## Appendix 1: Comparison of Commands with Other SDKs

***Table A1-1***Comparison with Other SDKs

|  |  |  |  |
| --- | --- | --- | --- |
| **Task** | **ArduPilot command In C++** | **Dronekit command In Python** | **PX4 C++ messages ROS-kind logic** |
| Screen interface: reading and writing | hal.console->printf()  hal.console->read() | Print  input() or raw\_input() | PX4\_INFO |
| Remote control reading or RC input | hal.rcin->read() | vehicle.channels[channel] | manual\_control\_setpoint.msg |
| Writing to motors or RC output | hal.rcout->write() | NOT TO MOTORS, ONLY TO VEHICLE:  send\_ned\_velocity(vx,vy,vz,duracion ) | actuator\_controls.msg  to vehicle  actuator\_direct.msg  test\_motor.msg  to motors |
| Analog reading | ch=hal.analogin->channel(channel)  read=ch->voltage\_average() | EXTERNAL, for example, analog ports on Raspberry Pi | adc\_report.msg |
| Digital reading and writing or GPIO | hal.gpio->read(pin)  hal.gpio->write(pin,value) | EXTERNAL, for example, GPIO libraries on Raspberry Pi | gpio\_led start |
| Wired or wireless serial UART reading/writing | hal.uart#->write()  hal.uart#->read()  with #=A,B,C, D | EXTERNAL, for example, with Raspberry Pi | modules/mavlink |
| Writing to SD card | DataFlash.WriteBlock(package,size) | EXTERNAL by using Raspberry or another command computer withfile.write  ormavutil.mavlink\_connection(filename) | <https://dev.px4.io/en/log/ulog_file_format.html> |
| Time invocation or delays | hal.scheduler->millis()  hal.scheduler->delay() | time.sleep(time) | hrt\_absolute\_time(); |
| Battery reading | battery.read( ) | vehicle.battery | battery\_status.msg |
| GPS reading | inertial\_nav.get\_position()  inertial\_nav.get\_velocity(); | vehicle.gps\_0 | vehicle\_gps\_position.msg  vehicle\_odometry.msg |
| Orientation reading | Ahrs  ins.get\_gyro() | vehicle.attitude | vehicle\_odometry.msg  sensor\_combined.msg |
| Altitude reading | barometer.get\_altitude() | vehicle.location | vehicle\_gps\_position.msg  vehicle\_odometry.msg |
| LED lightning | toshiba\_led.set\_rgb(R,G,B); | EXTERNAL | led\_control.msg |
| Signal filtering | LowPassFilter2pfloat filtername(parameters)  filtername.apply(signal); | EXTERNAL, additional Python apps | <https://dev.px4.ii/en/middleware/modules_estimesti.html> |
| Observations | Very complete libraries to fully use the autopilot, widely documented, and easy to use | Libraries for limited use, easy to implement, good documentation | The most complete libraries, difficult to use and with ROS style, not well documented |
| Webpage | <http://ardupilot.org/dev/docs/apmcopter-programming-libraries.html> | [http://python.dronekit.io](http://python.dronekit.io/) | <https://dev.px4.io/en/> |

## Appendix 2: Setup Extended Code

The following lines must be placed in each new program. It is advisable not to omit any of them in order to avoid incurring errors due to omission of data. In the best case, it is only advisable to add more lines of code as needed or to encapsulate certain specific lines in a function, as previously illustrated in the case of SD memory writing.

void setup()

{

ins.init(AP\_InertialSensor::COLD\_START,AP\_InertialSensor::RATE\_400HZ);

serial\_manager.init\_console();

serial\_manager.init();

compass.init();

compass.read();

ahrs.set\_compass(&compass);

gps.init(NULL,serial\_manager);

barometer.init();

barometer.calibrate();

DataFlash.Init(log\_structure, sizeof(log\_structure)/sizeof(log\_structure[0]));

if (DataFlash.NeedErase()) {

DataFlash.EraseAll();

}

log\_num = DataFlash.StartNewLog();

hal.scheduler->delay(100);

}

As you can see, the basic setup consists of starting the serial console (to at least send messages to a terminal), and then initializing the compass, the inertial sensor, the GPS, the barometer , and the combined module AHRS (so that the autopilot has a notion of its position and spatial orientation), and finally the storage module to the SD card. This initializer is the base mode and sometimes, as illustrated in the pertinent sections, it will be necessary to add lines (for items such as motors, LEDs, batteries, UART serial communication, and analog and digital ports).

## Appendix 3: Extended Header

This header information must be placed in each code file so that the programs can be executed.

The following header lines must be added to each new code file. They contain the invocation of all the necessary functions of the ArduPilot libraries. It is suggested to not modify them and in the best case, only add the necessary library. Do not remove any line of code if you are not sure about the library or command to be removed. Note that they have been taken almost completely from theardupilot.pde file. Just copy and paste the following:

// place the header here //

// c libraries

**#include** <math.h>

**#include** <stdio.h>

**#include** <stdlib.h>

**#include** <stdarg.h>

// Common dependencies

**#include** <AP\_Common.h>

**#include** <AP\_Progmem.h>

**#include** <AP\_Menu.h>

**#include** <AP\_Param.h>

**#include** <StorageManager.h>

// AP\_HAL

**#include** <AP\_HAL.h>

**#include** <AP\_HAL\_AVR.h>

**#include** <AP\_HAL\_SITL.h>

**#include** <AP\_HAL\_PX4.h>

**#include** <AP\_HAL\_VRBRAIN.h>

**#include** <AP\_HAL\_FLYMAPLE.h>

**#include** <AP\_HAL\_Linux.h>

**#include** <AP\_HAL\_Empty.h>

**#include** <AP\_Math.h>

// Application dependencies

**#include** <GCS.h>

**#include** <GCS\_MAVLink.h> // MAVLink GCS definitions

**#include** <AP\_SerialManager.h> // Serial manager library

**#include** <AP\_GPS.h> // ArduPilot GPS library

**#include** <DataFlash.h> // ArduPilot Mega Flash Memory // Library

**#include** <AP\_ADC.h> // ArduPilot Mega Analog to // Digital Converter Library

**#include** <AP\_ADC\_AnalogSource.h>

**#include** <AP\_Baro.h>

**#include** <AP\_Compass.h> // ArduPilot Mega Magnetometer // Library

**#include** <AP\_Math.h> // ArduPilot Mega Vector/Matrix // math Library

**#include** <AP\_Curve.h> // Curve used to linearlise // throttle pwm to thrust

**#include** <AP\_InertialSensor.h> // ArduPilot Mega Inertial // Sensor (accel & gyro) Library

**#include** <AP\_AHRS.h>

**#include** <AP\_NavEKF.h>

**#include** <AP\_Mission.h> // Mission command library

**#include** <AP\_Rally.h> // Rally point library

**#include** <AC\_PID.h> // PID library

**#include** <AC\_PI\_2D.h> // PID library (2-axis)

**#include** <AC\_HELI\_PID.h> // Heli specific Rate PID // library

**#include** <AC\_P.h> // P library

**#include** <AC\_AttitudeControl.h> // Attitude control library

**#include** <AC\_AttitudeControl\_Heli.h> // Attitude control // library for traditional // helicopter

**#include** <AC\_PosControl.h> // Position control library

**#include** <RC\_Channel.h> // RC Channel Library

**#include** <AP\_Motors.h> // AP Motors library

**#include** <AP\_RangeFinder.h> // Range finder library

**#include** <AP\_OpticalFlow.h> // Optical Flow library

**#include** <Filter.h> // Filter library

**#include** <AP\_Buffer.h> // APM FIFO Buffer

**#include** <AP\_Relay.h> // APM relay

**#include** <AP\_ServoRelayEvents.h>

**#include** <AP\_Camera.h> // Photo or video camera

**#include** <AP\_Mount.h> // Camera/Antenna mount

**#include** <AP\_Airspeed.h> // needed for AHRS build

**#include** <AP\_Vehicle.h> // needed for AHRS build

**#include** <AP\_InertialNav.h> // ArduPilot Mega inertial // navigation library

**#include** <AC\_WPNav.h> // ArduCopter waypoint // navigation library

**#include** <AC\_Circle.h> // circle navigation library

**#include** <AP\_Declination.h> // ArduPilot Mega Declination // Helper Library

**#include** <AC\_Fence.h> // Arducopter Fence library

**#include** <SITL.h> // software in the loop support

**#include** <AP\_Scheduler.h> // main loop scheduler

**#include** <AP\_RCMapper.h> // RC input mapping library

**#include** <AP\_Notify.h> // Notify library

**#include** <AP\_BattMonitor.h> // Battery monitor library

**#include** <AP\_BoardConfig.h> // board configuration library

**#include** <AP\_Frsky\_Telem.h>

**#if** SPRAYER == ENABLED

**#include** <AC\_Sprayer.h> // crop sprayer library

**#endif**

**#if** EPM\_ENABLED == ENABLED

**#include** <AP\_EPM.h> // EPM cargo gripper stuff

**#endif**

**#if** PARACHUTE == ENABLED

**#include** <AP\_Parachute.h> // Parachute release library

**#endif**

**#include** <AP\_LandingGear.h> // Landing Gear library

**#include** <AP\_Terrain.h>

**#include** <LowPassFilter2p.h>

// AP\_HAL to Arduino compatibility layer

**#include** "compat.h"

// Configuration

**#include** "defines.h"

**#include** "config.h"

**#include** "config\_channels.h"

// lines referring to the times of the pixhawk autopilot, which

// works at 400mhz or 0.0025seconds or 2500 microseconds

# define MAIN\_LOOP\_RATE 400

# define MAIN\_LOOP\_SECONDS 0.0025f

# define MAIN\_LOOP\_MICROS 2500

// statements referring to the autopilot objects, for example,

// gps-type objects barometer, compass, DataFlash, etc., all of

// them will subsequently be invoked in the corresponding code

const AP\_HAL::HAL& hal = AP\_HAL\_BOARD\_DRIVER;

static AP\_Scheduler scheduler;

static AP\_GPS gps;

static AP\_Baro barometer;

static AP\_InertialSensor ins;

static RangeFinder sonar;

static Compass compass;

static AP\_SerialManager serial\_manager;

static ToshibaLED\_PX4 toshiba\_led;

static AP\_BattMonitor battery;

//Data, BE CAREFUL, WHEN YOU READ THE SD SECTION DELETE THIS //BLOCK THERE

// you will learn how to use and external module and deep // details about these declarations

#define LOG\_MSG 0x01

#if CONFIG\_HAL\_BOARD == HAL\_BOARD\_PX4

static DataFlash\_File DataFlash("/fs/microsd/APM/LOGS");

#endif

struct PACKED log\_Datos{

LOG\_PACKET\_HEADER;

uint32\_t time\_ms;

float a\_roll;

float a\_pitch;

float a\_yaw;

float pos\_x;

float pos\_y;

float pos\_z;

};

static const struct LogStructure log\_structure[] PROGMEM = {

LOG\_COMMON\_STRUCTURES,

{LOG\_MSG, sizeof(log\_Datos),

"1", "Iffffff", "T\_MS,ROLL,PITCH,YAW,X\_POS,Y\_POS,Z\_POS"},

};

static uint16\_t log\_num; //Dataflash

// Inertial Navigation EKF

#if AP\_AHRS\_NAVEKF\_AVAILABLE

AP\_AHRS\_NavEKF ahrs(ins, barometer, gps, sonar);

#else

AP\_AHRS\_DCM ahrs(ins, barometer, gps);

#endif

static AP\_InertialNav\_NavEKF inertial\_nav(ahrs);

// place your code here //

Previously an abbreviated description of these libraries was made. For a more complete reference, consult the following documentation:

Alejandro Romero Galan, “Revision y modificacion del firmware de libre acceso arducopter para su uso en el proyecto airwhale,” Thesis, Universidad de Sevilla, 2015 (in Spanish).

<http://ardupilot.org/dev/docs/apmcopter-programming-libraries.html>

<https://github.com/ArduPilot/ardupilot/tree/master/libraries>

## Appendix 4: The Fully Functional Code

Here is the complete code , including the header and setup information:

// paste the header here //

// c libraries

#include <math.h>

#include <stdio.h>

#include <stdlib.h>

#include <stdarg.h>

// Common dependencies

#include <AP\_Common.h>

#include <AP\_Progmem.h>

#include <AP\_Menu.h>

#include <AP\_Param.h>

#include <StorageManager.h>

// AP\_HAL

#include <AP\_HAL.h>

#include <AP\_HAL\_AVR.h>

#include <AP\_HAL\_SITL.h>

#include <AP\_HAL\_PX4.h>

#include <AP\_HAL\_VRBRAIN.h>

#include <AP\_HAL\_FLYMAPLE.h>

#include <AP\_HAL\_Linux.h>

#include <AP\_HAL\_Empty.h>

#include <AP\_Math.h>

// Application dependencies

#include <GCS.h>

#include <GCS\_MAVLink.h> // MAVLink GCS definitions

#include <AP\_SerialManager.h> // Serial manager library

#include <AP\_GPS.h> // ArduPilot GPS library

#include <DataFlash.h> // ArduPilot Mega Flash Memory // Library

#include <AP\_ADC.h> // ArduPilot Mega Analog to // Digital Converter Library

#include <AP\_ADC\_AnalogSource.h>

#include <AP\_Baro.h>

#include <AP\_Compass.h> // ArduPilot Mega Magnetometer // Library

#include <AP\_Math.h> // ArduPilot Mega Vector/Matrix // math Library

#include <AP\_Curve.h> // Curve used to linearlise // throttle pwm to thrust

#include <AP\_InertialSensor.h> // ArduPilot Mega Inertial // Sensor (accel & gyro) Library

#include <AP\_AHRS.h>

#include <AP\_NavEKF.h>

#include <AP\_Mission.h> // Mission command library

#include <AP\_Rally.h> // Rally point library

#include <AC\_PID.h> // PID library

#include <AC\_PI\_2D.h> // PID library (2-axis)

#include <AC\_HELI\_PID.h> // Heli specific Rate PID // library

#include <AC\_P.h> // P library

#include <AC\_AttitudeControl.h> // Attitude control library

#include <AC\_AttitudeControl\_Heli.h> // Attitude control // library for traditional // helicopter

#include <AC\_PosControl.h> // Position control library

#include <RC\_Channel.h> // RC Channel Library

#include <AP\_Motors.h> // AP Motors library

#include <AP\_RangeFinder.h> // Range finder library

#include <AP\_OpticalFlow.h> // Optical Flow library

#include <Filter.h> // Filter library

#include <AP\_Buffer.h> // APM FIFO Buffer

#include <AP\_Relay.h> // APM relay

#include <AP\_ServoRelayEvents.h>

#include <AP\_Camera.h> // Photo or video camera

#include <AP\_Mount.h> // Camera/Antenna mount

#include <AP\_Airspeed.h> // needed for AHRS build

#include <AP\_Vehicle.h> // needed for AHRS build

#include <AP\_InertialNav.h> // ArduPilot Mega inertial // navigation library

#include <AC\_WPNav.h> // ArduCopter waypoint // navigation library

#include <AC\_Circle.h> // circle navigation library

#include <AP\_Declination.h> // ArduPilot Mega Declination // Helper Library

#include <AC\_Fence.h> // Arducopter Fence library

#include <SITL.h> // software in the loop support

#include <AP\_Scheduler.h> // main loop scheduler

#include <AP\_RCMapper.h> // RC input mapping library

#include <AP\_Notify.h> // Notify library

#include <AP\_BattMonitor.h> // Battery monitor library

#include <AP\_BoardConfig.h> // board configuration library

#include <AP\_Frsky\_Telem.h>

#if SPRAYER == ENABLED

#include <AC\_Sprayer.h> // crop sprayer library

#endif

#if EPM\_ENABLED == ENABLED

#include <AP\_EPM.h> // EPM cargo gripper stuff

#endif

#if PARACHUTE == ENABLED

#include <AP\_Parachute.h> // Parachute release library

#endif

#include <AP\_LandingGear.h> // Landing Gear library

#include <AP\_Terrain.h>

#include <LowPassFilter2p.h>

// AP\_HAL to Arduino compatibility layer

#include "compat.h"

// Configuration

#include "defines.h"

#include "config.h"

#include "config\_channels.h"

// lines referring to the times of the pixhawk autopilot, which

// works at 400mhz or 0.0025seconds or 2500 microseconds

# define MAIN\_LOOP\_RATE 400

# define MAIN\_LOOP\_SECONDS 0.0025f

# define MAIN\_LOOP\_MICROS 2500

// statements referring to the autopilot objects, for example, // gps-type objects barometer, compass, DataFlash, etc., all of

// them will subsequently be invoked in the corresponding code

const AP\_HAL::HAL& hal = AP\_HAL\_BOARD\_DRIVER;

static AP\_Scheduler scheduler;

static AP\_GPS gps;

static AP\_Baro barometer;

static AP\_InertialSensor ins;

static RangeFinder sonar;

static Compass compass;

static AP\_SerialManager serial\_manager;

static ToshibaLED\_PX4 toshiba\_led;

static AP\_BattMonitor battery;

//Data, BE CAREFUL, WHEN YOU READ THE SD SECTION DELETE THIS //BLOCK THERE

// you will learn how to use and external module and deep // details about these declarations

#define LOG\_MSG 0x01

#if CONFIG\_HAL\_BOARD == HAL\_BOARD\_PX4

static DataFlash\_File DataFlash("/fs/microsd/APM/LOGS");

#endif

struct PACKED log\_Datos{

LOG\_PACKET\_HEADER;

uint32\_t time\_ms;

float a\_roll;

float a\_pitch;

float a\_yaw;

float pos\_x;

float pos\_y;

float pos\_z;

};

static const struct LogStructure log\_structure[] PROGMEM = {

LOG\_COMMON\_STRUCTURES,

{LOG\_MSG, sizeof(log\_Datos),

"1", "Iffffff", "T\_MS,ROLL,PITCH,YAW,X\_POS,Y\_POS,Z\_POS"},

};

static uint16\_t log\_num; //Dataflash

// Inertial Navigation EKF

#if AP\_AHRS\_NAVEKF\_AVAILABLE

AP\_AHRS\_NavEKF ahrs(ins, barometer, gps, sonar);

#else

AP\_AHRS\_DCM ahrs(ins, barometer, gps);

#endif

static AP\_InertialNav\_NavEKF inertial\_nav(ahrs);

// place your code here //

// paste the setup here //

void setup()

{

ins.init(AP\_InertialSensor::COLD\_START,AP\_InertialSensor::RATE\_400HZ);

serial\_manager.init\_console();

serial\_manager.init();

compass.init();

compass.read();

ahrs.set\_compass(&compass);

gps.init(NULL,serial\_manager);

barometer.init();

barometer.calibrate();

DataFlash.Init(log\_structure, sizeof(log\_structure)/sizeof(log\_structure[0]));

if (DataFlash.NeedErase()) {

DataFlash.EraseAll();

}

log\_num = DataFlash.StartNewLog();

hal.scheduler->delay(100);

}

void loop(void)

{

hal.console->printf("Hello %d\n",hal.scheduler->micros());

hal.scheduler->delay(50);

}

AP\_HAL\_MAIN();

## Appendix 5: Helpful Keywords

The following is a list of thematic keywords . Feel free to use them in order to search for more information in your web browser.

* Quadrotor, quadcopter, aircraft
* Hover-altitude, attitude-orientation, steering
* Real time, OOP, modular programming, scheduler
* Euler angles, roll, pitch, yaw, quasi-velocities
* Linear systems, linearization, non-linear systems
* Autopilot, companion computer/development board
* SDK/software development kit, GUI/graphic user interface
* Data types
* PWM/pulse width modulation, PPM/pulse position modulation, RC
* Duty cycle
* BLDC/brushless DC, BDC/brushed DC, DC
* ESC/electronic speed control, BEC/battery elimination circuit, power module
* Propeller, frame
* Checksum, buffer, GPIO/general purpose input output
* Allocation matrix, thrust vectoring, tethered drone

## Appendix 6: Installing ArduPilot Libraries

There are various ways to install the libraries depending on the operating system, the computer platform, and the code editor used. However, this appendix presents a brief installation description based on Windows 7, Vista, and 10 32/64-bit operating systems using the preloaded Eclipse interface (if this interface is already available on your computer, additional changes may be necessary).

Since we are working with open source technologies, it is very common to deal with changes without prior notice, which may even affect the installation mode. Although a set of all the necessary programs is provided with this book (all of them are open source and free), it is advisable to be aware of the important changes if you want to update versions. You should also visit the corresponding forums.

