AN12178

A Simple Way of Using CapTouch on LPC800 MCUs

Rev.1.0 — 3 July 2018 Application note

Document information

Info	Content
Keywords	CapTouch, threshold, auto-compare, low-power
Abstract	This application note discusses the method of using CapTouch module on LPC800 MCUs to support the low-cost and hardware resource-limited platform. If all channels can share the same threshold, CapTouch module can detect the touch channel automatically by hardware.

Revision history

Rev	Date	Description
1.0	20180703	Initial version

Contact information

For more information, please visit: http://www.nxp.com

1. Overview

Some LPC800 MCUs, such as LPC845 and LPC804 integrate CapTouch module to support the touch solution. CapTouch module is a type of general converter that can sense the value of capacitance. It scans the capacitance of each pad on the touch panel and indicates the pad that is touched.

The earlier version of software for enabling the CapTouch feature used CapTouch module as a general converter. The software controlled the conversion and kept all the sensed capacitance values in memory. It compared the latest values sensed (find the maximum or the minimum value from the value group with the count of enabled channels) once the scan for a channel was done. It led to frequent scanning of interrupts and kept the CPU busy. Also, the complex comparing process was called and had to finish in the short time slice (between the continuous scan of channels). High frequency was therefore required and this led to high power consumption.

To achieve the touch use case with low cost and hardware resource-limited platform, a lower CPU workload and a lower CPU frequency is considered. In this application note, a simple software is used to enable the CapTouch feature, with low CPU load and a low CPU frequency. It is used to make use of the CapTouch hardware and enable the interrupts only when necessary.

Finally, an actual case to measure the low-power performance of CapTouch module on LPC804 is used to show the use of this simple method.

2. CapTouch module deep dive

Before discussing the new CapTouch software, let us review the hardware registers to ensure that all the hardware features are used to help in reducing the workload of the software.

2.1 CapTouch interrupt events

Five interrupt events generated during the working of CapTouch are:

- Yes Touch
- No Touch
- Poll Done
- Timeout
- Overrun

These five interrupts have their own responding bits in CAPT_INTENSET/CAPT_INTENCLR, CAPT_INTSTAT registers to enable/disable or get/clear status flags.

2.1.1 Yes Touch/No Touch

CAPT_INTSTAT[YESTOUCH]/CAPT_INTSTAT[NOTOUCH] flag bit for Yes Touch/No Touch interrupt is asserted based on the sensing value (the timing count for YH port or ACMP channel is triggered) of the channel is in the available touched range or not. The available range is defined by the CAPT_POLL_TCNT[TCNT] field (comparing threshold) and CAPT_POLL_TCNT[TCHLOWER] field (comparing direction). This comparison is done automatically by the CapTouch hardware following the scan for any channel. If both are enabled together, the interrupt occurs for each sensing channel and the software can fetch the sensing value for each channel. In the existing software for LPC845 they are enabled together, disabled together, and cleared flags together. In this case, the software captured the scan done event and fetched the sensing value for each channel.

2.1.2 Poll Done

CAPT_INTSTAT[POLLDONE] flag bit for Poll Done interrupt is asserted at the end of each polling round or when a Poll Now scan complete. A polling round includes a sequence of all the enabled X channels on the CapTouch. For Poll Now scan mode, there is only one-time polling round. For Continuous scan mode, there is increasing count of polling round one by one.

2.1.3 Timeout

CAPT_INSTAT[TIMEOUT] flag bit for timeout interrupt is asserted when the sensing value of the channel is beyond the range defined by CAPT_POLL_TCNT[TOUT] field. The triggering occasion is in the same time with Yes Touch / No Touch event. The trigger event is based on the comparison automatically by the CapTouch hardware like the Yes Touch / NO Touch, but the comparing object is the timeout value (CAPT_POLL_TCNT[TOUT]) instead of touch threshold value CAPT_POLL_TCNT[TCNT]). Timeout interrupt can be caused by some type of hardware error. For example, if the hardware system is changed while CapTouch is working, the timeout interrupt is generated (as all the available sensing value should not be timeout).

2.1.4 Overrun event

CAPT_INTSTAT[OVERRUN] flag bit for overrun interrupt is asserted when the old sensing result in CAPT_TOUCH register is covered by the new generated one before it is read out. Overrun always wastes the hardware resource since dummy working exists while the power and processing time are lost. CAPT_CTRL[WAIT] bit can enable a hardware function that forces to read the sensing value, or the next measurement is not launched. However, the overrun can be caused by the complex processing between scans of channel using too much time, and it can be considered as a type of a software error.

See Fig 1 for the waveform for CapTouch events.

