

Diseño de Microprocesadores

Práctica 2: SingleCycle

David Adrián Michel Torres
Eduardo Ethandrake Castillo Pulido

Realizar la implementación del procesador RISC-V single-Cycle visto en clase. Este procesador debe ser capaz de ejecutar el programa que implementa el factorial de n . El valor de n se transmitirá a través de una terminal serial, se recomienda el uso de docklight, pero puede cualquier otra terminal que sea capaz de transmitir valores hexadecimales directamente. n será recibida por el microprocesador por una UART la cual se tiene que integrar como periférico mapeado a memoria. Una vez procesado el factorial, el resultado será transmitido por la UART hacia la terminal en la PC. Note que la UART solo puede transmitir 8 bits a la vez, por lo cual para poder transmitir los 32 bits resultantes se tiene que hacer en paquetes de 8 bits, comenzando por los 8 bits más significativos.

- Una captura de pantalla de la simulación de cada instrucción implementada donde señales los puntos clave de la ejecución en particular (seguir formato de simulación).

auipc

auipc a3, 0x0000fc10

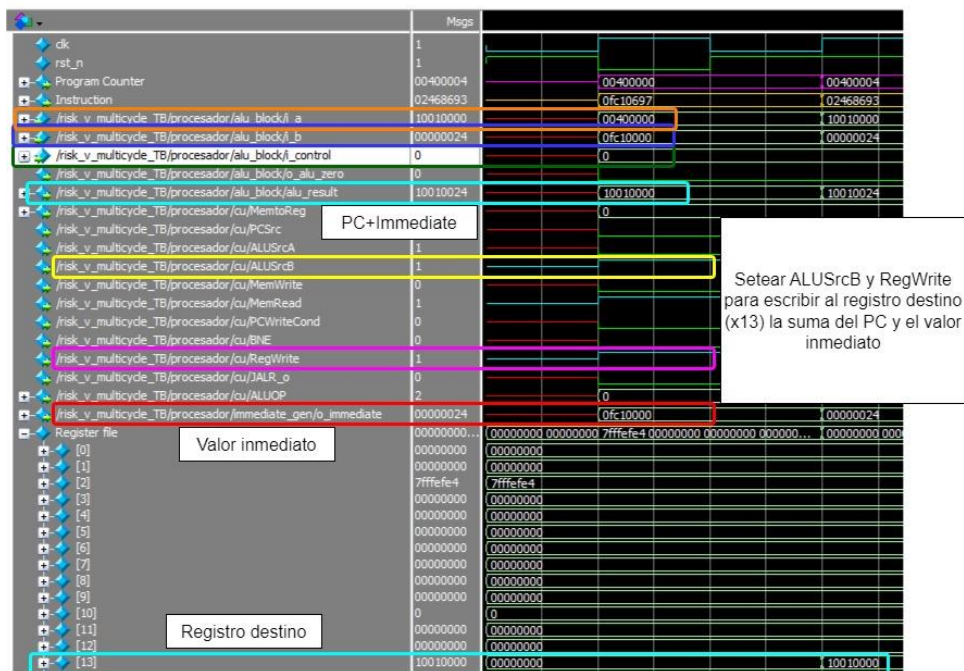


Imagen 1 simulación explicada de instrucción AUIPC

addi

addi a3, a3, 0x24

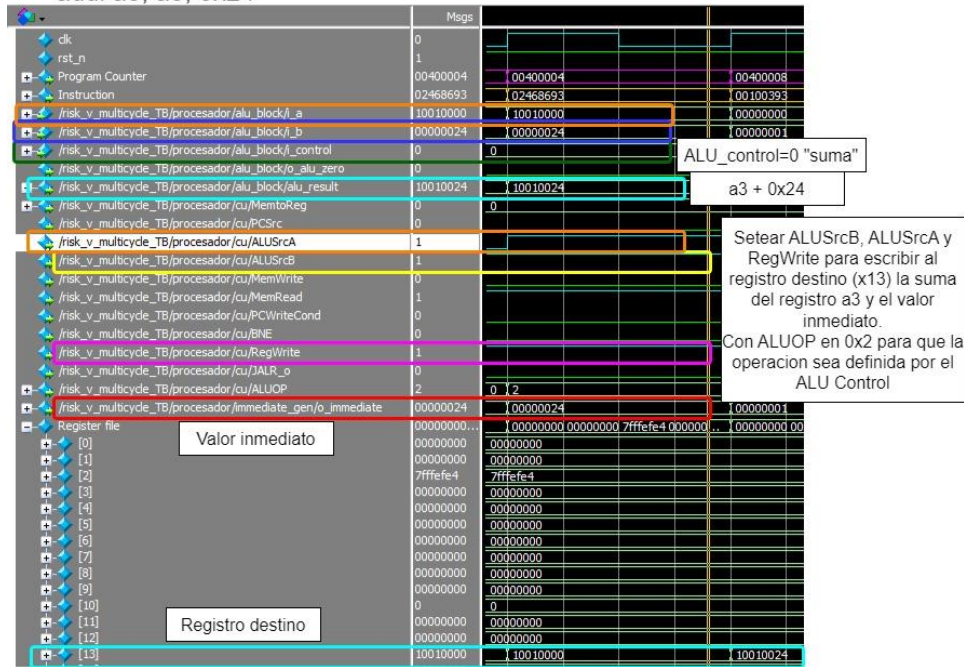


Imagen 2 Simulación explicada de instruccion ADDI

lw

lw a2, 0xC(a3)

