



Cool projects that will push your skills to the limit

Raspberry Pi Mechatronics Projects

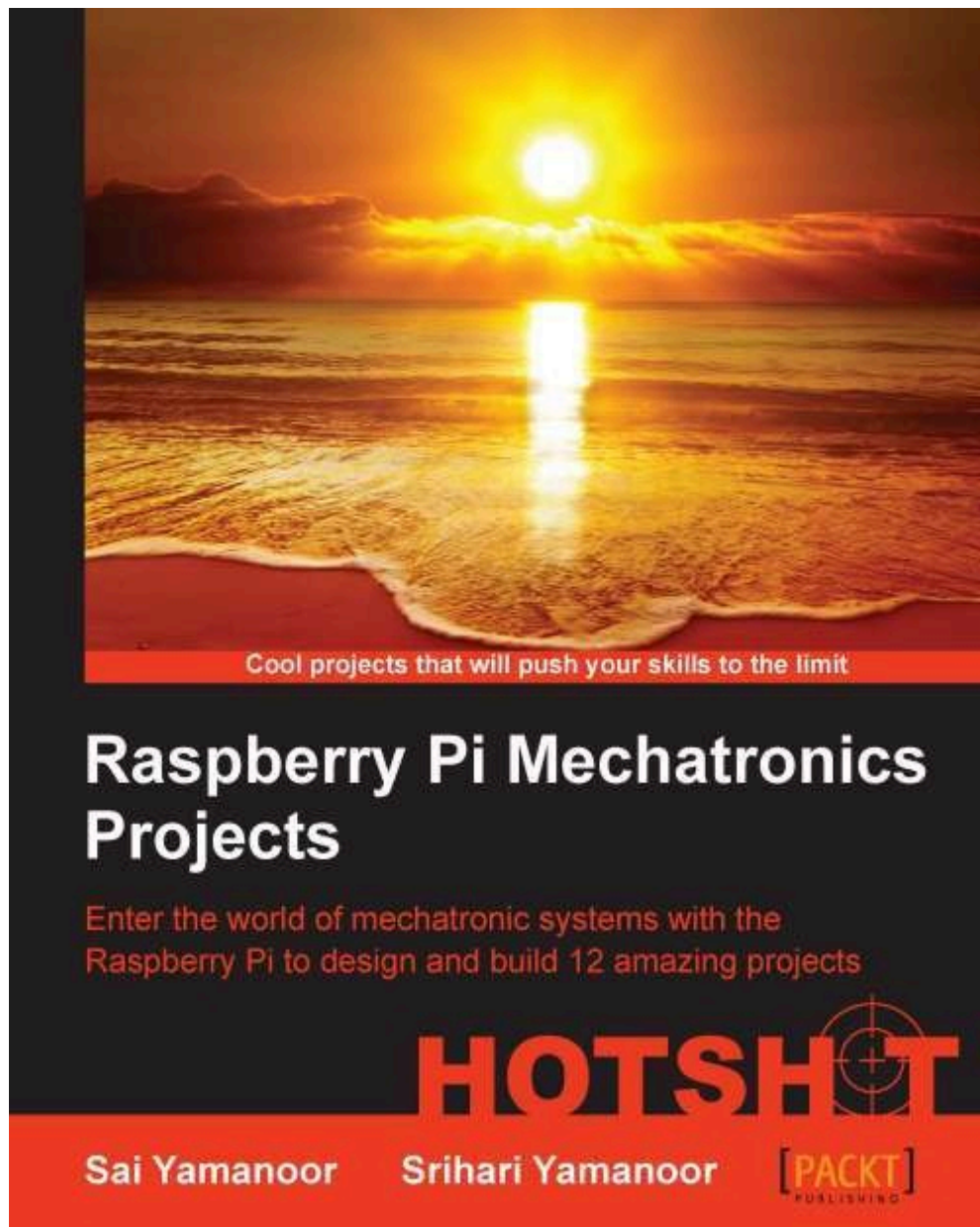
Enter the world of mechatronic systems with the Raspberry Pi to design and build 12 amazing projects

HOTSHOT

Sai Yamanoor

Srihari Yamanoor

[PACKT]
PUBLISHING



Raspberry Pi Mechatronics Projects

HOTSHOT

Table of Contents

[Raspberry Pi Mechatronics Projects HOTSHOT](#)

[Credits](#)

[About the Authors](#)

[About the Reviewers](#)

[www.PacktPub.com](#)

[Support files, eBooks, discount offers, and more](#)

[Why subscribe?](#)

[Free access for Packt account holders](#)

[Preface](#)

[What this book covers](#)

[What you need for this book](#)

[Who this book is for](#)

[Conventions](#)

[Mission briefing](#)

[Why is it awesome?](#)

[Your Hotshot objectives](#)

[Mission checklist](#)

[Task 1](#)

[Prepare for lift off](#)

[Engage thrusters](#)

[Objective complete – mini debriefing](#)

[Classified intel](#)

[Reader feedback](#)

[Customer support](#)

[Downloading the example code](#)

[Downloading the color images of this book](#)

[Errata](#)

[Piracy](#)

[Questions](#)

[1. Hello World](#)

[A quick introduction to the Raspberry Pi](#)

[Features of a Raspberry Pi](#)

[Where can I buy a Raspberry Pi?](#)

[Requirements to get started with the Raspberry Pi](#)

[Operating systems on the Raspberry Pi](#)

[Getting started with Raspbian](#)

[Downloading Raspbian](#)

[Flashing image on to the SD card](#)

[Windows](#)

[Linux](#)

[A command-line interface-based approach](#)

[Identifying the SD card mount point](#)

[Unmount the SD card](#)

[Flash the SD card](#)

[GUI-based approach](#)

[Setting up the Raspberry Pi](#)

[Mission briefing](#)

[Why is it awesome?](#)

[Your Hotshot objectives](#)

[Mission checklist](#)

[Hunting and gathering](#)

[Engage thrusters](#)

[Objective complete – mini debriefing](#)

[Setting up the GPIO](#)

[Engage thrusters](#)

[Installing GPIO libraries](#)

[Objective complete – mini debriefing](#)

[GPIO programming using Python](#)

[Engage thrusters](#)

[An alternative to quick2wire – RPi.GPIO](#)

[Objective complete – mini debriefing](#)

[Electrical output of our program](#)

[Engage thrusters](#)

[Objective complete – mini debriefing](#)

[Introduction to the Pi Crust – a prototyping platform for the Raspberry Pi](#)

[Mission accomplished](#)

[Hotshot challenge](#)

[2. A Raspberry WebIDE Example](#)

[Mission briefing](#)

[Why is it awesome?](#)

[Your Hotshot objectives](#)

[Mission checklist](#)

[Installation, features, and usage of the Occidentalis operating system from Adafruit](#)

[Prepare for lift off](#)

[Engage thrusters](#)

[Objective complete – mini debriefing](#)

[Setup of a remote login into the Raspberry Pi](#)

[Prepare for lift off](#)

[Engage thrusters](#)

[Objective complete – mini debriefing](#)

[Installation of the Raspberry Pi WebIDE](#)

[Engage thrusters](#)

[Objective complete – mini debriefing](#)

[Python development on the WebIDE](#)

[Prepare for lift off](#)

[Quick introduction to the I2C interface](#)

[Configuring the I2C interface on the Raspberry Pi](#)

[Engage thrusters](#)

[Objective complete – mini debriefing](#)

[Test and debugging examples using the WebIDE](#)

[Prepare for lift off](#)

[Engage thrusters](#)

[Objective complete – mini debriefing](#)

[Mission accomplished](#)

[Hotshot challenge](#)

[3. The Arduino Raspberry Pi Interface](#)