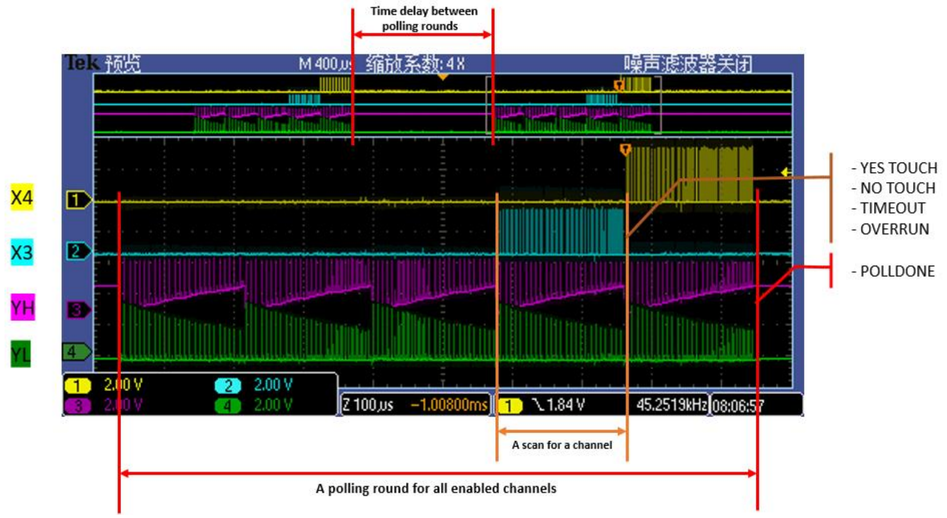


Fig 1. CapTouch events in continuous scan

As shown in Fig 1, the Yes Touch/No Touch/Timeout/Overrun interrupts might occur after the scan for each channel, and the Poll Down interrupt might occur after the scan for the whole sequence.

2.2 CapTouch timing

The basic operation of the CapTouch hardware is to toggle the X and Y pins to recharge the electrode of the touch panel. Multiple recharge cycles construct the scan for a channel and a sequence of scans for channel constructs the scan round. The continuous scan mode is the normal run mode of CapTouch. It launches the scan round repeatedly and asserts the interrupts after their conditions are met.

There are two groups of programable timing parameters in CapTouch registers: one is for the timing of recharge cycle with the four states, and the other is for the timing between the scan round when in the continuous mode.

2.2.1 The timing of recharge cycle

There are four steps in a CapTouch recharge cycle to read a sensing value of touch channel. The X and Y pins are set to different states. See Table 1. For additional information on the different states, see the UM11029, LPC84x user manual.

Table 1. CapTouch recharge cycle

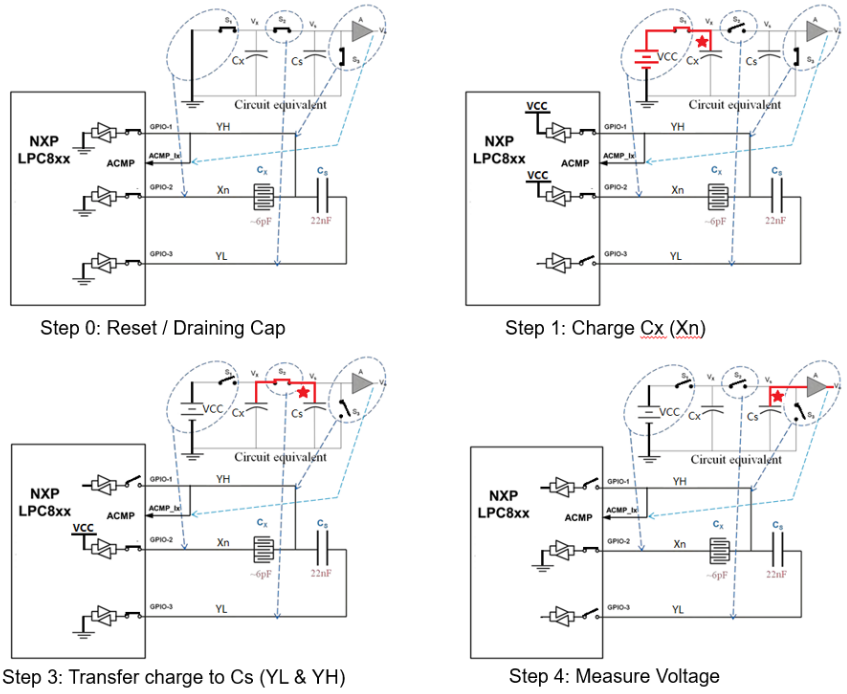
Step	State of loop	Active X	YH	YL	ACMP	Other X
0	Reset/Draining Cap	L	L	L	Don’t care	L
1	Charge X	H	H	High- Z	Input	High- Z[1]
2	Transfer charge to Y	H	High- Z	L	Input	High- Z
3	Measure voltage	L	High- Z	High- Z[2]	Input	High- Z

[1] Can be High-Z or Low. Selectable, based on system requirements and grounding.

[2] YH pad will be an input if using YH port pin measurement mode, High-Z is using analog comparator measurement mode.

To scan the channel, step1 to step 3 are repeated until the ACMP (or the YH port pin) reaches the threshold (Yes Touch or No Touch event) or timeout with many iterations beyond the available range in responding register field (Timeout event). When any event occurs, the available counter value is stored in the CAPT_TOUCH register as the result of measurement. Then it returns to step 0 to scan the next channel.

With the help of the external circuit including the capacitor of touch pad and accumulating capacitor, the energy is transferred from X to YH, and released along with YL. Once the pad is touched, the procedure of recharging is affected by the change in the capacity of the touch pad. The count of recharge cycle will change against the idle status and the CapTouch module will detect the touch event. See Fig 2 for the energy transfer situations on the external circuit responding to the four internal states of CapTouch module.



Step 0: Reset / Draining Cap

Step 1: Charge Cx (Xn)

Step 3: Transfer charge to Cs (YL & YH)

Step 4: Measure Voltage

Fig 2. Four steps of CapTouch with external circuit

Fig 3 shows the waveforms of CapTouch pins during the recharge cycle.

