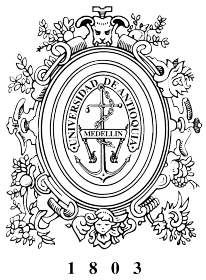


UNIVERSIDAD DE ANTIOQUIA

Facultad de Ingeniería

Informe del Parcial 1

Jeronimo Herrera Sanchez
Juan Camilo Sierra Gomez
Juan Carlos Murillo Florez

Informatica 2

Grupo Triple J

Profesores: Augusto Enrique Salazar Jimenez , Jonathan Ferney Gomez Hurtado

2 de marzo de 2022

Índice

Resumen

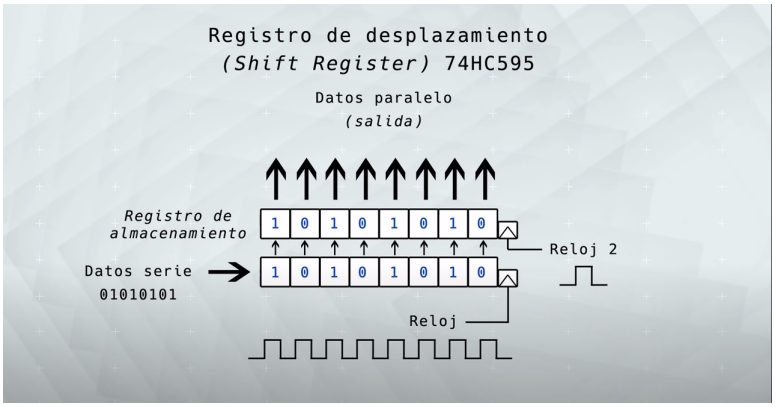
El propósito de este informe es conocer a profundidad la comunicación de dos Arduinos a su vez su implementación por medio de los puertos digitales los cuales se encargarían de enviar y recibir información de un Arduino a otro. Y con la ayuda del 74HC595 procesar esta información para después decodificarla y asi tomar la información valiosa que se quiere llevar.

1. Introducción

En el presente informe de Informática 2, referente al parcial 1 sobre la comunicación de dos Arduinos donde se puede enviar la información de manera codificada y por medio del 74HC595 se trate la información. además de dar a conocer las capacidades de los integrantes a la hora de desarrollar la solución de un problema con las herramientas y conocimientos dados en el curso de informática 2

2. Investigación del 74HC595

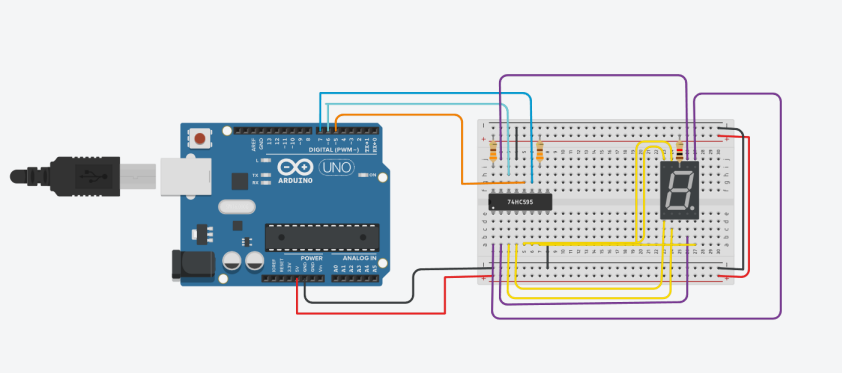
El 74HC595 permite controlar 8 salidas con tan solo 3 pines digitales, con este se puede encadenar a un segundo circuito y controlar 16 salidas con estos mismos 3 pines digitales. Este se tiene una entrada en serie y una salida paralelo de 8 bits. Con cada pulso del reloj se le dice al circuito integrado que lea la entrada y comience el proceso.[**Bitwise Ar.(6 nov 2021**]8 Al entrar los datos se desplaza a la derecha dandole espacio para el proximo bit, Pero sucede que al cargar los datos las salidas van cambiando de serie. Por esto lo ideal es que la salida no cambia hasta que se halla finalizado el problema de carga(esto lo resulve el circuito integrado con un circuito de almacenamiento. El cual posee tambien un reloj)[**CIRCUITS**, **I. (2003). 74HC595; 74HCT595.**] 8



