MICROCHIP

MICROCHIP快讯

ISSUE 25 2017.03 第廿五期
MICROCHIP NEWSLETTER

公司动态

新的一年，Microchip首席运营官兼总裁Ganesh Moorthy先生向各大媒体分享了他对半导体行业于2016及2017年的看法及展望，当中亦道出了来年市场热点及挑战。以下是其中访问内容。

Microchip 首席运营官兼总裁
Ganesh Moorthy 先生

问：2016年，您对半导体以及电子产业的整体表现的看法如何？

答：半导体行业在2016年的整体表现平平，增长率几乎和2015年持平。不过，下半年的表现优于上半年。基于这点，我们乐观预测2017年将会超越2016年。

问：您如何评价Microchip在2016年的表现？

答：Microchip在2016年的综合表现统计尚未公布，但根据前九个月的实际情况和我们对最后一个季度设定的最新指导数据，Microchip今年的表现将会创造纪录。这有一部分原因归于我们今年四月份的收购，一部分则源于我们在业务上的有机增长所带来的市场份额增长。

问：在哪些产品线以及应用部分表现格外突出？为什么？

答：Microchip为涉及数千种应用的超十万客户提供服务，其中既有直接客户，也有通过全球渠道伙伴网络获得支持的间接客户。这使我们很难把经营情况和任何一项具体的应用直接挂钩。如果站在产品线的角度，我们近90%的收益来自于单片机和模拟产品组合；如果站在终端市场的角度，我们涉及的应用领域十分广泛，但相比之下，工业市场、汽车市场和物联网市场的表现更加强劲。

问：您如何评价2016年中国半导体市场的发展情况？请分享贵司在2017年对于中国市场的目标。

答：中国是Microchip非常重要的市场，我们在客户研发和生产集中的区域设立了18个分部，每位员工都在尽职尽业地为中国数千个客户提供服务。我们非常重视2017年在中国的业务发展，将会为现有和未来的客户提供具有创新性和竞争力的解决方案，帮助他们取得成功。与此同时，我们会继续在全国范围内举行技术研讨会和动手实验课程及开展大学计划项目，以提高客户自身的技术水平并培养未来的工程师。

问：Microchip对2017年半导体产业的整体走向怎么看？特别是：

答：
- 并购整合的行业趋势及其对Microchip和整个产业的影响
  2015年和2016年这两年，半导体行业的并购整合连创纪录。我们预计2017年半导体行业的并购仍将继续，这一趋势对于Microchip和整个行业都是有益的。竞争力弱的公司会被更强大更系统的公司收购或重组。
- Microchip和整个行业将会面临的机遇与挑战，以及您的应对计划
  产业整合不会影响我们在市场和产品线上的竞争力。我们一直努力保障持续的增长和盈利（截至2016年9月，Microchip已经保持104个季度的持续盈利）。我们始终致力于市场创新，为客户提供出色的技术支持，为客户的持续发展而不断提升我们的生产能力和保证无EoL的产品，并重视有助于缩减成本的工程建设。

