利 尔 达 科 技 有 限 公 司

技术一部

LSD SCIENCE & TECHNOLOGY CO., LTD.

MSP430 相关 Q&A

Question & answer

版本号：V1.0

提交人：MSP430 项目组

整理时间：2009 年 02 月

目　　录

声明：所有问题均来自网络，部分答案也同样来自网路，答案仅供参考，并不能完全解决在使用中碰到的问题。所以客户在使用 MSP430 单片机时还是以实际情况来决定。

如有任何疑问欢迎与我们联系：haoqiang@lierda.com、gufeng@lierda.com、chenbing@lierda.com

第一章：开发工具类

Q1: 我自己做了一块 MSP430F149 的试验板，以前用下载线进行调试没有出现过问题，但是，最近我每次 make 后用下载线调试时，总是弹出一个窗口，给我提示：Could not find target status. 然后就死到那儿了，请问这是什么问题呢？

A1： 检查 Jtag 口线是否连接正常，如果 JTAG 口线连接正常，可能是供电不足，目标板加电再测试。

Q2：我用的 430f22x 学习套件,请问在 IAR Embedded Workbench 中仿真时如何看程序运行时间.

A2： 只有软件模拟下可以看， VIEW-REGISTER-CYCLECOUNT

Q3：请问各位 msp430 仿真器和编程器有什么区别啊?是不是我开发的时候这两个东西都得有?我目前用的是 msp430cg461x 系列或 msp430fg461x 系列，是不是很多仿真器和编程器都不支持?

A3： 一般来讲，仿真器是在先期调试程序时使用的，他不会烧断单片机熔丝，能把程序下载到单片机中，能够单步，跟踪，快速调试。编程器就没有这些调试功能，就是单纯把你做好的程序的编译后文件写到单片机中去，就和 51 的编程器一样，有加密熔丝烧断等功能，是在你产品成型后，生产时使用的

MSP430 的仿真器是使用 JTAG 接口的，分别有四线制的 JTAG、带 TEST 脚的四线 JTAG 和两线制的 SBWJTAG 三种接口，UIF 上三种都支持，并支持烧熔丝，UIF 就是 USB 接口的仿真；PIF 不能支持 SBWJTAG 接口，也不能烧熔丝，PIF 是并口的仿真器。任何一种 JTAG 接口的仿真器在烧断熔丝后都不能再仿真和写入，而 BSL 可以通过密码访问 FLASH 空间，读出写入均可，BSL 是串口实现的，但 BSL 不能仿真，注意部分器件不支持 BSL，如 F20XX 系列就不能用 BSL，烧掉了熔丝就变板砖。

MSP430 任何系列的仿真器只要接口方式一致都是兼容的，比如 FG461X，可以使用标准的带 TEST 的四线 JTAG，而 F22X4 也可以使用带 TEST 的四线 JTAG，当然 F22X4 还可以选择使用 SBWJTAG，它支持两种 JTAG 接口。如果不是 TI 标准的 430 系列用 JTAG 仿真器那就不行了，应该是不兼容的，不是什么“很多仿真器和编程器都不支持”，而是专用。

Q4： 初次使用 430 单片机，F149 。高速晶振 8M，低速 32K，下载程序时出现 security fuse **own 字样，无法烧录。请各位高人指点，应该是哪里有问题。

A4: 手工复下位应该基本可以解决，除非芯片熔丝真被烧掉了。如果熔丝断了,可以用 BSL 方式写入程序.

Q5： 运行到断点时程序不能自动结束，必须自己按 break。我这是程序跑飞了么？

A5：是你断点设置的多了吧。你在程序中,只设置这一个断点试试

Q6： 当在 RAM 中对其中一变量开辟的尺寸是 512 字节时,程序总占用 3.5K RAM 空间,可以正常烧到芯片中,也可以正常运行;

当对同一变量开辟的尺寸是 1024 字节，而其它均未改变时,程序总占用 4.0K 左右 RAM 空间,可以正常烧到芯片中,却无法运行!!!!

A6： 方法一 将一些变量定义成 no init 类型

方法二 修改 IAR 的 cstartup.s43 程序，具体方法如下：（iar310a 为例）

1 将 cstartup.s43 程序加载到用户自己的项目中，cstartup.s43 在 iar310a 的路径如下$TOOLKIT_DIR$\src\LIB\

2 修改 cstartup.s43 中的__program_start 子程序，加入关闭看门狗的命令 MOV #0x5A80,&0x0120

3 在 Project->Options->Linker->Config 页中选择 Override default programe ，并将 Entry lib 设置成 __program_start

Q7： 请问 AR 编译器的 s43 文件用什么编辑器打开？

A7：打开 IAR 编译器后就可以直接打开了；另外，记事本也可以打开

Q8：怎么设置才能使 2274 内部 dco 产生稳定的 16M 的时钟信号，还有就是 2274 是不是提供 FLL+ 功能！

A8：2XXX 没有 FLL+，但是 TI 在出厂芯片的时候已经做过出厂测试，你只要选择它头文件里的那个 16M 即可

Q9：程序编译时提示：Warning[Pe001]: last line of file ends without a newline ，这是怎么回事啊，应该怎么改呢？

A9：敲个回车就可以了

Q10：过去用 435/436 等因为仿真的几要线和端口都没有复用.所以很轻松就搞定.现在 1232 的仿真口和 P1 口是公用的,仿真时我把第一脚 TEST 接 VCC,这样可以下载仿真了,可是发现这复用的几个 P1 口就不能执行其正确的动作了,请教

A10：做仿真口的复用管脚在仿真时，被作为仿真功能管脚使。

Q11：USB 仿真器下载汇编程序时没有问题，但是下载 C 语言程序时，出报警信息

如下：

The stack plug-in failed to set a breakpoint on "main".The stack window will not be able to display stack contents. (You can change this setting in the Tools>Options dialog box.

在调试信息窗口出现 operation error.

A11：调试 c 程序时

在 Tools>Options dialog box 中 stack 要选中指向 main 函数处。

汇编和 c 要建不同的工程下调试。

Q12：装了 **MSP-FET430UIF** 仿真器，但是在 **AQ430** 的 **Options** 中，无法找到该仿真器，正常应该有 **LPT1/LPT2/LPT3/TIUSB** 四个选项，为什么看不到 **TIUSB**？

A12:先看看硬件管理器中有没有那个硬件，有则先用 **IAR** 试试**.**

Q 13：EZ430-RF2500 中的 USB debugging 可以调试 msp430 其他支持 Spy-Bi-Wire 系列的芯片吗?

A13：可以。

Q 14：用 TI 的 USB 仿真器，采用的是二线法。用 SD_16 采集数据，发现连接仿真器输出的数据正确，而把仿真器拔去，输出的数据就错了。这是为什么？

A14：怎么理解这个数据正确和错误？你怎么下的判断？我认为是你 接仿真器的时候目标板是从仿真器上取电的，拔掉仿真器是外部电源供电的，电源电压不同，将导致基准源改变，采样出来的值变化会很大。

Q15：sd-fet430 uif 仿真器不能外供电吗？

A15：可以，驱动电流较小，内部是光 MOS 管，要注意负载

Q16：430FET 下载出现问题时检查的一般思路

A16：在完全确定无虚焊、短路的情况下，一般注意以下事项：

1、仿真器的2、4脚电源
2、复位脚的RC电路
3、是否是SBW模式
4、JTAG复用引脚时注意外部电路的影响
5、外部有功率较大的器件，当下载时造成外部电路的功耗较大
6、JTAG线不能太长，一般在20CM以内比较合适
7、如果是 USB 的仿真器还要注意 USB 线不能太长，使用笔记本的还要注意 USB 上不能连接太多的负载

Q17：LSD-FET430UIF 仿真器，运用 IAR410 软件，器件是 F1121A，无法下载程序也无法仿真，直接将接口线换在原来的 FETP430IF 上，把 FET 设置换成并口后就可以，请问下，是什么原因造成的？
A17：检查一下驱动安装对吧，另外再检查下 USB 提供的电源和 USB 线的信号，USB 提供电源不足或 USB 线的信号受到了比较强的干扰都会影响下载。

Q18：430 加密用的密码是中断向量的简单组合呢？还是中断向量的加密组合？还是用的中断服务程序入口地址的简单组合？还是用的中断服务程序入口地址的加密组合？
A18：是入口地址的组合(存放在中断向量处的数据)，以前是简单组合，新的 5XX 就有些复杂了，用户自己还可以定义密码。

Q19：我的程序编译后,DEBUG 调试显示已成功 DOWN 到芯片里了,连着 EZ430 仿真器也能工作,但只要把 EZ430 脱开,PCB 重新上电,芯片怎么也不工作. 不知是什么原因?
A19：2013 的 RST 脚接 1 个上拉电阻就解决问题了。

Ｑ２０：通过标准的 BSL 方式是不需要编写单片机程序的？ 如果是的话，那自定义串口下载的优点和缺点又是什么？
A 20：用 BSL 不需要自己编写单片机程序，但需要另加入两根或 4 根编程线。自己写升级程序直接使用串口，但需要自己写自编程程序，此方式使用方便，还可以在线大批量升级，缺点是要占用 FLASH 空间，如果通讯协议严谨，再加上密码保护之类的算法，消耗 2 K 的ＦＬＡＳＨ资源还是比较正常的，所以小 FLASH 容量的就不推荐使用了。

Q21：在编译程序时,总是出现这样的错误信息,该怎么样解决呀?
Error[e46]: Undefined external "main" referred in ?cstart (D:\Program Files\iar\430\LIB\CLIB\cl430f.r43)
程序没有错误,是不是编译器本身有什么问题?
A21：是汇编程序时，把 OPTIONS 里的 ASSEM××ER ONLY ……选上。

Q22：UIF 连接不到 F2274，RST 信号的电容已去掉，用并口能连接上，不稳定，这是为什么？
A22：F22X4 支持两种 JTAG 接口，如果是 UIF 的话，个人建议使用 SBW 的，如果用 4 线制的话，会占用数个复用端口，你说连不上是不是就是因为复用端口上连有外围设备造成 JTAG 无法正常工作了呢。检查一下。

Q23：故障现象：有时 windows 能识别，多数不能识别；提示“未知 usb 设备”；挑机器－换台机器可能好使；升级失败；这是什么原因？
A23：1.使用 IAR 自动更新功能，只能通过 JTAG 口重新写入 f1612 程序。
2.EZ430 板上 TUSB3410 的复位脚 C9 电容小，增加到 2uF。最好增加旁路滤波电容，以提高电源质量。

Q24：使用 430 内置的 info flash 来存放一些配置信息。在用 fet 硬件仿真的时候，希望直接手动修改 memory 窗口中的的 info flash 的内容，但是老是弹出窗口说这些地址是不可访问等提示，请问有什么办法可以在硬件仿真的时候， 直接修改 info 中的内容么？

A24：flash 不能直接修改。这个得用编程器，仿真器或者软件编程修改 FLASH,你这样就能修改 FLASH 那也太……嘿嘿……

Q25： IAR 如何查看生成的代码的大小 ？

A25：编译后在信息窗口就是就可以提示,如果没有提示，请在 Tool/Options/Meassages/show build message:选为 All 就行了。

Q26：程序下不进去，用的仿真器是一头并口，一头 14 脚的 jtag，烧的时候总说找不到设备？

A26：RST 连上，去掉上面的电容。

Q27：给 MSP430F149 仿真的 JTAG 能给 MSP430F2002 仿真吗？
我把腿都对应上了，但程序下不进去，说是找不到器件.....
以前用同样的方法成功的给 449 仿真过!!! 难道 2002 用的是不同的 JTAG

A27：并口仿真器支持 2002，不过只能使用四线 JTAG 方式，RST 引脚必须连接，而且不要有复位电容。

Q28：430 如何将程序成功烧入？1.烧片子的具体的顺序？

A28：首先 option 里得选择正确的芯片型号，还要在 Debugger 选项卡里的 driver 选择 FET_Debugger，在按工具栏内的下载按钮（快捷键 ctrl+D）

Q29：MSP430F22X 学习板 SBW 用访真器不能进行调试？

A29： 原因可能两点
两线制是不能用并口访真器进行调试的，必须 USB 访真器
两线制 RST 并联的电容建议不接，否则造成程序无法正常下载

Q30：在研制带处理器的电子产品时，如何提高抗干扰能力和电磁兼容性？

A30：一、下面的一些系统要特别注意抗电磁干扰：
1、微控制器时钟频率特别高，总线周期特别快的系统。
2、系统含有大功率，大电流驱动电路，如产生火花的继电器，大电流开关等。
3、含微弱模拟信号电路以及高精度 A/D 变换电路的系统。

二、为增加系统的抗电磁干扰能力采取如下措施：
1、选用频率低的微控制器：

选用外时钟频率低的微控制器可以有效降低噪声和提高系统的抗干扰能力。同样频率的方波和正弦波，方波中的高频成份比正弦波多得多。虽然方波的高频成份的波的幅度，比基波小，但频率越高越容易发射出成为噪声源，微控制器产生的最有影响的高频噪声大约是时钟频率的 3 倍。

2、减小信号传输中的畸变

a、微控制器主要采用高速 CMOS 技术制造。信号输入端静态输入电流在 1mA 左右，输入电容 10PF 左右，输入阻抗相当高，高速 CMOS 电路的输出端都有相当的带载能力，即相当大的输出值，将一个门的输出端通

过一段很长线引到输入阻抗相当高的输入端，反射问题就很严重，它会引起信号畸变，增加系统噪声。当 Tpd＞Tr 时，就成了一个传输线问题，必须考虑信号反射，阻抗匹配等问题。

b、信号在印制板上的延迟时间与引线的特性阻抗有关，即与印制线路板材料的介电常数有关。可以粗略地认为，信号在印制板引线的传输速度，约为光速的 1/3 到 1/2 之间。微控制器构成的系统中常用逻辑电话元件的 Tr（标准延迟时间）为 3 到×××s 之间。

c、在印制线路板上，信号通过一个 7W 的电阻和一段 25cm 长的引线，线上延迟时间大致在 4~20ns 之间。也就是说，信号在印刷线路上的引线越短越好，最长不宜超过 25cm。而且过孔数目也应尽量少，最好不多于 2 个。

d、当信号的上升时间快于信号延迟时间，就要按照快电子学处理。此时要考虑传输线的阻抗匹配，对于一块印刷线路板上的集成块之间的信号传输，要避免出现 Td＞Trd 的情况，印刷线路板越大系统的速度就越不能太快。

e、用以下结论归纳印刷线路板设计的一个规则：

f、信号在印刷板上传输，其延迟时间不应大于所用器件的标称延迟时间。

3、减小信号线间的交叉干扰：

a、A 点一个上升时间为 Tr 的阶跃信号通过引线 AB 传向 B 端。信号在 AB 线上的延迟时间是 Td。在 D 点，由于 A 点信号的向前传输，到达 B 点后的信号反射和 AB 线的延迟，Td 时间以后会感应出一个宽度为 Tr 的页脉冲信号。在 C 点，由于 AB 上信号的传输与反射，会感应出一个宽度为信号在 AB 线上的延迟时间的两倍，即 2Td 的正脉冲信号。这就是信号间的交叉干扰。干扰信号的强度与 C 点信号的 di/at 有关，与线间距离有关。当两信号线不是很长时，AB 上看到的实际是两个脉冲的迭加。

b、CMOS 工艺制造的微控制由输入阻抗高，噪声高，噪声容限也很高，数字电路是迭加 100~200mv 噪声并不影响其工作。若图中 AB 线是一模拟信号，这种干扰就变为不能容忍。如印刷线路板为四层板，其中有一层是大面积的地，或双面板，信号线的反面是大面积的地时，这种信号间的交叉干扰就会变小。原因是，大面积的地减小了信号线的特性阻抗，信号在 D 端的反射大为减小。特性阻抗与信号线到地间的介质的介电常数的平方成反比，与介质厚度的自然对数成正比。若 AB 线为一模拟信号，要避免数字电路信号线 CD 对 AB 的干扰，AB 线下方要有大面积的地，AB 线到 CD 线的距离要大于 AB 线与地距离的 2~3 倍。可用局部屏蔽地，在有引结的一面引线左右两侧布以地线。

4、减小来自电源的噪声

电源在向系统提供能源的同时，也将其噪声加到所供电的电源上。电路中微控制器的复位线，中断线，以及其它一些控制线最容易受外界噪声的干扰。电网上的强干扰通过电源进入电路，即使电池供电的系统，电池本身也有高频噪声。模拟电路中的模拟信号更经受不住来自电源的干扰。

5、注意印刷线板与元器件的高频特性

a、在高频情况下，印刷线路板上的引线，过孔，电阻、电容、接插件的分布电感与电容等不可忽略。电容的分布电感不可忽略，电感的分布电容不可忽略。电阻产生对高频信号的反射，引线的分布电容会起作用，当长度大于噪声频率相应波长的 1/20 时，就产生天线效应，噪声通过引线向外发射。

b、印刷线路板的过孔大约引起 0.6pf 的电容。

c、一个集成电路本身的封装材料引入 2~6pf 电容。

d、一个线路板上的接插件，有 520nH 的分布电感。一个双列直扦的 24 引脚集成电路扦座，引入 4~×××H 的分布电感。

e、这些小的分布参数对于这行较低频率下的微控制器系统中是可以忽略不计的；而对于高速系统必须予以特别注意。

6、元件布置要合理分区

元件在印刷线路板上排列的位置要充分考虑抗电磁干扰问题，原则之一是各部件之间的引线要尽量短。在布局上，要把模拟信号部分，高速数字电路部分，噪声源部分（如继电器，大电流开关等）这三部分合理地分开，使相互间的信号耦合为最小。

7、处理好接地线

a、印刷电路板上，电源线和地线最重要。克服电磁干扰，最主要的手段就是接地。
b、对于双面板，地线布置特别讲究，通过采用单点接地法，电源和地是从电源的两端接到印刷线路板上来的，电源一个接点，地一个接点。印刷线路板上，要有多个返回地线，这些都会聚到回电源的那个接点上，就是所谓单点接地。所谓模拟地、数字地、大功率器件地开分，是指布线分开，而最后都汇集到这个接地点上来。与印刷线路板以外的信号相连时，通常采用屏蔽电缆。对于高频和数字信号，屏蔽电缆两端都接地。低频模拟信号用的屏蔽电缆，一端接地为好。
c、对噪声和干扰非常敏感的电路或高频噪声特别严重的电路应该用金属罩屏蔽起来。

8、用好去耦电容。

a、好的高频去耦电容可以去除高到 1GHZ 的高频成份。陶瓷片电容或多层陶瓷电容的高频特性较好。设计印刷线路板时，每个集成电路的电源，地之间都要加一个去耦电容。去耦电容有两个作用：一方面是本集成电路的蓄能电容，提供和吸收该集成电路开门关门瞬间的充放电能；另一方面旁路掉该器件的高频噪声。数字电路中典型的去耦电容为 0.1uf 的去耦电容有 5nH 分布电感，它的并行共振频率大约在 7MHz 左右，也就是说对于 10MHz 以下的噪声有较好的去耦作用，对 40MHz 以上的噪声几乎不起作用。
b、1uf，10uf 电容，并行共振频率在 20MHz 以上，去除高频率噪声的效果要好一些。在电源进入印刷板的地方和一个 1uf 或 10uf 的去高频电容往往是有利的，即使是用电池供电的系统也需要这种电容。
c、每 10 片左右的集成电路要加一片充放电电容，或称为蓄放电容，电容大小可选 10uf。最好不用电解电容，电解电容是两层溥膜卷起来的，这种卷起来的结构在高频时表现为电感，最好使用胆电容或聚碳酸酐电容。
d、去耦电容值的选取并不严格，可按 C=1/f 计算；即 10MHz 取 0.1uf，对微控制器构成的系统，取 0.1~0.01uf 之间都可以。

三、降低噪声与电磁干扰的一些经验。

1、能用低速芯片就不用高速的，高速芯片用在关键地方。
2、用串一个电阻的办法，降低控制电路上下沿跳变速率。
3、尽量为继电器等提供某种形式的阻尼。
4、使用满足系统要求的最低频率时钟。
5、时钟产生器尽量靠近到用该时钟的器件。石英晶体振荡器外壳要接地。
6、用地线将时钟区圈起来，时钟线尽量短。
7、I/O 驱动电路尽量靠近印刷板边，让其尽快离开印刷板。对进入印制板的信号要加滤波，从高噪声区来的信号也要加滤波，同时用串终端电阻的办法，减小信号反射。
8、MCD 无用端要接高，或接地，或定义成输出端，集成电路上该接电源地的端都要接，不要悬空。
9、闲置不用的门电路输入端不要悬空，闲置不用的运放正输入端接地，负输入端接输出端。 (10) 印制板尽量使用 45 折线而不用 90 折线布线以减小高频信号对外的发射与耦合。
10、印制板按频率和电流开关特性分区，噪声元件与非噪声元件要距离再远一些。

11、单面板和双面板用单点接电源和单点接地、电源线、地线尽量粗，经济是能承受的话用多层板以减小电源，地的容生电感。
12、时钟、总线、片选信号要远离 I/O 线和接插件。
模拟电压输入线、参考电压端要尽量远离数字电路信号线，特别是时钟。
13、对 A/D 类器件，数字部分与模拟部分宁可统一下也不要交叉。
14、时钟线垂直于 I/O 线比平行 I/O 线干扰小，时钟元件引脚远离 I/O 电缆。
15、元件引脚尽量短，去耦电容引脚尽量短。
16、关键的线要尽量粗，并在两边加上保护地。高速线要短要直。
17、对噪声敏感的线不要与大电流，高速开关线平行。
18、石英晶体下面以及对噪声敏感的器件下面不要走线。
19、弱信号电路，低频电路周围不要形成电流环路。
20、任何信号都不要形成环路，如不可避免，让环路区尽量小。
21、每个集成电路一个去耦电容。每个电解电容边上都要加一个小的高频旁路电容。
22、用大容量的钽电容或聚酷电容而不用电解电容作电路充放电储能电容。使用管状电容时，外壳要接地