[Mission briefing](#)

[Why is it awesome?](#)

[Where can you buy an Arduino?](#)

[Your Hotshot objectives](#)

[Mission checklist](#)

[Installing the Arduino IDE](#)

Prepare for lift off

Engage thrusters

Objective complete – mini debriefing

Programming the Arduino using the Raspberry Pi

Prepare for lift off

Engage thrusters

Objective complete – mini debriefing

Raspberry Pi AlaMode

Prepare for lift off

Engage thrusters

Objective complete – mini debriefing

The Weasley weather clock

Prepare for lift off

Engage thrusters

Objective complete – mini debriefing

Controlling the stepper using the Arduino

Prepare for lift off

Engage thrusters

Counterclockwise rotation of the stepper motor

Objective complete – mini debriefing

[Controlling the RGB LED Strip using the Arduino](#)

[Prepare for lift off](#)

[Engage thrusters](#)

[Objective complete – mini debriefing](#)

[I2C Communication using the Arduino \(optional\)](#)

[Prepare for lift off](#)

[Engage thrusters](#)

[The 7-Segment display](#)

[8x8 LED matrix](#)

[BlinkM](#)

[Proximity sensor](#)

[Objective complete – mini debriefing](#)

[Serial port communication with the Raspberry Pi](#)

[Engage thrusters](#)

[Python program for the Weasley clock](#)

[Arduino acknowledgement to control signal](#)

[Control flow for the weather clock](#)

[Temperature data](#)

[Control of the RGB LED strip](#)

[Control of stepper via serial port](#)

[Programming the weather forecast position](#)

[8x8 matrix control](#)

[BlinkM control](#)

[Objective complete – mini debriefing](#)

[Mission accomplished](#)

[Hotshot challenge](#)

[4. Christmas Light Sequencer](#)

[Mission briefing](#)

[Why is it awesome?](#)

[Your objectives](#)

[Mission checklist](#)

[Interface the devices to the Raspberry Pi](#)

[Prepare for lift off](#)

[Engage thrusters](#)

[Connecting multiple appliances to the Raspberry Pi](#)

[Objective complete – mini debriefing](#)

[Setting up the digitally addressable RGB matrix](#)

[Prepare for lift off](#)

[Where can I buy them?](#)

[Engage thrusters](#)

[How does it work?](#)

[Lighting up the RGB LED strip](#)

[An Arduino-based control](#)

[Objective complete – mini debriefing](#)

[Interface of an audio device](#)

[Prepare for lift off](#)

[Engage thrusters](#)

[Objective complete – mini debriefing](#)

[Installing the web server](#)

[Prepare for lift off](#)

[Engage thrusters](#)

[Objective complete – mission debriefing](#)

[Interfacing the web server](#)

[Prepare for lift off](#)

[Engage thrusters](#)

[Objective complete – mini debriefing](#)

[Mission accomplished](#)

[References](#)

[5. Internet of Things Example – An E-mail Alert Water Fountain](#)

[Mission briefing](#)

[Why is it awesome?](#)

[Your objectives](#)

[Mission checklist](#)

[Prepare for lift off](#)

[Engage thrusters](#)

[Objective complete – mini debriefing](#)

[Installation of the Flask framework](#)

[Prepare for lift off](#)

[Engage thrusters](#)

[Objective complete – mini debriefing](#)

[Controlling RGB LEDs from a web page](#)

[Prepare for lift off](#)

[Engage thrusters](#)

[Objective complete – mini debriefing](#)

[Setup of the e-mail alerts in the fountain](#)

[Prepare for lift off](#)

[Engage thrusters](#)

[Objective complete – mini debriefing](#)

[Mission accomplished](#)

[Hotshot challenge](#)

6. Raspberry Pi as a Personal Assistant

Mission briefing

Why is it awesome?

Your objectives

Mission checklist

Setting up the e-mail feed parser

Prepare for lift off

Engage thrusters

Setting up the parser for reminders and events

Prepare for lift off

Engage thrusters

Designing an enclosure design for the personal assistant

Prepare for lift off

Engage thrusters

Setting up the assembly for the dish monitor

Prepare for lift off

Engage thrusters

Objective complete – mini debriefing

Setting up sensors for the key alert system

Prepare for lift off

[Engage thrusters](#)

[Objective complete – mini debriefing](#)

[Mission accomplished](#)

[Hotshot challenges](#)

[References](#)

[7. Raspberry Pi-based Line Following Robot](#)

[Mission briefing](#)

[Why is it awesome?](#)

[Your Hotshot objectives](#)

[Mission checklist](#)

[Assembly of the Pi Plate for the line following robot](#)

[Prepare for lift off](#)

[Engage thrusters](#)

[Objective complete – mini debriefing](#)

[Sensor selection, interface, and data interpretation](#)

[Engage thrusters](#)

[Light Dependent Resistors](#)

[Infrared emitter/detector](#)

[Machine-vision-based line following robot](#)

[Sensor selection](#)

Objective complete – mission debriefing

Implementation of line following logic based on sensor data

Prepare for lift off

Engage thrusters

Objective complete – mini debriefing

Testing the motor driver circuit

Prepare for lift off

Engage thrusters

Objective complete – mini debriefing

Preparation of the line following course

Prepare for lift off

Engage thrusters

Use electrical tape and build a track

Designing the chassis for the line following robot

Prepare for lift off

Engage thrusters

Objective complete – mini debriefing

A step-by-step assembly of the robot

Prepare for lift off

Engage thrusters

[Objective complete – mini debriefing](#)

[Mission accomplished](#)

[Hotshot challenge](#)

[References](#)

[8. Connect Four Desktop Game using Raspberry Pi](#)

[Mission briefing](#)

[Why is it awesome?](#)

[Your objectives](#)

[Mission checklist](#)

[Hello world example with the pygame](#)

[Prepare for lift off](#)

[Engage thrusters](#)

[Playing sounds using the pygame module](#)

[Building menus using the pygame module](#)

[Objective complete – mini debriefing](#)

[Getting started with the Connect Four game \(UI and sound effects\)](#)

[Prepare for lift off](#)

[Engage Thrusters](#)

[Adding a menu for the game](#)

[Adding sounds to the game](#)

[Objective complete – mission debriefing](#)

[Interfacing game inputs using the PiFace module](#)

[Prepare for lift off](#)

[Engage thrusters](#)

[Objective complete – mini debriefing](#)

[A brief description of the Connect Four Game AI](#)

[Prepare for lift off](#)

[Engage thrusters](#)

[Main game loop](#)

[Objective complete – mini debriefing](#)

[Marquee design and control \(optional\)](#)

[Prepare for lift off](#)

[Engage thrusters](#)

[Objective complete – mini debriefing](#)

[Setup of the switches using a cardboard box](#)

[Prepare for lift off](#)

[Engage thrusters](#)

[Objective complete – mini debriefing](#)

[Mission accomplished](#)

[Hotshot challenge](#)

References

9. The Raspberry Pi-enabled Pet/Wildlife Monitor

Mission briefing

Why is it awesome?

