大家好，我是东华大学 2013 届，信息科学与技术学院自动化专业的陆雪纯，作为 NUEDC 实验室的负责人之一，我制作这份关于简易入门单片机 msp430，和 Altium Designer（配有库）的视频教程，希望对大家有所帮助！本视频尽量针对刚接触 C 语言的大一、大二的人，所以会讲解较为底层详细的东西。本教程中一些用【】符合框起来的名词，请按照生活中常规的意思去理解，不要和微机里面的专业术语混淆。

本视屏教程分为两个部分——MSP430 入门部分和 Altium Designer 入门部分。在 MSP430 中，我将从寄存器和内部电路简易原理开始教大家点亮流水灯、定时器中断、端口中断、ADC 的使用，了解了寄存器和内部电路原理之后，所有单片机几乎是如出一辙的，我个人认为，现在大多数人都直接选择用库函数，但是新手直接用库，他很难有一种入门感，甚至可能会一头雾水、没有概念，但如果你对于寄存器有了一定的了解，那么用其他芯片的一些函数库就是手到擒来。我选择 MSP430G2553 为我们教程中的开发板，因为我们学校信息系电子类的很多老师有这个开发板几十套，我们 NUEDC 实验室也有很多，所以我们东华大学的学生可以免费获得，所以选择它为教学板。我在百度云连接中提供所有的安装包、注册机、教程中的例程、MSP430G2553 的淘宝链接和授课稿以及其他相关有用的资料，并且用所教内容布置实训任务来帮助你们学以致用。在 Altium Designer 中，我已经用它绘制了飞思卡尔杯-平衡摄像头组的硬件电路板，已经打板并且投入使用，但教程中我只提供初代板（是我画的第一块板子，bug 很多），进行修改，删掉 UART 模块作为你们的练习，并将原理图、pcb 的布线，用到的集成库、安装包和其他有用的教程等都放入百度云链接，并且教会大家如何用 Message、board information、view configurations 进行排查，避免弱智的错误发生。

****

首先把我资料包里的软件安装破解好，然后必须要搞到板子。

如果你没有从老师和实验室那里搞到开发板，那么我给你开发板淘宝链接，如果链接失效就直接搜索 MSP430G2553，找那种有黑色方形纸盒子包装的（真希望店主给我广告费）：https://item.taobao.com/item.htm?spm=a230r.1.14.39.ebb2eb25pXeGu&id=14606699378&ns=1&abbucket=5#detail

打开文件后建立、保存工程的步骤，注意只要进行步骤一到四即可！！工程配置按照我给的工程的来！！！！！：https://wenku.baidu.com/view/4a2b278c04a1b0717ed5dd3f.html

****

好，那废话不多说！确保你手中有电脑，还有一块 MSP430G2553 的开发板以及连接线，我们就开始吧！

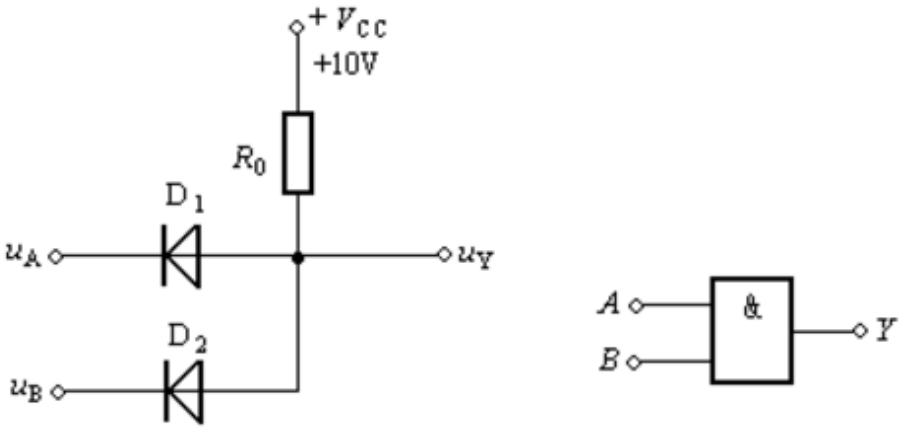
在教你们控制单片机之前，我将先用五分钟时间给你们讲解一些相关的知识，这些知识我个人认为很重要，他能让你们有个概念，避免一头雾水的生搬硬套 datasheet 上的规则，这也是和网上千篇一律的教程不一样的地方。

首先从小小二极管开始讲起：

二极管，我们高中就接触过，需要大于门坎电压（如硅管门坎电压为 0.5v）之后，才会被导通，导通的时候，二极管的正向导通电压是一个特定的值（硅管正向导通电压为 0.7v）。

假设 3v 及以上代表高电平，用“1”来表示，0.7 及以下代表低电平，用“0”来表示。

而这小小的二极管，按照我们的设计，组合成逻辑电路，就可以进行各种计算，比如下图的一个简单电路，它由二极管和电阻组成：



（这个电路具体工作原理，感兴趣的同学可以点击以下链接进一步了解：http://www.21ic.com/jichuzhishi/analog/questions/2013-10-10/193384.html）

这个电路起到什么作用呢？如果 A，B 两端都接入高电平“1”，那么 Y 端也会输出一个高电平“1”，如果 A，B 两端中至少有一端接入的是低电平“0”，那么 Y 端只会输出一个低电平“0”。也就是说，这个电路已经能实现一个“与”计算功能了。

（如果你不知道与运算是什么，可以点击这个链接 https://www.zybang.com/question/1f76ceaaec71acd0d753561eeabc8759.html）

