

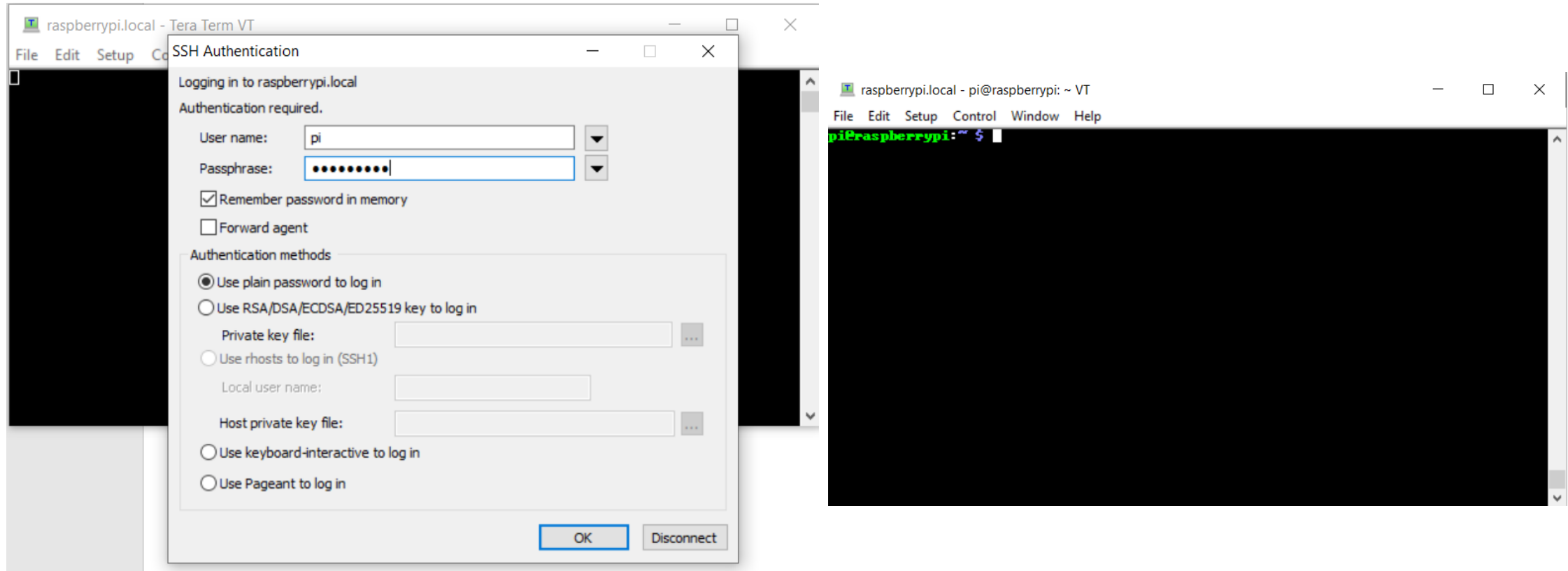
UCSD Embedded Linux Assignment 4

By

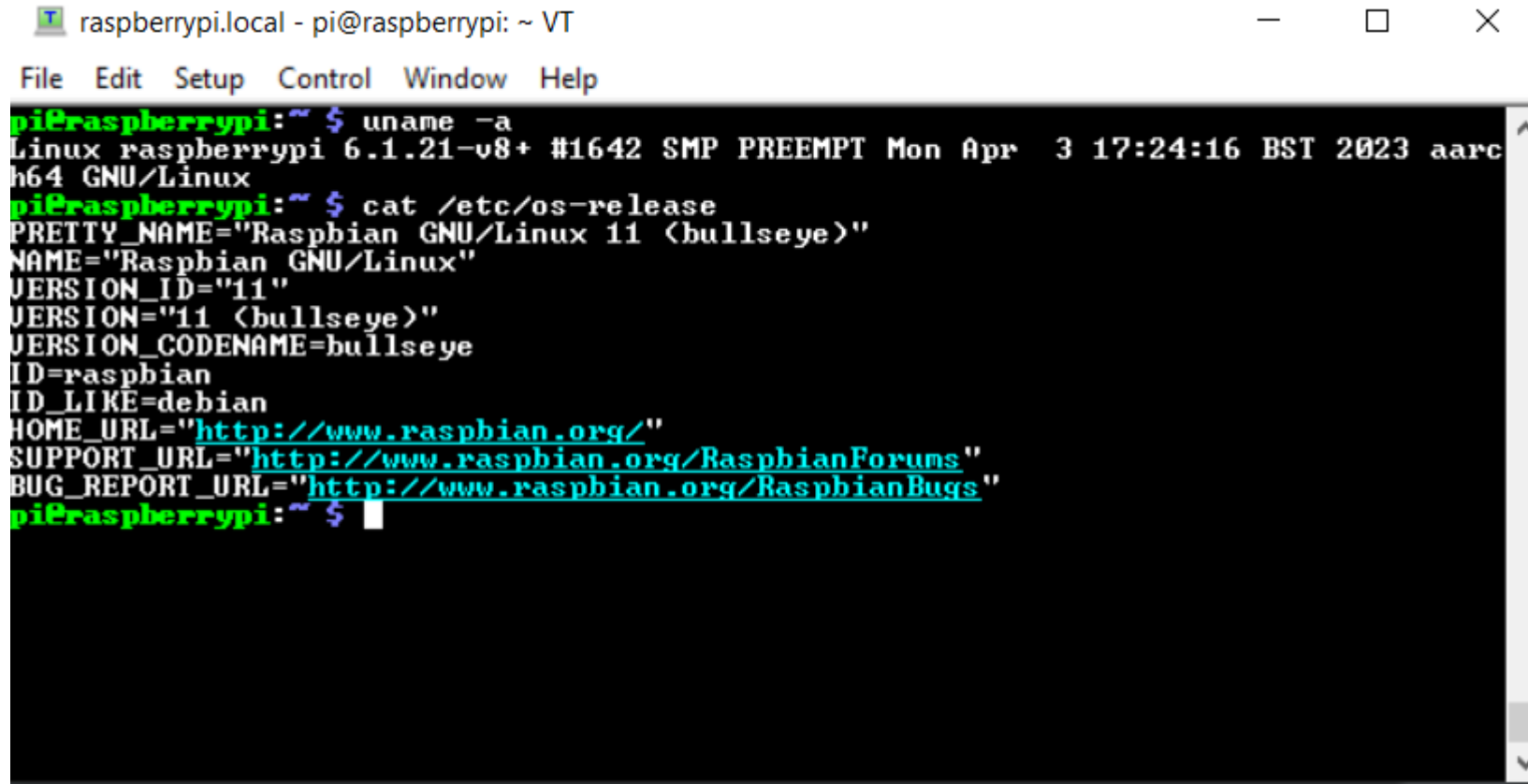
Hsuankai Chang

hsuankac@umich.edu

Step 1. Demo 1. RPi GPIO Access From C. Logon to RPi 4

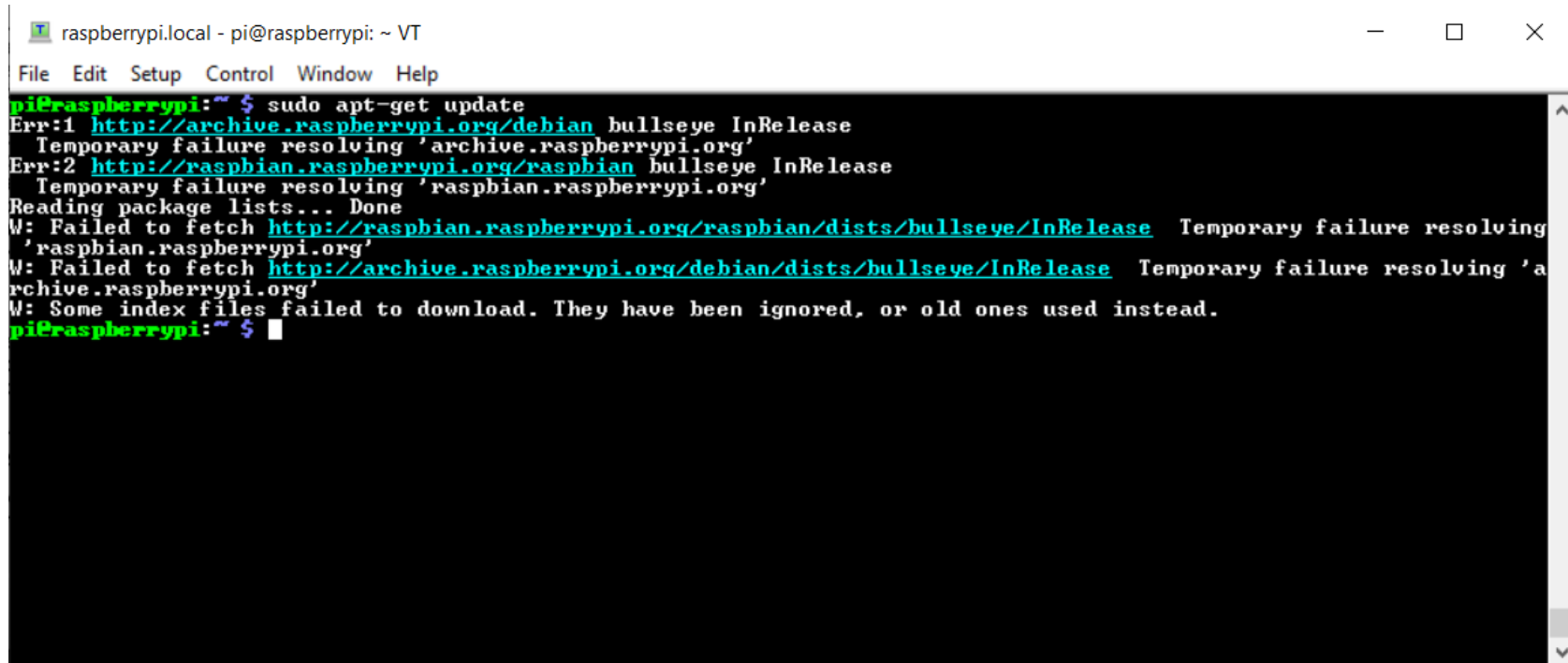


Step 2. RPi 4 Baseline

A terminal window titled 'raspberrypi.local - pi@raspberrypi: ~ VT' with standard window controls. The terminal shows the output of 'uname -a' and 'cat /etc/os-release'. The 'uname -a' output shows 'Linux raspberrypi 6.1.21-v8+ #1642 SMP PREEMPT Mon Apr 3 17:24:16 BST 2023 aarch64 GNU/Linux'. The 'cat /etc/os-release' output shows 'PRETTY_NAME="Raspbian GNU/Linux 11 (bullseye)"', 'NAME="Raspbian GNU/Linux"', 'VERSION_ID="11"', 'VERSION="11 (bullseye)"', 'VERSION_CODENAME=bullseye', 'ID=raspbian', 'ID_LIKE=debian', 'HOME_URL="http://www.raspbian.org/"', 'SUPPORT_URL="http://www.raspbian.org/RaspbianForums"', and 'BUG_REPORT_URL="http://www.raspbian.org/RaspbianBugs"'. The prompt 'pi@raspberrypi:~\$' is visible at the bottom.

```
pi@raspberrypi:~$ uname -a
Linux raspberrypi 6.1.21-v8+ #1642 SMP PREEMPT Mon Apr 3 17:24:16 BST 2023 aarch64 GNU/Linux