Your objectives

Mission checklist

Installing and testing the python-twisted framework

Prepare for lift off

Engage thrusters

Objective complete – mini debriefing

Setting up and testing Spark Core to interact with appliances

Prepare for lift off

Engage Thrusters

Objective complete – mini debriefing

Installing and testing the Google Coder project on the Raspberry Pi

Prepare for lift off

Engage thrusters

Objective complete – objective complete

A brief overview of a bird feeder that triggers a camera

Prepare for lift off

[Engage thrusters](#)

[Objective complete – mini debriefing](#)

[Alternatives and project ideas to consider](#)

[Mission accomplished](#)

[Hotshot challenge](#)

[10. Raspberry Pi Personal Health Monitor](#)

[Mission briefing](#)

[Why is it awesome?](#)

[Your Hotshot objectives](#)

[Mission checklist](#)

[Setting up a web server to record health parameters](#)

[Prepare for lift off](#)

[Engage thrusters](#)

[Objective complete – mini debriefing](#)

[A simple exercise tool using the Raspberry Pi](#)

[Prepare for lift off](#)

[Engage thrusters](#)

[A mini project idea](#)

[Objective complete – mini debriefing](#)

[Setting up an e-mail feed parser to refill a prescription as well as remind the user about](#)

[an upcoming doctor's appointment](#)

[Engage thrusters](#)

[Setting up a reminder for doctor's appointments](#)

[Objective complete – mini debriefing](#)

[A simple tool that reminds a person to take his/her pills](#)

[Prepare for lift off](#)

[Engage thrusters](#)

[Other project ideas](#)

[Objective complete – mini debriefing](#)

[Mission accomplished](#)

[Hotshot challenge](#)

[11. Home Automation using Raspberry Pi](#)

[Mission briefing](#)

[Why is it awesome?](#)

[Your objectives](#)

[Mission checklist](#)

[A simple example that controls limited indoor and outdoor lighting in the evening](#)

[Prepare for lift off](#)

[Software prerequisites](#)

[Engage thrusters](#)

[Objective complete – mini debriefing](#)

[Alternative solutions and some project ideas to consider](#)

[A customary web server example](#)

[Prepare for lift off](#)

[Engage thrusters](#)

[Objective complete – mini debriefing](#)

[Turning on a lawn sprinkler only when there is no rain forecast!](#)

[Prepare for lift off](#)

[Engage thrusters](#)

[Objective complete – mini debriefing](#)

[Gaining remote access to your Raspberry Pi to control appliances](#)

[Some project ideas to consider](#)

[Mission accomplished](#)

[12. Using a Raspberry Pi for Science and Education](#)

[Mission briefing](#)

[Why is it awesome?](#)

[Your Hotshot objectives](#)

[Improving your vocabulary using the Raspberry Pi](#)

[Prepare for lift off](#)

[Engage thrusters](#)

[Objective complete – mini debriefing](#)

[Raspberry Pi and Khan Academy](#)

[Some ideas to consider](#)

[Building a science fair exhibit using the Raspberry Pi](#)

[Prepare for lift off](#)

[Engage thrusters](#)

[Objective complete – mini debriefing](#)

[Some simple educational experiments using the Raspberry Pi](#)

[Hotshot challenge](#)

[13. Tips and Tricks](#)

[Mission briefing](#)

[Why is it awesome?](#)

[Your objectives](#)

[Mission checklist](#)

[Setting up Raspberry Pi as a development platform](#)

[#1 – Simple trick for Python development via remote login](#)

[#2 Web development using Google Coder](#)

[#3 Adafruit Occidentalis](#)

[#4 Java Development using the Raspberry Pi](#)

[#5 The Thingbox project](#)

[Remotely logging in the Raspberry Pi from anywhere on the Internet](#)

[Problems that might be encountered while using Raspberry Pi](#)

[SD card corruption](#)

[Power issues](#)

[Devices powered by USB](#)

[Cool add-on hardware sources developed for the Raspberry Pi platform](#)

[The Raspberry Pi model B versus B+, model A versus A+](#)

[Project ideas that can be implemented over a weekend](#)

[Useful resources for the Raspberry Pi](#)

[Raspberry Pi Foundation's forums:](#)

[Raspberry Pi Stack Exchange:](#)

[Element 14's Raspberry Pi community:](#)

[Mission accomplished](#)

[Index](#)

Raspberry Pi Mechatronics Projects

HOTSHOT

Raspberry Pi Mechatronics Projects

HOTSHOT

Copyright © 2015 Packt Publishing

All rights reserved. No part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the

prior written permission of the publisher, except in the case of brief quotations embedded in critical articles or reviews.

Every effort has been made in the preparation of this book to ensure the accuracy of the information presented. However, the information contained in this book is sold without warranty, either express or implied. Neither the authors, nor Packt Publishing, and its dealers and distributors will be held liable for any damages caused or alleged to be caused directly or indirectly by this book.

Packt Publishing has endeavored to provide trademark information about all of the

companies and products mentioned in this book by the appropriate use of capitals.

However, Packt Publishing cannot guarantee the accuracy of this information.

First published: February 2015

Production reference: 1190215

Published by Packt Publishing Ltd.

Livery Place

35 Livery Street

Birmingham B3 2PB, UK.

ISBN 978-1-84969-622-7

www.packtpub.com

Credits

Authors

Sai Yamanoor

Srihari Yamanoor

Reviewers

Guy Scheffer

Sreedhar Vaddi

Cheuk Yu

Commissioning Editor

Usha Iyer

Acquisition Editors

Usha Iyer

Rebecca Pedley

Sam Wood

Content Development Editor

Susmita Sabot

Technical Editor

Parag Topre

Copy Editor

Janbal Dharmaraj

Project Coordinator

Milton D'souza

Proofreaders

Simran Bhogal

Maria Gould

Ameesha Green

Paul Hindle

Indexer

Hemangini Bari

Graphics

Sheetal Aute

Disha Haria

Production Coordinators

Aparna Bhagat

Manu Joseph

Cover Work

Aparna Bhagat

About the Authors

Sai Yamanoor is a mechatronics engineer working for a private start-up school in the San Francisco Bay Area where he builds devices that help students achieve their full potential.

He completed his degree in mechanical engineering at Carnegie Mellon University,

Pittsburgh, PA, and his undergraduate work in mechatronics at Sri Krishna College of Engineering and Technology, Coimbatore, India. His interests, deeply rooted in DIY and open software and hardware cultures, include developing gadgets and apps that improve quality of life, Internet of Things, crowdfunding, education, and new technologies. In his spare time, he plays with various devices and architectures such as Raspberry Pi, Arduino, Galileo, Android devices, and others. Sai shares his adventures with mechatronics at the aptly named “Mechatronics Craze” blog at <http://mechatronicscraze.wordpress.com/>.

I want to thank my parents for encouraging me in all my endeavors and for making me what I am today. I am grateful to my brother who has helped shape my career all these years. I look forward to working with Srihari on similar projects in the future. I sincerely thank Mr. S. Balaji Raghavendra who has been a great source of inspiration to me while pursuing my undergraduate degree. I am also thankful to Susmita Sabot and the entire support team at Packt Publishing who were patient and understood the circumstances during difficult times.

Srihari Yamanoor is a nature photographer and mechanical engineer with experience working in medical device design and CAD/CAM. He completed his master’s in science as well as a degree in engineering from Stanford University. His work and interests in medical devices include women’s health, cancer, orthopedics, and cardiovascular diseases, as well as newer areas such as digital health and electronic health records. He has several thousands of hours of CAD experience behind him and is also certified among other things as a SolidWorks expert, simulation professional, and a SolidWorks instructor. His interests range from engineering and education to social entrepreneurship, animal welfare, and crowdfunding.

I would like to thank my parents, Narasimhan and Yasoda Venakatakrisnan, for their steadfast support in my education and efforts throughout the years, culminating in projects such as this book. Over the course of the years, many friends have helped both Sai and me grow in many ways. For this reason, I would like to say a big word of thanks, in no particular order, to Rika Catherine Hirachi, Anna Liu Jao, Shirupa Gupta, Patrick Nguyen, Vishnu Prasad Ramachandran, Andrew Eib, Christopher Ma, and Karthik Palaniappan. I am sure I am missing some names, and for this, I apologize in

advance. I would also like to thank my gracious mentors, Dr. Kevin Waldron, Dr. James Stubbs, Russell Sampson, Mukund Patel, and as I fondly call him, Cyril “Master”, all of whom have given me advice and shown me the way. I would like to especially thank my first cat, Squeaky, and the felines Bob, Saxon, and Fishbone, that have followed me, for inspiring me and pushing me to stay on track and keep an eye on the big picture.