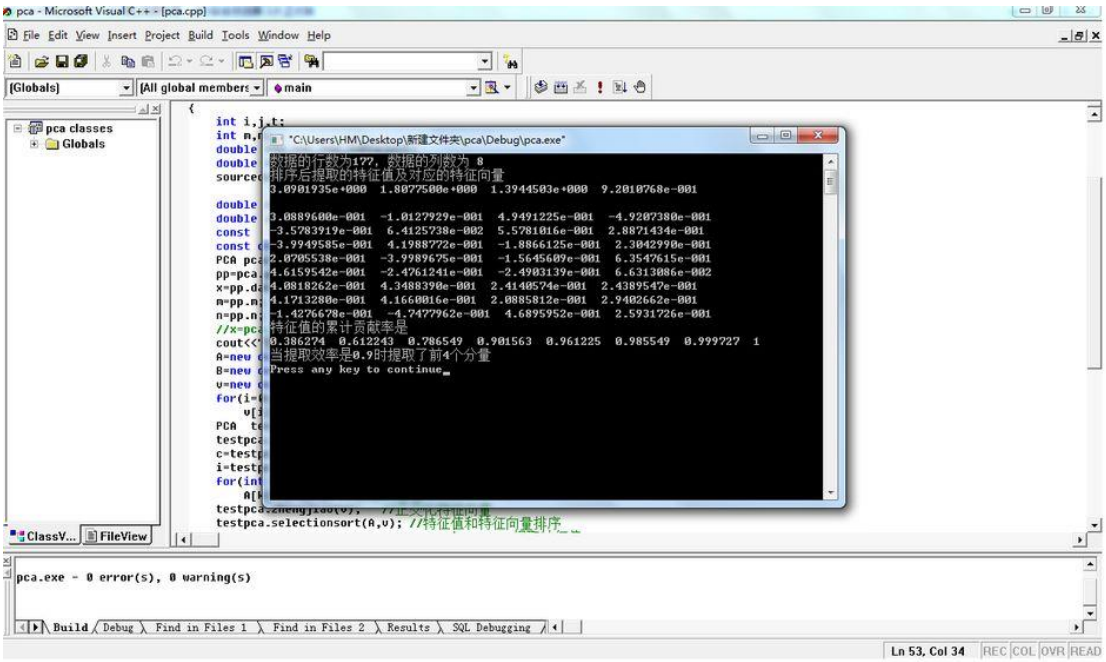
如果你忘了与运算是什么，没关系，暂时不要去了解，并不影响我要讲的东西。你只要记住，我讲这个电路只想要指出以下五点：

1. 这个电路，它的功能，是实现与运算，也就是一种【计算】。我们用的电脑，手机，要学习的单片机，他们都是由数以百万计的功能各异的电路，通过一定的规律组合、集成起来（所以为什么叫大规模集成电路，就是这个原理），来进行复杂的计算的（这就是为什么电脑也叫计算机），计算的过程，就是电脑运行的过程。

2. 这个电路里，只有高电平“1”和低电平“0”的概念，所以说，大规模集成电路——电脑在运行的时候，里面只有高低电平的出现，电脑只处理逻辑上的“0”和“1”，也就是我们学的二进制（一开始很多人都不知道学这个二进制干嘛，有何意义，01010 的），也就是——机器语言，机器语言是用二进制代码表示的，是计算机能直接识别和执行的。

我们现在学的 C 语言，或者以后会学的 C++，Java 之类的各种语言，都必须要被翻译成机器语言，计算机才能识别并执行，那么什么东西才能把 C 语言之类的高级语言翻译成机械语言呢？那就是我们很熟的软件——编译器了，也叫开发平台、开发工具等。比如我们东华大学的学生学 C 语言的时候，用到的编译器 Microsoft Visual C++，图标长这样：

界面长这样：



（后来我听说现在大一大二的小朋友又换编译器用 DevC++了）

而我们控制的单片机 MSP430 也要用到相应的编译器，我们的编译器是 IAR for MSP430（安装包百度云链接都有）

3. 在第 2 点中，我已经指出了高级语言、机器语言的概念，那么你们就应该明白一个道理：我们的 C 语言是高级语言，高级语言是高度封装了的（举个例子，你要装修你的家，你设计房间里需要用到空调、煤气灶等东西，你用高级语言写明了你的思想“要用空调、煤气灶”，然后你从商场购置来了空调和煤气灶，可以直接使用，不需要你从元器件一点点自己做个空调、煤气灶出来。类比到 C 语言里，高级语言就是 C 语言；空调、煤气灶等就是你可以直接使用的 printf()、scanf()函数，而商场就是函数库，所以你要包含这个库函数，也就是代码里要写上 #include<stdio.h>），同样的，我们操作单片机，我们的单片机型号是

MSP430G2553，那我们就需要包含头文件#include <MSP430G2553.h>

4. I/0 口（也叫 I/0 接口，或者输入输出管脚）的概念，I/O 就是 input、output 的缩写，输入输出的意思，I/O 口就是输入输出的管脚（管脚英语：PIN）。用以控制外围电路，什么是外围电路？看看开发板上，左下角有两个 led 灯，他们就是外围电路，他们和相应的管脚相连（P1.0　P1.6）
5. 仍然是在第二点中，我强调了：电脑在运行的时候，里面只有高低电平的出现，电脑只处理逻辑上的“0”和“1”。也就是说，微观上，在电脑的芯片里，程序、信息、指令的存储方式，只能由高电平“1”，低电平“0”两种量来表示，也就是二进制表示。

   单片机采用的是 TTL 电平，标准 TTL 输入高电平最小 2V，输出高电平最小 2.4V,典型值 3.4V，也就是说，在单片机当中，2v-5v 就属于高电平，常见高电平的典型值为 3.4v；输入低电平最大 0.8V，输出低电平最大 0.4V，典型值 0.2V。也就是说 0v-0.8v 属于低电平，常见的典型低电平为 0.2v。各个厂家生产的器件会有一些差异，高低电平的极限值会在标准值附近有一定的浮动。

   这里，我引进一句话（不准确，但足够帮你了解并使用单片机）：控制单片机，就是配置寄存器，而寄存器，是寄存【指令】的器件。[锦囊 1]

