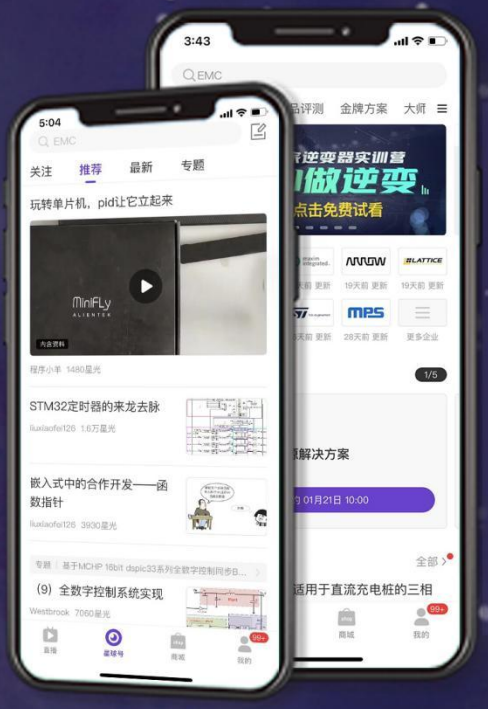
电源网 VIP 技术专刊

2022年度
精选内容

目录 Contents

电子星球

电子星球App是什么？



电子星球是电源网旗下电源电子行业APP。旨在打造一所属于电子工程师的终身学习的大学，成就百万电子人。通过线上线下结合的学习方式，把分散在各个地方的行业专家、民间达人的知识集合在一起，为工程师提供“省时间的高效知识服务”。

星球号是什么？

星球号是电源电子行业第一个付费自媒体平台。集结电子行业名企专家、顶尖大学教授，技术大牛与实战高手，旨在为读者呈现原创优质技术解读与实战分享。希望为读者提升专业能力，为创作者提升价值。加入我们成为星球号作者，你将获得稳定的收入，优先合作成为线下会议嘉宾、直播讲师、社群群主等，以及电源网旗下平台优先推广，提升个人品牌影响力。

联系微信：dianyuanqinghuai

APP下载

星球号

工程师眼中的「电源网论坛」

电源网论坛，中国权威的电源电子论坛（www.dianyuan.com/bbs/）

论坛是以电源电子技术聚合志同道合者的互动平台，网友以发帖回帖的形式来进行技术交流。

经过多年发展，论坛汇集了大批电源电子技术人才/名企高工/行业专家/技术大牛/实战高手，每日更新佳作层出不穷。其内容覆盖电源电子技术信息以及各大行业领域的应用，包括电源/工业/汽车/消费/医疗/通信等。

丰富高质的技术内容和精彩不断的定期活动，使得论坛得到了广大网友的一致认同，成为国内最专业的工程师首选平台。

扫码进入电源网论坛首页

「鼓励原创」论坛原创文章现金奖励计划

【现金奖励】

1. 技术原创帖一经发布，即可奖励50元，点击数达到1000并且有效回复数达到40后，作者可再奖励300元。
2. 发布50个及以上回复帖，奖励20元。

【发帖要求】

1. 帖子必须为原创，仅限原创，软文枪文，敬请绕道。
2. 发帖后请联系管理员（微信：18522870362）

【可以给您带来什么】

1. 利--发帖现金奖励，敲敲键盘成为任性土豪。
2. 名--我们会有专门的团队进行推广策划，推广原创内容的同时，也会协助打造个人品牌，真正的实现名利双收。

声明：最终解释权归电源网所有！

扫码查看鼓励原创计划详情

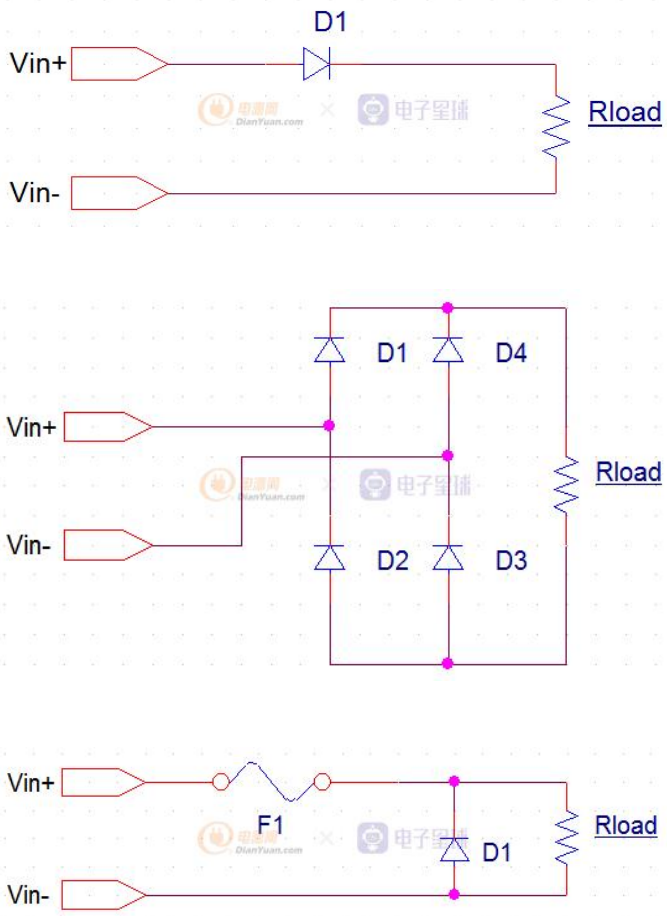
扫码添加论坛管理员

保护电路之一：DC 输入电源防反接电路设计

文 / 勤劳善良的人

为防止电源使用者有意无意将输入引线接错导致电源烧坏，一般的 DC 电源会在其输入端加防反接电路，常见方式为串联二极管（串 1 只或者 2 对管组成整流桥）或者并联二极管（反接时短路使大电流流经保险丝，最终使保险丝熔断开路），这种保护方式简单粗暴，成本低，适用在小电流电路中。

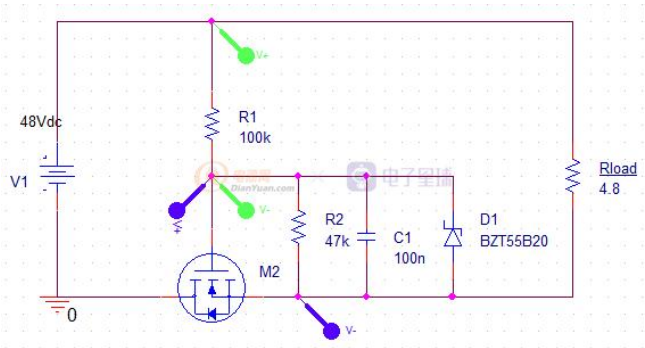
原理图放在下图中。



但是这种电路放在大电流电路中并不适用，首先咱们计算下功耗，拿流过 10A 回路电流举例，咱们选择 20A/100V 的肖特基二极管和 20A/100V 的 MOSFET 作比较：1.查规格书（Diode 规格书链接）中 If vs Vf 表格可知，Vf=0.72V@10A，所以二极管上功率 P=Vf*If=7.2W；2.查规格书（MOS 规格书链接）中 Id vs Rds 表格可知，Rds=0.03Ω@10A，所以 MOSFET 上功率 P=If*If*Rds=3W。两者功耗差别还是比较大的，因此在大电流电路中使用 MOSFET 设计防反接电路更加适宜。

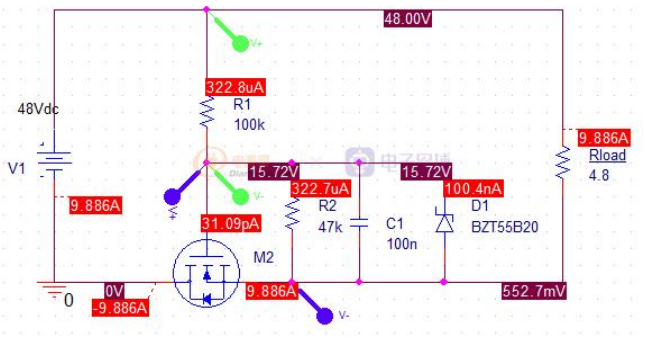
使用 MOSFET 设计防反接电路的优劣势也比较明显。优势：导通时回路电流流经 DS，阻抗相当于 MOSFET 的 Rds(on)，一般 MOS 的 Rds 都比较小，所以 MOSFET 的功耗很小，发热量也较小，这样就可以用较小的散热器，有利于实现产品小型化；劣势：如果用户将输入端正负极接反时，整个系统将不会工作。本人觉得使用的时候应该具体情况具体分析，适合自己的才是最好的。

下面将介绍防反接电路架构，注意以下电路中 MOSFET 的接法，D 接输入端负极，S 接后级负载端负极，工作原理：1.输入端正接时（上正下负），由于 MOSFET 存在体二极管，所以上电初始阶段 S 端被拉低，R1 和 R2 分压，其中 R2 上分得的电压为 MOSFET 提供偏置使其 DS 导通，此后 DS 就像开关一样被打开，维持稳定工作；2.输入端反接时（上负下正），DS 不通，无法构成回路，所以整个系统无法工作。

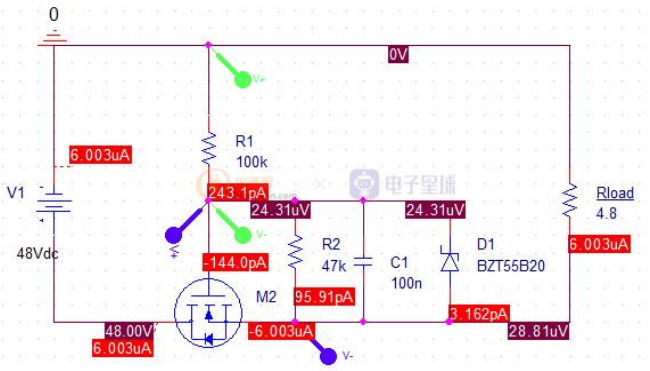


仿真结果（仿真使用 IRFP250 模型）：

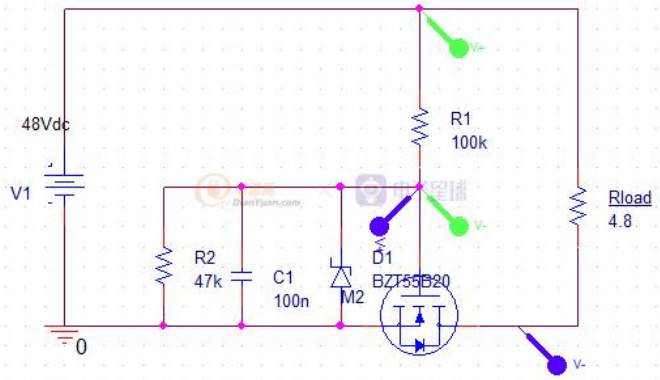
输入正接：



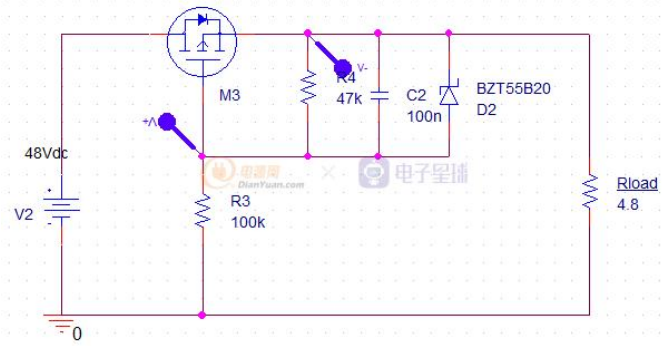
输入反接：



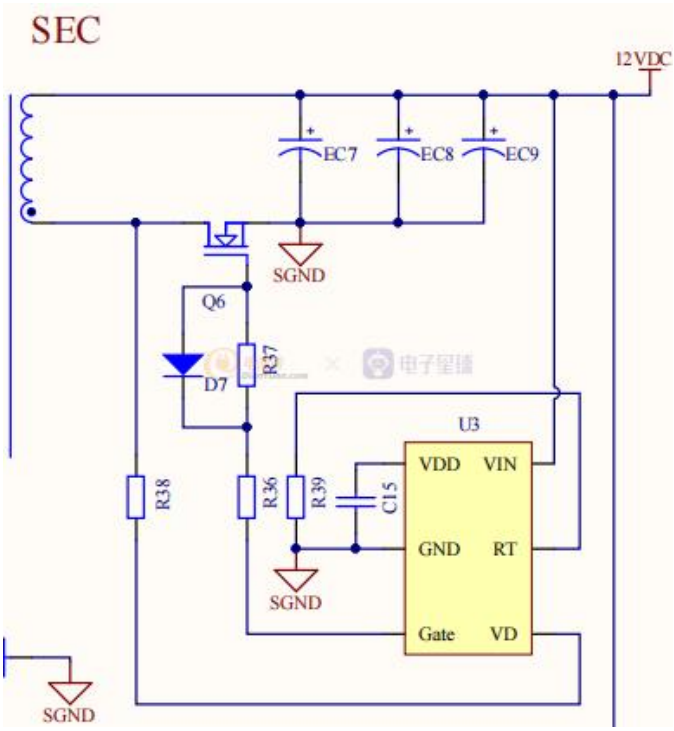
错误接法如下图，反接时起不到保护作用：



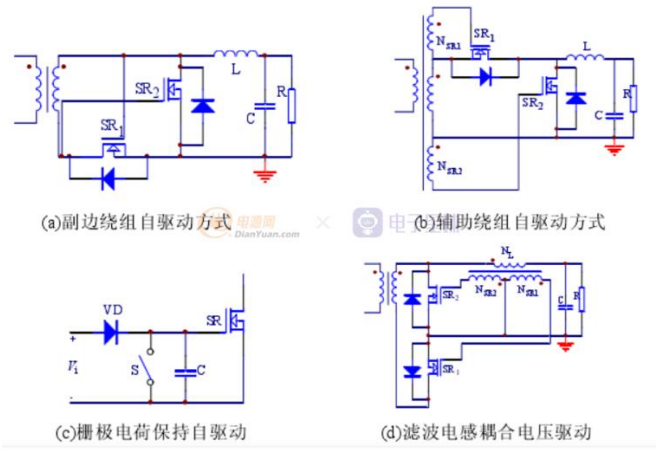
MOSFET 有两种接法，PMOS 和 NMOS，考虑到采购因素，我习惯性用 NMOS 多一些。同时也附上 PMOS 接法：



延伸电路：MOSFET 在同步整流中的应用，可以把变压器输出绕组看成为电压源，和上述防反接电路是同样的原理，控制芯片：OB2007MP（附规格书）。



其余常用同步整流电路。



留　言

互　动

一款汽车电子产品的 BCI 整改

文 / EMC 小白

谈到 EMC 整改，大家说的较多的是，干扰源，干扰路径，被干扰源三要素，也有很多人谈到 EMC 整改时，要先区分差模干扰，共模干扰，还有人提到要先判断传导干扰，还是辐射干扰，也有共模阻抗，感性耦合，容性耦合和空间辐射耦合，或者进行 RE 测试时，先判断是通过线束耦合，还是产品本身辐射等等，不知道大家怎么看待以上的理论分析。

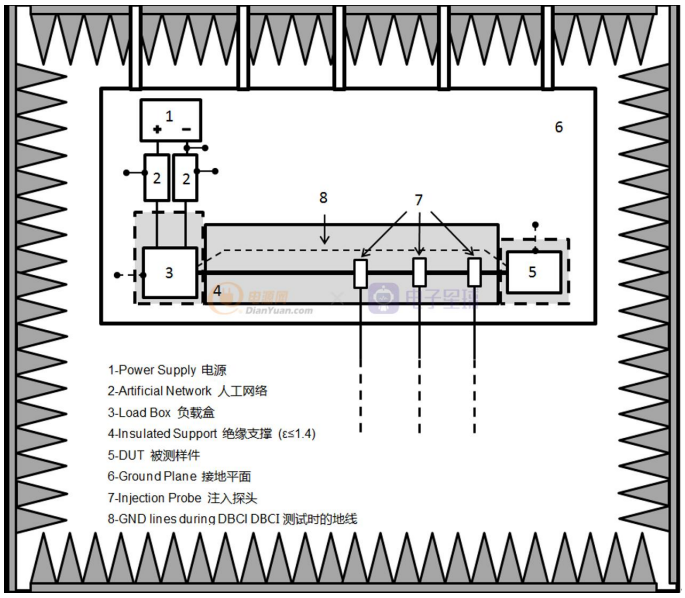
理论分析看得太多，如果没有深入思路彼此之间的联系，有时候自己也会困惑起来，遇到问题，我到底该按照什么思路来进行整改呢，是先判断差模干扰，共模干扰还是先判断以什么耦合方式进行干扰的，正所谓万变不离其宗，独孤九剑，以不变剑招破天下剑招，其实以上这些干扰方式谈论的都是一个要素，即干扰路径。用频谱分析仪或者经验排除法查找出干扰源是什么，再分析出潜在的干扰路径，对应辐射类问题，剩下的就是根据根据分析出的干扰路径逐一 debug；而对于抗干扰类问题，干扰源是确认的，再以差模，共模，公共阻抗耦合，容性耦合和感性耦合分析出潜在的干扰路径是什么，最后再结合产品原理分析以及排除法确认被干扰源，即可开展剩下的解决工作。

本文以一篇 BCI 实例，结合以上理论进行分析。

实验产品：

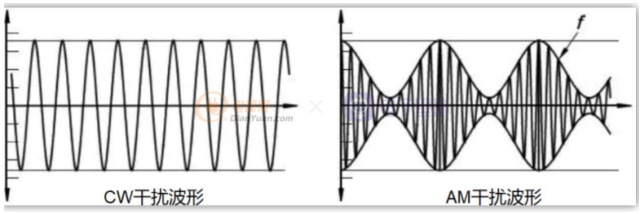
一款用于汽车无钥匙进入/启动的 NFC 产品。

实验布置图：



实验标准：

频率 Frequency(MHz)	Level 1 (dBuA)	Level 2 (dBuA)	方法 Method	调制方式 Modulation
1~15	86~102 86+13.61*log(f)	90~106 90+13.61*log(f)	DBCI [b]	CW, AM80 %
15~60	102	106	DBCI [b] & CBCI	CW, AM80 %
60~400	102~96 102-7.28*log(f/60)	106~100 106-7.28*log(f/60)	CBCI	CW, AM80 %
	功能状态 Function status		被测样件的测试线束都应置于注入钳之内。In the test frequency range all test wires of the DUT shall be routed inside of the injection probe.	
Region I	A	——		
Region II [a]	A	C		
Region III	——	A		



实验要求都是等级 A，即产品在实验过程中不允许出现任何的功能偏差。

实验现象：发现产品在进行 DBCI 和 CBCI 时，在 27.12MHz,40.68MHz 频点附近时出现读卡失效：

其他频点未出现失效。

实验问题分析：

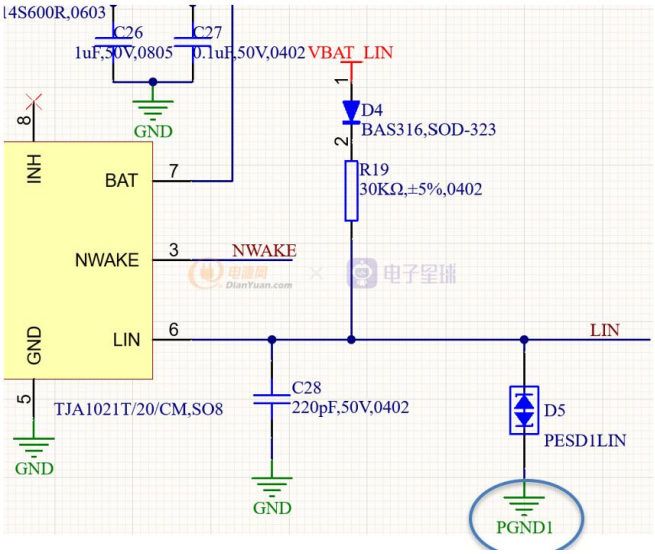
首先，干扰源很明确是 27.12MHz,40.68MHz 大电流干扰，干扰路径是传导耦合进入产品，再通过传导/辐射方式干扰，通过对产品的功能分析，最终被干扰源可能是 NFC 的供电电源，跟 NFC 读卡通信传输的 MCU 以及 NFC 芯片本身。

解决 EMC 问题，绝对是一项精力与金钱赛跑的技术活，面对的可是 EMC 实验室一小时大几百元的开销。如果从被干扰源出发，我们可能要做很多的排除法进行 dbug，可以先从干扰路径出发。

其次，干扰的路径是从线束耦合进入产品的，我们可以将产品分成两大部分，一部分是线束，一部分是产品本身，DBCI 是不包括 GND 线束，CBCI 是包括 GND 线束的，因为两者实验现象一致，为简单起见，我们先从 DBCI 开始分析，测试线束接口包括：

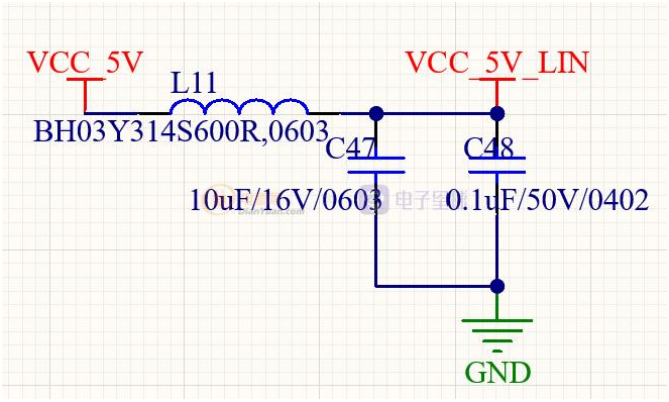
针脚名义 Pin Nominal	信号描述 Signal Description	端口类型 Port Type	备注 Remark
POWER	DC 9~16V	Power	
LIN	LIN，19200kbps	I/O	
NC	NC	/	
GND	地	Power	

其中 DBCI 时，只测试了 POWER 线束，LIN 线束，我们将 POWER 线束和 LIN 线束分开测试，发现给 POWER 线束注入大电流时，没有出现读卡失败，当给 LIN 线束注入大电流时，出现读卡失败了。分析 LIN 线束接口部分的电路：



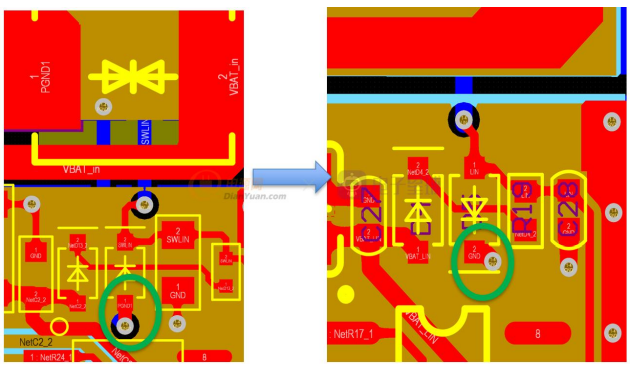
对于该部分电路，我们可以通过开头提到的各种理论分析，干扰途径包括，干扰通过 LIN 线束进入 GND 形成公共阻抗共模干扰耦合（例如通过 TVS 管进入 GND），干扰通过 LIN 线束攻击其他线束，例如电源，形成差模干扰耦合（例如通过 TVS 管进入 POWER），干扰通过 LIN 线束形成容性，感性耦合干扰到其他元器件（这部分的耦合应该可分为，电压驱动型和磁耦合型干扰，电压驱动下即因为形成高电压，对外形成耦合，磁耦合型即干扰形成回路后，再通过传导辐射耦合干扰到其他元器件），干扰通过天线形成空间对外辐射（排查了 layout，没有发现可形成直接对外辐射的天线）。

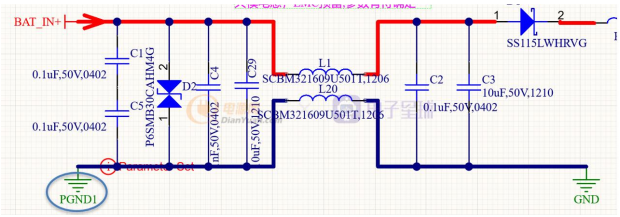
分析发现，通过 LIN 线束攻击电源：

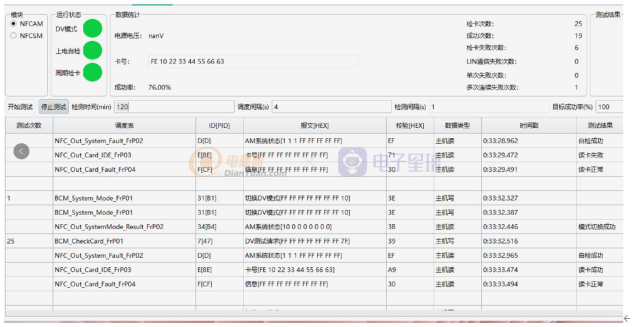


有可能通过 VCC_5V_LIN 耦合到 VCC_5V，我们尝试替换在 20MHz,40MHz 高阻抗的电感，发现对测试结果没有任何改善。

此时，我们发现 LIN 接口的 TVS 管的 GND 为 PGND1，当 BCI 干扰引入时，干扰信号最终会流入电源输入端的 PGND1，而 TVS 管到电源输入端的 PGND1 阻抗相对 GND 平面要大得多，流入的干扰信号会引起地弹，导致图示 A 点电位变高，该点的高电位信号会通过传导，耦合，辐射等多条路径干扰 PCB 板其他元器件工作（包括 NFC 芯片工作），进而引起 BCI 测试失效。而将 ESD 管地改为 GND 后，GND 平面阻抗很小，不会引起某点电位升高，干扰到其他元器件工作，BCI 测试通过。







总结：对于别人说的理论，我们要多多结合实例案例在应用中不断磨合深化，提炼出属于自己的理论，就像孔子所说，学的很多，但是不思考，则会迷惑，往往陷入其中，不知道具体该怎么做。

留　言

互　动

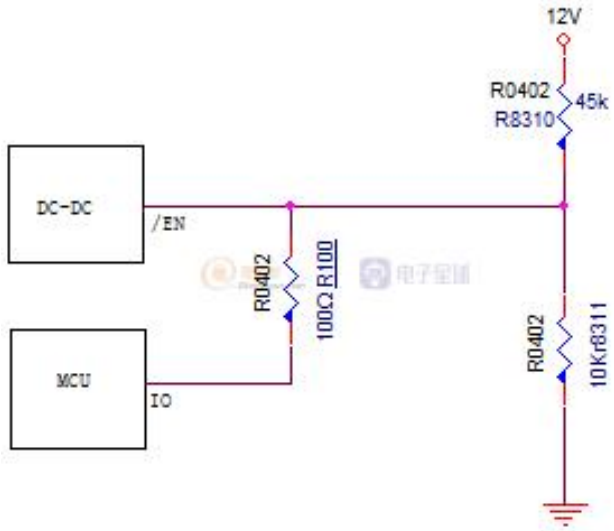
一款汽车电子静态电流整改

文 / 多宝鱼

汽车电子需要在产品中加入静态电流管理，为了保证汽车电平在存放状态不出现亏电状态，因此保证产品的静态电流符合车厂规定至关重要。

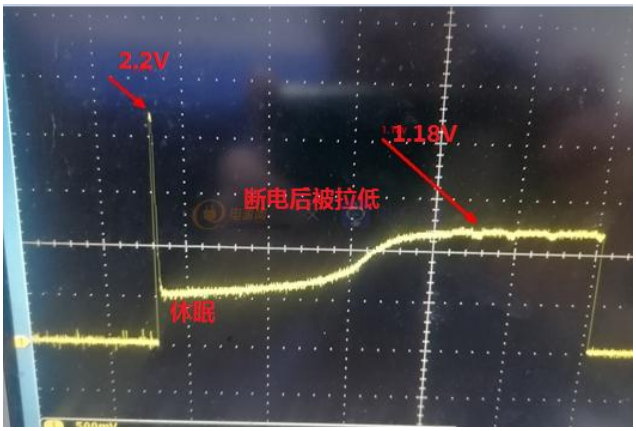
在汽车电子中需要使用 CAN 唤醒和休眠，如下图，在正常使用时，唤醒状态：CAN 被唤醒时 LDO 输出 3.3V 给到 MCU，MCU 拉低 EN 信号，DC-DC 工作。休眠状态：CAN 休眠 LDO 关闭，MCU 断电, DC-DC 不工作。

但在实际测试过程中，发现设备会出现不休眠的状态，因此需要分析为什么会出现不休眠的情况。



通过上图原理可知，正常状态 DC-DC EN 脚在高电平是不工作，在低电平工作，在 MCU 上电以后 MCU 通过拉低 IO 口，DC-DC 工作，MCU 断电状态，EN 脚通过电阻分压为高电平，DC-DC 不工作。

上述理论看似没问题，但是在实际测量过程中，发现在 MCU 断电状态，EN 脚的电压被拉低了，不能维持 2.2V 高电平，如下图，休眠后电压处于 0.4V-1.2V 之间（电压低于 0.4V 属于低电平，高于 1.2V 属于高电平），EN 脚不能维持高电平，启动 DC-DC 芯片，导致静态电流大，休眠失败。



所以一个问题，MCU 都掉电了，为啥还会把外围电路电压拉低？

带着这个疑问，问了一下供应商，看看 MCU 在掉电状态内部什么电路会把外围电路电压拉低。供应商给我的答复是有可能是 IO 口的防静电管，导致被分压，如下图：

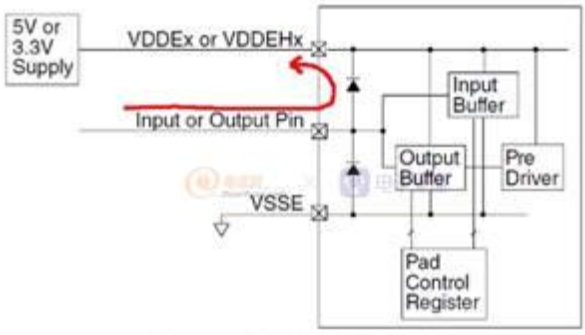
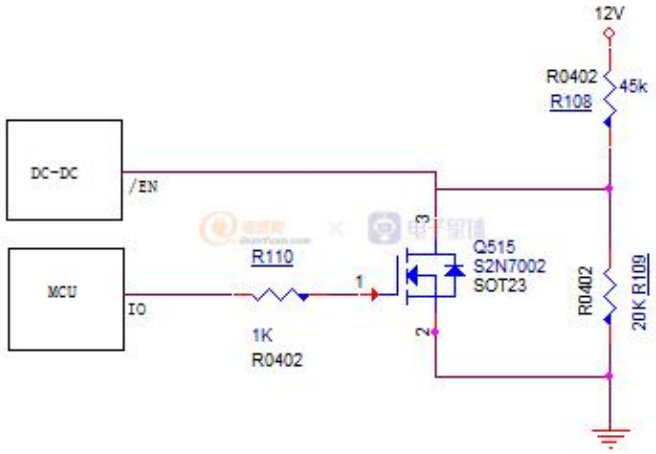


Figure 1. Typical I/O circuit

通过万用表二极管档，测量 IO 口，确实存在 0.7V 左右的二极管压差，可确定 MCU 内部存在防静电二极管，电流流向如上图所示。

知道原因后对原理图进行整改，如下图，增加 NMOS 开关来实现和 IO 的隔离，同时把分压电压调大，在 MCU 掉电以后，EN 脚一直为高电平，不会出现被拉低的情况，测试后无问题。



总结：MCU IO 口在掉电状态需要留意内部防静电二极管的存在，在做高低使能时充分考虑 IO 口掉电状态情况。

留　言

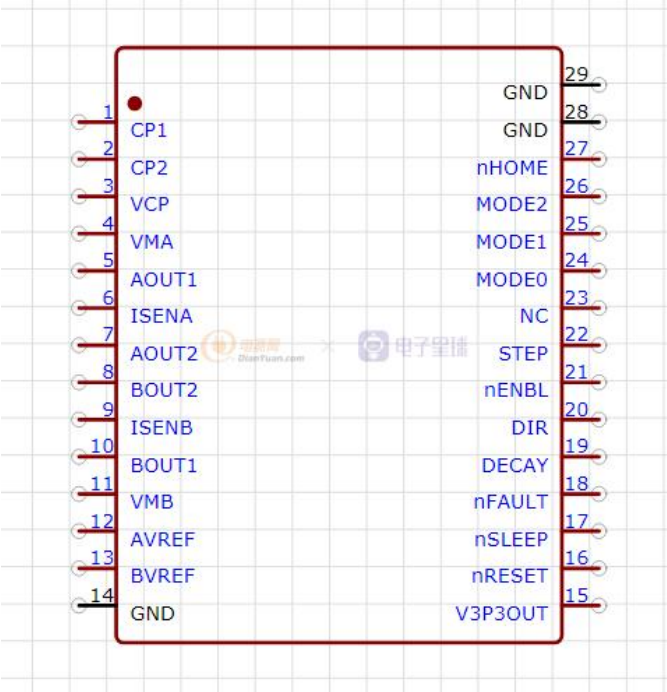
互　动

案例分析：主流步进电机闲时半流控制方案

文 / Elec 杂谈

步进电机是一种将电脉冲信号转换成相应角位移或线位移的电动机，在医疗仪器设备、计算机外设及存储设备、精密仪器、工业控制系统、办公自动化、机器人等领域中经常能看到它的身影。国内外各芯片厂商也纷纷抢占这一市场，推出了越来越多驱动芯片供应用工程师选择，工程师根据厂家提供的规格书可以很轻松地设计出符合自己所需的应用驱动。

目前市场主流的驱动芯片（如下图所示是其中一款驱动芯片）只需要几个 IO 口就可以使用它对电机实现精准控制。



在满足 21 脚 nENBL 输入低电平，此时 22 脚 STEP 输出 PWM，电机就可以运转。我们使用时会碰到一个问题，当 nENBL 被使能后，无论 STEP 是否有输出，系统都是满电流输出，因为一旦 nENBL 输入低电平，内部 H 桥输出使能。我们希望 STEP 有脉冲输出时（电机运行），电流输出正常，而当 STEP 没有脉冲输出时（电机暂停运行），电流可以大幅减小。有人想到电机停转时把 nENBL 拉高，这是不行的，nENBL 拉高，虽然电流为 0，但是此时电机将变为失锁状态，失去扭力。

我们再看 12、13 脚 VREF，这两个引脚通常都是通过一个可调电位器接到电源上。VREF 上分得的电压大小决定驱动器提供给负载电流的大小，而负载电流越大电机输出的扭力也就越大。回到我们的需求就可以把问题转换成当 STEP 有脉冲输出时，VREF 分得的电压较大，提供满足负载正常运行的输出电流，而当 STEP 无脉冲输出时，VREF 分得的电压较小，只需提供可以保证锁住电机的较小的电流（通常为电机正常工作时的一半），这样不仅可以有效降低电机闲时功耗，还可以减少电机和驱动器的热量，增加使用寿命。

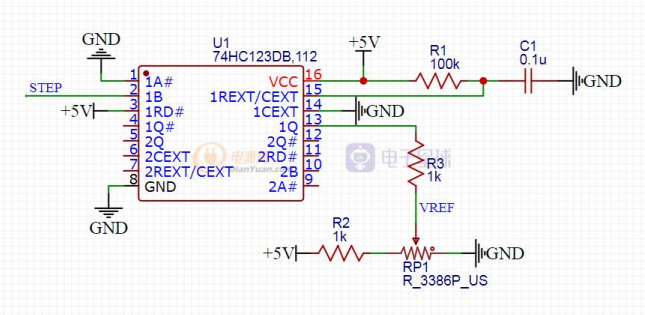
下面介绍一个主流方案，需要用到一个逻辑芯片——74HC123。

74HC123 是双路可重复触发的单稳态触发器，它输出的脉冲宽度取决于定时电阻 R 和定时电容 C，脉宽宽度 WP=R*C。先看一下真值表：

真值表

INPUT			OUTPUT	
n $\overline{RD}$	n $\overline{A}$	nB	nQ	n $\overline{Q}$
L	X	X	L	H
X	H	X	L	H
X	X	L	L	H
H	L	↑	⊓	⊔
H	↓	H	⊓	⊔
↑	L	H	⊓	⊔

我们需要将 STEP 脚接到 74HC123 的一个输入端，在 STEP 脚有脉冲输入和无脉冲输入的情况下，74HC123 的一个输出端分别输出高电平和低电平。蓝色框选的状态就是对应 nB 输入低电平，nQ 输出低电平；红色框选的状态对应 nB 输入方波信号并检测到上升沿，nQ 输出一个高电平脉冲信号，有人会有疑问：不是希望 nQ 输出高电平吗？之前提过这个脉冲信号的脉宽和 R、C 有关，我们只需给 R、C 取适当的值，使脉宽 WP 大于步进电机最慢转速下对应的 PWM 的周期，就会使 nQ 保持高电平，因为当检测到 nB 的上升沿后，nQ 输出高电平脉冲且脉宽较长，高电平一直维持到下一个周期上升沿又被检测，所以 nQ 会保持高电平不变。根据这些条件绘制了如下原理图：



