ARQUITECTURA DE COMPUTADORAS

Trabajo Práctico 2

Profesores:

- Santiago Rodriguez
- Martin Pereyra

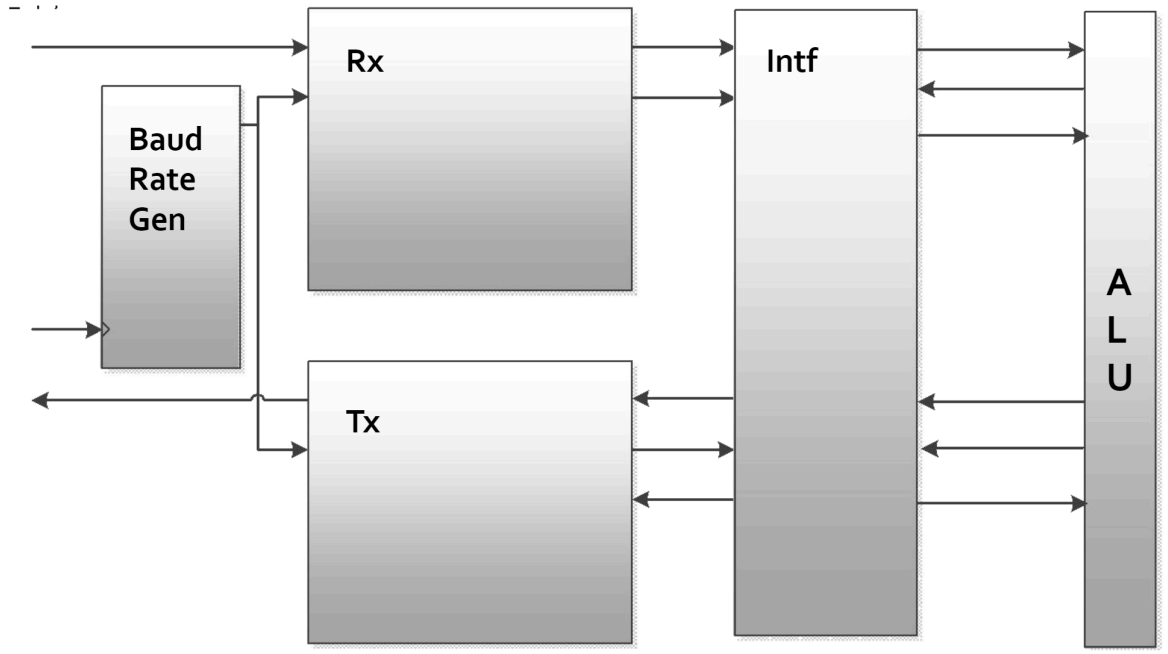
Alumnos:

- Sebastian Klincovitzky 41158451
- Ivo Ferrari 40326730

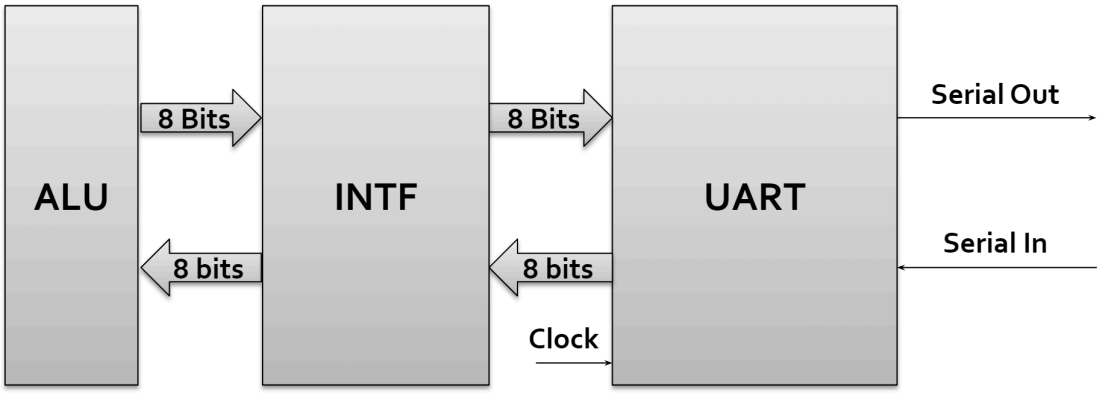
Objetivos

1. Implementar un módulo UART haciendo uso de la ALU del primer trabajo práctico.
2. Implementar la UART con y sin bit de paridad.