   这句话具体含义是什么，就由接下来的“点亮 LED 灯”来具体说明吧！

点亮流水灯：可以直接打开百度云链接里配置好的工程文件，直接烧写，熟练了之后，按照一开始给的教程链接，自己建立工程文件，自己配置函数。

我已经在之前说过，寄存器里面的指令以二进制存储。

代码中，等号左边是寄存器，等号右边是指令。

你要控制单片机去点亮外围电路的一个 LED 灯（高电平可以点亮），在我们的 MSP430 开发板中，管脚 P1.0 连接的是 LED1，我们就去试试点亮 LED1，那么你具体要做的是使单片机的管脚 P1.0 去“输出 高电平”，那么你需要做两步：

1. 让单片机连接着 LED1 的管脚 P1.0 的方向为输出（P1DIR）
2. 让输出的电平为高电平（P1OUT）

以上两步，都有专门对应的寄存器。“控制单片机，就是配置寄存器，而寄存器，是寄存【指令】的器件。[锦囊 1]”

那么这时候我们查阅 MSP430G2553 的 user’s guide:

8.2.3 Direction Registers PxDIR

Each bit in each PxDIR register selects the direction of the corresponding I/O pin, regardless of the selected function for the pin. PxDIR bits for I/O pins that are selected for other functions must be set as required by the other function.

Bit = 0: The port pin is switched to input direction

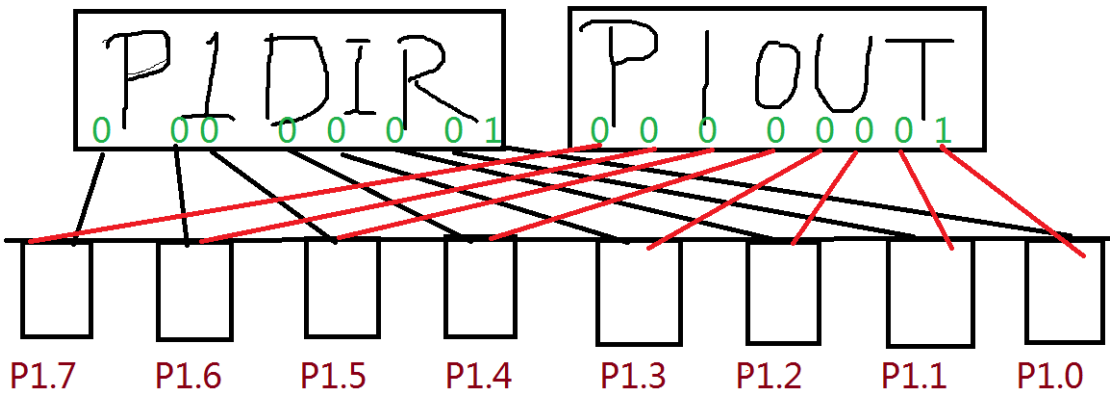
Bit = 1: The port pin is switched to output direction

8.2.2 Output Registers PxOUT

Each bit in each PxOUT register is the value to be output on the corresponding I/O pin when the pin is configured as I/O function, output direction, and the pullup/down resistor is disabled.

Bit = 0: The output is low

Bit = 1: The output is high



当你想控制 P1.0 输出 高电平　　P1DIR |= BIT0 =0000 0001

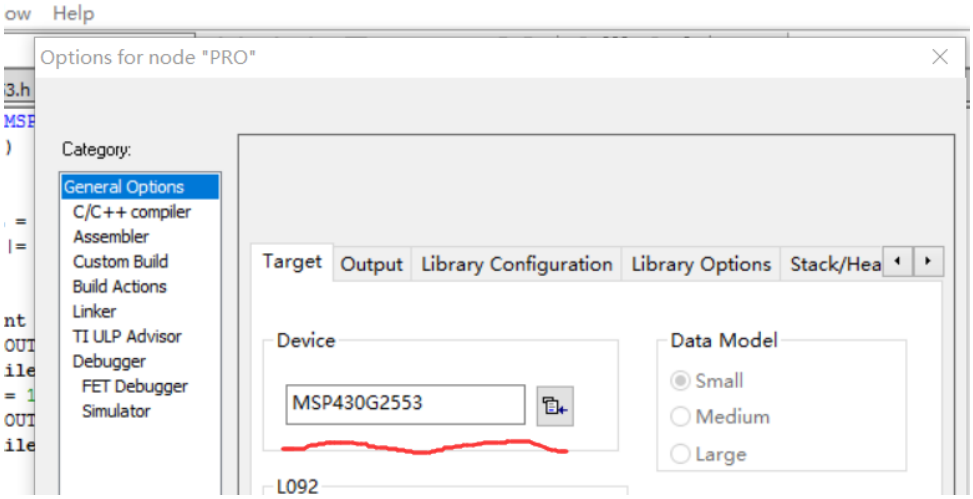
P1OUT |= BIT0 = 0000 0001

管脚的方向不是输入就是输出，管脚的状态不是高电平就是低电平（AD,DA 除外）[锦囊 2]