pi@raspberrypi:~$ cat /etc/os-release
PRETTY_NAME="Raspbian GNU/Linux 11 (bullseye)"
NAME="Raspbian GNU/Linux"
VERSION_ID="11"
VERSION="11 (bullseye)"
VERSION_CODENAME=bullseye
ID=raspbian
ID_LIKE=debian
HOME_URL="http://www.raspbian.org/"
SUPPORT_URL="http://www.raspbian.org/RaspbianForums"
BUG_REPORT_URL="http://www.raspbian.org/RaspbianBugs"
pi@raspberrypi:~$
```

Step 3. sudo apt-get update



```
raspberrypi.local - pi@raspberrypi: ~ VT
File Edit Setup Control Window Help
pi@raspberrypi:~$ sudo apt-get update
Err:1 http://archive.raspberrypi.org/debian bullseye InRelease
Temporary failure resolving 'archive.raspberrypi.org'
Err:2 http://raspbian.raspberrypi.org/raspbian bullseye InRelease
Temporary failure resolving 'raspbian.raspberrypi.org'
Reading package lists... Done
W: Failed to fetch http://raspbian.raspberrypi.org/raspbian/dists/bullseye/InRelease Temporary failure resolving
'raspbian.raspberrypi.org'
W: Failed to fetch http://archive.raspberrypi.org/debian/dists/bullseye/InRelease Temporary failure resolving 'a
rchive.raspberrypi.org'
W: Some index files failed to download. They have been ignored, or old ones used instead.
pi@raspberrypi:~$
```

