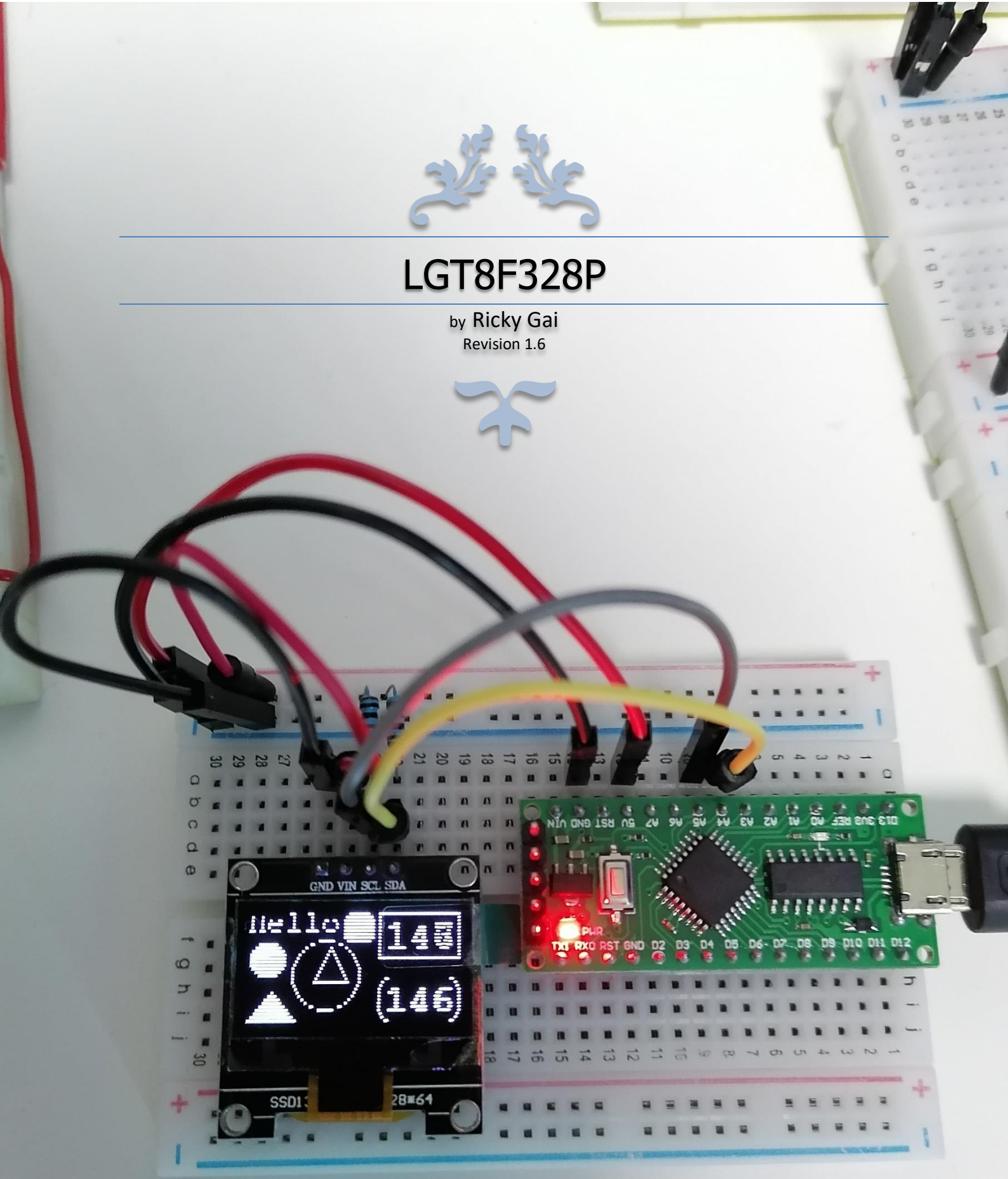
LGT8F328P

by Ricky Gai

Revision 1.6



JANUARY 18, 2021

Nexuz Innovation, Malaysia.
(MA0255412-M)

Introduction

About this Book

This is a hands-on information documented to illustrate the use of LGT8F328P chips that has been tested and run successfully on selected Arduino sketch projects using Arduino C/C++ programming language.

In the learning of Arduino platform and compatible microcontrollers, there often lack of information, incomplete online comments and the circuitry implemented may not be the same as the original circuit diagram, all these problems eventually living me to the darkness.

Therefore, I decided to write this book to document issues encountered and resolved via physical trial and error approach. It is not perfect, but with necessary guidelines to get through and move on.

To understand this book, it required some basic knowledge of electronics fundamental, PCB design and C/C++ software programming.

About the Author

Ricky Gai
is the founder and technical director of Nexuz Innovation, a small R&D IT company established in Kuala Lumpur, Malaysia.

