INTERCONEXIÓN DE PROCESADORES A UNA MEMORIA COMPARTIDA MEDIANTE UNA ARQUITECTURA DE INTERCONEXIÓN DE SISTEMA EN CHIP WISHBONE.

T.G No. 2027

JUAN PABLO BERNAL SÁENZ

TRABAJO DE GRADO PARA OPTAR POR EL TÍTULO DE INGENIERO ELECTRÓNICO.

DIRIGIDO POR:

ING. FRANCISCO FERNANDO VIVEROS MORENO

ING. LUISA FERNANDA GARCÍA VARGAS, Ph.D

PONTIFICIA UNIVERSIDAD JAVERIANA
FACULTAD DE INGENIERÍA
DEPARTAMENTO DE INGENIERÍA ELECTRÓNICA
BOGOTÁ D.C.
2021

RECTOR DE LA UNIVERSIDAD:	**JORGE HUMBERTO PELÁEZ PIEDRAHITA S. J.**
DECANO FACULTAD INGENIERÍA:	**LOPE HUGO BARRERO SOLANO, SCD.**
DIRECTOR DE CARRERA:	**JULIAN DAVID COLORADO MONTAÑO, Ph.D.**
DIRECTOR DE PROYECTO:	**FRANCISCO FERNANDO VIVEROS MORENO**
CODIRECTOR DE PROYECTO:	**LUISA FERNANDA GARCÍA VARGAS, Ph.D.**

ARTÍCULO 23 DE LA RESOLUCIÓN No. 13 DE JUNIO DE 1946.

"La universidad no se hace responsable de los conceptos emitidos por sus alumnos en sus proyectos de grado. Solo velará porque no se publique nada contrario al dogma y la moral católica y porque los trabajos no contengan ataques o polémicas puramente personales. Antes bien, que se vea en ellos el anhelo de buscar la verdad y la justicia".

AGRADECIMIENTOS.

"Gracias Dios que ha sido mi guía y fortaleza durante todo mi proceso en la universidad. Gracias a la universidad Javeriana por haberme formado como profesional y como ser humano en este proceso, por enseñarme que el esfuerzo siempre trae buenos resultados. Gracias a mis padres pilares fundamentales durante este proceso, su apoyo fue la fortaleza para no detenerme en este camino. Gracias a mi familia y personas que estuvieron acompañándome en este proceso. Gracias a mis directores de trabajo de grado quienes con su conocimiento y experiencia me apoyaron en la ejecución de este trabajo de grado."

TABLA DE CONTENIDO.

LISTA DE FIGURAS.

FIGURAS DE LOS ANEXOS.

LISTA DE ANEXOS.

1 INTRODUCCIÓN.

Los avances tecnológicos han resultado en que muchas aplicaciones sean *open source* (código abierto)[1], permitiendo que cualquier persona pueda tomar como base un proyecto y realizar las modificaciones que considere necesarias. Lo cual ha llevado a que hoy en día sea considerado el término de *hardware* libre para la implementación de diversas aplicaciones, como por ejemplo los procesadores[2].

En un sistema complejo en el que varios procesadores están conectados a una misma memoria, puede ocurrir que estos procesadores quieran acceder de manera simultánea a la memoria, pero al no existir un orden en el acceso a la memoria se genera un problema de flujo de datos que puede desencadenar que las tareas que realiza el procesador no sean ejecutadas de manera correcta.

El objetivo de este trabajo de grado es implementar una arquitectura de interconexión de sistema en chip que facilite el acceso ordenado de los diversos núcleos de procesamiento a la memoria, y así evitar pérdidas de información y retardos en la ejecución de las tareas de los procesadores, ya que se puede decidir el orden en el que los procesadores acceden a la memoria.

Para solucionar la problemática presentada se elige una arquitectura de interconexión WISHBONE la cual está basada en una arquitectura de *hardware* libre, esta arquitectura presenta ventajas respecto a buses de datos como el Coreconect diseñado por IBM[3] o el AMBA diseñado por ARM[4].

El diccionario Webster define la palabra WISHBONE como “Un hueso bifurcado en frente del esternón de un pájaro”[5], este hueso es comúnmente conocido como hueso de la suerte. De aquí surge el nombre de esta arquitectura, ya que en su primer esquema se intentaba encontrar un nombre que describiera un bus de datos bidireccional que usara multiplexores o una lógica de 3 estados,

Esto se logró formando una interfaz con caminos separados uno para la entrada y otro para la salida, cuando estos caminos se conectan a una lógica 3 estados tiene la forma de una configuración “Y” la cual se parece a un hueso de la suerte.[6]

Es común encontrar en documentos el término bus de datos WISHBONE, sin embargo, en el documento *“Wishbone B4 specification WISHBONE System-on-Chip (SoC) Interconnection Architecture for Portable IP Cores”* [6] se especifica que WISHBONE es una arquitectura de interconexión de sistema en chip.

El presente documento de trabajo de grado está dividido en ocho secciones, en la sección dos se presenta el objetivo general y los objetivos específicos, así como el alcance. La sección tres contiene el marco teórico, en donde primero se realiza una explicación detallada sobre las características de la arquitectura de interconexión WISHBONE así mismo se realiza una breve explicación acerca de la arquitectura RISC-V.

La sección cuatro está dividida en dos partes, la primera es el diseño de la arquitectura de interconexión WISHBONE, que incluye el diagrama de entradas y salidas, el diagrama de bloques, el AHPL, la descripción de las señales de la interconexión WISHBONE, la descripción del tipo de interconexión, y los esquemáticos. La segunda parte corresponde a la descripción en VHDL y a la implementación en la FPGA.

La sección cinco muestra los protocolos de pruebas, los resultados obtenidos de las simulaciones implementadas durante el diseño del sistema, así como las soluciones a los diferentes problemas que se presentaron en la elaboración del diseño. La sección seis corresponde a las conclusiones, incluyendo un análisis de posibles vías de investigación que podrían complementar el trabajo realizado, la sección siete corresponde a la bibliografía con las referencias utilizadas en el presente libro, por último, la sección 8 corresponde al listado de anexos con el enlace a la carpeta de *one drive* donde estos se encuentran.

2 OBJETIVO DEL PROYECTO

2.1 OBJETIVO GENERAL

- Implementar una arquitectura de interconexión de sistema en chip WISHBONE para la interconexión de 3 procesadores RISC-V con una memoria compartida.

2.2 OBJETIVOS ESPECÍFICOS.

- Determinar los requerimientos necesarios para la implementación de una arquitectura de interconexión WISHBONE
- Diseñar la arquitectura de interconexión de sistema en chip WISHBONE
- Diseñar un sistema que interconecte tres procesadores RISC-V con una memoria compartida por medio de la arquitectura de interconexión diseñada.
- Implementar la arquitectura de interconexión WISHBONE y el sistema multiprocesador por medio de una FPGA
- Diseñar un protocolo de pruebas que permita verificar el funcionamiento del sistema

2.3 ALCANCE FINAL.

Como alcance final del proyecto se desarrolló una arquitectura de interconexión WISHBONE con todos los bloques y señales necesarias para realizar la interconexión de 3 procesadores independientes con una memoria compartida. Este sistema se implementó en *hardware* en una FPGA DE2-115 y los resultados fueron visualizados por medio de un analizador de estados lógico.

En las señales propias de la arquitectura WISHBONE existen señales que le indican a cada maestro que tiene el acceso del bus, sin embargo, el procesador[7] no tiene una señal que le indique que tiene el control de la arquitectura de interconexión WISHBONE

Debido a esto fue necesario realizar una intervención al procesador RISC-V y se realizó una prueba para verificar el correcto funcionamiento de la intervención al procesador, luego se procedió al diseño de la arquitectura de interconexión, durante este proceso se realizaron pruebas a medida que se culminaba el diseño de cada uno de los bloques de la interconexión WISHBONE.

Finalmente, se verificó el funcionamiento total de la arquitectura de interconexión WISHBONE con los 3 procesadores accediendo a la memoria compartida, cada uno de ellos realizando un programa diferente verificando que las señales y bloques de la interconexión funcionan según lo diseñado.

3 MARCO TEÓRICO

En esta sección se presentan de manera resumida los conceptos básicos de este trabajo de grado

3.1 ARQUITECTURA DE INTERCONEXION WISHBONE.

La compañía Silicore Corporation realizó la primera investigación y presentó la arquitectura de interconexión de sistema en chip WISHBONE. Al principio ellos tenían todo el control sobre el diseño de esta arquitectura, luego entregaron las especificaciones al público [7] y permitieron que la comunidad de Opencores brindara la posibilidad a los diseñadores de implementar sus diseños basados en otros ya existentes y cumplir con el objetivo del *hardware* libre.

Las características de la arquitectura de interconexión WISHBONE pueden encontrase en el documento: *"Wishbone B4 specification WISHBONE System-on-Chip (SoC) Interconnection Architecture for Portable IP Cores*"[6], sus características más básicas son las siguientes:

- Simple, compacto y tiene una interfaz de hardware de núcleos que requiere muy pocas compuertas lógicas.
- Permite metodologías de diseño estructurado usadas por grandes equipos de proyectos.
- El protocolo de enlace permite que cada núcleo acelere su velocidad de transferencia de datos.

- Usa protocolos de transmisión de datos populares como el ciclo de lectura y escritura,
- Anchos de bus de datos y tamaños de operandos de hasta 64 bits.

Existen 4 diferentes maneras de implementar una arquitectura de interconexión WISHBONE según como los procesadores están conectados a ella. Las 4 formas son: punto a punto, flujo de datos, bus compartido e interconexión cruzada. Para la realización de este trabajo de grado se escogió la forma de interconexión de bus compartido, en la figura 1 se observa el diagrama en bloques de esta interconexión.

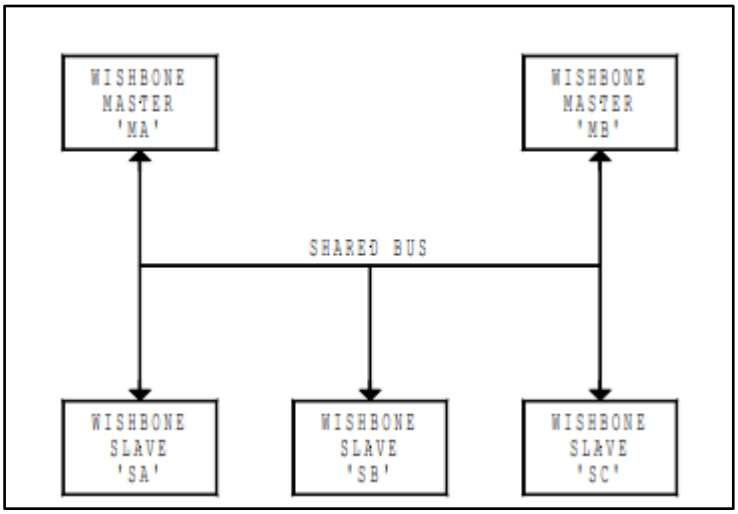

Figura 1. Interconexión de bus compartido. Tomada de [6]

Para el diseño de esta arquitectura de interconexión WISHBONE es necesario tener en cuenta que existen señales de entrada y salida para esclavos y maestros, donde el uso de algunas de estas es opcional, en la siguiente sección se presenta una descripción de las señales propias de la arquitectura de interconexión WISHBONE.

3.1 ARQUITECTURA RISC-V

La arquitectura RISC-V es una arquitectura nacida en la Universidad de California en Berkley, su principal objetivo es ser de uso libre, modificación y comercialización, esto permite reducir los costos de diseño al sistema que se está desarrollando.

Al ser una arquitectura de *hardware* libre, se estableció una versión base que con el paso del tiempo sufrió modificaciones que conllevaron a que se obtuviera una versión estable a la que los cambios que le faltan son mínimos.

La arquitectura RISC-V posee tres posibilidades de ancho de palabra (32,64,128) y tiene múltiples extensiones de instrucciones entre las que se encuentran las instrucciones tipo I para operaciones aritméticas, las instrucciones tipo M para multiplicación y división de números enteros, instrucciones tipo S para cargar datos a la memoria, entre otras extensiones de instrucciones.

Los procesadores RISC-V reciben su nombre basado en qué tipo de extensiones utiliza y el número de bits de palabra que tiene. En este caso el procesador seleccionado fue diseñado por Javier Barbosa [7], este tiene 32 bits de ancho de palabra y emplea instrucciones de tipo I por lo que este procesador recibe el nombre de RV32I, para información adicional sobre la arquitectura RISC-V consultar [8] (Página oficial de RISC-V).

4 DESARROLLO

4.1 DISEÑO

La arquitectura de interconexión WISHBONE diseñada es de bus compartido, esto indica que un conjunto de maestros comparte el bus que se interconecta con un conjunto de esclavos. El objetivo de este trabajo de grado es interconectar 3 procesadores con una memoria, por lo que en la interconexión es necesario el diseño de 3 maestros con un esclavo común.

Cada uno de los maestros recibe los buses de datos, direcciones, así como los habilitadores de lectura y escritura, en los maestros se generan las señales específicas de la arquitectura de interconexión. Además, se implementó un arbitrador que le da acceso a los procesadores al control de la arquitectura de interconexión.

