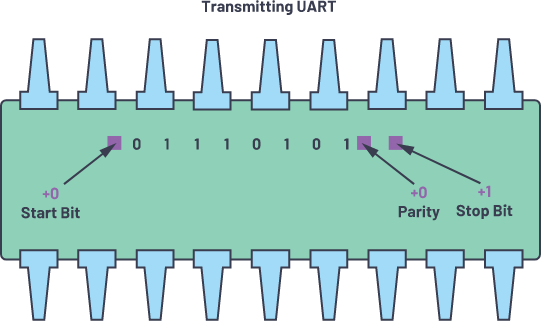
UART



Semester 3 Communication

UART Assignment

Nov 26, 2022.

Student: Andre Sanao

Course: Technology

# Introduction

In this assignment, we have to use a communication type to send and receive data. We have to program the functionality of the UART by software only without using the hardware registers. The following section will explain more in detail on how UART is implemented.

# Research

UART stands for Universal Asynchronous Reception and Transmission. It is a simple serial communication protocol that allows the host (Arduino) to communicate with serial devices. UART supports bidirectional, asynchronous and serial data transmission. It uses to pins on the Arduino which is pin 0(RX) for receiving and pin 1(TX) for transmitting. These can be connected between two devices for example two Arduino’s. UART is found on all types of Arduino boards which allows the Arduino to communicate with a computer due to its onboard USB to Serial converter.

# Design

During the implementation of UART code, we are required to use a state machine for each function. During each state the data received will go into various functions that will manipulate and verify if the received data is correct. The diagram below will show the process of how the code works. (figure 1)

Diagram

Description automatically generated

Figure 1 UART state machine diagram

# Testing

For the testing, I used 2 LED for this assignment. Red LED is for error that checks if the byte is not corrupted and the green LED is when the byte is received successfully. When a character is entered in the serial port, the data will be stored inside a variable. The variable will go into another state where the byte inside the variable will be checked for high bits. The high bits will increment a counter and go to the next state and check if the character is correct otherwise it will go into error state. The video that is attached to the assignment will give a visual demonstration of the UART assignment.

# Conclusion

To conclude this assignment, it was hard getting the right intervals in microseconds because of how fast you can make mistakes if you have codes that takes a lot of time to process. But in the end, I managed to get the right intervals even thought it took me days to get it right. Also getting the bits in the right position while debugging with Serial.print() can also delay the time and shift the bits by one.