TP2: ALU-UART

- Luna, Lihué Leandro
- Bonino, Francisco Ignacio

Introducción

El objetivo de este trabajo práctico es el de implementar una interfaz de comunicación UART para la unidad aritmético-lógica (ALU) desarrollada anteriormente.

Esta comunicación serie tiene como propósito el poder brindarle a la ALU los operandos y el operador por puerto serie (RX), y que el resultado sea transmitido por la misma vía de comunicación (TX).

Módulo UART

La función principal del módulo de comunicación serie de la PC es la de transmitir individualmente los bits de cada byte, y el módulo UART desarrollado es el que se encarga de recibirlos en una cola FIFO de recepción (FIFO-RX), reensamblar los bits para formar los bytes completos nuevamente, y enviar esta información a la ALU.

Una vez la ALU realiza la operación, el resultado es enviado a la cola de transmisión (FIFO-TX) y luego enviado por puerto serie a la PC que estará esperando estos datos.

La máquina de estados es representada por un módulo de interfaz entre la ALU y el módulo UART propiamente dicho.

Diagrama de bloques

Se muestra a continuación el diagrama de bloques que representa cómo se relaciona la comunicación UART de la PC con el módulo UART desarrollado y la ALU:

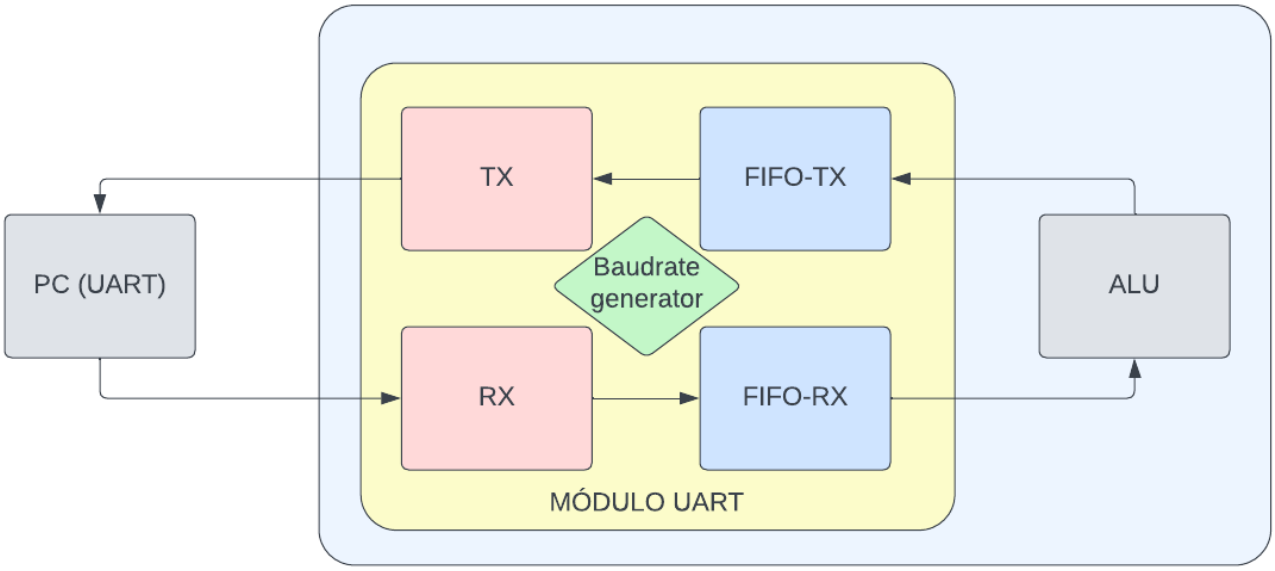


Diagrama de bloques para el módulo UART y la ALU

Tasa de baudios

Para lograr una fiable comunicación serie a 19200Bd teniendo un clock seteado a 50MHz, hubo que buscar el factor de división, que se logra mediante la fórmula:

DVSR = Clk_f / (16 * Bd)

En nuestro caso:

- Clk_f = $50*10^6$
- Bd = 19200

De este cálculo obtenemos que el parámetro DVSR para el módulo UART debe ser aproximadamente 163.