当 STEP 有方波输入，1Q 输出高电平，即 R2 和 R3 并联之后再和 RP1 分压；当 STEP 无方波输入，1Q 输出低电平，即 R3 和 RP1 并联再和 R2 分压。可以通过给电阻分配合适的阻值让前者分得的 VREF 的电压值将近后者的双倍，达到闲时 半流控制的效果。

留 言

互 动

PCB 板上的晶体不起振，为啥？如何调整？

文 / 硬件微讲堂

关于晶体，我已经发过两篇文章，它们主要讲关键术语和频偏计算：

①晶体在使用过程中需要注意哪些点？(一)-- 关键参数

②晶体在使用过程中需要注意哪些点？(二)--PCB 板实际频偏计算

前几天看到有粉丝留言让讲一下“无源晶体起振的条件”，这是个不错的问题，我觉得有必要聊聊，所以就有了这第 3 篇文章。

一、一道面试题

照例，先抛出来一道面试题：“PCB 板上的晶体不起振，可能是什么原因？”。这个问题比较常规，笔试题中标的概率比较高，实际使用时也可能会碰到这类问题。如何回答，10 秒时间自己先思考下。

二、负性阻抗(-R)是什么？

要回答这个问题，我觉得有必要先搞清楚什么是晶体的负性阻抗。

负性阻抗 ,英文:Negative Resistance ,简写:-R ,单位:Ω。我在网上搜了下，找到如下描述：负性阻抗是指从石英晶体共振子的二个端子往振荡线路看过去，所得到振荡线路在振荡频率时的阻抗特性值。

还有一种解释，是从 Murata 网站上看到的，如下：负性阻抗是指用阻抗表示的振荡电路的信号放大能力。

这两个定义，看起来都比较偏学术，不太通俗。我个人倾向于 Murata 的解释。大家知道大概意思就行了，不必纠结，重点是下面的内容。

三、(-R) 能做什么？

上面的定义，你可以不理解不清楚，但是下面这个你必须搞清楚弄明白。

敲黑板，这才是重点！！！

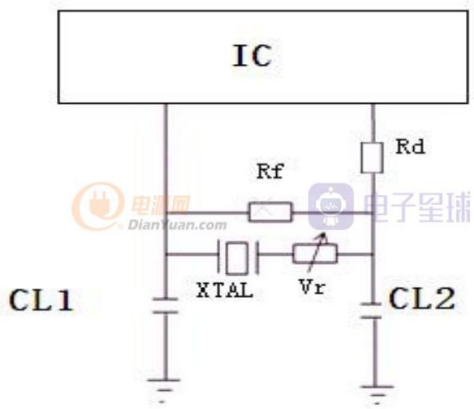
负性阻抗(-R)用于表征振荡裕量，即从振荡到振荡停止的裕量。

负性阻抗(-R)不是晶体的内部参数， 而是 PCB 电路的实测值，是晶体在 PCB 线路设计时的一个重要参数，可用于判断晶体振荡电路的稳定性。

如果在 PCB 上测出来的负性阻抗偏小，则表示该晶体振荡器的线路设计起振裕量不足，设计不合理。

四、(-R) 测量方法及步骤

说完负性阻抗(-R)的作用，再说下它的测量方法。



上图为负性阻抗测试的示意图。图中，Xtal 是晶体，Rf 是负反馈电阻，Rd 是限流电阻，CL1/CL2 是外部匹配电容，Vr 是可调电阻。

负性阻抗(-R)的测试步骤：

①将可调电阻 Vr 与晶体串联接入回路，如上图。Vr 是我们为了测试负性阻抗，刻意添加进去的

②调节可变电阻 Vr，使回路起振或停振。当回路刚停振时，测试 Vr 的阻值

③通过公式|-R|=RL+Vr，计算得到负性阻抗值

特别说明：RL 是晶体加负载电容的谐振阻抗，不是 ESR。网上看有些文章，直接写|-R|=ESR+Vr，这是不对的，不要被误导。

五、(-R) 阻值定量分析

根据上面的测量，此时已经得出负性阻抗(-R)的阻值|-R|。

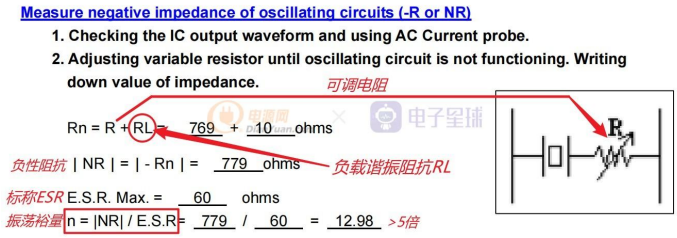
既然负性阻抗|-R|偏小表示晶体起振裕量不足，那|-R|阻值多少才算是一个合理值呢？

现在我们从定性分析已经进阶到定量分析，继续往下看。

至少是规格书标称 ESR 的 5 倍，才合适！阻值越大，振荡裕量越高，说明振荡越稳定。

六、实际案例

上面讲的一大堆，都是理论知识。



上图是某品牌的晶体匹配测试报告中负性阻抗(-R)和振荡裕量计算部分。

可以看出：可调电阻 R 实测值为 769Ω，RL 为 10Ω，则负性阻抗|-R|=779Ω。而该晶体标称的 ESR(max)=60Ω，则振荡裕量(倍率)=|-R|/ESR=12.98 倍，大于要求的 5 倍，满足起振要求。

七、晶体不起振的原因

再回到文章开头提到的面试题“PCB 板上的晶体不起振，可能是什么原因？”，负性阻抗偏小，振荡裕量不够，就是一个可能性比较大的原因。但这并不是唯一的原因，还有其他原因么？

当然有，其他原因有哪些，欢迎你在留言区交流。

八、晶体不起振，如何调整？

这也是一个不错的问题，我说 2 种方法：

方法①：减小外部匹配电容 CL1/CL2，以增大负性阻抗

方法②：更换标称值 ESR 更小的晶体，以增大振荡裕量

欢迎你在留言区交流调整措施。

九、总结

聊到这里，今天要说的也差不多了，总结下今天聊的内容：

①(-R)是指用阻抗表示的振荡电路的信号放大能力

②(-R)用于表征振荡裕量，即从振荡到振荡停止的裕量

③(-R)不是晶体的内部参数， 而是 PCB 电路的实测值

④(-R)可用于判断晶体振荡电路的稳定性

⑤(-R)的测试环境示意图和测试步骤

⑥(-R)的计算公式：|-R|=RL+Vr，特别说明 RL 不是 ESR

⑦(-R)的定量分析：至少是规格书标称 ESR 的 5 倍，才合适

⑧晶体不起振的原因：可能的原因之一是负性阻抗偏小，振荡裕量不够

附加题：

⑨晶体不起振的原因，除了负性阻抗偏小，其他还有哪些原因？

⑩晶体不起振，如何调整？已给出 2 种方法

怎么样？一个简短的问题，给出的回答可浅可深。我的助攻只能到这里，能否晋升到陆地神仙境，一剑开天门，就看你的造化了！

留　言

互　动

基于 GaN 器件双 Buck 逆变器（四）仿真分析

文 / Fourier

首先说一下在整个仿真过程中遇到的问题。最开始仿真只是照着原理图搭建 simulink 仿真图，然后对整个一个双 buck 逆变器进行仿真，最后有两种结果，

1.仿真失败，出现莫名其妙的错误；

2.仿真成功，输出莫名奇妙的结果。

一次成功的情况很少，几乎没有。

总结原因：

1.对 simulink 的仿真模块不熟悉，使用了错误的模块；

2.对电路不熟悉，电路参数，控制方法等。

对于电力电子仿真前，应该要做到这几点：

1. 计算好电路参数（最重要的是电感、电容和开关频率的匹配）；

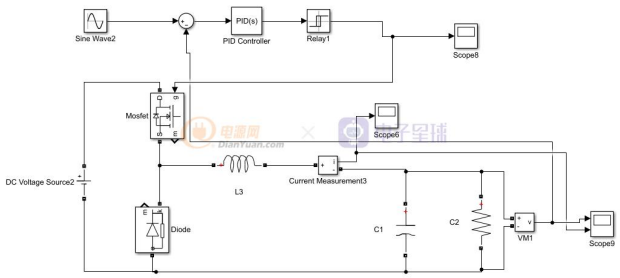
2.明白电路的控制方法，调制规则（如何产生控制信号、控制信号加在那个开关管）；

3. 要清楚理解电路的工作原理（不同控制信号，加在电路上电路的工作模块）；

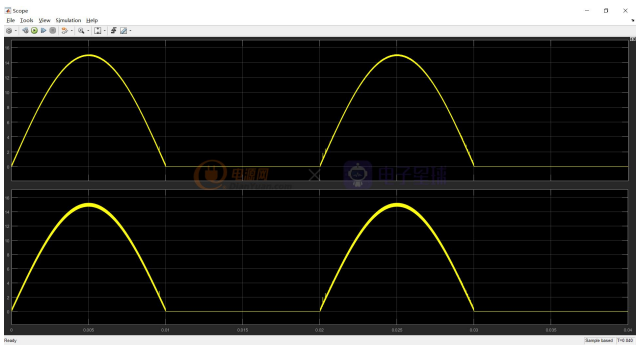
4.明白在仿真中要用到的模块的工作原理和适用范围（初学者可能会比较费力，仿真做多了就都熟了，如果要用到新的模块，最好的了解方式是通过 help 查看 simulink 给的技术文档）。

对双 buck 逆变器仿真，无论怎样都不能输出正弦波，以至于怀疑这个双 buck 逆变电路是否真的可以输出正弦波。在重新查看资料后，准备将双 buck 拆分，分成正负 buck 电路。

正负 buck 电路仿真

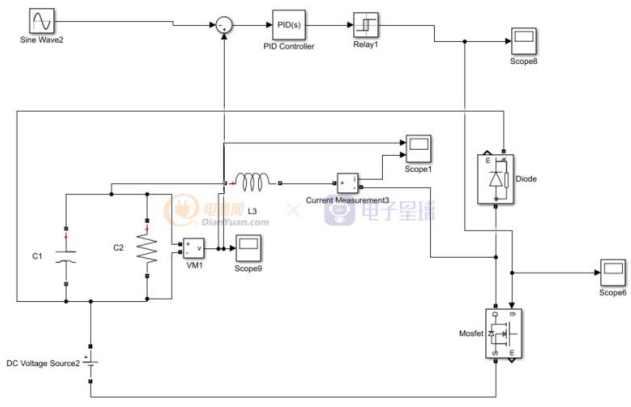


正 Buck 电路仿真图

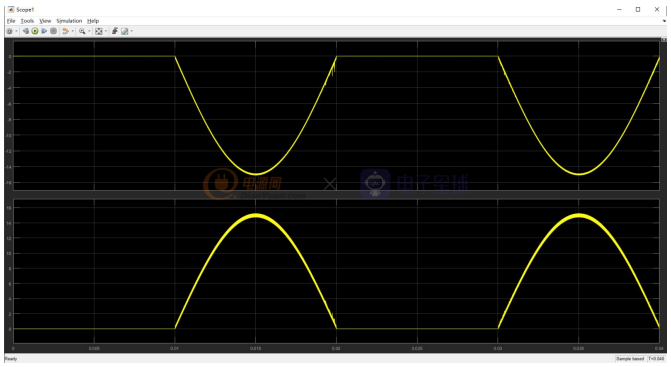


输出电压（上）和输出电流（下）

上面是传统 Buck 电路，采用简单的闭环控制，参考电压不是一个常数，而是正弦波，输出电压同为正弦波。跟前面分析相同，直流电压只为正的，buck 电路的输出电压与直流电压同向。只能输出正弦波的正半周期。



负 Buck 电路仿真图

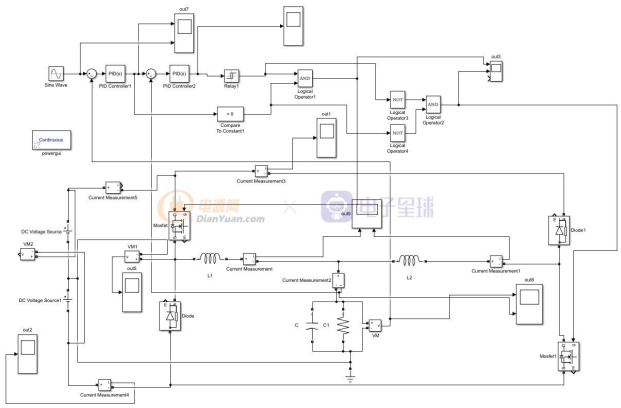


输出电压（上）和输出电流（下）

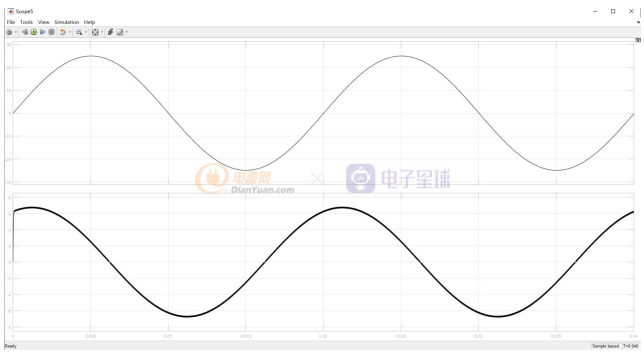
负 buck 电路也正常工作，输出正弦波负半周期。

双 Buck 半桥逆变器

将正负 buck 电路进行并联，在控制方面又加了电流环。

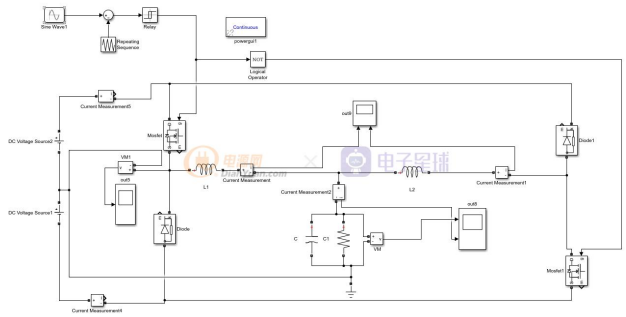


双 Buck 半桥逆变器闭环仿真图

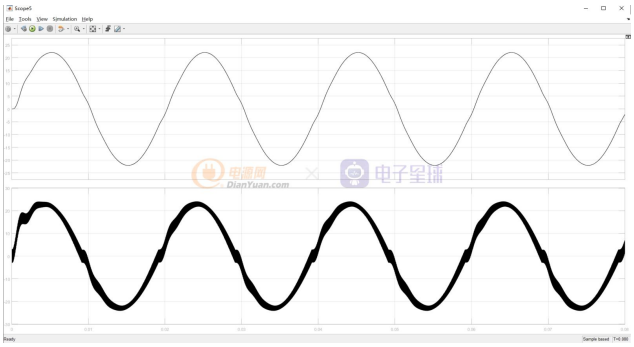


输出电压（上）和输出电流（下）

经过“拆分”和“组装”，双 Buck 半桥逆变器也算成功了。顺便做了下开环的仿真。



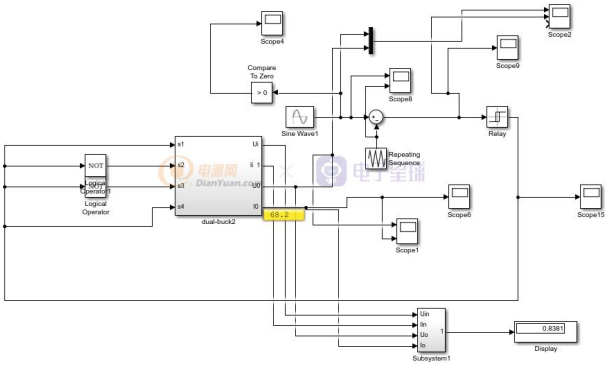
双 Buck 半桥逆变器开环仿真图



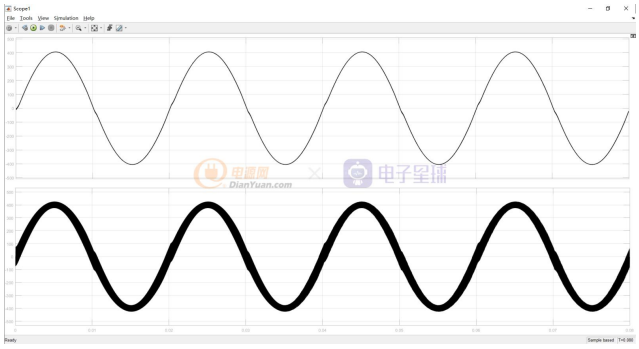
开环输出电压（上）和输出电流（下）

调制方式就是正常的 SPWM 调制，调制比为 0.8。开环也可以输出正弦波，因为没有电流电压环的控制，导致输出有些震荡和畸变，正弦波。经过了一系列的仿真，接下来对双 Buck 全桥逆变器的仿真就比较顺利了。

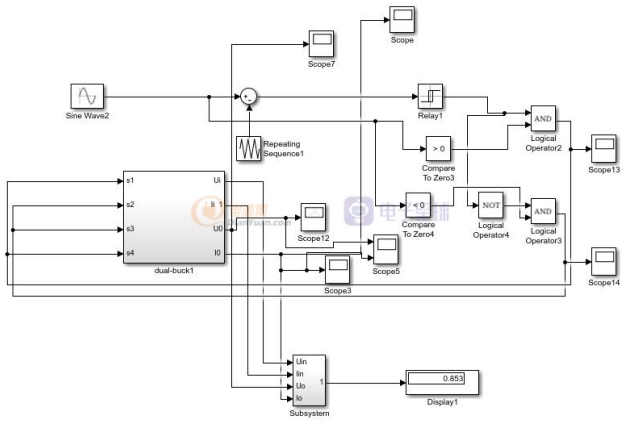
双 Buck 全桥逆变器



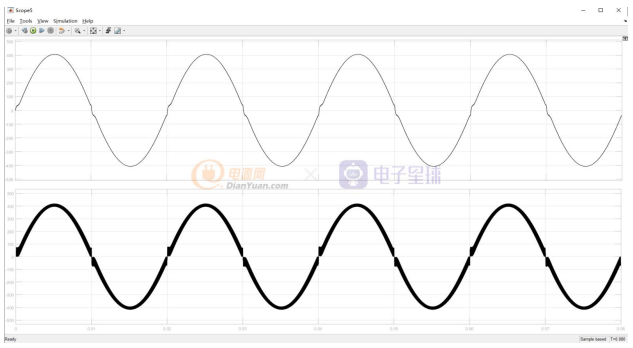
全周期控制



全周期控制输出电压（上）、输出电流（下）



半周期控制



半周期控制输出电压（上）、输出电流（下）

根据上面仿真结果可以看出，全周期控制的输出电流连续，输出电压在过零时几乎不存在畸变。半周期控制输出电流在过零处存在断续情况，且输出电压的过零畸变，较为严重。但是通过效率计算，半周期控制下的效率为 85.3%，全周期控制下效率为 83.81%。半周期控制下的效率还是较高的。

对比双 Buck 全桥逆变器和双 Buck 半桥逆变器，两者主电路都是 60V 的直流电源，都采用开环控制，SPWM 调制

调制比为 0.8 ,双 Buck 半桥逆变器的输出电压幅值为 20V 左右，双 Buck 全桥逆变器输出电压幅值为 40V 左右，可见双 Buck 全桥逆变器的直流电压利用率是双 Buck 半桥逆变器的二倍。

以上仿真结果全部符合前面的分析。

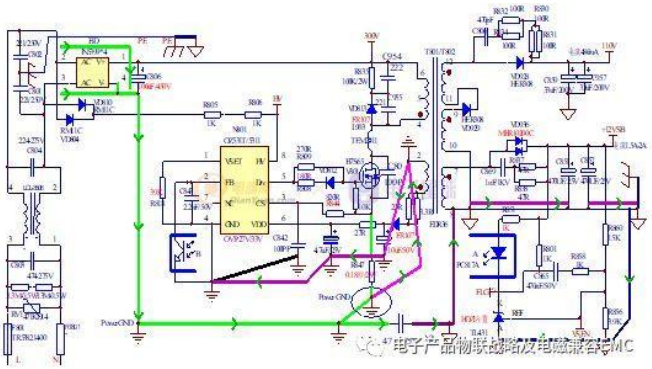
留　言

互　动

反激变换器的设计连载——RCD 吸收电路设计

文 / 杜佐兵

基本的反激变换器原理图如下所示，在需要对输入输出进行电气隔离的低功率<75W 的开关电源应用场合，反激变换器（ Flyback Converter ）是最常用的一种拓扑结构（ Topology ）。简单、可靠、低成本、易于实现是反激变换器突出的优点。接下来我将对电源的关键部分的设计进行说明！

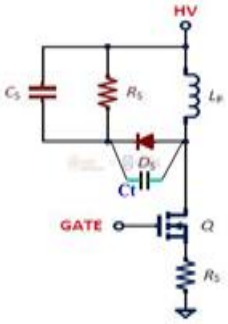


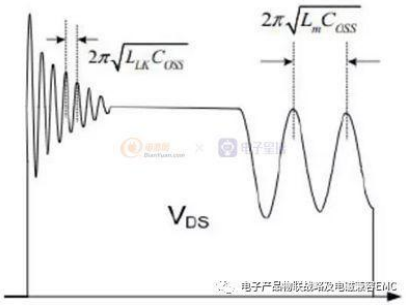
RCD 吸收电路的计算结果如下 :(上文已完成变压器关键设计)

11. Design RCD snubber			
Primary side leakage inductance (L_{lk})	20	uH	
Maximum Voltage of snubber capacitor (V_{sn})	134	V	
Maximum snubber capacitor voltage ripple	6.8	%	
Snubber resistor (R_{sn})=	99.0	kΩ	
Snubber capacitor (C_{sn})=	2.2	nF	
Power loss in snubber resistor (P_{sn})=	0.2	W	(In Normal Operation)
Peak drain current at V_{DC}^{max} (I_{ds2}) =	3.05	A	
Max Voltage of Csn at V_{DC}^{max} (V_{sn2})=	136	V	
Max Voltage stress of MOSFET (V_{ds}^{max})=	5[illegible]	V	

反激变换器在 MOS 关断的瞬间，由变压器漏感 LLK 与 MOS 管的输出电容造成的谐振尖峰加在 MOS 管的漏极，如果不加以限制，MOS 管的寿命将会大打折扣。因此需要采取措施，把这个尖峰吸收掉。

因此反激的 RCD 吸收电路设计对 FLY 的 EMI 及 MOS 的应力都有比较大的影响。对于<75w 的 FLY 设计，为了保证其参数的最佳化设计：





开关 MOS 管关断时的实际波形图

注意：RCD 吸收电路的设计对系统的 EMI 也会有很大的改善！

■ RCD 吸收电路（DS, CS,RS）将改变 MOSFET 关断时的突波振幅与振荡频率，进而改变了杂讯频谱。

■ 电压 Vds 波形改变了共模杂讯，电流 ID 波形改变了差模杂讯。

RCD 的钳位电路设计理论依据：

励磁电感能量可通过理想变压器耦合到副边，而漏感因为不耦合，能量不能传递到副边。如果不采取措施，漏感将通过寄生电容释放能量，引起电路电压过冲和振荡，影响电路工作性能，还会引起 EMI 问题，严重时会烧毁器件。为抑制其影响，可在变压器初级并联无源 RCD 钳位电路。

对于本设计我将理论和实际相结合，让大家对设计有更快速直观的认知，并且对于同类的设计直接分辨出优劣及问题！

先从理论计算上，给出我的计算方法：

RClamp 由下式决定：其中 Vclamp 一般比反射电压 Vor 高出 50～100V，LLK 为变压器初级漏感，以实测为准（本设计例中电感为 280uH，实测的最大漏感为 20uH）：

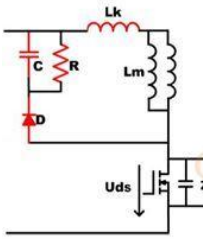
$$R_{clamp}=\frac{2\times V_{clamp}\times(V_{clamp}-V_{or})}{L_{Lk}\times f_{sw}\times I_{dspeak}^2}$$

CClamp 由下式决定：其中 Vripple 一般取 Vclamp 的 5%～10%是比较合理的：

$$C_{clamp}=\frac{V_{clamp}}{V_{ripple}\times f_{sw}\times R_{clamp}}$$

参考理论如下：

开关管截止时，漏感能量等于电容C上吸收的能量：



$$\frac{1}{2}C\left(U_{ds\_max}-U_{in}\right)^2-\frac{1}{2}CU_{reset}=\frac{1}{2}L_kI_P^2$$

U_{reset} 为电容C上的初始电压，U_{in} 为输入直流电压。

故：$$C=\frac{L_kI_P^2}{\left(U_{ds\,max}-U_{in}\right)^2-U_{reset}}$$

箝位电路的损耗为：$$P_{camp}=\frac{1}{2}L_kI_p^2f_s$$

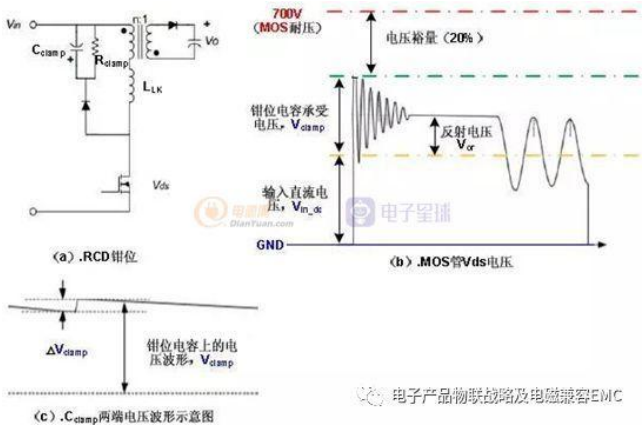
电阻R上的损耗为：$$P_R=\frac{\left(U_{ds\,max}-U_{in}\right)^2}{R}$$

故：$$R=\frac{2\left(U_{ds\,max}-U_{in}\right)^2}{L_kI_p^2f_s}$$

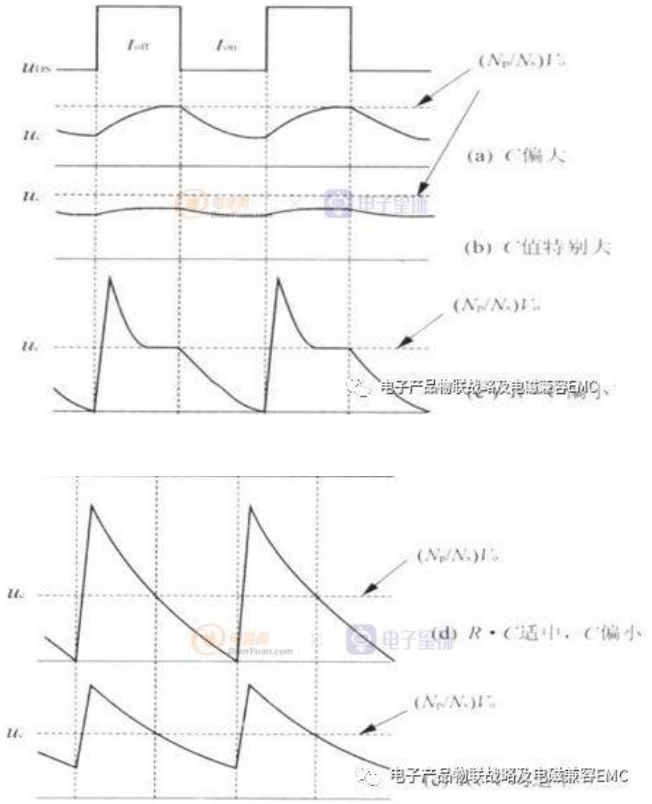
通过计算公式得出理论计算数据如下：

11. Design RCD snubber		
Primary side leakage inductance (L_{lk})	20	uH
Maximum Voltage of snubber capacitor (V_{sn})	134	V
Maximum snubber capacitor voltage ripple	6.8	%
Snubber resistor (R_{sn})=	99.0	kΩ
Snubber capacitor (C_{sn})=	2.2	nF
Power loss in snubber resistor (P_{sn})=	0.2	W (In Normal Operation)
Peak drain current at V_{DC}^{max} (I_{ds2}) =	3.05	A
Max Voltage of Csn at V_{DC}^{max} (V_{sn2})=	136	V
Max Voltage stress of MOSFET (V_{ds}^{max})=	511	V

理论的计算数据得出的分析原理结构图如下：



实际测试 RCD 钳位电容的电压波形 Data 分析：



引入 RCD 钳位电路，目的是消耗漏感能量，但不能消耗主励磁电感能量，否则会降低电路效率。要做到这点必须对 RC 参数进行优化设计，下面分析其工作原理：当 MOS-D 关断时，漏感 Lk 释能，二极管 Ds 导通时，C 上电压瞬间充上去，然后 D 截止，C 通过 R 放电。

设计的关键细节分析如下：

A.若 C 值较大，C 上电压缓慢上升，副边反激过冲小，变压器能量不能迅速传递到副边，见图(a)；B.若 C 值特别大，电压峰值小于副边反射电压，则钳位电容上电压将一直保持在副边反射电压 Vor 附近，即钳位电阻变为死负载，一直在消耗磁芯能量，见图(b);C.若 RC 值太小，C 上电压很快会降到副边反射电压，故在 MOS 开通前，钳位电阻只将成为反激变换器的死负载，消耗变压器的能量，降低效率，见图(c)：D.如果 RC 值取得比较合适，到 MOS 开通时，C 上电压放到接近副边反射电压，到下次导通时，C 上能量恰好可以释放完，见图(d)，这种情况钳位效果较好，但电容峰值电压大，器件应力高。

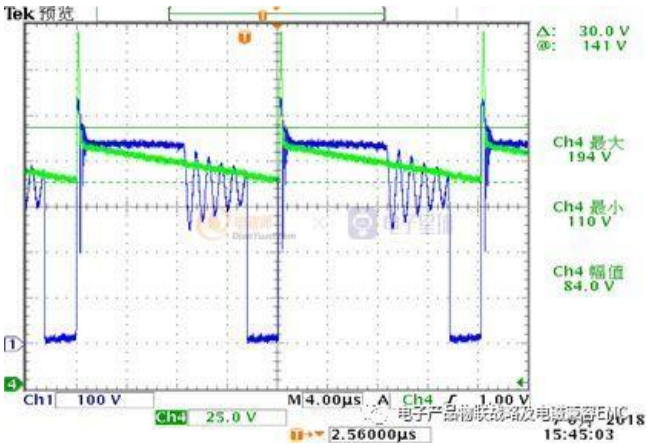
注意：B 和 C 两种方式是不允许的。A 种方式电压变化缓慢，能量不能被迅速传递；D 种方式电压峰值大，器件应力大。

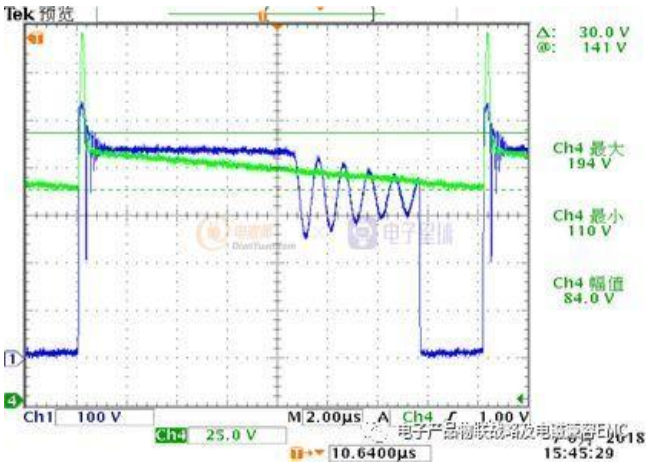
E.可折衷处理：在第 D 种方式基础上增大电容，降低电压峰值，同时调节 R，使到开关 MOS 开通时，C 上电压放到接近副边反射电压，之后 RC 继续放电至开关 MOS 下次开通，如图(e)所示。

我们进行实际参数的具体取值进行测试分析：

小的吸收电阻对开关 MOS 的开关应力确实有帮助。关键点要注意我们电路设计的要解决的问题点在哪儿，建议采用最佳设计来满足电路要求！

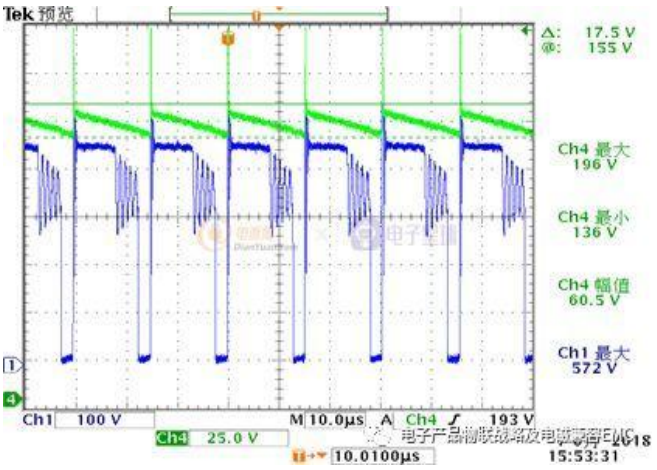
R=47K/2W　C=2.2nF/630V D=FR207

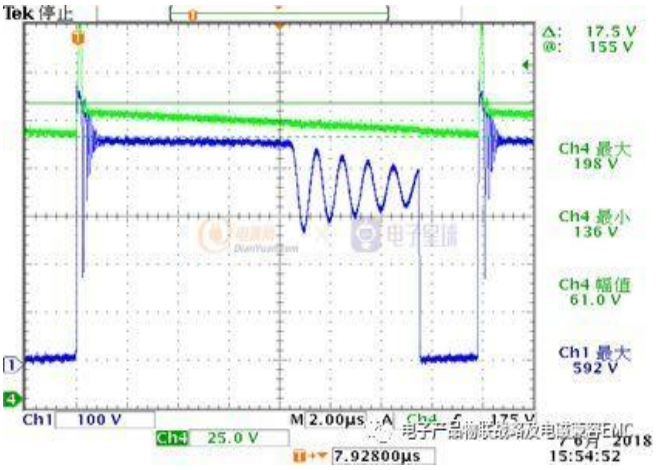




CH1: MOS-VDS　CH4: UC(钳位电容电压)

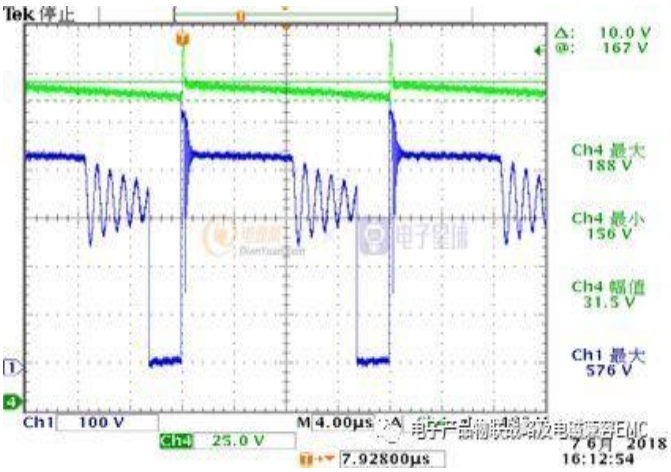
R=100K/2W　C=2.2nF/630V D=FR207

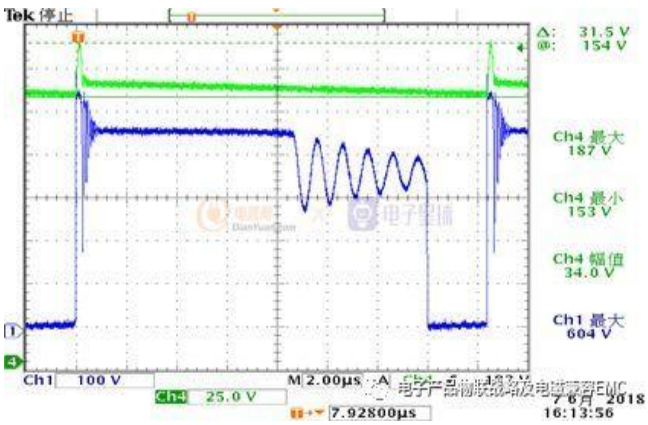




CH1: MOS-VDS　CH4: UC(钳位电容电压)

R=100K/2W　C=4.7nF/630V D=FR207





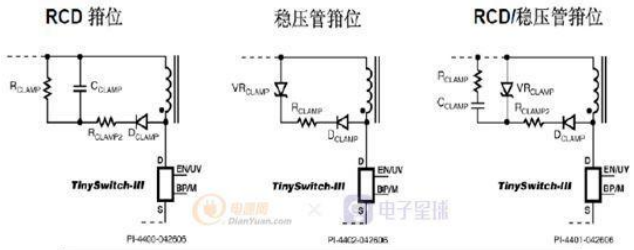
CH1: MOS-VDS　CH4: UC(钳位电容电压)