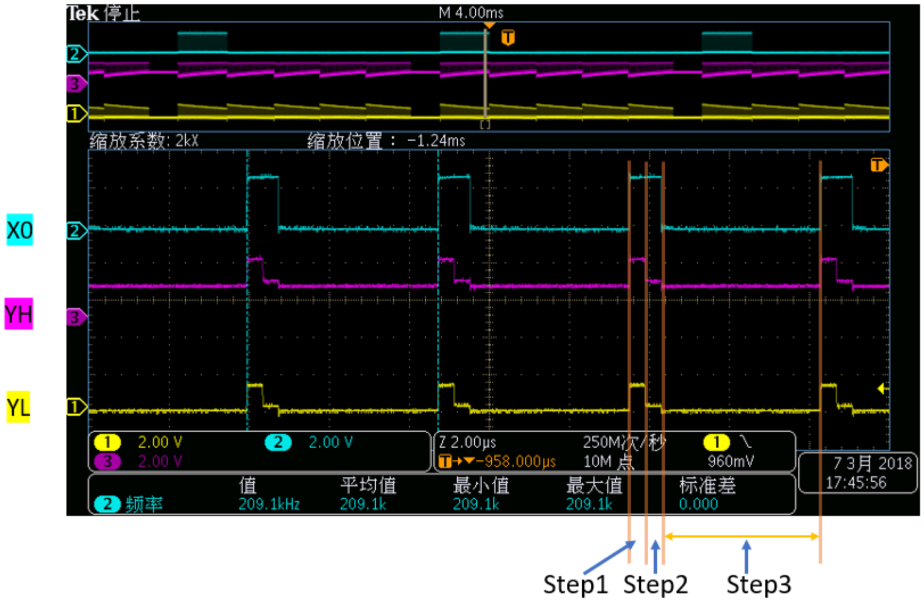


Fig 3. CapTouch pins waveform in the recharge cycle

As shown in Fig 3, step1 to step 3 are repeated in the recharge cycle for X0 channel.

The delay of step 3 can be configured in CAPT_POLL_TCNT[MDELAY], setting the time for holding/sampling time for ACMP/YH pin and comparing to check if the Yes Touch/No Touch happens. According to the board, the user should select the right value. The longer period extends the scan period for a channel (more energy is released through YL for each recharge cycle) and increase the discriminability. But if it is too long, the speed of increasing the energy becomes too slow and will finally timeout.

The delay in step1 and step 2 are driven by one FCLK period. FCLK is the basic clock unit for CapTouch timing feature. On LPC800 MCUs with CapTouch module, the clock source for FCLK is usually configured in SYSCON_CAPTCLKSEL register with the clock divider in CAPT_CTRL[FDIV] register field. See Fig 4.

