分类号_____ 密级_____
UDC _____

西華大學

硕士学位论文

基于开源软硬件技术的电力电子嵌入式计算平台

作 者 姓 名： 王 伟
学科、专业：电力电子与电力传动
学 号：212011080804002
指 导 教 师： 杨燕翔
完 成 日 期： 2014.5.1

Classified Index:
UDC:

Master Degree Dissertation

Based on Opensource Technology Power Electronics Embedded Computing Platform

Candidate: Wang Wei
Major : Electrical Engineering
Student ID: 212011080804002
Supervisor: Prof. Yang Yanxiang

May, 2014

西华大学学位论文独创性声明

作者郑重声明：所呈交的学位论文，是本人在导师的指导下进行研究工作所取得的成果。尽我所知，除文中已经注明引用内容和致谢的地方外，本论文不包含其他个人或集体已经发表的研究成果，也不包含其他已申请学位或其他用途使用过的成果。与我一同工作的同志对本研究所做的贡献均已在论文中做了明确的说明并表示了谢意。

若有不实之处，本人愿意承担相关法律责任。

学位论文作者签名：　　指导教师签名：
日期：　　日期

西华大学学位论文版权使用授权书

本学位论文作者完全了解学校有关保留、使用学位论文的规定，在校攻读学位期间论文工作的知识产权属于西华大学，同意学校保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅，西华大学可以将本论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复印手段保存和汇编本学位论文。(保密的论文在解密后遵守此规定)

学位论文作者签名：　　指导教师签名：
日期：　　日期

摘　　要

在过去的 20 多年，国内外众多专家学者致力于嵌入式软硬件技术的研究和开发，嵌入式软硬件技术也得到了长足的发展。随着电力工业、微电子技术和软件技术的飞速发展，结合电力电子嵌入计算平台的行业特殊性，强弱电结合的专业性和开发研究过程的难度性，采用开源软硬件技术的电力电子嵌入计算平台终将会随着嵌入式技术的发展成为一个技术热点。本文基于开源软硬件技术的电力电子嵌入计算平台，以无速度传感器技术为例，对电力电子开源软硬件计算平台展开全面、深入的研究。

本文首先介绍目前的两种开源硬件，分析了它们各自的特点，并介绍了两种开源硬件平台应用于电力电子嵌入计算平台的可行性，以及在这样的平台或系统的可能作用。然后，文章深入研究了两种开源硬件对应的软件生态环境。它从主机端软件环境的搭建，用户定制的操作系统镜像以及 python 脚本的应用程序设计三个方面来介绍树莓派 RaspberryPi 的软件开发环境；从嵌入式操作系统 freeRTOS，任务调度器 scheduler，CMSIS 微控制器软件接口标准，arduino 驱动软件设计，bootloader 程序方面来介绍 Arduino 软件开发环境。通过介绍分析感应电机无速度传感器的研究现状，梳理感应电机无速度传感器技术中涉及到的数学理论和数学模型，对一个电力电子嵌入计算平台的典型应用场景做出背景介绍。

作者选取感应电机无速度传感器作为开源软硬件的电力电子嵌入计算平台的应用实例，针对实际的问题解决正式采用开源软硬件技术和开发电力电子嵌入计算平台的快捷性给出试验结果，并证实了本文提出的基于开源软硬件技术的电力电子嵌入计算平台的使用价值。

关键词：开源软硬件技术；电力电子技术；嵌入式计算平台

Abstract

In the past twenty years, a great number of experts and scholars at abroad and home have been devoting themselves to researching and developing the embedded software and hardware technology, and it has made great progress. With the rapid development of Electronics Industry, Microelectronic Technology and software technology, it will be quite popular to use the power electronics embedded computing platform of open source software and hardware technology due to the industry particularity of power electronics embedded computing platform, the professional of combining strong and weak electricity and the difficulty of development process. This thesis cites sensor-less technology as an example to study power electronics embedded computing platform deeply and comprehensively on the base of the power electronics embedded computing platform of open source software and hardware technology.

This paper firstly introduces the two kinds of open source hardware and their own features, as well as the feasibility to use the hardware to power electronics embedded computing platform and the possible value of using it at this platform or in this system. Then, the author researches the corresponding software ecosystem deeply. It introduces the software development environment from the aspects of the establishment of host software environment, the operation system image of customer-designed and the application programming of python script; it introduces the software development environment of Arduino from the aspects of embedded operation system freeRTOS, task scheduler, CMSIS microcontroller software interface standard, driver software design of arduino and bootloader procedure. This paper introduces the background of typical application setting for a power electronics embedded computing platform through analyzing the research status of induction machine sensor and teasing the mathematical theories and mathematical models involved in the induction machine sensor technologies.

The author of this thesis cites the induction machine sensor of the power electronics embedded computing platform of open source software and hardware as the application example. It concludes the experimental results through solving the actual problems and using the open source software and hardware technology as well as the quick of developing the power electronics embedded computing platform. The experiment proves the value in use of the power electronics embedded computing platform based on the open source software and hardware technology.

Key Words：Open Source Software Hardware; Power Electronics; Embedded Computing Platform

目　　录

1 绪论

近年来，随着微电子技术的向前发展和软件技术的推陈出新，开源软硬件技术得到了迅速的发展和广泛的应用，涌现了一大批优秀的软硬件产品。其中我们最熟悉的智能手机就是开源软件技术的成功典范，其核心安卓系统就是在 Linux 系统的基础上不断发展而来的。开源硬件技术发源于欧洲，并在最近两年逐渐扩散到世界各地，深受专业人士和广大业余爱好者喜爱，开源硬件技术领域的典型代表包括阿都诺 Arduino 和树莓派 Raspberrypi。当前在传统的领域，广泛采用的模式依然是根据项目不同定制专用的检测控制电路平台，再基于此硬件平台定制用户代码。这样的开发周期往往长达数年，且开发代码的质量也会由于开发者的水平不同而参差不齐，因此，电路和代码后续的维护性非常差，这一情况在电力电子领域和汽车电子领域体现更加明显。同时，电力电子嵌入式计算平台一般采用专用的 DSP 计算芯片和以它制作成的专用开发板。这样就必然导致项目开发的高成本，高费用且代码闭源。另外，专用开发板也会给后期计算平台的硬件扩展性，软件扩展性，智能化应用带来不少麻烦，因为专用开发板的学习者少，而且学习者之间也会因将自己学习成果都视为保密文件而少有交流。这样不仅会产生高成本，而且会阻碍技术的不断创新。这也是为什么开源软硬件技术在消费类电子领域开展的相对成熟，在专业领域却不够充分的原因。目前国外的开源软件技术，开源硬件平台正如火如荼地开展中，而国内在开源方面做的却十分欠缺，更谈不上应用在电机控制系统这样的电力电子和电力传动专业领域运用开源软硬件技术。本文就是在此背景下，以电力电子领域为例，开展基于开源软硬件技术的电力电子计算平台研究，并以感应电机的无速度传感器为实例展开研究。

1.1 电力电子技术的发展现状

科技日报 2014 年 2 月 21 日，题为：下一代电力电子技术--美国的就业稻草，提到美国总统奥巴马和美国能源部在 1 月份宣布在北卡罗来州立大学建立下一代电力电子技术国家制造业创新研究所，这是一个价值 1.40 亿美元的研究项目。这个先进制造机构将联合学术界，政府部门，工业界，在应用领域，包括：电力电子设备，电力网络，电动汽车方面革新能源的有效性。这所高校联合了电气，计算机，材料多个学科的全球顶级专家和知名企业，例如：在电力电子领域的知名公司 ABB, 汽车电子零部件汽车 Delphi，元器件供应商 Toshiba 均加入到这个行列中。从此可见，电力电子技术，之于学术界，是一个热门领域，对于产业界，对于解决就业，也是一个不错的行业。

在具体应用领域，本文特别是以感应电机无速度传感器为例展开，回顾国内外现代电力电子与电力传动技术的发展现状。

1.2 开源硬件技术的国内外发展现状

当前主流的国内嵌入式计算平台有：1）微控制器，也俗称单片机，涵盖从 8 位到 32 位各类型的单片机，如英特尔的 51 单片机，富士通 MB90F 系列，德州仪器的 MSP430，意法电子的 STM32，瑞萨半导体的 V850 系列单片机等；2）数字信号处理器，如德州仪器的 C2000 系列 DSP，ADI 公司的数字信号处理器 Blackfin 系列；3）ARM 公司为传统微控制器和数字信号处理设计一类芯片内核，两小类，包括 Cortex-M3，Cortex-M4。其中 Cortex-M3 主要面向传统微处理器，而 Cortex-M4 主要面向传统数字信号处理器领域，将来 M4 有整合 MCU 和 DSP 两款芯片的趋势。

开源硬件一般按照计算机系统，外围设备，业余无线电，计算机零件，电话，机器人等类型来区分。开源是一个新兴但又古老的话题，最早的打印机，电脑的设计原理图都是公开的。很多项目在最初的一段开发的时间都是采用开源的方式开发的，许多商业公司如苹果和三星，为了各自的商业利益选择闭源，闭源在这段时间确实为 IT 技术的发展，产品的普及提供了财力推动，有利于创新。但是随着时代的发展，闭源却又阻碍了小创新公司的创新和个体的创新，因此开源又再次被广大技术爱好者提上研究日程。

可见开源硬件在传统 ICT（Information Communication Technology）信息与通信技术领域一直以来都有研究，但是在传统的电气领域，特别是古老而新兴的电力电子领域，研究得并不多。

1.3 开源软件技术的国内外发展现状

开源软件技术，是指面向公众开放源代码的软件技术。开源运动的重要人物有，理查德·斯托曼，林纳斯·托瓦兹。其重要产物主要指 Linux，最早是由理查德·斯托曼在 1980 年代末期提出了 GNU 宣言，也就是要创造一套全新的，自由的，与原有 UNIX 兼容的操作系统 GNU，最早的操作系统内核为 Hurd。后来由于在 1990 年代初期，Linux 的兴起，后都转向新的系统，也就有我们现在所称呼的 GNU/Linux。到 2003 年，安迪·鲁兵创办安卓团队，后被谷歌收购，在 2007 年后公开面向移动设备发布的商用的基于 Linux 的自由开源的操作系统。

在桌面操作系统和移动设备操作系统迅速发展的同时，传统工业控制领域由于芯片技术的发展，内存容量从原来的几百个字节发展到现在几十 K，几百 K 字节，代码空间也相应的增加。运行在芯片上的软件部分已经可以从简单汇编语言程序发展到前后台 C 语言系统，以及后来的嵌入式操作系统。

嵌入式软件系统，特别是微处理器 MCU 与数字信号处理 DSP 的软件系统的开发，也出现了 ucos, uITRON 等商用操作系统。开源实时操作系统出现了 freeRTOS 这样的优秀操作系统。为适应嵌入式系统网络升级，也有 uIP 这样的面向微处理器的 TCP/IP 协议栈。为了面向便携设备，也有 Powermoding 设计。为了嵌入式软件的网络升级，也有 bootloader 的软件设计[1]。

1.4 电力电子嵌入式计算平台的国内外发展现状

电力电子嵌入式计算平台，主要是选用 DSP, 如，德州仪器的 TMS320F2812 或者选用 FPGA, 如 Altera 公司的 SOPC 作为主控器。

1.5 本文的主要研究内容

随着电力工业和 IT 技术的飞速发展，电力工业再也不是原来的一个地区几个电厂，一个城市几个变电站的模式，而是结合了最现代化的发电控制技术，变电控制技术，微机综合保护技术，配电网自动化技术的各类电力电子嵌入式系统的广泛应用，具体是指广泛使用的电力测控仪表，计量仪表，控制和补偿设备。

本课题来源于研究生阶段的工程实践，但不拘泥于工程实践，它结合专业的特殊性和开源社区的广泛实践，作出有益的尝试。本课题以感应电机无速度传感器为例，介绍了基于开源硬件的电力电子嵌入计算平台的设计与实现。

本文的具体工作内容如下：

第二章讲述实际应用中的感应电机无速度传感器问题介绍，数学理论背景。

第三章开源硬件技术，调查研究全球范围内几款主流的开源硬件技术，从性能参数，价格，扩展性等各个方面进行比较。

第四章开源软件技术包括开源操作系统 Linux 和开源实时操作系统 freeRTOS。

第五章讲述一般的电力电子嵌入式平台的主要组成部分电流电压采样电路。

第六章是全文总结与展望。

2 感应电机无速度传感器的模型分析

本论文将以感应电机无速度传感器为应用实例展开前面几章讨论的开源软硬件技术的电力电子嵌入式计算平台。通过模型分析，重温感应电机无速度传感器的数学模型，并分析该数学模型在该电力电子嵌入式计算平台应用的可行性。感应电机无速度传感器本身是为了节约成本，特别是在速度精度要求不是特别高的感应电机调速系统中的。无速度传感器虽然节约了传感器，但是没有传感器的控制系统，反过来要求控制系统能够有性能良好的无速度传感器，能够很好地估算出实时的速度信息，便于系统开展实时控制工作。开源似乎意味着不稳定不安全，但是，开源就意味着开放，共享，在学术界和工程界形成开放的思维环境。通过价格便宜，随手可得的开源软硬件技术环境来验证学术界不断提出的新理论和新方法，将实际验证的结果迅速反馈给学术界的同仁，实现产学研一体化的快速结合。

要实现交流电机闭环控制，获得转子实时转速是一个必要的前提条件。传统的方法为，通过物理的速度传感器动态检测转速的实时转速，来测量转子的转速。这种方法有两个缺点：

1）速度传感器的成本较高。对于小型的电机控制系统，如果采用物理的速度传感器，整个系统的成本将大大增高。2）速度传感器容易损坏。速度传感器属于精密仪器，而电机的运行环境往往都是高振动、高污染，速度传感器较容易损坏。速度传感器的损坏为更换维护带来麻烦。

基于以上两个原因，不少学者在电机转速监测方面投入了很多的时间和精力，其中一个方面是无速度传感器的解决方案。

2.1 无速度传感器的研究现状

自上世纪七十年代开始，不少专家学者在无速度传感器方面投入研究。1975 年，A.Abondanti 教授等学者基于交流电机的稳态数学方程，推导出了交流电机转差频率的估计方法。这是交流电机无速度传感器控制方面，专家学者们做出的第一次研究，电机的调速比达到了 10:1。但该研究中的局限性在于，1. 电机的可调速度范围较小；2. 电机调速的动态指标质量无法完成；3. 感应电机的调速精度将因为电机的参数调整，而出现影响较大的情况。由于在模块推导中采用了交流电机的稳态数学方程，因此以上缺点无法在该方法中改进。在之后几年的学术研究成果中，一些学者尝试改进，但由于仍然采用电机的稳态数学方程，电机转速的估计精度与质量都没有得到较好的提高[1]。

M.Ishida 等学者于 1979 年，从一个新的方面出发，通过采用电机转子齿槽谐波的方法检测电机的转速。该方法的成果为在电机转速范围为额定转速到 300 转每分内，转速检测均能取得一定的精度。虽然精度仍然需要提升，但在转速估计方法中，未使用电机的稳态数学方程，为电机转速估计的研究给出了新的思路[3]。

R.Joetten 于 1983 年将无速度传感器应用于电机的控制，进一步推动了电力传动理论技术的发展。感应电机控制技术的逐步成熟和微控制器控制技术的发展，促使更多的学者将精力运用到感应电机无速度传感器控制技术的开发之中，由此许多用来估计转子磁场及转子转速的方法及模型被相继提出。从电机数学模型理想化程度方面，无速度传感器技术解决方案可分为以下两类：

1）基于电机理想数学模型的解决方案。电机的理想数学模型是指，预先提出一些假设条件，然后在这些假设条件基础上，根据电机电磁特性的动态方程，推导出的电机数学模块。其中所做的假定为：感应电机的三个绕组电圈完全一样，磁场沿一定方向按正弦规律分布；不考虑谐波磁场，只考虑基波磁场的作用；也不考虑磁滞、涡流、磁路饱和；不考虑电机绕组的其他效应[2]。该方法的缺点在于：1. 模块准确度受电机参数变化的影响比较大；2. 在同步频率为 0 或者在电机转速较低时，存在很难解决的稳定性问题。

2）感应电机的非理想数学模型的解决方法。从一些被忽略的感应电机非理想特性中寻找思路，如磁路饱和、集肤效应、人造转子凸极转子偏心齿槽效应等。这种采用非理想特性的办法，来检测电机转子位置、电机速度、以及转子磁链位置，对电机的参数变化具有较强的稳定性。齿谐波方法是不能用于低速范围，其它几种方案都可以用于低速范围。但这几种方案的缺点为，都对采样有较高的精度要求，并会不同程度地受负载或工作点影响[4]。

随着先进控制理论的发展，这些理论知识被应用到传统的电力电子与电力传动系统中，特别是速度的估计中，实现电机无速度传感器的控制。虽然这方面的应用正处试验阶段，实际应用较少。但相信随着研究的进展及科技的发展，会对无速度传感器矢量控制系统产生深远的影响。

2.2 无速度传感器的模型参考自适应法转速估计

S.Tamai 于 1987 年首次将模型参考自适应理论引入到交流传动，它的初衷是为了降低控制系统对过程参数变化、未建数学模型部分动态过程的影响。每次过程动态有所变化时，系统将能够知道这次变化，然后根据变化相应实时地调节系统的参数，进而保证控制精度。在之后的研究中，C.Schauder 第一次在异步电机转速辨识中使用模型参考自适应法，该方法基于电机稳定性理论，设计出异步电机转速辨识的方法。这种方法不仅