Q31：有关仿真器及编程器的问题：
A31：对于 51 系统来说，很容易理解编程器与仿真器的区别。
通俗的说，仿真器是用来仿真调试的，编程器是用来批量生产时对 MCU 进行烧写目标代码的。
对于 MSP430 来说，无论仿真还是烧写程序一般可以通过：JTAG/SBW/BSL 接口进行，这些概念仅是接口，并不代表哪个型号的编程器和仿真器，一般 JTAG/SBW 接口用于仿真接口，BSL 不能用于仿真，只能用于编程。编程器则三种接口都支持。所以并不能说 JTAG 只支持仿真，不支持编程，这是概念错误，JTAG 仅仅是一种接口协议而已。
下面简单描述一下三种接口的区别：
1、JTAG 是边界扫描技术，其在 430 内部有逻辑接口给 JTAG 使用，内部有若干个寄存器连接到 430 内部的总线上，所以 JTAG 可以访问到 430 的内部所有资源，包括对 FLASH 的读写操作。可以用于编程和仿真接口。主要连接线有 TMS/TCK/TDI/TDO/RST/TEST 等。
2、SBW 是 SPY-BI-WIRE 的简称。通常称为两线制 JTAG，主要用 SBWTCK 与 SBWTDIO，该接口主要用于小于 28 脚的 430 单片机及 5 系列的单片机。因为 28 脚以内的单片机 JTAG 接口与通用 IO 口复用，为了给用户预留更多的 IO，才推出 SBW 接口。同样 SBW 接口可以用于仿真器与编程器。
3、BSL 是 TI 在 MSP430 出厂时预先固化到 MCU 内部的一段代码，有点类似与 DSP 的 bootloader，但与 bootloader 有明显的区别，BSL 只能用于对 MCU 内部的 FLASH 访问，不能对访问其他资源。所以只能用作编程器接口。BSL 通过 UART 协议与编程器连接通信。编程器可以发送不同的通信命令来对 MCU 的存储器做不同的操作。BLS 的启动有些特殊，一般 430 复位启动时 PC 指针指向 FFFE 复位向量，但可以通过特殊的启动方式可以使 PC 指向 BSL 内部固化的程序。启动方式一般由 RST 与 TEST（或 TCK）引脚做一个稍复杂的启动逻辑后产生。
4、一般的 MCU 都有加密功能，430 如何实现加密的呢？外部对 430 内部的代码读写只能通过上述的三种方式，所以引入熔丝位，熔丝位只存在与 JTAG/SBW 接口逻辑内。BSL 没有熔丝。当熔丝烧断时（物理破坏，且不可恢复）JTAG/SBW 的访问将被禁止，此时只有 BSL 可以访问。而通过 BSL 对 MCU 的访问是需要 32 字节的密码，密码就是用户代码的中断向量表。
5、一般的仿真器型号有：UIF（USB 接口，支持 JTAG/SBW）、PIF（并口，只支持 JTAG）、EZ430（USB 接口，只支持 SBW 模式）。专业的编程器有 GANG430（串口，一拖 8 个目标板，支持 JTAG/SBW 及烧断熔丝功能）；多功能编程器（JTAG/SBW/BSL 及烧断熔丝功能）。这些编程器都可以支持离线烧写，即脱离计算机来对目标板烧写。当然也可以通过特殊的软件用仿真器来烧写，这类软件有 MSPFET、FET-PRO430 等。

Q32：最近调试程序，不知怎么老出现下面的问题，请问是怎么回事？
”erase check error at address 0x1100
retry erase operation?"
知道是怎么回事？
A32：最可能的就是芯片电压太低，或者芯片有虚焊的。

Q33：请问 MSP430F149 程序下载时的电源要提供的电流为多大？
A33：这个电流主要用于 flash 的擦写。擦和写的典型电流都是 3mA。

第二章：指令系统

Q1. IAR中怎样描述P2OUT.3脚，#define LCD_cs1 P2OUT.3; 对吗?
A1：430 不能位寻址，所以一般的位操作，都通过“与”来作用。#define LCD_cs1 (P2OUT&BIT3)。

Q2.__intrinsic是什么意思
A2：本征函数，不是C语言标准库，而是和MSP430汇编直接对应的函数，比如：
_NOP()
_EINT()
LPM0
引用 msp430xxxx.h，默认引用的头文件

Q3：430中使用C和汇编语言的区别，优缺点比较？
A3：严格来讲 430 的 C 是 ANSI C 的一个子集，与汇编的差别主要有：
1、C 有 if、（do）while、switch 等流程控制语句
2、C 有有限的数据格式，如 char、int、float、dou××e 等
3、对 430 最有特色的 R0--R15 的使用，C 不如汇编
4、430 的 C 不易进行 RAM 管理
5、430 各版本的 C，互相之间存在差异，好象 C 在 430 上还不成熟
6、C 的优点是在 PC 或 PDA 上，也就是在有操作系统的平台上，C 的优点才会表现出来，但那已经不是 430 的 C 了，而是 C++，它有丰富的数据类型，如结构、对象等
7、汇编的缺点，基本上就是 C 的优点，而汇编的优点基本上都是 C 的缺点
8、C 函数有类型限定的形参和返回值以及强制类型转换，还有格式化输出，发挥 C 的这些特长，再结合结构化程序设计技术，容易编出思路清晰、结构灵活、可移值性比较强的程序。汇编也可以满足上述要求，但相对来讲要复杂些。
9、初学者建议先用汇编，由于对硬件的操作更加直观灵活，有利于对 430 内部结构的熟悉和熟悉编程思路。等掌握了 30 的开发技能，又熟悉了系统的合理分配资源和组织资源时再学习用 c，那将使你进步最快的时候，因为 c 对程序的结构和调度真得很理想！
10、源代码隐藏性比较好，比较安全。

Q4：MSP430指令系统中，符号寻址方式有点不太理解。例如连续的两条指令是：
MOV.B #20H,0012H
MOV.B #34H,0200H
在反汇编窗口看这两条指令是：
mov.b #0x20,0x1112
mov.b #0x34,0x1300
不理解的是：
1.为什么反汇编后指令的第二个操作数地址变成了0x1112和0x1300？
2.上述反汇编指令似乎都在原有的地址基础上加上了0x1100，我知道保存程序的FLASH是从0x1100开始的。但实际在执行上述两条指令时，PC的值早就>0x1100.符号寻址操作数地址应该=(PC+0012H)、(PC+0200H)。而PC值在执行指令中应该是变化的？

A4：0012H 和 0200H 是绝对地址。问题中程序用的是立即寻址，实际上应该用绝对寻址

MOV.B #0030h,&0200h 就是前面操作数前面有一个&；没有‘&",MOV.B #0030h,0200h
那么地址就应该= PC addrress +0200(偏移量)。反编译出来的代码应该是
mov.b #0x20,&0x1112
mov.b #0x34,&0x1300
才对。

Q5：在程序里循环调用了sprintf函数，当循环了几次后程序就跑飞了，是堆栈大小不够吗？还有sprintf到底需要多大的堆栈，单步走了下，发现它使用堆栈都不是连续的？
A5：有可能是的 sprintf 函数堆栈数据把跳转地址挤掉，可以调开堆栈看一下。

Q6：问 MSP430 MOV 命令
立即数寻址
MOV @PC+, X(PC);
偏移量可变址寻址
MOV X(PC), Y(PC);
怎么理解？

A6：MOV @PC+, X(PC);
将 PC 所指单元的内容送给 PC+X 所在单元,然后 PC 本身加 1,指向相邻的下一单元.
MOV X(PC), Y(PC);
将 PC+X 所指单元的内容送给 PC+Y 所在单元

问题7：数据类型Small,Medium,Large的区别，以及怎么访问大于0X10000的地址空间？
数据放在大于 0X10000 的地址空间中，怎么去访问？同时 Small,Medium,Large 在何处可以体现它们的区别？
A7：可以访问。值得注意的是 option 里的 General Option 选项内的数据类型得选择 Large，程序大小，RAM 使用，运行速度都不一样。
1.F5XX 430X
2.选择 Large small medium 产生效果是改变指针变量所占字节个数.IAR 中指针变量默认 2 个字节,选择 large 占用 4 个字节,那么指针就可以访问超过 64K 范围的地址.
3.可以采用 IAR 内部函数, */
void __data20_write_char (unsigned long __addr,
unsigned char __value);

void __data20_write_short(unsigned long __addr,
unsigned short __value);

void __data20_write_long (unsigned long __addr,
unsigned long __value);

unsigned char __data20_read_char (unsigned long __addr);
unsigned long __data20_read_long (unsigned long __addr);

Q8：关于堆栈问题
问题：Error[e16]: Segment CSTACK (size: 0x50 align: 0x1) is too long for segment definition. At least 0x4 more bytes needed. The pro××em occurred while processing the

segment placement command "-Z(DATA)CSTACK+_STACK_SIZE#", where at the moment of placement the ××aila××e memory ranges were "CODE:5B4-600"
请问：这个错误是什么意思

A8：将 stack 设置减小,可通过编译,这时 stack 是静态的.
而会不会出错不是 stack 设置问题,而是在程序运行时,动态中,RAM 占用量大小所定.
减少程序中的变量个数，变量类型尽量小，够用就可以，能减少动态运行时 RAM 的占用量.

Q9：浮点数运算问题，既然2.2A编译器能实现64浮点位运算，那么它的运算程序在哪个文档里。
A9：要使用 64 位的浮点运算需要设置编译器
首先选择 project->option->general->target->dou××e floating point size
选中 "64"
再选择 project->option->XLINK->include->library->cl430d.r43
用 c++的朋友要选择 dl430d.r43 库
建议用整型数进行运算,比如像开方,64 位浮点数要用几十毫秒,而整型数随数据大小只要几微秒到几毫秒不等。

Q10：局部变量的问题，在中断服务程序中开中断响应其他中断，发现中断服务中定义的局部变量老被改动，换成全局变量没问题。想问一下这是啥原因？还想问一下c编程过程中应注意的细节？
A10：局部变量只是暂存变量,一般编译器会使用通用寄存器来保存这个变量值，汇编编程的话需要压栈操作，C 语言编程的话建议把变量申明为全局或者静态。

第三章：代码编程类

Q1： 我在一个子程序中定义了一个变量，是在其头文件中定义的，然后在主程序中包含了此头文件，当在主程序中引用此变量时 ，编译器没有报错，但是要 **DEBUG** 时出错了，告诉我是 **redefine in main**，请问这是怎么回事呢？

A1：肯定不能这样用了，相当于重复定义了．用外部变量声明可以解决这个问题．你可以在头文件中申明（加 **extern**），在 **C** 文件中定义。

Q2：这是我的接收函数，但是当我发送 **ko** 时，收到两个 **o**，发送 **ok** 时收到两个 **k,**这是怎么回事呢。

```
#pragma vector=UART1RX_VECTOR
__interrupt void usart1_rx (void)
{
unsigned j;
//IFG2 &= ~URXIFG1;
for (j=0;j<2;)
{
data[j] = RXBUF1;
j++;
}
if(j==2)
{
TX_Flag = 1;
}
```

A2：程序错了！！按你的程序，收到第一个字符 **K** 时，你将该字符填入数组，即数组内容为 {**K** ，**K**}；收到第二个字符 **O** 时，你将该字符填入数组，即数组内容为 {**O** ，**O**}，应该加一个判断，判断接收缓存里面的数据被存了，再存第二个。

Q3：问题: 想要根据一个数据 **send**，利用 **TA0** 产生中断，控制 **P2** 口产生 **3ms**、**1ms**、**3ms**、**1ms**、**3ms**、**1ms.....** 的一组方波（‘**1**’发送 **3ms**，‘**0**’发送 **1ms**），但程序写出来后，发现执行过程非常不稳定，忽快忽慢。不知道是单片机的问题还是程序的问题，请专家帮帮！

附程序：

```
void data_send (unsigned char send)
{
unsigned char i;
for (i=0;i<8;i++) //发送 8 位
{
if (send&0x01) //如果最低位为 1，发送 3ms
{
CCR0=189; //3ms 定时
CCTL0=CCIE; //打开中断
P2OUT|=BIT3; //P2 口变高电平
```

```
while (flag==0); //等待中断处理子程序返回标志位 flag=1
flag=0;  //flag 清零，以便下一位的中断
P2OUT&=~BIT3; //P2 口变低
}
else //当最低位为 0 时  中断 1ms
{
CCR0=63;
CCTL0=CCIE;
P2OUT|=BIT3;
while (flag==0);
flag=0;
P2OUT&=~send_BIT;
}
send=(send>>1); //左移一位
}
#pragma vector = TIMERA0_VECTOR //TA 中断
__interrupt void TA_start(void)
{
flag=1; //中断时间到，设置标志位
}
```

TA0 的时钟选用 0.5M 8 分频，近似计数值 63 为 1ms，使用连续计数方式。

```
A3：void data_send (unsigned char send)
{
unsigned char i;
for (i=0;i<8;i++) //发送 8 位
{
if (send&0x01) //如果最低位为 1，发送 3ms
{
CCR0=189; //3ms 定时
CCTL0=CCIE; //打开中断
P2OUT|=BIT3; //P2 口变高电平
while (flag==0); //等待中断处理子程序返回标志位 flag=1
flag=0;  //flag 清零，以便下一位的中断
P2OUT&=~BIT3; //P2 口变低
}
else //当最低位为 0 时  中断 1ms
{
CCR0=63;
CCTL0=CCIE;
P2OUT|=BIT3;
while (flag==0);
flag=0;
P2OUT&=~BIT3;
}
send=(send>>1); //左移一位
```

```
}
}
#pragma vector = TIMERA0_VECTOR //TA 中断
__interrupt void TA_start(void)
{
flag=1; //中断时间到，设置标志位
}
```

TACTL 中的 MC1、MC0 为 01 值

Q4：求 msp43 的 pwm 捕捉下降沿的程序。

A4：TACCTL2=CM1+CCIS_1+CAP; // 同步捕获模式，下降沿捕获

TACTL=TASSEL_2+MC_2+TACLR; // 连续计数方式，时钟设置为 SMCLK

I/O 口也需要设置为第二功能口，建议看 users guide 和 TI 例程。

Q5：C 文件 LINKER 后，报错 Error[e46]:Udefined external "send1_buff"referred in sys_evet(D:\sc1\Debug\Obj\sys_evet.r43，请教如何可解决此问题，变量 send1_buff 在文件里已定义了 extern。

A5：查看一下你的程序编译时先编译哪个文件,或在这个文件前先申明一下。

Q6：:DINT

```
MOV.W #FWKEY+WRT,&FCTL1 ; Write bit = 1
MOV.W #FWKEY+FSSEL0+FN0+FN1,&FCTL2
MOV.W #FWKEY,&FCTL3
TEST BIT #BUSY, &FCTL3
JNZ TEST
MOV.W R14,0(R12)
TEST1 BIT #BUSY, &FCTL3
JNZ TEST1
MOV.W #FWKEY,&FCTL1
MOV.W #FWKEY+LOCK,&FCTL3 ; Lock = 1
EINT
RET
```

想保存信息，R12 是地址，1080，R14 是内容，但是通过按键改变参数后，INFO 写入的确实 0，而不是要保存的数据。很奇怪。为什么不管什么数据，到 INFO 里都是 0？

A6：原因在于你只改变单个字，而 FLASH 是要整段擦除，然后在更改。MOV.W #FWKEY+FSSEL0+FN0+FN1,&FCTL2 ,flash 擦写时钟也是很重要的，经分频后时钟应该在 257-476KHz 的频率范围，偏离值太大，容易引起擦写的失败。

Q7：:为什么 485 程序只能单字节传输呢？单片机经过 232/485 转换器接到电脑，用串口精灵调试的。大家看看有什么问题啊？

```
#include <msp430x14x.h>
void main(void)
{ WDTCTL=WDTPW+WDTHOLD;
UCTL0 &=~SWRST;
UCTL0 |=CHAR;
```

```
UBR00 = 0X03;
UBR10 = 0X00;
UMCTL0 =0X4A;
UTCTL0 |= SSEL0;
ME1 |= UTXE0 + URXE0;
IE1|=URXIE0;
P3SEL |=0X30;
P3DIR |=0x70;
P3OUT&=~BIT6;
_BIS_SR(LPM3_bits + GIE); // Enter LPM3 w/ interrupt
}
#pragma vector=UART0RX_VECTOR
__interrupt void usart0_rx (void)
{
int i;
P3OUT|=BIT6;
while (!(IFG1 & UTXIFG0)); // USART1 TX buffer ready?
TXBUF0 =RXBUF0; // RXBUF1 to TXBUF1
for(i=0;i<10000;i++){_NOP();}
P3OUT&=~BIT6;
}
```

A7：程序问题：

1 发送第一个字节时候，中断服务程序中 **485** 使能端口操作（**P3OUT|=BIT6** 与 **P3OUT&=~BIT6**）之间的时间太长，导致这段时间的调试精灵发出的字节都丢失了。

2 一般可以先将所有的数据接受完成之后（即放到缓冲区后），在开始切换 4 8 5 端口发送数据，最后确保所有数据发送完成后在切换成接受方向主动发送,中断接收,在串口接收中断中不要做太多事情,只压接收到的字节数据。

Q8：初学 **430**，发现汇编语言中没有除法指令，请问版主和各位高手，能否提供一个 **32** 位**/16** 位的除法汇编程序，谢谢。

A8：**32** 位除以 **16** 位的自己改：

```
;ACCAx:除数,ACCBx:商,ACCCx:余数,ACCDx:被除数
;-------------------------------------------------------
Div48_32 MOV.B #030H,TEMP
CLR.W ACClierdalierda ;ACCBx:商
CLR.W ACCBM
CLR.W ACCBH
D_DIVS CLR.W ;ACCCx:余数
CLR.W ACCCH
DIV_LOOP CLRC
RLC.W ACCDL ;ACCDx:被除数
RLC.W ACCDM
RLC.W ACCDH
RLC.W ACCCL
RLC.W ACCCH
```

```
JC NOCHK1
MOV.W ACCCH,R15
SUB.W ACCAH,R15
JNZ NOCHK
MOV.W ACCCL,R15
SUB.W ACCAL,R15
NOCHK JNC NOGO
NOCHK1 SUB.W ACCAL,ACCCL
JC NOCHK2
DEC.W ACCCH
NOCHK2 SUB.W ACCAH,ACCCH
SETC
NOGO RLC.W ACClierdalierda
RLC.W ACCBM
RLC.W ACCBH
DEC.B TEMP
JNZ DIV_LOOP
RET
```

Q9：在 **TI** 的例子程序中有

TACCTL1 = CCIS0 + OUTMOD0 + CCIE;

TACCTL1 |= OUTMOD2;

怎么理解。

A9：**TACCTL1 = CCIS0 + OUTMOD0 ;** 是写整个寄存器**;**

执行后**,**寄存器 **TACCTL1** 的值**=0x1000+0x2000=0x3000;**

而 **TACCTL1 |= OUTMOD2;**只是把寄存器中的相关位置 **1,**剩余的位保持不变**;**

执行后**,**寄存器 **TACCTL1** 的 **BIT7(0x0080)**改写为 **1.**

Q10：在程序中作循环 ，执行时间比较长 。若循环次数少，则改变的数组中个数少 ；若多加，两次循环改变的个数就多。

A10：可能是堆栈溢出了！

Q11：想让 **menu_b** 定义在 **ROM** 区

unsigned int const menu_a[][3]={1,2,3};

unsigned int const menu_b[1]={&menu_a[0][0]};

IAR 编译器为何编译不了？但这样就行，但是 **menu_b** 是定义在 **RAM** 区的

unsigned int const menu_a[][3]={1,2,3};

unsigned int const *menu_b[1]={&menu_a[0][0]};

A11：**unsigned int const *menu_b[1]={&menu_a[0][0]};** “**unsigned int const ***” 表示指向 **const** 型（**ROM** 型）**unsigned int** 变量的指针，且由于前面没有再加一个 **const**（如 **const unsigned int const * xxx**），故按默认定义在 **RAM** 中。这是标准 **C** 的语法。！

Q12：最近才开始学 **MSP430**，看利尔达公司的一个汇编实验程序时，发现很多类似这样的指令，**mov.w #SHT0_2+ADC12ON,&ADC12CTL0** 这条指令是不是立即数寻址啊？如果是的话，**#**号后面不是直接加一个立即数

吗？这个上面怎么加了“SHT0_2+ADC12ON”这个啊？另外“SHT0_2+ADC12ON”这个都是“ADC12CTL0” 这个寄存器里面的一位或几位。这条指令是什么意思啊？

A12：SHT0_2，ADC12ON 已经宏定义好的，其实表示一个数， 这语句是对寄存器相应位置位。

Q13：用 **f147** 计算电压有效值，采样 **40** 个点，用平方和除以 **40** 再开根号，计算一次居然要 **10ms**，有这么长时间么？我的基础时钟模块这样设的

```
//set basic system time
BCSCTL1 |= XTS +XT2OFF; //HF model ,4.096M,XT2 close
BCSCTL2 |= SELM_3+SELS; //MCLK、SMCLK use LFXT1CLK,都不分频
//set ADC12
ADC12CTL0 |=SHT0_2 +REF2_5V+REFON +ADC12ON+MSC;
//采样周期为 8 个 ADC12CLK
//启用多次采样转换位
//2.5V 参考电平，打开 ADC 内核
//转换时间溢出中断和溢出中断未打开
ADC12CTL1 |=SHP + ADC12SSEL_1+CONSEQ_1; //使用采样定时器
//ACLK 做时钟源
ADC12MCTL0 |= INCH_0 + SREF_1; //ref+=Vref+, channel = A0
ADC12MCTL1 |= INCH_1 +SREF_1 + EOS; //ref+=Vref+, channel = A1
ADC12IE |= BIT1;
```