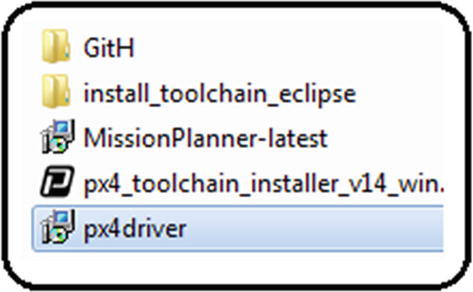
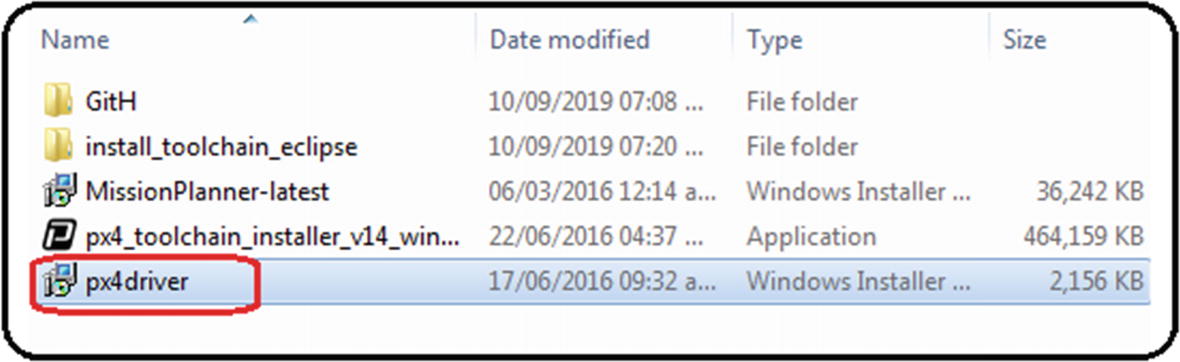
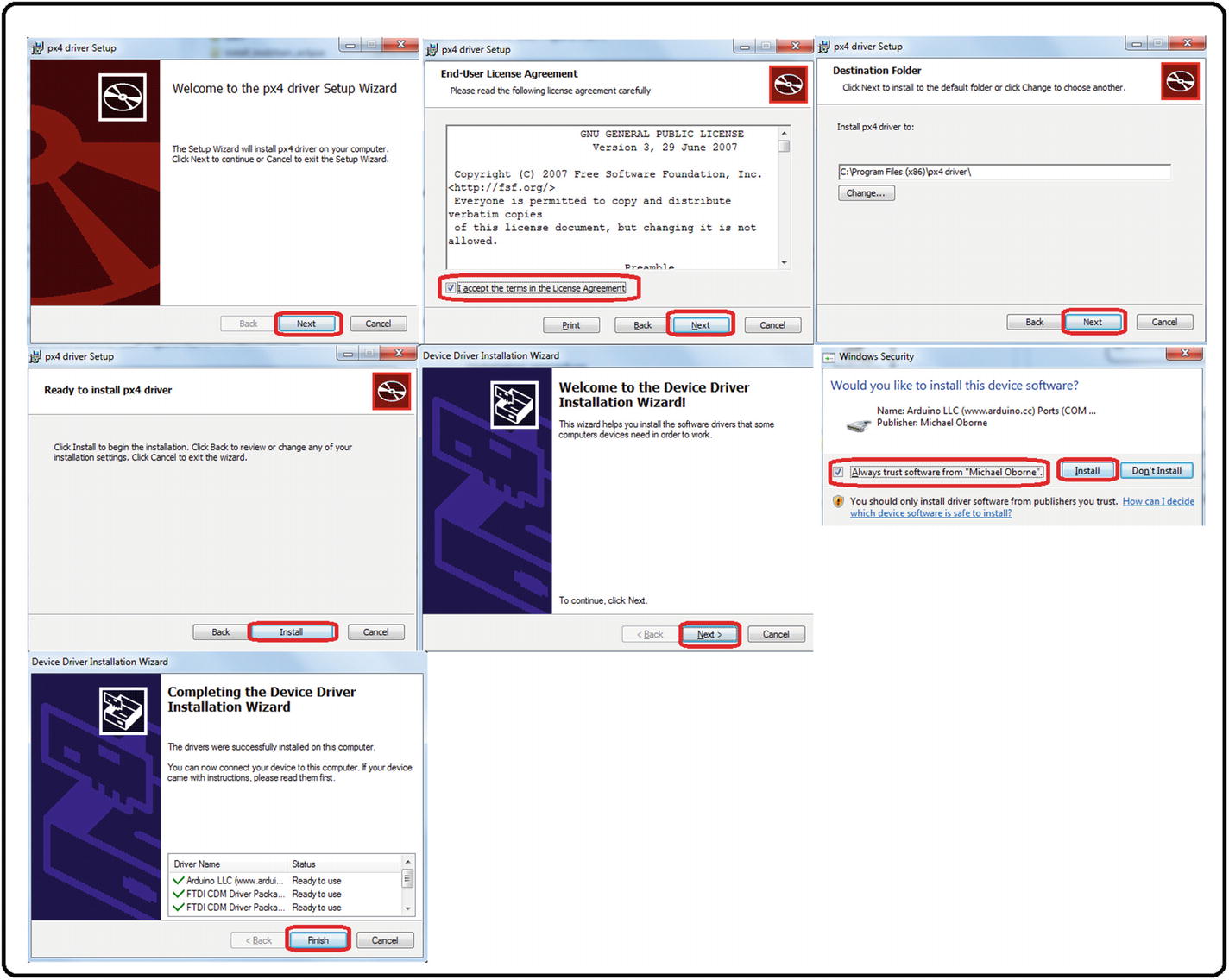
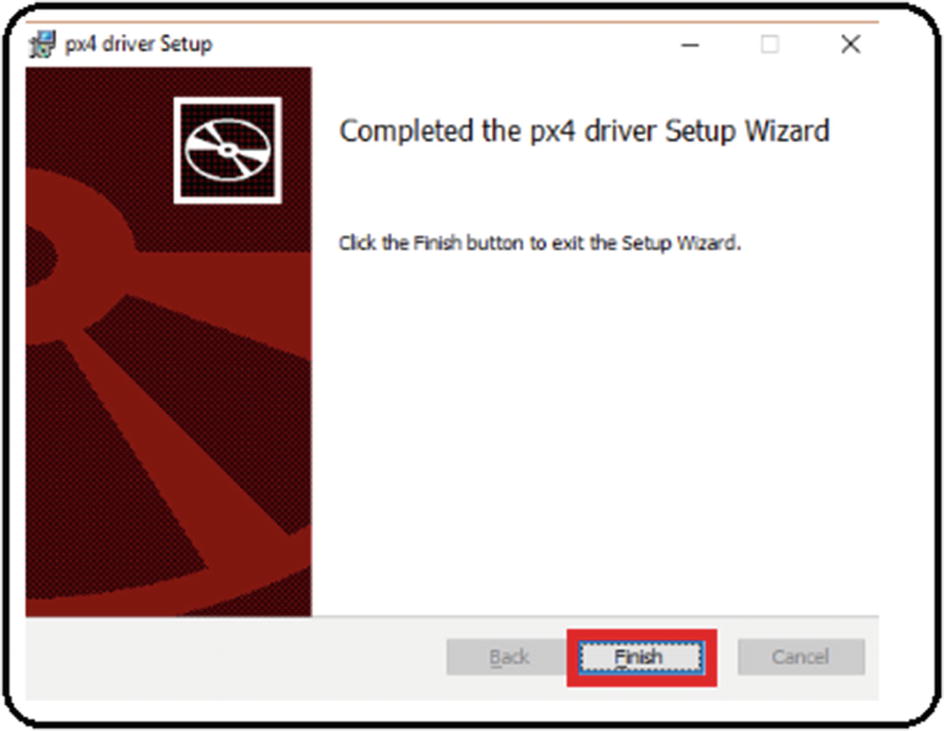
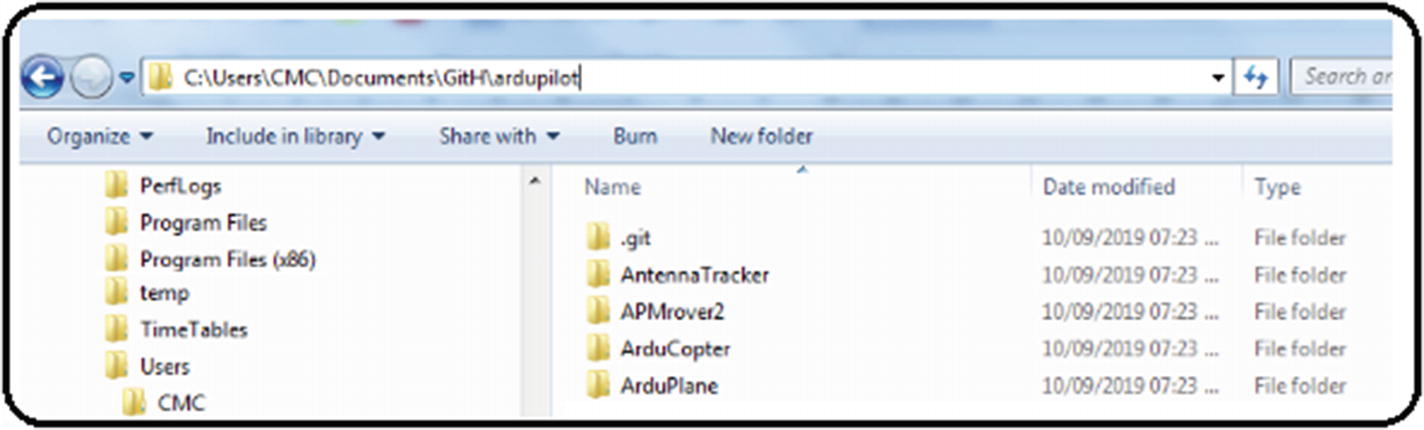
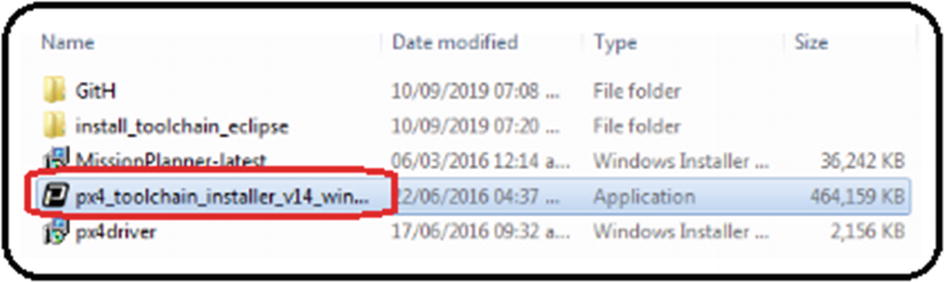
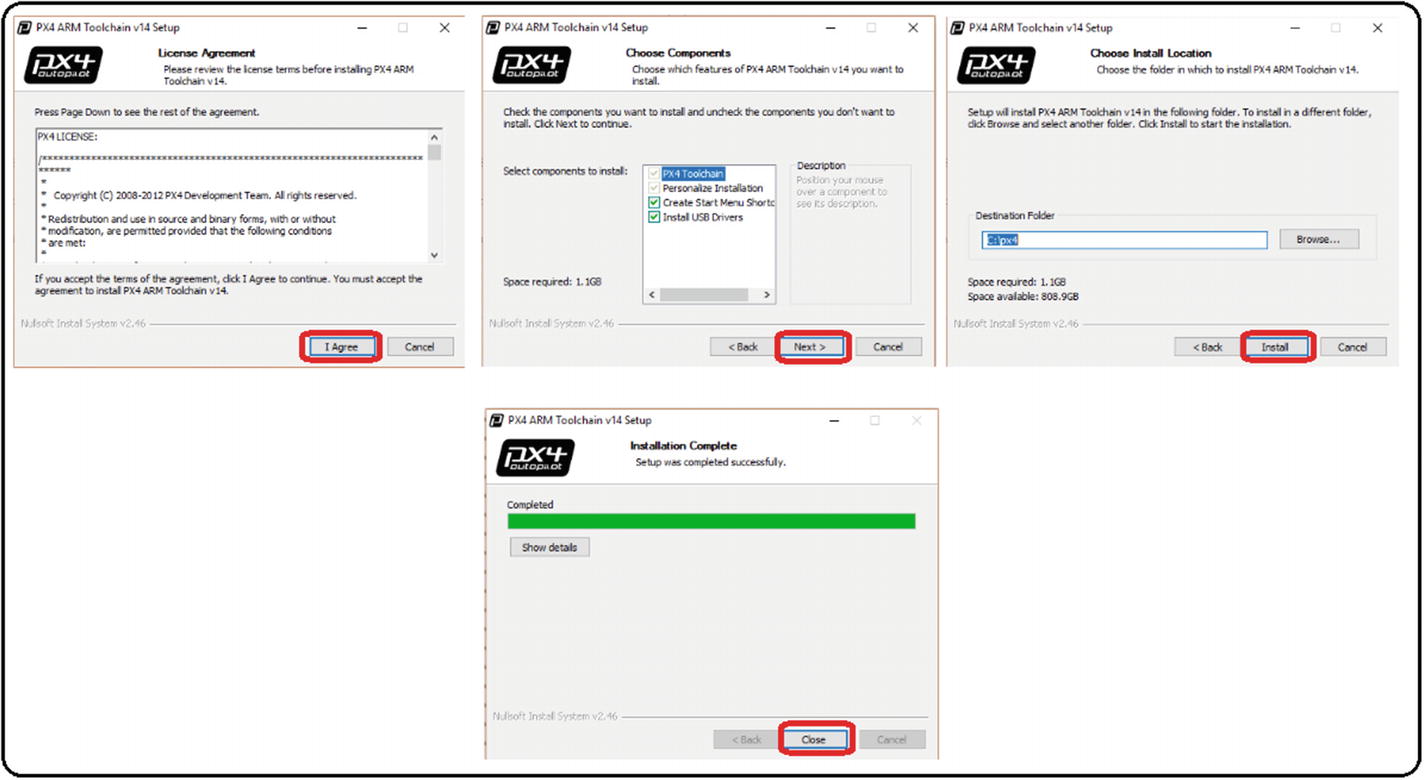
We remind you that the maximum scope of this book is to disseminate knowledge related to these technologies and it is not intended to go beyond any personal project. Any doubt in this regard, refer to web forums.

### “GENERIC” PROCEDURE

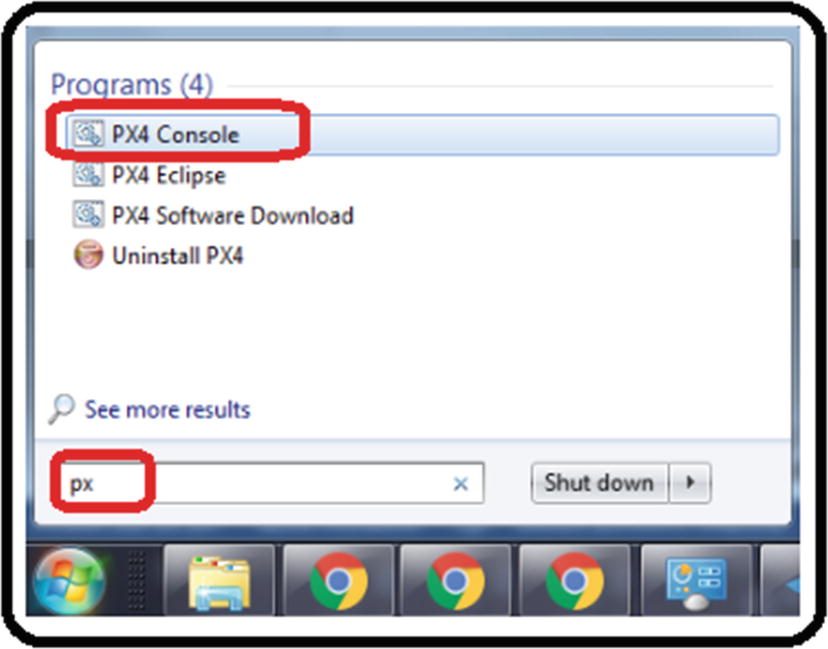
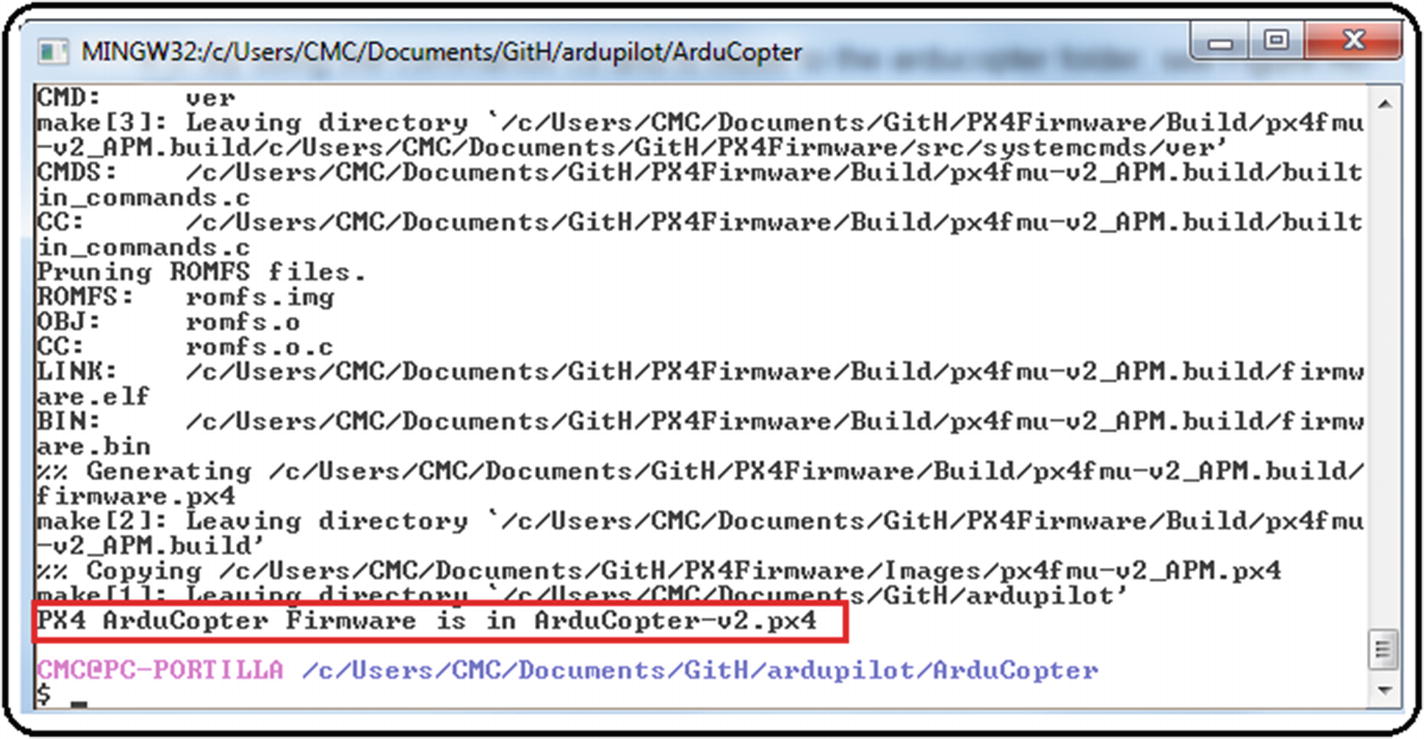
1. 1.
2. Install the driver.
4. 2.
5. Download the libraries.
7. 3.
8. Download the compiler.
10. 4.
11. Compile the libraries.
13. 5.
14. Customize the code editing interface.
16. 6.
17. Program custom code.
19. 7.
20. Compile and test it.

Installation requirements: Windows Vista or superior operating system at 32- or 64-bit, 4GB of RAM for the execution of the development interface, and USB 2.0 ports for using the Pixhawk.

### INSTALLATION PROCEDURE

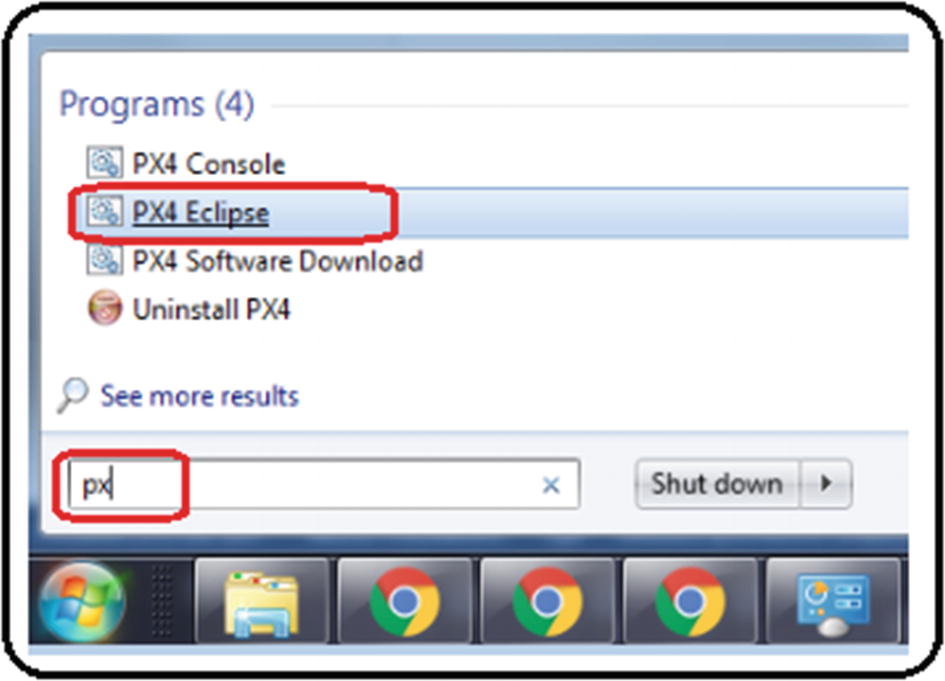
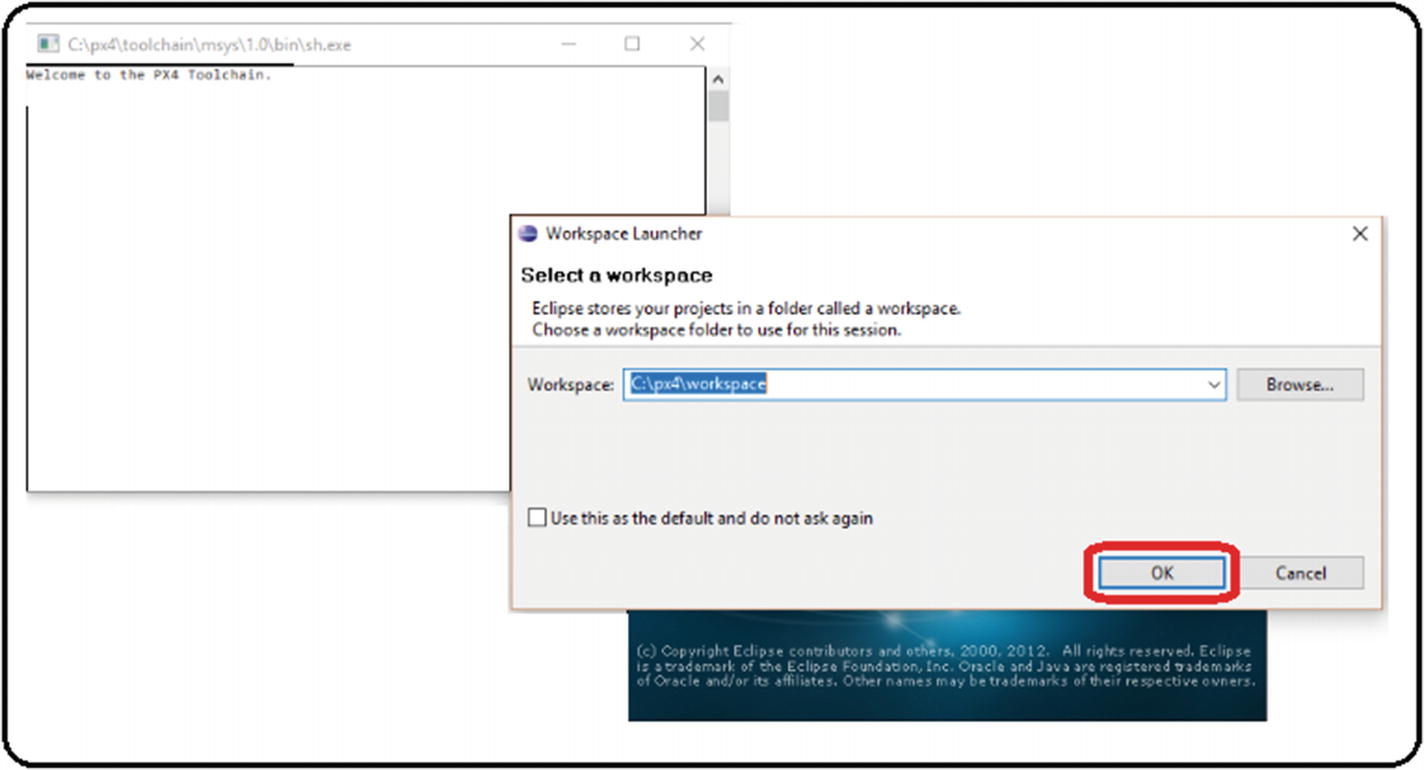
1. 1.
2. Download all the software : libraries, drivers, compiler, Mission Planner, and instructions. See Figure [A6-1](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig1) .
3. 
4. ***Figure A6-1***Provided software
6. 2.
7. Without modifying anything, place theGitH folder inside the following address, verifying that there are no accents, spaces, or special characters in the name of the folder:C:\Users\UserName\Documents
8. For example, this will work:
9. C:\Users\Fonseca\Documents
10. And this will not work because of the accent:
11. C:\Users\León\Documents
13. 3.
14. Once theGitH folder has been copied, make sure that the directory’s name is made up of 50 characters counting from “C:” to the “ardupilot” folder.
15. For example, this will work because it has 41 characters:
16. C:\Users\Fonseca\Documents\GitH\ardupilot
17. And this will not work because it has 51:
18. C:\Users\FonsecaMendezMend\Documents\GitH\ardupilot
19. To make it functional, it is possible to reduce the GitHub folder’s name (it is only possible to modify this folder; altering the others will affect the user’s computer):
20. C:\Users\FonsecaMendezMend\Documents\Gi\ardupilot
21. Now it has 49 characters and therefore it is useful.
22. Note that if you have more than 50 characters or spaces, or if you have accents or symbols, you can still install these libraries by creating a new administrator user account which will comply with the above requirements. Before this, you must ensure that you have removed all of the installation folders from your actual user account.
24. 4.
25. Install the Pixhawk drivers ; see Figure [A6-2](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig2) .
26. 
27. ***Figure A6-2***Autopilot drivers
29. 5.
30. Click the OK, Next, Accept, or Install buttons in the auxiliary windows that appear ; see Figure [A6-3](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig3) .
31. 
32. ***Figure A6-3***Driver installation, step 1
34. 6.
35. When this window appears , it’s an indicator that the drivers were correctly installed. Click Finish and close it; see Figure [A6-4](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig4) .
36. 
37. ***Figure A6-4***Driver installation, step 2
39. 7.
40. Install the toolchain , which is the software that contains the libraries’ compiler and the development interface (an Eclipse version).
42. 8.
43. Before installing it, verify that theardupilot folder created in step 1 exists; see Figure [A6-5](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig5) .
44. 
45. ***Figure A6-5*** Toolchain installation, step 1
47. 9.
48. Run the toolchain; see Figure [A6-6](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig6) .
49. 
50. ***Figure A6-6***Toolchain installation, step 2
52. 10.
53. Click the OK, Next, Accept , and Install buttons in the windows that appear; see Figure [A6-7](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig7) .
54. 
55. ***Figure A6-7***Toolchain installation, step 3

### COMPILING THE LIBRARIES

1. 1.
2. From the Windows Start menu , type “px”. You should automatically see the program named PX4 Console. Run it. See Figure [A6-8](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig8) .
3. 
4. ***Figure A6-8*** Compilation process, step 1
6. 2.
7. When executing it, the auxiliary screen shown in Figure [A6-9](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig9) should appear (note that the colors were inverted for presentation purposes of this book).
8. 
9. ***Figure A6-9*** Compilation process, step 2
11. 3.
12. Move to yourArduCopter folder using the linux-kind commandscd andls ; see Figure [A6-10](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig10) .
13. 
14. ***Figure A6-10*** Compilation process, step 3
16. 4.
17. Once there, execute themake px4-v2 command , which will compile the libraries to be used in the Pixhawk’s fmuv2 family including cloned versions; see Figure [A6-11](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig11) .
18. 
19. ***Figure A6-11*** Compilation process, step 4
21. 5.
22. With some time and patience, if the compilation was successful, the following message should automatically appear: “PX4 ArduCopter firmware is in ArduCopter-v2.px4.” Close the window with theexit command or by pressing the X button; see Figure [A6-12](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig12) .
23. 
24. ***Figure A6-12*** Compilation process, step 5