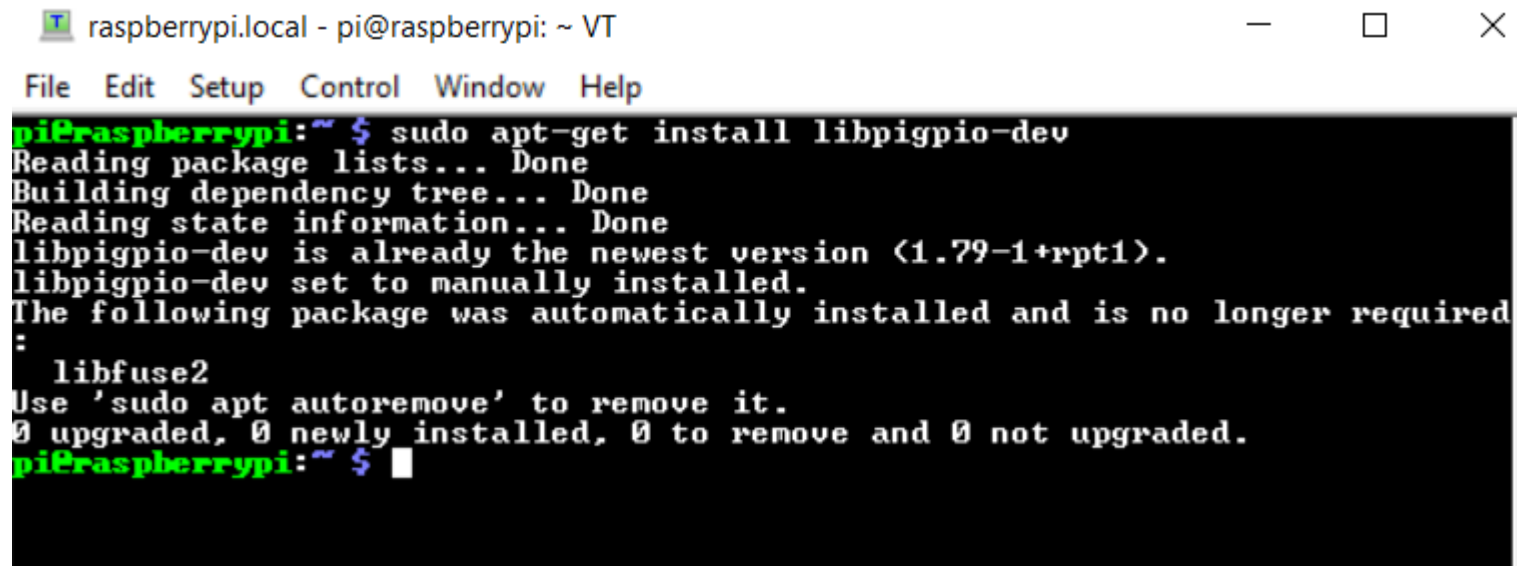
Step 4. sudo apt-cache search gpio

```
raspberrypi.local - pi@raspberrypi: ~ VT
File Edit Setup Control Window Help
pi@raspberrypi:~$ sudo apt-cache search gpio
airspyhf - HF+UHF software defined radio receiver - user runtime
gpio - Tools for interacting with Linux GPIO character device - binary
ledmon - Enclosure LED Utilities
libgpio-dev - C library for interacting with Linux GPIO device - static libraries and headers
libgpio-doc - C library for interacting with Linux GPIO device - library documentation
libgpio2 - C library for interacting with Linux GPIO device - shared libraries
libpigpio-dev - Client tools for Raspberry Pi GPIO control
libpigpio1 - Library for Raspberry Pi GPIO control
libpigpio-dev - Development headers for client libraries for Raspberry Pi GPIO control
libpigpio-if1 - Client library for Raspberry Pi GPIO control (deprecated)
libpigpio-if2-1 - Client library for Raspberry Pi GPIO control
pigpio - Raspberry Pi GPIO control transitional package.
pigpio-tools - Client tools for Raspberry Pi GPIO control
pigpiod - Client tools for Raspberry Pi GPIO control
python-periphery-doc - Peripheral I/O (Documentation)
python3-libgpio - Python bindings for libgpio (Python 3)
python3-periphery - Peripheral I/O (Python3 version)
python3-pigpio - Python module which talks to the pigpio daemon (Python 3)
python3-rpi.gpio - Module to control Raspberry Pi GPIO channels (Python 3)
rpi.gpio-common - Module to control Raspberry Pi GPIO channels (common files)
stm32flash - STM32 chip flashing utility using a serial bootloader
suxlink-gpio - GPIO control scripts SuxLink amateur radio server
python-gpiozero - Simple API for controlling devices attached to a Pi's GPIO pins.
python-gpiozero-doc - Simple API for controlling devices attached to a Pi's GPIO pins.
python-pigpio - Python module which talks to the pigpio daemon (Python 2)
python3-gpiozero - Simple API for controlling devices attached to a Pi's GPIO pins.
raspi-gpio - Dump the state of the BCM270x GPIOs
raspi-gpio-dbg - debug symbols for raspi-gpio
pi@raspberrypi:~$
```

Step 5. dpkg -L libpigpio-dev

```
raspberrypi.local - pi@raspberrypi: ~ VT
File Edit Setup Control Window Help
pi@raspberrypi:~$ dpkg -L libpigpio-dev
./
/usr
/usr/include
/usr/include/pigpio.h
/usr/lib
/usr/share
/usr/share/doc
/usr/share/doc/libpigpio-dev
/usr/share/doc/libpigpio-dev/changelog.Debian.gz
/usr/share/doc/libpigpio-dev/copyright
/usr/share/man
/usr/share/man/man3
/usr/share/man/man3/pigpio.3.gz
/usr/lib/libpigpio.so
pi@raspberrypi:~$
```

