Desafio 1

Informa2 S.A.S

Victor Manuel Jimenez Garcia
Jose Miguel Jaramillo Sanchez
Sebastian Garcia Morales

Despartamento de Ingeniería Electrónica y
Telecomunicaciones
Universidad de Antioquia
Medellín
Febrero 17 de 2022

Índice

1. Objetivos

- Aplicar los conocimientos adquiridos a lo largo del curso, demostrando apropiación de los fundamentos básicos del lenguaje de programacion C++.
- Desarrollar habilidades de investigación y redacción que permitan la adquisicion de nuevos conocimientos con el fin de solucionar problemas de la vida real.
- Demostrar la importancia y utilidad de la programación por hardware, así como el uso de módulos físicos para optimizar el uso de software en un diseño
- Diseñar un aplicativo en la plataforma de Arduino integrando programación de C++ para solucionar un desafio propuesto.

2. Introduccion

3. Marco Teorico

3.1. Conocimientos previos

A la hora de enfrentarse a un desafío lo más recomendable es dividirlos en varias etapas para trabajarlo más fácilmente, una primera etapa sería realizar una investigación de conceptos y componentes propuestos en el desafío. En este caso es necesario investigar el concepto de transferir información de forma serial y paralela cómo también identificar características, funcionalidades arquitectura, conexiones, alcances y limitaciones del circuito integrado 75HC595, por otro lado, ¿qué es un Arduino? y ¿cómo unirlo al circuito integrado mencionado anteriormente para lograr solucionar el desafío completo?.

Arduino es una plataforma de desarrollo basada en una placa electrónica de hardware libre que incorpora un microcontrolador re-programable y una serie de pines hembra. Estos permiten establecer conexiones entre el microcontrolador y los diferentes sensores y actuadores de una manera muy sencilla (principalmente con cables dupont). [1]
Este dispositivo es el que nos permitirá recibir los datos ingresados por el usuario y realizar la conversion a binario, ademas de funcionar tanto transmisor como receptor en el sistema de encriptacion.

La comunicacion entre arduinos ser realizará de forma serial, que es el proceso de enviar datos de caracter binario un bit a la vez, esto provee la ventaja de mantener la interfaz transmisor-receptor de forma simple y eficiente. [2]
Por lo tanto, para desencriptar, es necesario paralelizar dicha secuencia de bits

que luego seran las entradas de un circuito de logica combinacional encargado de comparar los datos de acuerdo a los parametros de desencriptacion.

Paralelizar no es mas que llevar la secuencia de bits que se desplazan como una sola fila, y transformarla en una columna. De esta forma si se tiene una secuencia serial de n bits, al paralelizar, el resultado es una columna de bits de n filas.

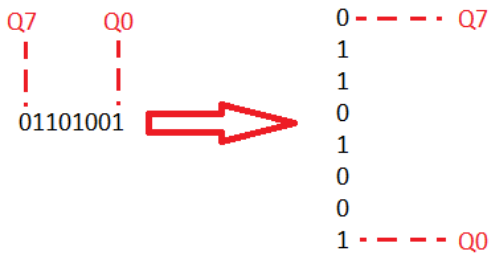

Figura 1: Ejemplo de paralelización

Esta accion de paralelizar la llevará acabo el circuito integrado 74HC595 tambien conocido como Registro de desplazamiento. Este chip de 16 pines, recibe una secuencia de 8 bits en un solo pin, y los va almacenando en cada una de las salidas para luego ser liberados como 8 señales independientes.

Dos definiciones que se deben de tener en cuenta para entender mejor el funcionamiento de todo el sistema son: comunicacion sincrona y comonicacion asincrona.

Comunicación sincrónica: Se da cuando el intercambio de mensajes sucede en tiempo real. Requiere que las dos partes (emisor y receptor) estén presentes en el mismo tiempo y espacio, ya sea físico o virtual. Por ejemplo, las llamadas telefónicas, las reuniones en la oficina o las videoconferencias.

Comunicación asincrónica: Sucede cuando los mensajes se intercambian sin importar el tiempo. Es decir, que no necesitan la atención inmediata del receptor, quien puede responder en el momento que decida o pueda hacerlo. Estamos hablando de medios como el correo electrónico, foros en línea, chats, mensajes de texto y documentos colaborativos. [3]

En este caso, la comunicacion se da de forma sincrona con ayuda de un pulso de reloj. En electrónica y especialmente en circuitos digitales síncronos, una señal de reloj es una señal usada para coordinar las acciones de dos o más circuitos, esta señal oscila entre estado alto y bajo, tambien conocido como flanco

de subida y de bajada, respectivamente, y gráficamente toma la forma de una onda cuadrada. [4]

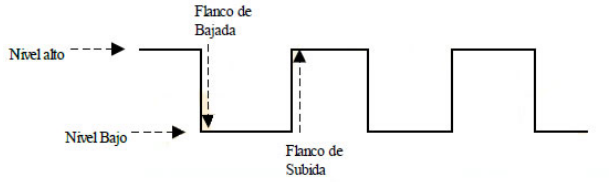

Figura 2: Diagrama de tiempo de una señal de reloj

A continuacion se muestra la distribucion de pines del circuito integrado 74HC595

