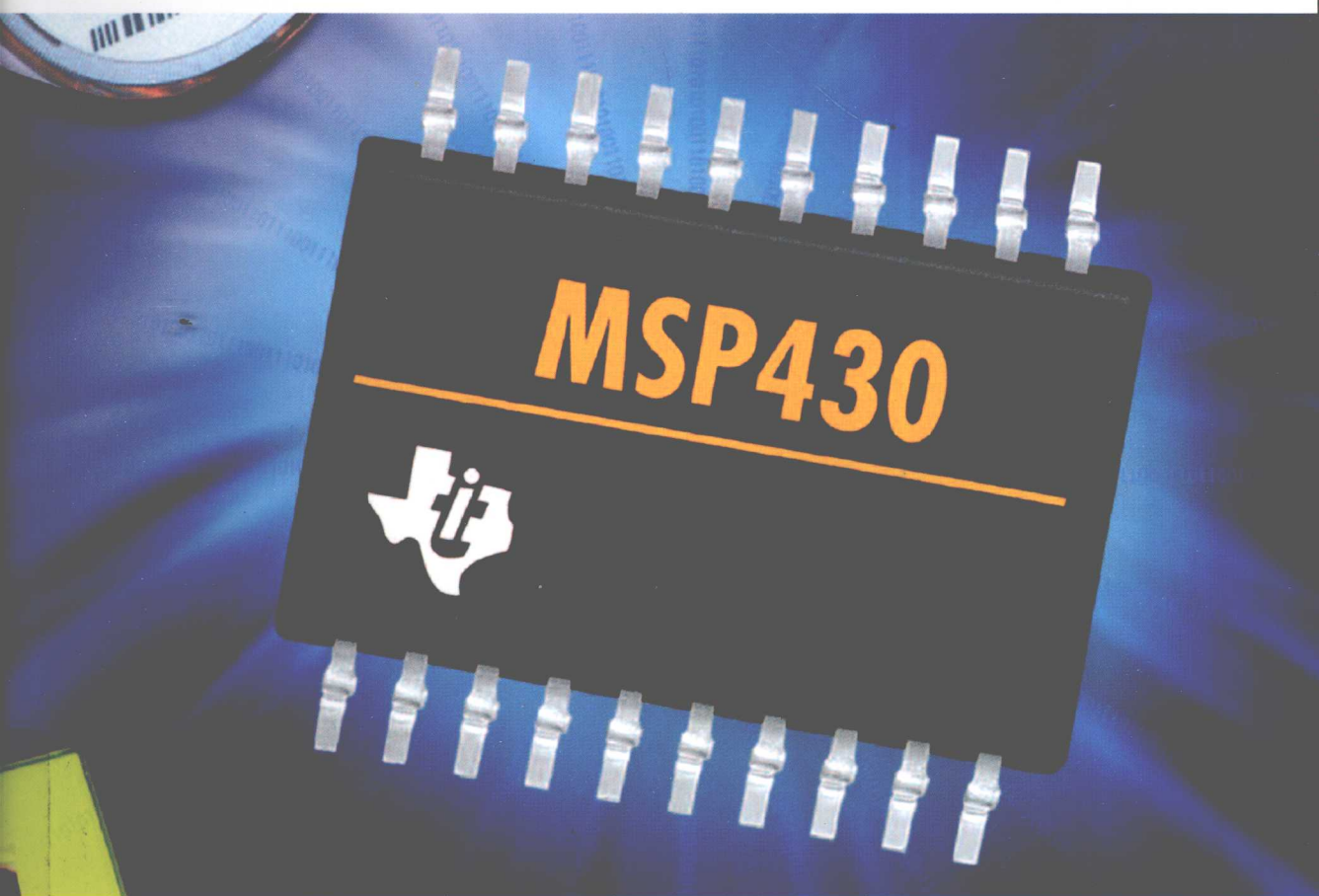




普通高等教育“十一五”规划教材

MSP430系列单片机系统 工程设计与实践



谢 楷 赵 建 编著



机械工业出版社
CHINA MACHINE PRESS



普通高等教育“十一五”规划教材

MSP430 系列单片机系统 工程设计与实践

谢 楷 赵 建 编著



机械工业出版社

本书以 MSP430 系列单片机(下面均用 MSP430 单片机)为例,介绍了超低功耗单片机系统软件设计、超低功耗外围电路设计、人机交互界面设计、嵌入式软件工程基础等基础知识和实践经验,使读者能够掌握超低功耗系统开发和设计所需的基本知识,并具有初步的软件结构规划能力。书中提供了全系列 MSP430 单片机的程序范例,并提供模块化程序库,可以让读者通过调用模块库内的函数,快速完成设计任务。掌握本书的内容,对于读者今后开发任何一款新的单片机都具有一定的帮助。

本书既可以作为本科生或研究生电子工程、测控技术与仪器、自动控制、机电一体化等专业的教学用书,也可以作为各类学生以及工程技术人员工程类参考书。

本书配有免费电子课件,欢迎选用本书作教材的老师登录 www.cmpedu.com 下载或发邮件到 Edmondyan@sina.com 索取。
Edmondyan@hotmail.com

图书在版编目(CIP)数据

MSP430 系列单片机系统工程设计与实践/谢楷, 赵建
编著. —北京:机械工业出版社, 2009. 7
普通高等教育“十一五”规划教材
ISBN 978-7-111-27386-8

I. M… II. ①谢…②赵… III. 单片微型计算机—系统设计—高等学校—教材 IV. TP368. 1

中国版本图书馆 CIP 数据核字(2009)第 092657 号

机械工业出版社(北京市百万庄大街 22 号 邮政编码 100037)

责任编辑: 贡克勤 版式设计: 张世琴 责任校对: 刘志文

封面设计: 姚 毅 责任印制: 李 妍

北京铭成印刷有限公司印刷

2009 年 7 月第 1 版第 1 次印刷

184mm × 260mm · 20 印张 · 491 千字

标准书号: ISBN 978-7-111-27386-8

定价: 38.00 元

凡购本书, 如有缺页、倒页、脱页, 由本社发行部调换

销售服务热线电话: (010)68326294

购书热线电话: (010)88379639 88379641 88379643

编辑热线电话: (010)88379725

封面无防伪标均为盗版

前言

MSP430 系列单片机是 TI(Texas Instruments, 美国德州仪器)公司近年来推出的一个优秀的 SOC 型混合微处理器产品系列, 它不仅具有 16 位高效的微处理器系统, 还具有丰富的、功能强大的外围电路资源, 其中也包括了许多高性能的模拟电路资源。目前, 在很多热门产品中都采用了 MSP430 系列的单片机, 其可贵之处在于它除了具备很好的数字/模拟信号处理能力外, 还具备了以极低功耗运行的特点, 可被广泛应用于要求低功耗、高性能、便携式的设备上, 即使在某些不需要低功耗的场合, 它仍然可以作为一款高性能单片机使用。