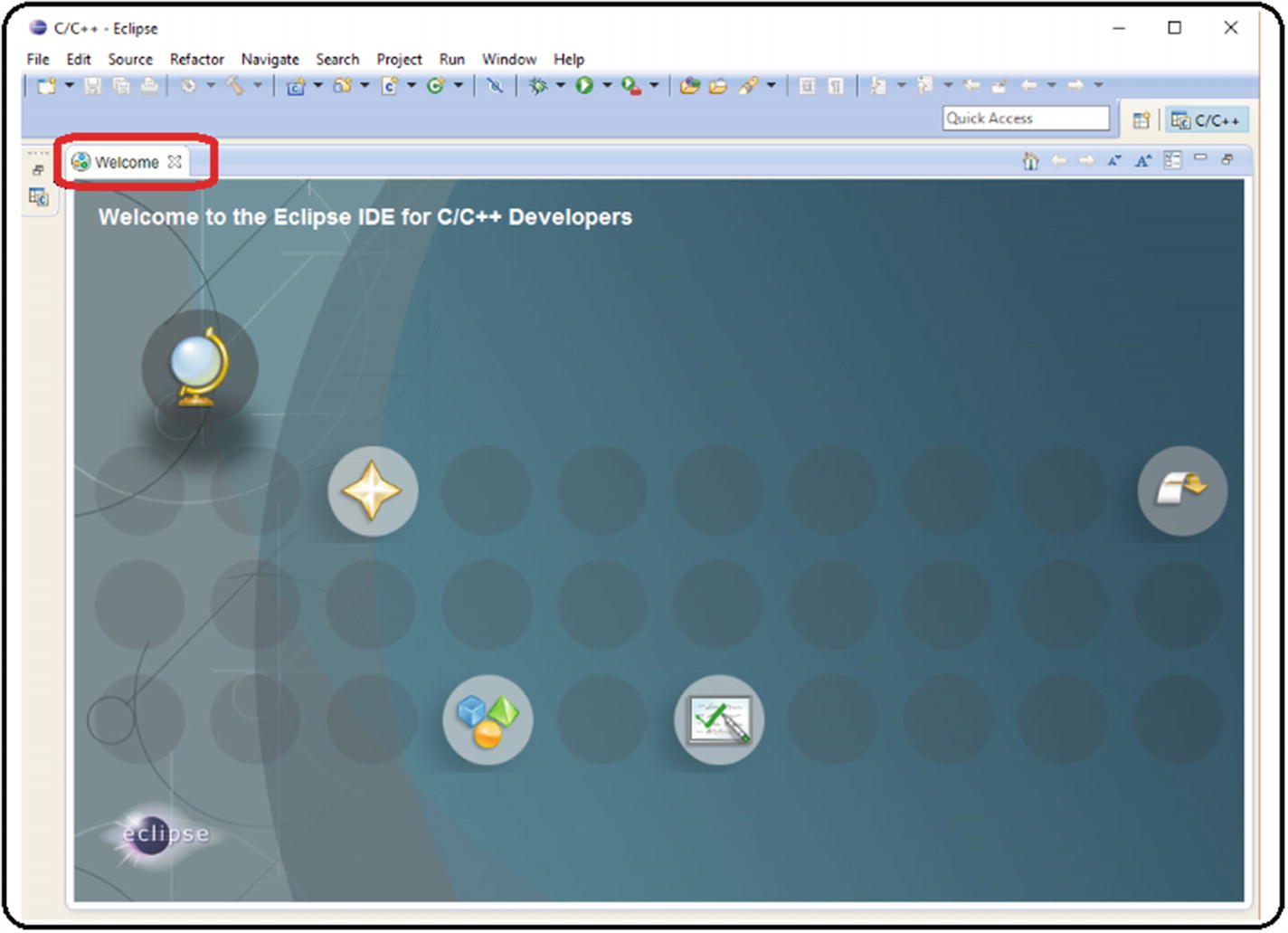
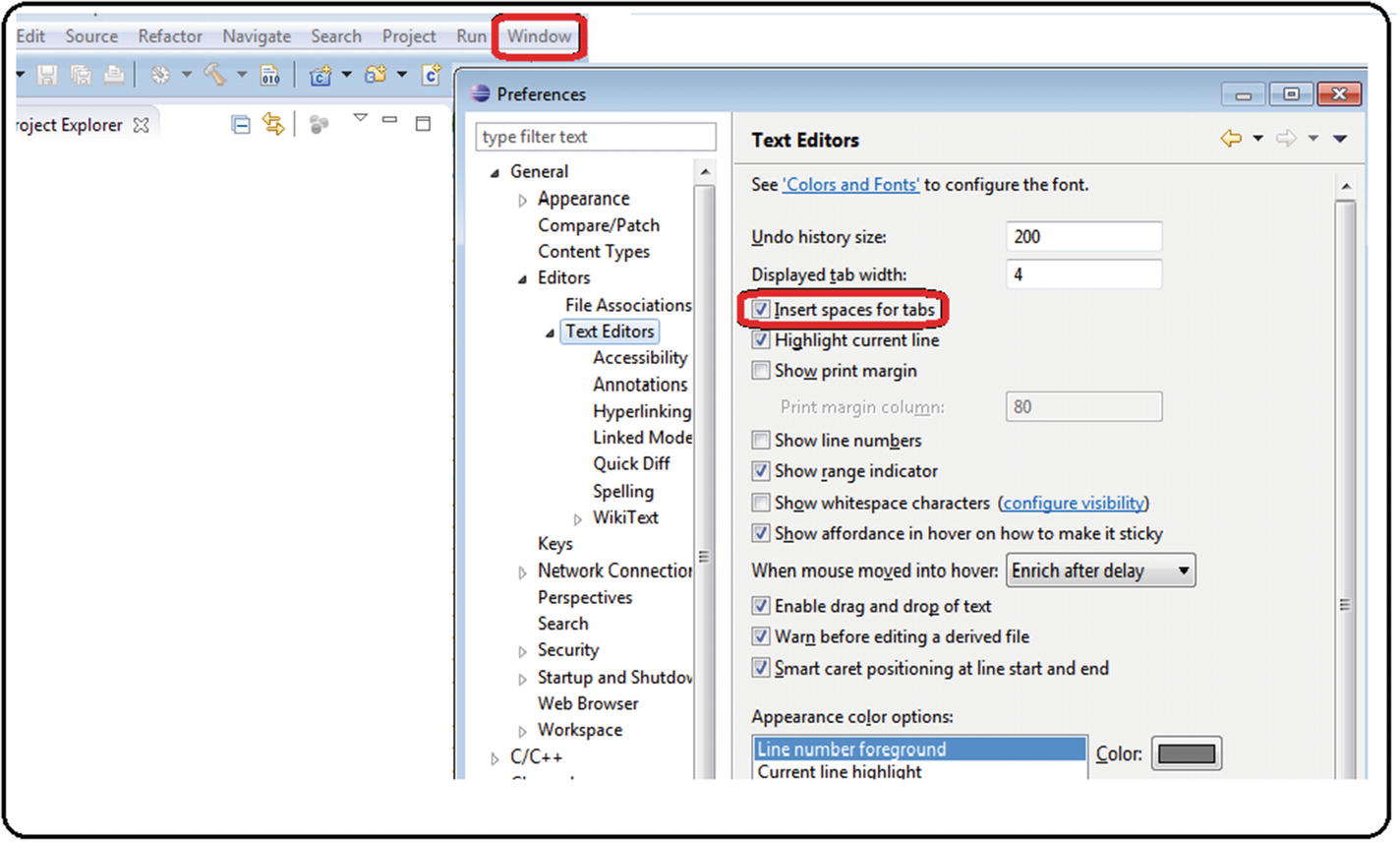
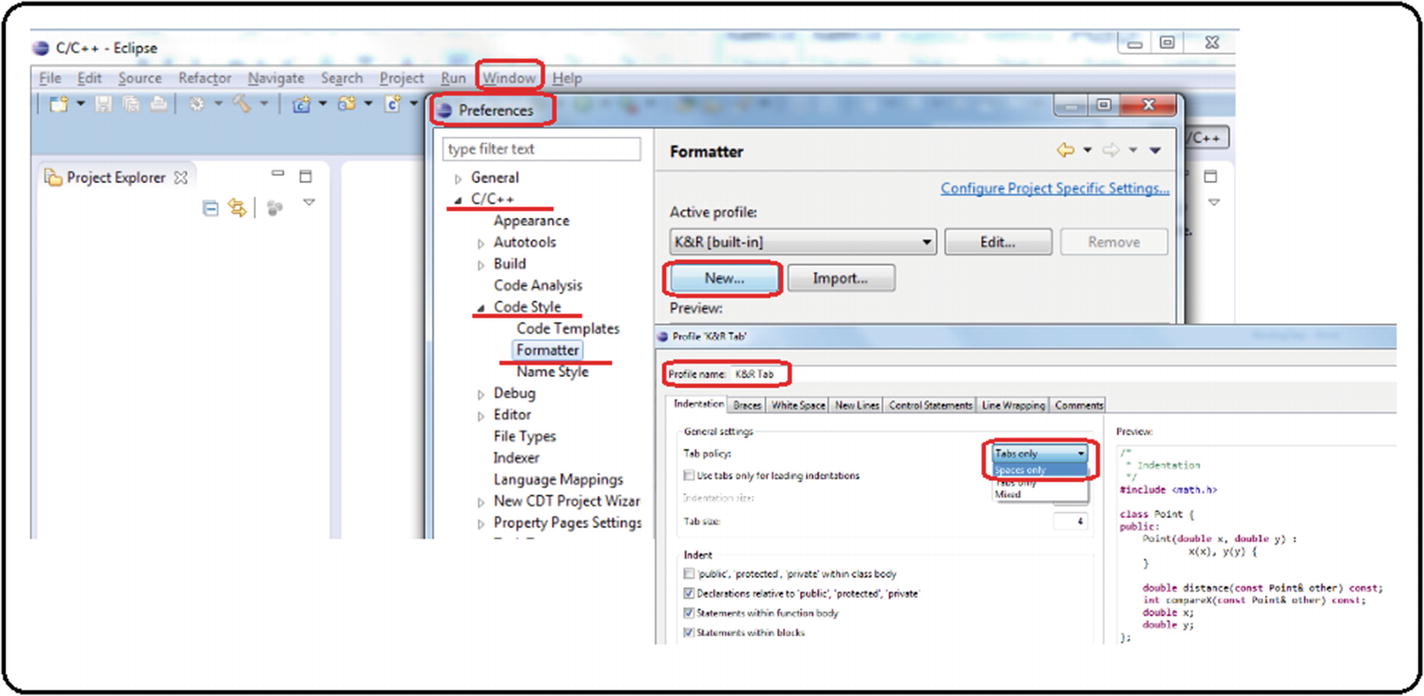
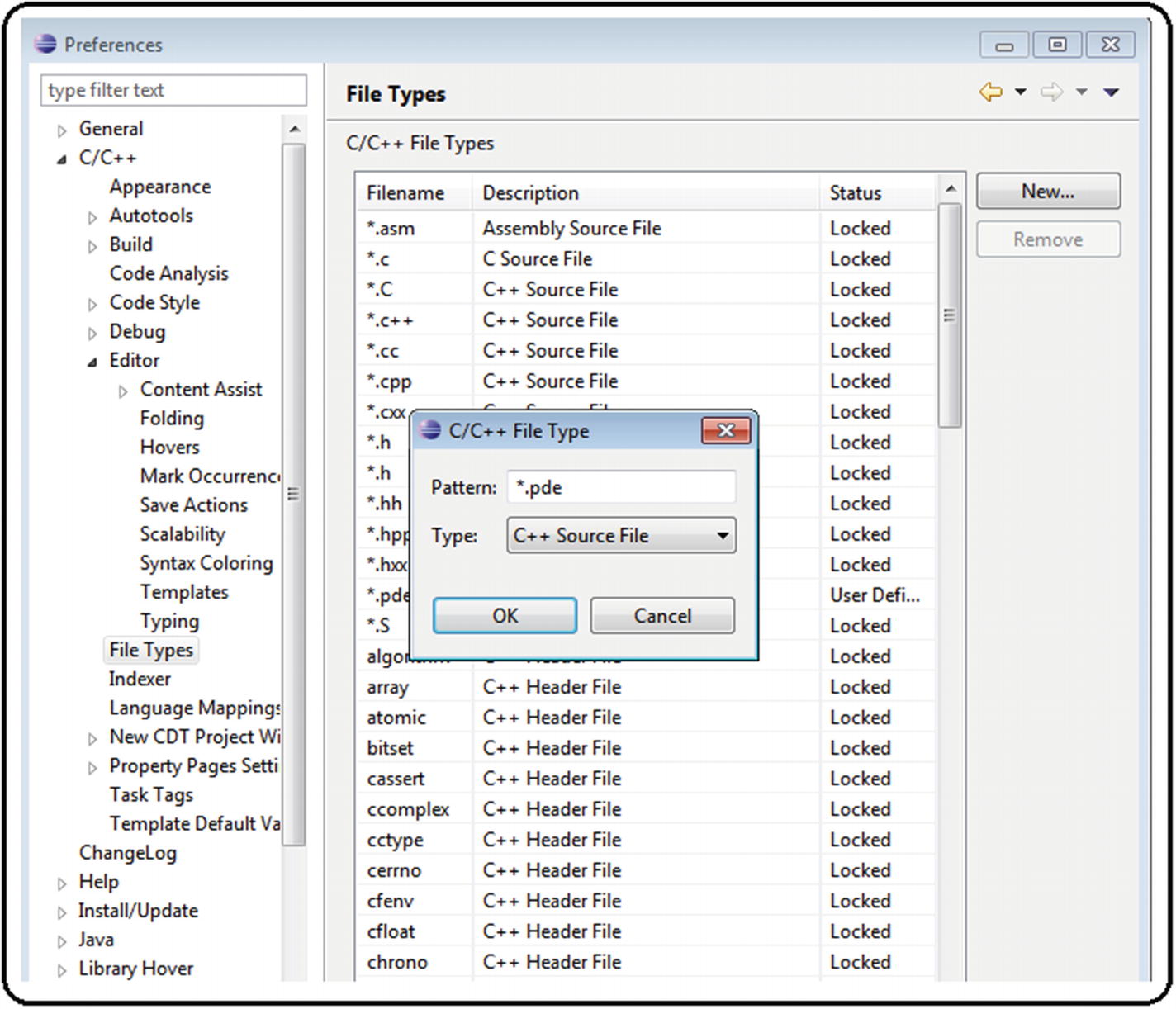
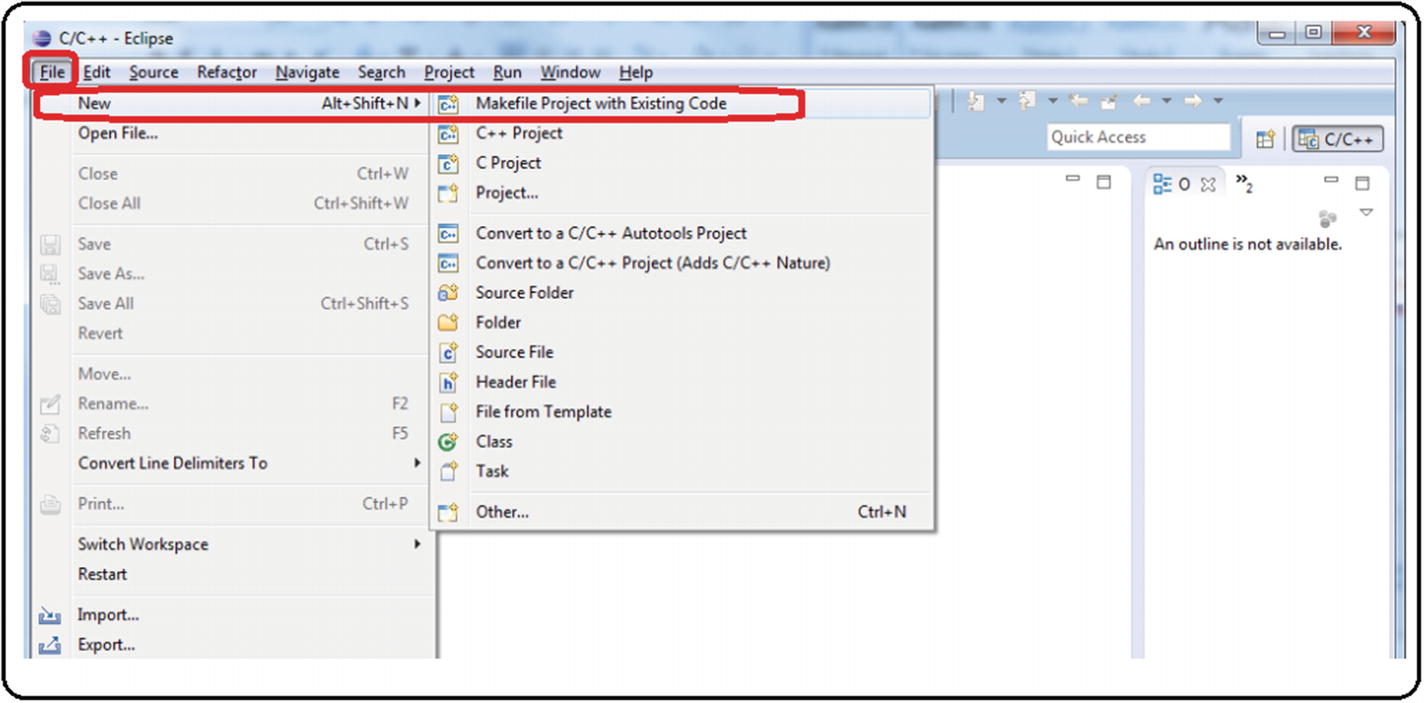
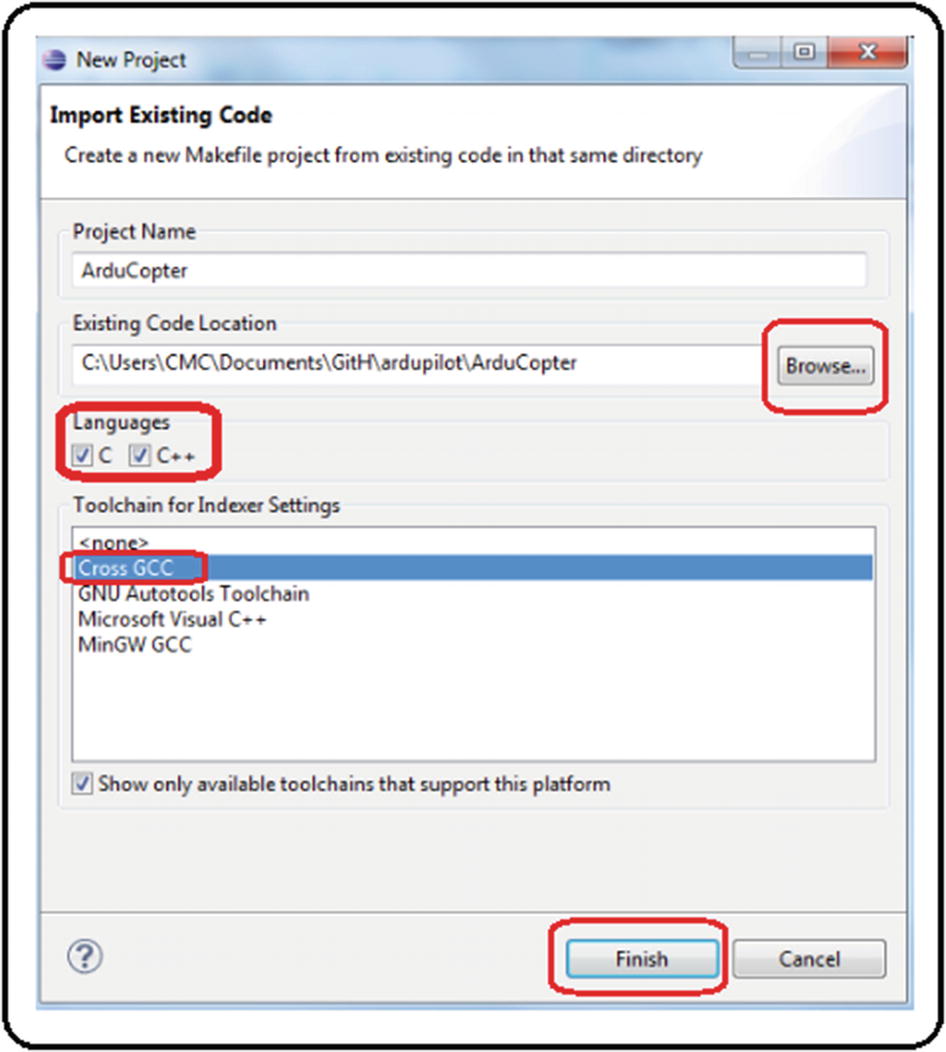
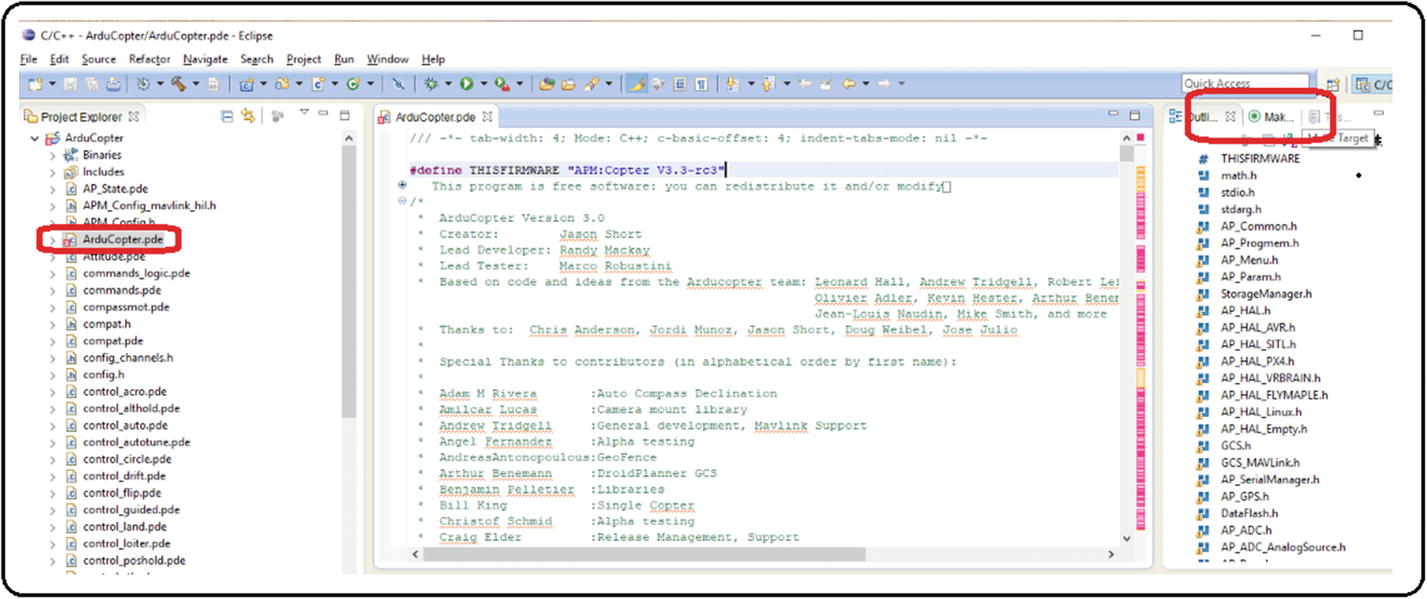
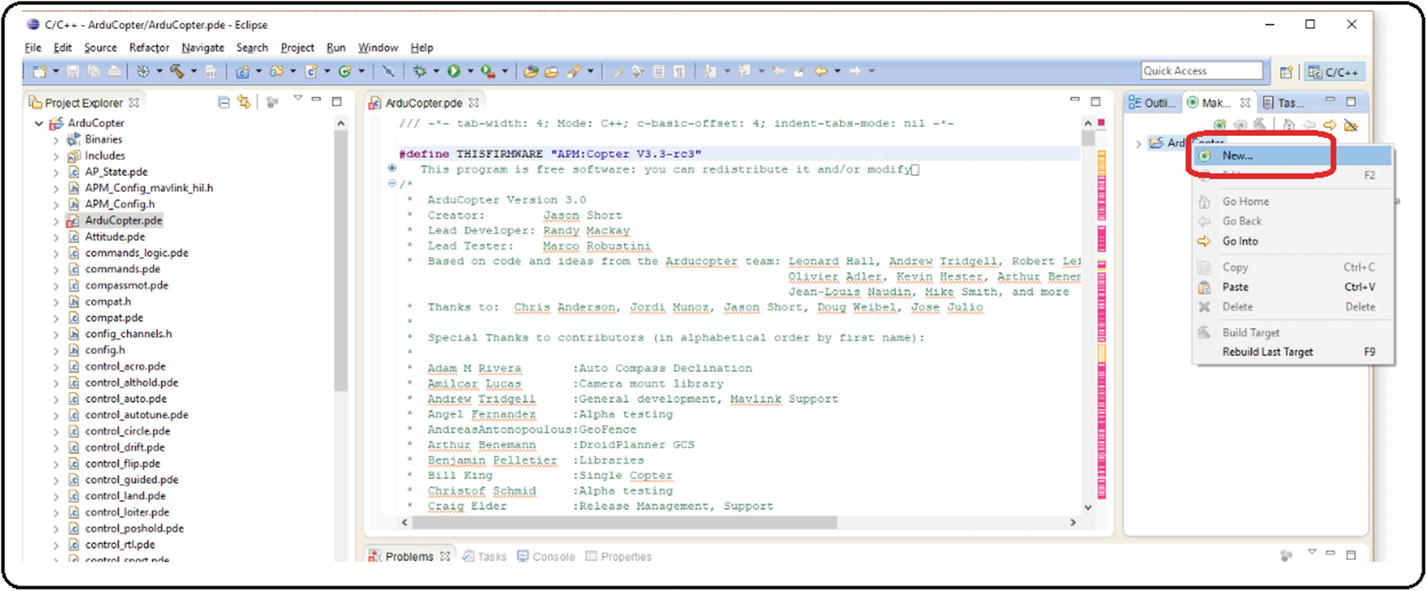
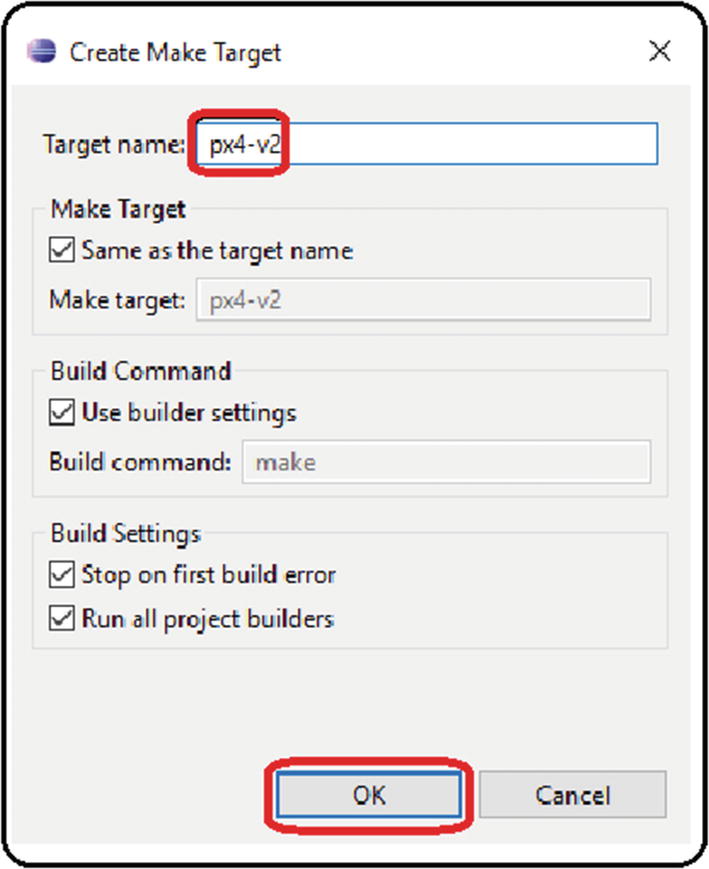
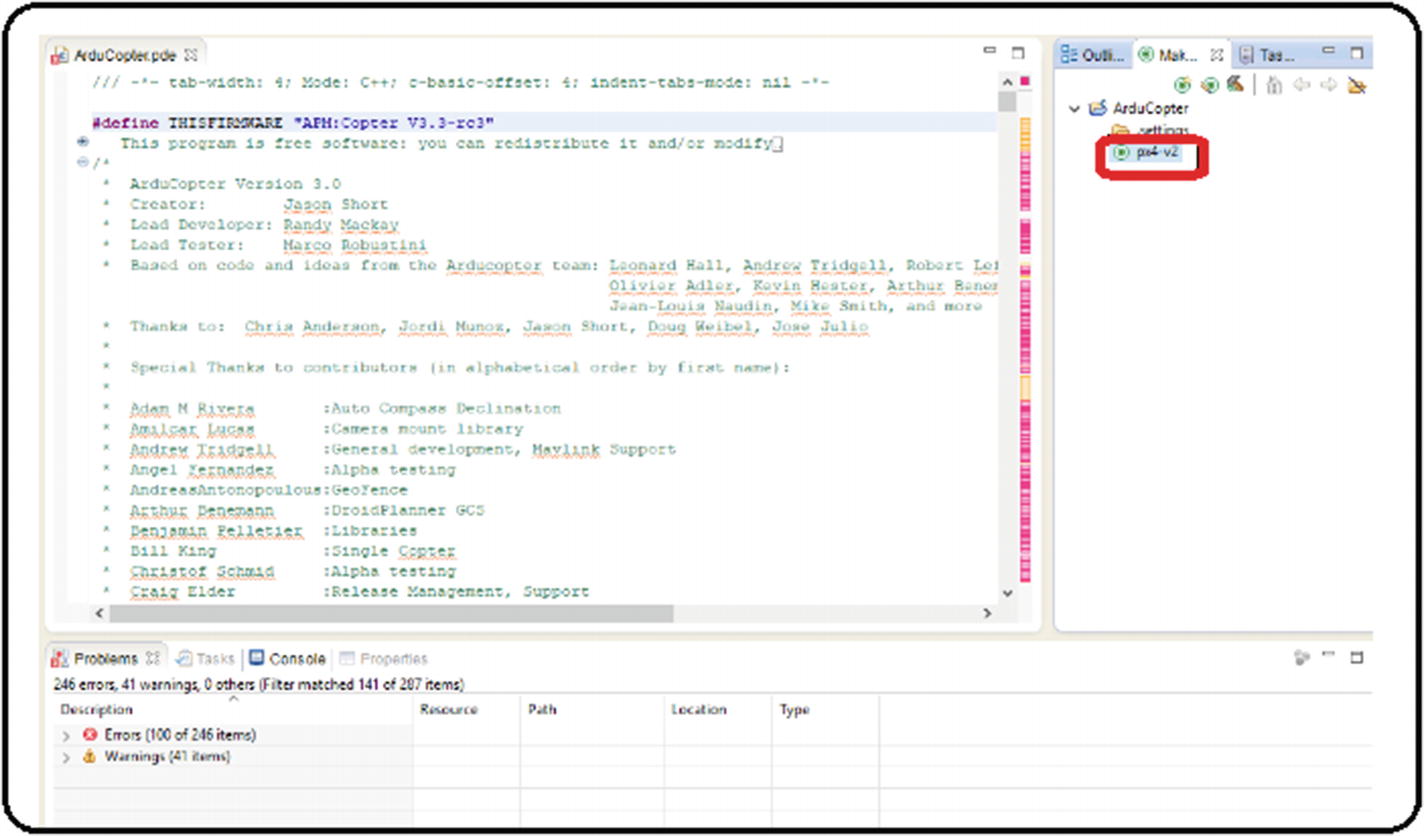
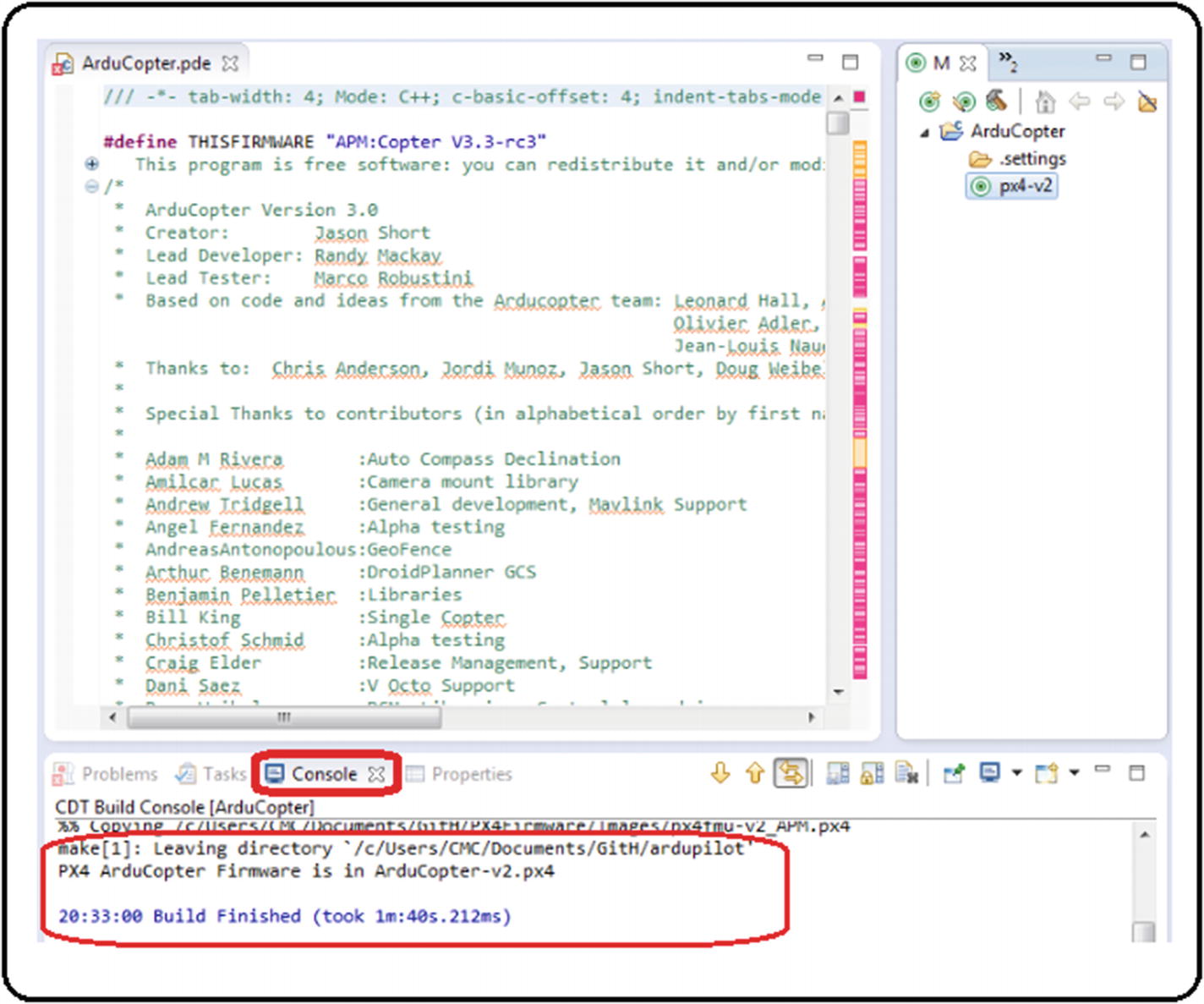
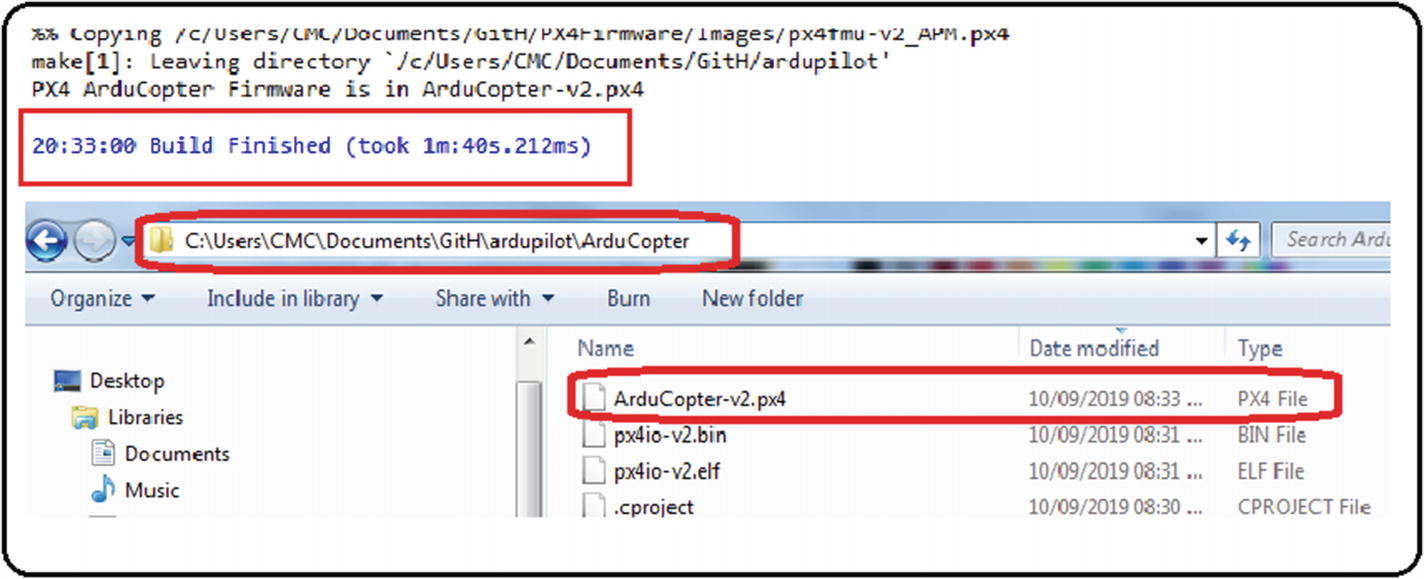
A frequent error during this procedure is the message indicating that theArducopter.pde file is not found. The way to solve it is to search via Windows Explorer in the directory where it is compiling for a file with a similar name, such asarducopter.pde . Rename it so that it matches in upper and lower cases with the name the error indicates and repeat the process from themake px4-v2 command.