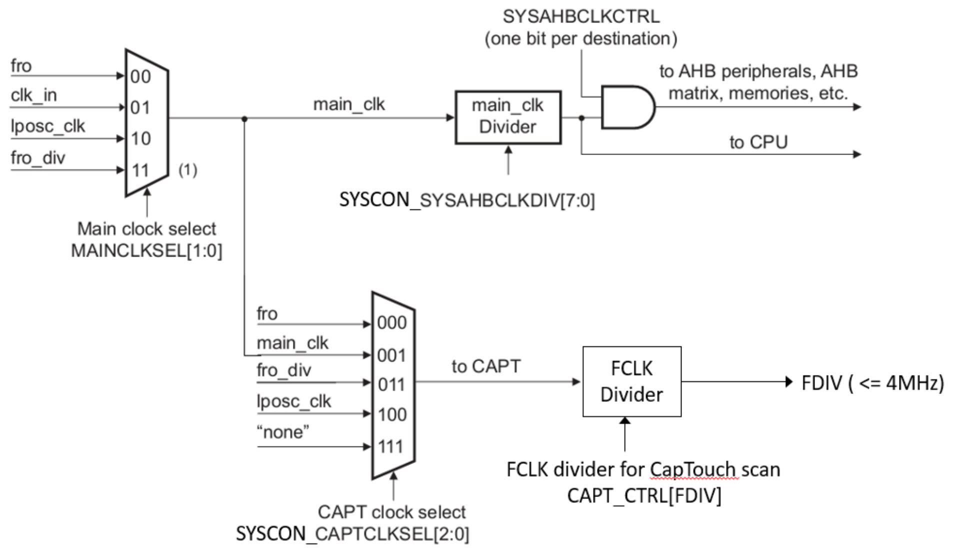


Fig 4. CapTouch clock source and divider

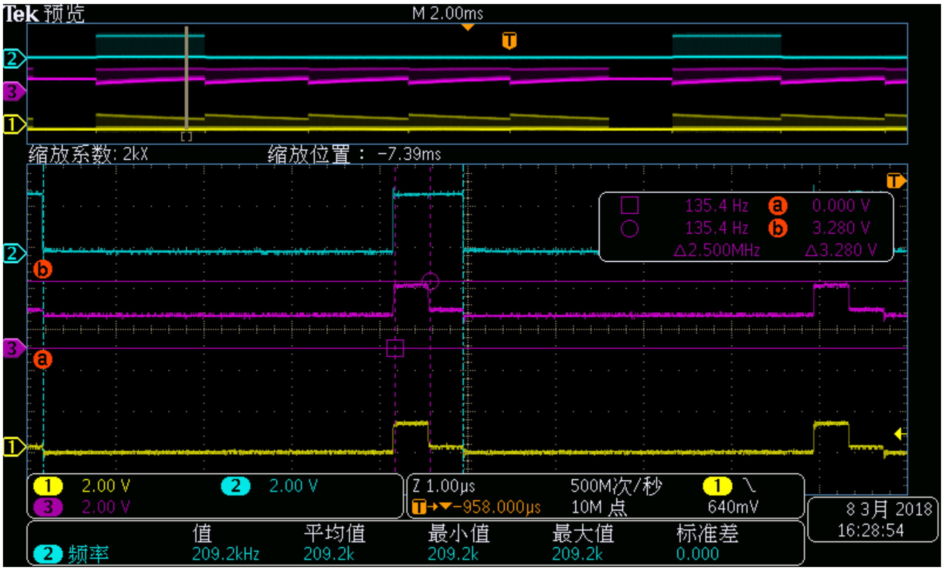
FCLK should not be higher than 4 MHz. Per the block-box test, CapTouch pins will not output the right wave with the FCLK of 5 MHz. The lower limit depends on the board hardware layout as the capacitor of pad will not accumulate the power during the recharge cycle with low FCLK, and the counter value exceeds the available range of CapTouch IP, the timeout event will occur. The block-box test on the test board (LPCXpresso804 board with Touch Shield board, OM40001), 1 MHz FCLK from LPOSC is available to support the CapTouch hardware normal working.

In the demo code aligned with Fig 3, FCLK is configured with the source of FRO 15 MHz and the divider of 6, and finally equal to 2.5 MHz.

```
/* CAPT_CTRL - Setup the working condition of CapTouch.
 *  0: 1 - POLLMODE: Scan method including: Inactive (stop the scan right now, control and status)
 *                                 PollNow (one-time scan and trigger by software)
 *                                 Continuous (continuously automatically by hardware)
 *  2: 3 - TYPE   : No other selection, the only available value is 0, can be reserved.
 *  4    - TRIGGER: Assert the scan done by checking the YH GPIO logic level or from the external ACMP module.
 *  5    - WAIT   : Enable blocking the scan before the available touch value is read.
 *  6: 8
 *  8:11 - FDIV   : Divider from CAPT clock source to FCLK.
 * 12:13 - XPINUSE: Logic state for inactivated X pins: High-Z (0x0) or Grounded (0x1).
 * 15    - INCHANGE: Check if this CAPT_CTRL register is accepting by hardware.
 * 31:16 - XPINSEL: Enable the activated X pins in the scan (or the sequence for continuous mode)
 */
while (0U ! (INCHANGE & LPC_CAPT->CTRL)) /* Waiting while the LPC_CAPT is in change. */
```

```
{}
LPC_CAPT->CTRL = POLLMODE_INACTIVE /* Will launch the scan later. */
        | TYPE_TRIGGER_ACMP /* TYPE_TRIGGER_ACMP */
        | WAIT /* NO wait, every data is processed. */
        | (((15000000UL/2500000UL)-1U) << FDIV) /* Divider for FCLK from clock source. */
        //| XPINUSE_HIGHZ /* X pins state when they are not active. */
        | (XPINUSE_LOW)
        | (0UL << XPINSEL) /* Will indicated the channel when launch the scan later. */
        ;
```

FCLK is shown as the delay of step 1 and step 2, measured by oscilloscope. See Fig 5 for the software setting.



	值	平均值	最小值	最大值	标准差
2 频率	209.2kHz	209.2k	209.2k	209.2k	0.000

Fig 5. CapTouch FCLK

As measured by the oscilloscope in the screen, the level ranges of X0 and YH are between GND (0V) and VCC of MCU (3.3V). The envelope of YL's voltage level acts like the YH's but has an offset, since its energy is inducted from YH through the capacitor of touch pad. The envelope of YH and YL are generated during step 3. YH accumulates the energy and the voltage of the envelop goes *higher*. YL releases the energy and the voltage of the envelope goes *lower*. While in the earlier phase, the energy for YH is less, a bigger change between the existing voltage level to VCC would generate more conducted energy through the electromagnetic field inside the capacitor of touch panel. That is the reason why YH goes high from initial low voltage while YL goes low from initial high voltage.

Fig 6 shows the delay of step 0. . It exists in the interval between scan of channels.

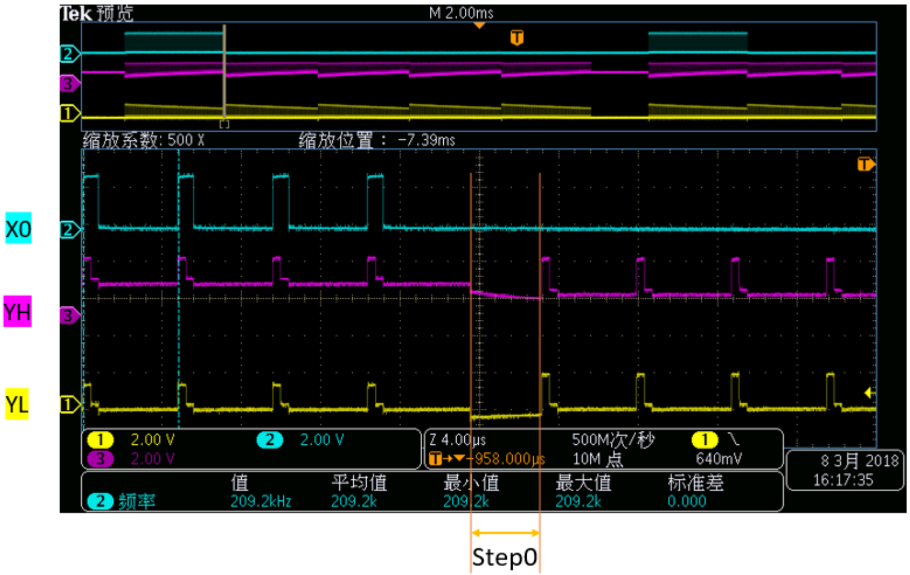


Fig 6. CapTouch step 0 in recharge cycle

The delay of step 0 can be configured in the CAPT_POLL_TCNT[RDELAY]. It specifies the number of divided FCLKs for which the module will remain in step 0 before starting the next X measurement in the polling round.