MSP430 系列单片机电路资源性能优异, 模拟与数字系统结合完美, 系列全面、技术先进、应用面广, 可用单芯片完成整个测控系统的设计, 特别适合在电子工程、测控技术与仪器、自动控制、机电一体化等专业的课程与实践教学中推广应用。

2003 年我们向 TI 公司申请了部分 MSP430 系列的单片机芯片和开发设备, 尝试将其应用在硕士研究生和高年级本科生的科技实践与毕业设计中。2005 年暑期全国大学生电子设计竞赛培训期间, 笔者在校内首次开设了 MSP430 系列单片机应用技术讲座。虽然课时很少, 且仅在小范围开展, 但学生的反响强烈, 表现出极大的兴趣。

2006 年我们和美国 TI 公司上海分公司与杭州利尔达科技有限公司共同建立了“MSP430 系列单片机联合实验室”, 把以 MSP430 系列单片机为代表的超低功耗单片机技术正式列入课程教学计划, 组织和指导学生运用 MSP430 系列单片机技术参加各种大学生科技竞赛活动, 并在全国大学生“挑战杯”课外学术科技作品竞赛中分别获得国家级特等奖、一等奖和二等奖各 1 项; 在全国大学生电子设计竞赛中先后获得全国一等奖 3 项、二等奖 2 项。在此期间, 我们所在的课题组利用 MSP430 系列单片机开发了数十种工业测控产品, 积累了大量的设计文档、技术方案等资料, 并且在教学和科研中总结了许多宝贵经验。

我们将这几年来积累的设计案例、课程讲稿、设计笔记、代码库、设计技巧等资料做了精选与汇总, 并结合课程教学与科研实践的经验, 编写了本书。内容上力求紧扣实际需要, 紧跟技术发展, 丰富实用, 通俗易懂, 而且结合了大量的工程实例, 使读者能够在按照章节循序渐进的学习过程中学会从工程实际出发, 体会产品设计过程中会遇到的各种问题, 并培养良好的编程习惯和设计风格。本书既可以作为本科生或研究生的课程教材, 也可以作为各类学生以及工程技术人员的工程类参考书。

当然, 既然本书是我们的经验总结, 就难免会有差错, 希望广大读者批评指正。最后, 再次衷心感谢美国 TI 公司上海分公司和杭州利尔达科技有限公司多年来对我们的帮助和支持。

谢 楷、赵 建

目 录

前言

第 1 章 MSP430 单片机入门基础	1
1.1 初识 MSP430 单片机	1
1.1.1 MSP430 单片机的应用 前景	2
1.1.2 MSP430 单片机的特点	2
1.1.3 MSP430 单片机最小系统	4
1.2 MSP430 单片机开发软件入门	6
1.3 MSP430 单片机 C 语言基础	13
1.3.1 变量	14
1.3.2 数学运算	15
1.3.3 位操作	16
1.3.4 寄存器操作	17
1.3.5 中断	20
1.3.6 内部函数	22
1.3.7 库函数	23
1.4 文件管理	23
1.5 代码优化	26
1.6 风格	27
1.6.1 变量命名规则	27
1.6.2 函数命名规则	28
1.6.3 表达式	29
1.6.4 风格一致性	29
1.6.5 注释	30
1.6.6 宏定义	31
1.7 可移植性	32
1.7.1 消除 CPU 差异	33
1.7.2 消除硬件差异	33
1.7.3 软件层次划分	34
1.7.4 接口	36
1.7.5 屏蔽	38
1.8 版本管理	41
本章小结	43
习题	43

第 2 章 MSP430 单片机的内部资源 ... 44

2.1 MSP430 单片机选型	44
2.2 I/O 口	45
2.2.1 I/O 口寄存器	45
2.2.2 I/O 口中断	46
2.2.3 “线与”逻辑	48
2.2.4 电平冲突	49
2.2.5 兼容性	49
2.2.6 电容感应式触控	50
2.3 时钟系统与低功耗模式	54
2.3.1 时钟系统结构与原理	54
2.3.2 低功耗模式	60
2.3.3 低功耗模式的应用	63
2.4 Basic Timer 基础定时器	67
2.4.1 Basic Timer 结构与原理	67
2.4.2 Basic Timer 中断	70
2.4.3 Basic Timer 的应用	71
2.5 LCD 控制器	74
2.5.1 LCD 的工作原理	74
2.5.2 LCD 与 MSP430 单片机的 连接	75
2.5.3 LCD 控制器的结构与原理	77
2.5.4 LCD 的显示缓存	79
2.5.5 LCD 控制器的应用	80
2.6 存储器与 Flash 控制器	87
2.6.1 MSP430 单片机的存储器 组织结构	87
2.6.2 Flash 控制器结构与原理	90
2.6.3 Flash 控制器的应用	96
2.7 16 位 ADC	103
2.7.1 SD16 模块的结构与原理	104
2.7.2 SD16 模块的中断	111
2.7.3 SD16 模块的电压测量应用	115
2.7.4 SD16 模块的误差及校准	120