从根本解决了电机速度辨识的稳定性问题，且具有静态误差小、动态性能好等优点。该方法在研究完成后，由于其方案的优越性，得到迅速地应用与发展[6]。在目前的实际应用产品中，采用模型参考自适应法来估计电机转速仍然占相当大的比例。

模型参考自适应法转速估计是指，基于模型参考自适应控制理论，将含有转速的数学模型作为可调模型，同时将不包含电机转速的数学方程作为参考模型。这两个模块在数学上看是类似的，因而他们具有相同的输出量。两个模块的输出量前者作为参考值，后者作为目标值，实际调节过程中，这两个模块输出量会出现差值，这个差值便被作为有利误差信号。根据此误差信号，搭建自适应规律，便可实时调节可调模型的转速，从而能够使可调模块的输出量趋近于参考模型的输出，达到辨识转速的目的。这种计算方法通过不断的在线调速，实现了对转速的闭环观测估计，具有很强的抗干扰性，具有较高的估计精度，有良好的应用意义。该方法的不足为：1.感应电机数学模型的参数变化对其比较大的影响，需要同时估算转子磁通， 2. 在一些速度范围内时，电机定子的反电动势较小，从而导致估计的转速质量不好[7]。

模型参考自适应控制系统 Model Reference Adaptive System， 简称 MRAS， 是在广泛使用的自适应控制系统。模型参考自适应系统一般由以下几部分组成的：参考模型、可调模型、控制器、被控对象和自适应机构。参考模型的作用为：计算基本回路的期望性能，其中基本回路由被控过程和可调模型构成。控制目标为，在系统运行中，达到参考模型与被控过程两者间保持动态一致性。如果二者的输出有一定的误差，则系统可调整被控过程，也就是可调模型的一些参数，从而使实际输出状态与参考模型的输出状态的偏差最小化，这些步骤通过自适应机构完成。一个模型参考自适应控制系统如图 2.1 所示。该系统是一个典型的并联模型，在该系统中，参考模块是期望的输出和状态，这个是在设计阶段完成。然后将参考模型的输出状态与可调模型的输出状态作比较，二者的差值作为广义误差信号。以广义误差信号作为输入，自适应机构通过一定的规则算法动态地修改可调模块调节器的部分参数，或者构造一个新的信号源，作为附加输入输送给可调模型。由此动态的调整可调模块的输出状态，使其逐步趋近于参考模型的输出状态。

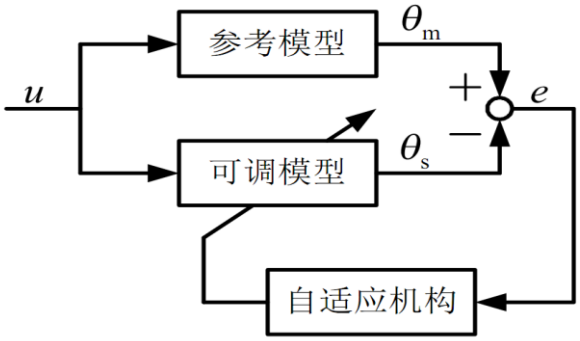

图 2.1　并联型模型参考自适应系统
Fig 2.1　Parallel MRAS

MRAS 的一个最大的优点为：抗干扰能力强，自适应能力高。其中实际可调模型与参考模型的之前的差值，通过自适应机构作为输入信号来调整可调模型，因此调控的实时性很强，并且是通过误差信号进行调节，因此调控肯定能够得到收敛。

其中参考模型的数学表达式为

$$\dot{x} = A_m x + B_m u \tag{2.1}$$

$$\theta_m = Cx \tag{2.2}$$

可调模型的数学表达式为

$$\dot{y} = A_m y + B_m u \tag{2.3}$$

$$\theta_s = Cy \tag{2.4}$$

其中，A_m、B_m 是恒定矩阵，x、y 为 n 维状态变量，u 是 m 维输入矢量，$A_s(t)$、$B_s(t)$ 是时变矩阵，θ_m 和 θ_s 是输出矢量，C 是恰当维数的输出矩阵，对参考模型和可调模型都选用矩阵 C。

定义状态广义误差为

$$e = x - y \tag{2.5}$$

输出广义误差为

$$\varepsilon = \theta_m - \theta_s = Ce \tag{2.6}$$

控制系统的性能与广义误差相关，二者存在一定的数学关系，比如可以用以下数学公式作为系统的一个性能指标

$$\sigma = \int e^T e(\tau) d\tau \tag{2.7}$$

系统控制的目标为使上式定义的值达到最小，这通过自适应机构完成。

对于 MRAS，在参考数学模型的敲定后，一个最重要的步骤为制定自适应规律。自适应律的制定过程中，为了简化设计，可作如下条件：

(1) 参考模型是确定常数的线性系统

(2) 可调模型与参考模型一定有相同的参考

(3) 在参数自适应的条件，自适应调节机构应该能够调节可调模型的所有参数

(4) 在自适应调节过程中，可调系统的参数仅随自适应机构的调节而变化，而不受其他因素的影响

(5) 只有输入矢量作用到系统上，而没有其他输入信号

(6) 两个模型参数之间的差别是不确定的

(7) 状态误差和输出误差是能够测试出来的

以上假设是理想情况的情况，大多数系统是无法真正满足以上条件。在这种情况下，有两种方法：1. 采用类似设定的方式；2. 调整的参考模型和可调模型，从而使用他们尽可能的满足以上条件，可以使自适应系统的设计变得简单一些。自适应控制机构的设计方法有：参数最优化法，李雅普诺夫稳定性理论法和波波夫超稳定性理论法。

基于电机数学方程的模型参考自适应系统。

根据之前理论的介绍，要构造感应电机无速度传感器的模型参考自适应系统，必须通过状态反馈来实现。感应电机状态反馈的前提条件是，获取相关的状态变量 x(t)。在实际的电机控制系统中，有些状态变量可通过测量得到，但有些则不能，此时只能通过其他方法间接得到所需的状态变量。一种常用的方法是，利用系统已知的状态信息，通过建立数学模型，计算出未知的所需的状态变量。这种方法也被称为系统状态估算法，用于状态估计，也即计算系统未知状态的子系统叫做状态估算器。

在现有的研究成果中，在当电机的转速通过 MRAS 方法估算得出后，以该转速信息为基础，可变化出更多需要的其他状态变量，作为参考模型的输出。一些文献中通过使用参考模型与可调模型的电磁转矩差值来计算电机的转速，在电机转速较低的情况下得到了较好的结果。一些文献通过使用电机的转子磁场，来作为状态广义误差，由此进行转速估计。一些文献通过使用参考模型与可调模型的定子反电动势差值，来进行转速估计。以上方法的缺点在于，当电机的转速较低时，或转速较接近零转速时，转速估计的精度及系统的动态性能存在较大的误差。文章将感应电机电流作为系统模型输出，搭建模型参考自适应系统，以提高估计精度。

在其他的坐标系下，感应电机的状态数学表达式为

$$p\begin{bmatrix} i_s \\ \varphi_r \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}\begin{bmatrix} i_s \\ \varphi_r \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u_s \tag{2.9}$$

$$i_s = Cx \tag{2.10}$$

以上式子中，定子电流 $i_s = \begin{bmatrix} i_{sa} & i_s \end{bmatrix}^T$，转子磁链 $\varphi_r = \begin{bmatrix} \varphi_{ra} & \varphi_r \end{bmatrix}^T$，定子电压 $u_s = \begin{bmatrix} u_{sa} & u_s \end{bmatrix}^T$，p 为微分算子。

$$A_{11}=-\{R_s/(\sigma L_s)+(1-\sigma)/(\sigma\tau_r)\}I \tag{2.11}$$

$$A_{12}=\{L_s/(\sigma L_s L_r\tau_r)\}I-\{L_m\omega_r/(\sigma L_s L_r)\}J \tag{2.12}$$

$$A_{21}=(L_m/\tau_r)I \tag{2.13}$$

$$A_{22}=-(1/\tau_r)I+\omega_r J \tag{2.14}$$

$$B=1/(\sigma L_s)I \tag{2.15}$$

$$C=\begin{bmatrix}I & 0\end{bmatrix} \tag{2.16}$$

其中 I 和 J 分别为单位矩阵和斜对称矩阵，如下式所示

$$I=\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix},\ J=\begin{bmatrix}0 & -1\\ 1 & 0\end{bmatrix}$$

式中各个符号意义如下所示

定子电阻 R_s

转子电阻 R_r

定子电阻 L_s

转子电感 L_r

互感 L_m

转子时间常数 $\tau_r=L_r/R_r$

漏感系数 $\sigma=1-L_m^2/(L_s L_r)$

转子角速度 ω_r

由以上数学公式，可看出电流 i_s 和 Phir 是状态变量，电压是输入量， 电流 i_s 又被作为输出量。

式(2.9)为电机的状态数学方程，此处作为模型自适应系统中的参考模型。从该式可看出，A12 与 A22 中含有电机转速 wr 这个变量，通过将转速状态观测值替换实际转速值，则转速估计系统中的可调模型为

$$p\begin{bmatrix}\hat{i}_s\\ \hat{\varphi}_r\end{bmatrix}=\begin{bmatrix}A_{11} & \hat{A}_{12}\\ A_{21} & \hat{A}_{22}\end{bmatrix}\begin{bmatrix}\hat{i}_s\\ \hat{\varphi}_r\end{bmatrix}+\begin{bmatrix}B\\ 0\end{bmatrix}u_s \tag{2.17}$$

其中“ˆ”表示转速观测值。

根据 MRAS 的原理，通过使用参考模型与可调模型的差值，作为调节可调模型参数的依据。所以此时的问题为选取哪个参数作为误差来源。考虑到易获得性的影响，由于电机的定子电流 i_s 测量较为方便，因此本文选择电机的定子电流作为模型参考自适应系统的误差来源。将定子电流实测值与观测值作差值，将式(2.9)和(2.17)的电流项相减，由此引入反馈矩阵 G

$$p(i_s-\hat{i}_s)=A_{11}(i_s-\hat{i}_s)+A_{12}\varphi_r-\hat{A}_{12}\hat{\varphi}_r+G(i_s-\hat{i}_s) \tag{2.18}$$

式中 G 表示观测增益矩阵，如下式所示

$$G=\begin{bmatrix} g_1 & g_2 \\ -g_2 & g_1 \end{bmatrix}^T \tag{2.19}$$

在实际系统中，电机转子磁链无法直接通过测量得到，磁链与电流的关系如下式所示

$$p\hat{\varphi}_r = A_{21}i_s + A_{22}\hat{\varphi}_r \tag{2.20}$$

由于定子电流 is 可通过测量得到，故通过式(2.21)可求出转子磁链 phir，并且可假设转子磁链的观测值和实测值相同，由此式(2.12)可变为

$$\begin{aligned} p(i_s-\hat{i}_s) &= A_{11}(i_s-\hat{i}_s)+G(i_s-\hat{i}_s)+(A_{12}-\hat{A}_{12})\hat{\varphi}_r \\ &= (A_{11}+G)(i_s-\hat{i}_s)+(A_{12}-\hat{A}_{12})\hat{\varphi}_r \\ &= (A_{11}+G)(i_s-\hat{i}_s)+\{L_m/(\sigma L_s L_r)\}(\omega_r-\hat{\omega}_r)J\hat{\varphi}_r \end{aligned} \tag{2.21}$$

从上式可看出，转速自适应律的实现原理为：首先计算广义误差信号，由此估算得出电机转速 ω_r，然后将 ω_r 代入电机可调模型中，即可计算出电机定子电流的估算值 i_s。当估算电机转速接近于实际电机转速时，即可调模型的系统特性接近于参考模型的系统特性时，广义误差信号就会接近于零，估计电流也将接近于实际电流。整个并联型 MRAS 结构图如图 2.3 所示。

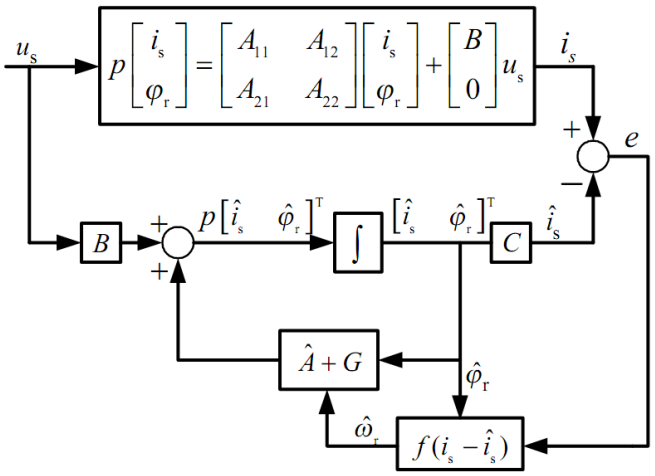

图 2.3　用于转速估计的并联型 MRAS 结构图

Fig 2.3　Parallel MRAS diagram used for speed-estimation

从以上推导可知， MRAS 方法的优点为：1. 保证系统有较好的稳态性能；2. 系统具有良好的动态性能。由于以上两个优点，MRAS 方法转速估计被大量应用于交流电机控制系统中，并能够取得较好的转速估计效果。

2.3 本章小结

本章首先回顾感应电机无速度传感器的研究现状，然后讲述无速度传感器的模型参考自适应法转速估计。

3 开源硬件技术

从 18 世纪富兰克林的电学发明到 19 世纪爱迪生的电灯发明，再到 20 世纪的计算机的发明，产生了众多发明家和极客的个人探索，而每一项新的发明和创造又都是在前人的基础上改进和创新的。开源，在成语“开源节流”中有开发水源，节制水流，顾名思义，在电子电气和计算学科领域的意思是开发技术资源。开源更是加快了发明的迭代效应，推进了新工业革命的发展[8]。

最近几年开源一词在电子电气和计算机领域中主要应用于硬件领域。开源硬件更是 2012 年之后的一个研究热点。本文选择两个最有代表性的平台进行介绍；这两块开源硬件平台是树莓派 RaspberryPi 和阿都诺 Arduino。它们在电力电子嵌入式计算平台中的作用是相互补充而不是相互替代；RaspberryPi 负责上位机监控和数据云端分析接口，而 Arduino 主要负责传统的单片机和实时信号处理。本章以下部分主要是详细介绍这两个硬件平台的基本情况。

3.1 开源硬件树莓派简述

树莓派主要负责上位机监控，网络传输数据和瘦服务器端。服务器现行主流的都是采用 Linux 软件。运行完整的 Linux 系统对硬件系统也有一定的要求，具体硬件参数如表 3.1

表 3.1 Rapberry Pi 的硬件参数列表
Table 3.1 Raspberry Pi hardware parameter list

编号	参数	说明
1	芯片	Broadcom BCM2835
2	处理器	700 MHz， ARM11 芯片核心
3	图形处理器	Broadcom VideoCore IV，OpenGL ES 2.0， 1080p30 h.264/MPEG-4 AVC high-profile decoder
4	内存	256 Megabytes (shared with GPU)
5	USB	2 (via integrated USB hub)
6	视频输出接口	RCA 接口, HDMI 接口
7	音频输出接口	3.5 mm jack, HDMI
8	板上存储介质	SD / MMC / SDIO card slot
9	网络	10/100 Ethernet (RJ45)
10	外设	8 × GPIO, UART, I2C bus, SPI bus
11	电源等级	500 mA (2.5 W) 700 mA (3.5 W)
12	电源接口	5 volt via MicroUSB or GPIO header
13	尺寸大小	长 85.60 × 宽 53.98 mm (3.370 × 2.125 in)
14	操作系统	Debian 的 GNU/Linux

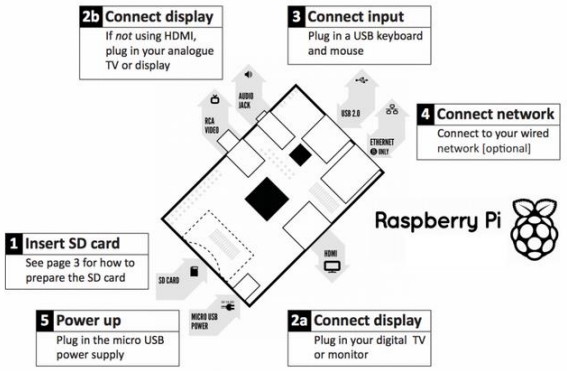

图 3.1　Raspberry Pi 外部接口框图

Fig.3.1　Rasperry Pi external interface diagram

本文选择树莓派的理由是：在短短两年内，单一产品以开源社区的方式卖出了上百万台，且这个数字还在不断上升。RaspberryPi 的单价被控制在 35 美金（约合人民币 200 元）的情况下，实现了一个完整的 Linux 系统，且开源大部分资料。为了能够成功移植 Debian Linux, 开源社区的志愿者对浮点处理相关功能做出了优化。

在传统的工控设计中，一般都会在嵌入式仪表旁放置一台安装有 windows 系统的工控机，然后，采用组态软件或者虚拟仪器的方式，实时显示从嵌入式计算平台传输上来的数据，并实时处理与控制。然而，文中的 Raspberrypi 在电力电子嵌入式计算平台应用现场就是一台没有显示器的工业监控。它通过有线或者无线网络接入 ethernet, 在网络中其他主机可以通过 mstsc 远程登录主机的方式登录到监控主机 Raspberrypi, 这样就能以图文的方式看到来自 arduino 的实时处理数据。