2.2.2 Timing between the scan round

There is an interval between the continuous scan rounds. It can be configured in the register field CAPT_ POLL_TCNT[POLL]. The actual delay time is 4096 FCLKs multiplied with the setting value. See Fig 7 for CapTouch delay between polling round.

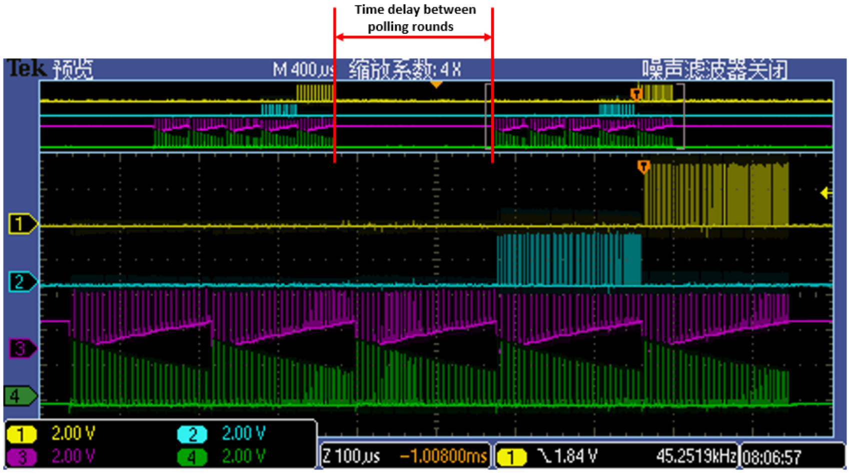


Fig 7. CapTouch delay between polling round

In application, the CapTouch does not have to scan the pins every time. It is because the timing resolution of human-machine interface does not need to be high with high power consumption. Lower frequency of scan rounds controlled by the hardware helps to reduce the power consumption.

The interval between the scan round is also reserved for the software process when the Poll Done event.

2.3 CapTouch hardware threshold

Generally, the threshold goes with the comparison method. In CapTouch module, there are two kinds of threshold: a voltage threshold and a counter threshold. They are setup by the hardware, so that the comparison can be executed automatically by the hardware.

2.3.1 Voltage threshold

The voltage threshold takes effect in step 3 of each recharge cycle and helps to decide if the scan for this channel is completed or requires more recharge cycles. In step 3, a hardware comparator is enabled (selected by CAPT_CTRL[TRIGGER] for YH port pin measurement or the external ACMP module integrated on the chip) to compare the voltage on YH pin and the threshold. If measuring voltage accumulated on the YH pin is still lower than the threshold voltage, more recharge cycles are still needed to increase the accumulated voltage on the YH pin. If the measuring value is higher than the threshold voltage, it is time to stop the recharge and record the count of recharge cycle into CAPT_TOUCH register. This counter value senses the value for the touch channel.

CAPT_CTRL[TRIGGER] register field decides where the voltage threshold is from. If the CAPT_CTRL[TRIGGER] = 0, it is for the *YH port pin measurement* mode. In this mode, the threshold value is the higher limit of TTL voltage range for logic 0, which is normally 0.8 V. In this mode, no additional hardware module is used, but the threshold value would not be customized. If the CAPT_CTRL[TRIGGER]=1, it is for the "ACMP measurement" module. In this mode, the on-chip ACMP module would be working in sync with the CapTouch module, and the threshold voltage can be setup in the ACMP module, for example, by the internal DAC in the ACMP. It means that the threshold value is programmable.

2.3.2 Counter threshold for YES Touch or NO Touch

The hardware threshold for YES Touch or NO Touch event for CapTouch module takes effect after the end of the scan for each channel and helps deciding if the scan for this channel is touched or not. YES Touch/NO Touch will depend on the comparison between the sensing value for this channel and pre-defined touch threshold value in CAPT_POLL_TCNT[TCNT] register field.

However, along with CAPT_POLL_TCNT[TCNT] field, another programmable condition for this comparison defined in CAPT_POLL_TCNT[TCHLOWER] register field. This bit setups if the lower or higher count than untouched sensor. When CAPT_POLL_TCNT[TCHLOWER] = 0, the YES Touch event is triggered on higher than CAPT_POLL_TCNT[TCNT] while the NO Touch event is on the lower range. When CAPT_POLL_TCNT[TCHLOWER] = 1, the direction is reverted, the YES Touch event is on the lower range, while the NO Touch event is on the higher range.