2.7.5 SD16 模块的超低功耗应用	122
2.7.6 SD16 模块的高精度应用	124
2.7.7 SD16 模块的内部温度 传感器	128
2.8 16 位定时器 Timer_A	131
2.8.1 Timer_A 定时器主计数模块的 结构与原理	131
2.8.2 Timer_A 定时器的捕获模块 ..	134
2.8.3 Timer_A 定时器的比较模块 ..	136
2.8.4 Timer_A 定时器的中断	145
2.8.5 Timer_A 定时器的应用	147
2.9 增强型异步串行通信接口	164
2.9.1 UART 的结构与原理	164
2.9.2 UART 的中断	173
2.9.3 UART 的高效率数据发送 应用	174
2.9.4 UART 的高效率数据接收 应用	179
2.9.5 UART 的高效率数据帧 接收与判别	182
本章小结	185
习题	185
第 3 章 单片机软件工程基础	188
3.1 前后台程序结构	188
3.1.1 任务	188
3.1.2 实时性	190
3.1.3 前后台程序的编写原则	192
3.1.4 函数重入	205
3.1.5 临界代码	207
3.1.6 前后台程序结构的特点	213
3.2 状态机建模	214
3.2.1 初识“状态机”	214
3.2.2 状态机模型的描述方法	220
3.2.3 通过状态转移图生成代码	223
3.2.4 状态机建模应用实例	225
3.3 事件触发程序结构	232
3.3.1 事件触发结构	232
3.3.2 事件引擎	233
3.3.3 中断优先级与中断嵌套	239
3.3.4 事件触发程序实例	240
本章小结	245
习题	246
第 4 章 人机交互	249
4.1 超级终端	249
4.1.1 初识超级终端	249
4.1.2 printf 与 scanf 函数的原理	250
4.1.3 printf 与 scanf 函数的应用	251
4.1.4 printf 与 scanf 函数的配置	256
4.1.5 超级终端的应用	257
4.1.6 超级终端人机交互应用实例 ..	258
4.2 菜单交互方式的设计	261
4.2.1 4 键菜单的交互方式	261
4.2.2 3 键菜单的交互方式	263
4.2.3 两键菜单的交互方式	264
4.2.4 单键菜单的交互方式	264
4.2.5 菜单交互的设计原则	264
4.3 菜单的软件实现方法	267
4.3.1 菜单的公共函数	267
4.3.2 独占 CPU 的菜单实现方法	270
4.3.3 基于状态机的菜单实现方法 ..	275
4.3.4 菜单超时退出的实现	279
本章小结	280
习题	281
第 5 章 超低功耗硬件电路设计	282
5.1 超低功耗系统的电源	282
5.1.1 常用电池及特性	282
5.1.2 超低功耗稳压电路	287
5.1.3 1.5V 电池升压电路	288
5.1.4 自动升/降压电路	289
5.2 电源管理	290
5.2.1 电源开关电路	291
5.2.2 外部电路的电源管理	294
5.2.3 零功耗电路设计	296
5.3 超低功耗信号调理电路	297
5.3.1 超低功耗运放选型	297
5.3.2 超低功耗的基本模拟电路 单元	299
5.4 功耗的计算与测量	304

5.4.1	MSP430 单片机系统功耗的估算方法	305
5.4.2	功耗的测量方法	305
	本章小结	307
	习题	308
附录	常用锂锰纽扣电池参数表	309
附录 A	常用锂锰纽扣电池参数表	309
附录 B	本书范例所使用的电路原理图	310
	参考文献	311

附录	309
附录 A	309
附录 B	310
参考文献	311

第 1 章 MSP430 单片机入门基础

本章介绍 MSP430 单片机开发的步骤、开发软件使用方法、编程语言、编译环境的基本设置以及软件工程基础知识。还特别加入了代码风格、可移植性与软件管理方面的基本知识。国内教材、参考书对这方面内容介绍较少，而本书将这些内容放在第 1 章的目的在于让读者了解：养成良好的编程习惯和软件思想才是好的开端。

1.1 初识 MSP430 单片机

在开始学习 MSP430 单片机之前，先来做一个实验，直观地感受一下 MSP430 单片机的“超低功耗”称号。准备一个柠檬(橙子、苹果、猕猴桃等酸性水果均可)；找几个废锌锰干电池，拆下外壳的锌皮并剪开，压平；再准备 3 把铜钥匙(或铜币)。

将水果切成 3 份，在果肉两侧分别贴上锌皮和铜钥匙，构成原电池：铜是正极，锌皮是负极，水果的酸性成分构成电解质。实测每节水果电池电压约 0.8V 左右。3 节水果电池串联达到 2.4V 即可为 MSP430 单片机系统供电，如图 1.1.1 所示。

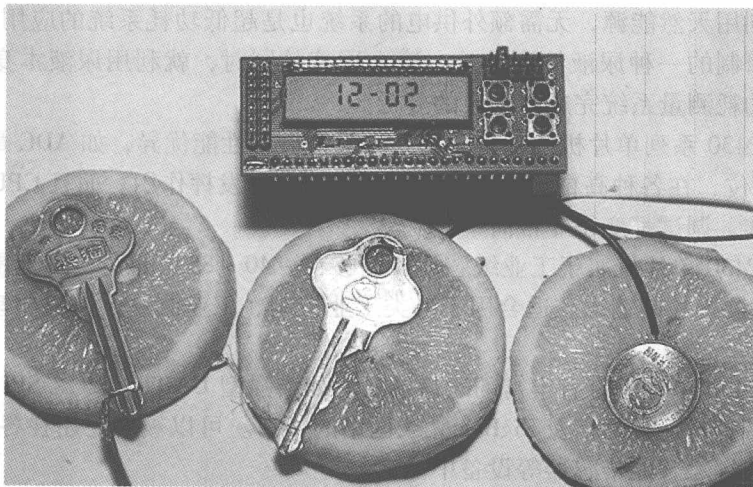


图 1.1.1 用水果电池供电的 MSP430 单片机系统

在 MSP430 单片机上运行一个电子表程序。只要水果不腐烂，能运行一个月以上，直到水果干枯。原电池只能提供极其微弱的电流(实测短路电流不到 $100\mu\text{A}$)；而在上述 MSP430 单片机系统中，运行一个电子表程序的 CPU 工作耗电不到 $2\mu\text{A}$ ，加上液晶耗电约 $3\mu\text{A}$ ，总电流不超过 $5\mu\text{A}$ 。对于一个水果电池来说，提供的电能绰绰有余。

再举个更直观的例子说明 $5\mu\text{A}$ 耗电的概念：一节普通的 CR2032 纽扣电池，能够为这个 MSP430 单片机运行电子表程序提供 5 年的电力。

1.1.1 MSP430 单片机的应用前景

从实验中,读者已经了解了超低功耗系统的概念:能够以极低(一般微安级)的耗电运行的系统。在实际应用中,许多产品和系统都对功耗提出了越来越严酷的要求,从而为 MSP430 单片机拓展了应用领域。

首当其冲的是便携式设备,随着便携式设备不断向小型化、轻量化、高精度、功能复杂化的发展方向,要求缩小电池体积、提高运算和处理能力、提高精度,与此同时还要求集成度不断提高,成本不断下降。MSP430 系列单片机不仅提供了强大的运算能力(16 位 CPU,目前最高能达到每秒 25 兆指令),而且能够以极低的功耗运行,并具有丰富的内部资源和各种模拟电路接口。这使得 MSP430 单片机不仅能够处理数字信号,还能够对模拟信号进行采集或处理。很多情况下,用 MSP430 单片机可以“单芯片”完成设计方案,这对提高产品集成度、提高生产效率、降低成本有着很大的帮助。