After receiving certification from Oxford Computer Engineering discipline in 1992, my carreer was mostly exposed to C/C++ system software development for decades about 30 years, since the MS-DOS time until today's Windows environments including real-time, networking, file system, 2D/3D games, software driver, application and mobile programming.

Nevertheless, much spare time devoted to further the electronics studies for two years before coming to Arduino platform, and my wife often staring at me. Arduino programming reminded me the MS-DOS season, it brought back memory of something like interrupt, vector and bootsector (eg. Bootloader in Arduino).

All the reference materials and source code are available via Github at:
https://github.com/rickygai/arduino

For any errors found, suggestions and questions, please do email to:
support@nexuzinnovation.com

Well, passion is everything and the key to success, I hope you find something useful here.

DISCLAIMER

Abbrevation	Descriptions
NEXUZ INNOVATION / AUTHOR	refers to the author, Ricky Gai.
READER / READER(S) / READER'S	refers to the person who read or knowledge transferred, accessed the circuitry setup based on the contents illustrated in this document.
COMPONENTS / EQUIPMENTS	refers to electronics components, tools, materials that used as part of the circuitry setup.
CONTENTS	Information described within the document, including software and hardware solutions or mathods described by the author.
IP / INTELLECTUAL PROPERTY / COPYRIGHT / PERMISSION	refers to the copyrighted materials (eg. Photo, Diagram, Source Code, Links) that owned by other creators.

THIS CONTENTS OF THIS DOCUMENT IS SOLELY BASED ON THE ELECTRONICS COMPONENTS OR MATERIALS ADOPTED AND TESTED BY AND AT THE AUTHOR'S PREMISES. DUE TO VOLTAGE SUPPLY VARIES FROM DIFFERENT COUNTRIES AND PHYSICAL SPECIFICATION OF COMPONENTS MAY CHANGE FROM TIME TO TIME, READER(S) TAKE OWN RESPONSIBILITY TO ACCESS THE EXPERIMENTAL SOLUTIONS BASED ON THE INFORMATION DESCRIBED IN THIS DOCUMENT.

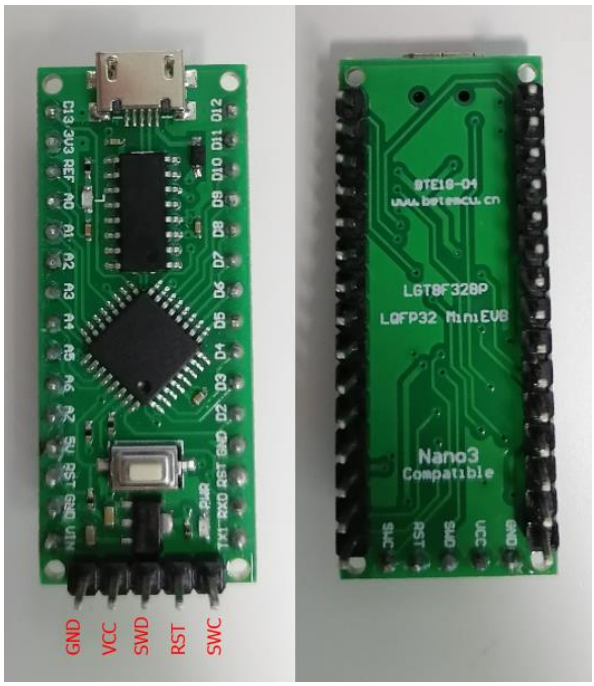
THE COMPANY NEXUZ INNOVATION AND THE AUTHOR BARES NO RESPONSIBILITY UPON ANY DAMAGE OR HARM IN CASE HAPPENS TO THE READER(S) OR READER'S SIDE, THIS INCLUDE READER'S RELATED SUCH AS HUMAN HEALTH, HARDWARE EQUIPMENTS AND OTHER LOSSES.

NEXUZ INNOVATION IS A SOLE PROPRIETORSHIP COMPANY WITH NO RELATION TO ANY HARDWARE MANUFACTURERS OR VENDORS MENTIONED IN THIS DOCUMENT, SUCH AS LOGIC GREEN, ARDUINO AND ETC. THE MENTIONED OF INTEGRATED CIRCUIT (IC) OR CHIPS PRODUCT MODELS ARE SOLELY FOR RESEARCH AND DEVELOPMENT PURPOSES ONLY.