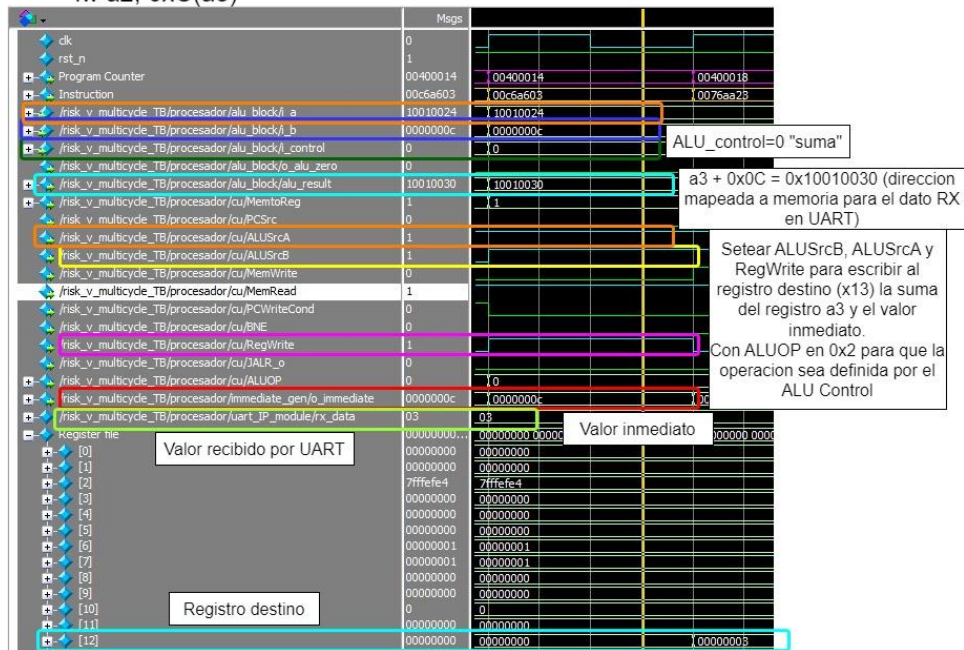


Imagen 3 Simulación explicada de instruccion LW

slti

siti t0, a2, 1,



Imagen 4 Simulacion explicada de instruccion SLTI

jlr

jlr zero,ra,0

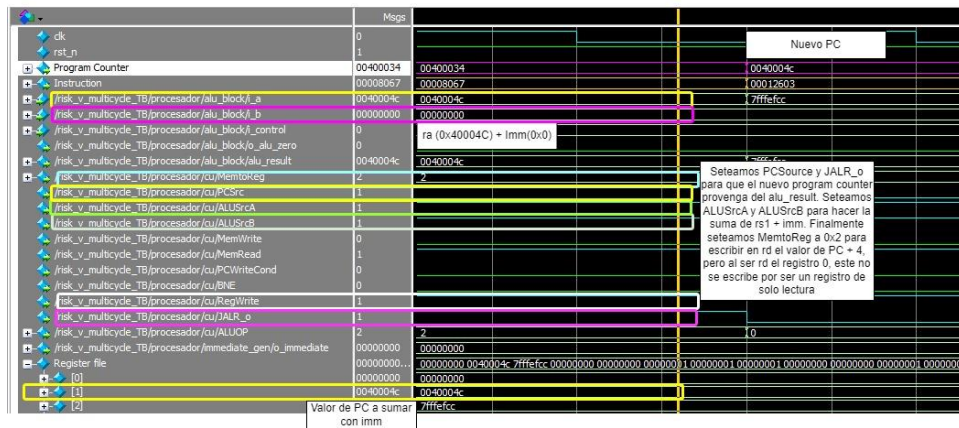


Imagen 5 Simulacion explicada de instruccion JALR

beq

beq t1, zero, get_uart_data

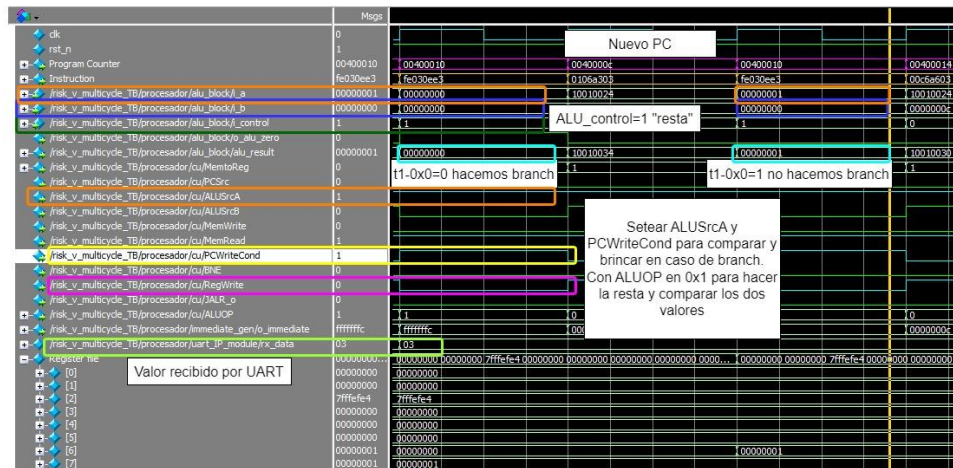


Imagen 6 Simulacion explicada de instruccion BEQ

bne

bne s0, zero, send_loop