About the Reviewers

Sreedhar Vaddi has 4 years of experience in big data, 4 years in cloud computing, and 15

years in Java. His experience includes work in the fields of big data security,

biomedical/healthcare, mobile security, financial services (wholesale, mortgage,

investment, and brokerage), the Web (marketplace, crowdsourcing, and advertising), news and entertainment, property and casualty insurance, software product development, and scientific research.

He has hands-on experience as an architect working on enterprise mobile applications, Hadoop, SaaS, PaaS, IaaS, the cloud, rich enterprise applications, n-tier enterprise systems, Enterprise Application Integration (EAI) in web-enabling legacy systems, IBM

Mainframe and SAP, and SOA web services with Java2 and JEE.

He has also worked in all phases of SDLC, namely designing, developing, performance testing, and implementing Web-2.0, Web-UI, server-side, middleware, and mobile applications. He has also worked with Big 5 Consulting and Fortune 100 companies. He has led teams of sizes varying from 2 to 80 across the globe in different time zones, cultures, and dialects.

He is an administrator of JEE containers (Geronimo, WebSphere, Weblogic, JBoss, and Tomcat) and CDH. He is also a technical consultant at a stealth mode start-up in animal care in the clinical trials industry. He is a diligent worker and quick learner with excellent problem solving, presentation, and mentoring skills.

Cheuk Yu is pursuing a bachelor's degree in electrical engineering at the University of California, Los Angeles (UCLA), with an emphasis on circuit and signal processing. He worked on multiple embedded systems projects using various microcontrollers such as Arduino, Teensy, and Raspberry Pi. As a student researcher for the ELFIN CubeSat Mission at UCLA, he also has hands-on experience working on printed circuit boards, including designing and testing avionics and ground support equipment boards. Besides work and study, he is also an amateur radio operator and astronomer.

www.PacktPub.com



Support files, eBooks, discount offers, and more

For support files and downloads related to your book, please visit www.PacktPub.com.

Did you know that Packt offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at www.PacktPub.com and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at <service@packtpub.com> for more details.

At www.PacktPub.com, you can also read a collection of free technical articles, sign up for a range of free newsletters and receive exclusive discounts and offers on Packt books and eBooks.

<https://www2.packtpub.com/books/subscription/packtlib>

Do you need instant solutions to your IT questions? PacktLib is Packt's online digital book library. Here, you can search, access, and read Packt's entire library of books.

Why subscribe?

Fully searchable across every book published by Packt

Copy and paste, print, and bookmark content

On demand and accessible via a web browser

Free access for Packt account holders

If you have an account with Packt at www.PacktPub.com, you can use this to access PacktLib today and view 9 entirely free books. Simply use your login credentials for immediate access.

Preface

The Raspberry Pi is a quaint example of technological innovations that come by once every two or three decades and set off a revolution that touches every aspect of human life from scientific exploration to entertainment to education. What is exciting this time around is that there are several such innovations happening simultaneously. They might sound like, and are in a way, buzzwords, but they are in fact real, and are changing our lives in so many ways, so few of which we are consciously aware of. We are speaking of innovations in entrepreneurship, such as crowdsourcing or the emergence of DIY with an almost cult-like following to innovations in technology that are current, such as 3D

printing and Internet of Everything, and innovations that are impending such as 4D

printing, smart manufacturing, and much more.

The Raspberry Pi has already found several uses as evidenced by hundreds of websites, project examples, and crowdsourcing campaigns. People have used the Pi to build robots, teach programming, restore old gaming consoles, stream videos, collect data, and do many other things. The Pi itself has gone through design iterations and continues to spur competing systems. People have taken advantage of this and have hooked up the Pi to many different ecosystems, expanding its ubiquity and utility.

The purpose of this book is to help those who are excited about the Raspberry Pi and have project ideas in mind, or would simply like to get their hands dirty and practice implementing projects so that they can then use that knowledge for other endeavors. We tried to take a practical approach, introducing the Pi in the first couple of project that you can skip if you are familiar with the Raspberry Pi and have used it before. We then discuss projects of increasing complexity. We have also attempted to diversify the projects to demonstrate various uses for the Pi throughout the book.

What this book covers

[Project 1](#), *Hello World*, will introduce the Raspberry Pi and remind you of things that you should know as you proceed through the book. We will set up the Raspberry Pi and blink an LED.

[Project 2](#), *A Raspberry WebIDE Example*, will take you through Python development using a browser. We will use the Adafruit WebIDE and lead you through the development process using Adafruit products as props for the examples.

[Project 3](#), *The Arduino Raspberry Pi Interface*, is inspired by the Harry Potter series, and in an effort to show our love for this series, we have created a clock similar to the innovative and endearing Weasley clock using the Raspberry Pi and an Arduino. This project, of course, allows the clock to report back weather conditions.

[Project 4](#), *Christmas Light Sequencer*, discusses holidays as the best time for DIY projects, and if you are ever starved of ideas, holidays are the best time to cook up ideas. We picked a Christmas-themed project to demonstrate controlling appliances connected to a local network using Raspberry Pi.

[Project 5](#), *Internet of Things Example – An E-mail Alert Water Fountain*, follows the theme of the previous project. We will show you how to control a decorative fountain such that anytime you receive a new e-mail, your fountain's light will flash an alert (this can also be switched to receive Twitter alerts).

[Project 6](#), *Raspberry Pi as a Personal Assistant*, will show you how you can use the Pi as your personal assistant, reminding you of chores, setting alarms, and doing everything at your command.

[Project 7](#), *Raspberry Pi-based Line Following Robot*, will take you through the simplest of all robots, the line following bot, with Raspberry Pi at the helm. This project will show you the way for this and other complex robotics projects you wish to implement.

[Project 8](#), *Connect Four Desktop Game using Raspberry Pi*, follows the instant-classic theme for the Raspberry Pi, which includes games. All manners of gaming applications, including the revival of archaic games to new ones and mods, are being created using the Raspberry Pi. We will implement the game using a push button interface.

[Project 9](#), *The Raspberry Pi-enabled Pet/Wildlife Monitor*, will walk you through a fun example of a bird feeder monitor that allows you to trigger a photo capture each time the bird approaches the feeder. You can think of extending this to other pets, or say, the feral cats in your colony.

[Project 10](#), *Raspberry Pi Personal Health Monitor*, is a project where we set up a simple web server to record our personal health parameters, build a simple tool to remind ourselves to remain physically active, and remind ourselves to refill a prescription via email alerts.

[Project 11](#), *Home Automation using Raspberry Pi*, is a project where we will show you how to control appliances using a twisted network where the appliance is connected to a platform such as an Arduino. The Raspberry Pi acts as a server that controls all such nodes in the network.

[Project 12](#), *Using a Raspberry Pi for Science and Education*, is a project packed with examples, including a vocabulary learning tool, a web host for Khan Academy, and a windmill generator science exhibit. Have fun learning and teaching!

[Project 13](#), *Tips and Tricks*, is a project packed with tips and tricks that will help you use the Raspberry Pi!

You can also get the videos, tips and tricks, and many more things related to the projects at

<http://diywithpi.com/>.

What you need for this book

To get started, you will need the Raspberry Pi, a monitor, a keyboard, a power source, and preferably, a laptop or personal computer running on any OS—Windows, Linux, or Apple.

Other hardware and software recommendations are made in the individual projects, and you are welcome to replace them with what you see fit. A working knowledge of Python is preferred, but not required.

