James Marvin, Marvin Newlin
Lab 3
29 April 2019

1.Can you toggle the GPIO pins (set them high/low)? How did you do it? How do you know it worked?

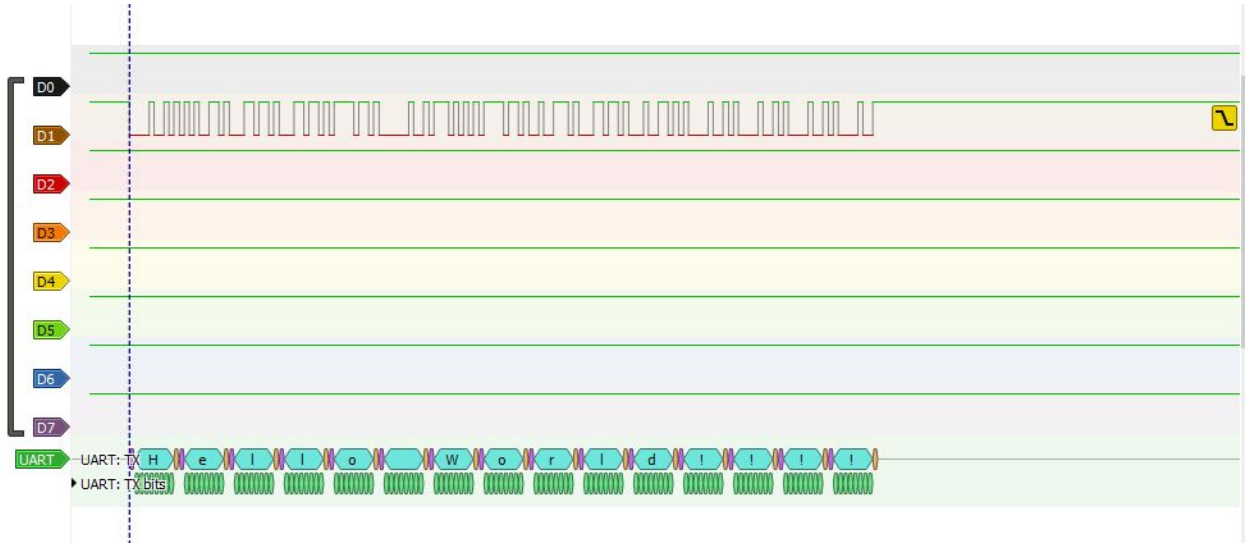
You can toggle the pins. The commands are **gpio mode <pin> out** and then **gpio write <pin> 0/1**. We knew the change occurred by examining the table in the **gpio readall** output. Here is an example of us changing GPIO. 0 to be high.

```
pi@raspberrypi:~ $ gpio mode 0 out
```

```
pi@raspberrypi:~ $ gpio readall
 +-----+-----+---------+------+---+---Pi 3B--+---+------+---------+-----+-----+
 | BCM | wPi |   Name  | Mode | V | Physical | V | Mode | Name    | wPi | BCM |
 +-----+-----+---------+------+---+----++----+---+------+---------+-----+-----+
 |     |     |    3.3v |      |   |  1 || 2  |   |      | 5v      |     |     |
 |   2 |   8 |   SDA.1 |   IN | 1 |  3 || 4  |   |      | 5v      |     |     |
 |   3 |   9 |   SCL.1 |   IN | 1 |  5 || 6  |   |      | 0v      |     |     |
 |   4 |   7 | GPIO. 7 |   IN | 1 |  7 || 8  | 1 | ALT5 | TxD     | 15  | 14  |
 |     |     |      0v |      |   |  9 || 10 | 1 | ALT5 | RxD     | 16  | 15  |
 |  17 |   0 | GPIO. 0 |  OUT | 0 | 11 || 12 | 0 | IN   | GPIO. 1 | 1   | 18  |
 |  27 |   2 | GPIO. 2 |   IN | 0 | 13 || 14 |   |      | 0v      |     |     |
 |  22 |   3 | GPIO. 3 |   IN | 0 | 15 || 16 | 0 | IN   | GPIO. 4 | 4   | 23  |
 |     |     |    3.3v |      |   | 17 || 18 | 0 | IN   | GPIO. 5 | 5   | 24  |
 |  10 |  12 |    MOSI |   IN | 0 | 19 || 20 |   |      | 0v      |     |     |
 |   9 |  13 |    MISO |   IN | 0 | 21 || 22 | 0 | IN   | GPIO. 6 | 6   | 25  |
 |  11 |  14 |    SCLK |   IN | 0 | 23 || 24 | 1 | IN   | CE0     | 10  | 8   |
 |     |     |      0v |      |   | 25 || 26 | 1 | IN   | CE1     | 11  | 7   |
 |   0 |  30 |   SDA.0 |   IN | 1 | 27 || 28 | 1 | IN   | SCL.0   | 31  | 1   |
 |   5 |  21 | GPIO.21 |   IN | 1 | 29 || 30 |   |      | 0v      |     |     |
 |   6 |  22 | GPIO.22 |   IN | 1 | 31 || 32 | 0 | IN   | GPIO.26 | 26  | 12  |
 |  13 |  23 | GPIO.23 |   IN | 0 | 33 || 34 |   |      | 0v      |     |     |
 |  19 |  24 | GPIO.24 |   IN | 0 | 35 || 36 | 0 | IN   | GPIO.27 | 27  | 16  |
 |  26 |  25 | GPIO.25 |   IN | 0 | 37 || 38 | 0 | IN   | GPIO.28 | 28  | 20  |
 |     |     |      0v |      |   | 39 || 40 | 0 | IN   | GPIO.29 | 29  | 21  |
 +-----+-----+---------+------+---+----++----+---+------+---------+-----+-----+
 | BCM | wPi |   Name  | Mode | V | Physical | V | Mode | Name    | wPi | BCM |
 +-----+-----+---------+------+---+---Pi 3B--+---+------+---------+-----+-----+
pi@raspberrypi:~ $ gpio write 0 1
pi@raspberrypi:~ $ gpio readall
 +-----+-----+---------+------+---+---Pi 3B--+---+------+---------+-----+-----+
 | BCM | wPi |   Name  | Mode | V | Physical | V | Mode | Name    | wPi | BCM |
 +-----+-----+---------+------+---+----++----+---+------+---------+-----+-----+
 |     |     |    3.3v |      |   |  1 || 2  |   |      | 5v      |     |     |
 |   2 |   8 |   SDA.1 |   IN | 1 |  3 || 4  |   |      | 5v      |     |     |
 |   3 |   9 |   SCL.1 |   IN | 1 |  5 || 6  |   |      | 0v      |     |     |
 |   4 |   7 | GPIO. 7 |   IN | 1 |  7 || 8  | 1 | ALT5 | TxD     | 15  | 14  |
 |     |     |      0v |      |   |  9 || 10 | 1 | ALT5 | RxD     | 16  | 15  |
 |  17 |   0 | GPIO. 0 |  OUT | 1 | 11 || 12 | 0 | IN   | GPIO. 1 | 1   | 18  |
 |  27 |   2 | GPIO. 2 |   IN | 0 | 13 || 14 |   |      | 0v      |     |     |
 |  22 |   3 | GPIO. 3 |   IN | 0 | 15 || 16 | 0 | IN   | GPIO. 4 | 4   | 23  |
 |     |     |    3.3v |      |   | 17 || 18 | 0 | IN   | GPIO. 5 | 5   | 24  |
```