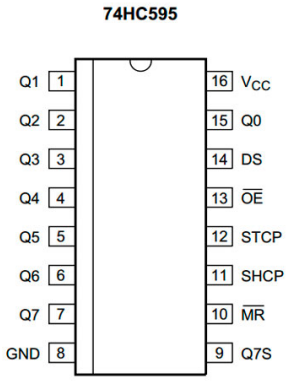

Figura 3: Pines IC 74HC595

Entradas:

GND (pin 8): conexion a tierra (0 V)
GND (pin 10): reinicio del registro (activo bajo)
SHCP (pin 11): señal de reloj
STCP (pin 12): pulso para liberar los datos
GND (pin 13): habilitar salida del registro (activo bajo)
DS (pin 14): entrada de datos serial
VCC (pin 16): conexion a fuente de voltaje (5 V)

Salidas:

Q0-Q7 (pines 1-7 y 15): salida de datos
Q7S (pin 9): salida de datos serial

El funcionamiento es el siguiente, la informacion serial entra por el **DS (pin 14)**, el integrado recibe cada bit cuando ocurre un flanco de subida por el **SHCP (pin 11)** y lo almacena en la salida de mas baja valor **Q0 (pin 15)**, a medida que van entrando mas bits, los datos que habian almacenados anteriormente se van desplazando desde **Q0** hasta **Q7** hasta completar el byte. Una vez hecho esto, se manda un flanco de subida en **STCP (pin 12)**, que se encargar de liberar los datos almacenados.

De esta forma el primer bit que entra, queda en la salida **Q7** y el ultimo en la salida **Q0**. Para ingresar un nuevo byte se debe borrar la informacion del registro, esto se hace mandando un flanco de bajada al pin **MR (pin 10)** y luego activando la salida del registro **(pin 12)**. [5]

4. Analisis del problema

4.1. Panorama del problema

El problema consiste en transferir una secuencia de bits encriptados desde un Arduino a otro de forma serial, desencriptandola antes de que llegue al receptor usando logica combinacional y un registro de desplazamiento.

Inicialmente la informacion se dará al Arduino en forma de arreglo numerico ingresandola por la consola serial del microcontrolador, este se encargará de realizar la conversion a binario y de generar los pulsos de reloj y reset necesarios para usar en el circuito integrado 74HC595.

La salida serial del Arduino, así como las señales de sincronización entraran al 74HC595 que se encargará de paralelizar los datos, entregando 8 salidas diferentes, que a su vez alimentaran las compuertas de un circuito de logica combinacional con una unica salida true o false de acuerdo a una referencia dada.

Al Arduino receptor le entrarán el reloj y los datos en forma serial, sin embargo solo admitirá aquella informacion que bajo ciertas condiciones dé como resultado un true en la logica combinacional, en otras palabras, la logica combinacional funcionará como un comparador que le dirá al receptor que informacion es correcta y cual deberá ser descartada, realizando de esta forma la desencriptacion de los datos.

En la siguiente seccion se presentarán a detalle todas las etapas de solución.

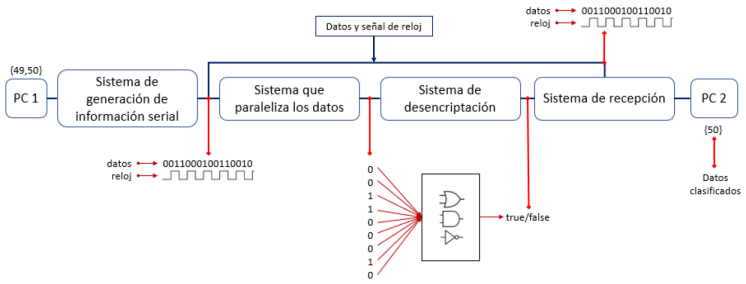

Figura 4: Esquema del sistema a implementar. Recuperado de [6]

4.2. Etapas de la solucion

Para afrontar el problema, se opta por dividirlo en distintas etapas o modulos, de forma que se pueda verificar el correcto funcionamiento de cada uno por separado. Una vez hecho esto se juntan todas las etapas para posteriormente construir el modelo final del sistema.

4.2.1. Circuito con el integrado 74HC595

Como primera etapa se revisa el funcionamiento del circuito integrado 74HC595 en el simulador Tinkercad con ayuda de leds, botones y suiches.

Para la alimentacion se usa una fuente de voltaje de 5 V, se realizan las respectivas conexiones de forma que cada led represente una salida del integrado y por facilidad se realiza el montaje solo para 4 bits. Las funciones de reloj, datos, y liberacion de datos se realizan con suiches y botones.

Para ingresar un dato se usa el boton reloj, que permitirá la entrada de un 1 o un 0, de acuerdo a la posicion que tenga el suiche deslizante (la izquierda representa 1 y la derecha 0). Luego de tener 4 datos ingresados se presiona el boton reloj de registro, mostrando los datos ingresados en los leds.