Who this book is for

This book is primarily aimed at hobbyists and do-it-yourself enthusiasts as well as those looking to implement specific projects using low cost hardware and software centered around the Raspberry Pi. Basic exposure to electronics, programming, and Internet usage is assumed. We have tried to take a pragmatic approach, implementing the projects and showing how you can do this on your own.

Conventions

In this book, you will find several headings appearing frequently. To give clear

instructions of how to complete a procedure or task, we use:

Mission briefing

This section explains what you will build, with a screenshot of the completed project.

Why is it awesome?

This section explains why the project is cool, unique, exciting, and interesting. It describes what advantage the project will give you.

Your Hotshot objectives

This section explains the eight major tasks required to complete your project.

Task 1

Task 2

Task 3

Task 4

Task 5

Task 6

Task 7

Task 8

Mission checklist

This section explains any pre-requisites for the project, such as resources or libraries that need to be downloaded, and so on.

Task 1

This section explains the task that you will perform.

Prepare for lift off

This section explains any preliminary work that you may need to do before beginning work on the task.

Engage thrusters

This section lists the steps required in order to complete the task.

Objective complete – mini debriefing

This section explains how the steps performed in the previous section allows us to complete the task. This section is mandatory.

Classified intel

This section provides some extra information relevant to the task.

You will also find a number of styles of text that distinguish between different kinds of information. Here are some examples of these styles, and an explanation of their meaning.

Code words in text, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input, and Twitter handles are shown as follows: “We will go through the important features of the pygame module, which is necessary to build our arcade game, including playing sounds and controlling the menu.”

A block of code is set as follows:

```
GPIO.output(25, True)
```

```
while 1:
```

```
    GPIO.output(25,False)
```

```
    sleep(1)
```

```
    GPIO.output(25,True)
```

```
    sleep(1)
```

Any command-line input or output is written as follows:

```
# cp /usr/src/asterisk-addons/configs/cdr_mysql.conf.sample  
/etc/asterisk/cdr_mysql.conf
```

New terms and **important words** are shown in bold. Words that you see on the screen, in menus or dialog boxes for example, appear in the text like this: “clicking the **Next** button moves you to the next screen”.

Note

Warnings or important notes appear in a box like this.

Tip

Tips and tricks appear like this.

Reader feedback

Feedback from our readers is always welcome. Let us know what you think about this book—what you liked or may have disliked. Reader feedback is important for us to develop titles that you really get the most out of.

To send us general feedback, simply send an e-mail to [<feedback@packtpub.com>](mailto:feedback@packtpub.com), and mention the book title via the subject of your message.

If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, see our author guide on www.packtpub.com/authors.

Customer support

Now that you are the proud owner of a Packt book, we have a number of things to help you to get the most from your purchase.

Downloading the example code

You can download the example code files for all Packt books you have purchased from your account at <http://www.packtpub.com>. If you purchased this book elsewhere, you can visit <http://www.packtpub.com/support> and register to have the files e-mailed directly to you.

Downloading the color images of this book

We also provide you a PDF file that has color images of the screenshots/diagrams used in this book. The color images will help you better understand the changes in the output. You can download this file from http://www.packtpub.com/sites/default/files/downloads/6227OT_ColoredImages.pdf.

Errata

Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you find a mistake in one of our books—maybe a mistake in the text or the code—we would be grateful if you would report this to us. By doing so, you can save other readers from frustration and help us improve subsequent versions of this book. If [you find any errata, please report them by visiting http://www.packtpub.com/submit-](http://www.packtpub.com/submit-errata)

[errata, selecting your book, clicking on the errata submission form link, and entering](http://www.packtpub.com/submit-errata) the details of your errata. Once your errata are verified, your submission will be accepted and the errata will be uploaded on our website, or added to any list of existing errata, under the Errata section of that title. Any existing errata can be viewed by selecting your title from <http://www.packtpub.com/support>.

Piracy

Piracy of copyright material on the Internet is an ongoing problem across all media. At Packt, we take the protection of our copyright and licenses very seriously. If you come across any illegal copies of our works, in any form, on the Internet, please provide us with the location address or website name immediately so that we can pursue a remedy.

Please contact us at [<copyright@packtpub.com>](mailto:copyright@packtpub.com) with a link to the suspected pirated material.

We appreciate your help in protecting our authors, and our ability to bring you valuable content.

Questions

Objective complete – mini debriefing

In this task, we finished setting up the desktop Connect Four game using Raspberry Pi.

Mission accomplished

In this project, we accomplished the following:

1. We used a Python script for the Connect Four Game and tweaked it to our requirements.
2. We implemented input controls using arcade style switches.
3. We wrote a program to control the marquee.
4. We also set up things on a cardboard box to play the game.

Hotshot challenge

In this project, we performed control of the input switches as a blocking operation. Can you figure out how to monitor switch press events by starting multiple threads? How can we improve the speed of the game?

References

1. *Making Games with Python and Pygame* is available at <http://inventwithpython.com/makinggames.pdf>.
2. *PiFace Guides* can be found at <http://www.piface.org.uk/guides/>.
3. [Pi-Lite Users' Guide can be found at http://openmicros.org/index.php/articles/94-ciseco-product-documentation/raspberry-pi/280-b040-pi-lite-beginners-guide](http://openmicros.org/index.php/articles/94-ciseco-product-documentation/raspberry-pi/280-b040-pi-lite-beginners-guide).

Chapter 9. The Raspberry Pi-enabled

Pet/Wildlife Monitor

In this project, we will discuss how the Raspberry Pi can interact with other devices on a network. This device can either be a laptop, an Arduino, or any other device connected to a network. We will make use of the capability of the Raspberry Pi to act as a control device in the network and implement a simple solution that can safely interact with our precious little animal friends.

Mission briefing

In this project, we will learn to build an interactive pet or wildlife (we call it wildlife in the context of animals that are not domesticated, for example, wild cats) monitor that can be monitored and controlled within a network (with some extra effort you should be able to control anywhere on the Web). This project presents three different possibilities where we can make use of the Raspberry Pi in combination with other devices to build an interactive device for pets that dispense treats to interface with devices that either interact with pets/feral cats/birds. This interactive device can either be a camera that captures a moment (for example, a wild bird coming and eating out of your bird feeder) or a relay board activated to feed treats to our animal friend while we are away from home.

Note

This project has been presented as an example to help a person enhance the quality of life for their animal friends. This project should not be construed as a replacement of human interaction for our animal friends. We totally understand that pets ease the stress in our lives and this project presents some examples to make their lives better.

Why is it awesome?

We are demonstrating the capabilities of the Raspberry Pi while building a pet monitor.

This can be adopted in any application where there needs to be exchange of information between devices. This project demonstrates the use of the Raspberry Pi in the betterment of our lives from another perspective. Our pets and other feral friends (for example, birds or feral cats) that visit us are our family members and we try our best to take care of them.

Our goal in this project is to make the Raspberry Pi do our job better.

Your objectives

In this project, we will explore the following topics:

Installing and testing the python-twisted framework

Setting up and testing Spark Core to interact with appliances

Installing and testing the Google Coder project on the Raspberry Pi

A brief overview of a bird feeder that triggers a camera

Mission checklist

For this project, we will need:

1. A Raspberry Pi plus SD card (at least 4GB in size)

2. Any one of the following boards:

An Arduino Ethernet board
(<http://arduino.cc/en/Main/arduinoBoardEthernet>) A Spark Core board
(<https://www.spark.io/>)

An Arduino Uno plus ESP8266 board (<http://www.banggood.com/ESP8266-Remote-Serial-Port-WIFI-Transceiver-Wireless-Module-p-947259.html>)

A few resistors and LEDs to test things out

A USB web camera or the Raspberry Pi's camera module

Installing and testing the python-twisted

framework

In this task, we will review the python-twisted framework

(<https://twistedmatrix.com/trac/>). The Twisted framework is an open source event-driven network engine that can be used to send control messages to other devices (for example, a pet feeder can be operated using Raspberry Pi to dispense solid food to our animal friends). The messaging protocol is called **AMP (Asynchronous Messaging Protocol)** which is available at <http://amp-protocol.net/>.