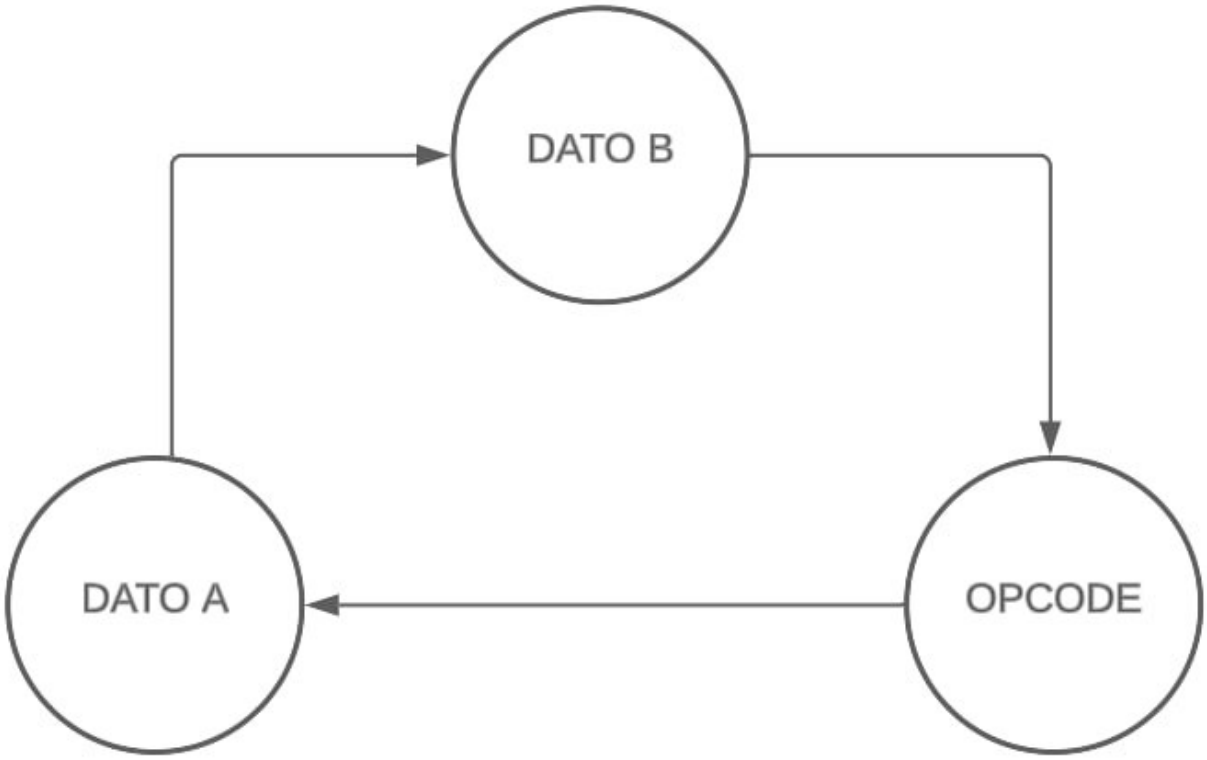
Máquina de estados

La máquina de estados implementada para este trabajo fue pensada para que sea sencilla, conteniendo así 3 estados:

- DATO A: En este estado, la máquina de estados está esperando recibir el primer dato de la PC, que representa el valor de entrada A para la ALU. En este estado, rd_signal se establece en 1, lo que indica a la PC que está lista para recibir datos. Cuando se recibe el dato de la PC, se almacena en la variable value_a, y luego el estado cambia al siguiente estado, que es DATO B.
- DATO B: En este estado, la máquina de estados está esperando recibir el segundo dato de la PC, que representa el valor de entrada B para la ALU. Al igual que en el estado anterior, rd_signal se establece en 1 para indicar que está listo para recibir datos. Cuando se recibe el dato de la PC, se almacena en la variable value_b, y luego el estado cambia al siguiente estado, que es OPCODE.
- OPCODE: En este estado, la máquina de estados está esperando recibir el tercer dato de la PC, que representa el código de operación (opcode) que se utilizará para realizar una operación específica en la ALU. Al igual que en los estados anteriores, rd_signal se establece en 1 para indicar que está listo para recibir datos. Cuando se recibe el dato de la PC, se almacena en la variable opcode. Luego, el resultado de la operación ALU se guarda en la variable o_result, y se coloca en la variable tx_data para enviar de vuelta a la PC. Finalmente, wr_signal se establece en 1 para indicar que está listo para enviar el resultado a la PC, y el estado vuelve al estado DATOA para esperar el próximo conjunto de datos.

