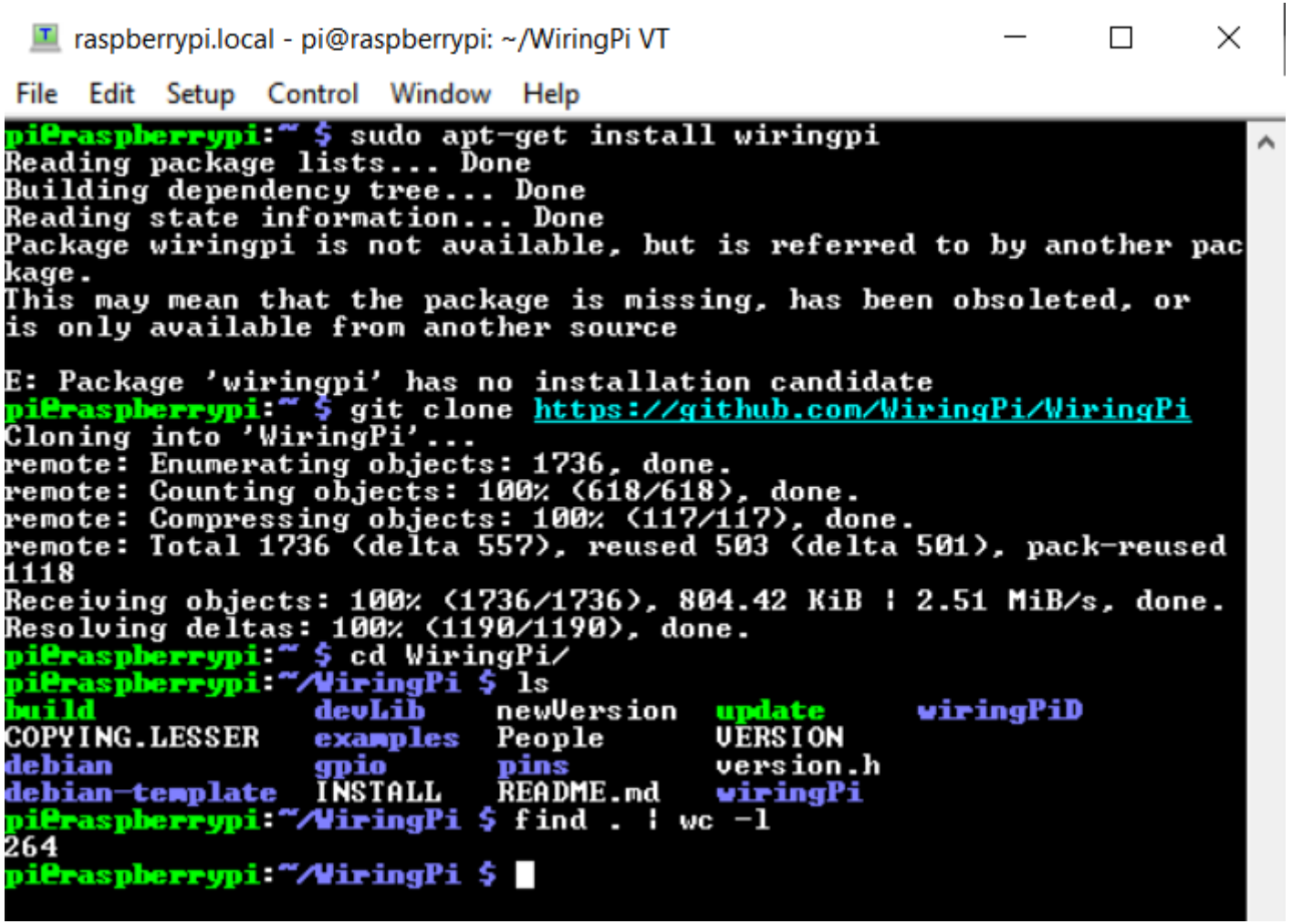
UCSD Embedded Linux Assignment 7

By

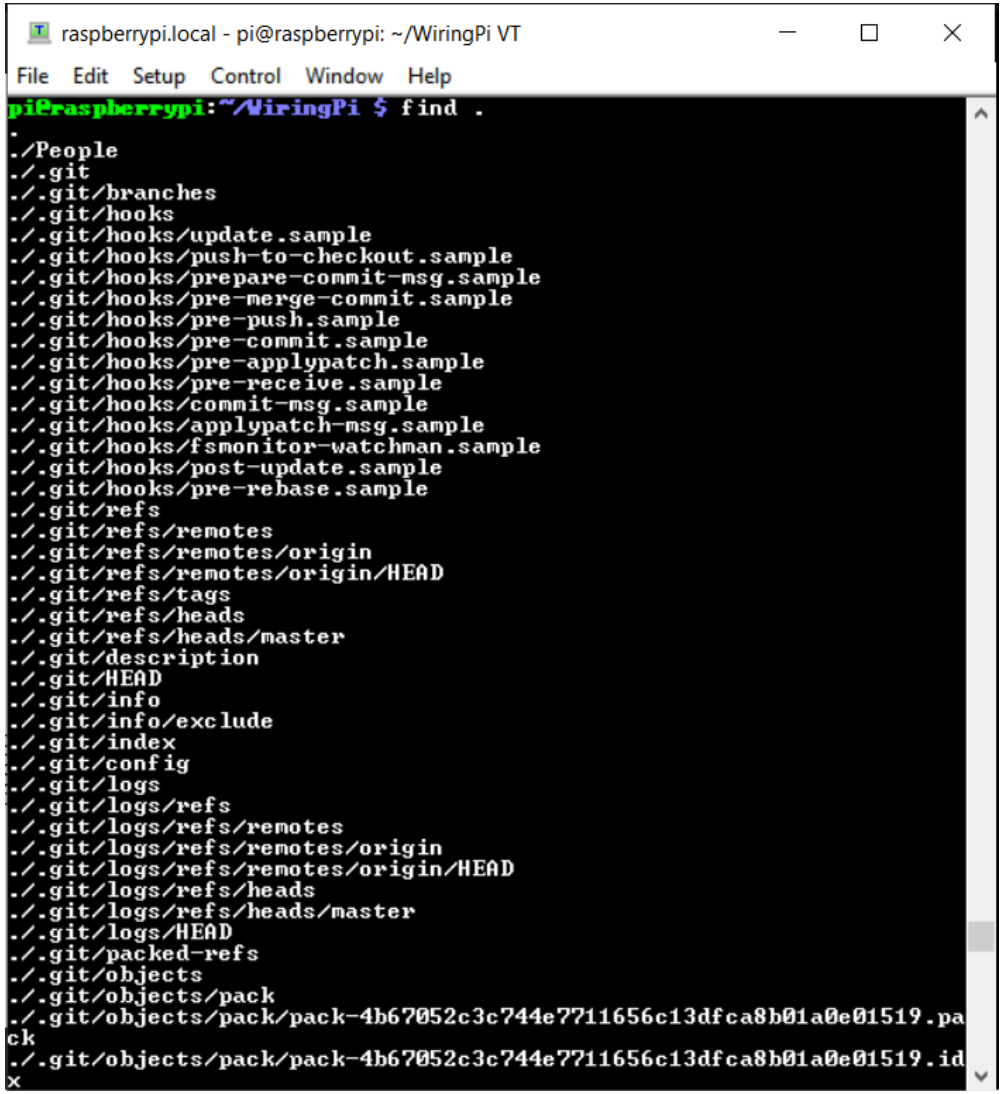
Hsuankai Chang

hsuankac@umich.edu

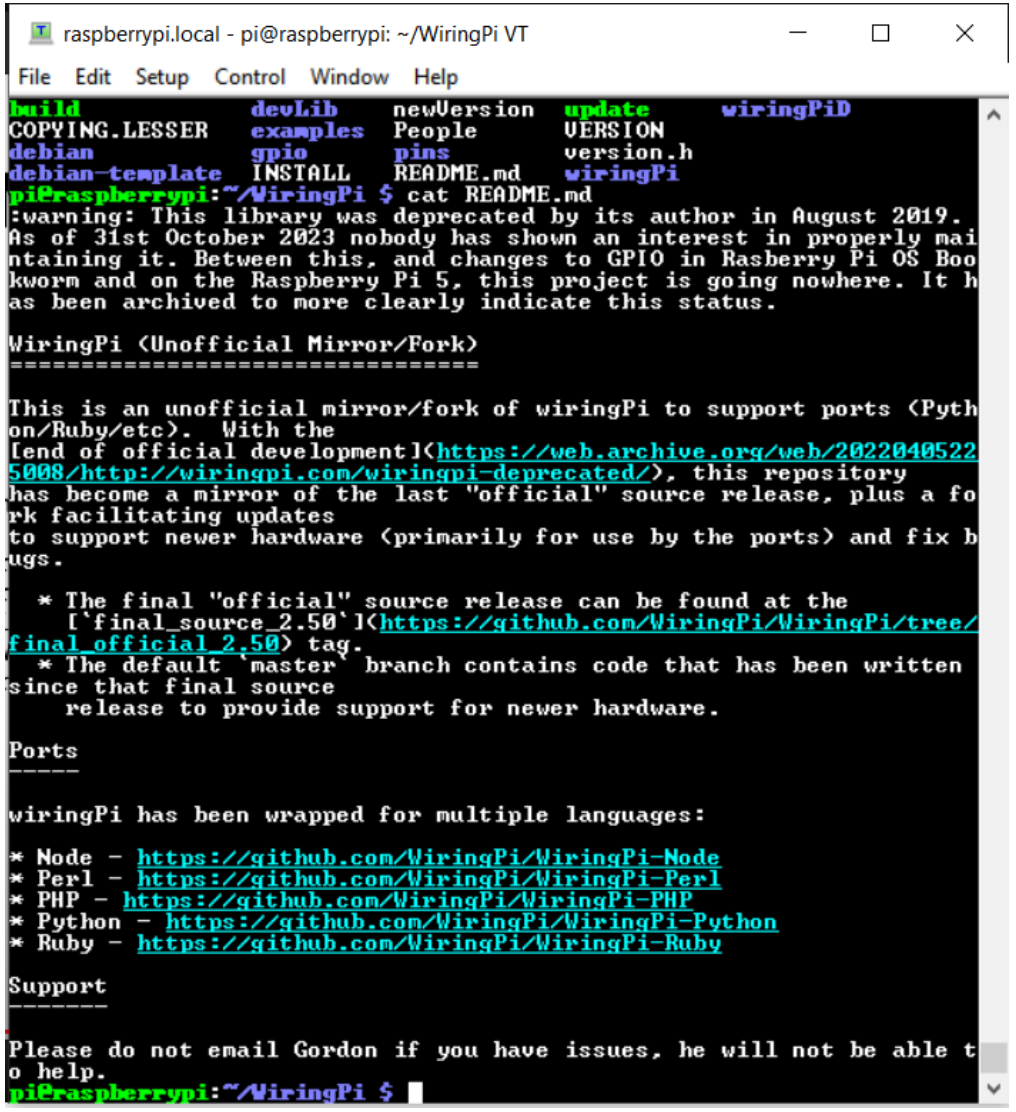
Step 1. Try to install wiringpi library



Step 2. find . In WiringPi



Step 3. WiringPi README.md file



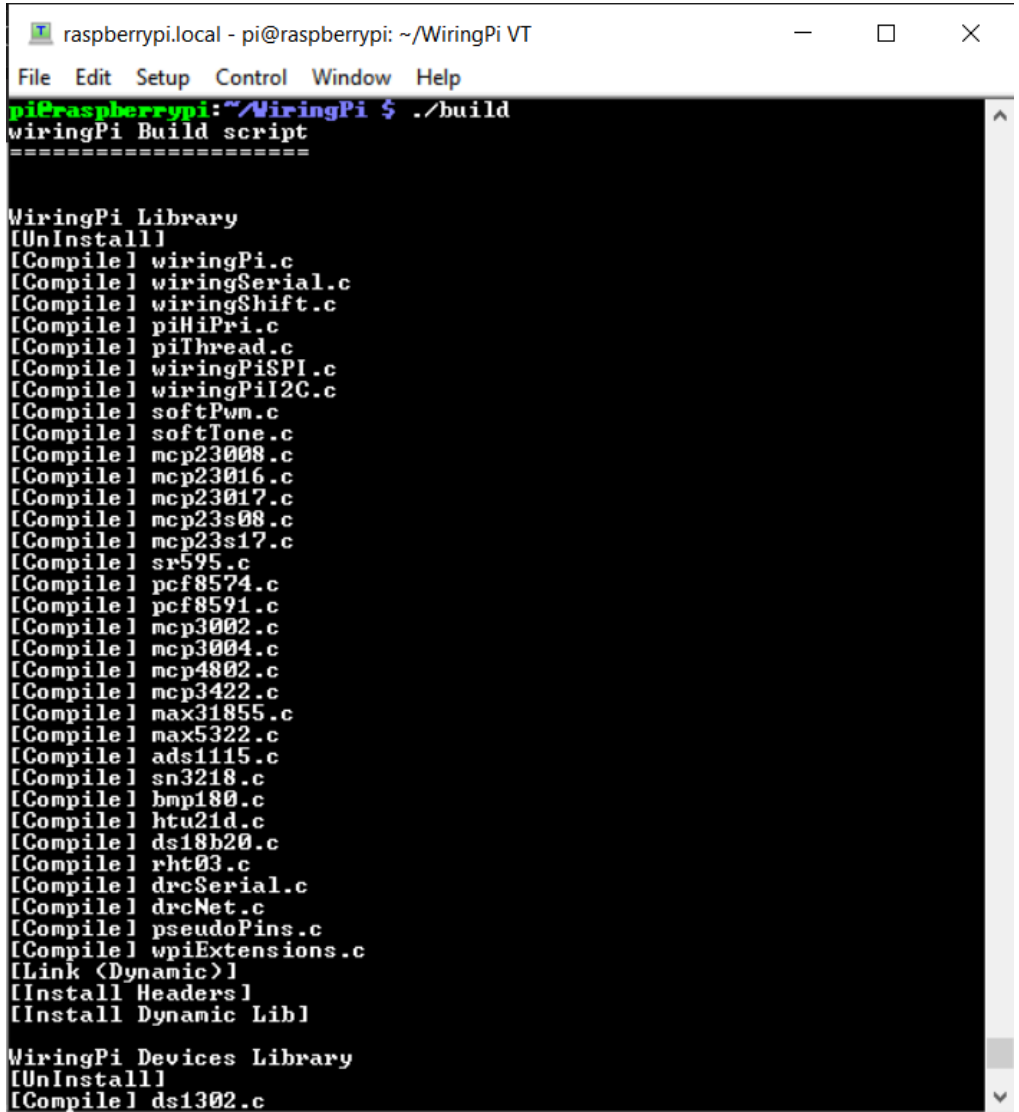
Step 4. Viewing the build script

raspberrypi.local - pi@raspberrypi: ~/WiringPi VT

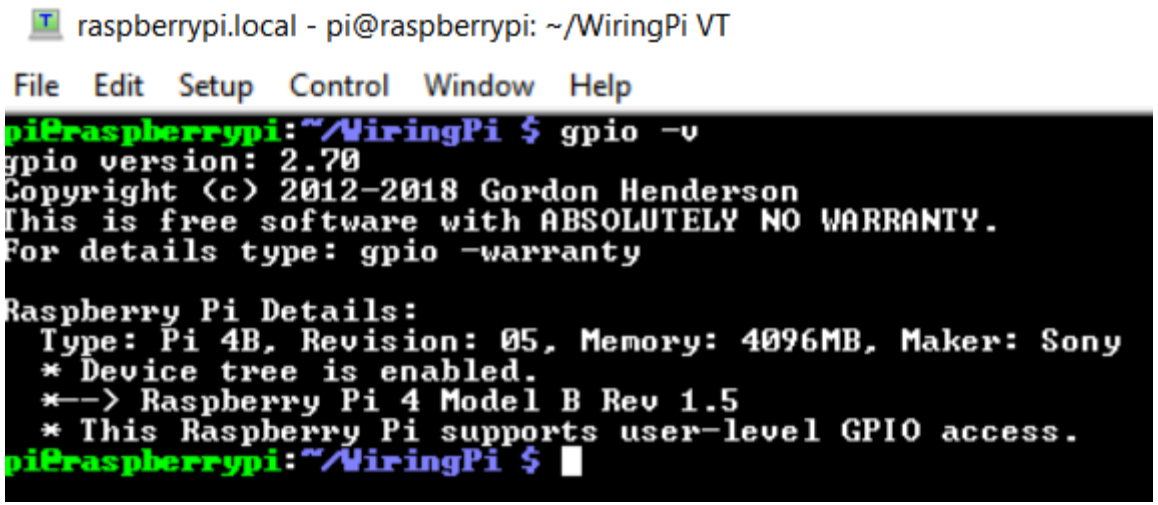
File Edit Setup Control Window Help

```
pi@raspberrypi:~/WiringPi $ cat build
#!/bin/sh -e

# build
#       Simple wiringPi build and install script
#
#       Copyright (c) 2012-2015 Gordon Henderson
#################################################################################
##############
# This file is part of wiringPi:
#       A "wiring" library for the Raspberry Pi
#
#    wiringPi is free software: you can redistribute it and/or modif
y
#    it under the terms of the GNU Lesser General Public License as
published by
#    the Free Software Foundation, either version 3 of the License,
or
#    (at your option) any later version.
#
#    wiringPi is distributed in the hope that it will be useful,
#    but WITHOUT ANY WARRANTY; without even the implied warranty of
#    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
#    GNU Lesser General Public License for more details.
#
#    You should have received a copy of the GNU Lesser General Publi
c License
#    along with wiringPi.  If not, see <http://www.gnu.org/licenses/
>.
#################################################################################
##############
#
# wiringPi is designed to run on a Raspberry Pi only.
#       However if you're clever enough to actually look at this scr
ipt to
#       see why it's not building for you, then good luck.
#
#       To everyone else: Stop using cheap alternatives. Support the
#       Raspberry Pi Foundation as they're the only ones putting mon
ey
#       back into education!
#################################################################################
##############

check_make_ok() {
  if [ $? != 0 ]; then
    echo ""
```