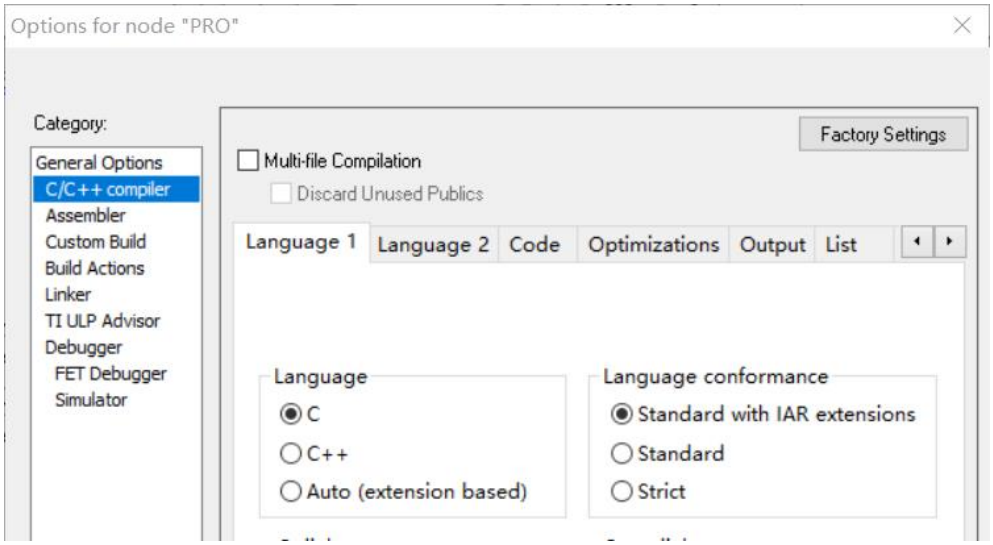
接下来就是要配置工程，然后烧录程序了！根据我上一次培训的经验来看，有很多学弟学妹想知道“我们为什么要配置工程？”这里我就要稍微说一下单片机的工作的的一些原理了。

首先要指出的是，电脑和人脑不一样，电脑里存储的任何信息（比如你烧写进去的程序、它自身计算的一些结果等）都是存放在存储单元的，这些存储单元都是有地址的，不同型号的单片机，内部的内存大小、存储结构是不一样的，所以你烧写程序，程序按照不同单片机的不同要求，按照应该被存入的单元的地址存入。你使用 C 语言编写或者 C++编写，它翻译成机器语言的方式也不一样（好比英语、法语书，肯定要对应的英语翻译、法语翻译来翻译），所以【相应的规则】也不同，你必须要告诉你的编译器你的芯片是什么型号，你用的什么语言写的代码，你的芯片和电脑是什么方式连接的（SWD 还是 JTAG 还是 JLINK 还是别的什么）