### INTERFACE CUSTOMIZATION AND RECOMPILATION FROM THE PRELOADED VERSION OF THE ECLIPSE EDITOR

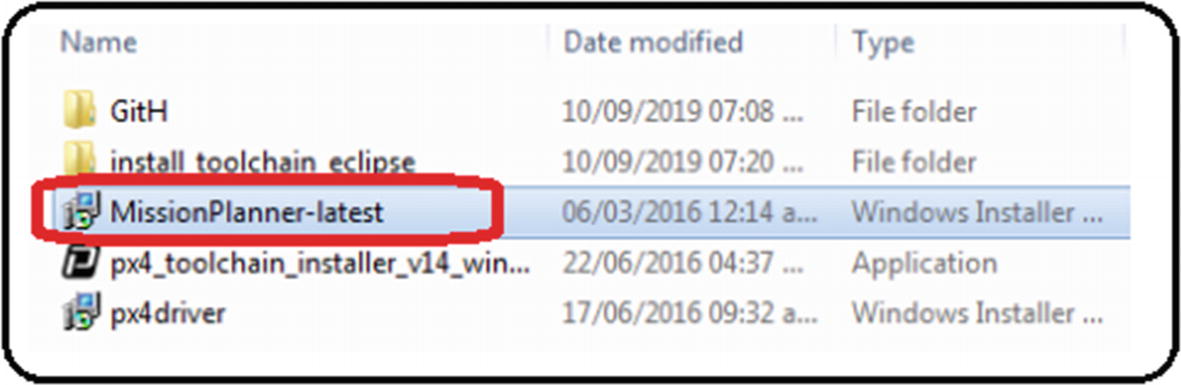
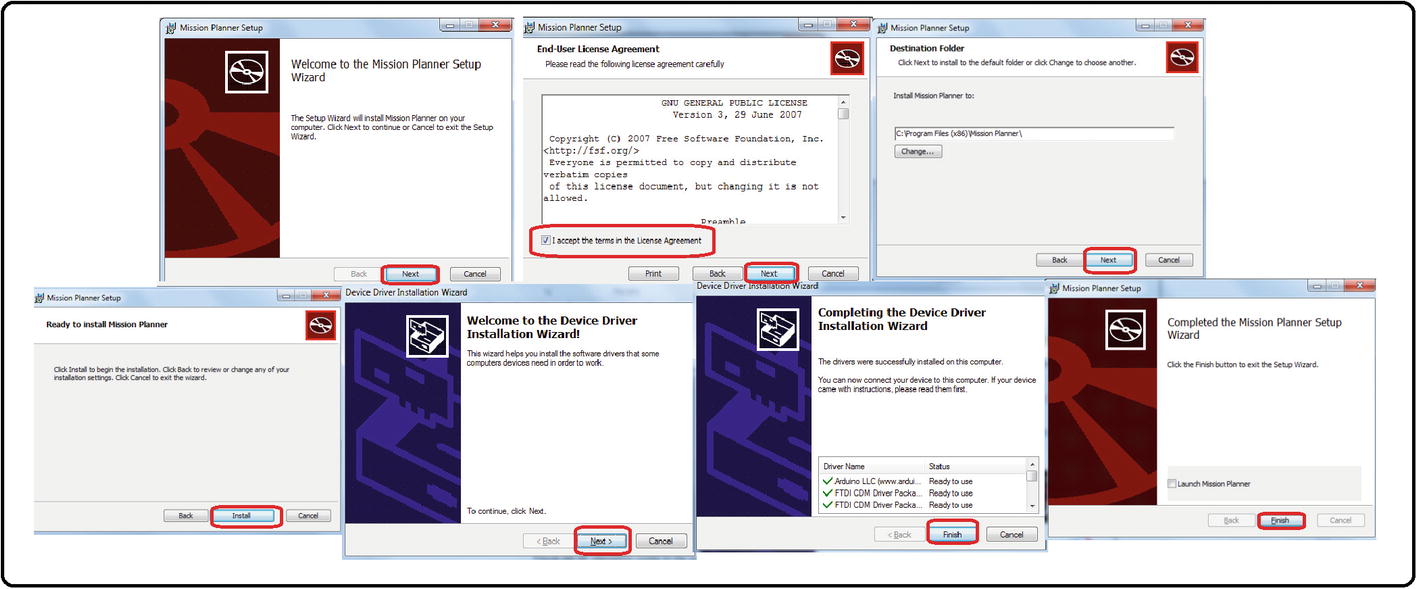
1. 1.
2. From the Windows Start menu, type “px”. The program named PX4 Eclipse should automatically appear. Run it. See Figure [A6-13](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig13) .
3. 
4. ***Figure A6-13*** Code editor customization, step 1
6. 2.
7. Click the OK button in the next window. It may take a while to appear . See Figure [A6-14](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig14) .
8. 
9. ***Figure A6-14***Code editor customization, step 2

A second frequent problem that occurs during this procedure appears when the OK button is pressed. If a warning message is displayed or the window is closed automatically, it means that a JAVA development package update is necessary. It is calledjdk-8u111-windows-i586.exe . In this case, we must be careful given that it is a third-party dependency, so we only provide the full name of the file as a reference.

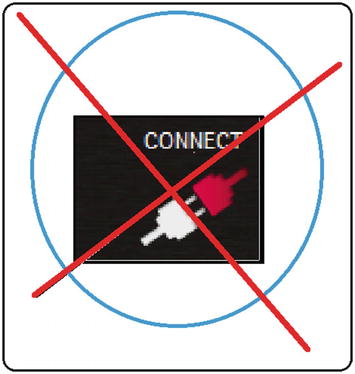
Once the update is installed, repeat the process from step 1 (this update’s download address and its license type is in the section “Licenses” at the beginning of this book).

1. 3.
2. If there is no error, the screen in Figure [A6-15](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig15) will appear. Close it by clicking the X in the Welcome tab .
3. 
4. ***Figure A6-15*** Code editor customization, step 3
6. 4.
7. Modify the preferences to have spaces instead of tabulations. Some programming languages use tabulations, but the ArduPilot libraries use spaces, so see Figure [A6-16](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig16) . Here is the path: Window ➤ Preferences ➤ General ➤ Editors ➤ Text Editors ➤ Insert spaces for tabs ➤ Apply ➤ OK.
8. 
9. ***Figure A6-16*** Code editor customization, step 4
11. 5.
12. Modify the preferences so that the code style uses spaces instead of indentation. See Figure [A6-17](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig17) . Here is the path: Window ➤ Preferences ➤ C/C++ ➤ Code Style ➤ Formatter ➤ New ➤ Write “K&R Tab” ➤ Change “Indentation” set to “Spaces only” ➤ Apply ➤ OK.
13. 
14. ***Figure A6-17*** Code editor customization, step 5
16. 6.
17. Associate files with a.pde extension to C++ source code (many of the source files of ArduPilot libraries have a.pde extension). See Figure [A6-18](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig18) . Here is the path: Windows ➤ Preferences ➤ C/C++ ➤ File Types ➤ New ➤ Write “∗.pde” ➤ Change the Type to C++ Source File ➤ OK ➤ OK.
18. 
19. ***Figure A6-18*** Code editor customization, step 6
21. 7.
22. Load the ArduCopter project . See Figure [A6-19](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig19) . Here is the path: File ➤ New ➤ Makefile Project with Existing Code.
23. 
24. ***Figure A6-19*** Code editor customization, step 7
26. 8.
27. In the auxiliary window that appears next, select the C and C ++ language checkboxes. Search for theArduCopter folder; if you select correctly, the ArduCopter project will appear automatically (note that the main project shares the name with the folder). Select the Cross GCC option and click the Finish button. See Figure [A6-20](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig20) .
28. 
29. ***Figure A6-20*** Code editor customization, step 8
31. 9.
32. In the project explorer , the ArduCopter project and all the auxiliary files will be loaded. Search for the main file calledArduCopter.pde . Open it and look for a green button on the right side named Make (under a sign that reads Quick Access); see Figure [A6-21](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig21) .
33. 
34. ***Figure A6-21*** Code editor customization, step 9
36. 10.
37. When pressing the green button called Make, it should show theArduCopter project folder. Right-click that folder. You should see another green button that says New. Click it. See Figure [A6-22](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig22) .
38. 
39. ***Figure A6-22*** Code editor customization, step 10
41. 11.
42. An auxiliary window will appear. Type “px4-v2” and then click the OK button. See Figure [A6-23](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig23) .
43. 
44. ***Figure A6-23***Code editor customization, step 11
46. 12.
47. A green button that reads px4-v2 will appear under the folder. Double left-click it to start the compilation. NEVER STOP THE COMPILATION. Even though you know the code is wrong, if you try to stop it, it will produce general errors in the operation of the computer. See Figure [A6-24](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig24) .
48. 
49. ***Figure A6-24***Code editor customization, step 12
51. 13.
52. In the lower menu, change to the Console section and ignore the problems section. If everything was done correctly, after a considerable compilation time, the message “Firmware is in ArduCopter-v2.px4” should appear in the Console menu followed by the exact date and time of completion and the message “Build Finished.” See Figure [A6-25](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig25) .
53. 
54. ***Figure A6-25*** Code editor customization, step 13
56. 14.
57. Check in theArduCopter folder that theArduCopter-v2.px4 file exists and that its modification date is exactly as indicated in the previous point. The files with the.px4 extension are the ones that will be uploaded into the autopilot. See Figure [A6-26](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig26) .
58. 
59. ***Figure A6-26*** Code editor customization, step 14
61. 15.
62. In theardupilot folder is a directory called hellodrone . Load the project and try to compile it. Then repeat the process of creating a new project as stated in the “Making New Projects by Using Eclipse” section. Try this with one of the examples of code previously shown (the projects about terminal writing or reading are a good starting point).

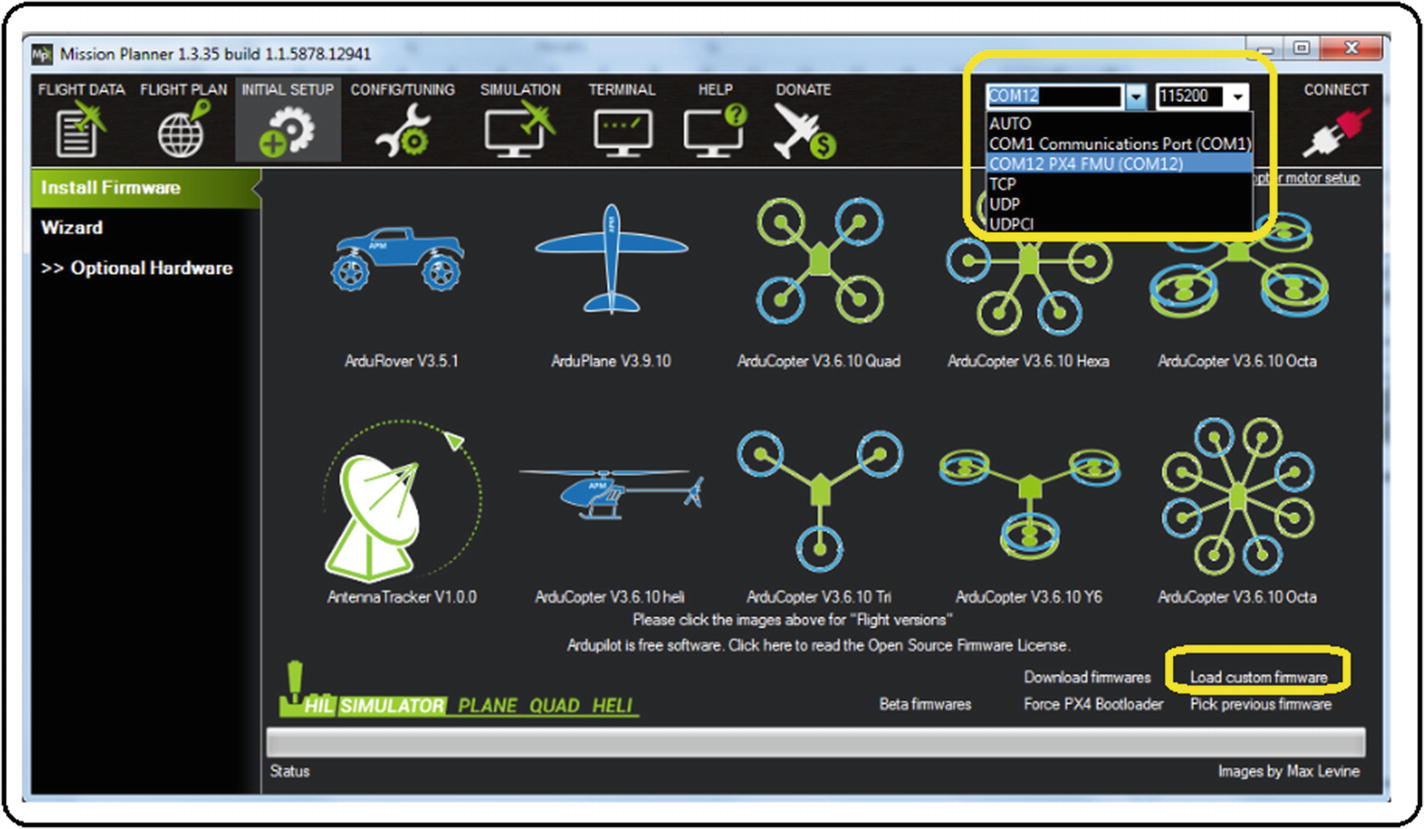
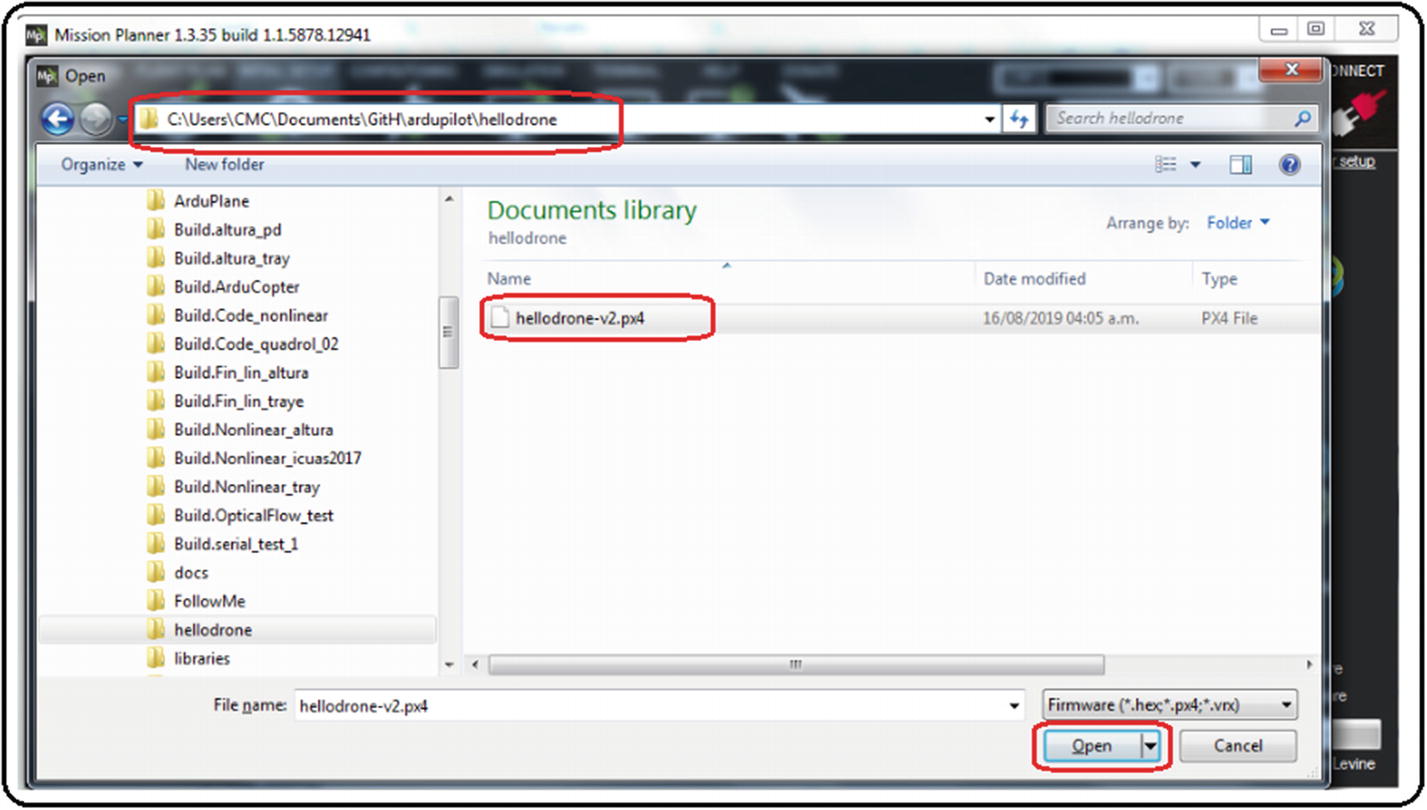
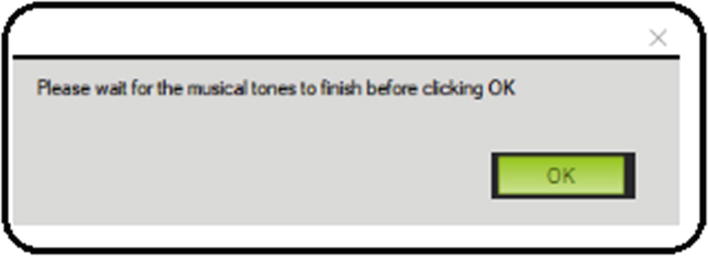
### UPLOADING ∗ .PX4 FILES TO THE AUTOPILOT

1. 1.
2. Install Mission Planner . See Figure [A6-27](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig27) .
3. 
4. ***Figure A6-27*** Uploading code to the autopilot, step 1
6. 2.
7. Click the Install, Accept , Ok, Next, or Finish buttons as many times as necessary, as shown in Figure [A6-28](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig28) .
8. 
9. ***Figure A6-28*** Uploading code to the autopilot, step 2

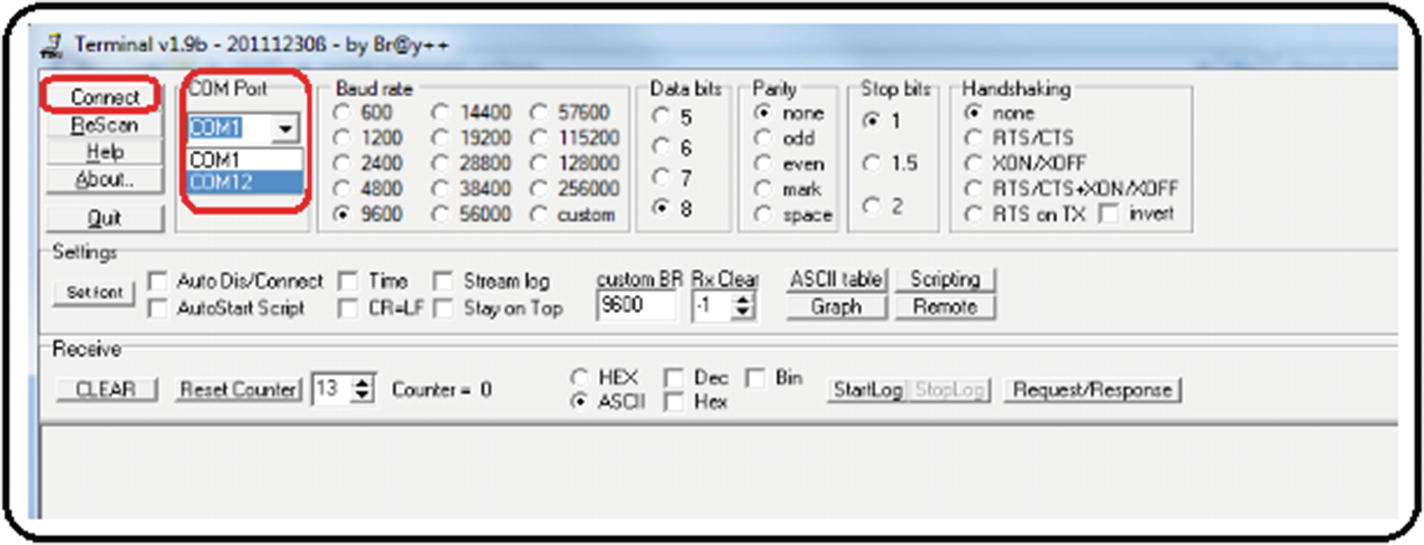
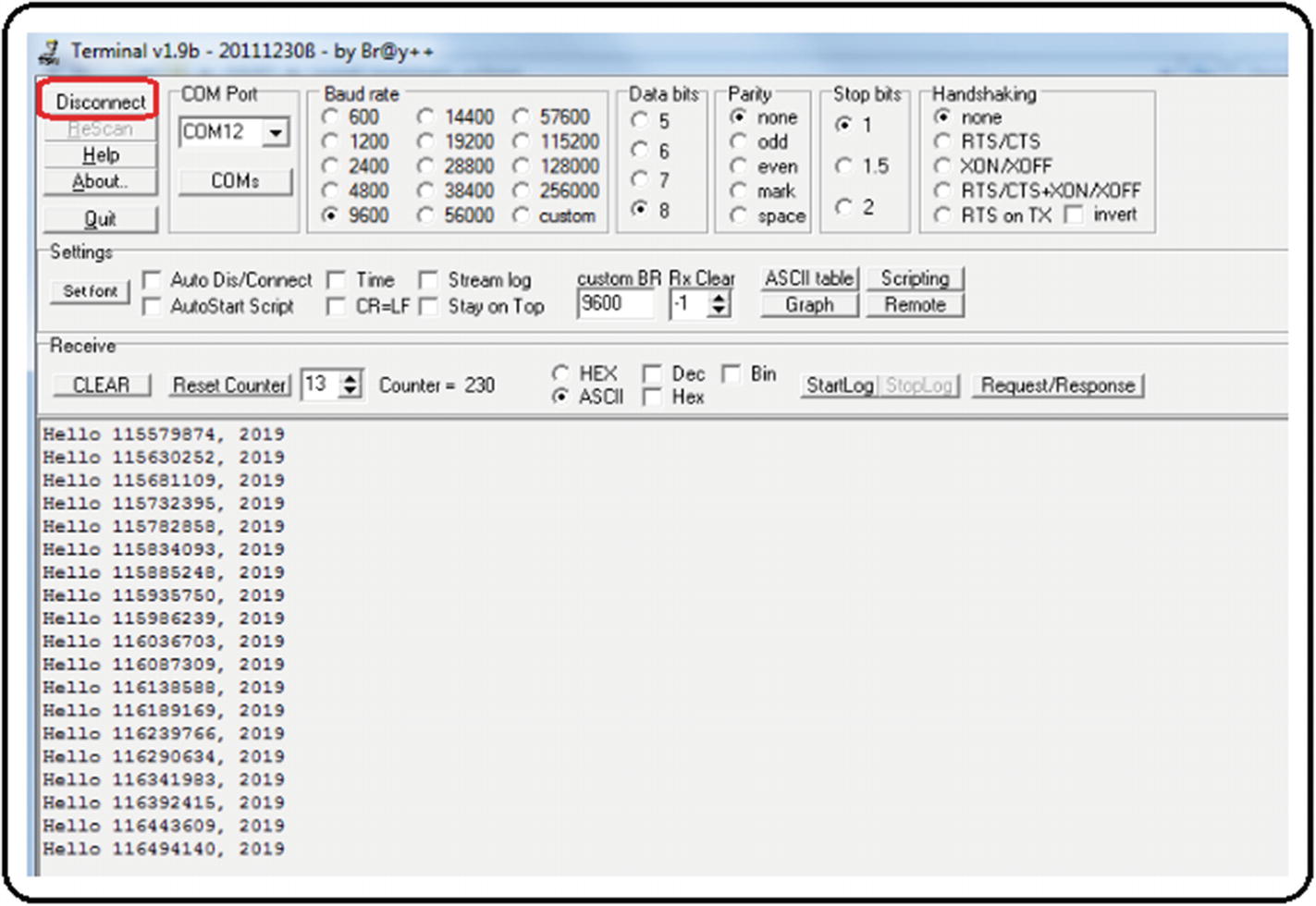
From this moment on, NEVER PRESS THE CONNECT BUTTON during sequence of steps. See Figure [A6-29](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig29) .