测试结论：R=47KR/2W- 100KR/2W C=2.2nF/630V 的 RC 参数是可行的！

在我的这个设计方案，我同时对 RCD 参数进行过极端参数评估，

实践与理论的评估先保留！后期如碰到案例再分享给大家。

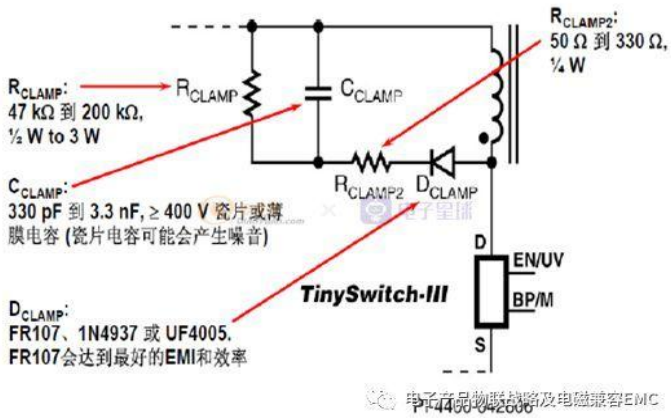
我们在实际的应用过程中，对系统的待机功耗及效率甚至 EMI 有更高的要求时 RCD 吸收电路会有较多的派生电路;提供设计给大家参考！

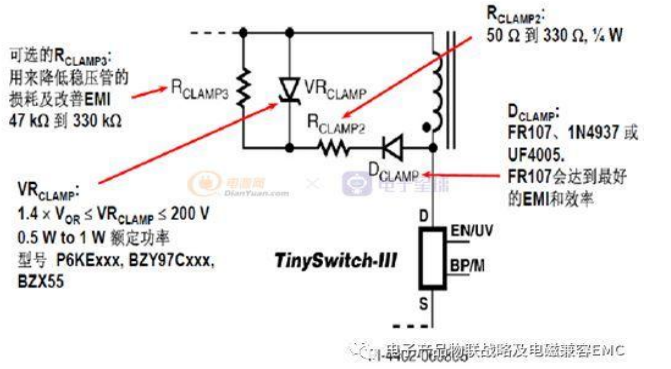


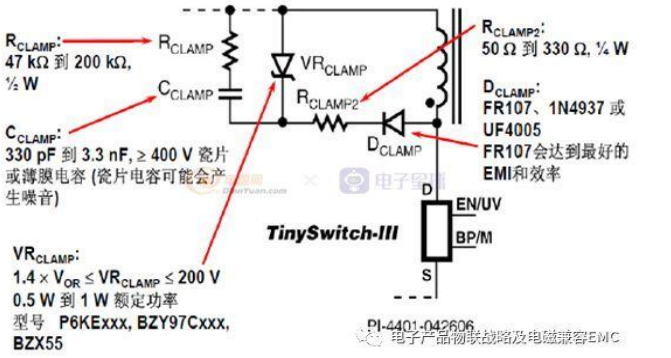
MOSFET Drain Clamp	RCD	Zener	RCD / Zener
Component cost	Lowest	Higher	Highest
No-load input power	Highest	Lowest	Low
Light-load efficiency	Lowest	Highest	High
EMI generation	Lowest	Highest	Low

留 言

互 动







针对需要最低空载功耗最高轻载效率和最低的 EMI 应用

根据上面的结构组合，我还有其它吸收电路组合。在实际应用遇到问题时，可以与我交流。

11. Design RCD snubber			
Primary side leakage inductance (L_{lk})	20	uH	
Maximum Voltage of snubber capacitor (V_{sn})	134	V	
Maximum snubber capacitor voltage ripple	6.8	%	
Snubber resistor (R_{sn})=	99.0	kΩ	
Snubber capacitor (C_{sn})=	2.2	nF	
Power loss in snubber resistor (P'_{sn})=	0.2	W	(In Normal Operation)
Peak drain current at V_{DC}^{max} (I_{ds2}) =	3.05	A	
Max Voltage of Csn at V_{DC}^{max} (V_{sn2})=	136	V	
Max Voltage stress of MOSFET (V_{ds}^{max})=	511	V	

SiC MOSFET 在恒定栅极偏压条件下的参数变化

文 / 英飞凌工业半导体

在正常使用器件时，由于半导体-氧化层界面处缺陷的产生和/或充放电，SiC MOSFET 的阈值电压可能略有漂移。阈值电压的漂移可能对器件的长期运行产生明显影响，具体取决于漂移量。由于这种漂移通常是向更大的电压值偏移，因此会导致器件的导通电阻变大。这又导致损耗增加，以及散热需求增大，从而可能缩短器件的使用寿命。因此，了解阈值电压的行为并考虑它对设计余量的影响非常重要。

这种现象在 Si 技术中已非常常见，被称之为“偏压温度不稳定性”（BTI）。考虑到 SiC 属于宽禁带半导体的事实，即，它不仅由硅（Si）而且由碳（C）原子组成，SiC/SiO2 界面的特性相比 Si/SiO2 界面稍有不同。在 SiC/SiO2 界面存在位于更大能量范围内的其它点缺陷类型，它们必须通过其它的氧化后处理（比如，用氧化氮代替氮氢混合气氛退火）进行钝化。此外，由于 SiC 的带隙较宽，在半导体与 SiO2 栅极氧化层之间更容易进行载流子交换。这些差异自然又会使得 SiC MOSFET 的电气特性和动态漂移特性相比 Si MOSFET 稍有改变。

很多努力已经付出在改善 SiC MOSFET 的性能上，但性能改善未必能带来更好的器件可靠性。为保证器件特性长期稳定，必须密切关注 BTI 这种漂移现象。在英飞凌，我们在追求一流器件性能的同时，也在设法实现最优异的器件可靠性。因此，我们开展了深入的研究，以期能够深入地了解潜在效应，评估 BTI 效应在现实应用中的影响，并制定出能够尽可能地抑制 BTI 效应的措施。

SiC MOSFET 在恒定栅极偏压条件下的参数变化（DC BTI）

1.DC BTI 简介

DC BTI 效应不仅存在于 SiC 功率器件中，在硅（Si）技术中也很常见。当在高温条件下给 Si 或 SiC MOSFET 的栅极施加恒定的 DC 偏压时，可以观察到阈值电压和导通电阻的变化。改变的幅度和极性取决于应力条件（偏压、时间、温度）。施加正栅极偏压应力（PBTI）时，通常可以观察到阈值电压向更高的电压偏移；而如果施加负栅极偏压应力（NBTI），阈值电压则向相反的方向偏移。这种效应是由 SiC/SiO2 或 Si/SiO2 界面处或附近的载流子捕获引起的，可以通过优化器件工艺控制在最低水平。为更好地了解和预测 SiC MOSFET 中的 DC BTI，英飞凌对这个问题展开了深入的研究，重点了解它相比 Si 技术存在哪些不同。就 Si MOSFET 而言 英飞凌过去已经对 BTI 有了扎实的了解，并且已与众多著名高校一道为科学进步作出了重大贡献。已经掌握的退化物理学和电气测量技术知识，如今已被用于研究英飞凌的 SiC 器件。事实上，尽管材料特性不同，Si 和 SiC 技术在 DC BTI 方面却存在许多相似之处。然而，它们在有些方面仍然存在不同，在测量和评估特定应用中的参数变化时必须考虑到这些不同。

2.测量 SiC 功率器件的 DC BTI

由 DC BTI 引起的阈值电压变化由两个分量组成：一个是快速、可恢复的分量，另一个是准永久（恢复很慢）的分量。准永久分量决定器件的长期漂移量，而快速分量能在短时间内恢复。

为了获得可比较的漂移值，已制定测定 BTI 漂移的工业标准，如 JESD22 和它的扩展标准 AEC-Q101。这些标准都是以 Si 技术为基础建立的，必须针对 SiC 技术进行完善，如下所述。

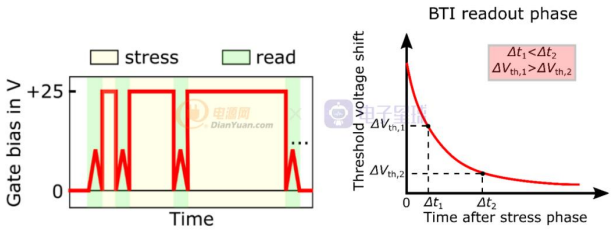


图 6.以 PBTI（脉冲 BTI）应力为例，典型的 DC BTI MSM（测量-应力-测量）序列。左图显示的是测量信号与时间的关系。右图显示的是阈值电压漂移的恢复与时间的关系，旨在表明读数延迟对提取的阈值电压漂移的影响。即使读数时间有很小的差异，提取的阈值电压漂移也有很大不同。

测量 DC BTI 的传统方法是以测量-应力-测量（MSM）为顺序，先反复地给栅极施加偏压和温度应力，然后读数，如图 6 中的左图所示。借助这种方法及合适的设备，以上所述的两个漂移分量都能被测量出来。但是，获得的阈值电压漂移在很大程度上取决于读数时间——即应力阶段与读数阶段之间的时间间隔，以及器件的状况。从图 6 中的右图可以看出，阈值电压漂移在应力结束后以指数级速度恢复。于是，即使读数时间有很小的差异——比如 1ms vs. 100ms，提取的阈值电压漂移也有很大不同。因此，这种简单的方法存在的缺点是重现性差，且难以区分阈值电压漂移中的完全可恢复的快速分量（滞后效应）与更加依赖于应用的准永久分量。

因为这个原因，英飞凌建议使用改进版的 BTI 测量序列，其中需要用到预处理脉冲，如图 7 所示。以预处理过的 PBTI 为例，读数阶段包含累积脉冲、在固定电流电平下的一次读

数、反向脉冲和二次读数。在所有序列都完成之后，即在二次读数时，留下的主要是准永久的 BTI 分量，它几乎无法恢复或者恢复很慢。这意味着，预处理使得测量结果更容易被重现，更不易受到读数延迟和器件状况的影响，并允许正确地区分滞后效应与漂移效应。

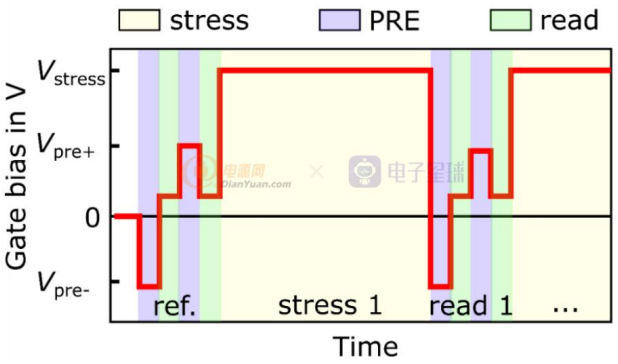


图 7.预处理过的 PBTI 的测量序列。读数阶段包含累积脉冲、一次读数、累积脉冲和二次读数。二次读数得到的是最稳定的、可重现的结果。

同一个读数阶段中的一次读数与二次读数之差代表阈值电压滞后现象。它随时间发生的漂移表示产生了新的界面态。预处理脉冲模拟的是栅极在应用中的开关过程，可将陷阱态转化为预定的电荷态，从而减少读数延迟与器件状态的影响。

3.SiC 和 Si 功率 MOSFET 的 DC BTI 比较

在以前发表的文章中 ,经常是说 SiC MOSFET 的漂移量显著高于 Si 功率器件。然而我们已经证明，英飞凌的 SiC 功率 MOSFET 具有的 NBTI 漂移量(负 BTI)很小 ,可与最先进的 Si 超结 MOSFET 器件相媲美（即使在给器件施加明显的过应力时)。这一结果是通过优化器件工艺来实现的。针对 SiC，我们给出了几种不同的工艺处理所带来的不同结果，以证明通过优化 SiC/SiO2 界面来改善或降低 BTI 的可能。

1）负偏压温度不稳定性（NBTI）

英飞凌研究了在 200°C 和-25V 的偏压应力下的 NBTI 漂移（图 8）。结果显示，通过几种工艺处理的改进，英飞凌 SiC MOSFET 的 NBTI 漂移可以减少一个数量级。在本试验的实验窗口中 ,最好的工艺改进版本所得到的 NBTI 漂移量 ,与 Si MOSFET 处于同一个数量级。SiC MOSFET 的漂移斜率甚至更小，表示随着应力施加时间的延长 ,它的漂移量将比 Si MOSFET 少。低 NBTI 是英飞凌 SiC MOSFET 器件的典型特征之一。

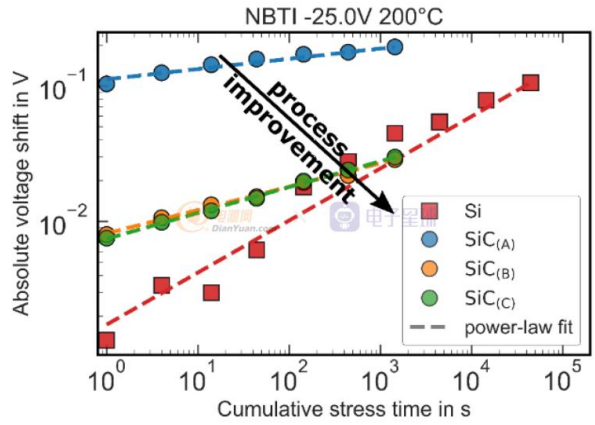


图 8.在 200°C 和-25V 的偏压应力下，NBTI 随时间的变化。通过改进处理工艺，英飞凌 SiC MOSFET 的总漂移量可被降到与同等的 Si 功率 MOSFET 类似的水平。

2）正偏压温度不稳定性（PBTI）

英飞凌研究了在 200°C 和+25V 的偏压应力下的 PBTI 漂移（图 9）。结果显示，Si 和 SiC 的 PBTI 有许多相似之处，而只有少许差异。

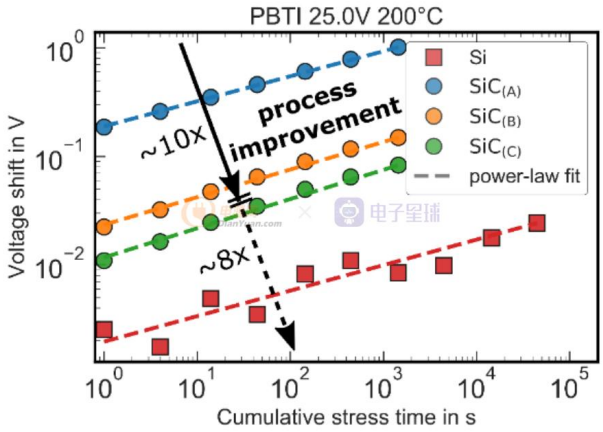


图 9.在 200°C 和+25V 的偏压应力下，PBTI 随时间的变化。取决于所用的技术和器件工艺 ,可以看到 Si 和 SiC 的 PBTI 随时间发生的变化是一致的，但绝对阈值电压漂移并不相同。SiC MOSFET 的 PBTI 更大，但仍然位于 100mV 的范围以内。

事实上，我们发现，SiC 和 Si 功率 MOSFET 的 PBTI 随时间发生的变化、电压加速（图 10）和与温度的关系都是一致的。

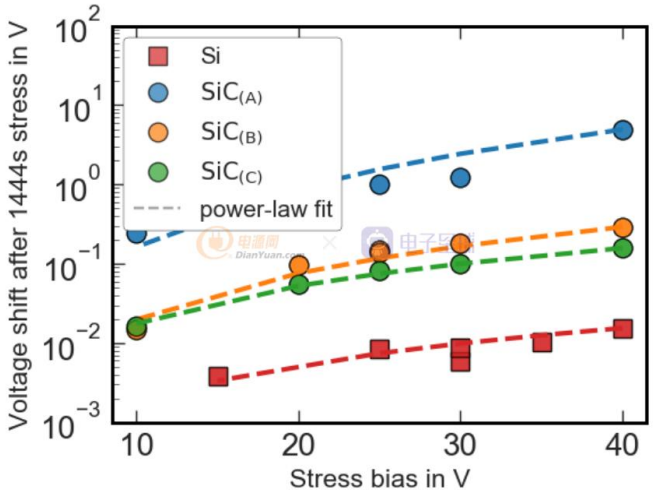


图 10.PBTI 在 200°C 下的电压加速。所有器件（无论是 SiC 还是 Si 技术）都显示出相同的电压加速，以及不同的绝对漂移。

剩余差异是绝对阈值电压漂移的补偿。通过优化器件处理，我们再次实现了漂移量降低一个数量级的目标，从而使得漂移量在本试验的实验窗口中落在了 100mV 的范围以内。然而，在这些试验条件下 ,最好的 SiC 器件的漂移仍是参比的 Si 器件样品的 8 倍左右。对于 Si 功率 MOSFET，PBTI 通常完全不是问题。所观察到的漂移补偿是 SiC 能带结构不同所导致的自然结果。

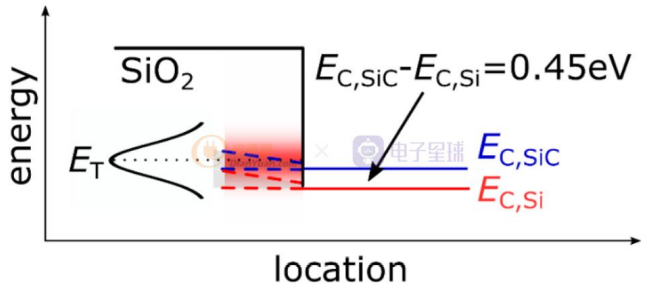


图 11.SiC/SiO2 和 Si/SiO2 界面的能带图。这两种技术的栅极氧化层中存在相同的陷阱分布。由于 SiC 的导带底更高，所以相比 Si，这一固有的陷阱能级更容易得到填充，这自然就使 SiC 的 PBTI 漂移更大——即使在假定 SiO2 的陷阱密度相同时。

图 11 显示的是 SiC/SiO2 和 Si/SiO2 界面的能带图 ,其中包含 SiO2 中靠近 SiC 导带边缘的一个已知的内在氧化物陷阱能级。正如我们在中所证明的，SiC 导带度更高使得电子更容易被捕获到该陷阱能级中，这是 SiC 器件在被施加 PBTI 应力后产生的漂移更大的主要原因。

3）DC BTI 漂移的建模

虽然 DC BTI 已经得到广泛的研究——尤其是 Si 技术的 DC BTI，但目前还没有被普遍认可的物理漂移模型。然而，利用实证幂律或捕获/释放时间图等经验模型，也可能进行寿命终期漂移预测。我们的研究表明，为 Si 技术开发和验证的预测模型（简化幂律和简化热激发模型），也能非常方便地用于英飞凌的 SiC MOSFET。因此，SiC MOSFET 的 DC BTI 漂移能向 Si 技术一样进行预测。

总结

SiC 的 DC BTI 是严重影响器件可靠性的一个问题。因此，必须通过优化器件工艺来将 DC BTI 降到最小 ,并利用合适的测量方法仔细地评估 DC BTI。然而，因为能使器件性能更好（RON x A 更小）的工艺条件，在 NBTI 或 PBTI 方面不一定就表现最好，所以必须采取谨慎的态度来对待 DC BTI。英飞凌的 SiC MOSFET 具有优异的器件性能，同时还拥有很小的 NBTI，可与最先进的 Si 功率 MOSFET 相媲美。SiC 器件的 PBTI 由于带隙更大而比 Si 技术略高 ,但仍位于 100mV 的范围以内。由于观察发现 SiC 的 PBTI 与时间、温度和偏压的关系与 Si 技术类似，所以可以断定它们对应的潜在物理机制是一样的，因此可以使用与 Si 技术相同的、同样有预测能力的建模方法。

留 言

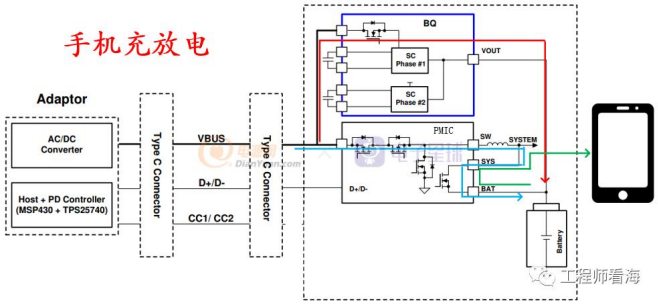
互 动

手机充、放电架构与工作流程讲解

文 / 工程师看海

电池充放电电路是手机中最关键的电路之一，是手机一切功能的源头，如果该电路出现问题会使得整个手机工作不稳定，甚至无法开机。手机的电是从电池来的，电池电压经过电源管理 IC 后，输出到各个负载，这个电源管理芯片叫做 PMIC，Power Management IC ，比如下图所示，电池的电经过 PMIC 后转换为一个叫做 system 的电，这就是手机的主电源，这个电源有的平台叫 Vsys，有的平台叫 VPH_PWR，总之万事万物都是想通的，不管叫什么电，手机来其他模块的电都是从这路电转换而来的，高端手机里有上百路电源，低端手机也有林林总总六七十路电源，都是从 Vsys 来的。

有极个别的情况是 Vsys 无法提供负载大电流要求，此时可以考虑直接从电池 Vbat 抽电，当然这是非常非常少见的应用和设计场景。



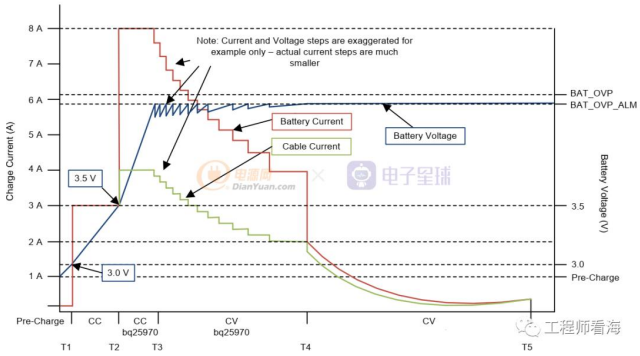
从电池的角度来看，它既放电为整个手机提供能量，也会被充电储存能量，放电时电流走的是输出路径，见上图绿色曲线路径，充电时走输入路径，见上图浅蓝色和红色路径，usb 充电线的充电电流经过 typec 连接器进来后经过 PMIC 或者辅助充电 IC 进入电池，实现充电功能；充电路径中红色的是电荷泵高功率充电，浅蓝色路径是 BUCK 低功率充电，其实把浅蓝色路径放过来就是 BOOST 升压结构，因此手机也可以升压，通过 typec 接口给其他设备用电。

有同学好奇，为什么充电还要走两个路径？

这两条充电路径一条是主充电路径，一条是辅助充电路径，辅助充电路径充电功率大，我们当前手机里的快充主要就是依靠辅助充电 IC 实现大功率充电的。

我们结合下图的充电电压电流曲线，再次深刻理解下手机充电过程，假如电池被过放，或长时间不使用，电量非常非常低，甚至低于 3.5V，下图中电池是从 3V 开始充电的，此时叫做 pre-charge 预充电，预充电过程就是主充电 IC 在工作，充电路径见上图浅蓝色曲线，USB 线缆上的电流和进入电池的电流基本一致，经过预充电后达到 T1 CC 阶段（CC 阶段是 Constant current 恒流阶段），这个阶段的特点是电池电压缓慢上升，而电流保持不变，图中的电流是稳定在 3A，而电池电压逐渐从 3V 上升到 3.5V，电池电量缓慢上升。

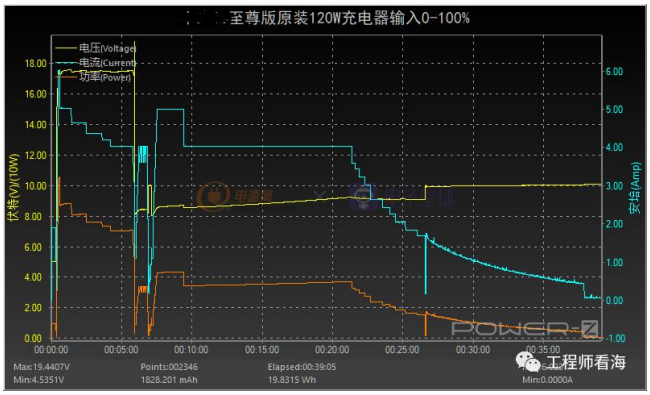
接下来到达时间 T2-T3 也是 CC 阶段，从 T2 开始，辅助充电 IC 开始介入充电过程，充电路径见上图红色曲线，此时的充电功率有了大幅变化，USB 充电线上的电流可以达到 4A，进入手机的电流是 USB 电流的 2 倍，大约是 8A，图里辅助充电 IC 是降压电荷泵充电架构，特点是电压减半，电流加倍，电池充电电流是 8A，假如电池电压是 4V，那么此时电池舜时充电功率就是 4V*8A 等于 32W，USB 提供的大约是 8V4A 也是 32W，电荷泵的原理参考以前文章：《一文理解电荷泵电源原理》。



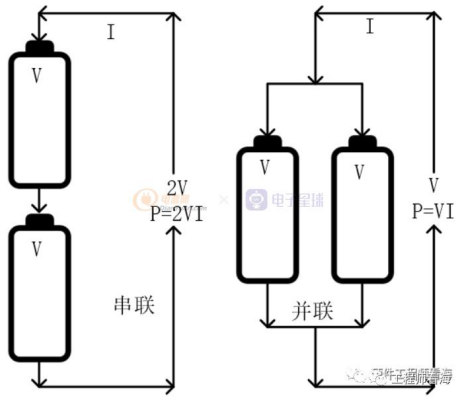
快充的持续时间是很短的，当电池到达一定程度后，充充电电流就会下降，充电过程进入 T3-T4，此时的特点是，电池电压不变，而电流逐渐降低，此时叫做 CV 过程，Constant voltage，恒压充电，不过呢，usb 电流和电池电流还是保持 2:1 的关系，此时的充电功率也不低。

T4 时间以后，充电功率就明显下降，辅助充电 IC 休息了，让主充电 IC 慢慢工作，此时是就进入 CV 阶段，电池慢慢也就充满电了。

以上就是手机充放电架构及工作流程的介绍，需要说一句的是，手机的电量和电压不是 100%正相关关系，在要求不高的场合我们可以用电池电压粗略估计电量，但是在手机这种对电量准确性要求高的场合，高精度体验友好的电量计设计是非常重要的，因此需要结合电压和电流对电量进行估计和拟合，比如有的电量计就用卡尔曼滤波估计电量，更简单点的做法是对电流积分来和电压互相补充来估计电量。此外，电池低电量时放电会特别快，不能让用户上一秒看手机还有 15%的电，下一秒就突然变成 1%了，甚至有的手机玩一玩游戏，电量反而蹦高了，这都是非常不友好的体验。



我们看下实际充电曲线，上图是某手机实测的充电曲线，黄色是 usb 电压，蓝色是 usb 电流，橙色是功率，大功率的持续时间只有 1 小段，该手机使用了更复杂的电池和充电架构设计：120W 秒充技术，它采用的是两颗电荷泵设计，将 USB 网络的 20V3A 高电压和高电流转换为两路 10V6A 电压电流，最终汇合成 10V12A 的大电流输入电池，实现 120W 高级秒充，为了实现 10V12A 电池充电，该手机使用双串电池架构，双电池串联的特点是：总电压升高、容量不变；双电池并联的特点是：总电压不变，容量升高。由于电池串联，总电压加倍，在总电流相同的前提下，串联设计将会带来更快的充电功率。



以上就是手机充电放电架构和工作流程的介绍，然而笔者更期望的还是电池技术本身的进步，容量更大、更稳定、充电更快的电池才是根本。

留　言

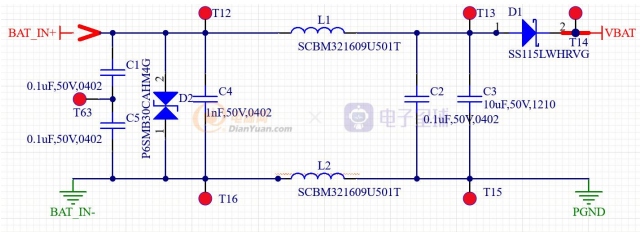
互　动

磁珠导致 BCI 测试失效的案例分析

文 / EMC 小白

磁珠其实就是单匝的线圈，也是是能量消耗元件，磁珠和电感的结构其实是类似的，一个主要区别的就是线圈匝数，选用电感主要考虑 DCR 小，电感感值稳定，抑制信号的原理是产生反向电动势，产生反向磁通，抑制原有磁通，而磁珠抑制信号主要是应用其电阻特性，因而匝数通常较少。

磁珠的主要原料为铁氧体。铁氧体是一种高磁导率，高电阻率的材料。这种材料的特点是高频损耗非常大。我们在信号传输中常利用磁珠高频特性进行滤波，如 L1,L2 所示。但是在应用磁珠时，我们通常会关注公共阻抗的耦合，而忽略了感性耦合，容性耦合。下面以一篇实例说明。



问题描述：

汽车电子控制器产品

表 1 供电特性

额定电压	电压范围	最大电流	备注
24Vdc		3A	商用车电压

表 2 接口特性

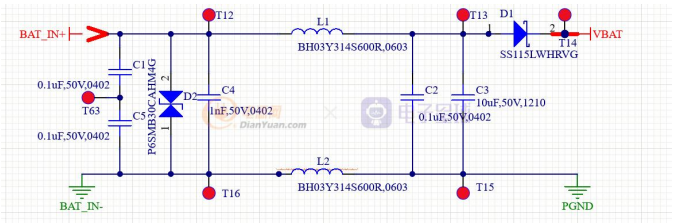
接口名称	工作电压	接口类型	电缆类型	备注
直流输入电源口	24Vdc	连接器	非屏蔽线	
Can 总线接口	3V	连接器	非屏蔽线	
采样信号线	/	连接器	非屏蔽线	

按照 GM 的 3097 BCI 测试标准测试：

频段	频率范围（MHz）
F1	1≤f≤10
F2	10≤f≤30
F3	30≤f≤80
F4	80≤f≤200
F5	200≤f≤400

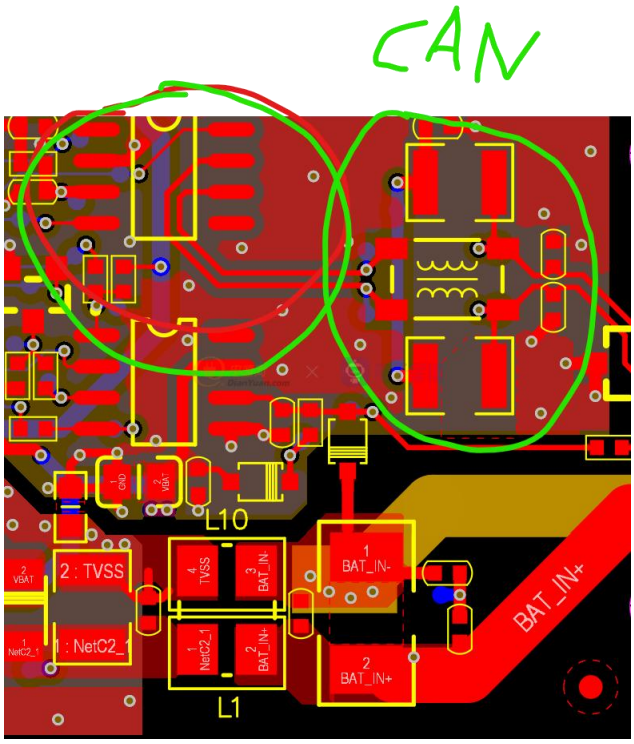
试验严酷等级	标准试验等级（mA）
L1	25
L2	50
L3	75
L4	100
X	协商值

在做 DBCI 实验时，发现 70MHz，80MHz，90MHz，通过 CAN 上位机接收数据异常。通过采用每根线束单独 BCI 测试时，发现当仅对电源线束进行 BCI 测试时，发现会出现 70MHz,80MHz,90MHz 上位机接收数据异常，而当对其他线束测试时，未发现异常，说明问题可能出现在电源线束上，从下图可以看出，当 BCI 干扰进入时，一部分干扰信号会通过电容滤波到地，一部分干扰被线束上磁珠已热能形式消耗，剩下的信号通过磁珠后进入下一级电路。

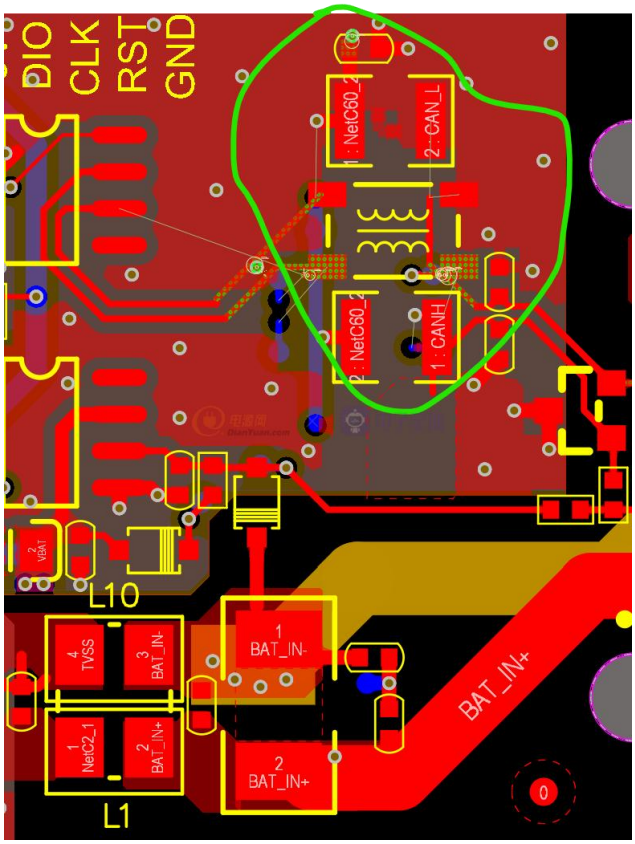


通过分析磁珠特性发现该磁珠在 100MHz 时具有 600R 的阻抗，按照正常理解，对 70MHz,80MHz，90MHz 应具有较好的抑制效果，为什么反而出现了通信异常的问题呢。

耦合通常分为公共阻抗耦合，容性耦合，感性耦合，辐射耦合，因为存在磁珠滤波，公共阻抗耦合可能性不高，剩下的就是容性耦合，感性耦合，辐射耦合，分析这部分 Layout，发现磁珠距离 CAN 电路很近。



通过减小容性耦合感性耦合方法有减小有效面积，改变介电常数，增加干扰源和被干扰源的距离，目前的情况，增加干扰源和被干扰源的距离更为切实有效：



改版后重新测试，发现在 70MHz,80MHz，90MHz 测试通过。磁珠的存在可以抑制对应频率的干扰信号能量，但

是该能量仍然会形成容性，感性耦合，辐射耦合，并攻击其他的信号线，这点我们可能会忽略掉。

留　言

互　动

电容触摸感应电路的 PCB 设计

文 / 勤劳善良的人

电容触摸感应检测的是电容值的微小变化，这种微弱的信号极易被外界干扰造成误触发，所以我们在设计中应该尤其注意元器件选择、元器件布局、线路板走线。好的设计规则是产品成功的关键，本篇介绍电容感应电路 PCB 设计中的注意事项。

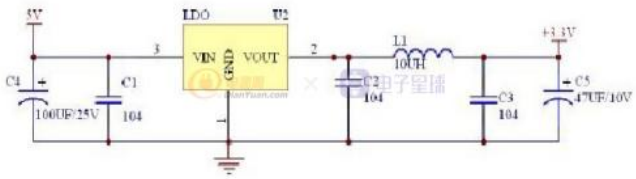
一、元器件选择

1.1 供电处理

在系统设计时应该把触摸电路部分当成是独立模块来设计。在取电时，应与其他模块并列采用“星型”连接方式，把取电处连接在输入电容 Vin、GND 根部，避免其他模块对触摸芯片电源造成干扰。

触摸芯片 VDD 要求低纹波，如果输入端纹波很大，那么需要使用 RC、LC 或者三端稳压器等稳压后再供给触摸芯片。如果成本限制的很严格，也可以分应用场合选择加或者不加。

一般情况下，应用在电池供电系统，或输入源为线性电源，此类电路可以选择不加稳压电路直接供给触摸芯片；应用在输入源为开关电源，或者系统中存在电机，或者系统中存在 RF 电路，此类电路需要加稳压电路给触摸芯片供电。



1.2 触摸面板

根据如下公式可知，影响电容大小的因素有：介质介电常数，正对面积，板间距离。

$$C = \frac{\varepsilon_0 \varepsilon_r A}{d}$$

应用设计中与以上三个影响因素对应的参数分别是：面板材料，感应盘大小，面板厚度。面板材料一般可选玻璃、亚克力、ABS，原则上是感应盘面积越大，面板厚度越薄，

得到的电容值就越大，反应就越灵敏，但是实际应用时感应盘面积不能太大，原因一是 PCB 空间不允许，原因二是太大的感应盘会导致极小的干扰造成误触发。实际应用时通常遵循以下原则：