A13:是计算的时候出了问题，首先要求平方，在开根号，这样是算需要很多的时间！

Q14：程序运行报错：

```
Mon Oct 13 09:56:08 2008: Target reset
Mon Oct 13 09:56:34 2008: Breakpoint hit: Code @ key.c:760.5
Mon Oct 13 09:56:34 2008: The stack 'Stack' is filled to 100% (80 bytes used out of 80). The warning threshold is set to 90%.
Mon Oct 13 09:56:34 2008: The stack pointer for stack 'Stack' (currently Memory:0x9A4) is outside the stack range (Memory:0x9B0 to Memory:0xA00)
Mon Oct 13 09:56:39 2008: The stack 'Stack' is filled to 100% (80 bytes used out of 80). The warning threshold is set to 90%.
Mon Oct 13 09:56:39 2008: The stack pointer for stack 'Stack' (currently Memory:0x9A4) is outside the stack range (Memory:0x9B0 to Memory:0xA00)请给与解释？
```

A14:可能是堆栽溢出了，修改一下 **STACK** 试试看！

Q15：The stack 'Stack' is filled to 100% (80 bytes used out of 80). The warning threshold is set to 90%.，这个 **LED** 时间显示程序，是不是循环太多，还是程序太冗余，下载后出现了上述错误，该如何修改？

A15:假如真是 **STACK** 的容量问题，就说明你的程序结构有问题了。一般显示这些功能用不了几个字的堆栈。程序中尽量不要出现递归类的函数。函数的参数及返回值也不要太多，如果多了，可以用指针。

Q16：问题：编写 **UART** 的中断函数时用到语句 **interrupt [UARTORX_VECTOR] void UARTORX(void)**
可是在编译时总是报错 **Error[Pe077]: this declaration has no storage class or type specifier**
Error[Pe065]: expected a ";"
在函数的总前面我已**#include <msp430x14x.h>**
#include <in430.h>

在 msp430x14x.h 中已经对 UARTORX_VECTOR 进行审明了啊。
请问需要进行怎么修改？
A16:是 IAR 调试软件的问题，
在 IAR1.26 版和以下都是 interrupt [UARTORX_VECTOR] void UARTORX(void)的写法，以上都是#pragma vector=TIMERB0_VECTOR
__interrupt void Timer_B (void)的写法。

Q17：P1DIR=0x10;
P1SEL=0x10;//p1.4 为 SMCLK
P2DIR=0x01;
P2SEL=0x01;//P2.0 为 ACLK
P5DIR=0x10;
P5SEL=0x10;//p5.4 为 MCLK
在设置好了时钟后，是不是能通过上面的语句在 p1.4,p2.0,p5.4 口用示波器看到时钟信号呢？但为什么没有看到？而用示波器观察晶振是起振的。
A17：程序没有死循环，跑到最后就复位了。原来的程序上加了语句：while(1)；后当程序跑到这个空循环时，再用示波器测各个引脚 时，就有 aclk ,mclk ,sclk。

Q18：运用定时器 A 做定时，程序思想正确却无法执行，在 tt==20 的地方无法执行
该怎么办？

```
#include <msp430x14x.h>

#define uint unsigned int
#define uchar unsigned char
uchar num,tt;
uchar ta××e[]={
0xc0,0xfc,0x24,0x4f,
0x66,0x6d,0x7d,0x07,
0x7f,0x6f,0x77,0x7c,
0x39,0x5e,0x79,0x71};
void delay(uint z);
void main(void)
{
// WDTCTL = WDTPW + WDTHOLD; // Stop watchdog timer

WDTCTL = WDTPW + WDTHOLD; // Stop WDT
TACTL = TASSEL1 + TACLR; // SMCLK, QINGCHU TAR

CCTL0 = CCIE; // CCR0 interrupt ena××ed
CCR0 = 40000;
TACTL |= MC0;
//
P1DIR = 0;
P2DIR = 0;
//
```

```
P1SEL = 0;
P2SEL = 0;
//
P1DIR |=BIT0;
P1DIR |=BIT1;
P1DIR |=BIT2;
P1DIR |=BIT3;
P1DIR |=BIT4;
P1DIR |=BIT5;
P1DIR |=BIT6;
P1DIR |=BIT7;
//
P2DIR |=BIT0;
P2DIR |=BIT1;
P2DIR |=BIT2;
P2DIR |=BIT3;
P2DIR |=BIT4;
P2DIR |=BIT5;
P2DIR |=BIT6;
//
P1OUT=0x00;
while(1)
{
就在此处不往下执行 ？？？ if (tt == 20)
{
tt=0;
num++;
P1OUT=ta××e[num];
//dula=0;
delay(1000);
}
}
}
void delay(uint z)
{
uint x,y;
for(x=z;x>0;x--)
for(y=110;y>0;y--);
}
#pragma vector=TIMERA0_VECTOR
__interrupt void Timer_A (void)
//interrupt [TIMERA0_VECTOR] viod TImerA_ISR(viod)
{
CCR0 = 40000;
```

```
tt++; // Add Offset to CCR0
}
```

A18：1、你的定时器没有启动
TimerA 的初始化设置应该为：

```
void init_TimerA(void)
{
TACTL = MC_0+TACLR;//定时器暂停，定时器和输入分频器复位

CCTL0 = CCIE; // 禁止捕获模式+选择比较模式+比较模式中断允许+OUT 对应于输入状态
CCR0 = 64000;//
TACTL=TASSEL_2 +ID_0+ MC_1;// 时钟源 SMCLK,8M 不分频，增记数模式，禁止定时器溢出中断,***启动定时器***
}
```

中断服务程序

```
#pragma vector=TIMERA0_VECTOR//比较器/捕获器 0 中断服务程序
__interrupt void Timer_A (void)
{
TACounter++;
}
```

2、注意在初始化你的 TimerA 之前,应该根据你的实际电路设置时。
3、注意开总中断允许

```
_EINT();//这一句放在初始化完成后,while()语句前。
```

Q19：C 语言写程序时候如何让单片机复位？
A19：对 WDT 或者是 FLASH 的口令写错都会导致单片机复位！

Q20：

```
#include<msp430x14x.h>
void main(void)
{
WDTCTL=WDTPW+WDTHOLD;//关狗
P1DIR=0XFF;
P1OUT=0X00;
TACCR1=200;
TACCTL1=CCIE;
_EINT();//使能
TACTL|=TASSEL_2+ID_2;//smclk
TACTL|=MC_2;//连续
LPM0;

}
#pragma vector=TIMERA1_VECTOR
__interrupt void time(void)
{
if(TAIV==0X02)
{
```

```
P1OUT^=0X01;
}

}
```

这样仿真时,进入中断,taiv=0;这是为什么?按理说 taiv=0x02

A20：仿真的时因为 jtag 已经读过 TAIV 了，和你用程序是一样的效果。读过就会复位。

Q21：问题：用 430fg437 开发，片内 flash 大小为 32K，编译得到的 d43 文件有 33K 多，怎样优化程序尺寸的？编程序时怎样估算占用 flash 的大小？

A21：用 SIZE 优化时,选 MIDDLE 比 MAXIUM 出错少。只要 C 编程时变量分配时限定关键词用好,如用 volatile 限定变量,循环不会优化掉。

总之,对于程序大问题,先用 middle,后用 maxum,一定能省出几百或者 1K 以上的代码。还有,优化时将 inline 前的勾去掉,别选它.否则代码反而增大。优化后如果某一模块的代码反而增加了,可选中该模块后单独对它选其他优化选项。

Q22：Error[e16]: Segment CSTACK (size: 0x50 align: 0x1) is too long for segment definition. At least 0x4 more bytes needed. The pro××em occurred while processing the
segment placement command "-Z(DATA)CSTACK+_STACK_SIZE#", where at the moment of placement the ××aila××e memory ranges were "CODE:5B4-600"

请问：这个错误是什么意思?

A22：stack 设置太小，在 option->general option->stack/heap 设置 stack/heap 更大一点

Q23：调试 ADS1258，已经可以读寄存器了，SPI 通信正常，可是现在遇到的问题是通过 SPI 对 ADS1258 的寄存器写数，读出来的数总是发的命令字，而不是我发的数据。请高手指点一下。

程序如下：

```
P5OUT&=~BIT0;
IFG2 &= ~URXIFG1; // Clear flag
U1TXBUF = 0x65; // Send address
while (!(IFG2&URXIFG1)); // Wait for TX to finish
IFG2 &= ~URXIFG1; // Clear flag
U1TXBUF = 0x01; // Load data for TX after addr
while (!(IFG2&URXIFG1)); // Wait for end of addr TXi--);
IFG2 &= ~URXIFG1; // Clear flag
P5OUT|=BIT0
```

希望寄存器能读出我写的数 0x01 可是现在读出来的数总是 0x65,这是什么原因?

A23：这是 SPI 的特殊性,本次通讯输出到 SL××E 的数据,要到下一次 SPI 通讯时才能得到结果.因此,在你的程序最后,再加三行语句进行一次 SPI 传输,就能得正确的 ADS1258 数据。每传递一次数据发送一次时钟。

Q24：问题：高通滤波器的算法问题，要求实时性比较好，可是现在的方法之前有个系数是 1/256，这样子用移位实现的话岂不是都为 0 了？在滤波器程序中，系数的大小，对 430 的 16 位宽度来说，实在太小了。

A24：在滤波器程序中,系数的大小,对 430 的 16 位宽度来说,实在太小了。

解决办法是:对事先已算好的系数,扩大 16,32,64,128.....等倍后,作为 430 计算用的常数,430 计算出结果后,再缩小相应的倍数。这种方法的优点是:只进行移位,甚至是取高 8 位或低 8 位,不用除法或乘法运算,结果的精度好。

Q25：C51 中的 XBYTE 用 430 的 C 语言如何表达?

A25：430 和 51 不一样，没有固定的数据线和地址线，都要自己用 io 去模拟。

Q26：三角函数用 430 单片机计算,有没有好的方法,有没有库函数可以直接用来计算的?

A26:TI 提供过一个文件 . "FPP413.exe",里面的.S430 文件的很全面的说明,包括程序内容,调用方法等. 用 C 语言，直接调用 sin,cos,tan 即可，只是代码量可能会迅速增大，速度也会明显收到影响如果是产生正弦波一类的应用，建议使用查表方式

Q27：如何从 430 里读回 hex 文件?

A27：没有加密可以通过 jtag 可以读出，加密后就只能通过 bsl 读出了，bsl 要密码的。

Q28：设计了一个 15k 的程序，在运行过程中偶尔会出现复位现象，开始以为是看门狗动作了，随后初始化时就关闭了看门狗，但是这种情况还是会发生，但是用 USB 仿真机全程仿真，就没有问题，不知道什么原因。我的程序主要是实时测量，进行运算，数学计算的地方较多，程序中还有定时写 flash、uart 通信、两个外部中断处理、timeA timeB 定时器中断处理等，会不会是默认堆栈 80 过小，导致了复位（堆栈我一直没有更改过）？最初我还以为晶振的干扰，对晶振的外壳做了接地处理，结果还是没有解决。我用的是 msp430F147

A28：在程序中作循环 ，执行时间比较长 。若循环次数少，则改变的数组中个数少 ；若多加，两次循环改变的个数就多。

Q29：芯片为 F149，用 c 写程序，复杂的公式怎么实现，比如：

y=(a-0.000012)^b，其中 b=-0.14778989013*10^(-4)

Q29：指数运算是 pow，不过整数次方，比如 2、3、4，建议采用连乘，效率更高。

Q30：在用 MSP430 编程的过程中，想得到浮点数的整数和小数部分，不知道哪位有解决的办法?

A30：先将小数变成整数，在将其个位、十位、百位取出来！

Q31：子函数内调用了 sprintf 函数，定义了指针变量*s 来访问格式化后的一整型变量的 ASCII 码：

char *s；sprintf(s,"%-6u",a)；

但仿真后发现 s 指向了另一个全局变量，运行后全局变量的值就改变了，如何解决？

A31：指针 S 没初始化。虽然你定义了一个指针，但是这个指针的地址是多少你没定义，编译过程中应该有警告的。

Q32：怎样用 430 编一个开 4 次方的程序？

A32：
```
#include <msp430x14x.h>
#include "math.h"
float ii,jj;
void main(void)
{
WDTCTL = WDTPW + WDTHOLD;
jj=100;
ii = sqrt(sqrt(jj));
while(1);
}
```

Q33：IAR 内嵌汇编是用 asm("")语句，但我想请问如何在 c 和汇编直接传递函数，比如用寄存器做了一些运算，运算结果在 R12,R13 里，如何返回他们的值到 c 程序中呢？

A33：C 语言与汇编语言混合使用有 3 中方式：

1、内部函数；

2、直接嵌入；

3、调用汇编模块；

调用内部函数：

内部函数本身是由汇编语言实现的，所以调用内部函数本身就是在 C 语言中嵌入汇编语言，不过内部函数数量很少，只能实现特定的功能。

直接嵌入：

使用_asm 或 asm 扩展关键字，这种方法是用起来很简单，但是编译器只是简单的将汇编语句嵌入程序中，不考虑与前后语句是否匹配，因此有可能会造成不稳定，它有几条限制：

编译器编译时使用不同的优化级别会忽略嵌入的汇编语句，或者不进行优化；

一些汇编指令不能嵌入；

不能访问局部变量；

不能声明语句标号。

调用汇编模块：

调用汇编模块时需注意以下 3 点：编写汇编模块时，必须严格遵从调用规则；必须在汇编模块中把函数声明为 PU××IC；调用时，或者将汇编函数声明为 extern，或者为汇编模块编写头文件，以便编译器可以找到函数的位置。

Q34：在做键盘程序时，遇到一个问题：**4** 个按键 **K1---K4** 分别接到 **msp430** 的 **P1.0---P1.3**，采用定时器每 **5mS** 中断扫描方式读取键值，**K1** 要求按下弹起后执行相应的程序；**K2** 要求长按 **2** 秒以上后每隔 **0.5** 秒执行一次相应的操作。这些如何实现？

A34：这样处理的：

1、扫描到按键被按下后，在状态变量 Keystate 中设置所有被按下按键的状态标志“KEYDOWN”；备份 Keystate 到 Keystatemem；若所有按键的状态都是零，退出；若有按键被按下，则启动 TA1 两秒定时，并允许 TA 中断，退出。

2、下次扫描按键的时候，再次构建 Keystate；比较 Keystate 和 Keystatemem，判断哪些按键已经被放开；保持任然保持按下状态的所有按键的“KEYDOWN”标志并清除其“KEYUP”标志；清除已经被放开的且有“KEYDOWN”标志的按键的“KEYDOWN”标志，设置其“KEYUP”标志。如果所有按键的状态都是“KEYUP”或零，则终止 TA 记时，禁止 TA 中断，退出。

3、一旦 TA 定时时间到，在 TA 中断服务程序中判定所有具有“KEYDOWN”标志的按键为“长按”，清除其“KEYDOWN”标志并设置其“HOLDDOWN”标志；关闭 TA 计时器，并禁止 TA 中断，退出。

4、在主程序的键盘功能分析程序中，依照各按键的“KEYUP”、“HOLDDOWN”状态，分别进行相应的处理。

5、在所有循环处理的按键功能都被完成之后，清除以上所有标志和状态变量。

6、完毕。

Q35：用的是 **449** 的片子，要用定时器 **A** 产生 **200Hz** 的采样频率，定时器时钟才用 **SMCLK,**为 **1M**，下面是所有程序：

```
#include <msp430x44x.h>
void init_TimerA(void);
void init_ADC12(void);
char adc_Flag = 0;
int newValue;
void main(void)
{
```

```
int datasign1=0, data_x1[500];
WDTCTL = WDTHOLD + WDTPW; //停止看门狗
_DINT(); //关闭中断
init_TimerA();
init_ADC12(); //初始化 ADC12
_EINT(); //使能中断
while(1)
{
if(adc_Flag ==1)
{
adc_Flag = 0;
data_x1[datasign1]=newValue;
datasign1++;
}
}
}

void init_ADC12(void)
{
P6SEL |= 0x01; // Ena××e A/D channel A0
ADC12CTL0 &= ~(ENC); //设置 ENC 为 0，从而修改 ADC12 寄存器的值
ADC12CTL0 = ADC12ON+MSC; // Turn on ADC12, set sampling time
ADC12CTL1 = SHP+CONSEQ_2+ADC12SSEL_1; // Use sampling timer, set mode,ADC12 时钟源为 MCLK.
ADC12IE = 0x01; // Ena××e ADC12IFG.0
ADC12MCTL0|=INCH_0+SREF_0;
ADC12CTL0 |= ENC; // Ena××e conversions
}

void init_TimerA(void)
{
FLL_CTL0 |= XCAP14PF; // Configure load caps
TACTL = TASSEL1 + TACLR; // SMCLK, clear TAR
CCTL0 = CCIE; // CCR0 interrupt ena××ed
CCR0 = 5000;
TACTL |= MC0; // Start Timer_a in upmode
}

#pragma vector=TIMERA0_VECTOR
__interrupt void Timer_A (void)
{
ADC12CTL0 &=~ENC;
newValue = ADC12MEM0;
adc_Flag = 1; //指示有数据要显示
ADC12CTL0 |= ENC+ADC12SC;
}
```

问题是，程序进入到了中断程序后就跳不出来了，所以无法往下进行
请问各位问题会出在哪呢？

A35：把 ADCIE 这句屏蔽掉，应该可以执行下句，此程序中断产生生不是 TA 中断，并且 CCR0 中断是自动清除的，产生中断的是 ADC 中断；由于配置 ADC 是多次采样单通道，并且把 MSN 打开，那么第一次 J 进入中断后将 ADCSC 打开 ADC 开始转换，完毕后自动又开始采样转换，反复，也就是说，在下一个 TA 中断来前其实 ADC 是一直反复采样转换也就是反复进入中断了.根据你的意图，我建议你在 ADC 里面配置为单通道单次转换，并且触发源可以直接选择用 TA 不需要再 ADCSC 。

Q36：如何在 **IAR** 中建立 **C** 函数库文件？

A36：其实跟做普通 C 项目一样，没什么太多的差异.只不过在 option 中选择 Library 即可.
一样可以分 Group，在每个 Group 中可以包含多个文件。

Q37：

```
#pragma vector=PORT1_VECTOR
__interrupt void Port1()
{
delay();
//unsigned char q0=0;
if((KEYIFG&KEY0) ==KEY0)
{
KEYIFG &= ~KEY0; //清除中断标志
g+=1;
if ((KEYIFG&KEY1) ==KEY1)
{
//处理 P1IN.5 中断
//KEYIFG &= ~KEY0; //清除中断标志
KEYIFG &= ~KEY1; //清除中断标志
//KeyDown=KEY_P15; //记录按下的键值
FLAG_KEY_3_5=1;//标识 2 个键同时按下。 次数为 0 标识没有同时按下 2 个键。
Num_KEY_3_5 +=1;
KeyZhi=5; //记录按下的键值
//q0=1;
}
}
else if((KEYIFG&KEY1) ==KEY1)
{
//处理 P1IN.6 中断

KEYIFG &= ~KEY1; //清除中断标志
s +=1;
//KeyDown=KEY_P16; //记录按下的键值
//q0=1;
}
else if((KEYIFG&KEY2) ==KEY2)
{
//处理 P1IN.7 中断
```

```
KEYIFG &= ~KEY2;
//清除中断标志
z+=1;
//KeyZhi=KEY_P17; //记录按下的键值
//q0=1;
}
else
{
//其他干扰引起的中断，不进行处理，只清除中断标志
KEYIFG=0;
}
//if(q0==1)
//{
//GoKey(0); //关闭键盘中断
//KeyTime=0;
//KeyCnt=0;
//GotimeDfA(100); //打开定时器 A
//}
//}
void delay()
{
for(i=5000;i!=0;i--);
}
```

请问各位,二键同时按下有时不响应,单键按下有时变量加 2 或 3,不是加 1,是不是延时有问题

A37：最好不要在中断中有延时程序，这样会降低 MCU 的响应。如果需要延时去抖动什么的，最好采用定时器，这样既提高了 MCU 的响应速度，又会降低供耗问题。

Q38：可不可以把常量地址赋给指针变量,如下:

代码:

```
const int abc;
    int *p = &abc;
```

主要是想在 abc 的地址处通过擦写 Flash 写入东西。我在 IAR 中报了个错,不知道是不是这个问题。

A38：Flash 擦除肯定是段擦除的,但是在擦除的时候需要指定段地址.

代码:

```
#pragma location = "INFOA"
const int xxx = 0;
const int abc = 123;
int *p = &abc;
p++;
```

如上面程序,常量 abc 存放在信息段 A 中,用来保存设置参数.

当在应用程序中改变此参数值时,需要整段擦除信息段 A,然后在常量 abc 的地址处写入其它值.

现在的问题是如何把常量 abc 的地址放到指针变量里去?

像上面代码中做的话,会报一个错误,意思是 const int*型的值不能用来初始化 int*型的变量,可以看到,其中 p 是 int*型的.