4.1.1 DIAGRAMA DE ENTRADAS Y SALIDAS

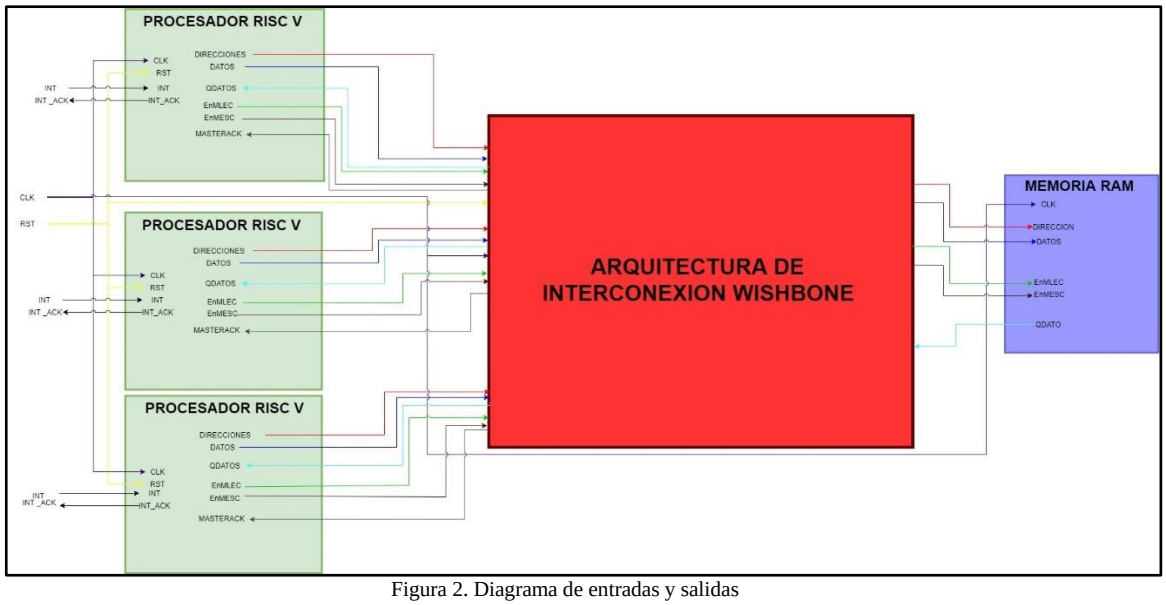

Figura 2. Diagrama de entradas y salidas

En esta sección se presenta la descripción de las señales de entrada y salida que componen la arquitectura WISHBONE, en la figura 2 se presenta el diagrama de entradas y salidas del sistema.

- DATOS: Bus de datos de 32 bits de ancho, para la comunicación de los procesadores con los maestros y la memoria con el esclavo.
- QDATOS: Bus de datos de 32 bits de ancho, para la comunicación de los procesadores con los maestros y la memoria con el esclavo.
- Direcciones: Bus de datos de 13 bits, esto para obtener el máximo número de posiciones de memoria que esta tiene (8192), esta memoria esta embebida en la FPGA.
- EnMLec: Señal de un bit, activa en alto, esta señal indica que la operación que está realizando el procesador es de lectura.
- EnMEsc: Señal de dos bits, esta señal indica que la operación que está realizando el procesador es de escritura, el valor de esta señal habilita la escritura de la memoria por un byte, dos bytes o cuatro bytes.
- CLK: Señal de un bit, esta señal es el reloj de todo el sistema, la frecuencia de este reloj está determinada por la FPGA, para este trabajo de grado la FPGA seleccionada tiene una frecuencia de trabajo de 50MHz, con esto este reloj del sistema tiene un periodo de 20 ns.

- RST: Señal de un bit activa en alto, esta señal sirve para realizar un reinicio general del funcionamiento de todo el sistema, es una señal asíncrona que le permite al usuario reiniciar el sistema en el momento en que lo desee.
- MASTERACK: Señal de un bit activa en alto, sirve para indicarle al procesador que ya dejara de tener control sobre la arquitectura y puede continuar con su funcionamiento normal.

4.1.2 DIAGRAMA DE BLOQUES

En la figura 3 se encuentra el diagrama de bloques del sistema (para ver en mejor resolución ir al ANEXO diagrama de bloques), en este apartado se muestra la descripción de cada uno de los bloques que componen el sistema, su función y las señales internas que posee, el bloque blanco con borde negro corresponde a la interconexión WISHBONE, los bloques verdes corresponden a los 3 procesadores, y el bloque azul corresponde a la memoria del sistema.

Debido a que los tres maestros comparten un esclavo, fue necesario generar selectores que dependen de cuál de los 3 procesadores tiene control de la arquitectura, estos permiten el paso una a la vez de las señales que vienen de los maestros, como la dirección, el dato y los habilitadores de lectura y escritura.

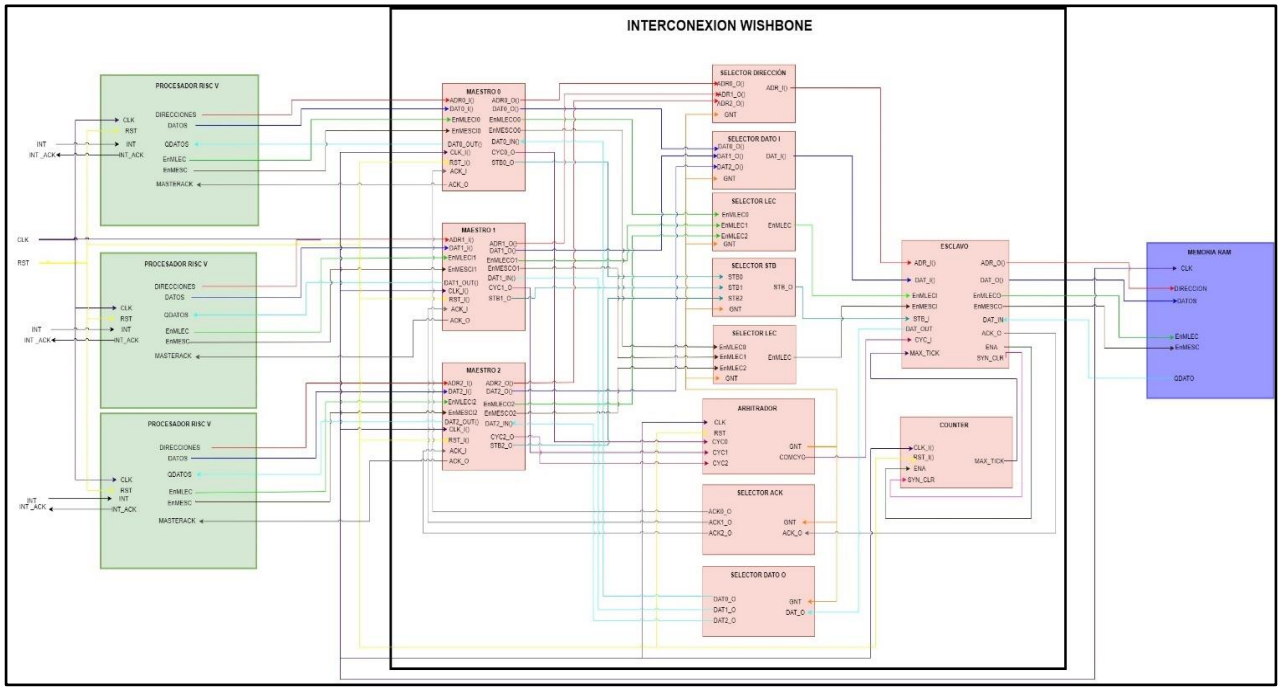

Figura 3. Diagrama de bloques

PROCESADOR: Dispositivo externo a la interconexión WISHBONE basado en la arquitectura RISC-V de tipo RV321, de 32 bits de ancho, *Señales de interacción del procesador:*

Direcciones: Bus de direcciones de 13 bits es de salida, sirve para indicar que posición de memoria se va a leer o en cual se va a escribir, es síncrona.

Datos: Bus de datos de 32 bits bidireccional sirve para cargar datos a la memoria

QDatos: Bus de datos de 32 bits bidireccional sirve para cargar datos que se reciben de la memoria.

EnMLec: Señal de un bit asíncrona es de salida, para indicar lectura de la memoria.

EnMEsc: Señal de dos bits asíncrona es de salida, para indicar escritura de la memoria.

CLK: Señal de un bit, es de entrada y corresponde al reloj del sistema.

RST: Señal de un bit activa en alto, esta señal sirve para realizar un reinicio general del funcionamiento de todo el sistema, es una señal asíncrona que le permite al usuario reiniciar el sistema en el momento en que lo desee.

MASTERACK: Señal de un bit activa en alto, sirve para indicarle al procesador que ya dejara de tener control sobre la arquitectura y puede continuar con su funcionamiento normal.

INT: Señal de tres bits de entrada, esta entrada externa al procesador sirve para indicarle al procesador que hay un periférico que necesita atención.

INT_ACK: Señal de un bit es de salida, sirve para indicarle a los periféricos que ya se está ejecutando una interrupción.

MEMORIA: Dispositivo externo a la interconexión WISHBONE, representa la memoria total del sistema, en ella se almacenan las instrucciones que los procesadores van a realizar, así como los resultados de las operaciones que estos lleven a cabo, esta memoria consta de 8192 posiciones de memoria,

Señales de interacción de la memoria

Direcciones: Bus de direcciones de 13 bits es de entrada sirve para indicar que posición de memoria se va a leer o en cual se va a escribir, es síncrona.

Datos: Bus de datos de 32 bits sirve para cargar datos a la memoria.

QDatos: Bus de datos de 32 bits sirve para para cargar datos que van hacia el procesador.

EnMLec: Señal de un bit asíncrona es de entrada y permite activar la lectura de la memoria.

EnMEsc: Señal de dos bits asíncrona es de entrada y permite activar la escritura de la memoria.

MAESTRO: Bloque encargado de recibir las señales provenientes del procesador y generar la lógica necesaria para general las señales que componen la arquitectura de interconexión WISHBONE

Señales de interacción del maestro

Direcciones: Bus de direcciones proveniente del procesador es de 13 bits bidireccional sirve para indicar que posición de memoria se va a leer o en cual se va a escribir, es síncrona.

Datos: Bus de datos proveniente del procesador o de la memoria de 32 bits bidireccional sirve para enviar datos a la memoria o para cargar al procesador datos que se reciben de la memoria.

EnMLec: Señal de un bit asíncrona es de entrada y permite activar la lectura de la memoria.

EnMEsc: Señal de dos bits asíncrona es de entrada y permite activar la escritura de la memoria.

CYC: Señal de salida es de un bit es propia de la arquitectura de interconexión WISHBONE, en el siguiente capítulo se explicará más a detalle esta señal.

STB: Señal de salida es de un bit es propia de la arquitectura de interconexión WISHBONE, en el siguiente capítulo se explicará más a detalle esta señal.

ACK: Señal de entrada es de un bit, es propia de la arquitectura de interconexión WISHBONE, en el siguiente capítulo se explicará más a detalle esta señal.

ESCLAVO: Bloque encargado de recibir las señales provenientes de los maestros, generar la lógica necesaria para general las señales que componen la arquitectura de interconexión WISHBONE, y enviar las señales y buses de datos hacia la memoria.

Señales de interacción del esclavo

Direcciones: Bus de direcciones proveniente del procesador es de 13 bits bidireccional sirve para indicar que posición de memoria se va a leer o en cual se va a escribir, es síncrona.

Datos: Bus de datos proveniente del procesador o de la memoria de 32 bits bidireccional sirve para enviar datos a la memoria o para cargar al procesador datos que se reciben de la memoria.

EnMLec: Señal de un bit asíncrona es de entrada y permite activar la lectura de la memoria.

EnMEsc: Señal de dos bits asíncrona es de entrada y permite activar la lectura de la memoria.

CYC: Señal de entrada es de un bit es propia de la arquitectura de interconexión WISHBONE, en el siguiente capítulo se explicará más a detalle esta señal.

STB: Señal de entrada es de un bit es propia de la arquitectura de interconexión WISHBONE, en el siguiente capítulo se explicará más a detalle esta señal.

ACK: Señal de salida es de un bit, es propia de la arquitectura de interconexión WISHBONE, en el siguiente capítulo se explicará más a detalle esta señal.

ARBITRADOR: Bloque encargado de generar la lógica necesaria para darle control del bus a cada uno de los procesadores, más adelante se explicará la lógica de funcionamiento de este bloque.

Señales de interacción del arbitrador

CYC: Señal de entrada de un bit proveniente de cada uno de los maestros, sirve para identificar cuál de los procesadores desea acceder a la memoria.

COMCYC: Señal de salida de un bit, cuando esta activa indica que alguno de los procesadores tiene control y acceso a la memoria.

GNT: Señal de salida de dos bits, sirve para indicar cuál de los procesadores tiene acceso a la memoria.

SELECTOR DIRECCION: Bloque combinatorio que recibe la señal GNT del arbitrador, y dependiendo del valor de esta, envía a la salida la dirección de memoria proveniente del procesador que tiene el control de la arquitectura de interconexión.

Señales de interacción del selector dirección

ADR_I: Bus de direcciones de entrada proveniente del maestro es de 13 bits.

ADR_O Bus de direcciones de salida sirve para indicar que posición de memoria se va a leer o en cual se va a escribir.

GNT: Señal de entrada de dos bits, sirve para indicar cuál de los procesadores tiene acceso a la memoria.

SELECTOR DATO I: Bloque combinatorio que recibe la señal GNT del arbitrador, y dependiendo del valor de esta, envía a la salida el dato que va hacia la memoria proveniente del procesador que tiene el control de la arquitectura de interconexión.

Señales de interacción del selector dato I

DAT_O: Bus de datos de entrada proveniente del procesador es de 32 bits.