Una vez montado el circuito, se evidencia que, aunque su comportamiento es el esperado, no se permitía ingresar mas información nueva ni borrar la existente, esto se debía a que no se había agregado un boton de reset en el pin 10 del integrado. Esta correccion ya se muestra en el circuito de la Figura 5.

Despues de haber garantizado el funcionamiento con botones y suiches, se procede a reemplazarlos por las salidas digitales del Arduino, en este caso el reset, el reloj de registro, la entrada de datos y el reloj estan conectados a los pines 4, 5, 6 y 7 respectivamente. Tambien se usan leds adicionales para llevar control de los pulsos que salen del Arduino como puede verse en la Figura 6.

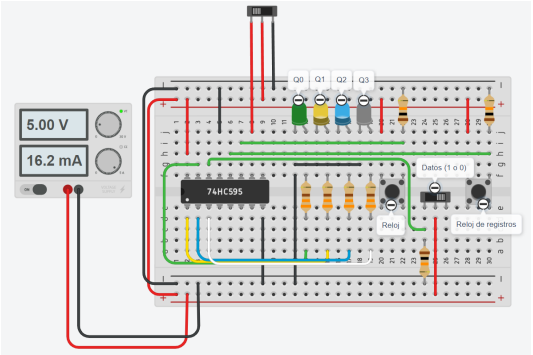

Figura 5: Primer montaje con el 74HC595

El pulso de reloj principal es implementado internamente en el Arduino con ayuda de un ciclo y delays. Los datos a transmitir provienen de un arreglo de unos y ceros que es recorrido con un for y cuyo valor solo es liberado cuando ocurre un flanco de subida del reloj. El reloj de registros tendrá un periodo de 4 veces el periodo del reloj principal, pues para este caso se está trabajando solo con 4 bits.

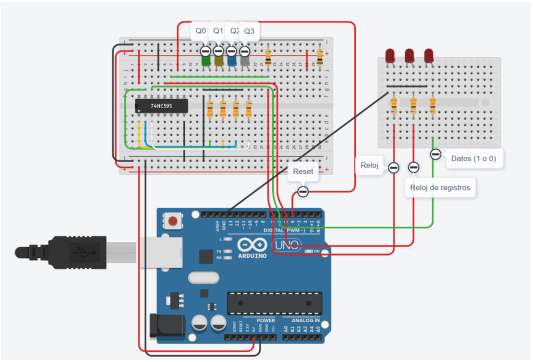

Figura 6: Montaje con 74HC595 y Arduino

4.2.2. Comunicacion serial entre arduinos

Para enviar la informacion se usan dos pines digitales en cada Arduino, configurándolos como pines de entradas y salida para el receptor y transmisor respectivamente. Uno de los pines es para el reloj principal y el otro para los datos en forma serial. Los pulsos de reloj son generados internamente en el Arduino transmisor con ayuda de un ciclo y diferentes delay.

En el Arduino receptor se usa un condicional que detecte si hay un flanco

de subida en el reloj principal, y si lo hay se imprime en la consola serial 1 o 0 segun el valor que haya en el pin de datos, ya sea un HIGH o un LOW. La impresion por consola es solo una medida de controlar el correcto funcionamiento del montaje.

Durante la implementación se presentó la dificultad de que el receptor tomaba varias veces el mismo valor debido a que la función loop se repite mucho más rápido de lo que cambian el reloj. Para solucionarlo, se redujo drásticamente el tiempo en alto del reloj hasta 0.5 ms, de forma que el receptor pueda recibir solo un dato a la vez y no varias veces el mismo dato generando errores. El montaje se muestra en la Figura 7.

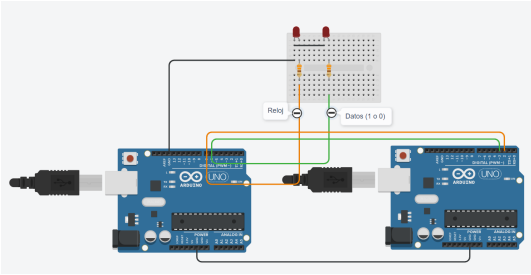

Figura 7: Montaje de Arduinos en comunicacion serial

5. Conclusiones

Referencias

[1] Arduino.cl. ¿qué es arduino? [Online]. Available: https://arduino.cl/que-es-arduino/

[2] O. Weis. Guía completa de especificaciones del puerto serie. [Online]. Available: https://www.virtual-serial-port.org/es/article/what-is-serial-port/

[3] E. editorial de Indeed. (2021, noviembre) Diferencias entre comunicación sincrónica y asincrónica. [Online]. Available: https://mx.indeed.com/orientacion-profesional/desarrollo-profesional/diferencias-comunicacion-sincronica-asincronica

[4] injgonzalez. (2011) El concepto de señal de reloj. [Online]. Available: https://sistemasdigitales2.files.wordpress.com/2011/11/semana3.pdf

[5] NXP. 74hc595 datasheet (pdf) - nxp semiconductors. [Online]. Available: https://pdf1.alldatasheet.com/datasheet-pdf/view/546557/NXP/74HC595.html

[6] A. S. Jimenez and J. F. G. Hurtado, “Desafio 1. informa2 sas,” 2022.