开源硬件越来越明显的优势是每一种硬件开源都有以供开发人员修改提升的性能和很强的扩展性，这样有利于极大地提高产出速度。树莓派（Raspberry Pi）就是一款这样的优秀开源硬件嵌入式平台，并且它依托于 Linux 开源软件，使其更加通用易用。Raspberry Pi 作为拥有 ARM11 与 GPU 两个处理器以及很多外围设备控制器的嵌入式硬件平台，能够成为一款流行的小型计算机系统的原因是其丰富的软件资源的决定性作用。这一小节将重点介绍 Raspberry Pi 的软硬件资源，以及如何利用这些资源打造自己的 Raspberry Pi 软件系统，并将其应用于电子电力系统。我们可以从官方网站 http://www.raspberrypi.org 上获得树莓派的硬件设计原理图。

本小节将重点介绍树莓派平台上的硬件接口。树莓派的硬件接口主要包含以下几个部分：输入接口，输出接口以及扩展接口与通信接口。输入接口主要包含两个 USB2.0 接口，树莓派的系统软件支持 USB 鼠标与键盘，并支持 USB 无线网卡。输出接口主要包含视频输出 AGV 和 HDMI，音频接口标准音频输出口。通信接口主要是网络接口。另

外，电路板上还有一个 SD 卡插槽，树莓派是将 SD 卡作为他的存储介质。扩展接口主要扩展了嵌入式系统中一些需要的常见接口, 如 I2C，SPI，UART。扩展接口如下图所示。

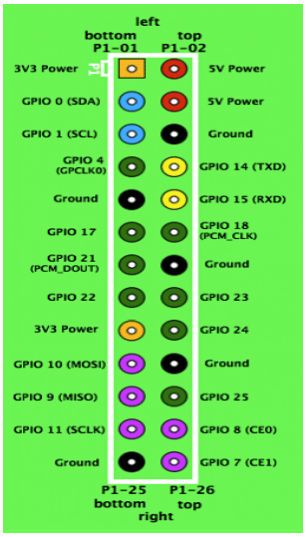

图 3.2　扩展接口定义图

Fig.3.2　Extension interface definition picture

扩展接口对于电力电子系统非常重要，因为它是树莓派与其他嵌入式控制器通信的桥梁。执行 IPC 协议后可以通过扩展接口使得树莓派与其他单片机控制器来进行通信，从而实现大系统来控制小系统的功能。这样就使得树莓派能更容易应用在电力电子系统中。当然，树莓派也可以通过扩展板的形式与其他单片机系统进行连接。如下图树莓派和阿都诺的转接板：

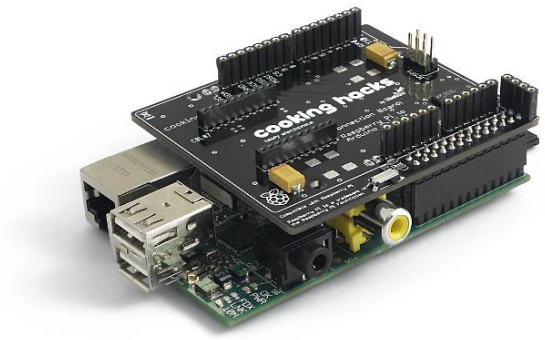

图 3.4　树莓派和阿都诺的转接板

Fig.3.4　Raspberry Pi and Arduino Adapter board

3.2 开源硬件阿都诺简述

阿都诺平台是便携、灵活组合且容易上手的开源硬件平台，包括硬件和软件。Arduino（意大利语），是 11 世纪古代意大利国王的名字。Arduino 最初是为了教学而设计的 IO 控制器，于 2005 年被 Massimo Banzi 和 David Cuartielles 商业开发，并采用 Arduino 作为产品的名字，慢慢地得到了世界各地的广泛认可。过去的几年已经衍生出各种类型 Arduino 的控制器，据统计，瑞萨半导体 RX60，德州仪器 MSP430,微芯的 PIC 单片机，意法半导体的 STM32，新塘半导体等众多厂商均有参照阿都诺 Arduino 设计的开发原型板。最近，Intel 也参与到其中，并发布了一款名叫伽利略 Galileo 的阿都诺 Arduino 开发板，它是第一款基于 Intel 架构的，硬件和软件与阿都诺都兼容的开源硬件[9]。

在以 Cortex-M 为内核的单片机中，意法半导体一直走在 Cortex 系列芯片的前列，其中具有 DSP 功能的控制器 ARM Cortex-M4 STM32F4XX 系列芯片最受人关注。它的芯片主要特性如下：具有浮点运算单元的中央处理器；最高主频可以高达 168Mhz，3 路 12 位模拟数字转换器，15 路通信接口，包括 3 路 I2C，4 路 USART/2 路 UART，3 路 SPI，2 路 I2S,2 路 CAN。高级连接包括 USB2.0 设备或者 OTG 设备，10/100M 以太网控制器。在数字信号处理领域大有代替专用 DSP 的趋势[10]。Embedded Pi 是目前在开源硬件社区中做得最好的 STM32 平台的 Arduino,因为它同时兼顾了 RaspberryPi 的连接。

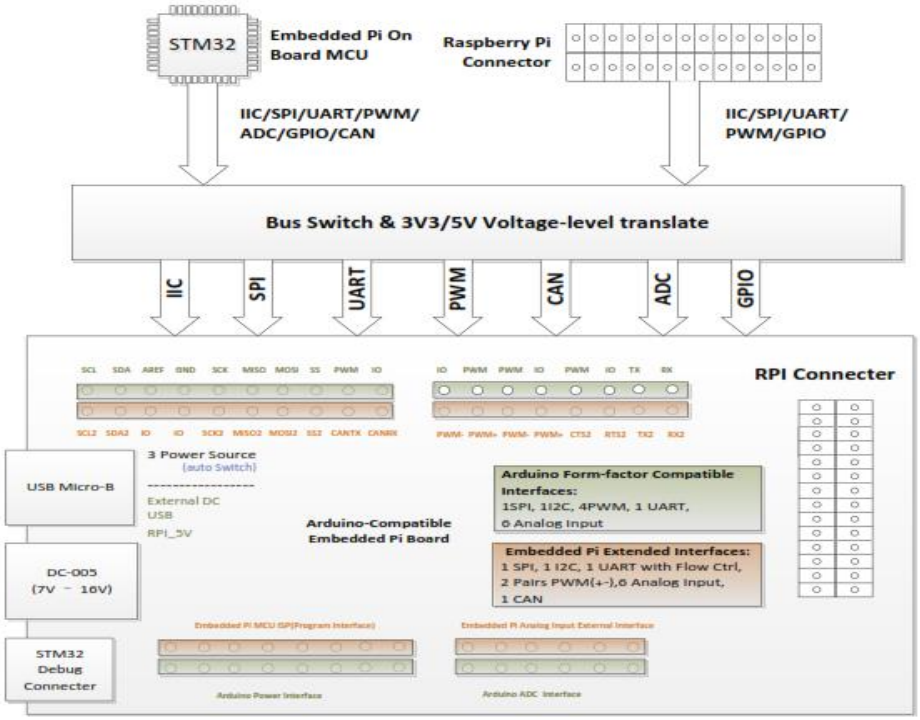

图 3.5　Embedded Pi 的硬件结构框图

Fig.3.5　The hardware diagram of Embedded Pi

由于 STM32 芯片间的管脚是相互兼容的，所以焊接过程中，可以轻易用 Cortex-M4 系列替换原设计中的 Cortex-M3 系列芯片。

本论文最大的创新不是应用 Arduino 的开放方式来开发典型的嵌入式软件，而是采用 Arduino 提倡的这种开源硬件的组件化思想。这为电力电子嵌入计算平台的开发过程，不断叠加传感器电路、控制电路和 IO 扩展电路的需要提供一种便捷方式。

什么是 Arduino shiled?

开源硬件平台最大的一个亮点是可重复使用硬件。它就如 LEGO 积木模块一样，可以不断重复使用，任意拼接进行创新。接下来，我们将 arduino 的开源盾板放在一起给大家介绍。

通用和共享是开源的最大精神。在 arduino 上使用过的开源盾板可通过一个简单的转接为 Raspberry Pi 所使用。而且 arduino 社区的兴起比 Raspberry Pi 社区早几年，所以它的资料相对完善一些，受众也相对较多，且积累的智力财富更多。据统计，arduino 的开源盾板有 400 件以上，提供者包括个人和超过两百家的供应商。

3.3 本章小结

本章首先从树莓派和阿都诺两款开源平台展开讲述开源平台的特性，然后将电流电压采样电路，电力电子控制电路如何与上述两款平台结合起来，最后讲述开源硬件依然需要考虑低功耗设计。

4 开源软件技术

开源项目最大的特点就是社区开发与志愿者贡献。我们在基于开源软硬件技术的电力电子嵌入计算平台的软件实现部分，最重要的是如何实现开源的精神，并充分利用好开源社区服务器，如服务器 github,代码版本管理工具 git（这些工具是全球顶级的开源项目都采用的模式）。开源软件技术同样实用于专业嵌入式系统领域。

4.1 开源软件技术的介绍以及发展

随着计算机硬件技术的发展，适应摩尔定律的硬件的发展速度使得 CPU 的处理速度越来越快[17]。计算机的硬件性能也越来越强大，但是软件的发展却远远滞后于计算机硬件的发展。软件变的原来越复杂，维护起来越来越难。大型软件的开发和维护已经成为制约计算机技术发展的瓶颈。伴随着软件设计的这些问题，在软件设计上人们提出了很多新的设计思路来解决这个问题，比如设计模式，模块化软件设计。然而随着软件的愈来愈复杂，这些新的方式已经没法在传统的软件设计方式中得以解决。在传统的设计方法中，一种软件的设计往往由一个公司或者一个组织进行开发维护。再严格的测试流程也无法完全测试出软件之中的存在的缺陷。往往很多缺陷是通过用户来进行反馈。再由开发软件的公司进行修正以及发布。这样的模式在很大程度上是浪费时间的，效率低下的。对于相同功能软件或软件模块，不同的公司由于版权的关系，必须重新进行开发，造成劳动力的重复。

基于传统软件设计的重重缺陷[18]，开源软件技术的兴起为彻底解决了这样的问题提供了可能。所谓的开源软件是指既要发布二进制文件，也要发布软件的源代码。使用软件的人也可以对软件进行修改。并且又使用软件源代码并且重新发布软件的权利。开源软件技术起步于九十年代，由理查德.斯托尔曼自由软件基金会发起。自由软件基金会的最初目的是设计一款自由并且开源的操作系统。但是一些原因，这个计划没有实现，虽然如此。自由软件基金会还是开发出来很多，现在广泛使用的操作系统外围软件。比如 VI，GCC，GDB 等等[19]。自由软件基金会的软件都遵循 GPL 协定，用户在获得软件的同时也或得了相应的软件源代码，但是如果对软件源代码进行了修改商用的话必须将源代码随着软件一同发布，因此 GPL 协定也叫做传染协定。自由软件是开源软件，但是开源软件并不都是自由软件[20]。目前开源软件有如下的几种：

（1）遵循 GPL 协定的自由软件。任何以商用为目的的对自由软件进行修改，都必须发布软件的源代码。无论是应用程序还是以库的形式提供。这是一种最严格的软件许可证

（2）遵循 LGPL 协定的开源软件[21]。遵循 LGPL 的软件可以只提供库软件而不用提供源

代码。这种软件许可证相对于 GPL 协定是比较宽松的。

（3）遵循 BSP 协定的开源软件。这是一种最宽松的软件许可证模式。任何商用的软件都可以不需要提供源代码。但是这样的许可证阻断了开源软件的传播。

随着开源软件基金会的发展，开源软件越来越得到人们的认可。但是对于操作系统开源的过程经历了很多曲折。最早开源的操作系统是 UNIX。在 UNIX 诞生于贝尔实验室，它由一名著名工程师写于 1969 年。该操作系统一降临就迅速的在实验室里风靡起来。1973 年肯汤普森在某个操作系统研讨会上撰写有关操作系统的学术文章。实验室外的工程师学者逐渐才认识了这款优秀的操作系统，之后 UNIX 迅速在全球得以逐渐流行起来。 1975 年美国加州大学伯克利分校的比尔乔伊将某语言编译系统整合在该操作系统里，并且以 BSD 命名进行发布，BSD 成为了 UNIX 发展过程中的一个里程碑。

之后，美国一家著名公司成立了专门的操作系统实验室，修改其操作系统的许可条款，提高授权费用。随着美国电话电报公司商业化 UNIX。UNIX 操作系统变得很难在个人计算机上流行。1980 年，IBM 成功推行个人计算机后，微软公司为其新电脑编写关键的软件，操作系统软件，也就是 windows 视窗操作系统的前身 MS-DOS 操作系统，软件与硬件第一次紧密联系在一起,操作系统 MS-DOS 逐渐成了个人计算机标准操作系统。

之后 UNIX 操作系统就分化成为了两股力量，商业化 UNIX 以及开源 UNIX 也就是 BSD。BSD 分化成许多不兼容的版本。这段期间开源操作系统面临着很大的挑战，于此同时也面临着巨大的机遇。1991 年，来自芬兰的 linus 发布了一款单内核的操作系统 linux，并开放了全部的源代码。随着 Linux 的开放的态度，吸引了全球大量的 IT 人才进行了 Linux 操作系统的开发。同时，开源操作系统正式的迅速发展起来。Linux 以其高效的内核，并且具有裁剪性，良好的移植性等很多优点，被应用于很多领域。

目前 Linux 操作系统正是开源软件技术发展的前沿。以下主要讲述 Linux 操作系统的组成以及特点。

计算机操作系统主要担负起管理计算机的硬件的职责。这主要包含：CPU，内存以及硬盘还有各种外部设备。计算机管理 CPU 主要是为了对进程进行调度。管理内存负责进程所有内存的分配与释放。Linux 作为一款优秀的操作系统，它具有大多数操作系统的几乎所有功能。除了这些还具有自己的一些特性。Linux 操作系统主要包含上层应用软件，Linux 内核和 shell 脚本。

上层应用软件，包含了界面程序以及系统正确运行的守护进程等。在 Linux 系统中最常见的的界面主要包含如下的几种：KDE 以及 GNOME. 这两种图形界面主要用于桌面计算机。界面程序必须依赖于一定的图形界面库才能实现。GTK+库就是 GNOME 采用的图形界面库。这种图形界面库采用 C 语言开发是完全开源的图形界面库，它是真正

免费的工业级图形界面开发工具。KDE 桌面环境基于某家著名计算机公司所开发的 Qt 程序库，全部采用 C++语言。很多 KDE 的桌面程序都是基于 QT 软件构架来进行开发的。随着互联网的发展，HTML5 规范的制定。基于 HTML5 开发的应用程序也逐渐的发展起来。HTML5 使得界面的开发更加的容易并且更容易基于网络进行开发。Shell 程序是很多应用程序运行的基础。应用程序需要在 shell 程序，继承 shell 的环境变量，并且行为受 shell 的控制。应用程序通过 shell 与内核打交道，这样更方便对应用程序进行管理。操作系统的最低层就是内核，Linux 操作系统也是，它主要包含如下的部分。

进程管理。负责操作系统进程的创建于调度， Linux 操作系统内核的调度算法非常的先进，2.6 版本内核采用的 O（1）调度算法，使得进程切换所需时间与进程的总数量是固定关系。3.0 版本内核使用的是公平调度算法，更能发挥操作系统的 CPU 的效能。并且对多核系统的调度更加的高效[22]。

文件系统管理。文件系统是评价衡量操作系统对于文件管理得是否高效的关键。Linux 内核采用虚拟文件系统来统筹文件系统。Linux 操作系统执行广泛的文件系统，对现在所有的文件系统 Linux 内核都能够支持。当出现新的文件系统的时候，采用的虚拟文件系统机制使得移植新的文件系统到 Linux 内核也相对的简单。当前 Linux 常用的文件系统有以下几种，其中包括 ext2,ext3。

设备管理：管理计算机的硬件设备是操作系统最基本的功能。管理计算机硬件模块的优秀办法，如 Linux 设备驱动模型，动态模块加载等。使得 Linux 内核支持设备方便的热插拔，减小了开发 Linux 设备驱动程序的难度。

（1）电源管理与网络。Linux 操作系统内核有着完善的电源管理模块，对设备的即插即用的支持非常的完善。

（2）网络功能：当前的技术要求所有操作系统都应该支持网络接口和协议。Linux 内核对网络有良好的体现，良好的网络的软件架构。在用户程序与数据接口之间相互传输数据，处理路由问题等。

