Intel® Galileo and Intel® Galileo Gen 2

API Features and Arduino Projects for Linux Programmers

Manoel Carlos Ramon

Apress open

Intel® Galileo and Intel® Galileo Gen 2: API Features and Arduino Projects for Linux Programmers

Manoel Carlos Ramon

Copyright © 2014 by Apress Media, LLC, all rights reserved

ApressOpen Rights: You have the right to copy, use and distribute this Work in its entirety, electronically without modification, for non-commercial purposes only. However, you have the additional right to use or alter any source code in this Work for any commercial or non-commercial purpose which must be accompanied by the licenses in (2) and (3) below to distribute the source code for instances of greater than 5 lines of code. Licenses (1), (2) and (3) below and the intervening text must be provided in any use of the text of the Work and fully describes the license granted herein to the Work.

(1) **License for Distribution of the Work:** This Work is copyrighted by Apress Media, LLC, all rights reserved. Use of this Work other than as provided for in this license is prohibited. By exercising any of the rights herein, you are accepting the terms of this license. You have the non-exclusive right to copy, use and distribute this English language Work in its entirety, electronically without modification except for those modifications necessary for formatting on specific devices, for all non-commercial purposes, in all media and formats known now or hereafter. While the advice and information in this Work are believed to be true and accurate at the date of publication, neither the authors nor the editors nor the publisher can accept any legal responsibility for any errors or omissions that may be made. The publisher makes no warranty, express or implied, with respect to the material contained herein.

If your distribution is solely Apress source code or uses Apress source code intact, the following licenses (2) and (3) must accompany the source code. If your use is an adaptation of the source code provided by Apress in this Work, then you must use only license (3).

(2) **License for Direct Reproduction of Apress Source Code:** This source code, from ***Intel® Galileo and Intel® Galileo Gen 2: API Features and Arduino Projects for Linux Programmers, ISBN 978-1-4302-6839-0*** is copyrighted by Apress Media, LLC, all rights reserved. Any direct reproduction of this Apress source code is permitted but must contain this license. The following license must be provided for any use of the source code from this product of greater than 5 lines wherein the code is adapted or altered from its original Apress form. This Apress code is presented AS IS and Apress makes no claims to, representations or warrantees as to the function, usability, accuracy or usefulness of this code.

(3) **License for Distribution of Adaptation of Apress Source Code:** Portions of the source code provided are used or adapted from ***Intel® Galileo and Intel® Galileo Gen 2: API Features and Arduino Projects for Linux Programmers, ISBN 978-1-4302-6839-0*** copyright Apress Media LLC. Any use or reuse of this Apress source code must contain this License. This Apress code is made available at Apress.com/9781430268390 as is and Apress makes no claims to, representations or warrantees as to the function, usability, accuracy or usefulness of this code.

ISBN-13 (pbk): 978-1-4302-6839-0

ISBN-13 (electronic): 978-1-4302-6838-3

Trademarked names, logos, and images may appear in this book. Rather than use a trademark symbol with every occurrence of a trademarked name, logo, or image we use the names, logos, and images only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

The use in this publication of trade names, trademarks, service marks, and similar terms, even if they are not identified as such, is not to be taken as an expression of opinion as to whether or not they are subject to proprietary rights.

While the advice and information in this book are believed to be true and accurate at the date of publication, neither the authors nor the editors nor the publisher can accept any legal responsibility for any errors or omissions that may be made. The publisher makes no warranty, express or implied, with respect to the material contained herein.

Managing Director: Welmoed Spahr
Associate Publisher: Jeffrey Pepper
Lead Editors: Steve Weiss (Apress); Patrick Hauke (Intel)
Coordinating Editor: Melissa Maldonado
Cover Designer: Anna Ishchenko

Distributed to the book trade worldwide by Springer Science+Business Media New York, 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com.

For information on translations, please e-mail rights@apress.com, or visit www.apress.com.

About ApressOpen

What Is ApressOpen?