感应电极面积	亚克力	普通玻璃	ABS
6mm×6mm	1.0mm	2.0mm	1.0mm
7mm×7mm	2.0mm	3.0mm	2.0mm
8mm×8mm	3.5mm	4.0mm	3.5mm
10mm×10mm	4.5mm	6.0mm	4.5mm
12mm×12mm	6.0mm	8.0mm	6.0mm
15mm×15mm	8.0mm	12mm	8.0mm

下面列出另外几种常用的面板材料，以供设计时参考。

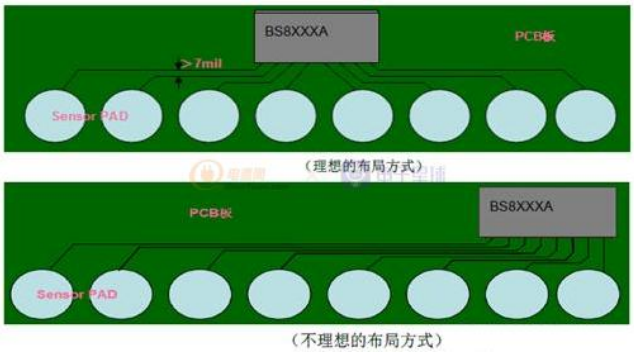
材料	介质常量
空气	1
木质	1.2~2.5
树脂玻璃	2.8
Mylar 聚脂薄膜	3.2
ABS	3.8~4.5
曝光板	4.6~4.9
玻璃（陶瓷）	6
玻璃（标准）	7.6~8.0

如果触摸面板与 PCB 板有一定距离，可以使用导电材料将感应盘延伸至面板背面，通常使用铜箔、金属片、平面弹簧、导电棉等，其中平面弹簧是最常用的，也是成本最低的。

二、元器件布局

1. 滤波电容、感应线上串联的电阻应紧贴触摸芯片放置。

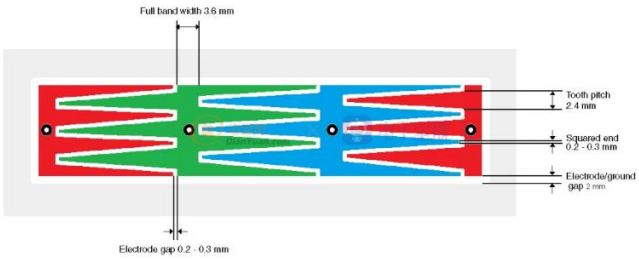
2. 如果有多个触摸感应盘，在 PCB 板空间允许的情况下，应尽量保证每个感应盘距离触摸芯片的距离相等，如下图：



3. 单键触摸感应盘可以设计成多种形状，圆形、正方形、三角形及异形，感应盘之间距离应大于 2.5mm，各感应盘大小应不小于 5mm*5mm，不大于 15mm*15mm，8mm-15mm 之间推荐使用圆形感应盘，5mm-8mm 推荐使用正方形感应盘来增大感应面积。

4. 当应用为滑条时，如果简单粗暴的使用规则感应盘排列形成一条线，要做到精细调节时就需要很多的感应盘，对应的是很多的 IO 口，对触摸芯片要求高。

因此处理滑条感应时通常采用“矢量”控制法，通过计算手指触摸导致各个传感通道增加的电容值与参考电容的比值来确定手指的具体位置，使用很少数量的感应盘即可实现多段式控制。



5. 滑轮触摸应用设计和滑条应用类似。

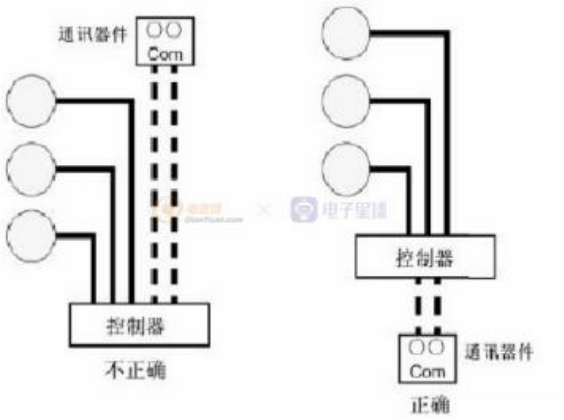
三、PCB 走线

在 PCB 工艺允许的情况下，感应盘到触摸芯片的连线应尽量短，线宽应尽量小，双面板设计时线宽 0.12mm-0.2mm（5mil-8mil）单面板设计时线宽 0.2mm-0.3mm（8mil-12mil），走线周围 1mm 不要走其他信号线，与铺地铜箔保持至少 1mm 间距；感应盘电气网络点与 PCB 板边应保持 3mm 以上距离，与金属外壳保持 5mm 以上距离。

如果把有感应盘一面定义为正面，另一面定义为反面，正面铺铜时应注意产品使用场合，铺铜会导致触摸灵敏度降低，建议铺铜与感应盘间隔大于 2mm，如果触摸面板很厚，或者应用在潮湿的环境中，建议正面感应盘周围不要铺铜。

反面铺铜通常采用网格状铜箔，铜箔面积与网格总面积比值不超过 30%，一般取线宽设置 6mil，网格尺寸 30mil，铜箔角度设置为 45 度，如果触摸面板很厚，反面铺铜可以选择不铺感应盘映射部分并保持大于 1mm 间距。

感应盘到触摸芯片的连线应尽量避免跨越强干扰、高频信号线，应远离脉冲信号，与其他信号平行走线时中间应铺地。



留　言

互　动

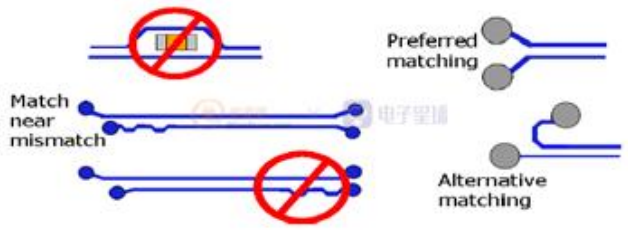
PCB 常见设计规则

文 / 广元兄

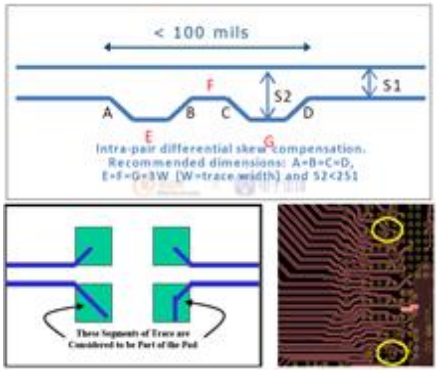
信号完整性的工作，很大一部分是对 PCB 走线规则的设定以及走线优化。规则设定得好是一个 PCB 设计的基础，也是最关键的部分。所以，设定规则来管控风险比出现风险解决来得更重要。预防管控的能力是未来信号完整性工程师的必备基础技能。预防管控 PCB 走线的风险，最基础的知识就是熟知常用走线规则。

1.线长匹配的原则是等时。考虑信号在表层和内层速率不同，所以一般建议就近匹配等长，这也就是很多资料里提到的“动态等长”。下图也给出了就近匹配的不同方式：

When trace length matching, the matching should be made as close as possible to the point where the length variation occurs, as shown in Figure 15-7, so the discontinuity won't propagate across the channel. For example, length matching in a chipset breakout area or connector pin field should occur within first 3.175 mm (125 mils) of the structure that causes mismatch.



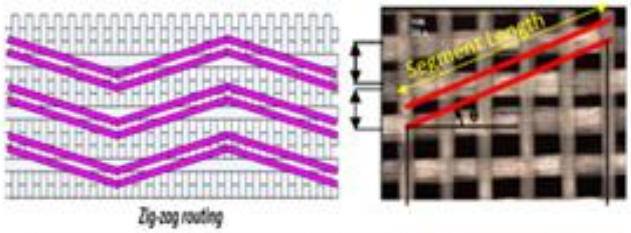
整条链路的等长匹配，如果 P/N 差别比较大，会在走线层面比较空的地方进行 3W2S，也有在 PAD 区域绕线来达到匹配。如下图所示：



2.高速信号建议走内层，就是为了做阻抗的管控。阻抗误差范围：微带线+/-15%，带状线+/-10%。

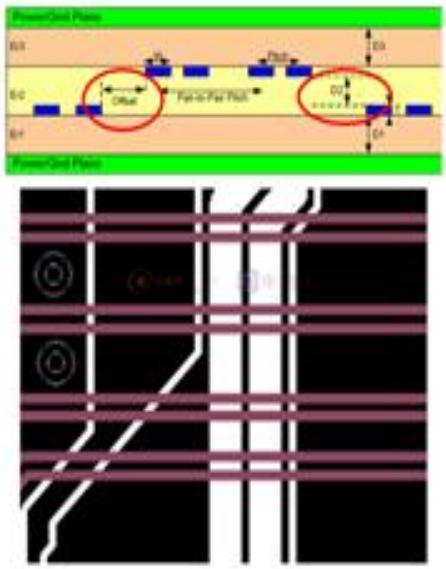
Target impedance is recommended for main route with +/-15% tolerance for Microstrip and +/-10% tolerance for Stripline/Dual Stripline.

3. 玻纤效应影响，走线锯齿状，对应信号速率走线长度的管控，如下图：

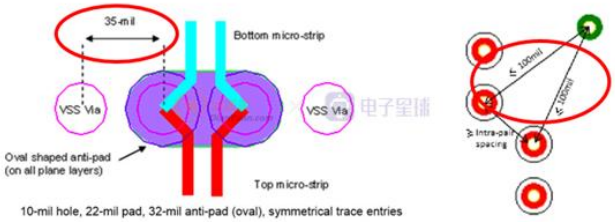


Transfer Speed	Max RSS Length in mm (see note-1)
2.5 GT/s	127
4 GT/s	101.6
5 GT/s	76.2
6.4 GT/s	63.5
8 GT/s	50.8
10 GT/s	50.8
16 GT/s	25.4
20 GT/s	12.7

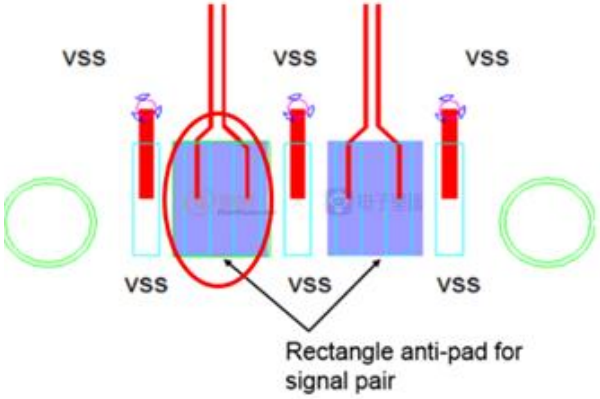
4. 消费类的一些产品会选择走双带线。双带线(Dual Stripline) 上下错开，不重复(Offset > 0)，最好是垂直走线。D2(4~9mil)。

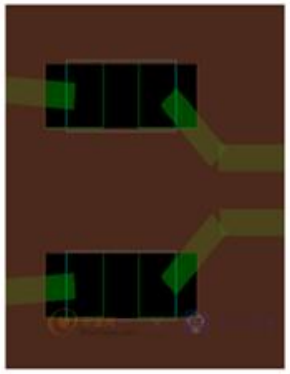


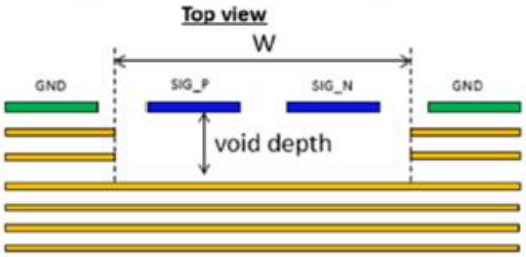
5. 过孔处阻抗突变的优化 ,过孔阻抗和反焊盘的大小、孔径、焊盘大小都有关系。参考层面发生改变，GND 回流孔距离上也要注意，有的资料给出的距离≤100mil。



6. SMT 连接器的处理方式，有的是相邻层挖空，也有推荐挖空 15mil，一般是往下两层走线层，但要注意第四层是否是地层参考的问题。







当然，PCB 设计规则不仅仅是这些走线规则的问题，从产品开始的叠层设计就需要管控，避免非对称叠层，板材的选择（PP&CORE），铜箔类型等等，再到走线层面规划：高速信号优先，高速信号残桩问题（高速产品），双带线模型走线问题（消费类产品）。

任何一款好产品的设计不是一个点，而是多个点的，不是一方面，而是多方面的优化结果。

留　言

互　动

5V，2A 反激变换器设计

文 / electronicLee

开关电源的设计是一个迭代过程，其中涉及许多变量，必须调整这些变量以获得优化的解决方案。然而，为了实现简单的低成本、低组件、单面板设计方法，需要权衡取舍。 本方案提供了一种使用 NCP1055 高压开关稳压器设计转换器的简单方法。易于遵循的分步电源设计，主要是输入模块、功率级、磁性元件、缓冲器、输出模块和反馈回路。 该电源要求 5.0 V、2.0 A 输出和 48%的最大占空比。符合 IEC 和 UL 要求。EMI 性能好，可实现 70%或更高的效率。这个功率段适合于辅助电源供电系统，满足绝大多数应用场合。

设计电源的第一步是定义和预先确定输入和输出参数。

通用输入电压范围：

$$V_{in(min)} = 85\ VAC,\ V_{in(max)} = 265\ VAC$$

输出：

$$V_{out} = 5.0\ V \pm 2\%,\ I_{out} = 2\ A$$

输入功率：

$$P_{in} = \frac{P_{out}}{est.eff}$$

$$P_{out} = 5 \cdot 2.0 = 10\ W$$
$$\therefore P_{in} = \frac{10}{0.78} = 12.82\ W$$

0.78 的效率是 Si MOS 反激式转换器的一般效率段。

低压和高压交流时的直流母线电压：

$$V_{peak(min)} = V_{in(min)} \cdot \sqrt{2} = 85 \cdot \sqrt{2} = 120.21\ VDC$$
$$V_{peak(max)} = V_{in(max)} \cdot \sqrt{2} = 265 \cdot \sqrt{2} = 374.77\ VDC$$

低压时平均电流：

$$I_{in(avg)} = \frac{P_{in}}{V_{in(low)}}$$

$$V_{in(low)} = V_{peak(min)} - V_{ripple} - V_{diode}$$

$$V_{ripple} = 32\%\ V_{peak(min)}$$

$$I_{in(avg)} = \frac{12.82}{80.2} = 0.160\ A$$

输入峰值电流：

$$I_{peak} = 2 \cdot I_{in(avg)} \cdot \frac{t_{sw}}{t_{on}}$$
$$I_{peak} = 2 \cdot 0.160 \cdot \frac{10\ \mu s}{4.8\ \mu s} = 0.667\ A$$

可以使用以下公式对器件损耗进行评估：

$$P_{loss} = P_{in}\ (1 - eff) \cdot P\%$$

其中 P%是总电源损耗中所需电路部分的损耗百分比。

通常，35%的损耗来自功率 MOSFET，60%来自输出整流器，5%来自磁性元件，5%来自其他来源。

基础器件选择

保险丝

保险丝 F1 保护电路免受开启时发生的电流浪涌。在本方案中，F1 的额定电流为 2.0 A、125 VAC。

EMI 滤波器

EMI 滤波器用于抑制共模和差模噪声，并且很大程度上取决于电路板布局、组件选择等。X 电容器 C1 和共模扼流圈 L1 放置在 AC 线路上以衰减差模噪声。EMI 电感器正在减缓任何瞬态电压浪涌以降低高频噪声。电容器和扼流圈都应放置在二极管电桥之前，并尽可能靠近 AC 线路输入，以尽量减少 RFI。

整流桥

为了选择合适的二极管桥式整流器，必须考虑正向和浪涌电流以及直流阻断电压的值。 浪涌电流可以达到平均输入有效值电流的五倍。因此，有必要选择能够处理如此大电流的整流器。

高压交流输入时，直流母线最高电压：

$$V_R \geq V_{peak(max)} = V_{in(max)} \cdot \sqrt{2} = 375\ \text{VDC}$$

导通电流值：

$$I_F \geq 1.5 \cdot I_{in(avg)} = 1.5 \cdot 0.160 = 0.240\ \text{A}$$

浪涌电流值：

$$I_{FSM} \geq 5 \cdot I_F = 5 \cdot 0.240 = 1.2\ \text{A}$$

输入电容：

输入大容量电容器 C2 的目的是保持整流后的线路电压并滤除共模噪声。它位于桥式整流器输出和地之间。大容量电容器的大小取决于峰值整流输入电压和纹波电压幅度。较大的电容会降低直流输入线上的纹波电压，但会在电源上电时产生较大的浪涌电流。假设纹波幅度约为低压线路峰值整流电压的 32%，则可以使用以下公式计算 Cbulk：

$$C_{bulk} = \frac{P_{in}}{fac \cdot (V_{peak(min)}^2 - V_{in(low)}^2)} = \frac{12.82}{60 \cdot (120^2 - 80.2^2)} = 27\ \mu\text{F}$$

选择最接近的低 ESR 的 33uF 标准电容器。铝电解是首选，因为它们坚固且可靠性高。

磁性器件计算

下一步是反激变压器的设计。磁的设计是整个设计过程中最重要和最精细的部分，因为它将决定电源的性能如何。 反激式变压器首先在初级绕组中激磁电流，从而将能量存储在变压器的励磁电感中。然后，当初级侧关闭时，励磁电感能量被传输到次级绕组。本方案用的是标准的 EFD20 尺寸。

为了使稳压器在更坏的情况下以非连续模式运行并最大化功率，最大导通时间为整个周期的 48%，因此最大初级电感是根据 48%的最大占空比计算的。使用比计算值更大的电感会导致电源输出超出调节范围。

$$L_{pri} = \frac{V_{in(low)} \cdot \partial\ max}{I_{peak} \cdot f_{op}}$$

$$L_{pri} = \frac{80.2 \cdot 0.48}{0.667 * 100 \cdot 10^3} = 0.577\ \text{mH}$$

反射电压：

$$V_{FB} = \frac{V_{in(low)} \cdot t_{on}}{t_{off}} = \frac{80.2 \cdot 4.8 \cdot 10^{-6}}{5.2 \cdot 10^{-6}} = 74.03\ \text{V, where } t_{on} \text{ is } 4.8\ \mu\text{s}$$

匝比：

$$\frac{N_{pri}}{N_{sec}} = \frac{V_{FB}}{V_{out} + V_F} = \frac{74.03}{5 + 0.525} = 13.4 \approx 13\ \text{turns}$$

通过重新计算校验上述方程并求解 Nsec 得到 ≈1 匝。

导通期间进入储存再原边励磁电感中的能量（当电源开关导通时）：

$$E_{stored} = \frac{L_{pri} \cdot I_{peak}^2}{2}$$

$$E_{stored} = \frac{0.577 \cdot 10^{-3} \cdot 0.667^2}{2} = 1.28 \times 10^{-4}\ \text{Joules}$$

可以使用以下等式仔细检查变压器的功率能力是否足够大以向输出提供足够的功率：

$$P_{in(core)} = \frac{L_{pri} \cdot I_{peak}^2}{2} \cdot f_{op} > P_{out}$$

$$P_{in(core)} = 1.28 \times 10^{-4} \cdot 100\ \text{kHz} = 12.8\ \text{W} > P_{out} = 10\ \text{W}$$

输出计算：

输出由二极管整流器、pi 滤波器和电压调节器组成。 对于输出电压低于 7.5 V 的反激式转换器，肖特基整流器可提

供最高效率，因此是最佳选择。 使用的肖特基是 1N5822，其中 VR = 40 V，IF = 3.0 A，VF = 0.525 V。这个整流的主要目的是取次级电压并将其转换为直流电压。 以下等式用于选择肖特基整流管：

最大反向峰值电压（在高压时计算）：

$$V_{Rout} > V_{out} + \left[V_{peak(max)} \cdot \frac{N_{sec}}{N_{pri}}\right]$$

$$V_{Rout} > 5 + \left[375 \cdot \frac{1}{13}\right] = 33.85\ V$$

对于不连续模式，最大正向峰值电流可以使用以下公式进行近似：

$$I_{Fout} = 4 \cdot I_{out} = 8\ A$$

二极管 D6 与 C7、C8、C9、L2 和 C11 一起对变压器次级进行整流并对输出进行滤波，以提供严格调节的直流输出。 电容 C7、C8 和 C9 并联放置以降低 ESR。 此外，C7、C8 和 C9 的额定电压应足够高，以使其能够承受电压尖峰和输出电压。 L2 和 C11 构成一个低通滤波器，可衰减高频噪声。

输出滤波电容：

$$C_{out} = \frac{I_{out(max)} \cdot T_{off(max)}}{V_{ripple(desired)}}$$

$$V_{ripple(desired)} = 40\ mV$$

$$C_{out} = \frac{8 \cdot 0.52 \cdot \frac{1}{100 \cdot 10^3}}{0.040} = 1040\ \mu F$$

输出滤波器扼流圈（设计用于 4.0 kHz 的截止频率）：

$$L = \left[\frac{1}{2 \cdot \pi \cdot f \cdot \sqrt{C}}\right]^2$$

$$C = C11 = 330\ \mu F$$

$$L = \left[\frac{1}{2 \cdot \pi \cdot 4k \cdot \sqrt{330\ \mu F}}\right]^2 = 4.8\ \mu H$$

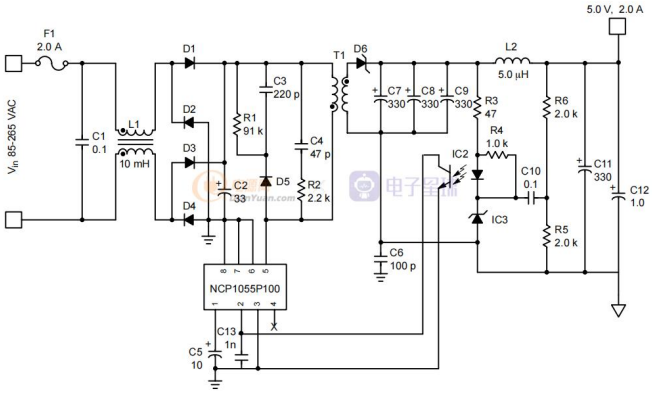
反馈环路：

反馈回路由光耦、431、补偿电容和电阻分压器组成。光耦合器将交流输入与直流输出隔离开来。TL431 用于调节输出电压。该 431 电压基准可使用两个外部电阻器在 Vref 至 36 V 范围内进行编程。它具有 1.0 mA 至 100 mA 的宽电流范围，是齐纳二极管的绝佳替代品。TL431 的参考电压设置为 2.5 V，输出电压为 5.0 V，通过电阻分压器 R5 和 R6（低容差, 2.0 k 电阻）。 TL431 监控 5.0 V 输出电压并将分压电压与其 2.5 V 内部基准电压进行比较。输出电压的小幅增加将导致并联稳压器开始导通，从而通过光耦合器的 LED 吸收电流。反过来，光耦合器晶体管变为正向偏置并开始驱动电流进入 NCP1055 的控制输入引脚。然后相应地调整电源开关占空比。 0.1 uF 的补偿电容器 C10 放置在 TL431 的阴极和参考引脚之间，以提高稳定性。电阻器 R3 将通过光耦合器的电流限制在安全水平，并防止损坏光耦合器。

RCD 电路：

由于功率晶体管漏极电压和变压器漏感的高 dv/dt 特性，当电源开关关闭时，漏极处会出现电压尖峰和振铃。 电阻器 R1、C3、D5 会影响 RCD 缓冲器。 与初级绕组并联的是 R2 和 C4，它们构成一个 RC 振铃阻尼器，可减慢 dv/dt 并降低峰值电压，从而减少高频噪声引起的振铃。 由于 i =C · dv/dt，增加电容也会降低电压纹波的幅度。 缓冲器和振铃阻尼器共同作用以保护 IC 免受大于 700 V 的瞬态电压影响，并降低辐射噪声。

最终原理图：



留 言

互 动

Linux 驱动 | 利用 DRIVER_ATTR 实现调试内核驱动方法

文 / 一口 Linux

1. 前言

很多朋友在调试驱动的时候，都会遇到这样一个场景：修改一个参数，然后调用某个内核中的函数，比如将某个 gpio 的值拉高/拉低，修改某个寄存器的值等等。

如果每一个参数都通过字符设备的 ioctl 接口，增加对应的 cmd，会比较麻烦，研究内核的计算机大牛们怎么会容忍这种事发生，于是设计出了 DRIVER_ATTR 这个宏，完美解决这个问题。下面一口君通过一个简单的实例，给大家讲解如何使用 DRIVER_ATTR。

2. DRIVER_ATTR 定义

该宏定义的文件如下：include/linux/device.h

```
struct driver_attribute {
    struct attribute attr;
    ssize_t (*show)(struct device_driver *driver, char *buf);
    ssize_t (*store)(struct device_driver *driver, const char *buf,
                     size_t count);
};
#define DRIVER_ATTR(_name, _mode, _show, _store) \
    struct driver_attribute driver_attr_##_name = __ATTR(_name, _mode, _show, _store)
```

__ATTR 定义于文件 include/linux/sysfs.h

```
#define __ATTR(_name, _mode, _show, _store) {
                \
    .attr = {.name = __stringify(_name), .mode = _mode },    \
    .show    = _show,                                        \
    .store   = _store,                                       \
}
```

说明：

_name：名称，也就是将在 sys fs 中生成的文件名称。

_mode：上述文件的访问权限，与普通文件相同，UGO 的格式，最高权限 0644，否则会报错。

_show：显示函数，cat 该文件时，此函数被调用。

_store：写函数，echo 内容到该文件时，此函数被调用。

3. 使用步骤

定义一个写操作的回调函数：

```
static ssize_t peng_test_store(struct device_driver *driver,
                               const char *buf, size_t count)
{
//对参数进行检查
    if(NULL == buf || count >255 || count == 0 || strnchr(buf, count, 0x20))
        return -1;

    printk("buf:%s count:%d\n",buf,count);

    return count;
```

声明该函数与文件节点关系：

```
static DRIVER_ATTR(peng, 0644, NULL, peng_test_store);
```

创建文件节点：

```
ret = driver_create_file(&(hello_driver.driver), &driver_attr_peng);
    if (ret < 0){
        dev_err(&pdev->dev, "could not create sysfs files\n");
        ret = -ENOENT;
    }
```

这几个名字之间关系如下：

4. 源码

本实验代码分为两个模块 device、driver， 分别定义结构体 platform_device、platform_driver 并注册到 platform 总线。完整源码如下：

device.c

```
#include <linux/init.h>
#include <linux/module.h>
#include <linux/platform_device.h>
#include <linux/ioport.h>
static void hello_release(struct device *dev)
{
	return;
}
static struct platform_device hello_device =
{
	.name = "duang",
	.id = -1,
	.dev.release = hello_release,
};
static int hello_init(void)
{
	printk("hello_init \n");
	return platform_device_register(&hello_device);

}
static void hello_exit(void)
{
	printk("hello_exit \n");
	platform_device_unregister(&hello_device);
	return;
}
MODULE_LICENSE("GPL");
module_init(hello_init);
module_exit(hello_exit);
```

driver.c

```
#include <linux/init.h>
#include <linux/module.h>
#include <linux/kdev_t.h>
#include <linux/fs.h>
#include <linux/cdev.h>
#include <linux/device.h>
#include <asm/io.h>
#include <linux/platform_device.h>
#include <linux/ioport.h>


static int hello_probe(struct platform_device *pdev);
static int hello_remove(struct platform_device *pdev);


static ssize_t peng_test_store(struct device_driver *driver,
                               const char *buf, size_t count)
{
	if(NULL == buf || count >255 || count == 0 || strnchr(buf, count, 0x20))
		return -1;

	printk("buf:%s count:%d\n",buf,count);

	return count;
}
static DRIVER_ATTR(peng, 0644, NULL, peng_test_store);

static struct platform_driver hello_driver =
{
	.probe = hello_probe,
	.driver.name = "duang",
	.remove = hello_remove,
};

struct resource *res;
static int hello_probe(struct platform_device *pdev)
{
	int ret;
	printk("match ok \n");
```

```
	ret = driver_create_file(&(hello_driver.driver), &driver_attr_peng);
	if (ret < 0){
		dev_err(&pdev->dev, "could not create sysfs files\n");
		ret = -ENOENT;
	}


	return 0;
}
static int hello_remove(struct platform_device *pdev)
{
	printk("hello_remove \n");
	return 0;
}

static int hello_init(void)
{
	printk("hello_init \n");
	return platform_driver_register(&hello_driver);
}
static void hello_exit(void)
{
	printk("hello_exit \n");
	platform_driver_unregister(&hello_driver);
	return;
}
MODULE_LICENSE("GPL");
module_init(hello_init);
module_exit(hello_exit);
```

Makefile

```
ifneq ($(KERNELRELEASE),)
obj-m:=device.o driver.o
else
KDIR :=/lib/modules/$(shell uname -r)/build
#KDIR :=/home/peng/linux-3.14
PWD  :=$(shell pwd)
all:
	make -C $(KDIR) M=$(PWD) modules
clean:
	rm -f *.ko *.o *.mod.o *.symvers *.cmd  *.mod.c *.order
endif
```

5. 编译运行

第一步，编译

```
root@ubuntu:/rootfs/code/attr# make
make -C /lib/modules/3.2.0-29-generic-pae/build M=/rootfs/code/attr modules
make[1]: 正在进入目录 `/usr/src/linux-headers-3.2.0-29-generic-pae'
  CC [M]  /rootfs/code/attr/driver.o
  Building modules, stage 2.
  MODPOST 2 modules
  CC      /rootfs/code/attr/driver.mod.o
  LD [M]  /rootfs/code/attr/driver.ko
make[1]:正在离开目录 `/usr/src/linux-headers-3.2.0-29-generic-pae'
```

第二步，加载模块驱动

```
root@ubuntu:/rootfs/code/attr# insmod device.ko
root@ubuntu:/rootfs/code/attr# insmod driver.ko
root@ubuntu:/rootfs/code/attr# dmesg
[32012.949347] hello_init
[32016.210118] hello_init
[32016.210127] match ok
```

第三步，查看生成的文件节点

```
root@ubuntu:/sys/bus/platform/drivers/duang# ls
bind  duang  peng  uevent  unbind
```

第四步，通过下面命令向节点输入一个数字（要管理员权限）

echo 1 > peng

```
root@ubuntu:/sys/bus/platform/drivers/duang# echo 1 > peng
root@ubuntu:/sys/bus/platform/drivers/duang# dmesg
[32012.949347] hello_init
[32016.210118] hello_init
[32016.210127] match ok
[32071.088055] buf:1
[32071.088056]  count:2
```

由结果可知，我们通过向文件 peng 写入一个字符，实现了调用函数 peng_test_store()，并且字符 1 传递给了参数 buf，字符个数传递给了 count。

其中目录 duang 是由结构体变量 hello_driver 给出：

```
static struct platform_driver hello_driver =
{
	.driver.name = "duang",
};
```

6. 一次注册多个节点

需要借助结构体

struct attribute 以及函数

```
/**
 * sysfs_create_group - given a directory kobject, create an attribute group
 * @kobj:The kobject to create the group on
 * @grp: The attribute group to create
 *
 * This function creates a group for the first time.  It will explicitly
 * warn and error if any of the attribute files being created already exist.
 *
 * Returns 0 on success or error.
 */
int sysfs_create_group(struct kobject *kobj,
                       const struct attribute_group *grp)
```

此处就不验证了，直接从内核找个例子给大家学习下吧。

drivers\input\touchscreen\ads7846.c

```
static ssize_t ads7846_pen_down_show(struct device *dev,
                                     struct device_attribute *attr, char *buf)
{
        struct ads7846 *ts = dev_get_drvdata(dev);

        return sprintf(buf, "%u\n", ts->pendown);
}

static DEVICE_ATTR(pen_down, S_IRUGO, ads7846_pen_down_show, NULL);

static ssize_t ads7846_disable_show(struct device *dev,
                                    struct device_attribute *attr, char *buf)
{
        struct ads7846 *ts = dev_get_drvdata(dev);

        return sprintf(buf, "%u\n", ts->disabled);
}

static ssize_t ads7846_disable_store(struct device *dev,
                                     struct device_attribute *attr,
                                     const char *buf, size_t count)
{
        struct ads7846 *ts = dev_get_drvdata(dev);
        unsigned int i;
        int err;

        err = kstrtouint(buf, 10, &i);
        if (err)
                return err;

        if (i)
                ads7846_disable(ts);
        else
                ads7846_enable(ts);

        return count;
}
static DEVICE_ATTR(disable, 0664, ads7846_disable_show, ads7846_disable_store);

static struct attribute *ads784x_attributes[] = {
        &dev_attr_pen_down.attr,
        &dev_attr_disable.attr,
        NULL,
};

static struct attribute_group ads784x_attr_group = {
        .attrs = ads784x_attributes,
};
```

```
        err = sysfs_create_group(&mydevice->dev.kobj, &ads784x_attr_group);
```

7. 补充

当然_ATTR 不是独生子女，他还有一系列的姊妹__ATTR_RO 宏只有读方法，__ATTR_NULL 等等。

如对设备的使用	DEVICE_ATTR
对驱动使用	DRIVER_ATTR
对总线使用	BUS_ATTR
对类别（class）使用	CLASS_ATTR

大家以后再调试驱动的时候，别忘了还有这些宏可以使用。

留　言

互　动

基于嵌入式的软件追踪技术（上）

文 / 程序小白

软件追踪技术主要应用在软件调试阶段上，我觉得这是一个非常牛的东西。学习 QP 至今，他是最让我震惊的两门技术之一。因为他是基于嵌入式方案的软件追踪，也就是说他在单片机上就能跑起来。当你在裸奔的年代，做着一些较为简单的应用，他会显得有些鸡肋，因为你可以让你的所有代码都跑在你的小脑袋里。但当你感觉脑袋不够用的时候，你就可以考虑用上这个东西了。

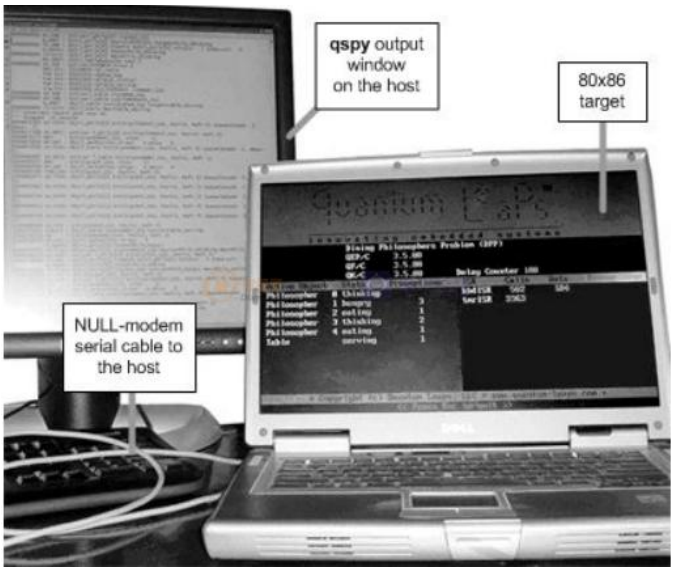
用传统方式调试代码。所谓的调试就是先让他跑起来，然后在既定输入下条件下，看看输出对不对。这种情况下，有个最强大的武器就是断点-单步调试，他非常好用。特别适合在一些裸奔或者顺序执行的情况下，因为这些操作虽然看不见，但是有一点能保证，那就是他是顺序执行的。

在多任务或者是事件驱动型系统中，单步调试的劣势就被放大了。因为系统在运行时，单步调试会带来一个最大的麻烦就是系统被暂停了。因为这种情况下你很难捕捉在某一时刻到底发生了什么样的交互（这种交互在裸机里面是不存在的，因为没有那一坨看不懂的 os 造成的一个既定的指向），才导致代码执行到了这里。这个时候因为很难用小脑袋去模拟软件执行过程多个对象交互过程中真实的情况，所以你需要想办法动态的去调试。printf 是一个非常不错的选择，但是很可惜，他并不适用于单片机（因为有很多应用是不带屏幕的，同样 printf 会造成系统性能的大幅下降，本身硬件就不怎么快）。

软件追踪技术相当于针对嵌入式处理器的应用，在printf方案上的一个超级版本进阶。他不光解决了单片机没有屏，和性能执行的问题，最关键的是，他能告诉你需要追踪啥？（哪些东西才是真正需要追踪的，通过追踪记录，可以完美的展现你的代码执行轨迹，他们到底是干了啥，怎么干的，有没有出格。）

QSPY 有两部分组成，一个是所谓的主机，就是一个上位机软件，这个大佬已经帮忙写好，并把源代码开放给了你。另一部分是需要内置到的代码工程当中的宏，在你不用QSPY的时候，直接禁用掉相应的宏就可以了。他们如何通信，其实形式有很多，完全取决你的硬件。最常用的办法就是一根串口线。