（3）内存管理：在系统算是最复杂的功能，又是基本功能的功能。内核具有高级的内存管理系统。主要包括三个方面，映射进程地址，内存空间分配与释放内存，数据缓冲。

Linux 操作系统的内核总体系统框图如下图所示：

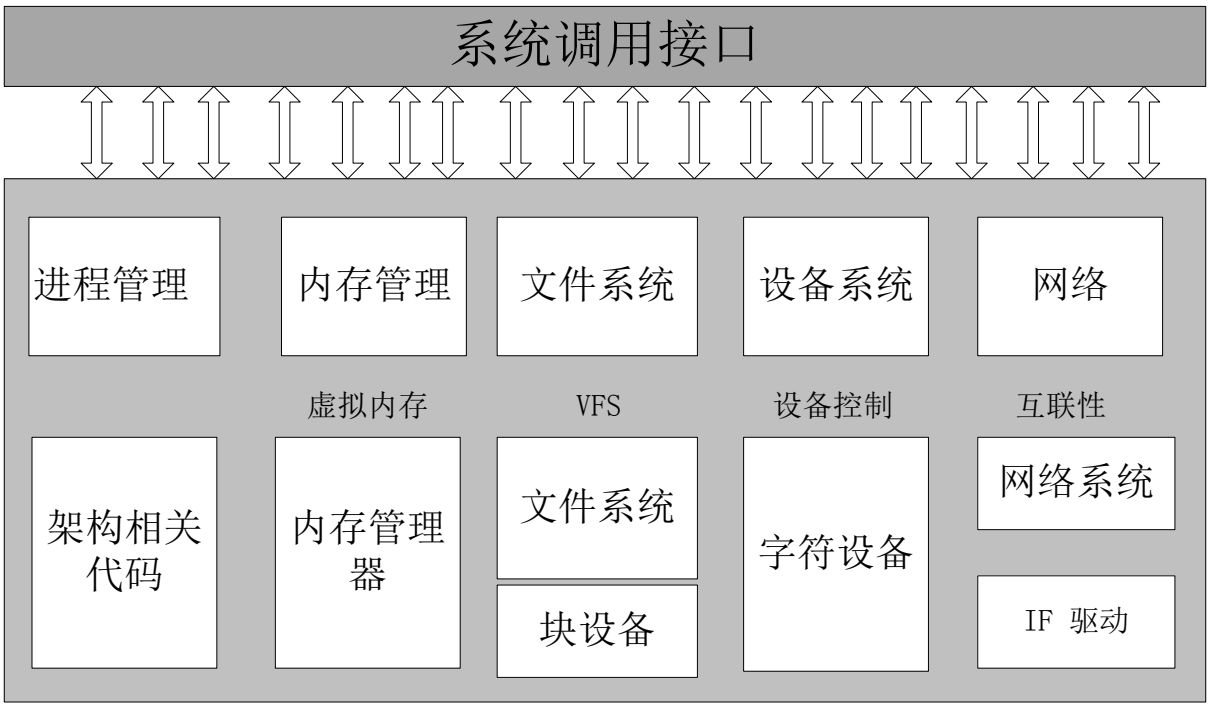

图 4.1　Linux 操作系统的内核总体系统框图
Fig.4.1　Linux Operating system kernel diagram

Linux 操作系统的开发。本小结重点介绍可以用于电子电力系统的嵌入式 Linux 操作系统的组成以及开发过程[23]。一个完整的嵌入式 Linux 操作系统主要分为如下几个部分：

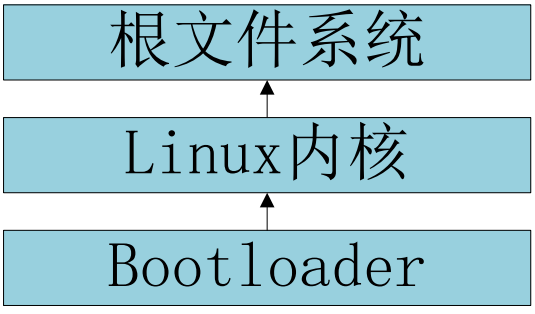

图 4.2　Linux 系统的主要部分
Fig.4.2　main part of Linux operating system

（1）启动加载程序：也就是 bootloader。Linux 通用的 bootloader 有如下几种，Grub 主要用于引导 PC 发行版的 Linux，如 ubuntu，debian 等都使用 Grub 作为 bootloader。U-boot，嵌入式系统通用引导程序，几乎所有的嵌入式 Linux 系统都使用 U-BOOT 作为引导程序。还有其他一些定制的引导程序。对于树莓派，由于其启动的特殊性，他的引

导程序不属于通用的引导程序。下面在 Raspberry 的启动流程中会重点介绍。Bootloader 主要实现是初始化系统必须的硬件，并将操作系统内核从存储介质中加载。

（2）Linux 内核，包含所有 Linux 内核的代码，上一小节已经重点介绍了 Linux 内核的组成以及特点

（3）根文件系统。根文件系统。根文件系统是 Linux 系统启动起来首先挂载的文件系统，他主要是功能是与组织管理文件到存储介质。所有的应用程序以及 C 库以及系统配置都是以文件存储在存储器上的。

嵌入式开发过程中，Linux 系统的分为主机与目标板的概念。进行嵌入式 Linux 系统的开发主要是完成如下的开发:bootloader，操作系统内核的移植，操作系统文件系统的创建。上层应用程序的开发。进行嵌入式 Linux 系统的开发首先需要搭建主机端的软件环境，并且建立主机端与目标机的通信机制。目前常见的通信方式有，串口，USB 以及网络。目前通过网络的通信方式最流行与最普遍。在主机端使用交叉工具链进行编译，并下载到目标机上运行。下面一小节主要介绍了嵌入式 Linux 主机端如何搭建软件环境。

4.2 电脑主机端软件环境的搭建

开发嵌入式 Linux 系统，首先需要在主机上安装一款 Linux 操作系统[24]。当然也可采用虚拟机的方式安装还可以采用单独安装 Linux 操作系统到 PC 上。推荐使用 Linux 与 windows 双系统的。这样更方便开发。主机端 Linux 操作系统一般采用发行版操作系统。目前常见的发行版操作系统有 debian, redhat, ubuntu 等。Ubuntu 操作系统目前是使用最多的桌面版本 Linux 操作系统。

安装完操作系统后，还需要安装交叉编译工具链，主要包括 gcc, gdb, 等。对于嵌入式处理器不同，采用的交叉编译工具链也不同。对于常用的 arm 处理器。可以使用 arm-linux-gcc4.4.3.

由于嵌入式目标机硬件系统的限制，因此在开发的前期需要通过网络文件系统来挂在主机端的文件系统供给嵌入式目标机使用。因此还需要搭建 NFS 服务器，目前最流行的就是 nfs-kernel-server 程序。安置 nfs-kernel-server 程序要遵循如下的步骤：

1. 在软件包管理器上部署 nfs-kernel-server，
2. 打开/etc/exports 文件，添加上：

/home/wang/work/nfsshare/rootfs_qtopia_qt4 *(rw,sync,no_root_squash)nfs，

在另外文件/etc/exports 中进行定义:

/home/wang/work/nfsshare/rootfs_qtopia_qt4：

＊ ： 所有

rw ：可读可写

sync：同步下载

no_root_squash：nfs

3. 再次启动服务服务：

#sudo /etc/init.d/nfs-kernel-server restart

4. 确认：#sudo mount -t nfs localhost:/home/wang/work/nfsshare/rootfs_qtopia_qt4 /mnt 发现在 mnt 中的内容和/home/wang/work/nfsshare/rootfs_qtopia_qt4 的文件内容是一致的话。那么证明安装是成功的。

在嵌入式 Linux 开发过程中，需要频繁的在主机端与目标机之间进行通信，传输程序等。通过 tftp 服务来传输能够使得开发的效率更高效[25]。ubuntu 下可以使用的 tftp 服务器有三种：(1) tftp(客户端)和 tftpd（服务器端） （2）atftp 和 atftpd （3）tftp-hpa 和 tftpd-hpa

其中最后一种 tftp 服务在 ubuntu 上面支持的更好。因此可以在 ubuntu 系统上安装 tftp-hpa 和 tftpd-hpa。下面介绍安装步骤：

（1） 清理之前安装的 tftp 服务与终端程序。

（2） 用 sudo apt-get install 安装客户端和服务器端，

（3） 建立 tftp 目录，然后设置访问权限：

```
sudo mkdir /tftpboot
sudo chmod 777 tftpboot/
```

（4）修改一些配置文件/etc/default/tftpd-hpa

```
sudo gedit /etc/default/tftpd-hpa
```

修改好后，内容如下：

```
# /etc/default/tftpd-hpa
RUN_DAEMON="yes"
OPTIONS="-l -s -c /tftpboot"
TFTP_USERNAME="sun"
TFTP_DIRECTORY="/tftpboot"
TFTP_ADDRESS="0.0.0.0:69"
TFTP_OPTIONS="--secure"
```

这里对参数进行一下说明：

-c

OPTIONS 选项。

TFTP_USERNAME 是用户名，我这里的是″raspi″，用户自定义。

TFTP_DIRECTORY 后面也是同样的根目录 ″/tftpboot″，用户自定义。

再次启动服务

sudo service tftpd-hpa restart

如果在结果中出现：tftpd-hpa start/running, process 3907 说明是安装方法是正确的：

raspi@sun-Crestline-ICH8M:~$ ps aux |grep tftp

root 15173 0.0 0.0 2348 124？ Ss 14:41 0:00 /usr/sbin/in.tftpd --listen --user sun --address 0.0.0.0:69 --secure /tftpboot

sun 14184 0.0 0.0 5628 768 pts/0 S+ 20:32 0:00 grep --color=auto tftp

（9）确认 tftp，先做本机测试

tftp 127.0.0.1

tftp>get u-boot.bin

tftp>quit

127.0.0.1，此地址一般都是本机地址。Get 命令是 tftp 的命令，可以从 tftp 服务器中下载你所需文件。输入？查看相关其他命令。其他常见的命令报考 get 下载命令，put 上传目录等。u-boot.bin 存在在 tftp 根目录下面的文件。

4.3 树莓派开源平台的嵌入式 Linux 操作系统组件的开发

通过上面小节的介绍，嵌入式 Linux 操作系统主要包含如下的几个部分：bootloader，Linux 内核以及根文件系统[26]。本小节将分别对这三个部分的开发进行介绍。

首先是 bootloader 的开发。bootloader 是在操作系统运行之前执行的程序。作用是将某些必须的硬件设备初始化，搭建内存空间的映射表，向内核传递参数，引导内核的启动。Bootloader 的开发与移植是嵌入式 Linux 系统最重要的工作。目前 bootloader 的种类很多，主要有：GRUB，VIVI，redboot, uboot 等。但是由于广泛的平台支持性以及越来越出色的性能和其开放的源码等特点，uboot 应用越来越广泛。下面就以 uboot 为例来介绍 bootloader 的移植于开发。

uboot是以命令行的形式操作的。其命令行模式非常接近linux的shell了。主要的命令分为一下几类：环境变量以及相关命令，串口传输命令，网络命令，nandflash操作命令，内存和寄存器操作命令，Norflash操作命令，USB操作命令，系统引导指令，下载烧写指令。

uboot的启动流程包括两个阶段。第一个阶段：初始化必须的硬件设备，初始化堆栈，为下一阶段c代码执行准备条件，复制自身到内存，跳转到第二阶段的C代码。第二

阶段：初始化本阶段所用的硬件设备，检测系统内存映射，将内核从flash调用到内存中，引导内核启动。

uboot的代码组织结构主要有以下

Cpu:与处理器有关的文件。目录中具有文件，cpu.c 和interrupt.c、start.S、u-boot.lds。cpu.c 负责初始化CPU、设置指令Cache 和数据Cache 等作用。interrupt.c 设置系统的各种中断和异常。文件start.S 是uboot启动时执行的第一个文件，它主要做最早的系统初始化，代码重定向和设置系统堆栈，为进入uboot第二阶段的C程序奠定基础u-boot.lds 链接脚本文件。

Board:目前支持的电路板的相关文件类型，包含内存SDRAM初始化代码、存储介质Flash底层驱动、电路板板级BSP。Makefile文件config. mk定义了TEXT_BASE，也就是代码的内存的起始地址。

Include:头文件，一定包括CPU的寄存器定义，文件系统，网络等, 目录configs的文件,是与目标板相关的头文件。

Lib_arm:重要的函数board. c,几乎所有的第二阶段的代码入口函数和初始化函数都在这个文件中。

Common：与上层一点的通用代码，uboot的命令分析软件代码、命令的应用代码cmd_*. c， Uboot环境变量处理代码env_*. c。

uboot的移植主要是选定一个和自己板级硬件配置相似的uboot自带的设置，根据他进行相应的修改。主要修改交叉编译工具链，board相关代码以及Cpu相关代码。最重要的修改是编译的全局配置文件[26]。

Linux内核的开发。上面的小节介绍了Linux内核的组成。在Linux内核中有很多代码是跨平台的，在不同的硬件平台不需要改动，并且代码经过很多程序员的验证，一般的开发人员是不需要开发这部分的内核代码的。由于Linux内核与硬件最相关的部分就是驱动部分的代码。而不同的系统平台由于硬件的不同。Linux内核需要做相应的移植。在Linux内核中驱动分为很多种，主要包含如下的驱动：

（1）字符驱动程序

（2）块设备驱动程序

（3）网络驱动程序

（4）USB 驱动程序

（5）PCI 驱动程序

这几部分的驱动程序区分并不明显，比如一种设备既可以是网络驱动程序也开始USB驱动程序。在Linux内核对中，驱动程序对不同硬件的支持有着很大的区别。这是因

为不同的硬件由于其结构功能的不同，内核层面上对其的支持非常的不同。因此对于不同的硬件驱动，内核层面的开发有着很大的区别。在Linux2. 6内核中，对于硬件设备的驱动的主要分为以下三种：

第一类是简单的设备，这样的设备Linux内核是不支持的。需要开发人员通过映射物理地址来控制硬件，这一类的驱动程序都比较简单，属于简单的内核模块。

第二种设备更具有一般性需要内核提供最小的支持。内核统一管理这一类的设备。抽象设备的功能，有驱动程序来注册接口完成设备的控制与管理，应用程序只需要调用系统调用就可以控制设备。

第三类设备也具有一般性，但是有一个特点是具有统一的接口。并且设备非常复杂。这样的设备，Linux内核提供完全的支持。这类设备的功能实现完全在内核中实现，对于底层设备寄存器层，驱动只需要填写少量的回调函数就可以了。这样的设备，如USB接口设备，网络设备等等[27]。

Linux操作系统的进程地址空间可以分为用户空间与内核空间。这两种空间的不同主要有两方面。第一方面是运行的CPU权限不同，内核空间运行在CPU的特权模式，有着对很多寄存器的访问权限。而用户空间运行在非特权模式。不能访问一些CPU寄存器。另外一方面是，用户空间与内核空间有着完全不同的线性地址映射。Linux系统不同地址空间的分配如下图所示：

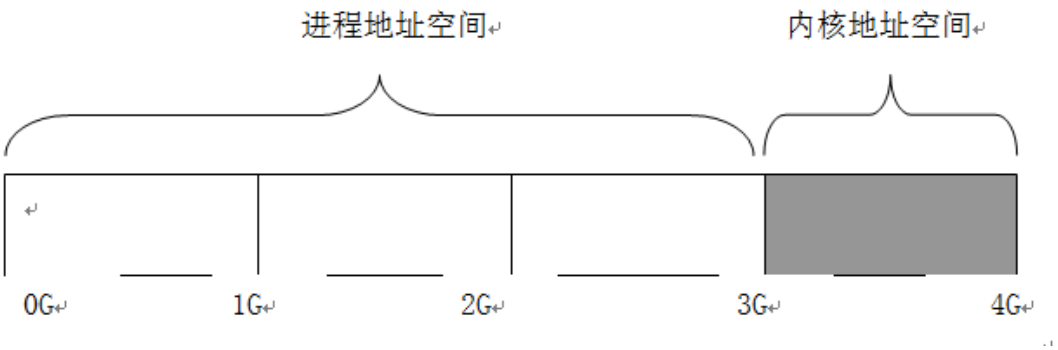

图 4. 3　Linux 地址空间分配

Fig.4.3　Linux address space assignment

根据 Linux 地址映射空间的不同。在 Linux 下开发驱动程序也要区分地址空间。如果在用户空间开发驱动程序那么可以大大的降低对内核的影响。但是效率比较低而且有些资源不能使用。但是在内核空间开发驱动程序，需要特别的注意。因为内核空间的所有代码共享一个地址空间，这样不健壮的代码会导致内核 Crash[28]。

Linux 根文件系统主要包含 Linux 运行必须的应用程序与库。在内核启动后，Linux 内核就会挂载根文件系统。这样应用程序调入内存开始运行。目前针对嵌入式 Linux 的分文见系统有很多主要有 jff2 这个是内核支持的，yaff2 需要打补丁才能在 Linux2.6 内核上面使用。但是新 3.0 以上的内核已经支持这个文件系统。还有 UBI 文件系统。对于嵌入式系统来说不能使用 PC 的硬盘来作为存储。必须使用 Nandfalsh 与 iNAND 等嵌入式存储设备。

对于 Linux 根文件系统的开发，主要是进行根文件系统的打包与格式转换，并且烧入的相应的存储介质中。Busybox 这款软件自带了很多嵌入式交叉编译的必须的程序。使用 busybox 来进行文件的打包能够节省开发人员的很多时间,将打包好的文件系统转换成相应存储介质的格式烧入到存储介质中。修改内核的启动参数，这样就完成了 Linux 根文件系统的开发。

Raspberry Pi 作为一个以教育为目的的卡片计算机。它的软件系统还是非常丰富的。目前树莓派安装的操作系统有如下：主要是 Raspbian，其次是 pidora，然后是 arch。他们都是典型的 Linux 操作系统。Raspbian 从 debian 移植过来的， 这个操作系统也是开源社区推荐的操作系统。而 pidora 移植于 Readhat 的免费版 fedora。除了这些典型的 Linux 系统，Raspberry Pi 还可以跑其他的操作系统。比如一些实时操作系统 uC/OS 等。而本小结主要介绍 Raspberry Pi 上的 Linux 操作系统的软件基础。