- ApressOpen is an open access book program that publishes high-quality technical and business information.
- ApressOpen eBooks are available for global, free, noncommercial use.
- ApressOpen eBooks are available in PDF, ePub, and Mobi formats.
- The user-friendly ApressOpen free eBook license is presented on the copyright page of this book.

This book is dedicated to my children—Mariana, Gabriela, and Matheus—and my lovely wife Najla, for their patience during several late nights and working weekends.

Contents at a Glance

About the Author xix

About the Technical Reviewers xxi

Acknowledgments xxiii

Introduction xxv

■Chapter 1: Intel Galileo and Intel Galileo Gen 2 1

■Chapter 2: Native Development 35

■Chapter 3: Arduino IDE and Wiring Language 93

■Chapter 4: New APIs and Hacks 145

■Chapter 5: Networking and Hacks... 217

■Chapter 6: Tweeting with REST API 1.1 289

■Chapter 7: Using OpenCV... 319

■Chapter 8: Creating a Soil Moisture Sensor 401

■Chapter 9: Home Automation and Dynamic Web........................ 425

■Chapter 10: Power over Ethernet (PoE) 499

■Chapter 11: Assembling and Controlling a Robotic Arm 509

■Chapter 12: Using an LTE Modem 579

Appendix A: Intel Galileo I/O and Muxing 605

Appendix B: Intel Galileo Gen 2 I/O and Muxing 611

Appendix C: Video Capturing 617

Appendix D: Picture Grabber 633

Index 643

Contents

About the Author xix

About the Technical Reviewers xxi

Acknowledgments xxiii

Introduction xxv

■Chapter 1: Intel Galileo and Intel Galileo Gen 2 1

The Beginnings.. 1

About this Book 2

Why Use Intel Galileo Boards?.. 3

The Software Advantages 4

The Hardware Advantages 4

Hardware Overview 6

The Processor: Intel Quark SoC X1000 6

Introducing Intel Galileo 8

Introducing Intel Galileo Gen 2... 17

Preparing Your Cables 23

The Serial Cable for Intel Galileo 24

The Serial Cable for Intel Galileo Gen 2 26

Testing the Serial Cables 27

Exploring the Linux Console 32

Testing the Data Cables 33

Summary.. 33

Chapter 2: Native Development ... 35

Introduction to the Yocto Build System ... 35

Yocto and this Book ... 36

Creating Your Own Intel Galileo Images ... 44

Preparing Your Computer ... 44

The SPI vs. SD Card Images ... 46

Building Intel Galileo Images ... 47

Building and Using the Cross-Compiler Toolchain ... 52

Creating a Hello World! ... 56

Debugging Native Applications ... 59

Booting Intel Galileo with Your Own Images ... 59

Booting from SD Card Images ... 59

Booting from SPI Card Images ... 64

What to Do If Intel Galileo Bricks ... 90

Summary ... 91

Chapter 3: Arduino IDE and Wiring Language ... 93

A Little Bit of History ... 93

The Intel Makers Community ... 94

Installing the Arduino IDE for Intel Galileo ... 94

Connecting Intel Galileo ... 96

Installing the Drivers and the Arduino IDE ... 97

Installing the Arduino IDE ... 97

Installing the IDE on Linux ... 98

Installing the Drivers ... 99

Understanding the Arduino IDE ... 101

Checking the Port and Board Selected ... 103

What Is a Sketch? 105

Compiling and Running the Sketch 106

Persisted and Not Persisted Sketches 107

Persisted Sketches and Long Time to Start 107

Debugging with Serial Console and Serial Communication 108

Serial.begin(int speed) 108

Serial.print(data) 108

Serial.println(data) 108

Serial.available( ) 109

Serial.read( ) 109

Printing Debug Messages and Using the Serial Console 109

The Arduino Language Reference and APIs 111

Structure 112

Digital I/O 113

Analog I/O 113

Time 115

Running Some Examples 116

Fade Example 116

Button Example 119

ReadAnalogVoltage Example 122

The Debounce Example 124

Updating the Firmware Using the IDE 127

Updating the Firmware with Different Firmware 129

Troubleshooting the Drivers 130

Serial Communication Issues with IDE on Windows 130

IDE Problems with Virtual Machines and 64-Bit Linux 135

Communicating Sketches with Linux Native Programs 136

is.Additionalfuture Project Example: Unread Email Alarm with Python and POSIX Functions 137