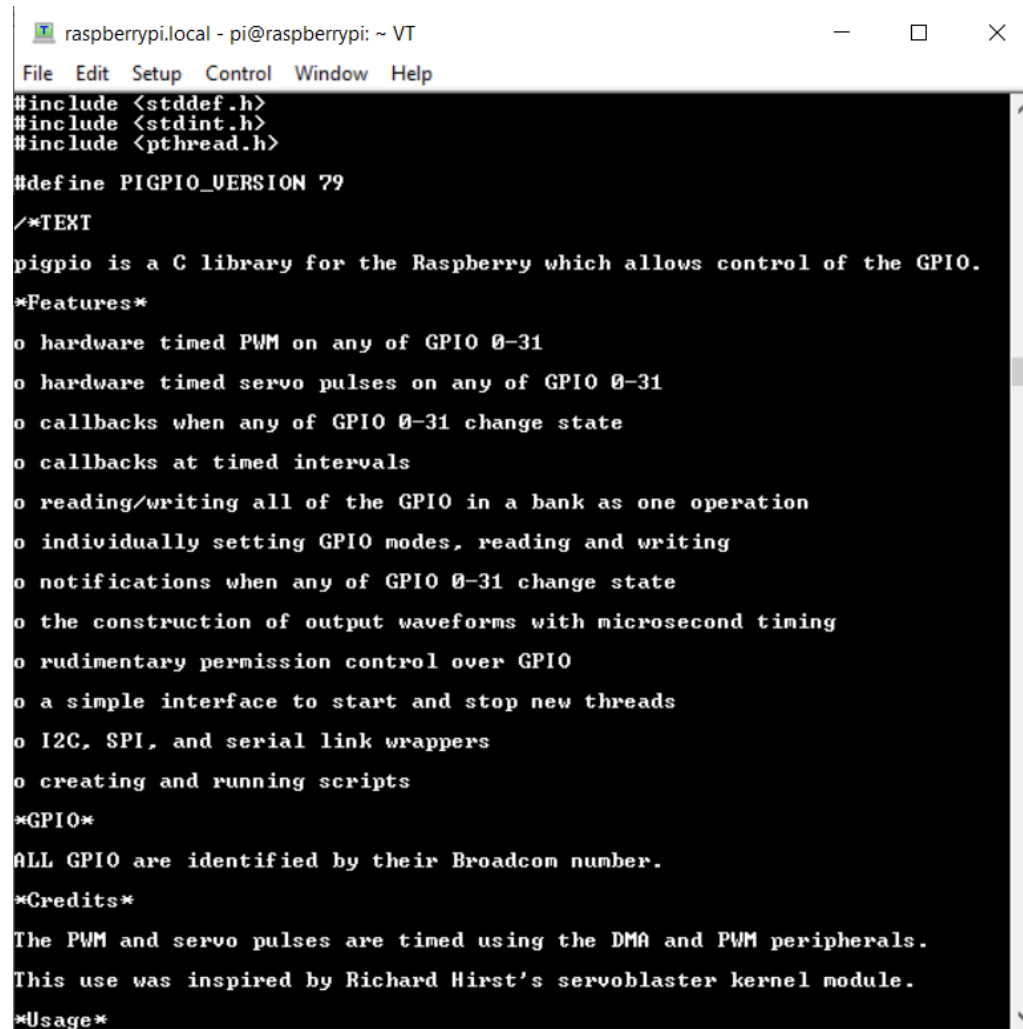
Step 6. sudo apt-get install libpigpio-dev



A terminal window titled "raspberrypi.local - pi@raspberrypi: ~ VT" with standard window controls. The terminal shows the command `sudo apt-get install libpigpio-dev` being executed. The output indicates that the package is already installed and is the newest version. It also lists `libfuse2` as a package that was automatically installed and is no longer required, with instructions on how to remove it using `sudo apt autoremove`. The terminal ends with a summary of package counts and a new prompt.

```
pi@raspberrypi:~ $ sudo apt-get install libpigpio-dev
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
libpigpio-dev is already the newest version (1.79-1+rpt1).
libpigpio-dev set to manually installed.
The following package was automatically installed and is no longer required:
  libfuse2
Use 'sudo apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
pi@raspberrypi:~ $
```

Step 7. View and understand the /usr/include/pigpio.h file

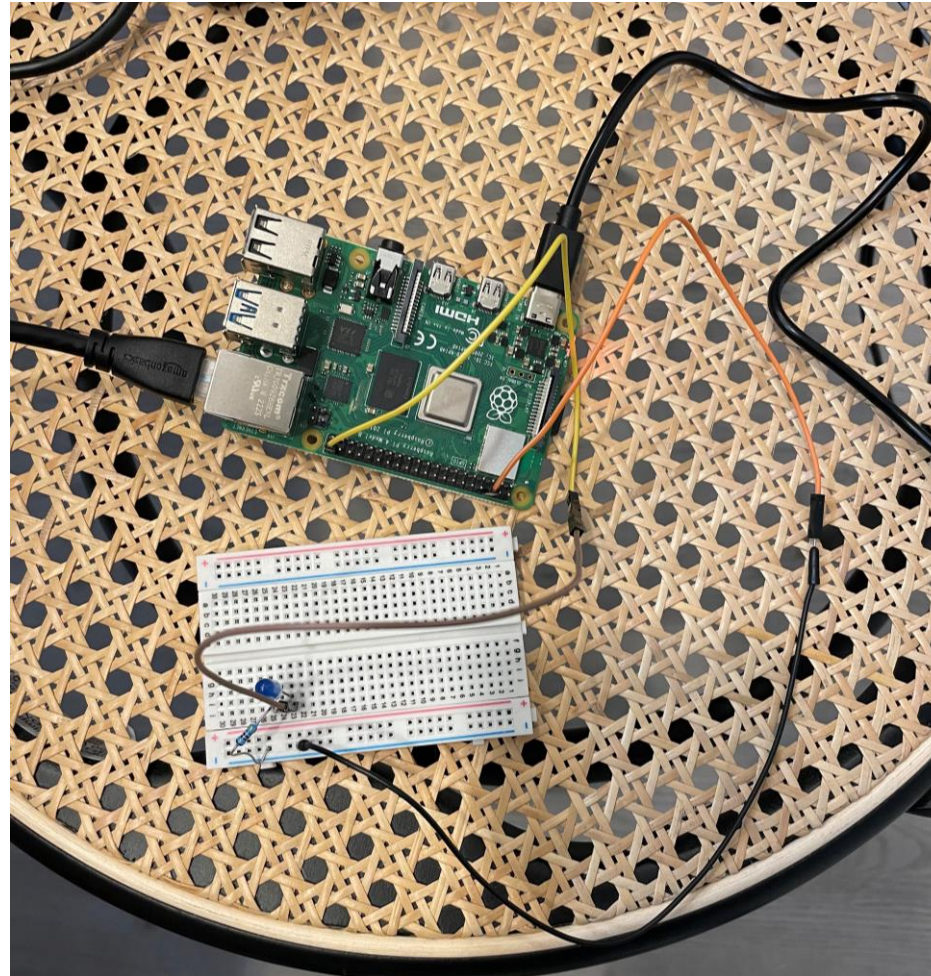


```
raspberrypi.local - pi@raspberrypi: ~ VT
File Edit Setup Control Window Help
#include <stddef.h>
#include <stdint.h>
#include <pthread.h>

#define PIGPIO_VERSION 79

/*TEXT
pigpio is a C library for the Raspberry which allows control of the GPIO.
*Features*
o hardware timed PWM on any of GPIO 0-31
o hardware timed servo pulses on any of GPIO 0-31
o callbacks when any of GPIO 0-31 change state
o callbacks at timed intervals
o reading/writing all of the GPIO in a bank as one operation
o individually setting GPIO modes, reading and writing
o notifications when any of GPIO 0-31 change state
o the construction of output waveforms with microsecond timing
o rudimentary permission control over GPIO
o a simple interface to start and stop new threads
o I2C, SPI, and serial link wrappers
o creating and running scripts
*GPIO*
ALL GPIO are identified by their Broadcom number.
*Credits*
The PWM and servo pulses are timed using the DMA and PWM peripherals.
This use was inspired by Richard Hirst's servoblaster kernel module.
*Usage*
```


