



普通高中教科书

信息技术

选择性必修 6 开源硬件项目设计

普通高中教科书

信息技术

选择性必修 6

开源硬件项目设计

总主编: 李晓明

副总主编: 赵健

本册主编: 方向忠

本册副主编: 杨晓哲

编写人员(按姓氏笔画排序):

方向忠 朱兰娟 苏宇彤 杨晓哲 吴俊杰 徐雄 奚骏 曾贊

责任编辑: 高烨

美术设计: 储平

普通高中教科书 信息技术 选择性必修6 开源硬件项目设计

上海市中小学(幼儿园)课程改革委员会组织编写

出版发行 华东师范大学出版社(上海市中山北路3663号)

印 刷 上海华顿书刊印刷有限公司

版 次 2021年3月第1版

印 次 2025年1月第7次

开 本 890毫米×1240毫米 1/16

印 张 9.25

字 数 163千字

书 号 ISBN 978-7-5760-0554-7

定 价 11.60元

版权所有·未经许可不得采用任何方式擅自复制或使用本产品任何部分·违者必究

如发现内容质量问题,请拨打电话 021-60821714

如发现印、装质量问题,影响阅读,请与华东师范大学出版社联系。电话: 021-60821711

全国物价举报电话: 12315

声明 按照《中华人民共和国著作权法》第二十五条有关规定,我们已尽量寻找著作权人支付报酬。著作权人如有关于支付报酬事宜可及时与出版社联系。

本册教材图片提供信息:

本册教材中的部分图片由全景网、视觉中国等图片网站提供。

致同学们

飞速发展的信息技术不断改变人们的思维与交往模式,身处科技发展日新月异的时代,坚持守正创新,培育创新文化,营造创新氛围,在熟练使用信息技术工具的基础上,“如何学习”“如何参与”“如何创造”是我们需要认真思考的问题。

当有一天,你不再满足于仅仅作为一个产品的使用者,你想知道它是怎样被设计和制造出来的,或者想为它添加新的功能,又或者想为他人设计新产品,你将如何开始?比如,你想设计一个智能水杯,用来提醒自己和家人多喝水;你想打造一个机器伙伴,以帮助自己养成良好的学习习惯;你想开发一个运动助手,通过它来更好地达成运动目标;你想尝试搭建一辆可以自动驾驶的无人小车……你可能还想到很多很多,你甚至想改造整个世界。

然而,你想到的这些与众不同的一切,可能没法直接找到现成的产品。你需要拥有一种全新的思维方式,以海纳百川的宽阔胸襟借鉴吸收人类一切优秀文明成果,共建共享知识体系,推动建设更加美好的世界。“开源硬件项目设计”可以帮助你用开源的视角重新审视与发现,哪些开源硬件可以支持你的创意,哪些开源代码可以支持你的构想,哪些开源系统可以支持你的设计。你需要运用开源社区,开展合作,获得支持,不断优化和迭代自己的作品。你也需要将自己新的发现、新的作品分享到开源社区,不仅仅为了展示自己,也为了方便其他人进行创造。

按照《普通高中信息技术课程标准(2017年版)》,“开源硬件项目设计”是选择性必修课程中的一个模块。在内容的呈现方面,教材按照开源硬件项目设计的流程,呈现从需求分析到发布共享的全过程;

在案例的选择方面,从制作简单作品开始,逐渐实现作品之间的互联和综合制作。在学习的过程中,同学们不仅要学习“怎么做”,更要综合运用科学、技术、工程、数学、艺术等多学科知识,理解“为什么”,感受动脑思考、动手制作、合作参与和创造作品的乐趣,践行开源与知识共享的理念,提升计算思维与创新能力,拓宽视野,“点燃”科技梦想,为实现创新驱动发展的目标,牢记初心使命、开创美好未来。

编 者

目 录

第一章 开源文化与开源硬件 ... 1

项目主题 初探智能水杯 ... 3

第一节 开源文化 ... 4

第二节 开源硬件平台及其结构 ... 12

第三节 开源硬件项目设计方法 ... 36

第二章 开源硬件系统的输入和输出 ... 45

项目主题 打造机器伙伴 ... 47

第一节 开源硬件系统的微控制器 ... 48

第二节 开源硬件系统的输入 ... 54

第三节 开源硬件系统的输出 ... 66

第三章 开源硬件系统的模块扩展和连接 ... 83

项目主题 开发运动助手 ... 85

第一节 通信方式简介 ... 86

第二节 通信方式应用 ... 89

第四章 开源硬件项目设计实践 ... 105

项目主题 搭建无人小车 ... 107

第一节 无人小车·整体设计 ... 108

第二节 无人小车·运动与遥控 ... 112

第三节 无人小车·智能控制 ... 118

第四节 无人小车·项目发布 ... 129

附录 开源硬件平台编程及输入/输出模块简介 ... 133

后记 ... 139



第一章

开源文化与开源硬件

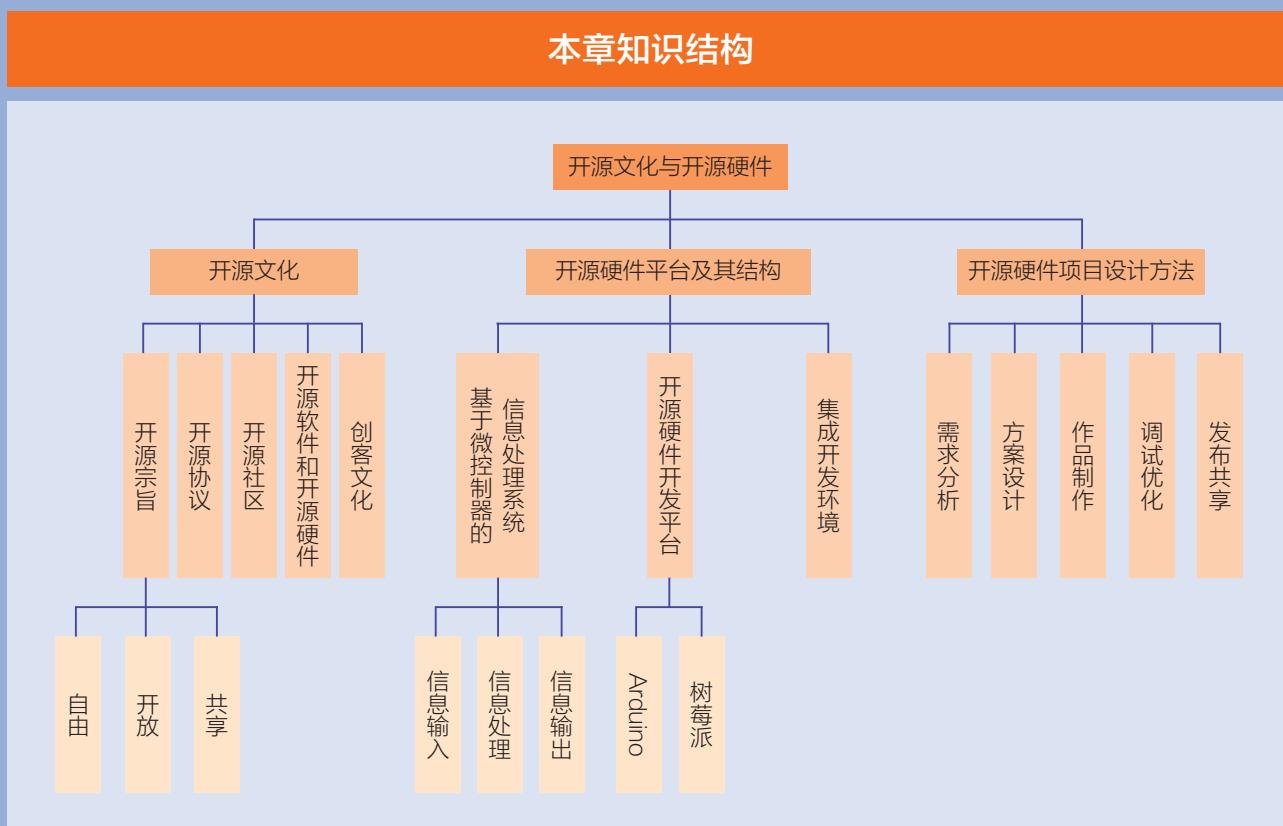
本章学习目标

-
- 了解开源文化、开源协议和开源社区，认识开源硬件的特征，理解开源的理念与知识共享的精神，理解知识产权保护的意义。
 - 了解计算机信息处理过程，知道开源和闭源的区别。
 - 知道常用开源硬件的种类、功能和基本组成结构。
 - 掌握开源硬件开发环境的基本使用方法。
 - 理解开源硬件项目的开发流程，初步掌握开源硬件项目的需求分析、方案设计、系统制作、调试优化、作品发布等的方法。
-

广泛应用的信息系统、不断扩增的数据以及不断迭代优化的算法,提高了采集、传输、存储、加工和使用信息的效率,深刻影响着人们的生活、工作和学习,推进着我国信息化智能化体系建设的发展。由硬件、软件、数据和人等要素组成的信息系统,既可以用“闭源”的方式也可以用“开源”的方式建设和开发。

日常生活中,很多互联网网站是基于开源的操作系统、WEB服务器和数据库开发出来的,一些智能手机中的Android操作系统源于开源的Linux,可以说,只要上网或者使用智能手机,我们就在不知不觉中使用了开源软件。

在本章的学习中,同学们将以创造人类共建共通共享的知识体系为目标,初步感受以“自由、开放和共享”为宗旨的开源文化,从信息处理过程的角度,了解开源硬件的起源和发展,尝试使用开源硬件和图形化编程软件进行简单的系统搭建和编程控制,理解开源硬件的结构和功能。在学习和认识开源世界、享受开源产品所带来的丰富多彩的生活、感受科技所创造的价值的同时,同学们要以开放创新的理念进一步思考:开源是为了什么?开源能创造什么?开源要遵守什么?



项 · 目 · 情 · 境

开源硬件为我们提供了一个全新的、可以不断进行创造的学习空间,生活中很多产品都可以使用开源硬件设计制作。从不起眼的烟雾报警器,到炫酷的呼吸灯,再到造型新颖独特的智能水杯,人们不断创造新的产品,让生活、工作和学习更便捷。

我们每天需要补充充足的水分,饮水不足会引发很多健康隐患,然而我们在忙碌的学习工作中常常忘记喝水。设想这样一个按时提醒饮水的智能水杯:早上起来洗漱完毕后,智能水杯提醒饮水,并显示水量和水温,饮水完毕,自动补水,间隔一段时间再次提醒饮水……如果将这样的智能水杯制作出来,我们就能很大程度上避免饮水不足的问题。如何运用开源软硬件实现智能水杯的提醒功能和显示功能?对于一些特殊人群,比如老人,智能水杯又该如何设计以满足他们的需求?

在这一章中,我们将围绕智能水杯的制作,学习开源硬件系统编程调试方法和开源硬件项目设计流程,完成智能水杯设计方案。

项 · 目 · 任 · 务

任务 1

分析常见的信息处理系统中的信息流向,从信息处理过程的角度,完成智能水杯的结构描述。

任务 2

探索典型的开源硬件组成结构和开发环境,选择合适的控制器及开发环境来设计智能水杯的电子控制系统。

任务 3

按照开源硬件项目的开发流程,完成智能水杯的项目设计。

第一节 开源文化

开源(open source),即开放源代码,最初指的是一种软件的发布模式。软件作者遵循“自由、开放和共享”的精神,在一定的协议规则约束下将软件源代码公开,让大家共享使用。这是起始于计算机科学领域的一种文化现象,知识成果的开放、共享激发了源源不断的创意。随着开源理念的不断深入和推广,“开源”的内涵和外延不断丰富和扩展,形成了一种更广义的开源文化。

体验思考

开源文化发展至今已经过数十年的历程,产生了许多开源软硬件成果。随着互联网的发展,开源更是渗透到了信息、教育、经济、生活等多个领域,它不只限于软件,还逐步涉及硬件,成为信息技术发展的重要推动力。

你了解开源文化吗?在你生活的周遭,你接触到的事物中有哪些体现了“自由、开放和共享”的精神?



图 1.1 开源软件标识

一、开源的起源和宗旨

“开源”一词来源于“自由软件”(free software),自由软件是指倡导尊重用户“自由”,将源代码公开,让用户可以自由使用、复制、修改和分发源代码的软件。但由于“free”在英文中有“自由”和“免费”双重含义,free software 容易被人误解为免费软件,再加上自由软件的反商业信条也让企业和公司对其敬而远之,因此在 1997 年的一个战略研讨会上,一批很有影响力的自由软件创始人决定改用开源软件(open source software)来描述自由、开放的理念。开源软件的标识如图 1.1 所示。

开源以“自由、开放和共享”为宗旨,以协议和合作的创新方式促进开源同商业的结合,即在提倡开放和知识共享的同时,允许人们以协议维护产权,以专利形式从产品中获取利益。在它的推进下,网景公司正式宣布将他们的旗舰产品网景浏览器的源代码免费开放,这一举动为开源运动的发展迎来了转机。

自由软件与免费软件不同,“free”虽有“免费”之意,但“免费”不等于“自由”。首先,免费软件只是免费提供给用户使用,并不公开源代码。其次,我们虽然可以免费使用某些软件,但却不能自由地使用,因为这些软件使用了数字版权管理(digital rights management,缩写为 DRM)技术后,便可以合法地追踪用户的使用行为。

作业练习

开源也为同学们学习编程提供了很好的机会,许多工程师都是从开源中开始学习和汲取灵感的。想知道你最喜欢的网页是如何运作的吗?在 Web 浏览器中点击“查看源代码”,尝试阅读网页的源代码,通过查阅资料了解一些代码的含义,思考源代码与网页效果之间的对应关系。

二、开源协议的作用

开源的基本理念是追求“自由、开放和共享”。但是,任何开源都不是简单地、无条件地、无理由地完全开放,而是要遵循一定的规则,即协议。在开源领域的协议被统称为开源协议或开源许可证(open source license),是指为了保护开源软件原作者的相关知识产权,在软件所有者与软件使用者之间设立的一种具有法律性质的虚拟合同,它也可被视为一种授权方式。

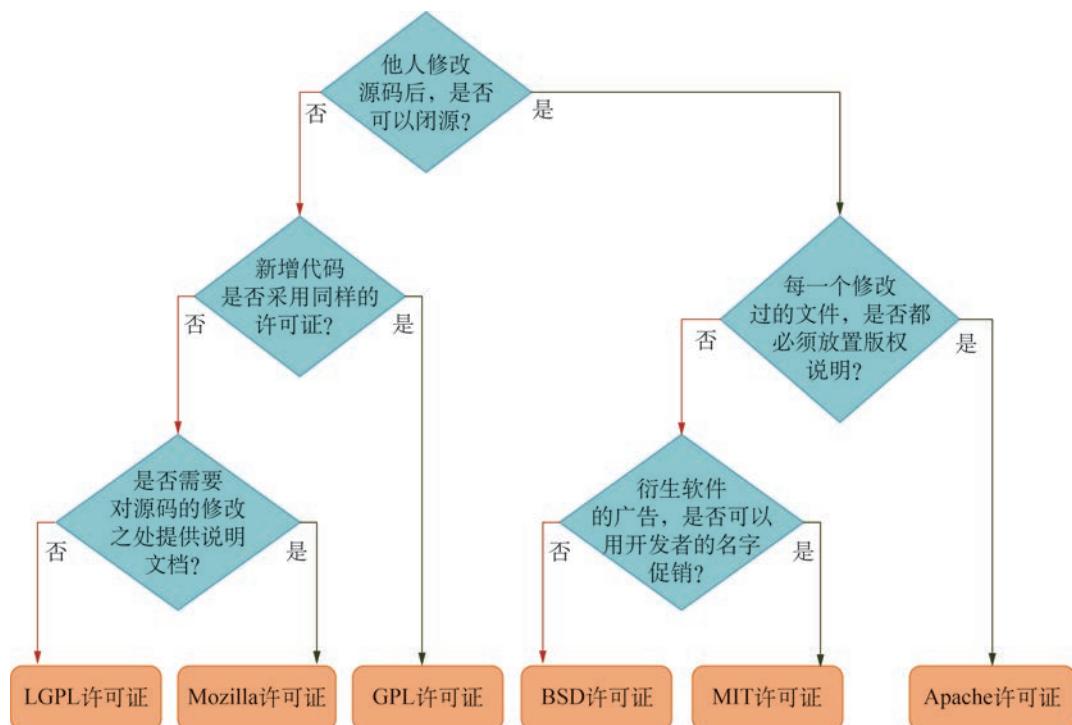
1. 通过网络搜索引擎,找一找有多少种开源协议。
2. 试着对不同的开源协议进行分类,说明为什么要有如此多不同的开源协议。

开源软件就是在开源许可证下发布的开放源代码的软件。开源许可证是授权使用、分享开源软件和其他产品的源代码、源设计等的条款,它提供了一种独特的产权模式,使得开源软件与专有软件一样,也受版权法保护。例如,它可以要求用户在所使用或修改的源代码中保留原作者名字和版权声明,或者要求用户再发布时必须遵循同一个许可证等。开源许可证的意义就在于对运行、复制、修改、发布受版权保护的软件等行为进行规范,保障原作者的合法权益,保护软件开发者共享合作的积极性,保障开源文化健康持续地发展。

通常开源许可证主要分为两大类：左版版权(copyleft)许可证和宽松式(permissive)许可证。copyleft是作为copyright(著作权)字面意义上的反义词而被发明出来的词汇，其含义是“给予任何人运行、复制、修改以及发布改变后程序的许可，但前提是发布的条款不能被改变”，也就是说任何人都可以自由使用左版版权源代码，但修改后的代码也必须采用左版版权许可证发布，不能闭源，而宽松式许可证则对用户几乎没有限制，用户可以修改开源代码后选择闭源。

目前国际公认的开源许可证有上百种，最常用的有GPL、LGPL、BSD、MIT、Mozilla、Apache、CC等。不同的开源许可证规定了不同的自由度和不同的商业模式，比如，是否允许商用，是否允许修改，修改后是否需要继续开源，开源是否必须继续使用相同协议授权等。图1.2描述了部分开源许可证之间的区别和适用情况。

图1.2 部分开源许可证之间的区别及适用情况



知识延伸

开源许可证的比较

表1.1中所列的是部分开源许可证之间的比较，出于不同的考虑，可以对开源的内容选取不同的许可证来保护。

表 1.1 部分开源许可证之间的比较

| 许可证 | 版本 | 是否包含源代码 | 是否允许链接 | 是否允许商业使用 | 是否允许私人使用 | 是否允许发布 | 是否允许修改 | 是否有专利许可 | 是否有转售许可 | 有无担保责任 | 有无商标 |
|----------------|-----|---------|--------|----------|----------|--------|--------|---------|---------|--------|------|
| Apache 许可证 | 2.0 | | | 是 | 是 | 是 | 是 | 有 | 有 | 有 | 有 |
| 3 句版 BSD 许可证 | | | | 是 | 是 | 是 | 是 | | 有 | 有 | 有 |
| 2 句版 BSD 许可证 | | | | 是 | 是 | 是 | 是 | | 有 | 有 | |
| GNU 通用公共许可证 | 2.0 | 是 | | 是 | 是 | 是 | 是 | 有 | 无 | 有 | |
| GNU 通用公共许可证 | 3.0 | 是 | | 是 | 是 | 是 | 是 | 有 | 有 | 有 | |
| GNU 宽通用公共许可证 | 2.1 | 是 | 是 | 是 | 是 | 是 | 是 | 有 | 有 | 有 | |
| GNU 宽通用公共许可证 | 3.0 | 是 | 是 | 是 | 是 | 是 | 是 | 有 | 有 | 有 | |
| MIT 许可证 | | | | 是 | 是 | 是 | 是 | | 有 | 有 | |
| Mozilla 公共许可证 | 2.0 | 是 | | 是 | 是 | 是 | 是 | 有 | 有 | 有 | 有 |
| Eclipse 公共许可证 | 1.0 | 是 | | 是 | 是 | 是 | 是 | 有 | 有 | 有 | |
| Affero 通用公共许可证 | | 是 | | 是 | 是 | 是 | 是 | | 有 | 有 | |
| 一般的著作权 | | | | 是 | 是 | 否 | 否 | | 无 | | |

作业练习

在开源运动中,关于版权保护一直争议不断。很多开发者由于不了解开源许可证的内容,在开放源代码的时候不知如何选择;也有的开发者在使用别人的开源代码时未遵循其原始许可证,由此带来了许多不必要的法律纠纷。

1. 拥有 GPL 许可证的开源软件 MySQL 是一个常用的数据库软件,其所属的公司于 2009 年被收购,有人担心新公司会将 MySQL 同其他产品合并而不再开源。你认为这种情况会发生吗? 为什么?

2. 如果你希望将自己开发的项目开源,并允许别人使用和修改你的代码,但要求使用者也必须公开源代码,那么你在发布项目时可以选用哪些许可证?

三、开源社区

开源文化的另一个重要标志是催生了一种新的组织结构——开源社区。所谓开源社区指的是在互联网上针对某个或某些开源代码而形成的一个共享学习与交流的平台。

探究活动

开源社区一般由开源技术的核心团队、领导者用户或者知名的企业搭建,由全世界的编程爱好者自发参与,比较著名的开源社区有开源中国社区、Linux 社区、码云社区、GitHub 社区等。在社区中,大家遵循相应的开源许可证,通过论坛、博客、跟帖、推送等方式发布源代码、探讨问题、分享成果、合作创新,对开源软硬件的开发、完善和发展起到了巨大的推动作用。

试着进入开源中国社区的技术问答栏目,查看最新提问和最新回答的内容,探究开源社区能为我们提供怎样的平台。

在开源社区兴起之前,绝大多数商业软件公司采用的都是传统的开发模式,在过去高度集中、严密的封闭式管理下,开发过程艰难而缓慢。与之相反,开源社区采用扁平化、开放式的集市模式,不仅软件的发明者,互联网上众多的志愿者都可以自由地参与开发,就如同一个开放的集市,所有的社区成员好比集市中的一个个商户,大家遵守既定规则,相互监督,共同增进和维护社区的繁荣。

Linux 就是典型的集市模式的例子。1992 年 Linux 的第一个 GPL 版本在网上发布,立刻吸引了众多软件人才的介入,迅速推动了 Linux 的发展。到 1993 年第一批发行版 Slackware 和 Debian 出现时,Linux 社区上已有 100 多位开发人员。开发者们自由地修改和发布程序版本,能很快发现错误代码并加以修复。在这种快速漏洞响应机制的作用下,Linux 不断地完善,社区的规模迅速扩大,软件质量、工作效率和品牌影响力也得到很大的提升。

四、从开源软件到开源硬件

探究活动

随着开源文化的快速推进,开源的理念逐渐从软件领域拓展到了硬件领域,开源硬件(open source hardware,缩写为 OSHW)应运而生。开源硬件是已公开的硬件设计,包括电子设计图、机械设计图、电路原理图和材料清单等。

1. 收集与开源软件和开源硬件相关的资料,了解从开源软件拓展到开源硬件的缘由和过程;
2. 分析开源硬件对开源文化发展的作用,以及它在许可证方面的特殊性。

1. 开源硬件的概念

开源硬件是指“可以通过公开渠道获得的硬件设计,任何人可以对

已有的设计进行学习、修改、发布、制作和销售”。需要指出的是，开源硬件并不是可免费使用的硬件产品，而是由开发者公开的硬件的所有设计信息，以及与开源硬件相关的驱动程序源代码，可供使用者学习及进一步开发。因此，开源硬件除了秉承开源文化的开放、共享之外，还在成本低、可二次开发、可裁剪等方面对开源文化的内涵作了延伸。

开源硬件对开源理念的延伸还体现在把开源软件惯用的开源协议规范带到了硬件共享领域，衍生出开源硬件许可证(open source hardware license)，如基于 GPL-2.0 的 TAPR 和 CERN 开源硬件许可证，基于 Apache-2.0 的 Solderpad 开源硬件许可证等。不过开源硬件许可证的普及度不高，更多的人还是直接使用开源软件许可证，例如 GPL/LGPL、CC 等协议就常被运用于开源硬件领域。

相对于开源软件，开源硬件涉及的对象比较特殊，不仅包括硬件产品的设计信息(如设计文件、源代码等)，还包括硬件产品的实物本身。从知识产权的角度来看，硬件产品的设计信息受到开源许可证的保护，但由此制造出来的实物产品则只受专利权的保护。因此除非开发者拥有硬件实物的产品专利，否则其他人还是可以在未经许可的情况下复制硬件产品而不会侵犯硬件设计文件的许可证。

2. 开源硬件的特征

开源硬件源于开源软件，并扩展了开源的适用范围，因此开源硬件具备开源软件某些基本特点(如源代码开放、遵循一定的协议规范等)，其特征可归纳如下：

(1) **开放共享性**。开源硬件的设计图、原理图、材料清单等资料都是公开的，而且往往具有标准化的接口和完善的文档，任何人都可以方便地获取和使用，并可以在对已有的设计进行学习、修改或扩展的基础上，再予以发布，进而形成更高层次的分享。

(2) **二次开发性**。开源硬件允许在开放的原始设计信息基础上进行二次开发。使用者可以复用别人的开发成果，对开源硬件进行裁剪和选择，通过修改、再造，形成满足用户个体需要的相应产品设备。

(3) **迭代创新性**。开源可以使开发者很好地利用已有的类似产品或者基础产品进行迭代，避免低价值的重复工作。同时更多开发者的参与也有助于硬件项目的完善和更新，当开发者不再受专利授权所困，越来越多地公开分享他们的创新时，他们便能在充满智慧和创造力的开源平台上，获得更多的免费帮助，从而改进自己的发明，拓展硬件产品的外延，实现快速的迭代与创新。

事实上,开源硬件不仅仅是硬件设计方法的开放,更是一种创新理念的开放。

知识延伸

开源硬件的起源与发展

开源硬件的概念最早由国际业余无线电爱好者在项目实践中形成。1997年,为了让硬件制造商能够自行认证他们开放的硬件产品,这些无线电爱好者首次发起了“开源硬件认证计划”,正式确立了“开源硬件”的概念,开源硬件正式成为了开源文化中的一个重要成员。开源硬件的标识如图1.3所示。

然而开源硬件项目的发展并不是一帆风顺的。由于半导体产业的特殊性和过高的生产成本,开源硬件曾一度陷入难以维持的困境。进入21世纪后,随着FPGA和SoC(片上系统)设计在嵌入式系统(embedded system)应用中异军突起,原先阻碍开源硬件发展的生产成本过高等问题得以解决,加之国际知名企业在开源项目上取得巨大的商业成功,开源硬件又迎来了新的发展契机,开源硬件项目也如雨后春笋般涌现,出现了如虚谷号、掌控板、Arduino、树莓派(Raspberry Pi)、BeagleBone、micro:bit、Edison可穿戴设备平台、MakerBot 3D打印等一大批项目。此外,2017年发射升空的立方体卫星UPSat几乎没有使用商用元件,全部采用开源软件和开源硬件,是第一颗发射到轨道上的开源卫星。随着开源硬件思想渗透至硬件设计的各个层面,开源硬件作为产业正式走上成熟的发展道路。



图1.3 开源硬件标识

3. 开源硬件与创客文化

随着硬件成本的逐渐降低和互联网、物联网的蓬勃发展,开源硬件得到广泛的应用。在开源文化的熏陶下,特别是在Arduino、树莓派等开源硬件项目的影响下,出现了一批不以营利为主要目的,热衷于整合资源将创意变为实际产品的创新之人,他们被称为“创客”(maker)。开源硬件为创客提供了实现创意的工具和开放的创造空间,随着当今世界产业链生态圈的不断完善,开源资源变得触手可及,创客队伍也不断壮大,由此萌生出很多有趣和有创意的开源项目。秉承着开放、合作、分享的理念,创客文化也从以兴趣为主导的亚文化逐渐演变成主流文化,融入到开源文化之中。

创客文化一般有三个特点:首先,创客使用数字化工具和电子元器件设计产品并制作模型样品,创客的一部分工作是通过添加传感器、编写控制程序、连接互联网等,使实物变得更加智能;其次,对于在开源社区上分享成果、开展合作创客们已经形成一定的规范,他们在网上发布成果后,数以百计的人可以共享他的设计创意和代码,协助进行错误修正,这对创客改进自己的产品大有裨益;再者,在遵守开源协议的前提下,可以通过一定的标准将开源硬件设计文件直接提供给商业制造服务商,用于制造所

设计的产品,这就大大缩短了从创意到产品的时间,尤其是随着制造行业信息化程度的提高,一些行业逐渐产生了自动化、智能化、定制化、个性化的生产方式,将创意和发明实物化和商业化的效率也大大提高了。

创客文化对教育也产生了较大的影响,许多开源硬件已进入学校课堂,便利的设计、多种多样的元件、开放的文化这三个特点得到充分发挥,创客教育应运而生,它旨在培养青少年综合学习和动手实践的能力,同时为科技创新注入了新的活力。然而要成为一名创客并非易事,首先,创客要用与众不同和充满创意的方式思考问题,同时要考虑这个创意的价值何在,它解决了什么问题,它能为他人或社区作出怎样的贡献等。其次,创意只是一个开始,如何实现创意则更为重要,创客需要具备用开源硬件将创意转化为实体物品的能力以及一定的编程能力,在设计和制作作品的过程中要反复调试,这意味着创客还需要学习更多的“造物工具”。高质量的创造是每位创客的追求,开发有用的实体物品、简洁优美的程序和信息资源,并把它们贡献给社会,创客从这样的挑战中获得创造的乐趣。最后,创客还应是怀有热情的志愿者,以开源的方式在社区中共享成果、创造价值。

知 识 延 伸

开源硬件的典型应用——RepRap

RepRap 是世界上首个多功能、可自我复制、能打印塑料实物的 3D 打印机原型机。RepRap 最初的设计是为了实现自我复制。RepRap 的许多部件可以由自身打印和生产,人们只要愿意花一些时间收集足够的材料,就可以通过复制和组装 RepRap 自身的部件来实现 RepRap 的自我复制。因此,从某种意义上说 RepRap 也是一台“技术免费”的 3D 打印机。

RepRap 遵循开源协议,从软件到硬件各种资料都在 GPL 许可证下发布。这意味着任何人都能够自由地改进和制造 RepRap,并无偿地分享给大家。RepRap 的开源特性充分体现了开源文化“自由、开放和共享”的精神。因此,凡是在最初的 RepRap 机型的基础上制作,在技术层面上能实现自我复制,并无偿地提供给大家使用的机型,都可以被称为“基于 RepRap”的项目。

作业练习

基于对开源软件、开源硬件的了解与认识,你和你的团队打算创建一个开源硬件项目。为此小组成员决定先明确一个成功的开源硬件项目通常具备什么样的共性特点,以及应该选择什么样的硬件许可证。

1. 找出几个成功的开源硬件项目案例,了解项目的团队构成和项目所有更新的版本,并尝试分析、归纳出成功的开源硬件项目的共性特点。
2. 搜集有关开源硬件许可证 TAPR、CERN、Solderpad 的内容,对照图 1.2 的部分开源许可证之间的区别及适用情况,试着将它们归入相应的类别,必要时可加以扩充。

第二节 开源硬件平台及其结构

我们在日常生活、工作、娱乐和消费等过程中,往往有意识或无意识地接触到各种各样具有不同功能、不同规模的信息系统。信息系统的构建和运行离不开计算机硬件系统的承载,涉及的硬件平台有开源、部分开源和闭源之分。从开发者的角度来看,选择一个开源的硬件平台意味着你是站在前人的肩膀上开始设计的,这不仅能够降低前期的开发成本,还能使你摆脱低价值的重复工作而投入到高价值的创新工作;从开发过程的角度来看,由于世界各地的优秀程序员参与代码错误或安全漏洞检查,开源项目的代码会有更高的灵活性和稳健性,能产生更多的创意。

体 验 思 考

每天早晨 6 点电子闹钟准时响起,小申同学起床,感应台灯自动点亮。吃早饭时,小申同学通过语音智能音箱了解当天的新闻和天气,然后刷交通卡乘坐地铁去学校,在学校门口刷校园卡后进校,中午使用校园卡支付午餐费用,然后到学校图书馆通过检索系统查找一本科普读物,并使用校园卡登记借阅。

小申同学上述这段经历中用到了哪些信息系统?它们分别是什么?生活中还有哪些常用的信息系统?

一、基于微控制器的信息处理系统

随着计算机的广泛应用、网络的普及以及大数据等技术的日渐成熟,信息时代已然到来。对信息资源的充分利用是信息社会的重要特点,而构建各种各样的信息系统是开发、利用和管理信息资源的重要方式。信息系统的运行不仅需要计算机承载,也越来越离不开网络。信息系统处理能力和系统规模等不同,所需要的计算机和网络也有区别。此处的“计算机”是一个宽泛的概念,它不仅可以指日常使用的PC、笔记本电脑,还可以是其他形式的计算机,小到只有指甲盖大小的微控制器,大到体型庞大的超级计算机。比如处理飞机机票和铁路车票预订的大型系统通常由高性能服务器、云计算资源等支持,需要较高的数据传输能力、处理能力和存储能力作保障;与之相比,家用的电子闹钟和小型扫地机器人等系统,所需要的计算能力就会小很多,如果系统在使用上有轻巧、低功耗、低成本、高可靠性等要求,则可以采用集成在单个芯片上的计算机(单片机)来实现。这类为实现特定

的功能而专门优化设计的特殊计算机被称为微控制器。与通用计算机的明显不同之处在于,微控制器一般不连接显示器、键盘等常见的输入/输出设备,用户也不能够随意改动在其上运行的用来驱动特定设备的软件,因此这类软件也被称为固件(firmware)。

1. 信息处理系统的组成

信息系统的运行平台无论采用哪些硬件或采用何种互连方式,一般都具有相同的工作过程,即都能够按照一定的目标和规则,从外部世界获取信息,再对获取到的信息进行处理,然后将处理后的结果反

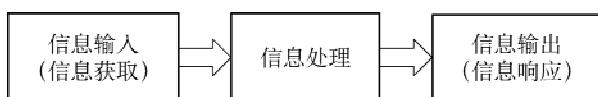


图 1.4 信息处理系统的工作过程

馈给信息接收方并作出响应。一般可将实现以上三个主要过程的模块作为一个整体,统称为信息处理系统,也就是说它由三个模块组成,即信息输入、信息处理和信息输出,如图 1.4 所示。

信息输入: 通过键盘、鼠标、传感器等输入设备,检测、识别和采集系统外部的信号源,根据采集到的信息的类型进行相应的转换和格式化等,完成信息的获取。

信息处理: 在中央处理器或微控制器的控制下,依照一定的算法规则对采集到的信息进行转换、判断、计算、加工、存储等,并根据信息接收者的要求作出相应的决策,完成输出信息的内容准备。

信息输出: 对信息处理得到的结果进行格式化、压缩、转换等,通过输出设备实现信息的定向传输和响应。用来完成特定任务的输出设备也被称为执行器或响应输出器件。

信息处理系统受益于计算机技术和设备,特别是开源软硬件设备,它们由于在性能、种类、价格和效率等方面的优势,正逐渐深入到我们的科研、教育、工业、农业、国防、日常生活等方方面面,由此涌现出许许多多为特定应用而设计的专用信息处理系统,这些信息处理系统在各个应用领域辅助或代替人类进行信息活动,已成为我们生产和生活的得力助手。

知识延伸

嵌入式系统

嵌入式系统是一类具有一个或多个特定功能的信息处理系统,一般都可进行实时计算,往往被用作某个设备的一个部件。这类系统一般都具有低功耗、高可靠性、可扩展等特点。而我们常见的个人电脑等通

用计算机用起来比较灵活,比如可替换硬盘、增加内存等,这样能满足更多用户的需求。

用在嵌入式系统中的处理器和微控制器发展很快,从最初的4位处理器(用于电饭煲等家电),到之后的8位处理器(用于游戏机、遥控器等)、16位处理器(用于机顶盒、路由器等)、32位处理器(用于网络设备、机器人等)等,其处理能力在不断增加,处理速度也从0.1 MIPS(million instructions per second,即每秒百万条指令)发展到超过1 000 MIPS。



图 1.5 烟雾报警器



图 1.6 烟雾报警器内部器件

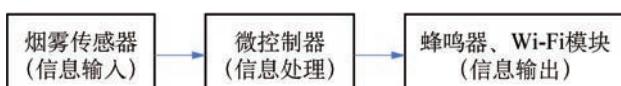


图 1.7 烟雾报警信息处理系统的组成



图 1.8 烟雾传感器



图 1.9 红外对管

2. 信息处理系统实例分析——烟雾报警器

在各种各样的公共场所,比如教室、图书馆、博物馆、咖啡厅等,都可以看到烟雾报警器的身影。它虽然是一个很小的装置,却成为现代安全设施的标配,默默地保障着人们的人身安全和财产安全。

如图1.5和1.6所示的烟雾报警器可视作小型的专用信息处理系统,它采用专门的烟雾检测装置,即烟雾传感器作为系统的输入模块,用集成度高、功耗小、成本低的微控制器作为烟雾信号处理模块,用蜂鸣器、Wi-Fi模块等作为输出模块,构成如图1.7所示的烟雾报警信息处理系统。

(1) 输入模块

传感器能感知外界信息,并能将得到的信息按一定规则变换成电信号或其他所需形式的信号。图1.8中的烟雾传感器的内部是如图1.9所示的一对侧斜向安装的红外光敏发射、接收对管。黑的为接收管,透明的是发射管。

由于发射管和接收管是侧斜向安装的,而光沿直线传播,没有烟雾时接收管是无法接收到发射管发出的红外信号的;当有烟雾进入烟雾传感器内部时,由于烟雾颗粒的作用,发射管发出的红外光发生散射,接收管便接收到红外信号。这个信号的强度与烟雾的浓度相关,因此在将该信号由电流转换成电压时,电压值的大小直接反映了外界烟雾浓度的大小。

烟雾传感器构成了信息处理系统的输入模块,它将得到的电压信号传送到微控制器后便完成了信息的获取。

(2) 处理模块

烟雾报警器的电路板上连接有一块微控制器芯片,它对接收到的烟雾信号进行处理。输入模块传送来的电压信号是一个在一定范围内(如在 0~3.3 V 之间)的模拟信号,而微控制器是数字信息处理器件,无法直接处理烟雾传感器传入的模拟信号,因此需要先把输入模块传送来的模拟信号转换成数字信号,这个通常由模数转换器件(analog to digital converter,简写为 ADC,或称为 A/D 转换器)完成,如图 1.10 所示。为方便用户使用,现今很多微控制器中都集成有 ADC,模拟信号可直接传入到微控制器的一些输入/输出接口(I/O 接口),再由这些接口传送进 ADC。



图 1.10 微控制器输入接口示意图



图 1.11 微控制器



图 1.12 蜂鸣器



图 1.13 Wi-Fi 模块



图 1.14 微控制器输出接口示意图

微控制器输出的都是数字信号,常用的输出模块多为数字器件(如蜂鸣器、Wi-Fi 模块等),可以直接接收微控制器传送来的数据。但当其输出信号用于驱动一些模拟端口的模块(如电机设备等)时,则需先将数字信号转换为模拟信号,一般由数模转换器件(digital to analog converter,简写为 DAC,或称为 D/A 转换器)进行转换后再输出,如图 1.14 所示。

烟雾报警器中的烟雾传感器、微控制器,以及蜂鸣器、Wi-Fi 模块,是构成信息输入、信息处理和信息输出三个模块的实体,它们通过相应的接口连接成一个有机的整体,组成烟雾报警信息处理系统。烟雾报警器的信息处理过程为:当周边环境的烟雾浓度较高时,烟雾传感器将这些数据传送到微控制器,微控制器将其与设定的阈值比较,若高于设定值则表示烟雾浓度超标,不超过该值则表示正常;微控制器根据判断结果控制输出模块,一旦烟雾超标,蜂鸣器便立刻发出报警声,同时 Wi-Fi 模块提醒管理人员采取措施。

烟雾报警器还可以设计成更加复杂的信息处理系统,如整合报警和处理功能,当烟雾浓度超标时,烟雾报警器不仅发出报警声,还能自动启动由微控制器控制的喷水装置,及时处理险情。

体验思考

若烟雾报警器是在一栋大楼中使用,如何将各个房间的报警信息汇总,实现集中管控?