Sistema a desarrollar



Diagramas usados de modelo



Receptor UART

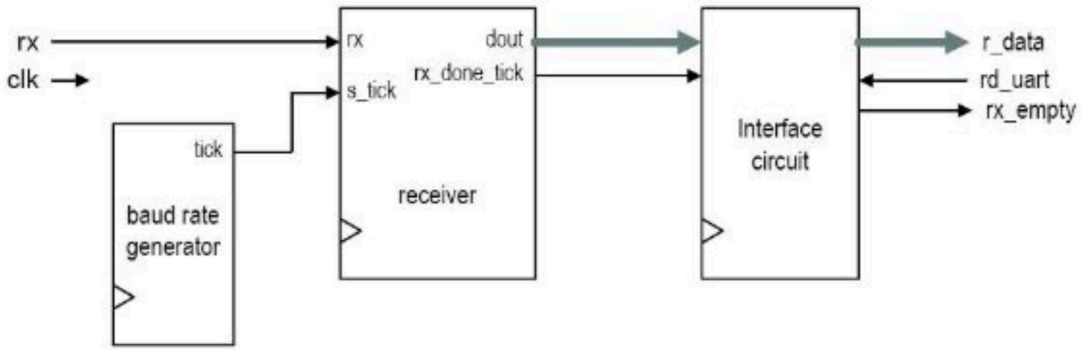
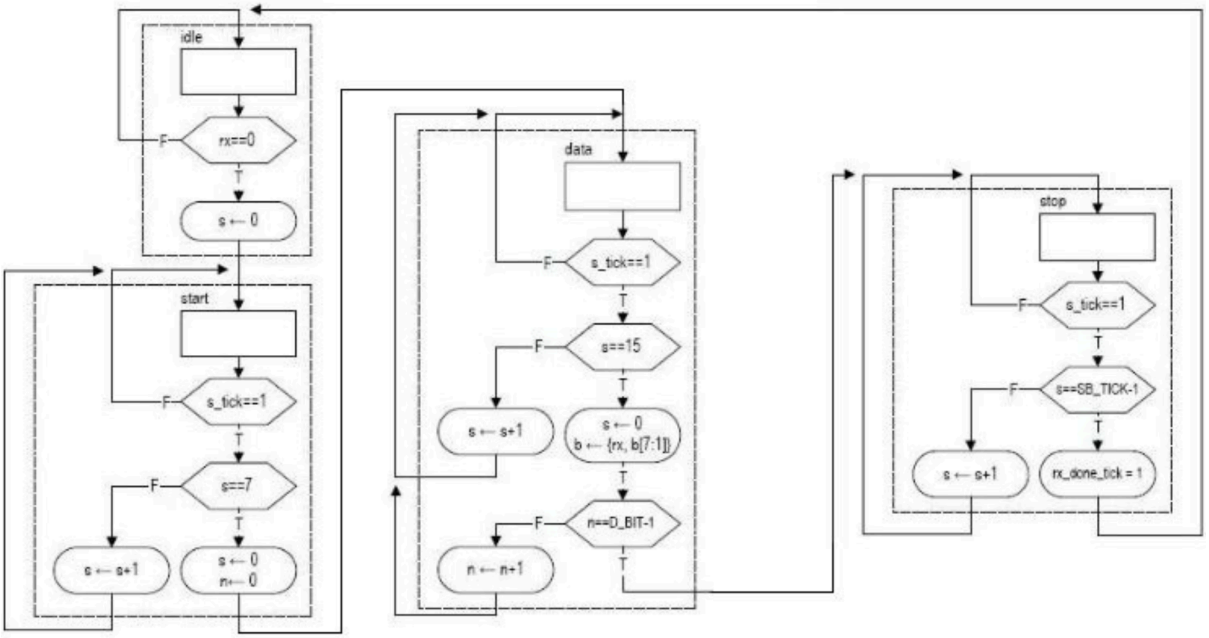
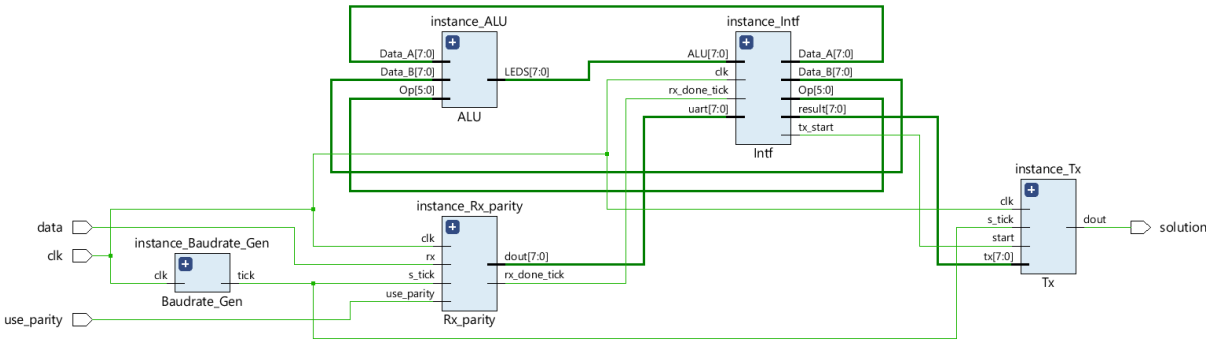