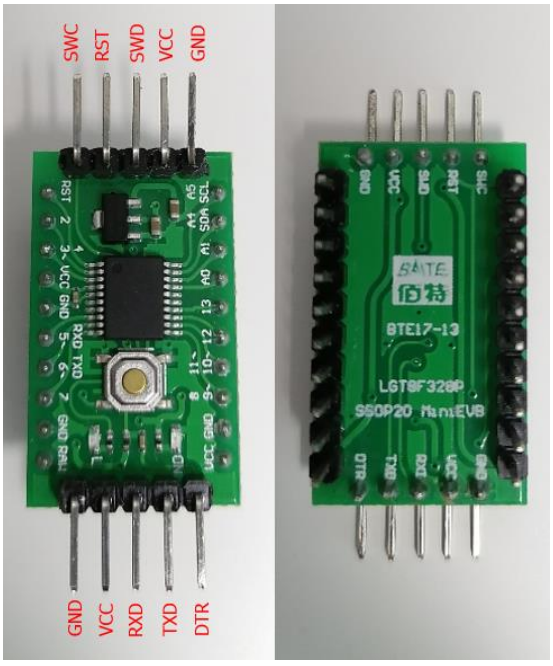
THE AUTHOR RESPECTS THE INTELLECTUAL PROPERTY FROM OTHER CREATORS, THIS DOCUMENTATION MAY SHARE SOME PARTLY EXTRACTED PORTION OF PHOTO OR DIAGRAM AS PART OF THE ILLUSTRATION USAGE. IF THERE IS ANY COPYRIGHT INFRINGEMENT, PLEASE DO INFORM THE AUTHOR TO EXCLUDE FROM THIS DOCUMENT.

Types of LGT8F328P chips

LGT8F328P LQFP32 MiniEVB

Pinout Diagram	References
	LGT8F328P is one of the Arduino clone chip created by Logic Green. Logic Green LGT8F328P datasheet (in Chinese) Logic Green official manuals (datasheets, circuit pinout diagrams) Ralph Bacon LGT8F328P-Arduino-Clone-Chip-ATMega328P (with English version of LGT8F328P datasheet)

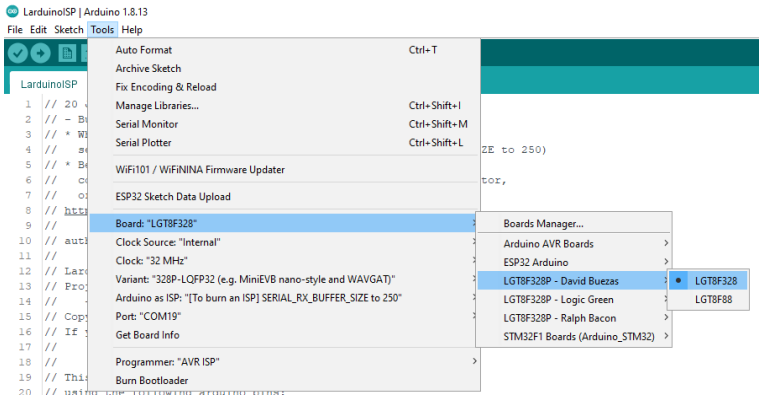
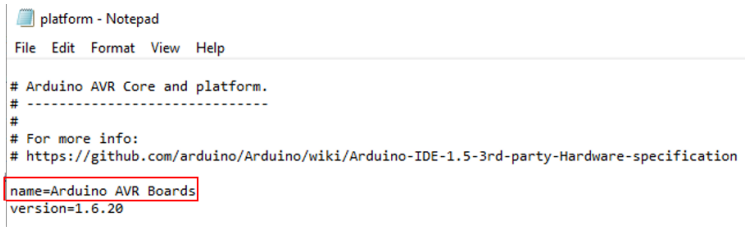
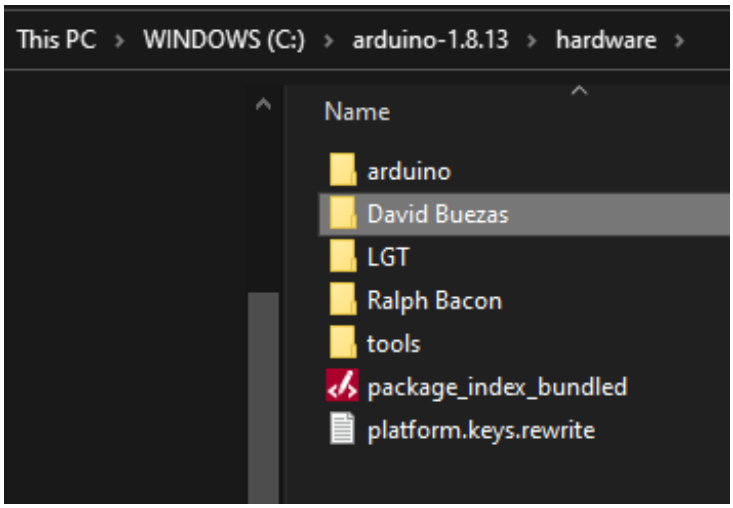
LGT8F328P SSOP20 MiniEVB

Pinout Diagram	References
	Same as above.

