# Memoria LCSE

## Contenido

# 1. RS232top

## 1.1 Registro de desplazamiento (Shift Register)

El bit que se recibe por una línea serie se va almacenando en una byte de la longitud deseada (en este caso 8 bits), para después ser entregado paralelamente, es decir, la palabra entera al dispositivo receptor.

## 1.2 Módulo TX

Para el modulo de transmisión se dispone de la palabra de datos de entrada; se está en estado *Idle* hasta cuando se recibe el valor alto de la variable *START*, entonces se pasa a estado *StartBit* y se comienza a mandar un bit, de valor '0', por la línea de transmisión a nivel bajo, indicando inicio de la trama de datos. Tras esperar el ancho de bit necesario, la maquina se mueve a *SendData* y se mandan los bits de datos de menos significativo a mayor significativo, esperando de nuevo un tiempo de ancho de bit calculado con *pulse_width* entre dato y dato; y aumentando el contador de *data_counter* para recorrer la palabra. Cuando se llega a la última posición del dato, se pasa al estado *StopBit* y se manda un último bit a nivel alto indicando la finalización de la trama. Entonces, de nuevo se espera en estado *Idle*.

Máquina de estados tipo Moore del TX:

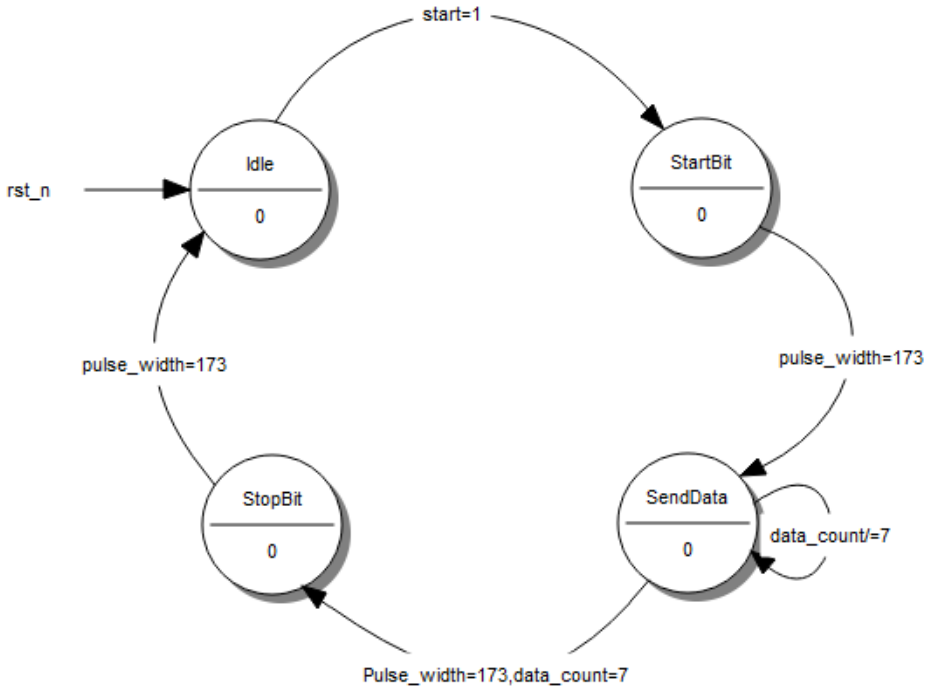



*Imagen 1.*

## 1.3 Modulo RX

Cuando la línea de datos contiene un nivel bajo, se salta al estado *StartBit*; cuando transcurren un ancho de bit, calculado con contador al igualarse a *bitCounter* se pasa al estado *RcvData*, estado en el cual se muestrea el valor de la línea de entrada *LineRD*_in en la mitad de su ancho de pulso, y se propaga directamente a la línea de salida (línea que en el conjunto final entra al registro de desplazamiento); se repite el proceso 8 veces, y entonces se salta al último estado *StopBit*. Si en el bit de parada se encuentra un nivel alto, (al no estar usando paridad) se supone que la palabra entera es correcta y entonces se habilita *valid_D* y se pasa de nuevo a estado *Idle*.