问：您认为哪些产品、应用或技术将成为2017年的市场热点？Microchip计划如何抢占市场先机？

答：我们认为工业市场、汽车市场和物联网市场，在2017年会是较为热门的终端市场。我们已经准备好抓住这些市场和其他应用市场的发展机会。

MICROCHIP发布2017财年第三季度财报

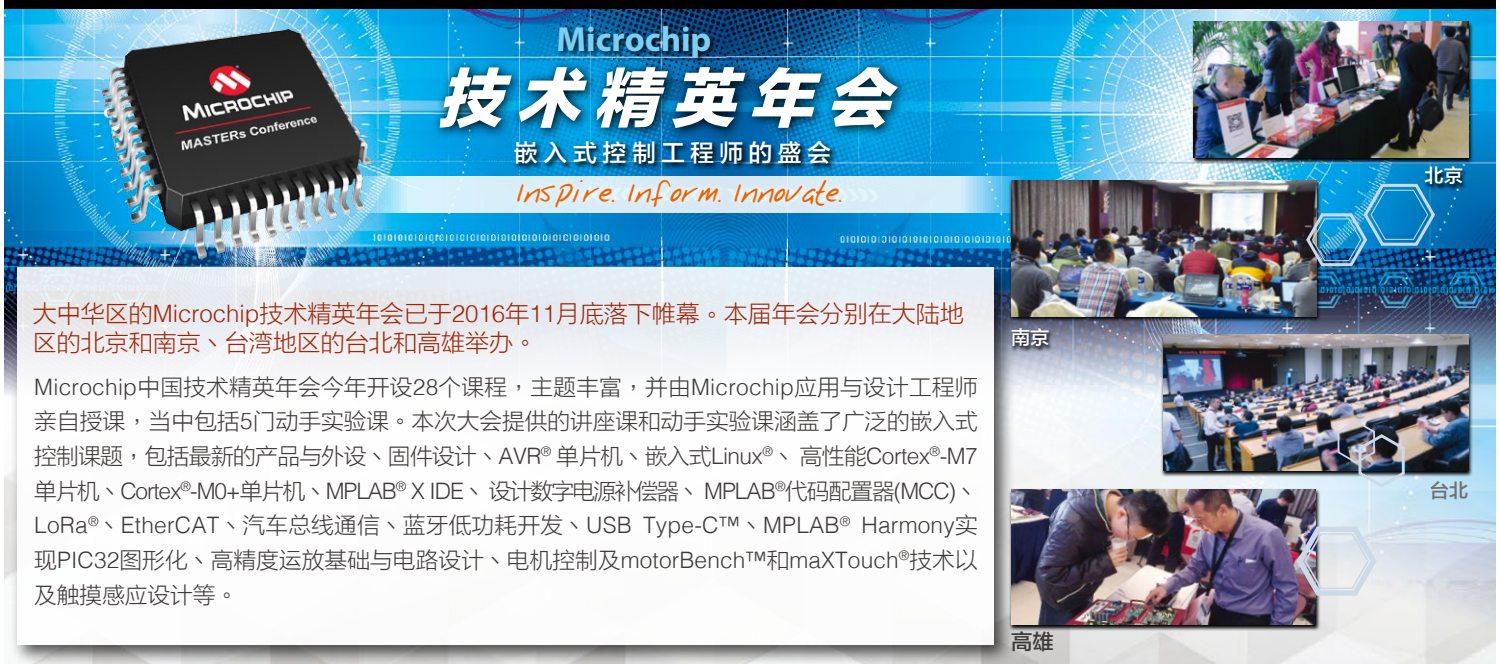
- **按照通用会计（GAAP）准则：**
  净销售额为8.344亿美元。由于收购Atmel，Microchip没有给出GAAP净销售额的预期值。
- **按照非通用会计（non-GAAP）准则：**
  净销售额为8.812亿美元，超出了本司于2016年11月29日给出的non-GAAP净销售预期值：8.389亿美元至8.651亿美元。
- **按照通用会计（GAAP）准则：**
  毛利率为55.8%；营业利润为1.181亿美元；持续经营的净利润为1.073亿美元，持续经营的摊薄后每股收益（EPS）为46美分。之前没有在First Call上对外发布关于GAAP EPS的估计值和预期值。
- **按照非通用会计（non-GAAP）准则：**
  毛利率为57.8%；营业利润创纪录，为2.891亿美元；持续经营的净利润创纪录，为2.465亿美元；持续经营的摊薄后每股收益创纪录，为1.05美元。本司于2016年11月29日更新的EPS预期值为87美分至94美分。
- **经营活动的现金流创纪录，为2.908亿美元。**
- **季度股息创纪录，为每股36.10美分。**

总结

Microchip首席执行官Steve Sanghi先生总结到：

“预计2017年3月份这个季度，我们的净销售总额将达到8.72亿至9.08亿美元。按照非通用会计准则（non-GAAP），在12月份的这个季度，我们的营业利率已达到32.8%。这一数字非常接近我们的长期目标——33%。目前，我们正在将原有的长期non-GAAP经营模式（59%的毛利率、26%的运营成本及33%的营业利率）调整为60%的毛利率、24%的运营成本及36%的营业利率。Microchip不采用 GAAP长期经营模式。”

活动花絮



大中华区的Microchip技术精英年会已于2016年11月底落下帷幕。本届年会分别在大陆地区的北京和南京、台湾地区的台北和高雄举办。

Microchip中国技术精英年会今年开设28个课程，主题丰富，并由Microchip应用与设计工程师亲自授课，当中包括5门动手实验课。本次大会提供的讲座课和动手实验课涵盖了广泛的嵌入式控制课题，包括最新的产品与外设、固件设计、AVR® 单片机、嵌入式Linux®、 高性能Cortex®-M7单片机、Cortex®-M0+单片机、MPLAB® X IDE、 设计数字电源补偿器、 MPLAB®代码配置器(MCC)、LoRa®、EtherCAT、汽车总线通信、蓝牙低功耗开发、USB Type-C™、MPLAB® Harmony实现PIC32图形化、高精度运放基础与电路设计、电机控制及motorBench™和maXTouch®技术以及触摸感应设计等。

另外，现场亦举行开发工具特价销售及赞助伙伴展览，吸引了工程师们纷纷驻足询问并购买。亚太区的Microchip技术精英年会亦同期于印度的班加罗尔及韩国的首尔举行。错过本次盛会的工程师，可以访问Microchip网站，留意下次研讨会的举办日期！

Microchip 嵌入式解决方案研讨会 2016 (秋)

Microchip嵌入式解决方案研讨会2016（ESS）于11月在台湾4个地点（台北、新竹、台中及高雄）举办。

此次研讨会专注于Microchip PIC®、AVR®及SAM微控制器、蓝牙连接及开发工具等。研讨会内更送出双MCU开发板及精美礼品，与会者均表示获益良多！

2016 物联网开发者大会

2016年12月9日（周五），2016物联网开发者大会（IoT Developers Conference 2016）已于北京北辰洲际酒店顺利举行！

Microchip是此次大会的黄金级赞助商。应用工程师薛祖旭（Tom Xue）在会上演讲《物联网连接技术和非接触式安全解决方案》，吸引了众多观众，座无虚席！场外更设有展位展示Microchip最新的物联网连接技术解决方案，超过1,000人参加了这一大会。