Prerequisition

Arduino IDE - Setup the LGT8F328P library

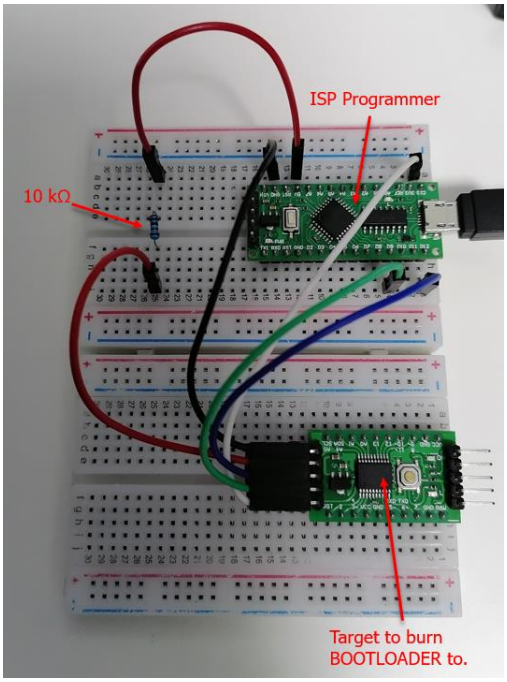
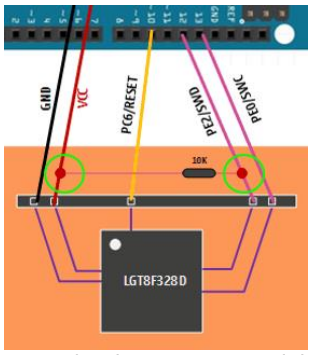
Download, install and configure the LGT8F328P libraries

Diagrams	Descriptions & References
It is very important to configure the installed LGT8F328P libraries as to appear under the Arduino IDE menu outstandingly such as below:  Figure 1: Multiple LGT8F328P libraries installed with orgainized  Figure 2: The default attribute name="Arduino AVR Boards"  Figure 3: The default attribute name="Arduino AVR Boards"	There are many versions of LGT8F328P libraries built by different authors. Here, I choose to install three board libraries from: Logic Green Ralph Bacon David Buezas To have a proper names listed as shown on **Figure 1**, you have to edit the file **platform.txt** from each of the author's library path installed, otherwise the listed items under "Board Manager" will be confusing such as below: Arduino AVR Boards Arduino AVR Boards Logic Green Arduino AVR Compatible Boards Depends of which version of Arduino IDE you are using, mine is 1.8.13 and it duplicated the default name "Arduino AVR Boards" because the platform.txt bundled under Logic Green and Ralph Bacon package file used the same default attribute **"name=Arduino AVR Boards"** as shown on **Figure 2**. To solve this problem, **I edited three of the platform.txt files** and renamed the attribute "name=" of each file respectively to something like below: name=LGT8F328P - Logic Green name=LGT8F328P - Ralph Bacon name=LGT8F328P - David Buezas To be more precisely, I renamed the library folder for Ralph Bacon and David Buezas as shown on **Figure 3**. In my case, these three platform.txt files are located under: C:\arduino-1.8.13\hardware\LGT\avr\platform.txt C:\arduino-1.8.13\hardware\Ralph Bacon\avr\platform.txt C:\arduino-1.8.13\hardware\David Buezas\avr\platform.txt The path locations above may be different, subjected to individuals' Arduino IDE installation.

Arduino Sketch Projects

LarduinoISP - How to burn the BOOTLOADER ?

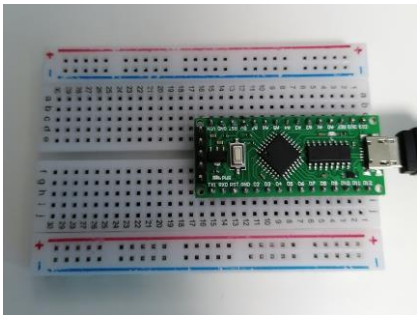
Introduction

Diagrams	Descriptions & References
 Figure 4: LarduinoISP working circuitry setup	LarduinoISP is the use of Arduino ATMega328P (eg. NANO, UNO) or LGT8F328P (LQFP32 MiniEVB) as **ISP Programmer** to burn BOOTLOADER to the Target LGT8F328P SSOP20 MiniEVB as shown at **Figure 4**. Here, I want to share some painful experiences that went through in dealing with the LarduinoISP circuitry setup. I started by referring to LGTMCU/LarduinoISP, at that time I was doubted about the short-circuit of the 10kΩ resistor between VCC and SWD pin as shown at **Figure 5**, with two green highlighted circles below:  Figure 5: LGT/LarduinoISP extracted diagram Really wasted much time on it, no one from the Arduino Forum can gives me at least a snapshot photo of how to setup the physical circuitry correctly. Based on **Figure 4**, I decided to remove the SWD (blue wire) from connecting to the 10kΩ pull-up resistor and connect directly to pin 12 of the ISP Programmer. Next, I used David Buezas's LarduinoISP sketch to upload to the ISP Programmer because his sketch is more recently updated. Somehow, still failed to upload the sketch (may be that time Arduino Nano was used as ISP Programmer). Later, a site called SuperUserNameMan/LGTISP, after reading its **usage** section, I begin to realize the whole story and quickly edit the **HardwareSerial.h** to to change the following instruction from: #define SERIAL_RX_BUFFER_SIZE 64 to: #define SERIAL_RX_BUFFER_SIZE 250 Eventually, LarduinoISP successfully burned the BOOTLOADER to the Target LGT8F328P SSOP20 MiniEVB. The entire circuitry is very straight forward actually but took me sometime to accomplish. We will cover more details about the hands-on process on next page.