Prepare for lift off

The Raspberry Pi (with an SD card flashed with the OS image) connected to the Internet is needed for this task along with an Arduino Ethernet Board (or any one of the boards mentioned earlier in the checklist. Network connectivity is essential). We will install the python-twisted framework and review a quick example to exchange messages between Arduino and the Raspberry Pi.

Engage thrusters

1. The python-twisted framework can be installed from the terminal of the Raspberry Pi as follows:

sudo apt-get install python-twisted

2. Once the installation is complete, it is time to review an example of the python-twisted framework to test the framework

(<http://twistedmatrix.com/documents/current/downloads/simplest.py>).

1. In this task, we will launch a server on the Raspberry Pi and try to communicate to the server within the local network.

2. Let's modify the preceding example to bind the python-twisted server to the IP

address of Raspberry Pi so that we can exchange messages with other devices on

the network. This device can either be an Arduino or a laptop.

3. We will modify the listenTCP method to bind the server to the IP address of the Raspberry Pi and listen to incoming messages at the port address, 8000.

```
reactor.listenTCP(8000,factory,50,'192.168.1.89')
```

4. The web server can be launched by executing a Python script.

5. Now, using a laptop that is connected to the same network, let's modify the

simpleclient.py script to send and receive messages to the server just

launched on the Raspberry Pi

(<https://twistedmatrix.com/documents/14.0.1/downloads/simpleclient.py>).

6. In order to connect to the server launched on Raspberry Pi, the connectTCP

method needs to be modified to include the IP address that the client has to

connect:

```
reactor.connectTCP('192.168.1.89', 8000, f)
```

7. In this example, the Raspberry Pi is the server and the laptop is the client. The server in this example echoes all incoming messages. Thus, when the client connects and sends a message, the output will be something like:

Server said: Hello, World!

connection lost

Connection lost - goodbye!

3. Now that we have installed the server, let's discuss a simple Arduino sketch to interact with the server launched on the Raspberry Pi. The sketch is available along with this project's downloads (TwistedFrameworkTest.ino).

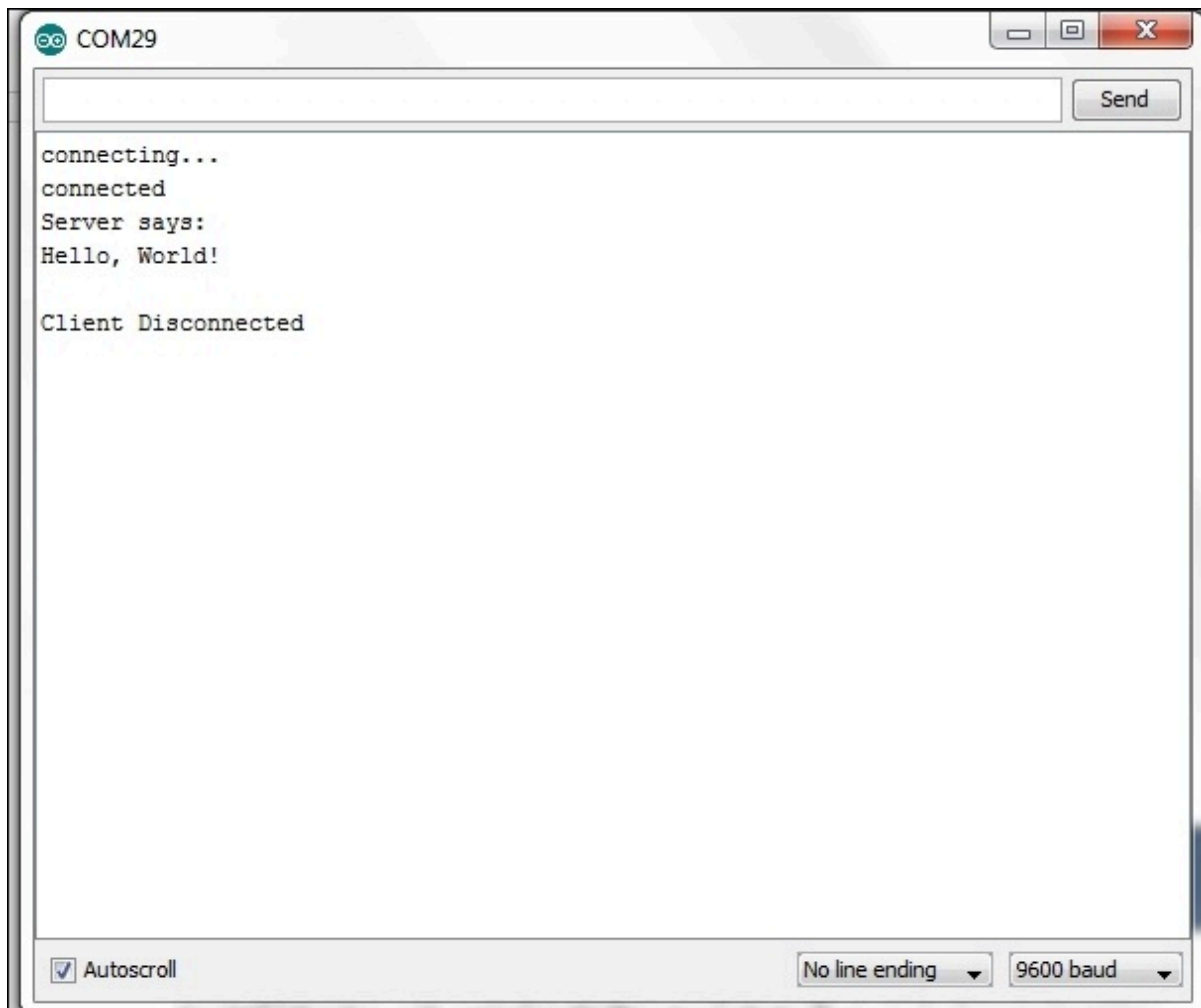
1. We will get started by declaring an IP address object that includes the IP address of Raspberry Pi:

```
//MAC Address of the Arduino
```

```
byte mac[] = { 0x90, 0xA2, 0xDA, 0x0F, 0x02, 0xFC };
```

```
//IP Address of the Raspberry Pi
```

```
IPAddress server( 192, 168, 1, 89);
```



2. The Arduino acts as a TCP client and connects to the Raspberry Pi:

```
if (client.connect(server, 8000)) {  
  Serial.println("connected");  
  client.println("Hello, World!");  
  client.println();  
  //Lets wait for the client to read and  
  //echo the message  
  //Note: A second's delay is a bit excessive  
  delay(1000);  
  //If there is a response from the server  
  //echo back the message  
  Serial.println("Server says:");  
  while(client.available()) {  
    char c = client.read();  
    Serial.print(c);  
  }  
  client.stop();  
  Serial.println("Client Disconnected");  
} else {  
  Serial.println("connection failed");  
}
```

}

3. The Arduino client connects to the Raspberry Pi and transmits the **Hello, World!** message. The Raspberry Pi echoes back the message to the Arduino client. A screenshot of the server's response to the client is shown as follows:

Interaction of the Arduino client with the Raspberry Pi

4. We demonstrated the ability to transmit and receive messages between Arduino and the Raspberry Pi.

Objective complete – mini debriefing

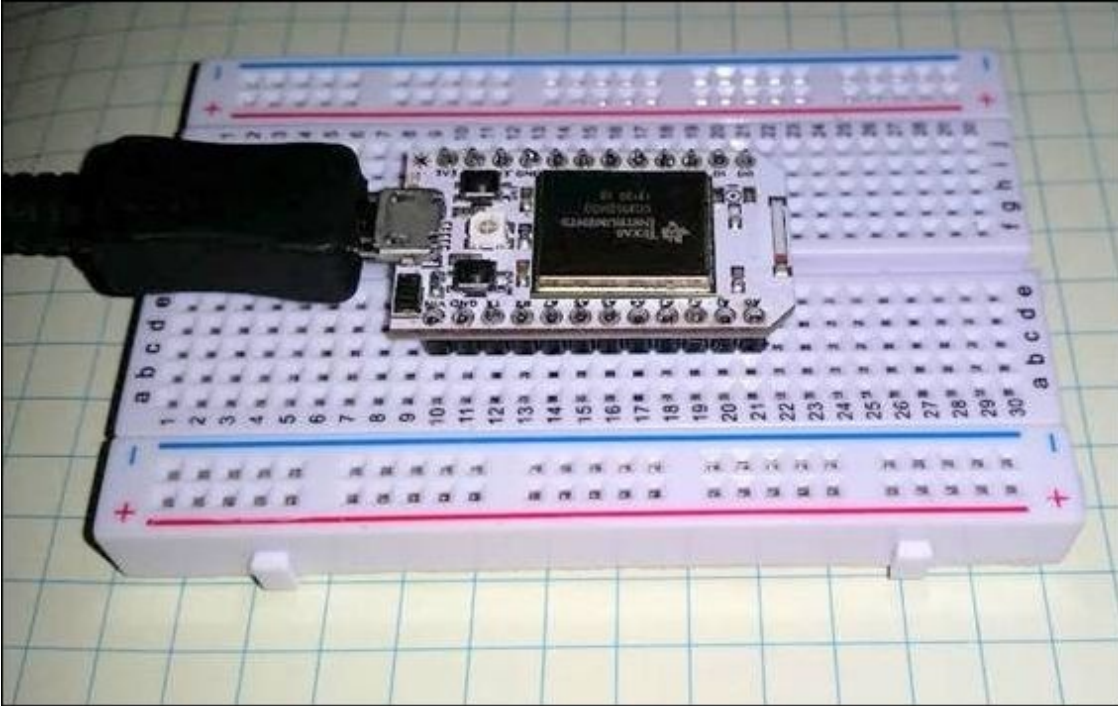
At the end of this project, we will discuss making use of the python-twisted framework and an Arduino Ethernet board to help you control a camera trigger or remotely dispense treats using a pet feeder.

Setting up and testing Spark Core to

interact with appliances

In this task, we will set up Spark Core. It is a development board that makes bringing Wi-Fi to hardware very easy. We will use the Core for this project because we love the simplicity of the setup and operation. You can read more about Spark Core at the official website, <https://www.spark.io/>.

You are welcome to use a similar device of your choosing. A good alternative will be to use an Arduino with a Wi-Fi adapter/shield (for example, the ESP8266 Wi-Fi module can be found at http://www.seeedstudio.com/wiki/WiFi_Serial_Transceiver_Module) or if you prefer, an Ethernet shield (as discussed in the previous task of this project).



Prepare for lift off

In this task, we will set up the Spark Core. Power up the Spark Core using a micro-USB

cable connected to a power source (5 V source). It can be powered using a battery pack, a laptop, or a 5 V USB wall wart. Based on your project needs, you can use a battery pack if the sensor node for the pet monitor has to be installed outdoors.

The easiest way to connect the Spark Core to your Wi-Fi network is through a phone application. This and other mechanisms are detailed on the Spark Core website (<http://docs.spark.io/connect/>). We will set up our Spark Core by downloading the application onto an Android phone and Samsung Galaxy S5 through the Google Play store (<https://play.google.com/store/apps/details?id=io.spark.core.android&hl=en>). The app searches for Spark Cores in range and if you supply the Wi-Fi password, it sends the code to the Core(s) and lets it connect to your Wi-Fi.

The SparkCore board

Engage Thrusters

We will warm up by implementing the LED example on Spark Core or something similar.

The documentation includes code to connect an LED with a resistor to the Core

(<http://docs.spark.io/examples/>). We decided to do something even simpler. There is already an LED connected to pin D7 and we decided to turn this one on instead.

All the apps you need can be downloaded to a custom web page available for each Spark Core through a username and password.

Once you have set up the Core, you will create a username and password. For different Cores, you will need different usernames. Using this, you can go to the Spark home page (<http://www.spark.io/>) and click on the **Build** button. This takes you to the home page of the development environment.

Using this development environment, we created a simple LED blinking application:

```
void setup() {  
  
  pinMode(7,OUTPUT);  
  
}  
  
void loop() {  
  
  // turn the LED on (HIGH is the voltage level)  
  
  digitalWrite(7, HIGH);  
  
  // wait for a second  
  
  delay(1000);  
  
  // turn the LED off by making the voltage LOW
```

```
digitalWrite(7, LOW);
```

```
// wait for a second
```

```
delay(1000);
```

```
}
```

Note

The preceding code is written in C programming language. If you are jumping directly to this project from other projects, we suggest that you familiarize yourself with the Arduino or Spark Core platform. Some learning resources are available at <https://learn.adafruit.com/search?q=Arduino>

As you can see, the app is quite simple; the app sets up the pin 7 as output. The program turns on the LED for a second and turns it off for one second.

Now, let's discuss an example that is more or less similar to the example discussed in the previous task:

```
TCPClient client;
```

```
// IP Address of the Raspberry Pi
```

```
byte server[] = { 192, 168, 1, 89 };
```

```
void setup()
```

```
{
```

```
  Serial.begin(9600);
```

```
  while(!Serial.available()) SPARK_WLAN_Loop();
```

```
  delay(1000);
```

```
  Serial.println("connecting...");
```

```
  if (client.connect(server, 8000)) {
```

```
Serial.println("connected");

client.println("Hello, World!");

client.println();

//Lets wait for the client to read and

//echo the message

//Note: A second's delay is a bit excessive
delay(1000);

//If there is a response from the server

//echo back the message

Serial.println("Server says:");

while(client.available()) {

char c = client.read();

Serial.print(c);

}

client.stop();

Serial.println("Client Disconnected");

} else {

Serial.println("connection failed");

}

}
```

```
void loop()

{

//Nothing to do here

}
```

The only difference between the previous example and this example is that the latter uses the TCPClient class while the former uses the EthernetClient class.

Objective complete – mini debriefing

That's it. You now have Spark Core, all set up and ready to go, with your username, and your own page to download apps and collect and transmit data through the Core.

Installing and testing the Google Coder

project on the Raspberry Pi

This is probably the simplest stage of the project as you should be familiar with setting up the Raspberry Pi for first use.

The Google Coder is a tool released by Google that enables people to learn web

development using the Raspberry Pi. It is a tool that provides an intuitive learning process.

After installing the Google Coder tool, we will test interfacing the GPIO pins of the Raspberry Pi to the web interface.

Prepare for lift off

We need to download the Google Coder image from its repository. The Google Coder

image is available at <http://googlecreativelab.github.io/coder/>.



Engage thrusters

1. Let's get started with setting up Google Coder for the Raspberry Pi. Once the image is downloaded, we have to install the image onto an SD card. If you are not familiar with the installation of the Raspbian image onto an SD card, the first project of this book is really helpful (we think!).