Diagrama de estados Rx



Módulos usados



Para el desarrollo de este trabajo creamos 6 módulos:

ALU

Módulo ALU utilizado en el primer trabajo práctico.

Baudrate_Gen

En base a un cálculo matemático haciendo uso de la frecuencia de clock y el baudrate deseado, da como salida los ticks para Rx y Tx.

```
module Baudrate_Gen
  #(
    parameter BAUDRATE = 19200,
    parameter CLOCK = 50000000, //revisar constraint
    parameter TICK_RATE = CLOCK / (BAUDRATE * 16),
    parameter N_TICK_COUNTER = $clog2(TICK_RATE)
```

Rx

Módulo receptor de la UART. Realiza un muestreo de la entrada rx contando la cantidad de ticks que recibe para buscar hacerlo al centro de la señal, tras recibir el byte completo, lo transmite.
Este módulo avanza haciendo uso de 4 estados: IDLE, START, DATA, STOP. De IDLE avanza a START cuando recibe un 0, en START avanza con 7 ticks, y DATA al llegar a 15 ticks guarda un bit, al acumular 8 (N_data-1) bits avanza a STOP, donde vuelve a IDLE al recibir N_STOP_TICKS ticks. Ademas de cambiar de estado, en STOP se activa la señal para indicar que se terminó de recibir el dato, y que el módulo siguiente sepa que puede aceptar el dato.

Rx_parity

Una evolución del módulo anterior, para poder hacer uso del bit de paridad, agrega una entrada para indicar si se debe usar paridad o no, y un quinto estado, PARITY, al que se llega antes de ir a STOP solo si así lo indica la señal de entrada.

Tx

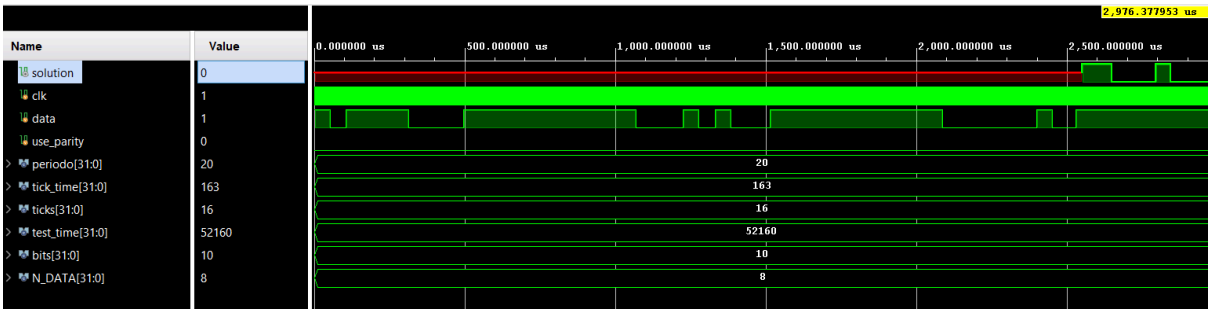
Módulo transmisor de la UART, recibe un byte, y al recibir la señal de comienzo, lo transmite bit a bit.
Para hacerlo, lo hace con 3 estados: IDLE, DATA, y STOP. De IDLE avanza con la señal start, en DATA transmite cada uno de los bits del dato, y STOP transmite un último bit de stop antes de volver a IDLE.