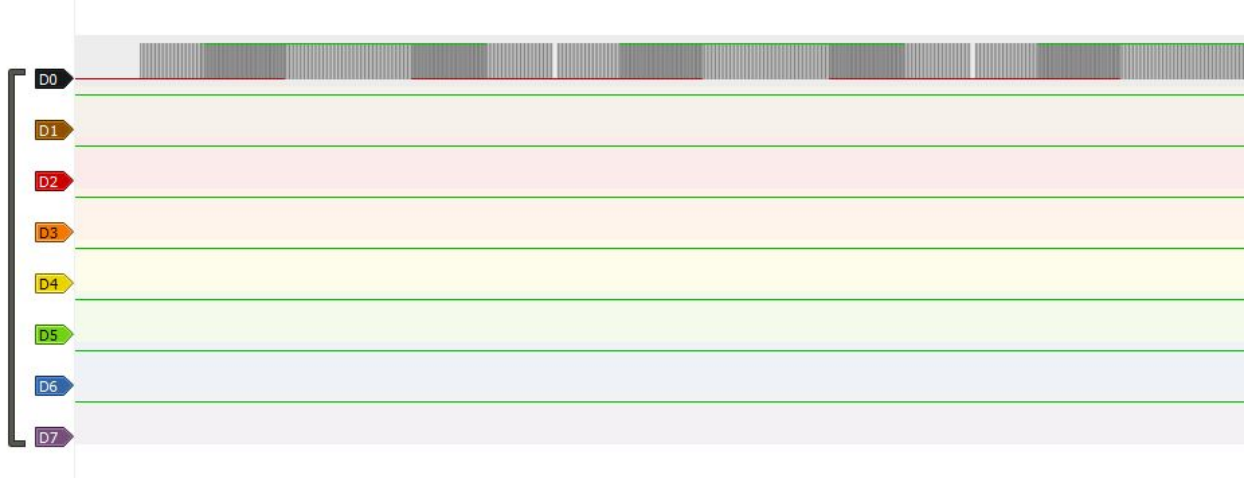
2. Instead of timing your press of the Run button on PulseView with Run on the Python script, set up a trigger condition. What is the advantage of this?