前面我们已经介绍了一个完整的 Linux 系统开发所需要的的三个重要组成部分。下面我们可以通过分析 Raspberry Pi 的 SD 卡来理解这三部分。在 windows 上面，插入树莓派可以启动的 SD 我们只能看到一个 boot 的盘，并且容量很小。其他的看不到。这是因为，windows 识别不了 windows 系统意外的文件系统，raspberry Pi 所用的根文件系统是 ext4。因此我们要分析树莓派的操作系统结构，需要一个 Linux 系统[29]。在 Linux 系统上，插入树莓派的 SD 卡，会识别出两个分区，一个是 boot 分区，一个是根文件系统分区。Boot 分区是 FAT32 的文件系统，而根文件系统分区是 ext4 的文件系统。因此在 windows 只能识别到 boot 分区。在 boot 分区内有如下的文件，他们的作用如下表：

表 4.1　树莓派的文件配置表

Table.4.1　Raspberry Pi file config table

编号	文件名	说明
1	bootcode.bin	树莓派 GPU 首先读取的文件，初始化 SRAM
2	cmdline.txt	内核参数配置文件，start.elf 会读取这个文件
3	config.txt	GPU 配置文件，如果没有执行默认配置。GPU 会读取这个文件来配置 GPU 与 ARM 启动方式以及读取一个二进制文件的名称。默认 ARM 读取第一个二进制文件名称是 kernel.img。可以通过这个文件来

		修改。
4	fixup.dat	用来配置 GPU 与 ARM 的内存使用分区
5	fixup_cd.dat	功能与 fixup.dat 类似，只是用于 config.txt 不同配置 *gpu_mem=16* 的情况
6	fixup_x.dat	GPU 的固件
7	issue.txt	
8	kernel.img	树莓派启动流程过后第一个执行的二进制文件，在这里是 Linux 内核
9	kernel_emergency.img	这个内核挂载的根文件系统是 ramdisk，也就是说这个 Linux 内核包含了根文件系统，这能够独立启动的内核。当正常的根文件系统被损坏可以启动这个来恢复系统。
10	start_cd.elf	功能与 stat.elf 类似，只是用于 config.txt 不同配置 *gpu_mem=16* 的情况
11	start.elf	初始化，内核启动参数，并且加载 kernel.img 到内存中执行。并跳转到加载地址开始执行内核代码。在这期间会通过读取 cmdline.txt 来传递给内核参数。当加载到内存后，复位 ARM 核，ARM 核开始工作，执行内核的代码

其中文件 bootcode.bin、start. elf、fixup. dat、config. txt 是构成了树莓派的 bootloader，kernel. img 是树莓派的 linux 内核。另外一个分区的所有内容是树莓派的根文件系统。至此树莓派的 Linux 操作系统的软件结构介绍完了。至于在启动的过程中这几个文件具体的作用在下一小节，Raspberry Pi 的启动流程中详细介绍[30]。

Raspberry Pi 启动流程

一个嵌入式系统的启动流程是由这个嵌入式系统采用的 CPU 来决定的。树莓派采用的 CPU 是 broadcom 公司的一款 SOC，具有 GPU 与 ARM 两个处理器核。由于这款芯片的限制，所以树莓派的启动流程与与一般的系统启动不一样。嵌入式系统的启动流程主要从上电的那一刻其一直到第一个二进制文件执行的过程。树莓派 SOC 由两个 CPU 核组成，一个是 broadcom 的 GPU 核，负责图形图像处理。另外一个是 ARM 公司的 ARM11 的和，控制 SOC 上的其他外设，如网络 SPI 以及 I2C 等。当给树莓派上电的时候，ARM 核是关闭的，首先起来的是 broadcom 的 GPU 核，当他完成自己的初始化后才会给 ARM 核复位，ARM 核才会完成自己的功能。 当系统上电时，树莓派会执行如下的步骤来启动系统：

（1）首先执行 SOC 的 bootrom, 初始化 SD 卡控制器，挂载 SD 卡的第一个分区。使系统能够访问 SD 卡第一个分区。并且在 SD 卡的第一个分区查找 bootcode. bin（与 loader. bin 合并）。并将 bootcode. bin 加载到 SOC 的 L2 cache 执行。此时的 SDRAM（板子上的 512M 内存）是不可用的。

（2）bootcode. bin 初始化内存控制器，使 SDRAM 可用，并读取 fixup. dat 给 GPU 与 ARM 划分内存分区。GPU 驱动 config. txt 配置自己，如果没有执行默认配置。然后在 SD 卡中的地一个分区需找 start.elf（如果 config.txtgpu_mem=16 的设置会查找 start_cd.elf），加载到内存中执行！这个文件是 GPU 的 firmware，初始化 GPU。

（3） start.elf 还有一个重要的任务就是初始化，内核启动参数，并且加载 kernel.img 到内存中执行。并跳转到加载地址开始执行内核代码。在这期间会通过读取 cmdline. txt 来传递给内核参数。当加载到内存后，复位 ARM 核，ARM 核开始工作，执行内核的代码

（4）至此，启动过程结束。系统的控制交给内核或者其他的可执行程序来完成[31]。

注意：bootcode.bin start.elf 是 broadcom 的闭源代码，由于涉及到 GPU 的初始化，broadcom 是不提供源码的，但是有国外爱好者已经返汇编出来了。代码量相当庞大，而且必须有 broadcom 提供的编译环境才可以编译，所以意义不是很大。因为 broadcom 芯片的限制，所以树莓派的启动流程与传统 Linux 不同。因此 Linux 系统上的常用引导工具 grub（常用于 PC） U-BOOT（嵌入式系统中）没有被使用。有人尝试着使用 U-BOOT 来引导内核，但是得必须将 U-BOOT 的二进制文件该名字成为 kernel.img，由 GPU 加载到内存，然后再由 U-BOOT 引导内核，其实没有这么做的必要。由于 start.elf 已经具备引导 Linux 内核的能力，所以不需要使用 U-BOOT 来引导了

树莓派的使用

当你拿到一个 Raspberry Pi 以及一张空的 SD 卡（4G 以及 4G 以上），由于 SD 卡没有系统软件，当你插上 SD 卡并接上电源，接上显示器。显示器是没有任何响应的。所以你需要自己制作可以启动的 SD 卡。本小结会介绍最简单的制作系统启动盘的过程，首先让 Raspberry 跑起来，至于定制系统以及软件的组成以及原理下面的章节会介绍。

如果你的机器只装了 windows 操作系统，那么你需要两个软件：

(1) SDFormatter 4.0 ：格式化 SD 工具软件，由于操作系统运行在 SD 上，所以需要 SD 卡充分格式化，以加快读写速度。

(2) win32diskimager-v0.9-binary： 这个是镜像写入工具，将下载的系统镜像写入到 SD 内[32]。

另外一个就是在 Raspberry 官网上下载的操作系统镜像：

20xx-xx-xx-wheezy-raspbian.img

首先使用第一个软件将 SD 卡格式化，然后使用第二个软件将操作系统镜像写入到 SD 内。SD 卡就可以用了，将 SD 卡插入树莓派。连接好显示器，接上电源你会看到系统跑了起来。第一个启动会有 raspi-config。需要设置一些配置。如果不小心设置错误，你也可以通过在命令行下输入 sudo raspi-config 来再次设置。

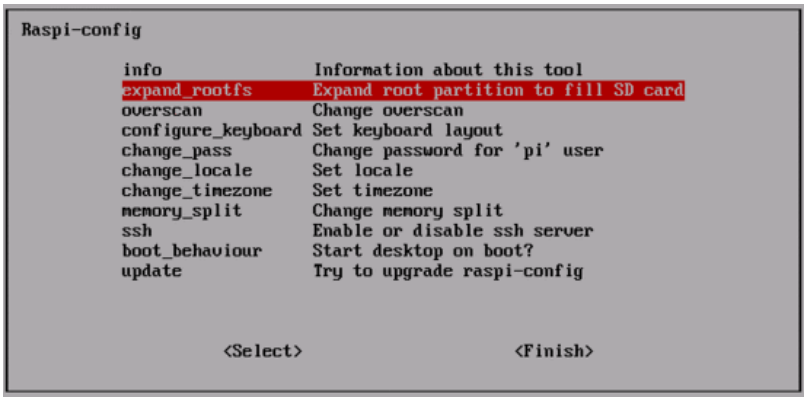

图 4.4　树莓派的编译配置选项

Fig.4.4　The Selection configuration of Raspberry Pi

首先，我们要选择 expand_rootfs。它的作用是扩展 Linux 系统的根文件系统。刚才我们写入的镜像的根文件系统大约 2G，一般我们会用大于 4G 的 SD 卡。用以上的做法，导致剩余的空间没有被利用，而且系统不能再装入其他的软件了，由于空间不足。设置 expand_rootfs 就充分利用了 SD 卡的空间。当然你也可以不选，进入系统自己来分区。这就需要具备一些 Linux 系统的使用经验。对于初次接触的人来说，还是上系统自动完成的好[33]。

选中 expand_rootfs 按下回车，会弹出一下的界面，显示已经成功扩展根文件系统，按下回车返回主界面。

然后要设置 overscan 选项。如果现在你的屏幕显示的图像并没有完全占用你的显示器空间，那么将 overscan 禁用掉，来让系统充分利用整个屏幕。但如果你的屏幕显示是 OK，就可以不用设置这步了。选中 overscan，按回车

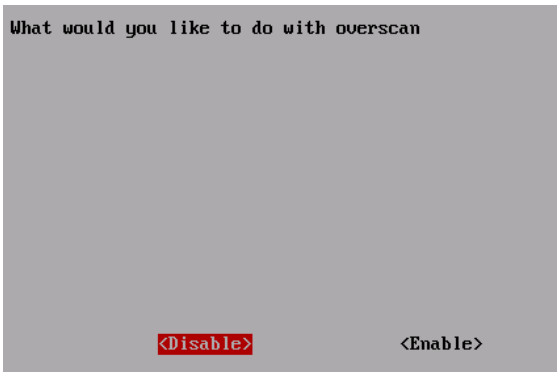

图4.5　overscan 配置方式

Fig.4.5　overscan configuration way

Raspbian 对键盘的识别也是不同，分为美式和英式。选中 configure_keyboard，然后按下回车。选择 Generic 105-key (Intl) PC 键盘，如下：

图4.6　键盘选择

Fig.4.6　Key selection

在选择键盘类型以后一定需要选择键盘布局。前面的都是英国键盘布局，选中其他（Other），然后再里面的列表选择 English（US）。

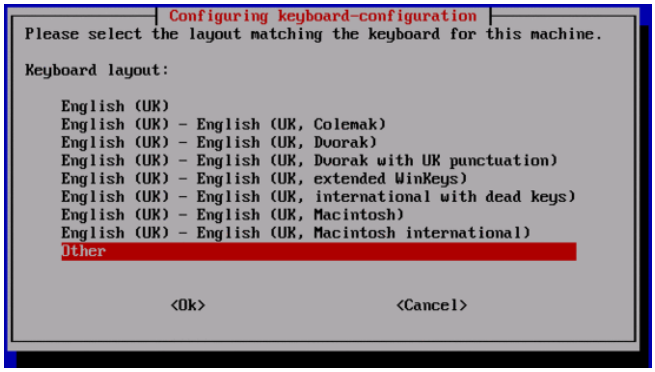

图4.7　键盘选项列表

Fig.4.7　Key selection list

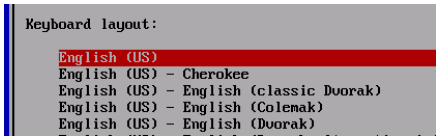

图4.8　选择语言

Fig.4.8　Language Selection

最后我们要设置用户名与密码，根据需要密码和用户名都是可以修改。

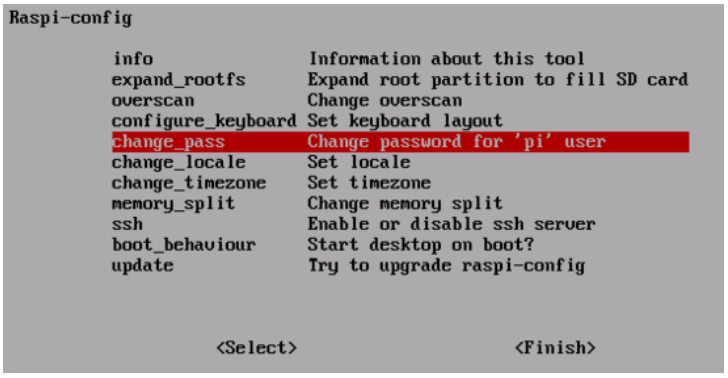

图4.9　修改密码

Fig.4.9　Change password for pi user

然后可以设置字符，这里我们选择 en_US.UTF-8，其他一些设置是无关紧要的设置，将树莓派重启以后，你会看到一个登录界面，在这里，用户名为”pi”，密码是刚才设置的密码，登录后第一件事情是启动图形界面。在命令行中输入：startx. 如果没有错误的话，会弹出树莓派的界面，那么恭喜你，Raspberry Pi 正常运行起来的[34]。

树莓派系统软件的编译与定制。Linux 操作系统非常重要的一个特点就是可裁剪，对于树莓派系统来说也可以通过裁剪来去除不需要的模块来提高系统运行的速度。或者增加模块来扩展系统的功能。本小结就介绍了树莓派系统软件的开发过程。树莓派系统软件的开发主要包含：Linux 内核的编译，根文件系统的编译与构造。定制操作系统镜像。

Linux 内核是嵌入式 Linux 系统不可或缺的组成元素，对应于树莓派就是 SD 卡第一个分区的 kernel.img 这个文件。他是由内核源码编译成的二进制文件，由树莓派的 bootloader 加载到内存中执行。主线的 Linux 内核源码编译的二进制是不能够使用的，因为他缺少了树莓派响应的驱动。所以需要移植 Linux 内核，移植 Linux 内核还是需要很多时间与精力的，还需要嵌入式系统底层软件的经验。不过庆幸的是已经有人完成了这部分的工作。并且在 github 上建立了 Linux 内核源码分支。编译内核所用的交叉编译工具也在上面提供。因此编译树莓派的 Linux 内核变的非常容易。只需要如下几步就可以完成。按部就班的做谁都会做，关键是理解每一个步骤含义，这样才能对系统有着更深入的理解。进而能够自己移植内核。

（1）获得交叉编译工具链。由于树莓派的 CPU 是 ARM11，而我们需要在主机上构建编译系统，那么需要安装交叉编译工具链，这里的交叉编译工具链是 arm-linux-gcc，前辈们已经配置与测试好了，所以在 github 上拷贝下来用就可以了。当然如果你对交叉编译工具链的制作有兴趣，也可以自己制作这个工具链。

（2）获得内核源代码

（3）清楚编译配置

（4）拷贝配置文件

（5）编译内核

（6）编译模块

（7）替换原来的内核

（8）实验是否替换成功

根文件系统的编译与构造。Linux 系统的根文件系统有连种：一种是内存文件系统，这种文件系统是只读的，存在于内存之中，可以与内核编译成一个文件就是 kernel.img，这种内核不需要其他的根文件系统就能独立启动起来，这种文件系统在内核自解压的时候创建。另外一种的存储介质文件系统，对于树莓派就是存放在 sd 卡中第二个分区的 ext4 文件系统。这种文件系统是可读可写的文件系统。但是需要存储介质的支持。下面就分别介绍树莓派这两种文件系统的制作。

（9）第一种内存文件系统，也叫 ramdisk。在编译内核的时候我们可以通过修改内核编译配置来将其编译到内核。首先获得内存文件系统目录

Git clone https://github.com/raspberrypi/target_fs

在内核配置文件中. config 加入如下的配置

CONFIG_BLK_DEV_INITRD=y CONFIG_INITRAMFS_SOURCE="../target_fs"

其中 target_fs 是从 github 上下载下来的文件目录。这样在编译内核的时候就将其编译进去了。这样内核启动的过程会先挂在 ramfs 到根目录。然后再挂载其他的文件系统。

4.4 树莓派开源平台的嵌入式 Linux 操作系统镜像的定制

我们可以从官方网站下载最新镜像，通过烧入 SD 卡来启动系统。这一小节将详细介绍通过我们自己的方式定义自己的操作系统镜像。在自己定制镜像之前，我们先分析一下官方下载的 img 镜像。官方提供的 img 文件其实是 SD 镜像文件，在 Linux 下用 fdisk 进行分析。输入如下的命令：

fdisk -l 2013-09-10-wheezy-raspbian.img

可以发现输出如下的信息：

Disk 2013-09-10-wheezy-raspbian.img: 1939 MB, 1939865600 bytes

255 heads, 63 sectors/track, 235 cylinders, total 3788800 sectors

Units = sectors of 1 * 512 = 512 bytes

Sector size (logical/physical): 512 bytes / 512 bytes

I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk identifier: 0x000b03b7

Device Boot Start End Blocks Id System

2013-09-10-wheezy-raspbian.img1 8192 122879 57344 c W95 FAT32 (LBA)
2013-09-10-wheezy-raspbian.img2 122880 3788799 1832960 83 Linux

可以发现这个 img 包含两个分区，一个分区是 FAT32 格式的启动分区， 另外一个分区是根文件系统。上面我们已经说到完整的 Linux 操作系统包含三个部分：bootloader，kernel，根文件系统。Raspbian 是基于 debain 的 Linux，所以也不例外的包含这三部分。那么这三个部分是怎么包含在 img 文件中的，我们通过 Linux 上的 image 分析工具来分析一下：

（1）首先要将 img 文件映射到虚拟的设备上，Linux 下是通过 losetup 命令来实现的，这个命令将 Img 文件映射到循环设备上。输入如下的命令。

losetup -f --show 2013-09-10-wheezy-raspbian.img。显示/devloop0，此后访问 /dev/loop0 设备就是访问 img 文件了。此时可以 fdisk -l /dev/loop0 显示的内容与访问 img 文件是一样。