Step 5. Build WiringPi



Step 6. gpio –v command



Step 7. strace -f -o /tmp/gpio.trace gpio write 21 1

raspberrypi.local - pi@raspberrypi: ~/WiringPi VT

File Edit Setup Control Window Help

```
1816  set_tls(0xb6fda940)                = 0
1816  mprotect(0xb6f1d000, 8192, PROT_READ) = 0
1816  mprotect(0xb6d3f000, 4096, PROT_READ) = 0
1816  mprotect(0xb6f4b000, 4096, PROT_READ) = 0
1816  mprotect(0xb6d5f000, 4096, PROT_READ) = 0
1816  mprotect(0xb6dce000, 4096, PROT_READ) = 0
1816  mprotect(0xb6f62000, 4096, PROT_READ) = 0
1816  mprotect(0xb6f81000, 4096, PROT_READ) = 0
1816  mprotect(0xb6faa000, 4096, PROT_READ) = 0
1816  mprotect(0x26000, 4096, PROT_READ) = 0
1816  mprotect(0xb6fde000, 4096, PROT_READ) = 0
1816  munmap(0xb6f84000, 76555)          = 0
1816  set_tid_address(0xb6fda4e8)        = 1816
1816  set_robust_list(0xb6fda4f0, 12)    = 0
1816  rt_sigaction(SIGRTMIN, {sa_handler=0xb6f276b8, sa_mask=[], sa_flags=SA_RESTO
RER|SA_SIGINFO, sa_restorer=0xb6dff910}, NULL, 8) = 0
1816  rt_sigaction(SIGRT_1, {sa_handler=0xb6f27774, sa_mask=[], sa_flags=SA_RESTOR
ER|SA_RESTART|SA_SIGINFO, sa_restorer=0xb6dff910}, NULL, 8) = 0
1816  rt_sigprocmask(SIG_UNBLOCK, [RTMIN RT_1], NULL, 8) = 0
1816  ugetrlimit(RLIMIT_STACK, {rlim_cur=8192*1024, rlim_max=RLIM_INFINITY}) = 0
1816  geteuid32()                        = 0
1816  brk(NULL)                          = 0x108f000
1816  brk(0x10b0000)                     = 0x10b0000
1816  openat(AT_FDCWD, "/proc/cpuinfo", O_RDONLY) = 3
1816  fstat64(3, {st_mode=S_IFREG|0444, st_size=0, ...}) = 0
1816  read(3, "processor\t: 0\nmodel name\t: ARMv7"..., 1024) = 1024
1816  read(3, "PU variant\t: 0x0\nCPU part\t: 0xd0"..., 1024) = 156
1816  _llseek(3, 0, [0], SEEK_SET)      = 0
1816  read(3, "processor\t: 0\nmodel name\t: ARMv7"..., 1024) = 1024
1816  read(3, "PU variant\t: 0x0\nCPU part\t: 0xd0"..., 1024) = 156
1816  close(3)                          = 0
1816  openat(AT_FDCWD, "/proc/cpuinfo", O_RDONLY) = 3
1816  fstat64(3, {st_mode=S_IFREG|0444, st_size=0, ...}) = 0
1816  read(3, "processor\t: 0\nmodel name\t: ARMv7"..., 1024) = 1024
1816  read(3, "PU variant\t: 0x0\nCPU part\t: 0xd0"..., 1024) = 156
1816  close(3)                          = 0
1816  openat(AT_FDCWD, "/dev/mem", O_RDWR|O_SYNC|O_CLOEXEC) = 3
1816  mmap2(NULL, 4096, PROT_READ|PROT_WRITE, MAP_SHARED, 3, 0xfe200000) = 0xb6fd9
000
1816  mmap2(NULL, 4096, PROT_READ|PROT_WRITE, MAP_SHARED, 3, 0xfe20c000) = 0xb6fd8
000
1816  mmap2(NULL, 4096, PROT_READ|PROT_WRITE, MAP_SHARED, 3, 0xfe101000) = 0xb6fd7
000
1816  mmap2(NULL, 4096, PROT_READ|PROT_WRITE, MAP_SHARED, 3, 0xfe100000) = 0xb6fd6
000
1816  mmap2(NULL, 4096, PROT_READ|PROT_WRITE, MAP_SHARED, 3, 0xfe00b000) = 0xb6fd5
000
1816  clock_gettime64(CLOCK_MONOTONIC_RAW, {tv_sec=2714, tv_nsec=994379901}) = 0
1816  exit_group(0)                      = ?
1816  +++ exited with 0 +++
```