Summary 143

Chapter 4: New APIs and Hacks 145

Servo API 145

The Theory versus Practice 145

The Mistake with Intel Galileo and Servos 150

What Is New in Servo API? 152

Serial, Serial1, and Serial2 Objects 157

Testing the Serial, Serial1, and Serial2 Objects 157

Improving the I/O Speed 162

The New APIs for I/O 163

The Tone API 184

What's New in the Tone API? 184

The pulseIn API 188

unsigned long pulseIn(uint8_t pin, uint8_t state, unsigned long timeout = 1000000) 188

What's New with pulseIn() 189

A Sample Running pulseIn() 189

Hacks 190

Hacking the Servo Library 190

Hacking the GPIO Expander for New PWM Frequencies 191

Single Code for Intel Galileo and Intel Galileo Gen 2 194

Project: DHT Sensor Library with Fast I/O APIs 195

Materials List 195

The DHT Sensor 195

A Workaround Using Tri-State Buffers 198

Creating a New Library for DHT11 Sensor 202

Summary 216

Chapter 5: Networking and Hacks 217

WiFi Cards 218

Setting Up the WiFi Mini-PCIe Card 218

Checking if the WiFi Card Was Recognized 222

Adding Support to a New WiFi Card 222

The WiFi API 224

Scanning the Wireless Networks 225

Connecting to the WPA or WEB 230

Hacking the WiFi Library 239

Step 1: Setting Up the WiFi Connection 239

Step 2: Restart the Wireless Connection 242

Step 3: Hacking the WiFi Class 243

Ethernet API 247

What's New with Ethernet API and Intel Galileo 248

Ethernet Example: Network Time Protocol (NTP) 248

Dynamic and Static IP Using Ethernet 253

Simplifying the Sketches by Removing the Ethernet Objects 271

Transferring Files Between Intel Galileo and Computers 271

Using ftp 272

Using scp or pscp 273

Using an SD Card 273

Using a USB Pen Drive 274

Hacking the IDE to Improve the Sketch Transfer 276

How the Hacked IDE Works 277

Hacking the Transfers 279

Summary 287

Chapter 6: Tweeting with REST API 1.1 289

Project Details 289

Material List 290

System and Hardware Clock 290

Assembling the Coin Battery 292

Workaround with 1.5V Batteries 293

Creating a Twitter Application Account 294

Step 1: Creating Your Twitter Account 294

Step 2: Creating the Twitter Application 296

Creating a Script to Post Messages on Twitter 301

Step 1: Identifying the API 301

Step 2: Generating the OAuth Signature with the OAuth Tool 303

Step 3: Creating a Script to Post on Twitter with Intel Galileo 308

Step 4: Creating a Sketch that Tweets 312

Ideas for Improving the Project 315

Flowers and Plants Communicating with You on a Social Network 315

Summary 318

Chapter 7: Using OpenCV 319

OpenCV Primer 319

Project Details 320

Materials List 321

USB Video Class 322

Preparing the BSP Software Image and Toolchain 323

Using eGlibc for Video4Linux Image 324

Increasing the rootfs Size 324

Disabling GPU Support on OpenCV 324

Building the SD Image and Toolchain 325

Development Library Packages 325

Connecting the Webcam 326

Introduction to Video4Linux 327

Exploring the Webcam Capabilities with V4L2-CTL 327

- Changing and Reading Camera Properties 329
- Pixel Formats and Resolution 330

Capturing Videos and Images with libv4l2 333

- A Program for Capturing Video 333
- A Program to Capture Images 347

Working with OpenCV 352

- Building Programs with OpenCV 353
- Capturing an Image with OpenCV 353
- The Same Software Written in Python 359
- Processing Images 360
- Detecting Edges 361
- Face and Eyes Detection 364
- Emotions Classification 372