作业练习

分析你周边的系统中哪些具备信息处理系统的构成要素。利用“信息输入——信息处理——信息输出”这根主线,将它们的各模块描述出来,填入表 1.2。

表 1.2 生活中信息处理系统的组成器件

| 信息处理系统 | 信息输入 | 信息处理 | 信息输出 |
|--------|------------------|------|--------------|
| 智能手环 | 运动传感器、脉搏传感器、触摸按键 | 微控制器 | 震动电机、显示屏、指示灯 |
| | | | |
| | | | |

二、常见的开源硬件开发平台

信息处理系统的运行依赖于有承载运行功能的硬件平台和相应的控制软件。开放性是基于开源平台的信息处理系统的特点之一,即允许用户在协议限制范围之内,从开源的核心硬件和应用于该硬件平台的软件出发,依据需要对系统进行二次开发,在已有的共享资源的基础上进行学习、改造、调试,甚至重新发布,从而搭建起功能更强大或更能适应特定环境的新系统,充分体现出开源的迭代创新特征。

体验思考

目前,开源硬件的典型代表是 Arduino 和树莓派系统,我们可以使用 Arduino 和树莓派制作出功能多样、极具创意的电子作品。观察几种典型的 Arduino 和树莓派开发板实物,阅读相关说明书,观察开发板上的各种标识信息,尝试回答:它们的品牌型号和供电方式是什么?你能否识别出其中的微控制器和输入/输出模块?

探究活动

随着开源硬件的兴起,关于开源硬件的创业成功案例不断涌现,开源硬件让创业者可以更轻松地将创意转化为现实。

1. 对比几种常用的开源硬件平台,了解它们的功能特点;
2. 对照典型的 Arduino 和树莓派开发板的硬件结构,归纳出它们的基本组成模块,并分析它们各自的适用场景。



图 1.15 Arduino 标识

1. Arduino 开发平台简介

Arduino 是目前较受初学者欢迎的一款开源硬件开发平台,由意大利一家高科技设计学校的师生在 2005 年联合设计完成。其标识如图 1.15 所示。

Arduino 开源平台包括硬件和软件两部分内容。硬件部分,即 Arduino 开发板,由微控制器、闪存(flash)以及一组通用输入/输出(general purpose input output, 缩写为 GPIO)接口等构成。软件部分主要包括用于编写 Arduino 程序的集成开发环境(Arduino IDE)



图 1.16 Arduino 开发板和 Arduino IDE

以及相关的开发包。Arduino 开发板和 Arduino IDE 如图 1.16 所示。

Arduino 是一款侧重于 I/O 性能的微控制器,支持底层设计,可以通过主板上的 I/O 引脚直接与开关、传感器、电机等各种相关部件连接,读取它们的状态信号,或者控制它们的运转;同时 Arduino 也支持与上层 PC 机的通信,可以实现交互式系统功能。Arduino 的主要优势是:

(1) 容易入门。Arduino 属于入门级的开源硬件开发平台,它提供的 Arduino IDE 支持图形化编程方式,开发工具包对硬件底层的操作进行了很好的封装,使得没有编程经验的人也能轻松上手,大大降低了学习难度和开发门槛。

(2) 可跨平台使用。Arduino 具有完善的开发环境,Arduino IDE 可以从网上免费下载,并且能够在 Windows、Macintosh、Linux 等主流操作系统上运行,适用范围广泛。

(3) 有较好的扩展性。Arduino 硬件电路板可以自行手动组装,还可以通过扩展板和自带的 I/O 接口与键盘、鼠标、传感器、电机、显示屏、蓝牙、Wi-Fi 等各种模块连接,实现功能扩展,完成模块化、标准化的设计,因此它也被称为“电子积木”。

(4) 有较大的开放性。Arduino 是起步比较早的开源硬件项目,不仅软硬件都完全公开,还提供了丰富完整的开源技术资料和完善活跃的支持社区。同时 Arduino 还预留了非常友好的第三方库开发接口,便于用户在成功实现了自己的设计后把成果分享给其他人,实现迭代更新。

2. 典型的 Arduino 开发板及其硬件结构

针对不同的应用需求,Arduino 开源平台提供了多种型号及众多衍生微控制器,如基于 8 位 CPU 的 Arduino Uno、Arduino Nano、Arduino Mega 2560 等,基于 32 位 CPU 的 Arduino Due、Arduino Exen Proto 等。

(1) Arduino Uno 开发板

Arduino Uno 是最常见的 Arduino 平台开发板之一,简单、实用、上手快。Uno 在意大利语中是“一”的意思,Arduino Uno 作为系列板中的第一个控制板,是 Arduino 平台的标准参考模板。Arduino Uno 采用一块 8 位的处理器芯片,开发板上同时具有闪存、内存、数字 I/O 接口、模拟输入接口、USB 接口、电源插座等,工作电压为 5 V,开发板

大小规格为 68.6 mm × 53.4 mm。

与大部分的开发板一样,Arduino Uno 开发板上的硬件资源大致可分为几大模块:微控制器最小系统模块、电源模块、调试模块以及 I/O 接口模块等。模块之间在电气和逻辑关系上相互联系和支持,共同构成一个完整的开发环境。其中,电源模块从外部接入电源并转换成稳定的工作电压,为开发板上的所有电气元件供能;调试模块转换 USB 接口信号,方便我们建立起电脑端与开发板之间的通信连接;微控制器最小系统模块是开发板的“大脑”,我们编写并下载的程序就在这一位置运行。Arduino Uno 开发板的组成如图 1.17 所示。



图 1.17 Arduino Uno
开发板的组成

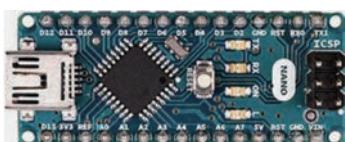


图 1.18 Arduino Nano 开发板

(2) 其他 Arduino 开发板

Arduino Nano 是 Arduino Uno 的微型版本,与后者最大的不同是没有电源插座,并且 USB 接口采用 Mini-B 型插座。Nano 开发板尺寸很小,只有 45 mm × 18 mm,可以直接插在面包板上使用,是专为空间有限的应用而设计的开发板。其外形如图 1.18 所示。



图 1.19 Arduino Mega 2560 开发板

Arduino Mega 2560(如图 1.19 所示)是 Arduino 系列中采用 8 位处理器芯片的最高配开发板,包含了单片机运行所需的所有要素。虽然其体积比前两个大(101.52 mm × 53.3 mm),但它的闪存空间大,I/O 引脚数多,通信接口多,适用于物联网项目等。

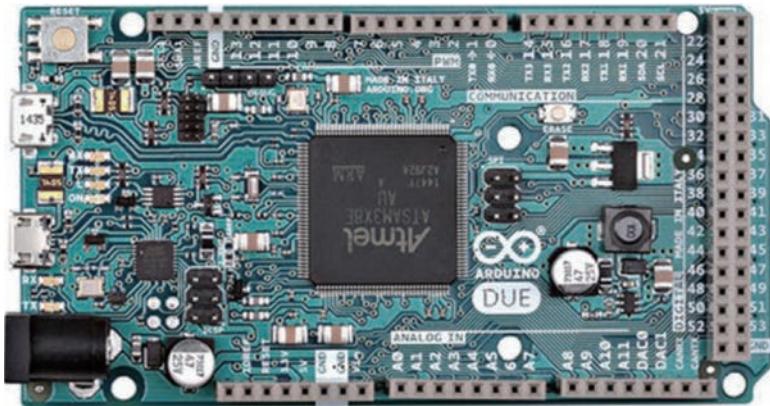


图 1.20 Arduino Due 开发板

Arduino Due(如图 1.20 所示)是基于 32 位 ARM Cortex-M3 处理器的第一个 Arduino 开发板,包含了支持微控制器所需的全部要素,只需通过 Micro USB 电缆将其连至计算机,或通过 AC-DC 适配器、电池为其供电,即可使其工作。Arduino Due 的大小与 Arduino Mega 2560 相同,但与其他 Arduino 开发板不同的是它的工作电压为 3.3 V。

3. 树莓派开发平台简介

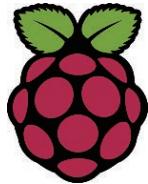


图 1.21 树莓派标识

树莓派是由英国的几位开发者历经 6 年时间开发,并于 2012 年正式发布的一款基于 ARM 架构的开源硬件开发平台。开发者最初的动机是为了提升学校计算机编程教育水平,让计算机变得有趣,以更多地激起人们对计算机科学的兴趣。因此命名时采用了大家喜欢的水果名称——“树莓”,“派”则是对通用编程语言“Python”的简称,最终的命名即为二者的组合“树莓派”。图 1.21 是树莓派的标识。

树莓派是一款侧重于计算性能的卡片式计算机,预装 Linux 操作系统,采用 ARM 架构处理器,配备较全面的接口,包括可供键盘、鼠标使用的 USB 接口,可与显示器或者电视机相连的高清多媒体接口 (high-definition multimedia interface, 缩写为 HDMI),以及快速以太网接口、SD 卡扩展接口、数字 GPIO 接口等常用接口,支持 Java、C、Python 等开发语言,虽然其外形仅有一张信用卡那么大,但它具备了一般个人电脑的基本功能,相当于一台微型化的低成本电脑。树莓派开发板的外形如图 1.22 所示。树莓派的主要特点有:

(1) 功能强大。树莓派自带多种多样的接口和强大的功能,不仅可以连接开关、传感器、电动机或微处理器等元器件,直接实现底层硬件的 I/O 控制,还能在较低的功耗水平下,像一般 PC 机那样稳定地

图 1.22 树莓派开发板

运行完整版的 Linux 操作系统,为用户提供开发更上层应用产品的可能。

(2) 硬件结构模块化。基于树莓派的所有扩展功能都有成品模块可供选用,用户只需像搭积木一样将模块插入相应的接口即可,这就大大降低了对用户的电子和硬件知识的要求,适合复杂信息处理系统的设计。

(3) 资源共享。树莓派拥有完善的社区支持,可提供完备的树莓派开发文档,为树莓派的系统迭代优化提供支持。

4. 典型的树莓派开发板及其硬件结构

树莓派是一个非常小巧的通用计算机,运算性能和智能手机相仿。树莓派如果连接上必要的外围设备(如电视机和键盘),就成了一台名副其实的低功耗通用电脑了,你能用它来做在一台普通计算机上所能做的事情。随着树莓派与物联网技术的结合,树莓派更是被广泛地应用于工业控制、机器人、智能家居、创客教育等领域。图 1.23 是树莓派外接设备的连接图。

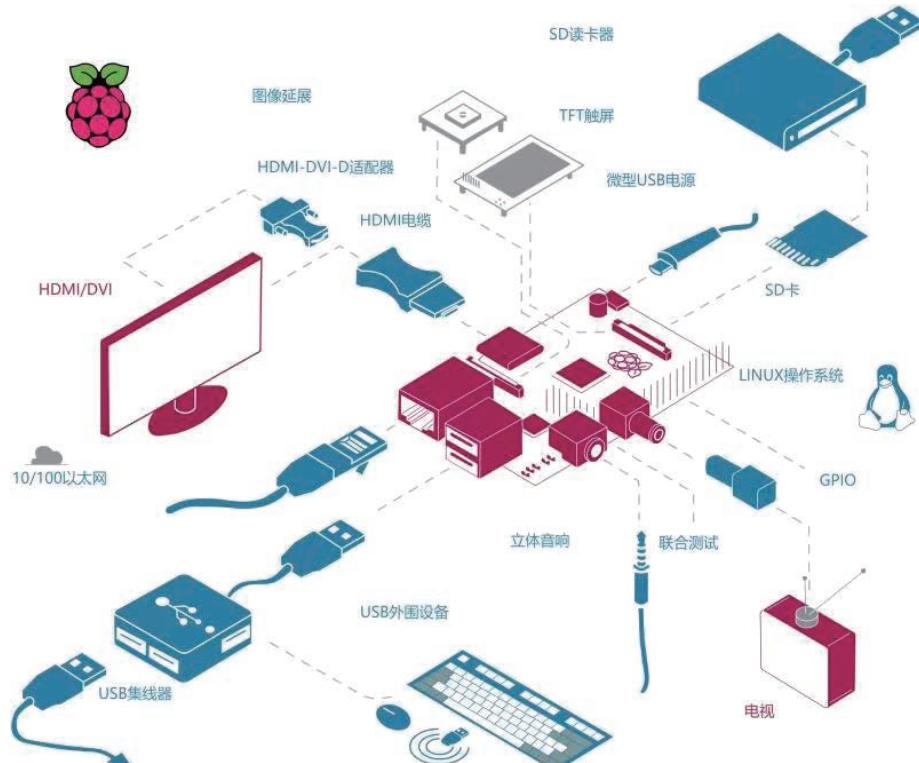


图 1.23 树莓派外接设备连接图

树莓派开源平台提供了多种型号的开发板,它们尽管型号不同,但基本硬件组成大致相同。以树莓派 3B+ 为例,树莓派 3B+ 开发板采用 64 位 ARM Cortex-A53 处理器,以 Micro SD 卡为存储模块,卡片周围含有多种多样的 I/O 接口模块,其中包括 USB 接口、以太网接口、模拟视频信号输出接口和高清多媒体接口等。

在图 1.24 所示的树莓派 3B+ 开发板中,可以看到其主要的硬件模块和它们所在的位置:

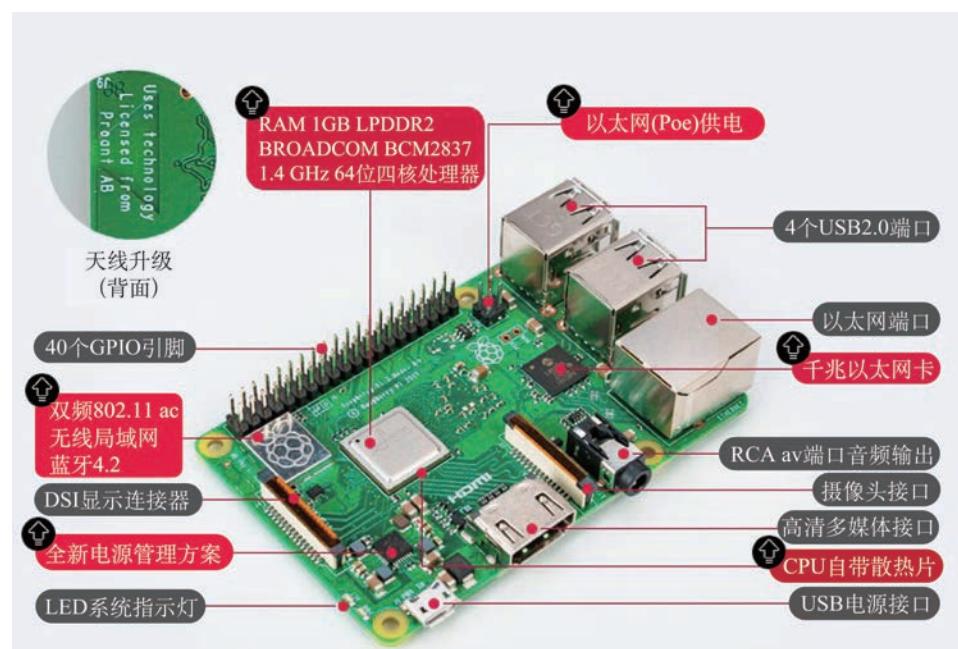


图 1.24 树莓派 3B+ 开发板

(1) 树莓派 3B+ 开发板的处理器芯片临近中心区域,它是由集成电路构成的 SoC 处理器模块,相当于开发板的“大脑”,具有信息处理、图形渲染以及输入/输出控制等功能。

(2) SoC 处理器模块下方的银色连接器是一个高清多媒体接口(媒体播放器和电视机顶盒上都有相同类型的接口),用高清多媒体接口连接显示器或电视机,可以提供高清分辨率的视频以及数字音频信号。

(3) 树莓派 3B+ 开发板左上方的两排插针是 GPIO 接口,这些接口最常见的用处是连接扩展板电路,通过它们可以将树莓派和其他硬件设备连接起来。

(4) 树莓派 3B+ 开发板的左下角是一个 Micro USB 电源接口,和目前的智能手机的接口一样,用一根 Micro USB 数据线将树莓派连接到合适的电源适配器上,树莓派就可以启动了。

(5) 树莓派 3B+ 开发板的左边底侧有一个 Micro SD 卡槽,可以

插入 SD 存储卡。一个安全性高的存储卡不但可以为操作系统、程序、数据和其他文件提供存储空间，而且不易使数据丢失。

5. Arduino 与树莓派的比较

Arduino 与树莓派是创客进行发明创造的两大利器，Arduino 和树莓派看起来十分相似，都是小型电路板，但实际上它们的区别很大。

Arduino 实质上是一个微控制器，由于微控制器只是计算机的一部分，功能有限，所以尽管 Arduino 可以通过一些小型的类 C 应用程序进行编程，但无法运行一个操作系统。树莓派则是一台完整的迷你计算机。它的核心是一款 ARM 微处理器，以 SD/Micro SD 卡作为内存和硬盘空间，外接键盘、鼠标、显示器等，可以加载 Linux 操作系统，运行复杂的应用程序。

树莓派侧重于计算性能，它只有数字 I/O 接口，缺乏模拟输入接口，而 Arduino 更侧重于 I/O 性能，不但包含数字 I/O 接口，还包含不少模拟输入接口，可以连接大量的数字传感器和模拟传感器，擅长处理模拟输入信号。

在许多方面结合使用树莓派和 Arduino，可以充分发挥两块电路板的优势。如果使用带有 GPIO 接口并含有类似于 AlaMode 这样兼容 Arduino 硬件的接口板，就可以将树莓派和 Arduino 结合起来，实现功能和性能的最优化。

作业练习

- 收集信息，从开源的内容和功能特点等角度，对比 Arduino 和树莓派，并给出它们的一些成功案例，完成表 1.3。

表 1.3 Arduino 与树莓派的对比

| 硬件平台 | 是否开源 | 功能特点 | 成功案例 |
|---------|------|------|------|
| Arduino | | | |
| 树莓派 | | | |

2. 上网搜集典型开源硬件平台的常用开发板信息,将它们的特性填入表 1.4。

表 1.4 常用的开源硬件开发板的特性

| 硬件平台 | 开发板名称 | 处理器位数 | 主频 | I/O 接口类型 | 开发语言 | 操作系统 | 特点及应用场合 |
|---------|-------|-------|--------|----------|--------|------|---------------------------|
| Arduino | Uno | 8 | 16 MHz | 数字和模拟 | 类 C 语言 | 不支持 | 最基础的开发板, 适用对性能要求不高的普通应用场合 |
| | Nano | | | | | | |
| | Due | | | | | | |
| 树莓派 | | | | | | | |
| 虚谷号 | | | | | | | |
| 掌控板 | | | | | | | |

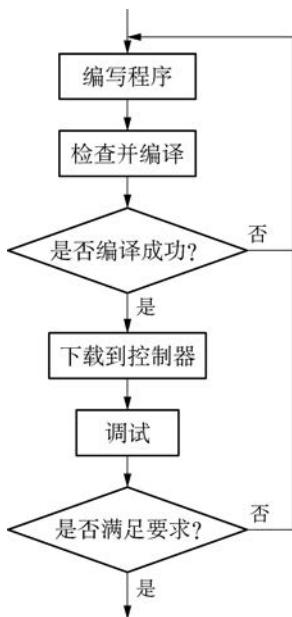


图 1.25 集成开发环境下的软件开发过程

三、集成开发环境(IDE)

要使计算机正常工作,仅仅有硬件系统是不够的,还需要软件系统的支撑。软件是计算机信息处理系统能够运行的基础,也提供了用户和硬件平台交互的方式。在进行项目开发前,首先需要搭建基于硬件平台的软件开发环境。

集成开发环境(integrated development environment, 缩写为 IDE)是提供程序开发环境的应用软件,它集成了程序代码编写、检查、编译、下载、调试等一系列功能,可以将编写、编译好的程序下载到控制器运行,并提供一定的调试手段,帮助我们测试程序,找错纠错。集成开发环境将用户在开发过程中所需要的工具或功能集成到一起,提供一体化服务,能够最大化地提高开发者的工作效率。集成开发环境下的软件开发过程如图 1.25 所示。

项 目 实 践

【项目器材】

电脑、Arduino 开发板、USB 数据线。

【操作步骤】

- 安装 Arduino IDE。可访问 Arduino 官方网站下载 Arduino IDE 软件。安装完成后的 Arduino IDE 的编程环境如图 1.26 所示。

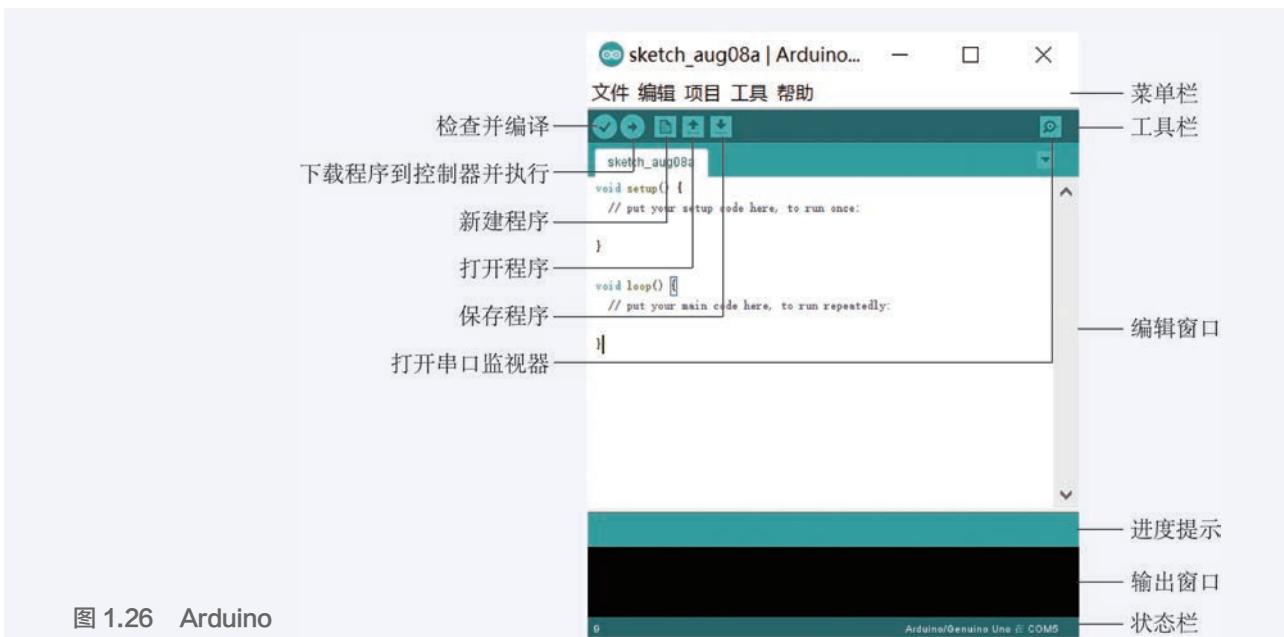


图 1.26 Arduino IDE 编程环境

2. 安装 Arduino 驱动。用 USB 数据线将 Arduino 开发板与电脑连接起来,第一次连接的时候,需要安装相应的驱动。驱动程序位于 Arduino 软件安装目录下的 Drivers 文件夹中。驱动程序安装完成之后,打开电脑的“设备管理器”,会发现在“端口”项中增加了一个 COM 端口设备,请记下该端口号。如图 1.27 所示,所用计算机通信端口号为 COM5。

请同学们将自己所用的开发板型号和其在电脑上所使用的串口号,填入表 1.5。

表 1.5 开发板型号及其串口号

| 开发板型号 | 串口号 |
|-------|-----|
| | |

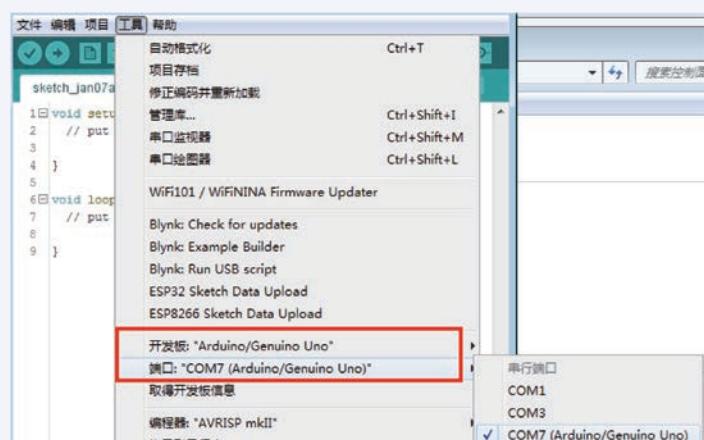


图 1.28 设置“开发板”和“端口”

4. 用 Arduino IDE 打开 Arduino 自带的 Blink 程序,如图 1.29 所示。

5. 点击工具栏中的  按钮将程序下载到 Arduino 微控制器自带的闪存,观察 Arduino 板上 LED 的闪烁。

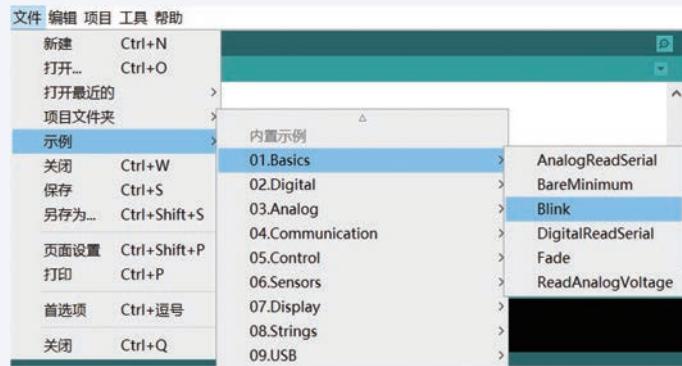


图 1.29 搜索 Blink 程序

技术支持

图形化编程

Arduino IDE 的默认编程环境支持源代码的文本编程方式,我们还可以通过在 Arduino IDE 中添加图形化的插件来搭建图形化编程环境。

1. ArduBlock 插件

ArduBlock 是一款可以配合 Arduino IDE 使用的第三方软件,它以图形化的方式来编制比较简单的程序。使用时不需要了解具体的编程语言的写法,只要将需要实现的功能以图形的方式告知编译系统,用类似于“搭积木”的“拖拉”方式,将左边相应的功能模块、控制模块、元件模块放入中间的编程窗口中,通过功能堆叠来编写程序,比较适合初学者。

将 ArduBlock 压缩包中的 libraries 和 tools 文件夹复制到 Arduino 的安装目录下,然后重新启动 Arduino 程序,在“工具”菜单中即可看到“ArduBlock”选项,如图 1.30 所示。点击“ArduBlock”,即可进入图形化编程环境。

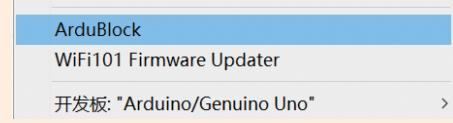


图 1.30 菜单栏内的“ArduBlock”选项

2. 米思齐(Mixly)编程环境

与 Arduino IDE 一样,米思齐也是一款免费开源的硬件编程软件,同时支持文本和图形化编程方式。米思齐基于 Blockly 和 Java8,支持 Arduino 和 micro: bit 硬件开源平台。它将图形化编程方式和文本编程方式融合在一起,用不同的颜色代表不同类型的功能块,用户只需直接拖动模块就可以编写出简单的程序。

项目实践

【项目器材】

电脑、Arduino 开发板、USB 数据线。

【操作步骤】

1. 用 USB 数据线连接好计算机和 Arduino 开发板,并设置好 Arduino IDE 中的“开发板”和“端口”。

2. 从开源平台网站上下载“Blink.abp”图形化程序文件，并用ArduBlock打开，同学们也可以自己完成图形化程序设计，以实现Arduino开发板板载LED(如图 1.31 所示)的闪烁(亮 1 s, 暗 1 s)。

3. 点击工具栏中的  按钮，将程序下载到 Arduino 微控制器并运行，观察 Arduino 开发板板载 LED 的实验效果。同时观察 Arduino IDE 界面上自动生成的文本代码。

注意：在 ArduBlock 中点击  按钮时，搭建好的图形化程序会在 Arduino IDE 中自动生成 C++ 文本代码，并自动完成编译，然后将编译后的目标代码存入到 Arduino 微控制器自带的闪存，并开始运行。

【流程示意】

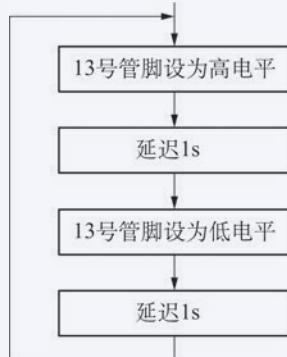
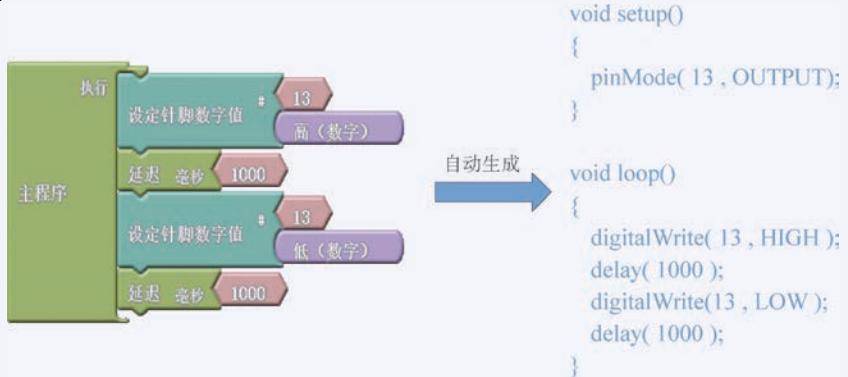


图 1.32 闪烁 LED 程序流程示意图

【关键代码】



【代码说明】

1. Arduino 程序的构成

Arduino 程序包括两部分，即初始化程序和主体程序。

(1) 初始化程序: void setup(){初始化程序代码}

初始化程序的代码在{}内完成，主要实现引脚的输入/输出方式、串行通信速率、变量初值等的设置。这部分在整个程序运行过程中只执行一次。

(2) 主体程序: void loop(){主体程序代码}



图 1.31 Arduino 开发板板载 LED

主体程序的代码在{}内完成,它包括执行程序时的各个过程、控制、输入/输出等。这部分在整个程序运行过程中可无限次地重复执行,直至断电或关机。

2. 设定数字引脚

可以用 digitalWrite()函数给一个数字引脚写入 HIGH 或者 LOW,即高电平或低电平。对 Arduino 开发板板载的 LED 而言,高电平可以点亮 LED,而低电平则熄灭 LED。

3. 设置延迟值

可以用 delay()函数设置程序延迟的时间(单位:ms, 1 s= 1 000 ms)。

作业练习

如果要求 LED 快速闪烁,如亮 0.1 s,暗 0.1 s,应如何修改程序?如果要求亮的时间长于暗的时间,那又要如何修改程序?

六、程序的调试

程序在运行过程中可能会出现运行结果不正确或失效等问题,这就需要在实际硬件环境中作进一步的调试。调试其实就是针对软件运行中出现的问题,通过现象找出原因的过程。

项 目 实 践

对于运行在开发板上的程序,由于缺少像台式机那样可以进行交互的外部设备,程序的调试变得较为困难。在上一节的两个活动中,我们是直接通过观察开发板板载 LED 的运行效果来检查程序是否正确运行的,但有时直接观察实验现象很容易产生误差,不能够精确定位出错位置。那么有没有办法利用我们的电脑显示器,把开发板上程序运行过程中的数据同步显示出来,让我们通过监控程序的运行来检测错误所在呢?