活动亮点

2017 中国（重庆）国际汽车技术展览会

2017中国（重庆）国际汽车技术展览会（China (Chongqing) International Automotive Electronics Technology Expo 2017）将于3月22-24日（周三至周五）举行。

今年是Microchip首次参加，展位位于展馆N5摊位T22-A。诚挚邀请您前来参观。我们将会展出汽车电子最前沿的解决方案，为您的客戶在汽车领域取得成功助力。展览期间还有许多汽车技术性研讨会和测试展。费用全免，欢迎参观。

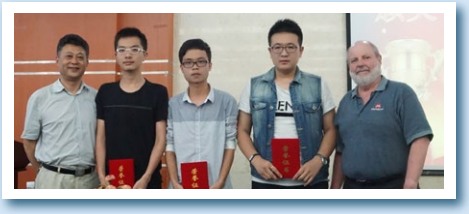
电子科技大学
Microchip公司2016奖学金颁奖仪式暨校园专题技术讲座

2016年下旬，第六届“微芯杯”电子设计大赛暨Microchip 2016中国大学奖学金颁奖典礼在电子科技大学顺利举行。

颁奖仪式由航空航天学院副院长李辉教授致开幕辞，并代表电子科技大学对Microchip公司的到来表示热烈的欢迎和衷心的感谢。一直以来Microchip公司和电子科技大学航空航天学院保持着密切的合作，提高学生动手能力、增长学生知识水平，共同进行着不懈的努力，希望双方之后的合作更加紧密，携手为祖国下一代的教育事业尽心尽力。

航空航天学院副院长李辉教授（左）和 Microchip 全球销售总监 Mr Ron Cates 先生（右）为微芯奖学金获奖同学颁奖

颁奖仪式结束后，Microchip公司资深应用工程师高伟在会上演讲了《未来的IoT世界将连接一切》的技术讲座。介绍了未来技术的发展方向以及Microchip公司的前沿技术、涉及领域。



航空航天学院副院长李辉教授（左）和 Microchip 全球销售总监 Mr Ron Cates 先生（右）为第六届“微芯杯”电子设计大赛 - 特等奖获奖同学颁奖

最新文档

宣传资料

- tinyAVR®器件宣传页
- 信号链解决方案宣传页
- 温度管理解决方案宣传页

数据手册

- PIC16F527数据手册
- PIC16(L)F18324/18344数据手册
- PIC16F570数据手册
- PIC16LF1566/1567数据手册
- dsPIC33EPXXGS202数据手册
- 16位单片机外设快速参考指南
- AWS IoT支持的IoT以太网工具包
- USB57XX系列宣传页
- PIC24FJ1024GA610/GB610系列数据手册
- 具有浮点单元的嵌入式连接（EF）PIC32MZ系列数据手册
- MIC2125/6数据手册
- UCS1003-1/2/3数据手册

用户指南

- MIC2125/6评估板用户指南
- MIC45404评估板用户指南
- MCP39F511N功率监视器演示板用户指南
- LoRa®技术评估套件用户指南
- MCP9902温度传感器评估板用户指南

参考手册

- 16位MCU和DSC程序员参考手册
- dsPIC33E/PIC24E FRM - 运放/比较器
- dsPIC33E/PIC24E FRM - 输出比较
- dsPIC33E/PIC24E FRM - 增强型控制器局域网（ECAN™）
- PIC32 FRM - 第 42 章 带有增强型PLL的振荡器
- PIC32 FRM - 第 49 章 加密引擎和随机数发生器（RNG）

应用笔记

- AN1536 - 超声波测距
- AN2007 - 100BASE-FX光纤介质支持
- AN2045 - 串行EEPROM与8位PIC®单片机的接口设计
- AN2049 - 单相BLDC电机驱动器
- AN2059 - LIN基础知识和8位PIC®单片机上实现的MCC LIN协议栈库
- AN2095 - 使用PIC®单片机实现远程无钥门禁和防盗锁止系统的晶体管线圈点火装置

最新视频

视频：MCP9902远程温度传感器

- MPLAB® XC编译器简介
- MPLAB® XC编译器许可证介绍
- MPLAB® XC编译器的下载与许可
- 如何实现快速循环
- PAC1921上桥臂功率/电流传感器
- LAN9252 EtherCAT®从控制器
- ADC ASPECTS 1 - 量化
- ADC ASPECTS 2 - 线性度（DNL和INL）
- TimeFlash MEMS振荡器在线编程工具包
- dsPIC® DSC磁卡读卡器演示
- 便携式气象站演示