We tried several different triggers and finally got the ASCII to show up by using the falling edge trigger and increasing the sample rate to 12 MHz.

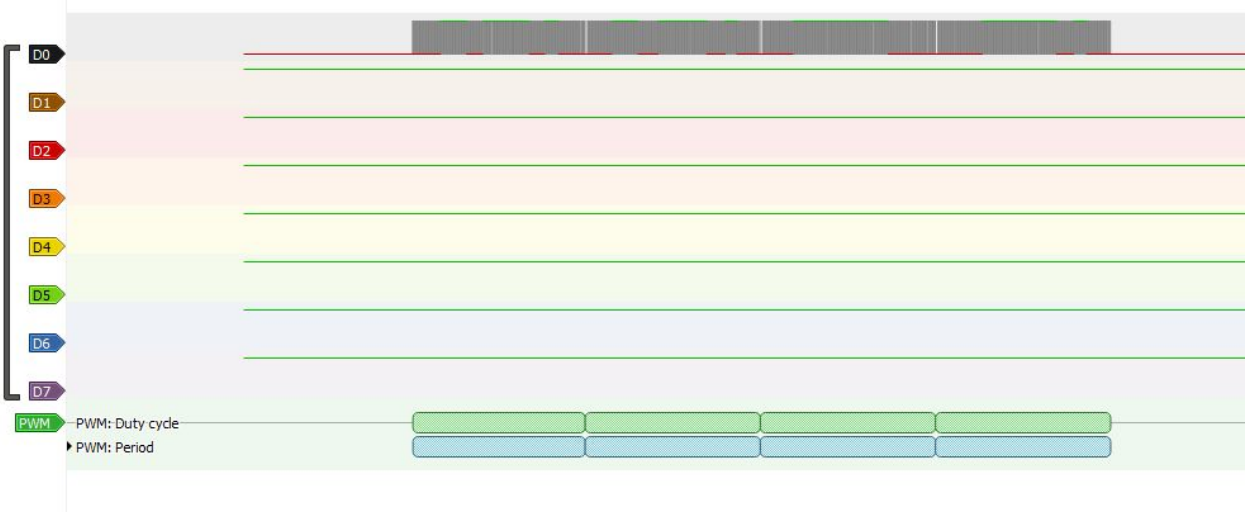


The advantage of a trigger is that it waits for the device to set it before sending any data. Before hello world is sent we do not receive any data on our screen because of the trigger.