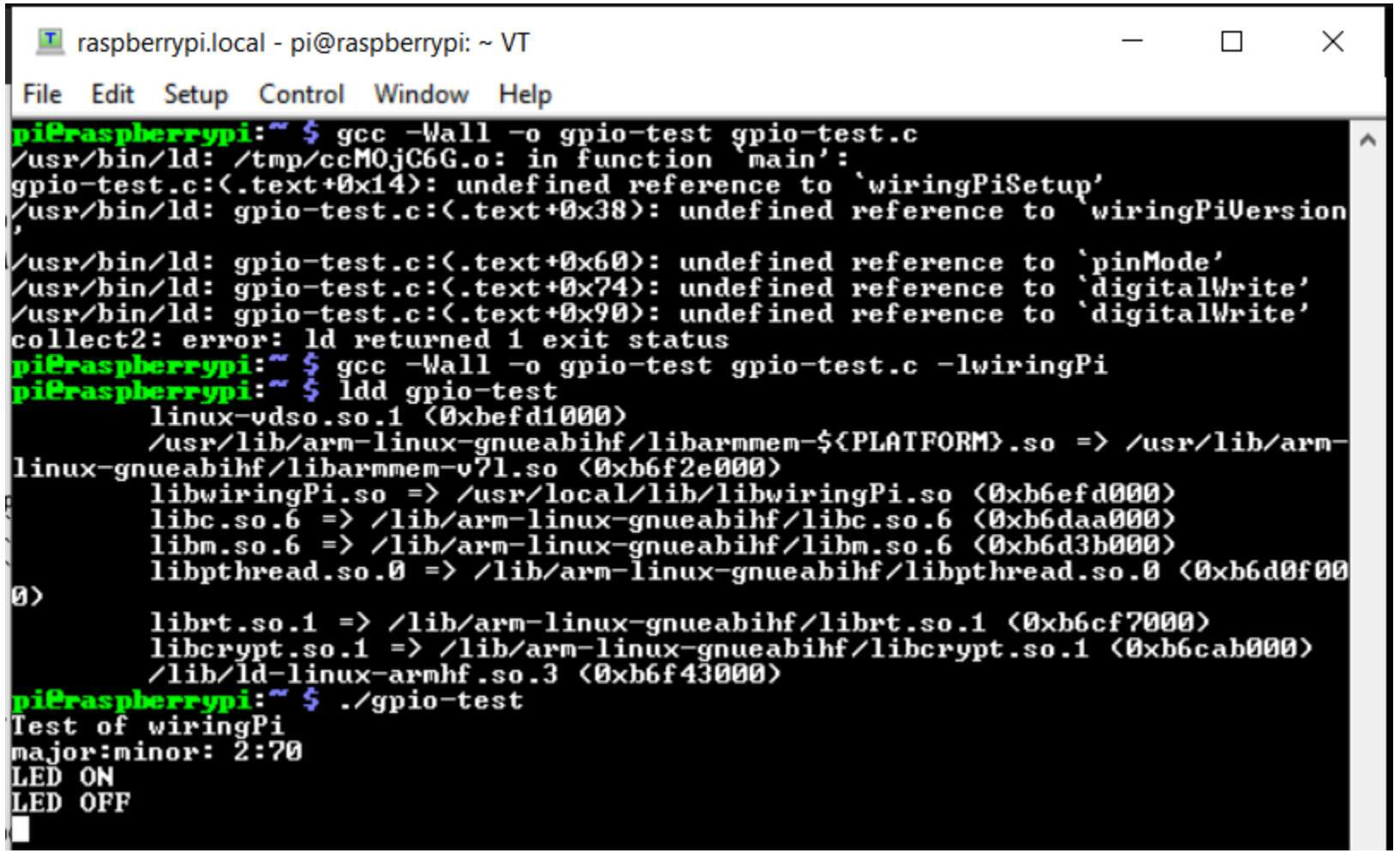
Step 8. /usr/local/include

```
pi@raspberrypi:~/WiringPi $ ls /usr/local/include
ads1115.h     max5322.h       mcp3004.h      sn3218.h
bmp180.h      maxdetect.h     mcp3422.h      softPwm.h
drcNet.h      mcp23008.h      mcp4802.h      softServo.h
drcSerial.h   mcp23016.h      pcf8574.h      softTone.h
ds1302.h      mcp23016reg.h   pcf8591.h      sr595.h
ds18b20.h     mcp23017.h      piFace.h       wiringPi.h
gertboard.h   mcp23s08.h      piGlow.h       wiringPiI2C.h
htu21d.h      mcp23s17.h      piNes.h        wiringPiSPI.h
lcd128x64.h   mcp23x0817.h    pseudoPins.h   wiringSerial.h
lcd.h         mcp23x08.h      rht03.h        wiringShift.h
max31855.h    mcp3002.h       scrollPhat.h   wpiExtensions.h
pi@raspberrypi:~/WiringPi $
```

Step 9. gpio-test.c file

```
pi@raspberrypi:~ $ cat gpio-test.c
#include <stdio.h>
#include <wiringPi.h>
#include <unistd.h>

int main()    {
    puts("Test of wiringPi");

    wiringPiSetup();

    int major = 42;
    int minor = 42;
    wiringPiVersion(&major, &minor);
    printf("major:minor: %d:%d\n", major, minor);

    int gpio21 = 21;
    pinMode(gpio21, OUTPUT);

    while(1) {
        puts("LED ON");
        digitalWrite(gpio21, 1);
        sleep(10);
        puts("LED OFF");
        digitalWrite(gpio21, 0);
        sleep(10);
    }

    return 0;
}
```

Step 10. Build and run the gpio-test.c file



Step 11. Edit the /boot/config.txt file, reboot and check the interface

```
#uncomment to overclock the arm. 700 MHz is the default.
#arm_freq=800

# Uncomment some or all of these to enable the optional hardware interfaces
dtparam=i2c_arm=on
#dtparam=i2s=on
dtparam=spi=on

# Uncomment this to enable infrared communication.
#dtoverlay=gpio-ir,gpio_pin=17
#dtoverlay=gpio-ir-tx,gpio_pin=18
dtoverlay=mcp2515-can0,oscillator=8000000,interrupt=25,spimaxfrequency=1000000

# Additional overlays and parameters are documented /boot/overlays/README

# Enable audio (loads snd_bcm2835)
dtparam=audio=on

# Automatically load overlays for detected cameras
camera_auto_detect=1

# Automatically load overlays for detected DSI displays
display_auto_detect=1
```

```
pi@raspberrypi:~ $ dmesg | grep CAN
[    7.294782] CAN device driver interface
pi@raspberrypi:~ $ dmesg | grep spi0
[    7.454334] mcp251x spi0.0 can0: MCP2515 successfully initialized.
pi@raspberrypi:~ $
```