（2） 现在已经有了块设备，但是还不能访问。因为他的两个分区还没有显示出来。此时我们可以通过使用 kpartx 命令来映射他的两个分区，这个命令在一般的 Linux 发行版上不是默认安装的，所以要自己安装 apt-get install kpartx 就可以完成安装[35]。输入：kpartx -av /dev/loop0。

显示如下的信息：

add map loop0p1 (254:0): 0 114688 linear /dev/loop0 8192
add map loop0p2 (254:1): 0 3665920 linear /dev/loop0 122880

然后在/dev/mapper 文件夹下面会出现两个文件 loop0p1 loop0p2 这两个文件就是 img 的两个分区。我们可以通过挂载他们来查看他们中的内容。

（3）首先在/media 目录下面建立两个文件夹

mkdir /media/boot
mkdir /media/rootfs
mount -t vfat /dev/mapper/loop0p1 /media/boot 来挂载第一个分区
mount -t ext4 /dev/mapper/loop0p2 /media/rootfs 来挂载第二个分区

（4） 查看 boot 文件夹的内容 ls /media/boot 发现里面有如下的文件。其中 bootcode.bin 以及 start.elf 是 bootloader，而 kernel. img 是 linux 内核

bootcode.bin config.txt fixup.dat issue.txt kernel.img start.elf
cmdline.txt fixup_cd.dat fixup_x.dat kernel_emergency.img start_cd.elf start_x.elf

这些内容就是我们前小节分析的 SD 卡启动分区之中的内容

（5） 查看 rootfs 文件夹的内容 ls /media/rootfs[36]。发现里面是很多文件以及文件夹。这个分区就是 Linux 的根文件系统。所有的应用程序以及系统的库都在这个分区内。

分析清了 img 文件的构成，那么我们可以定制自己的 Linux 操作系统镜像。定制自己的 Linux 操作系统镜像是这个过程的反过程。

（1）首先使用 dd 命令新建一个空白的 img 镜像

dd if=/dev/zero of=raspberry_myimage.img bs=2M count=1024

（2）然后使用 fdisk 将其格式化，并且分为两个区

（3）挂载到 loop 设备上

（4）将所需文件分别拷贝到相应的分区内。

Img 镜像就做好了。

4.5 树莓派开源平台的嵌入式 Linux 脚本应用程序的开发

树莓派（Raspberry Pi）推荐的官方编程语言是 phyton，Python（KK 英语发音：/ˈpaɪθən/），是一种面向对象，直译式语言，1989 年底发明，第一个公开发行版发行于 1991 年。Python 语法简捷而清晰，具有丰富和强大的类库。语言能够很轻松的把用其他语言制作的软件组件轻松联结一起。在树莓派上编写 phyton 语言非常简单，树莓派已经自带了 phyton 编译环境[37]。

4.6 阿都诺开源平台的嵌入式实时操作系统与调度器设计

原始的阿都诺平台是没有操作系统的，采用一个简单的 Processing 语言形式的程序设计模式。本文采用开源的嵌入式操作系统，freeRTOS，或者自行设计有限状态机作为阿都诺开源硬件的软件架构。freeRTOS 是一款容易使用的，高质量的，跨平台的实时操作系统，已经应用于 34 架构上，18 种工具链中，上百万的产品部署。该操作系统已经包括了任务管理，队列管理，内存管理，时间管理，消息队列， 具有 ucos-II 这类商用嵌入式实时操作系统类似的基本满足最小系统的要求。FreeRTOS 操作系统是完全免费的操作系统，源码完全开发，移植性强，裁剪性高，调度策略灵活[38]。

调度器设计，在 RAM 小于 8k 的时候，移植轻量级实时操作系统也是困难的时候，可以考虑采用调度器设计。调度器设计，可以在基于最简单的前后台，实现一个一般复杂度的有限状态机切换，为了完成一些消息传输，可以采用队列管理和内存管理的一些思想[39]。从这个角度来看，这就是一个操作系统的原型。

如果要用好 freeRTOS，可以与 ucos 比较学习。良好的数据结构知识和操作系统思维方式是最关键的。

4.7 阿都诺开源平台的微控制器软件接口标准

微控制器软件接口标准最早是由 ARM 公司提出，Microcontroller Software Interface Standard （CMSIS）微控制器软件接口标准是 Cortex-M 系列处理器，HAL 硬件抽象层。这是 ARM 公司统一微控制器软件接口提出的一个标准。本设计的初衷是为了减少更换芯片和开发工具，带来的时间和金钱损失。同样成熟的软件框架，在嵌入式领域，要实现很好的移植不是那么轻松的一件事情。太多的时间证明，更换芯片可能会带来风险。所以在一些关键行业，都会与芯片厂商签署一份保证供货协议。针对 ARM 系列芯片，不同的厂家会针对不同 ARM 内核出不同的外设的芯片，为了尽量减少软件与芯片硬件的耦合度[40]。

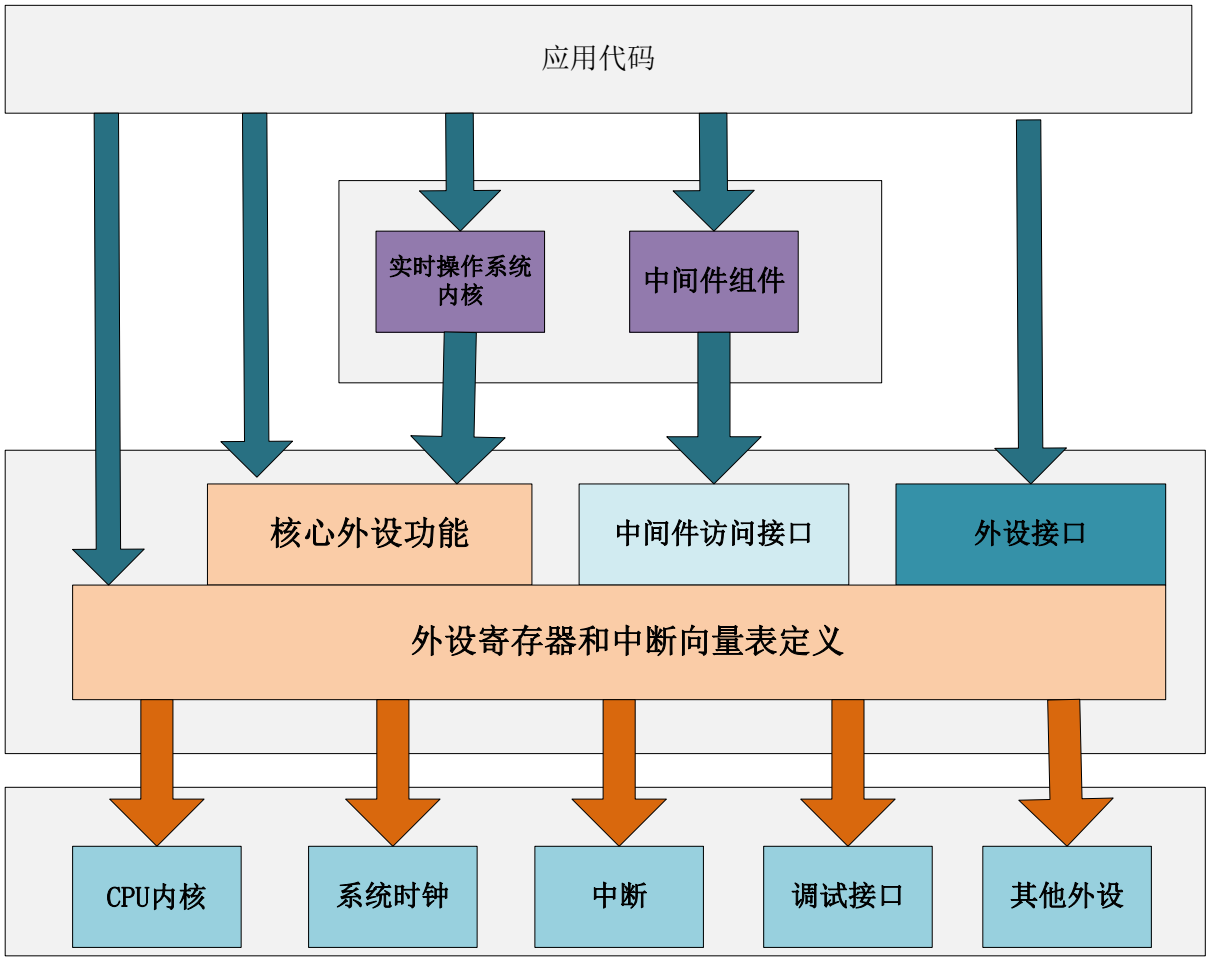

图 4.10 CMSIS 软件接口标准在整个嵌入式实时软件系统中的位置
Fig.4.10 CMSIS location in the embedded real-time software system

MCU 层为硬件驱动层，CMSIS 接口标准，RTOS 为操作系统层，USER 也就是真正的 APP。CMSIS 的架构，CMSIS 可以分为以下 3 个基本功能层，核内外设访问层，中间件访问层，设备访问层。

各层具体情况如下，核内外设访问层，定义了一些 Cortex 处理器内部的寄存器地址和功能函数，包含内核寄存器，非嵌套式中断，调试系统的访问。中间件访问层，讲述的是，定义一下中间件的通用 API，芯片厂商根据需要进行更新。设备访问层，与核内外设访问层也比较类似，也是主要包括芯片关键寄存器的外设的访问。这部分还包括了一些扩展外设中断的访问。由芯片厂商提供这部分代码[41]。

4.8 本章小结

本文首先详细讲述树莓派开源平台的嵌入式 Linux 开发环境的搭建，开发方法等，再仔细分析阿都诺开源平台的嵌入式实时操作系统与调度器设计，微控制器软件接口标准。

5 开源电力电子嵌入式计算平台的设计

伴随着计算机技术的发展，信息的互联性以及共享性变得越来越重要！电子电力平台也不例外[46]。为了使得电子电力平台所获取的信息能够更直观的展示以及达到更好的传输性，能够连接互联网并入数据库！通过智能操作系统对数据进行存储以及更深层的处理。这样电子电力平台的可扩展性以及互联性就大大的提高。为了使得电子电力系统拥有这样的能力。在 MCU 控制的传感器的另外一端需要智能操作系统平台的支持。借助于开源软硬件就可以使得这种智能操作系统快速的实现，即节约了成本又简化了设计。本章就以树莓派和阿都诺为例介绍基于开源软硬件的电子电力平台智能化以及互联化的实现[47]。

5.1 概论

在前面四章的基础上，本文分别讲述了电力电子嵌入式计算平台的基本概况后，迅速切换到嵌入式硬件设计和嵌入式软件设计的两个方面来展开。其中第二章重点讲述感应电机无速度传感器电路的模型分析。第三章分别讲述阿都诺和树莓派的硬件结构[48]。第四章重点讲述开源软件领域的 Linux 平台的文件系统，内核，应用 app 等，开源实时嵌入式操作系统的 freeRTOS，ARM 公司的微控制器软件接口标准，控制器驱动接口设计规则，bootloader 设计。本章站在电力电子的应用领域特征，作出了具体的框图设计。

5.2 总体思路

嵌入式系统一般都包括传感器电路部分，信号调理电路部分，交流转直流，直流转直流模块，微控制器模块部分，应用处理器部分。本文基于开源软硬件技术的电力电子嵌入计算平台系统（以后简称开源电力电子计算平台）框图，如图 5.1 所示，包括树莓派(Raspberry Pi)和与阿都诺兼容的 STM32F407 电路，电流传感器，电压传感器，交流转直流电路模块，电力电子控制电路接口，USB 转 TTL232 电路[49]。

在开源电力电子计算平台中，树莓派作为主机，在整个系统起到数据中转的作用。树莓派可以通过外接 HDMI 接口的显示器或者液晶显示屏，可以通过外接网线或者具有 USB 接口的 wifi 模块。

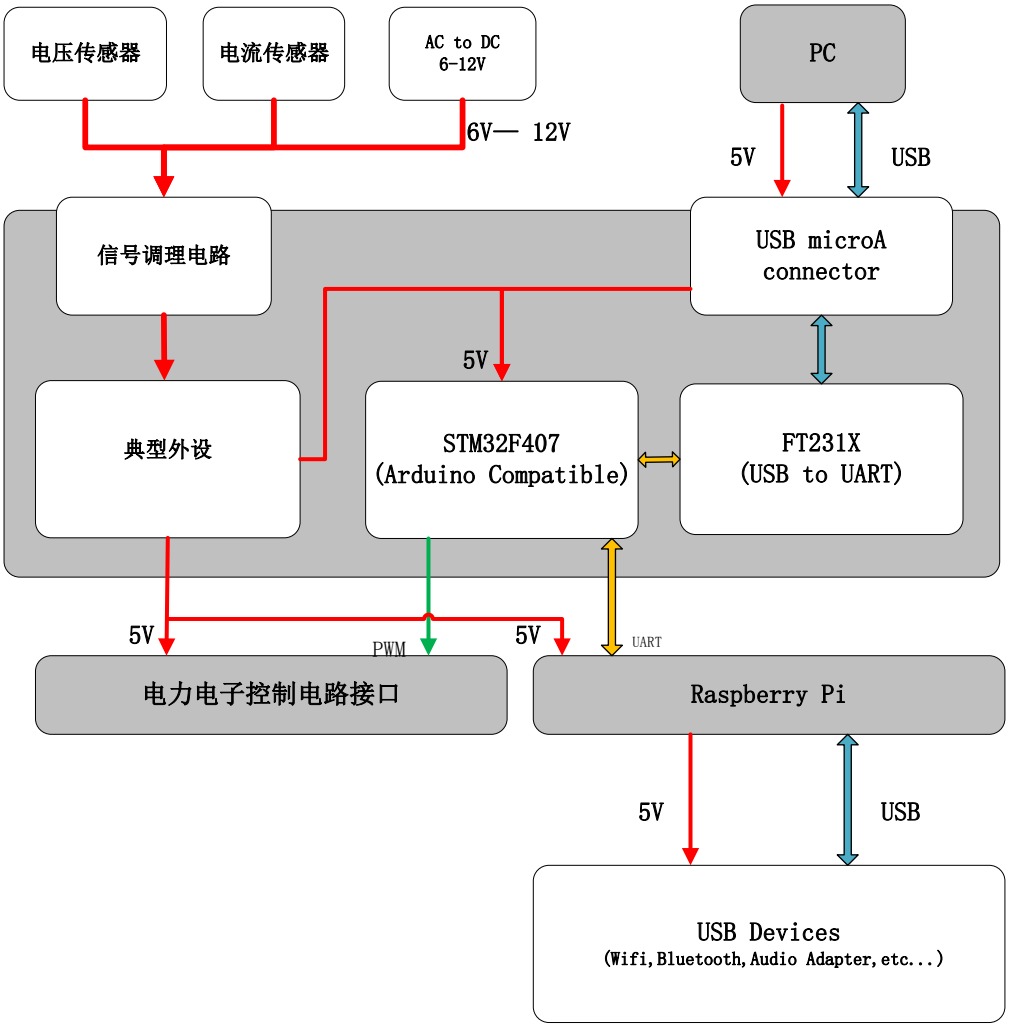

图 5.1 总体思路
Fig 5.1 Total Diagram

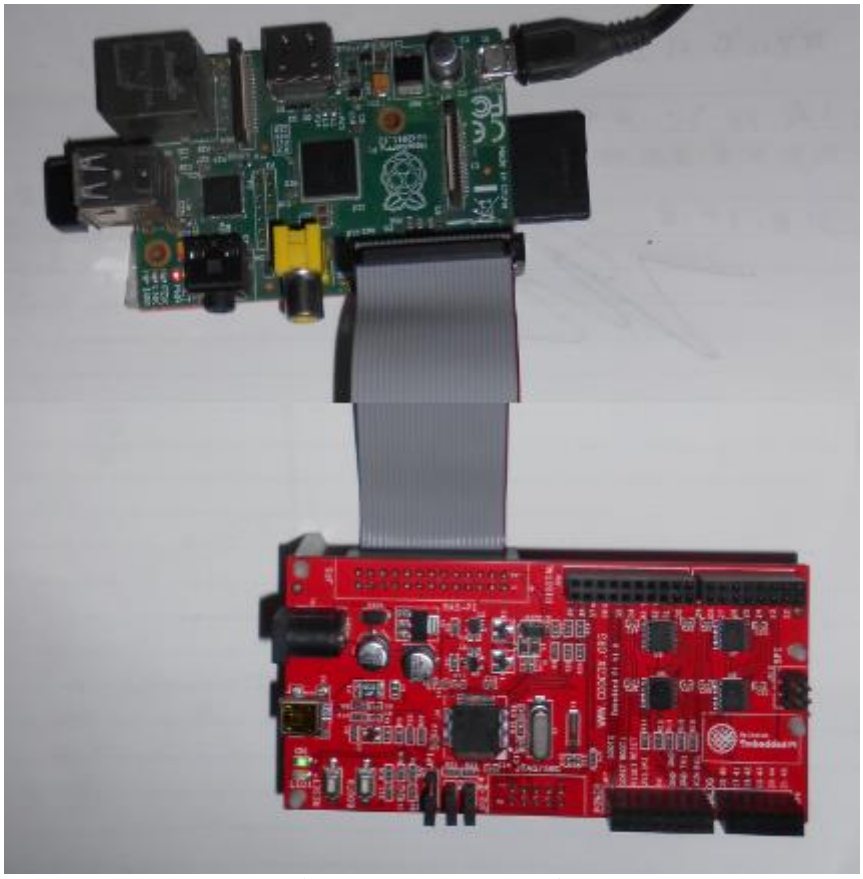

图 5.2　树莓派和阿都诺组合的实物图
Fig5.2　Raspberry Pi and Arduino Combination

图 5.2 中，上边绿色电路部分为树莓派，红色电路板为与阿都诺兼容的 STM32F407 原型板。树莓派与阿都诺电路板则是通过 20 管脚的扁平电路连接。

5.3　电流电压电路设计

电流电压采集，也称为模拟信号输入调理电路，电力电子和电力传动计算平台的相电压和相电流分别经过电流互感器和电压互感器变换之后，输出为标准的正负 10V 的模拟电压信号。电流电压采样电路可以有以下两种方案可供选择：