【项目器材】

电脑、Arduino 开发板、USB 数据线。

【操作步骤】

1. 用 USB 数据线连接好计算机和 Arduino 开发板。
2. 在 ArduBlock 中打开上一个活动中完成的程序“Blink.abp”,在原来程序的每个过程后面加上串口输出,以显示运行中的程序状态。其中 LED 亮时屏幕上显示“LED ON”,暗时显示“LED OFF”。
3. 点击工具栏中的  按钮,将程序下载到 Arduino 微控制器并运行,然后点击工具栏中的  按钮,打开串口监视器。仔细观察 Arduino 开发板板载 LED 的亮和灭,同时观察串口监视器上同步打印的字符。
4. 在 ArduBlock 中点击“另存为”按钮,将程序另存为“Blink Monitor 1”。

5. 在程序“Blink Monitor 1”上修改,删除显示延迟状态的输出语句,并将同步显示的字符由“LED ON”改为“-”,“LED OFF”改为“_”。
6. 点击工具栏中的 按钮,将程序下载到 Arduino 微控制器并运行,然后点击工具栏中的 按钮,打开串口监视器。
7. 在 ArduBlock 中点击“另存为”按钮,将程序另存为“Blink Monitor 2”。
8. 因为 ArduBlock 图形化编程命令比文本编程更少,控制不够多样,下面在 Arduino IDE 中对自动生成的文本程序进行修改,将两行打印回车语句“Serial.println();”删除,如下所示:

| 原代码 | 修改后代码 |
|--|--|
| <pre>void setup() { Serial.begin(9600); pinMode(13, OUTPUT); } void loop() { digitalWrite(13, HIGH); Serial.print(" - "); Serial.println(); delay(1000); digitalWrite(13, LOW); Serial.print(" _ "); Serial.println(); delay(1000); }</pre> | <pre>void setup() { Serial.begin(9600); pinMode(13, OUTPUT); } void loop() { digitalWrite(13, HIGH); Serial.print(" - "); delay(1000); digitalWrite(13, LOW); Serial.print(" _ "); delay(1000); }</pre> |

9. 然后点击工具栏中的 按钮将程序下载到 Arduino 微控制器。再点击 按钮,打开串口监视器(如图 1.33 所示)。仔细观察 LED 的亮和灭,同时观察串口监视器同步输出的字符。此时可看到代表低电平和高电平的简单方波信号(如图 1.34 所示)。



图 1.33 LED 亮灭的串口监视器

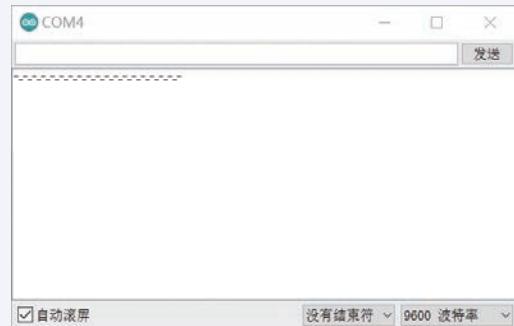


图 1.34 串口监视器显示的简单方波信号

【关键代码】

“Blink Monitor 1”程序如下：



“Blink Monitor 2”程序如下：



【代码说明】

1. 串口初始化设置函数 `Serial.begin(9600)` 将串口传输率设定为 9 600 bps。
2. 通过串口打印加回车，即 `Serial.println()` 函数，将数据打印到串口输出。注意：`Serial.println()` 函数打印输出数据后自动换行，而 `Serial.print()` 函数打印输出数据后不换行。`Serial.print()` 函数在 ArduBlock 中没有对应的图形化语句，所以在“Blink Monitor 2”程序生成对应文本程序后要对其进行适当修改。

作业练习

1. 和同学讨论一下，“Blink Monitor 1”和“Blink Monitor 2”程序的输出各有什么优点？

2. “Blink Monitor 1”程序中：

(1) 语句 `串口打印加回车 [LED ON]` 能否放到 `设定针脚数字值 # 13 高 (数字)` 之后？

(2) 语句 `串口打印加回车 [Delay of 1 second]` 能否放到 `延迟 毫秒 1000` 之后？

3. 在调试中，我们是需要监控程序的运行，还是只需要监控关键步骤、关键变量和关键数据的变化？

同学们有没有留心观察过现在的手机？当有未接来电或未查阅的短信时，手机上的指示灯会由暗到亮逐渐变化，就像人的呼吸一样，这是 LED 在微处理器控制下完成的显示效果，我们把这样显示的 LED 叫作呼吸灯。

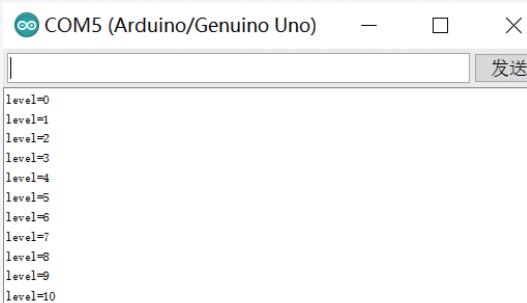
在前面的实例当中，我们都是用 Arduino 微控制器来控制 LED 的两种状态：亮或灭。那我们能否使 LED 慢慢变亮，再逐渐变暗，实现连续变化的呼吸灯效果呢？

【项目器材】

电脑、Arduino 开发板、万用表、示波器。

【操作步骤】

1. 连接好计算机和 Arduino 开发板。
2. 从开源平台网站上下载“fade.abp”图形化程序文件，并用 ArduBlock 打开，亦可自己完成图形化程序设计，实现 Arduino 开发板板载的 LED 由暗逐渐变亮，每 3 s 改变一次亮度，串口同步显示相应提示。
3. 点击工具栏中的  按钮，将程序下载到 Arduino 微控制器并运行，然后点击工具栏中的  按钮，打开串口监视器。在 Arduino 开发板板载 LED 亮度变化的同时，仔细观察串口监视器同步输出的字符（如图 1.35 所示）。



```
level=0
level=1
level=2
level=3
level=4
level=5
level=6
level=7
level=8
level=9
level=10
```

图 1.35 串口监视器字符

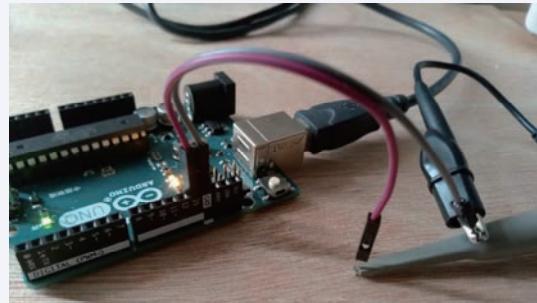


图 1.36 示波器接法

4. 小组合作，按图 1.36 所示用万用表检测 GND 和 13 号引脚的电压，注意红表笔接 13 号引脚，黑表笔接 GND。当串口监视器输出时，记录此时的电压值（保留一位小数），填入表 1.6。

表 1.6 记录万用表显示的电压值

| level | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|-------|---|---|---|---|---|---|---|---|---|----|
| 电压值 | | | | | | | | | | |

5. 用示波器的接地探针连接 GND，信号脚接 13 号引脚，按下示波器的“Autoset”按钮，示波器会自动调整设置（通道垂直档位、时基以及触发方式等），使波形显示达到最佳效果。观察如图 1.37 所示的波形，并在表 1.7 中记录波形下方的占空比（保留一位小数），占空比即一个周期内高电平宽度占周期的比例。

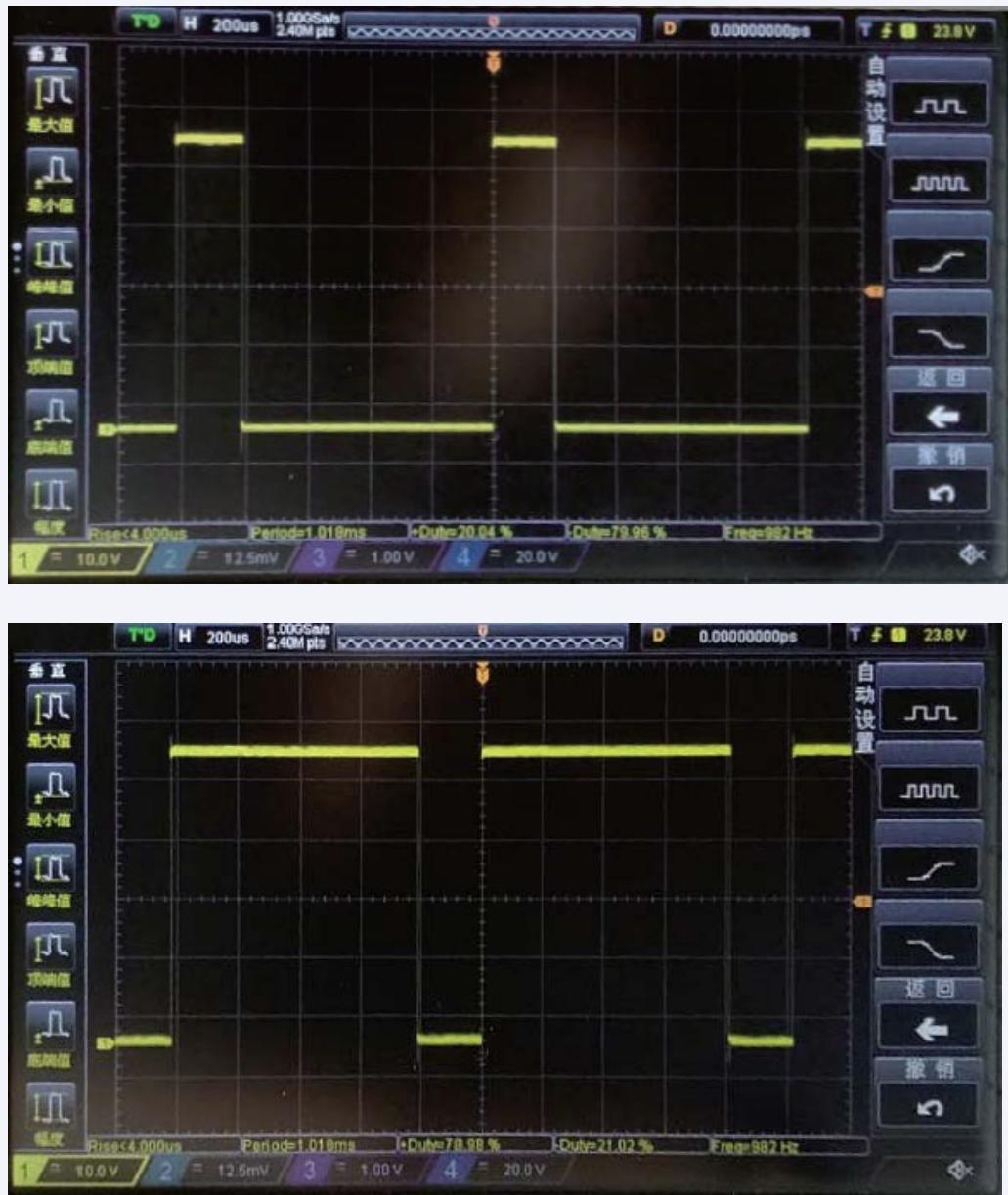


图 1.37 示波器显示的数据

表 1.7 记录示波器显示的占空比

| level | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|-------|---|---|---|---|---|---|---|---|---|----|
| 占空比 | | | | | | | | | | |

【流程示意】

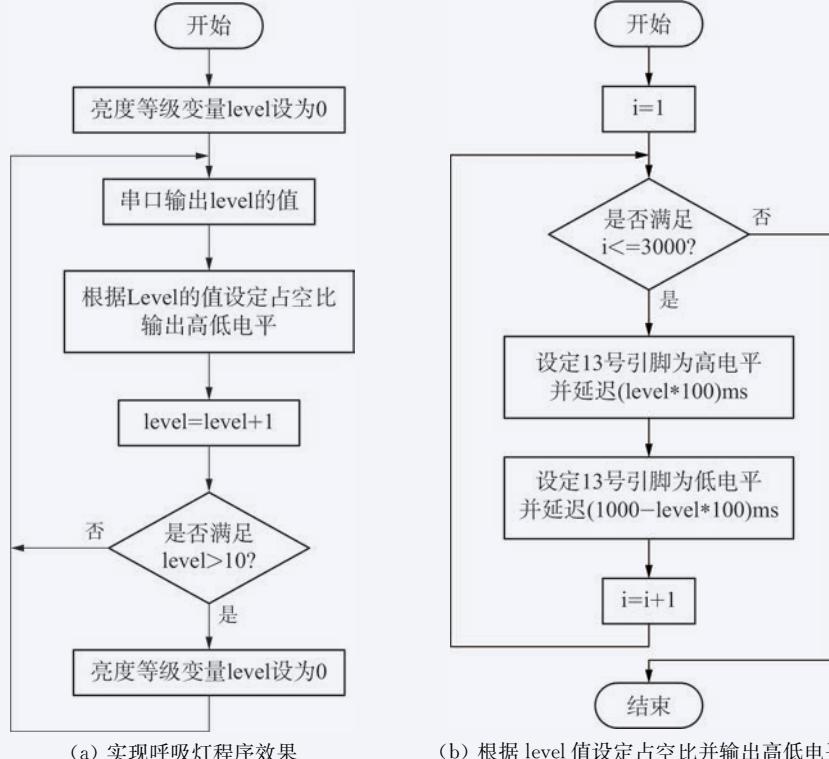
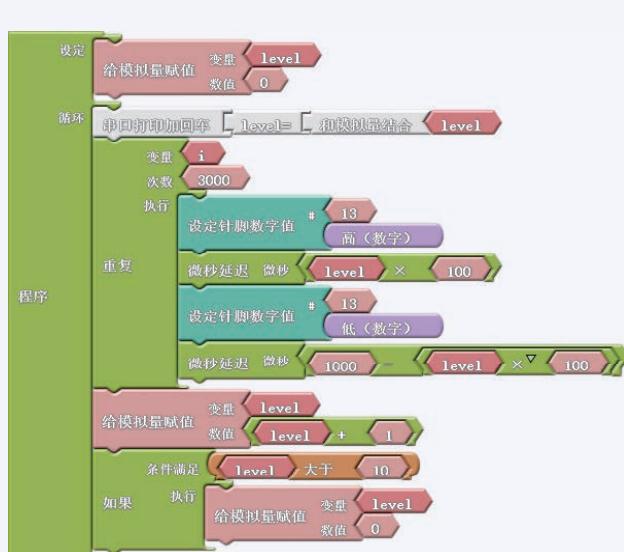


图 1.38 呼吸灯程序流程示意图

【关键代码】



```

int level = 0;

void setup(){
    Serial.begin(9600);
    pinMode(13, OUTPUT);
}

void loop(){
    Serial.print("level = ");
    Serial.println(level);
    for (int i = 0; i < 3000; i++) {
        digitalWrite(13, HIGH );
        delayMicroseconds( ( level * 100 ) );
        digitalWrite(13, LOW );
        delayMicroseconds( 1000 - level * 100 );
    }
    level++;
    if (level > 10) { level = 0 ;}
}

```

作业练习

- 观察程序中设定的占空比和 LED 亮暗变化的关系，并回答：程序中的变量 level 有什么作用？它的上限值 10 能够增加或减少吗？如果改变其上限值，实验现象会发生什么变化？
- 程序中的变量 i 有什么作用？它的上限值 3 000 能够增加或减少吗？如果改变其上限值，实验现象又会是什么变化？
- 这个活动中，为了能够正确地用万用表和示波器观察实验效果，我们设计的呼吸灯会跳跃式地逐步变亮，显示效果不是最佳。如果要实现更好的呼吸灯效果，应该如何修改程序？

知识延伸

示波器的使用

1. 示波器面板及按钮

各种型号的示波器面板式样及其按钮分布大相径庭，但基本功能是相近的。示波器按钮说明如图 1.39 所示。

2. 连接探针

探针样式如图 1.40 和 1.41 所示，旋紧连接在示波器上的信号输入端，用夹子接地，同时用测量头接触测量点，波形就会显示在示波器屏幕上。探针通常有 10 倍信号衰减或无衰减两种可调档位。

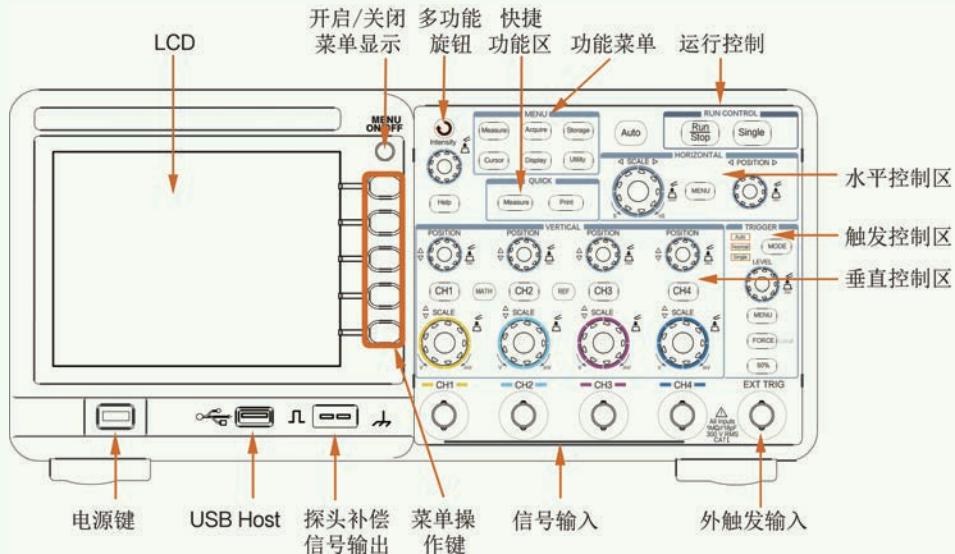


图 1.39 示波器面板按钮说明

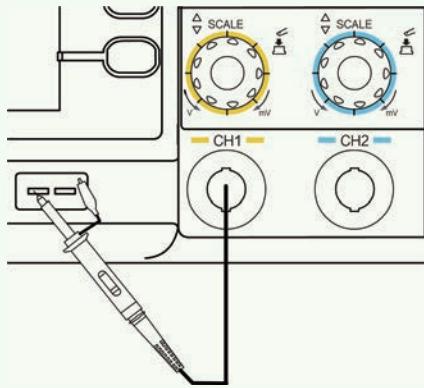


图 1.40 探头连接

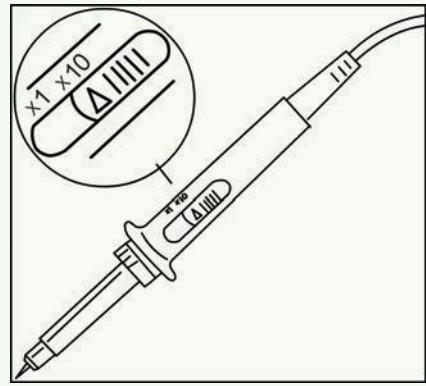


图 1.41 探头调节

测量时接地夹子要与被测电路的接地端相连,否则会造成测量不准确,甚至损坏设备。

3. 自动定标

按下“Auto Scale”键,可将示波器自动配置为对输入信号显示最佳效果。如果要使示波器恢复到以前的设置,可再次按下,取消自动定标。

4. 停止与重新扫描

示波器的功能就是显示不同时刻的信号波形,随着信号的不断输入,波形不断地显示在屏幕上,由于屏幕大小有限,波形看上去随时间在移动,不利于观测和记录参数数据。

“Run/Stop”键呈绿色,表示示波器正在运行,即示波器符合触发条件,正在采集数据。若要停止采集数据,可再次按下“Run/Stop”键,使波形固定。

“Run/Stop”键呈红色,表示数据采集已停止。若要开始采集数据,可再次按下“Run/Stop”键,使波形重新移动。

第三节 开源硬件项目设计方法

开源硬件项目的系统设计是一个反复迭代、逐步求精的过程。这个迭代过程一般包括以下五个步骤：需求分析、方案设计、作品制作、调试优化、作品发布。

体验思考

在进行创新实践时，需要理解项目要求，考虑如何充分利用开源硬件的特点，确定项目实施计划和方案，使项目能够顺利有序地进行，让设计的系统更好地满足相应需求。

通过智能水杯的开发实践，体验开源硬件项目的开发流程，并探讨：如何设计方案？如何选择开源硬件器材？设计制作完成的开源项目如何发布？

开源硬件项目的系统设计过程如图 1.42 所示。本节将以智能水杯设计为例，介绍开源硬件项目系统设计的各个步骤。

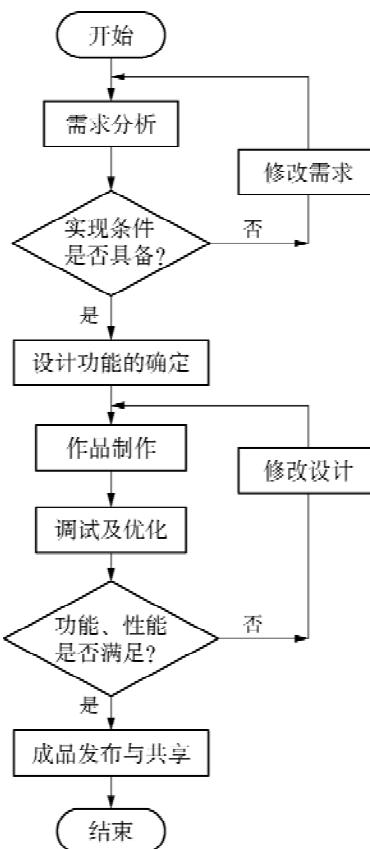


图 1.42 开源硬件项目系统设计的主要步骤

随着社会智能化程度的提高,人们对生活用品的智能化要求也不断提升。以水杯为例,传统的水杯没有显示、提醒功能,我们希望智能水杯具有显示水温和水量的功能,方便我们把握喝水的情况,同时,我们希望智能水杯能根据个人的生理状况和喜好,提供喝水提醒、缺水提醒等功能,使我们更科学、更合理地饮水。除此之外,智能水杯还应满足每个人对水杯外观、形状、装饰等个性化的审美需求。

一、需求分析

对项目进行需求分析有助于明确项目目标,可将需求分为核心需求和附加需求两类,核心需求是必须满足的,在满足核心需求的前提下,结合作品设计方案,利用有限资源尽可能地去实现更多附加需求。注意这两类需求可能会转换,尤其是在作品迭代时会有调整或增减。

探究活动

在发布开源硬件的网站和社区上有很多其他人分享的作品,它们是获得灵感、创新想法、完善作品、交流经验的重要渠道,善加利用可以帮助我们更有效地开发优秀作品。

【活动步骤】

1. 上网查阅智能水杯的相关资料,了解现有的或类似的产品,再通过用户调查、头脑风暴等方法,理解智能水杯项目的需求(如表 1.8 所示),提出项目的设计目标,并对项目的需求和功能进行细化。
2. 通过一定的方式,如网站、论坛、专家咨询、小组讨论等,获取相关的开源资料,判断和评估项目各功能实现的技术难点,明确在现有的技术、资料、资金、时间等限制条件下完成项目设计的可能性。
3. 从作品功能完整性、用户体验、成本因素等多方面对需求进行评估,不断调整需求内容直至通过评估。

表 1.8 智能水杯的需求分析

| | | |
|------|---------------|----|
| 核心需求 | 显示水温和剩余水量 | 采纳 |
| | 定时喝水提醒 | 采纳 |
| | 缺水提醒 | 采纳 |
| 附加需求 | 使用悦耳的音乐提醒 | 采纳 |
| | 定时的时间可以设置 | 采纳 |
| | 能远程查看水温、水量等状态 | 忽略 |
| | 外观设计体现个性化 | 采纳 |

二、方案设计

根据需求分析的结果论证和设计可行的系统实现方案,一般包括信息处理系统设计方案和功能结构设计方案两个部分。设计过程中应及时沟通设计信息并作交叉确认,使信息处理系统设计与功能结构设计相匹配,如果出现了二者不匹配的情况,就需要修改设计直到满足所有的匹配要求。

探究活动

智能水杯的方案设计可细分为两大部分:水杯的信息处理系统设计与水杯的功能结构设计。

【活动步骤】

1. 水杯的信息处理系统设计

项目设计中的信息处理系统一般也被称为电子控制系统。根据需求分析明确产品的主要功能,分析各个功能在信息输入、信息处理、信息输出等各模块中的对应关系,明确各功能模块之间信息通信的方式,确定需要使用的主要硬件器材,包括微控制器、传感器、执行器等。

(1) 功能描述

通电启动智能水杯,进行初始化设置,包括饮水的时间和缺水警示阈值等,将设定参数存入存储器。根据设定值启用硬件定时器,实现定时提醒喝水。

运行过程中用传感器获取水温、剩余水量等信息,实时监测剩余水量,并传送给微控制器;微控制器接收到传感器的信息,经处理后通过 OLED 显示屏将信息显示出来;当水杯内水量偏低时,执行器报警提醒加水。

(2) 器件选择

根据智能水杯的功能,选择相应器件。智能水杯功能与器件表如表 1.9 所示。

表 1.9 智能水杯功能与器件表

| 编号 | 功能 | 描述 | 对应模块 | 对应器件 |
|----|-------------|------------------|------|------------------|
| 1 | 测温 | 测量水的温度 | 输入模块 | 温度传感器 |
| 2 | 测剩余水量 | 测量水杯中剩余的水量 | 输入模块 | 压力或水位传感器 |
| 3 | 喝水提醒 | 指定时间提醒喝水 | 输出模块 | 定时器、蜂鸣器 |
| 4 | 时间和缺水警示阈值设定 | 设定喝水的时间和缺水警示阈值 | 输入模块 | 按键、OLED 显示屏、存储器 |
| 5 | 缺水提醒 | 水杯内水量过少时,提醒及时补充 | 输出模块 | 传感器、蜂鸣器 |
| 6 | 远程显示 | 向监护人远程显示测量信息 | 输出模块 | 蓝牙、Wi-Fi 等无线通信模块 |
| 7 | 水杯运行控制 | 控制水杯的正常运行,整合系统功能 | 处理模块 | 微控制器(含存储器、定时器) |

(3) 系统设计框图

电子控制系统的设计框图如图 1.43 所示：

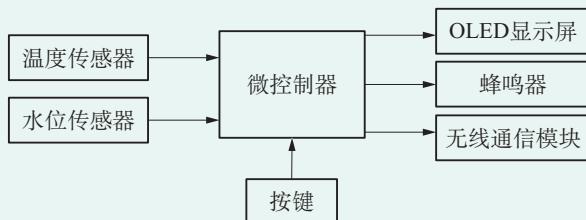


图 1.43 智能水杯电子控制系统设计框图

2. 水杯的功能结构设计

功能设计主要考虑作品在工作时操控方面的情况, 结构设计主要考虑作品的机械结构组成。

(1) 功能设计

在功能设计方面, 制定如表 1.10 所示的操作方案。

表 1.10 按键控制功能表

| 编号 | 功能键名称 | 功能说明 | 备注 |
|------|-------|-----------------|------------|
| 按键 1 | 测温 | 显示水杯中水的温度 | 快捷操作 |
| 按键 2 | 测剩余水量 | 显示水杯中剩余水量 | 快捷操作 |
| 按键 3 | 选定/退出 | 设定喝水时间、剩余水量警示阈值 | 用短按/长按操作菜单 |

(2) 结构设计

将结构设计细分为杯盖、杯体、杯底等部分, 杯体上嵌有 OLED 显示屏、按键等, 杯底安置控制板、电池和传感器模块等电子控制系统。同学们可以发挥创意, 制定智能水杯个性化的外观及装饰方案。

体验思考

以小组的形式开展以下活动:

1. 收集和比较常见智能水杯的功能和特点, 设计小组的智能水杯功能表。
2. 在开源社区中, 寻找智能水杯作品, 在分析该作品的基础上, 评估小组设计的智能水杯方案的可行性。
3. 思考在作品设计中应如何选用开源模块和闭源模块开展作品设计, 讨论开源文化对创新的意义。

三、作品制作

开源硬件项目的作品制作包括电子控制系统制作和结构造型制作两个任务。电子控制系统制作要完成硬件开发和软件固件开发两个部分,即根据功能设计选择合适的硬件器件,并选择恰当的开发环境编写程序;结构造型制作是指安装电子控制系统的机械结构,完成作品外观的制作。

探究活动

作品制作可以从搜索开源社区上的资源开始,选择一个类似作品,在他人经验和成果的基础上,进行修改和完善,使之符合自己的需求,快速完成应用系统的设计,有效提高自己的工作效率。

【活动步骤】

1. 电子控制系统制作

电子控制系统制作要确定控制器和外围器件,选出符合方案设计要求的器件的具体型号,在选定合适的开发环境和软件支持程序库后完成软件固件部分的开发。

(1) 控制器及开发环境选择

一般而言,要优先选择有开源社区支持、软件开发环境成熟的方案,这样可以在遇到问题时得到技术支持,并且尽可能避免其他因素的干扰,如果遇到开发问题也可以讨论咨询。表 1.11 给出了智能水杯控制器选择方案。

表 1.11 智能水杯控制器选择方案

| 控制器类型 | Arduino Uno | Arduino Nano | 树莓派 3B+ |
|-----------------|-------------|--------------|---------|
| 主控板尺寸是否满足要求 | 否 | 是 | 否 |
| 运算能力是否满足要求 | 是 | 是 | 是 |
| 存储容量是否满足要求 | 是 | 是 | 是 |
| 接口类型和引脚数量是否满足要求 | 是 | 是 | 是 |
| 是否支持电池供电 | 是 | 是 | 否 |
| 是否有开源社区 | 有 | 有 | 有 |
| 是否有良好的软件开发环境支持 | 有 | 有 | 有 |
| 结论 | 体积过大 | 适合 | 功耗太高 |

(2) 外围器件及驱动程序库选择

依据项目需求提出的每个器件分类,综合考虑性能、价格等因素,优先选取有高质量开源软件程序库支持的器件,具体确定每个分类模块或部件的器件型号。具体到智能水杯,其硬件配置如表 1.12 所示。

表 1.12 智能水杯硬件配置

| 编号 | 模块 | 关键参数 | 器件型号 | 硬件开源 | 开源软件库数目 | 综合评价 |
|----|----------|---|--------------------------|------|---------|------|
| 1 | 温度传感器 | 供电电压:3.3~5V 测量范围:0~100°C 测量精度: $\pm 0.5^{\circ}\text{C}$ (-10~85°C) | DS18B20 数字温度传感器 | 是 | 有 | 确定 |
| 2 | 压力传感器 | 压力感应范围:20~6 000 g 触发力:20g(默认电阻值小于 200 kΩ 时触发) | RP-C18.3-LT 电阻式压力传感器 | 是 | 有 | 确定 |
| 3 | 水位传感器 | 感应厚度(灵敏度)范围:0~13 mm 工作环境温度:0~105°C 液位检测精度: $\pm 0.5 \text{ mm}$ | XKC-Y25-T12V 智能非接触式液位传感器 | 是 | 有 | 确定 |
| 4 | 按键 | 工作电压:3.3~5V | 通用 | 是 | 有 | 确定 |
| 5 | OLED 显示屏 | 像素个数:128 列×64 行 工作电压:3.3~5V | SSD1306 OLED 液晶显示模块 | 是 | 2 个以上 | 确定 |
| 6 | 蜂鸣器 | 工作电压:5V | 通用 | 是 | 有 | 确定 |

(3) 硬件搭建

以开源社区中的类似方案设计为参考,根据图 1.43 所示智能水杯的设计原理,完成顶层框图和电路图设计、仿真、印刷板制作(依需要)及焊接等步骤,完成硬件搭建,并用开发环境完成固件流程设计、代码编写和测试。

2. 结构造型制作

依据在功能结构设计环节提出的要求,参照在器件选型环节确定的器件尺寸和系统框图环节确定的连接方法,明确成型作品的几何尺寸限制。特别需要注意的是,作品的组装过程需与电子模块安装流程配合,同时还要考虑维修的方便性。智能水杯的结构设计如图 1.44 所示。



图 1.44 个性化的智能水杯

3. 作品组装

将作品的电子控制系统模块和结构造型模块,按照设计的要求进行组装,检查完整性。

作业练习

在详细设计过程中,如果找不到一款完全合适的开源硬件开发板或驱动程序,是否意味着设计无法进行?请思考如何充分利用开源社区提供的开放资源提高创新设计的能力。

四、调试优化

完成作品各模块制作、集成后,还必须进行功能的测试与调试,优化设计方案并完善作品。

测试与调试要根据所测内容设计合适的流程,配置相应的环境,帮助确定并隔离出现问题的模块,查明原因并落实修正措施。

探究活动

在完成硬件模块搭建和软件编程后,要按照要求对每项功能进行测试,测试的目的是检验作品在功能、结构和性能等方面能否达到预定的要求。若发现错误,则修改相应软件和硬件,直到测试无误。在测

试的基础上,还要对设计方案和作品进行较为全面的评估,并依据评估结果,对系统进行优化。调试与优化主要包括:借助调试环境和测试工具,观察测试系统运行过程中的现象,发现错误及时纠正;调整硬件模块,提高信息检测与显示的准确性和可靠性;设计和优化作品的设计方案,增强作品的功能,提升系统的性能以及使用体验的友好性。

【活动步骤】

1. 功能测试

逐一测试智能水杯的功能,以列表形式(如表 1.13 所示)将测试时的现象描述清楚,如果存在功能错误,则需判断原因并加以修改。必要时可以重复上述的设计过程,直到所有功能都满足,并完成功能测试表。

表 1.13 智能水杯功能测试表

| 编号 | 功 能 | 现象描述 | 错误原因判断 |
|----|-------------|-----------|--------|
| 1 | 温度测量和显示 | 能正常显示水温 | 无错误 |
| 2 | 剩余水量显示和警示 | 能正常显示剩余水量 | 无错误 |
| 3 | 喝水提醒 | 能定时播放音乐 | 无错误 |
| 4 | 时间和缺水警示阈值设定 | 能正常设置 | 无错误 |

2. 作品优化

通过了测试的作品可进行演示,收集反馈信息后修改需求表,开始迭代优化,提高作品质量。

作业练习

- 在调试中发生功能错误,一般是采用什么样的方法定位错误发生的原因?如果水杯无法显示水温,应该先检查什么?再检查什么?
- 万用表和示波器在查找、定位错误时都是常用的仪器,两者在具体使用时又有哪些不同?

五、项目发布与共享

开源的意义在于,它一方面给了开发者一个较高的起点,使得他们能够从低价值的重复工作中解放出来,站在前人的肩膀上开展高价值的创新活动;另一方面,开源也给了发布者分享资源的舞台,使得发布者的想法和创意能够迅速传播。

开源硬件项目设计的最后一个重要的步骤是项目的发布与共享,这

既是我们为开源做贡献的一种方式,也是从开源获取回报的一个途径。因为通过开源,我们可以巩固现有技能、遇见“志同道合”的伙伴、寻找导师或者尝试帮助他人、学习领导和管理的艺术、探索更多创意,等等。为开源贡献力量,得到的回报就是能够学到更多、受教更多,并且可以从中锻炼很多能力。

探究活动

智能水杯设计制作完成后,可以作为开源项目设计的案例在某个开源社区上发布。发布前,先要了解开源硬件项目发布需要注意的有关事项,选择一个开源社区,学习开源硬件项目发布与共享的方法。

【活动步骤】

1. 注册项目账号。在开源社区网站注册项目专用的账号,并创建项目,上传软件代码和设计文档。
2. 合理地组织代码和文档。在分享源代码给公众之前,一方面需要认真考虑代码的组织方式,包括文件目录结构、代码风格等,另一方面要制作文档,给出明确的开发指南。这样,其他人就能方便地获取源文件,轻松地读懂源代码,全面地了解项目的设计思路,进而为项目的设计提供新的建议,由此真正达到资源共享、迭代创新的目的。
3. 选择开源许可证并发布项目。在开源硬件项目的设计正式发布之前,还有一件很重要的事情是选择一个合适的开源许可证。选好后应遵循开源许可证规则发布相关的硬件设计和软件代码,实现资源共享。
4. 宣传和维护。做好项目的宣传,让更多人参与到你的项目中。同时定期登录社区,查看相关的留言和建议,及时更新以完善项目的实施。

作业练习

1. 归纳总结开源硬件项目的设计过程,说明在开发开源项目的各个阶段分别需要提交哪些技术资料以及如何对项目进行评估。
2. 为开源做贡献并不意味着总要提交代码,也可以通过设计界面、编写文档、组织活动、检查他人代码等方法来为开源做贡献。举例说明你能为开源贡献什么。



第二章

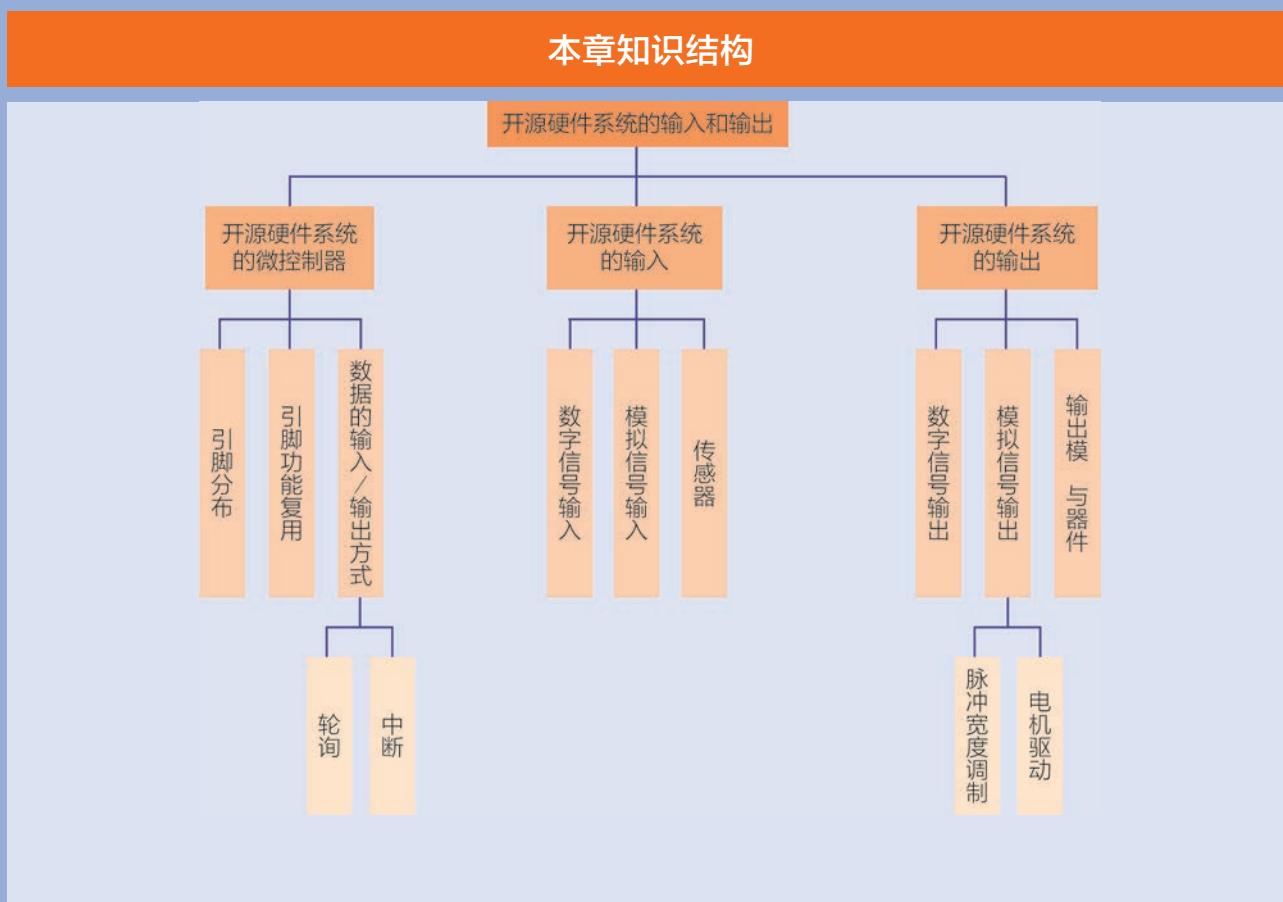
开源硬件系统的输入和输出

本章学习目标

-
- 通过制作开源硬件系统的输入和输出部分,理解开源硬件系统的基本组成,掌握输入与输出的多种实现方式。
 - 能根据项目需求分析设计开源硬件产品的开发方案,知道产品各组成部分的功能及相互间的调用关系。
 - 能根据设计方案和开源硬件使用说明选择合适的开源硬件,会测试与优化产品设计方案。
 - 能根据设计方案搭建产品的各种功能模块。
 - 通过设计、制作机器伙伴,经历发现问题与解决问题的过程,体验开源硬件项目的开发流程。
-

一个完整的信息处理系统,通常都会有输入与输出部分,例如,日常生活中的计算机主机,一般都连接了输入/输出设备,如打印机、键盘和鼠标等,有些设备只有输入功能,如键盘和鼠标,有些设备只有输出功能,如打印机。开源硬件系统有多种输入/输出方式和输入/输出设备可以灵活选择与搭配。传感器可以将环境中的声、光、电、磁、温度、湿度等转化为电信号,计算机对这些输入的电信号进行处理后,执行器响应处理结果,完成特定任务。灵活多样的传感器和执行器组合方式,构成了开源硬件系统的输入/输出设备,能够满足人们的不同需求。

在本章中同学们将学习开源硬件系统的输入与输出,灵活地调用与创造不同的输入与输出组合,动手制作简单的输入/输出产品。在设计与制作产品的过程中,同学们需要进一步思考:如何设计产品方案?如何选择合适的输入/输出方式?