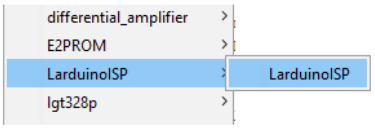
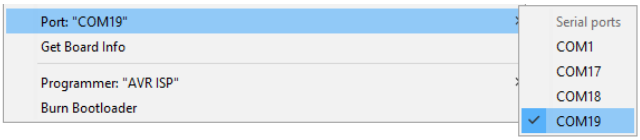
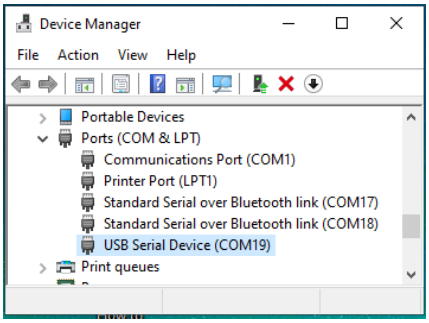
The Parts list

Components	Quantity
LGT8F328P LQFP32 MiniEVB	1
LGT8F328P SSOP20 MiniEVB	1
FTDI FT232RL	1
RESISTOR 10kΩ	1
10 cm Female-to-Male Jumper Wire	5
10 cm Female-to-Female Jumper Wire	5
10 cm Male-to-Male Jumper Wire	1
USB 2.0 Micro B cable connector	1
USB 2.0 Mini B cable connector	1
MB102 Mini Breadboard 8.5CM x 5.5CM 400 Holes	2

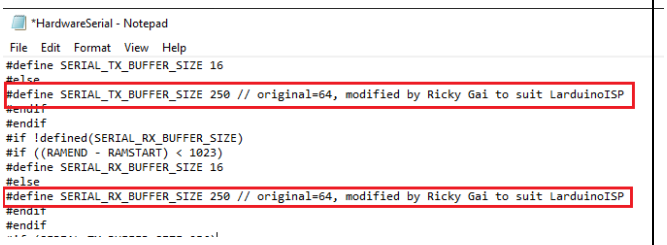
Installing the LGT8F328P board library

Illustrations	Descriptions
1. First, you have to complete the Download, install and configure the LGT8F328P libraries stage. For illustration purposes, we adopted David Buezas LGT8FX board library. 2. After you have completed step 1, put the LGT8F328P LQFP32 MiniEVB on the breadboard then connect the USB Micro B cable as shown on **Figure 6** to the PC.	 Figure 6: LGT8F328P LQFP32 MiniEVB on empty breadboard

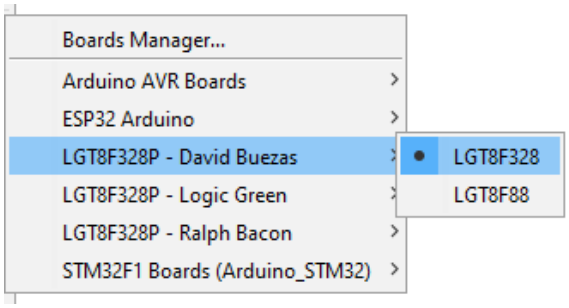
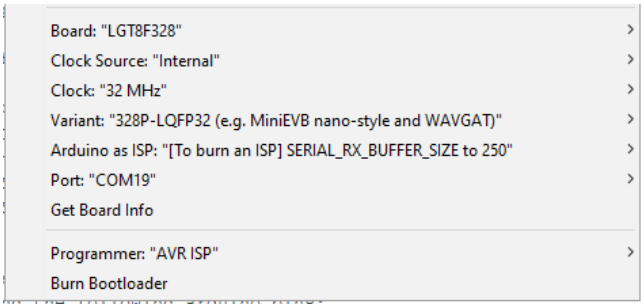
Opening the LarduinoISP sketch

Illustrations	Descriptions
3. At the Arduino IDE, File → Examples → LarduinoISP → LarduinoISP, click on it to open such as **Figure 7**.	 Figure 7: Opening the LarduinoISP sketch
4. Next, turn ON the LGT8F328P LQFP32 MiniEVB, the respective serial port should be detected, in this case it is COM19 under the Windows Device Manager at **Figure 9**. 5. Goto the File menu, Tools → Port: → select COM19 as shown below:  Figure 8: Selecting the LGT8F328P LQFP32 Serial Port	 Figure 9: LGT8F328P LQFP32 Serial Port under Device Manager

