Departamento de Computación,
Facultad de Ciencias Exactas y Naturales,
Universidad de Buenos Aires

Proyecto Final: Interruptor Digital

Diseño de Sistemas con FPGA

Primer Cuatrimestre de 2015

Alumno	LU	E-mail
Germán Augusto Romano	786/11	romano.german@live.com.ar

Índice

1. Introducción

En este informe se presenta el proyecto final de la materia *Diseño de Sistemas con FPGA*: un interruptor digital. El mismo brinda una protección por clave numérica del encendido/apagado de aquel dispositivo que tenga conectado a su salida.

El proyecto fue desarrollado sobre una placa *Digilent Nexys 3*, que cuenta con una FPGA *Xilinx Spartan 6*.

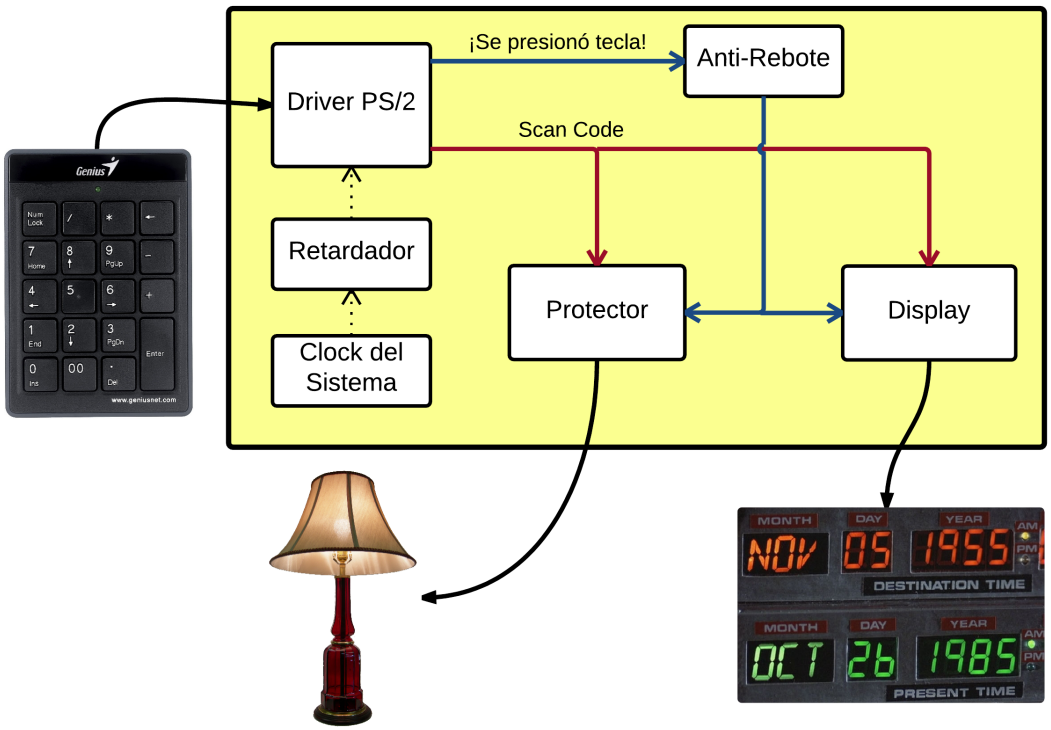
En las siguientes secciones se detallan las características generales del sistema, así como cada uno de sus componentes principales. También se deja en claro la organización del código fuente adjunto y algunas posibles extensiones al sistema.

2. Vista General

El ingreso de la clave al sistema se realiza mediante un teclado *Numpad USB*, para evitar, por ejemplo, los inconvenientes que traería tener que ingresar la clave a través de los pulsadores de la placa.

Los dígitos se muestran, a medida que son ingresados, en el display de la placa (4 componentes de siete segmentos).

El siguiente diagrama muestra los módulos que componen al sistema y sus interacciones:



- **Clock del Sistema** representa a la señal de clock de 100 MHz que brinda la placa.
- El módulo **Retardador** baja la frecuencia de la señal de clock del sistema a la mitad, dado que el Driver PS/2 funciona a 50 MHz.
- El **Driver PS/2** permite interpretar los comandos que envía el teclado, codificados originalmente mediante el protocolo PS/2.