***Figure A6-29*** Uploading code to the autopilot warning

1. 3.
2. Open Mission Planner, discard all messages about new updates, and go to the Initial Setup tab. Search for the Install Firmware tab . If an unexpected error message appears, just ignore it. See Figure [A6-30](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig30) .
3. 
4. ***Figure A6-30*** Uploading code to the autopilot, step 3
6. 4.
7. Connect the autopilot to the computer and look for it in the drop-down list where the AUTO tag is labeled; it usually comes up as a COM PX4 FMU. Once you have found it, select your device. Remember: NEVER PRESS THE CONNECT BUTTON. Now, look for the Load custom firmware button. If it does not appear, try to install previous or recent Mission Planner versions until this button is enabled. This button is essential to load custom software . See Figure [A6-31](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig31) .
8. 
9. ***Figure A6-31*** Uploading code to the autopilot, step 4
11. 5.
12. When the Load custom firmware button is there, click it. An auxiliary box will appear; in it you must indicate the location of the\*.px4 file to be uploaded into the autopilot. Select it, click the Open button, and follow the instructions on the screen. See Figure [A6-32](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig32) .
13. 
14. ***Figure A6-32*** Uploading code to the autopilot, step 5
16. 6.
17. If the upload was successful , a message indicating not to disconnect until you hear the buzzer will be displayed. Click the OK button. Now the autopilot can be disconnected and used. Mission Planner can be closed also. See Figure [A6-33](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig33) .
18. 
19. ***Figure A6-33***Uploading code to the autopilot, step 6

### TERMINAL TEST OF THE PREVIOUSLY LOADED PROGRAM

1. 1.
2. Reconnect the Pixhawk , remembering the address assigned in step 4 above. If you don’t know this address, look for it in devices and printers, or in the Windows device manager.
4. 2.
5. Open the terminal program that you prefer (we useTerminal.exe ) and select the COM port assigned to the Pixhawk autopilot. See Figure [A6-34](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig34) .
6. 
7. ***Figure A6-34***Testing code with a serial terminal, generic procedure Step 2
9. 3.
10. Press the Connect button on the terminal. As you can see, all the information displayed in your program by using the commandhal.console-> printf () is shown. NEVER UNPLUG THE PIXHAWK WITHOUT PRESSING THE DISCONNECT BUTTON. See Figure [A6-35](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig35) .
11. 
12. ***Figure A6-35*** Testing code with a serial terminal, generic procedure step 3

### REFERENCES AND SUGGESTED WEBSITES

Website about the different types of installation of the ArduPilot libraries: <http://ardupilot.org/dev/docs/building-the-code.html>

Websites about the type of installation seen specifically in this appendix:

<http://ardupilot.org/dev/docs/building-setup-windows.html#building-setup-windows>

<http://ardupilot.org/dev/docs/building-px4-with-make.html#building-px4-with-make>

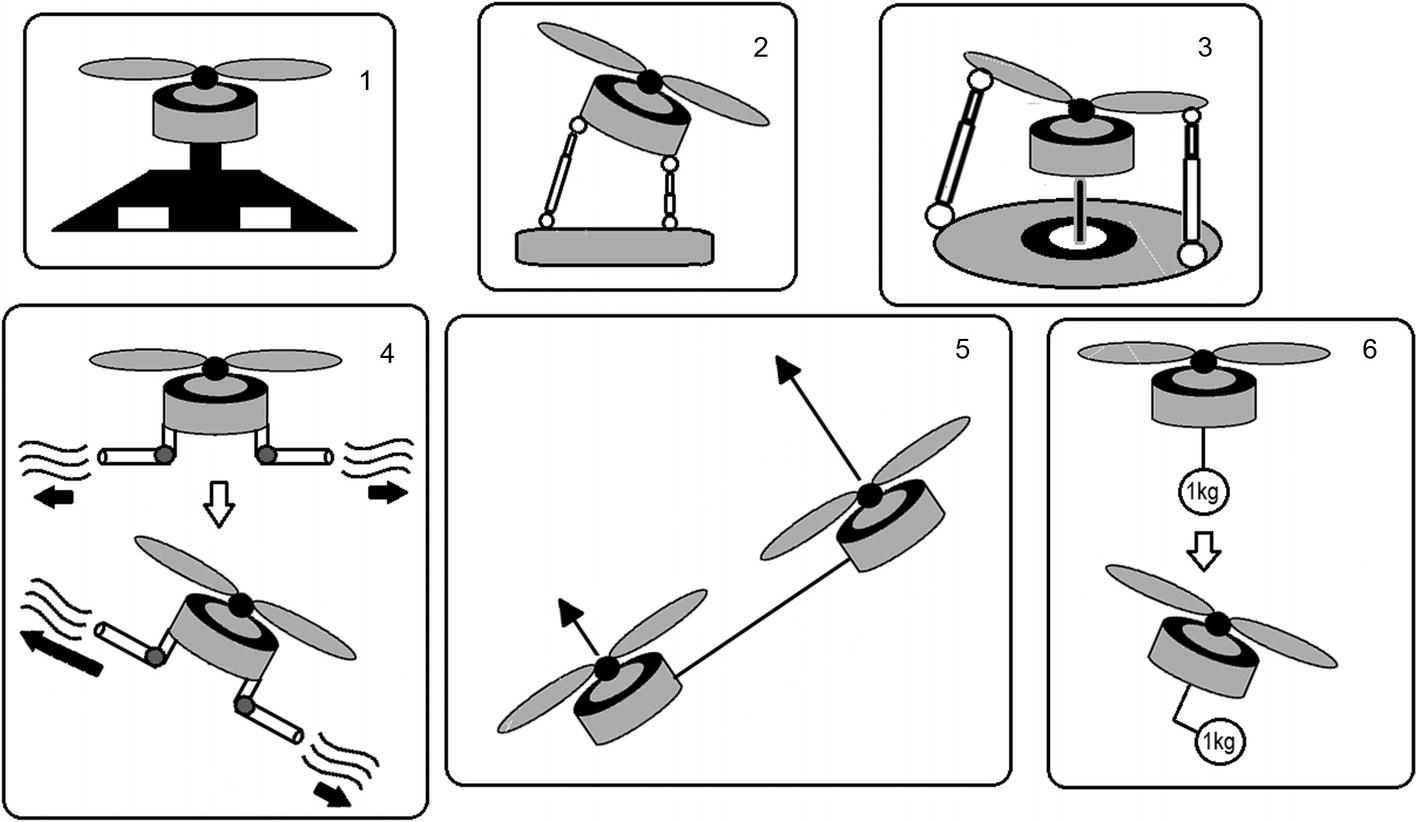
<http://ardupilot.org/dev/docs/editing-the-code-with-eclipse.html#editing-the-code-with-eclipse>

## Appendix 7: Thrust Vectoring

By controlling each one of the vehicle engines including the auxiliary servos, the ArduPilot libraries and Pixhawk autopilot become a team of great features. One of these features is the feasibility of designing unusual or non-existent systems. For this, two concepts are presented: thrust vectoring and omnidirectionality.

Thrust vectoring is the ability to regulate a motor’s main thrust direction. This is achieved through several methods; see Figure [A7-1](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig36) .

1. 1.
2. **With flaps** : This method has been used for decades by airplanes, ships, and cars. It uses one or more fixed main engines, and flaps are used to deflect the airflow (they are usually placed on the wings or the tail). Once the main flow has been redirected, the aircraft can change its flight direction.
4. 2.
5. **With direct movement of the motor** : This method is feasible in toy airplanes since it involves directly moving a full motor, which is already rotating at high speeds, and moving an object that is rotating at high speeds entails a lot of force by the servos that move this engine (due to gyroscopic effects). It has also been used in boats for several decades, where the pilot moves a helm or rudder that deflects the direction where the main engine pushes the boat.
7. 3.
8. **With direct movement of the propeller’s blades** : This is a very useful method in large aircraft such as helicopters. It allows deflecting the aircraft without moving the main rotor—just by moving the blade’s orientation by means of cyclic or collective plates or swashplates.
10. 4.
11. **With pneumatic and vacuum methods** : This is a similar idea to the use of flaps , but instead uses tubes that blow air, liquid, or generate a vacuum to divert the main airflow thrusted by the propellers.
13. 5.
14. **With variations on the effect of multiple engines** : This is how quadcopter drones work. All engines have a fixed position and direction of rotation in a rigid body, and the movement of the body in different directions is achieved by selectively varying each engine’s speed.
16. 6.
17. **With movement of a built-in mass** : In this case, a “massive” object placed in the vehicle’s center of gravity is used. If a direction change is wanted, that mass is moved, and the vehicle is forced to move in the direction where this mass has been placed. This example has been employed for decades with roller skates, kayaks, and motorcycles, where the drivers, in order to change vehicle’s direction, must tilt their body toward the side where they want to move.



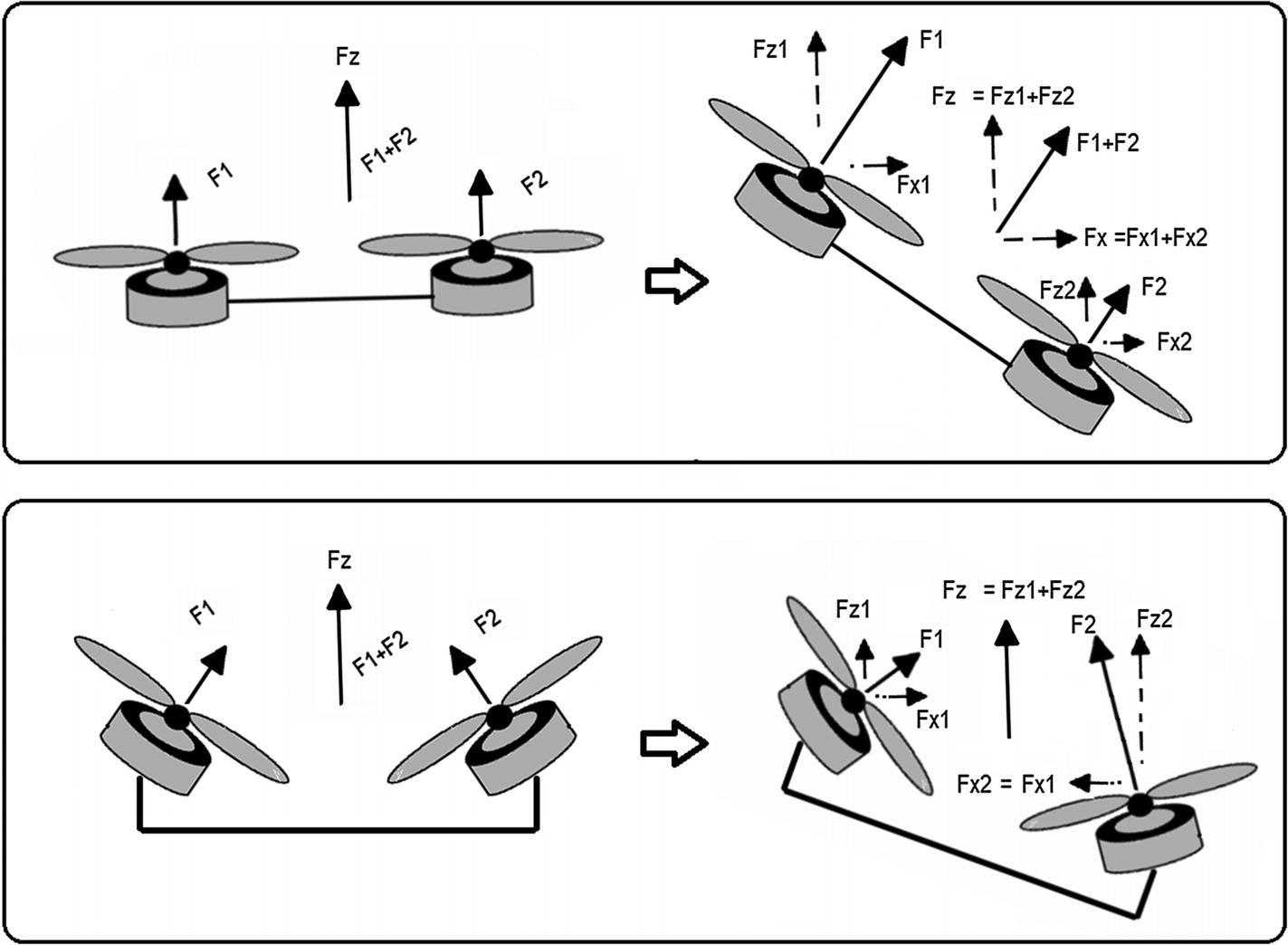
***Figure A7-1*** Thrust vectoring methods

## Appendix 8: Omnidirectionality

It is possible to introduce the concept of omnidirectionality by using one of the vectorization methods previously explained. It basically provides a vehicle with total mobility or the ability to achieve the reached position regardless of orientation.

For example, a standard quadcopter cannot tilt and stay floating at the same time because when it tilts, it tends to move in the direction it has been tilted.

But if a standard quadcopter (or underwater vehicle) is fitted with extra motors different to its planar configuration (like individual vectorizers for each motor), you can get a system that floats at any point in space with variable and independent orientation (maybe full movement, maybe restricted). See Figure [A8-1](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig37) .



***Figure A8-1*** How to obtain omnidirectionality

Note that the control of this kind of vehicle is no longer as trivial as the one presented here for the quadcopter, and you must read about geometric methods of acrobatic or aggressive maneuvers.

In summary, the omnidirectionality is highly feasible with the ArduPilot libraries and Pixhawk autopilot, but it requires greater immersion in mathematical theory.

### REFERENCES AND SUGGESTED WEBSITES

Articles on the different types of vectorization:

J. Pascoa, A. Dumas, M. Trancossi, P. Stewart, D. Vucinic. “A review of thrust-vectoring in support of a v/stol non-moving mechanical propulsion system.”*Open Engineering* , 3(3):374–388, 2013.

C. Bermes, S. Leutenegger, S. Bouabdallah, D. Schafroth, R. Siegwart. “New design of the steering mechanism for a mini coaxial helicopter.” In*Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on* , pages 1236–1241. IEEE, 2008.

J. Paulos, M. Yim. “Cyclic blade pitch control for small uav without a swashplate.” In*AIAA Atmospheric Flight Mechanics Conference* , page 1186, 2017.

X. Yuan, J. Zhu. “Inverse dynamic modeling and analysis of a coaxial helicopters swashplate mechanism.”*Mechanism and Machine Theory* , 113:208–230, 2017.

Articles on omnidirectional aircraft :

D. Brescianini, R. D’Andrea. “Design, modeling and control of an omni-directional aerial vehicle.” In*Robotics and Automation (ICRA), 2016 IEEE International Conference on* , pages 3261–3266. IEEE, 2016.

M. Tognon, A. Franchi. “Omnidirectional aerial vehicles with unidirectional thrusters: Theory, optimal design, and control.”*IEEE Robotics and Automation Letters* , 3(3):2277–2282, 2018.

A. Nikou, G. C. Gavridis, K. J. Kyriakopoulos. “Mechanical design, modelling and control of a novel aerial manipulator.” In*Robotics and Automation (ICRA), 2015 IEEE International Conference on* , pages 4698–4703. IEEE, 2015.

Application of vectorization and omnidirectionality in aerial robotic manipulators:

J. Mendoza-Mendoza, G. Sepulveda-Cervantes, C. Aguilar-Ibanez, M. Mendez, M. Reyes-Larios, P. Matabuena, J. Gonzalez-Avila. “Air-arm: A new kind of flying manipulator.” In*Research, Education and Development of Unmanned Aerial Systems (RED-UAS), 2015 Workshop on* , pages 278–287. IEEE, 2015.

[www.inrol.snu.ac.kr/](http://www.inrol.snu.ac.kr/)

S. Park, J. Her, J. Kim, D. Lee. “Design, modeling and control of omni-directional aerial robot.” In*Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on* , pages 1570–1575. IEEE, 2016.

M. Zhao, T. Anzai, F. Shi, X. Chen, K. Okada, M. Inaba. “Design, modeling, and control of an aerial robot dragon: A dual-rotor-embedded multilink robot with the ability of multi-degreeof-freedom aerial transformation.”*IEEE Robotics and Automation Letters* , 3(2):1176–1183, 2018.

[www.jsk.t.u-tokyo.ac.jp/~chou/](http://www.jsk.t.u-tokyo.ac.jp/~chou/)

D. Mellinger, M. Shomin, N. Michael, V. Kumar. “Cooperative grasping and transport using multiple quadrotors.” In*Distributed autonomous robotic systems* , pages 545–558. Springer, 2013.

Control methods employed with omnidirectional vehicles :

T. Lee. “Geometric controls for a tethered quadrotor uav.” In*Decision and Control (CDC), 2015 IEEE 54th Annual Conference on* , pages 2749–2754. IEEE, 2015.

D. Lee, C. Ha, Z. Zuo. “Backstepping control of quadrotor-type uavs and its application to teleoperation over the internet.” In*Intelligent Autonomous Systems 12* , pages 217–225. Springer, 2013.

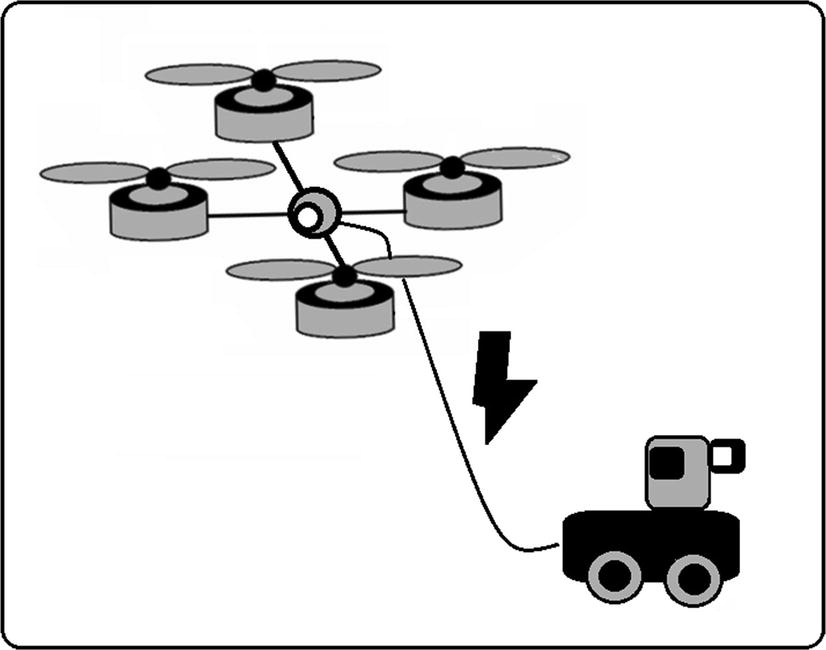
D. Mellinger, N. Michael, V. Kumar. “Trajectory generation and control for precise aggressive maneuvers with quadrotors.”*The International Journal of Robotics Research* , 31(5):664–674, 2012.

H. Abaunza, P. Castillo, A. Victorino, R. Lozano. “Dual quaternion modeling and control of a quad-rotor aerial manipulator.”*Journal of Intelligent & Robotic Systems* , pages 1–17, 2017.

## Appendix 9: Extended Power Methods

As a curious reader, you will have already realized that these vehicles consume a large amount of energy. A single basic brushless motor consumes 12V and 10A on average. This implies using high current energy sources and a power of at least 500 watts. Although LIPO batteries provide these characteristics and portability, they only last between 10 and 30 minutes of flight in average vehicles. In order to satisfy this level of power consumption, there are only three extended energy methods available in the market:

* **Internal combustion** : In this case , motors called glow-engines are used in aeromodelling. However, their application in multicopters is hard and it is only a matter of recent research.
* **Solar energy** : The area that a solar cell must occupy is only viable in fixed-wing aircraft. Research into small size, rotating-wing aircraft and multicopter vehicles is just being developed.
* **Direct electrical connection** : This is a viable way as long as you have the drone operating with an umbilical cord anchored to the ground or to a car. In this case, a ground source provides the necessary power to the vehicle through a series of transformers. It is an interesting option because the operating power allows for a very thin cable operating with high voltage and low current to be then converted to low voltage and high current, thus achieving mobility independence up to 500 meters. This method is widespread and you can find more about it by Googling “tethered drone” or “ tethered vehicle .” See Figure [A9-1](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig38) .



***Figure A9-1*** Tethered vehicle

### REFERENCES AND SUGGESTED WEBSITES

Morus project NATO: [www.fer.unizg.hr/morus](http://www.fer.unizg.hr/morus)

A small discussion of the known energy sources for a drone: [www.techinasia.com/talk/6-known-ways-power-a-drone](http://www.techinasia.com/talk/6-known-ways-power-a-drone)

Thesis and articles about the control of internal combustion engines applied to unmanned aerial vehicles: Paul D. Fjare. “Feedback speed control of a small two-stroke internal combustion engine that propels an unmanned aerial vehicle.” Master’s thesis, University of Nevada, 2014.

Tomislav Haus, Marko Car, Matko Orsag, Stjepan Bogdan. “Identification results of an internal combustion engine as a quadrotor propulsion system, Control and Automation (MED).”*2017 25th Mediterranean Conference on, IEEE, 2017* , pp. 713–718.

Solar drone article: M. Hasan Shaheed, Aly Abidali, Jibran Ahmed, Shakir Ahmed, Irmantas Burba, Pourshid Jan Fani, George Kwofie, Kazimierz Wojewoda, Antonio Munjiza. “Flying by the sun only: The solarcopter prototype.”*Aerospace Science and Technology 45* (2015), 209–214.

Articles on “tethered” drones or drones with specialized electrical extensions:

Beom W. Gu, Su Y. Choi, Young Soo Choi, Guowei Cai, Lakmal Seneviratne, Chun T. Rim. “Novel roaming and stationary tethered aerial robots for continuous mobile missions in nuclear power plants.”*Nuclear Engineering and Technology 48* (2016), no. 4, 982–996.

Christos Papachristos, Anthony Tzes. “The power-tethered uav-ugv team: A collaborative strategy for navigation in partially-mapped environments, Control and Automation (MED).”*2014 22nd Mediterranean Conference of, IEEE,* 2014, pp. 1153–1158.

Tethered units for sale: search keywords “tethered drone”

<http://sph-engineering.com/airmast>

[http://elistair.com](http://elistair.com/)

Future wireless power of drones and other vehicles as well as a state of the art on current energy technologies: Chun T. Rim, Chris Mi.*Wireless Power Transfer for Electric Vehicles and Mobile Devices* , John Wiley & Sons, 2017.

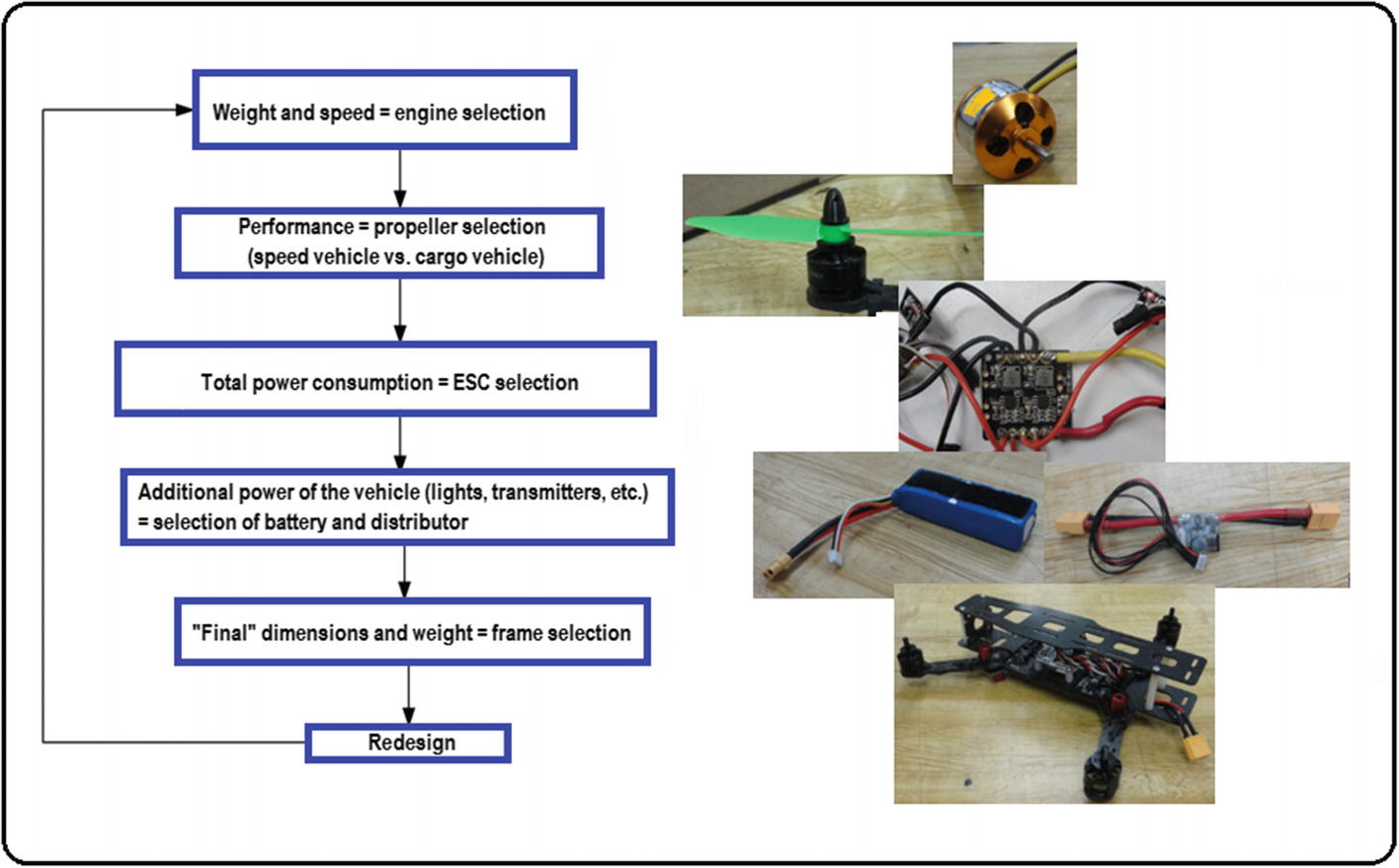
Compact and robust converters useful for tethered vehicles design: [www.vicorpower.com/](http://www.vicorpower.com/)

## Appendix 10: Summary of the Design of a Quadcopter

The design process of a multicopter is illustrated as a flow diagram. This contemplates three aspects that are considered standard and frequent in the design process: the body or vehicle, the brain or autopilot, and the external control or radio control selection. Notice that the selection of sensors has been omitted as it is a very variable task among end users. For example, some users will want cameras, some will prefer LIDARs, ultrasounds, etc. For more information in this regard, consult this appendix’s bibliography.