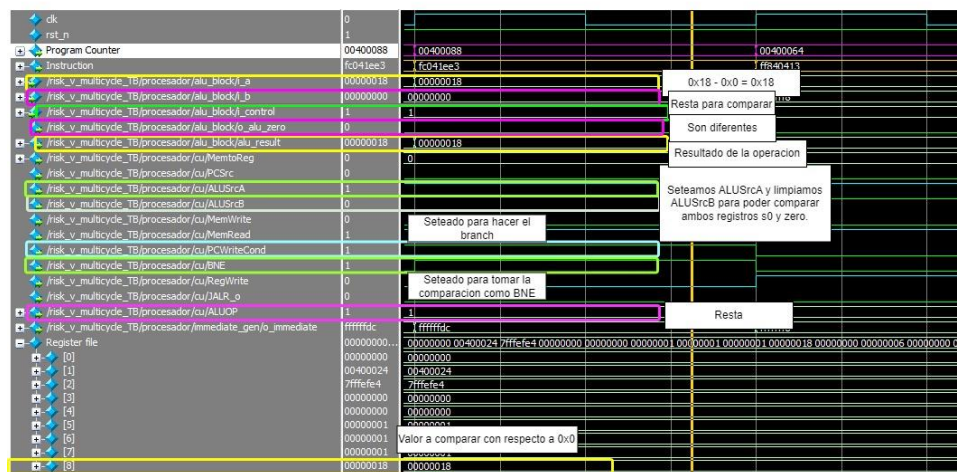


Imagen 7 Simulacion explicada de instruccion BNE

SW



Imagen 8 Simulacion explicada de instruccion SW

jal



Imagen 9 Simulacion explicada de instruccion JAL

mul

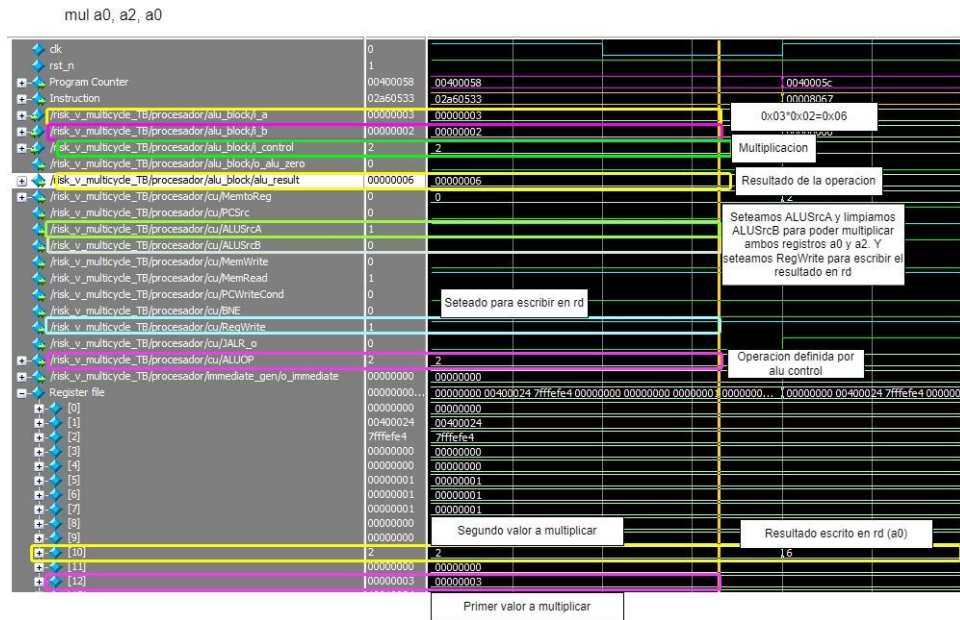


Imagen 10 Simulacion explicada de instruccion MUL

Srl

[illegible]

Imagen 11 Simulacion explicada de instruccion SRL

- Una captura de pantalla donde se demuestre el comportamiento recursivo del Factorial (uso de JAL y JR)