2.3.3 Counter threshold for timeout

Another hardware threshold for touch event, Timeout event, takes effect at the same time with the YES Touch/NO Touch comparison. Timeout event depends on the comparison between the sensing value for this channel and pre-defined timeout value in CAPT_POLL_TCNT[TOUT]. Once the sensing value for this channel is beyond available range defined by CAPT_POLL_TCNT[TOUT], it means that the timeout event has occurred.

3. Simplify software based on hardware feature

For reducing the power consumption of CapTouch in application, consider the following aspects:

- To simplify the software to leave more CPU work load for user application.
- To make maximum use of the hardware so that simpler software is used.
- To enable the low power mode in hardware to save power.
- To review the use of CapTouch module again.

3.1 Use of auto-compare by hardware

In general, the primary detection method for touch detection application, is comparison. Therefore, the hardware auto-compare function in CapTouch module must be used as much as possible. Normally, touch detection method compares values of all the enabled channels and find the largest or the smallest one, taking it as the touched channel.

In CapTouch module, *Counter Threshold for YES Touch or NO Touch* can be used to do the similar comparing work with sensing values and find which channel is touched. However, the CapTouch hardware uses a simple way to make the comparison between all the channels, taking third-part threshold value to distinguish the only touched one from others. It will reduce the complexity of calculation from $O(n^2)$ to $O(n)$. Choose the proper threshold value that will divide the touched channel and un-touched value into two parts.

3.2 Shared hardware threshold

When using the hardware auto-compare feature in CapTouch, the following problem should be considered. There is only one shared hardware threshold value. It means that the setting value in the CAPT_POLL_TCNT[TCNT] field is applied to all the sensing values of the channels, even their idle values of NO touch are not equal. Depending on the differences of the layout on the touch board, the sensing values for channels are not the same, with a different idle value for NO Touch and a different range for YES Touch. Therefore, in a thorough CapTouch solution, the software keeps all the relating values in memory and performs the necessary process for each channel by comparing the unified values according to the relating data. Finally, it judges whether there is a touch event. The thorough solutions have a huge requirement of hardware and software resource. For the simple method of using CapTouch on the low-cost and low power platform, trying to reduce the difference of sensing values by paying more attention on the layout of touch pad would be the acceptable way.

Selecting a proper threshold value for all the channels is a big factor for the simple CapTouch usage. It is good to update the threshold automatically according to the change in the environment condition. It is already implemented in the LPC845, which is a 32-bit ARM Cortex-M0+ MCU with 64 kB flash /16 kB RAM and up to 30 MHz operating frequency. For the ultra-low-cost platform of LPC804, which is a 32-bit ARM Cortex-M0+ MCU with 32 kB flash / 4 kB RAM and up to 15 MHz operating frequency, a simple method is more suitable (it can be used in the LPC845 also). In this application note document, the Freemaster GUI tool is used to monitor the sensing values for all the touch pad on demo board and selects a static threshold value manually for CapTouch hardware threshold register. See Fig 8.

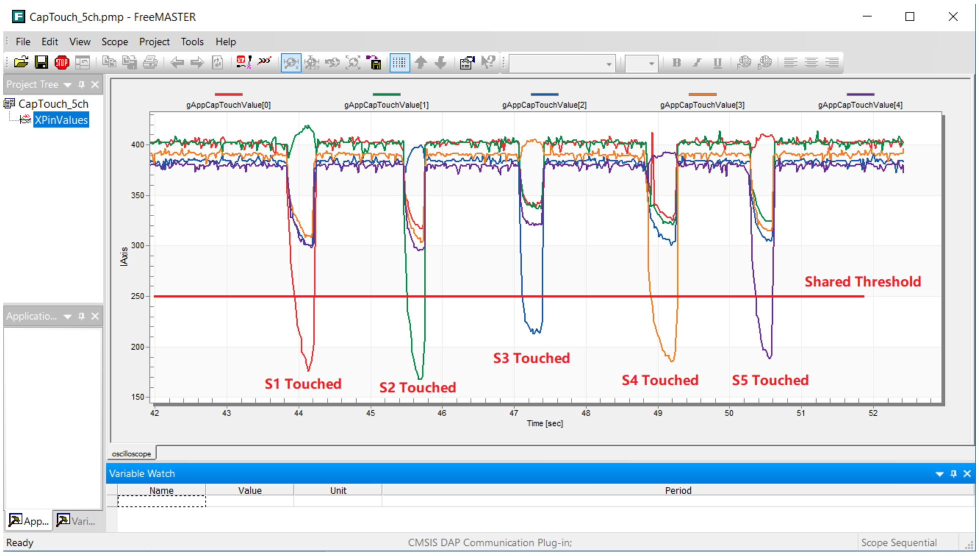


Fig 8. Using Freemaster tool to monitor sensing values and select static threshold.

The static threshold value loses some dynamic performance but is acceptable in most cases where the environment does not change.

3.3 Use the YES Touch interrupt event only

Although the CapTouch module provides up to five kinds of interrupts, not all the interrupts are necessary when running the simple application. Enabling all the interrupts with their response process makes the software work, but with high CPU workload. Only enabling the "YES Touch" interrupt and response is acceptable in the simple method of using CapTouch.

Some of the interrupts can be enabled during the debug period, to find the suitable configuration or to try the other method of detecting touch position. For example, enabling both YES Touch and NO Touch interrupt makes sure that there is always an interrupt after the scan of each channel. Software would then be able to read the sensing values and process with a customized algorithm.

3.4 Workflow of the simple CapTouch usage

See Fig 9 shows the workflow of using the CapTouch module on LPC800 MCU in a simple way.