将代码改成以下就不报错了

代码:

```
#pragma location = "INFOA"
const int xxx = 0;
const int abc = 123;
const int *p = &abc;
p++;
```

其实这个问题是"常量指针"和"指针常量"的问题。
我开始担心用 const 修饰 int* p 后会是 p 变为常量,而不能修改,就是 p++处会报错
后来查了下书,指针常量(指针本身是常量)定义的方法如下:
代码:

```
const int* const p = &abc;
```

上面第二个 const 才是是指针变为常量的因素。第一个 const 表示常量指针(指向常量的指针)
所以可以看出,第二段代码是正确的方法。
其实还有一种方法可以解决第一段代码中的报错问题
代码:

```
#pragma location = "INFOA"
const int xxx = 0;
const int abc = 123;
int *p = (int*) &abc;
p++;
```

如上,采用强制类型转换的方法也可以。

Q39：要把定义的常量放在代码区,该怎么定义啊!
A39：const
比如:

```
const uchar Number[]=
{
/*-- 文字: 0 --*/
/*-- 宋体 12; 此字体下对应的点阵为: 宽 x 高=8x16 --*/
0x00,0xE0,0x10,0x08,0x08,0x10,0xE0,0x00,0x00,0x0F,0x10,0x20,0x20,0x10,0x0F,0x00,//0
}
```

一般液晶字模就这样，放到程序存储区，避免占内存

Q40：１、在头文件 msp430x44x.h 中，宏定义#define TASSEL1 (0x0200)，后面的数字 0x0200 为干什么要加上括号（）；
２．汇编中，BIC #OscOff, SR 为什么要加上#，OscOff 应该理解为标号不是吗；
３．在头文件中#define BASICTIMER_VECTOR (0 * 2u) ,括号中的＂u＂代表什么？
A40： ()起保护作用
2, OscOff 因该有地方定义过
3,u 应该是无符号数

Q41：请问 IAR 中 CLIB 和 DLIB 区别?
问题：请问 IAR 中 CLIB 和 DLIB 区别?
A11：DLIB 是一个完整支持 C 标准的库文件，提供 IEEE754 浮点数计算库
CLIB 是一个相对精简的库文件，不完全支持 C 语言标准中队对库文件的要求，没有浮点库的支持，能够满足一般的需求。

Q42：这个提示是什么意思？

Library configuration file is not specified. Use --dlib_config, please see the compiler reference guide for details.

A42：没有指定库文件，这个提示已经告诉你使用 DLIB 库了，你就设置一下就可以了。

Q43：printf()函数是不是将需要打印的数据送到打印机啊？如果是他又是怎样让外围的硬件与打印机联系起来的呢？打印机的起始信号与应答信号他都能处理吗？

A43：单片机编程中，通常使用 printf(...)不是将数据写入打印机中，而是将数据进行格式化处理。但是更多的是使用 sprintf(....)，将格式化的数据写入函数指定的字符串中。因为 printf(...)函数在运行时需要调用 putchar(int value)函数，这是一个将字符写到“标准输出设备”的函数，若你想将数据写入特定的打印机，如微型打印机（在单片机中运行，而非 windows 中运行），那必须修改 putchar(....)函数的原程序 putchar.c 以满足在单片机中对该打印机的起始信号、应答信号、字符点阵转换等工作进行处理。不过一般微型打印机厂家都提供相应的驱动程序供用户使用，但对于特殊应用还是需要编程的。sprintf(....)的使用与 printf(...)完全相同，不同的是在 sprintf(....)中第一个形参是输出目标的字符串指针。而 printf(...)函数没有这个形参，而是使用 putchar(....)函数指定的标准输出设备作为输出目标。另外在运行时 sprintf(....)不调用 putchar(....)函数。

Q44：我用 **F2012** 的单片机

```
typedef struct{
U32 Band_Low; // Band low limit
U32 Band_High; // Band high limit
U32 Dem_Freq; // video IF
U32 Snd_Carr; // sound carrier frequency
U32 Hsync; // Hsync
}st_TV_Region;
#define GHz 1000000000
#define MHz 1000000
#define KHz 1000
#define Hz 1
```

程序中初始化的时候

```
const st_TV_Region c_Tv_Region[] = {
/* Low Limit , High Limit, Dem Freq , Snd Carrier, Hsync*/
/* PAL B/G */ {44*MHz, 862*MHz, 9*MHz, 5500*KHz, 15625*Hz},
/* PAL D/K */ {44*MHz, 862*MHz, 9*MHz, 6500*KHz, 15625*Hz},
/* NTSC-M */ {56*MHz, 806*MHz, 7*MHz, 4500*KHz, 15750*Hz},
};
```

5500*KHZ 6500*KHZ 4500*KHZ 的地方提示警告

Warning[Pe061]: integer operation result is out of range

如果 **5500 6500 4500** 后分别加上 **u**，就不会报错

如果直接写成 **5500000**，也不会出错。请问是怎么会使？

A44：在 IAR 中

#define 的值是根据后面的值来确定长度的

1000 它就以 int 型来表示了

5500*1000 肯定就超过 int 范围了

问题：关键我的数据类型是 uisigned long 类型啊

那请问如果我要保持个写法，应该怎么改。谢谢
如果 5500 后面加个 u，会对程序有影响么？
回复：没有其它方法了，只能如下：

```
const st_TV_Region c_Tv_Region[] = {
/* Low Limit , High Limit, Dem Freq , Snd Carrier, Hsync*/
/* PAL B/G */ {(U32)44*MHz, (U32)862*MHz, (U32)9*MHz, (U32)5500*KHz, (U32)15625*Hz},
/* PAL D/K */ {(U32)44*MHz, (U32)862*MHz, (U32)9*MHz, (U32)6500*KHz, (U32)15625*Hz},
/* NTSC-M */ {(U32)56*MHz, (U32)806*MHz, (U32)7*MHz, (U32)4500*KHz, (U32)15750*Hz}
};
```

Q45：问题：在 **430f149** 上运行 **uCOS-II** 操作系统刚开始的时候没什么问题，当需要做一个功能函数的时候编译没有错误但是在运行仿真的时候老是跳入 **DEBUG** 中提示：**the stack'stack'is filled to 100%.**
以为是堆栈益处调整堆栈大小以后编译出现错误。
The pro××em occurred while processing the segment placement command "-Z(DATA)CSTACK+_STACK_SIZE#", whereat the moment of placement the ××aila××e memory ranges were "CODE:29a-2ff" Reserved ranges relevant to this placement: 29a-2ff CSTACK
请问遇到过这个问题没有？有什么办法能解决呢？
A45：可能是局部变量过大造成的，也可能是设置 xcl 文件造成的，总觉得程序不可能占用 2k 的 RAM，看 CODE:29a-2ff 好象是你设置了 xcl 文件。

Q46:以前用的 **1612** 的 **DCO** 是由内部 **RC** 振荡器产生的频率.**4618** 的 **DCO** 频率是由低频振荡器的 **N+1** 倍决定的.不知道这样理解对不对.也就是说 **4618** 中,如果要使 **MCLK,SMCLK** 的频率来自 **DCO,**那就必须在 **XIN.XOUT** 处接表振.如果 **XIN.XOUT** 处不接.是不是 **DCO** 就没有频率输出。
A46：DCO 是内置振荡器，可以使用 FLL 模块对其进行标定，标定的基准就是外接的手表晶振，当没有外接手表晶振时，也可以使用 DCO 来做时钟。

Q47：如何在程序中定义位操作:如 **sbit pinINT = P3^2;**在 **msp430** 中能否实现这种操作,或有无可替找的方法**??**
A47：不能像上面这样定义，可用如下：

```
#define LED_ON P3OUT |= BIT2
#define LED_OFF P3OUT &=~ BIT2
```

程序中就直接使用 LED_ON LED_OFF 就和位操作一样了

Q48：有一段程序：

```
 const uchar HELLO[]="HELLO\n"; //因为这些字符是只读的，所以定义成
 const
 void fun1(uchar *str) //函数说明
 .....
main( )
 {......
 fun1(HELLO);
 .......
 }
 void fun1(uchar *str)
```

{.....
}
结果在编译时在调用 fun1 时出现 function argument imcompati××e with its declaration ,
将 const 换成 static 就 OK 了，函数就能正常调用。
是 const 类型的数组名不能作为参数传递吗，字符串数据是不是存在程序存储器中了，可用 const 定义时，我在程序中写如 str1=HELLO[3]，编译和运行也没问题阿，是不是与 MSP430 的寻址方法有关， 请大家赐教。

A48：用 const 类型的数组名是可以作为参数传递,但必须进行强制类型转换,如

```
const uchar HELLO[]="HELLO\n"; //因为这些字符是只读的，所以定义成
const
void fun1(uchar *str) //函数说明
.....
main( )
{.....
fun1((uchar *str)HELLO);
.....
}
void fun1(uchar *str)
{.....
}
```

Q49：问题：譬如在 C 程序中有如下一个子程序：

```
void Flash(void0
{
FlashBegin:
.................
.............
FlashEnd:
.................
}
```

在汇编程序中，想将 FlashBegin 与 FlashEnd 之间的那段程序拷贝到 RAM 中去。目前我是这样实现的：先将整个程序编译，找出该段程序所在存储空间的起始地址，然后在汇编程序中用立即数的方式直接表示这两个地址。这样做有个弊端，就是在程序进行修改之后，该段程序的起始地址会改变，因此，每次都需在汇编程序中修改这两个地址值，一不留神就会忘记，导致严重的错误。现在我想用 FlashBegin 与 FlashEnd 这两个标号来直接表示地址，请问在汇编程序中该如何引用？

A49：试试看！

```
；在汇编中加入：
;
extern called_by_asm;

rseg code
call #called_by_asm

end
/*******************
```

```
//C 中可以是：
void called_by_asm(void)
{
......//
}
```

Q50：想把一份表格存放在代码区，用的是 C 语言，不知道有什么方法？

A50：const unsigned char Ta××e[]={....}。

第四章：工作模式及功耗类

Q1：采用 MSP430F2012,用定时器 A 中断唤醒,定时器的时钟选择为 ACLK,外部接 32.768KHZ 晶振,LPM3 模式可以正常被定时器中断唤醒.因目前 IO 引脚不够用,想去掉外部的 32.768KHZ 晶振,当用利尔达的 USB JTAG 仿真器仿真时 LPM3 模式也能被定时器中断唤醒,但感觉每次唤醒的时间不一致,但是若不连仿真器,转为电源供电就完全不能被唤醒,不知为什么？关于 ACLK 时钟是否一定要接一个 32.768 的外部晶振?为了节省功耗必须用休眠模式,如果没有外部的 32.768KHZ 晶振是不是就唤不醒单片机?

A1：ACLK 时钟的来源可以是 VLOCLK 或者是 LFXT1CLK 所以不一定要接外部晶振 而在 LPM3 模式下 ACLK 是活动的，MCLK 和 SMCLK 被禁止所以不是没有外部的 32.768KHZ 就唤不醒单片机。需要你正确设置 ACLK 的来源。

Q2：1,MSP430 进入 LP 模式后,CPU 停止运行,那么,进入中断执行退出后,由于 SR 的恢复,导致还处于 LP 模式,是否意味着,CPU 在退出中断后立即停止了呢?

2,也就是说,进入 LP 模式后,要让非中断流程运行的话,只能在中断退出前把保存在堆栈里面的 SR 修改了？ 3,由于中断自动恢复保存的寄存器,要想在中断程序里面修改堆栈里面的保存的 SR,只能用汇编了？

A2：

1、是的。

2、是的。

3、是的。_BIS_SR_IRQ() 以及 _BIC_SR_IRQ() 函数可用。

Q3： 您好，我现在有个程序进入 **LPM3** 后拿万用表测电流在 **40uA** 左右，不是的 **datasheet** 里说的小于 **2uA**。万用表 **2mA** 档串接在电池和 **VCC** 之间。甚至于我写了最基本的初始化函数和 **main** 函数如下，测试电流值仍然在 **40** 多 **uA**。

我想知道如何才能准确测得 **LPM3** 下的电流，或者说是否万用表本身有影响？因为我们的程序要求有至少 **5** 年电池寿命，用的 **125mA** 的 **CR1632**，需要耗电在 **2.8uA** 下才行。

A3：我和你做的东西是差不多的，也用的是 **F201X** 或 **F21X1** 的片子，所以也测量过这个，对这个经验我有这几方面：

1、万用表的问题：有些万用表最小是 **2mA** 档，根据测量仪器的特性，接近满量程时的测量数据较准确，所以 **2mA** 档测量 **2uA** 的电流时，相差 **1000** 倍，极不精确。我测量时使用的是一块带有 **200uA** 档的 **DT830** 数字表，实测 **LPM3** 下只有 **0.7uA**，与 **F201X** 手册上标注一致。

2、湿度与 **PCB** 防潮问题：如果是实验板，因为上边没有阻焊膜，很容易受潮造成板上漏电，所以这种低功耗的东西一定要密封好，建议成品直接用树脂或是胶封起来，哪怕是热熔胶，也比暴露在空气中强。我测试的时候，就因为对着板子喘了口气，就发现表上电流示数开始变大了——这个电流实在是太小了，小到不能忽视任何干扰了。另外，要是手摸了板上子特别是电源两端的话，这个直接会造成几十到几百微安的电流，所以用手拿着板子测是极不科学的，要是不拿板子表笔不好扎的话，建议去电子商场买一对勾夹子，我买的 **1.5** 一个，**3** 块钱一对。勾好以后把板子悬空放着最好。

3、电路内的损耗电流：尤其是 **RC** 方式的 **SLOPE**，要想省电就必须严格控制电容的容量，越大越费电，或是说，如果你的电话容有余量的话，就要严格控制充电时间，否则充的时间长了，也一样会发生耗电增加的情况。最注意的就是，不要充上电之后去睡，醒了再放电检测，那样因为漏电的关系，损耗最大，哪怕你的电容容量很小也不行。

4、初始化的选择：初始化的时候，一定要把不用的东西全关掉，包括 **IO** 口全置为输出，并输出为低电平，而且不要选择 **REN**，或是像楼主的程序那样，**REN** 置 **0**。建议的初始化就是所有的 **SEL=0**，**REN=0**，**DIR=0FFH**，**OUT=0**。像楼主现在的初始化把 **P1SEL** 置了三位，那样就把那些模块一直选通了，可能会增加耗电。最好是模

块在使用前就不打开。当然如果为了利用模块的高阻性能而常开对应的比较器或 ADC 端口的话，建议置一下对应端口的 CAPD，这样可以进一步的减少寄生电流的产生。

Q4：问 CC2500 系列和 nRF2401 比较，功耗哪个更低？

A4：CC2500 和 nRF2401 的 datasheet,上很明显有 Power Down mode 模式下的一个功耗:900nA 和 1uA.而且功耗这块的算法,是看平均的功耗才为准..这就主要考虑到 WOR 模式.....

Q5：
```
#include <intrinsics.h>
#include <io430x14x.h>
int main()
{
// Stop watchdog timer to prevent time out reset
WDTCTL = WDTPW + WDTHOLD;
//initclock
init_clk();
init_io();
LPM0;
while(1);

}
```
Error[Pe020]: identifier "_BIS_SR" is undefined

以上程序为什么会出错？直接给 SR 赋值也不成，并且到头文件中也没找到 SR 的定义，这是出了什么问题呢？TA0 的时钟选用 0.5M 8 分频，近似计数值 63 为 1ms，使用连续计数方式。

A5：将头文件改成

```
#include<msp430x14x.h>
#include<in430.h>
#include <io430x14x.h>
```
即可正常运行。

Q6：芯片是 msp430F413,外围部件什么都没有接，只有一个 32768 的晶振，和芯片上电复位的几个电阻和电容，四个给段式液晶模块分压的 1M 电阻，再就是 JTAG 接口了，再没有任何东西了，没用的 IO 口都是悬空的，软件上都配成输出状态了，软件如下：

```
void Init(void)
{
P6DIR=0x00;
P6DIR |=BIT7; //
P6DIR |=BIT6; //
P6DIR |=BIT4; //
P6DIR |=BIT5; //
P6DIR |=BIT3; //
P6DIR |=BIT0; //
P6DIR |=BIT1; //
P6DIR |=BIT2; //
P3DIR |=BIT0; // 空闲引脚,置为输出状态
P3DIR |=BIT1; // 空闲引脚,置为输出状态
P2DIR |=BIT2; // 空闲引脚,置为输出状态
```

```
P2DIR |=BIT3; //  空闲引脚,置为输出状态
P2DIR |=BIT4; //  空闲引脚,置为输出状态
P2DIR |=BIT5; //  空闲引脚,置为输出状态
P2DIR |=BIT6; //  空闲引脚,置为输出状态
P2DIR |=BIT7; //  空闲引脚,置为输出状态
P2DIR |=BIT0; //  空闲引脚,置为输出状态
P2DIR |=BIT1; //  空闲引脚,置为输出状态
P6SEL = 0;
P1DIR |=BIT0;
P1DIR |=BIT1;
P1DIR |=BIT2;
}
void main(void)
{
WDTCTL=WDTPW+WDTHOLD; //禁止看门狗
Init(); //配置空闲管脚为输出状态
_EINT();
LPM4; //模式 4
_NOP();
_EINT(); //允许中断
delay1ms(1000);
while(1);
}
```

为啥低功耗电流怎么这么大，应该是 1uA 左右才对？

A6：1、CMOS 电路的电流消耗主要发生在 CMOS 管状态翻转的时刻。

2、设置成输入后，处于高阻状态的输入开关会发生未知的状态翻转，从而消耗电流。

3、设置成输出后，无论是上拉还是下拉，都会消耗电流。

4、建议普通 I/O 口设置成输出，并且悬空；或者设置成输入，并且上拉或下拉。

Q7：_BIS_SR(LPM3_bits);__bic_SR_register_on_exit(LPM3_bits),我看一些资料这两个函数都能进入到低功耗，但是第二个只能在中断程序中使用，第一个使用后程序还是在运行，说明一下两个的区别?

A7：前面是进入低功耗，后面的是退出低功耗。

Q8：低功耗设计中 430 接ＣＭＯＳ器件要注意 ？

A8：在低功耗产品设计中，当４３０的 i/o 口与 CMOS 器件接口时，比如ＬＣＤ之类的，为了省电，ＬＣＤ间歇供电，当ＬＣＤ关断时，与之相连的 430 的 i/o 口一定要设置为输入口或者设置为输出为１．否则 LCD 内部 cmos 器件的Ｉ／Ｏ口通过二极管导到其ＶＣＣ上，这样ＭＣＵ的电流变大了，功耗变的更大了.

Q9：程序总是不正确执行，运行就出现下述错误：

CPU is OFF (Low Power Mode) and interrupts are disa××ed! cannot excute Step/go

到底是怎么回事？看了 cpu 寄存器中 cpuoff =1 的，但没有设置低功耗模式呀？

A9：查下晶振是否正常；程序不能正常运行是一步都不能走么？如果只是某个位置，看看程序对外围控制是否会让时钟或者电源受到影响；再次确认是否真的没有设置低功耗，注意看低功耗的宏定义 。

Q10：当 CPU 关闭后，在中断函数里面的计算，比如全局变量 i 的累加等计算是由谁来执行的呢？为什么 CPU 关闭后还可以处理很多计算，赋值，判断等指令呢？

A10：在中断里 CPU 是处于 ACTIVE 模式的，进入中断后，DCO 会自动起震，430 退出低功耗模式。

Q11：这个设计,功耗如何才能降得很低?

用 F437 由于设计中有时钟功能,用 ACLK 产生所要的秒信号,所以系统不能断电,因此只能进入 LPM3 低功耗状态.而且有些数据要保存在 RAM 里面的，又这个仪器是要测信号的,因此要一个 8M 的晶振，还有段码 LCD。想问如何设计才能达到最低功耗要求呢?

现在的思路是：一个专门的按键,按下关(进入睡眠),再按一次开(唤醒)，但测得睡眠时的功耗有 3MA 左右.,这个外围是两个 MCP6002 运放,一个 232,一个红外芯片。是不是进入 LP3M 的时候,要把所有的外围模块都关了的?进入低功耗前把外部晶振关闭,是否会变成默认的内部晶振呢?这样能降不少功耗吧?定时器进入停止模式?LCD 关闭电压?AD 关闭?还有什么可以降低功耗的？

还要问下：做手持产品,外围的芯片是否都把它断电的呢?用 MOSFT 管吗?可否直接用单片机 IO 口来供芯片 VCC 呢?不需要时就把 IO 口改变就行了?这里工作时有 6MA,LP3M 时有 3MA,这样电池也用不了多久了.

