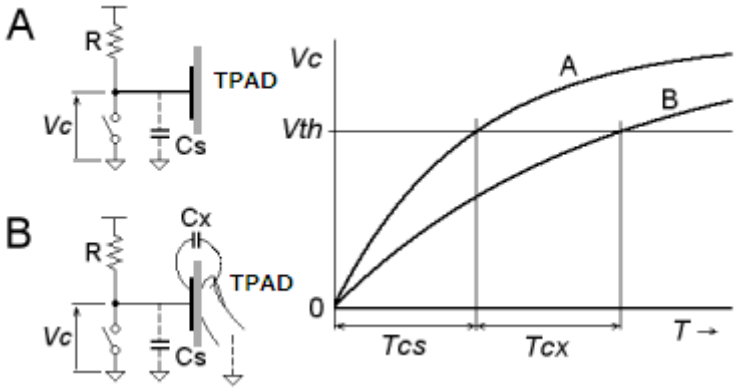
本文讲述的是如何在 XN12L 系列 MCU 上实现电容触摸按键的方法。主要的内容包括硬件设计和软件实现。文中涉及到的 XN12L 芯片的其它有关内容请参考用户手册。

电容触摸按键基本原理

电容触摸按键相对于传统的机械按键有寿命长、占用空间少、易于操作等诸多优点。其原理是利用电容对 RC 电路充放电时间影响来判定是否按键。实现原理如图下图所示：



图中 R 是外接的电容充电电阻，Cs 是没有触摸按下时 TPAD 与 PCB 之间的杂散电容。而 Cx 则是有手指按下的时候，手指与 TPAD 之间形成的电容。

先用开关将 Cs（或 Cs+Cx）上的电放尽，然后断开开关，让 R 给 Cs（或 Cs+Cx）充电，当没有手指触摸的时候，Cs 的充电曲线如图中的 A 曲线。而当有手指触摸的时候，手指和 TPAD 之间引入了新的电容 Cx，此时 Cs+Cx 的充电曲线如图中的 B 曲线。从上图可以看出，A、B 两种情况下，Vc 达到 Vth 的时间分别为 Tcs 和 Tcs+Tcx。其中，除了 Cs 和 Cx 我们需要计算，其他都是已知的，根据电容充放电公式：

Vc=V0*(1-e^(-t/RC))

其中 Vc 为电容电压，V0 为充电电压，R 为充电电阻，C 为电容容值，e 为自然底数，t 为充电时间。

在实际应用中，我们只需要调整 R，Cs 值，使得 MCU 可以稳定的区别是否触摸即可。R 可以通过外挂上拉电阻实现，而 Cs 与印刷版的 PAD 面积相关。

Xinnova XN12L 系列 MCU 概述

XN12L 系列是基于 ARM M0 内核的通用 MCU。该系列可以涵盖从低端到高端各种 MCU 应用，具有高性能，低成本，代码加密可靠等特点，是取代 8 位机 16 位理想的产品。与其它 MCU 相比，XN12L 系列指令精简，内含用于增强运算的 xDSP，主频更是可高达 100MHz，外设丰富实用，支持在线调试，在目前 MCU 市场上表现非凡。主要特点有：

- 高达 100MHz ARM Cortex M0 CPU
- 高达 88KB 用户 Flash 和 16KB SRAM
- xDSP 用于增强 MCU 运算功能
  - 32 位单周期除法器
  - CORDIC 运算器
  - CRC 校验
- 多种时钟系统供用户选择
  - 1%精度的内部晶振
  - 支持外部时钟和晶振
  - 内部 PLL
  - 支持实时时钟(RTC)
- 多达 3 个独立的 ADC 转换器更适合系统高速采样需求
  - 12 位，1MHz 采样率
  - 多达 12 路 ADC 通道
- 2 个模拟比较器
- 10 位 DAC,1MHz 转换率
- 4 个增强型系统定时/计数器，支持正交编码信号
- 集成的片上温度传感器
- 支持各种通讯接口
  - 4 个带波特率自动检测和 IrDA 功能 UART
  - 1 个 SPI
  - 1 个 Quad SPI （支持 Flash 4 IOs 数据传输）
  - 1 个 TWS（I2C 兼容）
- 支持内存，外设间的 DMA 大容量数据传输
- 支持故障诊断恢复功能（WDT/BOD)
- 支持睡眠，深度睡眠和掉电三种低功耗模式
- 数据和程序的高可靠和保密性能
  - 2 个 128 位密码的分区加密和保护技术, 确保片内数据安全和防知识产权的克隆
  - 加密模式下的应用二次开发，更好知识产权回报
- 单电源供电（3.3v）

XN12L MCU 电容触摸按键实现

在本案中，我们将通过用计数器的捕捉功能对输入管脚信号上升沿捕捉来实现。在 XN12L MCU 所有管脚都带有一个~10K 的上拉电阻并且缺省有效，这个电阻将作为 RC 电路的 R（用户也可以外挂上拉电阻，这样的话，需在程序中断开内部上拉）。PAD 的话使用一个 5 毛硬币接到 IO 管脚来模拟。软件过程如下:

```
//初始化计数器 TMR16B0 的捕捉功能
 //initial timer CT16B0_CAP0 to capture touch input, available pin PIO0_11/PIO0_28/PIO2_0
 XN_SYSCON->SYSAHBCLKCTRL |= 1<<7; //enable TMR16B0 clk
 XN_TMR16B0->PR = 1; //prescale set to 1, timer clk is 10MHz, same as main clk
 XN_TMR16B0->CTCR = (0 << 0) + (8<<8) + (8<<12); //set up timer config
 XN_TMR16B0->CCR = (1<<0)   | (1<<2); //capture tc at cap0 and enable int
…
//触摸按键判定
XN_GPIO2->DIR = 1<<0;       //设置 PIO2_0 管脚为输出
XN_IOCON->PIO2_0 =0x0090;   // PIO2_0 管脚输出低电平，对 PIO2_0 管脚外挂电容放电
```

```
//启动计数器
XN_TMR16B0->IR = 0xFF; //清除捕捉事件标志
XN_TMR16B0->TC = 0; //设置计数器初始值
XN_TMR16B0->TCR = 1;  //启动计数器
XN_IOCON->PIO2_0 =0x0091;  // 设置 PIO2_0 管脚为计数器 TMR16B0 的捕捉输入 CAP0， PIO2_0 管脚外挂电容被冲电
While((XN_TMR16B0->IR  & 0x10) !=0 ); //等待捕捉事件发生
XN_TMR16B0->TCR = 0;  //停止计数器
XN_TMR16B0->IR = 0xFF; //清除捕捉事件标志
cap_val = XN_TMR16B0->CR[0]; //读取计数器 16CB0 捕捉寄存器 0 值
//无触摸时，捕捉寄存器 0 的值保持不变（0x2A）；有触摸时，捕捉寄存器 0 的值变为（0x5F）
```

通过管脚不断的充放电，用户程序可以比较出在有无触摸捕捉寄存器 0 值的变化，从而判断按键状态。