3. Use WiringPi to setup one of the pins as a PWM output. See this example. View the resulting signal in PulseView. What can PulseView tell you about the signal?



It appears that the signal frequency increases periodically when setting the pin to PWM output. This is due to the brightness increasing and decreasing. You can see each for loop of the script because there are four distinct blocks labeled in the PWM duty cycle. The percentage goes from 0 to 100 and then resets and repeats.



In the below figure, within one of the 4 sections identified in the picture above, we see the transition of the power level up to 100 and then beginning to decrease back to 0 as the code shows.

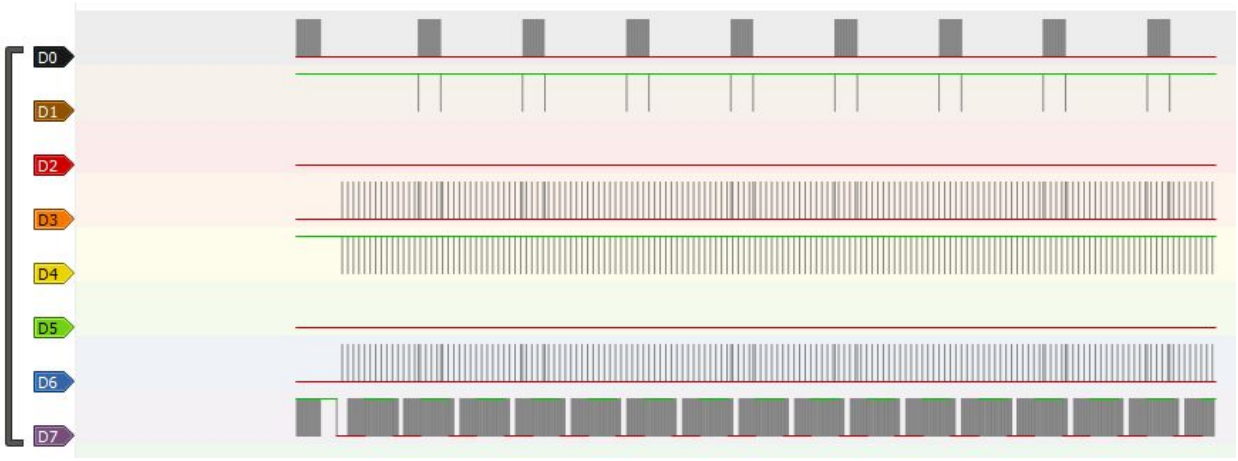
4. These examples have used serial data transmission. If you could instead send data across the pins in parallel, what would be the potential advantages and disadvantages?

Parallel data transfer is faster because there are more lines to send the data on, however parallel transmission is half duplex while serial transmission is full duplex. There are less errors and noise in serial transmission because one bit is being received at a time.

Decipher captured data

Download the le captured.sr from Canvas.
In PulseView, open (D) the le, which contains previously captured waveforms.
Use PulseView to analyze the waveforms.