Máquina de estados tipo Moore del RX:

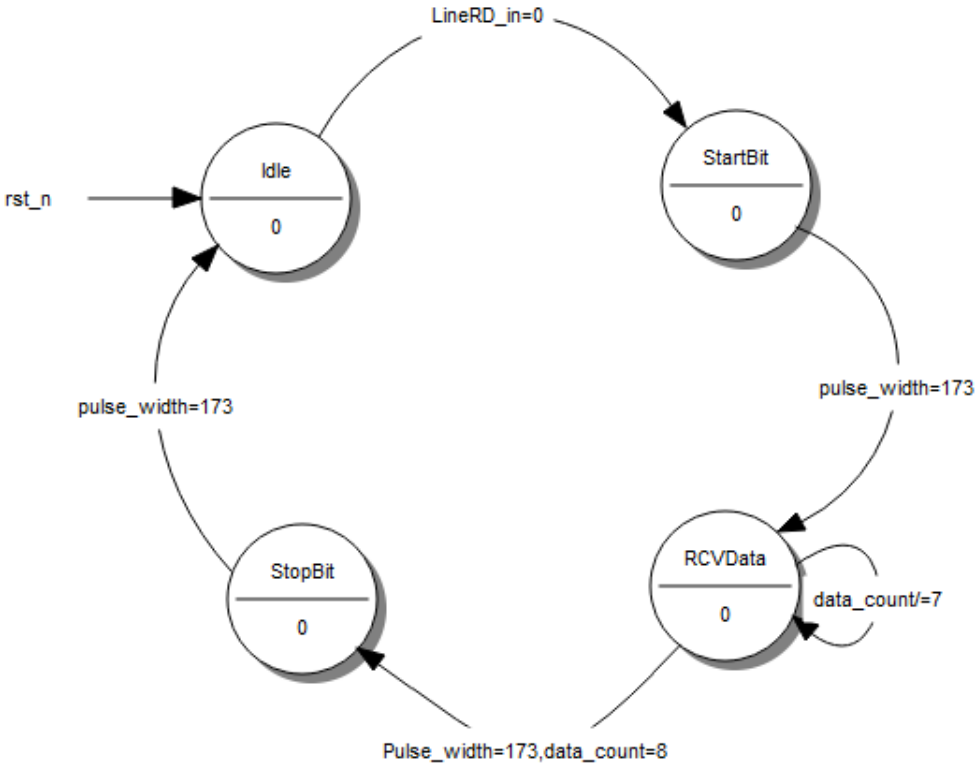



*Imagen 2.*

## 1.4 Reloj

Se ha instanciado una IP tipo clock _generator llamada *Clk_Gen*, en la cual tiene como entrada el reloj de la FPGA de 100 MHZ, obteniéndose como salida un reloj a 20 MHz que será dispuesto a todo el sistema restante.

## 1.5 FIFO

Se ha instanciado una IP de memoria tipo FIFO de 8 bits y 16 posiciones. La salida más importante que se ha de tener en cuenta es la que indica que la memoria no está vacía, *RX_EMPTY*.

## 2. RS232_DMA_RAMtop

El siguiente gran grupo de bloques es el dado por la terna: bloque RS232, bloque RAM y bloque DMA. En este punto el sistema es capaz de localizar datos en la memoria RAM y enviarlos por el canal de transmisión, así como recibir y descifrar datos llegados por la línea de recepción y guardarlos en las posiciones de memoria dedicadas a ello.