Step 8. Let's Connect LED to GPIO Pin 21



Step 9. led21_blink.c part 1

```
raspberrypi.local - pi@raspberrypi: ~ VT
File Edit Setup Control Window Help
pi@raspberrypi:~$ cat led21-blink.c
#include <stdio.h>
#include <pigpio.h>
#include <signal.h>
#include <unistd.h>

#define LED_PIN 21

int running = 1;
void handle_sig_int(int sig) {
    running = 0;
}

int main() {
    int result = gpioInitialise();
    if(result < 0) {
        fprintf(stderr, "gpioInitialise() failed\n");
        result = 1;
        goto getOut;
    }

    result = gpioSetMode(LED_PIN, PI_OUTPUT);
    if(result < 0) {
        fprintf(stderr, "gpioSetMode() failed\n");
        result = 1;
        goto getOut;
    }

    // We need to use signals
    int cfg = gpioCfgGetInternals();
    cfg |= PI_CFG_NOSIGHANDLER;
    gpioCfgSetInternals(cfg);

    signal(SIGINT, handle_sig_int);

    int toggle = 1;

    while(running) {
        sleep(1);
        result = gpioWrite(LED_PIN, toggle);
    }
}
```

Step 10. led21_blink.c part 2

```
raspberrypi.local - pi@raspberrypi: ~ VT
File Edit Setup Control Window Help
    running = 0;
}

int main() {
    int result = gpioInitialise();
    if(result < 0) {
        fprintf(stderr, "gpioInitialise() failed\n");
        result = 1;
        goto getOut;
    }

    result = gpioSetMode(LED_PIN, PI_OUTPUT);