2. Once the Raspberry Pi is powered up after flashing the SD card, MacBook users may launch the tool from a browser by going to <http://coder.local>.

3. Windows users have to install Apple's Bonjour Print Services for Windows

(<http://support.apple.com/kb/DL999>). Once installed, the Google Coder tool can be launched using a Chrome browser. Now, the Google Coder platform can be accessed at <http://coder.local>.

4. The setup process is really simple. If everything went as planned, we should be able to see the landing page of the Google Coder application.

The Google Coder landing page

5. Let's test the *Blinky Lights* example from the Google Coder project available at

http://googlecreativelab.github.io/coder-projects/projects/blinky_lights/. The example provides detailed instruction, including the circuitry and code required to set up a switch to read button states from the GPIO interface of the Raspberry Pi (We chose to skip the explanation as the example already provides a detailed elaboration of the example code).

6. The *Blinky Lights* example demonstrates the use of the node.js based GPIO helper

(<https://www.npmjs.com/package/pi-gpio>) to read/write GPIO pins.

Objective complete – objective complete

We will make use of the three tasks discussed in this project to put together a trigger that comes to eat out of a bird feeder.

A brief overview of a bird feeder that

triggers a camera

In this task, we will set up a bird feeder with a motion sensor. Whenever a bird flies into feed on the grains out of the feeder, the motion sensor is triggered to send a message to the Raspberry Pi. A camera module or a USB camera is connected to the Raspberry Pi, which captures some beautiful moments in our backyard.

Srihari Yamanoor (one of the authors of this book) has an extensive backyard where this bird feeder has been installed. Check out this book's website to find out whether we have captured any beautiful moments thus far!

Prepare for lift off

In order to build this interactive pet feeder, we need the following items:

A pet feeder. (Available from home improvement stores, for example, Home Depot,

Lowes, and so on).

The PIR sensor (<http://www.adafruit.com/product/189>).

Spark Core or any Wi-Fi enabled board.

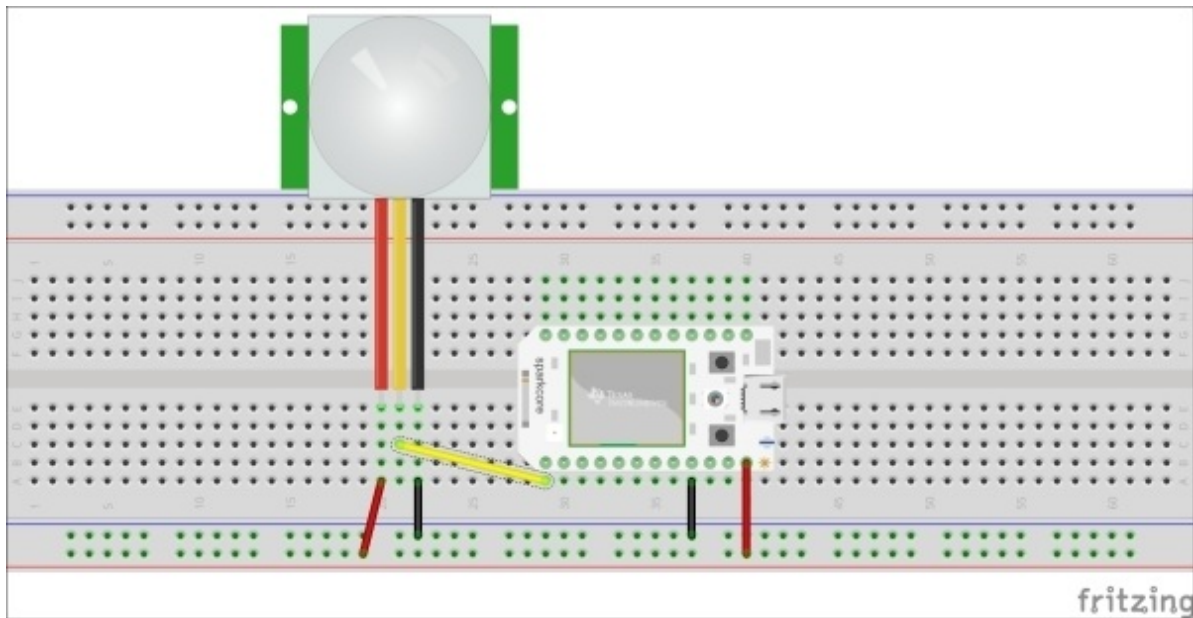
A USB battery pack (<http://www.adafruit.com/products/1959>),rd

(<http://www.adafruit.com/products/64>).

Raspberry Pi with a Twisted framework installed.

A Raspberry Pi camera module with a lens attachment. For example,

<http://store.cutedigi.com/raspberry-pi-camera-module-5mp-wide-angle-160-degree/>.



Engage thrusters

1. The first step involved in programming Spark Core is to detect the motion sensor events and publish those results to Raspberry Pi.
2. The sensor needs to be connected to SparkCore mounted on the breadboard as shown in the following figure:

A motion sensor connected to Spark Core

3. We will modify the example from Adafruit that prints the motion sensor events

<https://learn.adafruit.com/pir-passive-infrared-proximity-motion-sensor/using-a-pir>.

4. Whenever a motion event is detected, we need to send an alert to the python-

twisted server launched on Raspberry Pi:

```
if (client.connect(server, 8000)) {  
  
    Serial.println("Motion Detected");  
  
    client.println("Motion Detected");  
  
    client.println();  
  
    //Lets wait for the client to read and  
  
    //echo the message  
  
    //Note: A second's delay is a bit excessive  
  
    delay(1000);  
  
    client.stop();  
  
    Serial.println("Client Disconnected");  
  
    } else {  
  
    Serial.println("connection failed");  
  
    }  
}
```

5. In order to take pictures of the bird feeder when there is an incoming motion

detection event, the camera has to be interfaced to the Raspberry Pi as demonstrated in the video provided by the Raspberry Pi Foundation (<https://www.youtube.com/watch?v=GImeVqHQzsE>).

6. In order to trigger a camera to take a snapshot using the python-twisted server, we

need the picamera module (<https://pypi.python.org/pypi/picamera>). It can be installed as follows: **sudo apt-get install python-picamera**

7. Let's modify the python-twisted example, simpleserv.py, to import the picamera and datetime modules (to name the pictures captured by the camera using a timestamp).

8. If the incoming message is Motion Detected, we will take a picture and name the file using a timestamp.

9. In the callback method, DataReceived, let's capture an image:

with picamera.PiCamera() as camera:

camera.start_preview()

time.sleep(2)

camera.capture('img{timestamp:%Y-%m-%d-%H-%M}.jpg')



A bird trigger

Objective complete – mini debriefing

Check out this book's website to look at the Raspberry Pi and Spark Sensor setup that is used to take pictures. This is really helpful to learn more about the birds in your neighborhood.

Alternatives and project ideas to consider

1. In this project, we discussed the examples using the Spark Core and the Arduino Ethernet board. Another example to consider is the ESP8266 module, which is available at <http://www.banggood.com/buy/Esp8266.html>, and costs about 3 USD.

Refer to this book's website for setting up the ESP8266 module to emulate these

examples.

2. In [Project 4, Christmas Light Sequencer](#), we used a Power Switch Tail and a relay board example to control decorative appliances. Similarly, we can use them to control automatic pet feeders to dispense treats.

Mission accomplished

In this project, we interfaced built a small ecosystem of devices that were monitored by the Raspberry Pi. We used those examples to build a bird feeder!

Hotshot challenge

In this project, we discussed interfacing a sensor to take pictures of birds that come to feed from the bird feeder. Is it possible to upload the video feed to somewhere that's secure over the Internet? How do we do this?

Chapter 10. Raspberry Pi Personal Health