项 · 目 · 情 · 境

如今,随着中国加快推进科技自立自强,信息技术飞速发展,越来越多的机器装置正在以一种类似“机器人”的形态与我们的学习、工作、生活、娱乐紧密关联,“机器人”一词最早出现在20世纪初的一本科幻小说中,但机器人并非是在外形上与人类相似的机器装置,事实上科学家们认为“机器人是一种自动化的机器,能够依靠自身的动力和控制能力完成某种任务”。

机器人就像人类的“伙伴”一样为人类提供特定的服务,它获取输入数据,执行特定的输出,复杂的机器装置能代替人进行部分体力和脑力劳动,提高人民生活品质。在学习和生活中,我们常遇到一些小烦恼,比如:夜间需要开灯时,发现开关离得太远;学习时,发现自己始终无法集中注意力,学习效率不高;晚上沿路跑步时,路上的车辆太多,存在一定的安全隐患。那么,怎样利用开源硬件设计机器伙伴帮助我们解决这些实际问题呢?机器伙伴是如何获得输入信息,如何处理信息以及如何完成特定输出的呢?

本章我们将在分析上述问题的基础上,选择合适的开源硬件器材,设计制作自己的机器伙伴,编写程序并进行调试,最后在小组内展示、分享自己的作品。

项 · 目 · 任 · 务

任务 1

设计一个感应机器伙伴,使之能够感应到你的存在,并在特定情境下作出反应。

任务 2

设计一个定时提醒机器伙伴,使之能够自动地帮助我们更好地规划时间。

任务 3

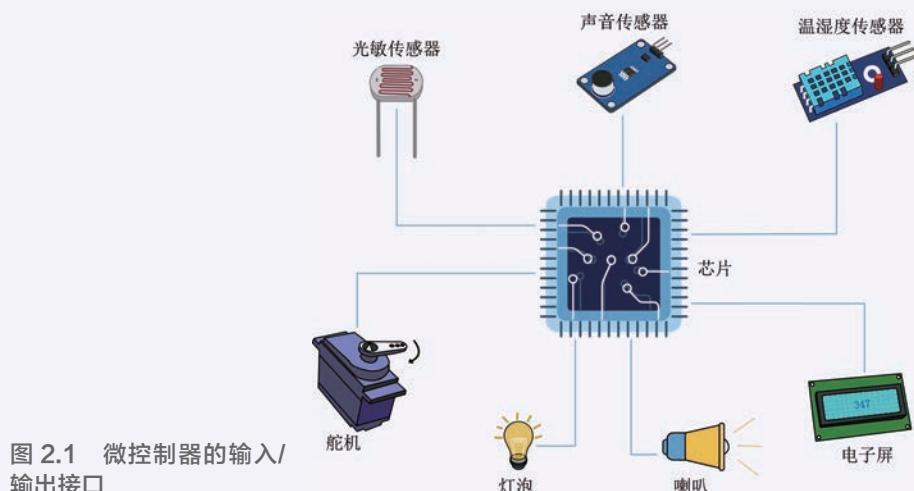
设计一个安全机器伙伴,使之能够发出个性化的醒目灯光,在夜晚提醒过往车辆。

第一节 开源硬件系统的微控制器

计算机的输入设备接收到的信息包括**数字量**和**模拟量**两大类,在开源硬件系统里,微控制器接收输入设备的信息进行处理,然后发送给输出设备,这些信息也有数字量和模拟量之分。

体 验 思 考

微控制器是如何处理数字量和模拟量的呢?图 2.1 展示的是微控制器的输入/输出接口,它是如何分辨出要从哪个接口读取数据,又是从哪个接口把数据发送出去的呢?

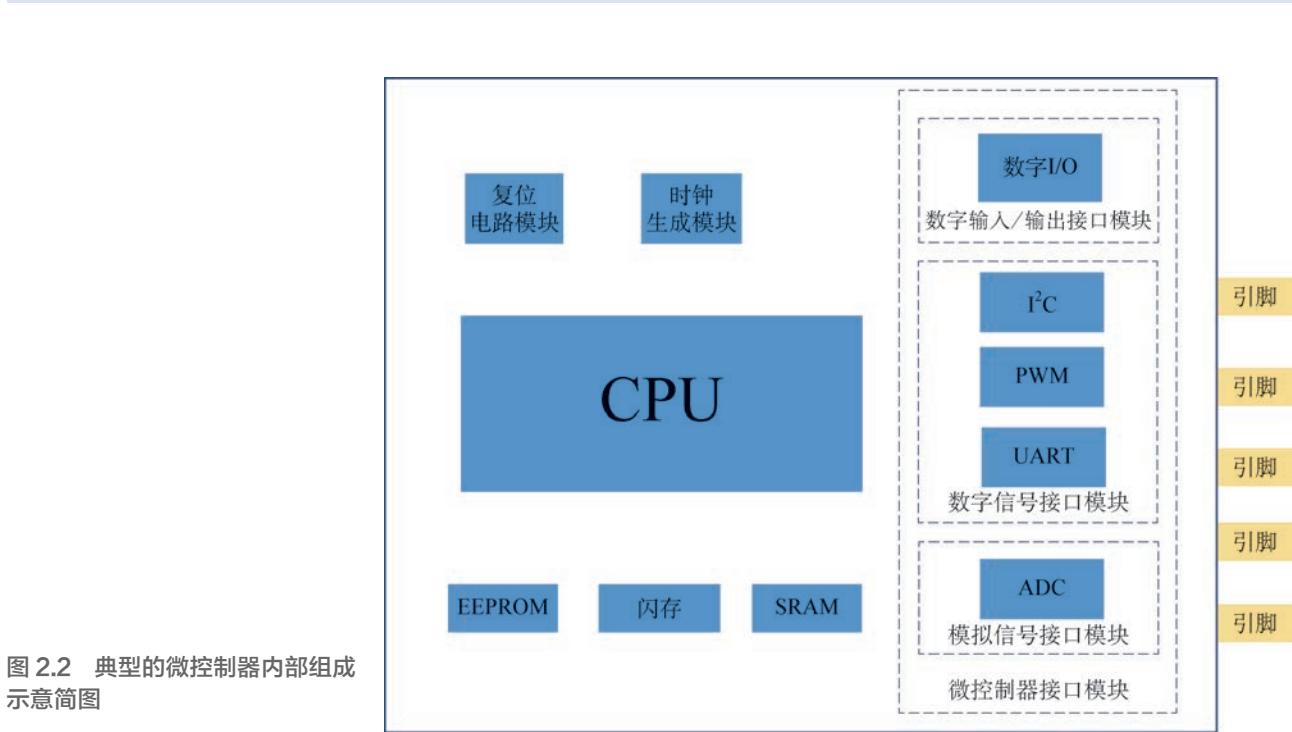


一、微控制器的组成

微控制器是通过芯片上的引脚与外部的输入/输出设备连接的。随着科技的快速发展,微控制器集成的功能模块增多,其功能也日益强大。与此同时,电子设备小型化对微控制器的封装尺寸提出了严格要求,芯片能够放置的引脚数目受到限制,其内部的所有模块一般无法同时连接到芯片引脚上。

以 Arduino Uno 开发板使用的微控制器为例来说明,该芯片内部组成如图 2.2 所示,可以看到微控制器芯片有多种接口模块。

组成个人计算机主板的必备器件,比如中央处理器、内存、输入/



输出接口等都可以在微控制器中找到。图 2.2 中,时钟生成模块产生时钟控制信号以控制系统运行;复位电路模块可产生内部复位信号以控制系统从最初始的状态开始运行;闪存保存程序固件代码;SRAM 用于系统运行时存储信息;数字 I/O 模块提供基本的输入/输出功能;芯片的边沿是引脚。

微控制器为了更好地适应不同应用场合的需要,还增加了一些常用的专用模块来增强系统的功能,如图 2.2 中的 UART 接口模块等。这些接口功能模块在使用时一般都需要通过引脚连接到外部的对应设备才能够发挥作用。为方便用户开发和使用,Arduino Uno 开发板将微控制器的大部分引脚连接到了开发板的接插件上。

二、微控制器的引脚

Arduino Uno 开发板的引脚分布如图 2.3 所示。Arduino Uno 开发板除了用黑色标记的电源、芯片复位等特殊的引脚之外,还有 20 个可用于通用输入/输出(GPIO)的引脚,它们被分为 3 个 GPIO 端口,分别是 PORTB、PORTC 和 PORTD,一般简写为 PB、PC 和 PD。因为微控制器芯片引脚数目的限制,大部分引脚无法做到单独应用于一个特定功能,所以很多数字引脚还同时连接到了内部的多个接口功能模块上。这些数字引脚也可以作为内部功能模块(如 UART、I²C、SPI 等)引脚、ADC 输入引脚或者 PWM 输出引脚使用。因此引脚的功能

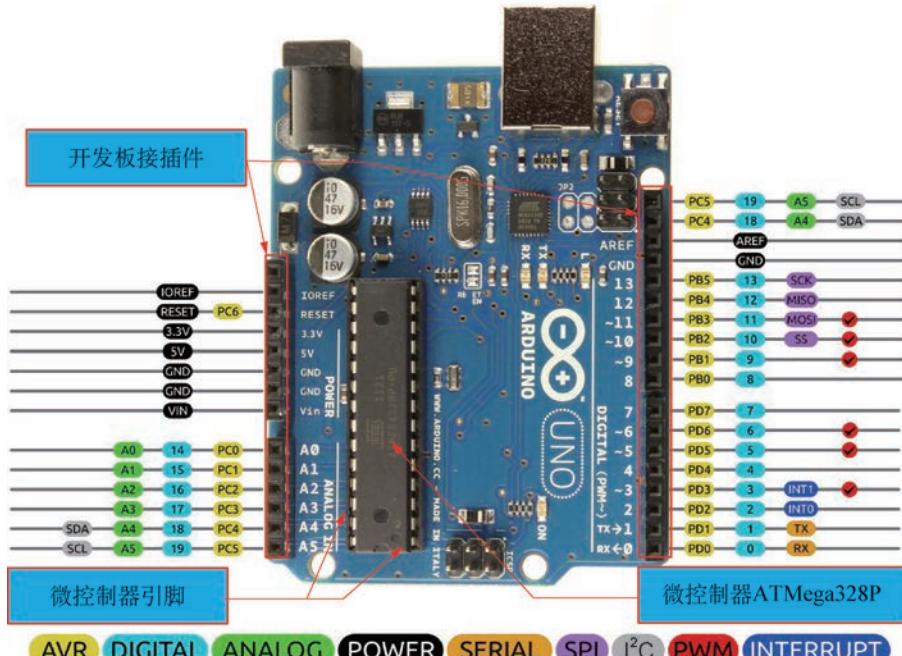


图 2.3 Arduino Uno 引脚和引脚复用功能示意图

都不是单一的,而是根据固件程序运行时对该引脚的配置,确定该引脚连接到哪个特定的功能模块或者是否作为普通的数字引脚使用,这样的设计被称为“引脚功能复用”。其特点是,具有复用功能的引脚在同一时间只能体现一种功能,比如引脚 A4 和 A5 如果被设置为用于 I²C 的通信,那么它们就无法同时被用于模拟输入。

引脚的复用功能可以在图 2.3 中查询。以 Arduino Uno 开发板上蓝色 14 号数字引脚为例,该引脚还连接有代表模拟输入功能的绿色“A0”标签和代表通用输入/输出功能的黄色“PC0”标签。据此可以了解,该引脚既可以作为数字输入/输出接口,又能连接内部的模数转换模块输入通道 ADC0。Arduino Uno 开发板的 3 号、5 号、6 号、9 号、10 号、11 号数字引脚除了具有数字输入和数字输出功能,还具有模拟输出(PWM 输出)的功能,在其引脚编号旁边有相应的“~”标志。

微控制器运行时复用引脚使用的实际功能是通过软件编程体现引脚配置的方式来确定的。示例如下:

(1) 实现数字输入配置

```
pinMode(14, INPUT);      //设定 14 号引脚为输入模式
value = digitalRead(14); //读取 14 号引脚信号电平
```

(2) 实现模拟输入配置

```
//模拟引脚不需要使用pinMode()函数,将其指定为输入或者输出模式  
value = analogRead(A0); //采集14号引脚上的模拟信号
```

(3) 实现数字输出配置

```
pinMode(14, OUTPUT); //设定14号引脚为输出模式  
digitalWrite(14, HIGH); //设定14号引脚输出高电平数字信号  
digitalWrite(14, LOW); //设定14号引脚输出低电平数字信号
```

(4) 实现模拟输出(PWM输出)配置

```
analogWrite(11, 255); //设定11号引脚为PWM(模拟)输出,取值范围0~255
```

三、数据的输入/输出方式

微控制器通过执行输入和输出相关的指令来完成数据的输入和输出。执行数据输入的条件是：外部设备的数据准备好提供到微控制器并且微控制器准备好接收。对应地，执行数据输出的条件是：外部设备准备好接收数据并且微控制器准备好发送数据。为有效完成数据的输入和输出，必须在对应的条件满足时才执行相应的输入和输出指令。可以采用轮询和中断这两种方法来完成微控制器的输入和输出。

轮询是让微控制器以一定的周期按顺序查询每一个外部设备，看它们是否有数据输入/输出的要求，若有，则进行相应的输入/输出服务，若无，就接着查询下一个外部设备。其明显的缺点是CPU利用率低，实时性比较差。比如，同学们在学校自修教室里上晚自习，并计划每学习一个小时休息十分钟，因此会每隔一段时间去查看手表，看看一个小时到了没有，如果没到则继续自习，如果到了就停止自习，在自修教室内开始休息十分钟。这种不断查看手表的方式对于自习的效率有很大的影响。此例中休息十分钟就好比是完成了一个输入/输出的服务，而每隔一段时间查看手表就好比是轮询。

中断则是指微控制器在正常运行程序的过程中，由于预先安排或突发随机事件，中断正在运行的程序，转到为应对此事件而专门设计的中断服务程序，中断服务程序的调用是由CPU来调度自动完成的。其优点是CPU利用率高，实时性好。仍以上晚自习为例，假设同学们

设置了每隔一个小时响一次的定时闹钟,听到闹钟响后,暂停自习并记录所学内容所在的页码,然后离开自修教室到操场上休息十分钟,休息完毕回到自修教室再从标记的页码开始继续自习。这样的方式较之前一种方式,自习的效率有所提高。此处,“在自修教室上晚自习”这个事件就相当于主程序,“闹钟响起”可认为是中断请求,而“每隔一个小时离开自修教室到操场上休息十分钟”就好比中断响应,“标记的页码”可认为是断点,“结束休息后回到自修教室继续从标记的页码开始继续学习”就好比中断返回,继续执行主程序。中断流程示意图如图 2.4 所示。

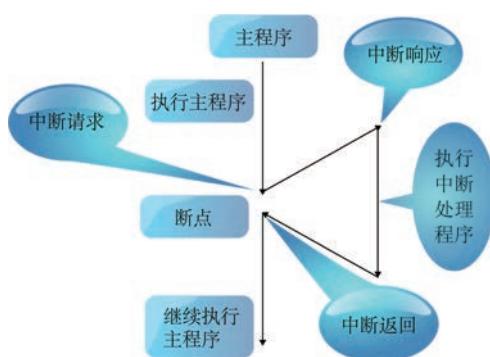


图 2.4 中断流程示意图

以下是一段程序,程序的功能是通过定时器中断,使得 13 号引脚的 LED 以 500 ms 的间隔闪烁。

```
#include<TimerOne.h>      //调用定时器扩展库 TimerOne 头文件
void setup()
{
    pinMode(13, OUTPUT);          //设定 13 号引脚为输出模式
    //初始化定时器,每 500ms 调用一次定时器中断服务函数
    Timer1.initialize(500000);
    //设定 reFresh() 函数为定时器中断子程序
    Timer1.attachInterrupt(reFresh);
}
void loop()  {  }
void reFresh() //中断服务程序
{
    //读取 13 号引脚的电压,并取反输出
    digitalWrite(13, !digitalRead(13));
}
```

其中,loop()函数是主程序,reFresh()函数是中断服务程序。程序运行后,首先执行主程序,这段主程序里没有任何代码,即什么也不执行,直到微控制器接收到定时器传来的中断请求,主程序暂停,生成一个断点,保存当前的主程序运行状态,转而响应中断,执行中断服务程序,待中断服务程序执行完毕,返回至主程序的断点处,恢复到先前的主程序运行状态,继续执行主程序,直到接收到下一个中断请求。

作业练习

查阅资料,比较常见的开源硬件开发板的数字和模拟引脚数量,填写表 2.1。

表 2.1 常见的开源硬件开发板的数字引脚与模拟引脚

| 硬件平台 | 开发板 | 数字引脚数 | 模拟引脚数 |
|---------|------|-------|-------|
| Arduino | Uno | | |
| | Nano | | |
| | Due | | |
| 树莓派 | | | |
| 虚谷号 | / | | |
| 掌控板 | / | | |

第二节 开源硬件系统的输入

开源硬件系统的输入可以有多种形式,如人机交互的鼠标、键盘、触摸屏等,也可以使用能自动采集数据的传感器。传感器可以感应周围的环境变化,可用来测量物理量,并将数据返回给系统处理,以便系统根据数据作出相应的响应。

体 验 思 考

人有五官,并且通过听觉、视觉、嗅觉、味觉、触觉来感知世界。如何让机器也能够尽可能地感知这个世界呢?我们可以为机器打造“五官”吗?

一、数字信号输入

1. 按键

按键是常见的数字输入型电子元件,按下时闭合,处于导通状态。按键交互一直以来是人机交互的常用方式。从机器主体的角度考虑,按键可以作为机器与外界交互或与人沟通的渠道(如图 2.5 所示)。比如生活中的各种家用电器,按下开关时,通常都有一个电源指示灯亮起,表明其处于工作状态。图 2.6 给出了几个常用的按键开关。



图 2.5 人机交互
中的按键

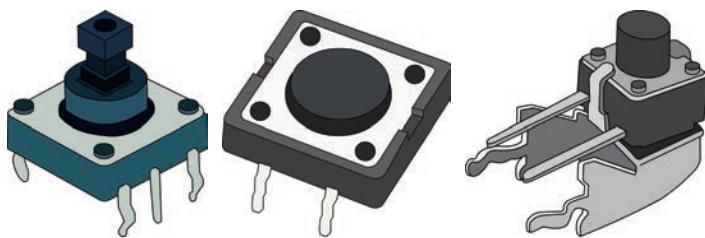


图 2.6 常用的按键开关

如果将按键或其他的开关型元件作为输入模块,需要将其状态的改变转化为相应电压的变化,才能使其被开源硬件系统识别。Arduino 开发板的数字引脚在没有外接电子元件时,其相应电压的状态是不确定的。可以用一个上拉电阻或下拉电阻把 Arduino 开发板引脚的状态确定下来。其中上拉电阻的一个引脚连接到正向电压 VCC,而下拉电阻的一个引脚直接接地。

上拉电阻的工作原理可参考图 2.7。当开关 S 断开时,根据欧姆定律, $V_{out} \approx V_{CC}$,即 V_{out} 得到高电平;当 S 闭合时,GND 和 V_{out} 短接, $V_{out} \approx 0$,即 V_{out} 得到低电平。下拉电阻的工作原理与上拉电阻的相似,如图 2.8 所示。

上拉电阻和下拉电阻不仅可以将开关状态转变为能被 Arduino 开发板识别的高、低电平,还可以防止在 S 闭合时 VCC 和 GND 出现短路。

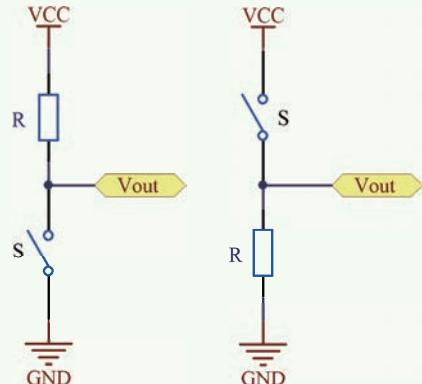


图 2.7 上拉电阻连线

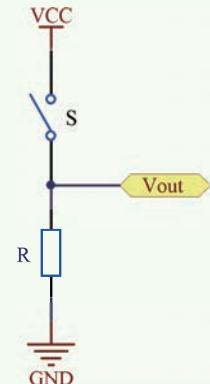


图 2.8 下拉电阻连线

2. 传感器与感官

人需要借助感觉器官来感知外界的各种信息,而我们的开源硬件系统可以利用传感器来感知外界信息。

传感器有多种分类,按照用途和应用场景来分,有开关传感器、温湿度传感器、超声波传感器、光敏传感器、压力传感器、声音传感器、加速度传感器、红外传感器、角度传感器、重力传感器、GPS 传感器等;按数据输入方式又可大体分为数字式传感器和模拟式传感器,数字式传感器能直接输入数字信号,模拟式传感器能采集外界连续变换的物理信号,它往往需要相应的连接引脚配备 ADC。

声音传感器能获取什么信息?查阅资料,试着在表 2.2 中填写与人的感官对应的传感器,以及它所能获取的信息。

表 2.2 人的五官感觉、超五官感觉与对应的传感器的比较

| 人的感觉 | 对应的传感器 | 传感器获取的信息 |
|------|--------|----------|
| 听觉 | 声音传感器 | 外界声音的强度 |
| 视觉 | | |

(续 表)

| 人的感觉 | 对应的传感器 | 传感器获取的信息 |
|----------|--------|----------|
| 触觉 | | |
| 味觉 | | |
| 嗅觉 | | |
| 非五官感觉 I | 红外线传感器 | |
| 非五官感觉 II | 陀螺仪传感器 | |

项 目 实 践

机器的开和关,可以通过按键来实现,按键是机器最常用的控制元件,LED 是机器最常用的状态显示元件。将按键用作机器的开关,通过手动控制 LED 的亮灭来显示机器的工作状态。

【项目器材】

Arduino Uno、按键开关、红色 M5 直插 LED、220 Ω 电阻、10 k Ω 电阻、面包板、杜邦线。

【系统框图】

机器开关系统框图如图 2.9 所示。



图 2.9 机器开关系统框图

【操作步骤】

1. 按照图 2.10 连接电路,将按键通过面包板接到 Arduino Uno 开发板数字 8 号引脚,红色 LED 接到数字 12 号引脚。(也可以用其他引脚,但最好不选用 0 号和 1 号引脚,因为 0 号和 1 号引脚同时也是串口通信接口,这属于引脚功能的复用。)

2. 编写程序,实现当按键按下时 LED 亮起。当按键按下时,数字 8 号引脚输入为高电平,控制数字 12 号引脚输出高电平,LED 亮起;当按键未按下时,数字 8 号引脚输入为低电平,控制数字 12 号引脚输出为低电平,LED 不亮。

【核心程序】

```
int ledpin = 12; // 定义 LED 接口为 12 号引脚
int inpin = 8; // 定义按键输入为 8 号引脚
int val; // 定义变量 val
void setup() {
```

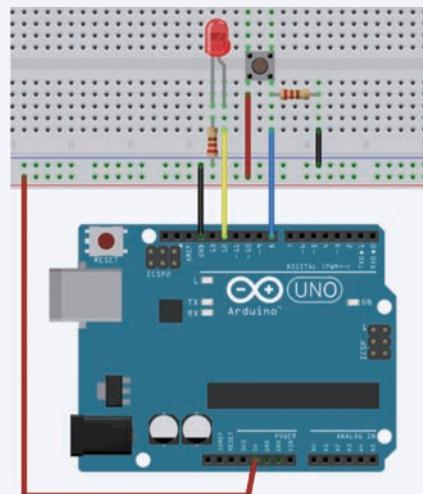


图 2.10 机器开关实物连接图

```

pinMode(ledpin, OUTPUT); // 定义 LED 接口为输出接口
pinMode(inpin, INPUT); // 定义按键接口为输入接口
}

void loop() {
    val = digitalRead(inpin); // 读取 8 号引脚电平值，并将其赋给 val
    if(val == LOW){ // 检测按键是否按下，按键按下时 LED 亮起
        digitalWrite(ledpin, LOW); // 给 12 号引脚低电平
    }
    else{
        digitalWrite(ledpin, HIGH); // 给 12 号引脚高电平
    }
}

```

体 验 思 考

- 能否将 loop() 中的程序代码用 digitalWrite(ledpin, digitalRead(inpin)); 来代替？
- 如要实现当按键按下再弹起后 LED 在点亮和熄灭两种状态之间切换，程序应如何设计？

技 术 支 持

面包板的使用

面包板(如图 2.11 所示)是一种通用插座板，各种电子元器件可根据需要插入或拔出，不需要进行焊接，这样可以快速构建电子电路的实验系统。另外，面包板上的元件可以重复使用，适合进行电子电路实验的组装、操作和调试。

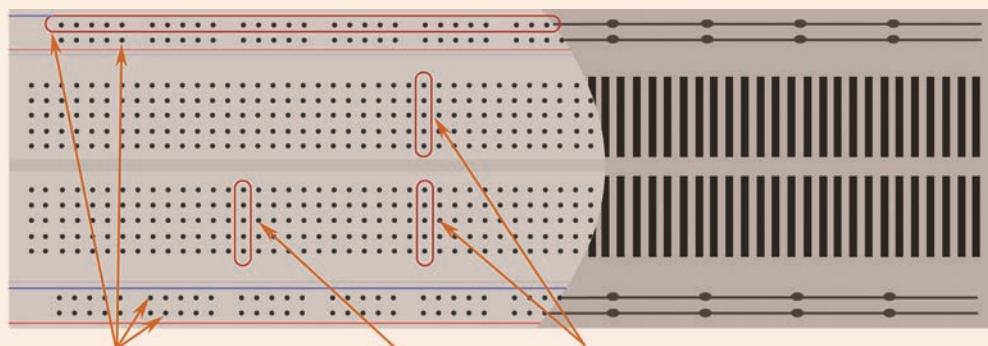


图 2.11 面包板正反面对比图

二、模拟信号输入

模数转换也是微控制器的重要功能之一。在 ATmega328P 的内部集成了一个 10 位精度的 ADC，该 ADC 可接受 0~5 V 的模拟电压输入。它将 PORTC 的 6 个输入/输出接口作为输入通道，对应在 Arduino Uno 开发板上即为 A0~A5 引脚。

1. 电位器



图 2.12 电位器

电位器(如图 2.12 所示)一般有三个引脚,它本质上是一个滑动变阻器,通过滑动或旋转接触的方式改变电阻的值,进而改变电路的输出电压。如果使用中间的引脚,则它可以作为可变电阻使用;如果只使用两边的引脚,则它可以作为定值电阻使用。电位器工作原理图如图 2.13 所示。它的三个引脚中,一般中间引脚用于输出可变电压,两边的引脚分别接地和接固定电压。

2. 模拟输入引脚的编程控制方法

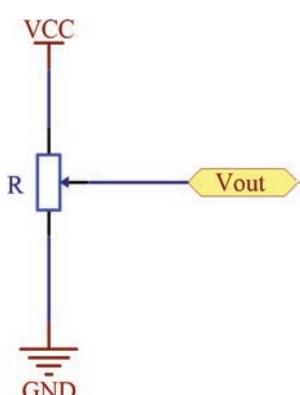


图 2.13 电位器工作原理图

Arduino Uno 开发板有 6 个模拟输入引脚(Arduino Mini 和 Arduino Nano 开发板有 8 个,Arduino Mega 2560 开发板有 16 个),每个引脚接有 10 位精度的 ADC,可以将外部的电压信号转换为能用 10 比特表示的数值。电压量程范围是 0~5 V,即输入电压 0~5 V 将转化为 0~1 023 的整数值,其精度为 $5 \text{ V} / (1 024 \text{ 个单位})$,每个单位约等于 0.004 9 V(即 4.9 mV)。

要从指定的模拟输入引脚读取值,需要用到 `analogRead(pin)` 函数,括号中的参数 pin 为相应引脚号,函数返回值为整型,范围为 0~1 023。

值得注意的是,如果模拟输入引脚没有进行任何连接,`analogRead()` 的返回值也会因为某些因素而波动。

项目实践

随着传感器类型的不断丰富,机器感知外部不同类型的信息也越来越方便。声音传感器能将外界声音信息转化为电信号,就好像给机器加上了“耳朵”,使其能感知外界声音大小的变化。

【项目器材】

Arduino Uno 开发板、声音传感器模块、杜邦线。

【系统框图】

机器耳朵系统框图如图 2.14 所示。



图 2.14 机器耳朵系统框图

【操作步骤】

1. 实物连接。按照图 2.15 连接, 其中声音传感器的信号输出端也可以连接到 A1~ A5 这些支持模拟输入的引脚。
2. 编写程序, 实现声音的获取和显示。
3. 观察结果。点击 Arduino IDE“工具”菜单下的“串口绘图器”命令, 在相应窗口中观察外界声音大小对生成波形的影响, 如图 2.16 所示。

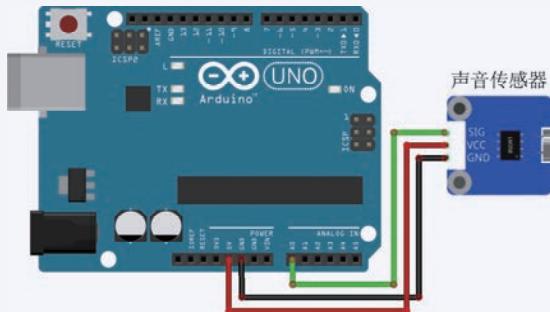


图 2.15 机器耳朵实物连接图

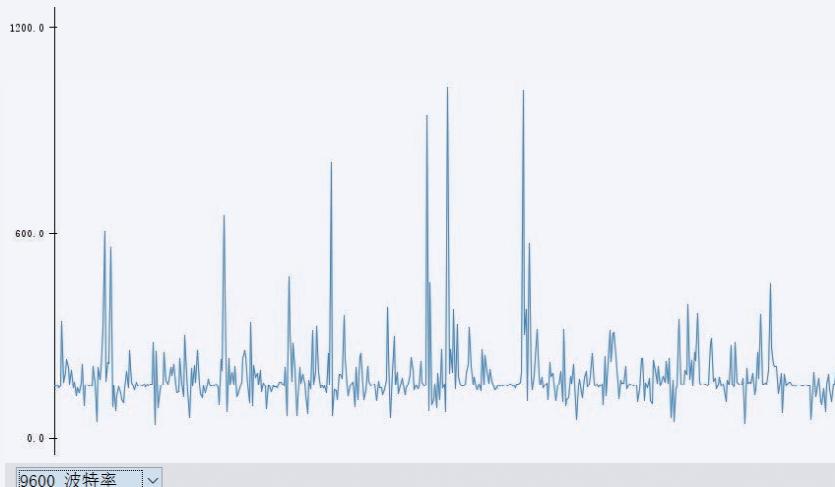


图 2.16 观察声音的波形

【关键程序】

```
void setup() {
    Serial.begin(9600); // 初始化串口传输率为 9600 bps
}

void loop() {
    int sensorValue = analogRead(A0); // 读取 A0 引脚的输入值
    Serial.println(sensorValue); // 串口窗口中显示获取的值
    delay(100); // 延迟 100 ms
}
```

作业练习

设计一个噪声监测仪,通过 LED 的亮度实时反映噪声的强弱: 噪声变大时,LED 变亮;当噪声超过一定检测值时,蜂鸣器响起。

三、机器伙伴·感应节能夜灯

项目实践

制作一个节能机器伙伴——感应节能夜灯。

【需求分析】

晚上,当有人经过时,灯自动亮起,过段时间之后,灯自动熄灭。

【功能设计】

- (1) 当检测到光线低于一定阈值并且有人经过时,点亮 LED。
- (2) LED 每次点亮会持续 30s,如果上一条件仍然满足,LED 继续点亮。

【软硬件设计】

- (1) 主控板的选择

根据项目需求和开源硬件各种开发板的特点,结合表 2.3,选择合适的开发板。你可以自己添加需求,看看主控板是否满足。

表 2.3 感应节能夜灯开发板选材表

| 考虑因素 | 项目要求 | 主控板是否满足项目要求 | |
|---------|--|-------------|-----|
| | | Arduino | 树莓派 |
| 尺寸 | 中等大小 | 是 | 是 |
| 运算能力 | 低 | 是 | 是 |
| 接口类型与数量 | 模拟输入接口 1 个 数字输入接口 1 个 数字输出接口 1 个 | | |
| 电池供电情况 | 支持电池供电 | | |
| 操作系统情况 | 不需要操作系统支持 | | |
| | | | |
| | | | |
| 结论 | | | |

(2) 检测元件的选择

检测光线可用数字式光敏传感器,检测人体可选用超声波传感器、声音传感器、人体红外传感器、摄像

头等。请查阅相关资料,比较这几个传感器,说明选用某个传感器的理由,并填写表 2.4。

表 2.4 传感器选择

| 传感器名称 | 输入信息类型 | 可行性分析 |
|---------|--------|-------|
| 超声波传感器 | | |
| 声音传感器 | | |
| 人体红外传感器 | | |
| 摄像头 | | |

(3) 系统框图

感应节能夜灯系统框图如图 2.17 所示。

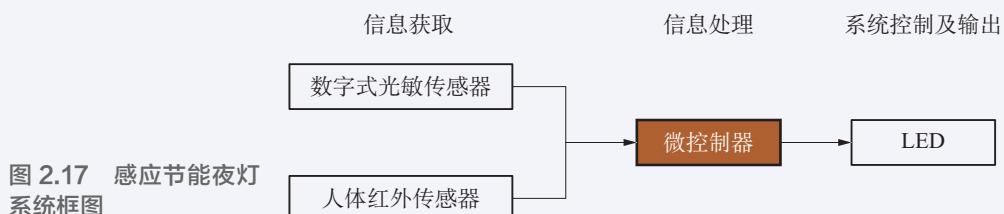


图 2.17 感应节能夜灯
系统框图

(4) 流程示意图

感应节能夜灯流程示意图如图 2.18 所示。

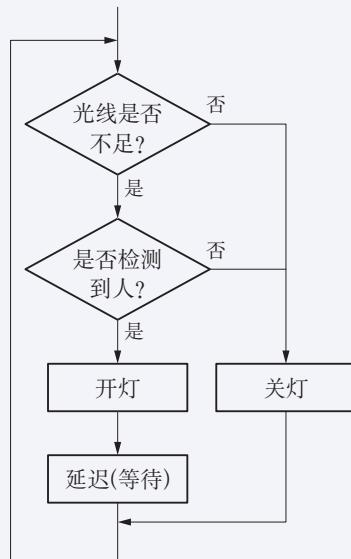


图 2.18 感应节能夜灯流程示意图

(5) 硬件模块搭建

准备的器材有：人体红外传感器 HC - SR501(或其他感应传感器)、光敏电阻、红色 M5 直插 LED、 220Ω 电阻、 $10k\Omega$ 电阻、面包板、杜邦线。

感应节能夜灯实物连接图如图 2.19 所示。

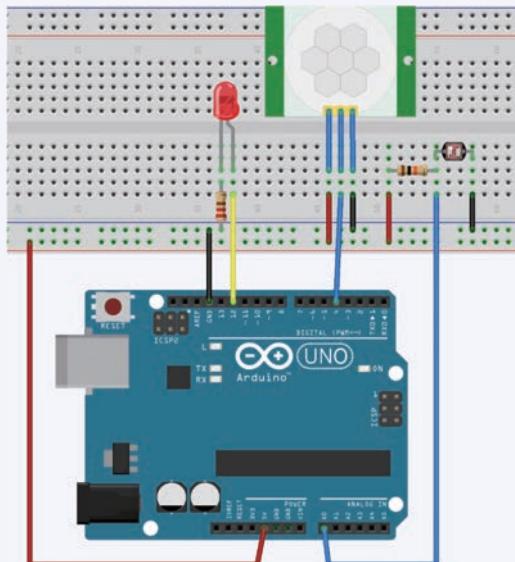


图 2.19 感应节能夜灯实物连接图

(6) 关键代码

```
// 获取光线的模拟数值,analogRead()函数的返回值范围是0~1023  
int lightValue = analogRead (lightSensor); // lightSensor 已定义为 A0  
  
// 获取当前人体红外值,1 为有人,0 为没人,sensorPin 已定义为 4 号引脚  
int humanHeatValue = digitalRead(sensorPin);  
  
// 当入射光弱时,光敏电阻增大,亮度阈值要根据实际情况调整  
if (lightValue>600&& humanHeatValue == 1) {  
    digitalWrite(ledPin, HIGH); //如果光线暗并且也有人经过,则亮灯  
    delay(30000);  
}  
else  digitalWrite(ledPin, LOW); //如果光线亮或没有人,则灭灯
```

【调试优化】

上述程序中将亮度阈值设为 600,我们具体操作时,需要根据实际情况进行修改。试着完善程序,增加相应的串口输出语句,在表 2.5 中记录在不同光线情况下 A0 引脚得到的模拟值。请结合表 2.5,用合适的数值代替程序中原来的亮度阈值,并查看运行结果。

表 2.5 不同光线情况下的运行结果

| 光线情况 | 很暗 | 较暗 | 较亮 | 很亮 |
|--------|----|----|----|----|
| 得到的模拟值 | | | | |

体 验 思 考

和同学讨论：为什么 LED 串联的是 $220\ \Omega$ 电阻，而光敏电阻串联的是 $10\ k\Omega$ 电阻？

技 术 支 持

超声波测距

超声波测距模块(如图 2.20 所示)可以通过发送和接收超声波，利用发送声波和接收到回波的时间差，以及声音的传播速度，来计算距离。设超声波在空气中的传播速度为 v ，测出发送声波和接收到回波的时间差为 t ，则模块到前方障碍物的距离 $s = v \times t/2$ 。

以常用的 HC-SR04 模块为例，我们来看看超声波测距模块的工作原理和过程。图 2.21 表明了超声波测距原理，首先向 HC-SR04 模块的 Trig 引脚输入持续至少 $10\ \mu s$ 的高电平脉冲，这时模块就会自动发送 8 个 $40\ kHz$ 的方波，并自动检测是否有信号返回。如果接收到返回信号，模块就会通过 Echo 引脚输出高电平，高电平持续的时间就是超声波从发送到返回的时间，据此就可以计算出模块到障碍物的距离 = (高电平时间 \times 声速)/2。



图 2.20 超声波测距模块结构图

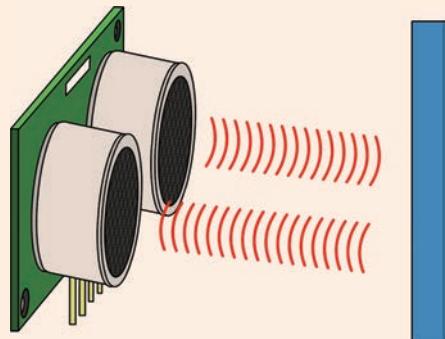


图 2.21 超声波测距原理

超声波测距引脚连接如表 2.6 所示，参考连接图如图 2.22 所示。

表 2.6 超声波测距引脚连接对照表

| Arduino Uno | 超声波模块 |
|-------------|-------|
| 5 V 引脚 | VCC |
| 3 号引脚 | Trig |
| 2 号引脚 | Echo |
| GND 引脚 | GND |

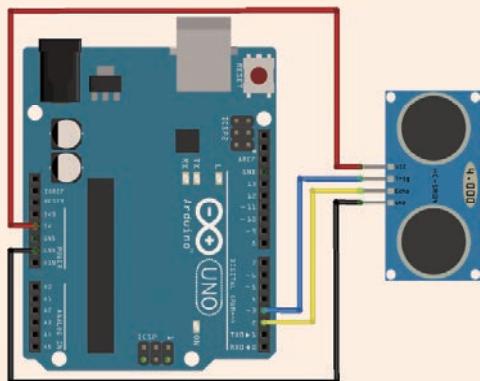


图 2.22 超声波测距各模块连接图

超声波测距的参考程序代码如下：

```
#define Trig 3 //超声波传感器引脚 Trig 连接到 Arduino Uno 开发板 3 号引脚
#define Echo 2 //超声波传感器引脚 Echo 连接到 Arduino Uno 开发板 2 号引脚
float dis(int trigpin, int echopin) {      //自定义测距函数
    float cm, temp;
    digitalWrite(trigpin, LOW); //给 Trig 发送一个低电平
    delayMicroseconds(2);     //等待 2 μs
    //给 Trig 输入持续至少 10 μs 的高电平信号以触发测距
    digitalWrite(trigpin, HIGH);
    delayMicroseconds(10);    //等待 10 μs
    digitalWrite(trigpin, LOW); //给 Trig 发送一个低电平
    temp = float(pulseIn(echopin, HIGH)); //存储回波等待时间, 单位为 μs
    cm = temp * 17 / 1000; //通过回波时间(μs)计算距离(cm)
    return cm;
}
void setup() {
    Serial.begin(9600);
    pinMode(Trig, OUTPUT);
    pinMode(Echo, INPUT);
}
void loop() {
    Serial.println(dis(Trig, Echo)); // 串口窗口中显示获取的值
    delay(100);                  // 延迟 100 ms
}
```

体 验 思 考

1. 上面程序中计算距离的公式为 $\text{temp} * 17 / 1000$, 该公式是怎么简化得到的?
2. 以上案例使用了超声波来测距, 同理, 能不能利用光来测距? 光的速度对测量距离来说有什么问题?