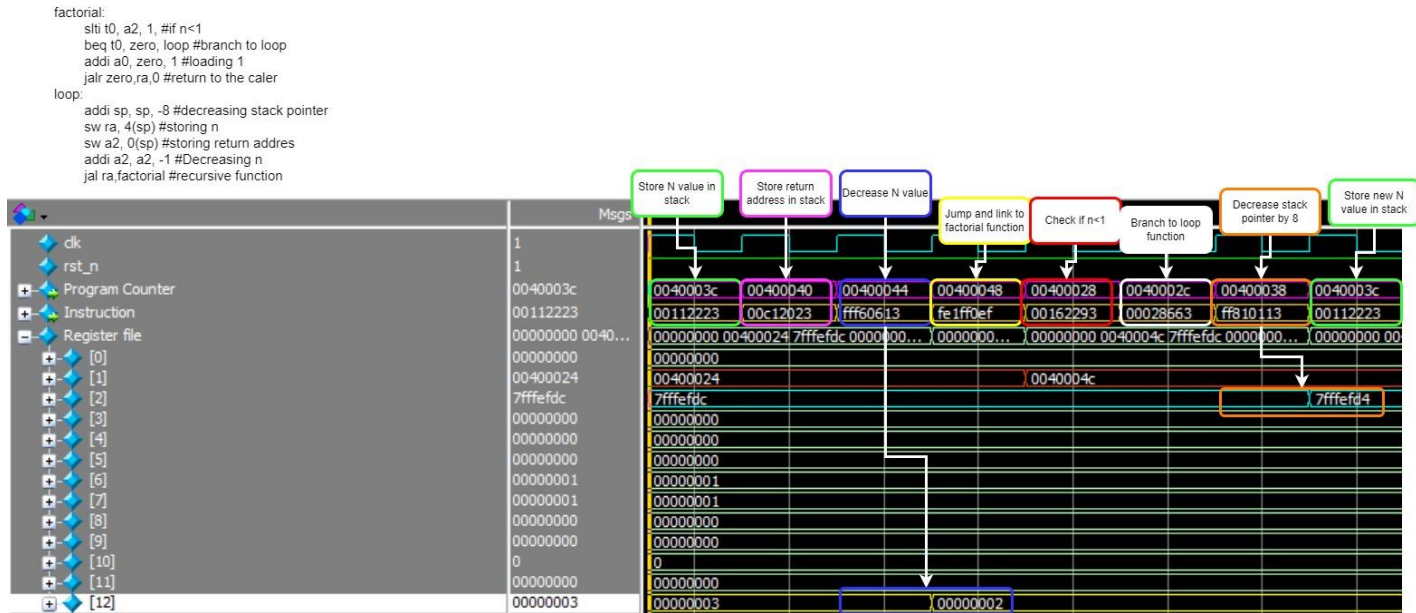


Imagen 1212 – Uso recursivo de instrucción JAL

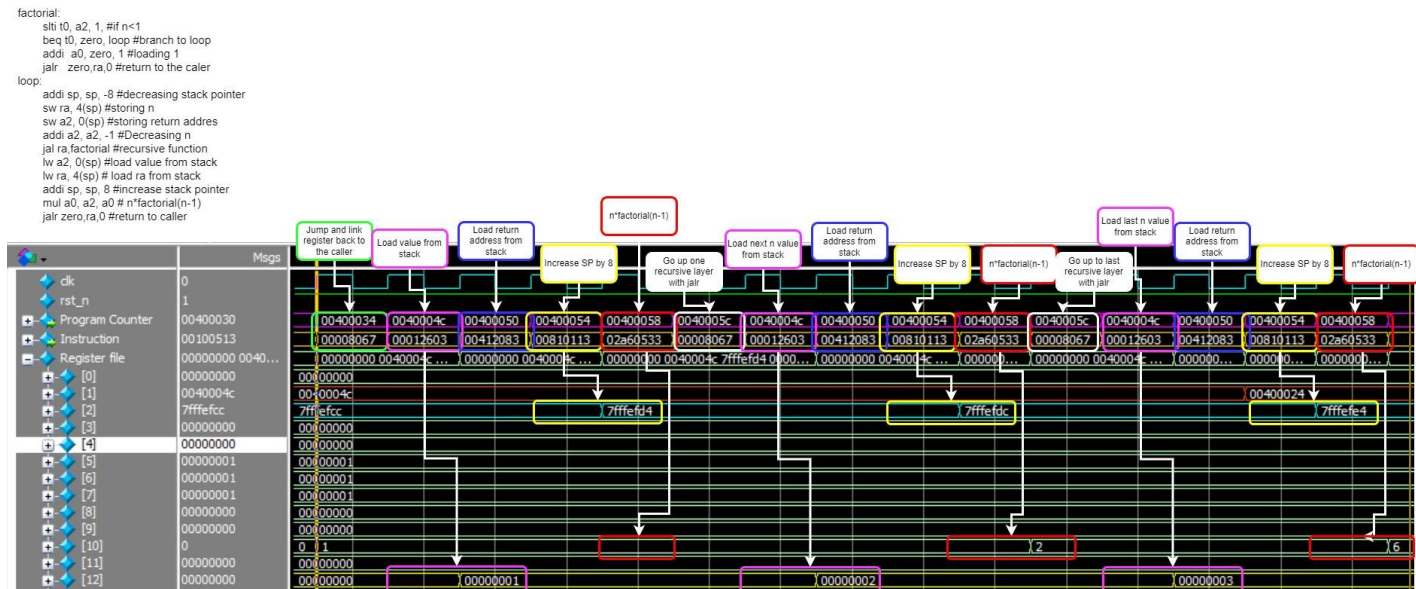


Imagen 1313 – Uso recursivo de instrucción JALR

- Frecuencia Máxima de operación**

La frecuencia máxima de operación del sistema a 0°C es de 41.87Mhz, mientras que el sistema a 85°C la frecuencia máxima es de 42.37Mhz. Por lo cual se puede obtener una frecuencia media de operación de 42Mhz.

Slow 1100mV 85C Model Fmax Summary				
<<Filter>>				
	Fmax	Restricted Fmax	Clock Name	Note
1	41.87 MHz	41.87 MHz	clk_pll	

Slow 1100mV 0C Model Fmax Summary				
<<Filter>>				
	Fmax	Restricted Fmax	Clock Name	Note
1	42.37 MHz	42.37 MHz	clk_pll	

Imagen 1414 – Frecuencia máxima de operación

- Una tabla donde se muestre el CPI de cada una de las instrucciones implementadas.**

Instruction	Type	Clocks cycles
auipc	U-type	1
addi	I-type	1
lw	I-type	1
slti	I-type	1
jalr	I-type	1
beq	B-type	1
bne	B-type	1
sw	S-type	1
jal	J-type	1
mul	R-type	1
srl	R-type	1
Type	CPI	
U-type	1	
I-type	1	
B-type	1	
S-type	1	
J-type	1	