DAT_I: Bus de datos de salida es de 32 bits para enviar datos a la memoria

GNT: Señal de entrada de dos bits, sirve para indicar cuál de los procesadores tiene acceso a la memoria.

SELECTOR DATO O: Bloque combinatorio que recibe la señal GNT del arbitrador, y dependiendo del valor de esta, envía a la salida el dato que viene desde la memoria al del procesador que tiene el control de la arquitectura de interconexión.

Señales de interacción del selector dato O

DAT_O: Bus de datos de entrada proveniente de la memoria de 32 bits.

DAT_I: Bus de datos de salida de 32 bits sirve para cargar datos al procesador.

GNT: Señal de entrada de dos bits, sirve para indicar cuál de los procesadores tiene acceso a la memoria.

SELECTOR ACK: Bloque combinatorio que recibe la señal GNT del arbitrador, y dependiendo del valor de esta, envía a la salida la señal de acknowledge correspondiente al procesador que tiene el control de la arquitectura de interconexión.

Señales de interacción del selector ack

ACK_I: Señal de entrada de 1 bit, corresponde a la señal de *acknowledge* generada en el esclavo

ACK_O: Señal de salida de 1 bit, que le permite al procesador que tiene acceso a continuar el programa que está ejecutando.

GNT: Señal de entrada de dos bits, sirve para indicar cuál de los procesadores tiene acceso a la memoria.

SELECTOR ESC: Bloque combinatorio que recibe la señal GNT del arbitrador, y dependiendo del valor de esta envía a la salida la señal del activador de escritura correspondiente al procesador que tiene el control de la arquitectura de interconexión.

Señales de interacción del selector esc

EnMescI: Señal de entrada de 2 bits proveniente del maestro.

EnMescO: Señal de salida de 2 bits, permite activar la escritura de la memoria.

GNT: Señal de entrada de dos bits, sirve para indicar cuál de los procesadores tiene acceso a la memoria.

SELECTOR LEC: Bloque combinatorio que recibe la señal GNT del arbitrador, y dependiendo del valor de

esta envía a la salida la señal del activador de escritura correspondiente al procesador que tiene el control de la arquitectura de interconexión.

Señales de interacción del selector lec

EnMLecI: Señal de entrada de 1 bit proveniente del maestro.

EnMLecO: Señal de salida de 1 bit, permite activar la escritura de la memoria.

GNT: Señal de entrada de dos bits, sirve para indicar cuál de los procesadores tiene acceso a la memoria.

COUNTER: Bloque combinatorio que realiza un conteo secuencial de 2 pulsos de reloj, recibe una señal que activa el conteo, y como salida tiene una señal de dos bits con el valor del conteo, y una señal de un bit que indica que el contador llegó a su valor máximo.

Señales de interacción del counter

Ena: Señal de entrada de 1 bit que inicia el conteo de pulsos de reloj.

Syn_clr: Señal de entrada de 1 bit, cuando esta activa devuelve el valor del conteo a 0.

Max_tick: Señal de salida de 1 bit, cuando esta activa indica que el valor del contador llego a su valor máximo.

Counter: Señal de salida de 4 bits, que indica el valor actual del contador.

4.1.3 DESCRIPCION DE LAS SEÑALES DE LA ARQUITECTURA WISHBONE

Para el diseño de la arquitectura de interconexión WISHBONE se tuvieron en cuenta las señales propias que esta tiene y se encuentran en [6], el uso de todas las señales no es obligatorio, en una sección posterior se mostrará cuáles de estas no fueron utilizadas, a continuación, se presenta una breve descripción de las señales propias de la arquitectura de interconexión WISHBONE.

4.1.3.1 SEÑALES COMUNES PARA ESCLAVOS Y MAESTROS.

- CLK_I: Coordina las actividades para la lógica interna dentro de la interconexión WISHBONE
- DAT_(): Vector de entrada usado para pasar datos binarios, los límites del vector están definido por el tamaño del puerto, puede tener un tamaño máximo de 64 bits
- DAT_I(): Vector de entrada usado para pasar datos binarios, los límites del vector están definido por el tamaño del puerto, puede tener un tamaño máximo de 64 bits
- DAT_O(): Vector de salida usado para pasar datos binarios, los límites del vector están definido por el tamaño del puerto, puede tener un tamaño máximo de 64 bits
- RST_I: Señal de entrada que fuerza el reinicio de la interfaz WISHBONE

4.1.3.2 SEÑALES PARA MAESTROS

- ACK_I: Señal de entrada de *acknowledge* que indica la terminación normal de un ciclo de bus.
- ADR_O(): Vector de salida de dirección usado para pasar una dirección binaria, el límite superior del vector corresponde al tamaño de la dirección de los procesadores.
- CYC_O: Señal de salida que indica que un ciclo de bus valido está en progreso, permanece activa mientras la duración de todos los ciclos de bus.
- STALL_I: Señal de entrada que indica que el esclavo no es capaz de aceptar la transferencia que está en la fila de transferencias.
- ERR_I: Señal de entrada que indica una terminación anormal del ciclo.
- LOCK_O: Señal de salida cuando está activa indica que el ciclo de bus actual no puede ser interrumpido, cuando una transferencia ha iniciado la interconexión no otorga el bus a otro maestro hasta que el maestro actual niegue la señal LOCK_O o CYC_O
- RTY_I: Señal de entrada que indica que la interfaz no está lista para enviar o recibir datos y el ciclo de

bus debe hacerse nuevamente

- SEL_O: Vector de salida indica donde datos validos se esperan en la señal (DAT_I()) durante ciclos de lectura y donde está ubicada en la señal (DAT_O()) durante ciclos de escritura.
- STB_O: Señal de salida que indica un ciclo valido de transferencia de datos, el esclavo activa alguna de las señales (ACK_I ERR_I o RTY_I) como respuesta a la afirmación de la señal STB_O.
- WE_O: Señal de salida que indica si el ciclo de bus actual es un ciclo de lectura o de escritura. La señal es negada durante ciclos de lectura y es afirmada durante ciclos de escritura.

4.1.3.3 SEÑALES PARA ESCLAVOS

- ACK_O: Señal de salida, cuando está activada indica la terminación de un ciclo de bus
- ADR_I(): Vector de entrada que es usado para pasar una dirección binaria
- CYC_I: Señal de entrada cuando está activada indica que un ciclo de bus valido está en progreso, la señal permanece afirmada durante la duración de todos los buses de reloj
- STALL_O Señal de salida que indica que el esclavo no puede aceptar transacciones adicionales a su fila de transacciones.
- ERR_O Señal de salida que indica una terminación anormal del ciclo
- LOCK_I Señal de entrada cuando esta afirmada indica que el ciclo de bus actual no puede ser interrumpido, un esclavo que reciba esta señal solo puede ser accedido por un solo maestro hasta que alguna de las señales LOCK_I o CYC_I sean negadas
- RTY_O Señal de salida indica que la interfaz no está lista para aceptar o enviar datos, y el ciclo debería ser reiniciado
- SEL_I Vector de salida indica donde datos validos se esperan en la señal (DAT_I()) durante ciclos de lectura y donde está ubicada en la señal (DAT_O()) durante ciclos de escritura
- STB_I Señal de entrada cuando esta afirmada indica que el esclavo está seleccionado, el esclavo activa alguna de las señales (ACK_O ERR_O o RTY_O) como respuesta a la afirmación de la señal
- WE_I Señal de entrada que indica si el ciclo de bus actual es un ciclo de lectura o de escritura. La señal es negada durante ciclos de lectura y es afirmada durante ciclos de escritura

4.1.4 DESCRIPCIÓN DEL MODO DE INTERCONEXION

El modo de interconexión seleccionado es el modo de bus compartido, esto quiere decir que un conjunto de procesadores comparte la interconexión WISHBONE con un conjunto de memorias y periféricos, en este caso 3 procesadores comparten una memoria por medio de la arquitectura de interconexión WISHBONE.

Dada la imposibilidad de que los 3 procesadores accedan a la memoria de manera simultánea, en el proceso de diseño de la arquitectura es necesario implementar un arbitrador, el cual se encargue de darle el control de la arquitectura de interconexión a cada uno de los procesadores. Existen diversas maneras de diseñar el arbitrador, para el diseño de la arquitectura WISHBONE se escogió el modo *round robin*, en la figura 4 se observa el esquema básico de funcionamiento del árbitro bajo el modo seleccionado.

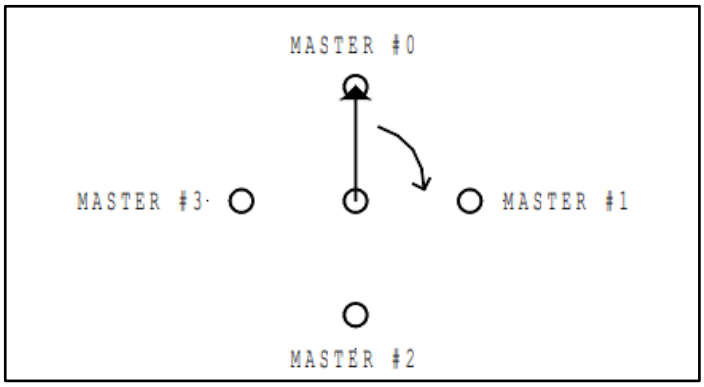

Figura 4. Esquema de árbitro *round robin*. Tomada de [6]

En este modo de diseño, el arbitrador le da el control de la arquitectura a los procesadores por turnos de manera secuencial, el procesador que tiene el control de la arquitectura debe utilizar las señales propias de la arquitectura WISHBONE para indicar en qué momento desea tener el control de la arquitectura, así mismo se utilizó un contador de dos ciclos de reloj para que cada procesador deje de tener el control de la arquitectura.

La señal CYC propia de la arquitectura WISHBONE es la que utiliza cada maestro para indicar que desea tener el control de la arquitectura de interconexión WISHBONE, en la máquina de estados del árbitro él recibe las señales CYC de los 3 procesadores y en un orden secuencial les concede el control de la arquitectura, la salida de la máquina de estados es la señal GNT esta señal indica cuál de los 3 procesadores es el que tiene el control de la arquitectura, en la figura 5 se observa la máquina de estados del funcionamiento del árbitro.

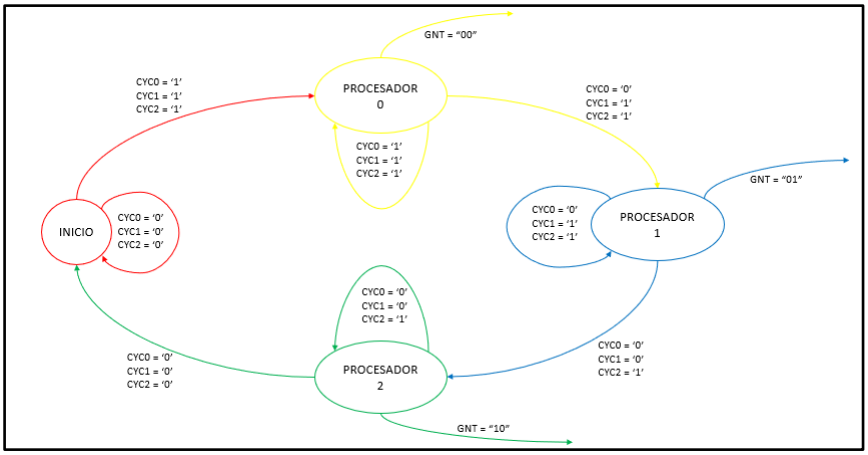

Figura 5. Máquina de estados del arbitrador

En el estado de INICIO las señales de CYC de cada procesador están en 0, indicando que ningún procesador desea tener acceso a la arquitectura, cuando las 3 señales CYC se encuentran en 1 indican que los 3 procesadores desean tener el control de la arquitectura, y es esta la condición de transición para ir al estado PROCESADOR0.

Mientras las señales CYC de los 3 procesadores sigan en 1 la máquina de estados se mantiene en el mismo estado, mientras la máquina esté en este estado la salida de la máquina de estados por medio de la señal GNT es 00 indicando que es el procesador 0 el que tiene el control de la arquitectura. Cuando la señal CYC0 se pone en 0 se cumple la condición de transición para pasar al estado PROCESADOR1.

Mientras las señales CYC de los procesadores 1 y 2 sigan en 1 la máquina de estados se mantiene en el mismo estado, mientras la máquina esté en este estado la salida de la máquina de estados por medio de la señal GNT es 01 indicando que es el procesador 1 el que tiene el control de la arquitectura. Cuando la señal CYC1 se pone en 0 se cumple la condición de transición para pasar al estado PROCESADOR2.

Mientras la señal CYC del procesador 2 sigan en 1 la máquina de estados se mantiene en el mismo estado, mientras la máquina esté en este estado la salida de la máquina de estados por medio de la señal GNT es 10 indicando que es el procesador 2 el que tiene el control de la arquitectura. Cuando las señales CYC de los 3 procesadores se ponen en 0 se cumple la condición de transición para regresar nuevamente al estado de INICIO.

En el modo de interconexión de bus compartido el cual fue diseñado, no es necesaria la utilización de todas las señales propias de la arquitectura interconexión WISHBONE, sin embargo, existe una secuencia básica para ciclos de lectura y escritura, en las figuras 6 y 7 se observa el funcionamiento de la arquitectura WISHBONE en los ciclos de lectura y escritura.