FUENTE: Bitwise Ar.(6 nov 2021)Arduino desde cero en Español

3. Ejemplo del 74HC595

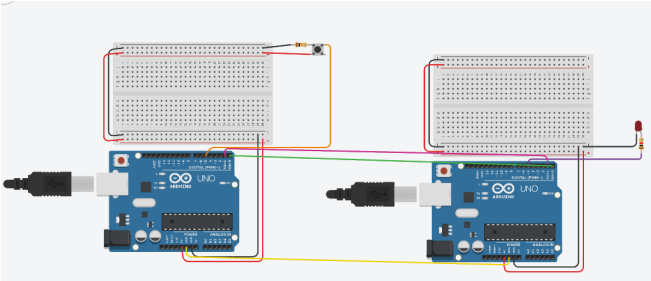
Figura 1. 74HC595



FUENTE: Autores

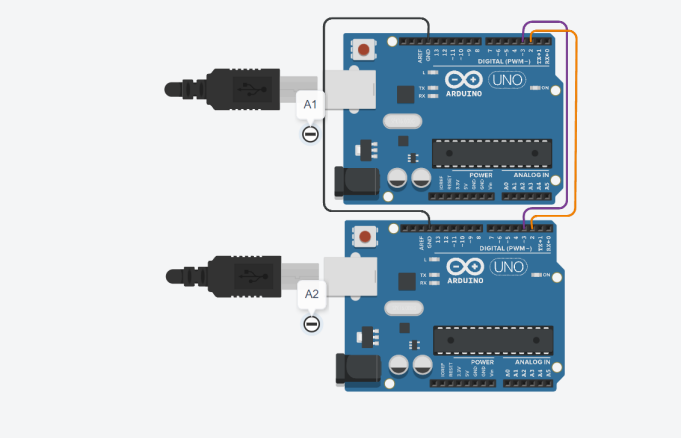
4. Ejemplos de envio de datos de un Arduino a otro Arduino

Figura 2. Arduinos pulsador



FUENTE: Autores

Figura 3. Arduinos



FUENTE: Autores

5. Resultados

En el ejemplo del 74HC595 Figura 1 se muestra el uso del chip de expansión 74HC595 para mostrar la palabra HOLA en un visualizador de 7 segmentos. Para hacer esto primero escogimos los puertos (en este caso 5,6,7), luegos creamos una variable constante tipo byte, para este caso le asignamos los bits y el orden manualmente (la idea es usar una función que lo haga) luego hacemos un ciclo que se repetirá tantas veces como la cantidad de bytes que queramos mostrar en el visor(para la palabra HOLA se repite 4 veces). Como ya sabemos o podemos ver en la documentación del chip, si se està usando un dispositivo con un reloj, debemos poner el pin del reloj en bajo con la función digitalWrite(clockPin, LOW).Luego invocamos la función shiftOut la cual desplaza un byte uno a la vez, comenzando desde el más significativo (el primero por la izquierda) o el menos significativo (el primero por la derecha). Esta función como parámetros de entrada pide; el pin de datos (este indica que barrita se va a encender), el clockPin: es un pin que estará apagado y se encenderá cuando ya se hayan pasado todos los bits, esto para que muestre completamente y al mismo tiempo el byte, lo que hace que se muestre la letra completa y no se vaya armando barrita por barrita. el bitOrder: aquí se debe escoger el orden en el que se va a desplazar los bits, si desde el más significativo (MSBFIRST) o el menos significativo (LSBFIRST). El Valor: esto es lo que queremos que se desplace (en este caso lo que queremos mostrar en el visor que sería nuestra variable constante tipo byte). Por último cabe resaltar que cuando el pin que se le pasa a la entrada del visor, es decir, el pin que representa el estado de una de las barritas está en cero (LOW), la barrita se enciende y si esta en uno(HIGH) se apaga. Es por esto que luego de invocar la función con sus respectivos parámetros de entrada ponemos el pin del reloj en alto con la función digitalWrite(clockPin, HIGH) para que se apague el visor y esté listo para la siguiente iteración y mostrar la siguiente letra.

En el primer ejemplo Figura 2 del envio de datos de los dos Arduinos podemos ver una comunicación entre dos arduinos (A1 y A2), para hacer que el primer Arduino (A1) envíe la