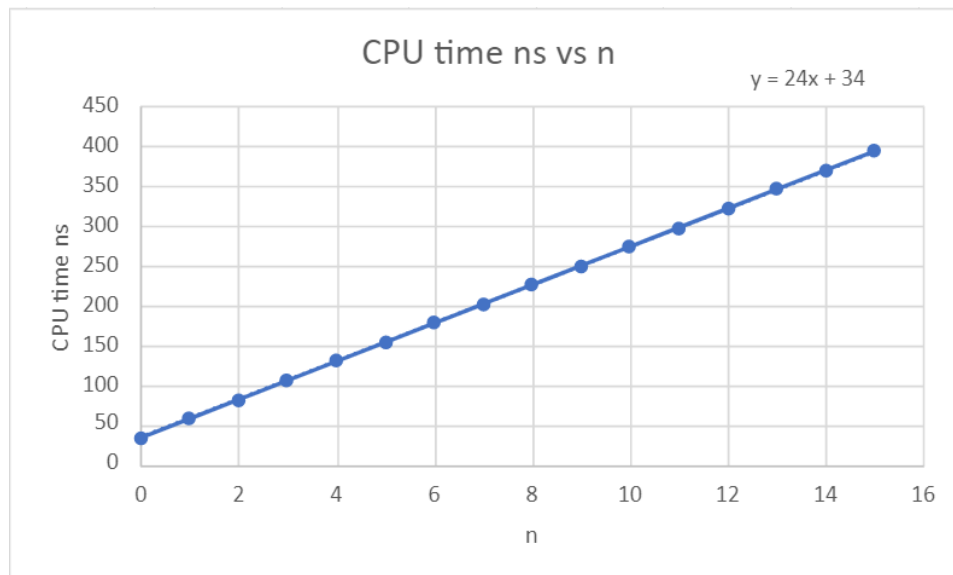
R-type	1
--------	---

Tabla 1 – CPI del programa

- Grafica donde se muestre la relación n vs para $n = 0$ hasta $n = 15$. time CPU

Tabla 2 – CPU time vs Numero de iteraciones

n	CPU time ns
0	34
1	58
2	82
3	106
4	130
5	154
6	178
7	202
8	226
9	250
10	274
11	298
12	322
13	346
14	370
15	394



- Resultados de síntesis en términos de logic elements (LEs)

Se puede observar en la Imagen 1616 que se está utilizando un total de 2,201 ALMs, esto simboliza un 5% del total. Y se esta utilizando 4 pines (Rx, Tx, clk, rst) lo cual simboliza menos del 1% de utilización de todo el sistema.

Flow Summary	
<<Filter>>	
Flow Status	Successful - Wed Apr 12 19:31:30 2023
Quartus Prime Version	21.1.1 Build 850 06/23/2022 SJ Lite Edition
Revision Name	risk_v_singlecycle
Top-level Entity Name	risc_v_top
Family	Cyclone V
Device	5CSXFC6D6F31C6
Timing Models	Final
Logic utilization (in ALMs)	2,201 / 41,910 (5 %)
Total registers	2950
Total pins	4 / 499 (< 1 %)
Total virtual pins	0
Total block memory bits	0 / 5,662,720 (0 %)
Total DSP Blocks	2 / 112 (2 %)
Total HSSI RX PCSs	0 / 9 (0 %)
Total HSSI PMA RX Deserializers	0 / 9 (0 %)
Total HSSI TX PCSs	0 / 9 (0 %)
Total HSSI PMA TX Serializers	0 / 9 (0 %)
Total PLLs	0 / 15 (0 %)
Total DLLs	0 / 4 (0 %)

Imagen 1616 – Numero de Les utilizados en el programa

- Microarquitectura en Visio o programa equivalente.