    if(result < 0) {
        fprintf(stderr, "gpioSetMode() failed\n");
        result = 1;
        goto getOut;
    }

    // We need to use signals
    int cfg = gpioCfgGetInternals();
    cfg |= PI_CFG_NOSIGHANDLER;
    gpioCfgSetInternals(cfg);

    signal(SIGINT, handle_sig_int);

    int toggle = 1;

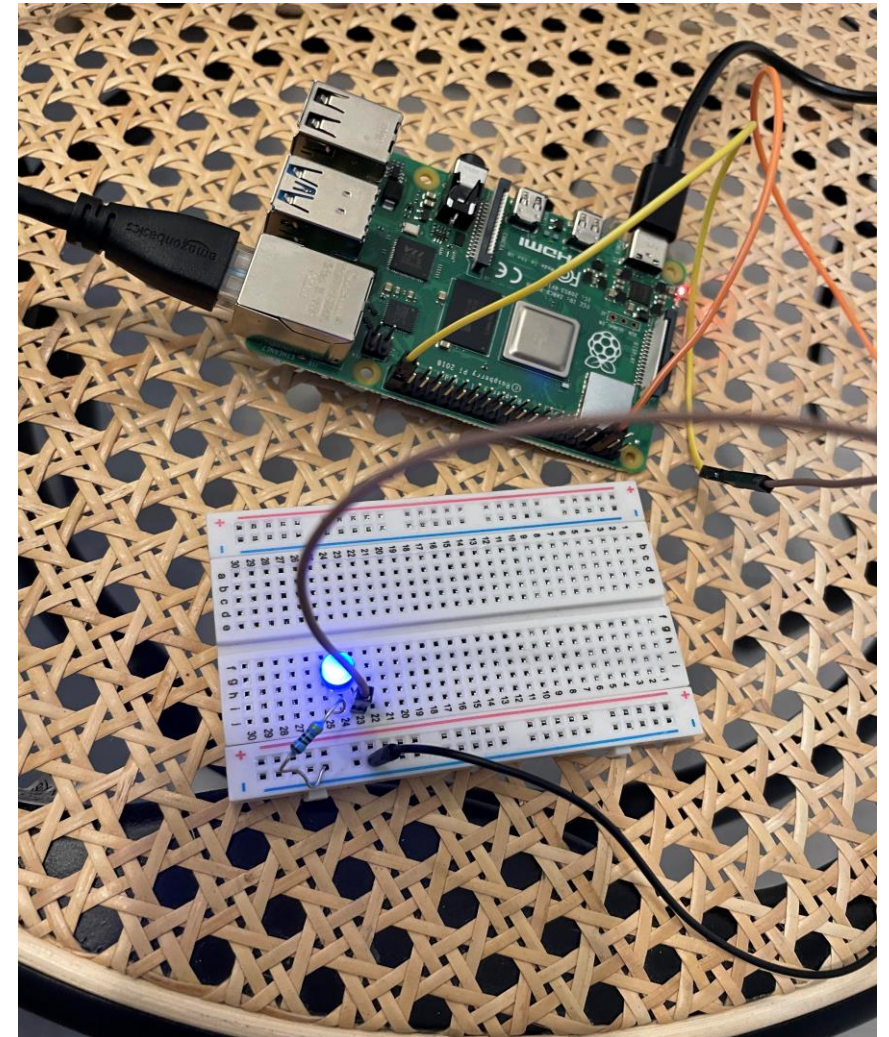
    while(running) {
        sleep(1);
        result = gpioWrite(LED_PIN, toggle);
        if(result < 0) break;
        toggle ^= 1; // Toggle bit
    }

getOut:
    gpioTerminate();
    return 0;
}

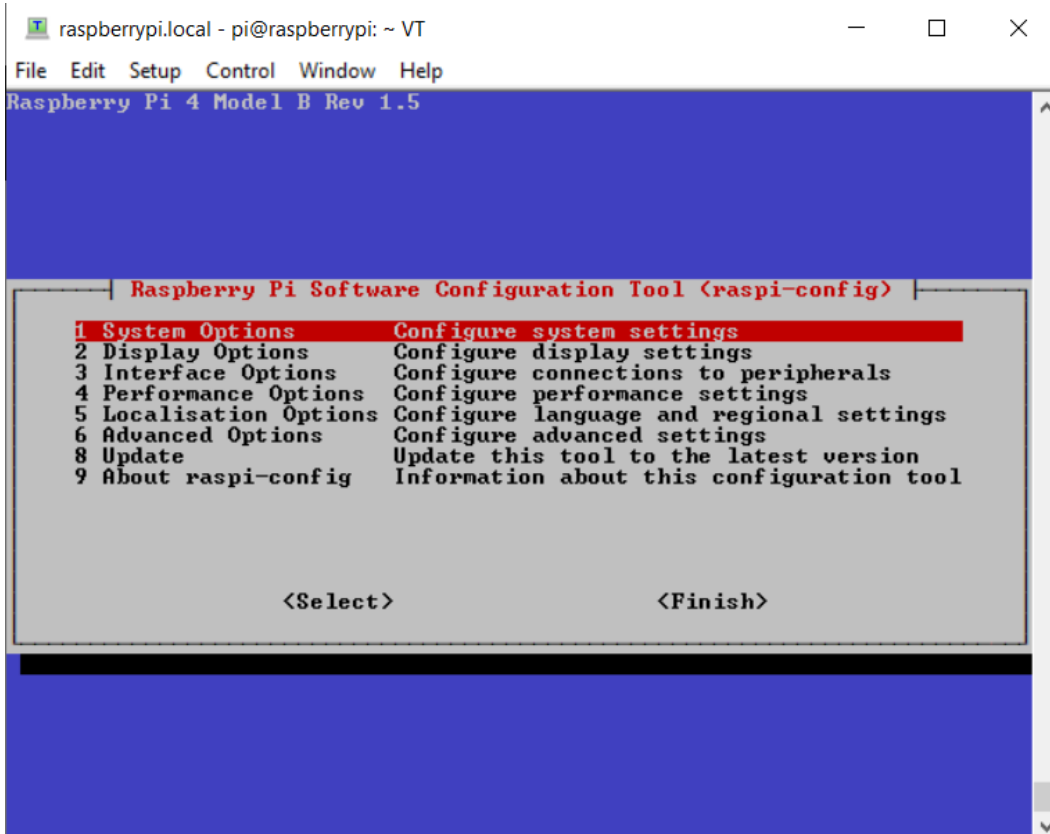
pi@raspberrypi:~ $
```


Step 11. build and run the code, the LED is indeed blinking

```
raspberrypi.local - pi@raspberrypi: ~ VT
File Edit Setup Control Window Help
pi@raspberrypi:~$ gcc -Wall -o led21-blink led21-blink.c -lpthread -lpigpio -lrt
pi@raspberrypi:~$ ldd led21-blink
        /usr/lib/arm-linux-gnueabi/libarmmem-${PLATFORM}.so => /usr/lib/arm-linux-gnueabi/
libarmmem-v81.so (0xf7eb6000)
        libpigpio.so.1 => /lib/libpigpio.so.1 (0xf7dbf000)
        libc.so.6 => /lib/arm-linux-gnueabi/libc.so.6 (0xf7c6b000)
        libpthread.so.0 => /lib/arm-linux-gnueabi/libpthread.so.0 (0xf7c3f000)
        /lib/ld-linux-armhf.so.3 (0xf7ecb000)
pi@raspberrypi:~$ sudo ./led21-blink
```