书本上讲的大概是这样，他在一台电脑上启动 QSPY 主机，另一台电脑上模拟嵌入式目标设备，中间一根线通信，如下：



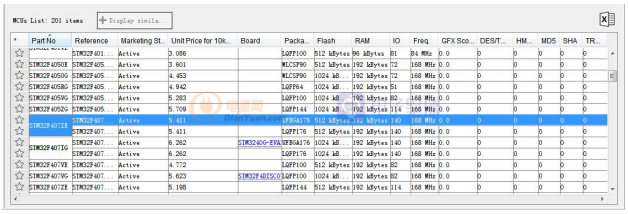
如何让其在自己的开发板上跑起来，并且根据我们内置的追踪记录一条一条的去分析一个真实的系统是如何执行的，是这篇文章的重点中的重点。因为内容有点长就分成上下两篇吧。上篇讲移植，下篇讲追踪分析。我的内容可能和书本的内容不一致，因为书本上的内容过期很久了，所以此篇就以 QP6.9.0 的真实版本作为实例去分析。

首先祭出我们的目标板，这是参加研讨会时收到的 F4 的开发板：

板子有了，先调我们的目标板的代码，主机部分的代码基本不需要我们改，因为这颗芯片是 F401，没有合适的 demo，所以就以 QP 提供的 F407 的 demo 为模板进行移植，说是移植，基本不怎么用改就能跑起来。先看看源码哪里需要改。

在调试的过程中，我遇到了两个问题，一个是主频的问题，因为芯片不同，所以，一定要当心看 F401 和 F407 之间到底有哪些区别。找区别，我不太习惯看芯片手册，还是 CUBEMX 用来顺手，熟悉 STM32 小伙伴应该都懂得。我在这个地方栽了一个跟头。

```
/*****************************************************************************/

/************************* PLL Parameters ************************************/
/* PLL_VCO = (HSE_VALUE or HSI_VALUE / PLL_M) * PLL_N */
#define PLL_M      8
#define PLL_N      336

/* SYSCLK = PLL_VCO / PLL_P */
#define PLL_P      8//2

/* USB OTG FS, SDIO and RNG Clock =  PLL_VCO / PLLQ */
#define PLL_Q      7

/*****************************************************************************/
```



图片可能看不太清，这里说一下，401 最快是 84M 的主频，407 最快是 168M 的主频，都是 F4，差别有点大，因为我们直接用的是 407，所以要仔细去看看时钟配置。

继续刚讲到的主频，先看一下 demo 的时钟是多少。一般来讲，demo 都是时钟有多快，我就配多快。代码如下截图：

```
if (HSEStatus == (uint32_t)0x01)
{
  /* Select regulator voltage output Scale 1 mode, System frequency up to 168 MHz */
  RCC->APB1ENR |= RCC_APB1ENR_PWREN;
  PWR->CR |= PWR_CR_VOS;

  /* HCLK = SYSCLK / 1*/
  RCC->CFGR |= RCC_CFGR_HPRE_DIV1;

  /* PCLK2 = HCLK / 2*/
  RCC->CFGR |= RCC_CFGR_PPRE2_DIV2;

  /* PCLK1 = HCLK / 4*/
  RCC->CFGR |= RCC_CFGR_PPRE1_DIV4;

  /* Configure the main PLL */
  RCC->PLLCFGR = PLL_M | (PLL_N << 6) | (((PLL_P >> 1) -1) << 16) |
                 (RCC_PLLCFGR_PLLSRC_HSE) | (PLL_Q << 24);

  /* Enable the main PLL */
  RCC->CR |= RCC_CR_PLLON;

  /* Wait till the main PLL is ready */
  while((RCC->CR & RCC_CR_PLLRDY) == 0)
  {
```

果不其然是168Mhz，这里我们要改下把他降下来，降到84M以内就好了，不然程序是跑不起来的。

```
-nucleo\qv\bsp.c(144) :     /* NOTE: SystemInit() already called from the startup code
ystem_stm32f4xx.h(80) : extern void SystemInit(void);
ystem_stm32f4xx.c(14) :   *      - SystemInit(): Setups the system clock (System clock source, PLL Mul
ystem_stm32f4xx.c(30) :   *   Then SystemInit() function is called, in "startup_stm32f4xx.s" file, to
ystem_stm32f4xx.c(33) :   * 3. If the system clock source selected by user fails to startup, the Syste
ystem_stm32f4xx.c(190) :   static void SystemInit_ExtMemCtl(void);
ystem_stm32f4xx.c(208) : void SystemInit(void)
ystem_stm32f4xx.c(234) :   SystemInit_ExtMemCtl();
```

时钟初始化在这个函数

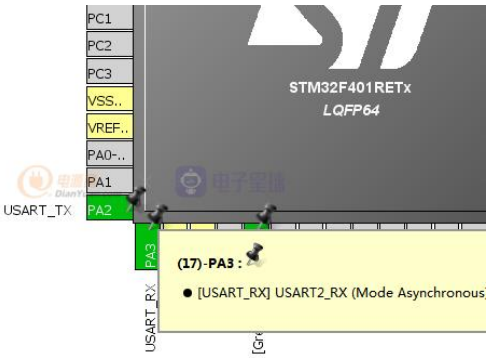
找到 PLL 相关参数，进行修改，降低主频，如下：

```
/*****************************************************************************/

/************************* PLL Parameters ************************************/
/* PLL_VCO = (HSE_VALUE or HSI_VALUE / PLL_M) * PLL_N */
#define PLL_M      8
#define PLL_N      336

/* SYSCLK = PLL_VCO / PLL_P */
#define PLL_P      8//2

/* USB OTG FS, SDIO and RNG Clock =  PLL_VCO / PLLQ */
#define PLL_Q      7

/*****************************************************************************/
```

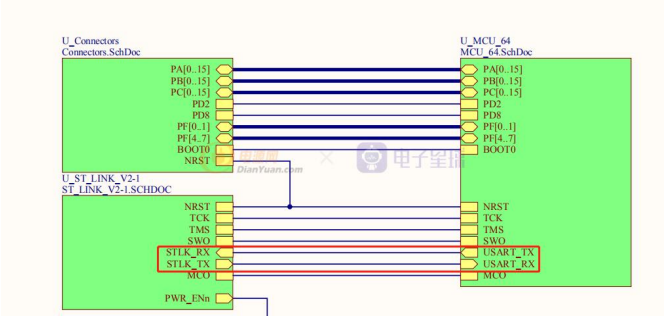
时钟改完了 ,接下来我们看看 demo 中 QSPY 是如何发送数据的。

```
void QV_onIdle(void) { /* CATION: called with interrupts DISABLED, NOTE01 */
    LED_GPIO_PORT->BSRRL = LED6_PIN; /* turn LED on  */
    __NOP(); /* wait a little to actually see the LED glow */
    __NOP();
    __NOP();
    __NOP();
    LED_GPIO_PORT->BSRRH = LED6_PIN; /* turn LED off */

#ifdef Q_SPY
    QF_INT_ENABLE();
    QS_rxParse();  /* parse all the received bytes */

    if ((USART2->SR & USART_FLAG_TXE) != 0) { /* TXE empty? */
        uint16_t b;

        QF_INT_DISABLE();
        b = QS_getByte();
        QF_INT_ENABLE();

        if (b != QS_EOD) {  /* not End-Of-Data? */
            USART2->DR  = (b & 0xFFU);  /* put into the DR register */
        }
    }
#elif defined NDEBUG
    /* Put the CPU and peripherals to the low-power mode.
    * you might need to customize the clock management for your application,
    * see the datasheet for your particular Cortex-M MCU.
    */
```

从 Q_SPY 部分代码中可以看到 ,其默认为 UART2 串口发送，所以要让我们的板子能够和他通信，就要将 UART2 串口接入主机。这里我找了原理图，看了当前 UART2 的配置。



UART2 为 PA2/PA3 引脚，看一下原理图是怎么接的，这个板子自带 stlinkV2 默认将 F401 的 uart2 和 stlink 接到了一起。



留 言

互 动

一招教你单片机固件快速瘦身

文 / 小麦大叔

1.前言

我们平时做项目的时候，随着代码量的增加，工程变得更加臃肿，但是可能实际上只使用到其中一部分函数，与此同时，还有一部分是已经定义但是没有被使用的函数，虽然我们不使用这些功能和函数 ,但它们往往会浪费我们的 ROM 和 RAM 的空间。

在使用静态库的时候，这种现象更加明显。比如，我们只需要使用静态库中的几个功能，但是编译器默认会把整个静态库全部链接到可执行程序中，从而导致可执行程序的大小大大增加。

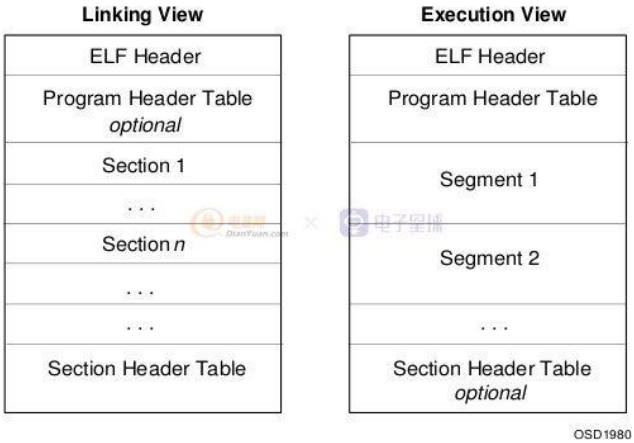
那该如何避免这种情况呢？把大部分工作交给编译器。我们只要告诉编译器不要把这些程序编译到可执行文件中即可。接下来我会详细解释。

2.ELF 格式

ELF（Executable and Linkable Format）是可执行和可链接格式。在 Linux 上 ELF 包括了链接过程中的目标文件（.o），共享库（.so）和可执行文件，同时还用于可加载的内核模块，因此作为链接过程中的目标文件也是通过 ELF 格式的文件来表示的。

ELF 的结构至少包含两个头：ELF 头、程序头，通常还会有一个节标头。

具体如下图所示：

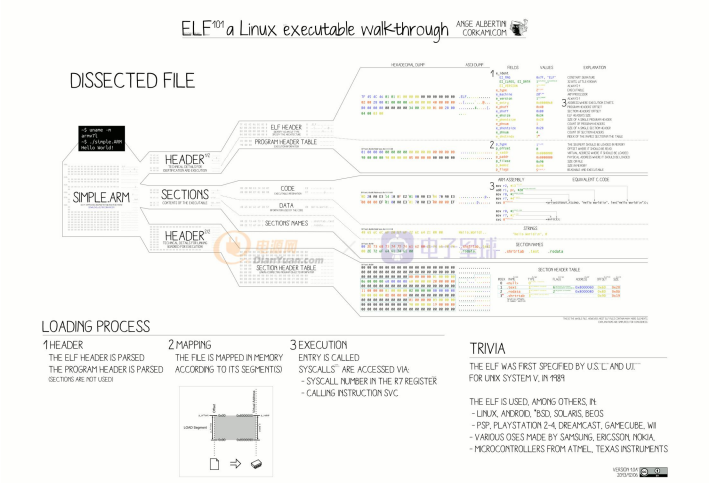


从图中我们可以看到，这里可以分为两种情况：

Linking：链接是按节划分的，在链接程序或库时会这样使用，这些部分包含一些目标文件信息，例如：数据，指令，重定位信息，符号，调试信息等等；

Exection：程序执行期间使用按段划分的执行视图。

下面是 ELF 格式文件的详细布局图：



3.编译器

通常在做 ARM 开发的时候 ,我们会使用到 ARMCC 和 GCC。这时可以参考相应编译器的手册，使用相应的编译命令就可以实现对程序的优化。

3.1 ARMCC

在 ARMCC 中，编译器通常将函数和数据放在一起，并且将每一个类别规整到同一个 section 中，如果在链接的时候发现某个 section 没有被使用，那么就会将这个 section 删除，从而减少可执行文件的大小。

可以使用--split_sections 编译器命令行选项来指示编译器为源文件中的每个函数生成一个 ELF 节，这样在链接的时候可以通过--remove 命令让链接器删除未使用的 section。

3.2 GCC

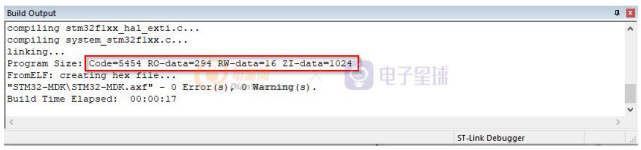
GCC 在编译时可以使用 -ffunction-sections 和 -fdata-sections 将每个函数或符号创建为一个 section； 链接阶段的时候，使用 -Wl，–gc-sections 来告诉链接器删除不需要的 section（其中-Wl, 表示后面的参数 -gc-sections 传递给链接器），这样就能减少最终的可执行程序的大小了。

4.IDE

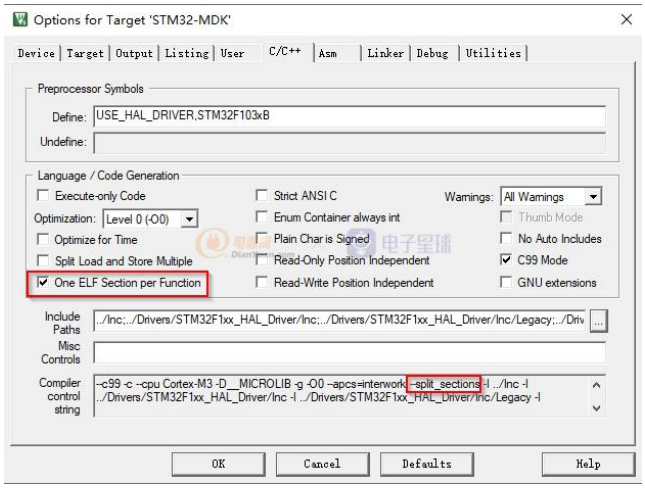
通常我们使用 IDE 的过程中，它已经帮我们做好了很多工作，比如上面提到的编译器命令需要我们自己手动写到 Makefile 中，但是在 IDE 只需要勾选相应的选项即可。

4.1 MDK 的设置

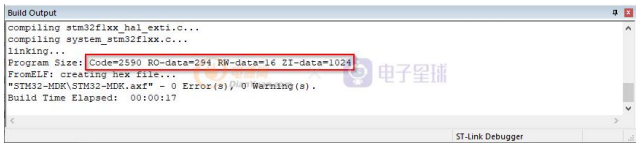
MDK 中使用的是 ARMCC 编译器，以 STM32 为例，纯净的 HAL 编译之后的结果如下图所示：



在工程的 Opentios 下勾选 One ELF Section per Function，发现在编译器命令自动追加了--splot_sections。



最终编译的结果如下，发现最终固件变小了很多。

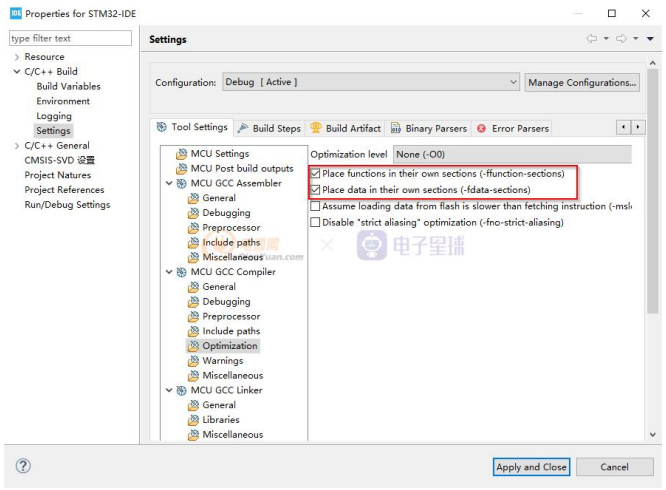


开启设置之后

4.2 CubeIDE

CubeIDE 中使用的是 arm-none-eabi-gcc，相同的代码与上面的基本相同，创建 CubeIDE 的工程，编译之后如下图所示：

同样在项目的属性设置中，增加--ffunction-sections 选项和-fdata-sections 选项之后，构建项目：



最终结果如下所示，发现固件的大小减小很多。

5.结论

本文对于如何删除编译过程中未使用的 section 做了简单地介绍。从 ELF 文件格式的角度出发，介绍了编译器 ARMCC 和 GCC 相应的命令以及 MDK，CubeIDE 中的相应配置，最终实验表明可以减少程序大小，另外编译器的优化等级-O1，-O2，-O3 也可以优化程序的大小以及执行时间。

留　言

互　动

图解法反激设计

文 / boy59

反激的设计资料多是以公式的方式表述不够形象，所以想用图形的方式将公式表述出来。

比如设计匝比时先确定占空比还是先确定反射电压？见下图（Vmax=300，Vmin=100，Vo=5，f=60KHz）。

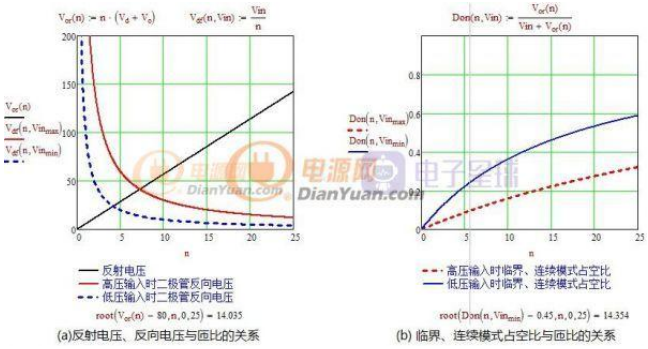


图 1-1 反射电压、占空比与匝比的关系

图(a)是反射电压、输出二极管反向电压与匝比的关系，匝比越大反射电压越高输出二极管反向电压越小。

图(b)是临界、连续模式时占空比与匝比的关系，匝比越大临界、连续模式的占空比越大。

占空比和反射电压二者是相互关联的，从上图可以通过直观的对比来选则一个匝比数比如取 n=14，此时反射电压 Vor≈80V，占空比 D≈0.45。如果匝比取 n=21，此时反射电压 Vor≈120V，占空比 D≈0.55，后者的效率会高一些不过需斜坡补偿。后面再讨论用图形法分析斜坡补偿和效率的问题。

再比如上面的例子，分别取电感 L=600uH 和电感 L=420uH，在匝比 n=14 的条件下看临界功率曲线。

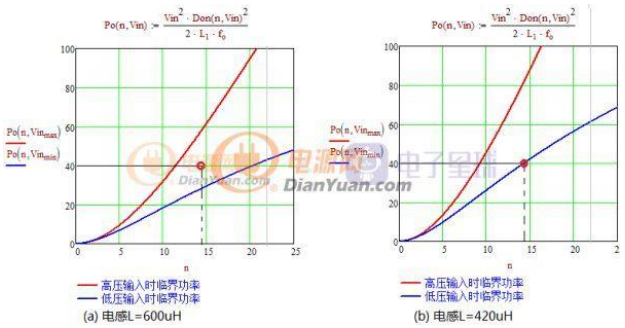


图 1-2 临界功率曲线与匝比的关系

假设输出功率为 40W，图(a)电感 L=600uH 低压输入时连续模式高压输入时断续模式，图(b)L=420uH 全程工作于临界或断续模式。

后续的图解分析大概包括变压器设计、电容选取、纹波、小信号、效率分析、EMI 设计等，期望能从这个过程中找出最优参数的设计方法。

一、输入电容

输入电路先近似等效为整流桥后接滤波电容和负载电阻，见下图。

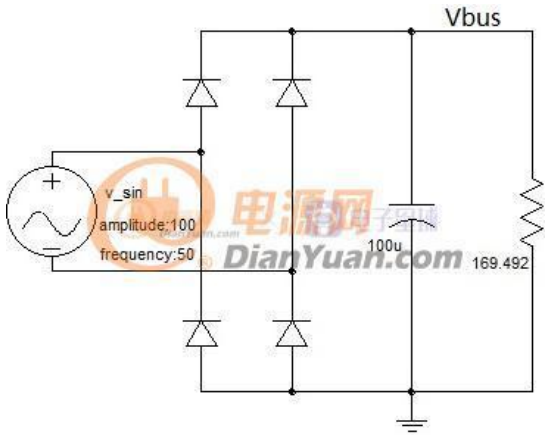


图 2-1 输入等效电路

电路的工作过程分为两部分，1、整流桥导通时母线电压 Vbus 等于整流后的输入电压，2、整流桥截止时变为 RC 放电电路。

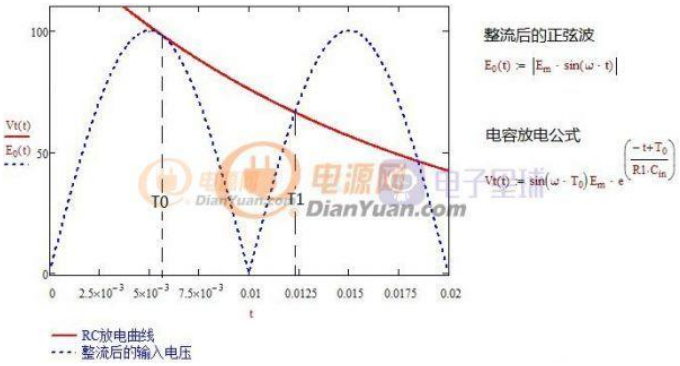


图 2-2 输入电路的母线电压波形构成

接下来是要确认 T0 和 T1 时刻，T0 时刻为两个曲线相切点 T1 时刻是两个曲线的交点，然后就可以画出母线电压 Vbus。

T0 时刻可以用两个 for 循环求解，这里借鉴资料中的公式（T0 的精度影响比较小）。在资料中多是将放电曲线近似为直线这会导致结果有些误差。

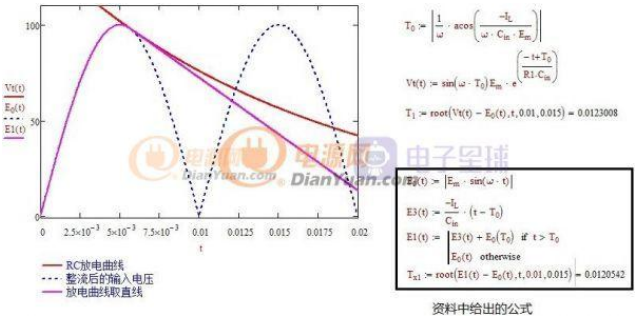


图 2-2 与资料中的线性放电曲线对比

再考虑整流桥的管压降减去 1.4V 得到最终的结果，同 Saber 的仿真结果进行对比如下：

图 2-3 Sabe 与 Mathcad 输入波形对比

如上图所示 Mathcad 的计算结果基本上同仿真的一样（上图输入电容为 200uH）。

接下来要对电流进行方程描述，首先去掉整流桥，电路变为由交流电直接驱动的阻容并联负载，此时的波形如下：

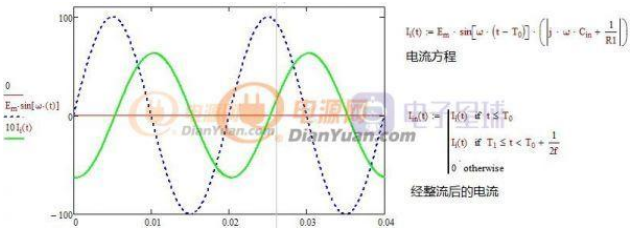


图 2-4 阻容负载的电压电流波形

负载上的电压为输入交流电压，电流超前于电压超前角跟 T0 有关（见上图方程），这时再把整流桥加上去，电压信号负的变正的对于方程描述就是取绝对值，电流取整流桥导通时段其它时段为零 ,重新生成的波形并同 Saber 对比如下：

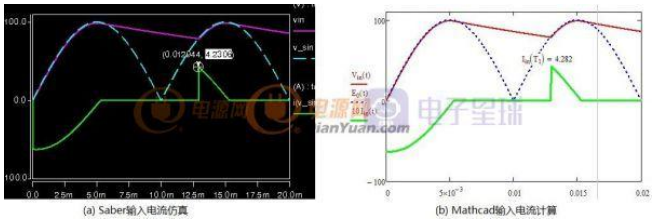


图 2-5 输入电流波形

如上图所示仿真和计算的结果非常的接近说明公式是准确的，不过遗憾的是大多数反激的应用都是恒功输出而不是恒阻输出的，所以一开始的等效电路就是不准确的，这个方程还需要再修正一下。

修正方法是把负载电阻换成与时间有关的函数既 R(t)=Vt(t)^2/Pin（恒功率），这时候方程变成了

$$V_t(t) := \sin(\omega \cdot T_0) E_m \cdot e^{\frac{(-t+T_0)\cdot P_{in}}{V_t(t)^2 \cdot C_{in}}}$$ （公式 1-1）

这个方程还不知道如何整理成单纯的 Vt(t)的函数，目前采用的是逐次逼近法。先假设 R(t)为恒定值代入方程求出 Vt(t)函数，将 Vt(t)代入恒功率方程求得一个跟时间 t 有关的 R1(t)，此 R1(t)再次代入方程求出 Vt2(t)，此 Vt2(t)再代入恒功率方程求出 R2(t)，R2(t)代入方程求 Vt3(t)……如此类推直到 n 次。

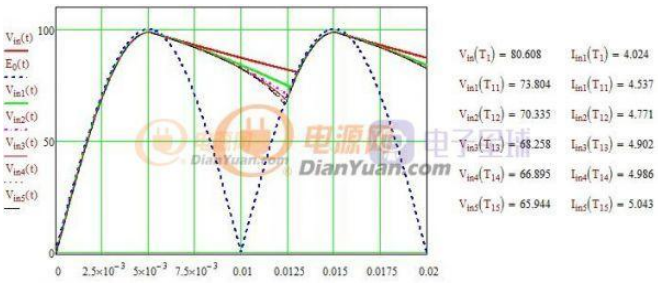


图 2-6 逐次逼近法计算输入电压波形

上图是逐次逼近法的计算结果，实际上这里还没有掌握逐次逼近法的精髓在上述结果中只有第一次逼近结果接近仿真值。下图是计算同仿真的对比：

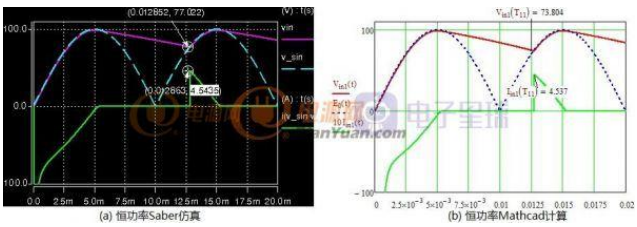


图 2-7 恒功模式下输入电容的波形对比

试过恒功率 50W-500W 计算结果和仿真结果还都比较接近，如果能整理出公式 1-1 的方程则结果应当更加理想。

有了电流方程后就可以计算出平均电流和有效电流从而进行损耗估算，电解电容的 ESR 可以用损耗角表示也可以用 50~80*10^-6 这个系数来估算如下图：

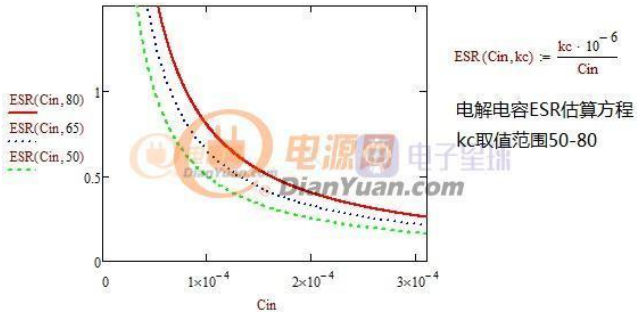


图 2-8 电解电容 ESR 与容量的关系

将之前的方程整理一下使输入电容 Cin 为自变量（X 轴）得到峰值电流和输入电容的关系：

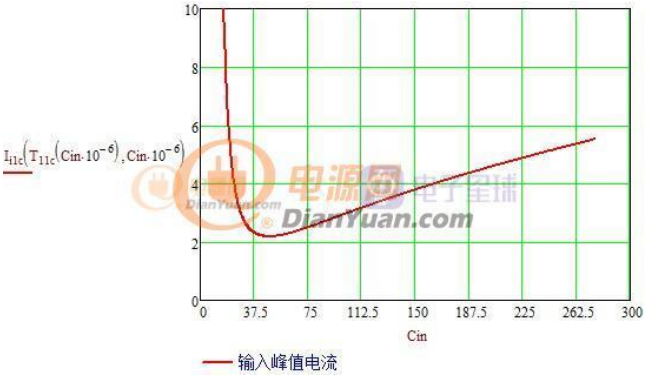


图 2-9 输入峰值电流和输入电容的关系

这样就可以估算出电解电容 ESR 和整流桥上的损耗：

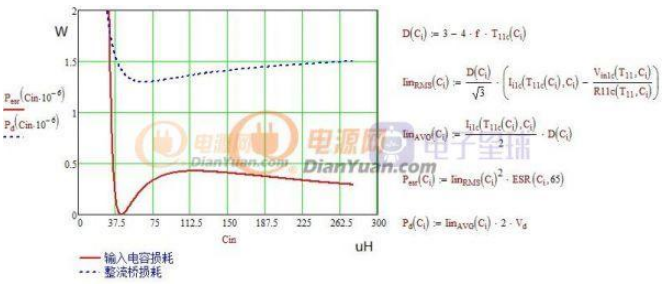


图 2-10 输入电容、整流桥损耗与输入电容容量的关系

上述数据的条件是输入电压峰值 100V、功率 50W、kc=65、二极管压降 0.7V。

下面是参考《精通开关电源设计》估算的电容寿命：

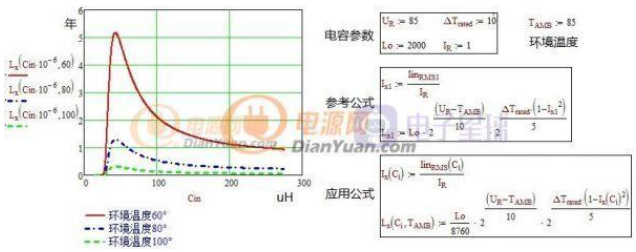


图 2-11 低性能电容寿命与容量、温度的关系

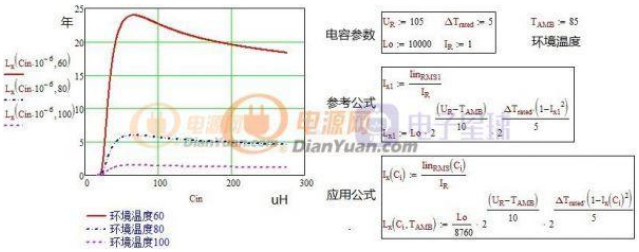


图 2-12 高性能电容寿命与容量、温度的关系

至此初步完成了输入电容的子模块设计后续再进行其它子模块整理，当完成所有子模块时整套电源系统就构建成功了。这个也是自我学习的一个过程，如果有分析不对的地方还望及时指正。

另外此输入电容是按最坏情况设计的，考虑到±20%的容量误差和容量下降 20%既为失效实际选择的电容是计算的 1.56 倍，可将此电容值代入方程来求解另一种情况下的极限值。

经过高人指点公式 1-1(恒功率输出)整理之后的表达式为：

$$Vt(t) := \sqrt{\left(E_m \cdot \sin(\omega \cdot T_0)\right)^2 - 2 \cdot P_{in} \cdot \frac{(t - T_0)}{C_{in}}}$$ 公式 1-2

将此方程代入后计算的电流、电压及波形同 Saber 仿真的基本一致。

电容恒功放电公式也可从下图推出：

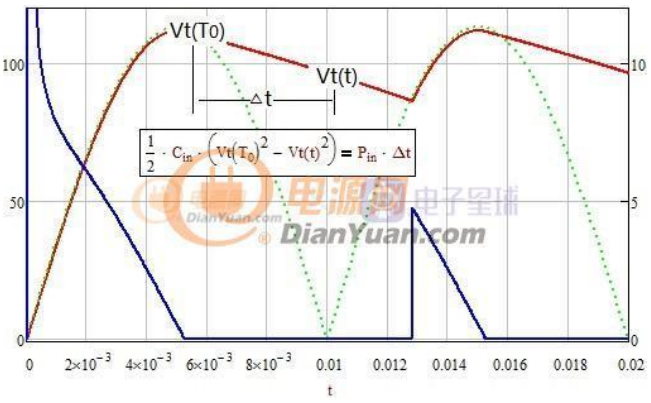


图 3-1 电容恒功放电公式图

输入电容 Mathcad 文件已上传方便大家参考 ,输入电流电压波形部分采用了微积分运算结果较精确。后面的损耗分析、寿命估算由于微积分运算速度较慢故采用了基波分析法。它们的峰值电流偏差如下：

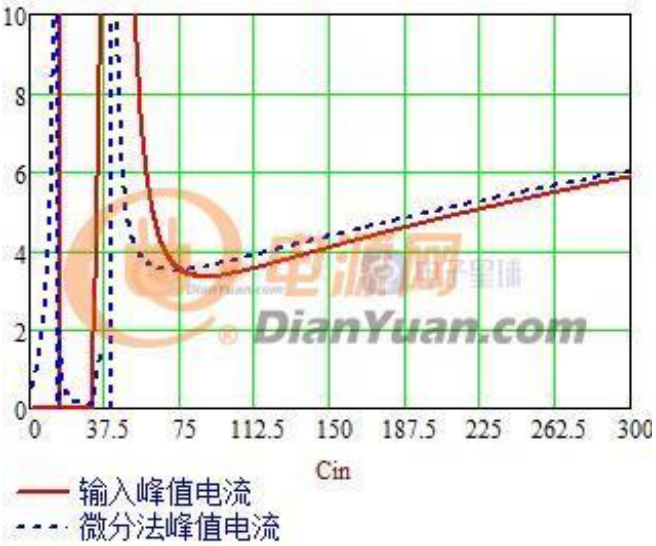
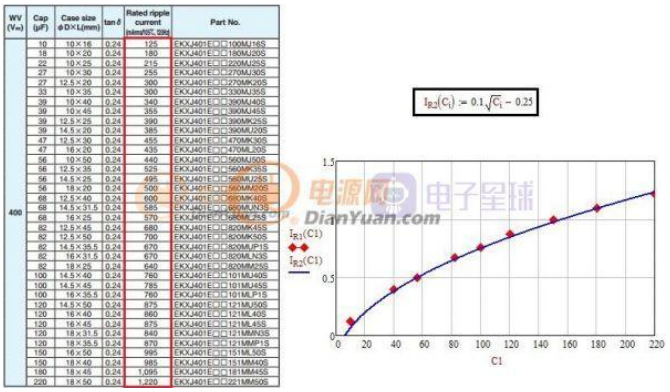
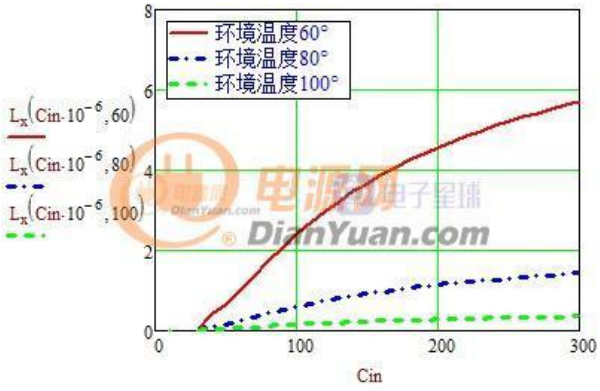


图 3-2 两种方法峰值电流偏差对比

电容寿命方程中的 IR 是个变量需要修正一下，参考一款电容。



得到这款电容近似的 IR 方程曲线，代入寿命估算方程重新绘制电容寿命曲线得：



留 言

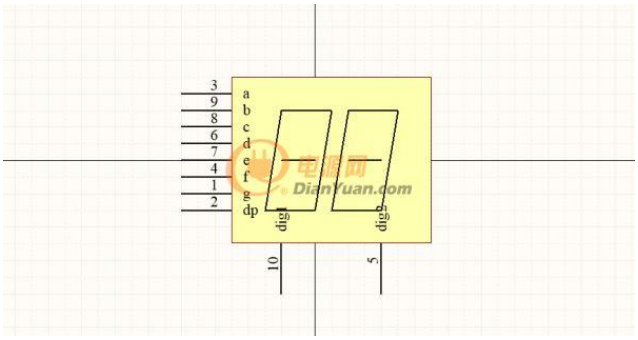
互 动

用 PCB 学习封装篇 :从入门到放弃，从放弃到入门

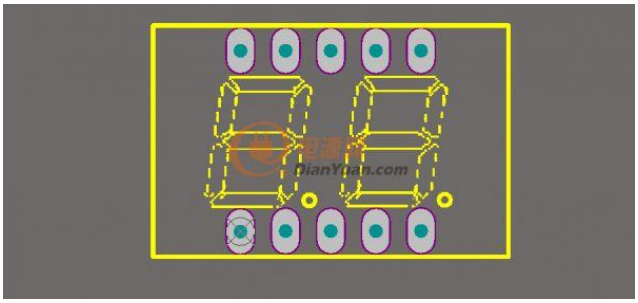
文 / 程序小白

之前的原理图设计我是一带而过，主要重心放在了 PCB 设计上。在设计 PCB 的过程中，有一点我们是绕不开的，那就是封装！这里我不会讲如何创建并绘制原理图封装和 PCB 封装 ,而是想告诉大家一个免费获取原理图封装和 PCB 封装，为我所用的小秘密。下面先简介一下封装的那些事，帮助那些如我一样半路出家的硬件工程师。