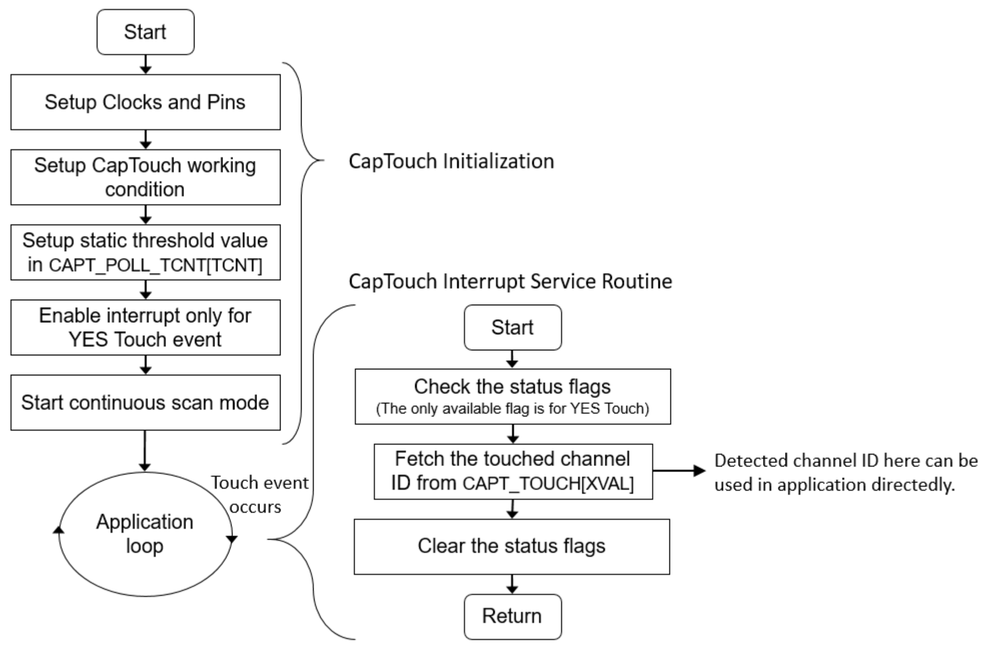

Start
Setup Clocks and Pins
Setup CapTouch working condition
Setup static threshold value in CAPT_POLL_TCNT[TCNT]
Enable interrupt only for YES Touch event
Start continuous scan mode
Application loop
Touch event occurs
CapTouch Initialization
CapTouch Interrupt Service Routine
Start
Check the status flags
(The only available flag is for YES Touch)
Fetch the touched channel ID from CAPT_TOUCH[XVAL]
Detected channel ID here can be used in application directly.
Clear the status flags
Return

Fig 9. Using CapTouch in the simple way

Use the following steps in the main () to enable a basic CapTouch application:

1. Setup the clock source/divider and pins for CapTouch.
2. Setup the CapTouch module for timing attributes, measure done trigger mode and grounded mode of Xpins.
3. Setup the static threshold value, this value can be decided manually.
4. Enable the only interrupt for Yes Touch event. This event is the only wake-up source for CapTouch.
5. Start the continuous scan by enabling the continuous mode in CapTouch module.

The CapTouch interrupt service routine is triggered once the CapTouch module detects the touch event automatically by the hardware. In this routine, besides the operation of getting and clearing flags, user code will read the touched channel directly from CAPT_TOUCH[XVAL] register filed. It is the most important one in CapTouch application.

4. A use case - low power measurement on LPC804

Most consumer electronics require low power to survive for a longer duration as they work on battery power supply. The CapTouch module on LPC804 MCU can survive in deep-sleep and power-down mode. In this test case for low power, it expects the hardware and software to be as simple as possible so that it can work in the resource-limited hardware platform. Using the feature that can be automatically done by hardware reduces the power than using the software. The complex CPU will need more power to do the same work with the software. There is a test case to measure the static low power performance of CapTouch module on LPC804, to provide the information for future low power design. The simple way of using CapTouch mentioned in this application note is applied to the measure process.