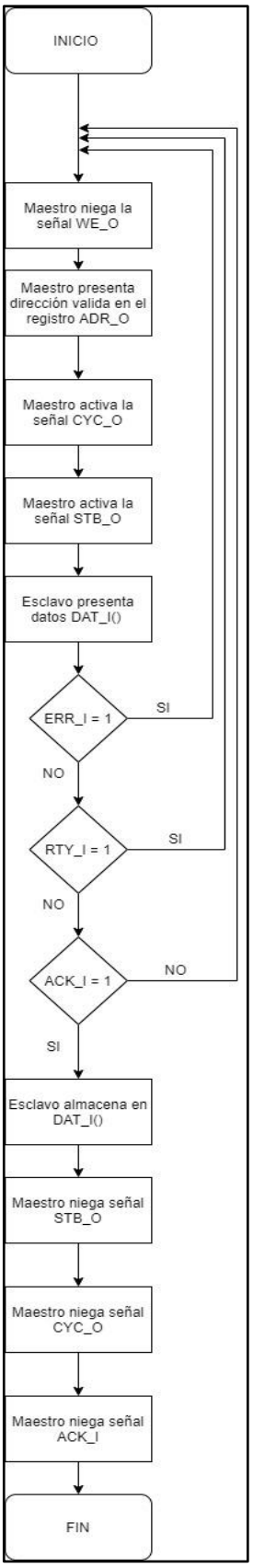

Figura 6. Diagrama de flujo del ciclo de lectura de la interconexión WISHBONE

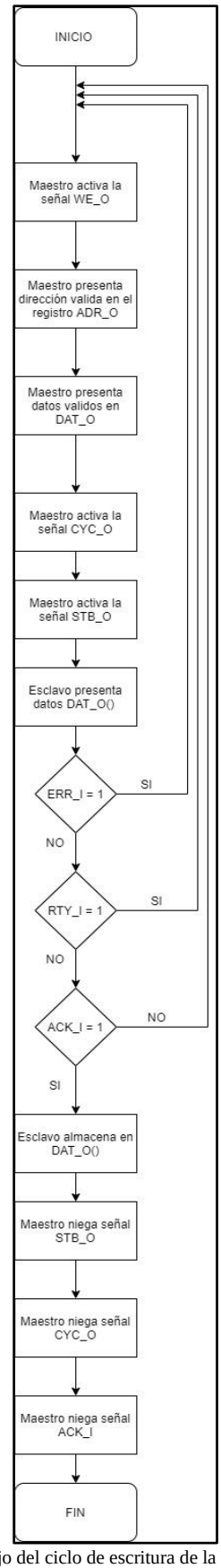

Figura 7. Diagrama de flujo del ciclo de escritura de la interconexión WISHBONE

Cabe destacar que para la arquitectura de interconexión WISHBONE existen diferentes tipos de interconexión fuera del seleccionado (Bus compartido), como punto a punto, flujo de datos, sin embargo, el tipo de interconexión seleccionado es el más útil cuando se desea realizar la interconexión de más de un maestro con más de un esclavo.

Aunque en este método se interconexión cada maestro debe esperar hasta volver a tener el control de la arquitectura. La principal ventaja que tiene este tipo de interconexión se encuentra en que los sistemas son relativamente compactos, y en general requiere una menor cantidad de compuertas logicas y recursos de enrutamiento.

Para la realización de la arquitectura de interconexión de sistema en chip WISHBONE los estándares en ingeniería que se tendrán en cuenta se mencionan en el documento: "WISHBONE System-on-Chip (SoC)Interconnection Architecture for Portable IP Cores"[6], para el manejo de los procesadores se observan las características presentadas en la página web de RISC-V[8].

4.1.5 AHPL

Dentro de la arquitectura de interconexión WISHBONE la señal CYC generada en cada uno de los maestros indica cuál de ellos desea tener acceso y control de la arquitectura. Sin embargo, es necesario que los procesadores tengan una señal que les indique que tienen acceso y control de la arquitectura.

Para llevar esto a cabo fue necesario intervenir el procesador que se está utilizando en la arquitectura de interconexión, y se generó una señal llamada MASTERACK. Esta señal proviene de los maestros y corresponde a la señal de ACK propia de la arquitectura WISHBONE, el objetivo de esta intervención es que el procesador realice una espera en todos los estados que desea acceder a memoria.

Para lograr esto fue necesario modificar el AHPL del procesador RISC-V, específicamente se modificó la máquina de estados del control, los estados modificados fueron todos aquellos estados en los que las señales de lectura y escritura se activan para acceder a la memoria, estas modificaciones se pueden observar en el anexo A. Ver ANEXO A.

4.1.6 ESQUEMATICOS

En el anexo B se observan los circuitos esquemáticos correspondientes a cada uno de los bloques que componen el sistema, los procesadores, los maestros, el esclavo, la memoria, el contador, el arbitrador y los selectores. Ver ANEXO B

4.1.7 DESCRIPCIÓN EN VHDL

En el ANEXO C se encuentra la descripción en *hardware* de todo el sistema por medio de VHDL, para su desarrollo e implementación se utilizó el software Quartus II, esto para verificar que no se generaran errores en el funcionamiento del sistema.

Debido a que fue necesario intervenir el procesador, luego de la modificación al AHPL del sistema fue necesario modificar el VHDL del procesador [7] para poder verificar que las modificaciones realizadas cumplen con el objetivo esperado.

Dado que el procesador seleccionado diseñado por Javier Barbosa [7] ya tiene una completa descripción en hardware por medio de VHDL, todos los bloques adicionales de la arquitectura de interconexión WISHBONE se diseñaron y su descripción en VHDL fue añadida a los bloques que ya estaban diseñados.

Así mismo se utilizaron las herramientas *Waveform* y *ModelSim Starter edition,* esto con el fin de realizar las simulaciones y verificar los resultados de las pruebas. Ver ANEXO C donde está el VHDL del proyecto implementado en Quartus II.

5. PROTOCOLOS DE PRUEBAS Y ANÁLISIS DE RESULTADOS

5.1 PROROCOLOS DE PRUEBAS

Los protocolos de pruebas se diseñaron para realizarlos en simulación, así como en *hardware.* Para realizar las pruebas en *hardware* se utilizó la FPGA DE2-115 Cyclone IV. El reloj de esta tarjeta tiene una frecuencia de 50MHz es decir un periodo de 20ns.

5.2 SIMULACIONES

Las simulaciones de pruebas del sistema se realizaron en 3 fases. La primera fase corresponde a las pruebas del funcionamiento del procesador después de las modificaciones para la compatibilidad con la arquitectura WISHBONE, la segunda fase corresponde a las pruebas del funcionamiento del ejemplo de interconexión de bus compartido que se encuentra en [6] así como una simulación de la arquitectura con un procesador y memoria que se encuentran en [7], y la tercera fase corresponde a las pruebas del funcionamiento del sistema en general de la interconexión WISHBONE con los 3 procesadores y la memoria que se encuentran en [7].

- En la primera fase se realizó la simulación del funcionamiento del procesador[7] esto para determinar que las instrucciones se ejecutan de manera correcta, las instrucciones seleccionadas para esta fase son de tipo S y de I load y observar que los estados por los que pasa la máquina de estados de control corresponden a estas instrucciones seleccionadas.

Por otra parte, se buscaba , se verificó el tiempo que tarda el procesador en cada estado de la máquina de estados, y el tiempo que tarda la memoria en responder a las señales de lectura y escritura.

La primera prueba realizada en esta etapa fue al procesador [8], el objetivo de esta prueba era entender el funcionamiento del procesador y probar los tipos de instrucciones S y I load, en la figura 8 se muestra el resultado de la prueba realizada en la herramienta waveform de Quartus II.

En la prueba se escribió en la posición 0 de memoria que el procesador inicia la ejecución del programa en la posición 1 de memoria, en la posición 1 de memoria la instrucción que se debe ejecutar es una instrucción tipo I que carga el dato que se encuentra en la posición de memoria 5 en el registro 20, en la posición de memoria 2 la instrucción que se debe ejecutar es una instrucción tipo S que carga un byte del registro 20 en la posición de memoria 30.

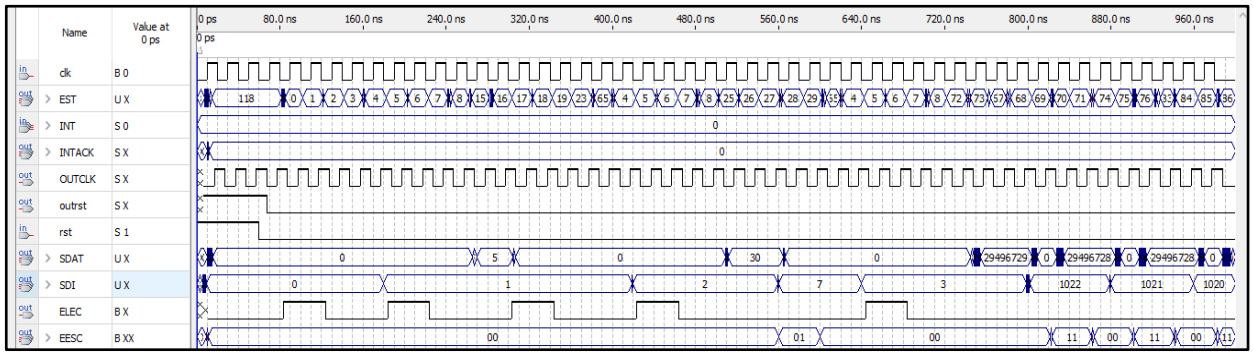

Figura 8. Simulación del funcionamiento del procesador

Se observa que el procesador ejecuta las dos instrucciones correctamente, la señal SDAT muestra primero el valor 5 que corresponde a la posición de memoria que fue designada en la instrucción tipo I, y luego muestra el valor 30 que corresponde a la posición de memoria designada en la instrucción tipo S.

El estado 8 del procesador es el estado en el que se realiza la decodificación de la instrucción, se observa que el procesador luego del estado 8 va a los estados correctos para ejecutar las instrucciones I y S, luego se observa que el procesador va al estado 72 ya que en la posición de memoria 3 se encuentra una instrucción errónea, y luego pasa a la posición 1022 la cual contiene las instrucciones que debe ejecutar cuando se encuentra una instrucción errónea.

Dentro de la arquitectura de interconexión WISHBONE está la señal CYC para indicarle a cada uno de los maestros que tiene el control de la interconexión, pero es necesario que los procesadores tengan una señal que les indique que tienen el control del bus, para esto se modificó el AHPL del procesador (Ver Anexo A) y se agregó una señal llamada MASTERACK.

En todos los estados en los que el procesador va a acceder a memoria, realiza una espera hasta que la señal MASTERACK sea activada indicando que el procesador ya no tiene el control de la arquitectura y puede continuar con su funcionamiento normal.

Para esto se modificó también el VHDL del procesador y se realizó una simulación en la que se buscaba probar que el procesador se quedara en todos los estados que accede a memoria a la espera de que la señal MASTERACK sea activada para continuar con su funcionamiento, en la figura 9 se observa el resultado de la simulación.

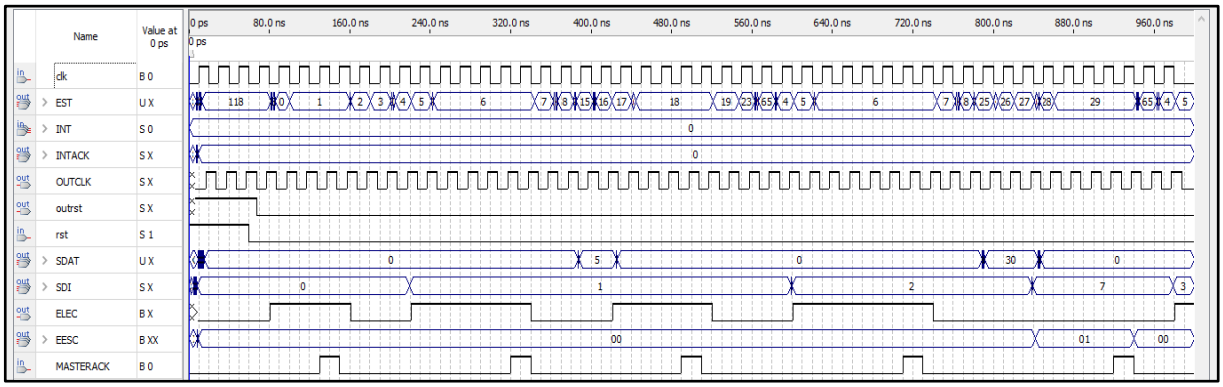

Figura 9. Simulación del procesador incluyendo la señal MASTERACK

En la simulación se observa que en los estados 1,6,18 y 29 (en estos estados el procesador va a acceder a la memoria) el procesador se queda a la espera de que la señal MASTERACK sea activada y solo en ese momento continua con su funcionamiento.

- En la segunda fase se realizó la simulación del ejemplo de bus compartido que se encuentra en [6], el objetivo de esta simulación era entender el funcionamiento de las señales propias de la arquitectura WISHBONE, entender el funcionamiento básico del árbitro, y como la arquitectura realiza la conexión entre los procesadores y un conjunto de memorias.

Así mismo se deseaba ver el funcionamiento de algunos de los bloques del ejemplo, con la memoria y el procesador diseñados en [7], se realizaron simulaciones con un procesador y la memoria diseñada, comprobando que el árbitro solo le da control del bus a este procesador, y las instrucciones se ejecutan correctamente. Estas simulaciones se realizaron en el software *ModelSim Starter edition.*

En el funcionamiento del procesador, en la posición 0 de memoria el usuario debe escribir la posición de memoria de inicio del programa que debe ejecutar, es por esto por lo que se realizó una prueba en la que se eligió que la dirección de inicio de memoria fuera la 2, en la figura10 se observa el resultado de esta prueba realizada mediante ModelSim Starter edition.