A11：进入低功耗前把所有能关的电源全关掉。IO 口供电只能提供很小的电流，除非你供电很小，否则你还是采用 MOS 管或者其它开关进行控制。

Q12：很简单的一个问题。430 在低功耗的时候，闲置的口线应该是输入还是输出？？

A12：不用的 I/O 口设置为输入时接地，或者设置为输出时悬空。

第五章：复位系统类

Q1： 用的 **msp430f437**，用到了 **12** 路 **ad** 通道测量电压，利用 **flash** 信息段存储 **PT** 变比，**id** 号，电压计算系数等装置参数，精度校准后参数写入 **flash** 信息段保存，可以比较准的测量工频电压，但是现在我在这个基础上加入了温度补偿程序，即增加一个 **ad** 通道利用 **430** 的温度传感器测温，结果平时运行正常，但是在校准精度时（利用 **uart** 通信校准，**9600bps**）有时候就会出现复位情况，不是每次都这样。我用两个程序作比较，都含有 **flash** 写操作（校准后肯定要写入 **msp** 的 **flash** 信息段保存参数），区别在于：后面一个程序多了 **1** 个 **AD** 通道采集温度，于是也多了几个变量（**static** 变量，全局变量），这个两个程序就一个不复位，一个复位，我怀疑是我的堆栈什么的设小了，不知道有没道理。我用的 **IAR3.41**，编译后下面说：**xxxxcode**，**xxxxdata**，（**80absolute**），不明白什么意思？然后我 **jtag** 调试时，设置断点后在断点处暂停后下面指出：**the stack is filled to 100% (80 bytes used out of 80)**，**the warning threshhold is set to 90%**，这个有什么影响吗？

另外，用 **flash** 保存参数感觉会影响中断执行（毕竟 **flash** 写操作要关闭中断的），导致我的 **led** 不是等间隔定时轮询显示（我的 **led** 显示靠 **1ms** 中断来进行段选），所以会有明暗变化。请问有什么好方法改善吗？

A1：知道你问题所在了，你用的是默认 **80BITS** 堆栈，而你写 **FLASH** 时是不是有个大数组临时变量呀？你在选项里把堆栈设大点就没事了！**LED** 你可用静态显示！

Q2：应用背景：仪表做老化试验，需要周期性的上电工作。比如，加电源-〉工作 **10** 秒钟-〉断电-〉停止 **10** 秒钟-〉再上电，周期重复。单片机使用看门狗，看门狗周期 **1000ms**，保证正确清除看门狗。

现象：单片机上电后有时不能初始化，根本不能运行程序？

A2：**1**，请使用外部看门狗！**2** ，检查你的电路电源，是不是使用了大容量的电容，造成复位有问题。这 **2** 种方法都不错，有些电路掉电 **10** 秒钟后，单片机仍然处在有效工作电压，

Q3：请问 **msp430** 怎么手动复位啊**?**是不是连到 **RST/NMI** 上**?**但是这个脚不是和 **JTAG** 连吗**?**我看到一些资料上说复位的话还要上拉电阻或者复位电路。

A3：**JTAG** 功能只在下载程序时候使用，正常工作中 **RST** 可以连接一个按键，按下按键实现 **430** 手动复位。上拉电阻是上电复位用的，手工复位一个 **BUTTON** 就行了。**MSP430** 单片机低电平复位。

Q4：一个产品在做瞬时通断电试验时**,**发现有部分样机有死机现象，试验是这样的：对 **40** 台样机，通电几秒钟，再断电几秒钟，反复的上下电，断电的时间分别设为 **1S--10S,**发现有 **4** 台样机，在不同的断电时间，再上电后，不能启动，复位的看门狗（**706**），不断的有复位信号，但是样机不再工作了。除非断电将 **149** 的电源电压降为 **0**，再上电才能工作。发现死机的 **149** 的电压降到为 **0.X** 伏时，再上电，就死机了，可重复发生。本产品不带电池。该怎么办？

A4：由于外部电容的影响，几秒钟的断电，也许不可能让单片机完全停止工作。建议使用复位芯片。

Q5：上电复位和硬件看门狗复位有什么区别吗，在程序里将两者分开，请问有办法将两者分开吗？

A5：上电复位时，内存被清零或为任意值，看门狗清零时并没有断电，内存里的原有信息被保留，同时上电复位无法通过标志位来判别，看门狗复位才可以通过 **WDTIFG** 来判别。

同时注意 **RESET** 之后：

1、判断有无复位标志，若有，则为 **WDT** 复位；若无，则为上电复位，并且设置复位标志。手动按键复位也同此理。

2、保证复位标志在复位程序中不被清除。汇编好办，那是自己在控制 **RAM** 清除程序，**C** 呢，就要注意了。

3、要注意快速断电**/**上电的问题。处理不好的话，不但 **RAM** 中原先的内容有可能还存在，而且 **MCU** 复位很可能会失败

Q6：用 **FG4618** 做了一个工程,本片子有 **8K** 的 **RAM,**，定义的全局数组变量还不到 **4K,**发现编译完成后下载程序,调试界面不能复位**(IAR** 编译器**),**只要将全局数组变量减小定义,这个问题就没有了，这是为什么？

A6：**RAM** 并不只是全局变量占用，包括进入中断及局部变量都需要分配 **ram** 空间，这会导致在某些时段 **RAM** 空间不够。

Q7：用 **msp430f2012** 需不需要复位电阻和复位电容？因为我发现 **TI** 的 **EZ430** 开发工具上是没有复位电路的。

A7：查下晶振是否正常；程序不能正常运行是一步都不能走么？如果只是某个位置，看看程序对外围控制是否会让时钟或者电源受到影响；再次确认是否真的没有设置低功耗，注意看低功耗的宏定义 。

Q8：当 **CPU** 关闭后，在中断函数里面的计算，比如全局变量 **i** 的累加等计算是由谁来执行的呢？为什么 **CPU** 关闭后还可以处理很多计算，赋值，判断等指令呢？

A8：在中断里 **CPU** 是处于 **ACTIVE** 模式的，进入中断后，**DCO** 会自动起震，**430** 退出低功耗模式。

Q9：软件可不可以自使ＭＣＵ　ＰＯＲ（**RST)?**

A9：**PoR** 是指 **power on reset;**只能由 **reset** 脚或者 **svs** 硬件产生。**PUC** 是指 **power up clear,**多种情况可以生成。具体参考 **user's guide.**

Q10：用ＭＳＰ４３０Ｆ１４９做项目，但ＲＯＭ写满后换成２４１８的（１１６Ｋ）的片子，编译软件用的是ＩＡＲ　４.１１Ｂ版本的，写进去后，原来ＩＩＣ通信的都不行了，用示波器看ＩＩＣ的时钟和数据波形都有，从２４Ｃ５１２里就是读不出数据（在１４９里正常的）；还遇到另一个问题：因为我这产品有时间显示，用的是外部２Ｓ的信号输入，有时候４分钟自动复位一次（很有规律），有时候无规律的自动复位．再者个人认为２系列的没有１系列的稳定抗干扰能力强。请发表看法？

A10：首先纠正一个观点，**2** 系列的稳定性要好与 **1** 系列。再者，从 **149** 换到 **2418**，时钟上要做出一些调整，具体请参照用户指南和数据手册。很有规律的出现复位，建议用最小系统检测程序是否有误，或者外接电路的不正当操作。无规律的自动复位，请检查供电是否稳定，因 **2** 系列多了 **BOR** 模块，多了复位的触发源。

小提示：**430** 的 **I/O** 口单个供电能力最高可达到 **15mA**，但是整个芯片的总供电能力只有 **100mA** 左右。

Q11：用 **IAR** 进行 **C** 语言编程,请问如果定义全局变量并赋给初始值，在 **main** 函数中不再赋值，在上电复位后是怎么被赋的值啊？**430** 在上电后进入 **main** 函数之前都做了些什么操作啊？

A11：应该是按全局变量那样定义，你单步调试一下，看看该变量的值有没有变 。可以看 **IAR** 中编译后的汇编代码是从 **CSTART_END** 开始的,可以在 **CSTART** 中设置断点,观察到如果设置了全局变量并且赋了初值,则**IAR** 编译器会吧初值定义到 **FLASH** 空间中，在 **CSTART** 中在把 **FLASH** 中的初值复制到全局变量的 **RAM** 中的，所以对全局变量的赋值会占用 **FLASH** 空间，对于小容量的 **FLASH** 芯片，如 **2011** 等，要尽量的少用这种定义方法。

Q12：用 **msp430f149**，用的是 **IAR**，程序在用仿真器仿真的时候能正常运行，但是停止仿真，去掉仿真器，再重新上电的时候就不能正常运行了，请问这是怎么回事啊？

A12：**1**、你先确定，重新上电后单片机有没有复位运行。可以加一个 **LED** 做为标志，如果单片机跑起来，让这个 **LED** 不停的闪；如果没有，就检查复位电路。

2、停止仿真是会出现程序停止的现象，手动将 **RST** 与地短接复位可以恢复。但只有部分型号的芯片会有这个问题，比如 **149..**开始仿真也会出现程序烧不进的现象，大部分时候也可以通过手动复位解决；

3、最终检查出是复位问题。

Q13：运行到断点，程序就复位，不能停在断点处，跟程序量比较大有关系吗？请问是什么原因 ？

A13：1、因为 **430** 单片机看门狗复位时间默认为 **32ms**，如果你的程序在断点前的执行时间大于 **32ms**，当然不能运行到断点，还有如果你的程序自身编写有问题，比如通讯数据过多和 **AD** 采样时间太长，以及还有可能有死循环，都会导致程序断点不停。

2、如果看门狗是关的，那么你可以检查下，在程序运行到断点前是不是把某个中断打开了，在运行到这个断点前，该中断产生要进入中断；但是有可能你的中断向量是否写对，写错程序会跑飞，你可以看下 。

第六章：看门狗及定时器类

Q1. 定时器两个中断 TAIE 和 CCIE，有什么区别？两个中断的中断向量一样吗？

A1：TAIE 和 CCIE 指的是不同事件。

TAIE 指 TAR 计数器溢出，从 65535 到 0 的变化，由 TAIFG 引起的。

CCIE 指捕获到相应信号（捕获模式下）；定时时间到（比较模式下）。由 CCIFG 引起的。两个中断的中断向量不一样，TAIFG 一般进 TIMERA1_VECTOR;CCIFG 的话要看用的是哪个定时器如果是 CCR0 的话就进 TIMERA0_VECTOR，如果是 CCR1,CCR2……则进 TIMERA1_VECTOR。（中断向量的写法因器件不同而有所不同）

Q2：用的 msp430f1481,奇怪的是定时器 TA 能进中断,但中断标志一直存在,因此程序老在中断中跑，我用的是 8M 晶振,XT2ON。

A2：你最好说明是哪个标志位没有被清 0。没有设置相应的中断使能，是不会引起中断发生的；中断频率太快,在调试状态中,刚把中断标志清除,又有一个新的中断产生了,所以给人的错觉是清不掉

Q3. 芯片用的是 MSP430F447，从 P1.2 口输入外部脉冲，我用 TIMEB 定时，Port1 中断触发的，程序干不了别的，光中断了，程序好像也进不到定时中断里去。

A3：如上所述分析，可能是：

1、MCLK 可能过低，来不及处理指令

2、TB 的中断服务程序太长，所以光中断了，其实是中断服务程序运行时间太长了。

3、定时时间内的脉冲个数，如果脉冲频率很高，本来就是在不停的进入中断，如果频率高到一定程程度之后，都不用如不用中断，直接去判定标志位，因为进出中断也耗时间的。

Q4. 用 F149 的定时器 B 的捕捉功能，遇到问题,在等待捕捉时,读取 TBR 的值总是随机数。

A4：TBR 是一直再跑的，看 TBCCRx。因为捕获事件发生时，硬件会自动把 TBR 的值保存到 TBCCRx 中。

Q 5 ：想输出 PWM，在中断响应后能改变 PWM 的频率吗？

A5：调整相应的定时器配置，PWM 的频率和占空比都是可以改变的，不过周期频率一般都是由 CCR0 据顶的。

Q6： time A 定时器输出模块中 EQUx 和 EQU0 有什么区别？它们有什么用？

A6：捕获/比较器在比较模式时设置 EQUX 信号有差别：

当 TAR 的值大于或等于 CCR0 的中的数字时，EQU0=1

当 TAR 的值等于相应的 CCR1 或 CCR2 的值时，EQU1=1 或 EQU2=1

EQUx 和 EQU0 它们是用来控制输出单元的，软件中可以不用设置，由硬件自动触发

Q7. 关于 F2274 的 I/O 口中断的问题 ?希望通过 set 或者 clear P1IES 可以设置 P1 口在上升沿时触发中断还

是下降沿时中断，能不能搞成一个电平触发的？如高电平中断或低电平中断？

A7:：430 的 I/O 口中断都是沿触发的，没有 51 的电平触发模式。不过你可以在 I/O 口中断服务程序里查询电平状态，如果不是你期望的电平出现了，才清 I/O 口中断标志，让中断程序退出。

Q8. 在中断能使用全局变量？

问题：在用全局变量时，连接时提示错误：

Error[e46]: Undefined external "?cstart_init_copy" referred in main (C:\Documents and Settings\homex\桌面\comsp430\Debug\Obj\main.r43)

Error[e46]: Undefined external "?cstart_init_zero" referred in main (C:\Documents and Settings\homex\桌面\comsp430\Debug\Obj\main.r43)

Error[e46]: Undefined external "?ShiftRight32u_4" referred in main (C:\Documents and Settings\homex\桌面\comsp430\Debug\Obj\main.r43)

Error[e46]: Undefined external "?Mul32" referred in main (C:\Documents and Settings\homex\桌面\comsp430\Debug\Obj\main.r43)

Error[e46]: Undefined external "?DivMod32s" referred in main (C:\Documents and Settings\homex\桌面\comsp430\Debug\Obj\main.r43)

Error[e46]: Undefined external "?DivMod16s" referred in main (C:\Documents and Settings\homex\桌面\comsp430\Debug\Obj\main.r43)

A8：这种情况一般是多个 C 文件都申明了同一变量名造成的，把所用的变量在中断所在的文件中做外部变量说明即可。另外建议不要把项目文件夹放在桌面上。

Q9：timerA 不能进入中断检查的一般思路？

A9：不能进入中断一般检查思路:

1:是否开所属模块中断和总中断

2:所属模块所用时钟是否有效

3：触发条件成立没？标志位是否置位。

Q10. MSP430F149 关于时钟的问题

问题：时钟采用8MHz，那么执行for(i=0;i<980;i++);会花费多少ms?是不是1ms?另外，若是采用ADC12自带的时钟，在RC振荡器在5MHz，且不分频，执行一次采集（采样和转换）会花费多长时间？

Q10：执行for(i=0;i<980;i++);这个得去看汇编 看消耗了几个机器周期 这样是看不出来的。当然一般的做法是把程序运行在仿真状态下，在执行执行for(i=0;i<980;i++)之前记下此时的CYCLECOUNTER（VIEW-->register）该语句执行完毕之后记下这时的CYCLECOUNTER，2者相减，在乘以MCLK的周期。

若是采用ADC12自带的时钟，在RC振荡器在5MHz，且不分频：ADC12采样是12个ADC12CLK 外加一个ADC12CLK 用来把结果存到ADC12MEM。所以时间t=13*1/5M 秒。

Q11：430F149,中 TACTL 中 SCCI 是什么功能？

A11：Latched capture signal (read)处于捕获模式时，接入的信号状态 **0/1**。

Q12：怎么利用 F155 实现计数脉冲功能 ?将一个矩形脉冲序列送入单片机计算其脉冲数并根据脉冲数调节其增益。怎么实现呢？主要是计算脉冲数。用的高速晶振是 8M 的。我先设置 TA 的 CCR2＝10000。但是脉冲来之后每个高电平都要触发中断，还有有每次中断为什么都要 1ms 到 1s 啊。小于这个不行吗？

A12：1）用定时器做的话，也可以将定时器设置在捕获状态下，如上升沿捕获，当定时器捕获到上升沿时会产生一次中断，此时定时器会记录当前计数器的值到 CCRX，您可以把这个值放到指定的变量里，两次中断的记数差值就是你实际计数个数，这样你可以根据你计数个数调整增益，另外如果要计算时间的话只要将个数乘以定时器时钟就可以。

2）用捕获的方式就不用设置 **1mS~1S** 啦。你可以把 **8MHZ** 当 **TIMERA** 时钟，最小可以到 **1/8uS**，另外由于 **1MHZ** 捕获信号与 **8MHZ** 比较接近，如果采用两次捕获计算一个脉冲宽度精度不高，可以多采几次，如 **100** 次求得平均，这样精度会高很多。

Q13：MSP430F2101 和 MSP430F1101A 同时用了定时器 A 中断，它们的头文件以及各引脚使用有什么不同？各项是可兼容？

问题：产品（吸尘器控制器）本来用的是芯片 **MSP430F1101A**，在使用过程中发现因电压不稳定会导致其死机。于是打算换成具有 **BOR** 功能的 **MSP430F2101**。当更换了头文件后，把用于 **MSP430F1101A** 的代码烧录到 **MSP430F2101** 中，发现无法进入定时器 **A** 中断程序。我怀疑 **MSP430F2101** 和 **MSP430F1101A** 的定时器 **A** 有所不同，但经过阅读手册，发现并没有什么不同，使用了相同的晶振，**MSP430F2101** 和 **MSP4301101A** 的引脚是兼容的。那问题是什么呢？

A13：MSP430F2101 和 **MSP430F1101A** 的 **PIN5** 和 **PIN6** 引脚虽然是兼容的，但是不完全兼容：前者的引脚除了可外接晶振外，还可作为一般的 **I/O** 口和比较器输入端，而后者仅作为外接晶振。在 **MSP430F1101A** 的代码中使用了语句 **P1SEL=P2SEL=0x0;**把 **PIN5** 和 **PIN6** 的外围模块功能给禁止了，最终造成 **MCU** 的晶振无法正常工作。而 **TIMER_A** 又是依赖于晶振工作的。所以导致了无法进入 **TIMER_A** 的中断子程序。

Q14:timer 用 vlo 做 aclk 源如何进入 lpm3,定时 1 分钟，再退出 lpm3?1 分钟可以实现？

A14：硬件定时器结合软件计数器即可

Q16：程序执行完成后，进入 **LPM3** 模式，大约过 **3** 秒钟之后，又再进入正常模式，即从 **LPM3** 退出，再一次从头开始执行程序，执行完成后，再一次进入 **LPM3** 模式。如此往复循环下去。请问这个能不能通过定时器 **A** 来实现？

A16：能。

1、在主程序中完成初始化后进入 **LPM3** 模式。

2、在 **LPM3** 模式下，**ACLK** 必须工作，并且 **TA** 选择 **ACLK** 为时钟源。

3、当 **TA** 中断时，**CPU** 在进入 **TA** 中断服务前，会自动唤醒。

4、在 **TA** 中断服务中调用要被周期性执行的程序。

5、从 **TA** 中断服务返回后，**CPU** 又会自动恢复到中断发生前的 **LPM3** 模式（即，主程序会永远停留在进入 **LPM3** 模式的那条语句中程序）。

Q17：中断向量和中断标志寄存器 的区别？

A18：IICIFG 是中断标志，指示是否产生中断。IICIV 是中断向量，是中断代码的入口地址。MCU 首先查询是否有中断标志产生，如果有，再查询中断向量，转去执行中断服务程序。

Q18：单片机是 8M 的，脉冲是 1M 的。请问单片机不能及时响应吧？有什么好方法使它能及时响应？

A18：用高速信号计数:从 TACLK、或者 INCLK 脚输入，从 TAR 中取数据。读取 TAR，如：

```
int A;
A=TAR;
```

When the TACLK is asynchronous to the CPU clock, any read from TAR should occur while the timer is not operating or the results may be unpredicta××e. Alternatively, the timer may be read multiple times while operating, and a majority vote taken in software to determine the correct reading. Any write to TAR will take effect immediately.

Q19：BTCTL=BTDIV+BTIP1+BTIP0;为什么中断了 125 毫秒？

A21：32768/256/16 = 8hz=125ms。

Q20:捕获时，上升沿捕获，

第 1 次捕获的数据 TAR 放到 CCR0！ BH0 = CCR0;

第 2 次捕获的数据 TAR 放到 CCR0！ BH1 = CCR0;

一个脉冲周期 = BH1-BH0;

但是如果第一次捕获时 CCR0 很大，大到快要接近 0xffff；TAR 继续计数

TAR 溢出后！是什么个情况？ 是从 0 开始吗？

如果是从 0 开始，第 2 次捕获的数据如果小于第一次的数据！那该怎么办！？

A20：需要开捕获中断，记录溢出次数,当溢出次数只有一次时任然可以用 T=BH1－BH0，这时刚好是补码。当溢出次数超过一次，那待测脉冲周期 T=65536*（溢出次数-1）+ BH1－BH0。

Q21：设定 CCR0 作为 PWM 波形的周期、

设定 CCR1 作为 PWM 波形的占空比

改变 CCR1 来改变占空比

我的问题是在什么情况下可以改变 CCR1;

任意时刻都可以吗？

还是想 TAR 那样！必须在停止模式下才可以改变其值！？？

如果是任意时刻!

那么当 CCR1 改变后!

如果改变后的 CCR1> 改变前的 CCR1! 波形是怎么个情况??

如果改变后的 CCR1< 改变前的 CCR1! 波形是怎么个情况? ?

A21: CCR1 被改变时,新的 CCR1 值与老的 CCR1 的值大小比较,会影响改变时刻后第一个波形的占空比,以后的波形就是你所期望的了.请读 TI 的 slau056f(msp430fx4xx family user's guide)的 TIMER_A 部分

Q22: 用的是 MSP430F169 单片机!

但是我有点不清楚! 那个是 SMCLK 那个是 ACLK 了!

看一下我的初始化!

```
void init_tima（void）
{
TACTL = TASSEL_1+TACLR; //清除数据 ， 设定波特率
CCTL0 = CCIE; //定时中断
CCR0 = 163; //周期 10ms 32768/163/2=100.5HZ =10ms
TACTL |= MC0; //增计数模式
}
```

我用得是 32768 和 4M 两种晶振! 在我的记忆中 ACLK = 32768HZ SMCLK = 4MHZ(不知道对不对)

TASSEL_1 TASSEL_0

0 0 TACLK (TACL)

0 1 ACLK

1 0 SMCLK

1 1 INCLK(外部输入时钟)

问题是:

TACTL = TASSEL_1+TACLR; 我这个初始化后! 我发现感觉我的时钟是 32768HZ 呀而不是 4M ! 这是为什么?

如果设置真是 32768HZ ,那么 4M 的怎么设置呀!! ? ? ?