Este proceso se repite continuamente mientras la interfaz UART esté activa y haya datos disponibles para enviar y recibir.

Gráficamente:



Máquina de estados diseñada

Interfaz gráfica

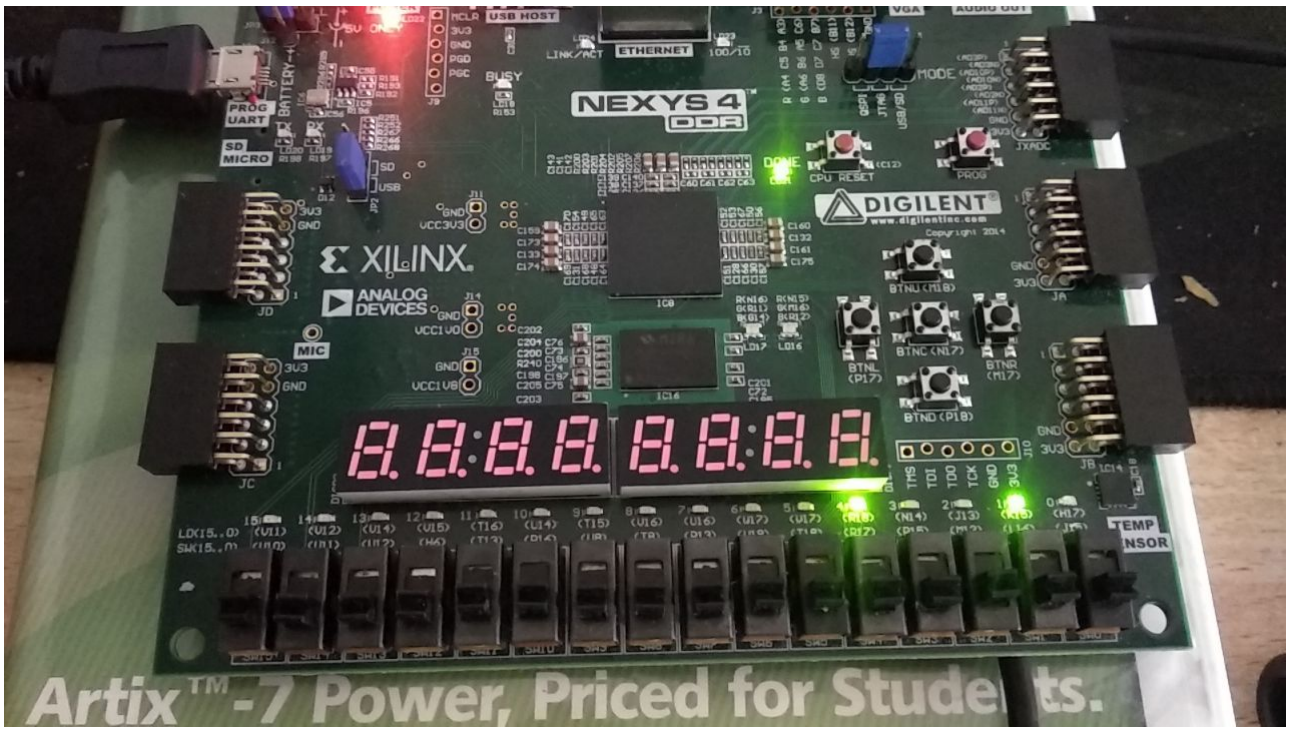
Se optó por desarrollar una interfaz gráfica (GUI) en Python para abstraerse más aún de la conexión serie y que el manejo sea . Esta GUI consta de un campo donde se ingresa el puerto a conectarse (COMx en caso de usar Windows), dos campos donde se deben ingresar los operandos a manipular, y un campo de selección desplegable en el cual figuran las operaciones soportadas por la ALU. Estas operaciones son:

- ADD
- SUB
- AND
- OR
- XOR
- NOR
- SRL
- SRA

Una vez la conexión serie se haya establecido y se hayan ingresado los operandos y el operador, se podrá enviar estos datos mediante un botón etiquetado como "GO!", el cual se encarga de enviar cada dato por puerto serie a la placa (habiendo convertido previamente el código de operación a su equivalente en binario para que la ALU pueda reconocerlo correctamente), quedando a la espera del resultado que devolverá la ALU mediante el módulo UART, el cual se muestra en su respectivo campo cuando esté disponible.
Si la operación recibida no es reconocida, la suma se toma por defecto.

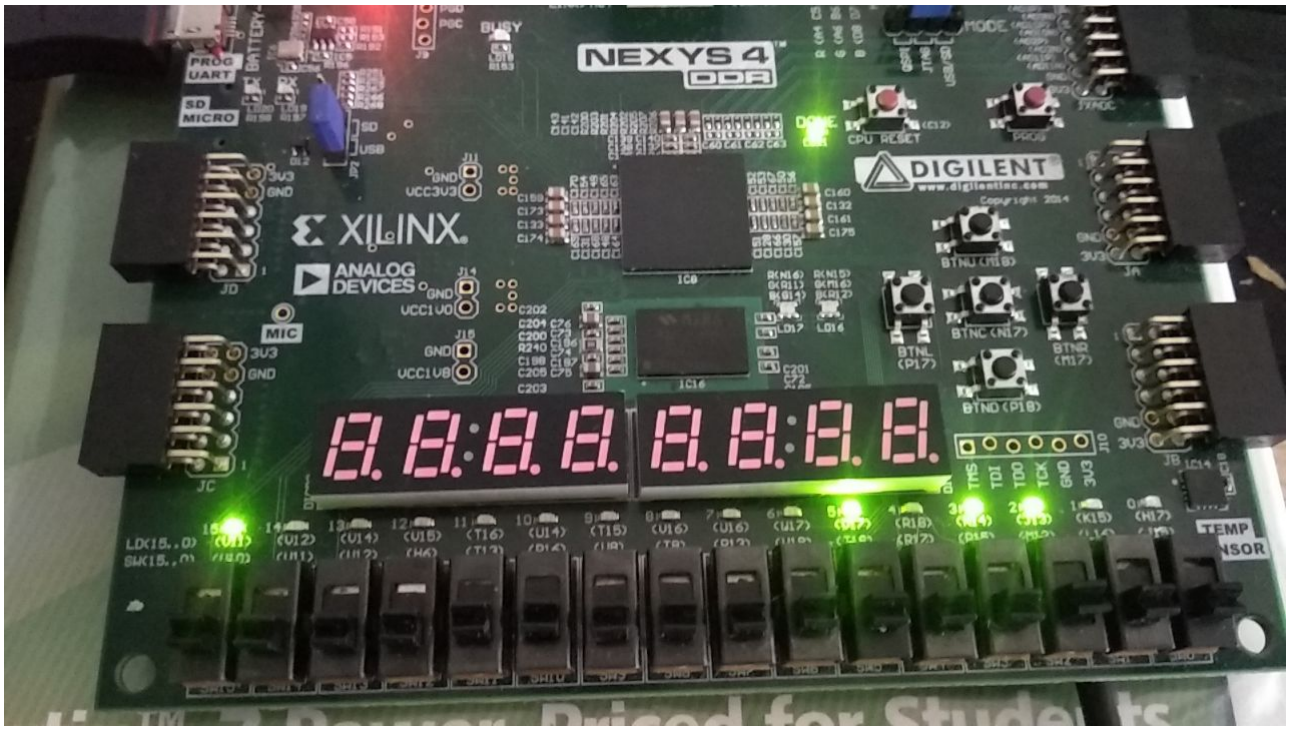
Funcionamiento

- Suma:



18 = 0b00010010

- Suma con carry:



300 = 0b00101100 ***+ carry***