Ideas for Improving the Project 399

- Integrating Your Emotions with a Robotic Head 400
- Expanding the Classifications 400
- Improving the Emotion Classification Using Large Databases 400

Improving the Emotion Classification for Several Faces 400

Summary 400

Chapter 8: Creating a Soil Moisture Sensor 401
Project Details 401
Material List 402
Assembling the Moisture Sensors.. 403
Mechanical Assembly 403
Electrical Assembly 404
Assembling the Flower Face 406
Testing the Flower Face with the Software 410
Reviewing the Code 413
Running the Project.. 416
Calibration Procedure 416
Ideas for Improving the Project 421
Increasing the Number of Sensors 421
Automatic Irrigation 421
Using Appropriate Wires 423
Using a Commercial Sensor 423
Tweeting 424
Summary... 424
Chapter 9: Home Automation and Dynamic Web 425
Project Details 425
The Software Architecture 426
Materials List.. 428
Integrating the Components Individually... 428
Testing the Keypad 428
Testing the PIR Sensor 438
The PIR Sensor Connection 440
Testing the YwRobot Relay Module 443
Testing the TMP36 Temperature Sensor 447

Creating the Sketch 451
Sending UDP Messages 451
Receiving UDP Messages 452
Joining All Code in a Single Sketch 455
Creating Your Own Web Server with node.js 465
Updating node.js 465
About the npm 467
Installing Cheerio 469
Installing socket.io 469
The Web Page 470
Writing the Web Server Code 474
Running the Home Automation System 486
Ideas for Improving the Project 489
Power of Ethernet (PoE) 489
Using express and node.js 489
Changing the Web Page and Web Server Without Experience with the Web 490
Creating an Analogic Keypad and Having More I/Os Available 495
Adding a Username and Password 497
Using the DHT11 Sensor 497
Summary 497
Chapter 10: Power over Ethernet (PoE) 499
Project Details 499
Materials List 500
Assembling the PoE Module 500
Powering On with the PoE Injector 504
How It Works 506
Ideas for Improving the Project 506
Summary 507

Chapter 11: Assembling and Controlling a Robotic Arm 509
An introduction to Robotic Arm Theory 510
Using an Online Tool to Calculate Your Servos... 514
Considerations About This Project 516
The Robotic Arm 516
Assembling the Robotic Arm 519
Step 1: Preparing the Servos.. 521
Step 2: Assembling the Base 522
Step 3: Assembling the Shoulder... 526
Step 4: Assembling the Elbow 528
Step 5: Assembling the Wrist.. 531
Step 6: Assembling the Mechanical Gripper... 532
Step 7: Assembling a Base 535
Controlling the Robotic Arm 537
Building a Servo Control Board... 538
The Hardware 538
Assembling the Board 544
The Software 547
A Gripper Based on Coffee and a Balloon.. 570
Preparing the Coffee Gripper.. 573
A Sketch for the Coffee Gripper.. 575
Ideas for Improving this Project 577
Summary... 577
Chapter 12: Using an LTE Modem 579
An Introduction to XMM7160 and XMM7260..................................... 580
Project Details 581
Materials List.. 581
Considerations Related to Antennas 582

