

CMPE-240 Laboratory Exercise 02

UART Polling

By submitting this report, I attest that its contents are wholly my individual writing about this exercise and that they reflect the submitted code. I further acknowledge that permitted collaboration for this exercise consists only of discussions of concepts with course staff and fellow students; however, other than code provided by the instructor for this exercise, all code was developed by me.

Daniel Santoro
Performed: 10/03/16
Submitted : 10/03/16

Lecture Section: 01
Professor: Alessandro Sarra
TA: Kevin Millar, Adrian Cruzat, Humza Syed

Abstract

The goal of the lab is to test out a UART connection from a computer to the Raspberry Pi. UART was enabled for the Raspbian OS and then enabled for custom functions. Functions were written to allow for the reading and writing of strings. When run, the Pi outputted “Hello, World!” six times, allowed user input and returned the input in uppercase. This will allow for a user to control and debug a program with input and have visible output.

Design Methodology

The UART connection to the GPIO pins was set up in the given order. (See below figure.)



Figure 2.1 – UART Connection to GPIO

In the config.txt file of the Raspberry Pi image, UART needed to be enabled and Bluetooth needed to be disabled. (See below figure).

```
# Enable UART
enable_uart=1

# Disable bluetooth (UART and bluetooth share the same resources,
# so both can't be active at the same time)
dtoverlay=pi3-disable-bt
```

Functions for get_char, put_char, get_string, and put_string were implemented to interface with the Pi and allow inputting and printing of text. In the provided uart.c file, code was written for each of the functions for the commented sections.

Putty was used to connect over serial port via UART to the Pi. (See below figure.) To find the COM number to connect to, device manager was used (on Windows 10) and the COM number was listed under Ports.

PuTTY Configuration



Category:

- Session
 - Logging
- Terminal
 - Keyboard
 - Bell
 - Features
- Window
 - Appearance
 - Behaviour
 - Translation
 - Selection
 - Colours
- Connection
 - Data
 - Proxy
 - Telnet
 - Rlogin
 - SSH
 - Serial

Basic options for your PuTTY session

Specify the destination you want to connect to

Serial line: COM4 Speed: 115200

Connection type:

☐ Raw ☐ Telnet ☐ Rlogin ☐ SSH ☒ Serial

Load, save or delete a stored session

Saved Sessions

RP13
Gladius
NEWYORK
Phalanx CE Server
RP13
pixy
rasberypi
rasberypi-2

Load

Save

Delete

Close window on exit:

☐ Always ☐ Never ☒ Only on clean exit

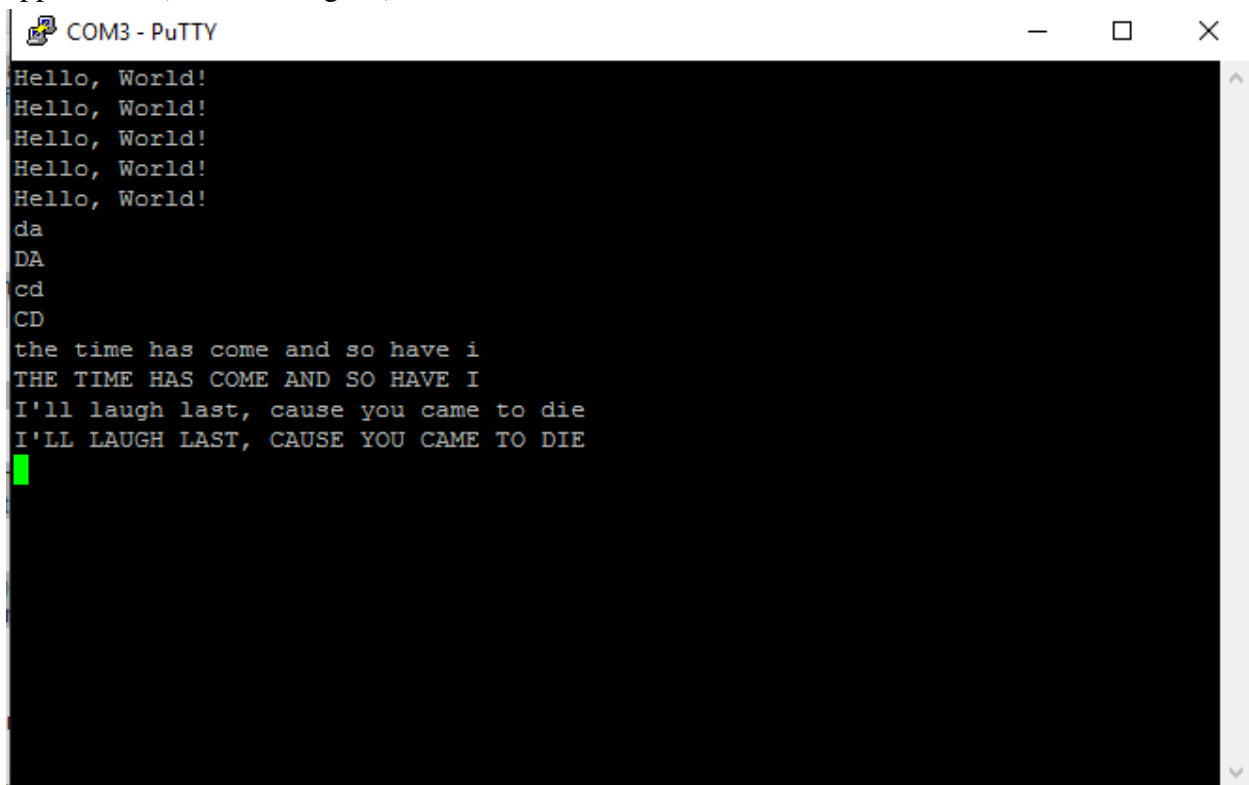
About

Open

Cancel

Results and Analysis

After the program printed out “Hello, World!” six times, text inputted would be returned in upper case. (See below figure.)



```
COM3 - PuTTY
Hello, World!
Hello, World!
Hello, World!
Hello, World!
Hello, World!
da
DA
cd
CD
the time has come and so have i
THE TIME HAS COME AND SO HAVE I
I'll laugh last, cause you came to die
I'LL LAUGH LAST, CAUSE YOU CAME TO DIE
█
```

Conclusions

The lab was successful since the purpose of the lab was completed (the ability to write via UART to the Pi and see output from the Pi).

A difficulty during the lab was when there was no output to the Putty connection. Going through the code revealed that one of the wait functions `wait_for_rx_slot` was set to `TX_FIFO_FULL` during the prelab. Another issue which was the cause of the problem was that Bluetooth was not disabled in the configuration file. After adding the disable to the configuration file, putty was able to connect to the Pi.

A terminal, via UART, could be used to connect to the Pi and have input written to a buffer on the Pi which could be used to perform debugging or other commands. In the lab, the input functionality was used to modify the input string and print the modified input to the terminal.