知 识 延 伸

ToF

ToF 即 time of flight, 指通过光的传播时间来计算距离, 其基本原理是根据发射器发射和接收器接收光脉冲的时间差计算光脉冲的往返时间, 进而计算摄像机与物体之间的距离。这种方式要求发射器和接收器尽可能地集成, 时间测量尽可能准确, 这样才能保证距离测量的准确性。

作业练习

当病人在医院输液时, 输液速度一般不能及时显示, 且结束后需要病人按动按钮通知护士前来换液。能不能做一个自动装置既可检测输液速度, 又可以在输液结束后自动传唤? 和同学分组讨论, 并提交输液速度提示器的项目设计方案。

第三节 开源硬件系统的输出

多种多样的传感器让机器的输入方式更灵活,机器可以更好地感知和响应外界信息。一个完整的机器既有多种输入方式又有多种形式的输出。开源硬件项目中,输出方式分为数字信号输出和模拟信号输出,可以输出各种声光效果、显示图像、做机械动作等。

体 验 思 考

如果将机器和人体比较,能否找出与执行部件对应的人体表情语言和肢体动作?

一、数字信号输出

用 Arduino 可以完成的交互产品有很多,最常见也最常用的是声光展示。实现灯光效果最常用的是 LED,而能够发出声音的最常用的元器件是蜂鸣器或音频播放模块。可以用蜂鸣器为机器添加上属于它的“嘴巴”,让机器发声。

技 术 支 持

常用元器件及其使用方法

1. LED

LED 即发光二极管,是半导体二极管的一种,能够将电能转化为可见光,具有体积小、亮度高、功耗低、效率高、寿命长的特点。LED 常在电路及仪器中用作指示灯,或用来显示文字及数字。图 2.23 是 LED 效果图。

LED 具有红、绿、蓝等多种颜色,可以根据需要选用。一般来说,外置的 LED 接到 Arduino 开发板上需要串联限流电阻。Arduino Uno 开发板集成了一个可编程控制的 LED。常见的 LED 如图 2.24 所示。

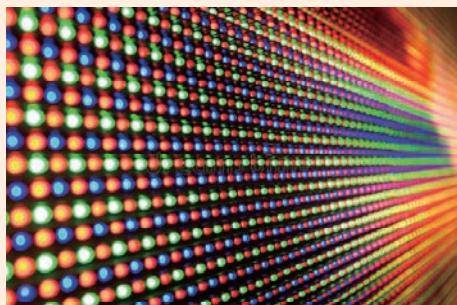


图 2.23 LED 效果图



图 2.24 常见的 LED

2. 蜂鸣器的使用方法

蜂鸣器(如图 2.25 所示)主要可分为有源蜂鸣器和无源蜂鸣器两种,有源蜂鸣器内部带有震荡源,只要通电就能发声,而无源蜂鸣器内部不含震荡源,需要以一定频率的方波去驱动才能发声。

蜂鸣器简单控制程序如下:

```
for(i = 0;i<80;i++) {  
    //蜂鸣器连接在 8 号引脚,高电平时发声音,低电平时不发声音  
    digitalWrite(8, HIGH);  
    delay(1);  
    digitalWrite(8, LOW);  
    delay(1); //也可试着将参数改为 2,看能否产生不一样的声音  
}
```



图 2.25 蜂鸣器

如要让蜂鸣器发出特定频率的声音可使用 tone() 函数,有以下两种格式:

- (1) tone(pin, frequency),向指定引脚发送相应频率的方波,需要执行 noTone() 函数来停止。
- (2) tone(pin, frequency, duration),参数 duration 是发送方波持续的时间,到时自动停止发送,不需要执行 noTone() 函数。

利用 tone() 函数播放音乐时,各个音符对应的频率可参考表 2.7。

表 2.7 C 调各音符频率对照表

| 高音 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|--------|-------|-------|-------|-------|-------|-------|-------|
| 频率(Hz) | 1 046 | 1 174 | 1 318 | 1 396 | 1 567 | 1 760 | 1 975 |
| 中音 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 频率(Hz) | 523 | 587 | 659 | 698 | 783 | 880 | 987 |
| 低音 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 频率(Hz) | 261 | 293 | 329 | 349 | 391 | 440 | 493 |

项目实践

做一个简单的音乐播放器,来帮我们缓解学习和工作中的疲劳。

【项目器材】

Arduino Uno 开发板、蜂鸣器、杜邦线。

【操作步骤】

按图 2.26 连接各模块, 编写程序, 让蜂鸣器发出有规律的乐音。

【关键代码】

```
#define Length 250 //指定每个节拍 250 ms, 可自己调节
#define PINOUT 12 //指定蜂鸣器接在 12 号引脚
// 将低音、中音、高音的音符频率分别放入数组中, 方便调用
int ToneL[] {261, 293, 329, 349, 391, 440, 493};
int ToneM[] {523, 587, 659, 698, 783, 880, 987};
int ToneH[] {1046, 1174, 1318, 1396, 1567, 1760, 1975};
//将需要发出的音符放入数组, 方便调用
int Sound[] {1, 2, 3, 4, 5, 6, 7};
//每个音的节拍数,1 表示 1 拍,0.5 表示半拍
float Tone_Time[] {1, 1, 1, 1, 1, 1, 1};
void setup() {
    pinMode(PINOUT, OUTPUT); //设定指定引脚为输出模式
}
void loop() {
    int j; //定义一个循环变量
    //循环调用音符数组中的每个音符
    for(int i=0;i<sizeof(Sound)/sizeof(int);i++)
        {//数组下标以 0 开始, 音符 1 对应 ToneM[0] 存放的频率
            j=Sound[i]-1;
            //让蜂鸣器发出音符对应频率的声音,持续时间为相应节拍数
            tone(PINOUT, ToneM[j]);
            delay(Length * Tone_Time[i]);
        }
    noTone(PINOUT); //关闭声音
    delay(2000); //静音 2 s
}
```

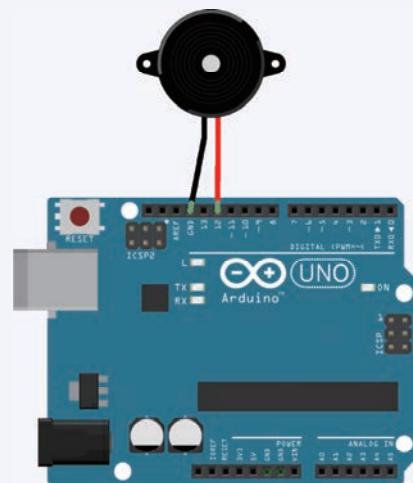


图 2.26 音乐播放器实物连接图

体 验 思 考

1. 上述程序只是实现了中音的播放, 如果要播放高音和低音, 应如何修改程序?
2. 上述程序实现了通电后自动演奏音乐, 如想添加一个按键来控制蜂鸣器奏乐, 硬件连接和程序应如何修改?

作业练习

参考图 2.27,设计一个光敏电子琴,当手在几个不同的光敏电阻上移动,改变它获得的光线时,可以让蜂鸣器发出不同的乐音。

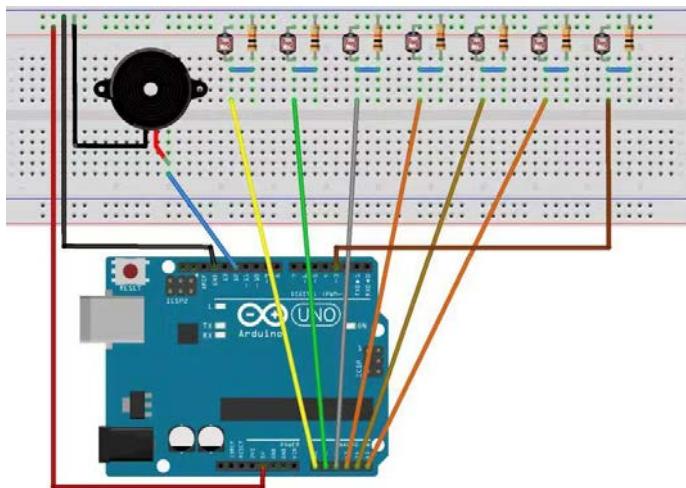
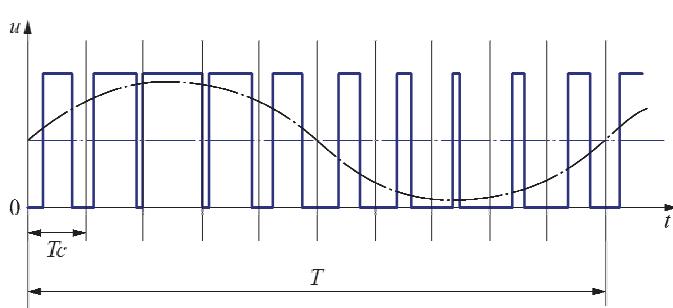


图 2.27 光敏电子琴

二、模拟信号输出

由于处理器处理的都是数字信号,如果想输出模拟信号,需要通过数模转换器件 DAC 来完成数字信号向模拟信号的转换。大部分的微控制器都没有内置 DAC,数模转换功能需要外部附加的模块来实现。



1. 脉冲宽度调制技术

脉冲宽度调制技术简称脉宽调制(pulse width modulation, 缩写为 PWM),是一种利用数字方波信号来近似替代模拟信号的控制技术。PWM 信号是一种周期固定,但脉宽占空比会依据需要而改变的信号。PWM 的示意图如图 2.28 所示。

图 2.28 PWM 示意图

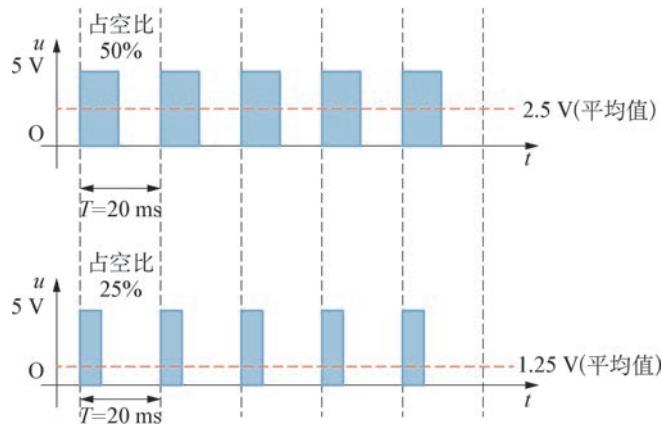


图 2.29 脉冲宽度调制原理图

PWM 对模拟信号的近似替代是通过产生方波的方式来实现的,通俗来讲就是快速地通电、断电。如图 2.29 所示,在很短的一段时间内,比如 20 ms,让高电平持续 10 ms(即通电 10 ms),低电平持续 10 ms(即断电 10 ms)。以这样的 20 ms 为单位,在连续的时间轴上不断地重复这样的 20 ms,产生的信号就是一个周期为 20 ms,占空比为 50% 的 PWM 信号。如果高电平的电压是 5 V,那么这一信号就可以近似替代 2.5 V 的模拟电压。直观上也可以作如下理解:一段

时间内只有一半的时间提供了 5 V 的电压,就相当于在整段时间内只提供了 2.5 V 的电压;类似地,一段时间内只有四分之一的时间提供 5 V 电压,就相当于在整段时间内只提供了 1.25 V 的电压。

PWM 在通信行业常用于信号的调制解调,而在电子应用方面常用于灯光亮度控制、电机调速、舵机控制等。许多微控制器内部都设有 PWM 控制器,Arduino 系列更是将 PWM 信号的输出封装成了方便我们使用的函数 `analogWrite()`。这一函数将占空比 0~100 映射到了 0~255 区间,也就是说,如果我们想向某个引脚 pin 输出占空比 100% 的 PWM 信号,可以用 `analogWrite(pin, 255)` 来实现,如果想输出占空比为 50% 的 PWM 信号,则可以用 `analogWrite(pin, 128)` 来实现。

PWM 替代模拟信号的成本低,其端口可以和普通的数字端口通用,尽管输出电压精度比 DAC 低,但还是被许多微控制器采用,以实现近似的数模转换功能。

体验思考

PWM 输出的方波信号相对于模拟信号,除了实现成本低、简单易行外,还有什么其他的优点?

项目实践

在第一章的呼吸灯项目实践中,实现呼吸灯效果的板载 LED 接在 13 号引脚上,而 13 号引脚不具有

PWM 输出功能,我们是通过程序模拟 PWM 的工作方式实现了呼吸灯效果。下面我们使用其他具有 PWM 输出功能的引脚外接 LED,通过 analogWrite()函数来实现呼吸灯效果。

【项目器材】

Arduino Uno 开发板、红色 M5 直插 LED、220 Ω 电阻、面包板、杜邦线。

【操作步骤】

1. 实物连接。按照图 2.30 连接,LED 的正极接在 11 号引脚,也可以接在其他支持 PWM 输出的引脚,LED 的负极接地。

2. 编写程序。

【关键程序】

```
void setup()
{
    pinMode(11, OUTPUT);
}

void loop()
{
    for(int a = 0; a <= 255; a++)          //通过循环来控制 PWM 占空比逐步增加
    {
        analogWrite(11, a);
        delay(10);                      //当前亮度维持 10 ms
    }
    for (int a = 255; a >= 0; a--)    //通过循环来控制 PWM 占空比逐步减少
    {
        analogWrite(11, a);
        delay(10);                      //当前亮度维持 10 ms
    }
    delay(500);                      //每次“呼吸”后等待 0.5 s
}
```

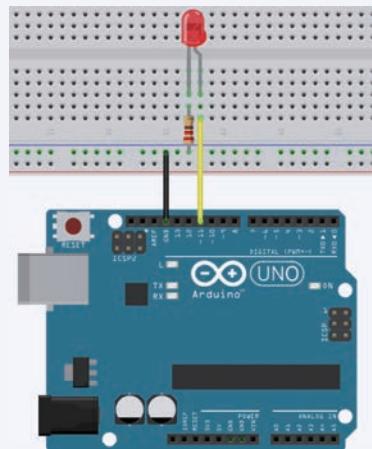


图 2.30 呼吸灯实物连接图

2. 电机

电机俗称“马达”,它能将电能转换成动能,电机工作时可以产生驱动转矩,可作为各种机械运动的动力源。

电机有很多分类:按工作电源来划分,可分为直流电机和交流电机;按工作原理来划分,可分为异步电机、同步电机;按使用用途划分,

可分为动力驱动为主的驱动用电机和能进行精细运动控制的控制用电机,其中控制用电机又可分为步进电机和伺服电机。

(1) 直流电机的驱动

直流电机一般只有两个引脚,分别接在相应电源的正负极上。接上电源后,直流电机就可旋转,如果将连接两个引脚的电线位置调换一下,则直流电机将反向旋转。

让直流电机旋转需要较大的功率,所以不能直接用开发板为其供电,我们需要另外连接电源,并搭建相应电路来驱动,如三极管放大电路和 H 桥直流电机控制电路,也可以使用专门的电机驱动模块来驱动直流电机。

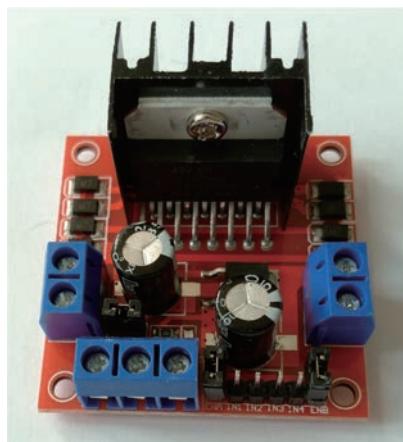


图 2.31 L298N 电机驱动模块

L298N 是常用的包含 H 桥的电机控制芯片,如图 2.31 所示。L298N 作为电机驱动模块,可以独立控制两路电机,驱动两个电机工作。

L298N 模块中,IN1~IN4 则是两路电机输出的转向控制引脚。L298N 电机驱动模块逻辑功能表如表 2.8 所示。当 IN1 和 IN2 同为高电平或低电平时,电机输出端 A 连接的电机都会停止转动;当 IN1 和 IN2 输入信号为一高一低时,输出端 A 连接的电机会朝着不同的方向转动。电机输出端 B 的转向则由 IN3 和 IN4 以同样的方式控制。

表 2.8 L298N 电机驱动模块逻辑功能表

| 直流电机 | 旋转方式 | IN1 | IN2 | IN3 | IN4 | ENA | ENB |
|------|------|-----|-----|-----|-----|--------|--------|
| 电机 A | 正转 | 高 | 低 | / | / | PWM 信号 | / |
| | 反转 | 低 | 高 | / | / | PWM 信号 | / |
| | 停止 | 高 | 高 | / | / | PWM 信号 | / |
| | 停止 | 低 | 低 | / | / | PWM 信号 | / |
| 电机 B | 正转 | / | / | 高 | 低 | / | PWM 信号 |
| | 反转 | / | / | 低 | 高 | / | PWM 信号 |
| | 停止 | / | / | 高 | 高 | / | PWM 信号 |
| | 停止 | / | / | 低 | 低 | / | PWM 信号 |

这样,我们就可以通过给 L298N 模块单独供电,并用微控制器给 ENA、ENB 以及 IN1~IN4 提供控制信号的方式来实现对电机转向、转速的控制。



图 2.32 常用舵机

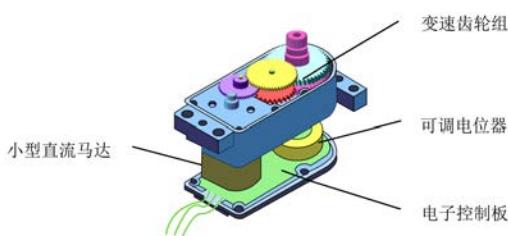


图 2.33 常用舵机的机械结构

(2) 舵机

舵机是伺服电机的一种,伺服电机的转子转动角度可以精确控制,由内置的伺服算法和齿轮提供位置反馈。

常见的舵机(如图 2.32 所示)由直流电机和电位器构成,常被应用在开源硬件项目开发中。其内部结构如图 2.33 所示,电位器作为反馈控制传感器,可监控内部芯片和调整转子位置,其中位置控制用 PWM 信号。舵机的 PWM 输入只是传输一个信号,其脉冲宽度直接对应舵机旋转角度(如图 2.34 所示),但它并不输出功率,功率一般直接由电源提供。常见的舵机有三个引脚,分别是接地、电源正极和控制端。

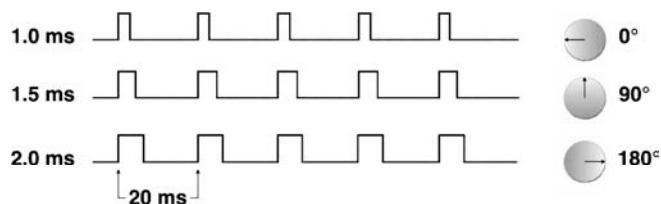


图 2.34 舵机的旋转角度控制

舵机的控制程序如下:

```
#include <Servo.h> // 引用舵机的头文件
Servo myservo; // 创建一个舵机对象
int pos = 0; // 定义舵机的初始角度为 0°
void setup() {
    // 指定舵机的数据线连接的引脚为 9 号引脚, 其能进行 PWM 输出
    myservo.attach(9);
}
void loop() {
    // 舵机从 0° 转动到 180°, 每个角度停留 15 ms
    for (pos = 0; pos <= 180; pos += 1) {myservo.write(pos); delay(15); }
}
```

放在书桌上的小风扇可以帮我们驱散夏日的炎热。自制小风扇，通过一个旋钮来控制风扇的转速，同时 LED 显示风扇的转速：当风扇转动变快时，LED 变亮；风扇转动变慢时，LED 变暗。

【系统框图】

调速小风扇系统框图如图 2.35 所示。

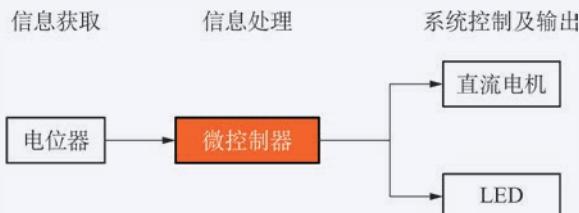


图 2.35 调速小风扇系统框图

【项目器材】

Arduino Uno 开发板、小型 6V 直流电机、L298N 直流电机驱动模块、外接 12V 电源模块、红色 M5 直插 LED、 220Ω 电阻、B50K 电位器、面包板、杜邦线。

【操作步骤】

1. 连接好计算机与 Arduino Uno 开发板。
2. 按图 2.36 所示用杜邦线连接各活动器材。注意所有设备均需要公共接地。除电源线外，L298N 模块的 IN1 和 IN2 引脚连接到 Arduino Uno 开发板的某两个普通数字 I/O 接口，这里选择连接到 7 号引脚和 8 号引脚；ENA 引脚连接到 Arduino Uno 开发板的某一个可进行 PWM 输出的数字 I/O 接口，这里选择 6 号引脚。电位器中间引脚连接到模拟输入接口 A0，另外两个引脚分别接地和 5V 电压。LED 的长脚连接到可进行 PWM 输出的数字 I/O 接口。
3. 适当修改示例程序，对程序进行调试、编译、下载。
4. 旋转电位器旋钮，观察风扇转速和 LED 亮度的变化，同时观察串口监视窗口的输出。

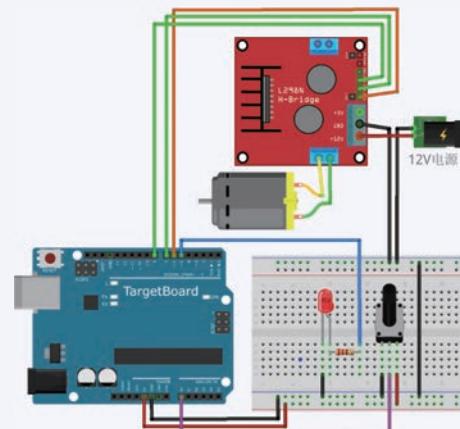


图 2.36 调速小风扇实物连接图

【流程示意】

调速小风扇流程示意图如图 2.37 所示。

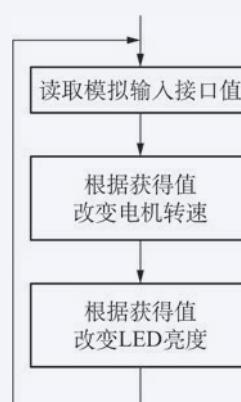


图 2.37 调速小风扇流程示意图

【关键代码】

```
const byte L298N_IN1 = 8; // 定义 L298N 驱动模块 IN1 连接引脚号  
const byte L298N_IN2 = 7; // 定义 L298N 驱动模块 IN2 连接引脚号  
const byte L298N_ENA = 6; // 定义 L298N 驱动模块 ENA 连接引脚号  
void setup() {  
    pinMode(L298N_IN1, OUTPUT);  
    pinMode(L298N_IN2, OUTPUT);  
    digitalWrite(L298N_IN1, HIGH);  
    digitalWrite(L298N_IN2, LOW); // IN1 高电平, IN2 低电平, 电机正向旋转  
}  
void loop() {  
    int val = analogRead(A0); // 读取模拟输入接口 A0 的值, 范围应是 0~1023  
    Serial.println(val); // 显示出 val 的值, 方便进行调试  
    analogWrite(L298N_ENA, (val/4)); // 控制电机转速  
    analogWrite(5, (val/4)); // 控制 LED 亮度  
}
```

体 验 思 考

1. 如果要让电机反向旋转, 应该如何修改程序?
2. 当用语句 `analogWrite()` 函数控制电机时, 为什么要将模拟输入接口 A0 获得值 `val` 除以 4 作为参数?
3. 仔细观察串口监视窗口的输出, 当程序中 `val` 的值比较小时, 电机是否旋转? 如果不旋转, 怎样才能使电机旋转以达到更精细的控制效果?

三、机器伙伴·番茄钟

项 目 实 践

制作一个时间规划机器伙伴——番茄钟, 如图 2.38 所示。

【需求分析】

番茄工作法是一种简单易行的、微观的时间管理方法, 即将工作或学习的阶段性时间默认为 25 分钟, 规定这段时间内专注于工作或学习, 中途不做任何与任务无关的事, 直到番茄钟响起或提示灯亮起, 这样可以暂时休息一下, 然后准备进入下一个番茄钟时间。

设计一个番茄钟, 当把它放在书桌上时, 它可以帮助我们集中注意力, 更

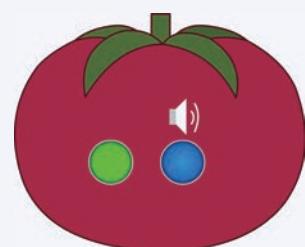


图 2.38 番茄钟

好地规划和管理时间,提高学习效率,养成节约时间和专心学习的习惯。图 2.39 是番茄钟的使用示意图。



【功能设计】

番茄钟的基本功能有：至少设计两个按钮，一个用来开始或重置计时，另一个具有关闭定时功能。默认 1 个学习阶段为 25 分钟，规定的时间一到，蜂鸣器发声，演奏乐曲，同时 LED 闪烁。

此外，还可以添加按钮改变定时的时长，添加多个 LED，显示剩余时间，也可采用不同颜色的 LED 或根据 LED 的不同亮度来显示时间。填写表 2.9 和表 2.10，分析项目活动中 LED 和按钮的功能。

表 2.9 灯的状态及其含义

| 灯的状态 | 表达含义 |
|-------|------|
| 红灯亮 | 计时中 |
| 红灯闪烁 | 时间到 |
| 红灯灭 | 关闭状态 |
| | |

表 2.10 按钮及其对应的功能

| 按钮操作 | 功 能 |
|---------|---------|
| 按钮 1 按下 | 开始计时/重置 |
| 按钮 2 按下 | 关闭计时 |
| 按钮 3 按下 | 显示剩余时间 |
| | |

【软硬件设计】

(1) 系统框图

番茄钟系统框图如图 2.40 所示。

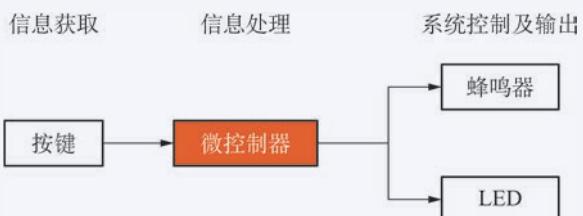


图 2.40 番茄钟系统框图

(2) 流程示意图

番茄钟流程示意图如图 2.41 所示。

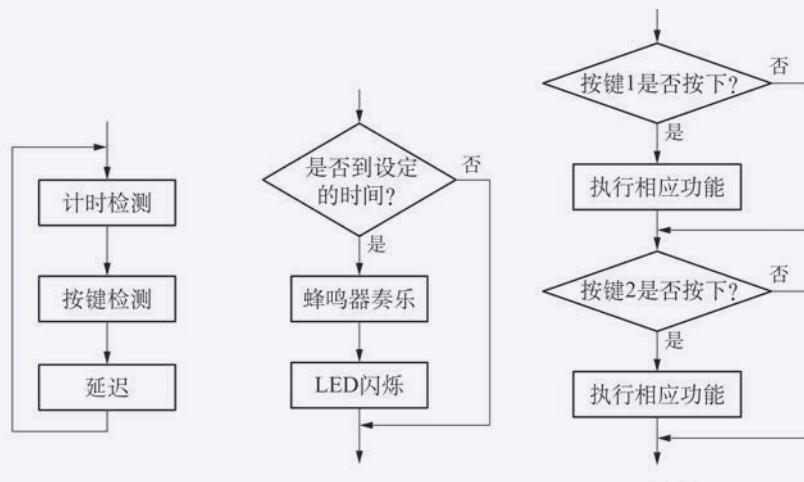


图 2.41 番茄钟流程示意图 (a) 番茄钟程序流程图 (b) 计时检测模块流程图 (c) 按键检测模块流程图

(3) 硬件模块搭建

准备的器材有：按键开关、蜂鸣器、红色 M5 直插 LED、 220Ω 电阻、 $10k\Omega$ 电阻、面包板、杜邦线。实物搭建图如图 2.42 所示。

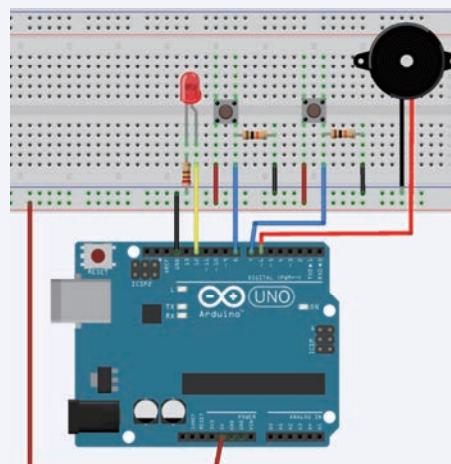


图 2.42 番茄钟实物连接图

(4) 关键代码

计时检测部分关键代码如下：

```
currentMillis = millis(); // 获取当前系统运行时间  
// 当前时间和开始计时时间之差大于预设值时执行相应语句  
if (currentMillis - previousMillis > Timeinterval)  
{ ..... } // 执行具体功能的相应语句
```

按键检测部分关键代码如下：

```
button1LastState = button1State; // 更新按键状态  
button1State = digitalRead(7); // 检测按键 1 的电平输入  
// 按键 1 刚按下的瞬间, 7 号引脚的输入由低电平变为高电平  
if(button1State && !button1LastState)  
{ ..... } // 执行具体功能的相应语句  
delay(100); // 隔 0.1 s 检测一次按键状态
```

体 验 思 考

- 参考的关键代码中, 检测的是按钮按下瞬间的电平变化, 如果要检测按键在手指松开弹起时的电平变化, 要如何修改程序?
- 流程示意图中, 按键检测模块采用不断轮询检查的方式, 也就是在程序中读取相应引脚的输入来判断是否有按键被按下。即使没有按键被按下, 端口的读取还是要进行。试着讨论一下, 这样会带来什么问题?
- 能否采用中断方式来检测按键? 这时流程示意图和程序应如何修改?

技 术 支 持

millis()函数与外部中断函数

millis()函数: 返回 Arduino 开发板从运行当前程序开始的毫秒数。函数返回值将在约 50 天后溢出(归零)。

外部中断函数: 其格式为 attachInterrupt(中断号, 服务程序名, 触发条件)。Arduino Uno 和 Nano 只有引脚 2 和引脚 3 支持外部中断, 中断号分别是 0 和 1。外部中断的触发条件取值及其含义如表 2.11 所示。

表 2.11 不同的触发条件取值及其含义

| 触发条件取值 | 含 义 |
|---------|-------------------|
| LOW | 低电平 |
| CHANGE | 引脚电平发生变化 |
| RISING | 引脚电平由 0 变 1, 即上升沿 |
| FALLING | 引脚电平由 1 变 0, 即下降沿 |

体 验 思 考

如果几个按钮同时按下,会产生怎样的运行结果?