### VEHICLE DESIGN

The vehicle design is shown in Figure [A10-1](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig39) .

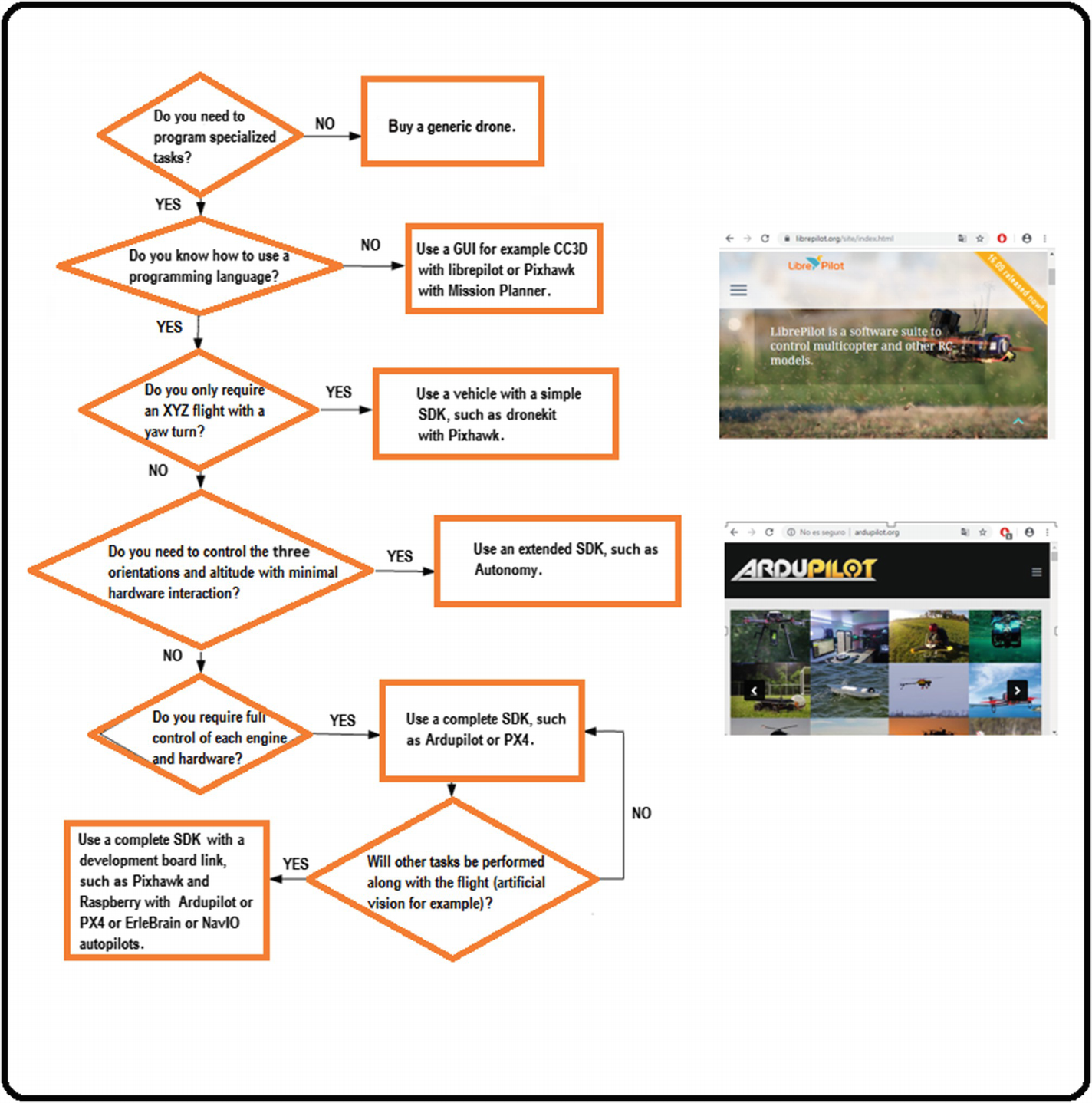


***Figure A10-1*** Multicopter vehicle design

1. 1.
2. **Weight to move + maximum flight speed** : This point concerns motor selection.
4. 2.
5. **Performance of flight (Is it more of an agile, a cargo, or mixed type of vehicle?)** : The answer indicates the selection of propellers.
7. 3.
8. **Motor and propellers power consumption** : This implies a selection of ESCs and BECs.
10. 4.
11. **Total power consumption = ESCs consumption + the rest of the drone (radios, stabilizers, cameras etc)** : This concerns battery selection.
13. 5.
14. **Total dimensions based on the selection of the previous equipment** : This is related to frame selection.
16. 6.
17. A full design means you’re ready to build. Otherwise , a redesign means going back to step 1.

### AUTOPILOT SELECTION

The autopilot selection is shown in Figure [A10-2](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig40) .

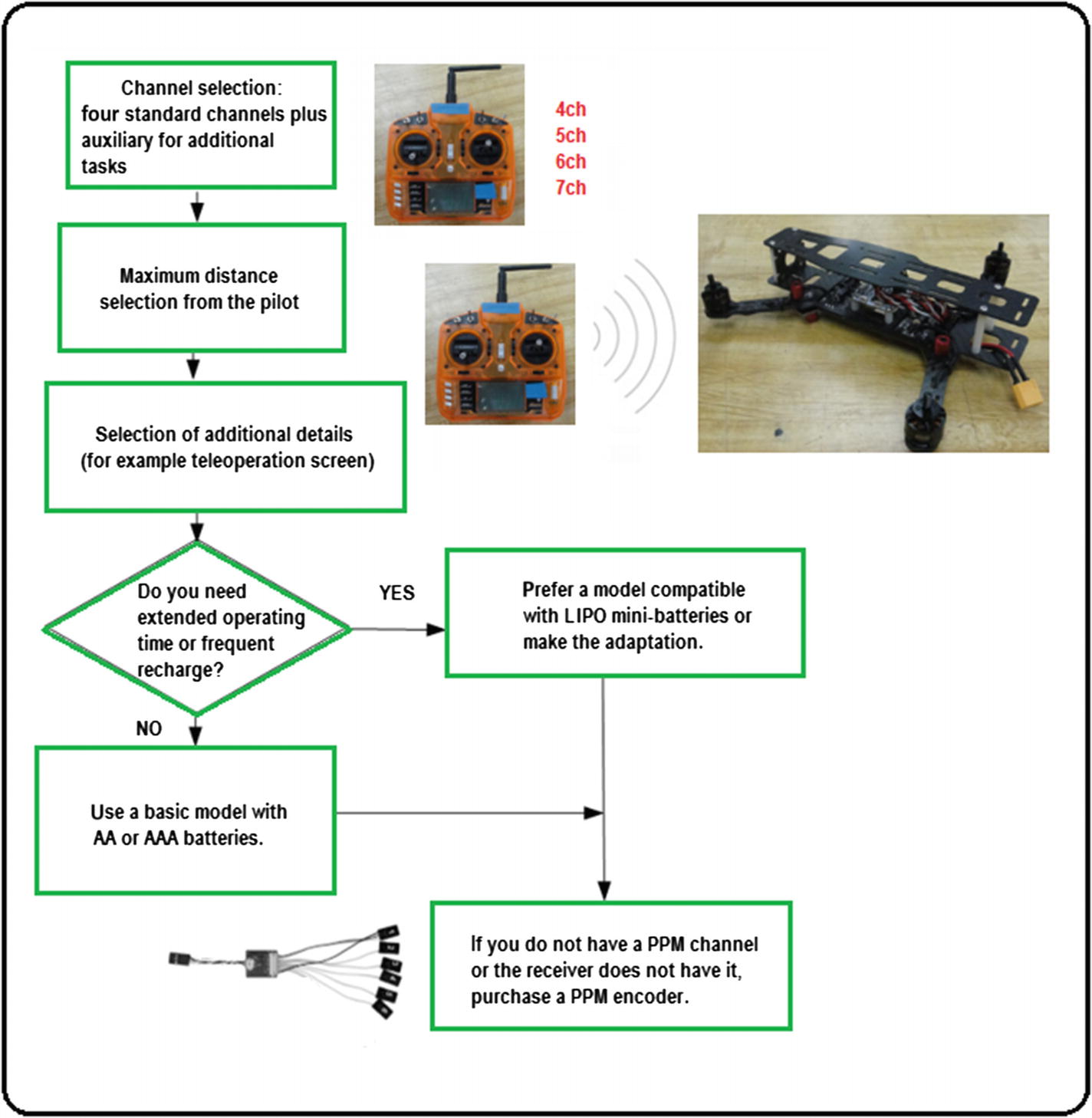


***Figure A10-2***Multicopter autopilot selection

1. 1.
2. Can the required application be done with manual operations? Remember that the GUI has a very robust control, while the SDK allows the reader to operate with the least detail possible but the control and its robustness will be designed by the user.
3. IF the answer is yes, use autopilots based on GUI type CC3D.
4. IF the answer is no, use autopilots based on SDKs, if you know how to program.
6. 2.
7. Does the application demand a particular flight mode X Y Z and turning angle?
8. IF yes, you could use Mission Planner scripts or a simplified SDK.
10. 3.
11. IF no, does the application demand a flight mode with total angular variation and altitude?
12. IF yes, look for an extended SDK.
14. 4.
15. IF not, does the application demand independent control of each engine (for example, a new prototype that does not exist)?
16. IF yes, you must look for a full SDK such as ArduPilot, and a good autopilot such as the Pixhawk.
18. 5.
19. Is the Pixhawk enough for your task?
20. IF no, use development boards combined with autopilots, such as the ErleBrain.

### SELECTION OF THE REMOTE CONTROL

Figure [A10-3](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig41) shows the selection of the remote control .



***Figure A10-3*** Remote control selection

1. 1.
2. How many tasks are required in addition to the four basic movements? For example, a six-channel radio where the two additional channels are levers of type ON/OFF will have a total of four combinations for four other tasks:
3. Aux1 On Aux2 On
4. Aux1 On Aux2 Off
5. Aux1 Off Aux2 On
6. Aux1 Off Aux2 Off
7. Result = four-channel radio + number of extra channels
9. 2.
10. Determine the maximum range distance.
12. 3.
13. Determine additional features.
15. 4.
16. Do you want an extended operation time?
17. If yes, opt for a model with optional LIPO battery power.
19. 5.
20. Does your control have a PPM port? If not, you must buy an adapter.

### REFERENCES AND WEBSITES

Maker-style books that facilitate the understanding and selection of various components of a multicopter:

II Davis, Robert James*. Arduino Flying Projects: How to Build Multicopters, from 100mm to 550mm* , CreateSpace Independent Publishing Platform, 2017.

Terry Kilby, Belinda Kilby.*Getting Started with Drones: Build and Customize Your Own Quadcopter* , Maker Media, Inc., 2015.

Vasilis Tzivaras .*Building a Quadcopter with Arduino* , Packt Publishing Ltd, 2016.

A very complete article that deals with the process of designing, modeling, and controlling a multicopter from the scientific point of view: Hyunsoo Yang, Yongseok Lee, Sang-Yun Jeon, Dongjun Lee. “Multi-rotor drone tutorial: systems, mechanics, control and state estimation.”*Intelligent Service Robotics 10* , 2017, no. 2, 79–93.

On the use of Python scripts in Mission Planner:

<http://ardupilot.org/planner/docs/using-python-scripts-in-mission-planner.html>

<https://github.com/ArduPilot/MissionPlanner/tree/master/Scripts>

About the SDK autonomy for the Parrot Bebop: <https://bebop-autonomy.readthedocs.io/en/latest/>

On the different platforms, prebuilt drones, and navigation cards supported by the ArduPilot libraries: <http://ardupilot.org/dev/docs/building-the-code.htm>

Alternative project Crazyflie: [www.bitcraze.io/getting-started-with-development/](http://www.bitcraze.io/getting-started-with-development/)

## Appendix 11: Working with Header Files

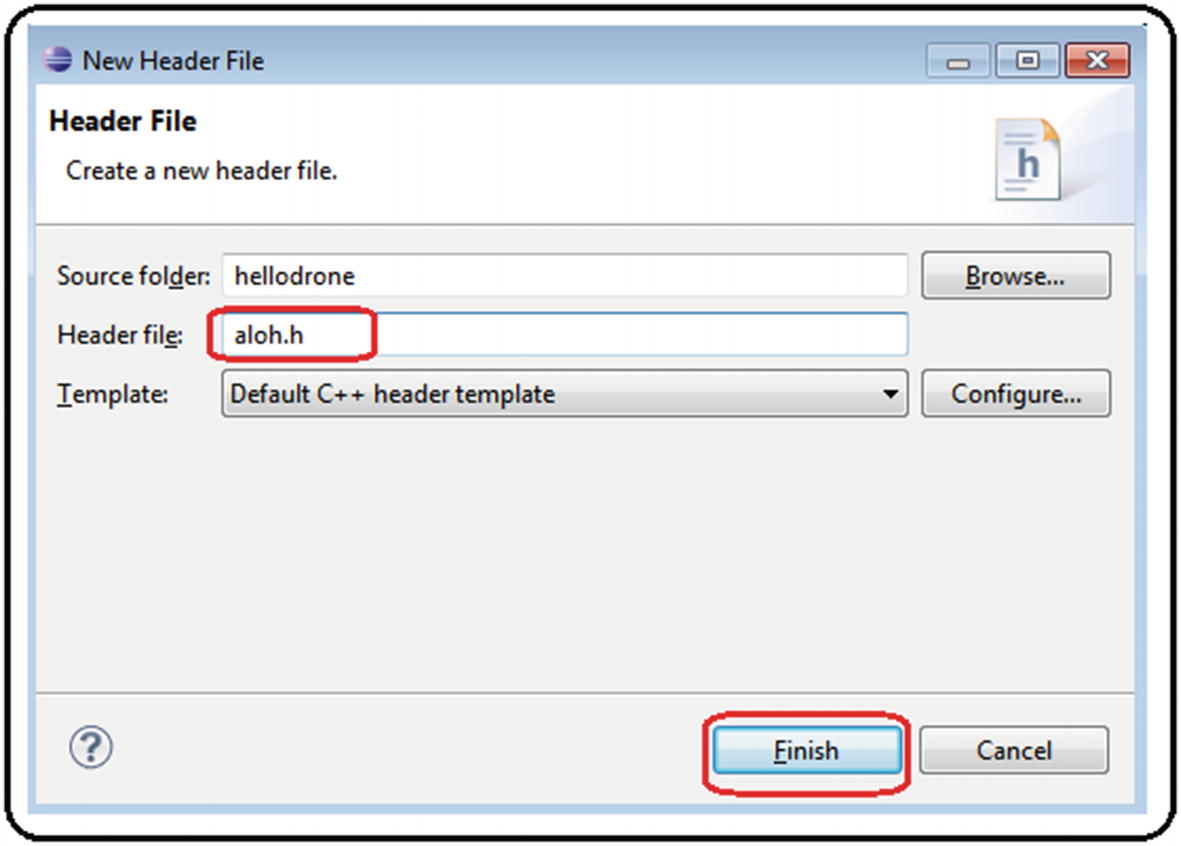
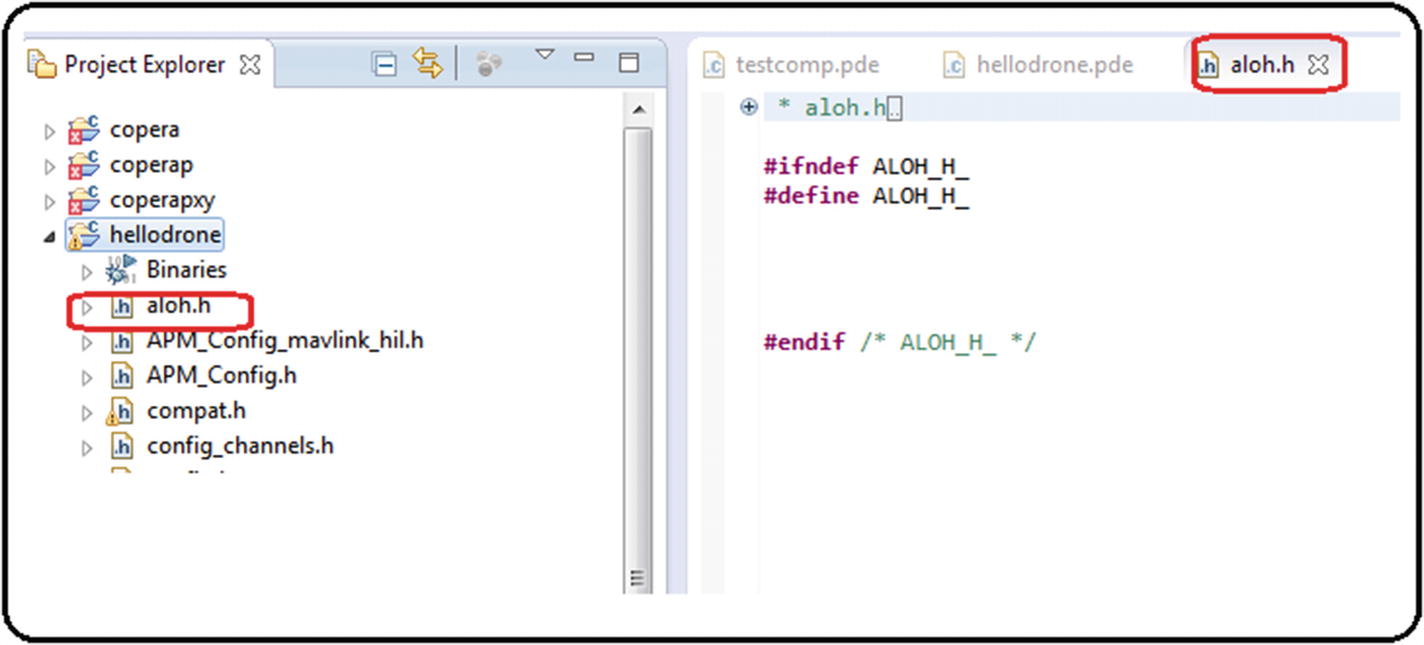
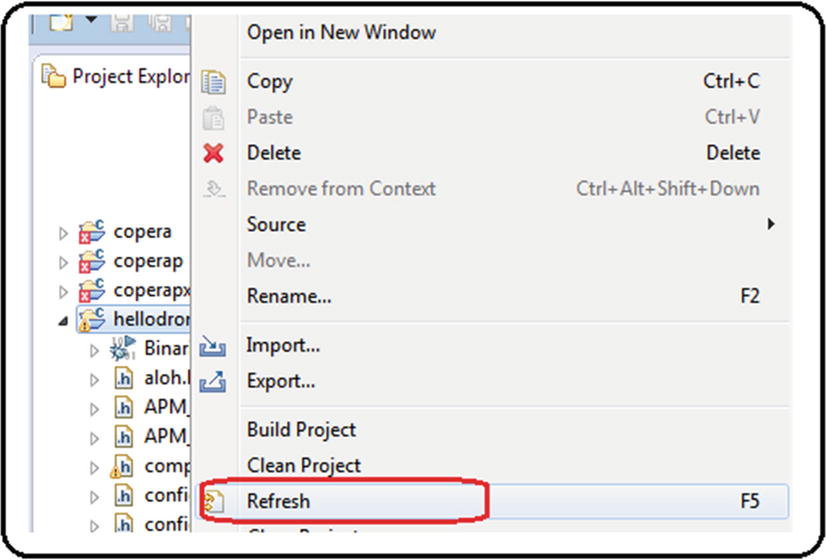
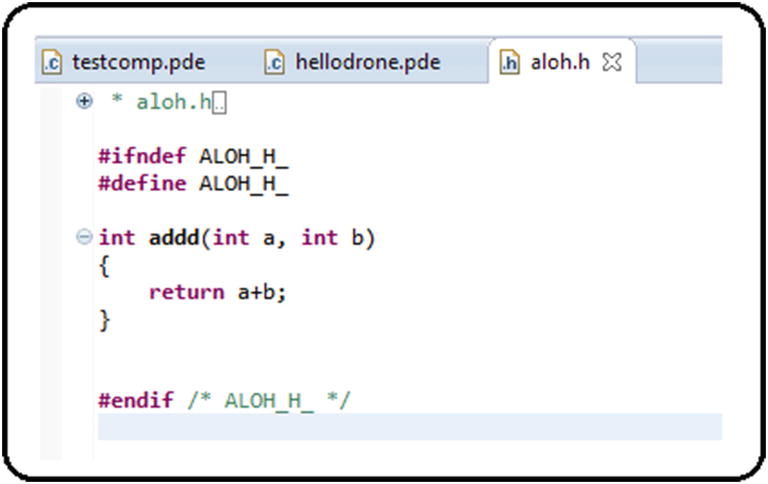
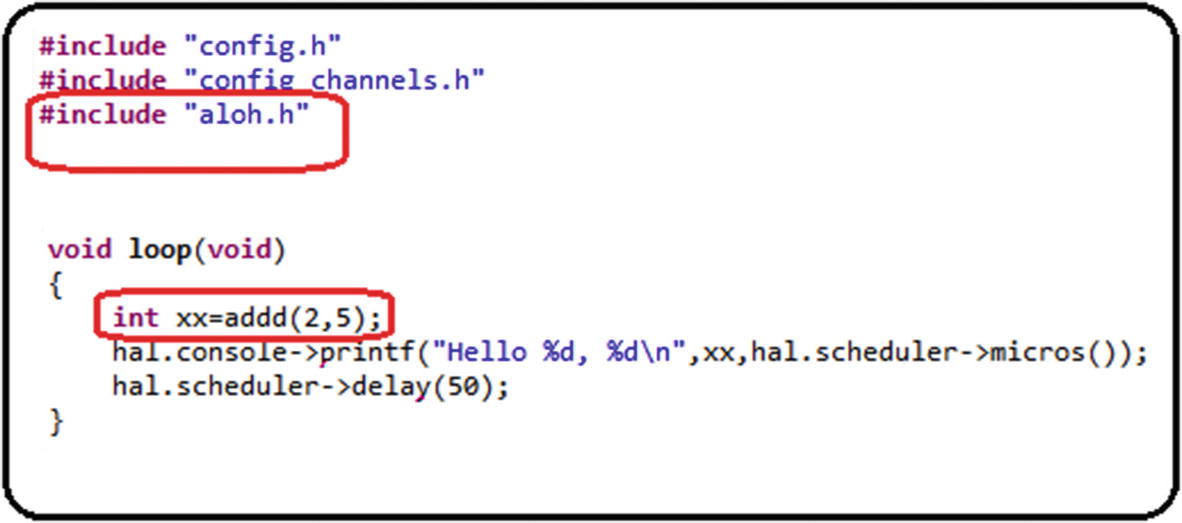
You probably want to work with additional header files beyond those already included in the extended header (defines.h ,configs.h , andcompat.h ). This is possible under the following restrictions. The distribution of ArduPilot libraries included with this book is limited for use with internal libraries (internal with respect to the project). This way, it is only possible to use internal header files (which are defined in the project folder). Also, these header files must contain declarations and definitions in a single file with the extension.h (not as in the common way found in many software projects where the declaration is indicated in an.h file and the definition in a.c or.cpp file).

Having said that, we recommend generating header files that contain very simple definitions, constants, or functions (port registers, control constants, communication speeds, etc.).

Any other way to use header files, such as using commands from external libraries other than ArduPilot, is left as your responsibility. This way, you can search in forums or verify through them if an improvement is available with recent versions or alternative versions of the ArduPilot libraries.

ATTENTION: Do not attempt to invoke the extended header within a header file. For reasons of compilation with the distribution included with this book, Eclipse does not detect the extended header encoded within the header file. You must copy the extended header to the main file for each one of your projects.

Having indicated the characteristics and restrictions of these header files , the procedure to create them is as follows:

1. 1.
2. In Eclipse, right-click the project folder, look for the New tab, and then search for and click the Header File. See Figure [A11-1](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig42) .
3. 
4. ***Figure A11-1***Making header files, step 1
6. 2.
7. Next, you must assign a name with the extension.h ; in this example, it’saloh.h . Then click the Finish button. See Figure [A11-2](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig43) .
8. 
9. ***Figure A11-2***Making header files, step 2
11. 3.
12. Once created , it will appear automatically. Now open the file and edit it. See Figure [A11-3](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig44) .
13. 
14. ***Figure A11-3***Making header files, step 3
16. 4.
17. If it does not appear automatically, right-click the project folder and look for the Refresh tab. See Figure [A11-4](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig45) .
18. 
19. ***Figure A11-4***Making header files, step 4
21. 5.
22. Edit your header file with simple definitions or functions. See Figure [A11-5](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig46) .
23. 
24. ***Figure A11-5***Making header files, step 5
26. 6.
27. To use the definitions or functions coded in the header file, you must first indicate the header’s name between quotation marks in the includes-section of the extended header. After that, the content of these definitions or functions can be used within the main code or in secondary modules. See Figure [A11-6](https://learning.oreilly.com/library/view/advanced-robotic-vehicles/9781484255315/A487932_1_En_BookBackmatter_OnlinePDF.html#Fig47) .
28. 
29. ***Figure A11-6***Making header files, step 6

Index

A

ArduPilot elementary code, parts

Allocation matrix

arm affects, components

control components

factors

geometrical explanation

yaw-dependant

Antivibration mount

Arduino’s serial plotter

Arduino’s transmitter code

ArduPilot-compatible motors

ArduPilot libraries

coding and compiling

compatibilities and projects

display

error checking

compilation process

error line

explicit error message

intentional error

project, secondary files

feedback

file extensions

interface

physical execution

problems

Program flow

projects

ArduPilot libraries employ objects

ArduPilot libraries installation

code editor customization

ArduCopter.pde. file

ArduCopter project

ArduCopter-v2.px4 file

auxiliary window

Console section

C++ source file

hellodrone

jdk-8u111-windows-i586.exe.