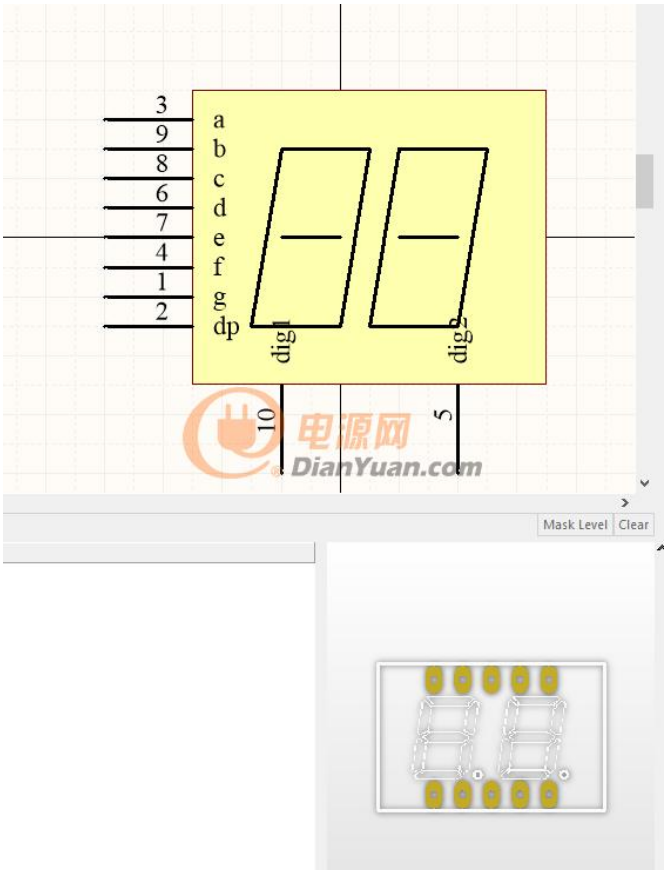
封装又分为：原理图封装（画原理图用）likethis：



PCB 封装（画 PCB 用）likethis：

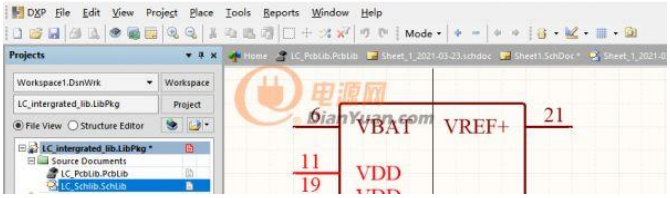


一个完整的元件需要同时拥有原理图封装和 PCB 封装：



其实这篇文章不是教大家如何去画原理图封装或是 PCB 封装，而是教大家如何获取一个新的元器件的封装。

首先，你要建立一个集成库的工程，并且添加原理图库文件和 PCB 库文件，至于这个步骤，大家百度就好了，建立完成工程如下：

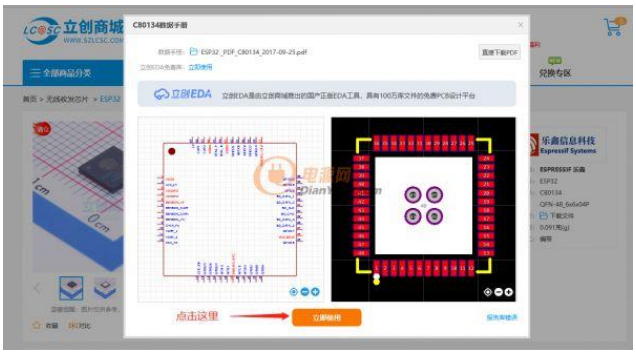


为什么这个集成库用 LC 命名呢，因为我们免费获取的元件封装就是来自于立创商城，或者是说立创 EDA 在线编辑软件，从网上打过板子或者是买过物料的小伙伴应该对它并不陌生。

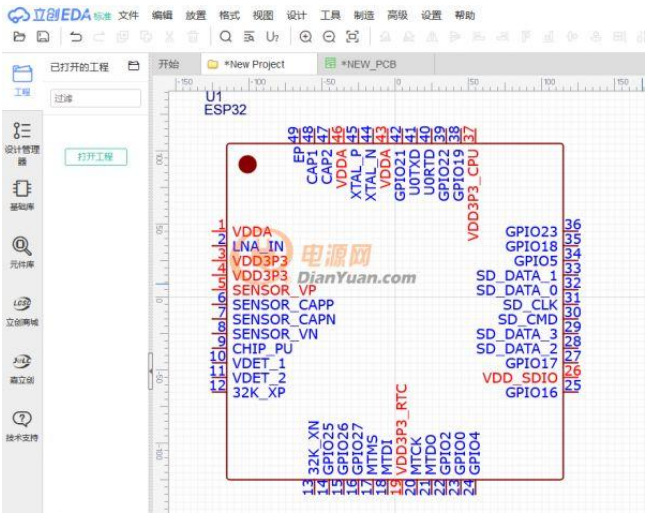
接下来我们以乐鑫的 ESP32 这颗 IC 为例，演示如何从立创获取这颗 IC 的原理图封装和 PCB 封装，并最终导入到我们的集成库中。

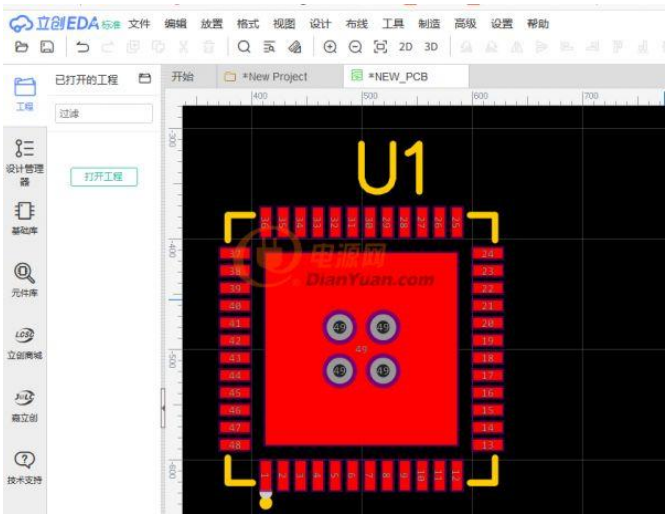
首先我们搜索 ESP32 这颗 IC，如下：

弹窗如下，这里提示建议大家先登录，不然无法完成以下操作。

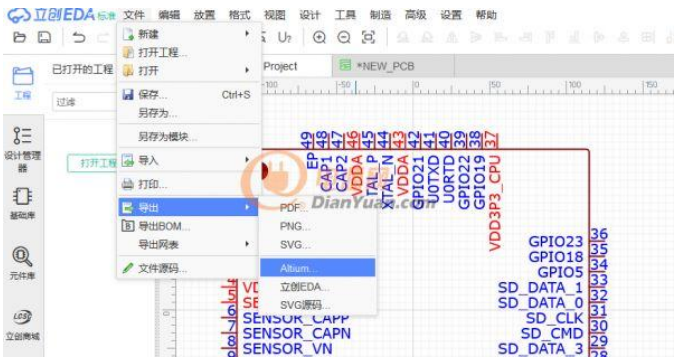


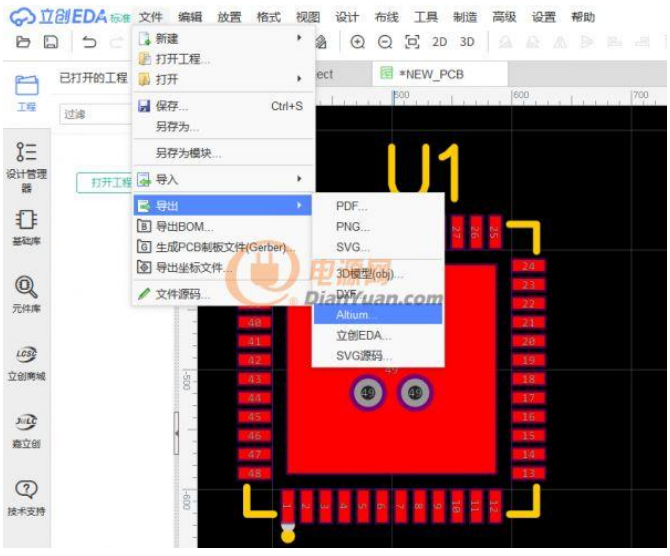
点击立即使用会进入到在线编辑器，这里我们会看到原理图封装和 PCB 封装。



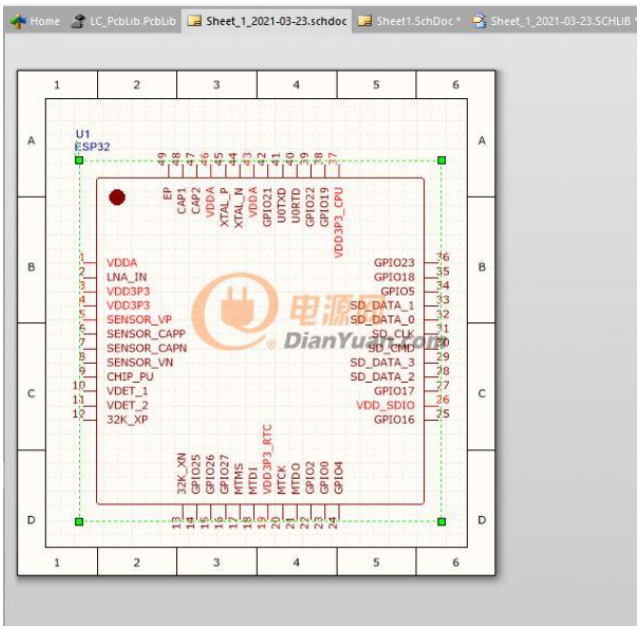


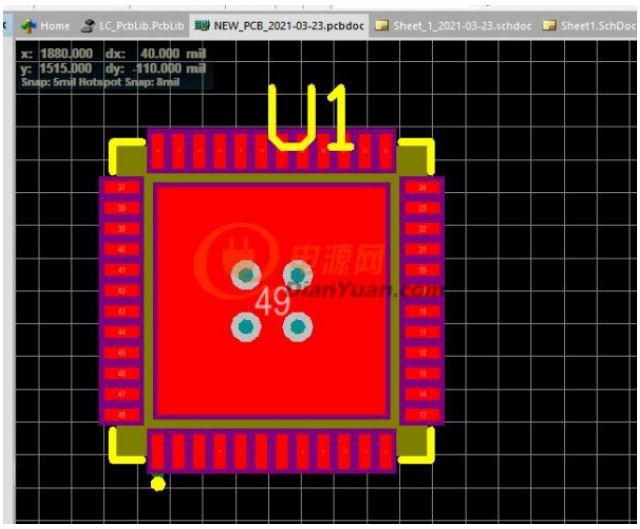
接下来就需要把这两个文件下载下来并保存为 AD 的文件格式，如下：



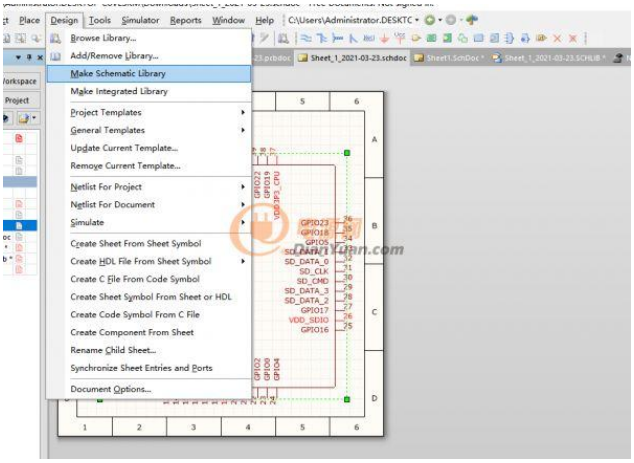


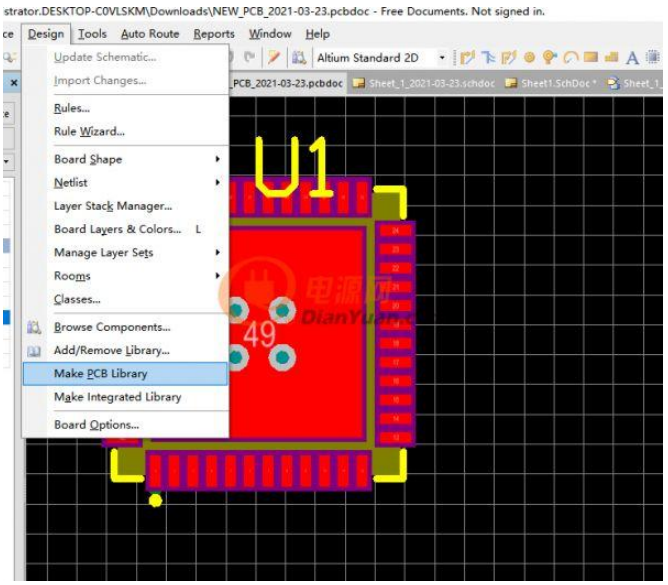
将保存到本地的文件用 AD 打开，如下：



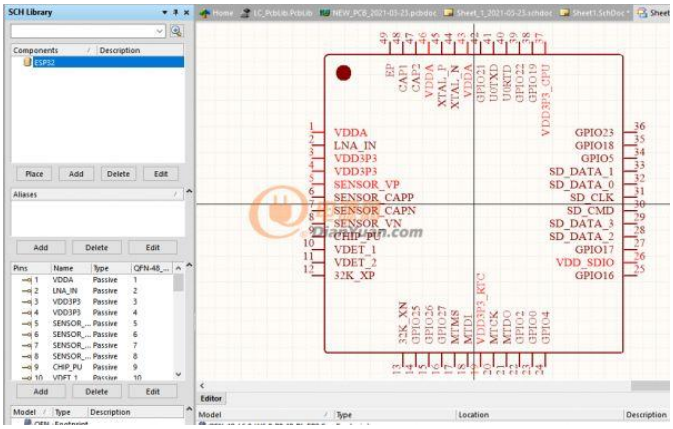


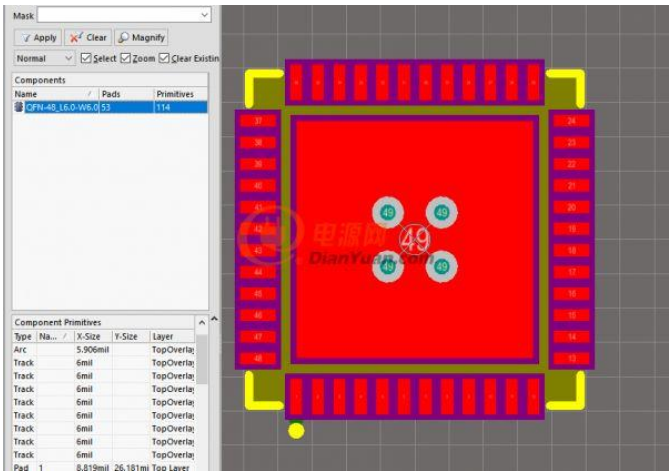
细心的小伙伴会发现它们都是 doc 文件，并不是我们我们想要的库文件，所以这里需要从原理图和 PCB 中提取封装，如下操作：



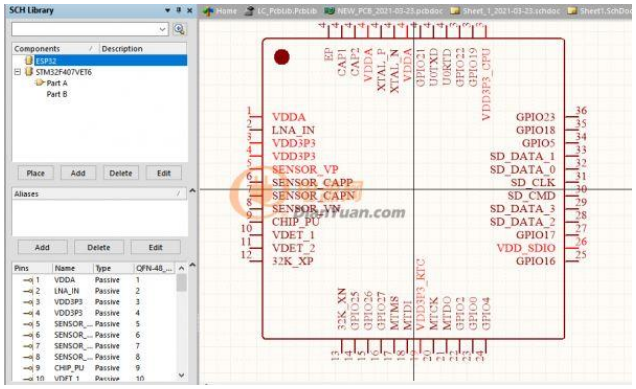


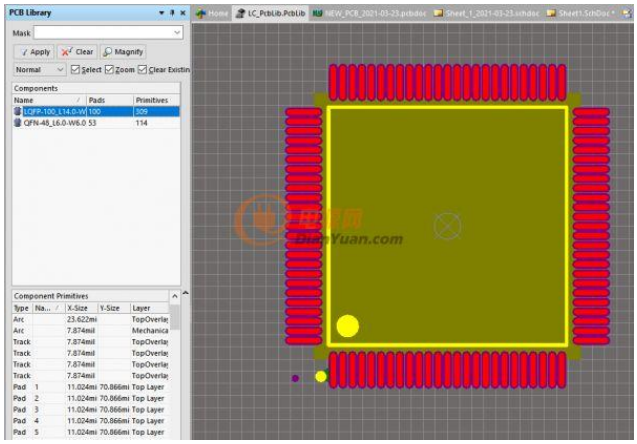
提取完成以后，会生成两个 lib 库文件，分别为原理图库文件和 PCB 库文件，如下：



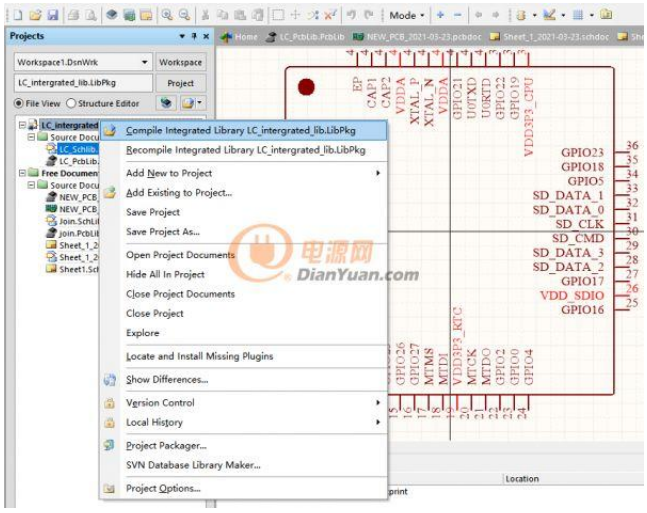


这个时候,我们可以把这两个文件拖入我们现有的集成库工程中，点击编译生成对应封装的元件，当然这样做并不专业。专业点的做法就是扩展我们现有的集成库编辑文件，当然也不麻烦，复制粘贴就可以搞定，如下：



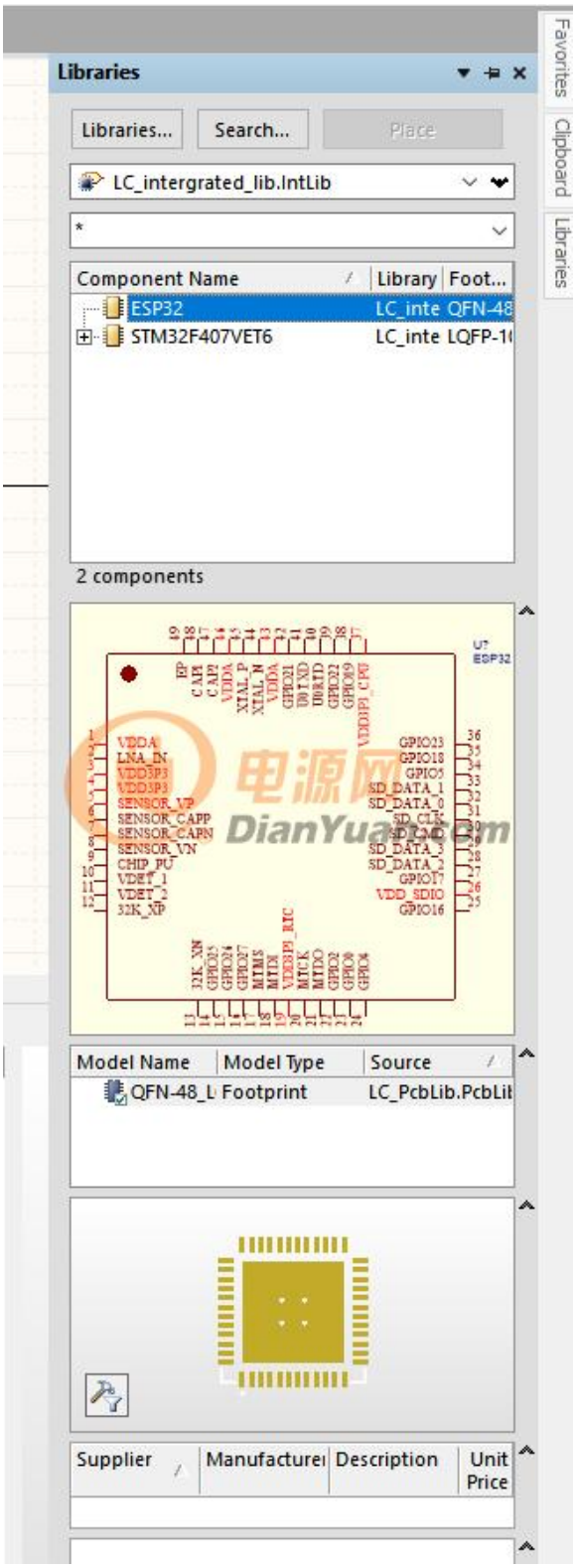


把他们分别粘贴到我们的 SCH 和 PCB 库中以后，只需要右击我们的集成库工程，第一项编译，等待一小会儿就可以生成对应元件的封装了：

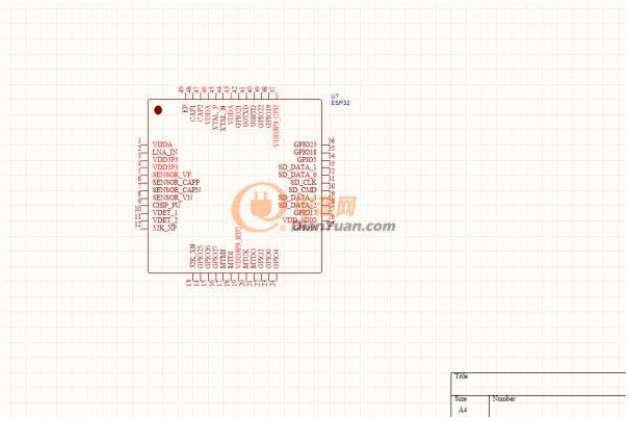


生成以后建议检查一下 message 信息打印 ,好习惯要养成：

没有错误哈，这个时候，你看一下右侧的 library 栏中，就会有我们的 ESP32 的元件了 ,对应的有原理图封装和 PCB 封装，如下：



这个时候我们就可以设计的原理图中，添加我们新的元件：



以后你需要什么样的 IC 芯片都可以尝试去上面找一找，有的话就可以省去很多的麻烦，你学会了吗？

留　言

互　动

基于 STM8 控制的单极性倍频调制 SPWM

文 / 宛东骄子

我做过好几种 SPWM，单极性单边调制，双边调制，单极性倍频调制，都各有特点。

单极性单边调制普遍用在高频逆变器后级，特点是电路简单，稳压电路和驱动电路好做，缺点是管子发热不均匀。

单极性双边调制普遍用在工频逆变器的逆变桥和部分高频逆变器后级，特点是各桥臂损耗和温升都差不多，驱动部分稍复杂，必需要做各种隔离。

单极性倍频调制其特点和单机型双边调制差不多，不同的是由于倍频的原因，开关器件的工作频率一般只有常规驱动方式的一半，即 10KHz 左右。

我用常规的 STM8 单片机试着做了一下，抓了一些图片与大家分享一下。

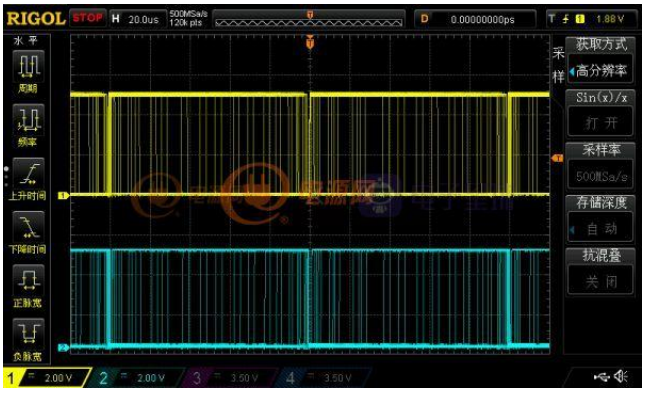
芯片输出脚位：TIM1_CH1 和 TIM1_CH1N,互补关系，插入死区，经过逻辑保护器件和驱动器件控制左边半桥。

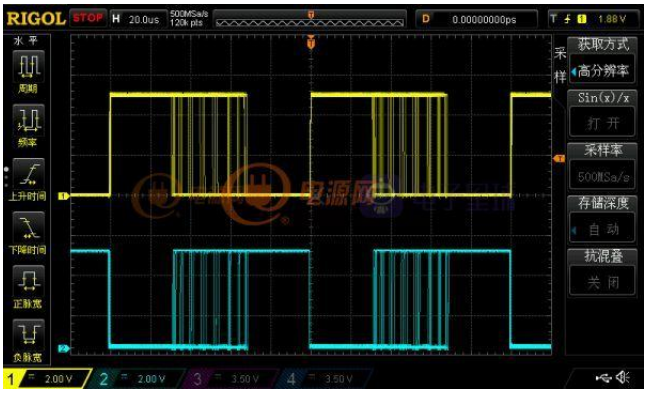
TIM1_CH2 和 TIM1_CH2N,互补关系，插入死区，经过逻辑保护器件和驱动器件控制右边半桥。

TIM1_CH1 和 TIM1_CH2 是交错 180°的关系，设置为中间对齐模式。

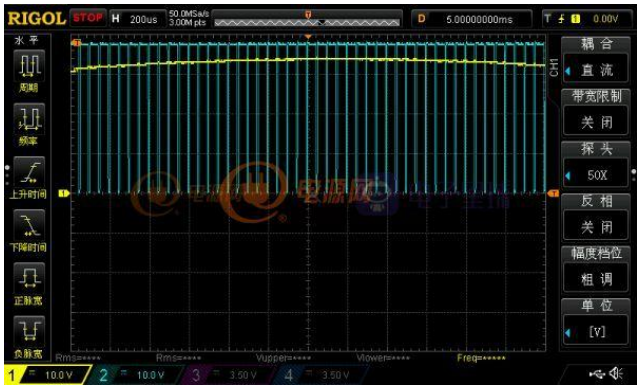
下面，高清大图来啦：

通道 1、2、3、4 分别是 TIM1_CH1、、TIM1_CH1N、、TIM1_CH2、、TIM1_CH2N，上面是四路波形同时展现的状态。

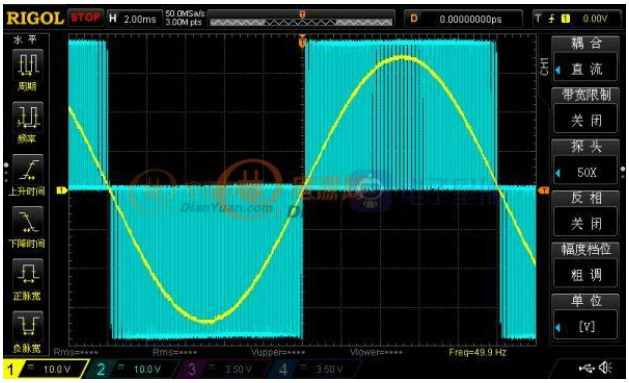


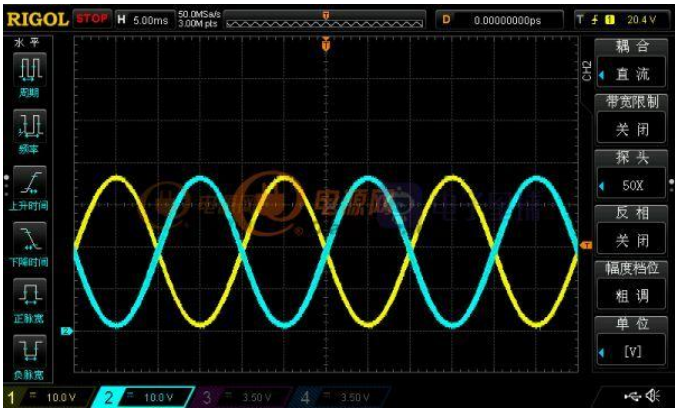


上面两幅是左边桥臂上下两管的驱动信号波形，是互补状态。

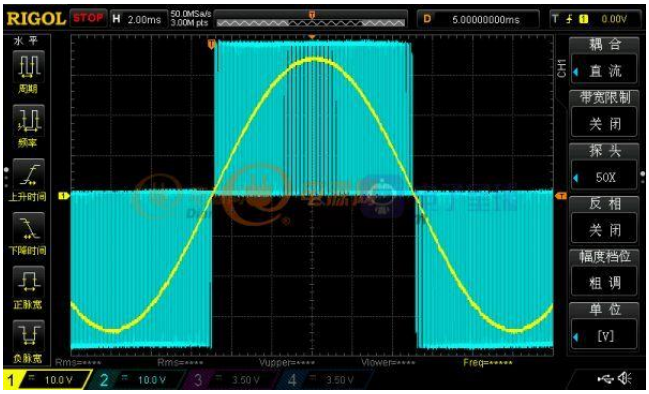


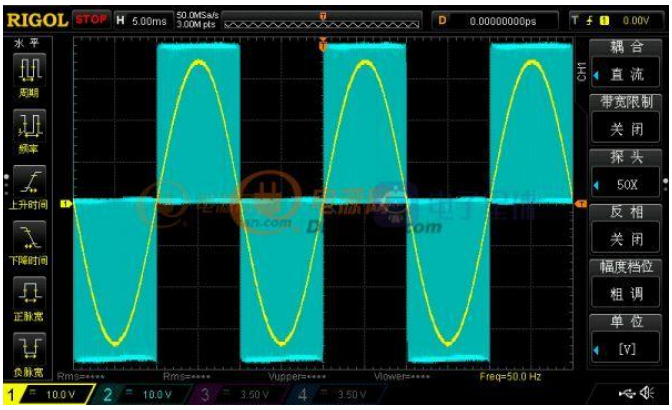
上面这幅是左边桥臂中点经过 LC 对地滤波后的波形，直流耦合状态测得。



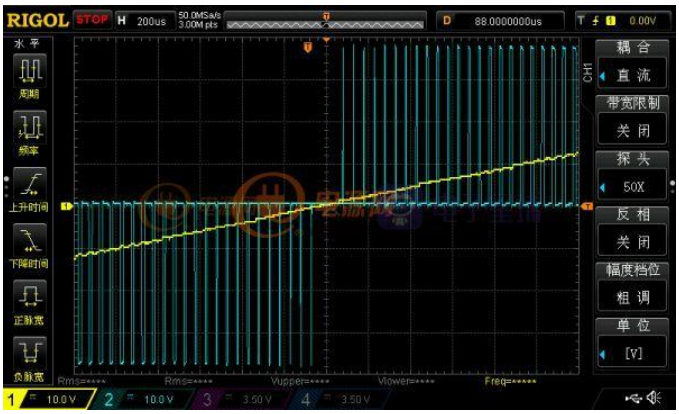


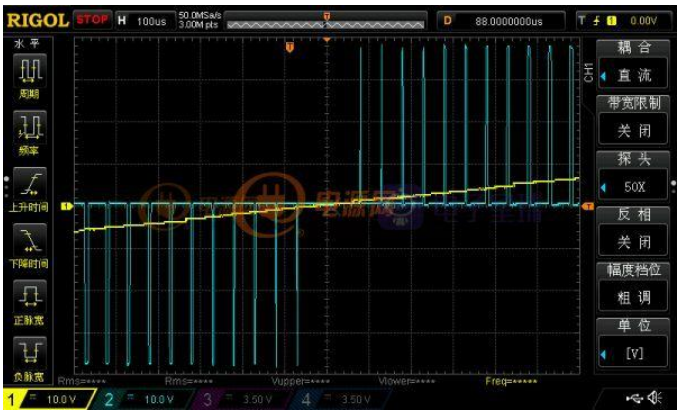
上面这幅是左右两桥臂中点单独对地 LC 滤波后，分别用两个通道测得。



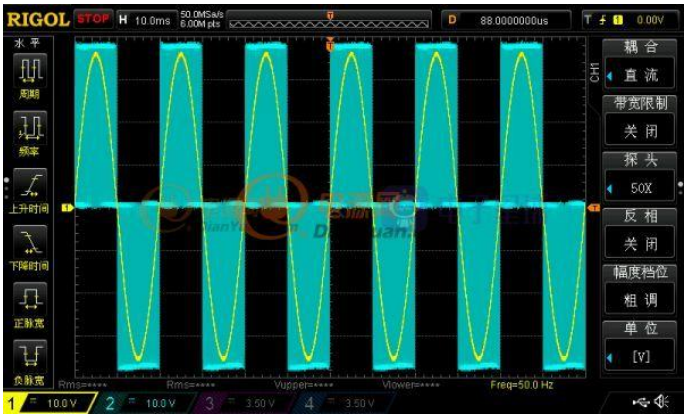


上面这几幅是左右桥臂中点经过 LC 滤波前、后的波形。





过零点的波形细节。



整整齐齐。

以上图片清晰地展现出单极性倍频调制 SPWM 的各个关键波形点，不敢说拍得很好，但也算是比较全面的图片了。

留　言

互　动

功率器件的里程碑——英飞凌 CoolGaN™功率器件测评

文 / javike

功率管是电力电子产品的基本构成单元，持续发展至今。

近年来碳化硅和氮化镓材料的功率器件推出，基于宽禁带半导体材料的功率管作为一种更先进的功率管正在被广泛的应用，其速度快，频带高，与硅等传统的半导体材料相比，它能够让器件在更高的饱和电子迁移率、频率和电压下运行。

硅的带隙是 1.17 电子伏特，碳化硅是 3.263 电子伏特，氮化镓是 3.47 电子伏特。

Table 2.1 Bandgap parameters and effective density of states of some semiconductors

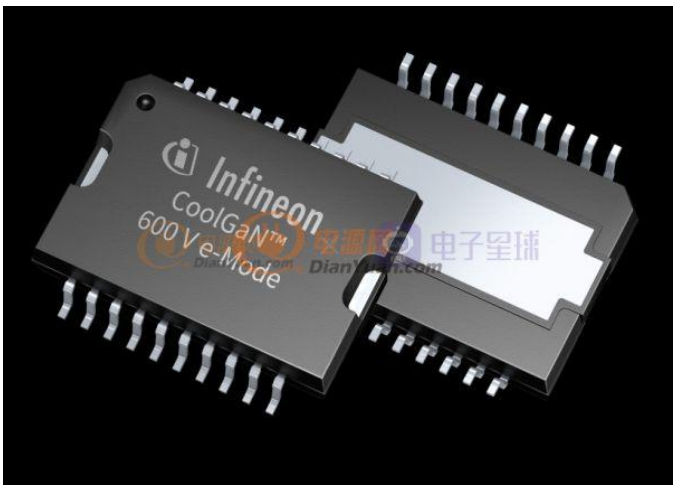
	Si	GaAs	4H-SiC	GaN
$E_g(0)$ (eV)	1.170	1.519	3.263	3.47
$\alpha \times 10^4$ (eV/K)	4.73	5.405	6.5	7.7
β (K)	636	204	1300	600
$E_g(300)$ (eV)	1.124	1.422	3.23	3.39
$N_C(300)$ (cm^{-3})	2.86×10^{19}	4.7×10^{17}	1.69×10^{19}	2.2×10^{18}
$N_V(300)$ (cm^{-3})	3.10×10^{19}	7.0×10^{18}	2.49×10^{19}	4.6×10^{19}

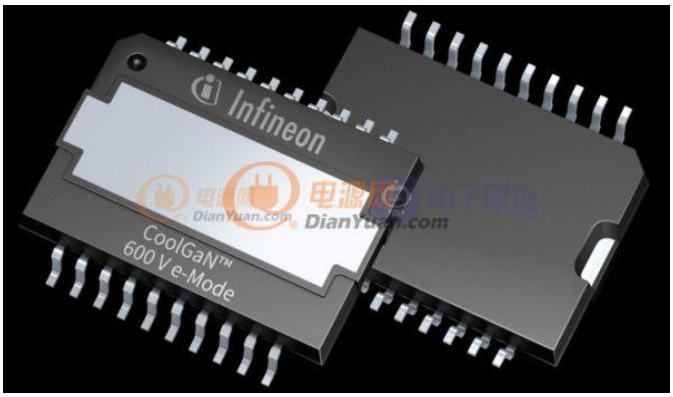
The source is "Josef Lutz, Heinrich Schlangenotto, Uwe Scheuermann, Rik De Doncker. Semiconductor Power Devices, Physics, Characteristics, Reliability, Spinger 2011"