四、机器伙伴·安全警示灯

项 目 实 践

制作一个安全机器伙伴——安全警示灯。

【需求分析】

当人们在道路上跑步或骑行时,尤其是处在较暗的外界环境中时,一个安全警示灯显得很有必要,它能够发出醒目的灯光,来提醒附近的车辆,从而保障我们的安全。

【功能设计】

可将警示灯做成像章或臂章样式,可以佩戴在衣服或背包上。警示灯可以用多个LED来组成有个性的图案造型,可以将多个颜色的LED按照颜色分组串联,分别控制,把彩色LED放在图案中心;要能够通过按钮来控制LED闪烁的频率,如慢闪、快闪,使不同颜色的LED分别点亮等。

参照图 2.43 所示安全警示灯造型,小组讨论,用多个 LED 组成安全警示灯的图案,绘制草图。

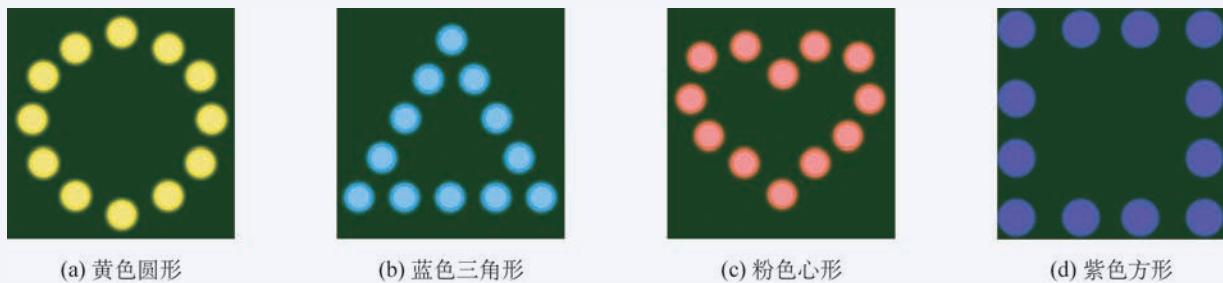


图 2.43 安全警示灯效果图

【软硬件设计】

(1) 主控板的选择

根据项目需求和开源硬件各种开发板的特点,结合表 2.12,选择合适的开发板。也可以自己添加需求,思考主控板是否满足。

表 2.12 安全警示灯选材表

| 考虑因素 | 项目要求 | 主控板是否满足项目要求 | | |
|----------|--|-------------|--------------|-----|
| | | Arduino Uno | Arduino Nano | 树莓派 |
| 尺寸 | 小型 | 否 | 是 | 否 |
| 运算能力 | 低 | 是 | 是 | 是 |
| 接口类型与数量 | 数字输入接口 1 个 数字输出接口 2 个 PWM 输出接口 3 个 | | | |
| 纽扣电池供电情况 | 支持纽扣电池供电 | | | |
| 操作系统情况 | 不需要操作系统支持 | | | |
| | | | | |
| | | | | |
| 结论 | | | | |

(2) 系统框图

安全警示灯系统框图如图 2.44 所示。



图 2.44 安全警示灯系统框图

(3) 流程示意图

安全警示灯流程示意图如图 2.45 所示。

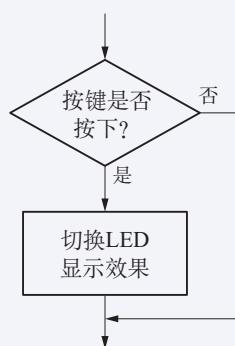


图 2.45 安全警示灯
流程示意图

(4) 硬件模块搭建

准备的器材有：红色 M5 直插 LED、按键开关、 220Ω 电阻、 $10k\Omega$ 电阻、面包板、杜邦线。

此处以简单的直线型图案为例，实物搭建图如图 2.46 所示。如要实现图 2.43 效果图，可在实物板上重新摆放 LED 的位置，使之形成相应图案。

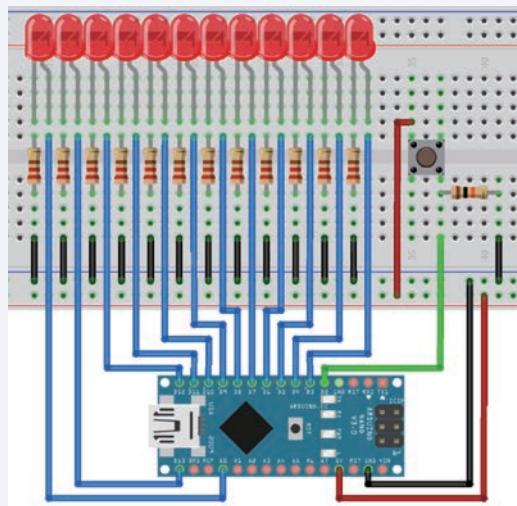


图 2.46 安全警示灯实物连接图

(5) 核心程序

```
int delaytime = 50; // 预设 LED 闪烁的延迟时间
int led_status = 0; // 预设 LED 的状态，可通过点击按钮，调用中断处理函数，改变其值
void setup() {
    for(int i=3;i<=14;i++) {pinMode(i, OUTPUT);}
    attachInterrupt(0, Press_button, RISING); // 在 2 号引脚捕捉中断
}
void loop() {
    if (led_status==0) { // LED 常亮
        for(int i=3;i<=14;i++) {digitalWrite(i, HIGH);}
    }
    if (led_status==1) { // LED 快闪
        for(int i=3;i<=14;i++) {digitalWrite(i, HIGH);}
        delay(delaytime);
        for(int i=3;i<=14;i++) {digitalWrite(i, LOW);}
        delay(delaytime);
    }
    if (led_status==3) { // LED 常灭
    }
}
```

```
    for(int i=3;i<=14;i++) {digitalWrite(i, LOW);}
}
}

void Press_button(){ // 定义中断处理函数
    led_status = (led_status + 1) %3; //状态变量在 0~2 之间切换
}
```

体 验 思 考

能否再增加几种 LED 的显示效果,如 LED 慢闪效果或 LED 逐个点亮的流水灯效果,如何相应地修改程序?

作业练习

经过以上项目实践,我们已经完成了感应节能夜灯、番茄钟和安全警示灯的制作。能否把这三个项目作为三个模块整合在一个作品中,让它能根据实际需要,通过按钮切换到想要的模块? 你还能试着再设计一个充满创意的机器伙伴吗? 和同学分组讨论,并提交相应的项目设计方案。



第三章

开源硬件系统的模块 扩展和连接

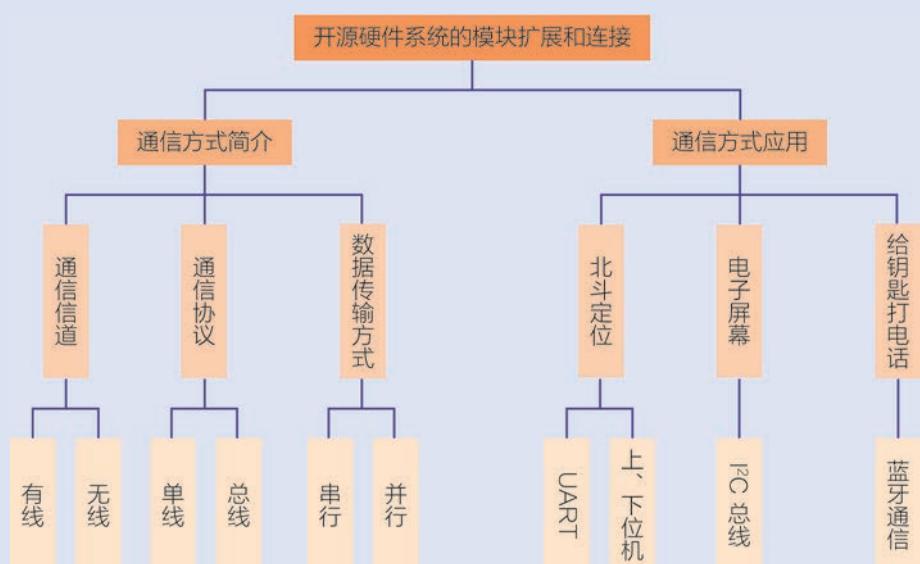
本章学习目标

-
- 了解开源硬件扩展与连接的基本概念、基本原理,掌握常用的编程方法。
 - 能通过扩展模块的连接,运用编程语言,完成开源硬件系统的扩展。
 - 结合学习实践,能在不同连接、不同通信方式下搭建由多种传感器、执行器共同组成的应用系统。
-

信息系统之间的“沟通”被称为通信,开源硬件有许多功能强大的扩展模块,一般都带有特定的通信方式。多个模块之间通过这些特定的通信实现协同工作,从而使信息系统的性能得到整体的提升。我们设计信息系统时必须坚持系统观念,认识到“万事万物是相互联系、相互依存的”。在不久的未来,人们将进入一个万物互联的时代,试想一下:一个冰箱可以根据用户的需求,结合用户的习惯,给出所缺食物的采购方案,甚至通过网络自动完成订货;一个蔬菜大棚能够根据环境温度、湿度以及土壤特性等数据,自动进行通风和喷洒灌溉。这些物联网的应用场景将不断涌现,而应用往往是由多种复杂的硬件组合而成,硬件之间的通信保证了系统的正常运行。

本章将利用开源硬件及一些扩展模块,通过各种硬件的连接、组合、控制,实现开源硬件系统的扩展与连接,进而构建开源硬件新的应用场景。当同学们拿到一个功能强大的模块的时候,别急着去使用,先阅读它的说明书,并进一步思考:这个扩展模块的连接规则是怎样的?如何根据需求选择合适的通信方式?思考后再埋头苦干,试验该模块和单片机的连接。

本章知识结构



项 · 目 · 情 · 境

随着开源硬件的发展和各种传感器的应用,大大小小的开源系统广泛存在于我们生活的方方面面,它们作为我们的生活助手,给我们提供帮助,提升我们的生活质量。

运动是保持健康的重要方法,但有时会遇到一些问题,比如:我们看不到自己的运动数据,难以判断是否达成预定的运动目标;跑步时不知道自己的准确位置,难以判断运动路程;运动过程中如果身体不适难以提前预警,等等。那么怎样利用开源硬件帮助我们解决这些实际问题呢?这些开源硬件和模块之间是如何通过连接和信息交互,充分发挥各部分的作用,组成一个能实现特定功能的系统的?

在本章的学习中,我们需要在分析需求的基础上,小组合作设计运动助手方案作品,运用有线及无线通信,并采用蓝牙、Wi-Fi 等通信模块,制作相关产品来解决问题,可通过连接起更多设备实现万物互联的设计目标。

项 · 目 · 任 · 务

任务 1

开发一个具有定位功能的运动助手,将北斗定位模块与微控制器模块连接,实现位置信息的获取。

任务 2

开发一个具有显示功能的运动助手,将显示模块与开源微控制器模块连接,实现基本信息显示的功能。

任务 3

开发一个具有搜寻功能的运动助手,使用智能手机向连接有蓝牙模块的物件发送信息,通过智能手机实现发声和发光等功能。

第一节 通信方式简介

组成开源硬件系统的各个内部模块、系统间都需要信息的交换和传递。通过互联可以完成信息互通，实现更复杂、多样的功能，连接系统的设备数目和种类也会达到更大的规模，进而形成万物互联的物联网。为了合理并可靠地构建开源硬件系统，选择合适的通信方式很重要，一般需要考虑通信信道、通信协议和数据传输方式。

一、通信信道的选择

借助传输通道(也就是传输媒介)，信息的接收方可以接收到来自发送方的信息，通道一般分为有线和无线两种。常见的杜邦线、USB线、HDMI信号线都是以其内部的导电金属为媒介完成信息传输通信的，属于有线通信；蓝牙耳机、手机以电磁波为传输媒介传输信息，属于无线通信。

在第一章的烟雾报警器案例中，烟雾传感器和蜂鸣器通过在印刷线路板上的导线连接到微控制器，它们之间的信号传输采用的是直接连接的有线通信方式，而大楼中的火灾信息系统通过Wi-Fi连接分布在各楼层的烟雾报警器，采用的是无线通信方式。

在开源硬件系统内部，一般都采用有线的方式连接各个模块，这样的有线连接又分为直接连接和总线连接；连接到系统外部的设备或系统时，有线和无线两种方式均可采用。

体 验 思 考

为避免在公共场合打扰到他人，人们在接听电话或欣赏音乐时都会使用耳机。如果采用连接线直接把耳机连接到手机，那么语音信号就通过有线传输的方式传输；蓝牙耳机则是使用无线方式传输语音信号。举一些在日常生活中采用有线通信和无线通信传输同类信息的例子，并比较两种传输方式的优缺点。

二、通信协议的选择

当需要传输的信息为单个的数字信号“0”或“1”时，可以将器件或者模块的接口通过导线直接连接到微控制器的输入/输出端口，此时

采用一对一的通信方式,只要微控制器对端口进行一次读或写就可以完成全部信息的传输,通信信道被信息的发送方和接收方独占,无法实现更大范围的信息共享,这种方式一般用于短距离传输的场合。比如,按键开关的闭合或断开的信息可以用一个数字信号表达,所以可以把开关直接连接到微控制器的输入端,微控制器可以读取此端口的信息,获得按键开关的状态,这样就完成了信息的传输,如果将微控制器的端口直接连接到 LED,就可以通过向这个端口输出高电平或者低电平来控制 LED 的亮或灭。

为了应对复杂的通信环境,也为了更远更有效地传输数据,需要选择合适的通信方式,采用适当的通信协议。借助通信协议的约束和规范,信息传输在可靠性、安全性和传输距离等方面都有了保障。不同的通信协议所适用的场合和能够解决的问题都不同,需要根据设计要求选择。采用总线类型协议可以实现通信信道的共享,连接更多的模块或设备,如图 3.1 所示。例如,USB 总线可以同时连接符合 USB 接口协议要求的 U 盘、鼠标、键盘等多种类型的多个设备。

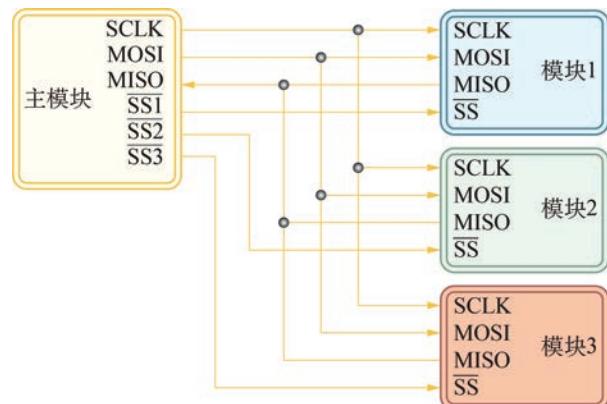


图 3.1 某种总线协议的模块连接示意图

体 验 思 考

采用红外遥控技术的家电遥控器已经非常普遍,很多家庭都有多个遥控器。通常,这些遥控器都可能使用相同波长的红外信号来发送信息。在使用中会发现,与家电配套的遥控器对同品牌的同款家电都可以正确操作,但对其他品牌或不同款的家电无法操作,这是什么原因?

三、数据传输方式的选择

数据信息可以逐位顺序的方式传送,此方式称为串行通信(如图3.2所示);也可以在同一时间多位数据信息一起传送,这种方式称为并行通信(如图3.3所示)。串行通信一般用于模块之间、设备之间或系统之间的数据传输;并行方式主要应用于数据量大、传输距离近的器件间或电路板上模块间的信号传输。由于通信技术的发展,单条线上数据的传输速率越来越大,串行通信已经可以满足大多数系统对传输速率的要求,此外,串行通信可以极大地减少通信接口引脚数目,在高度集成的微控制器应用系统中串行通信已成为各模块间通信的主要方式。

图3.2给出了8位数据信息以串行通信方式从设备A传送到设备B的示意图。图3.3给出了8位数据信息以并行通信方式从设备A传送到设备B的示意图。在串行通信方式下,8位数据将分8次按照排列顺序逐个通过一条信号线传送;在并行通信方式下,8位数据则分别通过8条信号线一次性完成传送。

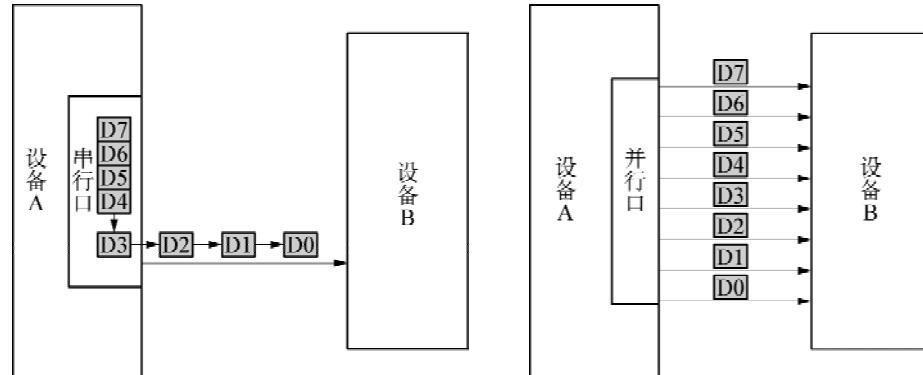


图3.2 串行通信示意图

图3.3 并行通信示意图

第二节 通信方式应用

在设计开源硬件项目时,常常会通过增加扩展模块来增加系统的功能,微控制器与扩展模块之间的通信可以用总线通信的方式来完成。比如为保存系统运行时采集的数据及运算结果,系统的存储容量需要增加。此时就可以添加一个包含存储芯片的存储模块,采用总线连接的方法来实现系统的简单扩容。

常用的总线通信方式分为有线和无线两种。常见的有线连接类型有 UART 串行通信接口、I²C 总线、SPI 总线等;常见的无线连接类型有蓝牙和 Wi-Fi 等。传输信息需要考虑传输数据量的大小、传输速率、传输时间延迟、通信信道和功耗等多方面的因素。微控制器的通信接口种类及数量、扩展模块具有的接口种类、需要同时连接的模块数目,将影响可选择的总线通信方式的范围。总线通信的具体方案要根据微控制器和扩展模块的通信接口参数、所传输信息的特点等因素综合考虑确定。

一、运动助手·北斗定位

以前人们去一个陌生的地方往往需要带上纸质的地图,现在,人们只要打开手机就能查看路线,这不仅更加快捷方便,而且当开启手机上的定位功能,就可以知道自己在哪里,并且可以根据自己的位置,搜索周边的景点、餐厅、商城等,而这一切的基础就是定位技术。

北斗卫星导航系统(BeiDou navigation satellite system,简写为 BDS)是中国自行研制、自主建设、独立运行的全球卫星导航系统,由北斗接收模块获得北斗定位卫星发射的信号,据此可实现授时和定位的功能。

安装了北斗接收模块的手机可以实现实时的精确定位。但是,离开了手机,我们如何进行定位呢?尝试使用开源硬件及扩展模块设计自己的定位仪,并用定位仪获取位置信息。

技术 支 持

UART 串行通信接口

通用异步收发传输器(universal asynchronous receiver/transmitter,缩写为 UART)支持点对点的双向通信,因其连接简便、协议易实现而在开源硬件系统中得到了广泛的应用。几乎所有的微控制器都提供 UART 串行通信接口(缩写为 UART 接口),并且内置了专门处理 UART 通信协议的 UART 通信模块,以方便用户利用此接口扩展系统功能。UART 通信模块的功能很多,仅使用 UART 模块提供的接收数据引脚 RX 和发送数据引脚 TX 就可以实现最基本的 UART 通信。二者都通过各自的 RX 引脚接收对方的 TX 引脚发送的数

据,也可以通过各自的 TX 引脚发送数据到对方的 RX 引脚。在实际使用时,通信的双方要对 UART 串行通信接口采用相同的参数配置,以保证数据的正常传输。

知识延伸

上位机与下位机

自动化控制领域的信息系统往往具备较为复杂的功能,需要完成数据的采集、分析、汇总、显示,以及对于底层设备的控制等,此外还需要实现人机交互,因此会使用多个计算设备相互配合来完成。如图 3.4 所示,在系统中负责人机交互、发布具体操作指令的计算机被称为上位机;而负责接收命令、承担基础数据采集、执行控制命令、实现底层操作的计算机或设备被称为下位机,常用的有微控制器。上、下位机之间交换信息需要通信协议的支持,几乎所有的计算机和微控制器都具备 UART 接口,因此 UART 接口在上、下位机互联中得到广泛应用。



图 3.4 上、下位机连接示意图

知识延伸

USB 转 UART

早期的台式计算机一般会提供符合 RS - 232 电气接口规范的 UART 接口,为了增强信息传输的抗干扰能力,实现远距离传输,其引脚上的电压范围为 -15~15 V。台式计算机的 UART 接口与 Arduino Uno 开发板的 UART 接口不匹配,二者不可以直接连接,而现在的便携式计算机几乎都取消了 UART 接口,但几乎都具备 USB 接口,因此 Arduino Uno 开发板上放置了一块 USB 转 UART 芯片(参见第一章图 1.17),计算机可以识别此 UART 接口,并借助它实现与 Arduino Uno 开发板上的微控制器通信。

探究活动

Arduino Uno 的板载 UART 接口和计算机的 RS - 232 串行接口在电气性能方面存在差异,查阅相关资料并进行学习探究,填写表 3.1,并思考:UART 接口连接两个设备时,需要连接哪些线?为什么?

表 3.1 RS - 232 串行接口与 UART 接口的比较

| | 信号电压范围(V) | 信号传输距离(m) | 数据传输速率 |
|-----------------------|-----------|-----------|--------|
| 计算机的 RS - 232 串行接口 | | | — |
| Arduino Uno 的 UART 接口 | | | |

【需求分析】

外出旅游,特别是自驾游时,有了一个定位设备就可以随时知道自己所在的位置,方便规划路径,为旅行的顺利进行保驾护航。

【功能设计】

利用北斗接收模块和计算机制作一个旅行必备的定位仪。北斗接收模块与微控制器组成北斗定位模块,可以获得所在地的经、纬度数据,属于系统的下位机;电脑构成上位机,完成显示、后续处理等操作。北斗定位模块与计算机通过串口通信的方式连接,实时地将位置数据传送到计算机上显示。

【软硬件设计】**(1) 器材选型**

定位模块选用串行接口的北斗定位模块,微控制器选用 Arduino Uno。具体器材包括:Arduino Uno、北斗接收模块、USB 数据线、2~3 米 SMA 接头的天线(室内用时可选)、计算机。

(2) 硬件模块搭建

按图 3.5 所示连接 Arduino Uno 和北斗接收模块,供电成功后模块上的 TXD 指示灯会以 1 s 的时间间隔闪烁。

(3) 流程示意图

北斗定位程序的流程示意图如图 3.6 所示。

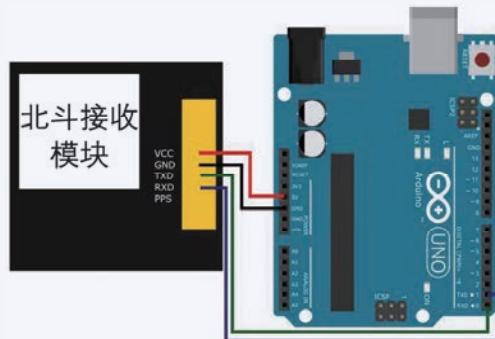


图 3.5 Arduino Uno 与北斗接收模块连接图

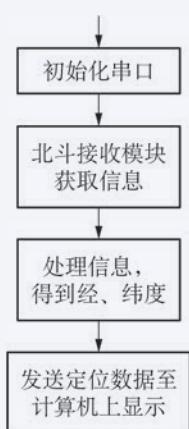


图 3.6 北斗定位程序流程示意图

(4) 关键代码

编写相关代码,编译并下载到 Arduino。

```

#define BeiDouSerial Serial
int LED=13; //指定 LED 指示灯端口为 13 号引脚
struct
  
```

```

{
    char BeiDou_Buffer[80];
    bool isGetData;           //获取到 GPS 数据时为“真”
    bool isParseData;         //解析完成时为“真”
    char UTCTime[11];        //获得 UTC 时间
    char latitude[11];       //获得纬度
    char longitude[12];      //获得经度
    bool isUsefull;          //定位信息有效时为“真”
} Save_Data;

void setup() {
    BeiDouSerial.begin(9600); //设置 UART 串口通信参与 9600 波特,与北斗接收模块 UART 配置一致
}

void loop() {
    /* 北斗接收模块定位信息的获取、处理、输出等功能已封装成函数,方便使用 */
    BeiDouRead();           //获取 GPS 数据
    parseBeiDouBuffer();    //解析 GPS 数据
    printBeiDouBuffer();    //输出解析后的数据
}

```

【调试优化】

- (1) 打开串口监视器,设置串口传输速率为 9 600 bps,没有结束符。
- (2) 若模块还未完成定位,会收到如图 3.7 所示的信息。

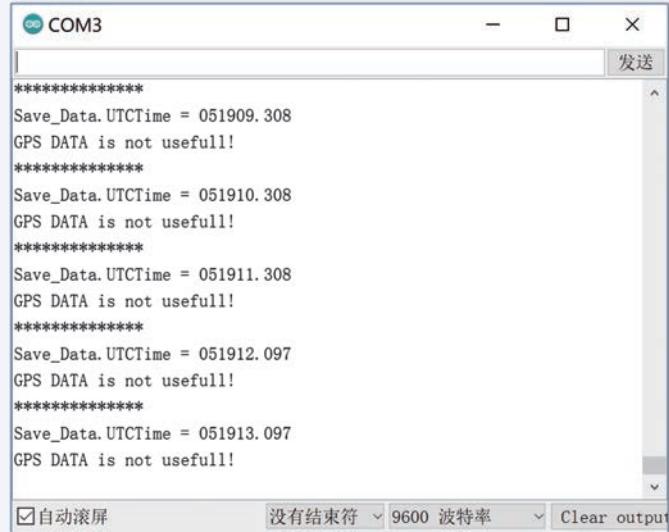
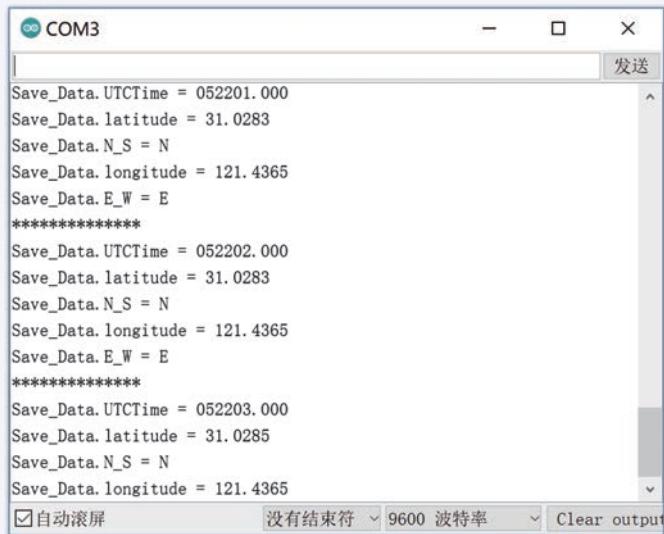


图 3.7 未完成定位时的提示信息

若模块完成定位,会收到解析后的定位信息(如图 3.8 所示),同时北斗模块上的指示灯会以 1 s 的时间间隔闪烁。



```
Save_Data.UTCTime = 052201.000
Save_Data.latitude = 31.0283
Save_Data.N_S = N
Save_Data.longitude = 121.4365
Save_Data.E_W = E
*****
Save_Data.UTCTime = 052202.000
Save_Data.latitude = 31.0283
Save_Data.N_S = N
Save_Data.longitude = 121.4365
Save_Data.E_W = E
*****
Save_Data.UTCTime = 052203.000
Save_Data.latitude = 31.0285
Save_Data.N_S = N
Save_Data.longitude = 121.4365
```

图 3.8 完成定位后的提示信息

(3) 复制得到的经、纬度信息,并至定位网站查询,验证自己得到的定位信息是否准确。

体 验 思 考

1. 可否带上北斗定位仪出去旅游并用它来记录整个旅游轨迹呢? 如何让它实现记录功能呢?
2. 用北斗模块可以获取经、纬度的数据,如何将其与电子地图结合,设计完成一个专属的导航系统,让参观校园的游客可以方便地到达各参观点?

二、运动助手·电子屏幕

信息处理系统项目设计一般通过显示运行结果及数据来实现人机交互。简单的信息传递可以用 LED 等简单模块直接完成,而复杂一些的信息就需要由液晶显示屏、OLED 屏和数码管等显示模块来呈现,这也是开源硬件项目设计中的常见做法。

这类显示模块安装了显示器件,以 OLED 屏为例,屏幕的显示需要通过显示的位置、显示的字模(也就是图形)、显示的颜色等参数来控制,因此会给模块配备一个专用的显示器件控制器,它从微控制器

接收显示的命令和数据,完成显示操作。这类专用显示控制器一般会包含一个很小的内置常用字库,并留出少量的内存存储不在常用字库内的汉字或图形,以满足更多样的显示需要。该类模块一般都会提供总线连接接口,方便与微控制器连接。

技术 支 持

I²C 通信接口

I²C(inter-integrated circuit)是一种串行通信总线,广泛地用于低速模块之间的短距离通信,具有多种传输速率,可连接多个设备。I²C 使用两根信号线,分别是串行数据线 SDA(serial data)和串行时钟线 SCL(serial clock),典型的电压值为 3.3 V 或 5 V。连接在 I²C 总线上的所有 I²C 模块要把各自的数据线和时钟线分别连接在一起,如图 3.9 所示,完成连接后的数据线和时钟线都要设置为高电平。常见的应用包括 EEPROM 存储芯片、外置的低速 ADC 和 DAC、低速传感器模块、即插即用设备的初始信息获取等。

I²C 总线协议有很多的版本,定义了支持各种不同传输速率的总线工作模式,传输速率覆盖了 100~5 000 Kbps 的范围。

总线上的 I²C 模块被称为节点,为保障 I²C 模块正常使用,每个节点都将被赋予一个各不相同的地址值,该地址值就是对应的节点在总线上的唯一标识。地址值用 7 个比特表示,其中 16 个地址值为保留值,不能用来给节点赋值。因此一条 I²C 总线最多可以包含 112 个地址不同的模块,只要知道某一节点的地址值,就可以通过 I²C 总线与之建立通信。

微控制器上的通信端口是非常有限的,如果需要用到的扩展模块数量较大、种类较多,可以使用 I²C 总线接口来连接。对于前文所述的在系统中添加含 I²C 总线接口的存储模块的方案,只要每个存储模块都能够配置成各不相同的 I²C 总线地址,就可以方便地实现系统存储的扩容,同时微控制器可以通过各模块所设定的 I²C 总线地址与之建立通信,实现信息交换。

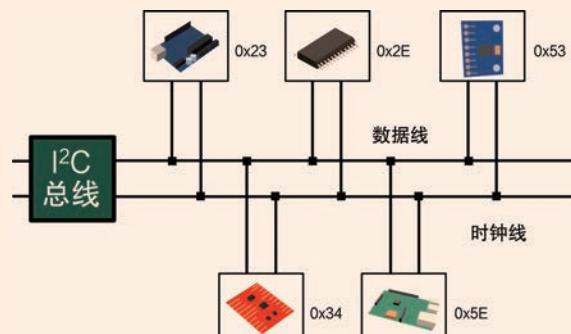


图 3.9 I²C 模块在 I²C 总线上的连接示意图

知 识 延 伸

什么是 SPI?

SPI(serial peripheral interface)是一种同步串行通信总线,包括:时钟信号、主机输入信号、主机输出信号和片选信号线,广泛地应用在开源硬件项目设计中,例如应用于 SD 卡、LCD 显示模块等的连接。

探究活动

1. 某开源硬件项目设计团队正在评估一个设计方案,选取了包含1个UART接口、1个I²C总线接口和1个SPI总线接口的微处理器作为开源硬件的控制器,现需要连接9个开源硬件扩展模块(假设这些模块都有I²C或SPI或UART接口的型号可供选择),选择这些模块的接口需要考虑哪些因素?

如果其中有1个模块只有UART接口的型号可供选择,试提出尽可能多的微处理器与扩展模块的连接方案。如果其中有1个模块只有SPI总线接口的型号可供选择,试提出尽可能多的连接方案。

2. 查阅资料,了解SPI与I²C的特点,比较I²C总线、SPI总线和UART接口,并填写表3.2。

表3.2 I²C总线、SPI总线和UART接口的比较

| 通信协议 | 通信速率 | 节点数量 | 典型应用场合 |
|------------------|------|------|--------|
| I ² C | | | |
| SPI | | | |
| UART | | | |

知识延伸

扩展模块驱动程序库

显示模块上的显示屏幕有色彩、分辨率等参数;其上的控制芯片决定了显示屏的功能,芯片的正常工作依赖于对微控制器芯片的正确操控,而操控此类显示模块的一系列程序就被称为显示驱动程序,这些程序组合在一起就构成了驱动程序库。编写这类程序需要对芯片内部结构、显示模式等诸多条件有足够的了解,编写的难度较大。因此,一般会选用现成的支持该显示控制芯片的驱动程序库,不同的程序库的功能和接口都会有所不同。

图3.10所示的OLED模块的控制芯片为SSD1306,两个开放源代码的U8g2驱动库和Adafruit驱动库可以实现常用的功能。在Arduino IDE菜单项中选取“工具→管理库”,打开“库管理器”,在“主题”中选择“显示”,可以很方便地看到能够使用的驱动程序库并了解其支持的芯片种类。可根据需要添加程序库完成安装,一般还会同时安装有价值的参考例程。

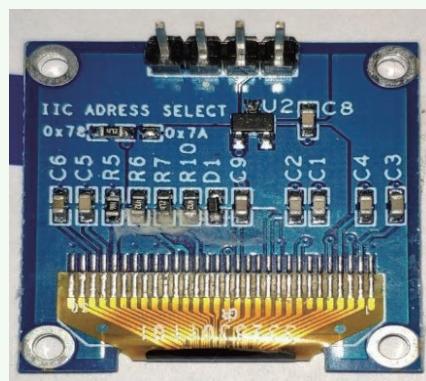


图3.10 I²C接口的OLED模块

【需求分析】

生命在于运动,适当的运动有益健康。为方便运动锻炼,需要将运动提醒信息发送到屏幕上。设计一个计时装置,当运动时间达到设定值,该装置可在 OLED 屏上给出提示语。

【软硬件设计】**(1) 器材选型**

对所选的关键器件进行评估,此处所选择的显示模块居于重要地位。根据表 3.3 完成对三种控制芯片的评估。

表 3.3 电子屏幕选材表

| 模块名称 | 关键芯片型号 | 接口类型 | 驱动程序支持 | 评估意见 |
|------|---------|------------------------|--------|------|
| 显示模块 | SSD1306 | I ² C 或 SPI | 2 种以上 | 合格 |
| | | | | |
| | | | | |

准备的器材有: Arduino Uno、含 I²C 总线接口的 OLED 模块、连接线。

(2) 硬件模块搭建

本项目中的 OLED 模块为 I²C 总线,共有四根线,分别是电源线、接地线、SDA 和 SCL;Arduino Uno 开发板连接插件上的 I²C 总线,SDA 信号引脚编号为 A4,SCL 信号引脚编号为 A5。

按照 I²C 总线的使用要求,设计连接图并检查其正确性。确认后按图 3.11 所示连接 Arduino Uno 和 OLED 显示模块。

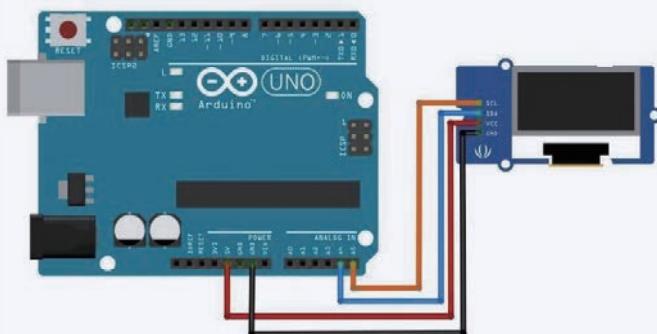


图 3.11 Arduino Uno 与 OLED 模块连接图

(3) 总线硬件配置确认

当有多个模块连接在一起时,要确保各个模块的地址没有冲突。模块地址的数值一般可通过查阅数据手册获知,如果数值是可变的,还需要对照实物确认实际的数值。该 OLED 模块缺省的电阻焊接位置所对应的是默认的 I²C 地址,数值为 0x78。

(4) 显示驱动库添加

点击 Arduino IDE 主界面的“项目→加载库→管理库”，用“U8g2”搜索找到库后，点击安装，可以完成显示驱动库的添加，如图 3.12 所示。



图 3.12 U8g2 库的搜索添加

(5) 例程运行

从 U8g2 的示例程序中选择例程 U8g2logo，下载后屏幕显示如图 3.13，这表明硬件连接成功，程序下载正确。

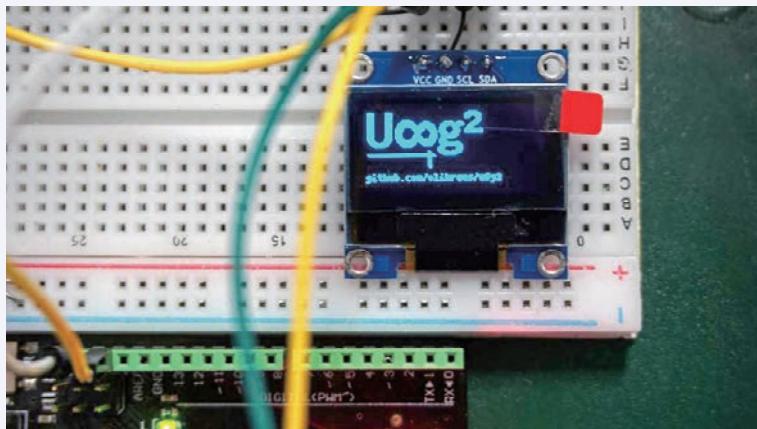


图 3.13 U8g2logo 例程的显示结果

(6) 关键程序

```
# include<U8g2lib.h> //加载 U8g2 库
U8G2_SSD1306_128X64_NONAME_F_SW_I2C u8g2(U8G2_R0, A5, A4, U8X8_PIN_NONE);
void setup() {
```

```
u8g2.begin();
u8g2.enableUTF8Print();      //开启 UTF-8 支持
u8g2.setFont(u8g2_font_unifont_t_chinese2); //调用中文字库
u8g2.setFontDirection(0);
u8g2.clearBuffer();         //清 OLED 屏幕
u8g2.setCursor(0,15); //设定字符的左下角位置
u8g2.print("Hello OpenSource"); //输出要显示的内容
u8g2.setCursor(0,40); //设定字符的左下角位置
u8g2.print("开源,你好");
u8g2.sendBuffer(); //在 OLED 屏幕上显示提示信息
delay(5000); //欢迎词显示 5s
}
```

```
void loop() {
u8g2.setFont(u8g2_font_unifont_t_chinese2);
u8g2.setFontDirection(0);
u8g2.clearBuffer();
u8g2.setCursor(0,40);
u8g2.print("每天活动一小时");
u8g2.sendBuffer();
delay(5000);
}
```

【调试优化】

(1) 运行结果

程序正常运行后, OLED 显示屏显示欢迎词, 如图 3.14 所示。



图 3.14 OLED 屏显示欢迎词



图 3.15 OLED 屏显示提示信息

5 s 后, OLED 屏显示正常的提示信息, 如图 3.15 所示。

(2) 自定义汉字字库

U8g2 字库除了能显示各种字符外还能显示少量中文(最多几百个字),可以根据需要选用,也可以自建字库。自建字库的好处是可以只把需要显示的汉字放入字库中,节约闪存空间,留下更多的空间存放程序。

作业练习

利用已经连接好的 Arduino Uno 开发板与 OLED 模块,再外接一个按键,要求当按键按下时,OLED 屏上显示自己喜欢的一句格言。

体 验 思 考

1. 查阅资料,比较三种常用的显示器件,填写表 3.4。

表 3.4 三种常用显示器件比较

| 名称 | 显示效果 | 应用场景 | 特点 |
|------|------|------|----|
| 液晶 | | | |
| 数码管 | | | |
| OLED | | | |

2. 计步器可以计量人体运动步数,再通过统计距离、速度、时间等数据,进而估算卡路里或热量消耗,可用来控制运动量,防止运动不足或运动过量。计步器大多用到了三轴加速度传感器,可通过公式“消耗的卡路里= 体重(kg) × 距离(km) × 运动系数(k)”来计算相关量,其中跑步的运动系数大于走路,走路的运动系数又大于骑自行车。