Updating HardwareSerial.H

Illustrations	Descriptions and Fundamental Units
6. Open the file HardwareSerial.H using Notepad: C:/arduino-1.8.13/hardware/arduino/avr/cores/arduino/HardwareSerial.h 7. Change "#define SERIAL_RX_BUFFER_SIZE 64" to "#define SERIAL_RX_BUFFER_SIZE 250" as shown on **Figure 10**. 8. Save the file.	 Figure 10: HardwareSerial.H updates

Uploading LarduinoISP sketch to ISP Programmer (LGT8F328P LGFP32 MiniEVB)

Illustrations	Descriptions and Fundamental Units
9. Goto File menu, Tools → Board: → LGT8F328P - David Buezas → select LGT8F328 as shown on **Figure 11**.	 Figure 11: Selecting David Buezas LGT8F328 board library
10. Make sure under the Tools menu options, all settings are based on **Figure 12**. 11. Upload the sketch by clicking  button.	 Figure 12: David Buezas LGT8F328 settings

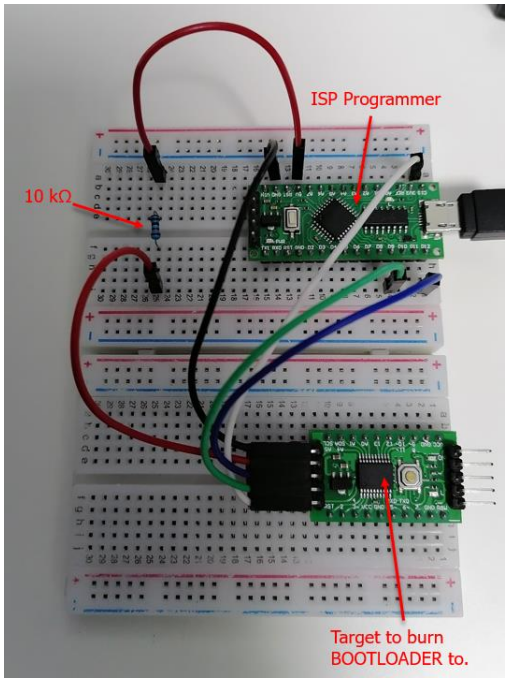
The success messages of LarduinoISP sketch uploaded to ISP Programmer.

Descriptions
12. If the LarduinoISP sketch is successfully uploaded to ISP Programmer (LGT8F328P LQFP32 MiniEVB), you should be seeing messages below:

```
avrdude: AVR device initialized and ready to accept instructions

Reading | ################################################## | 100% 0.00s

avrdude: Device signature = 0x1e950f (probably m328p)
avrdude: reading input file "C:\Users\RICKYG~1\AppData\Local\Temp\arduino_build_659691/LarduinoISP.ino.hex"
avrdude: writing flash (5420 bytes):

Writing | ################################################## | 100% 1.34s

avrdude: 5420 bytes of flash written
avrdude: verifying flash memory against C:\Users\RICKYG~1\AppData\Local\Temp\arduino_build_659691/LarduinoISP.ino.hex:
avrdude: load data flash data from input file C:\Users\RICKYG~1\AppData\Local\Temp\arduino_build_659691/LarduinoISP.ino.hex:
avrdude: input file C:\Users\RICKYG~1\AppData\Local\Temp\arduino_build_659691/LarduinoISP.ino.hex contains 5420 bytes
avrdude: reading on-chip flash data:

Reading | ################################################## | 100% 1.05s

avrdude: verifying ...
avrdude: 5420 bytes of flash verified

avrdude done.  Thank you.
```

Figure 13: The success messages of LarduinoISP sketch uploaded to ISP Programmer

Wiring up the LarduinoISP circuitry

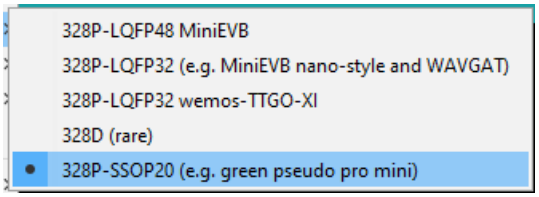
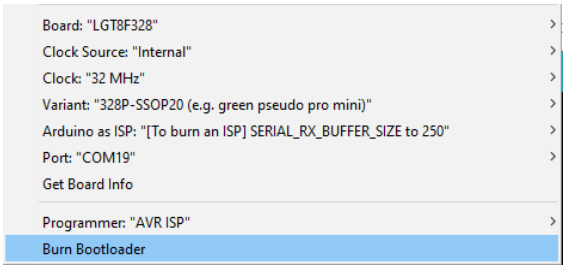
Illustrations	Descriptions
13. Now, the ISP Programmer is ready. 14. Turn off the ISP Programmer. 15. We are going to wire up the LarduinoISP circuitry just like **Figure 15**. 16. LarduinoISP wire connection table:	

LGT8F32P SSOP20 (Target)	LGT8F328 LQFP32 (ISP Programmer)
GND	GND
VCC	5V
SWD	D12
RST	D10
SWC	D13