防水功能强大的外设触摸控制器令Microchip MCU如虎添翼

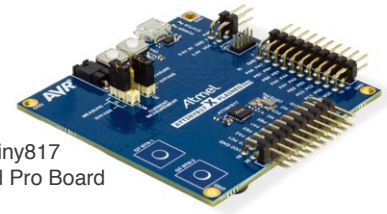
触摸外设取得新进展，首获START代码配置器支持

Microchip近日为其最新发布的ATtiny817/816/814单片机系列器件添加了一个新的增强型外设触摸控制器（PTC）。新的PTC器件是一款尺寸小巧、性价比高的独立于内核的外设（CIP），可在标准单片机上实现高性能的电容式触摸传感功能。该PTC具备出色的耐水性，用户可以通过START代码配置器对其进行配置，使用起来简单方便。

CIP是一种被设计为可独立处理任务、无需代码或CPU监管即可自行维持运行的器件。作为一款CIP器件，外设触摸控制器简化了触摸传感功能的实现，为设计人员提供了相当的灵活性，使其可以专注于应用的其它方面。

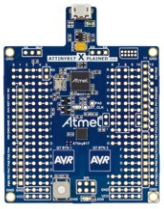
ATtiny817 QTouch® Water Demonstration Board

使用PTC器件的触摸解决方案结合了先进的噪声处理、耐水性触摸功能和低功耗触摸唤醒操作。IEC 61000-4-6的传导抗扰度为15Vrms，这使得客户能够轻松通过最严格的EMC标准，尤其是在家电和汽车市场。强大的耐水触摸功能使得产品得以在户外使用，极大地提升了用户体验。此外，它能在功耗更低的情况下实现先进的休眠和唤醒功能，是可穿戴应用和其它电池供电应用的理想选择。



ATtiny817 Xplained Pro Board

Microchip生态系统提供的软件支持使得易用性得到进一步提升。而ATtiny817是产品线中第一款获得基于web的工具链START代码配置器及QTouch®代码紧凑型模块化固件库支持的产品。作为一款基于web的工具，START可以让用户始终受益于最新、最先进的触摸库。



ATtiny817 Xplained Board

“Microchip仍将是触摸技术领域的领导者”，Microchip触摸传感部副总裁Fanie Duvenhage表示：“在打造这款最新外设触摸控制器的过程中，我们在继续为设计人员提供应用实现的简便性时还寻求在防水性能和抗噪性能上的突破。这种努力换来的便是一款性能一流的CIP器件，它对ATtiny817/816/814系列、并且在未来对多数MCU来说都可谓是一款锦上添花的器件。”

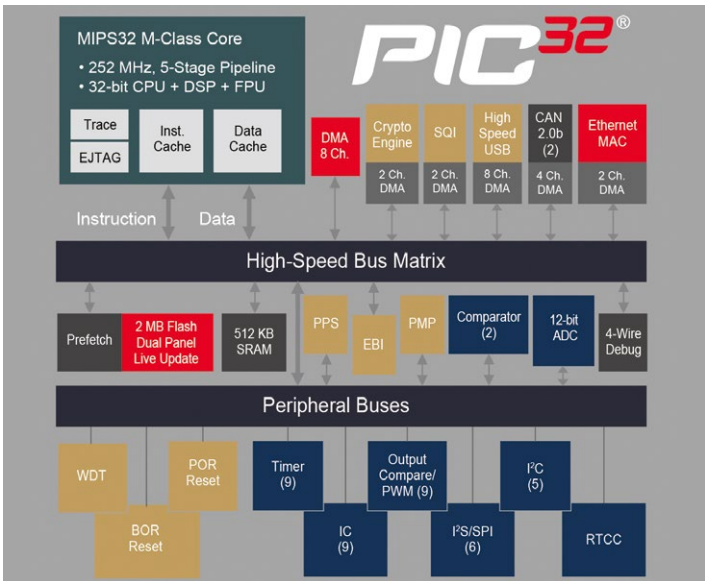
欲了解更多有关Microchip外设触摸控制器的信息，请访问：
www.atmel.com/products/microcontrollers/avr/tinyAVR.aspx

扫一扫 访问更多信息

Microchip 32位PIC32MZ EF单片机系列喜添新成员，部分器件可支持扩展级温度范围

PIC32MZ EF系列是Microchip首个符合汽车电子委员会制定的AEC-Q100一级（-40°C至125°C）规范的PIC32 MCU产品系列

Microchip近日扩展了旗下32位PIC32MZ EF单片机（MCU）系列，增添了支持扩展级温度范围的产品和支持工业级温度范围的高速（250 MHz）产品。新器件为Microchip推出的首个汽车级PIC32 MCU产品系列。



PIC32MZ EF系列新器件拥有频率高达250 MHz的高性能内核、一个集成浮点运算单元（FPU）、丰富的外设以及包含控制器局域网（CAN）在内的各种功能卓越的连接选项。支持扩展级温度范围的新产品组合特别适用于汽车和工业领域的几种关键应用，因为这些应用都需要确保产品在更高温度条件下的可靠性能和稳定性。

此外，Microchip还选择扩充了我们支持工业级温度范围（-40°C至85°C）的部分产品组合，以提供更高的处理速度（250 MHz/795 Core-Mark）。该系列高速器件适用于诸如高清音频等应用，这些应用往往需要稍高一些的内核性能以实现更快速的音频处理和传输。