- (1) 通过编程制作一个运动卡路里计算器,并结合自身情况用该计算器估算一天消耗的卡路里。
- (2) 查找人体必需营养物质的单位热量,估算自己一天摄入的卡路里,并与(1)中估算出的数值比较,判断摄入的卡路里和消耗的卡路里是否平衡。

三、运动助手·给钥匙打电话

日常生活中我们会遇到需要找一些物件的场合,比如,某些物品可能被我们随手放置,一时间难以找到;又比如,我们在外出旅行时可能带了很多的行李,每件行李都照顾周全是件费心力的事情;再如,某时刻我们想找手机,却怎么也想不起来放在了哪里。将无线信道通信的蓝牙模块和开源硬件设计结合起来,类似的这些事情都可以很好地解决。

技术支持

蓝牙通信

蓝牙是一种无线通信标准,可实现模块间、设备间的短距通信,在无需许可的 2.45 GHz 开放频段工作。蓝牙技术具有功耗低、速率大、支持多设备连接等特点,在计算机、电信、网络与消费性电子产品等领域得到了广泛的应用。

蓝牙模块并没有集成到 Arduino Uno 的微控制器中,使用 Arduino 时需要外接该模块,其作用是将无线信道所接收的数据发送到微控制器或反之。蓝牙模块有无线蓝牙通信和有线通信两个接口,因此,它既可以通过无线通信接口,也可以通过有线通信接口,与微控制器展开信息交换,而这个接口一般是 UART。

图 3.16 所示的是一个技术设计方案的草图。图中有蓝牙接口的智能手机通过无线信道完成与蓝牙模块的信息交互,遵守的是蓝牙协议;与此同时,蓝牙模块通过 UART 接口连接到 Arduino Uno 开发板,遵守的是 UART 接口协议。微控制器完成信息交互后根据结果操控受控器件。

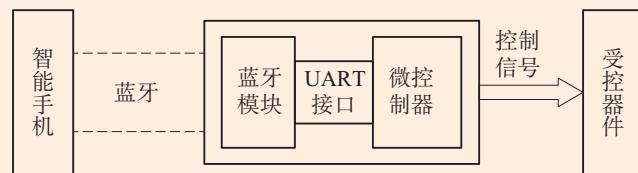


图 3.16 智能手机控制示意图

体验思考

1. 观察身边的事物,有哪些使用蓝牙来进行连接的设备?
2. 体验蓝牙通信的性能边界,自由选择以下内容做实验,记录并分析实验数据。
 - (1) 蓝牙通信距离测试:完成蓝牙耳机与手机的连接,在手机上播放音乐,分别在教室内空地、教室走廊、相邻或间隔的教室内等不同距离进行测试,记录所听到音乐的音量,直到音乐完全听不见。分析记录的数据,将其绘制成曲线图并作解释。
 - (2) 蓝牙天线体验:完成蓝牙耳机与手机的连接,在手机上播放音乐,用各种不同材质、不同密封程度的盒子等包裹或放置手机,直到播放的音乐完全听不见。分析实验结果并据此设计一个独特的蓝牙信号屏蔽方案。
 - (3) 蓝牙信道传输速率测试:把两部已经用蓝牙连接的手机拉开不同的距离,传输同样数据大小的文件,记录完成时间。分析记录的数据,将其绘制成曲线图并作解释。

你是否有过这样的经历,刚刚才看到的钥匙,怎么就找不到了?这个时候多想给钥匙“打个电话”,让钥匙发出铃声回应。通过蓝牙模块与开源硬件,实现用手机找钥匙的功能。

【需求分析】

可以先做一个能挂在钥匙串上的小挂件,小挂件在需要时能够发声和发光,方便定位和寻找。

【功能设计】

在钥匙串的小挂件上安装蓝牙接收模块,手机可使用内置的蓝牙与其连接,可向其发送命令激活发声和发光的功能。示意图如图 3.17 所示。

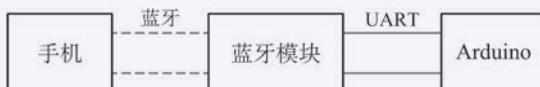


图 3.17 手机控制 Arduino 示意图

【软硬件设计】

(1) 项目器材

蓝牙模块、Arduino Nano、连接线、智能手机、万用表、示波器。

(2) 硬件模块搭建

按图 3.18 连接好 Arduino Nano 和蓝牙模块,通电正常后,蓝牙模块上的指示灯会以 0.5s 的时间间隔闪烁。

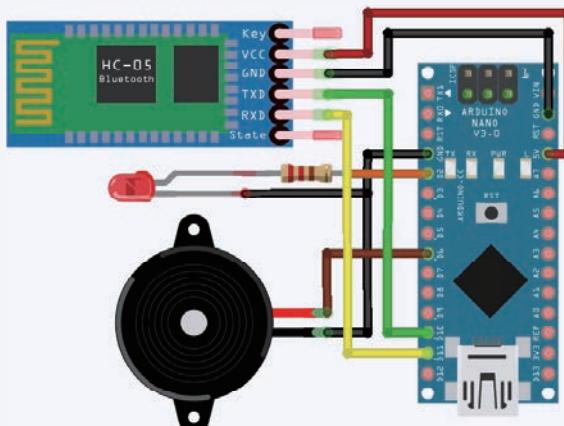


图 3.18 蓝牙模块与 Arduino Nano 连接图

通过手机自带的蓝牙模块,搜索到与 Arduino Nano 连接的蓝牙模块。配对成功后,建立起手机与 Arduino Nano 的信息通道。

(3) 流程示意图

“给钥匙打电话”程序流程示意图如图 3.19 所示。

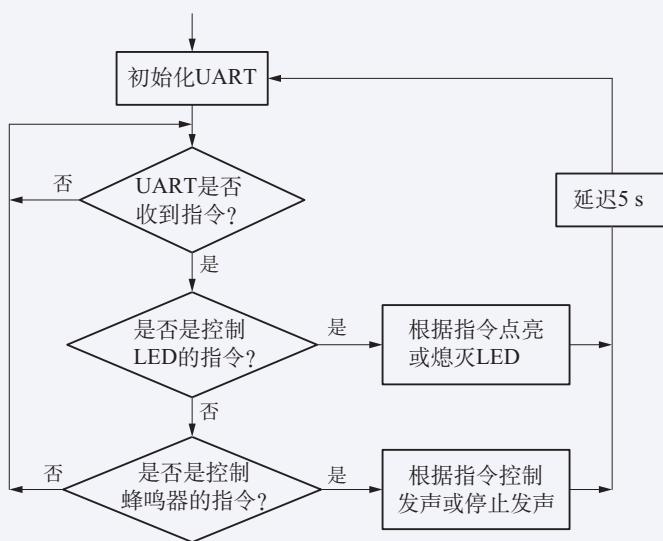


图 3.19 “给钥匙打电话”程序流程示意图

(4) 关键代码

编写程序，完成编译和下载。在 Arduino Nano 的串口上，接收手机通过自带的蓝牙模块发送的控制信息。关键程序如下：

```

#include <SoftwareSerial.h>
#define ledPin 2
#define buzPin 6
int val;
SoftwareSerial BT(10, 11); //D10 为 RX, 接 HC05 的 TXD, D11 为 TX, 接 HC05 的 RXD

void setup() {
    Serial.begin(38400); //硬件内置串口
    Serial.println("BT is ready!");
    BT.begin(38400); //软件模拟串口
    pinMode(ledPin, OUTPUT);
}

void loop() {
    if(Serial.available()) {
        val = Serial.read();
        BT.print(val); //Arduino 向手机传送数据
    }

    if(BT.available()) { //手机向 Arduino 传送数据

```

```

val = BT.read();
Serial.print(val);
if(val == '1') { //手机向蓝牙发送字符“1”，命令蓝牙点亮 LED
    BT.print("LED_ON"); //蓝牙接收命令，并向手机发送反馈信息
    digitalWrite(ledPin, HIGH); //LED 点亮
    delay(2000);
}
else if(val == '2') { //手机向蓝牙发送字符“2”，命令蓝牙奏响蜂鸣器
    BT.print("BUZ_ON"); //蓝牙接收命令，并向手机发送反馈信息
    tone(buzPin, 1000, 1000); //蜂鸣器奏响 1 秒钟
}
else {
}
digitalWrite(ledPin, LOW);
}
}

```

(5) 发送指令

打开手机上的蓝牙控制 App, 选择要找的设备, 发送相应的控制指令, 就可以实现找钥匙的功能, 如图 3.20 所示。



图 3.20 手机 APP 控制指令及响应

【调试优化】

- (1) 手机通过 App 发送指令后, 用万用表观察连接 LED 灯的 Arduino Nano 的 D2 端口的电平情况。
- (2) 手机通过 App 发送指令后, 用示波器观察连接蜂鸣器的 Arduino Nano 的 D6 端口的信号波形。

作业练习

1. 查阅关于 Wi-Fi 模块的资料,想一想: 如何利用 Wi-Fi 模块实现开关的远程控制?
2. 设计一个利用开源硬件完成的方案设计,当手机离开你一定的距离时能够自动地发出提示,提醒你不要遗落手机。

探究活动

【需求分析】

随着人们对健康越来越重视,越来越多的人加入到锻炼的行列中。长跑作为一项深受人们喜爱的活动,参与的人数非常多,这同时也对活动的安全性提出了更高的要求。特别是在长距离、高强度的锻炼或比赛中,需要为活动者或比赛选手提供身体状况的检测、警示,减少意外的发生。这样一方面让使用者实时知晓身体状况数据,自行调整运动量,另一方面,使用者如果出现身体不适,又能够及时报警,尽快获得帮助。

【方案设计】

利用开源硬件并结合脉搏监测传感器,设计一个可以自动测量心率的检测仪: 使用者在正常运动时,它可以实时显示数据,供使用者参考决定后续的运动量; 当数据超过设定的安全值时,它会及时提醒; 当使用者出现身体不适等紧急情况时,它可以通过“一键求助”的方式将位置信息发送到安全中心,方便救护人员施救。

参考如图 3.21 所示的方案设计图,完成一个具有脉搏监测和紧急求助功能的运动助手的方案设计,论证方案可行性,并绘制详细的实物连接图。

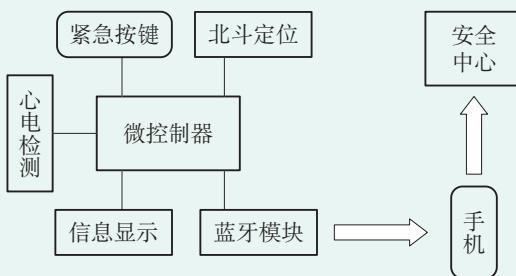


图 3.21 运动助手示意图



第四章

开源硬件项目设计实践

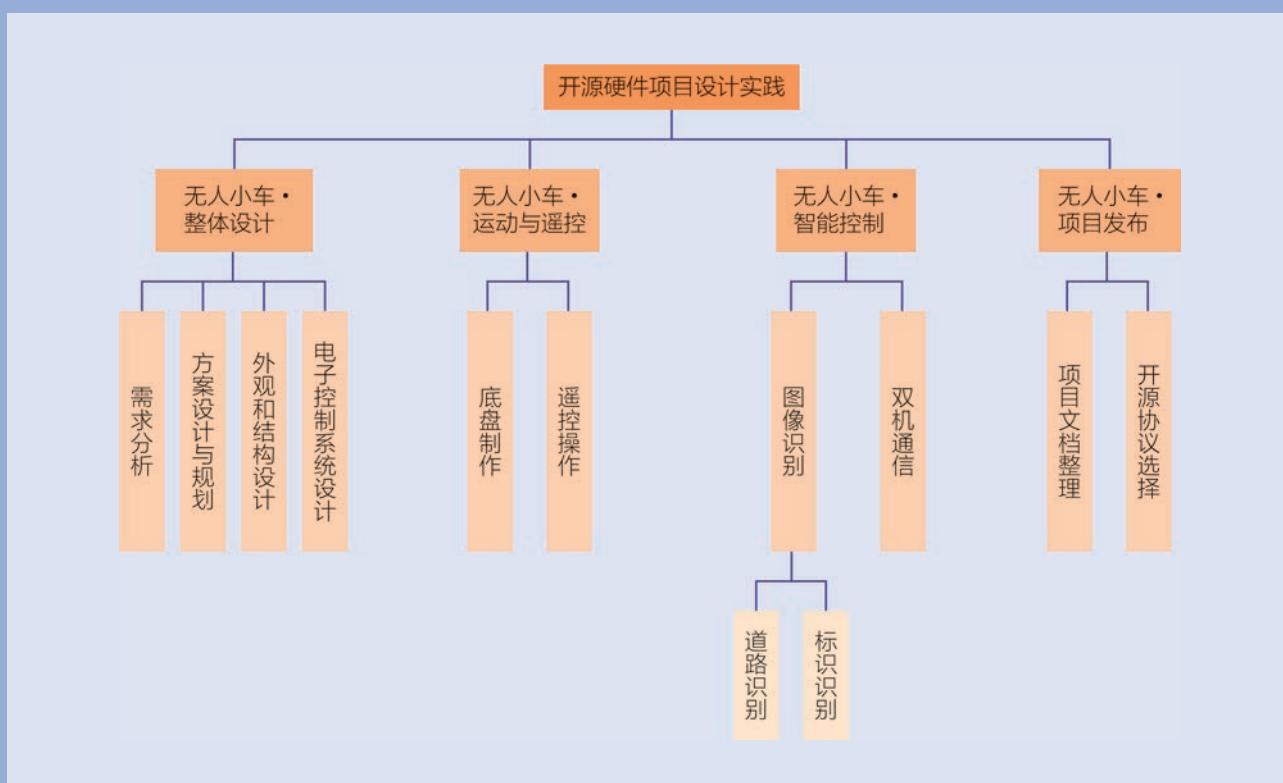
本章学习目标

-
- 了解基于开源硬件的自动驾驶小车的设计方案,理解自动驾驶小车各组成部分的功能及相互间的连接关系,能根据要求选择合适的开源硬件型号。
 - 会使用开源硬件的设计工具和编程语言,能根据设计方案实现自动驾驶小车运动控制和图像识别等功能。
 - 掌握开源协议的运用,能将自动驾驶小车项目发布至开源社区。
-

开源硬件中各类扩展模块的连接和组合,使人们可以用更灵活多样的方式进行创造。各类开源硬件模块结构不同、性能不同,应用场景也不同,它们之间通过合适的组合与连接,进行合作,能充分发挥各模块的优势,从而形成一个功能更强大的整体。较为复杂的开源硬件项目,往往需要对更多类型的开源硬件进行组合,编制更复杂的程序,制作难度更高的产品造型,进行更细致的调试。在这种情况下,开源硬件项目实施的五个步骤,可以支持项目有计划地推进和实施。作品的创意设计和开发制作往往不是一蹴而就的,而是需要不断优化和迭代。

同学们在本章的学习中,将完成一个以自动驾驶为主题的开源硬件项目的设计和实践,学习开源硬件项目设计的方法,体验开源硬件项目设计的完整流程。在设计与制作的过程中,同学们要结合实际情况,进一步思考:开源硬件项目的限制条件有哪些?如何在此基础上进一步明确需求?产品如何不断优化和迭代?在作品制作过程中,如何进行安全的操作?

本章知识结构



项 · 目 · 情 · 境

随着开源硬件的日益普及,开源硬件作品的制作渐渐地从原来的简单小系统向更为复杂的系统,从单个独立系统向多个系统相互配合的更高级系统的方向发展。电子技术、智能控制和人工智能等技术快速发展使得越来越多的先进技术运用在汽车领域,自动驾驶汽车能够辅助驾驶员驾驶车辆,甚至实现自动驾驶。

从物流小车到货运重卡,自动驾驶汽车将被应用在更广泛、更复杂的场景中,从而实现新兴的智能交通,构建起新型的未来城市。也许某一天,当你去取快递的时候,看到送快递的是一辆自动驾驶送货小车,或者,你在打开出租车门的时候,发现车上并没有司机……自动驾驶是一个非常复杂的问题,需要环境感知、行为决策、路径规划和运动控制等技术的综合运用。如何制作一个能在限定场景条件下自动驾驶的无人小车?如何模拟无人小车的驾驶环境?如何实现小车的运动控制和图像识别功能?

本章我们将首先通过设计一个较为简单的基于 Arduino 的遥控小车完成初步的作品制作,体验开源硬件项目设计的流程和方法;在此基础上,用仿真道路模拟遥控小车的驾驶环境,为遥控小车增加一个专门用于道路识别和标识检测的基于树莓派的模块,实现双机通信,完成制作后通过开源社区分享作品。

项 · 目 · 任 · 务

任务 1

制作需求列表、功能列表、外观模型,依次完成基于开源硬件的无人小车的需求分析、功能分析、外观设计。

任务 2

利用开源硬件 Arduino 完成小车的组装,进而实现小车的前进、后退、转弯等运动控制功能;用树莓派实现图像识别功能,进而实现智能控制。

任务 3

整理、归档无人小车项目中的各类文件,按照规范将项目发布至开源社区。

第一节 无人小车·整体设计

真实的马路交通情境是复杂多变的，本项目将对一个简易的模拟道路进行需求分析。该模拟道路包含一个仿真道路、“右转”标志牌、“停止”标志牌和红绿灯。项目的目标是使小车能根据标识前进、右转、停止等，控制小车的方式包括人工或自动。

体验思考

除了上述交通标志外，你能列举出自动驾驶小车还有哪些物体需要识别？

一、需求分析

在本项目中，自动驾驶小车实际是一辆装上传感器和智能控制系统的电动小车。我们将制作一个可不断优化的自动驾驶小车，首先完成蓝牙控制，然后再以图像识别等人工智能的方式进行运动控制。完整的需求列表如表 4.1 所示：

表 4.1 自动驾驶小车需求列表

| 核心需求 | 小车在平地上启动、向前行驶、右转 | 采纳 |
|------|------------------|---------|
| 附加需求 | 使用平板电脑通过蓝牙遥控小车 | 采纳 |
| | 识别道路、“停止”交通标志 | 设计迭代时采纳 |
| | 使用语音控制小车 | 忽略 |
| | 小车刹车、倒车 | 设计迭代时采纳 |

在做项目的需求分析时，需要了解各类微控制器的特点。从各种微控制器的优势着手，准确地设计出技术方案。本项目中树莓派能够支持自动驾驶小车的图像识别功能，而 Arduino 不支持，因此需要使用树莓派构建小车的智能控制系统；树莓派虽然也支持小车的动力系统，但不如 Arduino 连接动力系统方便，因此在运动控制部分选择使用 Arduino。

二、功能设计

根据小车的需求分析详细设计功能。首先搭建一个遥控小车,通过人工发送“右转”“停止”等指令,控制小车在仿真道路上的行驶;之后在此基础上搭建人工智能自动驾驶小车,使其在仿真道路上能自动地根据交通标志进行判断,完成“右转”等任务。功能设计如表 4.2 所示。

表 4.2 自动驾驶小车功能列表

| 功能项 | 要 求 | |
|-----------|--|--|
| 小车的基本架构 | | |
| 尺寸 | 要满足应用场景的要求和限制 | |
| 结构 | 既要有一定的承载,又具有较好的灵活拆卸性 | |
| 运动 | 既要便于控制,又满足运动的灵活操控 | |
| 通信方式 | 当小车与控制设备距离较近,且设备间干扰较少时,可以选用蓝牙通信;当距离较远时,可以选用 Wi-Fi | |
| 电池 | 在选择电池的时候,需要考虑电池的能量密度,能量密度越大,则续航时间越长,体积越小,可选用锂电池作为系统的动力电池 | |
| 电机 | 在选择电机的时候,需要考虑电机的扭矩,扭矩越大,则负载越大 | |
| 小车的两种控制体系 | | |
| | “人工”控制 | “智能”行驶 |
| 小车控制设备 | 由操控者下达指令,指令发送到小车后,再在小车的主控模块控制下,完成指定的运动 | 将信息传送到控制设备后,由控制设备根据这些信息,采用智能的方式,经过运算处理后,自动驾驶 |
| 人机交互设备 | 小车信息的显示要清晰、准确,可以选用液晶显示屏或 OLED 屏;同时,下达指令的按键表示要清晰、易懂 | 在“智能”的运动模式下,交互的方式可以选用语音交互方式 |

体 验 思 考

根据上述需求,请通过查找资料、上网搜索等方式,选择合适硬件和对应型号,并给出选择的原因。在选择硬件的过程中,有哪些性能因素需要考虑?请根据搜索到的硬件情况,填写表 4.3。

表 4.3 自动驾驶小车选材表

| 功能模块 | 硬件及型号 | 选择原因 |
|----------|---------|------|
| 小车动力主控模块 | Arduino | |
| 小车智能处理模块 | 树莓派 | |
| 动力系统 | | |
| 人机交互设备 | | |

三、外观设计

外观设计包括自动驾驶小车的形状、图案、色彩的设计，其中形状设计是指各个3D零件的造型设计，需要充分考虑小车的内部构造，确保其能容纳所要用到的电子模块，包括各类电路板和传感器等。图4.1是一辆创意自动驾驶小车的效果图。



图4.1 小车效果图

四、电子控制系统设计

根据功能设计，进一步实现小车的整体制作。设计方案包括上、下位机的选择，各个模块与上、下位机的连接等。

将Arduino、通信模块、电机驱动模块等拼接在小车上，构成下位机，完成场地信息的获取、传递、执行运动指令等功能。用Arduino控制板连接多种传感器，比如红外传感器、超声波传感器等，以获取场地信息；通过Arduino控制板连接蓝牙模块、Wi-Fi模块、信号连接线等，以实现通信；将Arduino控制板连接电机驱动模块等，使小车能执行运动指令。

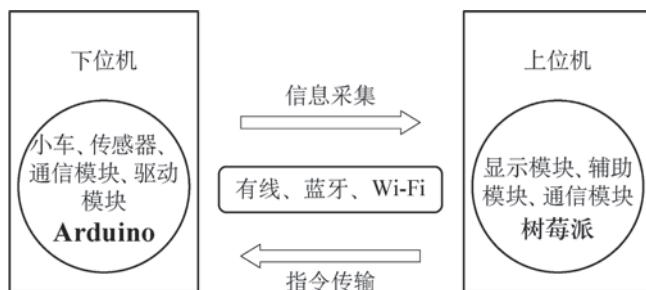


图4.2 小车控制实现示意图

树莓派与语音识别模块、图像识别模块、蓝牙模块、显示模块组成运动小车的控制设备,构成上位机,可以接收下位机传送的信息。根据下位机传送的信息和上位机采集的图像及语音等信息,进行判断和处理(人工或自动)后,将结果或后续的小车运动指令传送给下位机。

通过上、下位机的协调配合,小车既能由操控者控制,又能自动驾驶。同学们可以循序渐进,依次按需求进行设计、制作。

作业练习

- 根据小车的设计方案,你能选择适合小车的通信方式吗?请填写表4.4。

表4.4 自动驾驶小车通信方式的选择

| 通信方式 | 能否作为小车的通信方式 | 请选择其一作为通信方式,并写下选择原因 |
|------------------|-------------|---------------------|
| 串口通信 | | |
| 蓝牙 | | |
| Wi-Fi | | |
| I ² C | | |
| | | |

- 语音指令是一种应用场景十分广泛的控制方式。你能说出语音指令控制的优点和缺点吗?请填写表4.5。

表4.5 语音指令控制的优缺点

| 优 点 | 缺 点 |
|-----|-----|
| | |
| | |

第二节 无人小车·运动与遥控

遥控小车曾给同学们的童年带来许多的欢乐和惊喜。在这一节中,我们首先利用开源硬件及一些模块,制作一个运动的小车,让它在程序的控制下,完成“前进”“后退”“转弯”等动作;接着利用平板电脑,通过蓝牙与小车实现信息的交互,小车既可以接收平板电脑发出的指令,完成指定的动作,又可以将运动状态等信息回传至平板电脑。通过这两个部分的实践,制作一个遥控小车,为后续制作自动驾驶小车做准备。

体验思考

如果要制作一个具有运动控制功能的小车,应该如何规划小车的运动控制?运动控制由哪些部分组成?

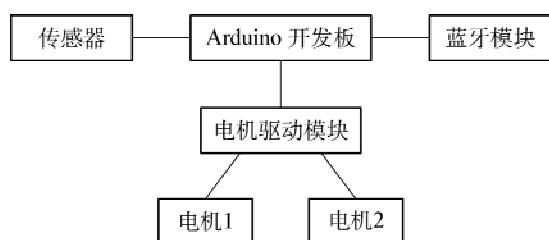


图 4.3 运动小车的系统框图

为实现上述功能,制作遥控小车所需的器材有:Arduino 开发板、电机驱动模块、电机、蓝牙模块、传感器、车轮、车架等。从整个信息处理系统的组成上看,小车既是一个传感器的载体——利用安装在小车上的多个传感器获取需要的信息,又是一个信息处理系统功能的体现者——在控制器的控制下完成指定的运动。运动小车的系统框图如图 4.3 所示。

一、运动功能



图 4.4 运动小车底盘图

小车的主体包括车架、车轮、固定件等,控制部分由 Arduino 开发板及电机控制板组成,其他还包括电机、电池、连接线等。运动小车底盘(如图 4.4 所示)的 3 个轮子中的两个后轮分别与两个电机相连,由电机带动轮子的转动,它们作为驱动轮,可以独立控制;前轮是一个万向轮。因为电机可以正转或反转,所以每个轮子可以完成前进或后退动作。

微控制器的 I/O 接口输出不同的信号组合,可以控制每个电机的转动,继而控制连接在电机上的轮子的运动。两个后轮的运动组合可以形成小车的各种运动(如表 4.6 所示)。

表 4.6 轮子转动与小车运动对应关系表

| 编号 | 左后轮 | 右后轮 | 小车运动 |
|----|-----|-----|---------|
| 1 | 前进 | 前进 | 小车直线前进 |
| 2 | 后退 | 后退 | 小车直线后退 |
| 3 | 前进 | 停止 | 小车向右转弯 |
| 4 | 停止 | 前进 | 小车向左转弯 |
| 5 | 前进 | 后退 | 小车顺时针转圈 |
| 6 | 后退 | 前进 | 小车逆时针转圈 |

体验思考

如果小车有 4 个轮子,且 4 个轮子均可以独立控制,分析一下轮子转动与小车运动的对应关系。

项目实践

【项目器材】

Arduino 开发板、Arduino 扩展板、电机驱动模块 L298N、超声波传感器、光电巡线传感器、车体、车轮、电池。

【操作步骤】

1. 将各种部件正确组装成一辆运动小车,其中包括底盘、轮子、电机等的安装。
2. 在制作完成的小车的基础上,将 Arduino 开发板、电机驱动模块、传感器、电池等安装到小车上,如图 4.5 所示。

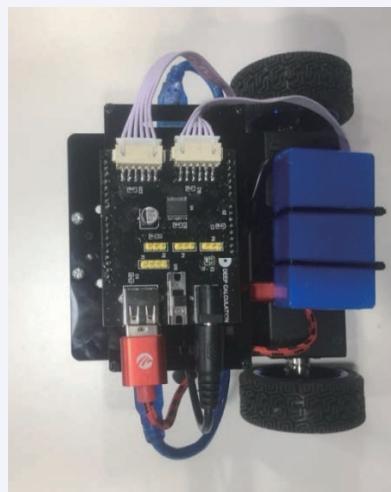


图 4.5 运动小车的完整组装

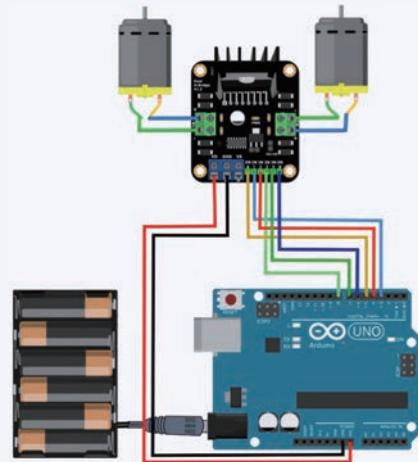


图 4.6 Arduino 与电机驱动连接图

3. 各模块按图 4.6 所示完成电路连接。
4. 编写小车运动控制程序。

【关键代码】

```
void carStop() { //自定义小车停止函数
    digitalWrite(L298N_IN1, LOW);
    digitalWrite(L298N_IN2, LOW);
    analogWrite(L298N_ENA, 0);

    digitalWrite(L298N_IN3, LOW);
    digitalWrite(L298N_IN4, LOW);
    analogWrite(L298N_ENB, 0);

}

void carAdvance(int leftSpeed, int rightSpeed) { //自定义小车前进函数
    digitalWrite(L298N_IN1, HIGH);
    digitalWrite(L298N_IN2, LOW);
    analogWrite(L298N_ENA, leftSpeed); //控制小车前进速度

    digitalWrite(L298N_IN3, HIGH);
    digitalWrite(L298N_IN4, LOW);
    //为保持小车直行前进,参数 rightSpeed 和 leftSpeed 应该一样
    analogWrite(L298N_ENB, rightSpeed);
}

void carTurnLeft(int leftSpeed, int rightSpeed) { //自定义小车左转函数
    digitalWrite(L298N_IN1, LOW);
    digitalWrite(L298N_IN2, LOW);
    analogWrite(L298N_ENA, 0);

    digitalWrite(L298N_IN3, HIGH);
    digitalWrite(L298N_IN4, LOW);
    analogWrite(L298N_ENB, rightSpeed);

}
```

体 验 思 考

1. 以上程序实现了小车的停止、前进、左转,那么要实现小车的其他运动方式该如何编写程序?
2. 若在一辆正常向前运动的小车前面放置一块障碍物,如何利用安装的传感器,设计相应的程序,使得小车能够在障碍物前自动停下来?

二、遥控功能

小车的遥控可以采用蓝牙或 Wi-Fi,当控制距离不远时,蓝牙是一种常用的通信方式,可实现平板电脑与小车的信息交互。利用平板电脑自带的蓝牙模块与遥控小车上的蓝牙模块进行通信,由平板电脑发出指令,控制小车的运动,小车的相关信息(包括传感器测得的数据)也可以通过蓝牙回传到平板电脑。

蓝牙模块与微控制器通过串口连接完成通信,微控制器向串口发送数据,通过蓝牙模块就可以将信息发送到对应的平板电脑上;微控制器以读取串口数据的方式,得到平板电脑发送来的数据或控制指令。遥控小车通信示意图如图 4.7 所示。

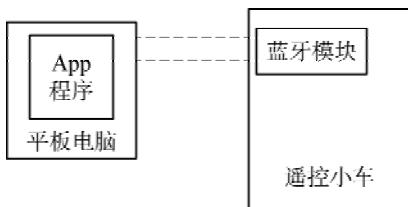


图 4.7 遥控小车通信示意图

项目实践

在已经制作好的运动小车的基础上增加蓝牙模块,平板电脑通过蓝牙模块对小车进行遥控。

【项目器材】

Arduino 开发板、Arduino 扩展板、电池、蓝牙模块、车体、超声波传感器、光电巡线传感器、车轮。

【项目步骤】

1. 如图 4.8 所示,将蓝牙模块连接在安装好的运动小车上。
2. 安装 App 到平板电脑中。

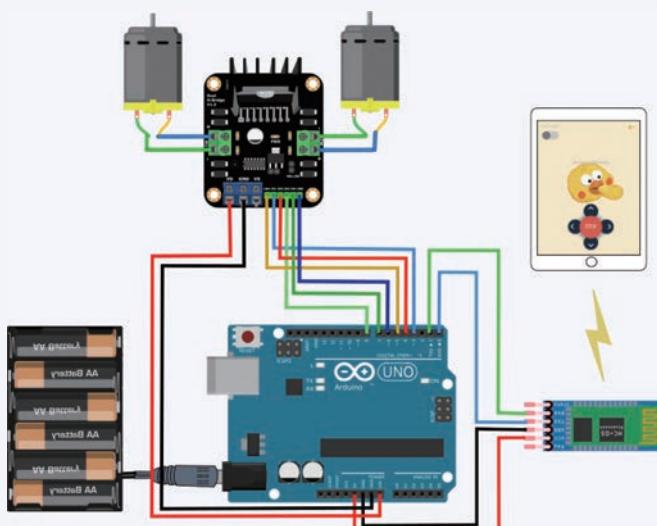


图 4.8 运动小车与蓝牙模块连接图

3. 编写蓝牙通信程序。

4. 如图 4.9 所示,将各个部件、模块组装成一辆遥控小车。

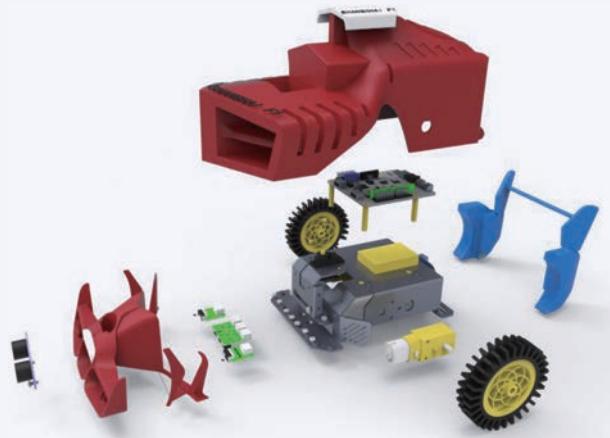


图 4.9 遥控小车的完整组装

【关键代码】

蓝牙接收的程序如下:

```
void setup(){
Serial.begin(115200); //初始化串口并设置波特率为 115200
}

void loop(){
char val;
val = Serial.read(); //微控制器读取串口数据
if(val!= -1){ //当读取到数据
    //执行相应的程序代码(此处具体代码已略去)
}
}
```

蓝牙发送的程序如下:

```
void setup(){
Serial.begin(115200); //初始化串口并设置波特率为 115200
}

void loop(){
Serial.print(val); //将变量 val 对应的数据通过蓝牙发送出去
delay(500);
}
```

本项目中的协议是指通信双方遵循的规则和约定,要使双方协同工作实现信息交换和资源共享,它们之间必须有共同的语言,即交流什么、怎样交流,这样才能让双方互相“听懂”对方。比如,平板电脑与小车可按表 4.7 约定,当遥控小车接收到 000 时,就停止运动。

表 4.7 对遥控小车的约定

| | | | |
|-----|-----|-----|-----|
| 000 | 001 | 010 | 011 |
| 停止 | 前进 | 左转 | 右转 |

体 验 思 考

1. 如果要求平板电脑控制小车运动的速度,比如要求小车以最高速度的一半前进,该如何设计协议?
2. 除了蓝牙,对小车的遥控还可以用什么方式?它们分别在何种条件、何种场合适用?

作业练习

1. 请将小车的组成部件填入表 4.8。

表 4.8 小车组成部件

| 名称 | 部件 | 特点 |
|--------|----|----|
| 小车基本组件 | | |
| 传感器件 | | |
| 通信器件 | | |
| | | |
| | | |

2. 试着设计一个记录小车运动情况的程序,它能将小车的运动数据回传到平板电脑,并记录下来。

第三节 无人小车·智能控制

长久以来,机器不能像人类那样轻松地识别交通标志,以至于自动驾驶研发一直处于瓶颈期。直到近两年,由于人工智能芯片的算力突破和海量数据的累积,自动驾驶迎来了春天。

智能控制是指小车能根据场景信息进行判断,自动完成直行、转弯等运动控制。在第一节的“电子控制系统设计”中提到,Arduino 作为下位机搭载小车的驱动模块,树莓派作为上位机搭载图像识别等模块。我们已经在上一节中解决了小车的下位机运动控制问题,这一节我们将利用上位机,也就是通过树莓派,利用图像识别模块,完成小车的智能控制。

体验思考

自动驾驶是依靠人工智能、视觉计算、雷达、监控装置和全球定位系统协同工作,让控制器在没有任何人操作的情况下自动安全地操控机动车辆。自动驾驶可以让汽车行驶变得更加节能高效,可以让驾驶员从驾驶操作中解放出来,也将对社会经济、生态环保等领域产生巨大的效益。你认为自动驾驶需要哪些图像识别技术?



图 4.10 自动小车图像识别模块

有了图像识别功能,小车就能在一定场景下对图像信息进行逻辑判断,实现真正的“自动”行驶。在之前的系统框图上,新增图像识别模块,如图 4.10 所示。

一、图像识别

自动驾驶小车的图像识别功能有两个重要任务:一是道路识别,即保证小车根据路况安全行驶,在直道上直行,在弯道上转弯,项目将以仿真道路(如图 4.11 所示)为例进行巡线;二是交通标志识别,即让小车“看懂”交通标志,并按标志自动操作,项目将以“停止”标志牌(如图 4.12 所示)为例进行识别。



图 4.11 仿真道路



图 4.12 “停止”标志牌

本节的场景设置与技术支持如表 4.9 所示。

表 4.9 自动驾驶小车场景设置

| 场景 | 说明 | 微控制器 | 核心技术 |
|--------|--------------|------|------|
| 道路识别 | 可按道路直行、转弯 | 树莓派 | 边缘检测 |
| 交通标志识别 | 可识别“停止”等交通标志 | 树莓派 | 目标检测 |

完成这两项任务后,小车就能在道路上巡线行驶,并能自动地根据交通标志作出判断。

项目实践

如果要使自动驾驶小车在道路上安全行驶,那么,怎么用树莓派进行巡线等道路识别呢?