Figure 14: LarduinoISP wire connection

Figure 15: LarduinoISP circuitry setup

To burn BOOTLOADER to Target LGT8F328P SSOP20 MiniEVB

Illustrations	Descriptions
17. We are now going to burn the BOOTLOADER to the Target LGT8F328P SSOP20 MiniEVB. 18. Turn on the ISP Programmer. 19. At the File menu, Tools → Variant: → select "328P-SSOP20 (e.g. green pseudo pro mini)" as shown on **Figure 16**. 20. Ignore the "Arduino as ISP", select Programmer as "AVR ISP". 21. Click on the "Burn Bootloader".	 Figure 16: Change the Variant to Target LGT8F328P SSOP20  Figure 17: To Burn Bootloader

The success messages of burning the BOOTLOADER to the Target.

Descriptions
22. If the BOOTLOADER is successful, you will see the message below:

```
avrdude: AVR device initialized and ready to accept instructions

Reading | ################################################## | 100% 0.00s

avrdude: Device signature = 0x1e950f (probably m328p)
avrdude: NOTE: "flash" memory has been specified, an erase cycle will be performed
         To disable this feature, specify the -D option.
avrdude: erasing chip
avrdude: reading input file "C:\arduino-1.8.13\hardware\David Buezas\avr/bootloaders/lgt8fx8ps20/optiboot_lgt8f328ps20.hex"
avrdude: writing flash (30720 bytes):

Writing | ################################################## | 100% 0.25s

avrdude: 30720 bytes of flash written
avrdude: verifying flash memory against C:\arduino-1.8.13\hardware\David Buezas\avr/bootloaders/lgt8fx8ps20/optiboot_lgt8f328ps20.hex:
avrdude: load data flash data from input file C:\arduino-1.8.13\hardware\David Buezas\avr/bootloaders/lgt8fx8ps20/optiboot_lgt8f328ps20.hex:
avrdude: input file C:\arduino-1.8.13\hardware\David Buezas\avr/bootloaders/lgt8fx8ps20/optiboot_lgt8f328ps20.hex contains 30720 bytes
avrdude: reading on-chip flash data:

Reading | ################################################## | 100% 0.02s

avrdude: verifying ...
avrdude: 30720 bytes of flash verified
avrdude: reading input file "0x3f"
avrdude: writing lock (1 bytes):

Writing | ################################################## | 100% 0.01s

avrdude: 1 bytes of lock written
avrdude: verifying lock memory against 0x3f:
avrdude: load data lock data from input file 0x3f:
avrdude: input file 0x3f contains 1 bytes
avrdude: reading on-chip lock data:

Reading | ################################################## | 100% 0.00s

avrdude: verifying ...
avrdude: 1 bytes of lock verified

avrdude done.  Thank you.
```

Figure 18: The success messages of burning BOOTLOADER to Target LGT8F328P SSOP20 MiniEVB

Testing the LGT8F328P SSOP20 MiniEVB with new BOOTLOADER burned.

Illustrations	Descriptions and Fundamental Units

23. Congratulations, the Target LGT8F328P SSOP20 MiniEVB is burned with new BOOTLOADER image.

24. Here, we are going to do a test on it by uploading a fast blinking sketch to see the significant changing effects. If it is uploaded successfully, the LGT8F328P SSOP20 MiniEVB should be blinking faster at 64 ms interval.

25. Turn off the ISP Programmer, connect the LGT8F328P SSOP20 MiniEVB to FTDI FT232RL as shown on **Figure 20**.

26. The FTDI FT232RL connection, **its CTS pin is not required here**:

LGT8F32P SSOP20	FTDI FT232RL
GND	GND
VCC	5V
RXD	TXD
TXD	RXD
DTR	DTR

Figure 19: The standalone LGT8F328P SSOP20 MiniEVB pin connection with FTDI FT232RL