A22: 若要使用 4M 为时钟源,可设 BCSCTL2 = SELS,选择 SMCLK 为 XT2CLK ,设 TACTL = TASSEL_1,选择 Timer_A clock source 为 SMCLK。

Q23: 用 430 的 timer_A 产生 PWM 波来驱动步进电机,我将 430 的 PWM 波输出管脚直接连接到步进电机驱动器,现在发现不是很稳定。我想通过一个芯片提高 430 输出的 PWM 波的驱动能力, 我应该用什么芯片?

A23：步进电机驱动器有用 PWM 波驱动应该是用频率脉冲驱动。

Q24．关于 TIMERA 计数问题

用 TIMERA 对外来脉冲计数，脉冲输入 TACLK，读到的值总是不准，不知道各位有什么高招？TIMERA 益出有什么好办法可以连续计数？

A24：

1、应该是漏计了，输入频率不是很高，最高也就是 20KHZ。

2、TIMERA 我用的是连续计数到 0XFFFF 后中断的方式，在溢出中断里累计中断次数。然后应用程序里面用溢出次数*65535+TAR 值，就是连续计数值。

Q25：用 430F413 定时器 A 对外部引脚（48 脚，P1.5/TACLK/ACLK） 的 2M 脉冲计数，计数始终为 0，而对内部 ACLK 或 MCLK 计数则正常，不知如何解决？

A25：P1.5 脚要在 P1SEL 中设置为模块功能 P1SEL |= BIT5 ;

Q26：知道定时器可输出自定义占空比的PWM波，找了些参考程序看，大概是这样的

```
TACTL=TASSEL0+TACLR+MC0;
CCTL0=CCIE;
CCR0=360;
CCR1=327;
CCTL1=OUTMOD_2;
P2SEL |= BIT5;
```

这样是不是直接从P2.5端口输出PWM波，芯片不能工作在低功耗下啊？怎么样让芯片工作在低功耗，等有波形翻转时产生中断啊？

A26：OUTMOD_7 我推荐 那样比较好计算指定占空比时 CCR1 与 CCR0 的值，这种情况下芯片可以工作在低功耗模式下 只要你保证在该模式中你的 TA 时钟源仍然开启就行，至于波形翻转这些都是硬件完成，无需 CPU 干预。

Q27：如何实现间隔 1 小时触发定时器中断进行采集，对定时器如何设置呢？如果我用 32K 的时钟源，最多是不是只能实现 2 秒触发一次？

A27：最多 2s 呢，设置一个计数器，计数 1800 次。

Q28：想用 TA 的 CCR0 定时中断，增记数模式，但我看书上的 CCIFG0 的设置只有在 CCR0 与 TAR 的值相等时才置位，下一个周期马上又复位了。要是此时正在运行一条指令，那岂不是错过了进入中断了吗?

A28：CCIFG0 应该是响应了中断服务程序后才复位的或者用指令清除,不会错过的.。

Q29：MSP430 中断所需的最小输入脉宽是多少？

A29:最小中断脉宽必须大于 1.5 主时钟周期 (MCLK)，以确保中断有效。有关问题请参阅器件特定的数据表。

Q30：430 在低功耗模式下， 是哪几个模式（LPM0,LPM1,LPM2,LPM3,LPM4) 下，还支持看门狗。LPM3，LPM4 支持看门狗吗。

A30：在只有在 LPM4 模式下，看门狗才关掉，其他模式下只要选择了看门狗模式，时钟会一直跑。

Q31：在某种模式下把看门狗使能，那么功耗将增加多少？

A31：这个定时器模块不会引起额外的功耗(lpm3 模式下，不论是否开启看门狗，系统耗电会低于 1uA)。

Q32：不管什么模式，看门狗最长间隔时间能达到多少？

A32：因你设置的看门狗的时钟和寄存器配置而定。只要你提供给 WDT 的时钟源没关断，WDT 就一直工作。参看头文件:

```
#define WDT_MDLY_32 (WDTPW+WDTTMSEL+WDTCNTCL) /* 32ms interval (default) */
#define WDT_MDLY_8 (WDTPW+WDTTMSEL+WDTCNTCL+WDTIS0) /* 8ms " */
#define WDT_MDLY_0_5 (WDTPW+WDTTMSEL+WDTCNTCL+WDTIS1) /* 0.5ms " */
#define WDT_MDLY_0_064 (WDTPW+WDTTMSEL+WDTCNTCL+WDTIS1+WDTIS0) /* 0.064ms " */
/* WDT is clocked by fACLK (assumed 32KHz) */
#define WDT_ADLY_1000 (WDTPW+WDTTMSEL+WDTCNTCL+WDTSSEL) /* 1000ms " */
#define WDT_ADLY_250 (WDTPW+WDTTMSEL+WDTCNTCL+WDTSSEL+WDTIS0) /* 250ms " */
#define WDT_ADLY_16 (WDTPW+WDTTMSEL+WDTCNTCL+WDTSSEL+WDTIS1) /* 16ms " */
#define WDT_ADLY_1_9 (WDTPW+WDTTMSEL+WDTCNTCL+WDTSSEL+WDTIS1+WDTIS0) /* 1.9ms " */
/* Watchdog mode -> reset after expired time */
/* WDT is clocked by f**CLK (assumed 1MHz) */
#define WDT_MRST_32 (WDTPW+WDTCNTCL) /* 32ms interval (default) */
#define WDT_MRST_8 (WDTPW+WDTCNTCL+WDTIS0) /* 8ms " */
#define WDT_MRST_0_5 (WDTPW+WDTCNTCL+WDTIS1) /* 0.5ms " */
#define WDT_MRST_0_064 (WDTPW+WDTCNTCL+WDTIS1+WDTIS0) /* 0.064ms " */
/* WDT is clocked by fACLK (assumed 32KHz) */
#define WDT_ARST_1000 (WDTPW+WDTCNTCL+WDTSSEL) /* 1000ms " */
#define WDT_ARST_250 (WDTPW+WDTCNTCL+WDTSSEL+WDTIS0) /* 250ms " */
#define WDT_ARST_16 (WDTPW+WDTCNTCL+WDTSSEL+WDTIS1) /* 16ms " */
#define WDT_ARST_1_9 (WDTPW+WDTCNTCL+WDTSSEL+WDTIS1+WDTIS0) /* 1.9ms " */
```

Q33：系统复位后,如何判断是看门狗引起的复位呢,还是上电引起的复位呢?能不能通过看门狗中断标记来判断？是不是看门狗做定时器时才能使能看门狗中断控制位？

A33：看门狗引起复位会在标志寄存器中的 WDTIFG 为 1,而上电复位为 0。

Q34：我用的是 F169 的片子，看门狗时钟源用的是辅助时钟源（ACLK），XT1 接的是 32.768 的晶振，初始化看门狗程序：WDTCTL |= WDTPW+WDTCNTCL+WDTSSEL+WDTIS0;也就是 250ms，主程序用的是 DCO，大概频率是 2.5Mhz。每次不打开看门狗工作正常，一旦打开看门狗了，程序初始化看门狗以后就不知道跑哪去了，请问是什么原因啊？

A34：WDTCTL |= WDTPW+WDTCNTCL+WDTSSEL+WDTIS0;写法并不能达到你预期的结果。因为 POR 后，WDTCTL 不等于 0。可以采用 WDTCTL=XXXXX 的写法，直接赋值。

Q35：语句 *WDTCTL = WDTPW + WDTHOLD;* *// 停止看门狗*

WDTCTL 是 16 位的寄存器，可是 WDTPW + WDTHOLD 是什么意思啊？
WDTCTL&＝0X0010 不也是停止(watchdog timer is stopped)吗？有什么区别？
A35：WDTCTL = WDTPW + WDTHOLD，是关看门狗,WDTPW 是密钥,对 WDT 或 flash 寄存器访问是要密钥的,WDTPW 就是写入密钥,否则产生复位。WDTCTL&＝0X0010，没有写密钥，WDTCTL 根本不能设置成 0X0010，而是系统直接复位了。

Q36：一般使用如下指令进行看门狗停止：WDTCTL = WDTPW + WDTHOLD; //关闭看门狗。
那么看门狗打开是否可以采用指令：WDTCTL &= ~(WDTPW + WDTHOLD); //打开看门狗 ？
A36：需要理解 WDTPW、WDTHOLD 的具体意思。这些都在头文件中的宏定义。WDTPW 为寄存器访问密钥，如果写入错误的密钥，会产生复位；WDTHOLD 为看门狗时钟关闭；寄存器 WDTTMSEL 选择看门狗是工作在定时器模式还是在看门狗模式，所以要打开看门狗应该 WDTCTL = WDTPW + WDTTMSEL;具体定时时间可以看头文件，头文件中都有注释。

Q37：在程序中间采用如下指令：WDTCTL = WDTPW + WDTCNTCL;//清开门狗，防止在程序正常运行中复位。是否正确？
A37：错误，程序一样会复位。WDTCNTL 为清除计数器，WDTHOLD 为关闭计数器，两个寄存器的概念不一样，使用时需要注意。

Q38：看门狗的时钟设置为：BCSCTL2 = SELM_2 + SELS + DIVS_3; //选择 MCLK、SMLK 为 XT2 分频因子 8
以上为我程序中的设置情况，可是感觉看门狗不能正常工作，不知道是何原因？
A38：相应的 .h 头文件中有相应定义，时间长度、时钟源都定好了：

```
#define WDT_MRST_32 (WDTPW+WDTCNTCL) /* 32ms interval (default) */
#define WDT_MRST_8 (WDTPW+WDTCNTCL+WDTIS0) /* 8ms " */
#define WDT_MRST_0_5 (WDTPW+WDTCNTCL+WDTIS1) /* 0.5ms " */
#define WDT_MRST_0_064 (WDTPW+WDTCNTCL+WDTIS1+WDTIS0) /* 0.064ms " */
/* WDT is clocked by fACLK (assumed 32KHz) */
#define WDT_ARST_1000 (WDTPW+WDTCNTCL+WDTSSEL) /* 1000ms " */
#define WDT_ARST_250 (WDTPW+WDTCNTCL+WDTSSEL+WDTIS0) /* 250ms " */
#define WDT_ARST_16 (WDTPW+WDTCNTCL+WDTSSEL+WDTIS1) /* 16ms " */
#define WDT_ARST_1_9 (WDTPW+WDTCNTCL+WDTSSEL+WDTIS1+WDTIS0) /* 1.9ms "
```

Q39：定时器有两个中断 TAIE 和 CCIE，有什么区别？两个中断的中断向量一样吗？
A39：TAIE 和 CCIE 指的是不同事件。
TAIE 指 TAR 计数器溢出，从 65535 到 0 的变化。产生的是溢出中断 TAIFG。
CCIE 指捕获到相应信号（捕获模式下）；定时时间到（比较模式下）。产生 CCIFG 中断。
两个中断的中断向量不一样。

Q40：我用的 msp430f1481,奇怪的是定时器 TA 能进中断,但中断标志一直存在,因此程序老在中断中跑，我用的是 8M 晶振,XT2ON？
A40：没有设置相应的中断使能，是不会引起中断发生的；中断频率太快,在调试状态中,刚把中断标志清除,又有一个新的中断产生了,所以给人的错觉是清不掉

Q41：芯片用的是 MSP430F447，从 P1.2 口输入外部脉冲，我用 TIMEB 定时，Port1 中断触发的，程序干不了别的，光中断了，程序好像也进不到定时中断里去。
A41：如上所述分析，可能是：

1、MCLK 可能过低，来不及处理指令

2、TB 的中断服务程序太长，所以光中断了，其实是中断运行时间太长了。

3、定时时间内的脉冲个数，如果脉冲频率很高，本来就是在不停的进入中断，如果频率高到一定程程度之后，都不用如不用中断，直接去判定标志位，因为进出中断也耗时间的。

Q42：用 F149 的定时器 B 的捕捉功能，遇到问题,在等待捕捉时,读取 TBR 的值总是随机数,我用软件在线调试观察的。

A42：别读 TBR。读 TBCCRx 就 OK 了，捕获到信号时 TBR 的值自动复制到 TBCCRX 的值。

Q43：我想输出 PWM，在中断响应后能改变 PWM 的频率吗？

A43：调整相应的定时器配置，PWM 的频率和占空比都是可以改变的。CCR0 的值改变周期，CCR1（和输出对应的寄存器）改变占空比。

Q44：time A 定时器输出模块中 EQUx 和 EQU0 有什么区别？它们有什么用？

A44：捕获/比较器在比较模式时设置 EQUX 信号有差别：

当 TAR 的值大于或等于 CCR0 的中的数字时，EQU0=1

当 TAR 的值等于相应的 CCR1 或 CCR2 的值时，EQU1=1 或 EQU2=1

EQUx 和 EQU0 它们是用来控制输出单元的，软件中可以不用设置，由硬件自动触发。综上所述，EQUX 可以理解为一个信号，是为了描述方便加上的一个名字。

Q45：通过 set 或者 clear P1IES 可以设置 P1 口在上升沿时触发中断还是下降沿时中断，能不能搞成一个电平触发的？如高电平中断或低电平中断？

A45：不可以的。一定要是一个跳沿，并且这个跳沿的时间最小要大于 20ns。

Q46：timerA 不能进入中断检查的一般思路？

A46：不能进入中断一般检查思路:

1：是否开所属模块中断和总中断。

2：所属模块所用时钟是否有效。

3：中断向量是不是写正确。

Q47：时钟采用 8MHz，那么执行 for(i=0;i<980;i++);会花费多少 ms?是不是 1ms?另外，若是采用 ADC12 自带的时钟，在 RC 振荡器在 5MHz，且不分频，执行一次采集（采样和转换）会花费多长时间？

A47：执行 for(i=0;i<980;i++);这个得去看汇编，看消耗了几个机器周期这样是看不出来的。若是采用 ADC12 自带的时钟,在 RC 振荡器在 5MHz,且不分频:ADC12 的采样时间根据具体的寄存器设定,转换是 12 个 ADC12CLK 外加一个 ADC12CLK 用来把结果存到 ADC12MEM 是 13 个 CLK。

Q48：430F149,中 TACTL 中 SCCI 是什么功能？

A48：SCCI 寄存器为当前捕获到的状态，带锁存功能。

Q49：将一个矩形脉冲序列送入单片机计算其脉冲数并根据脉冲数调节其增益。怎么实现呢？主要是计算脉冲数。用的高速晶振是 8M 的。我先设置 TA 的 CCR2＝10000。但是脉冲来之后每个高电平都要触发中断，还有有每次中断为什么都要 1ms 到 1s 啊。小于这个不行吗？

A49：1、用定时器做的话，也可以将定时器设置在捕获状态下，如上升沿捕获，当定时器捕获到上升沿时会产生一次中断，此时定时器会记录当前计数器的值到 CCRX，您可以把这个值放到指定的变量里，两次中断的

记数差值就是你实际计数个数，这样你可以根据你计数个数调整增益，另外如果要计算时间的话只要将个数乘以定时器时钟就可以。
用捕获的方式就不用设置 1mS~1S 啦。你可以把 8MHZ 当 TIMERA 时钟，最小可以到 1/8uS，另外由于 1MHZ 捕获信号与 8MHZ 比较接近，如果采用两次捕获计算一个脉冲宽度精度不高，可以多采几次，如 100 次求得平均，这样精度会高很多。

Q50：产品（吸尘器控制器）本来用的是芯片 MSP430F1101A，在使用过程中发现因电压不稳定会导致其死机。于是打算换成具有 BOR 功能的 MSP430F2101。当更换了头文件后，把用于 MSP430F1101A 的代码烧录到 MSP430F2101 中，发现无法进入定时器 A 中断程序。我怀疑 MSP430F2101 和 MSP430F1101A 的定时器 A 有所不同，但经过阅读手册，发现并没有什么不同，使用了相同的晶振，MSP430F2101 和 MSP4301101A 的引脚是兼容的。那问题是什么呢？
A50：MSP430F2101 和 MSP430F1101A 的 PIN5 和 PIN6 引脚虽然是兼容的，但是不完全兼容：前者的引脚除了可外接晶振外，还可作为一般的 I/O 口和比较器输入端，而后者仅作为外接晶振。在 MSP430F1101A 的代码中使用了语句 P1SEL=P2SEL=0x0;把 PIN5 和 PIN6 的外围模块功能给禁止了，最终造成 MCU 的晶振无法正常工作。而 TIMER_A 又是依赖于晶振工作的。所以导致了无法进入 TIMER_A 的中断子程序。

Q51：timer 用 vlo 做 aclk 源如何进入 lpm3,定时 1 分钟，再退出 lpm3?1 分钟可以实现嘛？
A51：硬件定时器结合软件计数器即可。只要时钟在就可以实现 1 分钟的定时。

Q52：使用 MSP430 产生 PWM 波形，一般抖动问题如何解决？
A52：提高时钟精度。

Q53：程序执行完成后，进入 LPM3 模式，大约过 3 秒钟之后，又再进入正常模式，即从 LPM3 退出，再一次从头开始执行程序，执行完成后，再一次进入 LPM3 模式。如此往复循环下去。请问这个能不能通过定时器 A 来实现？
A53：能。
1、在主程序中完成初始化后进入 LPM3 模式。
2、在 LPM3 模式下，ACLK 必须工作，并且 TA 选择 ACLK 为时钟源。
3、当 TA 中断时，CPU 在进入 TA 中断服务前，会自动唤醒。
4、在 TA 中断服务中调用要被周期性执行的程序。
5、从 TA 中断服务返回后，CPU 又会自动恢复到中断发生前的 LPM3 模式（即，主程序会永远停留在进入 LPM3 模式的那条语句中程序）。

Q54：使用定时器 A 定时每 20 分钟主动发送一次短信，采用 CCR1 定时，在中断处理程序中发送短信，发送 AT 命令，然后在串口中断接受回应数据，同时要是定时中断接受数据超时判断，超时判断用 CCR0,1ms 中断一次也就是说在 CCR1 定时中断处理程序中还要进行串口接受中断和 CCR0 定时中断，在 CCR1 中断处理过程中可以响应串口接受中断和 CCR0 中断吗？如果不行，有什么办法可以替代呢？
A54：可以中断嵌套，在中断服务子程序中打开总中断使能就可以。

Q55：中断向量和中断标志寄存器的区别？
A55：IFG 是中断标志，指示是否产生中断。IV 是中断向量，是中断代码的入口地址。MCU 首先查询是否有中断标志产生，如果有，再查询中断向量，转去执行中断服务程序。

Q56：单片机是 8M 的，脉冲是 1M 的。请问单片机不能及时响应吧？有什么好方法使它能及时响应？
A56：MCU 主时钟和 TA 时钟选择 8M 时钟源，捕获 1M 脉冲，从 TAR 中取数据。

Q57：问题：因为项目需要，用 MSP430FE42X，设计多路电力参数变送器。各方面来说，FE42X 都比较满意，但是它只能够输出两路 PWM 信号用于 DAC，差一路。在不使用外部 DAC 以及电位器调整外，有什么稍微简单一点的办法用来生成第三路 DAC 吗？

A57：FE427 内部有 3 个 timer ,可以用内部的 timer 的比较捕获功能来做。

Q58：BTCTL=BTDIV+BTIP1+BTIP0;为什么中断了 125 毫秒？

A58：参考寄存器配置说明可以知道 32768/256/16 = 8hz=125ms。

Q59：捕获时，上升沿捕获，
第 1 次捕获的数据 TAR 放到 CCR0！ BH0 = CCR0;
第 2 次捕获的数据 TAR 放到 CCR0！ BH1 = CCR0;
一个脉冲周期 = BH1-BH0;
但是如果第一次捕获时 CCR0 很大，大到快要接近 0xffff；TAR 继续计数
TAR 溢出后！是什么个情况？ 是从 0 开始吗？
如果是从 0 开始，第 2 次捕获的数据如果小于第一次的数据！那该怎么办！

A59：在捕获时需要打开定时器溢出中断，记录溢出次数。

Q60：设定 CCR0 作为 PWM 波形的周期、
设定 CCR1 作为 PWM 波形的占空比
改变 CCR1 来改变占空比
我的问题是在什么情况下可以改变 CCR1；
任意时刻都可以吗？
还是想 TAR 那样！必须在停止模式下才可以改变其值！？？
如果是任意时刻！
那么当 CCR1 改变后！
如果改变后的 CCR1> 改变前的 CCR1！波形是怎么个情况??
如果改变后的 CCR1< 改变前的 CCR1！波形是怎么个情况？？

A60：CCR1 被改变时,新的 CCR1 值与老的 CCR1 的值大小比较,会影响改变时刻后第一个波形的占空比,以后的波形就是你所期望的了。

Q61：用的是 MSP430F169 单片机！
但是我有点闹不清楚！那个是 SMCLK 那个是 ACLK 了！看一下我的初始化！

```
void init_tima（void）
{
TACTL = TASSEL_1+TACLR; //清除数据 ， 设定波特率
CCTL0 = CCIE; //定时中断
CCR0 = 163; //周期 10ms 32768/163/2=100.5HZ =10ms
TACTL |= MC0; //增计数模式
}
```

我用得是 32768 和 4M 两种晶振！在我的记忆中 ACLK = 32768HZ SMCLK = 4MHZ(不知道对不对)
TASSEL_1 TASSEL_0
0 0 TACLK (TACL)
0 1 ACLK
1 0 MCLK

1 1 INCLK(外部输入时钟)
问题是：
TACTL = TASSEL_1+TACLR； 我这个初始化后！我发现感觉我的时钟是 32768HZ 呀而不是 4M ！这是为什么？
如果设置真是 32768HZ ,那么 4M 的怎么设置呀！！？？？
A61:TASSEL_1 时钟选择是 ACLK 所以始终是 32768HZ。若要使用 4M 为时钟源，可设 BCSCTL2 = SELS,选择 SMCLK 为 XT2CLK ，设 TACTL = TASSEL_2，选择 Timer_A clock source 为 SMCLK。