（1）采用 ADE7758 或者 ATT7022 或者 SD3003 作为数据采集芯片

ADI 公司的 ADE7758 是一款精密的具有串行数据接口输出，脉冲数据输出的电能测量芯片。芯片内部包括两个 ADC、一个数字积分器、参考电路、温度传感器、以及计算有功、无功和有效值计算[11]。SD3003 是 ATT7022 是一款电能专用计量芯片；它适用于三相三线，三相四线制的应用场景[12]。杭州晶华微电子推出的 SOC 计量芯片，已经实现国产化量产。

采用专用计量芯片为标准的 Arduino shield 扩展板，使用起来比较方便的，单片机与嵌入式系统只需通过 I2C 或者 SPI 或者串口的方式，以通信消息的方式访问芯片的寄存器，电力参数和工况信息的采集是由芯片本身的算法逻辑和信号调理电路实现。我们要特别注意的是一定要做好强电与弱电的隔离。强弱电隔离一般都是两种形式：变压器磁隔离和光电隔离。

（2）采用 DSP 或者 MCU 实时计算电力参数，需要自己设计一套算法。

电流电压采样电路虽然可以采用以上两种方案，但是为了开放和重复使用硬件，所以我们采用 Arduino 的硬件接口方案，其中电流电压采用 GROVE 四针接口电路板[14]。

具体的电路设计如下图 5.3 所示

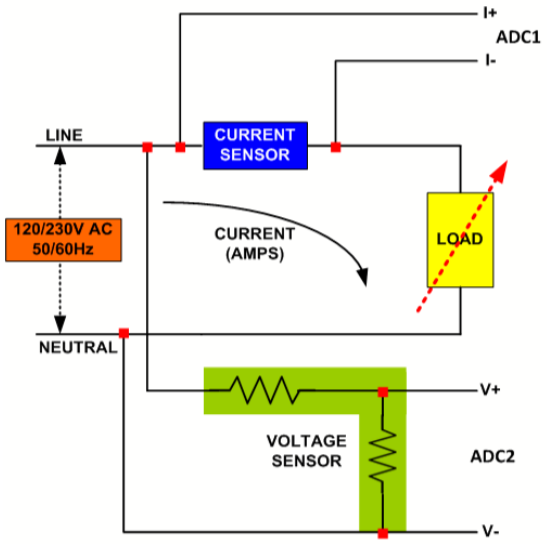

图 5.3　单相的电流传感器和电压互感器电路原理框图

Fig.5.3　The schematic diagram of Single-phase current transformer and voltage transformer

从主回路过来的电压一般都是 230V 或者 120V，但可以分压到 930mv。电压的模拟前端包括尖峰保护压敏电阻 R3，在压敏电阻之后是一个简单电压分压器和 RC 低通滤波器反锯齿滤波器电路。

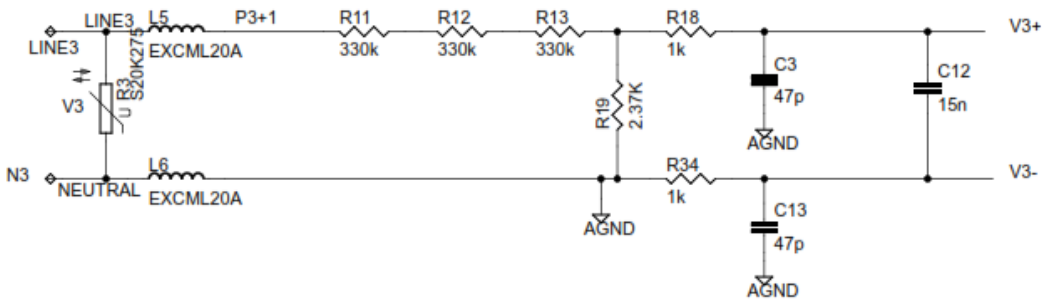

图 5.4　电压输入的模拟前端电路设计

Fig.5.4　Analog circuit design of voltage input

上图显示了来自主回路为 230v 电路的模拟信号前端，电压被分压为 549mv，峰值为 779mv。这个电压在微控制器的模拟限制范围内[15]，安全空缺有 15%的范围。这个空缺允许在电压脉冲状态下进行精确测量。

电流输入，下图为电路输入的模拟前端电路设计；它稍微不同于电压输入的模拟前端设计。

负载电阻的选择应考虑 CT 电流互感器的电流范围和转化比。在这个设计中，负载电阻大约是 13 欧，反锯齿电路包括电阻和电容，以及负载电阻[16]。转换器的输入信号完全是差分输入的电压，其范围是±930mv 之间。在转换比设置为 1 的情况下。

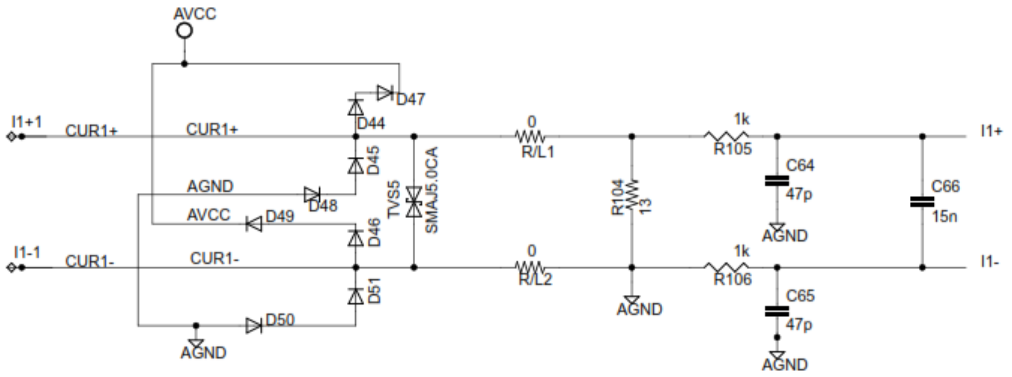

图 5.5　电流输入的模拟前端电路设计

Fig.5.5　Analog circuit design of current input

本章将描述三相电的测量软件实现。第一个小节描述了 MSP430 的各种外设建立。紧接着，完整的测量软件实现被描述为两个主要流程：前台和后台流程。外设包括 $\sum\Delta$ 模拟数字转换器，时钟系统，定时器，LCD 和看门狗时钟。对于三相电压测量系统来说，最少需要 6 个 $\sum\Delta$ 来测量三相的电压和电流，而很多微控制器芯片的外设已有多路独立

的$\sum\Delta$模拟数字转换器。代码设计了具有有限的反篡改特性，但是代码支持中性电流的测量。$\sum\Delta$的时钟来源于系统时钟，被配置为 16Mhz。它采用的频率是 $f_s = f_M / OSR$，OSR 为 256，模块频率为 1.048576MHZ；结果是 4.96ksps。配置为固定采样频率每个采样间隔中。实时时钟是实时时钟模块可以配置为精度为 1 秒中断。基于 1 秒中断，这个时间和日期可以按照需要通过软件升级。

MSP430 的 LCD 控制器可支持多达 8-mux 显示 320 段。由于内部充电泵可以达到良好的对比度，在电流设计中，LCD 控制器可以配置为 4-mux 模式，所以，可以使用 160 段，ACLK/64 的刷新频，也就是 512hz.对于信息来说，这些参数可以支持 LCD 显示。

前台程序

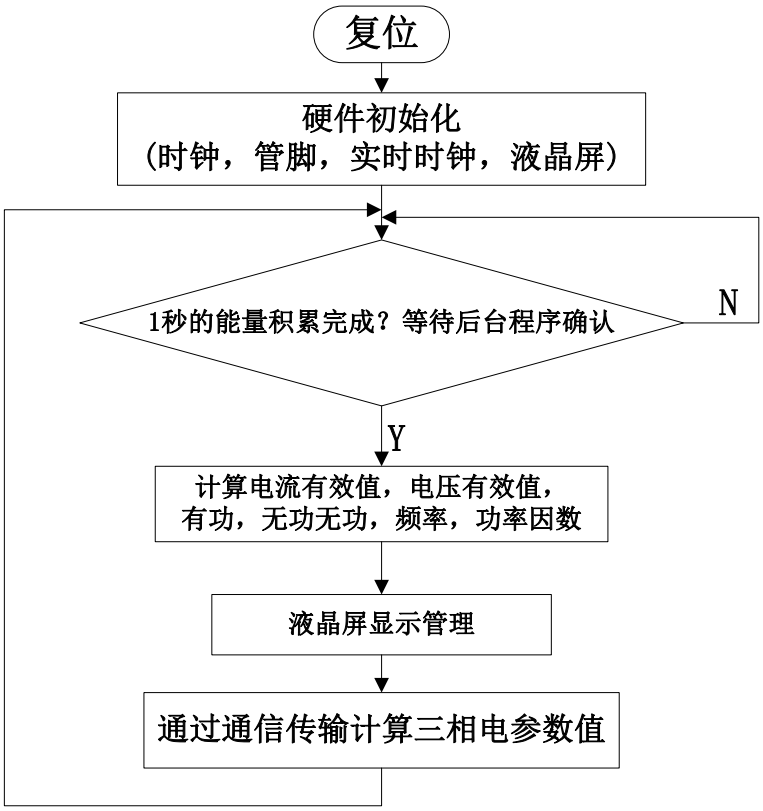

图 5.6　前台程序流程图
Fig.5.6　program diagram

前台程序包括复位之后的 MSP430 硬件初始化和软件初始化。 硬件初始化包括时钟 SD24(Sigma delta)、端口配置、实时时钟、LCD 和外设。第二步，1 秒的能量积累，等

待后台程序的确认。第三步，计算平均电流和电压，有功，视在功率，无功功率，频率，功率因数等电参数。第四步，LCD 显示管理。第五步，发送数据到通信通道。

在硬件初始化完成之后，前台程序就等待后台程序通知他最新计算的仪表参数。这个参数通知是通过当一帧数据处理完成之后的状态标志来实现。数据帧包括处理后的电流，电压，有功，无功。这个等同于与电压信号同步的 50/60 个周期的数据积累。除此之外，采用计算器。

计算方法的进一步研究。

后台程序：

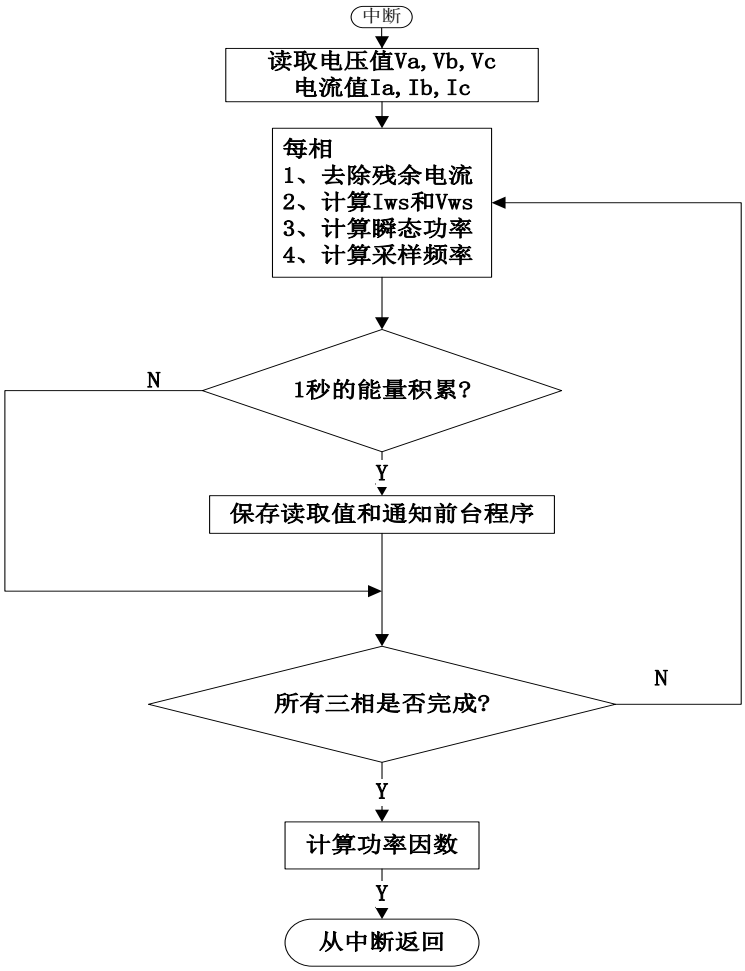

图 5.7　后台程序流程

Fig.5.7　Program diagram

数据结构：

```
struct phase_parms_s
{
    int32_t active_power;
    uint16_t V_rms;
    uint16_t I_rms;
    int32_t V_dc_estimate[2];
    int16_t V_history[4];
    int16_t V_sq_accum[3];
    int16_t V_sq_accum_logged[3];
    struct current_senor_parms_s current;
    struct current_senor_parms_s neutral;
    int16_t sample_count_logged;
    int32_t active_power_counter;
    int32_t energy_counter;
    uint32_t consumed_energy;
    int since_last;
    int16_t last_V_sample;
    uint16_t status;
    int8_t V_endstops;
    int8_t V_history_index;
    uint8_t energy_pulse_remaining_time;
}
```

5.4 电力电子控制电路设计

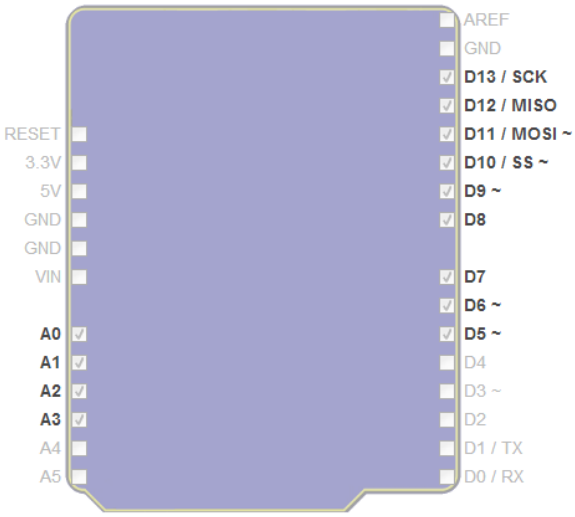

图 5.8　arduino 的外部接口图
Fig.5.8　arduino external interface

Arduino 的外部电路接口包括电源输出 5V 和 3.3V，复位输入，电源输入；模拟输入 A0-A5；数字输入 D0-13；通信接口 UART、SPI、I2C。层叠，组件，标准化，高度抽象化。软件提升了硬件价值，整个工具链开发。

机械机构按照大小尺寸:宽 70mmX 高 54mm，插针排列方式如图 3.11 所示。主控制器，各类传感器，各类执行机构，显示接口，通信接口，都采用这种方式来设计他们的对外接口。

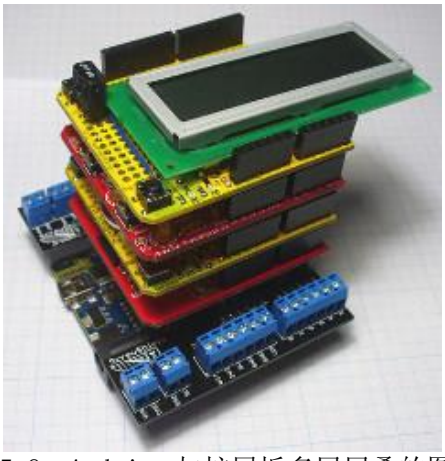

图 5.9　Arduino 与扩展板多层层叠的图片
Fig.5.9　Arduino and Extension board

如此外部接口与一些具体行业的术语结合，电力行业有四遥，遥控，遥信，遥调，遥测，对应 arduino 的外部接口，分别为数字输出，数字输入，模拟输入，模拟输出；

这些基本符合我们应用场合的硬件要求。可以选取几个典型的电力电子电路交流转换直流，直流转换直流,直流转换交流，交流转换交流其中一部分作为 arduino 的开源盾板。

5.5 硬件低功耗设计

开源硬件的低功耗设计不仅在电池供电这类电量敏感的领域广泛应用，也作为现代低碳节能环保的重要方向被深度挖掘。

5.6 驱动软件设计

驱动设计，按照 posix 标准来设计。POSIX(portable operating system interface),指的是操作系统应该为应用程序提供什么样的接口标准。这套标准最早是来自 UNIX 系统，他为程序移植提供了一种标准方式。以下面几个典型的 posix 标准函数为例展开，讲述，嵌入式软件系统也需要标准化驱动设计，Ioctl，设备驱动程序中对设备的 IO 设备进行管理的函数，包括对设备的一些特性进行控制。Open 可以返回文件句柄，也是一种指针。Read ,Write,Close 方式也是类似的[42]。如何在嵌入式系统中实现以上函数。