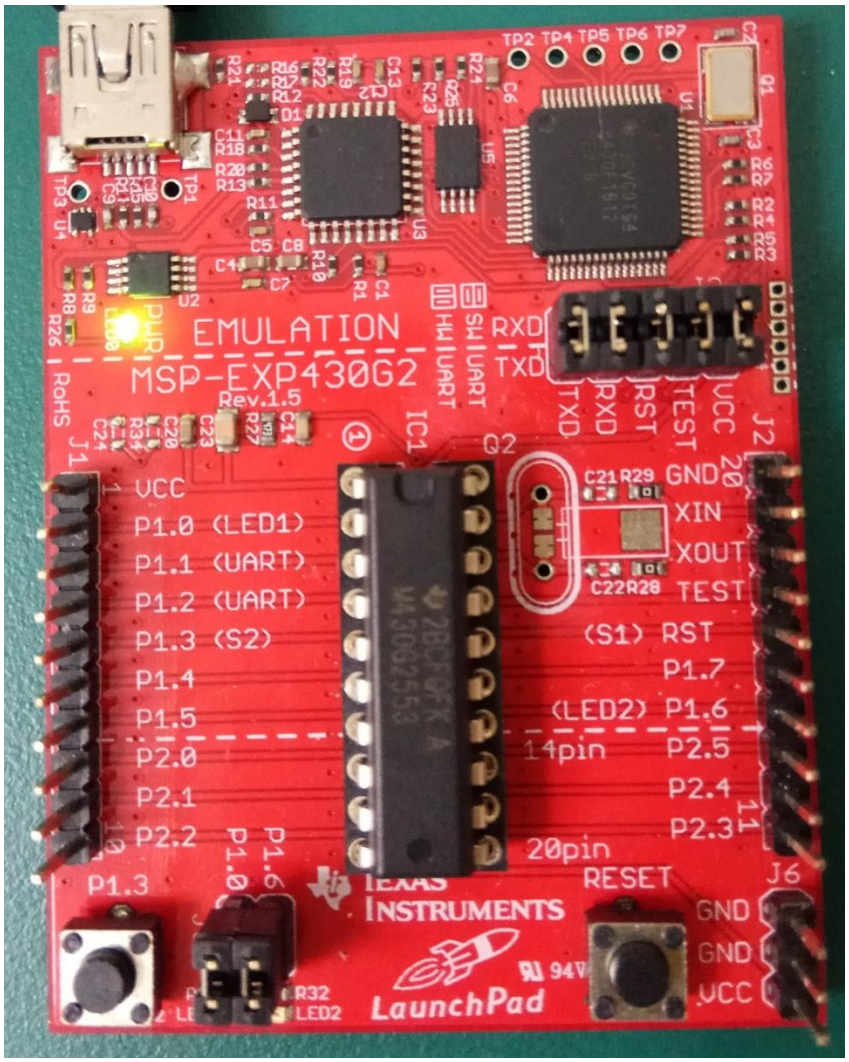
下图是工程配置 option 中的选定单片机型号



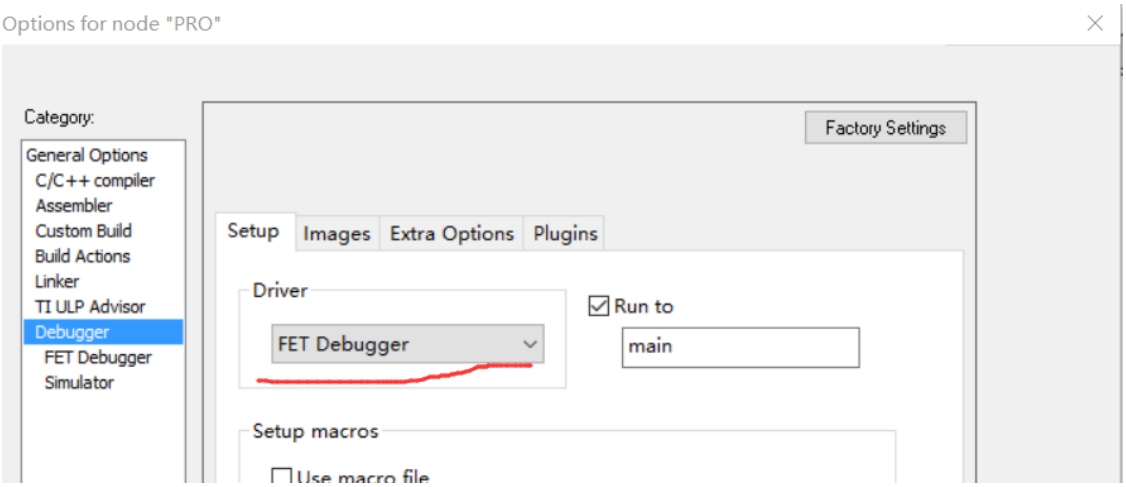
选定你用的语言

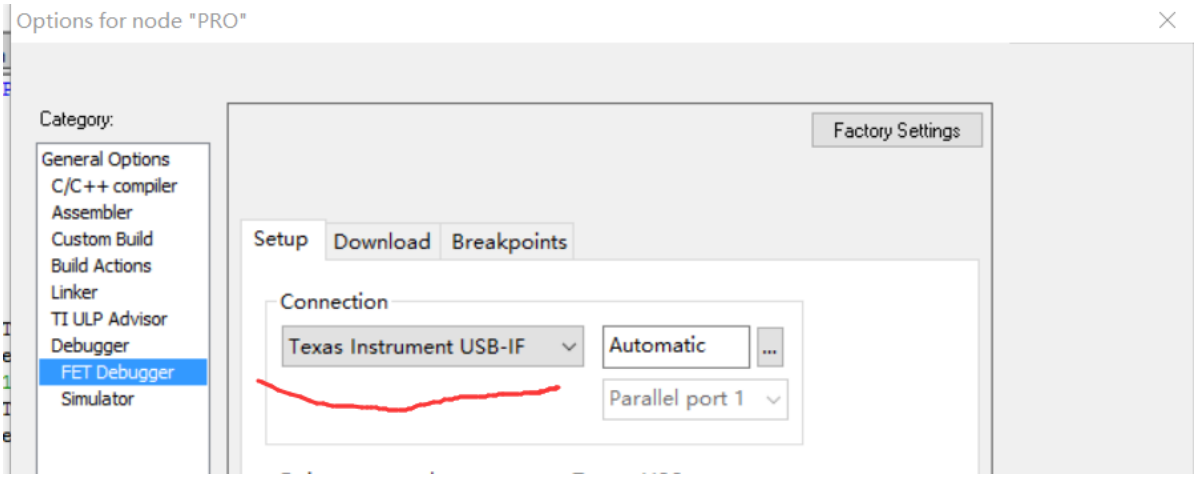


我们开发板上是把仿真器也做进去了的，具体看图靠上三分之一处，有个白色字体 EMULATION，所以那一部分都是硬件仿真器部分，EMULATION 是仿真器的意思，毫无疑问是硬件仿真，所以我们点选硬件仿真。而且我们的连接线也是 USB 连接线，所以连接选项 connection 按照图勾选。



FET DEBUGGER 中文翻译就是硬件仿真
（对仿真好奇的同学点这里：
https://zhidao.baidu.com/question/449972284.html）





好了，配置完之后，点一下菜单栏中的绿色三角，就能烧录了，你会发现，LED0，亮了……亮了……这是成功的第一步，我在这个小例子里面和之前都把很多延伸的概念讲了，这些概念，应用到其他各种型号的单片机，都应该可以触类旁通。

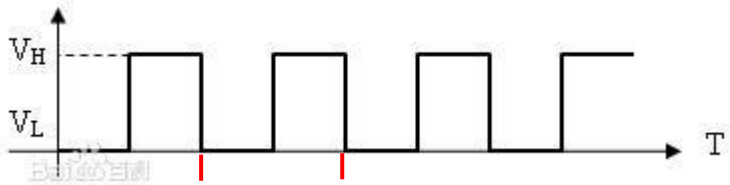
**********定时器**********
那么，接下来就要讲解定时器了。定时器是单片机中最重要的内容之一，你以后做的所有开发，都不太可能脱离定时器。
那么，定时器是个什么东西呢？顾名思义，就是——“你定下来，过了 200 秒后，你要停下手头的活，去做另一件事”

这句短短的话，包含了很多信息
1.“秒”，如同“时”、“分”一样，是个计时单位，也是一个衡量的基准单位，你不能说，“你过了 200 之后，去做某事情，200 什么？200 秒还是 200 分钟？”所以你要配置一个计时单位，要让计数器“过了 200 个计时单位后，去做某事”，这里我们计时单位选的是 1/12KHZ　秒

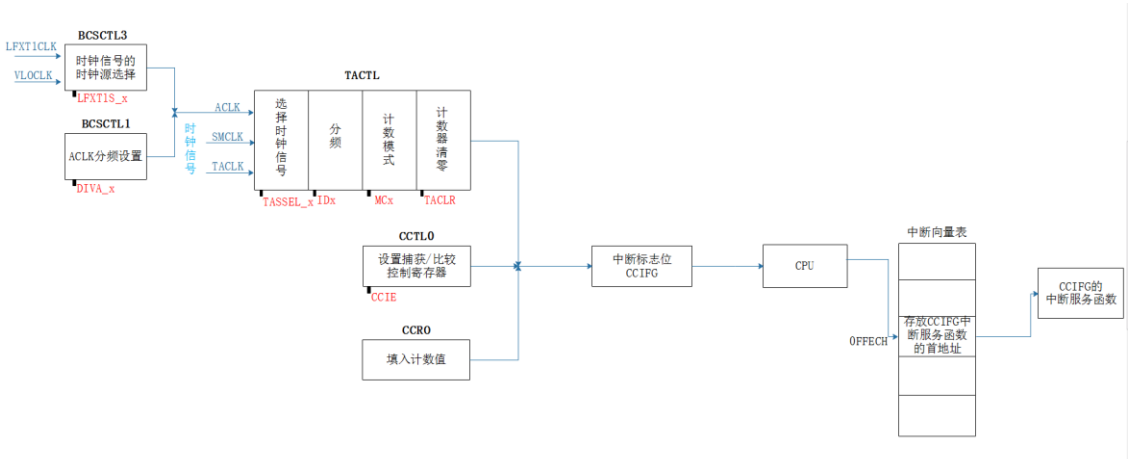
2.“200”是一个阈值，毫无疑问，我们程序里面写的 CCR0=12000 就是说阈值是 12000，当记到 12000 个计数单位后，就要进入中断了 。也就是 1/12000 * 12000 =1 秒
3.题目中的这句话还有一个隐藏条件，就是说，“你【知道】，你过了 500 秒后要去做某事”，如果你不知道你要去做某事，那就不会执行中断行为，所以要开启中断_EINT(); 计时，就是从使能中断后开始的