Intf

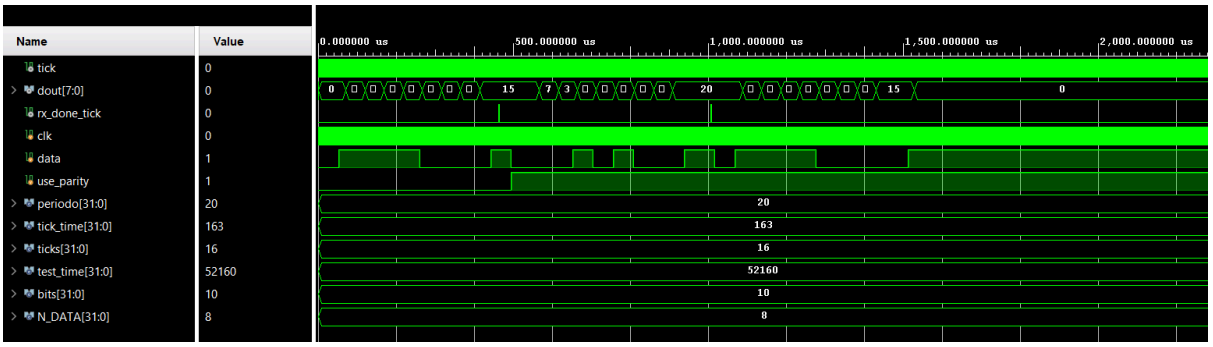
Módulo entre la ALU y la recepción y transmisión de datos. Rx recibe y le pasa los datos y operandos, este se los pasa a la ALU, recibe el resultado, y se los envía a Tx.
Consta de 4 estados, FIRST_DATA, SECOND_DATA, OPERATION, y SEND. En los primeros 3 estados avanza según flag de haber recibido los datos del Rx, y SEND luego de habilitar la flag del Tx,, vuelve naturalmente a FIRST_DATA.

TestBench

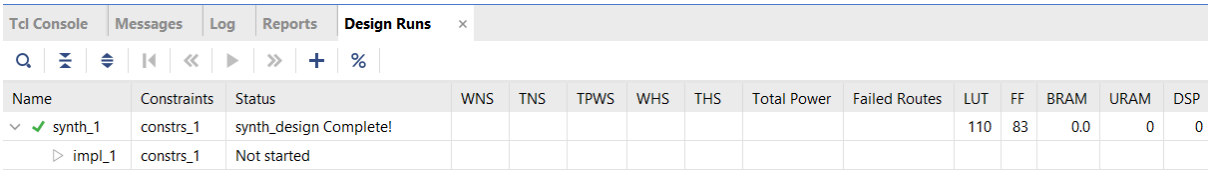
En nuestro trabajo, hemos realizado un TestBench por cada módulo, a continuación capturas de la simulación del TOP donde se realiza suma entre 15 y 20:



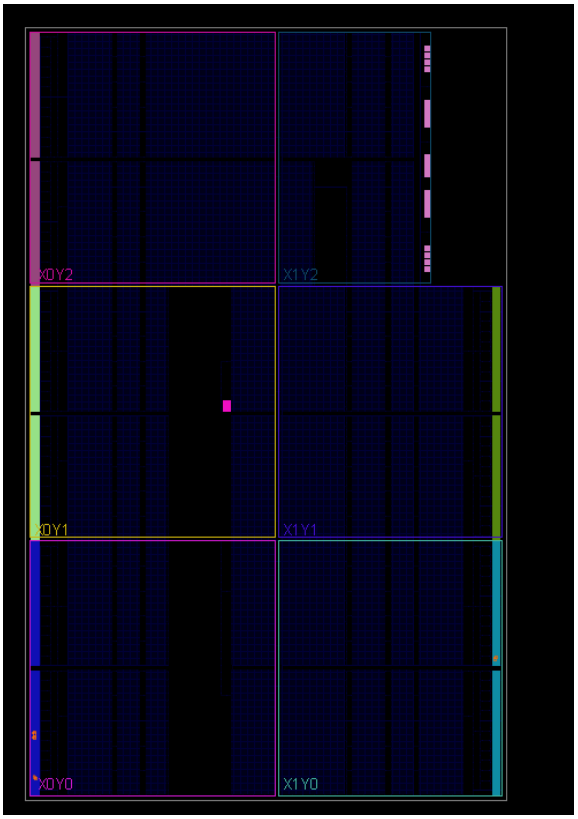
A continuación captura de la simulación de Rx_parity, donde se recibe un 15 con la paridad desactivada, luego se activa paridad y se envía un 20 con bit de paridad 1, y luego 15 (00001111) con bit de paridad 0 (estando la paridad activada), por lo que no se activa rx_done, y luego se reinicia:



TOP post síntesis:



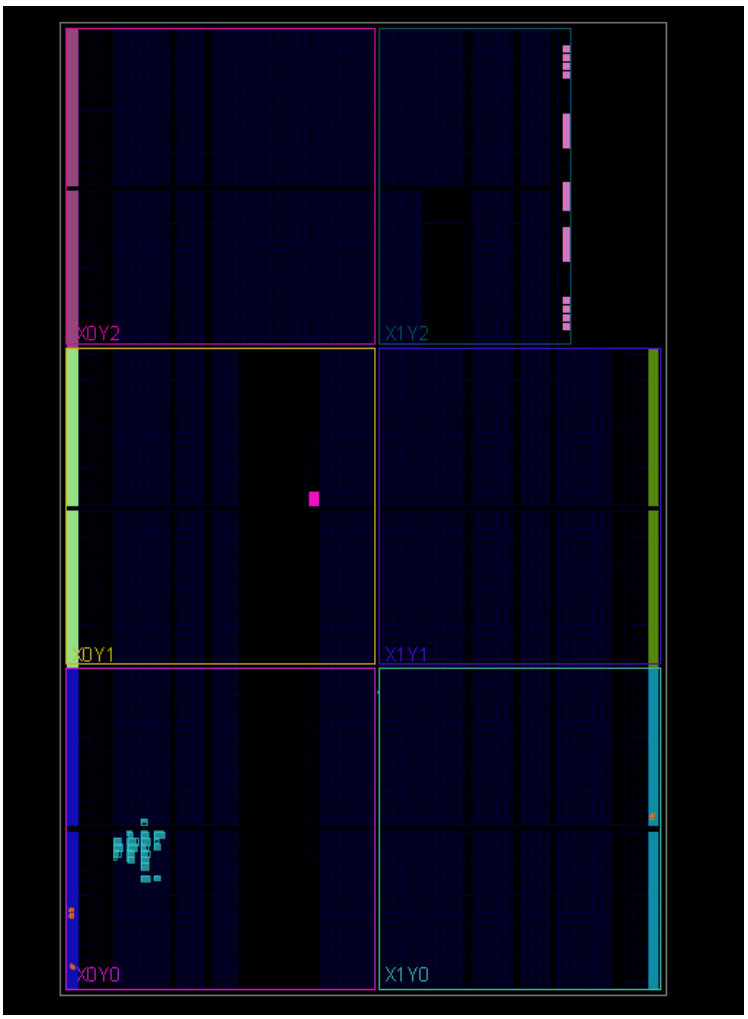
Name	Constraints	Status	WNS	TNS	TPWS	WHS	THS	Total Power	Failed Routes	LUT	FF	BRAM	URAM	DSP
synth_1	constrs_1	synth_design Complete!								110	83	0.0	0	0
impl_1	constrs_1	Not started												