Step 12. ifconfig –a, setup the bitrate and ifconfig can0

raspberrypi.local - pi@raspberrypi: ~ VT

File Edit Setup Control Window Help

```
pi@raspberrypi:~ $ ifconfig -a
can0: flags=128<NOARP>  mtu 16
        unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00  txqueuelen 10  (
UNSPEC)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 169.254.195.45  netmask 255.255.0.0  broadcast 169.254.255.255
        inet6 fe80::f442:a915:8e7c:9ba3  prefixlen 64  scopeid 0x20<link>
        ether d8:3a:dd:42:a7:ee  txqueuelen 1000  (Ethernet)
        RX packets 742  bytes 62223 (60.7 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 338  bytes 42582 (41.5 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 23  bytes 2468 (2.4 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 23  bytes 2468 (2.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.4.86  netmask 255.255.252.0  broadcast 192.168.7.255
        inet6 fd06:39de:26fb:1:b300:104a:cbc2:299f  prefixlen 64  scopeid 0x0<gl
obal>
        inet6 fe80::81cc:3b68:dc1e:5beb  prefixlen 64  scopeid 0x20<link>
        ether d8:3a:dd:42:a7:f0  txqueuelen 1000  (Ethernet)
```

```
pi@raspberrypi:~ $ sudo ip link set can0 up type can bitrate 500000
pi@raspberrypi:~ $ ifconfig can0
can0: flags=193<UP,RUNNING,NOARP>  mtu 16
        unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00  txqueuelen 10  (
UNSPEC)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

pi@raspberrypi:~ $
```

Step 13. can-utils

```
pi@raspberrypi:~ $ sudo apt install can-utils
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following package was automatically installed and is no longer required:
  libfuse2
Use 'sudo apt autoremove' to remove it.
The following NEW packages will be installed:
  can-utils
0 upgraded, 1 newly installed, 0 to remove and 20 not upgraded.
Need to get 118 kB of archives.
After this operation, 492 kB of additional disk space will be used.
Get:1 http://raspbian.mirror.axinja.net/raspbian bullseye/main armhf can-utils a
rmhf 2020.11.0-1 [118 kB]
Fetched 118 kB in 2s (64.8 kB/s)
Selecting previously unselected package can-utils.
(Reading database ... 160873 files and directories currently installed.)
Preparing to unpack .../can-utils_2020.11.0-1_armhf.deb ...
Unpacking can-utils (2020.11.0-1) ...
Setting up can-utils (2020.11.0-1) ...
Processing triggers for man-db (2.9.4-2) ...
pi@raspberrypi:~ $ dpkg -L can-utils
/.
/usr
/usr/bin
/usr/bin/asc2log
/usr/bin/bcmserver
/usr/bin/can-calc-bit-timing
/usr/bin/canbusload
/usr/bin/candump
/usr/bin/canfdtest
/usr/bin/cangen
/usr/bin/cangw
/usr/bin/canlogserver
```

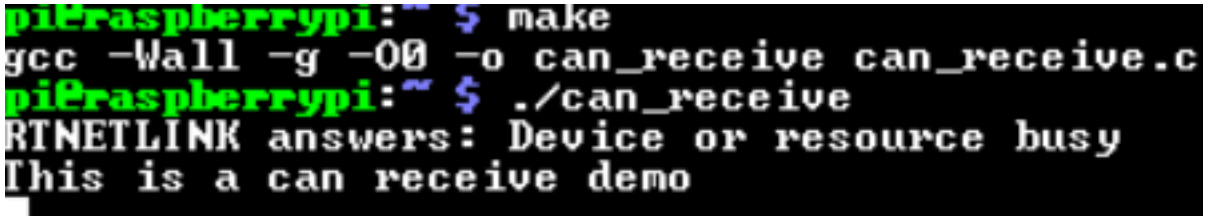
Step 14. can_receive.c and Makefile

```
pi@raspberrypi:~ $ cat can_receive.c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <net/if.h>
#include <sys/ioctl.h>
#include <sys/socket.h>
#include <linux/can.h>
#include <linux/can/raw.h>

int main()
{
    int ret;
    int s, nbytes;
    struct sockaddr_can addr;
    struct ifreq ifr;
    struct can_frame frame;

    memset(&frame, 0, sizeof(struct can_frame));

    system("sudo ip link set can0 type can bitrate 100000");
    system("sudo ifconfig can0 up");
    printf("This is a can receive demo\r\n");

    // 1. Create socket
    s = socket(PF_CAN, SOCK_RAW, CAN_RAW);
    if(s < 0) {
        perror("socket PF_CAN failed");
        return 1;
    }

    // 2. Specify CAN0 device
    strcpy(ifr.ifr_name, "can0");
    ret = ioctl(s, SIOCGIFINDEX, &ifr);
    if(ret < 0) {
        perror("ioctl failed");
        return 1;
    }

    // 3. Bind the socket to can0
    addr.can_family = PF_CAN;
    addr.can_ifindex = ifr.ifr_ifindex;
    ret = bind(s, (struct sockaddr*)&addr, sizeof(addr));
    if(ret < 0) {
        perror("bind failed");
        return 1;
    }
```

```
    // 4. Define receive rules
    struct can_filter rfilter[1];
    rfilter[0].can_id = 0x123;
    rfilter[0].can_mask = CAN_SFF_MASK;
    setsockopt(s, SOL_CAN_RAW, CAN_RAW_FILTER, &rfilter, sizeof(rfilter));

    // 5. Receive data and exit
    while(1) {
        nbytes = read(s, &frame, sizeof(frame));
        if(nbytes > 0) {
            printf("can_id = 0x%X\r\ncan_dlc = %d\r\n", frame.can_id, frame.ca
n_dlc);
            int i = 0;
            for(i = 0; i < 9; i++)
                printf("data[%d] = %d\r\n", i, frame.data[i]);
            break;
        }
    }

    // 6. Close the socket and can0
    close(s);
    system("sudo ifconfig can0 down");

    return 0;
}
```

```
pi@raspberrypi:~ $ cat Makefile
CC = gcc
CFLAGS = -Wall -g -O0

can_receive:can_receive.c
        $(CC) $(CFLAGS) -o $@ $^

clean:
        $(RM) can_receive .*.sw?
```