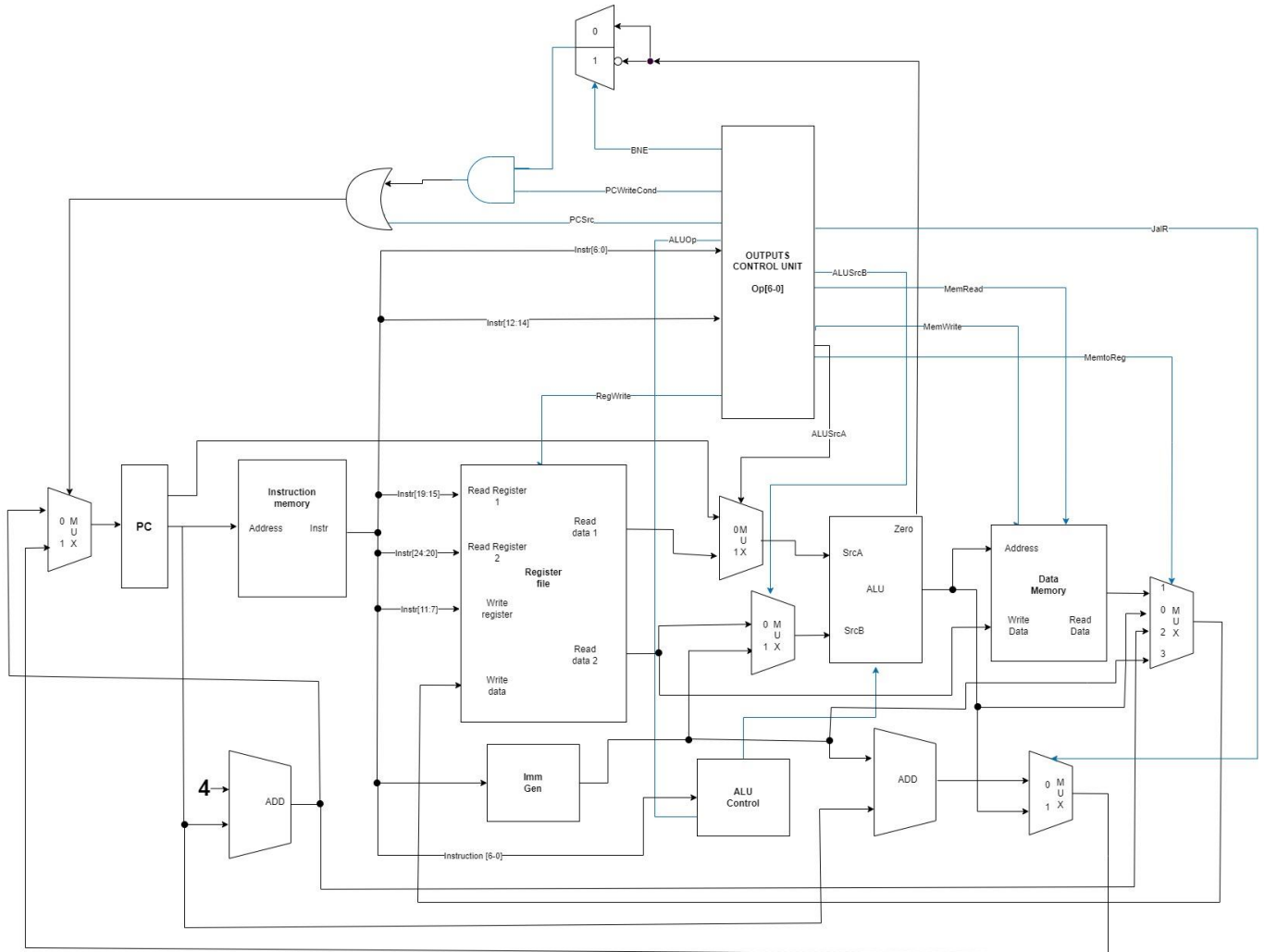


Imagen 1717 – Arquitectura singlecycle

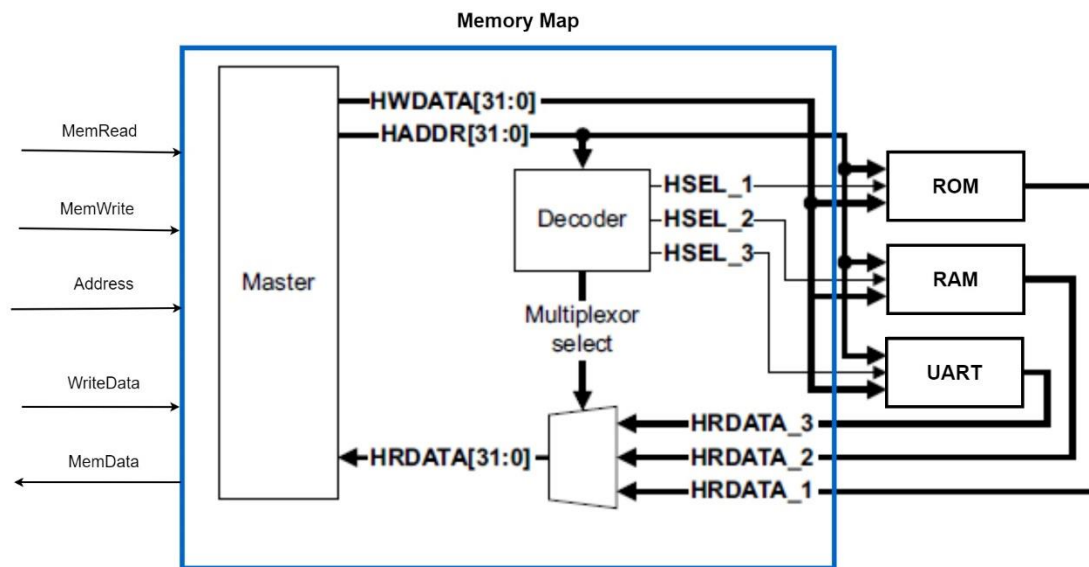


Imagen 18 18- Arquitectura memory map

- Captura de pantalla de las señales internas usando el Signal Tap donde se muestre la ejecución recursiva.