其次,超低功耗特性使得产品电池寿命终身化成为可能。一般的电子产品按 5~8 年寿命考虑,如果电池的预期寿命能达到 8~10 年,在产品整个生命周期内无需更换电池。这些产品可以将电池固化在内部,不需更换。例如某些野外安装的气象传感器,可以在无需更换电池的情况下,连续记录数年的气象数据直到寿命终结。

第三,利用 MSP430 单片机的超低功耗特性,让一些新的微弱能源为单片机系统供电成为可能。例如太阳能电池、信号线窃电、电缆附近磁场、温差能量、射频辐射、人体运动的动能等。这种利用天然能源,无需额外供电的系统也是超低功耗系统的应用发展方向之一。例如日本最近研制的一种尿液检测卡片,滴入尿液的同时,就利用尿液本身作为电解质发电,驱动超低功耗测量系统完成对尿液的分析。

第四,MSP430 系列单片机内部集成的各种模拟设备性能优异,如 ADC 最高可达 16 位,DAC 可以达 12 位,在各种高精度测量、控制领域都可以发挥作用。而且 CPU 与模拟设备的结合,使得校准、调试都变得非常简单。

第五,MSP430 单片机属于工业级芯片,能够在 $-40\sim 85^{\circ}\text{C}$ 的宽温度范围内工作,并且带有 PWM 发生器等控制输出。适合用于各类工业控制、工业测量、电机驱动、变频器、逆变器等。

第六,MSP430 单片机具有丰富的通信端口,如增强型 UART、I2C、SPI、USI 等,新推出的 5 系列还带有对射频通信及 ZigBee 无线网络的接口。可以被广泛用在各种协议转换器、数据转发器、中继、智能传感器等设备中。

最后,即使在某些不需要超低功耗的场合,让 MSP430 单片机全速运行,作为普通的单片机使用,仍然具有强大的运算能力。相比同价位的单片机系统,优势仍然十分明显。

1.1.2 MSP430 单片机的特点

MSP430 单片机最显著的特点是能够超低功耗运行。除了超低功耗特性以外,MSP430 单片机还具有许多其他特点。这里先做个简单概括,在后续的章节中,读者对这些特点将会有深入的体会。

首先,MSP430 单片机引入了“时钟系统”的概念。将 CPU、外围功能模块、休眠唤醒机制三者所需的时钟独立,而且可以通过软件设置时钟分频、倍频系数,为不同速度的设备

提供不同速度的时钟,并且可以随时将某些暂时不工作模块的时钟关闭。这种独特的时钟系统还可以实现系统不同深度的休眠,让整个系统以间歇工作方式最大限度地节约电力。

其次,MSP430 单片机内核是 16 位 RISC 处理器,单指令周期,其运算能力和速度都具有一定的优势。某些内部带有硬件乘法器的型号,在处理能力上在更胜一筹,结合 DMA 控制器甚至能完成某些 DSP 的运算功能。CPU 在 1.8 ~ 3.6V 宽电压范围内都可以工作,延长了电池的使用时间。

第三,MSP430 单片机采用模块化结构,每一种模块(内部资源)都具有独立而完整的结构,在不同型号的单片机中,同一种模块的使用方法和寄存器都是相同的。这为学习和开发 MSP430 单片机提供了便利。同一家族中不同型号的单片机,实际上就是不同功能模块的组合。此外,TI 公司每年都会新推出新系列、新型号的单片机,以及新的模块。模块化结构的另一优点是可以单独开启或关闭某些模块,只激活需要使用的模块,以节省电力。

第四,MSP430 单片机采用冯·诺依曼结构。寄存器以及数据段(RAM 区)与代码段(Flash 区)统一编址。如果将代码搬移到 RAM 区,同样可以运行,并且每一款 MSP430 单片机都集成有 Flash 控制器,通过它可以对 Flash 区进行擦写操作。这种存储机制可以很方便地实现在线升级甚至远程升级功能。例如可以在开发时写一段升级代码,将升级所需的代码复制到 RAM 区,再擦除整个代码区,然后运行 RAM 区的升级代码,最后将新版本的固件烧写到 Flash 内,就实现了固件更新。

第五,TI 公司具有雄厚的模拟技术实力。在 MSP430 单片机家族中,丰富的、性能卓越的模拟设备是一大特色。利用 MSP430 单片机,可以单芯片完成模拟信号的产生、变换、放大、采样、处理等任务。这对缩小产品体积、降低成本有着重要的意义。

第六,MSP430 单片机是一个不断更新、不断发展壮大的家族。每年都会有新的型号发布,不断会有新的系列推出,而且陆续推出的各种新型号单片机性能越来越强、功耗越来越低、性价比也不断提高。

在本书大部的范例以及附录的电路图中,用到的单片机型号是 MSP430FE425。现以该单片机为例,来直观了解一下 MSP430 单片机丰富的资源和强大功能:

- 16 位 RISC 指令集处理器,速度可达八百万条指令/秒。
- 512B RAM(数据) + 16KB Flash 存储器(代码)。
- 2 段 128B 的 Info Flash,可作掉电存储器使用。
- 内置 Flash 控制器,所有程序未用 Flash 均可作数据存储用。
- 内置时钟管理单元(FLL+),可软件设置 CPU 时钟倍频。
- 3 个独立的 16 位高精度 ADC。
- 内置可编程增益放大器(1 ~ 32 倍)。
- 内置温度传感器。
- 内置 1.2V 基准源和输出缓冲器。
- 内置 128 段 LCD 驱动器,可直接驱动段码式液晶板。
- 1 个增强型 UART 串口,自带波特率发生器,支持 7/8 位数据、支持奇偶校验、支持 9 位地址模式、带自动数据帧判别功能、可配置成 SPI 模式...
- 内置看门狗,也可配置成定时器使用。
- Basic Timer 定时器,方便产生 1/2"s 定时。