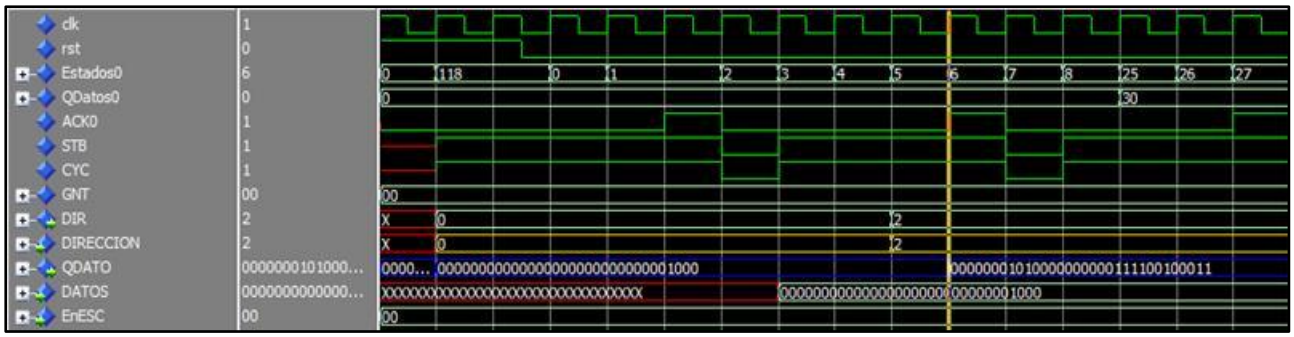

Figura 10. Simulación del procesador con dirección de inicio diferente

Hay que tener en cuenta que la memoria esta direccionada por bytes, esto quiere decir que en la posición 0 de memoria escribir un 4 corresponde a la posición 1 de memoria, para que la dirección de inicio sea la 2 en la posición de memoria hay que escribir el número 0, es por esto qué en la señal QDATO se observa el 8 que está en la posición 0 de memoria, con lo cual se confirma que el resultado que se obtuvo es el esperado.

- En la tercera fase se realizó la simulación de los 3 procesadores con la memoria, el objetivo de esta simulación era comprobar que el árbitro realizaba correctamente su función y les brinda el acceso a los procesadores de manera correcta, de igual forma se buscaba verificar que las señales propias de la interconexión WISHBONE utilizadas en la conexión de los procesadores con la memoria cumplen con su correcto funcionamiento.

Debido a que se tienen 3 procesadores y normalmente cada uno de ellos ejecuta programas distintos, la posición de memoria de inicio de cada uno de ellos debe ser distinta, por lo que se realizó una prueba en la que se agregó una señal externa de entrada mediante la cual el usuario pudiera decidir cuál es la dirección de inicio, el valor de esta señal se estableció en 1, en la figura 11 se observa el resultado de esta simulación.

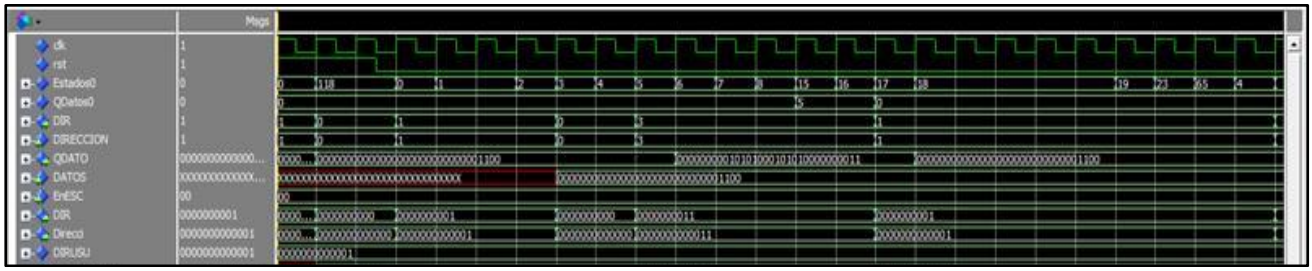

Figura 11. Simulación del procesador con dirección ingresada por el usuario

El resultado obtenido mediante la herramienta ModelSim Altera Starter Edition, muestra que en los estados 0,1,2 que son los estados en los que el procesador recibe la dirección de inicio del programa a ejecutar la posición de memoria en la que se encuentra el sistema es la 1, lo cual corresponde al valor que tiene la señal ingresada por el usuario que es 1.

El diseño del procesador RISC-V que se escogió para este proyecto de grado [7] establece que en la posición de memoria 0 el usuario debe determinar la posición de memoria en la que se encuentra el inicio del programa que se va a ejecutar.

Como cada procesador debe ejecutar un programa distinto es necesario que la posición de memoria de inicio de cada procesador deba ser distinto, para esto fue necesario realizar una intervención en el procesador para que el procesador 0 inicie en la posición 4 de memoria, el procesador 1 inicie en la posición 10 de memoria, y que el procesador 2 inicie en la posición 15 de memoria.

En las primeras simulaciones realizadas los tres procesadores ejecutaban el mismo programa, pero en la realidad cada procesador debe ser independiente razón por la cual cada uno de ellos debe ejecutar un conjunto de instrucciones diferente por lo que, en esta fase, se decidió que cada uno de los procesadores debía ejecutar un programa distinto.

Para el procesador 0 el programa que se ejecuta incorpora instrucciones de tipo S, estas instrucciones sirven para cargar datos a la memoria, en la figura 8 se muestran las instrucciones que ejecuta el procesador 0, con su descripción en assembler, su descripción en binario y las posiciones de memoria en las que se ejecutan estas instrucciones.

DIRECION DE MEMORIA	INSTRUCCIÓN	ENSAMBLE	BINARIO
3	Sumar el valor del registro 0 con el valor inmediato	ADDI 21 0 12	00000000101000000000101010010011
4	Comparación menor entre el valor del registro 21 con el valor inmediato	SLTI 22 21 20	00000001010010101010101100010011
5	Resta entre el registro 21 y el registo 0	SUB 23 21 0	01000000000010101000101110110011

Figura 12. Programa que ejecuta el procesador 0

Para el procesador 1 el programa que se ejecuta incorpora instrucciones de tipo I y de tipo I subrutina, estas instrucciones permiten realizar operaciones aritméticas, así como saltos en instrucciones para realizar subrutinas, en la figura 9 se muestran las instrucciones que ejecuta el procesador 0, con su descripción en assembler, su descripción en binario y las posiciones de memoria en las que se ejecutan estas instrucciones.

DIRECION DE MEMORIA	INSTRUCCIÓN	ENSAMBLE	BINARIO
11	Salto a subrutina a la posicion de memoria 20	JALR 22 0 80	00000101000000000000101101100111
20	Sumar el valor del registro 21 con el valor inmediato	ADDI 21 21 14	00000000111010101000101010010011
21	Fin de subrutina regreso a la posicion 11	JALR 22 0 44	00000010110000000000101101100111
12	Comparación menor con signo entre el valor del registro 21 con el valor inmediato	SLTI 22 21 20	00000001010010101010101100010011

Figura 13. Programa que ejecuta el procesador 1

Para el procesador 2 el programa que se ejecuta incorpora instrucciones de tipo I y tipo R, estas instrucciones permiten realizar operaciones aritméticas entre registros y entre un registro y un número que determina el usuario, en la figura 10 se muestran las instrucciones que ejecuta el procesador 0, con su descripción en assembler, su descripción en binario y las posiciones de memoria en las que se ejecutan estas instrucciones.

DIRECION DE MEMORIA	INSTRUCCIÓN	ENSAMBLE	BINARIO
15	Cargar dato en el registro 20	LB x20 x0 17	00000000 10100000 00010100 00000011
16	Cargar un byte en la posicion de memoria 30	SB x20 36(x0)	00000001 01000000 00001111 00100011

Figura 14. Programa que ejecuta el procesador 2

Luego de definir los programas de cada procesador se procedió a realizar la simulación de los 3 procesadores con la memoria y las señales propias de la arquitectura WISHBONE seleccionadas para este diseño. Las señales propias de la arquitectura seleccionadas son las básicas para su correcto funcionamiento estas señales son las señales CYC, STB, ACK, ADR, DAT y WE.

Esta simulación se realizó mediante la herramienta *ModelSim Starter edition*, en las figuras 11 y 12 se observa el resultado de las simulaciones, en la figura 15 se observan los primeros 500 ns de simulación y en la figura 12 se observan los siguientes 500 ns de simulación.

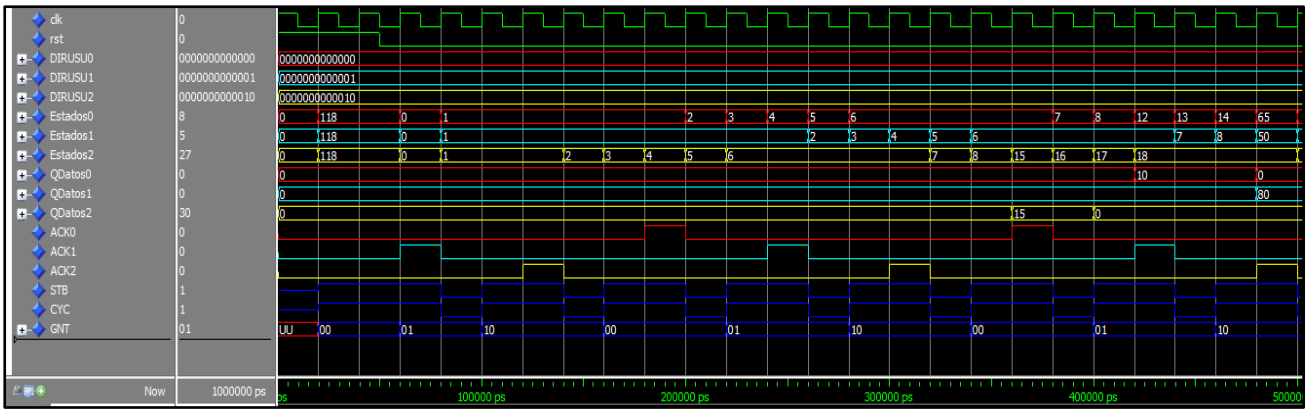

Figura 15. Simulación de la arquitectura WISHBONE parte 1

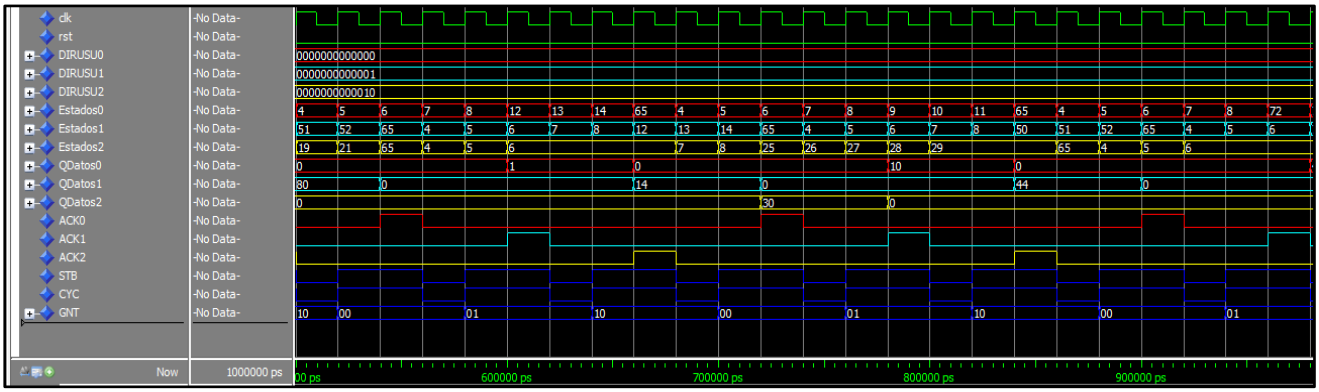

Figura 16. Simulación de la arquitectura WISHBONE parte 2

En color verde se observa primero la señal del reloj del sistema, este reloj tiene una frecuencia de 50Mhz con lo cual se obtiene un periodo de 20ms, la siguiente señal es de reset, se observa que esta señal inicia en 1 esto quiere decir que el sistema aún no ha iniciado, luego cambia a 0 lo que indica que el sistema puede iniciar su operación normal.

En color rojo se observan las señales correspondientes al procesador 0, la señal DIRUSU0 que tiene el valor 0, indica que en la posición de memoria 0 se encuentra la posición de memoria en donde se inicia el programa que debe ejecutar el procesador 0 (posición 4).

La señal Estados0 indica los estados de la máquina de estados de control por los que pasa este procesador, en la figura 15 se observa que pasa por los primeros 8 estados, en este estado decodifica la instrucción que debe ejecutar y pasa al estado 12 debido a que la instrucción que debe ejecutar es de tipo I, y en los siguientes estados ejecuta esta instrucción satisfactoriamente.

En la figura 16 se observa que regresa al estado 4 y pasa por los estados 5, 6, 7 y 8 y realiza la decodificación de la segunda instrucción, pasa nuevamente al estado 12 ya que la instrucción que se debe ejecutar es de tipo I, en los siguientes estados ejecuta la instrucción satisfactoriamente.

