KYLE GOFF, BINSEN QIAN, & HARRY CHENG

Binsen Qian, PhD candidate in mechanical engineering; Kyle Goff, undergraduate computer engineering student; and Harry Cheng, professor and director of the UC Davis C-STEM Center.
magpi.cc/2xEGEa9

USE A GUI TO CONTROL GPIO PINS

Discover a simple way to interact with GPIO pins using C-STEM

You'll Need

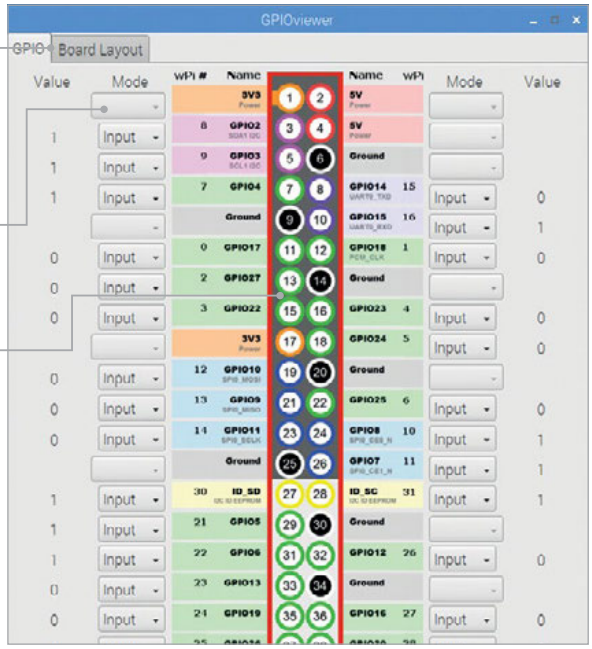
- Raspberry Pi 3
- C-STEMbian **magpi.cc/2p3JUNP**
- Breadboard
- Jumper wires
- 1 × LED
- 1 × 220Ω resistor (Red-Red-Brown)

TROUBLE-SHOOTING HELP

Remember to use GPIOviewer for testing the pins before programming. If this fails, check the LED is plugged in properly and the wires for all components are in the correct spots.

One of the easiest ways to get started on a Raspberry Pi is with C-STEM Studio on the C-STEMbian operating system. It removes many of the hassles associated with programming in C by using C/C++ interpreter Ch. Ch is superset of C with many high-level extensions. It can run C code without compilation. This article will demonstrate how to get started controlling GPIO pins on the Pi using GPIOviewer and WiringPi with a project in Ch and highlight some key features of the system.

To make use of these programming tools, you should install the C-STEMbian operating system (**magpi.cc/2p3JUNP**) which contains C-STEM Studio. This free, open-source operating system contains all the necessary tools for robotics and physical computing. Additionally, it is a superset of Raspbian, so all the familiar features will still be there. If you already have Raspbian installed, the C-STEM modules can be installed separately on top. Step-by-step guides will assist you in setting up and accessing the Raspberry Pi if needed.



To better understand the layout of the board, click on the Board Layout tab

Change the mode of a specific pin by selecting input, output, or PWM

The name of each pin can be found in the centre of GPIOviewer

While this article will focus on ChIDE, GPIOviewer, and the WiringPi package, there are many additional features included in C-STEMbian:

- **C-STEM Studio (magpi.cc/2xEGEa9), a platform for hands-on integrated learning of computing, science, technology, engineering, and mathematics (C-STEM).**
- **Ch Mindstorms Controller for controlling multiple LEGO MINDSTORMS (EV3 or NXT) from the Raspberry Pi using a simple user interface**
- **Linkbot Labs which, like the MINDSTORMS equivalent, gives a user interface for total control of one or more Linkbots.**
- **Ch Arduino for controlling and programming in Ch an Arduino Uno's pins from the Raspberry Pi.**

Make a circuit

The program in this tutorial would be useless without a circuit to test it on. If you have one, use a breakout board to make the wiring process clearer. Otherwise, wire the pins directly from the Pi. Take a wire from the GPIO 4 pin and connect it to an empty row of the breadboard. Then, attach the positive lead of an LED to this row. From the negative lead of the LED, attach a 220Ω (Red-Red-Brown) resistor to ground.

Before programming, we can use GPIOviewer, a helpful feature of the C-STEMbian operating system. To use it, navigate to the big 'C' on the top of the desktop window.

Once open, navigate to 'Ch Raspberry Pi' and click Launch in the bottom right-hand corner. This will open up GPIOviewer, which allows total control of all GPIO pins of the Raspberry Pi. In this view, you can change pin modes between input, output, and PWM (with a slider). For this circuit, find GPIO pin 4 on GPIOviewer and set it to output.

Ensure the circuit is set up and working properly by switching between high and low outputs. If the light turns on, you are ready to program.