4.手头的活（main（））、去做某事（中断服务函数），这是两个不同的行动，相当于你本来执行的一个动作，现在要去执行另一个动作（所以通过中断向量表里记录的中断函数的首地址，跳倒中断函数里，去执行中断服务函数）
#pragma vector=TIMER0_A0_VECTOR//表示给这个向量表中的【定时器中断】所对应的地址进行赋值，并赋值为函数 void Timer_A (void)的首地址。
其中，关键字__interrupt 则表示这个函数是【中断服务函数】这种特殊函数。
5.你接完电话之后，回来继续做你原来做的事情（中断标志位清零，CPU 一看，清零了，就认为中断的任务已经完成，于是就退出了中断）

这里讲解一个时钟的概念（真的是很基础的知识点啊）
时钟，顾名思义，是一种计时用具，我们可以用它来记录时间。在单片机中，时钟信号的表现形式为“频率固定，只有高电平（V_H）低电平(V_L)的矩形波”



时钟边沿就是图中的竖线，从低电平变成高电平的叫上跳沿，反之叫下跳沿。单片机可以计算时钟边沿的个数，比如时钟的频率为 2Hz，那么时钟的周期(图中两端红线内的部分)就是 0.5s，如果单片机记录到了 2 个下跳沿，那么两个下跳沿中的时间就是 0.5s，如果记录到了 3 个下跳沿，就是 1s。这就是一种计时功能，所以叫做时钟。



**********端口中断**********

1.你知道这个是电话铃声（init 里的配置符合中断的条件），且有电话铃声响的时候，“你要去”接电话（开启中断）（且中断标志置 1，一旦中断标志位置 1，CPU 立刻去执行中断工作）

中断标志说明的是“当前有中断请求”，CPU 如果要响应的话，还必须要使能这个中断。也就是说工作的原理是：先判断“中断请求标志”再判断这个中断使能标志是否 Enable 了，或者讲是否这两个标志是否都有效了，系统才会响应这个中断。
不开中断使能，只是置位中断标志的话，就像我们定义了一个用户的 bit 标志，我们只是给它置 1 或者 0,但我们从来不判断它为 1 的时候做什么，为 0 的时候做什么是一样的道理。

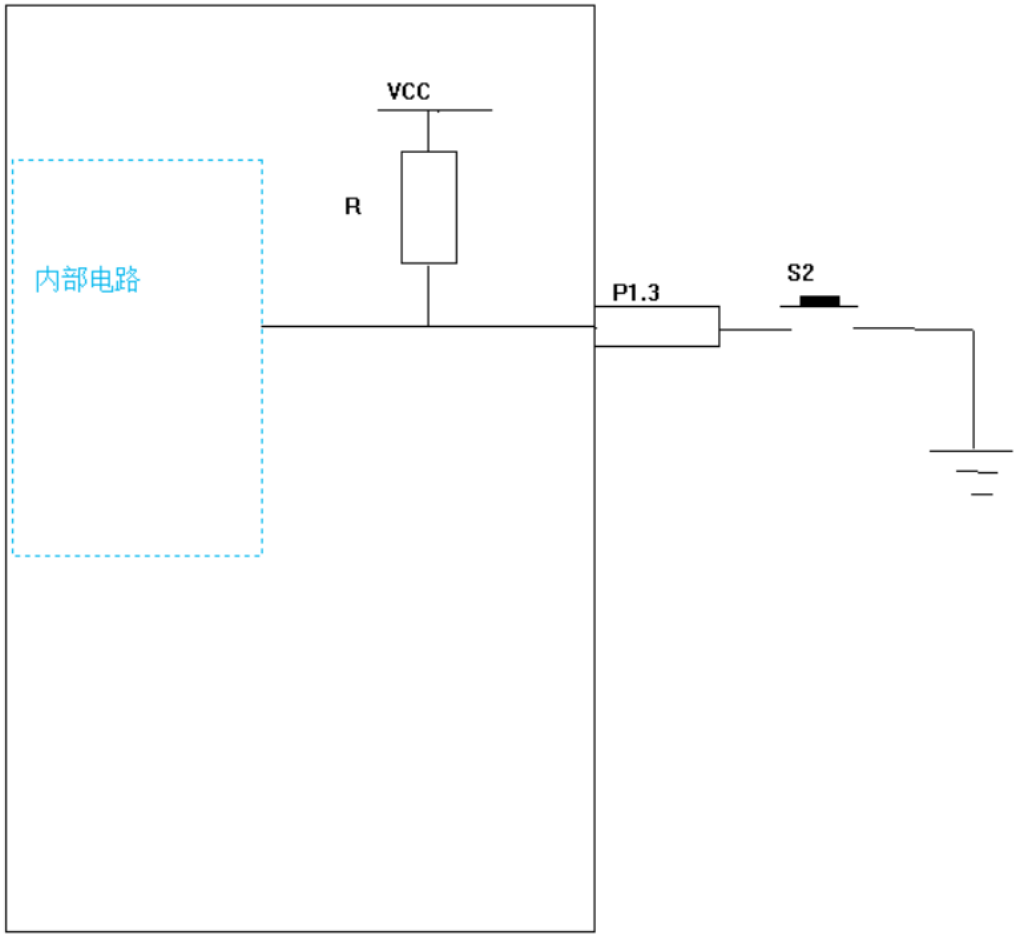
你的例子：如果允许 T0 中断，你再人为的置 T0 中断标志，CPU 就会响应了。