- 16 位 TA 定时器，带 3 路捕获和 2 路 PWM 发生器。
- 自带 BOR 检测电路，能自动躲避上电瞬间的毛刺并产生可靠的复位信号。
- 14 个双向 I/O 口，每个 I/O 口均可作为中断源。
- 自带仿真调试功能和调试接口，支持两个断点。
- 内置电能计量模块，可无需 CPU 干预的情况下自动完成交流电压、电流有效值、功率、有功电能等测量和计量工作(和 ADC 不能同时用)。

从上面的资源列表中可看到，MSP430 系列单片机不仅仅是一个微处理器系统，还包含模拟电路、模数/数模转换、显示接口等丰富的外设。这种将一个完整系统所需的所有设备集成在一块芯片上的体系结构被称为“片上系统”(System On Chip, SOC)。

参考 IT 公司的选型表，可以发现不同的单片机型号实际上就是不同的模块组合。当然，模块越多、容量越大的单片机价格越高。在单片机选型的时候，可根据实际项目的需要，选择最合适的型号。即使有些资源浪费不用，成本也比自己用分立芯片搭建系统要低得多！

MSP430 系列单片机详细的型号列表、资源列表和参考价格可以查阅每年的《MSP430 单片机选型手册》(登录 TI 官方网站 www.ti.com 下载)。

1.1.3 MSP430 单片机最小系统

在传统的微处理器系统中，要让系统运行，至少要提供电源、时钟和复位信号，而在 MSP430 单片机中，内部就带有复位电路(BOR)、片内数控时钟源(DCO)，因此只需外加电源即可构成可运行的最小系统，见图 1.1.2a。对于某些早期没有 BOR 模块的单片机(主要是 11x、12x、13x、14x 系列)，还必须提供复位逻辑电路，典型的方法是利用 IMP809-T 之类的复位/电源检测芯片提供复位信号，见图 1.1.2b。

如果在不含 BOR 模块的 MSP430 单片机系统中，也按照 1.1.2a 接线，省去复位芯片，则每次断电后必须等到电源电压降到接近 0V 才能再次上电，否则复位将是不可靠的。因为电源系统中的滤波电容难免有残压，如果电源开关快速断开-接通，将可能使系统进入错乱状态。

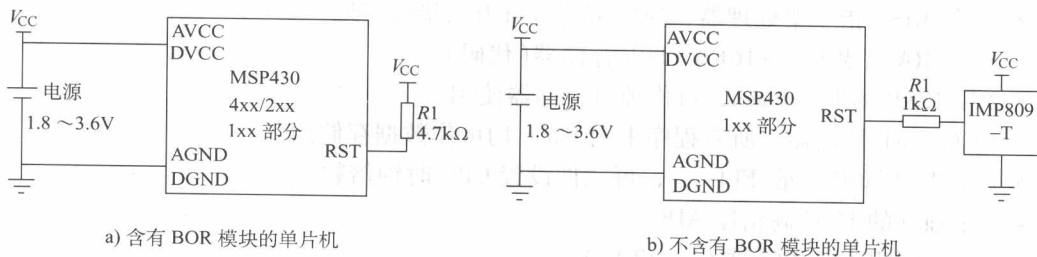


图 1.1.2 MSP430 单片机最小系统

在上述最小系统中，使用的是内部的数控振荡器(Digit Controlled Oscillator, DCO)，大部分 MSP430 单片机在复位后 DCO 输出频率被默认设定在 800kHz ~ 1MHz，可以为系统提供时钟源，并且随时可以通过软件改变 DCO 的振荡频率。但内部 DCO 的误差很大(20%)，且受温度影响严重。只适合为 CPU 运算提供时钟，或在对时间误差要求极其宽松的场合。对于需要较为精确定时的场合，如波特率产生、日历计时、精确定时、时间测量等应用中，必须提供外部晶体作为时钟源。MSP430 单片机通常使用 32.768kHz 的手表晶振作为外部时钟。

这个低频振荡一般向内部低速设备提供时钟，并作为定时唤醒 CPU 用，仅在 CPU 需要运算时才使用 DCO 提供的高速(但不准确的)时钟。

如果系统中不仅需要高速时钟，还要求高速时钟是准确的，就要使用高频晶振(大于 1MHz 的晶振称为高频晶振)。高频晶振有两种接法：第一种是直接替代 32.768kHz 晶振，并在软件中将晶体振荡器部分配置为高速模式(详见 2.3 节)；第二种是接在高速晶振专用的管脚上。采用第一种方法时，由于没有低速时钟，系统将无法低功耗休眠，采用第二种方法可以保留 32.768kHz 低速时钟，但并不是所有型号的单片机都带有高速晶振专用管脚(仅在 x3x 以上系列才有)。注意，无论高速晶振接在何处，都要自备 20~30pF 的匹配电容。各种晶振的接法见图 1.1.3。

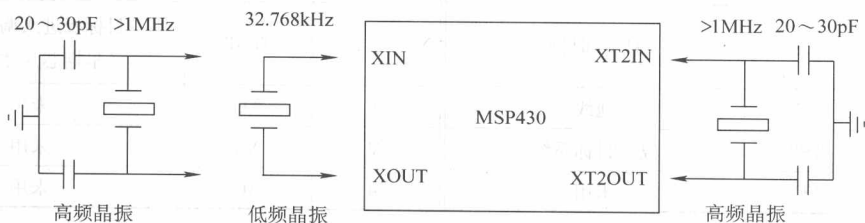


图 1.1.3 MSP430 单片机的外部时钟接法

这里特别要说明一下 MSP430F4xx 系列单片机内部带有锁频环电路，可以精确地对 32.768kHz 时钟进行倍频，无需高频晶振也能获得准确的高频时钟。另外，2xx 系列的部分单片机带有 VLO (Very Low Oscillator) 模块，可以在内部产生低频振荡，在不需要精确定时的应用中可以省略外部 32.768kHz 低频晶振，进一步降低成本。

最后，在单片机系统中还需要有调试接口，才能下载或调试程序代码。目前，MSP430 单片机有 3 种接口可供选用：JTAG 调试接口、SBW 调试接口和 BSL 接口。

JTAG 调试接口是成本最低的程序下载、仿真、调试的接口。全系列的 MSP430 单片机都具有 JTAG 接口。它需要 5~6 根线(4 根信号线,1 根复位线,早期单片机还需要 1 根 TEST 线)与计算机连接，可以实现代码的下载、仿真以及烧断保密熔丝的操作。

SBW 接口的功能与 JTAG 功能一样，但只需两根线(1 根信号线,1 根复位线)即可完成 JTAG 接口所有的功能。SBW (Spy-Bi-Wire, 双面间谍)这一有趣的名称因此得名。目前只有 2xx 系列和 5xx 系列的单片机带有 SBW 接口。