Luego de ejecutar esta instrucción, una vez más el procesador regresa al estado 4, pasa por los estados 5, 6, 7 y en el estado 8 decodifica la tercera instrucción, pasa al estado 10 ya que la instrucción que debe ejecutar es de tipo R, en los siguientes estados ejecuta esta instrucción satisfactoriamente.

La señal QDatos0 es la señal de salida del procesador, esta señal muestra el resultado de las instrucciones ejecutadas, en la figura 15 y en la figura 16 se observa que las instrucciones se ejecutaron de manera satisfactoria y que los datos que salen de cada procesador corresponden de manera correcta al resultado esperado según las instrucciones que se deseaban ejecutar.

La señal ACK0 corresponde a la señal interna de la arquitectura de interconexión WISHBONE, esta señal es generada en el bloque esclavo, esta señal le indica a este procesador que dejara de tener el control de la arquitectura y que los datos que provienen de la memoria ya van hacia este procesador.

En color celeste se observan las señales correspondientes al procesador 1, la señal DIRUSU1 que tiene el valor 1, indica que en la posición de memoria 1 se encuentra la posición de memoria en donde se inicia el programa que debe ejecutar el procesador 1 (posición 10).

La señal Estados1 indica los estados de la máquina de estados de control por los que pasa este procesador, en la figura 15 se observa que pasa por los primeros 8 estados, en este estado decodifica la instrucción que debe ejecutar y pasa al estado 50 debido a que la instrucción que debe ejecutar es de tipo I subrutina, y en los siguientes estados ejecuta esta instrucción satisfactoriamente.

En la figura 16 se observa que regresa al estado 4 y pasa por los estados 5, 6, 7 y 8 y nuevamente realiza la decodificación de la segunda instrucción, pasa al estado 12 ya que la instrucción que se debe ejecutar es de tipo I, en los siguientes estados ejecuta la instrucción satisfactoriamente.

Luego de ejecutar esta instrucción, una vez más el procesador regresa al estado 4, pasa por los estados 5, 6, 7 y en el estado 8 decodifica la tercera instrucción, pasa al estado 50 ya que la instrucción que debe ejecutar es de tipo I subrutina ya que debe terminar la subrutina, en los siguientes estados ejecuta esta instrucción satisfactoriamente.

La señal QDatos1 es la señal de salida del procesador, esta señal muestra el resultado de las instrucciones ejecutadas, en la figura 15 y en la figura 16 se observa que las instrucciones se ejecutaron de manera satisfactoria y que los datos que salen de cada procesador corresponden de manera correcta al resultado esperado según las instrucciones que se deseaban ejecutar.

La señal ACK1 corresponde a la señal interna de la arquitectura de interconexión WISHBONE, esta señal es generada en el bloque esclavo, esta señal le indica a este procesador que dejara de tener el control de la arquitectura y que los datos que provienen de la memoria ya van hacia este procesador.

En color amarillo se observan las señales correspondientes al procesador 2, la señal DIRUSU2 que tiene el valor 2, indica que en la posición de memoria 2 se encuentra la posición de memoria en donde se inicia el programa que debe ejecutar el procesador 2 (posición 15).

La señal Estados2 indica los estados de la máquina de estados de control por los que pasa este procesador, en la figura 15 se observa que pasa por los primeros 8 estados, en este estado decodifica la instrucción que debe ejecutar y pasa al estado 15 debido a que la instrucción que debe ejecutar es de tipo I Load, y en los siguientes estados ejecuta esta instrucción satisfactoriamente.

En la figura 16 se observa que regresa al estado 4 y pasa por los estados 5, 6, 7 y 8 y nuevamente realiza la decodificación de la segunda instrucción, pasa al estado 25 ya que la instrucción que se debe ejecutar es de tipo S, en los siguientes estados ejecuta la instrucción satisfactoriamente.

La señal QDatos2 es la señal de salida del procesador, esta señal muestra el resultado de las instrucciones ejecutadas, en la figura 15 y en la figura 16 se observa que las instrucciones se ejecutaron de manera satisfactoria y que los datos que salen de cada procesador corresponden de manera correcta al resultado esperado según las instrucciones que se deseaban ejecutar.

La señal ACK2 corresponde a la señal interna de la arquitectura de interconexión WISHBONE, esta señal es generada en el bloque esclavo, esta señal le indica a este procesador que dejara de tener el control de la arquitectura y que los datos que provienen de la memoria ya van hacia este procesador.

En color azul se observan dos de las señales de la arquitectura de interconexión WISHBONE relevantes para la simulación, la señal CYC y la señal STB, señales que se activan de manera cíclica según el diseño del árbitro, se observa que cada dos ciclos de reloj esta señal se niega y se vuelve a activar indicando que el siguiente procesador desea tener el control de la arquitectura WISHBONE.

La última señal que se observa también en color azul oscuro es la señal GNT esta señal es la salida del arbitrador y esta indica cuál de los procesadores es el que tiene el control de la arquitectura, se observa que funciona correctamente ya que cuando está en 00 el procesador que tiene el control es el 0, cuando es 01 el procesador que tiene el control es el 1 y cuando es 10 el procesador que tiene el control es el 2.

5.3 IMPLEMENTACIÓN EN LA FPGA

Por medio del uso del analizador de estados lógicos, se verificó que en hardware el funcionamiento del sistema. La FPGA seleccionada para este trabajo de grado únicamente tiene 40 pines de los cuales 36 se pueden utilizar para observar las señales en el analizador, debido a esto no fue posible observar todas las señales al mismo tiempo en el analizador, pero se dividieron las pruebas para observar resultados relevantes.

En la primera prueba que se realizó en el analizador de estados, se deseaba observar que los procesadores pasan por los estados que les corresponde, acceden a las posiciones de memoria especificadas, y que el árbitro asigna el control de la arquitectura a los procesadores de manera secuencial según lo diseñado. En las figuras 17 y 18 se observa el resultado de esta prueba.

Para las siguientes pruebas, se tomaron las señales individuales de cada uno de los procesadores, esto con el fin de verificar que cada uno de los procesadores ejecuta correctamente el programa que se designó. Estas pruebas permitieron observar una mayor cantidad de señales y con esto comprobar que el resultado obtenido es el esperado.

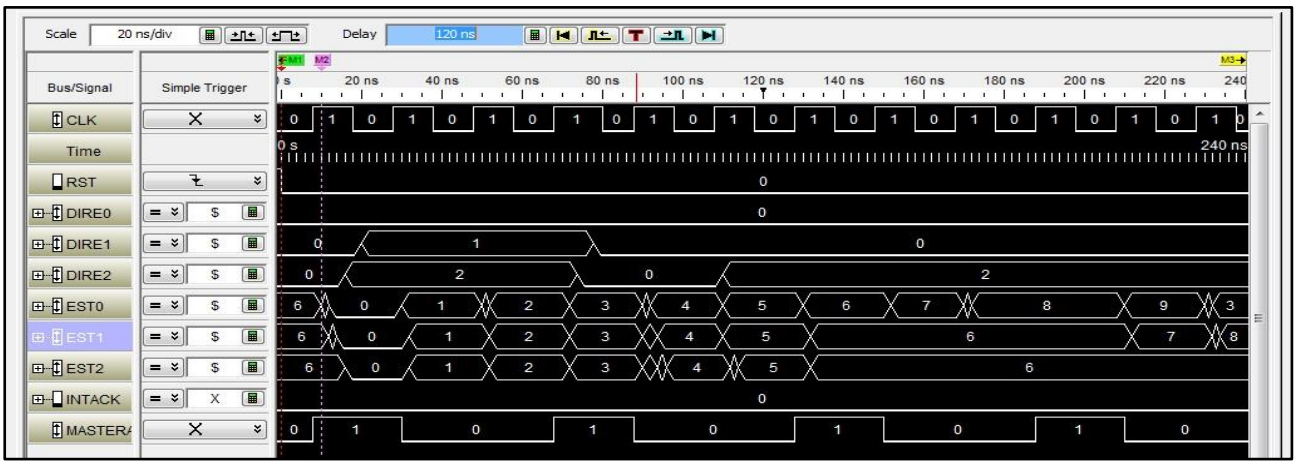

Figura 17. Implementación de los tres procesadores parte 1

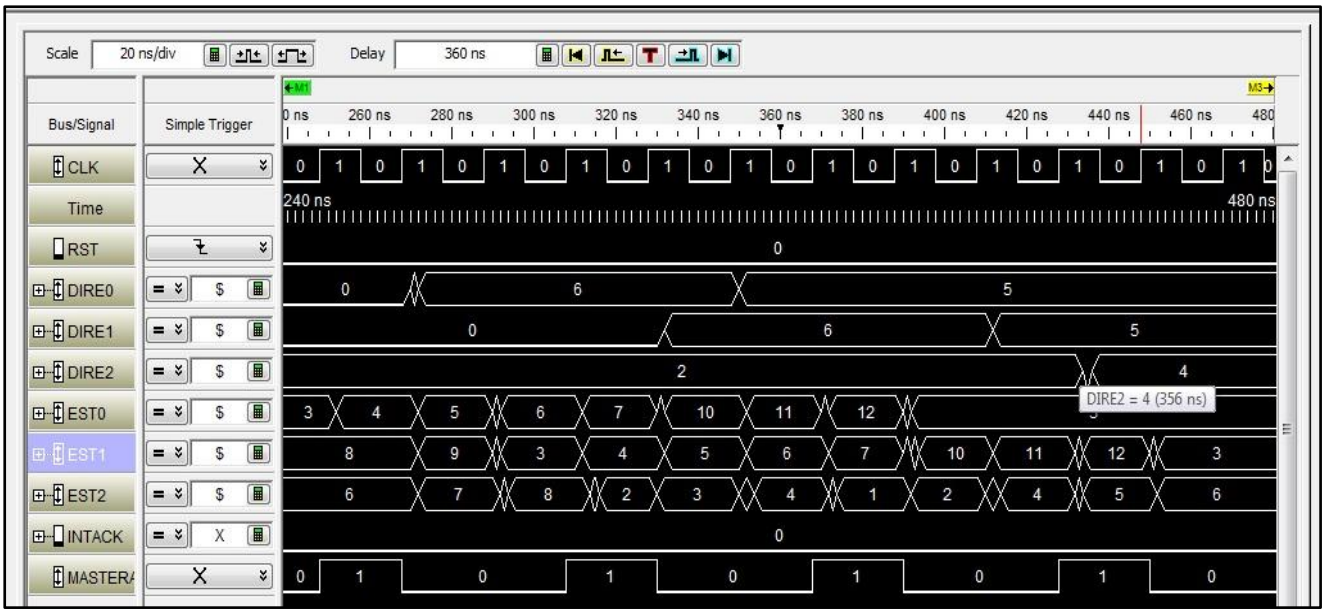

Figura 18. Implementación de los tres procesadores parte 2

En la figura 17 se observa que cada uno de los procesadores accede a la posición de memoria de inicio seleccionada para cada uno, para el procesador 0 la posición de memoria 0, para el procesador 1 la posición de memoria 1 y el procesador 2 la posición de memoria 2.

Como se ha explicado en secciones anteriores el diseño del árbitro concede la arquitectura de interconexión a cada procesador de manera secuencial, y cada procesador mantiene el control de la arquitectura WISHBONE hasta que el maestro niegue la señal CYC indicando que dejara de tener el control de la arquitectura.

Basados en esto se observa que el funcionamiento del árbitro es correcto, en la figura 17 se observa que los 3 procesadores pasan por los estados del 0 al 5 y se quedan en el estado 6 ya que en este los procesadores desean acceder a la memoria y se quedan en este estado a la espera de recibir la señal de ACK para continuar con el programa establecido.

Se observa que a los 130 ns los 3 procesadores llegan al estado 6, un ciclo ciclos de reloj después es decir a los 150ns el procesador 0 recibe la señal de ACK dejando de tener control de la arquitectura avanzando su funcionamiento pasando al estado 7. Dos ciclos de reloj después el procesador 1 recibe la señal de ACK que dura un ciclo de reloj es decir a los 210 ns, dejando de tener el control de la arquitectura y continuando su funcionamiento pasando al estado 7.

En la figura 18 se observa que dos ciclos de reloj después del ACK del procesador 1, el procesador 2 recibe la señal de ACK que dura un ciclo de reloj esto se produce a los 270 ns en este momento deja de tener el control de la arquitectura y continua con su correcto funcionamiento pasando al estado 7.

Luego de verificar que el árbitro concede el acceso a la arquitectura de manera correcta a los procesadores, que pasan por los estados de manera correcta y por las posiciones de memoria establecidas en los programas, se procedió a realizar una prueba del funcionamiento del procesador 0. En las siguientes figuras se observa el resultado de esta prueba.