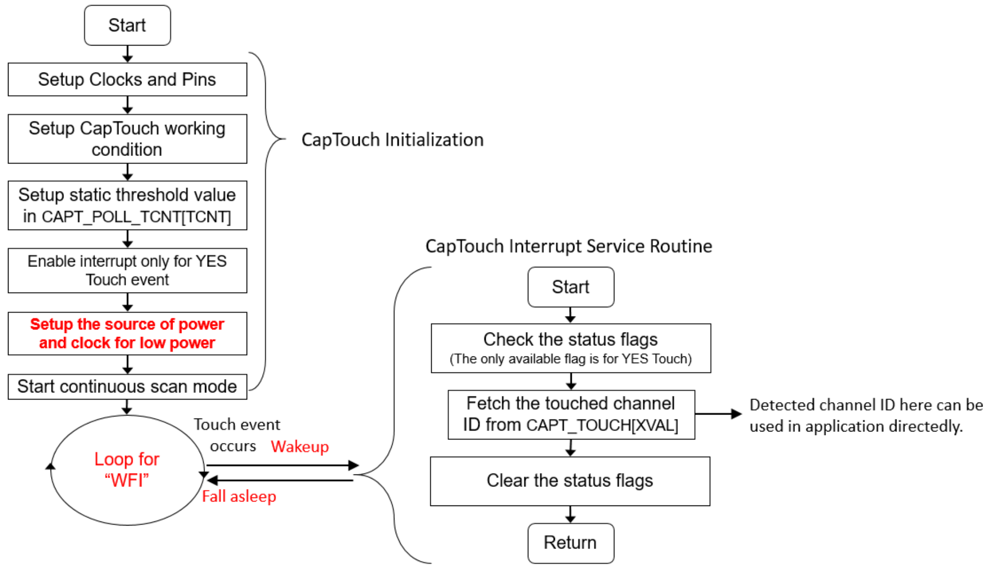

Start
Setup Clocks and Pins
Setup CapTouch working condition
Setup static threshold value in CAPT_POLL_TCNT[TCNT]
Enable interrupt only for YES Touch event
Setup the source of power and clock for low power
Start continuous scan mode
CapTouch Initialization
Loop for "WFI"
Touch event occurs
Wakeup
Fall asleep
CapTouch Interrupt Service Routine
Start
Check the status flags
(The only available flag is for YES Touch)
Fetch the touched channel ID from CAPT_TOUCH[XVAL]
Detected channel ID here can be used in application directedly.
Clear the status flags
Return

Fig 10. Using CapTouch for low power case

Fig 10 shows the diagram of a low-power use case of CapTouch. It is based on the basic simple method, while adding the steps that prepare and enter the low-power mode. When in the low-power mode, the CapTouch module keeps scanning while most modules are powered down. CapTouch module automatically detects the touch event through the hardware and wakes up the MCU when there is a touch event. In the example project, some LED lights are used to show the different touched channels. When all the process is completed, it returns to the low-power state and waits for another wake-up event.

In the test case, the CapTouch is used in the LPOSC as the clock source with 1 MHz. The FCLK frequency for CapTouch is equal to LPOSCs, while all the other modules are powered down including BOD. See Table 2 for the summary of measured data.

Table 2. LPC804 CapTouch static power consumption

Power mode	Current consumption
Normal run	1468 µA
Deep-sleep	471 µA
Power-down	332 µA

It is reasonable for the CapTouch to cost so much current consumption because the voltage level of CapTouch pins are toggled in a high frequency (1 MHz) even in the low-power mode. The measurement result is only for the working performance of CapTouch. In the actual low-power touch use case, the CapTouch module can be optionally shutdown to save more power.

5. Conclusion

This application note gives a detailed information about the CapTouch module and provides the description of some critical concepts of CapTouch, including the four steps of recharge cycle, polling round, and interrupt events. The snapshot of waveform shows the CapTouch working conditions.

Based on the knowledge of the CapTouch module, a simple way is provided to enable the touch function on hardware resource-limited platform, LPC804, for example. It makes use of the hardware auto-compare feature to save the software workload and simplifies the interrupt trigger. But the limitation is with the shared hardware threshold, where all the electrical characteristics of touch pads should be the same so that their sensing value would apply for the same threshold.

A real case, measuring the low-power performance of CapTouch is used to evaluate the simple way of using CapTouch, and it can capture the touch event in low-power mode. This case proves that the simple way is feasible and also provides some measuring data for CapTouch low power use case.

6. Legal information

6.1 Definitions

Draft — The document is a draft version only. The content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included herein and shall have no liability for the consequences of use of such information.

6.2 Disclaimers

Limited warranty and liability — 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. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the *Terms and conditions of commercial sale* of NXP Semiconductors.

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.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or 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 and its suppliers accept 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.

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 is suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Translations — A non-English (translated) version of a document is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Evaluation products — This product is provided on an "as is" and "with all faults" basis for evaluation purposes only. NXP Semiconductors, its affiliates and their suppliers expressly disclaim all warranties, whether express, implied or statutory, including but not limited to the implied warranties of non-infringement, merchantability and fitness for a particular purpose. The entire risk as to the quality, or arising out of the use or performance, of this product remains with customer.

In no event shall NXP Semiconductors, its affiliates or their suppliers be liable to customer for any special, indirect, consequential, punitive or incidental damages (including without limitation damages for loss of business, business interruption, loss of use, loss of data or information, and the like) arising out the use of or inability to use the product, whether or not based on tort (including negligence), strict liability, breach of contract, breach of warranty or any other theory, even if advised of the possibility of such damages.

Notwithstanding any damages that customer might incur for any reason whatsoever (including without limitation, all damages referenced above and all direct or general damages), the entire liability of NXP Semiconductors, its affiliates and their suppliers and customer's exclusive remedy for all of the foregoing shall be limited to actual damages incurred by customer based on reasonable reliance up to the greater of the amount actually paid by customer for the product or five dollars (US$5.00). The foregoing limitations, exclusions and disclaimers shall apply to the maximum extent permitted by applicable law, even if any remedy fails of its essential purpose.

6.3 Licenses

Purchase of NXP <xxx> components
<License statement text>

6.4 Patents

Notice is herewith given that the subject device uses one or more of the following patents and that each of these patents may have corresponding patents in other jurisdictions.

<Patent ID> — owned by <Company name>

6.5 Trademarks

Notice: All referenced brands, product names, service names and trademarks are property of their respective owners.

<Name> — is a trademark of NXP B.V.

7. List of figures

8. List of tables

9. Contents

Please be aware that important notices concerning this document and the product(s) described herein, have been included in the section 'Legal information'.

For more information, visit: http://www.nxp.com

Date of release: 3 July 2018
Document identifier: AN12178