英飞凌是目前唯一覆盖普通硅、碳化硅、氮化镓三种工艺的功率管的公司。

提到英飞凌，大家都知道他的 CoolMOS™ Mosfet，另外英飞凌还有 600V 以上的碳化硅二极管和 1200V 以上的碳化硅功率管，以及 600V 的 CoolGaN™ 产品，其优良的特性，包括无寄生体二极管、无反向恢复、可以双向导通，可以实现更多完美的拓扑以及更高频和高效的电源设计。

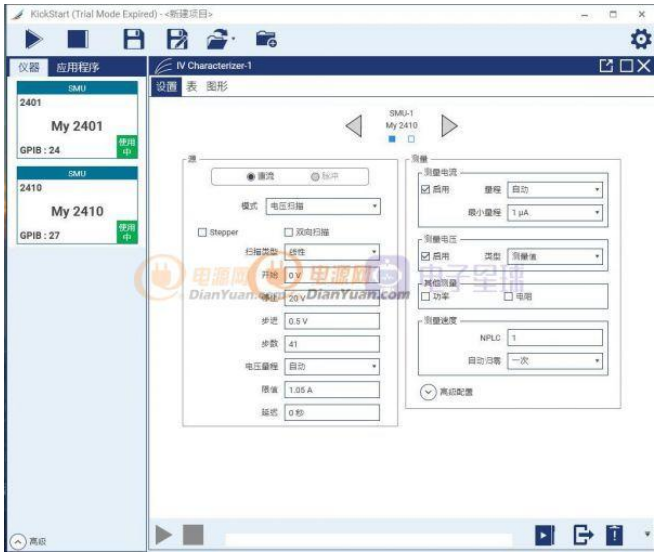
配合英飞凌的 CoolGaN™专用驱动 1EDF5673K 可以大大的简化设计。

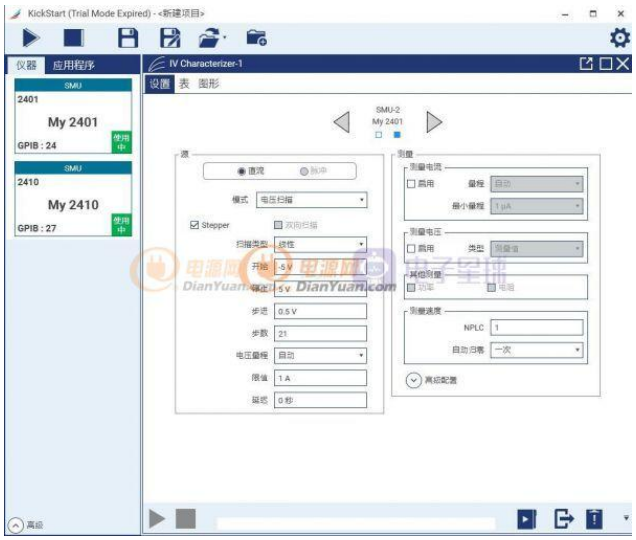




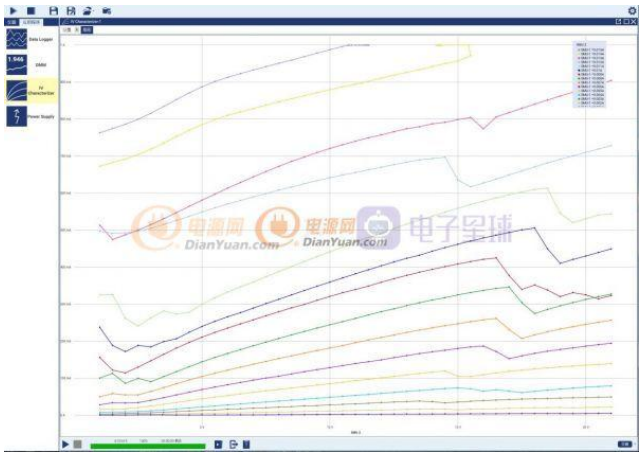
首先采用源表对英飞凌 CoolGaN™——IGO60R070D1 进行 IV 曲线的测量。

利用 2 台泰克（吉时利）的 6.5 位源表联机测量。





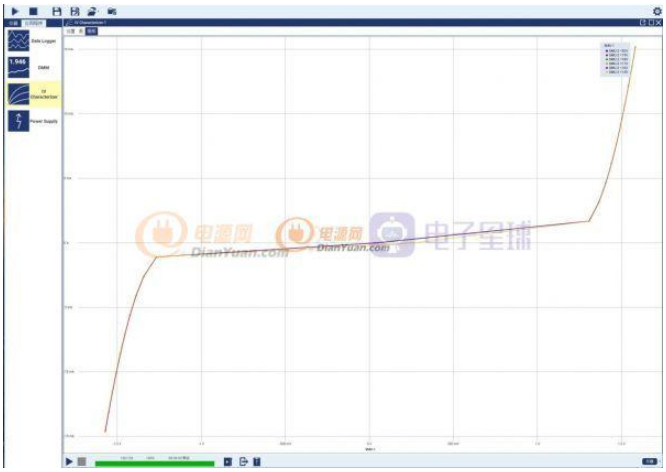
一台用于驱动信号的供给 ,另一台用于 VDS 和 IDS 电压电流的测量。通过上位机联动操作测试。



规格书给出的驱动所需要的最大平均电流是 20mA,设置 Vgs 电压限制为 5V , 测试 Igs 电流从 0.1mA 到 15mA 的 IV 曲线如上图。

从图中可以看出，Igs 和 Ids 的线性关系还是比较好的，在 Igs=14mA、Vds>15V 进入完全导通状态，在 Igs=15mA、Vds>11V 进入完全导通状态。

设定 Vds 为 15-20V，测试 Igs 电流从-15mA 到+15mA 时的 Vgs 电压的 IV 曲线。



从图中可以看出，CoolGaN™ IGO60R070D1 正负电流驱动的对称性非常好，而且趋势非常明显，在电流满足的情况下，需要的 Vgs 电压也非常低。在电路设计中我们知道，弱电流信号往往比弱电压信号的抗干扰能力更强，所以，CoolGaN™在电源中应用会比电压驱动氮化镓功率管更稳定和可靠。

不过在高频开关电源的应用中，还是需要按常规做法做到驱动回路尽量短小，将驱动线路中的寄生电感降低至最小，毕竟电感会抑制电流的上升。同时，由于 CoolGaN™的导通域值比较低，所以在高 DV/DT 和高 DI/DT 电路中，还是有必要在开关瞬间加入负压关断来抑制干扰。

建议采用英飞凌推出的 CoolGaN™ 专用驱动芯片 1EDF5673K、1EDF5673F 和 1EDS5663H，其不同于传统功率 MOSFET 的栅极驱动 IC，这个针对英飞凌 CoolGaN™量身定制的栅极驱动 IC 可提供负输出电压，以快速关断氮化镓开关。

留 言

互 动

工业应用中传感器数字 I/O 模块的选择

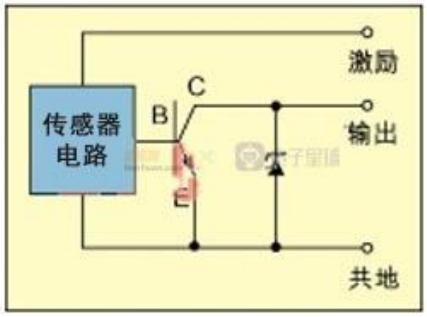
文 / Au 砸金子

数字传感器驱动电路

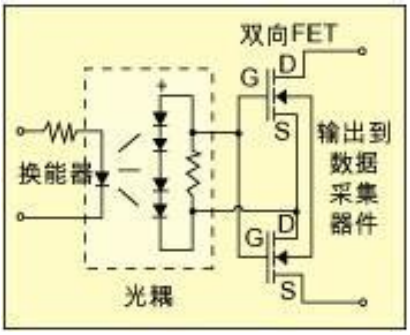
按照实现传感器输出的电子元件分类。用于驱动数字传感器输出的电子元件通常有三种：机械继电器、晶体管和双向 FET 器件。

机械继电器是一种电磁器件，闭合触点时能够接通电路，断开触点时能够切断电路。这种器件能够承受高压下的电流负载。机械继电器相对于固态继电器而言属于低速器件，因此通常用于以线路状态表征输出结果的传感器中。这种器件常出现的问题是触点老化和阻抗增大，其触点的寿命取决于电流负荷和工作频率。在与计数器或者数字 I/O 模块连接时，机械继电器有时会由于触点反弹而输出不确定的结果。

晶体管是一种用于控制 DC 电流的固态器件，有两种类型：NPN 型和 PNP 型，通常在低直流电源的传感器中用作输出开关。图 2 给出了一个 NPN(电流吸收) 型集电极开路晶体管。



双向 FET 器件是以一种叫做双向 FET 输出的结构实现的，如图 3 所示。该结构有许多优点，但最重要的是它能够直接与 TTL 和 CMOS 电路接口，而且它具有关态泄漏电流低和响应速度快的特性。FET 的意思是场效应晶体管，它是一种最适合于用作数字传感器输出的器件，因为其工作特性接近理想模型。



数字传感器 I/O 模块的选择

在选择数字 I/O 模块时，我们必须回答以下三个问题：

1. 传感器输出的数字信号的什么特征能够表征传感器的测量结果?从列表 1 你就能选择一组数字 I/O 模块。

2. 传感器的输出是吸入型输出还是泵出型输出?表 1 同样也提供了一组根据吸入和泵出能力分类的 NI 数字 I/O 板，如果你所采用的传感器是吸入型输出，那么你就需要选择一块泵出型输入的 I/O 板，反之亦然。

3. 用来实现传感器输出的是什么元件?这时我们可能就需要考虑传感器的关态泄漏电流和开态最小保持电流。比较表 2 中传感器的规格和数据捕获设备的规格，你就能选出最适合你传感器的数字 I/O 设备。

在选择过程中，除了考虑电压之外，以下两个条件也必须受到重视(有时我们会将其忽略)：

1. 传感器的关态泄漏电流≤I/O 板的低态最大输入电流;

2. 传感器的最小保持电流≤I/O 板的高态最大输入电流

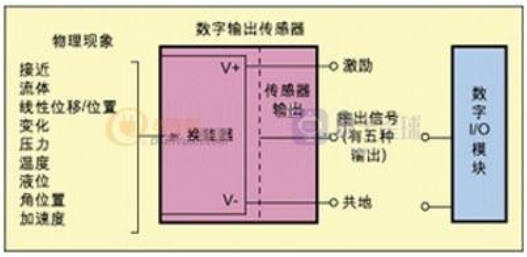
数字传感器以及由数字信号驱动的激励器的应用非常广泛，几乎所有现实中的变量(如温度、流量、压力、速度等等)测量中都可以找到数字传感器的应用，其数字输出有多种格式，本文首先根据输出信号和电路接口的类型对数字传感器分类，然后指出在选择与传感器接口的数字 I/O 模块时应注意哪些问题。

数字传感器与模拟传感器不同，模拟传感器的输出值是一个在整个输出范围内连续变化的值，而数字传感器的输出值只有两种，非“0”即“1”.轻触开关(touch switch)就是数字传感器的一个最简单的例子，在未被按下时，轻触开关通常是一个阻抗无穷大的开路电路，而按下之后，就变成一个阻抗为零的短路电路。在将数字传感器与数据捕获设备对接时，必须考虑一些可能影响接口性能的关键因素。下文中，我们将介绍这些关键因素，并为你提供一种简易方法，以指导你正确选择适合你应用的数据捕获设备。

数字传感器的分类

由于技术进步，市场上出现了各种各样复杂的数字传感器，而且如今的传感器已经能够产生一长串的开关状态转换，使用这些传感器时，输出脉冲序列的频率特性，甚至是脉冲形状都能表征传感器的测量结果，从而使得连续测量变为可能。

图 1 按照传感器输出的驱动信号的信号特征对传感器进行分类(有 5 种)，其中，45%的数字传感器是以数字线的开关状态表征输出，35 %以输出信号的频率表征输出，12 %则是以输出信号的占空比来表征输出 ,另有 6%以时间间隔表征，2%以脉冲数表征。



在选择捕获传感器数据所需的工业数字模块时，能够表达传感器测量结果的信号特征是第一个需要考虑的选择参数。在决定你所选择的数字 I/O 模块中是否需要计数器时，这一参数十分重要。

吸入和泵出这两个术语定义了负载中直流电流的流向控制。吸入型器件为电流提供一条到地的通路，不负责为设备供电。凡名字中包含 NPN、集电极开路和 IEC 负逻辑这些术语的器件都属于吸入型器件。泵出型器件提供电源或一个正电压，将电流灌入负载。凡名字中包含 PNP、发射极开路、常低和 IEC 正逻辑等术语的器件都属于泵出型器件。

吸入和泵出的概念与用来完成操作的元件无关(不论元件是晶体管、机械继电器还是其它)。这一概念对任何 DC 电路均适用，但用于实现这一电路的元件可以有多种选择。

2 线传感器或 3 线传感器

对于定义传感器开态和关态时电流和电压的关系而言，十分重要。一个传感器可按以下方式被归为 2 线传感器或 3 线传感器。

2 线传感器

这种传感器与数据捕获设备串联，当传感器未被激活时，它只吸收一个最小操作电流，这一电流值等于传感器的关态泄漏电流，有些传感器厂商也称之为残余电流。当传感器没有连接数据捕获设备，而直接连接到其它负载上时，就不存在残余电流问题，例如在工业环境中，这些 2 线传感器就常常直接连接到电机，以及类似电机的低阻设备。

但如果传感器所需的残余电流高于数字 I/O 模块能够提供的电流，那么问题就出现了。这时数字 I/O 模块可能会因为传感器吸收的电流高于它准备提供的电流而错误地将关态检测为开态。工业应用中的大多数 2 线传感器的关态泄漏电流或残余电流都不高于 1.7 毫安。

与关态类似，传感器要维持开态也需要一个最小电流，这就是最小保持电流(minimum holding current)，其值通常在 3 毫安到 20 毫安之间。如果数字 I/O 模块不能吸收或提供该电流，那么传感器将无法正常工作。

3 线传感器并不直接通过数字输出线获取能量，而是从一个激励终端获取能量，某些厂商也将这种传感器叫做线路驱动传感器。这类传感器从数字 I/O 模块处吸收的电流叫做负荷电流，通常为 20 毫安左右。但注意，该电流是由激励终端提供的。

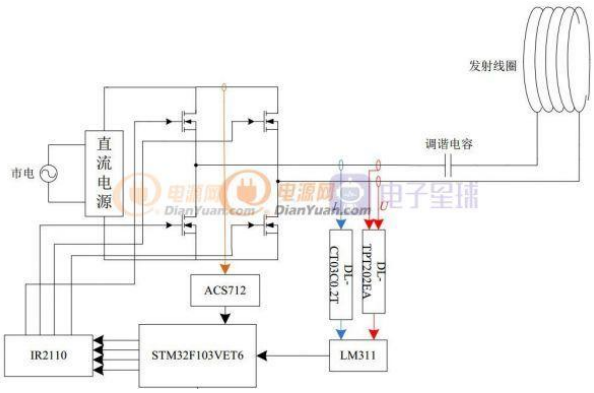
留　言

互　动

全桥+移相全桥逆变电源设计

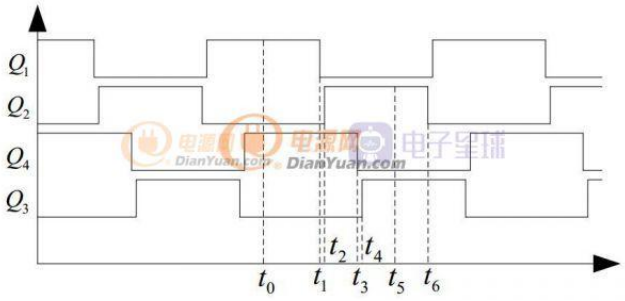
文 / 快乐的小天使

采用全桥的方式在逆变电路中应用非常广泛， 控制电路比较简单， 一般运用到中、 大功率场合。常见的全桥逆变控制方式有脉冲频率调制(PFM)，脉冲密度调制(PDM)，脉冲宽度调制(PWM) ，正弦脉冲宽度调制(SPWM)和空间矢量PWM(SVPWM)调制等。PWM 技术是逆变电路中运用最广泛的控制技术，本设计作为无线传能的前级电路，系统设计如下：



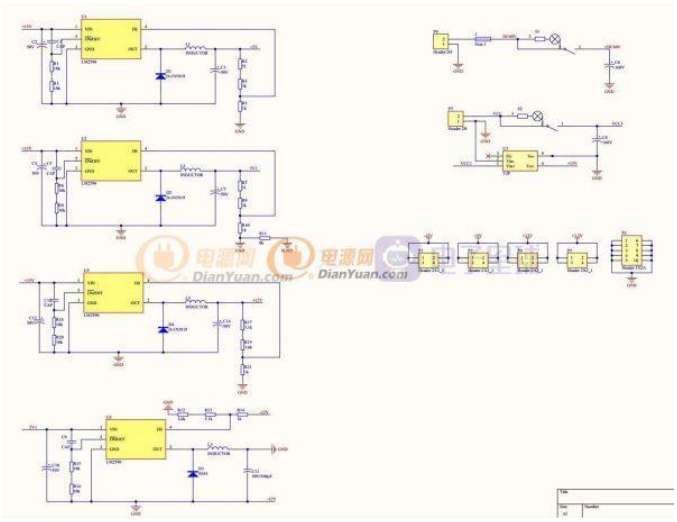
从市电接入一个直流开关电源为 MCR-WPT 电源供电，MCR-WPT 电源分为辅助电源部分、主控部分、逆变部分和信号采集及调理部分。主控芯片输出带有移相角的 PWM 信号经隔离驱动电路后驱动四个 MOSFET ，逆变输出接电容和发射线圈组成的串联谐振回路。输出的电压和电流通过高频互感器采即，将采集到的辛信号经调流电路输入给主控芯片作锁相处理。逆变环节中直流母线电流通过霍尔电流传感器采集，经跟随和限幅电路后通过主控芯片的 ADC 完成电流闭环。

单片机驱动波形



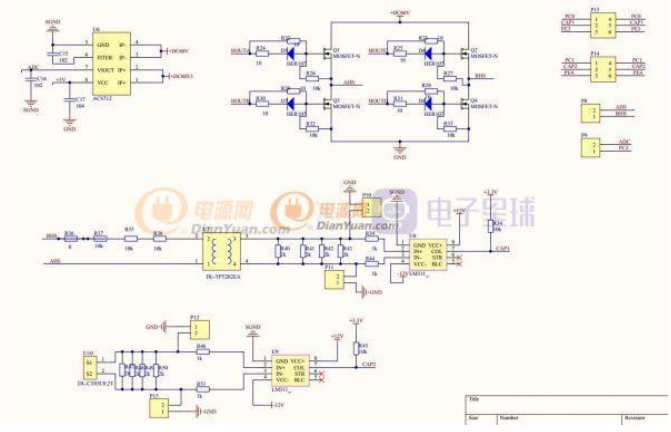
辅助电源电路

辅助电源电路使用 DC-DC 降压芯片 LM2596， 分别输出 5V、 12V、 -12V 给单片机、 IR2110 和 LM311 供电。 LM2596 最大输入电压 40V 电流 3A，输出稳压值决定于其 4 脚的反馈电压，通过微小的改动可以实现正电压输入负电压输出的转换。



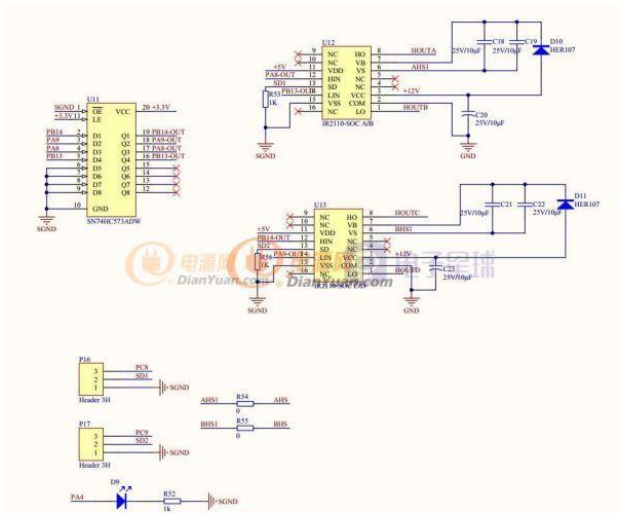
全桥电路和电压电流采样

本设计采用全桥逆变，需要四个 MOSFET。电源在谐振时电流很大，所以需要可以承受大电流的 MOSFET，选用 IR 公司生产的 IRF3205 MOS 管。



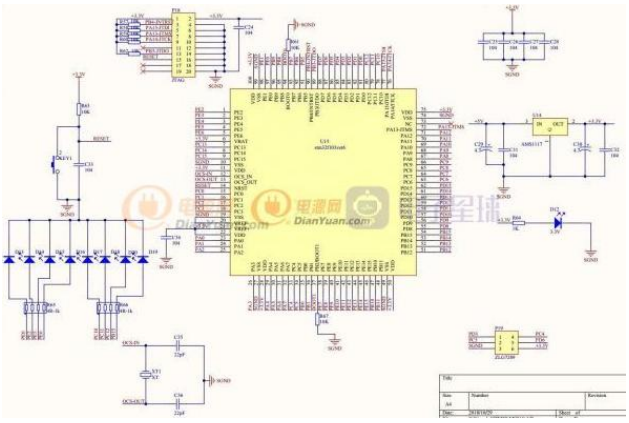
驱动电路设计

由于主控芯片输出的 PWM 信号不足以驱动 MOSFET，所以需要驱动芯片驱动全桥逆变电路。本设计的隔离驱动芯片采用 IR 公司生产的 IR2110，需要两片 IR2110 才能驱动四个 MOSFET。 IR2110 具有自举功能，外围电路比较简单，自举电容配置好。



单片机最小系统电路

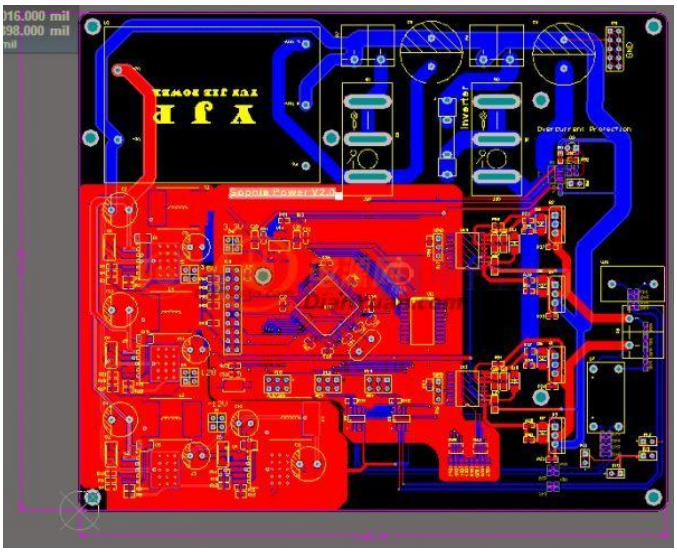
STM32F103VET6 最小系统包括电源电路、 时钟电路、复位电路和程序下载电路。 芯片供电电压为 3.3V，使用 AMS1117 提供 3.3V 电压，其输入为 5V 由辅助电源提供。时钟电路采用 8M 晶振，配合两个 22pF 起振电容。复位电路采用上拉复位。程序下载使用 JLINK 下载，故引出 JTAG 接口， JTAG 接口的数据线和时钟线均接上拉电阻。



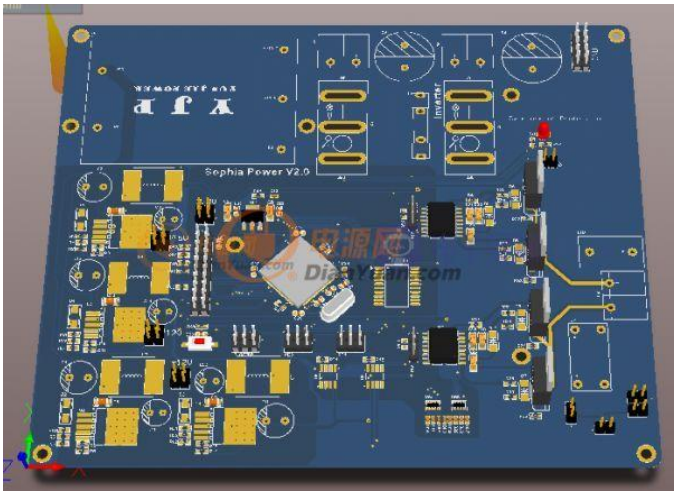
单片机资源分配，两个外部中断用来计算相位差、 一个外部中断用来读取按键值、 一个定时器中断用来 PI 运算、一个定时器作为计数器使用、一个定时器输出两对带有死区的移相 PWM 信号。

PCB 电路设计

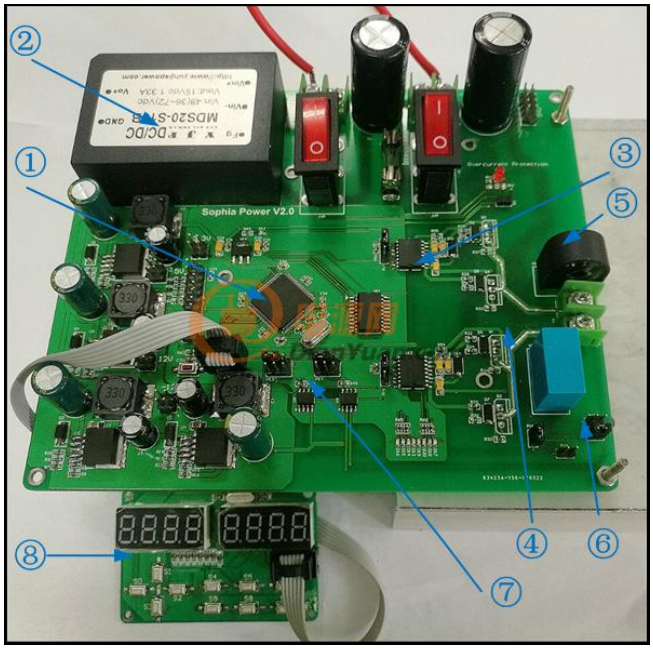
主要包含三部分，辅助电源部分，控制部分和逆变部分。



PCB 3D 效果图

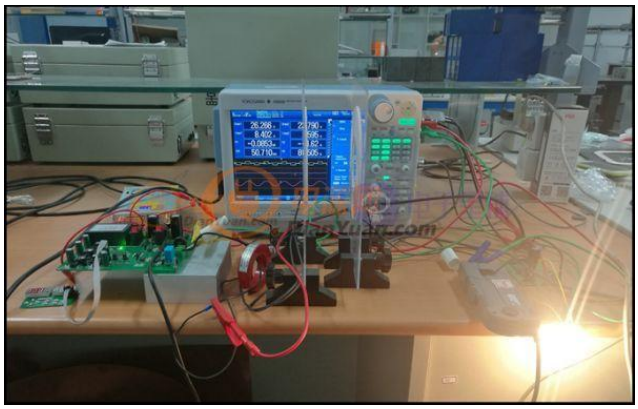


实物作品

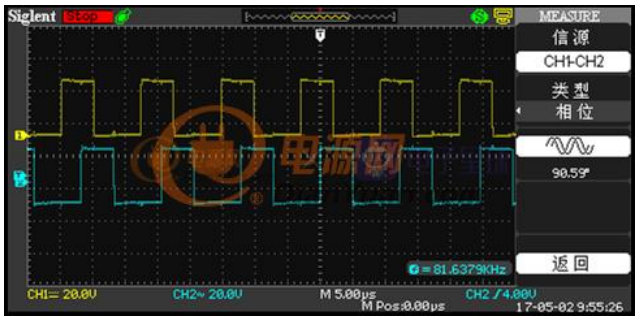


图中标号 1 为 STM32F103VET6 主控芯片，标号 2 为 DC-DC 降压隔离电源 ,标号 3 为 IR2110 驱动芯片，标号 4 为以 IRF3205 组成的全桥逆变电路 ,标号 5 为电流互感器 ,标号 6 为电压互感器， 标号 7 位以比较器 LM311 组成的信号调理电路，标号 8 为 ZLG7289 人机交互模块，它包含 10 个按键和 8 位数码管。

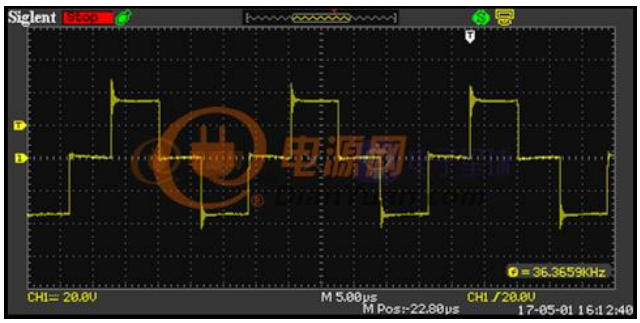
测试环境，灯泡成功点亮。



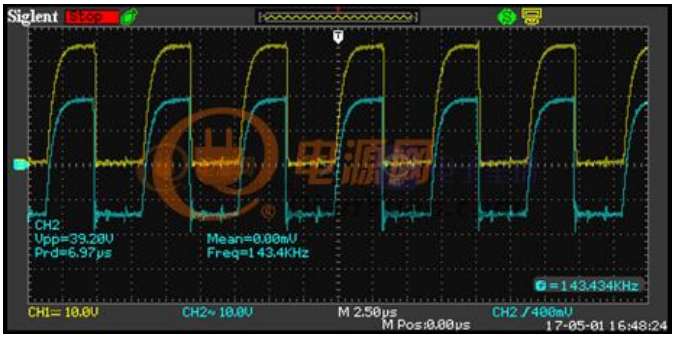
移相 90 度 PWM 输出



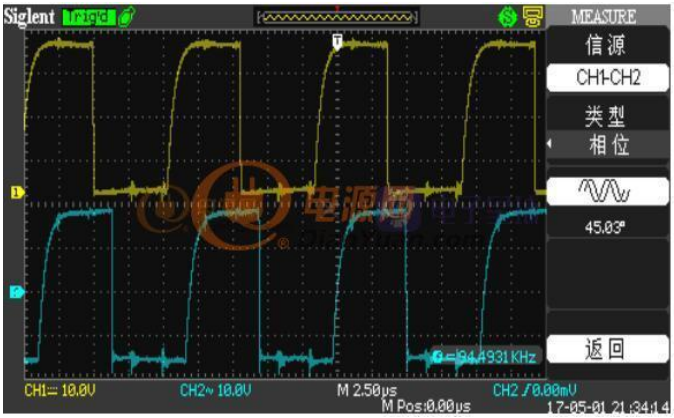
移相 90 度输出波形



锁相 0 度调试成功



锁相 45 度



测试参数	实验数据	描述
U_1	21.287V	发射线圈输出电压
I_1	9.808A	发射线圈输出电流
P_1	163.4W	发射线圈输出功率
U_2	24.761V	接收线圈输出电压
I_2	5.943A	接收线圈输出电流
P_2	146.9W	接收线圈输出功率
η	89.436%	效率
f	53.655kHZ	谐振频率

留　言

互　动

硬件小白成长记：给单片机加个开关

文 / 程序小白

第一篇点灯的成长之路已经完成，你可以用单片机来控制如何点灯玩耍了，至于具体如何写代码让单片机完成这个点灯的动作，还是往后放一放，看看其它一些常用的电路设计，这里讲一个，有关于单片机的输入开关量检测的设计。

开关量检测是在单片机控制中非常常用和简单的一类处理了，经常用到的开关有光电开关，水位开关，压力开关，微动开关等等，其实很多，这里我们讲一下无源开关的检测设计（有源的后面讲），也叫干接点输入，就是闭合和断开无极性。（你可以理解为家里常用的插座开关，只是一个通断的控制，电源由外部提供）。

首先，简单介绍一下，单片机在检测输入是通过检测电平的高低状态来完成的，所以如果直接把一个开关接到单片机上市没有任何意义的，其二在设计一开关量输入检测电路时有一些需要注意的问题：

1.输入到单片机的电平不能超过单片机能够承受的最大耐压值，一旦超过会怎样，超过单片机的引脚会出现短暂性失效和永久性失效两种状态，第一种还好，一般断电修复电路，重新上电，单片机正常工作，另一种就只能换单片机了。（关于单片机的引脚耐受电压是不同的，以 3.3v 供电的 STM32F103RCT6 为例，有的引脚可以承受 5V，有点引脚可以承受 3.3V，打开他的手册你看引脚上带有 FT 的就是耐受 5v 的引脚，如果电压再高就不能保证了。）

说再多都是空口白话，还是从 datasheet 中截取一段，大家看着更直白：

Pins				Pin name	Type(1)	I / O level(2)	Main function(3) (after reset)	Alternate functions(3)(4)	
LQFP100	LQFP64	TFBGA64	LQFP48					Default	Remap
90	56	A4	40	PB4	I/O	FT	NJTRST		PB4 / TIM3_CH1 SPI1_MISO
91	57	C4	41	PB5	I/O		PB5	I2C1_SMBA / TIM16_BKIN	TIM3_CH2 / SPI1_MOSI
92	58	D3	42	PB6	I/O	FT	PB6	I2C1_SCL(12)/ TIM4_CH1(11)(12) TIM16_CH1N	USART1_TX

1. I = input, O = output, S = supply, HiZ= high impedance.
2. FT= 5 V tolerant.

2.假如本来接开关的端口，被接入了电源该怎么办（同样两个接入端，遇到不懂的人谁管他带电不带电先接上去再

讲嘛。），如何有效防止这种意外的发生呢，也是设计硬件时需要考虑的问题。

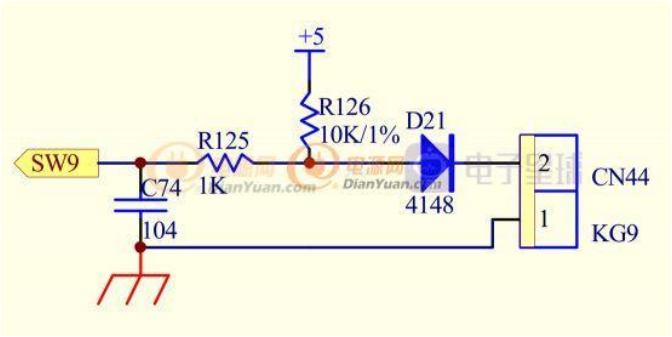
3.因为输入很容易引入干扰，像静电干扰/电磁干扰，影响小导致你检测信号的不正常，影响大直接损坏你的检测电路。

4.有时候你还需要通过电路来检测一下这个开关量是不是好用的呢，万一一上来就是坏的，万年常开万年常闭，又该如何去判断呢。

其实硬件设计出能用的电路很简单，身为小白的我学校学的那点知识可能就够用了，但是如何设计出简洁稳定低成本的电路却是一门艺术，你不光要从设计者的角度考虑问题，你还要站在使用者的角度去考虑问题，不断地对电路进行升级，伴随着产品的升级，让他越来越好。

闲话扯远了，下面让我们来进入一下正题，那些年大牛们钟爱的那些电路：

先来一个某位大牛曾设计的电路我们看一下，然后小白带你一点点的分析，先上图：



CN44,是常用的端子编号，表示第 44 号端子，这里接开关的输入端。

SW9,是单片机输入口的编号，这里的电平是单片机检测的输入端，也是我们需要的测试端。

端口确认了，接下来分析一下这个电路的原理，首先假设接入开关处于断开状态，SW9 的电位由+5 接 10K 电阻接 1K 电阻接过来，所以这个点为高电平。

当接入开关在闭合状态时，SW9 端的电位由 R125 右侧决定，通过分析为 0.7V，也就是 4148 二极管抬升部分的电位。高位 5v 低位 0.7v，这个电路是可用的哈。

下面来分析一下电路为什么要这么搭：

首先接 5v 的 R126 为 10K，主要作用就是限流，因为当前电路我们只做电平检测，所以电路取大一些来电路整体的功耗。

再来看 104 对地接一个电容怎么看：104 的电容主要用来屏蔽高频干扰信号，所以这里主要用来屏蔽干扰。

再来看 104 电容串接 1k 的电阻，主要是用来防止电容充放电时，造成的冲击干扰，串个电阻限下流，因容值本身就比较小，所以这里选择串接一个较大的 1k 的电阻。

再来看看这个二极管 4148，这里稍微展开看一下。下面先看下这个 4148 的 datasheet 吧。

先看下一手册给出的参数，第一组手册给出的是最大额定值的参数：

Operating Temperature: -65°C to +200°C
Storage Temperature: -65°C to +200°C
Operating Current: 200 mA @ T_A = +25°C
Derating Factor: 1.14 mA/°C Above T_A = + 25°C
Surge Current A: 2A, sine wave, P_W = 8.3ms
Surge Current B: 1.41A, square wave, P_W = 8.3ms

运行温度和存储稳定都是-65 到 200 度。

最大通过电流 200mA 在环境温度为 25 度的情况下，根据实际经验随着环境温度越高，其通过的最大电流值会降低，实际再设计电路时，要注意不要在最大额定值的附近设计，一旦环境温度提高会导致硬件损坏。

降额因数：1.14mA/度，这个没大研究明白，先略过。

浪涌电流（正玄波）2A，最高持续 8.3ms

浪涌电流（方波）1.41A,最高持续 8.3ms

接下来是额定参数，25 度时候（好像接触的 datasheet 额定参数都是 25 度下给出的）：

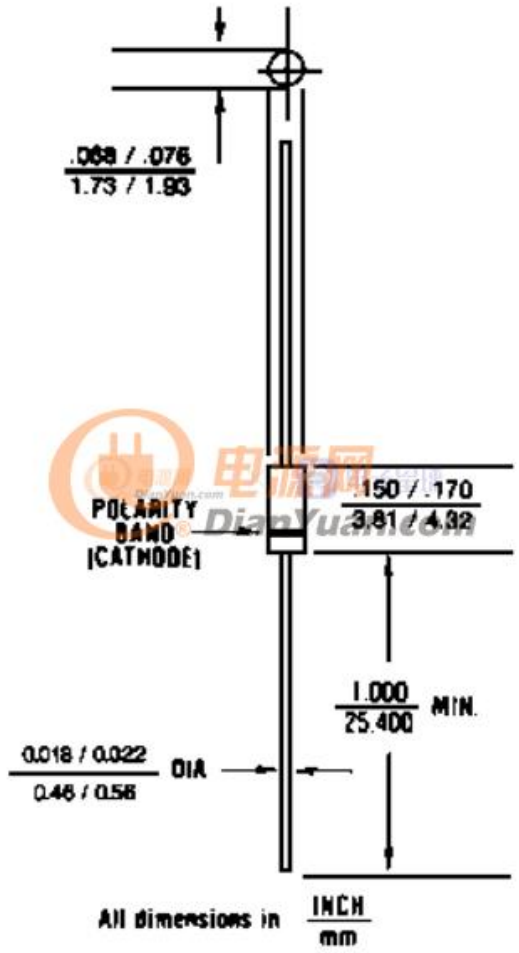
ELECTRICAL CHARACTERISTICS @ 25°C, unless otherwise specified.