5.7 升级工具设计

Bootloader 是什么？

不管是在单片机还是在片上系统上都会用到，便于在产品设计制造维护后期，为了在一个相对闭塞的机械外壳外，提供一个对原有应用代码存在的 bug 进行修正或者非现场维护。Bootloader 是芯片启动程序加载的第一段程序。在功能上与 bios（basic input/output system）,这是用在计算机或者微控制器启动计算机，在完成一定的硬件初始化工作以及人机交互后，用来加载操作系统的程序。当应用程序需要重新更新时，我们可以将芯片程序 app 程序通过指令跳转到 bootloader 区域，这样，app 处于非活动区域。当芯片运行在 bootloader 态，通过外部接口，如 CAN,USB,UART,SPI,LIN 等通信接口介绍自动化升级命令或者手动输入命令，让芯片处于数据接收准备模式。下一步，启动文件传输，将文件以一定的方式，如每帧 252 字节的方式传输给芯片。在基于 arduino 和 RaspberryPi 的开源硬件平台上，我们采用 linux 的脚本命令传输，开源社区有两个经典 avrdude 和 stm32loader. py 实现升级[43]。

Avrdude 是一族使用在 AVR 单片机上,运用 ISP 接口升级 ROM 和 EEPROM 内容的工具，上传，下载，操作等功能。它具有以下几个特征：1、命令行用户接口，包含了所有的下载和升级特性，makefile 文件中的自动化测试。2、交互性验证和修改各类内存区域，也就是所谓的终端模式，提供了一个修改操作参数。3、据了解是可以运行在各类符合

POSIX 标准的操作系统上，当然也包含 win32 系统。使用现成的操作系统驱动。4、支持广泛的编程硬件工具。5、支持 Intel Hex, Motorola S-record, 原始二进制文件作为输入输出文件。

stm32loader.py 是一个 python 语言写的软件工具，用于在 Raspberrypi 的 Linux 系统上升级。告知 stm32 bootloader 要升级和下载固件了。

本文以 STM32F4XX 系列芯片为例，讲述新一代的单片机的 bootloader 如何设计 STM32 采用 HCC 公司提供的基于 USB，串行接口的 bootloader。理论上来看，可以采用任意接口，包括 USB,串口，SPI,LIN，I2C，只是根据项目的性能要求选择不同的数据传输方式而已[44]。

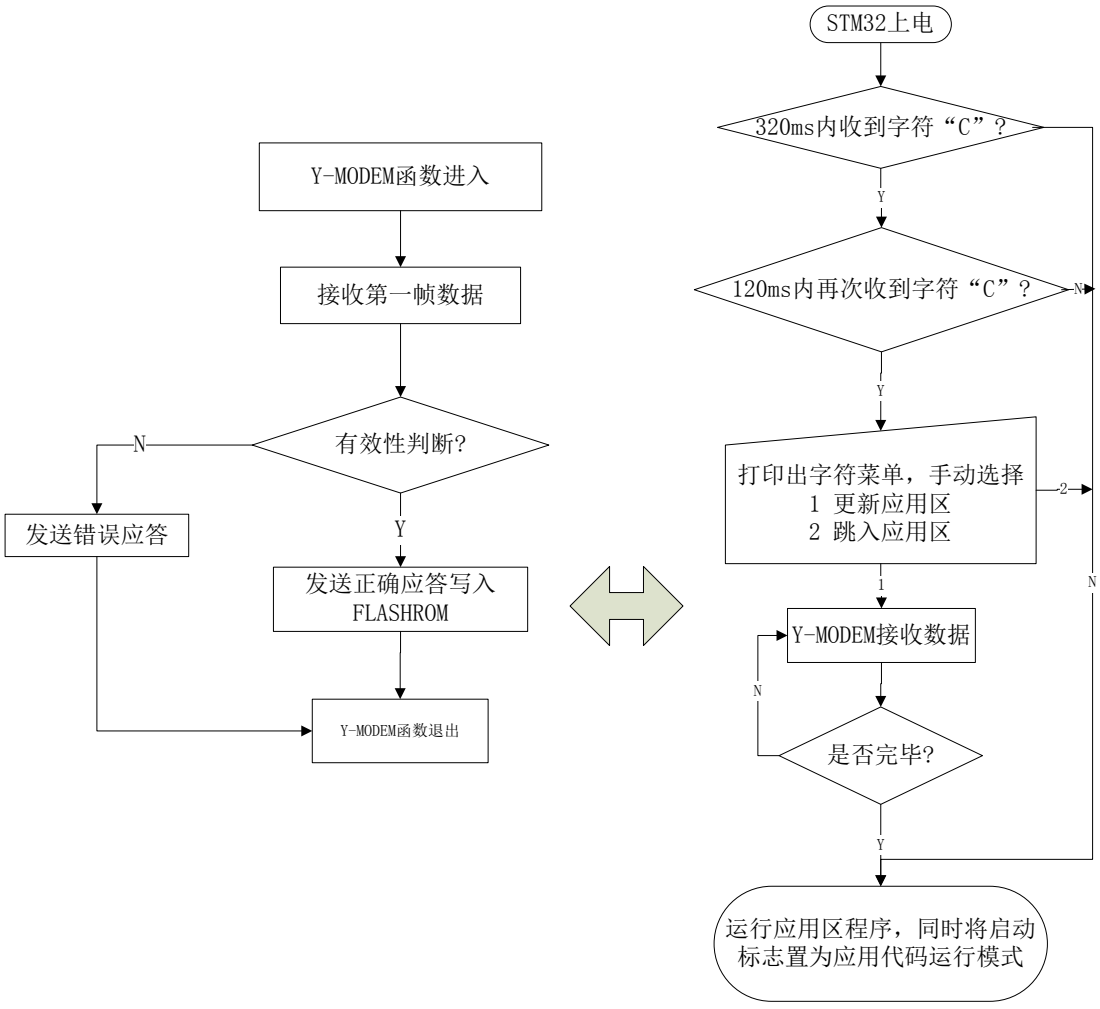

图 5.10　基于 Y-modem 协议的 bootloader 设计流程图

Fig.5.10　the design program of bootloader based on Y-modem

Bootloader 的基本软件思想是，串口按照一定的通信协议，一般是 X-modem, Y-modem, 从串口接收数据和写数据到 FLASHROM 中，两者可以同步。一边接收数据，一边进行数据 checksum 校验，校验完成之后，再进入下一帧数据的接收和写数据[45]。

5.8 开源项目管理方式探究

开源项目最大的特点就是社区开发，志愿者贡献。我们在基于开源软硬件技术的电力电子嵌入计算平台的软件实现部分，最重要的是如何实现开源的精神。充分利用好开源社区服务器，如服务器 github, 工具 git，这些工具是全球顶级的开源项目都采用的模式[50]。

如何做好迭代和测试。

版本控制系统，在开源项目开发中扮演了一个非常主要的角色。不管代码和文档在开发中总是不断变化逐渐完善的，但是由于这些开发的工作难免在开发过程中有这样那样的失误，误删除误增加一些代码引入一些不应该的问题。在没有版本控制系统之前，我们只能定期做简单的拷贝和备份，但是这样的工作会一方面占硬盘空间，另外一方面效率不高，回溯能力不强。跟人感觉这样的软件开发就像用作坊开发的模式来做软件开发，不利于大型的项目开发和维护。所以引入版本控制系统，一般公司都做了这样的系统，但是这样的系统在软件管理经理那里一般都会有这样一个要求必须满足一定的需要再做提交。基于这样一个两难的情况，一方面我们需要版本控制，但是公司的版本控制系统又必须要满足这样一个限制要求。

能不能有一种，本地独立维护的版本控制系统呢，这就是集中式版本控制和分布式版本控制系统的关键区别所在。

针对我们本地的比较频繁的代码修改，以及我们对频繁修改的跟踪和回溯的需求，选择 git 将是非常明智的选择。git 在构建本地化代码控制系统方面有一下以下独特的优势：

1. 分支更快，更容易
2. 支持离线工作，本地提交可稍后提交到服务器上。

Git 是一款开源的版本控制系统，它属于分布式类型，可以高效的处理大大小小的软件项目版本之间管理。Git 最开始是 Linux Torvalds 管理 Linux 内核开发的一个开源源码的版本控制软件。最开始只是为了当时广泛使用的 BitKeeper 的。正是因为 linux 的推动促进了 git 在开源社区获得了广泛的认可，并被很多知名开源软件项目所使用，用于后续专门的领域研究性文档的管理。

5.9 本章小结

本章首先讲述树莓派和阿都诺在电力电子嵌入计算平台设计框架的情况，然后简单讲述了开源软硬件技术的知识管理和项目管理知识。

6 全文总结与展望

6.1 总结

本论文通过数学理论分析，后续采用开源数值计算软件 scilab 仿真和实验。本论文试验部分主要包括硬件平台的搭建，主要数学模型的分析，以及在其他平台中验证经验。本文采用开源硬件之后，搭建电力电子嵌入计算平台，省去硬件制板，焊接，调试三个环节。软硬件调试，原计划将仿真结果直接运行在实际的开源硬件平台上树莓派和阿都诺组合的电力电子嵌入计算平台。从选题，对开源硬件的选择，对开源硬件的熟悉，对开源软件的学习，以及开源计算软件的摸索。这些方法大大提高效率，减少算法转化为程序的时间，提高了传统电子设计中反复使用的传感器电路模块，核心板模块，充分吸收了在开源社区中别人优秀的 C 算法，代码风格。通过实验和学习可以认为在电力电子嵌入计算应用的一些场景是可以采用开源的软硬件技术的。

6.2 进一步工作的展望

开源软硬件技术的研究不仅是传统的 IT 行业的一个技术热点，而且也是研究性能优良的现代工业控制的关键技术。对于基于开源软硬件的电力电子嵌入计算平台的研究，不仅要具有良好的电力电子技术的理论基础，更要有优秀的开源软硬件思维和动手能力，其中特别如何应用开源软件技术的能力。本文在开源硬件，开源软件做了深入研究，通过研究分析，初步认为开源软件技术可以应用于电力电子计算上。下一步，作者打算结合开源数字计算软件 scilab 在嵌入式 Linux 得以实时仿真，通过传统 C 语言将仿真结果和仿真流程转换为具体实施代码，运行在根据无速度传感器的控制系统上。后续的工作还有很多要展开。

由于作者从事开源软件技术应用于电力电子行业的时间还比较短，水平还比较有限，论文错误之处在所难免，肯定各位老师和同行专家指正。

参 考 文 献

[1]柳兰,龚学余,肖金凤.无速度传感器感应电机控制系统的现状与展望[J].现代电子技术.2006.
[2]徐志根,刘忠举.基于 MRAS 的无速度传感器异步电机 DTC 系统[J].电力电子技术.2006.
[3]李自成.感应电机无速度传感器矢量控制系统的研究[D].硕博学位论文.2005.
[4]郝晓弘,高超,李桂素等.基于空间矢量无速度传感器的直接转矩控制[J].电气传动.2007.
[5]徐中领,李桥梁.交流感应电机无速度传感器矢量控制系统设计[J].安徽电气工程职业技术学院学报.2006.
[5] Gumus B, Ozdemir M. Sensorless vector control of a Permanent magnet synchronuous motor with fuzzy logic observer [J].Electrical Engineering. 2006,8(8):395-402.
[6] Edelbaher G,Jenernik K, Member S, et al.Low-speed sensorless control of induction machine[J]. IEEE Transactions on Industrial Electronics. 2006, 53(1): 120-130.
[7] Holtz J. Sensorless control of induction motor drives [J]. Proceeding of the IEEE. 2002,90 (8):1359-1395.
[8]王伟.基于开源软硬件技术的嵌入计算平台研究与实践[J].单片机与嵌入式系统应用.2013.11.
[9]宾俊.基于 Arduino 和 Python 搭建的实时在线 pH 测量平台[J].计算机与应用化学.2013.10.
[10]王大虎等.基于 Arduino 控制板的压力采集监测系统设计[J].河南理工大学学报.2013.4.
[11]张晓丹.应用计算方法教程[M].机械工业出版社,2008.
[12]胡广书.数字信号处理[M].清华大学出版社,2006,26(18): 71~76.
[13]严蔚敏.数据结构[M].清华大学出版社, 2012.
[14]马龙华.基于 scilab 的 ARM-linux 嵌入式计算及应用[M].科学出版社 , 2011.
[15]谭浩强.C 程序设计[M].清华大学出版社 , 2011.
[16]钱能.C++程序设计教程[M].清华大学出版社 , 2011.
[17]Raspberry Pi Home Automation with Arduino.pdf.
[18]库少平.EAGLE 电路原理图与 PCB 设计方法及应用[M].北京航空航天大学出版社，2013.
[19]翁恺.Arduino 技术内幕[M].人民邮电出版社,2013.
[20]张宝玲.基于 Arduino 的趣味电子制作[M], 科学出版社，2011.
[21]唐乐.Arduino+Android 互动智作[M]，科学出版社，2013.
[22]杨继志.Arduino 从基础到实践[M]，电子工业出版社，2013.
[23]程晨.自律型机器人制作入门—基于 Arduino[M],北京航空航天大学出版社，2013.
[24]刘槮楠.Arduino 编程从零开始[M],科学出版社,2013.
[25]于欣龙.爱上 Arduino[M],人民邮电出版社,2012.
[26]孙骏荣.Arduino 一试就上手，科学出版社,2013.
[27]程晨.Arduino 电子设计实战指南[M],机械工业出版社，2013.
[28]程晨.Arduino 开发实战指南[M],机械工业出版,2013.
[29]谭亮.新媒体互动艺术—Processing 的应用[M],广东高等教育出版社,2013.

[30]周自恒.Android 应用开发入门[M],人民邮电出版社，2013.
[31]马龙华.基于 scilab 的 ARM-linux 嵌入式计算及应用[M].科学出版社 ,2011.
[32]魏永明.Linux 设备驱动程序[M].中国电力出版社，2006.
[33]陈健.Linux 程序设计[M]，人民邮电出版社，2010.
[34]董西成.Hadoop 技术内幕,深入解析 MapReduce 架构设计与实现原理[M],机械工业出版社,2013.
[35]吕晶.开源软件之道[M].电子工业出版社，2010.
[36]康万新.毕业设计指导及案例剖析应用电子技术方向[M].清华大学出版社，2007.
[38]陈森.面向电力电子系统的 TMS320F2812 通用平台的设计与实现[D]. 硕博学位论文.2008.
[39]Peng F Z. Robust speed identification for speed-sensorless vector control of induction motors [J]. IEEE Transactions on Industry Applications. 2004, 5(30): 1234-1241.
[40]Kerkman R J, Seibel B J, Bowan T M. A new flux and stator resistance identifier for AC drive systems [J]. Electrical Engineering in Japan. 2000, 110(4): 129-139.
[41]Leohard W. Control of Electric Drives, 3rd ed. Berlin, Germany: Springer-Verlag, 2001.
[42]刘晋.基于嵌入式操作系统的电力电子控制平台的研制[D].硕博学位论文.2003.
[43]吴斌.普适计算环境下轻量级中间件平台的研究和实现[D].硕博学位论文.2005.
[44]胡海兵.电力电子集成系统中的数字控制平台研究[D].硕博学位论文.2007.
[45]侯思名.基于开源软硬件的智能家居系统设计与实现陈儒敏[J].现代计算机.2013.
[46]王文汉.开源需要创新[J].2008.
[47]郑悦.同一个开源,同一个梦想[J].中国计算机用户.2006.
[48]程文婷.开源软件开发者激励机制研究[J].网络法律评论.2011.
[49]张宏烈.面向可重构系统的资源管理与软/硬件划分研究[D].硕士论.2011.
[50]吴广霖.μCOS 的 μCIP 协议栈在 ARM 系统中的实现[J].计算机工程与应用. 2005.
[51]刘晋.基于嵌入式操作系统的电力电子控制平台的研制[D].硕士论文.2003.

攻读硕士学位期间发表论文及科研成果

1　**王伟**. 单片机与嵌入式系统应用，2013 年，11 卷（期）：基于开源软硬件的嵌入计算平台研究与实践

致　　谢

光阴似箭，日月如梭！美好的三年研究生时光随着毕业论文的完成已经接近尾声。本论文是在我的导师杨燕翔教授的悉心指导和不懈支持下完成的；衷心的感谢老师在十分忙碌的教学工作中仍挤出时间来指导我选题，查阅资料，撰写论文，并仔细指导我修改论文。在此向杨老师致以最诚挚的谢意和最崇高的敬意。

非常感谢养育我的西华大学，母校浓厚的学习氛围，舒适的生活环境令我终生难忘。在西华大学学习期间，我的恩师们不仅向我传道授业解惑，还让我明白了许多做人的道理。感谢郑萍老师，她严谨的治学态度，精益求精的工作作风对我影响深远。感谢黄小莉老师，她深渊的专业知识，诲人不倦的高尚师德我将铭记在心。感谢所有的西华教师，感谢你们让我度过了美好的研究生学习生活。

感谢陪伴了我三年的室友、同学和各位实验室同仁。感谢他们为我的学习和生活提供的帮助和建议。正因为有了室友、同学和各位实验室同仁的帮助，我的研究生学习生活才五彩斑斓，正因为有了室友、同学和各位实验室同仁的建议，研究生学习生涯才得以顺利完成。

最后，衷心感谢我的朋友及家人。感谢我的好友罗富强和姚振国，感谢他们对我选题及撰写论文所提供的独到见解与无私帮助。感谢我的妻子陈玖红女士，在论文的语言和格式方面作大量的修改工作。感谢我的父母，感谢父母含辛茹苦将我养大，感谢他们对我的理解与支持。