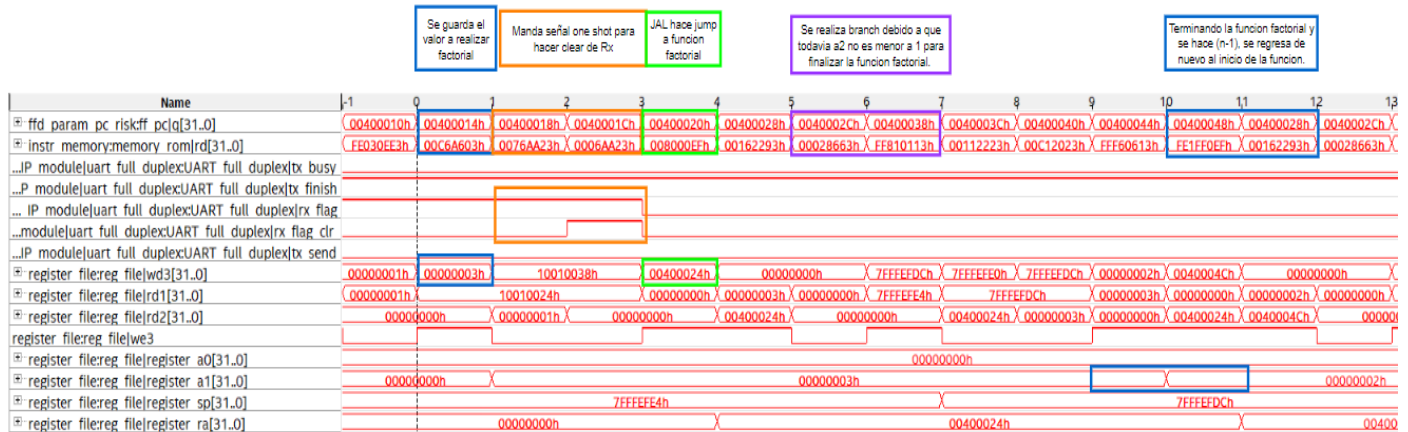


Imagen 1919 – Signal tap recursividad 1

En la Imagen 1919 se puede observar el inicio de nuestra secuencia en la cual se recibe por el Rx el numero a realizar factorial, que en este caso es el número 3. El programa realiza escrituras al UART para hacer clear de la señal Rx. Y también se inicia la función factorial que una vez que se termina y se realiza la resta de N-1, regresamos de nuevo a la función para realizar el calculo de nuevo hasta que $N < 1$.

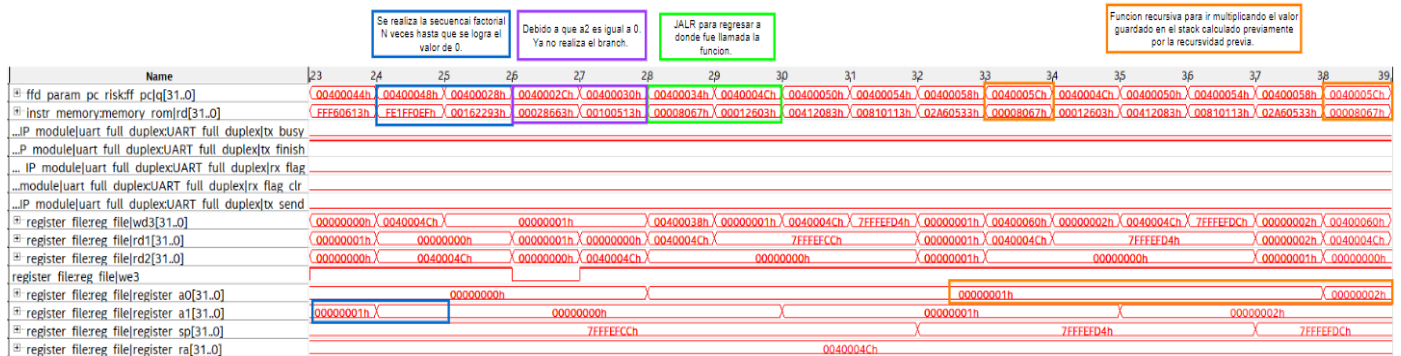


Imagen 2020 – Signal tap recursividad 2

Una vez que se terminó la recursividad del factorial en el cual se llegó $N = 0$. Se procede a realizar otra secuencia recursiva en la cual se va a estar multiplicando los datos guardados en el stack para poder obtener el factorial que va a ser calculado en el registro A0.

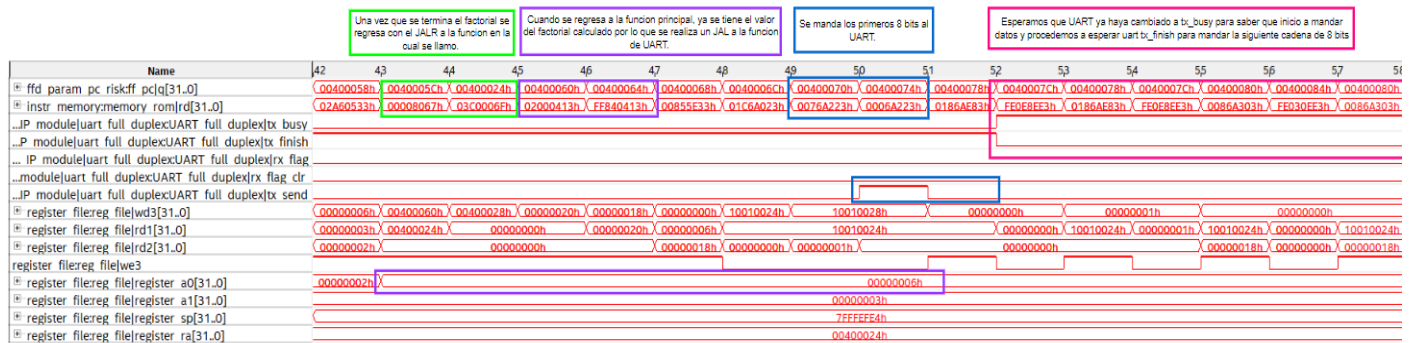


Imagen 21 21 – Signal tap recursividad 3

Cuando ya se tiene calculado el valor del factorial en el registro A0, el JALR nos va a regresar cuando fue llamada la función en la etiqueta principal para poder hacer un jump a la secuencia del UART, en la cual toma el valor del factorial, realiza un shift para poder mandar del MSB al LSB y poder verlo en la terminal adecuadamente. Una vez que hace ese shift manda la señal UART_TX send al UART y espera a que el UART inicie con la transmisión y la termine para poder hacer esta secuencia 4 veces para poder transmitir los 32 bits en 4 cadenas de 8 bits.

- Todas las fuentes deben ser debidamente comentadas.

Los archivos se comentaron en la parte superior de cada uno, especificando el nombre del módulo, función y fecha. Así como ciertos comentarios en los archivos. Estos archivos se encuentran en el repositorio en la carpeta /src.