1. At first investigation, how do the signals relate to each other?

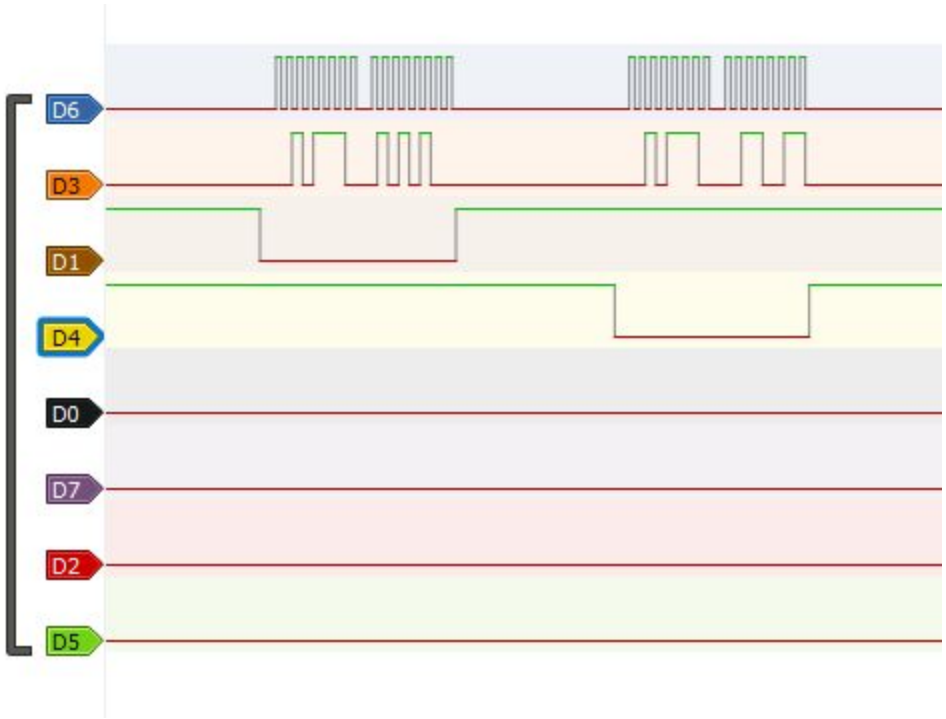


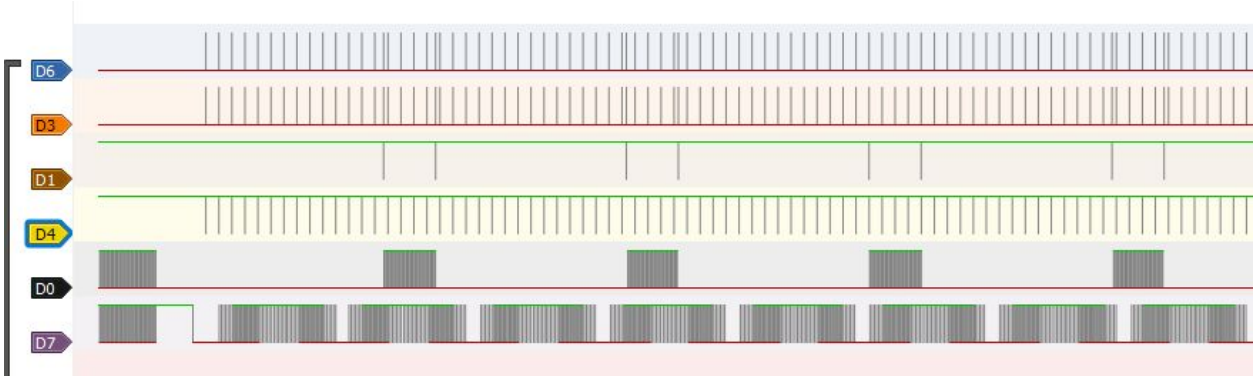
D0 and D1 have signals that appear to be in lock step with each other. D3 and D6 appear to have the same frequency and values. D7 does not appear to be related to any of the other signals.

2. Do any signals appear to be common functions, e.g., clock, enable, etc.?

D4 appears to be a clock function. D1 appears to be an enable function for D0 because when it changes value D0 can be high or low.

3. Which protocols are in use?





- The protocol is Serial Peripheral Interface (SPI)
- D6 is clock
- D3 is Master Out Slave In
- D1 is one of the slave selects
- D4 is the other slave selects
- D7 is PWM
- D0 is a timer
- Note that we don't see a master in slave out channel, indicating that the slave devices do not communicate with the master device.

4. Describe the big-picture scenario of the capture, including what kind of device(s) were in use.

- There are two devices that receive messages, but don't communicate back to the master
- The slave selector tells the slaves which slave the master is talking to

Feedback

1. This lab took us 8 hours to complete.
2. This lab overall was not too bad but the capture portion was pretty difficult. I think it would have been less difficult if we did some pulse view examples in class and talked more in depth about the various serial protocols and how they look in wave form.
3. I think the lab is an appropriate assessment, I would just incorporate the class practice alluded to in feedback answer 2.