PX4 Eclipse

spaces

Welcome tab

compiling libraries

auxiliary screen

Linux-kind commands cd and ls

PX4 ArduCopter firmware

PX4 Console

px4-v2 command

Pixhawk drivers

software download

terminal test

toolchain

uploading .px4 files

Firmware tab

load custom software

location

Mission Planner

warning

websites

Auxiliary robotics components

battery tester/monitor

brushless motors

distributors

embedded on-board computer

ESC

fasteners

frame

function

GPS module

LIPO battery

passive antivibration modules

power module

propellers

remote control

silicon wire

special connector

special Pixhawk components

structure

telemetry module

thermofit

B

Bias translation concept

Bicopter

Brushed/DC motors

Brushless (BLDC) motors

components

description

equipment

keyword

process

Servos and Radios

testing with Arduino

writing to

C

Class definition

Coaxial system

Code optimization

files

hardware and software

Pixhawk autopilot connection

RC output signals

robots

writing to DC motors

writing to motors

Computational efficiency, mathematical equality

Control methods

closed*vs.* open loop

theory and practice

allocation matrix

linear systems control

roll and pitch

sinusoidal forces

spring and damper

translational and rotational models

tuning process

D

Data use and storage

Decoupled tasks

altitude

characteristics

control components

optional tasks

motor

trajectory

orientation

physical explanation

position

remote control

Dirty integration/rectangular approximation algorithm

Distributors

Drone

body frames

coordinate frame

geometric configuration

matrix form

movements

multicopter

problems

propeller

propulsion/allocation matrix

quadcopters

reference frames

rotational frames

self-rotating effect

Drone flight implementation

components

label

tasks

Dynamic rotational equations

body frame

forces and torques

pqr notation

trigonometric properties

Dynamic translation equations

E

Electronic speed controller (ESC)

Euler angles*vs.* pqr variables

Extended code, setup

Extended energy methods

direct electrical connection

internal combustion

solar energy

websites

Extended header

External Position Sensors Reading (GPS)

dirty integrator algorithm

read planar positions and velocities

F

Fasteners

Flight\_modes() function

Flight modes

aggressive, 283ArduPilot libraries

automatic mode

classifications

*vs* . control methods*vs* . task planning

kinodynamic

rudeness

soft

Functional code

G

GPS module

H

Header files

Hovering

I, J

Input and output operations

altitude and vertical velocity readings

analog sensors, reading

angular singularity

angular velocity readings

auxiliary channels

battery reading

command

digital reading and writing

drift

electrical noise

filtering signals

header

code

commands

LED, using visual alerts

map values, function

position and orientation internal sensors reading

saturation values, function

setup

state machine

terminal reading

Installation and coding

K

Keyboard motor calibration

L

LIPO battery

Log\_Write functions

Lyapunov methods

M

Map function

Mathematical stabilization methods

microSD card

flight data

Mission planner interface

Module

Motor effect

roll/pitch lever

throttle lever

yaw lever

Multicopter design process

autopilot selection

remote control selection

vehicle design

websites

N

Newton’s second law

Normalization

O

Omnidirectionality

websites

Orientation and position

P

Pixhawk autopilot

clones*vs* . originals

commercial autopilot*vs* . design

custom code, uploading

data types

definition

development card

features

GUI*vs* . SDK interfaces

ports

PPM, PWM, 1000/2000, implementation

projects

SDKs, types

types

Pixhawk code

Pixhawk’s receptor code

Pixhawk type X configuration

Pixhawk type X quadcopter

Planar vertical takeoff and landing (PVTOL)

coaxial

control analysis

non-coaxial

Polling method

Programming ArduPilot code, models

Propellers

Propulsion matrix

Q

Quadcopter

coordinate frames

linear-attitude equations

non-linear equations

resume of

R

Radio control motor calibration

Radio reading

control signals

limiting values

map function

saturation function

tests

Read\_radio() function

Real time mode

control.pde module

data.pde module

hovering control, quadcopter

Module radio.pde

pose.pde module

state machine

Real-time working environment, ArduPilot libraries

execution time, measuring

runtime task

scheduler

system clock invoation

linker

parts

scheduler

Remote control components

S

Saturation function

Save\_data() function

Scheduler

SDK commands, comparison

Serial.write() command

Servomotors

Signal biasing, effects of

Signal saturation, effects of

Software development kit (SDK)

Special Pixhawk components

Spring-damper type system

Stepper motors

ArduPilot libraries

Pixhawk

Symmetric quadcopter angles

T, U

Telemetry module

Tethered vehicle

Thematic keywords

Thermofit

Thrust vectoring method

Time management, ArduPilot libraries

Translation value

Trigonometric properties

V

Variable

Velocity kinematic relations

angular variations

Euler angles

frames*vs.* translational velocities

planar axes

translational velocities

W

Wired/wireless serial communication

checksum method

reception

sending

components

connection

data reception process and recap

data verification procedure

description

development boards

external sensors

geometrical interpretation of module and residue

polling method

sending data

telemetry device configuration

UART devices

XOR type checksum

X, Y, Z

XOR checksum method

# **Embedded Programming with Modern C++ Cookbook**

Copyright © 2020 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 author, nor Packt Publishing or its dealers and distributors, will be held liable for any damages caused or alleged to have been 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.

**Commissioning Editor:** Richa Tripathi

**Acquisition Editor:** Vincy Davis

**Content Development Editor:** Pathikrit Roy

**Senior Editor:** Storm Mann

**Technical Editor:** Pradeep Sahu

**Copy Editor:** Safis Editing

**Project Coordinator:** Francy Puthiry

**Proofreader:** Safis Editing

**Indexer:** Pratik Shirodkar

**Production Designer:** Jyoti Chauhan

First published: April 2020

Production reference: 1170420

Published by Packt Publishing Ltd.

Livery Place

35 Livery Street

Birmingham

B3 2PB, UK.

ISBN 978-1-83882-104-3

[www.packt.com](http://www.packt.com/)

# **About the author**

**Igor Viarheichyk** works as an engineering manager at Samsung, developing a safety-critical middleware platform for advanced driver assistance systems aimed at specialized automotive embedded platforms. Prior to joining Samsung, in the past 20 years of his career, he has played different roles, from software engineer to software architect, to engineering manager in a variety of projects, and he has gained vast experience in the areas of system programming, embedded programming, network protocols, distributed and fault-tolerant systems, and software internationalization. Though he knows and actively uses programming languages such as C, Java, and Python, C++ is his language of choice to implement large-scale, high-performance applications.

*I would like to thank Pathikrit Roy, Content Development Editor, and Tanvi Bhatt, Project Manager of this book, for their guidance, attention to detail, and dedication, essential to bring this book to life.*

*Many thanks go to Antonio Calderone for the technical review of the book and for testing all code samples. Your suggestions were extremely valuable.*

*Finally, I want to thank my family for their patience and support.*

# **About the reviewer**

**Antonino Calderone** has worked in the computer software industry for over 20 years as a software engineer in various domains, including telecommunications and networks, embedded systems, cybersecurity, machine learning algorithms, and DBMS. He has worked for companies such as Ericsson, Intel, and McAfee. He has also been a security architect, technical writer, and teacher in programming courses in C++ and design patterns. Antonino is an author and maintainer of several open source projects, including mipOS, an RTOS for SoC microcontrollers, and nuBASIC, a language designed for educational purposes. He was also a contributor to the magazine *Computer Programming*, one of the most well-known programming magazines in Italy in the 90s and 2000s.

# **Packt is searching for authors like you**

If you're interested in becoming an author for Packt, please visit [authors.packtpub.com](http://authors.packtpub.com/) and apply today. We have worked with thousands of developers and tech professionals, just like you, to help them share their insight with the global tech community. You can make a general application, apply for a specific hot topic that we are recruiting an author for, or submit your own idea.

# **Table of Contents**

1. [Preface](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/0269324f-145a-4de4-b0ab-b57e1ac6b799.xhtml)
   1. [Who this book is for](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/ebef314b-a3fa-476b-bd79-9e717af110e7.xhtml)
   2. [What this book covers](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/91a4d42d-ed21-4d4a-8e43-9b6cd9dc4c18.xhtml)
   3. [To get the most out of this book](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/73a38ba8-8f67-410c-be38-95ce4e2fd856.xhtml)
      1. [Download the example code files](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/416caad9-e1a1-45fd-ab2c-ad4d7528b0a4.xhtml)
      2. [Download the color images](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/eb4921e9-9f01-4e86-84d8-3a6656b82d68.xhtml)
      3. [Conventions used](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/5d053fdf-8521-4926-b686-5c3013f82dea.xhtml)
   4. [Sections](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/cc716b1b-5019-4a17-9dc3-f4ab3381f659.xhtml)
      1. [Getting ready](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/9cd93899-b24e-425d-af51-7df37e9b4ed8.xhtml)
      2. [How to do it&#x2026;](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/804d1639-d415-455c-93f4-140e4800efc1.xhtml)
      3. [How it works&#x2026;](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/2c709317-7392-44ee-b662-b7a8741932e2.xhtml)
      4. [There's more&#x2026;](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/e0280f7c-358c-483f-bc11-3aae8d542559.xhtml)
      5. [See also](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/348864a6-e6bc-446a-9f96-58ee1abacfd9.xhtml)
   5. [Get in touch](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/07926dcf-7e7f-45b4-b277-543a12177392.xhtml)
      1. [Reviews](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/07f1cf3e-3450-4342-89c9-cd2cd86a2ac1.xhtml)
2. [Fundamentals of Embedded Systems](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/f1f00b5a-3564-48c2-a50f-4c612ce00967.xhtml)
   1. [Exploring embedded systems](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/63bfd501-ed83-40b7-b506-ffc4919a8140.xhtml)
      1. [How are they different from desktop or web applications?](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/73c8667f-f3e8-4e53-b401-f0623721dd4b.xhtml)
      2. [Types of embedded systems](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/01ab2e14-5731-4378-ae57-232a97d909ce.xhtml)
      3. [Microcontrollers](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/4030a196-9461-4eb1-ab97-93968d7589ae.xhtml)
      4. [System on Chip](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/2ef853ba-7798-416c-b484-fd395ae973bd.xhtml)
      5. [Application-specific integrated circuits](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/c3d72309-467a-4d38-927c-61c539954f13.xhtml)
      6. [Field programmable gate arrays](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/a7958d23-8afe-4292-a1cf-5dcd129f073d.xhtml)
   2. [Working with limited resources](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/12335766-1122-40a9-9c43-df59d1454341.xhtml)
   3. [Looking at performance implications](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/1ce61393-5581-4a0a-988c-5142ca2568f6.xhtml)
   4. [Working with different architectures](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/815c5412-20ac-4cd8-a2af-6284491e2731.xhtml)
      1. [Endianness](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/6958eafb-bb4b-4eab-bf28-10ad7bfab6b4.xhtml)
      2. [Alignment](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/06426472-a1ff-48d4-847a-157109fef6a6.xhtml)
      3. [Fixed-width integer types](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/49b2f01a-f8fa-4ed4-ab92-5c4e7c7638e9.xhtml)
   5. [Working with hardware errors](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/febd8705-facc-494c-b3f1-2c857eba2d5f.xhtml)
      1. [Early versions of hardware](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/88d05d79-722c-4efc-bccb-73f2103102f5.xhtml)
      2. [Hardware is unreliable](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/0f456809-445b-4598-b71c-d392c2736d5f.xhtml)
      3. [The influence of environmental conditions](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/020a02fc-2e97-45f0-9a43-1cbbc5b7850d.xhtml)
   6. [Using C++ for embedded development](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/c15c690e-3f88-4866-a751-738e61743be1.xhtml)
      1. [You don't pay for what you don't use](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/f74831ee-88aa-4b0a-ba4d-ea4a43d3e144.xhtml)
      2. [Object-oriented programming to time the code complexity](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/90c43af1-677b-47b4-811a-a728cb07ca8e.xhtml)
      3. [Resource acquisition is initialization](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/1b35c59c-a9d4-4193-b273-bfd288d070bf.xhtml)
      4. [Exceptions](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/0df2e08f-b211-43b2-8b3b-8169671bac1b.xhtml)
      5. [The powerful standard library](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/6cd4f9e5-81eb-44d2-b883-de655b38bef2.xhtml)
      6. [Threads and a memory model as part of the language specification](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/0cdf7bb0-e7f8-4beb-b8fd-a2e490156c9a.xhtml)
   7. [Deploying software remotely](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/170056d9-2b90-462e-984e-02369ce7d93f.xhtml)
   8. [Running software remotely](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/cf9624e0-396a-44e1-8f85-1b1c5e12273f.xhtml)
   9. [Logging and diagnostics](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/5f5b32ba-5e4d-457d-9af6-94b0e2e69be1.xhtml)
   10. [Summary](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/ed16ac40-037a-407f-bb88-f40dc9af861a.xhtml)
3. [Setting Up the Environment](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/7c7e4cbe-659a-4ce5-8519-7e23ca277141.xhtml)
   1. [Setting up the build system in a Docker container](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/c2e2ac0d-2b78-4e21-8e51-8a8a66406c73.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/180acf57-341c-47ce-b53f-c9961be7a2c4.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/9e45a5d7-83e7-42a2-86b2-142c3ba53437.xhtml)
      3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/80c903c5-b9c7-47de-9a04-5af15766a247.xhtml)
   2. [Working with emulators](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/d5b0de31-a61c-4831-a58f-5af60f02e055.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/21bcd47c-d6b9-4353-942f-5a11ac8023ed.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/2f3aca1e-81b3-4f0b-9e2d-68c40e649a0d.xhtml)
      3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/3e3fdf72-5e77-4b90-b217-f15a6a1009bc.xhtml)
   3. [Cross-compilation](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/5d431ecb-0294-4bec-add7-3a76f51076c7.xhtml)
      1. [Getting ready](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/d3e2ca77-ed3d-4595-bf4c-04a607fcc85f.xhtml)
      2. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/cdcf399b-6474-4aab-8aa5-fcf3dd49d128.xhtml)
      3. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/ec294f06-ecfd-4cb1-8900-5583ec373605.xhtml)
      4. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/d18913f9-c30d-4c1d-8369-a3658a97f1af.xhtml)
   4. [Connecting to the embedded system](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/567cd12f-db4c-468b-b67b-7e3ea3308d06.xhtml)
      1. [Getting ready](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/5528cce2-d7ae-454a-b3a4-50e4fac6b65b.xhtml)
      2. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/19df1283-3263-4c59-a9eb-0801e86b2811.xhtml)
      3. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/11f14abd-3166-4fbd-b2ea-39aa8a580546.xhtml)
      4. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/e5d61f22-1346-4348-b886-6c1e438949b3.xhtml)
   5. [Debugging embedded applications](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/be2a6ab8-3080-41b7-a9fe-8342e37e56af.xhtml)
      1. [Getting ready](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/f8801e51-2be0-4002-abf8-2b853449ce02.xhtml)
      2. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/a60b1248-5193-44ff-9511-2216b386470a.xhtml)
      3. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/db2e7fdc-e964-40a1-a697-e007cdbf8f7e.xhtml)
      4. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/d37e1034-a0f4-4a22-b5f0-70462efe5f02.xhtml)
   6. [Using gdbserver for remote debugging](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/b438adba-5ca0-4f1a-bbab-ee0af8f36dc7.xhtml)
      1. [Getting ready](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/293e0dd7-8649-4949-9bdc-1c5dd2f56760.xhtml)
      2. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/8cef48f1-fb90-467a-a4bf-e4afd93e3cea.xhtml)
      3. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/1b5ca380-3821-4890-956f-1425e91f5d1c.xhtml)
      4. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/571e3730-7042-4b0a-b2e6-99462b2d9485.xhtml)
   7. [Using CMake as a build system](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/5c0ffcc3-01de-4a01-84a9-dd51333f1ab6.xhtml)
      1. [Getting ready](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/b987ab7d-3058-411a-9e3d-a6c2805f544b.xhtml)
      2. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/c9635b75-c3df-4115-9077-9a3f46f59e43.xhtml)
      3. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/49af5227-853f-43ca-b206-17066dd86b74.xhtml)
      4. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/6dab1b25-4857-4729-98d5-61578a3c96f8.xhtml)
4. [Working with Different Architectures](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/6a1c9def-a2ae-437d-a246-376006f022b3.xhtml)
   1. [Exploring fixed-width integer types](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/75a5afbe-1e75-4b0d-8485-bc6b690e984e.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/041b10ea-a14e-4ef9-8260-8da1fffe2f3a.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/31482e95-1521-470e-8d1e-19c49c6ca3a7.xhtml)
      3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/6a624e88-61cd-4737-9f11-8670169dac71.xhtml)
   2. [Working with the size\_t type](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/0015868f-c55e-4dd2-b84f-725181162211.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/0ef272dd-a804-4932-8d58-7c9fc8a36439.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/7dcea76e-1f3d-4731-8f9f-2f0a865ffb7e.xhtml)
      3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/f64a108b-48c9-474c-8ec3-7eaa4a81e6cc.xhtml)
   3. [Detecting the endianness of the platform](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/a377f29a-69cb-41dc-9876-bdbb4caea97a.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/51cab548-d189-4134-97e9-65200c7362ae.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/bd78c590-35ff-4821-98b0-b52c42e5ba17.xhtml)
      3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/94ea0068-a93f-4c6c-83f9-1246e6d326ed.xhtml)
   4. [Converting the endianness](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/80dd4d0a-563c-410f-bb5e-7dc037fa65e2.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/f336dc96-de31-4f9b-aaed-3c2039801f59.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/c06bfbde-40e4-4cd2-9c50-18f84ec39912.xhtml)
   5. [Working with data alignment](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/73ab1a27-16b1-4111-901f-a52201f91f5f.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/680d0d35-5791-46f5-8440-70b5c0e2936a.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/973bb9c5-f02a-45d9-9ce2-5e677d051071.xhtml)
      3. [There's more...&#xA0;](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/f72edfe7-901e-4392-884e-4ce7057a56e4.xhtml)
   6. [Working with packed structures](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/6c5926bf-952a-43ab-9786-a771a9308efb.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/666a6338-ce27-42d8-bfd4-1e6e18bc0359.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/fbc956ac-afce-49c0-9068-8c974ff5d9fb.xhtml)
      3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/b238d1b0-a541-4892-87f3-96fd3a14aac5.xhtml)
   7. [Aligning data with cache lines](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/7e79925c-14d7-4545-9b4e-1ecfa3724fe7.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/56665821-f055-4364-8f3f-e450e9969030.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/c1d2c613-af26-4655-88df-932c24ac6cb3.xhtml)
      3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/4480e634-cd49-46ad-95f2-430177cd6d7c.xhtml)
5. [Handling Interrupts](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/2ef9e6e7-5714-46cf-96e4-77e00c956304.xhtml)
   1. [Data polling](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/0d3f271e-11dd-4842-a410-bfec6c7c1268.xhtml)
   2. [Interrupt service routines](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/03838fd0-6506-4b3b-904d-1acee962e429.xhtml)
   3. [General considerations for ISRs](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/00de3162-0aee-4f18-84d2-ee41c0a0a994.xhtml)
   4. [8051 microcontroller interrupts](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/dabaf509-e586-4e38-9302-c4b5fcc29f58.xhtml)
   5. [Implementing an interrupt service routine](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/8ced8963-4b7d-4b13-a92a-139c7d08d028.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/10211987-5b94-4f51-aba4-75485b03e9be.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/61afbfa2-3624-44ed-97b7-78da8c78d5ef.xhtml)
   6. [Generating a 5 kHz square signal using 8-bit auto-reload mode](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/4b58f66f-2c63-4b5a-ad72-758268e306ea.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/01f5e80c-af2d-4c6c-b2ad-f858df1ff868.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/6f70bf81-92c5-4185-8855-9eb6c6ec3991.xhtml)
   7. [Using Timer 1 as an event counter to count a 1 Hz pulse](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/ca017168-5c5a-47ce-baa5-d3a3e0927a2a.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/87d4e4fb-f565-4488-ad5f-d378806071b2.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/f544af45-6a06-48a9-b70d-56bacc61f80a.xhtml)
      3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/3c5851c6-d61d-42b6-9428-92af484b8fff.xhtml)
   8. [Receiving and transmitting data serially](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/8a6130f3-0fd3-41e3-882c-b6b08882c702.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/bacc8692-d7b1-47f7-82c1-377befeda526.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/1ed4c2f4-7a05-4d31-b731-687b022d8eba.xhtml)
      3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/663b00ba-2e5f-431c-bfe9-a02c1d213015.xhtml)
6. [Debugging, Logging, and Profiling](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/8c8ebe40-5e7e-490e-a5ee-1a88710b78a2.xhtml)
   1. [Technical requirements](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/4ba28f50-7d4e-4c1e-9e17-bad30fabb0a3.xhtml)
   2. [Running your applications in the GDB](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/c91aecf0-fd8a-4c47-9ee4-eb47755a3661.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/1209bc77-a61d-4bbc-a295-8f4adcac9399.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/93971d1c-0f5a-4f86-ba9c-c9d47f35c187.xhtml)
      3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/25fac9a7-32fb-4f1e-9ecc-2857756faf30.xhtml)
   3. [Working with breakpoints](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/8f2d3ec2-8bd2-4043-b5b8-903f3d6f6a93.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/4bf02555-f17d-493f-98d1-408b57d9e088.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/17fe18a4-c176-41b2-bf3a-95be7ece3af9.xhtml)
      3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/b2ab0e34-18f2-4218-a57e-3a2c9b1f2f2d.xhtml)
   4. [Working with core dumps](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/eb6a8a2e-9d82-44c3-a1ab-2c57b85fcb98.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/857feb57-a536-4852-9281-9379be2e0c7c.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/6d366e62-0405-4a2d-a8ca-f18bfdb07a61.xhtml)
      3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/00af4a93-9cf7-4fdf-bcd9-1b11dae19fba.xhtml)
   5. [Using gdbserver for debugging](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/928c7178-9a61-44b7-90ef-36436dc065f6.xhtml)
      1. [Getting ready...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/9c3f69b5-6795-4761-aefe-ab20f4cabe4a.xhtml)
      2. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/4d2a1c3b-039b-4046-966b-d83b02c50cca.xhtml)
      3. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/362bb7a8-7c90-45b1-8ef6-12f9e9c94f6f.xhtml)
   6. [Adding debug logging](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/c47166f3-8971-492e-9c3d-6234c4582552.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/7377b628-9789-491f-9ce3-bb5cd7ebccf5.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/5f2bd728-26af-4468-8546-6817e2c34b04.xhtml)
      3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/a373e68b-93ef-4ee6-876a-03673423c22b.xhtml)
   7. [Working with debug and release builds](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/c4af68af-315c-45c7-b16a-98eb1727a2d9.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/98f87ac3-5734-477a-b48e-0c5183ef508e.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/c74885c2-cff3-4a4f-9bc9-c706d2bbe2f8.xhtml)
      3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/35532e3f-6a89-40d8-93ca-7a4033efa1f8.xhtml)
7. [Memory Management](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/a6d7d28d-8bb3-4141-a134-316601fa8bff.xhtml)
   1. [Using dynamic memory allocation](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/6d854383-dab0-4970-87b5-e784e3aa9c1d.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/b431928e-047e-4576-92e5-08ca5de9d9de.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/0e968123-9234-4700-bb08-265bf4bc9557.xhtml)
   2. [Exploring object pools](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/b0c48dc5-4ea2-48a8-af68-4d0ec9f76a27.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/ee81dec6-6b66-4b72-b10d-a3dbb530da27.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/663d0a99-572f-487d-a827-e3d961e38239.xhtml)
      3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/88320af0-c813-41bb-9ea7-282f6bd11d96.xhtml)
   3. [Using ring buffers](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/d50d6bf4-a8a3-4ed3-843c-059af3209e31.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/a97b7ef0-9282-4b8c-a83a-300ea098d267.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/d40afc28-85a9-434b-96eb-a145cbf28d48.xhtml)
   4. [Using shared memory](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/ff91dc08-7c1c-4af2-a147-f6c7e70c4d51.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/8a983220-4919-485a-829b-139046a875e6.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/e6e0b8b5-f5a7-4e16-929b-673f3cbcebfd.xhtml)
      3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/b6e7c516-0343-41f4-9399-445524e79484.xhtml)
   5. [Using specialized memory](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/7ee12321-fbaf-4382-a185-a8612f5a43f7.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/51139457-0dcf-471b-a455-8f2ad9db9a66.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/e1b12c90-fce5-4ece-9153-d0c828f26c8c.xhtml)
      3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/780a63de-0740-44aa-b46c-df6e7d091129.xhtml)
8. [Multithreading and Synchronization](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/005b6d09-8c01-48e6-8855-983862ac07d1.xhtml)
   1. [Exploring thread support in C++](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/7238c467-26f7-4696-bfcf-7e934f6aecdc.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/b5fa1cbc-2682-454b-945e-6a41fe6bd1a1.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/07bf48c6-3a3d-4cdd-b014-790d4e3bfa0f.xhtml)
   2. [Exploring data synchronization](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/1dce5bbe-d326-43a6-878e-3a769a728b88.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/79b900c4-fbe3-4753-9464-9bc855998adc.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/666de2e2-981e-48f0-95e1-e5ecc370d46c.xhtml)
      3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/56073129-0fad-4490-a553-86525b82e911.xhtml)
   3. [Using condition variables](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/0f375cc5-db7f-44e0-b404-99248f6baa85.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/56ae84cf-89f6-4bf7-b74f-4201b9f60ab3.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/04352660-823e-40dd-909f-46f420d3b625.xhtml)
      3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/d73ecb46-3544-427d-ae8d-d631a6904599.xhtml)
   4. [Using atomic variables](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/15457b32-1c4b-4b6d-998f-0b801e33f53b.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/0a0c9d84-c279-4066-9c5e-443ae92d97ec.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/8278520b-105e-420b-8e80-7f768cf844b0.xhtml)
      3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/44a96a92-8201-42cf-a512-9892c64bd028.xhtml)
   5. [Using the&#xA0;C++ memory model](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/f0ec5f13-6063-4153-b811-01d2d1b443d3.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/eb9e0530-db0c-4432-96cc-5c27661520a3.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/ab4d3c75-c3fc-4f3e-880f-4771d01fdeb0.xhtml)
      3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/97115739-5c29-478b-9573-3b348cc80b70.xhtml)
   6. [Exploring lock-free synchronization](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/a75083b8-5480-4727-88b7-12a2b1ab6440.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/d55fa4b1-8b0f-4e66-9803-14f25c219e28.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/6ef97314-a133-4fd5-8dd4-53d3e3ea78bf.xhtml)
      3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/59f1d188-5704-4001-8bf3-6f5b6e9a7804.xhtml)
   7. [Using atomic variables in shared memory](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/5c88d9b7-24cd-48c8-a00b-603912cd9d5e.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/4aecbe99-3c52-4a0a-bb66-694f5788a47f.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/8f385f60-8f62-4b59-a80f-17a757d33cec.xhtml)
   8. [Exploring async functions and futures](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/c28c682d-02fc-4b7c-85f2-c952314ab2b0.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/ffcdcfeb-bc27-4a55-921c-1a98e272c8f3.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/f11482cc-f784-4098-ae42-b5425d9c14b4.xhtml)
      3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/730d1f20-65ed-444b-82ee-754c862810a7.xhtml)
9. [Communication and Serialization](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/ea5334c5-eb78-4147-b53d-5caae134197f.xhtml)
   1. [Using inter-process communication in applications](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/d960e191-c59a-4ef3-90d7-86da4af9aa29.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/673fbefb-7886-41cf-bf76-95909959d2f7.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/ded856f0-3faf-4d90-81ad-21579b2b98ce.xhtml)
      3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/e5cf1741-ab38-4cfe-902f-1b73357fedf1.xhtml)
   2. [Exploring the mechanisms of inter-process communication](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/e0c712d7-b8b5-4247-8bb1-9acb3fec4596.xhtml)
      1. [Getting ready](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/51a98835-6237-45d9-92ac-36091a4b17eb.xhtml)
      2. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/a3ddb37a-999d-44b4-a34c-0dc27eba5688.xhtml)
      3. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/87a7efe0-1483-4b15-8104-f951ed55717c.xhtml)
      4. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/07c6546b-5860-4d6b-ab15-da392695b044.xhtml)
   3. [Learning about message queue and publisher-subscriber models](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/2b5d12ee-f7e7-4043-a68f-b1d59a60dda8.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/26ff3851-bcaa-488d-a0f9-9d89d2ccfd2b.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/5aab4895-bc65-4c57-ad67-a34e9e88d81a.xhtml)
      3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/0b984f2a-7dba-451d-b42a-dae5bffa0408.xhtml)
   4. [Using C++ lambdas for callbacks](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/06e3b927-e98f-400d-afe8-68c398a00ebb.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/f5dcbc92-4135-4425-b612-eac90c32255b.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/51f26032-f1fa-4d4d-b58b-b9535bd96b8c.xhtml)
      3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/8db57a72-29a2-4bb1-99b7-32df91ca9b4f.xhtml)
   5. [Exploring data serialization](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/438b5d9b-38fa-470f-9346-7c80ecf3e367.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/02f4c887-cb0c-45ed-8fbd-2d16e12e6c6f.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/82bdac96-3912-470f-9b02-53c94fb21934.xhtml)
      3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/426da5ed-a2e1-44f5-935e-1c84ee175a89.xhtml)
   6. [Using the FlatBuffers library](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/f76677c4-1ccf-40ad-afd1-2040f1856b14.xhtml)
      1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/e218df60-6751-49e2-a906-a4ad88c9e773.xhtml)
      2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/e6293c99-43f3-4282-a3d9-19f044151533.xhtml)
      3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/1c32dcab-c32c-4ab1-8bf1-1b79d2a65625.xhtml)
10. [Peripherals](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/374d92b0-0194-46a6-a8c1-1d2ca4d3cc9a.xhtml)
    1. [Controlling devices connected via GPIO](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/b6351d82-d739-4ffa-82f5-8d6dbe9f19da.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/036e6cfa-4833-450f-85bc-dcc7fa78fc43.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/aab2528d-9fa1-4fe3-9a25-9c7039ee5b1b.xhtml)
    2. [Exploring pulse-width modulation](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/08cfc22c-dcc4-4298-8c58-1ef9b55ca74c.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/90dd2f49-d693-4403-9e47-af231cf8c325.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/6da0166c-cda4-425c-9dbf-7087ba5dfbcf.xhtml)
       3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/2cb231f9-6ef7-4538-8076-cd2fbc540a0d.xhtml)
    3. [Using ioctl to access a real-time clock in Linux](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/0abb88f2-2667-4855-8b28-ac258d8945bf.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/1398260b-bc82-4cc6-9181-89f404c330ff.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/46df03a6-93c8-4542-a77c-38bb9e63720d.xhtml)
       3. [There's more](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/93a29015-9397-422f-ae07-56b41f73560c.xhtml)
    4. [Using libgpiod to control GPIO pins](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/ffe6f279-006c-45e0-ae09-69267e8e2826.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/30069073-f84e-4c67-871f-bfb3767a2792.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/09fa5396-7702-4ede-b837-0d4d68347e0c.xhtml)
       3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/5b031f23-3d72-46c4-9056-87b882553ccd.xhtml)
    5. [Controlling I2C peripheral devices](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/07d8035a-b38f-45d2-a250-b39ed67e81f3.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/2de77055-46ae-456a-bb03-0715c7fa3b33.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/3cf381de-bbd5-41a5-97f2-6e137adc3d2f.xhtml)
       3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/fadf3e6f-7fc4-460d-b0d0-67a993a18370.xhtml)
11. [Reducing Power Consumption](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/39481f50-3bd5-4157-832d-40c3bbb9b607.xhtml)
    1. [Technical requirements](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/8ebc58c6-7f5f-4a46-a2e5-218481571e3b.xhtml)
    2. [Exploring power-saving modes in Linux](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/a3272989-05d3-44f9-9d75-dd6736ca8307.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/8111a633-b64f-42e1-a23c-e7c6746ea90d.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/47229034-35cd-4666-82c7-846b5ceea127.xhtml)
       3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/a6e922a1-98e1-41e4-9656-a71c38ce0e94.xhtml)
    3. [Waking up using RTC](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/701edf0d-8084-4b3a-ac48-d2bc012db45f.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/0d2aff88-297b-4957-aaa9-11dc8310660a.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/c5316b01-f5fc-4140-a936-97a4a972b166.xhtml)
       3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/649b10bd-53a8-40d7-b402-d0f7c23e54b4.xhtml)
    4. [Controlling the autosuspend of USB devices](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/dc531ef7-cd4a-4767-984f-c72240872a26.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/f02edca7-b1de-4e1c-8467-51490c241652.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/e730338d-c8d9-4417-8744-08c3f4188e65.xhtml)
       3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/ac75f5e8-0946-49b4-90f6-09c0334a657c.xhtml)
    5. [Configuring CPU frequency](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/1b615341-49d5-449e-88e3-37c4344592d1.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/b8eb80a3-8a24-4fd0-8bdd-54575df6435f.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/694abc71-3707-45d5-8656-5c36e53a2d96.xhtml)
       3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/6c089106-ea4e-4e54-a867-331adcde2d6d.xhtml)
    6. [Using events for waiting](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/e8b1ee94-7768-4da6-9184-1b2367e00a14.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/ea9ba21a-a63c-42f3-b93a-3c688f974903.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/f48c3404-e54a-4a8e-876e-3886b6056382.xhtml)
       3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/b874b083-a134-4c26-9280-da7cab5f2a84.xhtml)
    7. [Profiling power consumption with PowerTOP](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/5a3a2ba0-cce0-4ac7-bae6-fc3f1f3bf0ba.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/f9e560b9-3f4d-496b-8f1a-edda8a0f9ae9.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/0f8ad72a-d3d7-405f-964a-8c24048ea4be.xhtml)
       3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/d4f2af3a-ef75-477f-842b-66a4c9de012c.xhtml)
12. [Time Points and Intervals](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/94b37ae2-a82e-44b5-bed9-f491fdf669f3.xhtml)
    1. [Exploring the C++ Chrono library](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/cac7a2df-0a25-4fab-b09c-b2623b25923c.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/6b9c108a-aec0-41ed-bbd1-5d4f71950b8f.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/9ab59287-3cf8-4a1f-90ed-2820fb6661c8.xhtml)
       3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/a5df9a2b-c11d-4e0c-bc8d-4370e2ce1cf0.xhtml)
    2. [Measuring time intervals](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/4f17d324-776a-4aa9-888e-0dfa611d77ac.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/3cf2b30e-0e64-45d4-9372-b1f359fb9112.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/ee6eb850-66f2-443a-8255-0d909e76856e.xhtml)
       3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/d25ed962-7448-4b08-a8cf-f9ff6c99c59e.xhtml)
    3. [Working with delays](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/80221595-1afa-423e-b1c0-34719b843cfa.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/cd206825-582d-42cf-93b4-7bfed765a1df.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/238e46ba-95ab-480a-8633-dcee27064f4b.xhtml)
       3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/050989ca-4d80-4ca7-85bf-930a6dd04f07.xhtml)
    4. [Using the monotonic clock](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/befa652a-6623-4371-8e6d-317c9c77bca4.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/b9e7099f-86af-4bc5-9f43-744709e399d7.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/a15a3553-2d37-4981-b215-f2d3ba647dc8.xhtml)
       3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/040051fe-e0d5-49c3-a921-39ba12825e24.xhtml)
    5. [Using POSIX timestamps](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/9f57c249-e8ee-4179-a598-1915dc18faf6.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/0ea8e0d3-71c6-4719-86bd-eb578a3eb707.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/1445460f-a9d8-4a3f-9fdd-82e52c5f840f.xhtml)
       3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/4ab9622d-0e19-48c1-b42f-87a6024a51b6.xhtml)
13. [Error Handling and Fault Tolerance](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/334b7510-57b1-43cf-8a56-034a11db95f1.xhtml)
    1. [Working with error codes](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/c2724f9c-beb3-41a6-a794-7a84c5144c1e.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/d03bee52-10fd-4bd6-999e-1288dc3956de.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/05865559-36f1-46ac-82ec-d77585707193.xhtml)
       3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/7f1b5e23-fea9-4024-a540-47e1732d20f2.xhtml)
    2. [Using exceptions for error handling](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/2ba86678-5d59-496a-9886-93c8366219c6.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/cd5b6f9c-4e14-4514-b65a-e1c495b62f9b.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/e1128722-6a95-418b-b8c7-2468c9cf6f45.xhtml)
       3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/bad3097e-072e-40d6-8d71-0e62bce74a14.xhtml)
    3. [Using constant references when catching exceptions](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/e0cfcd42-5b7b-40d2-8f76-23a4ff047a52.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/6f0d06f1-e065-4472-b124-86c930aa551c.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/c821e5d6-8f52-4d3e-9a23-106113021837.xhtml)
       3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/69bd1c93-35cd-4ad0-b68c-ca15e2cd48e9.xhtml)
    4. [Tackling static objects](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/d30133c4-e4ea-42e0-be2d-a702188112bf.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/81448743-65e2-4b9e-a830-69d1b9117c92.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/26530a0a-f803-4412-844b-c3b4aaad76b7.xhtml)
    5. [Working with watchdogs](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/06bebdb8-3c1a-4eb3-81ce-604ed1609b16.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/98d407ff-18f7-4593-969b-789cd73c96b0.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/a6299280-64a4-4aef-9ec5-6655b4b5fd96.xhtml)
    6. [Exploring heartbeats for highly available systems](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/710d977f-8b83-4efc-9566-fe249a4f5990.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/932a8530-21b1-489f-9736-5b7983fd5093.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/bf489f8c-0a78-4917-9ebb-7e76cb54806f.xhtml)
       3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/2134138c-2fd1-420d-ab36-c33fca89b8f6.xhtml)
    7. [Implementing software debouncing logic](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/8fdfbe39-9ef2-463a-b286-a5024e12eb05.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/d458ac4c-5033-4b6e-84a4-f5a3f6aa766a.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/866e5d92-87fa-4d13-a90c-3d65200ce849.xhtml)
14. [Guidelines for Real-Time Systems](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/87cb9462-11d9-4a31-841f-ad2f124ec4e7.xhtml)
    1. [Using real-time schedulers in Linux](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/409f5aad-e2e4-4a9e-9a87-cab6632e453a.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/1ce95606-da74-439a-87e7-55a6c17c2bf1.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/c1293dcb-6231-4f94-9efd-617940971480.xhtml)
    2. [Using statically allocated memory](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/d9ca611b-7056-402e-90b8-ba457701271e.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/53ad3e92-f7cb-4b1b-85d7-a78f26679dc2.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/19d68ca1-45a9-4449-9df9-5742be30dd0b.xhtml)
       3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/c9cbc8d5-549f-4092-a986-2f043d72ffeb.xhtml)
    3. [Avoiding exceptions for error handling](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/80305529-2edd-4176-a3e2-fd2ed193bd2b.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/11cd8b9b-e086-4413-afc3-cec31f289b49.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/38d45144-8430-461a-b59d-9b39ee89857d.xhtml)
       3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/a83d4847-acb9-47d8-a2e7-7234d0ee7d5c.xhtml)
    4. [Exploring real-time operating systems](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/2dcc28ee-7a0a-4ba3-a986-58d5e2f82f26.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/aadf2412-db93-4798-83c4-146cf0101c9d.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/26bf4d37-3096-462d-b92d-22280a41bd24.xhtml)
       3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/6d9f8527-2a65-42ac-be98-64165c3d8051.xhtml)
15. [Guidelines for Safety-Critical Systems](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/0ee5fd15-2cc6-4806-8001-4d897e0e81fd.xhtml)
    1. [Using the return values of all functions](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/0a3a39d0-7fc1-484b-afe7-fb0beefcf7de.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/e6c1326e-e9e2-4eeb-ad04-0a659c92d9de.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/49053ddb-a211-4e75-a418-657a611a2ee9.xhtml)
       3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/655d20f9-106c-414a-950f-4c60090b0b0e.xhtml)
    2. [Using static code analyzers](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/69dc091e-bf32-42e7-9ff0-1761d309576e.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/03f5d6c2-b3ea-4103-beca-61f520ea1d31.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/16ac6ce8-a749-4b59-ae24-cb032552a089.xhtml)
       3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/c8d0eadb-f6ff-4c15-a2c9-add61fbdccc6.xhtml)
    3. [Using preconditions and postconditions](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/c92d7b12-3bc9-4c7e-9b5f-5288572c3a4f.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/17b488a2-ddf0-4089-a17e-a4f1c9cc54ce.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/dc40823f-6d88-41fa-9f78-976312e0ec2d.xhtml)
       3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/1bb44c90-2c1b-4b28-afce-3b1de381c619.xhtml)
    4. [Exploring the formal validation of code correctness](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/0b7ff8cb-0fed-413c-9f92-088e8232e950.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/bf1b8312-4577-4126-8ecf-556cd48ea4bd.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/a2769051-b6c3-4658-a57b-86f6f887ddd0.xhtml)
       3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/216cd550-e38b-49fe-a244-e3b1c5deb1c0.xhtml)
16. [Microcontroller Programming](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/c06fbff9-bb3a-4adf-b85c-3ffebd7f8264.xhtml)
    1. [Setting up the development environment](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/0313eb7b-cad8-465b-bf0c-46268ec663c4.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/10f0975f-9381-4996-9ade-e24e172c8234.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/2e42e098-9d85-4c0f-9d54-b1c63dbf80ed.xhtml)
       3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/fa02e7ea-b2ff-4812-9169-e3fa48ba7417.xhtml)
    2. [Compiling and uploading a program](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/96a3df87-196d-44cf-b921-f3b9dcdfabdb.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/5dfe560e-f614-46e4-92e4-685480b7dba8.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/739279d2-1449-45c9-8d52-2b1d8ac2d0ac.xhtml)
       3. [There's more...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/0d1e2b7a-6b82-483f-a483-3154d34330b4.xhtml)
    3. [Debugging microcontroller code](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/56d9427f-76cb-4e61-8730-d343ea619290.xhtml)
       1. [How to do it...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/0f1b12f9-76fa-415b-9996-c8320a1d5a35.xhtml)
       2. [How it works...](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/88afe8c5-56fb-4434-9e34-65b05aca1c89.xhtml)
17. [Other Books You May Enjoy](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/3f4b4ea1-a407-487d-8981-5f92468a307a.xhtml)
    1. [Leave a review - let other readers know what you think](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/db9fe2fe-032e-4981-bfba-1e6657a77088.xhtml)