- Usar el repositorio DM_Apellido1_Apellido2_RISC_V y colocar la implementación en un branch con el nombre single_Cycle

- Definición de las direcciones utilizadas para la comunicación con la UART (mapeo de direcciones utilizadas en relación con los registros internos).**

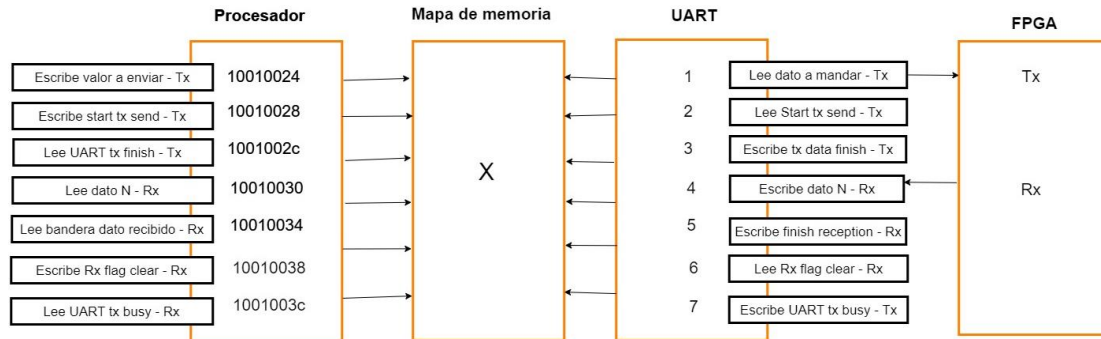
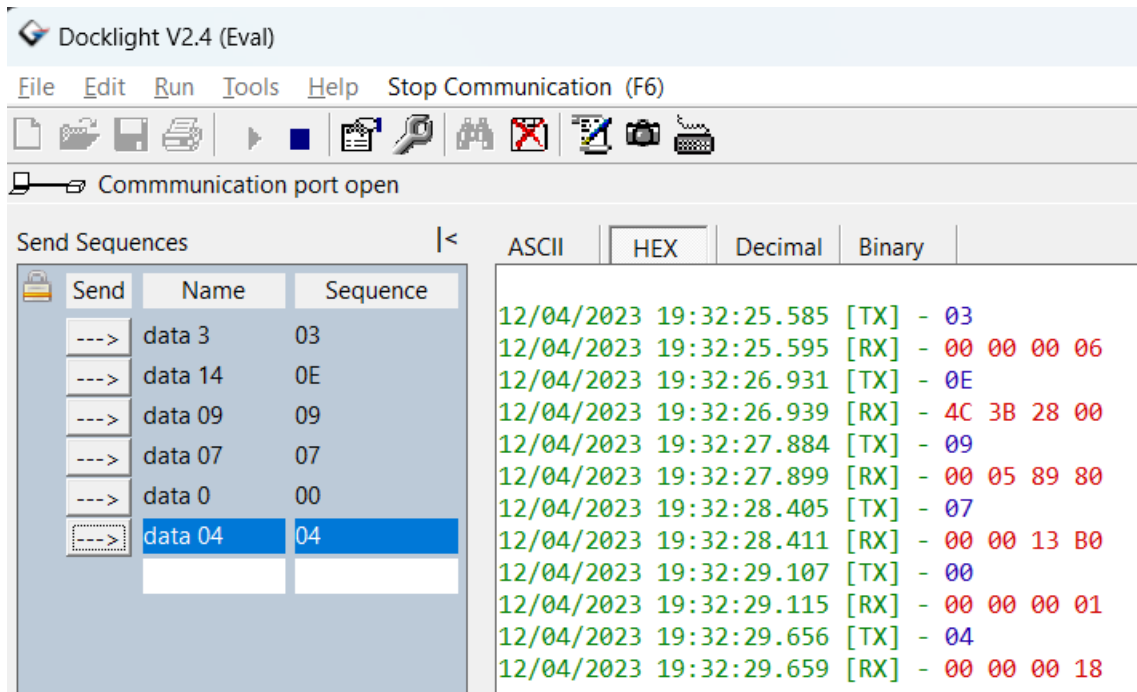


Imagen 22 22– Mapeo de direcciones procesador & UART

- Resultados singlecycle - UART**



La imagen muestra la interfaz de Docklight V2.4 (Eval) con la comunicación port abierta. Se visualizan las secuencias de envío y recepción de datos en tiempo real.

Send	Name	Sequence	Timestamp	Direction	Data (HEX)
---	data 3	03	12/04/2023 19:32:25.585	[TX]	03
---	data 14	0E	12/04/2023 19:32:25.595	[RX]	00 00 00 06
---	data 09	09	12/04/2023 19:32:26.931	[TX]	0E
---	data 07	07	12/04/2023 19:32:26.939	[RX]	4C 3B 28 00
---	data 0	00	12/04/2023 19:32:27.884	[TX]	09
---	data 0	00	12/04/2023 19:32:27.899	[RX]	00 05 89 80
---	data 04	04	12/04/2023 19:32:28.405	[TX]	07
---	data 04	04	12/04/2023 19:32:28.411	[RX]	00 00 13 B0
---			12/04/2023 19:32:29.107	[TX]	00
---			12/04/2023 19:32:29.115	[RX]	00 00 00 01
---			12/04/2023 19:32:29.656	[TX]	04
---			12/04/2023 19:32:29.659	[RX]	00 00 00 18

Imagen 23 23 – Uso de docklight para demostración de resultados de la practica

- Programa en ensamblador de utilizado para la comunicación.

Código ensamblador ubicado dentro del repositorio en el directorio “asm” bajo el nombre de “factorial.asm”