“自从PIC32MZ EF系列产品问世以来，采用该产品的客户群一直保持着强劲的增长势头”，Microchip MCU32产品部副总裁Rod Drake表示：“此次推出PIC32系列首个支持扩展级温度范围的产品组合并取得AEC-Q100 一级标准认证，可谓是Microchip产品发展蓝图中的一个重大里程碑。新推出的高速器件大幅提升了性能，可满足处理密集型应用的需求，这是该系列产品令人振奋的一大进步。”

支持扩展级温度范围的汽车级PIC32MZ EF系列器件现已开始供货。
欲了解更多相关信息，请访问
www.microchip.com/PIC32MZEF_Main1946

扫一扫 访问更多信息

设计分享

让触摸界面风“雨”无阻

全新ATtiny系列单片机采用具有超强防水性能的触摸界面

最近发布的全新tinyAVR®系列8位单片机（MCU）是首款采用独立于内核的外设（CIP）的tinyAVR单片机，它不仅仅在Microchip的历史上占据了里程碑地位，同时对各触摸应用研发商来说也是突破性的进步。这得益于ATtiny817/816/814系列器件采用体积小且具成本效益的CIP，即外设触摸控制器（PTC），使得从数据采集到过采样和阈值对比都能通过高性能电容式触摸实现。通过这种硬件技术改进，可以将宝贵的MCU资源用于执行其他系统任务，同时提供一流的触摸性能、卓越的降噪特性，并实现低功耗运行。Microchip赋予PTC的一大优势是支持防水触控。随着越来越多的产品和应用采用触摸界面，用户的期望也随之提高。用户希望自己的触摸界面在各种条件下都可以运作良好，不需要洗净或者擦干手就可以使用，不会因为戴着手套或者表面有水就无法操作。但是，由于水会使电容式触摸电路发生短路，所以触摸界面通常不适合在有水珠、雨水、雾水、汗水或者溢水的环境下使用。

为此，PTC会发出保护信号（驱动屏蔽层），让MCU不受水造成触摸短路的影响。全新tinyAVR系列MCU可提供这种基于硬件的防水触摸功能，而且无需MCU内核介入。与**PIC16(L)F156x**系列、**PIC16(L)F188xx**系列和**PIC18(L)FxxK40**系列的8位MCU一样，此系列MCU可同时驱动多个防水护罩，因此可实现户外应用的防水键盘、甚至是2D触控板。除了防水性能，全新tinyAVR系列中的PTC结合运用PTC模块化触摸库，还能满足严苛的EMC标准以及家用电器和汽车应用的最新要求。其IEC/EN 61000-4-6防传导额定值为15Vrms。此外，触摸唤醒操作的功耗也非常低，约为6µA。这是目前为止体积最小（3×3mm 20引脚QFN）且成本效益最高的AVR®器件，它的出现推动了触摸界面向创新和专业化发展。但它的意义远不止如此。这些新器件的高性能以及更轻松的MCU配置方式，均获得Microchip软件生态系统的支持。ATtiny817/816/814是首批受Atmel START支持的8位产品。**Atmel START**是一款可帮助设计人员配置软件组件的在线工具。因此通过这款基于网页的工具，您可以获得最新、最前沿的触摸库资源。模块化的固件**QTouch® Library**允许您只在设计中集成所需的触摸功能，内存占用量大大减少。集成的**QTouch Composer**开发平台是一套完整的电容式按钮、滚动条和滚轮的开发和调整工具，它提供基于向导的设计流程，直观地引导您完成整个设计过程。虽然全新系列tinyAVR MCU的硬件和软件工具都推出了重要更新，但仍保留自电容和互电容感应特性，并支持1.8V至5.5V的电源电压范围。Microchip拥有几十年的触摸技术经验，数百万成熟的解决方案已成功部署于众多不同市场，能够帮助您在新设计中添加防水触摸界面。我们还能够帮助您充分发挥全新tinyAVR系列MCU中PTC模块的优势。如果您希望使用PIC® MCU，我们可为您提供带有电容分压器（CVD）或具有计算功能的（ADC2）先进ADC的器件。这些器件均受MPLAB® X集成开发环境和MPLAB代码配置器支持。