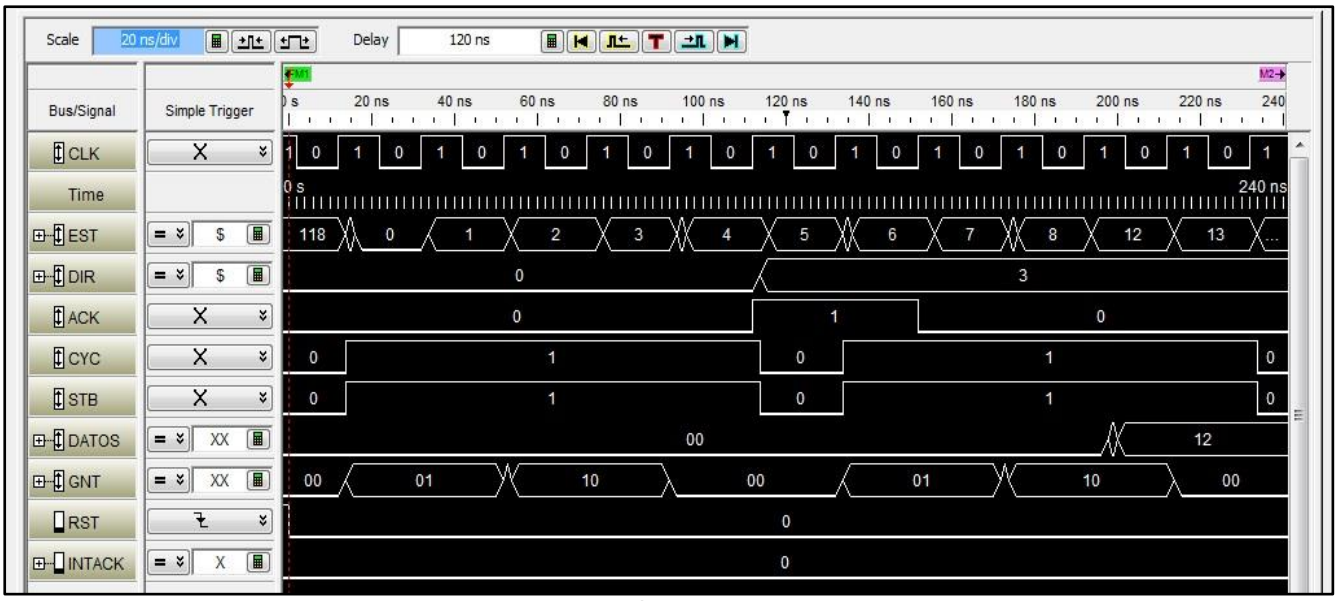

Figura 19. Implementación del procesador 0 parte 1.

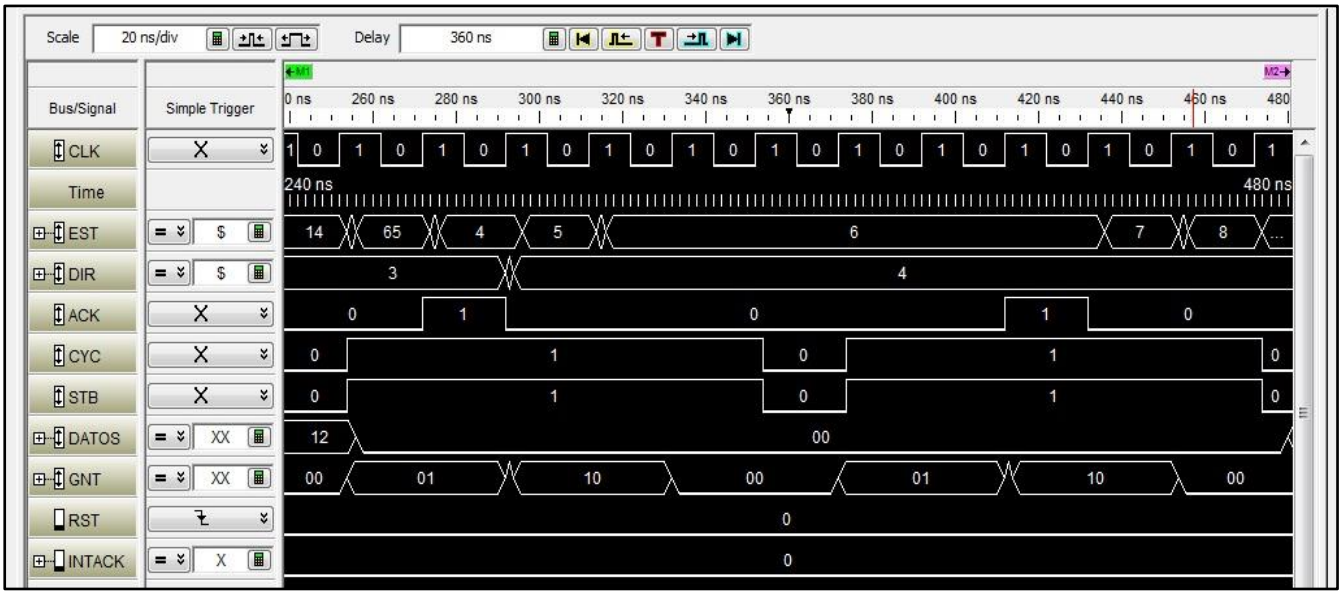

Figura 20. Implementación del procesador 0 parte 2

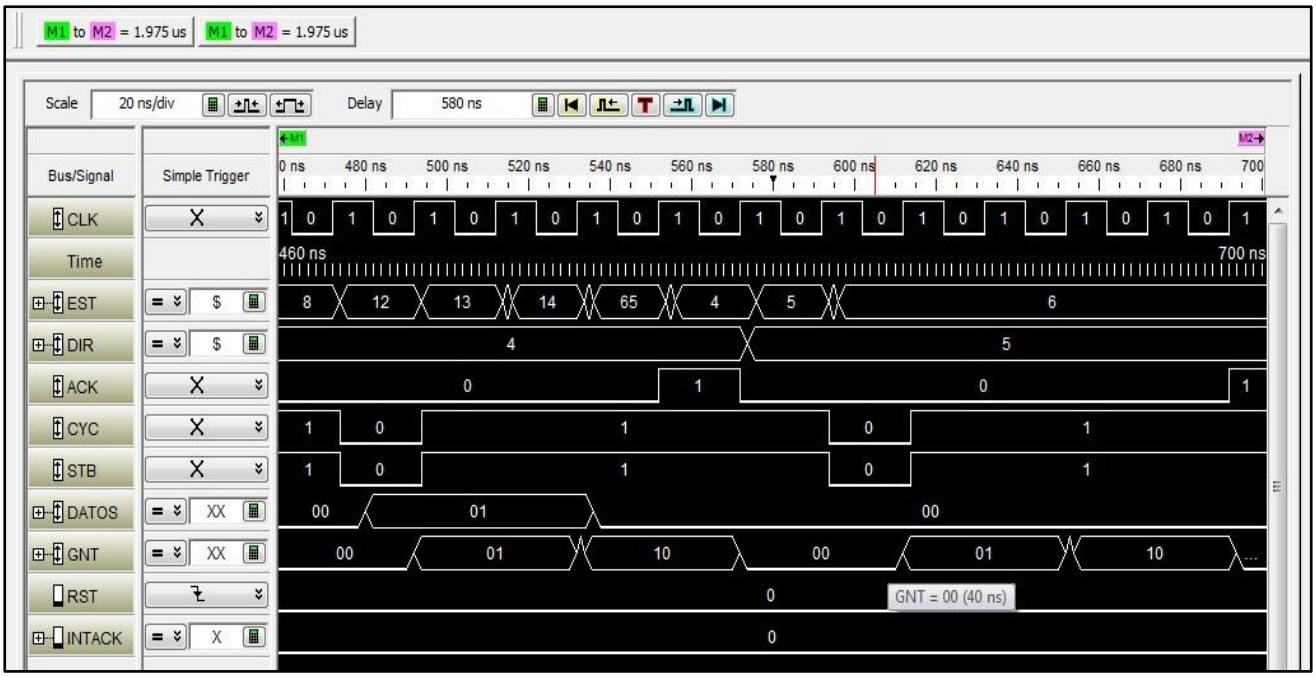

Figura 21. Implementación del procesador 0 parte 3.

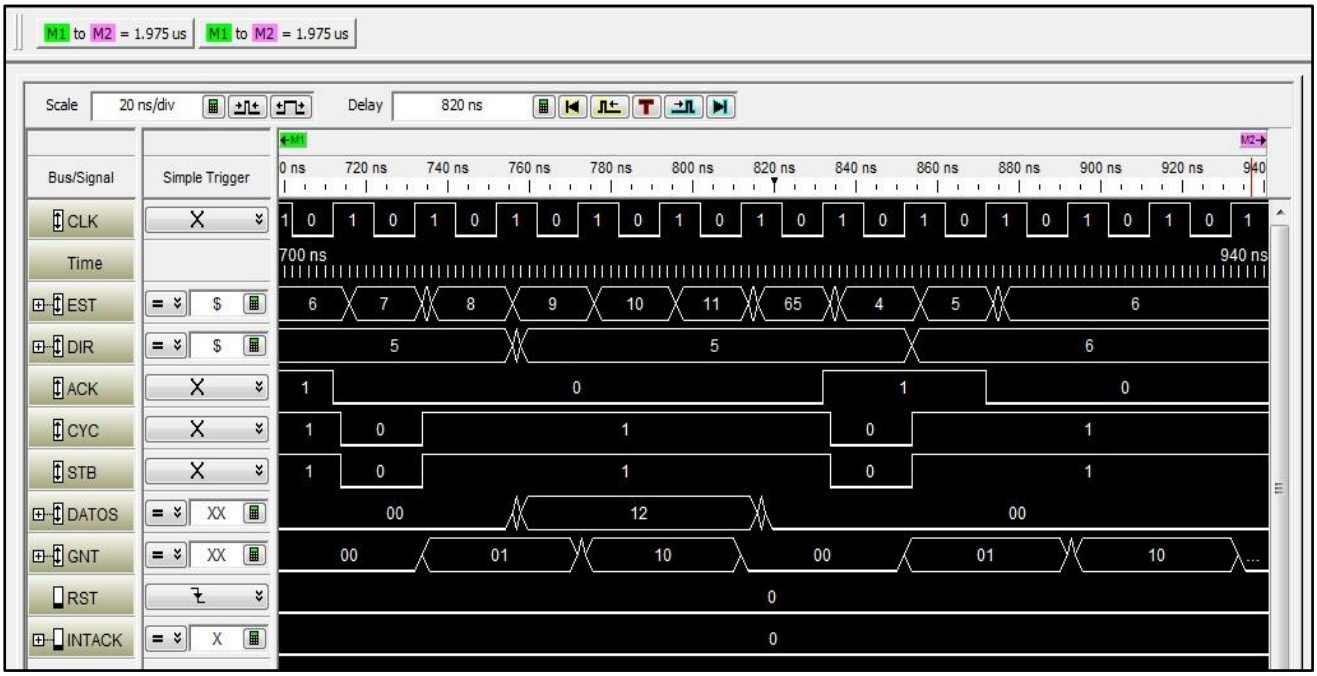

Figura 22. Implementación del procesador 0 parte 4.

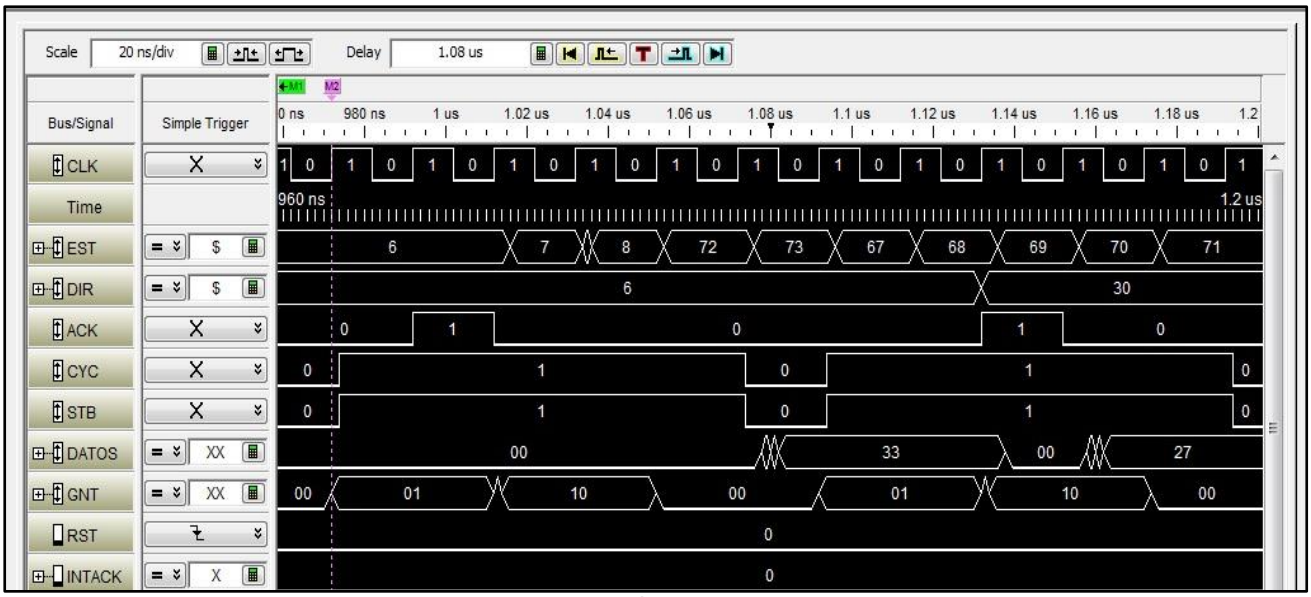

Figura 23. Implementación del procesador 0 parte 5.

En la figura 19 se observa que cuando la señal de reset toma el valor de 0 el procesador inicia su funcionamiento, pasa por los estados 0 al 6 en ese momento recibe la señal de ACK, pasa a los estados 7 y 8 y decodifica la instrucción que debe realizar y por esto pasa al estado 12. También se observa que el procesador inicia en la posición 0 donde debe buscar la posición de memoria donde se encuentra la primera instrucción del programa que debe ejecutar, la cual es la posición 3. A los 200 ns se observa que el bus de datos toma el valor de 12 confirmando que la instrucción se ejecutó de manera satisfactoria.