通过 JTAG 或者 SBW 接口不仅能够下载和调试程序，还能够在调试完毕后通过一定的指令烧断保密熔丝，使调试接口自毁。自毁后 JTAG 或 SBW 接口将失效，再也无法通过它读取内部代码，避免代码被他人读取或复制，从而保护知识产权。

对于烧断了熔丝的单片机，只能通过 BSL (Bootstrap Loader, 引导加载)接口来更新代码。除了最低端的几款型号之外 (F11x、F201x)，全系列的 MSP430 单片机都带有 BSL 接口。BSL 接口一般利用芯片上两根 IO 口(大多是 P1.1 和 P2.2, 具体参见相应芯片的文档)与计算机串口相连，可以对程序代码进行擦除、更新和校验。其中校验过程需要读取程序代码，为避免代码被非法读取，在读取之前必须和中断向量表核对，全部核对正确后才能读取代码区的内容。由于非法读取者不可能事先知道中断向量表的内容(各中断入口位置)，从而使得只有代码所有者才能读取代码进行校验。

在设计阶段和原理样机阶段，要对程序不断进行编写、修改、仿真和调试工作，只需要

JTAG 或 SBW 接口, 不进行烧熔丝操作。一旦产品定型, 在发布和量产阶段, 不再需要调试程序, 只需要烧写代码, 并且一定要烧毁熔丝, 要留 BSL 接口。为稳妥起见, 如果电路板面积足够, 仍然建议将 JTAG(或 SBW)调试接口和 BSL 接口都留出。

JTAG 调试接口标准要求采用双排 14 针连接器, 各管脚定义如表 1.1.1 所示。

表 1.1.1 JTAG 调试接口管脚定义

管脚	定义	功 能	管脚	定义	功 能
1	TDO	JTAG 数据出	2	VCCO	电源输出至目标板
3	TDI	JTAG 数据入	4	VCCI	目标板对 JTAG 供电
5	TMS	JTAG 模式选择	6	CLK	对目标板提供时钟
7	TCK	JTAG 时钟	8	TEST	目标板进入调试模式 (早期芯片才使用)
9	GND	地线	10	NC	未用
11	RST	复位目标系统	12	NC	未用
13	NC	未用	14	NC	未用

其中 TDI、TDO、TMS、TCK 这 4 根信号线要与单片机上对应功能的管脚连接, RST 信号要与单片机的复位管脚连接。早期的 11x 和 12xx 系列单片机还需要将 TSET 信号与芯片的 TSET 管脚相连。注意单片机如果采用外部复位电路, 外部复位电路的输出要串联一只 1k Ω 电阻再与单片机复位管脚连接, 以避免 JTAG 给出的复位信号与复位芯片给出的复位信号互相冲突(见图 1.1.2b)。

在开发过程的前期, 第一步通常可以只焊接最小系统相关的元件, 先将单片机运行起来, 再逐步焊接和调试各种外围模块。

1.2 MSP430 单片机开发软件入门

国内普及的 MSP430 开发软件种类不多, 主要有 IAR 公司的 Embedded Workbench for MSP430(以下简称 EW430)和 AQ430 两大类。目前 IAR 的用户居多, 本书所有的例程也以 IAR EW430 作为开发环境。IAR EW430 软件提供了工程管理、程序编辑、代码下载、调试等所有功能, 还提供了一个针对 430 处理器的编译器(ICC430 编译器)。整个开发过程所需的全部功能由一套软件全部提供, 这类软件被称为“集成开发环境”, 简称 IDE(Integrated Develop Environment)。而且 IAR Embedded Workbench 系列开发软件涵盖了目前大部分主流的 8/16/32 位处理器系统, 软件界面和操作方法保持不变。只要学会其中一种, 就可以很顺利地过渡到另一种新处理器的开发工作(如 IAR Embedded Workbench for ARM 等)。

读者可以登陆 IAR 的官方网站(www.iar.com)下载各种开发工具的试用版, 一般都提供两种试用版本: 4KB 限制版和 1 个月限制版。前者可以永久免费使用, 但编译生成的机器码不允许超出 4KB, 否则不予编译。后者无代码大小限制, 但只能试用 1 个月。

IAR EW430 安装运行后界面见图 1.2.1, 和大部分的开发工具一样, 它也采用了 VC 风格的界面。位于主窗体左侧的是工程管理窗, 负责工程内文件的添、删除、管理、浏览等功

能。右侧是编辑窗,在该窗口内编写程序。

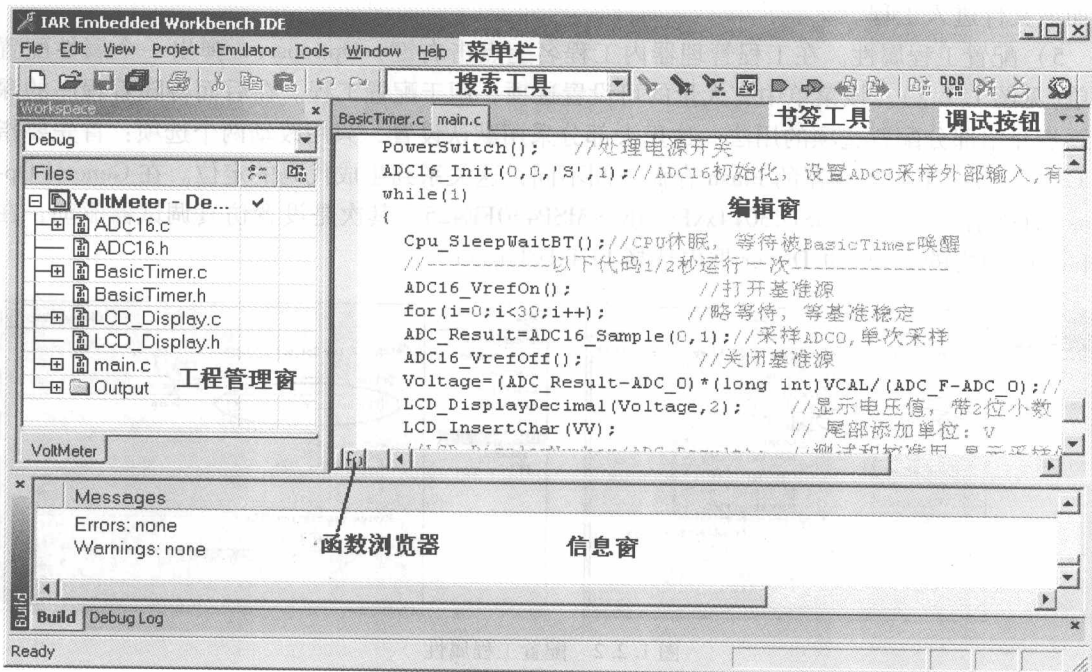


图 1.2.1 IAR EW430 编辑界面

位于主窗口下方的是信息窗。编译过程的相关信息、最终代码大小、RAM 开销情况、程序错误提示等内容会显示在该窗口内。另外软件界面中还有一些实用的小工具,如搜索和替换工具、书签工具、函数快捷浏览器等。