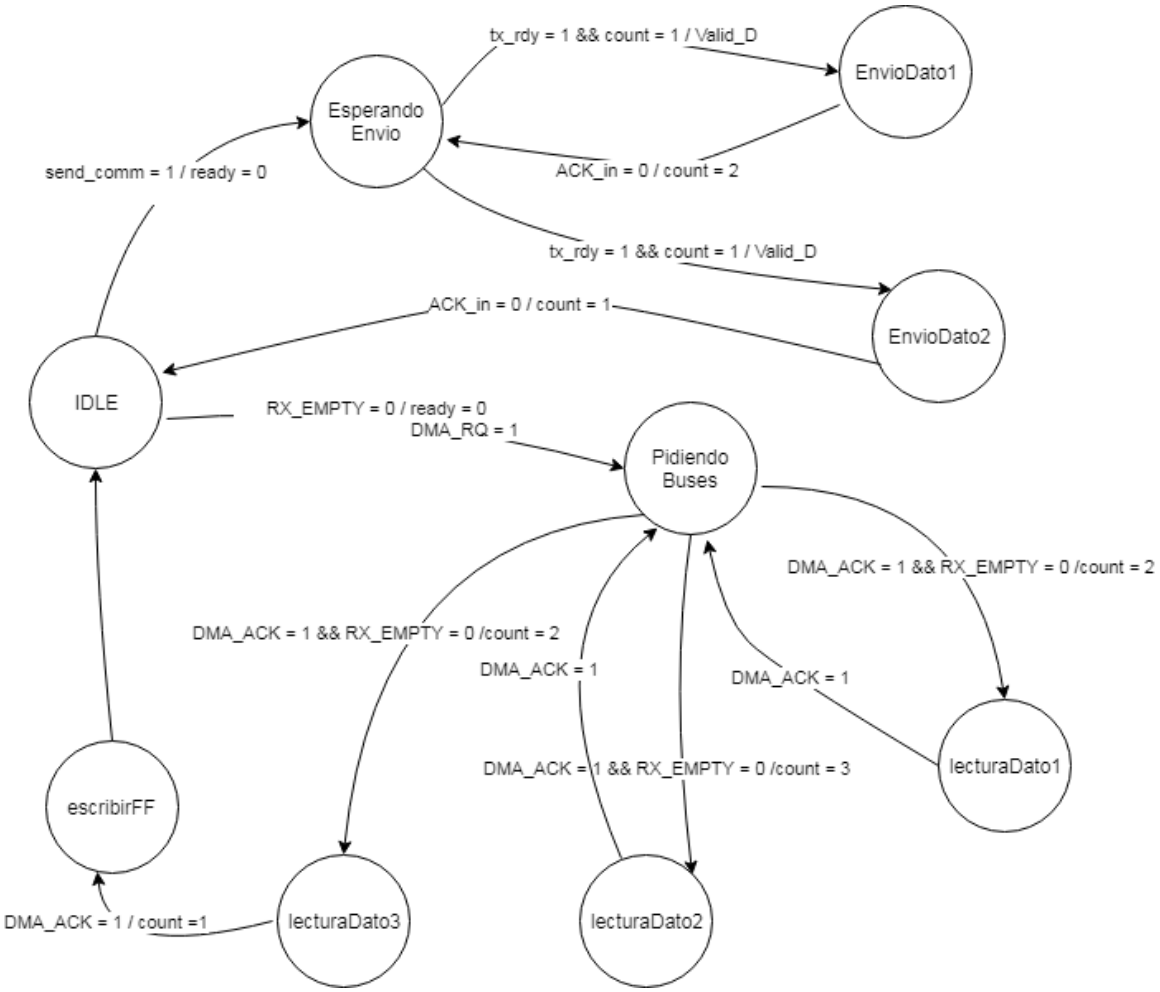
### 2.1 RS232top

*Bloque anteriormente explicado aquí.*

### 2.2 DMA

Cuando la DMA recibe un nivel bajo de *RX_EMPTY* desde el bloque de recepción, la DMA pide buses para poder colocar en la RAM el dato que se acaba de recibir, esto ocurrirá en grupos de tres bytes puesto que son el tamaño de instrucciones que se esperan en el sistema completo. En cambio si la unidad de control principal desea enviar datos, la DMA recibirá un nivel alto de *Send_comm*, lo que hace que entre en los estados de enviar datos, enviando y esperando la respuesta satisfactoria del RS232.