有关各种解决方案的信息，请访问我们的**触摸和输入传感设计中心**或联系**当地的Microchip销售办事处**。

新的起点

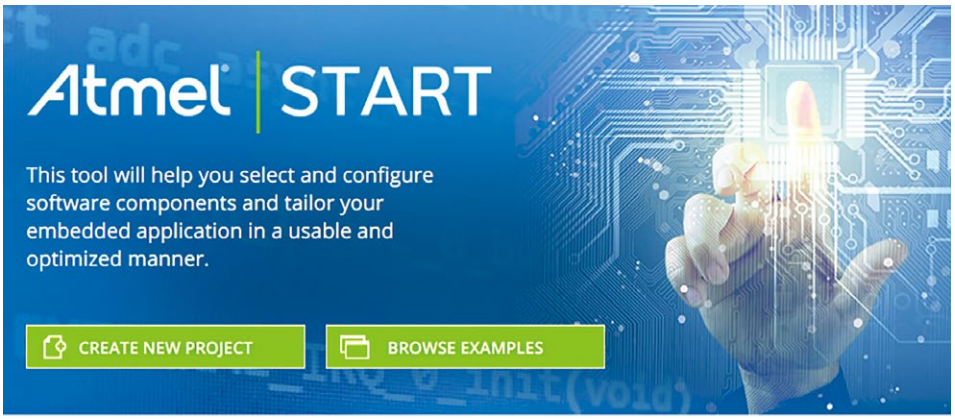
基于网页的Atmel START支持AVR® 单片机

Atmel START

是一款基于网页的免费软件配置和发布工具。它最近的更新包含了对某些8位AVR®单片机的基本支持。其主要功能包括引脚配置，可查看时钟和预分频器的配置，它也是最新发布的ATtiny817/816/814/417器件中事件系统的图形化配置工具。通过项目面板可以添加和配置所有外设。由于全新的tinyAVR®器件配备外设触摸控制器（PTC），因此通过QTouch®器件配置工具可以简化在项目中添加触摸按钮的操作。

简介

Atmel START可以从浏览器直接启动，便于您进行AVR和SAM MCU开发工作。在主页中，您可以选择创建新项目，或者先浏览示例项目，帮助您尽快开始设计。这些示例代码与应用笔记和参考设计中的相似，可以直接应用，也可以根据自己的要求轻松修改。如果选择“Create New Project”（新建项目）选项，会转到代码生成区。通过图形用户界面，您可以为自己的MCU配置多种特性和外设，并导出到所需的C编译器。如果需要，也可将代码导入**Atmel Studio**。无论您何时想配置其他外设，您都可以利用Atmel Studio随时重新配置自己的Atmel START项目。代码生成器也含有中间件，但主要是针对SAM MCU。



Atmel START 主页

Atmel START发展迅速，不断推陈出新。目前已提供一些教程视频，未来还将推出更多教程。

扫一扫 观看教程视频

https://goo.gl/n198me

专家意见↘

MPLAB® Harmony之学习篇（三）——— 创建一个Harmony项目

李建平 — Microchip MCU32产品部应用工程师

本文用于指导用户建立一个基于 Harmony 的工程项目，通过对使用 MHC 配置并产生代码，加深对 Harmony，MHC 的认识。
有关 Harmony 和 MHC 的介绍，请参考系列学习篇之《Harmony 设计理念和优点》和《了解 MHC》。

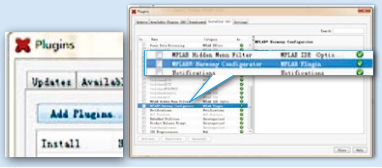
（一）软件工具

A. 集成开发平台MPLAB® X
下载地址：www.microchip.com/mplab/mplab-x-ide

B. 编译器：XC32
下载地址：www.microchip.com/mplab/compilers

C. 软件库：Harmony
下载地址：www.microchip.com/mplab/mplab-harmony

D. MHC
MHC是Harmony自带的集成配置工具，如果你已经安装了MPLAB X和Harmony，你可以在MPLAB X > Tools > Plugins找到它。



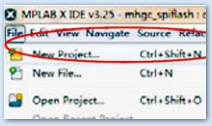
如果它的版本不是最新的版本，你可以通过Plugins > Updates自动更新，也可以通过Plugins > Downloaded > Add Plugins手动安装，MHC安装文件的放在Harmony目录下：C:\microchip\harmony\v1_07\utilities\mhc

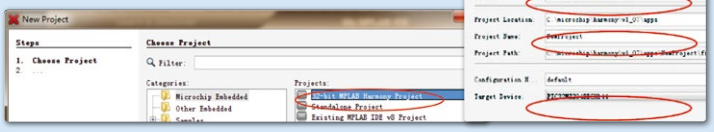
（二）创建项目 Project 步骤

步骤1. 打开MPLAB X。

“开始>MPLAB X IDE Vx.xx”或者
“开始> Microchip > MPLAB X IDE > MPLAB X IDE Vx.xx”

步骤2. 创建一个基于Harmony的工程项目。





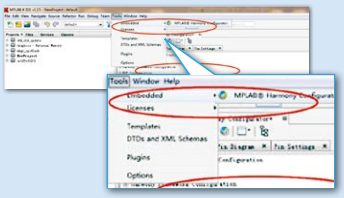
（三）通过 MHC 配置 Project

MHC 是一个集成的代码配置工具，它以简易，方便的勾选配置代替了繁复的代码编写工作，当然合理的配置步骤将使你收到事半功倍的效果。

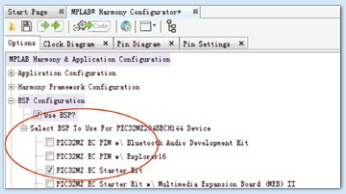
我们推荐的配置步骤是：

选择 BSP Configuration（可选）> 配置 Clock > 配置其他 Configuration Bits > 定义引脚 > 配置 Third Part Libraries（可选）> 根据项目需要选择 Drivers > 根据项目需要选择 System Services > 选择需要的 Libraries 和 Stack。
下面以一个工程实例介绍一下 MHC 配置的过程。