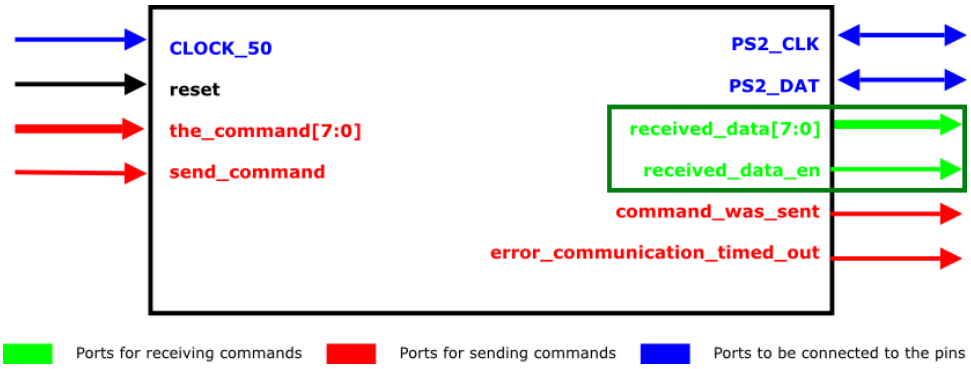
- El módulo **Anti-Rebote** evita las señales "fantasma" que suelen acompañar el pulsado de un botón mecánico.
- **Display** se encarga de hacer visibles los dígitos ingresados, mediante el display de siete segmentos de la placa.
- El módulo **Protector** tiene por tarea comparar la clave ingresada con la clave almacenada, y activar o desactivar la salida protegida según corresponda.

3. Componentes Principales

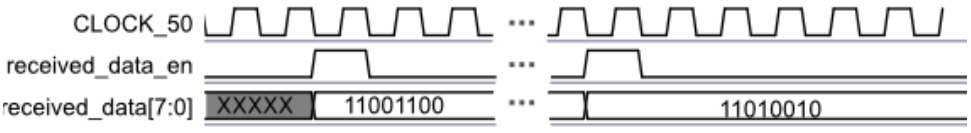
A continuación, se explica más detalladamente el funcionamiento de cada uno de los principales componentes del sistema:

3.1. Driver PS/2

El Driver PS/2[2] permite interpretar fácilmente los comandos que ingresan a través del teclado sin tener que lidiar con el protocolo PS/2. Tiene la capacidad de manejar una comunicación bidireccional; sin embargo, en este proyecto sólo se utiliza la sección encargada de los datos entrantes (encuadrada en verde). Este driver brinda la siguiente interfaz:



- **CLOCK_50** es el puerto por donde entra la señal de clock del sistema, luego de bajar su frecuencia a la mitad mediante el módulo *Retardador*.
- La señal **reset** está conectada a uno de los pulsadores de la placa, mediante el cual se reinician las operaciones del módulo.
- Las señales **PS2_CLK** y **PS2_DAT** son las que corresponden al bus de comunicación PS/2.
- Los puertos de comunicación saliente, marcados en rojo, no son relevantes a los alcances de este proyecto.
- Los *Scan Code* de las teclas presionadas se leen mediante el puerto **received_data**. Dicha señal consta de 8 bits.
- La señal **received_data_en** se encontrará en alto durante un ciclo de clock cada vez que se reciba un nuevo dato a través del puerto anterior. Dicha interacción se detalla en el siguiente diagrama:



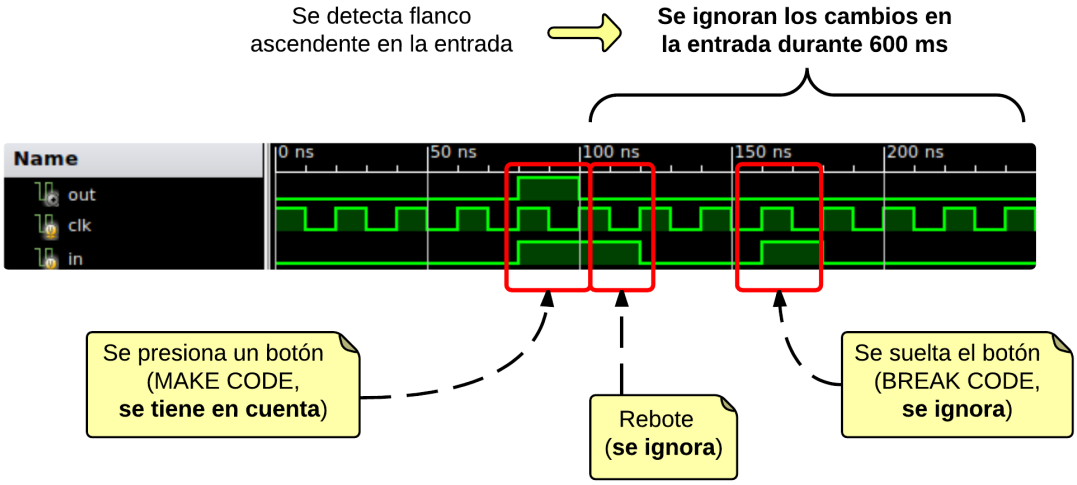
3.2. Módulo Anti-Rebote

Como es habitual, la interacción con dispositivos mecánicos (en este caso las teclas del *Numpad*) trae aparejadas ciertas complicaciones, como lo son las señales de rebote. Esto es, al pulsar una tecla se registran en realidad varias pulsaciones, debido a sutilezas del dispositivo o al pulso de quien presionó la tecla.

Además, el protocolo de comunicación de los teclados consiste en enviar el *Make Code* de la tecla que se presiona y el *Break Code* de la misma una vez que se soltó. Para el ingreso de la clave sólo nos interesa conocer los *Make Code* de las teclas, pudiendose interpretar los otros códigos como falsas pulsaciones.

El módulo *Anti-Rebote* resuelve dichas complicaciones. Consiste en una simple FSM que, cuando registra un flanco ascendente en la señal de entrada, pone en alto la señal de salida durante un único ciclo de clock e ignora los cambios en la señal de entrada durante cierta cantidad de ciclos de clock, evitando de esta manera transmitir las señales de rebote y los *Break Code*. Dicha cantidad de ciclos de clock fue calculada para evitar estas señales durante un lapso de medio segundo, aproximadamente.

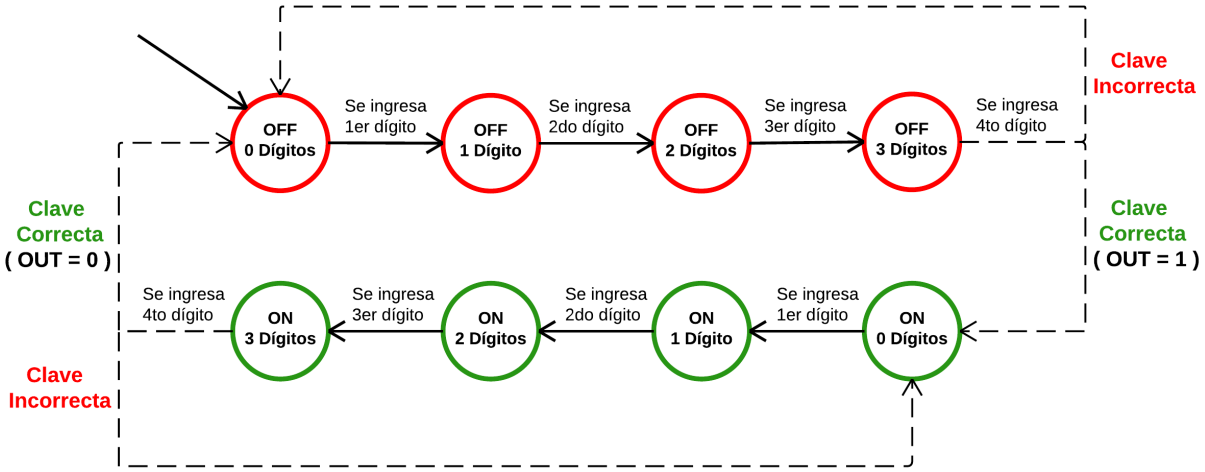
El siguiente diagrama muestra el comportamiento de este módulo:



3.3. Protector

El módulo *Protector* es el encargado de registrar los dígitos que se van ingresando y de verificar la correctitud de la clave ingresada, comparándola con la clave almacenada en el sistema. Si se trata de la clave correcta, procede a cambiar el estado de la señal de salida. De lo contrario, la salida del sistema se mantiene en su estado y el módulo queda a la espera de un nuevo ingreso de la clave.