树莓派的巡线功能主要依靠图像识别技术中的边缘检测技术来完成。在检测到道路边缘后,小车可通过算法,保证车体在道路中央行驶。通过接下来的项目,利用自制的交通地图模拟巡线,使小车能自动判别直行、转弯等场景,体验巡线技术。

要完成巡线等操作,需要利用 Python 的图像处理拓展包 OpenCV。OpenCV 具有功能齐全的图像处理函数,可支持树莓派进行巡线操作。

【项目器材】

遥控小车、树莓派 3B+、仿真地图、摄像头、显示器(含 HDMI 接口)、鼠标、键盘、数据线。

【操作步骤】

1. 用 Python 的拓展包 OpenCV 读取外接摄像头的数据,并完成二值化处理,即把读入的彩色图像(如图 4.13 所示)转成灰度图(如图 4.14 所示),方便后续计算。



图 4.13 小车视角下的道路图像

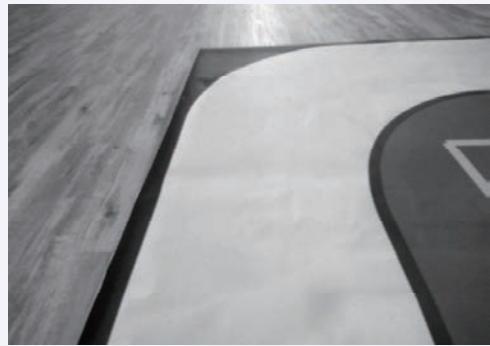


图 4.14 二值化后的道路图像

2. 边缘检测。使用 OpenCV 的边缘检测函数检测车道的边缘, 提取车道信息, 效果如图 4.15 所示。



图 4.15 边缘检测后的道路图像

3. 判断小车是否在道路中央, 若未在中央则进行矫正。
4. 完成矫正后, 利用树莓派向 Arduino 发送前进指令。如果想设置小车以每秒 20 个编码器脉冲的速度前进, 则通过树莓派发送指令给 Arduino, 如“m 20 20”, Arduino 接收到指令后, 运行函数 carAdvance, 进而控制电机的转速。

【关键代码】

```
import serial  
ser_a = serial.Serial('/dev/ttyACM0', 9600) #开启串口  
ser_a.write("m 20 20 ") #树莓派向 Arduino 发送命令“m 20 20”  
ser_a.close() #关闭串口
```

小车在直道、弯道上行驶的过程中需要进行中央矫正处理。当方向与直道不平行时,小车会自动矫正,保持与直道平行行驶。

可以观察到,边缘检测后的仿真道路边缘是白色的。通常的做法是选取图像中的一行像素,计算并判断这一行像素中表示道路的白色像素的中间点(图中蓝色B点)是否与摄像区域的中间点(图中红色A点)重合,由此可获知小车是否在道路中间行驶。如图 4.16 所示,摄像区域中心点 A 在道路中心点 B 的左侧,则让左轮加快转速,使得小车向右偏转一点,从而使 A、B 两点重合,实现道路中央矫正。同理,在弯道时通过类似的算法来完成道路中央矫正。

主要代码如下:

```
#找到白色像素的中心点 B 的位置
center = (index[0] + index[1]) / 2

#计算出 center 与视频区域中心点 A 的偏移量,假设视频区域中心点的位置为 320
direction = center - 320

#若偏差太大,超过 200,则使小车停止
if abs(direction) > 200:
    ser_a.write("m 0 0") #树莓派向 Arduino 发送命令“m 0 0”

#前进
elif direction == 0:
    ser_a.write("m 20 20") #树莓派向 Arduino 发送命令“m 20 20”

#向右矫正
elif direction > 0:
    ser_a.write("m 30 20") #树莓派向 Arduino 发送命令“m 30 20”

#向左矫正
elif direction < 0:
    ser_a.write("m 20 30") #树莓派向 Arduino 发送命令“m 20 30”
```

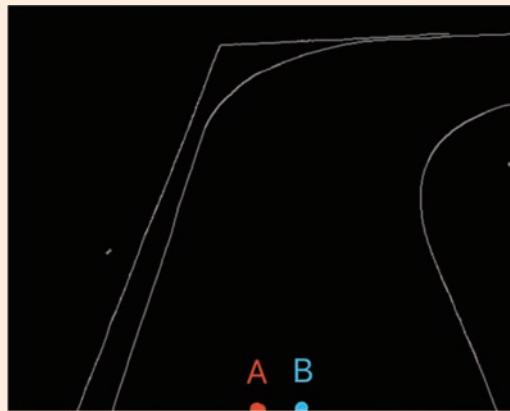


图 4.16 道路中央矫正示意图

自动驾驶汽车的安全行驶需要汽车在自动驾驶的过程中准确识别出交通标志,并作出正确反应,识别交通标志这一能力需要目标检测技术的支持。

目标检测是指找出图像中的目标物体,并确定它们的位置和大小,目标检测对于人类来说并不困难,

人类的视觉系统很容易识别出某个场景里的目标物体，并对物体进行定位。但对机器而言，由于各类物体有不同的外观、形状、姿态，加之成像时的光照、背景等杂乱因素的干扰，目标检测会非常困难。机器需要通过不断地“学习”，才能更好地进行目标检测。

在本项目中，树莓派能满足所有图像处理的技术要求。通过连接摄像头，树莓派可以接收摄像头传输的数据流。那如何才能让机器“知道”眼前的东西是什么呢？通过本项目，体验让机器“看懂世界”的主要技术——目标检测。要完成目标检测，需要利用 Python 的拓展包 TensorFlow。TensorFlow 是开源的深度学习框架，可让人工智能工程师轻松构建各种复杂的算法模型。

【项目器材】

遥控小车、树莓派 3B+、“停止”标志牌、摄像头、显示器(含 HDMI 接口)、鼠标、键盘、数据线。

【实物搭建】

智能小车实物连接如图 4.17 所示，包括显示器、摄像头、鼠标、键盘的连接。

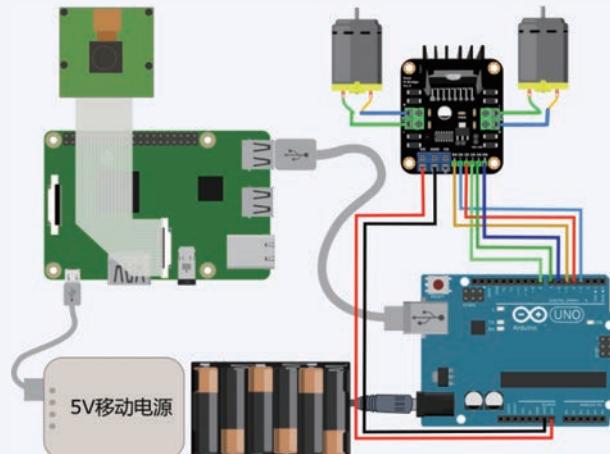


图 4.17 智能小车实物连接图

【操作步骤】

1. 利用手机或电脑收集训练数据，用相关软件做数据标注，并用 Python 开源的深度学习框架训练模型(如图 4.18 所示)。

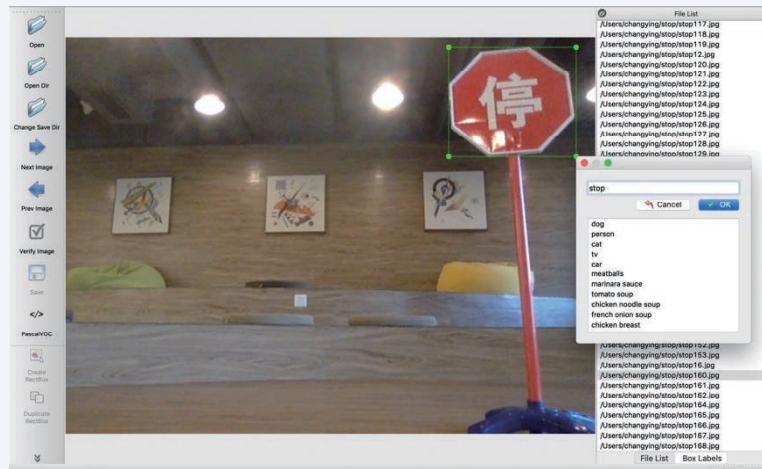


图 4.18 用 Python 开源的深度学习框架训练模型

2. 在树莓派中安装 OpenCV 和 TensorFlow 等, 搭建测试环境。安装完成后, 就能在树莓派中载入拓展包, 测试环境如图 4.19 所示。

```
pi@raspberrypi:~ $ python3
Python 3.5.3 (default, Sep 27 2018, 17:25:39)
[GCC 6.3.0 20170516] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import cv2
>>> import tensorflow as tf
>>>
```

图 4.19 测试环境

3. 利用树莓派的 USB 接口连接摄像头, 在命令行中运行 Python 脚本 detect.py, 测试模型。如果机器识别达到一定的阈值, 则代表标志牌被正确识别。图 4.20 显示的是目标检测结果。



图 4.20 目标检测结果

【关键代码】

```
while(True):
    ret, frame = camera.read() # 读取摄像头数据
    frame_expanded = np.expand_dims(frame, axis=0) # 将摄像头的数据转换为模型可接受的输入
    # 大小
    (boxes, scores, classes, num) = sess.run(
        [detection_boxes, detection_scores, detection_classes, num_detections],
        feed_dict={image_tensor: frame_expanded}) # 运行模型
    cv2.imshow('traffic sign', frame) # 显示图片
```

树莓派接收到摄像数据后, 通过目标检测模型分辨出“停止”标志牌, 向 Arduino 发送操作指令, 如“m 0 0”。Arduino 接收到指令后, 运行事先烧录好的 carStop 函数, 使两个电机的转速为 0, 从而完成“停止”操作。

```
import serial
ser_a = serial.Serial('/dev/ttyACM0', 9600) # 开启串口
ser_a.write("m 0 0") # 树莓派向 Arduino 发送命令“m 0 0”
```

```
ser_a.close() #关闭串口
```

至此,我们完成了图像识别中的目标检测任务。将道路识别与交通标志识别结合在一起,就能实现特定场景下的自动驾驶了。

体 验 思 考

如图 4.21 所示,激光雷达(laser detection and ranging,简写为 LiDAR)凭借其优异性能已成为不可或缺的环境感知传感器。车载激光雷达因其具有可准确获取目标的三维信息、分辨率高、抗干扰能力强、探测范围广、近全天候工作等优点,在智能驾驶环境感知系统中占据重要地位。

请同学们查阅资料,还有哪些能够提升汽车驾驶环境感知能力的技术?



图 4.21 激光雷达

二、双机通信

树莓派与 Arduino 的通信共有两种常见的实现方式,一种为 USB 连接,一种为串口连接。两个开源硬件的连接遵循一般的通信协议,如 USB 协议规范 2.0、RS-232 串口通信协议等。

Arduino 具有多种多样的外部接口,不同于树莓派的 I/O 接口,Arduino 具有模拟输入接口,可以测量 I/O 接口上的模拟值。Arduino 与树莓派进行串口通信的方式一般有两种:一是通过树莓派的 USB 接口进行通信,二是通过 GPIO 接口进行通信。

1. 树莓派与 Arduino 通过 USB 接口进行通信

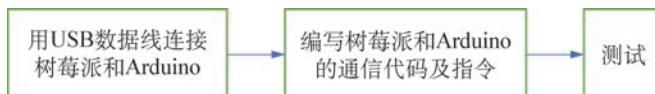


图 4.22 树莓派和 Arduino 通过 USB 接口进行通信的流程图

通过 USB 数据线把树莓派和 Arduino 连接起来，在连接正常的情况下，可以在树莓派终端查看两者连接端口的名称；然后编写树莓派和 Arduino 的通信代码及指令，并完成上传；最终在树莓派上测试通信结果。流程图如 4.22 所示。

(1) 用 USB 数据线连接树莓派和 Arduino

按照流程图，先用 USB 数据线连接树莓派和 Arduino。硬件连接如图 4.23 所示。

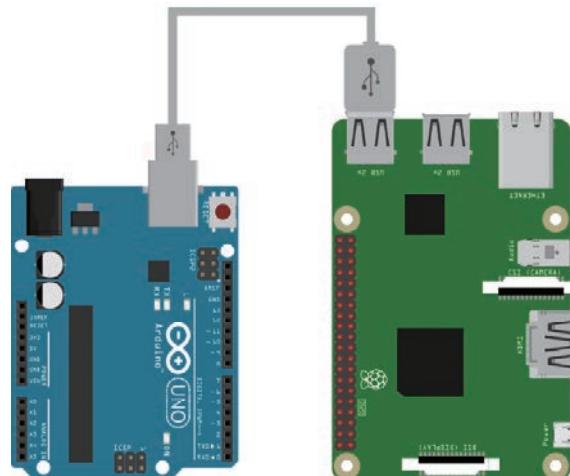


图 4.23 树莓派和 Arduino 通过 USB 接口连接的实物搭建图

(2) 编写树莓派和 Arduino 的通信代码及指令

以 USB 接口连接为例，展示核心测试代码。测试内容为树莓派向 Arduino 发送字符串“Hello Arduino”，若 Arduino 回复树莓派“Hello Raspberry”，则表示连接成功。

(3) 主要代码

① Arduino 烧录程序

```
void setup()
{
    Serial.begin(9600); // 指定数据传输速率为 9600 bps
```

```

}

void loop()
{
    if(Serial.available())
    {
        if("Hello Arduino" == Serial.read())// 接收树莓派传来的字符串,进行条件判断
            Serial.println("Hello Raspberry");// 回复树莓派
    }
}

```

② Python 脚本测试树莓派与 Arduino 的通信

```

import serial
ser_a = serial.Serial('/dev/ttyACM0', 9600) #开启串口
ser_a.write("Hello Arduino ") #树莓派向 Arduino 发送字符“Hello Arduino”
receive = ser_a.readline() #树莓派读取 Arduino 回复的字符串
print(receive) #打印 Arduino 回复的内容
ser_a.close() #关闭串口

```

③ 测试

若连接树莓派的屏幕上显示了“Hello Raspberry”,则说明连接成功;否则,需逐行排查错误。

2. 树莓派和 Arduino 通过 GPIO 接口进行通信

树莓派与 Arduino 通过 GPIO 接口进行通信的流程及串口连接分别如图 4.24 和图 4.25 所示。



图 4.24 树莓派和 Arduino 通过 GPIO 接口进行通信的流程图

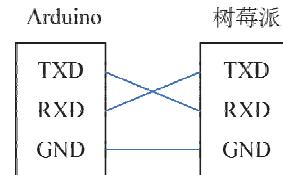


图 4.25 树莓派和 Arduino 的串口连接示意图

通过 GPIO 接口,能够实现 Arduino 与树莓派的通信。首先将 Arduino 接收的数据传输到树莓派中进行处理,再由树莓派发送命令给 Arduino,并根据规定的通信协议执行命令。这样就充分利用了两

种开源硬件的优势,使二者相辅相成,协同完成智能驾驶任务。树莓派与 Arduino 通过 GPIO 接口连接如图 4.26 所示。

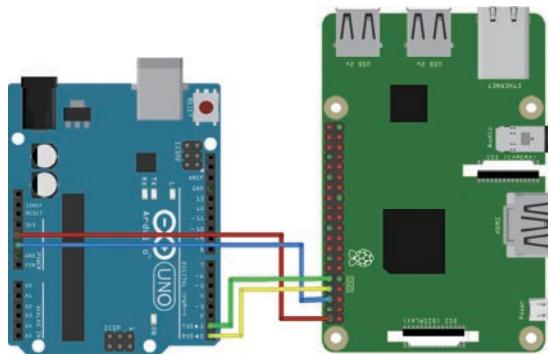


图 4.26 树莓派与 Arduino 通过 GPIO 接口连接的实物搭建图

探究活动

下位机 Arduino 和上位机树莓派通过串口通信建立信息交互通道。树莓派发送指令给 Arduino, Arduino 执行相应的动作或者返回传感器信息给树莓派,这需要制定双方共同遵守的通信协议。这类协议可以复杂也可以很简单,一般以满足需要并适当考虑使用场合的拓展为宜。可靠性要求高的场合还需要对发送和接收的数据做校验,以保证数据的完整性。

假设采用以下的通信协议:

- (1) 命令以字符串的形式发送,命令和数据间用空格分割。
- (2) 命令由三个字段组成,第一个字段是一个字母长度的命令位,第二个字段为数据位 1,第三个字段为数据位 2。

示例如下:如果想从下位机模拟引脚 A3 中读取数据,则发送指令“a 3”,如果想设置小车以每秒 20 个编码器脉冲的速度前进,则发送指令“m 20 20”。这里的“a”“m”等指令均可在 Arduino 上自定义。

制定表 4.10 所示的指令说明表,以实现所要求的控制命令。

表 4.10 指令说明表

| 指令 | 命令位(可自定义) | 数据位 1(可自定义) | 数据位 2(可自定义) |
|-------------|-----------|-------------|----------------------|
| 读取 AD 接口的值 | ‘a’ | AD 接口的引脚号 | 无 |
| 设置 I/O 接口的值 | ‘c’ | I/O 接口的引脚号 | 0 或者 1,0 为低电平,1 为高电平 |
| 读取小车的速度 | | | |
| 设置小车的速度 | | | |
| 小车左转 | | | |
| 小车右转 | | | |

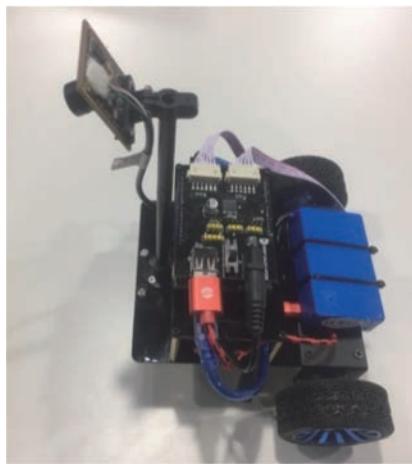


图 4.27 自动驾驶小车

我们在上一节的运动小车的基础上搭载了树莓派,再连接摄像头,就能完成最终的自动驾驶小车了。图 4.27 为拼装完整的实物图。

接下来对小车进行调试。测试与调试自动驾驶小车时,经常会遇到一些制作过程中无法想象的问题。例如,在调试过程中经常碰到小车无法正确地执行指令,小车无法正确转弯等问题。那么通过什么方式能解决上述类似的问题?

首先需要知道引发问题的模块有哪些,并进行逐一排查。然后试着借鉴前人的经验解决问题,借鉴的方式有很多,可以翻阅书籍,或利用互联网上的搜索工具搜索,也可以在开源社区里提问。最后根据获得的结果,结合自身所遇到的问题,有针对性地进行调试。

调试过程通常分为以下几步:

- (1) 针对发生问题的模块进行断点调试。
- (2) 通过打印的方式判断错误的类型。
- (3) 重新运行代码,查看修改后的代码是否会引发新的问题。

其中,断点调试是一种很实用的排查错误的技巧。断点调试即在程序的某一行设置一个间断点,程序运行到这一行时就会停止,之后依次往下对每一行进行调试,如果报错,则打印出程序中变量的值进行分析,并根据错误类型修改代码。我们以图像识别的代码为例,其断点调试的方式如图 4.28 所示。

```
131     for frame1 in camera.capture_continuous(rawCapture, format="bgr", use_video_port=True):  
132  
133         frame = frame1.array  
134         frame.setflags(write=1)  
135         frame_expanded = np.expand_dims(frame, axis=0)  
136  
137         ● (boxes, scores, classes, num) = sess.run(  
138             [detection_boxes, detection_scores, detection_classes, num_detections],  
139             feed_dict={image_tensor: frame_expanded})  
140  
141         vis_util.visualize_boxes_and_labels_on_image_array(  
142             frame,  
143             np.squeeze(boxes),  
144             np.squeeze(classes).astype(np.int32),  
145             np.squeeze(scores),  
146             category_index,  
147             use_normalized_coordinates=True,  
148             line_thickness=8,  
149             min_score_thresh=0.10)
```

图 4.28 断点调试(图中红点即为“断点”)

至此,我们完成了由“运动小车”到“自动驾驶小车”的整个开发过程,包括分析、设计、选材、开发、组装、调试、优化等一系列环节。同学们可以通过动手实践,设计并开发一辆自动驾驶小车,还可以利用课余时间查阅资料,了解自动驾驶相关技术和产业的前沿发展情况。

第四节 无人小车·项目发布

发布一个开源项目是人们为开源做贡献的一种直接而有效的途径,不仅可以传播自己的创意、为别人提供思路,也可以收获别人的帮助,改进和提高项目的质量。需要注意的是,当一个项目被开源,这意味着任何人都可以查看、运行、修改和发布该项目,因此在开源项目发布的具体操作流程上必须遵循特定的发布规范,选择合适的社区,相关权限可通过附加的开源许可证来规定。

体 验 思 考

如何将项目发布至开源社区? 如何更新维护你的开源项目?

一、开启一个开源项目

如果你乐意让他人对你的工作进行查看和反馈,你就应该开源你的项目。首先,开源项目都应包括以下文档:

(1) 许可证(license)。假如某个项目源代码开放,但是没有任何的许可协议,那它就不能称为开源项目,所以启动开源项目时,务必选择相应的许可证。

(2) 自述文件(README)。它不仅用于说明如何使用你的项目,还可以解释你的项目为什么重要,以及该项目可以为你的用户做什么。

(3) 贡献指南(CONTRIBUTING)。它告诉项目参与者如何参与你的项目,包括如何提交错误报告、如何建立一个新功能、如何配置开发环境和运行测试等。

(4) 行为准则。它主要为项目参与者的行为设定基本规则,比如参与社区活动时的一些礼仪、说话方式、行为等,有助于形成一种友好的氛围。

其次,应为项目选择一个容易记住、有创意、能表达项目用意的名称,但名称不能涉嫌侵权或和已存在的项目冲突。

再次,要保持项目所使用的代码风格一致,项目要有明确的功能、方法,代码注释要清晰。

最后,将所有内容检查一遍,如果没有问题,点击“publish”,完成发布。

二、Arduino 开源项目发布示例和许可证的选择

图 4.29 是 Arduino 开源项目的文档列表,它包含许可证、自述文件和贡献指南等文件。

| facchinm NetworkUpload. use ipAddress to start JSch session ... | | | Latest commit adabb00 22 hours ago |
|---|--------------------------|---|------------------------------------|
| | .settings | Update eclipse java-formatter settings | 4 years ago |
| | app | Avoid NPE and spurious unlock if serial monitor is clicked at upload ... | 9 days ago |
| | arduino-core | NetworkUpload. use ipAddress to start JSch session | 22 hours ago |
| | build | Added comment to address A-Style Autoformatter (#7550) | 13 days ago |
| | hardware | Update signatures for avrdude-6.3-arduino17 | last month |
| | libraries | Delete builtin libraries sources | 11 months ago |
| | .classpath | Add log4j dependencies | 29 days ago |
| | .gitignore | Amended .gitignore to ignore liblistSerials dev dependency. | 3 years ago |
| | .project | Fix eclipse project files | 8 years ago |
| | CONTRIBUTING.md | 贡献指南 Remove mention of Playground being a publicly editable wiki from CONT... | 3 months ago |
| | ISSUE_TEMPLATE.md | Update issue template to reflect change in GitHub's new issue page | 2 months ago |
| | PULL_REQUEST_TEMPLATE.md | Rename PULL_REQUEST_TEMPLATE_md to PULL_REQUEST_TEMPLATE.md | 13 days ago |
| | README.md | 自述文件 update link http -> https | 2 years ago |
| | lib_sync | Added lib_sync, utility script | 4 years ago |
| | license.txt | 许可证 Replacing Processing's text files with Arduino's (e.g. todo.txt) | 10 years ago |

图 4.29 Arduino 开源项目的文档列表

许可证分为开源硬件许可证和开源软件许可证。Arduino 的开源硬件许可证选用 CC BY - SA 3.0;开源软件许可证有两个,即 GNU GPL 2.0 和 GNU LGPL 2.1。GPL/LGPL 协议需要把开源者的许可证复制到项目的 License 文件里。那么 Arduino 的软件是参考哪个开源项目开发的呢?

Arduino 的发展历史要从 Processing 讲起,简单来说,Processing 是一个开源的集成开发环境(IDE),它的开源许可证是 GNU GPL 2.0,然后 Wiring 在 Processing 这个 IDE(其界面如图 4.30 所示)里增加了核心库开源代码(如图 4.31 所示),它就继承了 Processing 的 GNU GPL 2.0 协议。

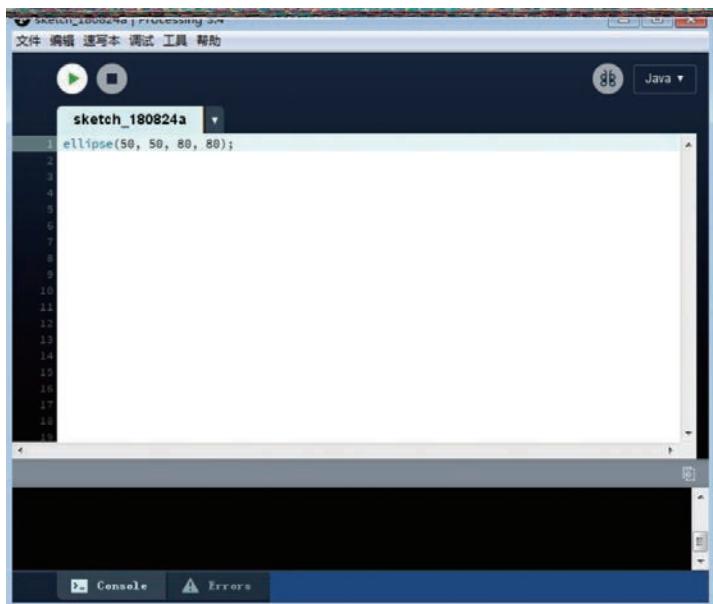


图 4.30 Processing IDE 界面

| | |
|----------------------------------|----------------------------------|
| Input/Output | |
| <i>Pin digital Input/Output</i> | <i>Time</i> |
| <i>digitalRead()</i> | <i>delay()</i> |
| <i>digitalWrite()</i> | <i>delayMicroseconds()</i> |
| <i>pinMode()</i> | <i>micros()</i> |
| <i>Pin digital Input pullup</i> | <i>millis()</i> |
| <i>noPullup()</i> | |
| <i>pullup()</i> | |
| <i>Port digital Input/Output</i> | <i>Pulse Input (polled)</i> |
| <i>portMode()</i> | <i>pulseIn()</i> |
| <i>portRead()</i> | |
| <i>portWrite()</i> | |
| <i>Pin Bit Shifting</i> | <i>Interrupts</i> |
| <i>shiftIn()</i> | <i>attachInterrupt()</i> |
| <i>shiftOut()</i> | <i>detachInterrupt()</i> |
| <i>Pin analog Input</i> | <i>interruptMode()</i> |
| <i>analogRead()</i> | <i>interrupts()</i> |
| <i>analogReference()</i> | <i>noInterrupts()</i> |
| <i>Pin PWM (analog) Output</i> | <i>Tone output generation</i> |
| <i>analogWrite()</i> | <i>getTonePolyphony()</i> |
| <i>noAnalogWrite()</i> | <i>noTone()</i> |
| <i>Pin PWM Advanced Settings</i> | <i>setTonePolyphony()</i> |
| <i>setPWMPrescale()</i> | <i>tone()</i> |
| <i>setPWMResolution()</i> | <i>Serial Communication</i> |
| | <i>Serial</i> |
| | <i>Power Management</i> |
| | <i>disablePower()</i> |
| | <i>enablePower()</i> |
| | <i>Advanced Power Management</i> |
| | <i>noSleep()</i> |
| | <i>sleep()</i> |
| | <i>sleepMode()</i> |

图 4.31 Wiring 核心库开源代码

Arduino IDE 是基于 Processing IDE 开发的,核心库开源代码是基于 Wiring 开发的,所以它也就继承了 Processing 和 Wiring 的 GNU GPL 2.0 协议,同时根据自己的要求新增了 GNU LGPL 2.1 协议,其发展脉络如图 4.32 所示。

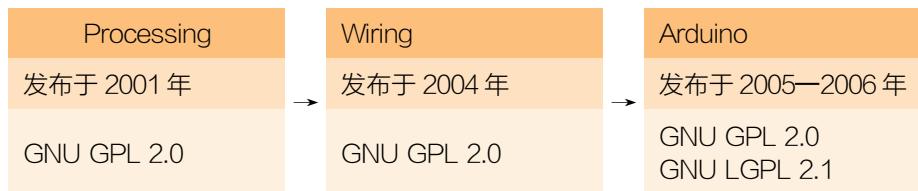


图 4.32 Arduino IDE 的发展脉络

以 Arduino 项目开源为例,开源项目发布前,需要按照规则分类设计文档,选择开源许可证,编写自述文件及其他相关文件等。

(1) 按照规则新建多个文件夹,如专门存放小车 3D 模型文件的“3D”文件夹,专门存放原理图、PCB 等硬件设计资料的“hw”文件夹等,并将各类设计文档放入相应的文件夹内,如图 4.33 所示。

| | |
|-----------------|--------------------------|
| 3D | —— 存放小车3D模型文件 |
| hw | —— 存放硬件设计资料 |
| docs | —— 存放其他文档 |
| drivers | —— 存放驱动源码 |
| examples | —— 存放例程源码 |
| extmod | —— 存放外部模块的资料（红外、激光传感器） |
| py | —— 存放Python源码（语音识别、图像识别） |
| LICENSE | —— 开源许可证文件 |
| README.md | —— 自述文件 |
| CONTRIBUTING.md | —— 贡献指南 |

图 4.33 文件夹及其保存文件类型

(2) 选择硬件许可证和软件许可证，并将选定的开源许可证写到托管仓库根目录的 License 文件夹内，注意不同的许可证有各自不同的要求。

(3) 编写 README. md 文件，保存在托管仓库的根目录下，它是一个介绍性的说明文件，主要解释项目有何用处、为何发起，以及如何快速入门等。不同的文件夹里需要编写相应的 README. md 文件来说明本文件夹内存放的内容。

(4) 编写其他文档，如贡献指南、行为准则以及教程等。

(5) 软件版本更新。

发布一个开源项目以后，要对项目不断地进行迭代和优化，相应会产生很多版本。大型的代码托管平台有专门用于版本更新的管理工具，例如在国内的码云或国外的 GitHub 上使用 git 来上传(更新)代码。

完成上述步骤后，自动驾驶小车项目就可以发布至开源社区了。

作业练习

选择一个开源社区，诸如 GitHub、码云等，将本章完成的自动驾驶小车项目规范地发布至平台上。

附录 开源硬件平台编程及输入/输出模块简介

1. Arduino 编程语法简介

(1) 程序结构

① Arduino 程序的构成

Arduino 程序必须包括两部分：

a. 初始化程序：void setup() {初始化程序代码}

b. 主体程序：void loop() {主体程序代码}

② 库函数调用

include 用于调取外部的库文件。

如：# include <DHT.h> // 调用与温湿度传感器使用相关的库文件

③ 自定义函数调用

为了确保程序结构清晰，将程序中可以重复使用或用来独立地完成某个功能的代码，以函数的形式单独放置，这样也可使得程序的可读性加强。如：

```
void loop()
{
    FlashNumber(1);
}

void FlashNumber(int pos)
{
    ...
}
```

④ # define 定义常量

如：# define ledPin 3 // 将 ledPin 定义为 3，方便调用和后期修改

(2) 常用符号

① ;(分号)：用于表示一句代码的结束。

② {}(花括号或称大括号): 成对出现, 将包含在其中的多条语句组合成复合语句。

③ // (单行注释): 将其所在行后面的文字作为注释, 不会被编译和被处理器执行。

④ /* */ (多行注释): 将“/*”和“*/”包含的多行文字作为注释。

(3) 流程控制

① if

格式: if(条件表达式){

.....//满足条件时执行的语句组

}

② if...else

格式: if(条件表达式){

.....//满足条件时执行的语句组

}

else

{

.....//不满足条件时执行的语句组

}

③ for

格式: for(初始化语句; 循环条件; 增量计数){

.....//满足条件时

重复执行的语句组

}

④ while

格式: while(条件表达式){

.....//满足条件时重复执行的语句组

}

(4) 常用运算

| 算术运算 | 赋值 | 加 | 减 | 乘 | 除 | 取余 |
|------|----|---|---|---|---|----|
| 运算符 | = | + | - | * | / | % |

| 关系运算 | 等于 | 不等于 | 大于 | 小于 | 大于等于 | 小于等于 |
|------|----|-----|----|----|------|------|
| 运算符 | == | != | > | < | >= | <= |

| | | | |
|------|----|---|---|
| 逻辑运算 | 与 | 或 | 非 |
| 运算符 | && | | ! |

(5) 常量表示

① HIGH|LOW(引脚电压定义) //定义高或低电平,也可直接用 1 或 0 表示

② true|false(布尔常量) //定义逻辑真或假,当结果是 0 时,为 false;当结果是非 0 时,为 true

③ INPUT|OUTPUT(数字引脚定义)

(6) 常用函数

① pinMode(pin, mode): 将指定的引脚 pin 配置成输入或输出,即 INPUT 或 OUTPUT。

② digitalRead(pin): 读取指定数字引脚 pin 的值,返回 HIGH 或 LOW。

③ digitalWrite(pin, value): 给一个数字引脚 pin 写入 HIGH 或 LOW。

④ analogRead(pin): 从指定模拟引脚 pin 读取数据值,返回 0~1 023 之间的整数值。

⑤ analogWrite(pin, value): 从一个模拟引脚 pin 输出值 value (PWM)。value 最小为 0,即占空比为 0;value 最大为 255,即占空比为 100%。

⑥ millis(): 返回 Arduino 开发板从运行当前程序开始的毫秒数。

⑦ delay(): 使程序暂停指定的时间(单位: ms)。

⑧ delayMicroseconds(): 使程序暂停指定的一段时间(单位: μ s)。

⑨ map(value, fromLow, fromHigh, toLow, toHigh): 将一个数从一个范围映射到另外一个范围。

⑩ attachInterrupt (interrupt, function, mode): 当指定引脚 interrupt 发生外部中断时,调用一个指定函数 function, mode 为触发条件。

⑪ Serial.begin(speed): 初始化串口,设置数据传输速率为 speed (单位: bps)。

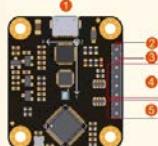
⑫ Serial.print(): 从串口打印输出数据,不换行。

⑬ Serial.println(): 从串口打印输出数据,并换行。

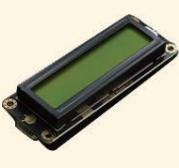
2. 开源硬件系统中常用的输入模块

| 名称 | 图片 | 简介 |
|----------|---|-----------------------------------|
| 声音传感器 |  | 用于检测周围环境中的声音强度。 |
| 超声波测距传感器 |  | 用于检测传感器与目标物之间的距离。 |
| 人体红外传感器 |  | 用于探测人或动物身体发射的红外线。 |
| 光敏传感器 |  | 用于将光信号转换为电信号。 |
| 气体传感器 |  | 用于检测可燃性气体和有毒烟雾。 |
| 温湿度传感器 |  | 用于检测环境中的温度、湿度信息。 |
| 加速度传感器 |  | 用于检测物体运动过程中的加速度, 获取物体运动姿态或运动方向信息。 |

(续 表)

| 名称 | 图片 | 简介 |
|---------|---|-----------------------------------|
| 陀螺仪传感器 |  | 用于测量角速度,即物体围绕轴旋转的速度。 |
| 霍尔传感器 |  | 用于检测周围磁场强度。 |
| 火焰传感器 |  | 用于探测火源或其他波长在 760~1 100 nm 范围内的光源。 |
| 土壤传感器 |  | 用于检测土壤的水分信息。 |
| 红外对管传感器 |  | 用于实现传输信号、遥控、避障等功能。 |

3. 开源硬件系统中常用的输出模块

| 名称 | 图 片 | 简 介 |
|---------|---|------------------------|
| LED 数码管 |  | 多个发光二极管组成数字 8 形状的显示器件。 |
| 液晶屏 |  | LED 技术的显示器件(黑白屏)。 |

(续 表)

| 名称 | 图 片 | 简 介 |
|---------------|---|---------------------------------|
| OLED 显示屏 |  | OLED 技术的显示器件(彩色屏)。 |
| 继电器 |  | 弱电(微控制器)控制强电(交流电、大电压、大电流)的控制器件。 |
| MP3 播放器 |  | 播放 MP3 音乐文件的器件。 |
| 直流电机驱动模块和直流电机 |  | 能将电能转换为机械能,驱动直流电机运转。 |
| 步进电机驱动模块和步进电机 |  | 通过脉冲信号精确控制角位移,驱动步进电机运转。 |
| 舵机 |  | 控制精度高、响应速度快,用于多足机器人、机械臂和监控云台等。 |

后记

本册教科书依据教育部《普通高中信息技术课程标准(2017年版2020年修订)》编写,并经国家教材委员会专家委员会审核通过。全体编写人员认真领会国家基础教育改革精神,精心研究当代信息社会的人才培养要求,广泛调研上海及各地高中信息技术教育的现状和挑战,深入了解高中学生的学习需求,并汲取了上海市《普通高中信息科技(试用本)》的编写经验。

编写过程中,上海市中小学(幼儿园)课程改革委员会专家工作委员会,上海市教育委员会教学研究室,上海市课程方案教育教学研究基地、上海市心理教育教学研究基地、上海市基础教育教材建设研究基地、上海市信息科技教育教学研究基地(上海高校“立德树人”人文社会科学重点研究基地)及基地所在单位华东师范大学等单位给予了大力支持,在此表示感谢!

本册教科书出版之前,我们已通过多种渠道与教科书选用作品(包括照片、画作)的作者进行了联系,得到了他们的大力支持。对此,我们衷心地表示感谢!恳请尚未联系到的作者与我们联系,以便出版社及时支付相关稿酬。

我们真诚地希望广大教师、学生及家长在使用本册教科书的过程中提出宝贵意见。我们将集思广益,不断修订,使教科书趋于完善。

编者

普通高中教科书

信息技术

选择性必修 6

开源硬件项目设计



绿色印刷产品

ISBN 978-7-5760-0554-7



9 787576 005547 >

定价：11.60元