# **Preface**

For a long time, development for embedded systems required either plain C or assembly language. There was a host of good reasons for this. The hardware did not have enough resources to run applications written in higher-level programming languages, such as C++, Java, or Python, but more importantly, there was no real need to write software in these languages. Limited hardware resources put a limit on software complexity, the functionality of embedded applications remained relatively simple, and the capabilities of C were sufficient to implement it.

As a result of the progress in hardware development, more and more embedded systems nowadays are powered by inexpensive yet powerful System-on-Chip capable of running a general-purpose multitasking operating system such as Linux.

Growing hardware capabilities demand more complex software, and more and more often C++ becomes the language of choice for new embedded systems. With its *you don't pay for what you don't use* approach it allows developers to create applications that use computational and memory resources, like applications written in C, but gives developers many more tools for dealing with complexity and safer resource management, such as object-oriented programming and the RAII idiom.

Seasoned embedded developers with substantial experience in C often tend to write code in C++ in a similar, habitual way, considering this language just as an object-oriented extension of C, a *C with classes*. Modern C++, however, has its own best practices and concepts that, properly used, help developers avoid common pitfalls and allow them to do a lot in a few lines of code.

On the other side, developers with C++ experience entering the world of embedded systems should be aware of the requirements, limitations, and capabilities of specific hardware platforms and application domains and design their C++ code accordingly.

The goal of this book is to bridge this gap and demonstrate how features and best practices of modern C++ can be ap

# **Who this book is for**

This book is for developers and electronic hardware, software, and system-on-chip engineers who want to build effective embedded programs in C++.

The world of embedded systems is vast. This book tries to cover one type of them, the SoCs running Linux OS, such as Raspberry Pi or BeagleBoard, briefly touching low-level microcontrollers such as Arduino.

Familiarity with C++ is expected, but no deep knowledge of C++ or experience with embedded systems is required.

# **What this book covers**

[Chapter 1](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/f1f00b5a-3564-48c2-a50f-4c612ce00967.xhtml), *Fundamentals of Embedded Systems*, defines what embedded systems are, how they are different from other systems, why specific programming techniques are needed, and why C++ is good and in many cases the best choice for embedded development. It outlines the constraints and challenges that embedded developers encounter in their everyday work: limited system resources and CPU performance, dealing with hardware errors, and remote debugging.

[Chapter 2](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/7c7e4cbe-659a-4ce5-8519-7e23ca277141.xhtml), *Setting Up the Environment*, explains the differences in a development environment for embedded systems compared to web or desktop application development and goes through concepts of the build and target system, cross-compilation and cross-toolkits, the serial console, and the remote shell. It provides practical steps for setting up virtualized build and target hosts for the most common desktop configurations running Windows, macOS, or Linux.

[Chapter 3](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/6a1c9def-a2ae-437d-a246-376006f022b3.xhtml), *Working with Different Architectures*, explains how to take into account important differences in CPU architectures and memory configuration of target systems in your C++ code.

[Chapter 4](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/2ef9e6e7-5714-46cf-96e4-77e00c956304.xhtml), *Handling Interrupts*, covers the low-level concepts of interrupts and interrupt service routines. In modern OSes, even developers or device drivers have to use a higher-level API provided by the OS. That is why we explore the interrupt techniques using the 8051 microcontroller.

[Chapter 5](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/8c8ebe40-5e7e-490e-a5ee-1a88710b78a2.xhtml), *Debugging, Logging, and Profiling*, covers debugging techniques specific to Linux-based embedded systems, such as running gdb directly on the target board, setting up gdbserver for remote debugging, and the importance of logging for debugging and failure root cause analysis.

[Chapter 6](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/a6d7d28d-8bb3-4141-a134-316601fa8bff.xhtml), *Memory Management*, provides several recipes and best practices of memory allocation that will be helpful for developers of embedded systems. We discuss why dynamic memory allocation is avoided in embedded applications and what alternatives can be considered for fast, deterministic memory allocation.

[Chapter 7](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/005b6d09-8c01-48e6-8855-983862ac07d1.xhtml), *Multithreading and Synchronization*, explains how to use the functions and classes provided by the standard library of C++ to implement efficient multithreading applications that can utilize all the power of the modern multicore CPUs.

[Chapter 8](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/ea5334c5-eb78-4147-b53d-5caae134197f.xhtml), *Communication and Serialization*, covers the concepts, challenges, and best practices for inter-process and inter-system communications, such as sockets, pipes, shared memory, and memory-efficient serialization using the FlatBuffers library. Decoupling applications into independent components that talk to each other using well-defined asynchronous protocols is a de facto standard way of scaling a software system while keeping it fast and fault-tolerant.

[Chapter 9](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/374d92b0-0194-46a6-a8c1-1d2ca4d3cc9a.xhtml), *Peripherals*, explains how to work with various peripheral devices in C++ programs. Though most device communication APIs do not depend on a particular programming language, we will learn how to use the power of C++ to write wrappers that are convenient for developers and help prevent common resource leaking errors.

[Chapter 10](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/39481f50-3bd5-4157-832d-40c3bbb9b607.xhtml), *Reducing Power Consumption*, explores the best practices for writing energy-efficient applications and utilizing the power management functions of the OS. It provides several practical recipes for Linux-based embedded systems, but the same concepts can be expanded to any OS and any platform.

[Chapter 11](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/94b37ae2-a82e-44b5-bed9-f491fdf669f3.xhtml), *Time Points and Intervals*, covers various topics related to time manipulations, from measuring intervals to adding delays. We will learn about the API provided by the standard C++ Chrono library and how it can be used efficiently to build portable embedded applications.

[Chapter 12](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/334b7510-57b1-43cf-8a56-034a11db95f1.xhtml), *Error Handling and Fault Tolerance*, explores possible implementations and best practices of error handling for embedded applications written in C++. It explains how to use C++ exceptions efficiently and compares it to alternatives such as traditional error codes and complex return types. It touches on basic fault-tolerance mechanisms such as watchdog timers and heartbeats.

[Chapter 13](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/87cb9462-11d9-4a31-841f-ad2f124ec4e7.xhtml), *Guidelines for Real-Time Systems*, covers the specifics of real-time systems. It briefly describes how real-time systems are defined and what kinds of real-time systems exist. It contains practical recipes on how to make the behavior of applications more deterministic, a crucial requirement for real-time systems.

[Chapter 14](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/0ee5fd15-2cc6-4806-8001-4d897e0e81fd.xhtml), *Guidelines for Safety-Critical Systems*, explains what safety-critical systems are and how they are different from other embedded systems. It covers development methodologies and tools that are required when working on safety-critical systems, from following formalized coding guidelines such as MISRA, AUTOSAR, or JSF to using static code analysis or formal software validation tools.

[Chapter 15](https://learning.oreilly.com/library/view/embedded-programming-with/9781838821043/c06fbff9-bb3a-4adf-b85c-3ffebd7f8264.xhtml), *Microcontroller Programming*, outlines basic concepts of writing, compiling, and debugging C++ code for microcontrollers. We will learn how to set up the development environment using the widely used Arduino board as an example.

# **To get the most out of this book**

Development for embedded systems implies that your applications will interact with some sort of specialized hardware—a specific SoC platform, a specific microcontroller, or a specific peripheral device. There is a huge variety of possible hardware configurations, along with specialized OSes or IDEs that are needed to work with those hardware setups.

The goal of this book is to let everyone start learning about programming for embedded systems without investing too much in hardware. That is why most of the recipes are aimed at working in a virtualized Linux environment or an emulator. Some of the recipes, however, may require physical hardware. These recipes were designed to be run on either a Raspberry Pi or an Arduino, the two most widely used and inexpensive platforms that can be obtained relatively easily.

|  |  |
| --- | --- |
| **Software/Hardware covered in the book** | **OS requirements** |
| Docker  (<https://www.docker.com/products/docker-desktop>) | * Microsoft Windows 10 Professional or Enterprise 64-bit * macOS 10.13 or newer * Ubuntu Linux 16.04 or newer * Debian Linux Stretch (9) or Buster (10) * Fedora Linux 30 or newer |
| QEMU  (<https://www.qemu.org/download/>) | * Windows 8 or newer (32-bit or 64-bit) * macOS 10.7 or newer * Linux (various distributions) |
| Raspberry Pi 3 Model B+ |  |
| Arduino UNO R3 or ELEGOO UNO R3 |  |

**If you are using the digital version of this book, we advise you to type the code yourself or access the code via the GitHub repository (link available in the next section). Doing so will help you avoid any potential errors related to the copying and pasting of code.**

# **Download the example code files**

You can download the example code files for this book from your account at [www.packt.com](http://www.packt.com/). If you purchased this book elsewhere, you can visit [www.packtpub.com/support](https://www.packtpub.com/support) and register to have the files emailed directly to you.

You can download the code files by following these steps:

1. Log in or register at [www.packt.com](http://www.packt.com/).
2. Select the Support tab.
3. Click on Code Downloads.
4. Enter the name of the book in the Search box and follow the onscreen instructions.

Once the file is downloaded, please make sure that you unzip or extract the folder using the latest version of:

* WinRAR/7-Zip for Windows
* Zipeg/iZip/UnRarX for Mac
* 7-Zip/PeaZip for Linux

The code bundle for the book is also hosted on GitHub at <https://github.com/PacktPublishing/Embedded-Programming-with-Modern-CPP-Cookbook>. In case there's an update to the code, it will be updated on the existing GitHub repository.

We also have other code bundles from our rich catalog of books and videos available at <https://github.com/PacktPublishing/>. Check them out!

# **Download the color images**

We also provide a PDF file that has color images of the screenshots/diagrams used in this book. You can download it here: <https://static.packt-cdn.com/downloads/9781838821043_ColorImages.pdf>.

# **Conventions used**

There are a number of text conventions used throughout this book.

CodeInText: Indicates code words in text, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input, and Twitter handles. Here is an example: "Run the hello application under gdbserver."

A block of code is set as follows:

#include <iostream>

int main() {

std::cout << "Hello, world!" << std::endl;

return 0;

}

When we wish to draw your attention to a particular part of a code block, the relevant lines or items are set in bold:

#include <iostream>

int main() {

std::cout << "Hello, world!" << std::endl;

**return 0;**

}

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

**$ docker run -ti -v $HOME/test:/mnt ubuntu:bionic**

**Bold**: Indicates a new term, an important word, or words that you see onscreen. For example, words in menus or dialog boxes appear in the text like this. Here is an example: "The best way to configure cross-compilation for CMake is by using the so-called **toolchain** files"

*Warnings or important notes appear like this.*

*Tips and tricks appear like this.*

# **Sections**

In this book, you will find several headings that appear frequently (*Getting ready*, *How to do it...*, *How it works...*, *There's more...*, and *See also*).

To give clear instructions on how to complete a recipe, use these sections as follows:

# **Getting ready**

This section tells you what to expect in the recipe and describes how to set up any software or any preliminary settings required for the recipe.

# **How to do it…**

This section contains the steps required to follow the recipe.

# **How it works…**

This section usually consists of a detailed explanation of what happened in the previous section.

# **There's more…**

This section consists of additional information about the recipe in order to make you more knowledgeable about the recipe.

# **See also**

This section provides helpful links to other useful information for the recipe.

# **Get in touch**

Feedback from our readers is always welcome.

**General feedback**: If you have questions about any aspect of this book, mention the book title in the subject of your message and email us at customercare@packtpub.com.

**Errata**: Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you have found a mistake in this book, we would be grateful if you would report this to us. Please visit [www.packtpub.com/support/errata](https://www.packtpub.com/support/errata), selecting your book, clicking on the Errata Submission Form link, and entering the details.

**Piracy**: If you come across any illegal copies of our works in any form on the Internet, we would be grateful if you would provide us with the location address or website name. Please contact us at copyright@packt.com with a link to the material.

**If you are interested in becoming an author**: If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, please visit [authors.packtpub.com](http://authors.packtpub.com/).

# **Reviews**

Please leave a review. Once you have read and used this book, why not leave a review on the site that you purchased it from? Potential readers can then see and use your unbiased opinion to make purchase decisions, we at Packt can understand what you think about our products, and our authors can see your feedback on their book. Thank you!

For more information about Packt, please visit [packt.com](http://www.packt.com/).

# **Fundamentals of Embedded Systems**

Embedded systems are computer systems that combine hardware and software components to solve a specific task within a larger system or device. Unlike general-purpose computers, they are heavily specialized and optimized to perform only one task but do it really well.

They are everywhere around us, but we rarely notice them. You can find them in virtually every home appliance or gadget, such as a microwave oven, TV set, network-attached storage, or smart thermostat. Your car contains several interconnected embedded systems that handle brakes, fuel injection, and infotainment.

In this chapter, we are going to deal with the following topics on embedded systems:

* Exploring embedded systems
* Working with limited resources
* Looking at performance implications
* Working with different architectures
* Working with hardware errors
* Using C++ for embedded development
* Deploying software remotely
* Running software remotely
* Logging and diagnostics