Preparing the Hardware 583

Step 1: Preparing the SIM Card 583

Step 2: The NGFF/mPCIe Adaptor 585

Step 3: Adapting the Modem Card 586

Step 4: Connecting the Antennas 587

Step 5: Connecting the Modem to Intel Galileo 588

Preparing the Software 589

Step 1: Checking the Modem 589

Step 2: Loading the CDC-ACM Driver 590

Step 3: Checking if the Modem Responds to AT Commands 591

Step 4: Checking the SIM Card Connection 592

Step 5: Configuring the APN and Attaching it to the Network 593

Step 6: Creating an IP Interface with pppd 596

Step 7: Testing the Internet Connection 600

Testing the Internet Bandwidth 600

Why Aren't the Download/Upload Bandwidths Optimal? 601

Ideas for Improving the Project 601

Improving the Chat Script 601

Loading the cdc-acm Driver Automatically 603

Summary 604

Appendix A: Intel Galileo I/O and Muxing 605

Appendix B: Intel Galileo Gen 2 I/O and Muxing 611

Appendix C: Video Capturing 617

Appendix D: Picture Grabber 633

Index 643

About the Author

Manoel Carlos Ramon was born in Brazil. He started learning software programming when he was 11 years old, using a TK-85—an 8-bit computer based on a Z80 processor with only 16KB of RAM. His passion for programming and understanding computers started with watching *Bit & Bytes*, a Canadian TV series that aired on TV Cultura, a Brazilian public TV network.

A few years later, Manoel created his first pirate FM transmitter that, instead of music, transmitted programs written in Basic to exchange software saved on cassette tapes with his friends via radio. He created his first commercial software in 1986, at age 13. This was software for storage control, written in DbaseII and running on a CP500/M80 computer. He learned and programmed in Z80 Assembler using his first MSX computer in 1988. Later he bought an IBM PC XT with an Intel 8080 processor, on which he created the first Brazilian software able to compute with precision the lux intensity of irregular surfaces based on photometric curves, written in Turbo C.

Manoel earned a BsSc in electric engineering in 1996, graduating with honors. He has been working in research and development of mobile devices and tablets since graduation. Manoel contributed to porting the first Java virtual machine fully certified by Sun Microsystems in Motorola phones. He has also worked with Linux kernel and device drivers, J2ME, Android NDK/SDK/ADK, audio path, power management, and modem 2G/3G/4G integration, including the first tablet with an LTE modem released in South America.

Manoel has worked on several projects in the Americas, Asia, and Europe. He migrated to the United States in 2011. Manoel is currently part of Intel Corporation's New Devices Group, where he is a member of the team responsible for the development of the Intel Galileo, the Intel Galileo Gen 2, and the Intel Edison Kit for Arduino.

Manoel lives in California with his wife and children. He admires Dr. Steve Mann, who is considered the real father of wearable computers, and David Rolfe, a notable Assembler programmer who created classic arcade games in the 1980s.

About the Technical Reviewers

Krzysztof Sywula joined the Intel Corporation in 2010. He was initially assigned to the LAN Access Division, which provided him a background in UEFI and the Ethernet. He joined the Quark Software team in 2012, where he works on different areas of the chip, gaining an extensive and deep understanding of the workflows of low-level components.

Dino Tinitigan graduated from the University of Nevada Las Vegas (UNLV) with a BS in computer engineering. He is currently a software engineer for Intel, working on maker boards like the Galileo and the Edison. In his spare time he tinkers with robotics, home automation, and wearables.

Acknowledgments

Many people contributed in different ways along the process of writing this book. I hereby express my deepest gratitude to them:

The Apress team, who made this book possible, especially Melissa Maldonado, James Markham, and Steve Weiss.

Patrick Hauke, the Intel Technology Book Program Manager who believed in me and in the idea of this book.

Dino Tinitigan and Krzysztof Sywula for their excellent contributions on this book as technical reviewers, and also as part of the Intel Galileo development team.

Jorgen Ostulan and Dan Nguyen for their support with antenna selections for LTE frequencies and XMM7160 modem information.

Kevin Clarke for his collaboration as tester and developer, going deeply into the Linux signals issues.

Daniel O'Donovan for his amazing collaboration in developing not only the Arduino IDE, but also Linux drivers.

Francis Crowe and Srikanth Vasuki for the fantastic job designing, supporting, and validating the Intel Galileo Gen 2 hardware.