在 IAR 的软件中,不支持对单个 C 语言文件进行编译。所有的程序都必须在建立工程后才能编译。考虑到一个设计中可能包含多个 CPU 或多个版本的程序,EW430 中还引入了工作空间(Workspace)的概念。每个 Workspace 对应着一个设计,每个设计中允许添加若干个工程(Project),每个工程对应一个完整的程序。工程之下,才是若干个源文件。所以即使只写一个文件,也要先建立工作空间和工程。

参照下面的步骤,我们开始尝试为 MSP430 单片机写第一个程序:

- 1) 在 File 菜单里面选择 File-> New-> Workspace, 建立一个空工作空间。
- 2) 在 Project 菜单里面选择 Project-> Create New Project, 在当前空间建立一个新工程。新建过程中,首先会弹出语言选择窗,选择该工程的编程语言。EW430 支持 430 汇编、C 语言和 C++ 三种编程语言。本书的程序均采用 C 语言编写。选择 C-> main, 将自动生成 main.c 并添加进工程。如果不需要自动生成第一个文件,可以选择 empty project, 以后自行新建并添加文件。下一步将提示选择工程路径并输入工程名(例如 first.ewp)。并且最好新建一个文件夹,将工程保存在文件夹内以方便管理。
- 3) 在 File 菜单里选择 File-> Save Workspace, 提示选择路径并输入文件名(例如 first.eww), 保存当前工作空间。
- 4) 在工程管理窗内会看到新建的名为 first 的工程,若在第 2 步选择了“C-> main”,则在工程内已经包含了 main.c, 双击文件,在编辑窗打开后即可编辑。若在第 2 步选择了

“empty project”，则需新建一个文件，保存成 xxx.c 文件，在工程名上右键->Add files 手动添加该文件进入工程。

5) 配置工程属性。在工程管理器内工程名上点击右键->Options... 弹出一个属性配置菜单(见图 1.2.2)。该菜单中有大量的可设置选项，用于配置工程。在后面将随着程序的深入逐步介绍部分配置选项的用法。这里大部分采用默认设置，只用改动两个选项：首先要指定 CPU 型号，不同处理器的 Flash 存储空间不同，这关系到生成代码的定位。在 General Options 项内选择 Device->MSP430x4xxFamily->MSP430FE425。其次是设置仿真调试器驱动，在 Debugger 项内的 Setup 页 Driver 框内选择 FET Debugger。

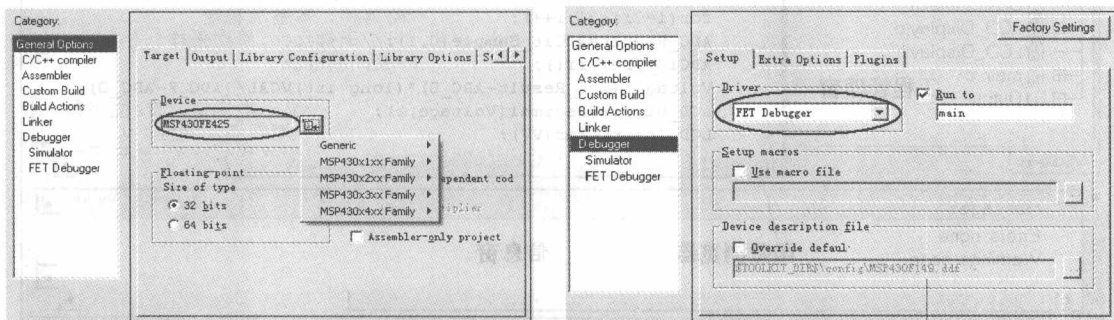


图 1.2.2 配置工程属性

若读者没有目标板，可以使用默认设置(Debugger 设为 Simulator)，即软件仿真模式，可以模拟运行代码，能调试纯软件的程序，如某些数据处理算法。但不能从实际硬件上反映出运行结果。

6) 连接 FET-Debugger，见图 1.2.3。“FET”的含义是“Flash 仿真工具”(Flash Emulation Tools)。在 MSP430 单片机内部，自带了下载程序的接口(JTAG 接口)。通过单片机的 4 根专用管脚与 PC 的并口(打印机接口)相连即可向单片机下载程序。除了下载程序外，通过该接口还能查看和更改 CPU 的各种寄存器与 RAM 内容、能够暂停 CPU 运行，从而能够实现仿真和调试功能。MSP430 单片机需要 3V 逻辑电平，PC 并口是 5V 逻辑电平，FET Debugger 实际上是一个电平转换器，在 PC 并口和 MSP430 单片机的 JTAG 接口之间起了数据桥梁作用。由于每次调试都要写单片机的 Flash 代码存储器，调试完毕单片机就可脱机工作，“Flash 仿真工具”因此得名。除此之外，FET Debugger 还能从计算机并口上窃取 10mA 左右的电流，提供给目标系统用。在大部分应用中，足够 MSP430 单片机系统使用，在调试过程中不需自备电源为目标板供电。