27. Turn on the FTDI FT232RL that will power up LGT8F328P SSOP20 MiniEVB.

28. NOTE: This will create a new serial port COM22 interface as shown on **Figure 21**.

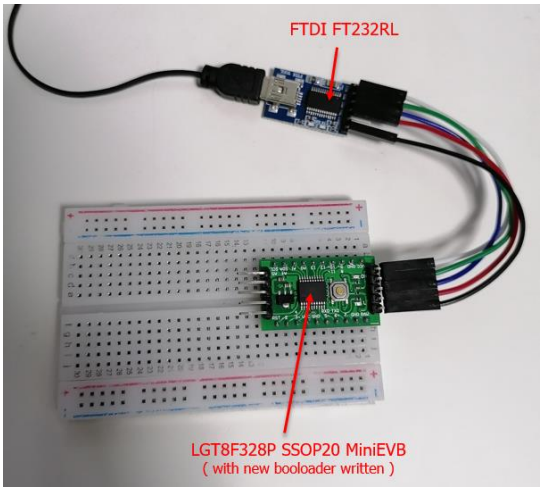


Figure 20: The standalone LGT8F328P SSOP20 MiniEVB with FTDI FT232RL interface

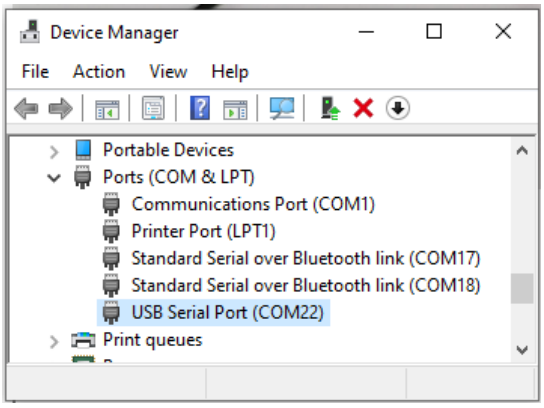


Figure 21: FTDI FT232RL new serial port interface

29. Take a look at the LGT8F328P SSOP20 MiniEVB by default itself, it is blinking at internal roughly two seconds as shown on **Figure 22**.

    To see the different, later when the new sketch blink64ms.ino is uploaded to it, the blue LED should be blinking faster at 64 ms interval.

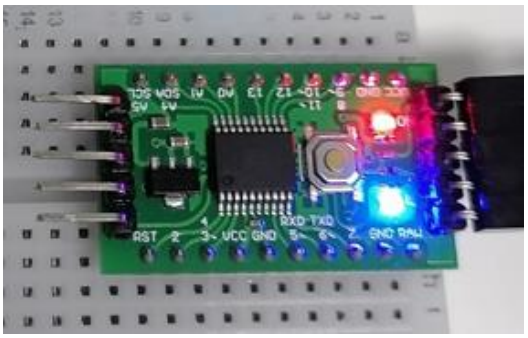


Figure 22: FTDI FT232RL default blinking (blue light) two seconds interval

Uploading blink64ms.ino sketch to LGT8F328P SSOP20 MiniEVB via FT232RL

Descriptions

30. Here, we are to upload a sketch called blink64ms.ino to the LGT8F328P SSOP20 MiniEVB since the Bootloader already written, if it is working normally after a reset, the Bootloader should starts and jumps to the program address where the blink64ms.ino program will be loaded and run.

31. Select the correct Port COM22 (FT232RL) from the menu as shown below:

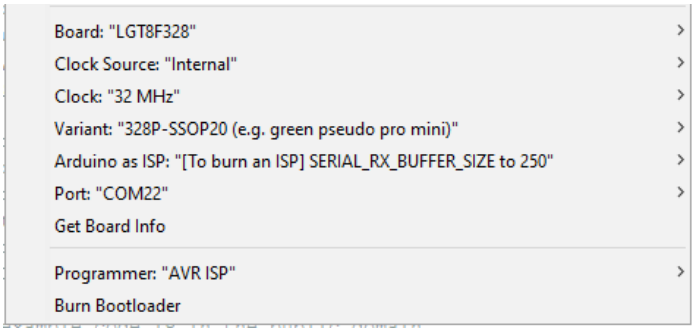


Figure 23: Selecting the FTDI FT232RL serial port COM22

32. At Arduino IDE, open the blink64ms.ino sketch then click the  button to upload sketch.

33. If it is uploaded successfully you will notice the blue LED will blink faster continuously.

The success messages of uploading blink64ms.ino to LGT8F328 SSOP20 MiniEVB

Descriptions and Fundamental Units

34. The success messages after blink64ms.ino sketch uploaded to LGT8F328P SSOP20 MiniEVB will look like below:

```
avrdude: AVR device initialized and ready to accept instructions

Reading | ################################################## | 100% 0.00s

avrdude: Device signature = 0x1e950f (probably m328p)
avrdude: reading input file "C:\Users\RICKYG~1\AppData\Local\Temp\arduino_build_82099/blink64ms.ino.hex"
avrdude: writing flash (1104 bytes):

Writing | ################################################## | 100% 0.48s

avrdude: 1104 bytes of flash written
avrdude: verifying flash memory against C:\Users\RICKYG~1\AppData\Local\Temp\arduino_build_82099/blink64ms.ino.hex:
avrdude: load data flash data from input file C:\Users\RICKYG~1\AppData\Local\Temp\arduino_build_82099/blink64ms.ino.hex:
avrdude: input file C:\Users\RICKYG~1\AppData\Local\Temp\arduino_build_82099/blink64ms.ino.hex contains 1104 bytes
avrdude: reading on-chip flash data:

Reading | ################################################## | 100% 0.47s

avrdude: verifying ...
avrdude: 1104 bytes of flash verified

avrdude done.  Thank you.
```

Figure 24: The success messages of uploading blink64ms.ino sketch to LGT8F328P SSOP20 MiniEVB

35. DONE.