Example blink code in C

The classic way of programming this circuit on the Raspberry Pi would resemble the C code in **blink.c**, which includes the wiringPi library. This code also works in Ch interpretively without compilation.

Language
>C

NAME:
blink.c
DOWNLOAD:
magpi.cc/BlinkPi

```
/* File: blink.c */
#include <wiringPi.h>

int main() {
   wiringPiSetupGpio();
   pinMode(4, OUTPUT);
   while(1) {
      digitalWrite(4, HIGH);
      delay(500);
      digitalWrite(4, LOW);
      delay(500);
   }
   return 0 ;
}
```

The code uses GPIO pin numbering and sets pin 4 to an output. Then, it enters a while loop that cycles the LED every half second between on and off. Notice the standard **int main** function that requires a '**return 0** statement.

Equivalent Code in Ch

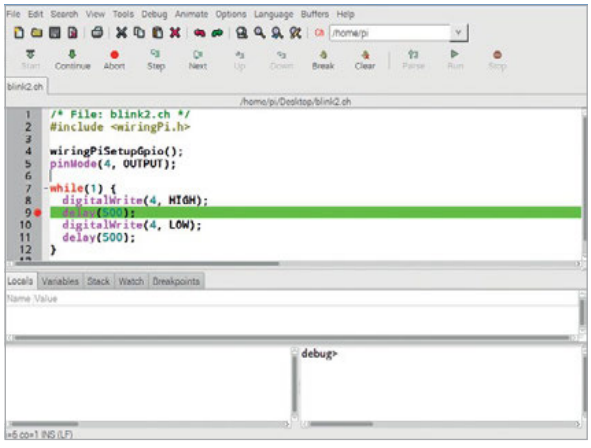
However, to run the code just in Ch, the program can be simplified to the code in **blink2.ch**.

Language
>Ch

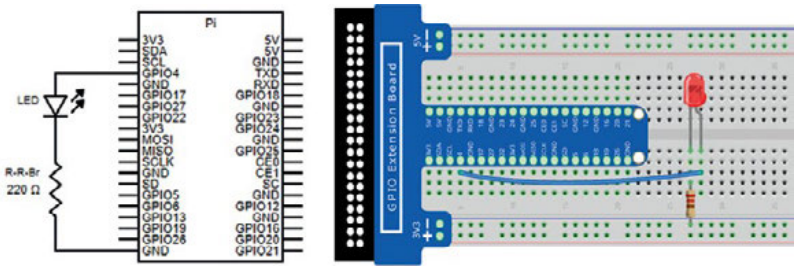
NAME:
blink2.ch
DOWNLOAD:
magpi.cc/BlinkPi

```
/* File: blink2.ch */
#include <wiringPi.h>

wiringPiSetupGpio();
pinMode(4, OUTPUT);
while(1) {
    digitalWrite(4, HIGH);
    delay(500);
    digitalWrite(4, LOW);
    delay(500);
}
```



Above ChIDE provides easy-to-use debugging tools to step through Ch code



Above Set the circuit up as shown in the diagram, with a breakout board or with wiring directly from the Pi

Notice that the **int main** function is gone. Just like Python and other scripting languages, there is no need to create any functions for a simple program. This is just one of the many benefits of using Ch. While this method of programming is most common for users coming from C programming, there is another way.

Scripting in Ch

Besides writing code resembling C, it is possible to write scripting code in Ch. The same program as before is rewritten in **blink3.ch**.

Language
>Ch

NAME:
blink3.ch
DOWNLOAD:
magpi.cc/BlinkPi

```
#!/bin/ch
/* File: blink3.ch*/

gpio -g mode 4 out
while(1) {
  gpio -g write 4 1
  delay(500);
  gpio -g write 4 0
  delay(500);
}
```

Ch provides both methods as an option so that you can choose whichever is more comfortable for your style. To run the Ch code, it is easiest to use ChIDE. To launch it, click the magnifying glass icon next to the big 'C' you clicked to open C-STEM Studio. Alternatively, navigate to 'Programming with Ch' within C-STEM Studio's menu. Type the code into the code editing pane. To run it, click the Run button at the top of the interface above the code. If all goes well, the LED should blink on and off indefinitely.

Debugging

Another useful feature of Ch is the ChIDE debugging feature. Like any popular IDE, the ChIDE provides easy to use debugging tools to step through Ch code. To step through the code, use the debug control panel at the top of the window. You can step into functions or through code with the Step and Next buttons, and you are able to set break points by clicking in the grey line number column between the numbers and text. For example, set a break point by clicking in the grey area to the right of line 9. Then, click the Continue button. Every time you click this button, it will run the program until it hits another breakpoint. Since this is a while loop, it will hit this break point every time.

FOR HELP AND NEW IDEAS

Open the 'Learn Physical Computing with Raspberry Pi' textbook in the Ch Raspberry Pi section of C-STEM Studio for more circuits and programs.