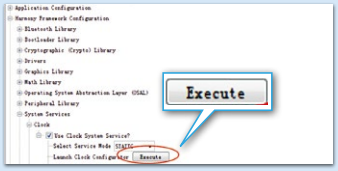
步骤1.打开MHC配置工具（在新建项目以后MHC是默认打开的）

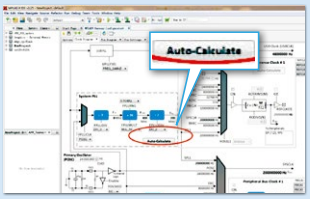


步骤2.如果用到了Microchip官方评估板，可以直接选择BSP。

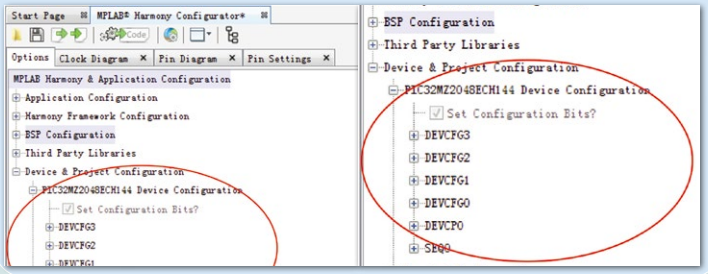


步骤3. 根据项目要求配置系统时钟，“Auto-Calculate”功能可以自动计算各种时钟。

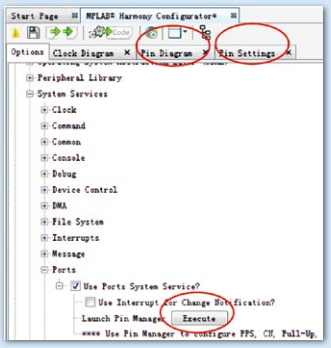




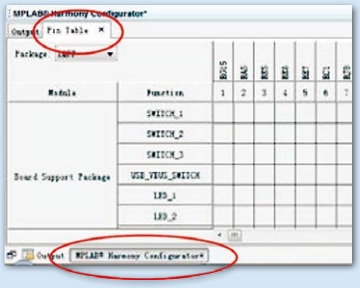
步骤4. 选择其他的Configuration Bits。



步骤5. 根据需要定义引脚。



步骤6. 选择引脚PPS。



步骤7. 根据需要增加其他的Drivers，System Services，Libraries等等。

（四）产生代码

（五）参考代码

档案名称：NewProject.zip

至此，你已经成功建立了一个基于 Harmony 的项目工程！

问与答

问：由MHC产生的代码可以直接编译运行吗？

答：是，MHC 生成的代码是一个完整的工程，可以直接编译运行，但是它是一个没有应用代码的空循环。

问：由MHC生成的Project是否必须放在Harmony文件夹下？

答：否，只需要在 MHC 中选择 Harmony 路径就可以建立 Project 与 Harmony 的联系了，不需要放在 Harmony 的文件夹下，不过 Harmony 自带的例程都是放在 Harmony 文件夹下的。

问：在MHC中可以通过System Service自动配置系统时钟，也可以通过Device & Project Configuration中的熔丝位选择来配置系统时钟，这两者的效果是一样的吗？

答：是，这两者都是实现了对熔丝位的配置，但是 System Service 配置系统时钟更加直观。

MPLAB® Harmony之学习篇（四）——添加自己的应用程序

李建平 — Microchip MCU32产品部应用工程师

在《如何创建自己的 Harmony 工程》文章中，我们介绍了怎么创建一个基于 Harmony 的工程项目，当然，新建的这个工程是没有实际的应用功能的。

本文通过一个工程实例，介绍一下如何基于已经建立好的 Harmony 工程项目添加自己的应用程序。
通过本文，用户能够在 Harmony 架构下添加自己的应用代码，同时也理解了如何应用 Harmony 里的驱动等资源实现自己的功能。

在添加应用代码的时候我们首先要明白两个基本原则：

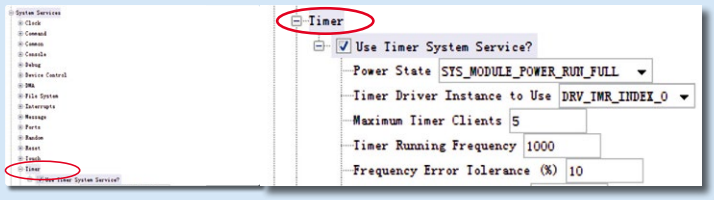
第一：添加应用代码的时候尽量在 app.c 或者新建应用文件，不要在 Harmony 自动产生的文件中添加，以保证 Harmony 模块化的分层结构。
第二：Harmony 驱动层的代码都是基于状态机的机制，所以在添加应用代码时，我们同样建议使用状态机。