2.手头的活（main（））、去接电话（中断服务函数），这是两个不同的行动，相当于你本来执行的一个动作，现在要去执行另一个动作（所以跳倒中断函数里）

MSP430 内部有一个中断向量表，只要往里面对应向量（实际也是某地址）填相应函数的首地址，该中断发生时，就能根据填入的函数首地址自动【跳转到该函数】。

#pragma vector=这句话就是给这个中断向量表进行赋值的。
#pragma vector=PORT1_VECTOR 表示给这个向量表中的【外部端口中断】所对应的地址进行赋值，并赋值为函数 void Port_1(void)的首地址。
其中，关键字__interrupt 则表示这个函数是【中断服务函数】这种特殊函数。

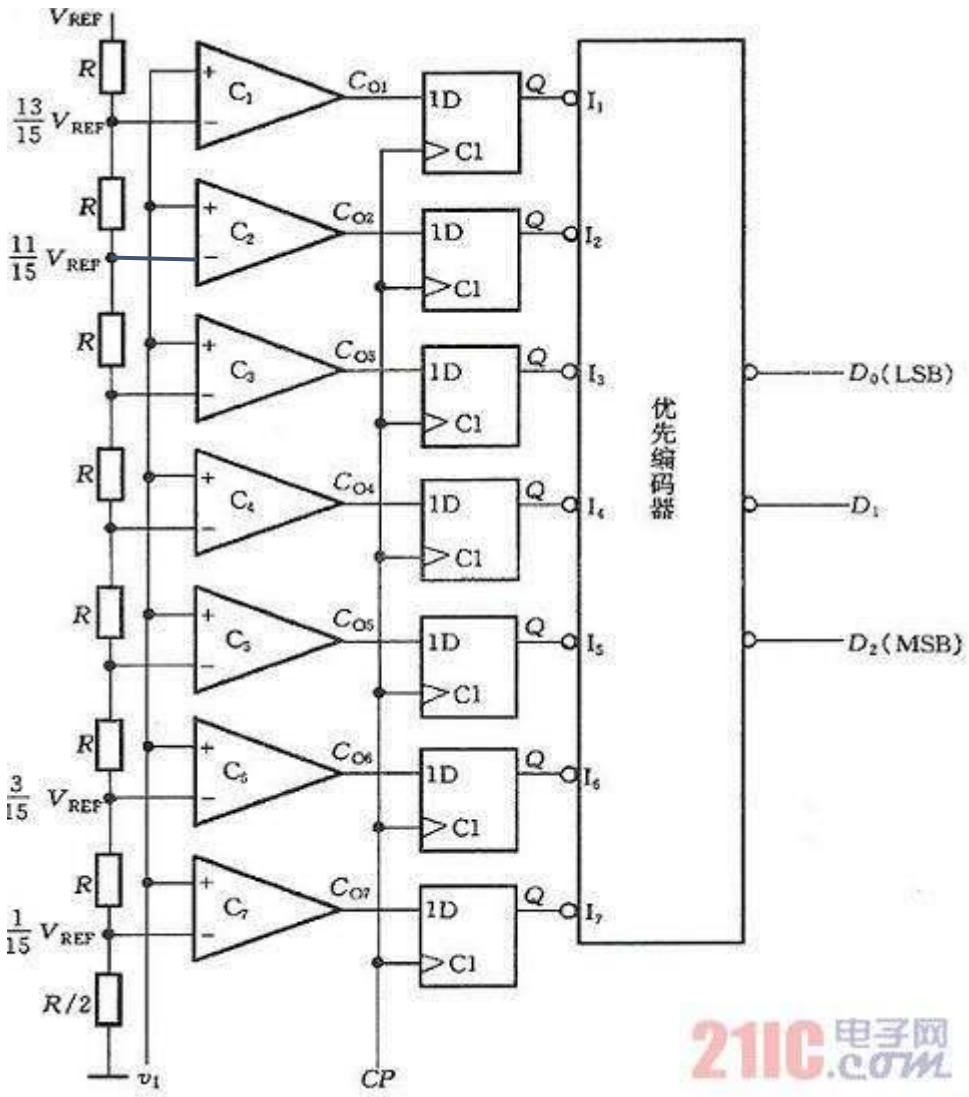
3.你接完电话之后，回来继续做你原来做的事情（中断标志位清零，CPU 一看，清零了，就认为中断的任务已经完成，于是就退出了中断）



按键消抖：http://baike.baidu.com/link?url=MxLcKIvzshA8ITWRxcVhgk-3nbn6wwzOu1mGlkzvnjo6TsborM9n4q5sypp5w213hznFMAMvFJGr52RacjtbYgYiO-xGVCu8WjsCE-JHC8gb8gd-SKWPKOcg27kuk8Vt

上拉电阻、下拉电阻：http://www.dzsc.com/data/2016-6-18/109978.html
http://www.bubuko.com/infodetail-449501.html
https://wenku.baidu.com/view/1623511552d380eb62946d55.html

**********ADC**********
模拟量：表示某种待转换量（可能是温度、湿度）的程度的电压值（可能是2.1v，也可能是1.3v）。
数字量：仅用0,1（高低电平）来表示的量。


V_{REF}
R
$\frac{13}{15}V_{REF}$
$\frac{11}{15}V_{REF}$
$\frac{3}{15}V_{REF}$
$\frac{1}{15}V_{REF}$
R/2
v_I
C_1
C_2
C_3
C_4
C_5
C_6
C_7
C_{O1}
C_{O2}
C_{O3}
C_{O4}
C_{O5}
C_{O6}
C_{O7}
1D
C1
Q
I_1
I_2
I_3
I_4
I_5
I_6
I_7
优先编码器
D_0(LSB)
D_1
D_2(MSB)
CP