Máquina de estados de la DMA:



## 2.3 RAM

Módulo ofrecido por el profesor de la asignatura al cual se le han hecho mínimas modificaciones. Las necesarias para disponer de todas las direcciones de memoria que se piden a continuación:

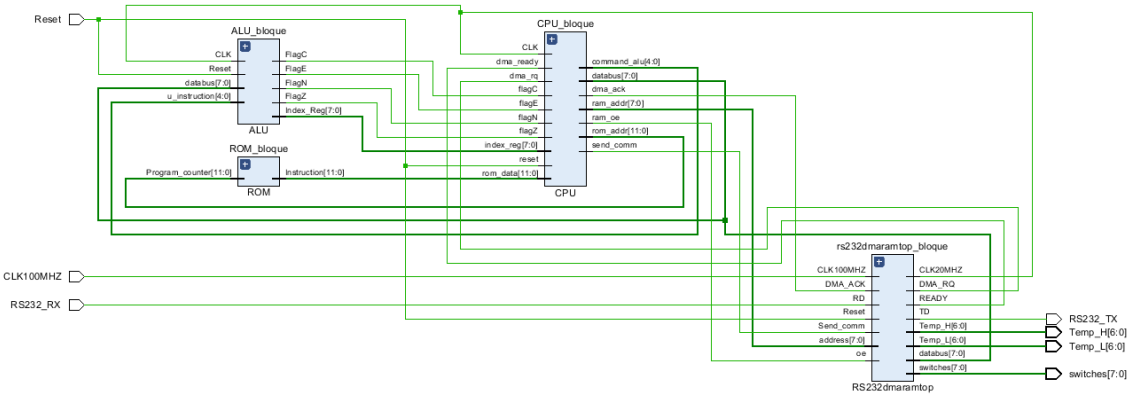
| Dirección | Alias | Función |
|---|---|---|
| 0x00 | DMA_RX_Buffer (MSB) | Byte más significativo de la reserva para el controlador DMA (recepción) |
| 0x01 | DMA_RX_Buffer | Byte intermedio de la reserva para el controlador DMA (recepción) |
| 0x02 | DMA_RX_Buffer (LSB) | Byte menos significativo de la reserva para el controlador DMA (recepción) |
| 0x03 | NEW_INST | Flag que indica la llegada de un nuevo comando por la línea serie |
| 0x04 | DMA_TX_Buffer (MSB) | Byte más significativo de la reserva para el controlador DMA (transmisión) |
| 0x05 | DMA_TX_Buffer (LSB) | Byte menos significativo de la reserva para el controlador DMA (transmisión) |
| 0x06 … 0x0F | Reservado | Para posterior ampliación |
| 0x10 … 0x17 | SWITCH(0..7) | Zona de control de interruptores |
| 0x18 … 0x1F | Reservado | Para posterior ampliación |
| 0x20 … 0x29 | LEVER(0..9) | Zona de control de actuadores |
| 0x2A … 0x30 | Reservado | Para posterior ampliación |
| 0x31 | T_STAT | Temperatura fijada en el termostato |
| 0x32 … 0x3F | Reservado | Para posterior ampliación |
| 0x40 … 0xFF | GP_RAM | Memoria de propósito general |

Así como recogida de datos de los switches y el manejo los displays de 7 segmentos de la placa.

# 3. PICtop

Estructura que alberga todos los bloques del sistema:

- RS232TX
- RS232RX
- ShiftRegister
- FIFO
- Clock_generator
- DMA
- ALU
- RAM
- ROM
- CPU




## 3.1 RS232_DMA_RAMtop

*Bloque anteriormente explicado aquí.*

## 3.2 ALU

Bloque que responde a la señal *u_instruction* proveniente de la CPU, la cual le indica que operación debe realizar. Trabajo simplificado gracias a la librería *PIC_pkg.vhd* que recoge la mayoría de variables necesarias ya definidas así como el tipo *alu_op* que contiene todos los valores de la señal *u_instruction.*

## 3.3 CPU

Bloque que contiene la maquina de estados para recoger las instrucciones de programa recorriendo la ROM y ejecutando saltos dentro de ella si es necesario y los comandos necesarios para gobernar la DMA y la ALU.

## 3.4 ROM

Bloque entregado con los archivos del profesor.