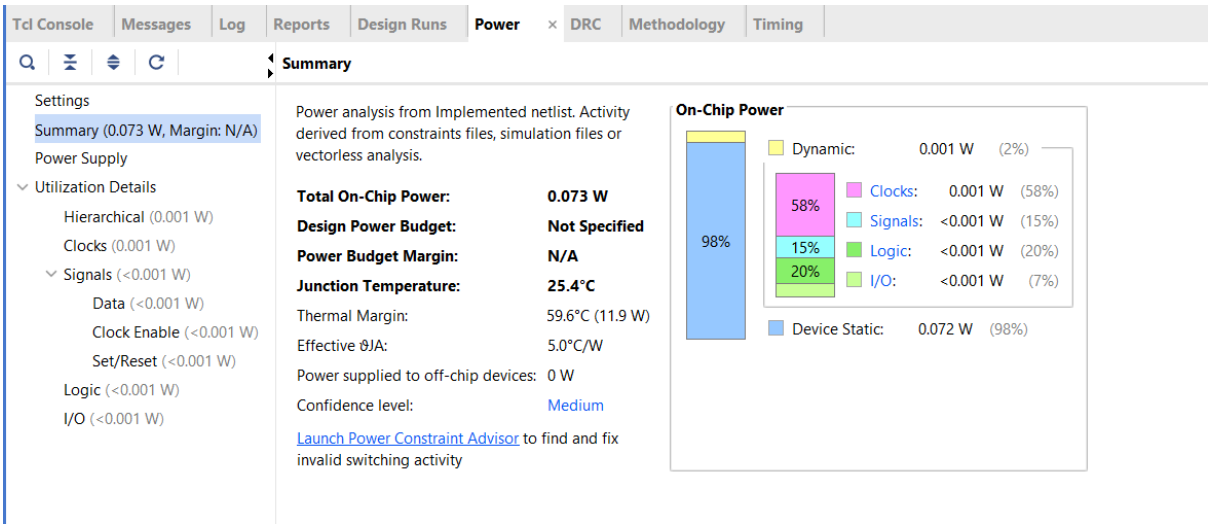


Compilación correcta del top:

Tcl Console | Messages | Log | Reports | **Design Runs** | Power | DRC | Methodology | Timing

Name	Constraints	Status	WNS	TNS	TPWS	WHS	THS	Total Power	Failed Routes	LUT	FF	BRAM	URAM	DSP
✓ synth_1	constrs_1	synth_design Complete!								110	83	0.0	0	0
✓ impl_1	constrs_1	route_design Complete!	5.041	0.000	0.000	0.138	0.000	0.073	0	105	83	0.0	0	0


X0Y2
X1Y2
X0Y1
X1Y1
X0Y0
X1Y0


Tcl Console
Messages
Log
Reports
Design Runs
Power
DRC
Methodology
Timing
Summary
Settings
Summary (0.073 W, Margin: N/A)
Power Supply
Utilization Details
Hierarchical (0.001 W)
Clocks (0.001 W)
Signals (<0.001 W)
Data (<0.001 W)
Clock Enable (<0.001 W)
Set/Reset (<0.001 W)
Logic (<0.001 W)
I/O (<0.001 W)
Power analysis from Implemented netlist. Activity derived from constraints files, simulation files or vectorless analysis.
Total On-Chip Power: 0.073 W
Design Power Budget: Not Specified
Power Budget Margin: N/A
Junction Temperature: 25.4°C
Thermal Margin: 59.6°C (11.9 W)
Effective θJA: 5.0°C/W
Power supplied to off-chip devices: 0 W
Confidence level: Medium
Launch Power Constraint Advisor to find and fix invalid switching activity
On-Chip Power
98%
Dynamic: 0.001 W (2%)
58%
Clocks: 0.001 W (58%)
Signals: <0.001 W (15%)
15%
Logic: <0.001 W (20%)
20%
I/O: <0.001 W (7%)
Device Static: 0.072 W (98%)