Step 12. Demo 2 I2C Access from C. Configuration of I2C



raspberrypi.local - pi@raspberrypi: ~ VT

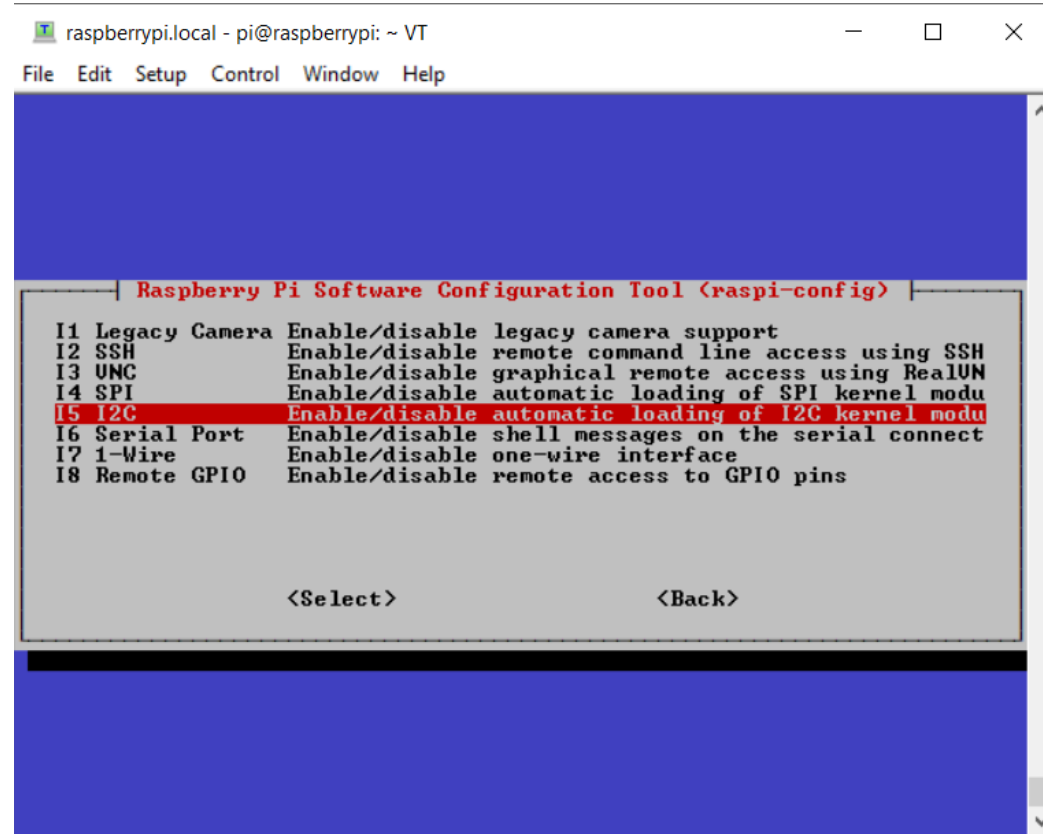
File Edit Setup Control Window Help

Raspberry Pi 4 Model B Rev 1.5

Raspberry Pi Software Configuration Tool (raspi-config)

1 System Options	Configure system settings
2 Display Options	Configure display settings
3 Interface Options	Configure connections to peripherals
4 Performance Options	Configure performance settings
5 Localisation Options	Configure language and regional settings
6 Advanced Options	Configure advanced settings
8 Update	Update this tool to the latest version
9 About raspi-config	Information about this configuration tool

<Select> <Finish>



raspberrypi.local - pi@raspberrypi: ~ VT

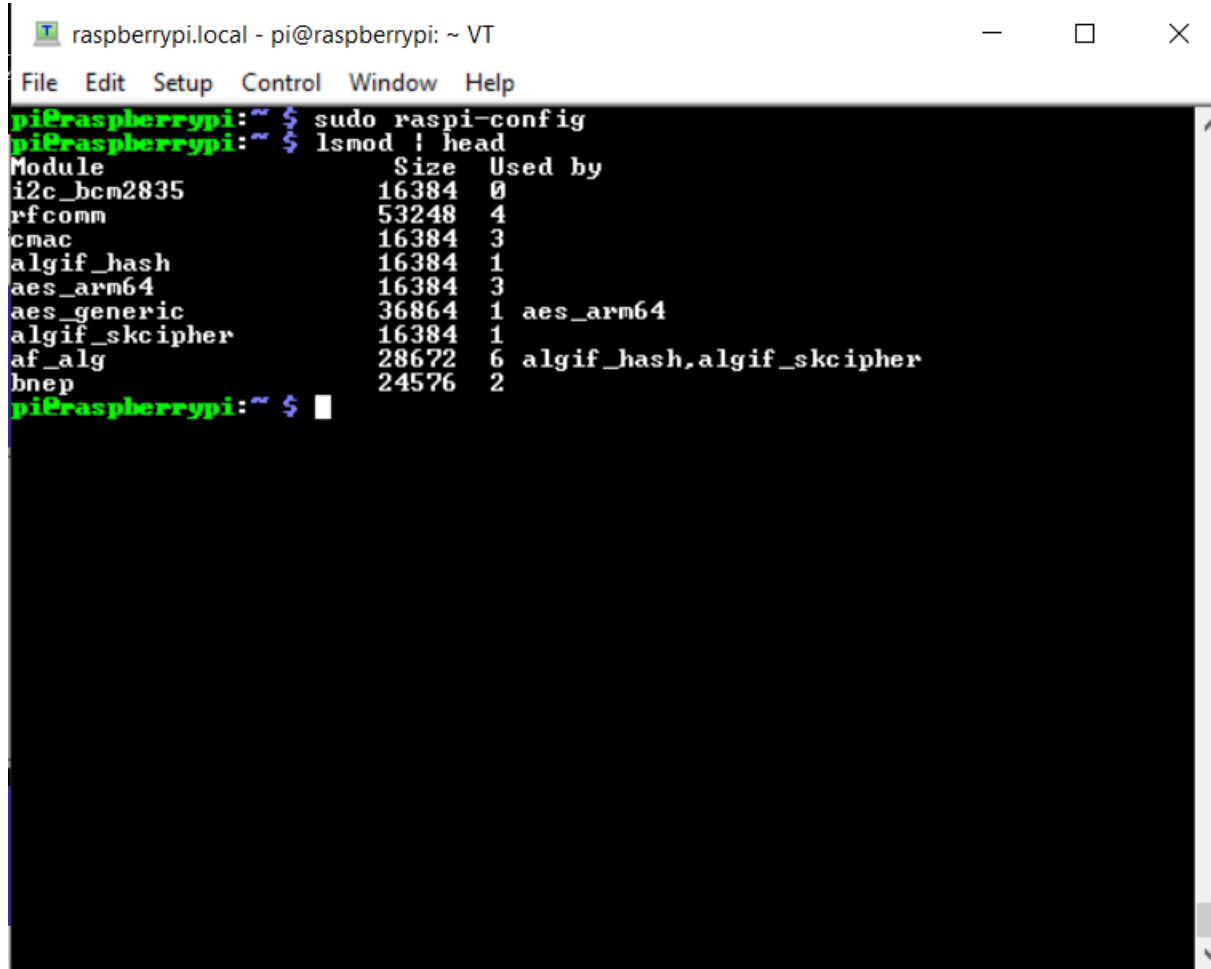
File Edit Setup Control Window Help

Raspberry Pi Software Configuration Tool (raspi-config)

I1 Legacy Camera	Enable/disable legacy camera support
I2 SSH	Enable/disable remote command line access using SSH
I3 UNC	Enable/disable graphical remote access using RealVNC
I4 SPI	Enable/disable automatic loading of SPI kernel module
I5 I2C	Enable/disable automatic loading of I2C kernel module
I6 Serial Port	Enable/disable shell messages on the serial connect
I7 1-Wire	Enable/disable one-wire interface
I8 Remote GPIO	Enable/disable remote access to GPIO pins