información al segundo Arduino lo que hicimos fue usar la librería “include ¡SoftwareSerial.h¿” la cual se ha desarrollado para permitir la comunicación en serie en otros pines digitales del Arduino. Entonces se escogen los pines o puertos a utilizar para la comunicación en este caso usamos el uno y el cero (TX= 1, RX= 0), pero se puede establecer comunicación en otros prines. Invocamos la función “SoftwareSerial mySerial(RX, TX);” que recibe como parámetros de entrada los puertos escogidos para la comunicación, primero el RX y después el TX con los valores invertidos (es decir el valor del puerto TX en el valor de RX y viceversa) para el arduino que envía y para el que recibe si lo dejamos normal sin invertir (como están conectados TX=1 y RX=0), inicializamos los puertos serial (aparte del Serial.begin(9600) se debe usar mySerial.begin(4800)), y los pines tanto del arduino que envía (inicializar el pulsador) como el que recibe (inicializar el LED). Como esta función nos permite enviar un caracter o número a la vez entonces a la hora de implementar esto con el circuito conformado por un pulsador y un LED; lo que hacemos es que leemos el puerto donde está conectado el pulsador y si este nos retorna un alto, enviamos una “H” para indicar que se está presionando (HIGH) el pulsador, de lo contrario una “L” para indicar que NO se está presionando (LOW) el pulsado. Por medio de la función “mySerial.write()” que recibe como parámetro un carácter o número, y lo envía al otro arduino, podemos saber en el otro arduino el estado del pulsador; ya que solo debemos hacer básicamente lo mismo, invocamos de nuevo la función y leemos el valor (en este caso letra que esta trajo del otro arduino) por medio de la función “mySerial.read()” la cual no lleva parámetros de entrada y retorna el valor guardado (en este caso lo guardamos en una variable tipo char llamada estado). Por último nos preguntamos si los que nos retorno la función (es decir lo que hay en estado), es igual a “H” y si lo es encendemos el LED de lo contrario si hay una “L” lo apagamos. De esta manera tenemos un LED que se enciende cuando en el otro arduino se presiona el pulsador.

Como en el ejemplo anterior en la Figura 3 pudimos ver una comunicación entre dos arduinos (A1 y A2), para hacer que desde el primer Arduino (A1) se envíe el estado de un pulsador al segundo Arduino para encender un LED, en este lo que hicimos fue básicamente lo mismo solo que en este debemos enviar un arreglo que en este caso es de enteros. Como esta función nos permite enviar un caracter o número a la vez entonces a la hora de implementar esto para enviar el arreglo; lo que hacemos es que por medio de un ciclo vamos recorriendo e indexando el arreglo en cada iteración y por medio de la función “mySerial.write()” (que recibe como parámetro un carácter o número), le ponemos como parámetro el arreglo indexado en la iteración (en este caso cop2[i]), y esto lo envía al otro arduino. Para recibir la información debemos hacer básicamente lo mismo, invocamos de nuevo la función y leemos el valor (en este caso los números del arreglo que están llegando del otro arduino) por medio de la función “mySerial.read()” la cual no lleva parámetros de entrada y retorna el valor guardado (en este caso lo guardamos en una variable tipo char llamada estado), y lo vamos guardando en un arreglo y para esto lo que hacemos es que por medio de otro ciclo vamos recorriendo e indexando el arreglo en cada iteración y a la posición indexada le asignamos el valor que nos retorna la función “mySerial.read()”. Por último para imprimir este arreglo lo que hacemos es que en cada iteración imprimimos el valor de la posición indexada luego que se le asignó su respectivo valor.

6. Retos que surgieron

A la hora del envío y recepción de datos entre los dos Arduinos surgieron bastantes inconvenientes ya que en ciertos casos no entraba en ciclos o mandaban información que no era deseada. También en la investigación que se realizó para solucionar el bloque de desencriptado y así poder clasificar la información valiosa. La pantalla LCD también genero errores de interpretación de la información que se le enviaba, en algunas ocasiones Imprimía solo el final del arreglo y en otros se repetía la información de la primera línea.

7. Conclusión

En conclusión, 74HC595 es una herramienta muy útil. con tan solo 3 salidas digitales podemos tener 8 salidas y si se juntara más de estos en serie se podría obtener el número de salidas que sean necesarias. también con esta podremos tratar la información con su registro de almacenamiento el cual entra al terminar la entrada de datos en serie y gracias a la ayuda de su reloj integrado podremos enviar la información sin ninguna perdida de bits. con los resultados obtenidos con el ejemplo del 74HC595 se pudo hacer una mejor idea de su implementación e importancia a la hora de realizar la comunicación de los dos Arduino para él envió de información. ya con los ejemplos de la comunicacion de los dos arduinos se pudo implementar la transmicion de datos para el circuito final. con la ayuda de algunas liberias se puedo mandar la informacion de un Arduino a otro. con la ayuda de algunas Librerías se puedo mandar la información de un Arduino a otro, en el primer Arduino se saca la información de un arreglo y se envía por medio de los puertos RX y TX la información y en el segundo se reconstruye lo que va llegando

8. Referencias

1. Logosímbolo UdeA
2. CIRCUITS, I. (2003). 74HC595; 74HCT595.

3. Bitwise Ar.(6 nov 2021)Arduino desde cero en Español - Capítulo 70 - 74HC595 Registro de desplazamiento (Shift register)
Youtube.https://www.youtube.com/watch?v=LFqIA3ZvZE8&t=426s&ab_channel=BitwiseAr

4. Imagen de funcionamiento del 74HC595 (Autores)
5. author = Arduino, year = 22 de febrero de 2022,
title = `https://www.arduino.cc/reference/en/`