La siguiente FSM muestra el comportamiento de dicho módulo:

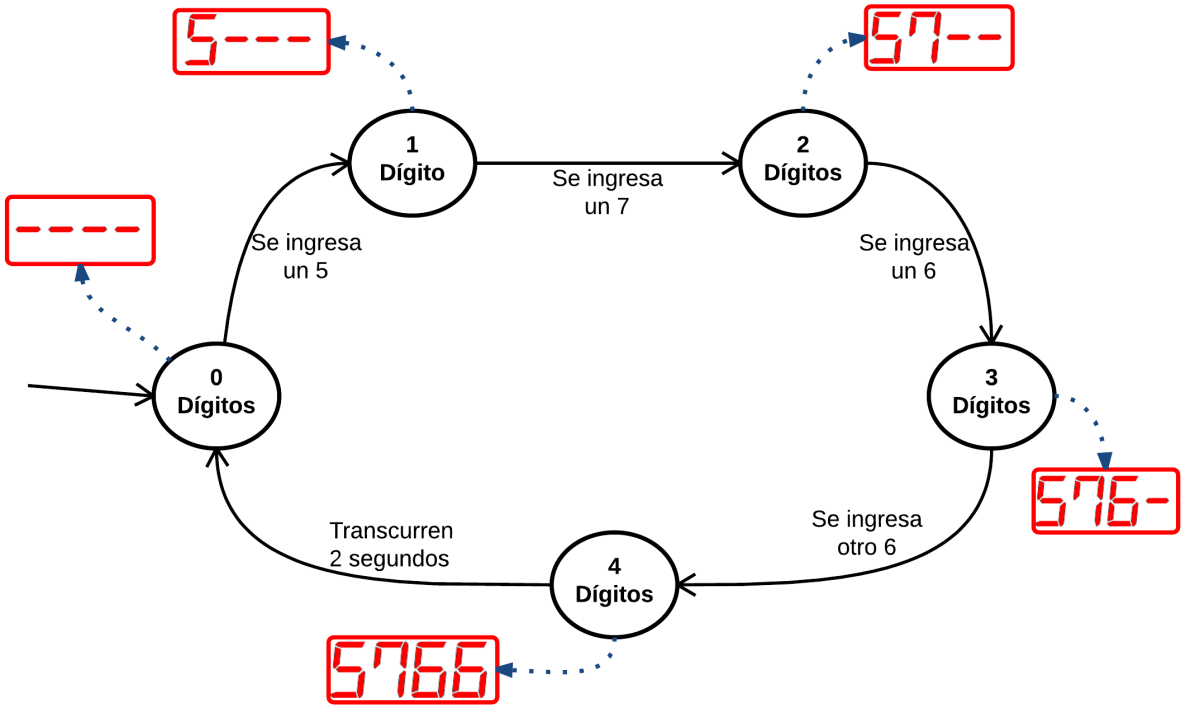


3.4. Display

Este módulo, que funciona de manera totalmente independiente al módulo *Protector*, permite visualizar los dígitos ingresados mediante el teclado. Obtiene los *Scan Code* mediante la señal *received_data* del *Driver PS/2* y los pasa al display de siete segmentos de la placa, convirtiendolos antes a un formato legible por dicho componente.

Una vez ingresados los cuatro dígitos, el módulo los muestra por pantalla durante aproximadamente dos segundos y procede a limpiar la misma, mostrando cuatro guiones.

A continuación se muestra el comportamiento del módulo al ingresar los dígitos 5766:



4. Código Fuente

Cada uno de los módulos se encuentra en una carpeta con su nombre. Allí se incluyen los archivos *Verilog* ".v" (tanto los módulos en sí como archivos de test, en algunos casos), los proyectos ".xise" con sus dependencias y los archivos en código máquina ".bit".

Los archivos correspondientes al *Driver PS/2* se encuentran en la carpeta *ps2/PS2_Controller/*.

El proyecto principal (es decir, aquel que contempla en su totalidad al sistema aquí descripto), se encuentra en *protector/protector.xise*.

5. Trabajo Futuro

A continuación se proponen algunas posibles mejoras/extensiones al proyecto:

- Seteo manual de la clave. Esto evitaría tener que modificar el código fuente y recompilar el sistema para cambiar la clave.
- Agregar un tiempo de bloqueo tras cierta cantidad de intentos fallidos, para dificultar los ataques por fuerza bruta.
- Conectar la salida del sistema a un relé. Esto permitiría controlar prácticamente cualquier tipo de dispositivo: lámparas, cerraduras electrónicas, equipos de alta potencia, etc.

Referencias

[1] Pong P. Chu, FPGA Prototyping By Verilog Examples, 2008.

[2] Jonathan Rose - University of Toronto, Sitio Web Personal, `http://www.eecg.toronto.edu/~jayar/ece241_08F/AudioVideoCores/ps2/ps2.html`