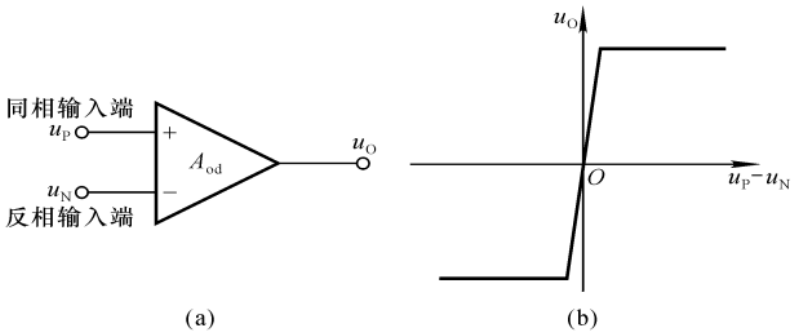


图 4.1.2　集成运放的符号和电压传输特性

(a) 符号　(b) 电压传输特性

集成运放的输出电压 u_o 与输入电压(即同相输入端与反相输入端之间的差值电压)之间的关系曲线称为**电压传输特性**,即

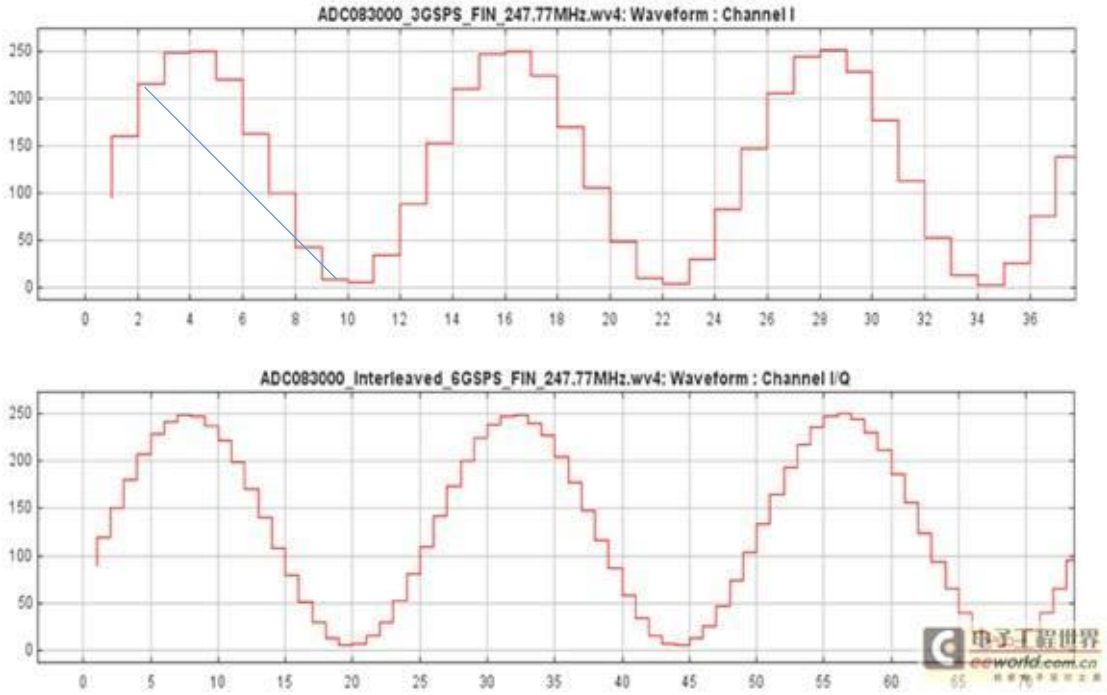
$$u_o = f(u_P - u_N) \tag{4.1.1}$$

对于正、负两路电源供电的集成运放,电压传输特性如图 4.1.2(b)所示。从图示曲线可以看出,集成运放有线性放大区域(称为**线性区**)和饱和区域(称为**非线性区**)两部分。在线性区,曲线的斜率为电压放大倍数;在非线性区,输出电压只有两种可能的情况,+ U_{oM} 或 − U_{oM} 。

十进制	二进制（3 位）
0	000
1	001
2	010
3	011
4	100

5	101
6	110
7	111

采样频率：



《MSP430 单片机基础与实践》

```
int_adc()
{
    P6SEL = 0x01;                                   //选择 AD 通道
    ADC12CTL0 = ADC12ON + SHT0_2 + REF2_5V + REFON;  //采样保持时间为 16 个 ADC12CLK
    ADC12CTL1 = ADC12SSEL1 + ADC12SSEL1;             //参考电压开启,选择 2.5 V
    ADC12MCTL0 = 0x10;                              // ref + = REF2_5V, channel = A0
    ADC12IE = 0x01;                                 // 使能转换中断
    ADC12CTL0 = ENC;                                // 使能 A/D 转换器
}
#pragma vector = ADC_VECTOR
__interrupt void ADC12ISR (void)
{
    while((ADC12CTL1&0X01) == 1);                   //等待转换完
    adc_flag = 1;
    AD_TEMP = ADC12MEM0;                            //设置 A/D 转换完成标志,并读取 ADC 值
}
```

软件部分告一段落，非常感谢大家看到这里，如果对你们有一些帮助，那我真的很高兴。我个人（不谦虚的讲），水平在实验室里真的非常一般，而且我一开始逻辑性天赋都很差，硬咬牙才学会了这些，效率上和最终熟练度、掌握知识的深度广度，与其他那些思维方式很好、逻辑性很强的人相比有很大差距，所以我水平的确有限，如果有错误，请直接弹幕，感谢大家不吝赐教！