（一）示例项目功能介绍

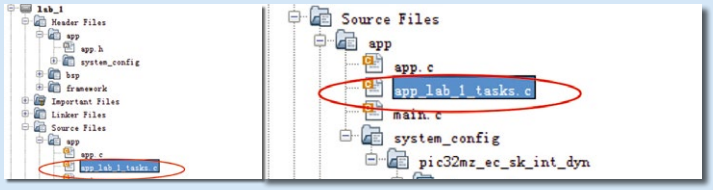
本文希望添加用户代码，实现一个 LED 的 1Hz 闪烁。

（二）添加用户代码的步骤

步骤 1. 在System Services添加应用程序需要用到的服务：Timer



步骤 2 增加app_xx_task.c函数（根据实际需要的任务数量添加）。



步骤 3. 在app_xx_task.c增加对任务状态机的定义。

```
typedef enum
{
/* Try to start the delay in this state */
APP_LAB1_TASK_STATE_START_DELAY = 0,
/* Wait for the delay to complete */
3
APP_LAB1_TASK_STATE_WAIT_DELAY
}APP_LAB1_TASK_STATE;
static APP_LAB1_TASK_STATE appLab1TaskState;
```

注意状态机的设计要简洁合理，需要将程序设计成合理的时间块，本例程中的任务主要是等待延时，当延时到达的时候就将LED取反输出，并重新计时。所以设计了左面的状态机：

步骤 4. 在 app_xx_task.c 中增加对任务调用资源的定义。

```
static SYS_TMR_HANDLE sysTMRHandle;
```

步骤 5. 在 app_xx_task.c 中增加任务的初始化。

```
void APP_LAB1_Initialize(void)
{
/* Set the initial state of the task */
APP_LAB1_TASK_STATE appLab1TaskState = APP_LAB1_TASK_STATE_START_DELAY;
/* Set the Timer System Service handle to something known */
sysTMRHandle = SYS_TMR_HANDLE_INVALID;
}
```

步骤 6. 在 app_xx_task.c 中增加任务的实现函数。

```
void APP_LAB1_Tasks(void)
{
switch(appLab1TaskState)
{
case APP_LAB1_TASK_STATE_START_DELAY:
{
/* In this state try to start the delay. If the delay could not
* be started, then try again. */
/* Step 1*/
sysTMRHandle = SYS_TMR_DelayMS(1000);
if(sysTMRHandle != SYS_TMR_HANDLE_INVALID)
{
/* This means the delay has started. Let's change the
state to
* waiting for delay to complete. */
appLab1TaskState = APP_LAB1_TASK_STATE_
WAIT_DELAY;
}
break;
}
case APP_LAB1_TASK_STATE_WAIT_DELAY:
/* Step 2 */
if(SYS_TMR_DelayStatusGet(sysTMRHandle)
== true)
{
/* This means the delay has completed. Lets
toggle the LED */
BSP_LEDToggle(BSP_LED_1);
/* Now let' s start the delay again */
appLab1TaskState = APP_LAB1_TASK_
STATE_START_DELAY;
}
default:
break;
}
}
```

步骤 7. 在 app.h 中增加应用程序的状态机。

```
typedef enum
{
/* Application's state machine's initial state. */
APP_STATE_INIT=0,
/* TODO: Define states used by the application state machine. */
APP_STATE_RUNNING
} APP_STATES;
```

步骤 8. app.h 中申明客户的任务初始化和实现函数。

```
void APP_LAB1_Initialize(void);
void APP_LAB1_Tasks(void);
```

步骤 9. 在 app.c 中申明并调用任务的初始化和实现函数。

```
void APP_Initialize ( void )
{
/* Place the App state machine in its initial state. */
appData.state = APP_STATE_INIT;
/* TODO: Initialize your application's state machine
and other
* parameters.
*/
APP_LAB1_Initialize();
}
void APP_Tasks ( void )
{
/* Check the application's current state. */
switch ( appData.state )
{
/* Application's initial state. */
case APP_STATE_INIT:
{
appData.state = APP_STATE_RUNNING;
break;
}
case APP_STATE_RUNNING:
{
/* Call the Application Tasks */
APP_LAB1_Tasks();
}
/* TODO: implement your application state
machine.*/
/* The default state should never be
executed. */
default:
{
/* TODO: Handle error in application's state
machine. */
6
break;
}
}
}
```

（三）参考程序

档案名称：lab_1.zip

问与答

问：在Harmony里面添加自己的用户代码时要注意什么？

答：第一，要将代码添加在应用层，尽量不要改变MHC生成的代码，最好根据每个任务新建app_xx.c的文件；

第二，根据自己的应用层任务合理设计状态机。

问：在Harmony架构下，如果驱动层需要用到应用层的资源的时候如何处理？

答：可以增加回调函数，实现驱动层对应用层资源的调用。

问：在Harmony架构下，用户自己添加的代码如何嵌到MHC生成的代码中运行？

答：在 void APP_Initialize (void) 函数中调用任务的初始化函数，在 void APP_Tasks (void) 函数中调用任务的 Task 函数就可以运行用户的代码了，就是这么简单！