V_{BR} @100μA	V_{RWM}	I_0	V_{f1} @I_F = 10 mA	V_{f2} @I_F = 100 mA	t_{rr}
Volts	Volts (pk)	mA	V dc	V dc	n sec
100	75	200	0.8	1.2	5

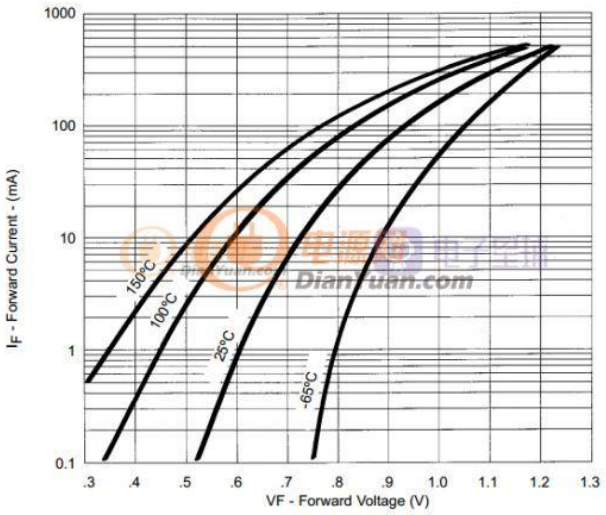
I_{R1} @ 20 V dc	I_{R2} @ 75 V dc	I_{R3} @ 20 V T_A = 150°C	I_{R4} @ 75 V T_A = 150°C	CAPACITANCE @ 0 V	CAPACITANCE @ 1.5 V
nA	μA	μA	μA	pF	pF
25	0.5	35	75	4.0	2.8

这里基本大体说下，不一一展开了，大部分参数我都是猜的啊 哈哈哈，希望没猜错，最大反向电压 100V，超过就报废，最大反向方波电压 75V ,正常使用时压降大概在 0.8~1.2 之间的样子，电流不同，电压不同，非绝对值，后面看其曲线你就更能直观的看出他的特性。

继续翻翻翻，这个是封装尺寸：



接下来看一下，伏安特性曲线（这里不是课本给出的，而是厂家给出的）：



从图中可以看出，温度不同，元件的特性也不同，但是基本走势是一样的，开启电压，电流上升趋势，跟当年课本上学习的还是很像的。

留 言

互 动

软蓝牙+FM 功能低功耗设计

文 / yjyd6677

设计带收音机功能蓝牙耳机：高通 QCC300X 系列+FMIC——RDA5807M

QCC300X 应该输出 1.8V 的 VDD 端口却输出电压为 2.35V，关机后 1.8V 的 VDD 端口依然输出 2.35V 电压不变，待机电流有 180uA。

对于头戴式蓝牙耳机来说，待机功耗算非常大了，因为单功能的蓝牙耳机正常情况 1.8V 在 5S 内会掉至 0V ,待机电流在 1uA 以内。

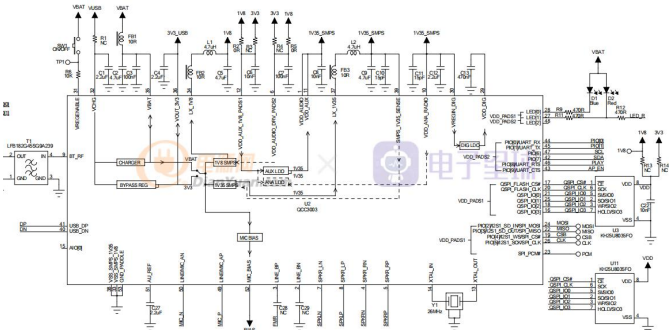


图 1 QCC3003 应用示例图

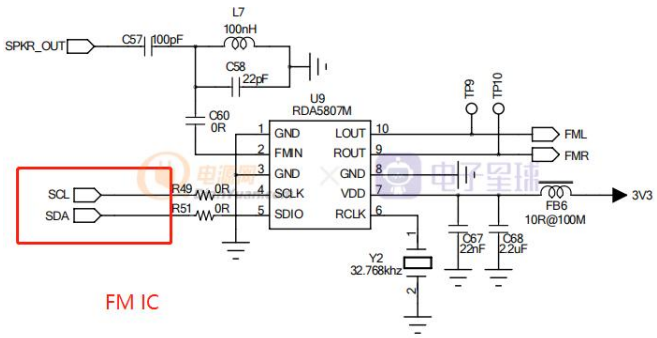


图 2 FMIC——RDA5807M

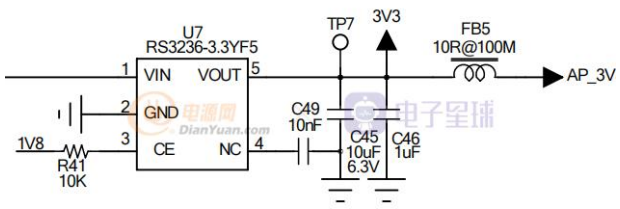


图 3 3.3VLDO

工作原理 ：

开机 :关机状态下长按开关机键 SW1 3S 以上 QCC300X 启动输出 1.8V 1.35V。

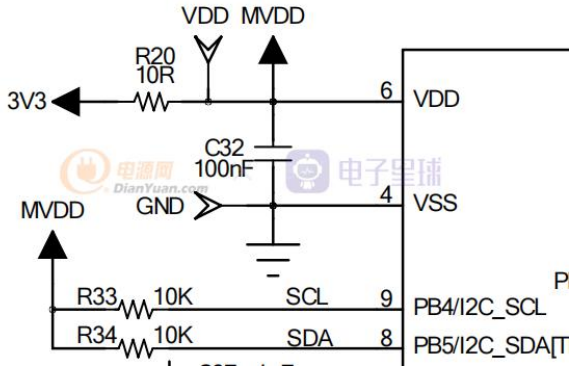
1.8V 驱动 3.3 VLDO RS3236 使能端，LDO 输出 3.3V 给 FMIC 供电。通过开关机键可以切换蓝牙模式与收音机模式。在收音模式下音量调节和上下电台切换都是通过 QCC300X 的 I2C 端口(PIO6 PIO7)与 RDA5807M 的 I2C 相连通信完成。

关机：开机状态下长按开关机键 SW1 3S 以上。

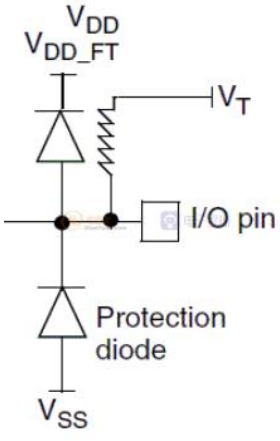
第一次发帖先起头，后续再说明原因及解决方案。

一、先说说 QCC300X 应该输出 1.8V 端口却输出电压为 2.35V。

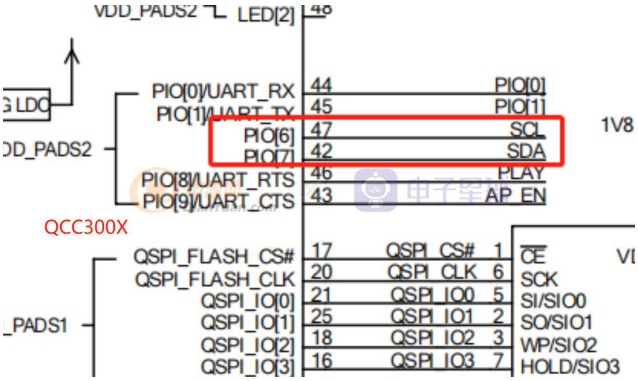
I2 C 通信接口都有上拉，如下图：



大多数 IO 口内部电路如下图，VDD 接 1. 8 V。



QCC300X 启动后 1.8 V 控制 LDO 使能端输出 3.3V，3.3V 通过电阻到 QCC300X I2C 端口 PIO6 \PIO7 约有 2.8V 电压，2.8V 电压再通过 IO 口的稳压二极管降压后向 VDD 端口供电将 1.8V 抬高到 2.35V。



二、关机后 1.8V 的 VDD 端口依然输出 2.35V 电压不变，待机电流有 180uA。

在关机后下正常情况下，因为滤波电容的存在 VDD 在 5S 内从 1.8V 降至 0V ,但是 3.3V LDO 通过 IO 稳压管抬高 VDD 电压到 2.35V，VDD 是 LDO 的使能端控制信号，形成自锁循环，这样就不能完全关机。

三、解决方案

LDO 的使能端信号改为 IO 口控制，在关机后可以快速拉低 LDO 使能端信号，让 3.3V 无输出，也不会抬高 VDD 电压。

留 言

互 动

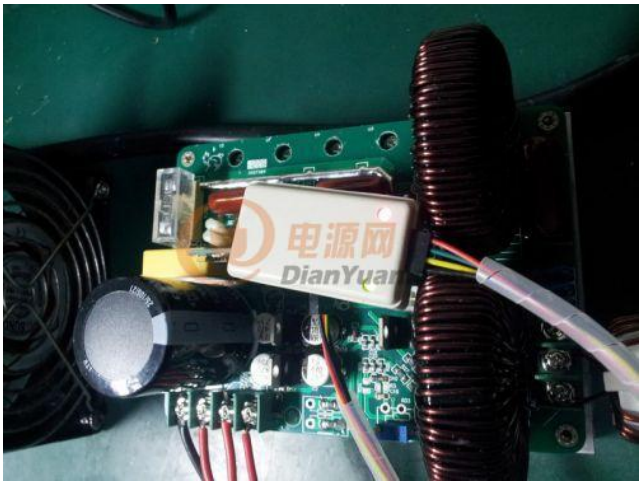
原来逆变器也可以这样玩！与主控板匹配的蓝牙模块做好了

文 / plc_avr

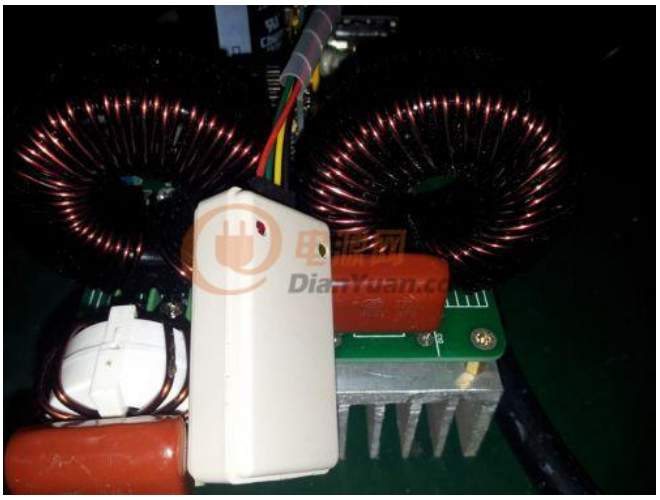
与主控板匹配的蓝牙模块做好了，原来逆变器也可以这样玩！从此俺的调试可以不用线了，哈哈。晒图！

1.天生的哥俩好 ,核心控制主角.....SPWM V1.2 主控板和蓝牙控制模块。因手机拍的角度有问题，实际的蓝牙模块比主控板要小的多。呵呵。各位将就着看吧。

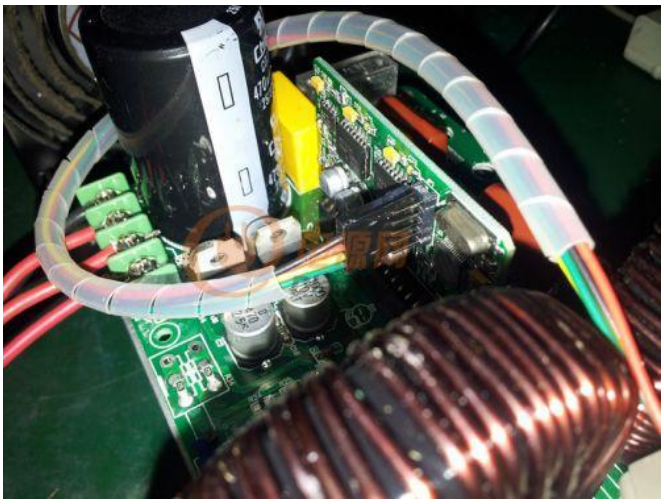
2.抗干绕稳定性测试，电感是电磁辐射大户，就靠它身上吧。链接依然给力，与上位机通讯没有丢包现象发生。



3.换个位置，依然给力。



4.与主控板接口特写。



至于输出的波形图我就不帖了。扫码继续阅读。

留 言

互 动

自制作：手机蓝牙逆变器

文 / HL_ZXM

近期做了一个创意出来，成果分享：

逆变器为 12V 正弦波逆变器 ,前级和后级采用隔离方式，DC12V 输入，输出 220V50HZ 正弦波电压可以轻松带起感性负载，最大功率 300W (足功率，峰值功率 600W)。本产品是真正意义上的蓝牙逆变器，可以无线遥控逆变器，无线上传、下传数据！不是市面上的用个蓝牙 MP3、蓝牙耳机加在逆变器里，那种蓝牙和逆变器半毛钱关系都没有。

1.4 位 LED 数码管显示：(1) “t” 温度显示；(2) “u” DC 电压显示；(3) “P” 输出功率显示；(4)循环显示。(具体显示方式可以通过手机 APP 设置！)

2.逆变器软启动功能：(1)2 秒钟；(2)1 秒钟；(3)0.7 秒钟 ;(4)0.35 秒钟 ;(5)取消软启动功能。(通过手机 APP 设置！)

3.可以定时关机逆变器，最长能定时 999 分钟：如，定时 30 分钟 ,在 30 分钟后逆变器就自动关机。(通过手机 APP 设置！)

4.可以定时关机和开机逆变器，定时开机最长 999 分钟关机最长 999 分钟：如，定时开 20 分钟关 50 分钟，逆变器就会开机 20 分钟后关闭，待关机 50 分钟后自动开机，一直循环。(通过手机 APP 设置！)

5.逆变器在与蓝牙通讯距离范围内 (10 米),可以通过手机 APP 随时 “打开” 或者 “关闭” 逆变 器。(通过手机 APP 的 “开” “关” 按钮操作)

6.手机 APP 上实时显示逆变器的 DC 电压、温度、输出功率等信息，用户一目了然，方便管理。

7.逆变器的 220V 输出可以任意短路、长时间短路 MOS 不会发热 (短路一天、一个月……)。因为短路电流是空载电流的一半这样 (如：空载电流 520ma，那么短路后的电流是 280ma 左右)，可以短路开机，在短路期间蜂鸣器一直循环响 2 声报警，短路排除逆变器自动恢复输出。

8.输入 DC 高压保护：输入电压大于 15V 时，自动关闭输出电压蜂鸣器一直循环响 5 声报警，待电压小于 13V 时自动恢复输出电压。

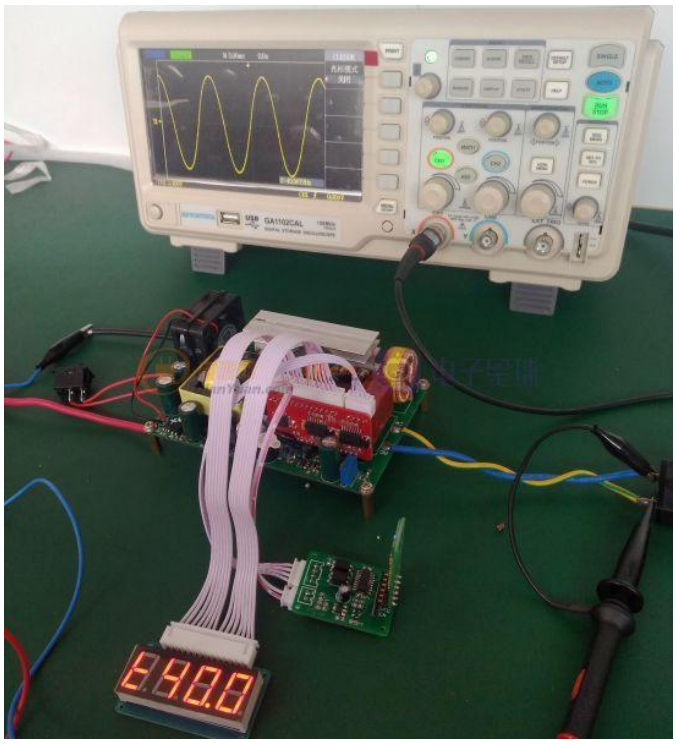
9.输入 DC 低压保护：输入电压小于 11V 时，蜂鸣器一直循环响 6 声报警，小于 10.5V 时，自动关闭输出电压，待电压大于 12V 时自动恢复输出电压。

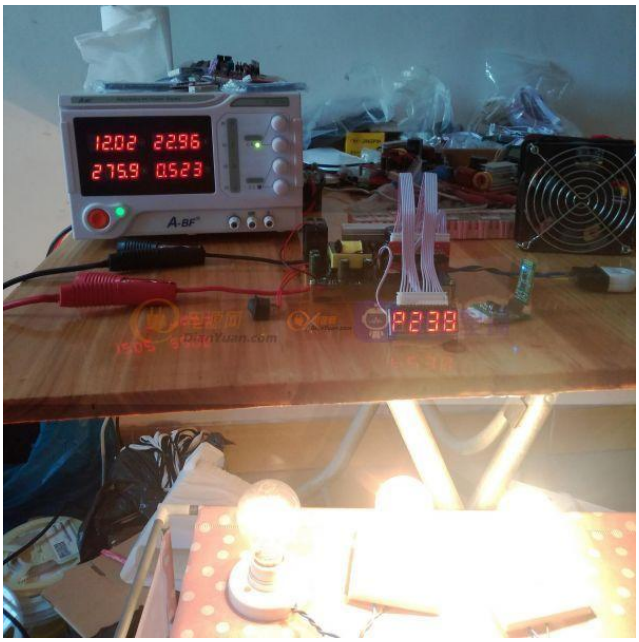
10.温度保护：主板温度大于 45 度，风扇自动开启，温度大于 80 度时关闭输出电压，蜂鸣器一直循环响 4 声报警，待温度下降到 45 度时，自动恢复输出电压。

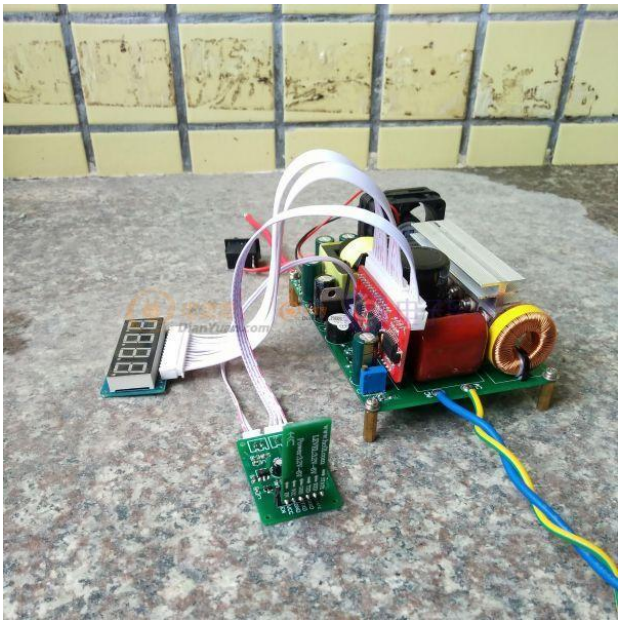
11.过载保护（过流保护、过冲保护）：逆变器的输出功率或者输出电流过大时（300W），逆变器连续 5 秒过载后，会自动关闭输出，蜂鸣器一直循环响 3 声报警。

12.智能驱动风扇：温度达到 45 度以上或者负载超过 100W 时会立即启动风扇。

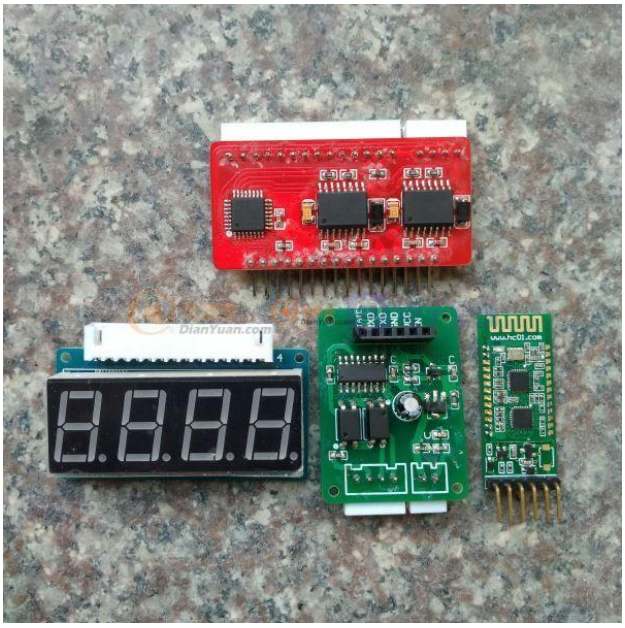
13.输出电压可调 ,最低调到 190V 输出 ,最高可以调到 270V 输出，本逆变器的 DC 输入不可以反接，反接就会烧 MOS 管、保险丝等前级元件。



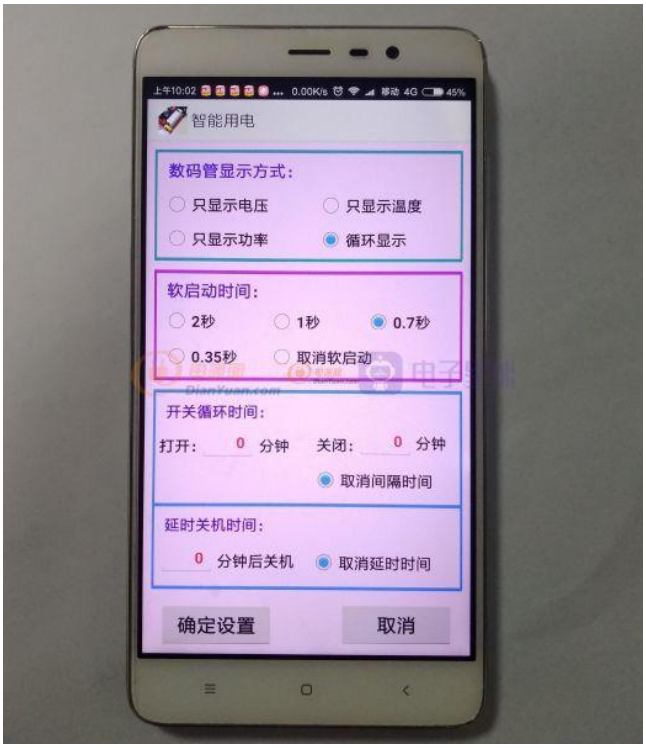





逆变器
APP软件
百度文库
今日头条
广东移动
百度
抖音短视频
蓝牙逆变器
高德地图
短信
拨号
联系人
相机




智慧用电
欢迎使用
输入电压：
12.0V
主机温度：
42.7℃
输出功率：
238W
打开
关闭
功能设置
结束程序



以上为个人意见！欢迎大家讨论！！！

留　言

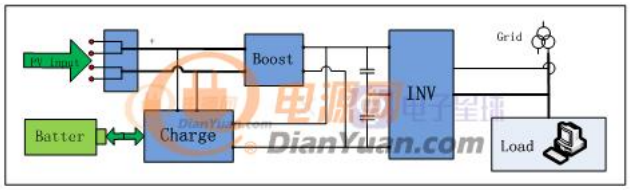
互　动

简单图解家用光伏储能逆变系统

文 / Richie_Li

近几年，随着光伏发电快速发展，家用储能系统也随着配套起来，本帖简单图解一下家用储能系统。

通过拆解几家光伏储能逆变器，经过分析，我认为光伏储能逆变器一般可分为 5 大功能模块：



1.Charge 模块：负责电池的充放电，以及接收指令，实现能量分配。

2.BOOST 模块：负责追踪太阳能面板的最大功率点。

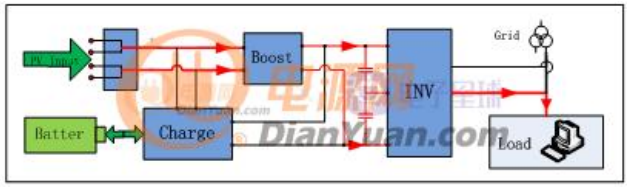
3.INV 逆变模块：将直流电转换为交流电输送出去；

4.市电电流采样模块：实时采集市电电流信息，将电流信息反馈给主控板。

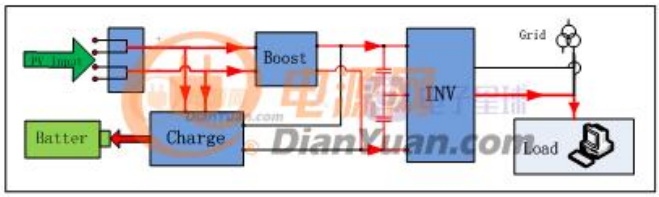
5.控制模块：实现 Charge 模块、BOOST 模块以及 INV 逆变模块之间的能量调度。

具体工作模式能量流动如下：

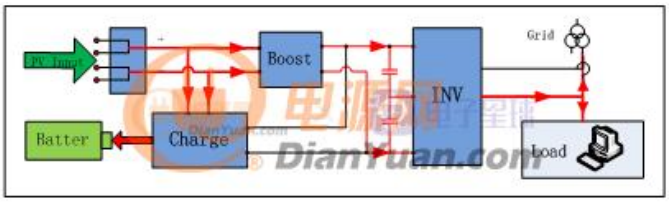
模式一：太阳能能量刚好与家用负载平衡



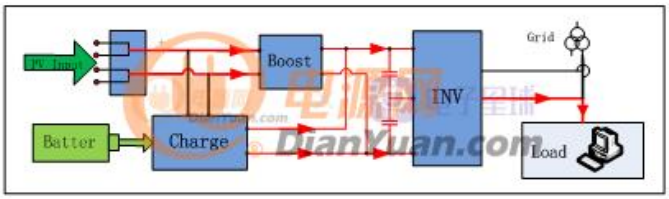
模式二：太阳能能量大于家用负载，其余能量给电池充电



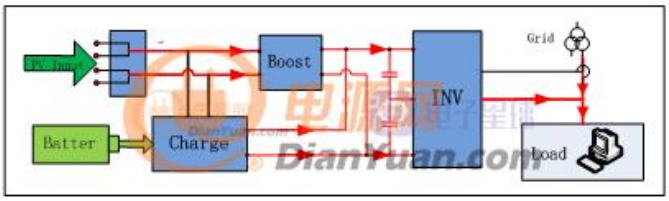
模式三：太阳能能量大于家用负载和电池充电，多余能量输送给电网



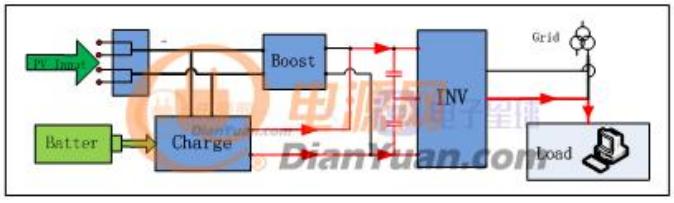
模式四：太阳能能量小于家用负载，电池放电供给负载



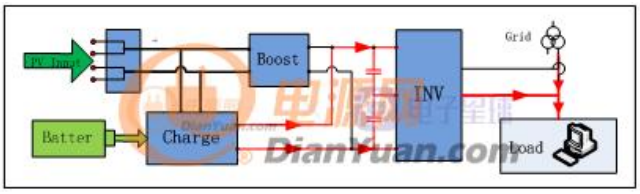
模式五：太阳能能量和电池放电能量小于负载功率，市电补偿所缺能量



模式六：晚上家用负载功率较小时，电池释放能量供电

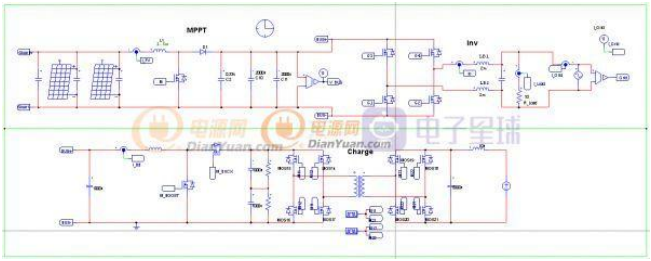


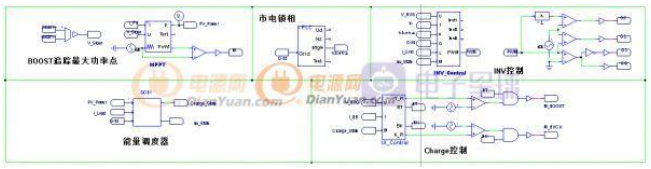
模式七：晚上家用负载功率较大时，电池释放能量不足，市电补偿所缺能量



通过图解，我们大致了解家用储能系统的基本构架。接下来，再进一步深入理解家用储能系统。

仿真原理图以及控制架构如下：





1.BOOST 做 MPPT；

2.H4 桥逆变输出；

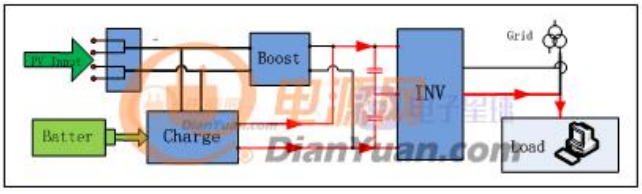
3.Boost_Buck 加上全桥隔离做电池充放；

4.能量调度器分配能量平衡；

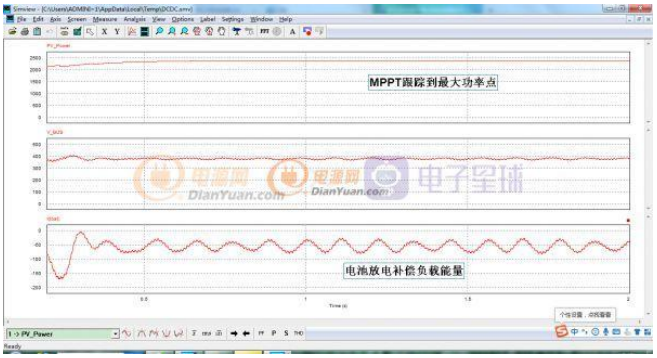
接下来进行带载仿真。

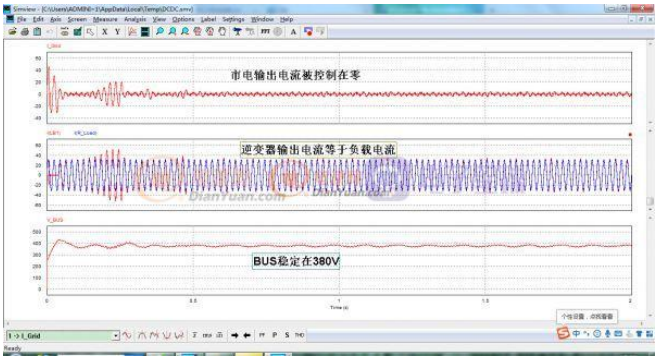
实验一 :光伏能量为 2500W ,负载 10R(功率 4840W)

能量分析：光伏能量不足以提供负载，电池系统需补偿能量



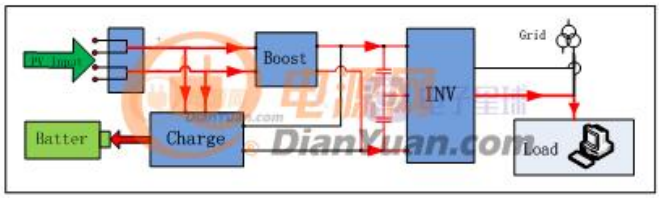
观看防真效果（从仿真结果看，效果很好）



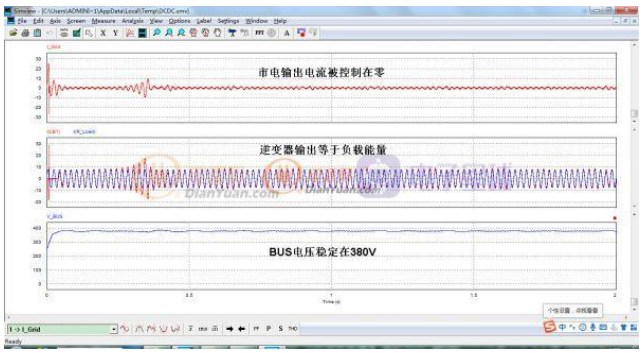


实验二：光伏能量为 2500W，负载 40R（功率 1210W）

能量分析：光伏能量大于负载，电池系统需吸收能量

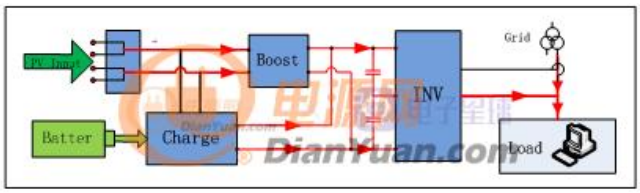


观看防真效果（从仿真结果看效果很好）

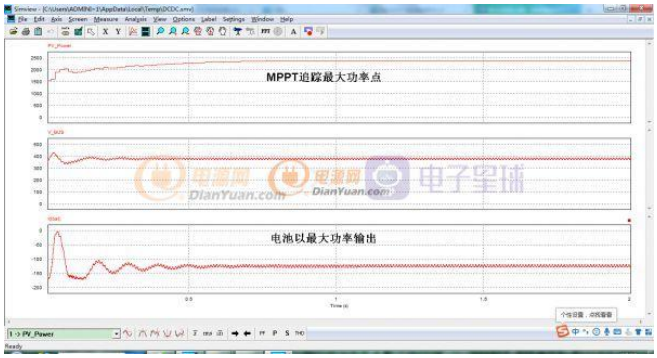


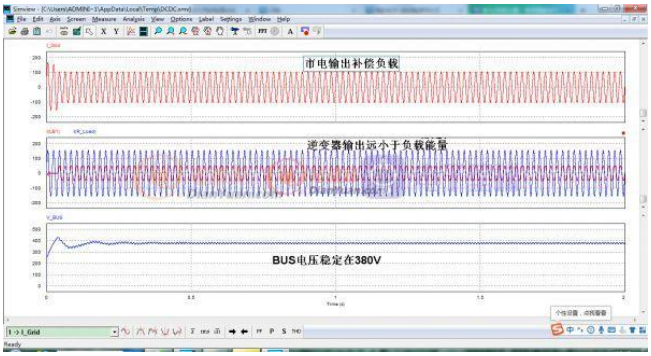
实验三 :光伏能量为 2500W ,负载 2R(功率 24200W)

能量分析：光伏能量远小于负载，电池系统最大功率输出，市电补偿负载



观看防真效果（从仿真结果看效果很好）





留 言

互 动

关于本刊电子版的版权声明

2022 电源网 VIP 会员技术专刊精选

DIANYUAN VIP Member Technical Special Issue

电源网

地址：天津市南开区清新大厦 A 座九层

电话：400-003-2006

邮箱：editor@netbroad.com

网址：https://www.dianyuan.com

创作者招募

微信联络：dianyuanqinghuai