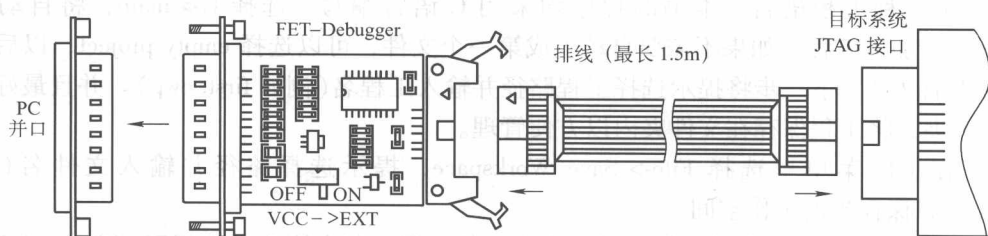


图 1.2.3 FET-Debugger 的连接

7) 在编辑窗口输入第一个 MSP430 单片机程序:

```
/* 该程序让 P2.0 口上的 LED 闪烁 */
#include "msp430x42x.h" /* 430 单片机寄存器头文件 */
void main(void)          // 主程序
{
    int i;
    WDTCTL = WDTPW + WDTHOLD; // 停止看门狗
    FLL_CTL0 |= XCAP18PF;     // 设置晶振匹配电容 18pF 左右
    P2DIR |= 0x01;           // P2.0 设为输出
    while(1)                 // 永远循环,单片机程序不能结束
    {
        for(i=0; i<20000; i++); // 延迟
        P2OUT ^= 0x01;         // P2.0 取反(闪烁)
    }
}
```

8) 在 Project 菜单点击 Rebuild All, 开始编译。随后信息窗将提示编译结果:

```
70 bytes of CODE memory
80 bytes of DATA memory (+4 absolute)
2 bytes of CONST memory
Total number of errors: 0
Total number of warnings: 0
```

上述信息说明, 代码编译后, 生成的机器码占用了 70B Flash ROM 空间, 80B 的 RAM。没有错误也没有警告。若提示出错, 则根据错误提示在编辑窗修改程序后重新编译。

编译通过之后, 开始在目标板上运行程序。首先要将编译生成的机器码下载到单片机的片内 Flash 存储器中。点击位于主界面上的“开始调试”按钮 , 弹出如图 1.2.4 所示的下载进程框。

若遇到“找不到设备”的提示, 应检查目标板电源是否正常、是否有其他程序占用了并口、在 BIOS 里将并口设为 ECP 方式、连接线是否有断线或短路? 若遇到“数据写入错误”的提示, 多半是电源电压不足 2.7V 造成的。Flash 存储器写入时要求电源电压在 2.7V 以上。若计算机没有并口, 可以使用 USB 仿真机, 但价格比并口 FET-Debugger 高。

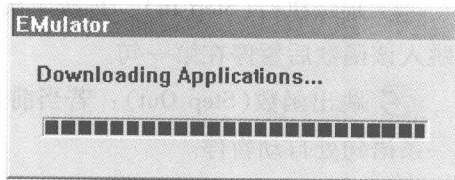


图 1.2.4 下载进程框

下载过程结束后, EW430 软件自动进入调试 (Debug) 状态, 界面也会随之改变 (见图 1.2.5)。

点击“”按钮全速执行, P2.0 口的 LED 开始闪烁。第一个程序运行成功!

然而实际中, 几乎不会有一次成功的程序。编译过程中只能检查出程序中的语法错误,

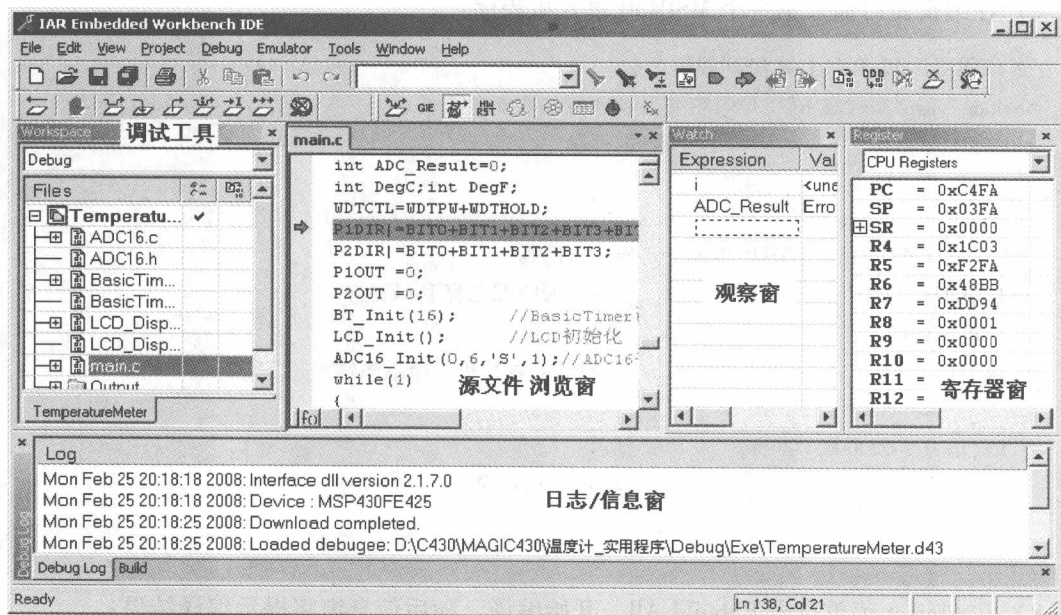


图 1.2.5 EW430 调试界面

不能检查出功能错误、逻辑错误、表达式写错、下标溢出等人为错误。因此在 EW430 进入调试界面后，主要功能不再是编写程序，而是检查和排除各种错误。

调试界面中提供了许多强大的调试工具，供调试、控制和跟踪程序流程：

 **全速执行 (Go)**：按该按钮后，程序全速执行。

 **暂停程序 (Break)**：只要程序处于运行状态，按该按钮可以强制暂停程序。

 **程序复位 (Reset)**：按该按钮可以将程序复位，并暂停在第一条句。

 **单步执行 (Step Over)**：每按一次该按钮，程序执行一条语句后暂停。如果遇到函数，则执行完该函数后暂停。

 **跟踪执行 (Step In)**：每按一次该按钮，程序执行一条语句后暂停。如果遇到函数，则跳入该函数后暂停在第一条句。

 **跳出函数 (Step Out)**：若当前程序处于某函数，按该按钮后，执行到函数返回前最后一条语句处自动暂停。

 **运行至光标处 (Run To)**：点击该按钮后，程序全速运行，直到运行到光标所在语句自动暂停。

 **断点 (Toggle Break Point)**：断点是一种常用调试手段，类似于陷阱。点击该按钮后，光标所在行会变红，提示该行已被设置成一个断点 (Break Point)。程序无论因为何种原因 (单步、全速) 运行到该行，都会自动被暂停。再按一次断点按钮，可取消该断点。

 **重新下载 (Make & Download)**：在发现错误之后，可以直接在编辑窗口修改源代码，然后点击该按钮，重新编译并自动下载。