Step 15. Build and run the code

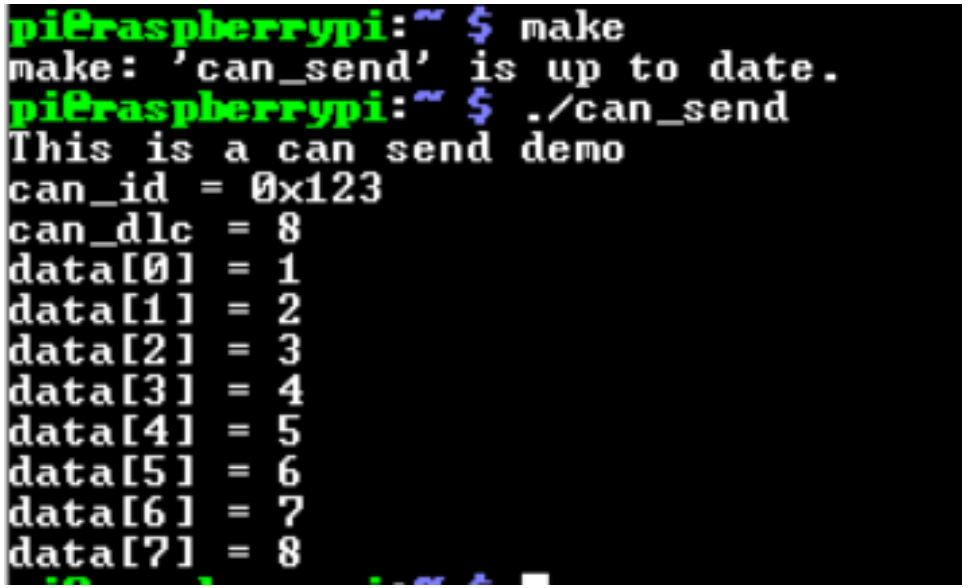


Step 16. can_send.c file

```
pi@raspberrypi:~ $ cat can_send.c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <net/if.h>
#include <sys/ioctl.h>
#include <sys/socket.h>
#include <linux/can.h>
#include <linux/can/raw.h>

int main()
{
    int ret;
    int s, nbytes;
    struct sockaddr_can addr;
    struct ifreq ifr;
    struct can_frame frame;

    memset(&frame, 0, sizeof(struct can_frame));

    system("sudo ip link set can0 type can bitrate 100000");
    system("sudo ifconfig can0 up");
    printf("This is a can send demo\r\n");

    // 1. Create socket
    s = socket(PF_CAN, SOCK_RAW, CAN_RAW);
    if(s < 0) {
        perror("socket PF_CAN failed");
        return 1;
    }

    // 2. Specify CAN0 device
    strcpy(ifr.ifr_name, "can0");
    ret = ioctl(s, SIOCGIFINDEX, &ifr);
    if(ret < 0) {
        perror("ioctl failed");
        return 1;
    }

    // 3. Bind the socket to can0
    addr.can_family = AF_CAN;
    addr.can_ifindex = ifr.ifr_ifindex;
    ret = bind(s, (struct sockaddr*)&addr, sizeof(addr));
    if(ret < 0) {
        perror("bind failed");
        return 1;
    }
```

```
    // 4. Disabling filtering rules, do not receive packets, only send
    setsockopt(s, SOL_CAN_RAW, CAN_RAW_FILTER, NULL, 0);

    // 5. Set send data
    frame.can_id = 0x123;
    frame.can_dlc = 8;
    frame.data[0] = 1;
    frame.data[1] = 2;
    frame.data[2] = 3;
    frame.data[3] = 4;
    frame.data[4] = 5;
    frame.data[5] = 6;
    frame.data[6] = 7;
    frame.data[7] = 8;

    printf("can_id = 0x%X\r\n", frame.can_id);
    printf("can_dlc = %d\r\n", frame.can_dlc);
    int i = 0;
    for(i = 0; i < 8; i++)
        printf("data[%d] = %d\r\n", i, frame.data[i]);

   // 6. Send message
   nbytes = write(s, &frame, sizeof(frame));
   if(nbytes != sizeof(frame)){
        printf("Send Error frame[0]!\r\n");
        system("sudo ifconfig can0 down");
    }

    // 7. Close the socket and can0
    close(s);
    system("sudo ifconfig can0 down");

    return 0;
}
```

Step 17. Build and run the code



Appendix, hardware connection, using cheaper version of MCP2515 CAN module