<Select> <Back>

Step 13. lsmod | head



```
raspberrypi.local - pi@raspberrypi: ~ VT
File Edit Setup Control Window Help
pi@raspberrypi:~$ sudo raspi-config
pi@raspberrypi:~$ lsmod | head
Module              Size  Used by
i2c_bcm2835         16384  0
rfcomm              53248  4
cmac                 16384  3
algif_hash          16384  1
aes_arm64           16384  3
aes_generic         36864  1 aes_arm64
algif_skcipher       16384  1
af_alg              28672  6 algif_hash,algif_skcipher
bnep                 24576  2
pi@raspberrypi:~$
```

The image shows a terminal window titled "raspberrypi.local - pi@raspberrypi: ~ VT". The window has a menu bar with "File", "Edit", "Setup", "Control", "Window", and "Help". The terminal output shows the user running "sudo raspi-config" and then "lsmod | head". The output of "lsmod | head" is a table with three columns: "Module", "Size", and "Used by". The table lists several kernel modules and their usage. The user's prompt "pi@raspberrypi:~\$" is visible at the bottom of the terminal.

Module	Size	Used by
i2c_bcm2835	16384	0
rfcomm	53248	4
cmac	16384	3
algif_hash	16384	1
aes_arm64	16384	3
aes_generic	36864	1 aes_arm64
algif_skcipher	16384	1
af_alg	28672	6 algif_hash,algif_skcipher
bnep	24576	2

Step 14. i2c-led.c file

```
raspberrypi.local - pi@raspberrypi: ~ VT
File Edit Setup Control Window Help
pi@raspberrypi:~$ cat i2c-led.c
#include <stdio.h>
#include <pigpio.h>
#include <signal.h>
#include <unistd.h>

#define MCP23008_I2C_ADDR 0x20

int running = 1;
void handle_sig_int(int sig) {
    running = 0;
}

int main() {
    int result = gpioInitialise();
    if(result < 0) {
        fprintf(stderr, "gpioInitialise() failed\n");
        result = 1;
        goto getOut;
    }

    // We need to use signals
    int cfg = gpioCfgGetInternals();
    cfg != PI_CFG_NOSIGHANDLER;
    gpioCfgSetInternals(cfg);

    signal(SIGINT, handle_sig_int);

    unsigned i2cBus = 1;
    unsigned i2cAddr = MCP23008_I2C_ADDR;
    unsigned i2cFlags = 0;
    int i2cHandle = i2cOpen(i2cBus, i2cAddr, i2cFlags);
    if(i2cHandle < 0) {
        fprintf(stderr, "i2cOpen() failed\n");
        goto getOut;
    }

    // Set MCP23008 I/O direction for all pins as output(0x00)
    unsigned i2cReg = 0;
    unsigned i2cValue = 0;
```

```
raspberrypi.local - pi@raspberrypi: ~ VT
File Edit Setup Control Window Help
    cfg != PI_CFG_NOSIGHANDLER;
    gpioCfgSetInternals(cfg);

    signal(SIGINT, handle_sig_int);

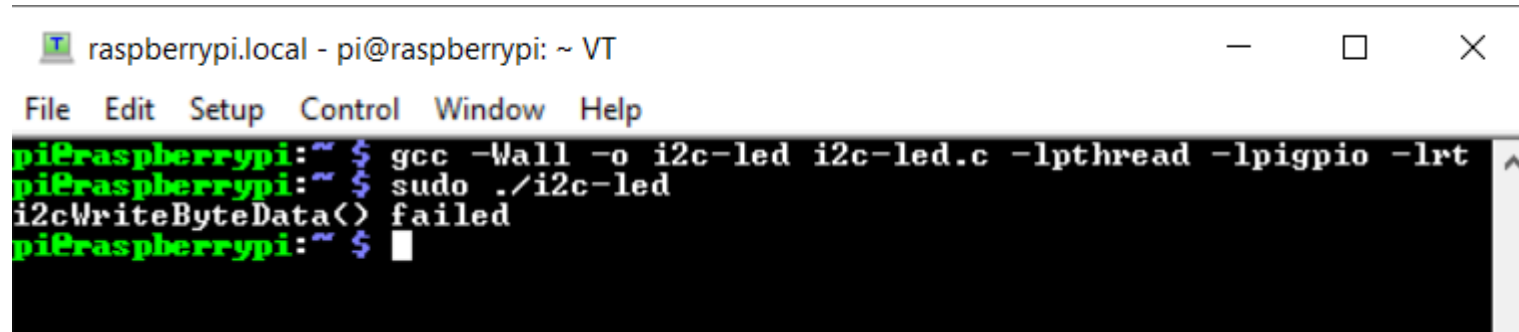
    unsigned i2cBus = 1;
    unsigned i2cAddr = MCP23008_I2C_ADDR;
    unsigned i2cFlags = 0;
    int i2cHandle = i2cOpen(i2cBus, i2cAddr, i2cFlags);
    if(i2cHandle < 0) {
        fprintf(stderr, "i2cOpen() failed\n");
        goto getOut;
    }

    // Set MCP23008 I/O direction for all pins as output(0x00)
    unsigned i2cReg = 0;
    unsigned i2cValue = 0;
    result = i2cWriteByteData(i2cHandle, i2cReg, i2cValue);
    if(result != 0) {
        fprintf(stderr, "i2cWriteByteData() failed\n");
        goto getOut;
    }

    // Toggle the pins connected to the MCP23008 I/O expander
    i2cReg = 0x09;
    while(1) {
        i2cValue = 0xff;
        i2cWriteByteData(i2cHandle, i2cReg, i2cValue);
        sleep(5);
        i2cValue = 0;
        i2cWriteByteData(i2cHandle, i2cReg, i2cValue);
        sleep(5);
    }

getOut:
    if(i2cHandle >= 0) i2cClose(i2cHandle);
    gpioTerminate();
    return 0;
}
pi@raspberrypi:~$
```

Step 15. build the code, build is successful. I do not have MCP23008 to test, so this is the end of my demo



```
raspberrypi.local - pi@raspberrypi: ~ VT
File Edit Setup Control Window Help
pi@raspberrypi:~$ gcc -Wall -o i2c-led i2c-led.c -lpthread -lpigpio -lrt
pi@raspberrypi:~$ sudo ./i2c-led
i2cWriteByteData() failed
pi@raspberrypi:~$
```