I would especially like to say that I am very thankful to the whole open source community that works continuously on the Linux operation system and on the Arduino IDE. Of course, I could not list all names here, but I would like to express my gratitude in a Linux bash terminal:

```
cd $LINUX_REPOSITORIES; git log --pretty=format:"%an"|sort -u
cd $ARDUINO_REPOSITORIES; git log --pretty=format:"%an"|sort -u
```

Introduction

The Intel Galileo maker development board was announced on October 4, 2013 at the Rome Maker Fair. It was the first for the Intel Corporation. The board was based on the Intel Quark SoC X1000 application processor, powered by Linux OS, and compatible with Arduino reference APIs. The Intel Galileo introduced several extra software and hardware features that exceeded most other Arduino boards available on the market.

Intel received thousands of feedback comments from makers on Intel's Maker forums. Based on this feedback, Intel later unveiled the Intel Galileo Gen 2, a new version with the same Intel Quark SoC application processor but improved hardware.

This book discusses the hardware and the software for the Intel Galileo and the Intel Galileo Gen 2, introducing Arduino reference APIs and APIs created especially for Intel Galileo boards. Practical projects show how to make use of the Linux capabilities and to aggregate this potential in Arduino sketches.

What Is in this Book?

Chapter 1 discusses the hardware design of the Intel Galileo and the Intel Galileo Gen 2, as well as the construction of serial and FTDI cables for debugging using Linux terminal consoles.

Chapter 2 explains how the Yocto build system works and how to generate your custom SPI and SD card images. It also presents how to compile, install, and use the toolchains for native applications development, and discusses procedures to recover bricked Intel Galileo boards.

Chapter 3 shows how to install and use the Arduino IDE, and how to install the drivers needed in the computer or virtual machine used, running real examples of interacting sketches with simple circuits. It also brings a practical project that integrates Python, POSIX calls, and sketches to send an alert when an email is received.

Chapter 4 discusses the new APIs and hacking techniques created especially for the Intel Galileo and Intel Galileo Gen 2 boards. It contains a broad discussion of clusters architecture, and how GPIOs are distributed and their respective speed limits. A practical project demonstrates how to overcome the Intel Galileo's limitations and how to make the DHT11 temperature sensor work.

Chapter 5 presents networking APIs and hackings using an Ethernet adapter and WiFi mPCIe cards. It also explains how to install new WiFi cards and how to share Internet access between the Intel Galileo and computers. This chapter also explains how to hack the Arduino IDE to download sketches using network interfaces instead of a USB.

Chapter 6 offers a practical project on tweeting using Intel Galileo boards with new OAuth authentication and without intermediary computers or servers. The project uses a RTC (real-time clock) with external coin batteries and WiFi mPCIe cards.

Chapter 7 shows techniques using V42L and OpenCV libraries, as well as how to capture images and videos and detect facial expressions and emotions using a webcam. This chapter also explains how to change the Linux BSD to support eglibc instead uClibc and to generate the toolchain to compile C/C++ programs. There are also examples of OpenCV in Python.

Chapter 8 presents a low-cost project to create moisture sensors based in scrap materials and galvanized nails.

Chapter 9 shows a practical home automation project implementing a web server using node.js, interacting with multiple sensors for motion and temperature, and using keypads and switch relays.

Chapter 10 explains how to install and use PoE (Power of Ethernet) modules with the Intel Galileo Gen 2.

Chapter 11 discusses basic principles in robotics and how to design and control a robotic arm using analog controllers. It also presents a practical project using a 6 DOF robotic arm with a mechanical gripper and another one built with ground coffee.

Chapter 12 discusses how to connect a XMM 7160 LTE modem and use data channels in real networks using Intel Galileo boards.

Chapter 13 is a bonus chapter available online. It presents a practical project on how to design and build a low-cost robot head with animatronic eyes and a mouth that expresses emotions. This chapter is available online at `http://www.apress.com/9781430268390`, under the Source Code/Downloads tab.