Q62：用 430 的 timer_A 产生 PWM 波来驱动步进电机，我将 430 的 PWM 波输出管脚直接连接到步进电机驱动器，现在发现不是很稳定。我想通过一个芯片提高 430 输出的 PWM 波的驱动能力，我应该用什么芯片？
A62：430 估计没有那么大的驱动电流，最好选用专门的步进电机驱动芯片。

Q63：用 TIMERA 对外来脉冲计数，脉冲输入 TACLK，读到的值总是不准，不知道各位有什么高招？TIMERA 益出有什么好办法可以连续计数？
A63：TIMERA 我用的是连续计数到 0XFFFF 后中断的方式，在溢出中断里累计中断次数。然后应用程序里面用溢出次数*65535+TAR 值，就是连续计数值。

Q64：用 430F413 定时器 A 对外部引脚（48 脚，P1.5/TACLK/ACLK） 的 2M 脉冲计数，计数始终为 0，而对内部 ACLK 或 MCLK 计数则正常，不知如何解决？
A64：理解错了 P1.5/TACLK 为外部时钟输入，不是说捕获脉冲输入。

Q65：使用了 Timer_B 的单元 1 和单元 2 测量频率，计数过程中发生的溢出中断，如何判断这个溢出中断是哪个单元产生的？
A65：TIMER_B 使用两个中断向量 TBCCR0 中断向量 CCIFG 和 TBIV 中断向量。TBIV 包含所有其他的 CCIFG 和 TBIFG。

Q66：知道定时器可输出自定义占空比的 PWM 波，找了些参考程序看，大概是这样的
TACTL=TASSEL0+TACLR+MC0;
CCTL0=CCIE;
CCR0=360;
CCR1=327;
CCTL1=OUTMOD_2;
P2SEL |= BIT5;
这样是不是直接从 P2.5 端口输出 PWM 波，芯片不能工作在低功耗下啊？怎么样让芯片工作在低功耗，等有波形翻转时产生中断啊？
A66：推荐使用 OUTMOD_7 模式那样比较好计算指定占空比时 CCR1 与 CCR0 的值，这种情况下芯片可以工作在低功耗模式下只要你保证在该模式中你的 TA 时钟源仍然开启就行，至于波形翻转产生中断这种，直接把 CCR1IE 开起来就行了。每一种低功耗开启和关闭的时钟不一样，所以一样能够进入低功耗的。

Q67：问题：#include<msp430x14x.h>
void main(void)
{
WDTCTL=WDTPW+WDTHOLD;//关狗
P1DIR=0XFF;
P1OUT=0X00;

```
TACCR1=200;
TACCTL1=CCIE;
_EINT();//使能
TACTL|=TASSEL_2+ID_2;//smclk
TACTL|=MC_2;//连续
LPM0;
}
#pragma vector=TIMERA1_VECTOR
__interrupt void time(void)
{
if(TAIV==0X02)
{
P1OUT^=0X01;
}
}
```

这样仿真时,进入中断,taiv=0;这是为什么?按理说 taiv=0x02

A67：TA 有两个中断源：

TA0_VECTOR 这个中断源里面只包含有 CC0IFG 这一个中断向量，也就是说 TACCR0 的比较中断和捕获中断都在这个中断向量表中。

TA1_VECTOR 这个中断源中包含有除 CC0IFG 后的其它中断向量，有 CC1IFG、CC2IFG、TAIFG，由 TAIV 的值来确定进入那个中断向量中。

Q68：用 430 单片机进行连续的数据采集，每次持续时间得几十秒，但是还要用单片机实现实时时钟功能，期间每秒都得进 Timer 中断计时。这样势必会影响数据的连续采集。这种矛盾该怎么解决呢？

A68：数据采集启动后,有一定的时间才能完成 AD 转换.这段时间内,CPU 要么空转,要么进行时钟处理.

因此我认为 CPU 干这两件工作完全胜任,只要:

1. 提高 CPU 时钟到 8MHz。
2. 用中断方式进行 AD 及时钟处理,中断程序要高效小巧。
3. 数据采集从微观上讲,并不是连续不停进行的(这是模拟电路的带宽及数据稳定所决定的),因此合理选择单位时间内 AD 转换的次数。
4. 要连续采集几十秒数据,单位时间内 AD 转换的次数应与系统的数据存储空间大小一并考虑。

Q69：如何实现间隔 1 小时触发定时器中断进行采集，对定时器如何设置呢？如果我用 32K 的时钟源，最多是不是只能实现 2 秒触发一次？

A69：时钟选择 32768HZ，计数的最大范围为 65535，最多为 2s 呢。要实现一个小时的延时只需要计算中断次数就可以了。

Q70：我想用 TA 的 CCR0 定时中断，增记数模式，但我看书上的 CCIFG0 的设置只有在 CCR0 与 TAR 的值相等时才置位，下一个周期马上又复位了。要是此时正在运行一条指令，那岂不是错过了进入中断了吗。可能我的理解有错，请指教。

A70：CCIFG0 应该相应了中断服务程序后才复位啊！或者用指令清除,不会错过的.。

Q71：用 430 进行脉冲记数问题，是不是可以用 TIMER－－A 的捕捉功能呢？？

A71：可以的，TA 中的捕获功能就是做这个事情的，记录脉冲的个数。

Q72：一个 **TA** 或 **TB** 最多只能产生两个中断吗。要不是的话，中断向量表该怎么处理？？

A72：应该说是两个中断向量地址，不能说产生两个中断。**TA**、**TB** 总 **4** 个中断向量地址，**TA**、**TB** 各有两个，如 **TIMER_A** 的比较模式采用 **CCR0**、**CCR1**、**CCR2** 就可以产生三个中断，但中断入口只有两个，**CCR0** 占用一个，**CCR1**、**CCR2**、**TAIFG** 占用一个，通过中断标志来判断是那个中断源引起的中断。

Q73：MSP430 中断所需的最小输入脉宽是多少？

A73：最小中断脉宽必须大于 **1.5** 主时钟周期 **(MCLK)**，以确保中断有效。有关问题请参阅器件特定的数据表。

第七章：系统时钟类

Q1：想找一款时钟芯片能和 MSP430 结合的，最好是并口的，这样不用占用串口

A1：节约成本考虑：使用 430 的定时器可以做一个实时时钟 RTC,不会引起多余的功耗。

另外，市面上时钟芯片多为 I2C 接口，可以通过 430 的 2 个普通 I/O 口来模拟实现，而不占用硬件串口。

Q2. 430f435 晶振最大能接多少？能接 32.768 兆吗

A2: msp430f4xx 最大只能接 8M，数据手册有相关说明

Q3：选择低频晶体(LFXT1CLK)做 ACLK.系统进入低功耗模式四(LPM4),这时引发一个外部中断,在中断服务程序里,DCO 是启动了吧,那么低频晶体有没有起振啊?

A3：进入低功耗 4 所有的时钟均停止，当外部中断来临时，DCO 自动被唤醒，然后由 DCO 去引导晶振起振。

Q4：用**msp430f147**外接了一个**6M**晶振，占用两个串口工作，一个波特率设置为**2400**，另一个设置为**115200**，但是运行的时候发现，波特率为**115200**的那个串口不能正常进中断接收数据，经过不断试验，发现当其改为**38400**的时候就可以正常工作了，但是波特率再往上调通讯都不正常。以前做过一个试验，只用一个串口，同样外接**6M**晶振，波特率设置到**57600**都没有问题。问题出现在哪呢？

A4：注意你的232芯片的最大工作波特率,可能是这方面存在问题。另外你测试的时候最好只有一个串口再工作，另外提醒下：小数寄存器是否配置妥当？

Q5：MSP430 F2 系列的外接晶振一般多大？我看大部分资料上写的都是 32768 晶振。可以大点吗？比如 1M 或者 4M。

A5:这个最好看下用户指南和数据手册，如果至此高频模式的话，那么配置成 1M 和 4M 都是可以的。

Q6:

问题：假如波特率为9600，时钟频率为32768，波特率控制寄存器为32768/9600=3.41 .UBR00 = 0x03;

UBR10 = 0x00;

UMCTL0 = 0x4A;

这个 0x4A 是如何来的啊？

A6：是为了修正 0.41 的偏差，其中 1 最好均匀分布。比如说 0x4a，

2 进制码是 1001010，这其实就是 3 个 1 均匀分布么

Q7：DCO 可以调到很高的频率，系统主时钟有上限吗？是多少？

A7:看是哪个系列的了，1XXX 和 4XXX 在 8MHZ 左右，2XXX 多数可跑 16MHZ。另外还与工作电压有关。结合你的实际情况选择

Q8:

1、XTIN XTOUT 接低速晶振，最高可以接多大频率晶振，可以接 8M 的吗？

2、用 DCO 作为主时钟，最高频率可以到多少？

3、XT2IN XT2OUT 接高速时钟，最高可以接多大频率晶振？（看资料，好像有的可以 8M，有的可以到 16M）

A8：1 .将 xt1 改成高速模式 可以接 8m 晶振

2. 根据系列不同而不同 比如最新的 5xx 系列能到 25m 而 2xx 则是 16m

3 . 这个得看你选择的芯片型号

Q9：以 57600 波特率实现 PC 与 1611 通信。PC 给 1611 发送 11 22 33 44 55 66 77 88，单片机只能收到 11 44 77 总是隔 2 个收到一个数据；单片机给 PC 发数据的话，PC 都能收到。

```
void Baudrate57600(void)
{
UBR0_0=0x8B;//波特率发生器选择 SMCLK,57600
UBR1_0=0x00;
UMCTL_0=0x00;
```

}

怎么回事？

A9：1、1611 的收发速度小于 PC 的,从你提供的情况,大约为 PC 的 1/4.降低波特率或改进 MSP1611 的收发代码执行效率.

2、建议 MCLK 再提高一些，MCLK 不要和 UART 的时钟是同一个时钟源，再试试。

Q10：要使串口 2 输出波特率为 115200 是不是必须要在 XT2 加个 8M 晶振呀？用加电容吗？

A10：不需要那么大,可以考虑用 DCO 产生，但要注意校准，如果用 XT2 最好要加电容的。

Q 11： 我用的是 430F449 我想用 XT2 作为我的 MCLK，但是怎么也切换不过去，一直都是 DCO 再工作的，我打电话给技术支持，按照他们说的都切换不过去，请问大家谁有好的办法不？或者说下这个切换的过程，最好有程序，谢谢了

A11：由于 XT2 时钟失效自动切换到 DCO。因此程序中首先打开 XT2，判断晶振失效标志位，直到 XT2 正常工作，OFIFG 为 0 后再执行其它程序。可用以下程序再做尝试：

```
void init_XT2(void)
{
  unsigned int   t;
  FLL_CTL1 &= ~XT2OFF;
   do{
    IFG1 &= ~OFIFG;
    for(t=0xfff;t>0;t--);
  }
  while(IFG1 & OFIFG);
  FLL_CTL1 = SELM1;                    //MCLK-->XT2=4MHz
```

Q12：CALBC1_1MHZ 这个值到底是多少啊，在那有它的定义？

A12：对于 2XX 系列由于 DCO 没有锁频环，因此不能够自动校准频率，需要通过手动配置 DCO MOD RESEL 来校准频率。2XX 单片机在出厂时候，TI 已经对常用 1MHZ，8MHZ，12MHZ，16MHZ DCO MOD RESEL 寄存器配置已经配好，保存在信息段里面。具体位置还有定义你可在 2 系列的头文件里查找。

第八章：LCD 显示驱动类

Q1：用 MSP430F470 来驱动段式液晶,发现有的段不该显示的时候,却有点模糊,该显示的段位,却亮度达不到有些段位,比较小的段就比较清晰。

A1：常见的 41X,42X,43X,44X 芯片的驱动需要外部的调压电阻，3 个 1M 电阻可以调整到 100K-1M 之间，达到较好的亮度和对比度。IO 选择为 LCD 功能，并且没有被 LCD 应用到的，要悬空。

Q2：我用的是 430F413 的片子，配的是点正液晶显示器。在整个电路调好后，有部分的电路液晶显示，放置一段时间后液晶出现浅显（不该显得都显出来了）。

A2：MSP430F413 自带 LCD 段式液晶驱动器，配的是点阵液晶，估计是液晶驱动程序没有写正确。

Q3：晶体一般都是接 32768，然后使用液晶很正常。我打算将晶体接 6M 的替换 32768，那么液晶还能正常显示吗

A3：看你所用的 LCM 模块时序极限是多少 HZ，然后看 6M 情况下，MSP430 去驱动 LCM 时，程序时间会不会超过这个极限频率，如果超过，得加延时。

Q4：msp430fw427 制作水表 液晶显示模块，能用 LCD1602 吗？

A4：1602 是点阵液晶吧！那样比较费电，还是用段式液晶好 1602 是点阵液晶吧！那样比较费电，还是用段式液晶好 FW427 本身自带了段式液晶存储器，一般水表商均自己开模去做的段式液晶 ，毕竟水表所需表达信息量并不多， 没必要使用点阵液晶。

Q5：I/O 怎样直接驱动 LCD，如何做？

A5：建议采用带 LCD 驱动的芯片。可采用 MSP430F4xx 系列的芯片。如果不用带 LCD 驱动的芯片，可以用 IO 口仿 LCD 的波形，比较复杂些，不过也能做出来。可以到 TI 网站上去下载关于 LCD 的应用报告。

Q6：用 CMOS 系列芯片驱动 LED 数码管，进行 LED 数码管的静态驱动，但不知道用那一款 CMOS 芯片，有什么好方法？

A6：CD4094 或 HC4094 芯片，74LS245/74hc245/74hc373 也可以,可选的芯片比较多,觉得 CD4094 可以，这个成本比较高的吧,建议可以采用一个升压的电源管理芯片+一个普通的移位寄存器来实现。CAT4238+74HC164 来实现。像 74hc373 这些都可以。一个数码管可以静态；2 片 74hc373 扫描方式可以驱动 8 个 LED 数码管。

Q7：段式液晶和点阵式液晶区别和不同的应用？

A7：段式液晶显示的信息较少，但是便宜，驱动电路简单。点阵液晶，显示信息丰富，可以显示汉字，图片，但是比较贵，驱动电路比较复杂，驱动程序也要比段式液晶的驱动程序复杂许多。

第九章：通信类

Q 1: 430 串口中，有个 R/D 控制线，在接收上位机的数据，但本身的数据有无发送完毕不知道啊，什么时候才可置低 R/d 位来接收数据啊？好像 430 没有发送完中断标志

A1：字节主动发送，一般都能发出去，除非你的的时钟有问题．可以用程序检测，

半双工通讯，可以多发送一个字节，作为判断，当最后个字节（作为判断用）写入发送缓存产生中断时，在中断里改变R/D状态。

Q 2: 我想做个 6 个节点的网络，最远的距离为 2m
F20 系列的 USI 做 3 线 SPI 模式用是否能满足要求？
能否提供其他低端的 430 芯片？

A2：SPI 是一种高速的，全双工，同步的通信总线，并且在芯片的管脚上只占用四根线，节约了芯片的管脚，同时为 PCB 的布局上节省空间，提供方便，正是出于这种简单易用的特性，现在越来越多的芯片集成了这种通信协议..
但作为芯片间的通信,距离不能太远,如果你想做 2m,建议你用串口 232 去做!

Q3： 从 PC 端，用串口调试助手发送一个字 MSP430 可以接收到，但是我发送一个字符串 MSP430 就接收不到了，不知道为何，程序如下：

```
#pragma vector=UART0RX_VECTOR
__interrupt void usart0_rx(void)
{
RecBuf[revcont] = RXBUF0; // RXBUF0 to TXBUF0
revcont++;
}
```

用上面的中断程序接收到后，发送 RecBuf 到 PC 串口....我如果发送 0x01 0x02 到 MSP430 后 ，再从 MSP430 发送到 PC,我接收到的是 0xE0

A3： 查查你的程序,在中断程序中的 revcont++; 会不灰导致 RecBuf[]越界;发送 0x01 0x02 0x03 0x04 在串口调试助手里 应该是 01 02 03 04 并点以十六进制传输.否则 PC 将会以 ASCII 码形式发出.

Q4：本人想利用单片机的定时器,DMA,Flash 常数表和 DA 产生一定频率的正弦波.希望 CPU 在初始化设置好之后不再占用 CPU 资源。有没有可能 DMA 输送一段数据,在我的时钟控制下一个一个的往 DA 送数,当这一段数据传送完成之后传送地址自动返回到这段数据开始再送呢？

A4：你的这种思路是可以实现的，不过要设置好 DMA
1、在“块地址到单地址”编址模式下，采用“重复块传送”方式；
2、要由中断来触发 DMA 的数据传送，从而实现 DAC 的数据更新，这时要使用 CPU
方法 1，即可实现无 CPU 干预的情况下数据的 direct memory access

Q4：请问,无线通讯系统,增加通信距离都有哪些方法呢?最有效最常用的

A4：加放大器，换增益更大的天线，降低通讯速率，降低载波频率，加大发射功率，基本就这些了。主要是硬件上，发射功率和你的高频电路负载的匹配

Q5：Modulation bits. These bits select the modulation for BRCLK. 调制位，不明白 umctl 是什么作用

A5：调整器，举个例子，BRCLK = 32768hz，要产生 2400 波特率，分频器分频系数为 32768/2400 = 13.65，所

以设置分频器的计数值为 13.接下来调整寄存器的值来设置小数部分的 0.65，调整器是 8 位，在调整器里要设置的 1 的个数是 0.65*8 = 5.
具体可以参考 USER'S GUIDE 或者介绍 430 单片机原理的书

Q6：低功耗的产品应该是不接ＸＴ**2**直接用ＤＣＯ的吗？用**msp149**，产品用到ＵＡＲＴ不知道ＤＣＯ稳定否？当波特率**9600**，**msp149**的ＤＣＯ最大频率多少啊？
A6：低功耗产品建议使用内部的 DCO，1 系列的 DCO 稳定度比起晶振确实不是很好，但是如果只是给 UART 提供时钟源，而工作的波特率是 9600 的话，还是可以胜任的，149 的 DCO 频率最大能达到多少还是看 149 的数据手册吧，内有详细介绍。

Q7：异步串口通信分别什么时候产生发送中断和接收中断？
A7：应该是先中断，在中断中发送数据，接收数据是一个字节收完才产生中断的。

Q8：**MSP430**的**BSL**下载的硬件电路是采取串口与**MSP430**的直连。如果复用串口信号线的话，会与普通的串口通信冲突吗？ 另外 用于**BSL**下载的各信号线需要做什么处理？
A8：**BSL**只是芯片中固定位置的一段程序，在执行应用程序它是不起作用的，所以不会与普通的串口通讯冲突。**BSL**下载的各信号线基本没有什么特殊的要求，如果复用时只要不将它们直接接地或者**VCC**就可以上拉下拉是没问题的，不过能不复用最好。

Q9：比如：**702.0**，在**MSP430**中占四个字节，为**00 80 2f 44**。是不是通过指针，一次发一个字节，发四次？而在上位机上用**VC**编程序时，串口接收数据时，怎样把这四个字节还原为**702.0**呢？
A9：简单的方便使用指针，分四次发，接收时直接使用指针存入一个浮点数中。

```
ptr = (char *) & fData;
ptr[1] = Rec1Buff[m];
ptr[0] = Rec1Buff[m + 1];
ptr[0] = Rec1Buff[m + 2];
ptr[0] = Rec1Buff[m + 3];
```

Q10：使用**USCI**模块**B**进行**IIC**通信，配置完寄存器后，置位＊＊＊**STT**位后，从示波器观察并没看到**SCL**和**SDA**线上发出起始信号，请问可能是因为什么原因？
A10：**2**系列为了降低功耗 只有在把数据写到发送缓存的时候才会产生时钟。

Q11.问题：请问**MSP430F247**的**SPI**总线的**SOMI**和**SIMO**上如果加上了**4.7K**的上拉电阻对总线有影响吗？
A11：一般来说增加上拉可以提高驱动能力，但是**SPI**可以寄存器配置是空闲高电平还是低，因此如果要上拉需要软件设置一致，否则出错。

Q12：**MSP430 SPI** 或 **UART** 的速度？
A12：在 **SPI** 主模式下，通信速率可以达到 **4Mbps**，而在 **UART** 模式下，速率也可达到 **2Mbps**。**USART** 可进行配置，以便同时支持同步 **(SPI)** 与异步 **(UART)** 操作，并且可从几个内部及外部时钟源（与 **CPU** 时钟无关）中进行选择。在 **SPI** 主模式下，**USART** 的运行速率可达到应用时钟的 **1/2**。例如，如果使用 **8MHz** 时钟，则 **SPI** 主模式的传输速率可达到 **4Mbps**。在 **UART** 模式下，实现可靠通信至少要求每位 **3** 或 **4** 个时钟。例如，**8MHz** 时钟除以 **4** 可以支持高达 **2Mbps** 的速率。**MSP430xxxx** 用户指南中提供了有关 **USART** 功能的完整说明，其网址是：**http://www.ti.com/msp430**。

第十章：IO 端口类

Q1：请问 430 的 I/O 中断能不能可靠的响应 60ns 的脉冲信号，就是来了一个 60ns 的脉冲，430 的中断会有丢失吗？

A1：端口支持的最高8M的时钟，无法响应这么快的频率。

Q2：430是3.3V供电，如果我想使P2.7管脚置高，然后等待低电平中断。接5V左右的电压接上拉电阻行不行？需不需要用分压电路到3.3V

A2：你可能要进行电平变换，430I/O口最好不要接5v的

Q3：当Jtag接口把程序下载到单片机之后，Jtag接口还能不能用作普通IO口？

A3: 和GPIO端口复用的JTAG引脚，只要不在仿真状态，就是普通的i/o引脚。

Q4：我在产品的测试中经常会发现，某一管脚输出电压不正常，本来应该是高电平，可是就有那么一个是低电平，而且不影响cpu其他管脚正常工作，有没有遇到相同问题的高手，能帮忙解决下吗，出现这个情况是不是 cpu就是坏了，不能正常使用了？

A4：首先确定该端口的PxSEL对应的位是0，比如2系的P2.6和P2.7默认与XIN和XOUT复用，如果不设置就是不能正确输出的。

然后才是看是不是虚焊，最后确认IO可能是坏的。

Q5：430单片机能不能用I/O口仿并口通讯，仿的时候需要注意什么，是不是只要把PXDIR设置成输出就行

A5：可以。向外写的时候要设为输出，读取数据时要设为输入，根据情况随时控制PXDIR

Q6：MSP430 刚刚上电时I/O管脚的状态是高电平还是高阻态还是低电平啊？

A7： I/O在设置之前的状态是输入状态，电平不定的，这个最简单了，你只要测试以下就可以知道答案。关于I/O及其他寄存器的初始值，在相关的系列Users guide 中有描述。

Q8. P1 REN 这个寄存器是什么用途的？

A8：是控制上下拉是否启用的。用法是 DIR 置为 0 也就是输入状态时，通过 OUT 的值控制上拉还是下拉。当 REN 置位时，就被拉，否则输入是高阻的。

Q9：在同一个程序中,前半段用的是 P3.1(SIMO) P3.3(UCLK) 的 SPI 模式,然后在接下来的程序中要 P3.3,P3.3 作为普通 IO 口输出时钟. 用什么指令能使 SPI 工作完后清除 SPI 功能,变成好象没使用过 SPI 模式前一样？

A9：可以的，IO 口的 IO 功能及辅助功能可以随时切换的。一般而言只要更改 PxSEL 寄存器即可。具体可以查看一下相应的用户指南及数据手册

Q10：低功耗设计中 430 接ＣＭＯＳ器件要注意点

A10：在低功耗产品设计中，当４３０的 i/o 口与 CMOS 器件接口时，比如ＬＣＤ之类的，为了省电，ＬＣＤ间歇供电，当ＬＣＤ关断时，与之相连的 430 的 i/o 口一定要设置为输入口或者设置为输出为１．否则 LCD 内部 cmos 器件的Ｉ／Ｏ口通过二极管导到其ＶＣＣ上，这样ＭＣＵ的电流变大了，功耗变的更大了．

Q11：430 I/O 最大驱动电流是多少呀？

A11：每个 I/O 输出电流最大允许 6mA，数据手册（datasheet）上有详细说明。

Q12.：msp430201X 的电源电压为 1.8V，其 IO 口电平电压是不是也是 1.8V 呢，对最大输出电流会不会影响呢？

A12：若 DVCC 采用 1.8V 的，则 I/O 的高电平不会超过 1.8V 的，其 CPU 核的电压与外围模块的电压用的是一个电压，不过在其 F5xx 系列中，CPU 核电压与外围模块电压分开了。

A13：请问：如果不用的口,方向设做什么好,输入还是输出?为什么?

Q13：输出低电平，或者设成输入，并接地。效果是差不多的

Q14：在 F149 芯片应用时使用了其中的两个串口进行通讯，现在想增加通讯的串口数量，不知道 F149 芯片能不能进行端口功能的扩展，若能，怎样实现？另外怎样在外部扩展 ROM 区，具体怎么实现，请指教！

A14：可以用定时器模拟几个串口，timera 专门为这个功能做了特殊处理，请参考例程。

430 不开放总线所以扩展 ROM 是不可以的。

Q15：JTAG 与 I/O 功能之间的 MSP430 引脚复用？

A15：四个引脚 P1.7 - P1.4 在 20 与 28 引脚 MSP430F1xx 器件上均同时具有 I/O 与 JTAG 功能。这些引脚的默认功能是，当器件通电时具有 I/O 功能。当测试引脚拉高时，则将这些引脚选为 JTAG。当使用交互式系统内调试程序时，这些器件的 FET 会将这些引脚处于 JTAG 模式下。如欲了解有关在使用调试程序时从 JTAG 模式发布引脚的信息，敬请参阅《FET 工具用户指南》。

注：如果将外部电路附加到共享引脚上，则必须考虑 JTAG 信号对引脚的相互影响。

如果通过 JTAG 对该器件进行系统内交互式编程或调试，则需考虑电路将产生的影响。如果电路将增加共享引脚的负载或偏置，进而干扰 JTAG 通信，则应考虑这一点。更高引脚计数器件具有仅可用于调试与编程的专用 JTAG 引脚。

Q16：MSP430 I/O 引脚的汲极电流与源极电流？

A16： MSP430 未指定来自 I/O 引脚的最大绝对电流。如欲了解 Voh 与 Vol 的规范，敬请参阅数据表。其中显示了每个 I/O 引脚均可提供几毫安的电流，但输出电压将随着电流的增大而发生变化。这些规格的附注通常提供了要维持特定电压，所有组合的输出提供的最大总电流。 MSP430 I/O 不适于驱动高电流的 20mA LED。

第十一章：FLASH 存储类

Ｑ１：用 IAR Embedded Workbench for MSP430 通过 JTAG 往 MSP430 上写程序。为了知道片内程序的版本，必须读出 Flash 中内容。什么工具软件可以通过 JTAG 口实现这个功能？

A1: 熔丝未烧断的话，做个空程序的项目，然后在C-SPY选项里选择保留未改变的区域，DEBUG后看MEMORY里的内容！熔丝烧断的话只能用编程器或BSL，如果加密了，只能通过BSL来读了，不过你要知道中断向量表的32个字节的内容，即密码。

Q2：MSP430F449中我想把一个整形的数A存入某地方掉电也不丢失，作为以后程序运行的参数。是放在flash里面吗?用代码怎么实现?

A2：如果只是个巴字节的话就把它作为数组或者变量定义到FLASH就可以了。

如： const unsigned char Value@0x0C000；

Ｑ3: 430里面Flash的主存储区和信息存储区有什么区别么？是不是程序是保存在主存储区里面的?那信息存储区是存什么信息的呢？存进去之后是不是随时能够读写出来呢?

A3: Flash分主Flash和信息Flash。如你所说,Flash主存储区主要来存储程序。信息Flash就是保存数据用的，可以随时读取。不过从物理特性而言他们是一模一样的，就段的长度有区别。当然信息段你也可以用来存储程序使用。

Q4：在 flash 单字节写的时候,如我写在D区,可以不用全部清D区吗?因为D区还有先前保存的数据.

A4： 不可以。只能先读出然后全部擦除再重新写入,如果FLASH可以如您说的那么操作,那就是EEPROM了

Q5： 目前使用的单片机是5V供电，买的flash存储芯片3.3V供电，单片机和存储芯片的连接是怎么的呢？我查看一些相关资料，说是可以直接连接，但是直接连接有问题，请问如何连接？？需要什么样的方法？？

A5： 最好在FLASH与MCU之间接一个电平转换芯片，或者用MOFET管转换一下电平。

Q6：请问擦写 FLASH 选择频率时,有没有要特别注意的,如果我用 8M 的,不分频,这样可以吗?

A6：要注意啊，擦写Flash的频率在250～470K（具体可以查查手册）；必须对FCTL2进行设置，使频率在这个范围内。

Q7: 用 jtag 接口往 430 中写程序会不会破坏原有 flash 信息段保存的数据啊？

A7：取消擦除信息段选项，即在 IAR project 的 options 选项中 DEBUG 下的选项。

Q8：从数据手册上来看，在写的过程中，好像要求判断BUSY位来决定接下来的操作，可是为什么TI的例程里面没有判断BUSY bit ？是不是MCU会自动判断？

A8：430对FLASH的写操作多种。比如块写（××OCK WRITE）和字节/字写（BYTE/WORD WRITE），在不同模式下，允许写的操作也不同。比如在字节写（BYTE WRITE）操作下，允许在FLASH MEMORY或在RAM下进行。

当在FLASH MEMORY下进行写操作时，此时CPU是挂起的，直到写操作完成它才能执行下一个指令操作，

您看到的例子，可能就是这方面的。但是当擦除程序是在RAM内初始化的，那么CPU就可以一直执行，如果不判断ＢＵＳＹ位的话，当RAM内代码执行完毕（us级），ＰＣ跳转到FLASH，而此时FLASH还再做擦除操作（ms级）就会造成非法访问，产生不可屏蔽中断。

Q9：MSP430 的编译器默认是将数组定义在 RAM 里面。请问如果我要将定义的数组直接保存在 flash 里面应该如何定义呢？是不是要修改编译器的某个配置信息，然后直接用 const 定义呢？

A9：这个如何分配，以及用什么关键字都是由编译器决定的，如果你用的是 IAR Embedded Workbench for 430，那再在数组声明的时候，前面加一个CONST那么在编译的时候就会将数组分配到FLASH 空间。如果不加 const 默认的情况是分配到 RAM 空间的。

第十二章：AD 转换类

Q1：用 430f1132 采样是 10 位的 ad 数据，而用 f169 的 12 位 da 输出，请问那边采过来的数据可以直接用吗？还是需要有一个换算才能用？

A1：如果参考电源一样的话，直接除 10 后再乘 12 就可以了。

Q2：在2274的ADC10里，
现在的

```
BCSCTL1= CALBC1_8MHZ;
DCOCTL = CALBC1_8MHZ;
void ADC10_SET(void)
{
ADC10AE0 |= BIT1; // 选择 ADC10 A1(P2.1)采样通道
ADC10DTC0 |= ADC10CT+ENC; // 选择连续转移数据
ADC10DTC1 = 62; // 连续采样62次中断一次
ADC10SA = (uint)ad_value+1; // ADC10 数据转移起始地址
ADC10CTL1 = INCH_1+ADC10DIV_1; // 选择第10通道,为片内温度传感器
ADC10CTL0 = REFON+SREF_0; // 打开1.5V正参考,地为负参考
ADC10CTL0 |= ADC10SHT_3+ADC10IE; // 打开ADC10内核,设定采样保持
```

时间为64个ADC10CLK,使能ADC10中断

```
ADC10CTL0 &= ~(ENC+ADC10ON); // 关闭 ADC10 转换
}
```

现在采样时间是125US,采样62次中断一次,现在要指把采样时间稍微的加长一点如何设置？

A2：ADC10CTL1 中的 ADC10SSEL 没有设置，说明默认 0，也就是说 ADC10 模块的时钟是自带的时钟 ADC10OSC，频率一般都是5MHz，也就是说AD模块的fadc10=5M，Tadc10=0.2us
ADC10CTL0 中的 ADC10SHT 为 3，也就是说采样时间是 64 个 Tadc10 也就是 64*0.2=12.8us
不知道你的 125us 从何而来。如果 AD 时钟用 1M，相应采样时间是 64us，采样时间不是想设多少就可以设多少的 从上面的计算就可以看出 是由时钟和 SHT 决定的

Q3：初步的设想是使用模拟开关切换16路电阻值，阻值在0-1000之间，精度0.5或1;
现在遇到的困难是，4067的开关电阻在3V供电的情况下较大，300 400的样子，而且此阻值在输入电压变化时也有上百的变化，不适合用作做积分转换。
是不是有什么办法来消除开关电阻的影响，或者是使用slop A/D法是不是无法实现多路转换？或者是通过软件可以校正误差？重复精度如何？

A3：切换稳定后再测量,SLOP 的方法本身就很慢,测量完一次需要等 1 秒后再测下一次,这样精度才高.

Q4：ad 转换中 inch_10 表示第十通道，可是第十通道是哪个管脚啊？

A4：一般是内部温度传感器

Q5：MSP430F149 AD 的输入阻抗有多大？

A5：RC<2000 欧*30PF

Q6：MSP430 ADC12 模块的速度？

A6： ADC12 的转换速率是转换所需的 ADC12CLK 以及时钟的一项功能。ADC12CLK 的近似最小值与最

大值分别为 500kHz 及 6.5MHZ。速度最快的整个转换过程可以在 17 个周期内完成（13 个周期进行转换，4 个周期进行采样及保持）。6.5MHz/17 = 382ksps。ADC12 的运行速率不能低于最小值的 ADC12CLK，但在软件的控制下，采样门可以无限制保持打开状态。如欲了解有关采样与转换时间规范的更多详情，敬请参阅数据表。

Q7：利用 MSP430F1611 制作一个函数发生器，信号最后经 DAC12 输出，函数波形方波，正弦波，三角波等，频率要求不高 200Hz 以下，应该如何进行就？

A7：

1、在 FLA**SH** 中存储波形数据

2、联合使用 **DMA** 和 **DAC12**

3、**DMA** 设定在“块到字”编址模式，“字到字重复”传输模式

4、用 **TA** 或 **TB** 的 **PWM** 功能来触发 **DAC12** 数据装载，再用 **DAC12** 的 **DAC12IFG** 来触发 **DMA** 数据传输，这样实现由 **TA** 或 **TB** 来连动 **DAC12** 和 **DMA**

5、**TA** 或 **TB** 的 **PWM** 周期乘以波形点数就是输出波形周期

比如 **TA** 或 **TB** 的 **PWM** 周期是 **0.00015625** 秒，而没周期波形数据为 **32** 点，那么 **0.00015625*32=0.05** 秒，即输出信号频率为 **2**00Hz

Q8：**msp430f449**如何将模拟量转化成数字量，**adc12** 的参考电压如何设置？

A8：根据参考电压 如果参考电压事 **2.5V** ， **12** 位就是 **2** 的 **12** 次方**=4096**，那就是把 **2.5V** 满幅的输出定义为 **4096**，测到的数据是 **2005**，那么测量的电压结果就是（**2005/4096**）***2.5V** 你可以在软件中设置 **watch** 选项，查看 **ADC12MEM** 的值

Q9：SD16SD16 采集电压范围 V=+-（VREF/2/GAIN），我的硬件接法为： Vin1+ -->|A0.0+ ，Vin1- -->|A0.0- 那就是说 可以转换的输入电压 V=(Vin1+)- (Vin1-) ,的范围为=+-（VREF/2/GAIN）， 这样理解对吗？SD16CCTL 有 SD16DF 位就是数据格式 为双精度或单精度，这与输入电压的正负有什么关系？

A9：V=(Vin1+)- (Vin1-) 的范围<=+-（VREF/2/GAIN），最好是的范围是在 +-（VREF/2/GAIN）*80％ 。但是Vin＋或者Vin－不要超过Vcc。这个具体的datasheet上都有具体说明的。SD16DF只是数据输出格式的问题，这个具体在MSP430x4xx Family User's Guide 412页有具体陈述。输入范围跟数据格式没有多大的关系，只是在计算分辨率的时候跟数据输出格式有关系。

Q１０：用VREFON开通SD16的Vref后，需要等待多长时间才能进行AD转换？

A10：SD16的相关Vref引脚经常有外接电容，这是要等待一定时间。

如果没有外接，启动后等几个周期就可以用。

在接外接电容时：

1、VMIDON=1时，VREFON后须等待100us

2、VMIDON=0时，VREFON后须等待5ms

Q 11：在使用**MSP430F4250**的**SD16**的时候，发现的问题：

1、使用中断方式，采集的结果稳定性较好，使用查询方式，稳定性不好

2、我在使用**SD16**中断方式的时候，在**main**里循环显示数码管，数码管显示**AD**采集的数据，但是程序只在**SD**中断里，显示的程序一直没有执行，不知道是怎么回事？

A11：原因是**:SD**的中断优先于显示中断**.**

解决方法**:**定义一个数组**,**保存**SD**的结果**.**在**SD**中断中仅进行数据保存工作**,**在**MAIN()**中处理数据及显示**.**这样**SD**中断占用时间很小**.**

```
unsigned short Buffer[ n ] ;
unsigned short *pBuffer ;
void main( void )
{
....
...
pBuffer=Buffer ;
...
...
}

//SD中断
__interrupt void SD_Int( void )
{
*pBuffer++ = AD数据;
if ( pBuffer >= Buffer + n )
{
pBuffer = Buffer ;
}
}
```

第十三章：电源类

Q1：请问 **msp430**（我用的 **4619**）的 VCC,DVCC,VSS,DVSS 怎么接啊？模拟的和数字的一样吗？

A1：**CC** 就是正，**SS** 就是负，**A** 是模拟电，**D** 是数字电，**A** 的都接在一起，**D** 的都接在一起，地线要分开布，中间用 **000** 贴片连接，也就是传说中的单点连接。**A** 和 **D** 的正电源间如果信号源有固有的频率或频率范围建议中间用适当的电感连接，并在两侧加适当的退耦电容，以防止数字部分的信号干扰模拟信号。

要求不是很精的时候，模拟电源和数字电源多数都是连通的。

Q2：如果直接用干电池（比如两节 **AA** 电池）不经过 **PWIC** 给 **MSP430** 供电，时间长了电池电压下降，这时候是不是芯片的参数，比如说 **Voh/Vol/Vih/Vil** 甚至时钟频率都要发生变化了。如果电池电压降得远低于 **3V**，可能会导致电路工作不正常。这个理解是不是正确？

A2：可以这么理解，**VCC** 电压下降了，相应的 **IO** 口电平会变化，内部 **DCO** 震荡频率也会变化，如果远低于 **3V** 会导致电路工作不正常，但可以利用 **SVS** 模块来监控电压，或利用 **AD** 采样电池电压，当低于一定程度时关闭相关电路甚至 **MCU** 本身。

Q3：430 的电源范围很宽，一般来说电压越低，功耗越小，不知大家都使用的几伏的电压？以前用的 **3.3V** 的，现在想用 **3V**，不知稳定性怎样？在 **JTAG** 下载程序时有没有什么问题？

A3：具体用几伏电压的电源，要看实际情况：

1、在允许条件下，电源电压越低，**MCU** 能耗越低。

2、**GIO** 或其它功能引脚，比如 **LCD**、**ADC**、**USART** 等与外部电路或器件之间电压的匹配。

3、如果用了电源管理器件，过低的输出电压会增加该器件输入端与输出端之间的压降，从而增加其本身的能耗

举例：**3** 节 **1.5V** 铝电池

1、新电池电压 **3*1.55 = 4.65V**

2、电池能量快消耗完毕时的电压 **3*1.2V = 3.6V**

3、如果采用 **DC/DC** 或 **LDO** 作 **MCU** 的电源，则其输入电压与输出电压之差应大于 **0.3V**

4、选 **MCU** 电源电压为 **3V** 或 **3.3V** 都可以

5、如果考虑到防止电池电压接反以及作为电源保险， 最好在 **DC/DC** 或 **LDO** 前面接一只肖特基二极管，那么该二极管约 **0.2V** 的压降也要考虑进去

其它情况，可以类比。

Q4：MSP430F449 芯片，供电是 **3V**，无意中吧电源接反了，大概有几秒钟，但芯片没怎么热，怎么判断是否坏？

A4：测 **VCC** 和 **VSS** 间的双向导通情况：

用万用表电阻档测 **VCC** 对 **VSS**，电阻值不能低于 **M** 欧级别。

用万用表二极管档测 **VSS** 对 **VCC**，显示值不能大于几百欧姆。

或者也可以用仿真器测试。

Q5：用 **430** 做了个系统,用在发电机上,但目前发电机出来的信号干扰太大,**CPU** 容易死机,脉冲峰值有近 **400V**,有效值大概 **30V** 左右,电源波形很差,要把这个干扰去掉,使 **CPU** 不死机。有什么方法呢？

A5：滤除这种噪声，可以考虑一下加上瞬态抑制二极管。但是瞬态抑制二极管是可以解决一定干扰不能完全解决问题，还要考虑跟大地滤波。

Q6：如果一节电池是 **2400** 毫安时。如果放电电流为 **1** 毫安，就可以用 **2400** 小时，也就是说可以用 **100** 天？

A6：不对！

通常所说的电池额定放电容量是指在额定放电电流下放电到特定电压时的电池放电容量。当实际放电电流比额定放电电流低的时候，实际放电容量会有所增加；间歇放电情况下，实际放电容量也比连续放电情况下相对有所增加。同时当电池放电时，其电压也有下降，当下降到 **MCU** 不能工作时，就不能在放电了。举个例子：比如额定容量为 **2400mAh** 的电池，它的额定放电电流是 **50mA**，那额定放电时间就是 **2400/50=48** 小时。

如果实际放电电流是 **20mA**，小于额定放电电流，由于电池的理化特性，实际放电时间会比 **2400/20=120** 小时要长。反过来，如果超常规放电，放电电流远大于额定放电电流，则电池的容量就会减小，甚至会缩短电池的使用寿命。

Q7：1602 与 **430** 电平不匹配问题，用什么芯片进行电平转换较好。

A7：建议选用 **TI** 的 **SN74LVTH245,TI** 的 **74LVC245**

1、TI 的 **SN74LVTH245** 是专门为低压 **3.3V** 而设计的芯片，但也可以用于 **5V** 系统（**These octal bus transceivers are designed specifically for low-voltage (3.3-V) VCC operation, but with the capability to provide a TTL interface to a 5-V system environment.**）其用 **ABT** 技术（**Advanced BiCMOS Technology**）。输入电压 **VCC** 范围是 **2.7~3.6**。

2、TI 的 **74LVC245** 是通用的转换芯片，**VCC** 在 **1.65~3.6V** 之间。 其输出电压最大值是 **VCC**，输入可以是 **5V**，所以可以用作电平转换。

74LS245 是通用型 **3** 态 **transceiver,**其 **VCC** 范围 **4.5~5.5V**。