En la figura 20 se evidencia que el procesador pasa al estado 65 para verificar si se generó una interrupción, lo cual no ocurre por lo que regresa al estado 4 y nuevamente se queda en el estado 6 hasta que recibe la señal de ACK, y continua a los estados 7 y 8 donde decodifica la nueva instrucción a ejecutar, el bus de direcciones pasa a la posición de memoria 4 para ejecutar la segunda instrucción del programa establecido.

En la figura 21 se observa que el procesador luego de decodificar la instrucción pasa al estado 12 nuevamente a ejecutar la segunda instrucción, a los 480 ns se observa que el bus de datos toma el valor de 1 lo cual permite afirmar que la instrucción se ejecutó de manera correcta.

Al finalizar la instrucción el bus de estados pasa luego al estado 65 a verificar si se produjo una interrupción, lo cual no ocurre por lo que regresa al estado 4 y va al estado 6 nuevamente a esperar la señal de ACK, el bus de direcciones pasa a la posición de memoria 5 en donde se encuentra la tercera instrucción del programa a ejecutar.

En la figura 22 se muestra que el procesador recibe la señal de ACK por lo que el bus de estados pasa al estado 7 y luego al 8 a decodificar la tercera instrucción que debe ejecutar el procesador es por esto que pasa al estado 9, a los 760 ns el bus de datos toma el valor de 12 esto demuestra que la instrucción se ejecutó de manera correcta. Al finalizar la instrucción el bus de estados pasa luego al estado 65 a verificar si se produjo una interrupción, lo cual no ocurre por lo que regresa al estado 4. El bus de direcciones pasa a la posición de memoria 6 en busca de una nueva instrucción a ejecutar.

En la figura 23 se observa que el procesador recibe nuevamente la señal de ACK y en el estado 8 decodifica la instrucción que debe ejecutar, pero debido a que la instrucción es errónea el estado al que pasa es al 72 que ejecuta una rutina de excepción, es por esto que se observa que el bus de direcciones se va a la posición 1022 (se observa el valor de 30 debido a que por la limitación del número de pines no fue posible observar todos los bits del bus de direcciones).

El resultado de esta prueba demostró que el procesador 0 ejecuta correctamente el programa designado, y que por medio de la arquitectura el tiempo de ejecución del programa fue de aproximadamente 900 ns. La siguiente prueba correspondía a verificar el funcionamiento del procesador 1, en las siguientes figuras se observa el resultado de esta prueba.

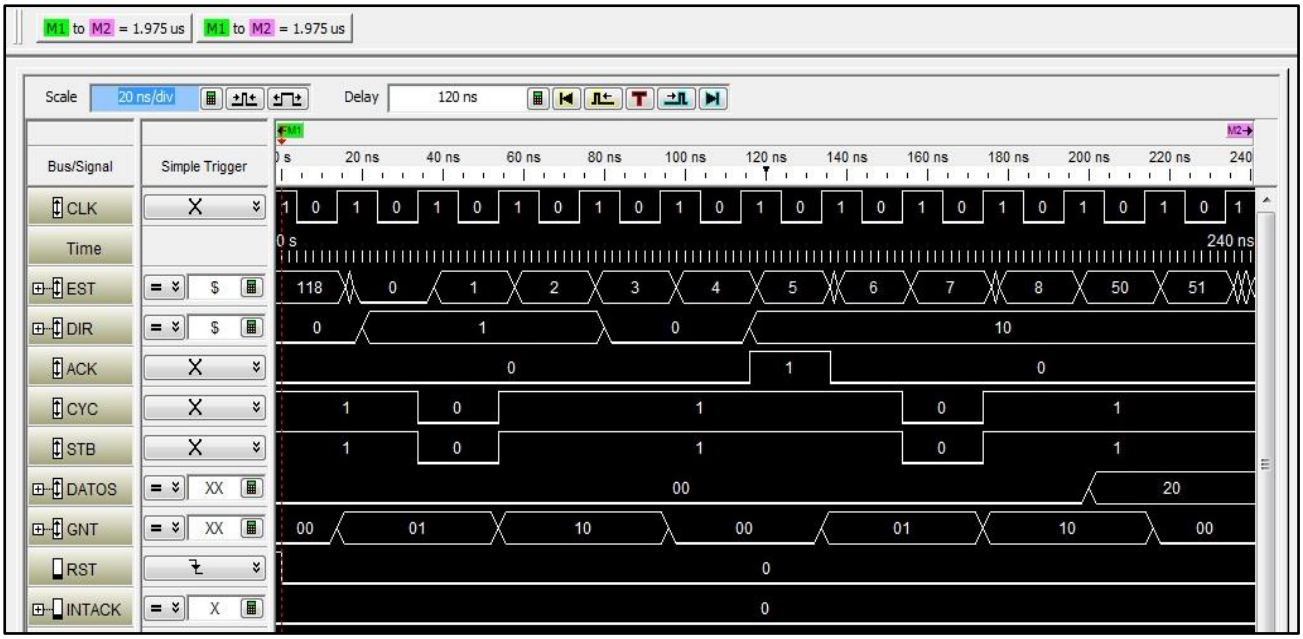

Figura 24. Implementación del procesador 1 parte 1.

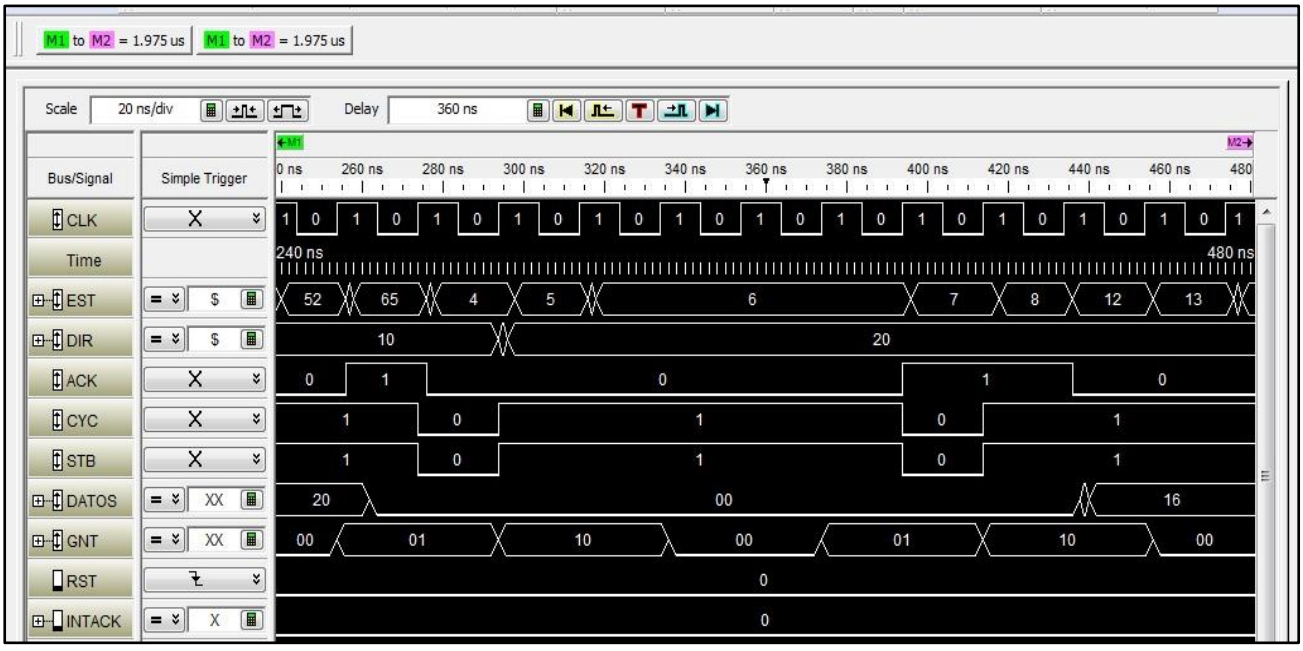

Figura 25. Implementación del procesador 1 parte 2.

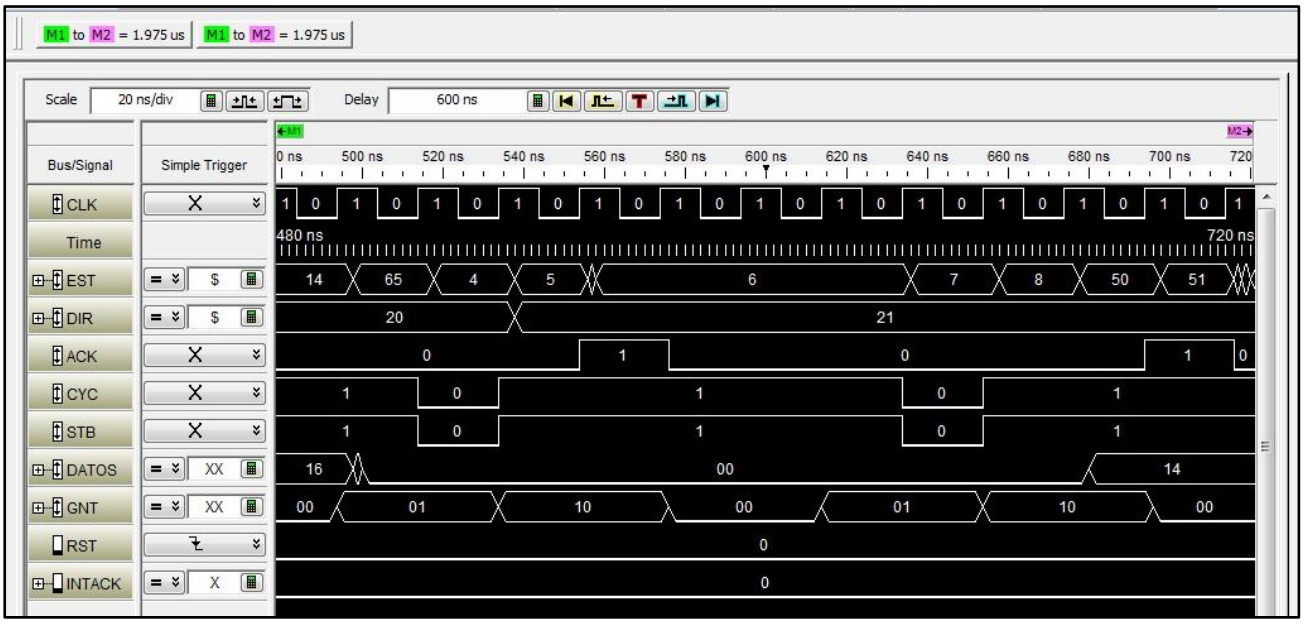

Figura 26. Implementación del procesador 1 parte 3.

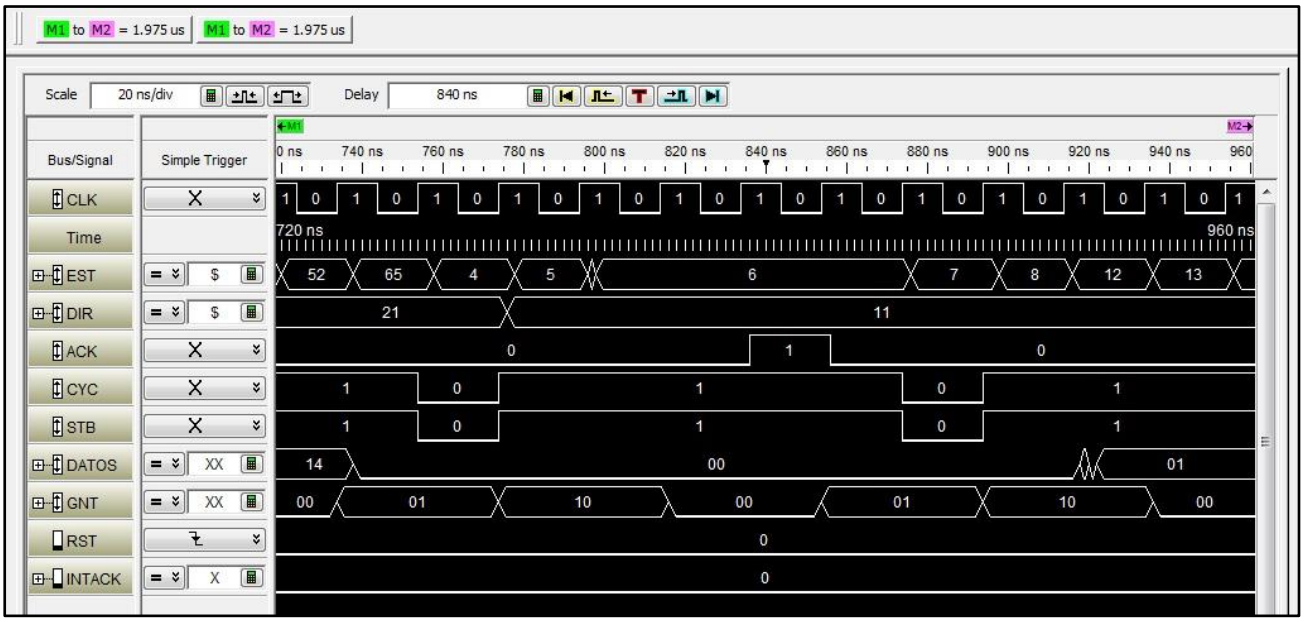

Figura 27. Implementación del procesador 1 parte 4.

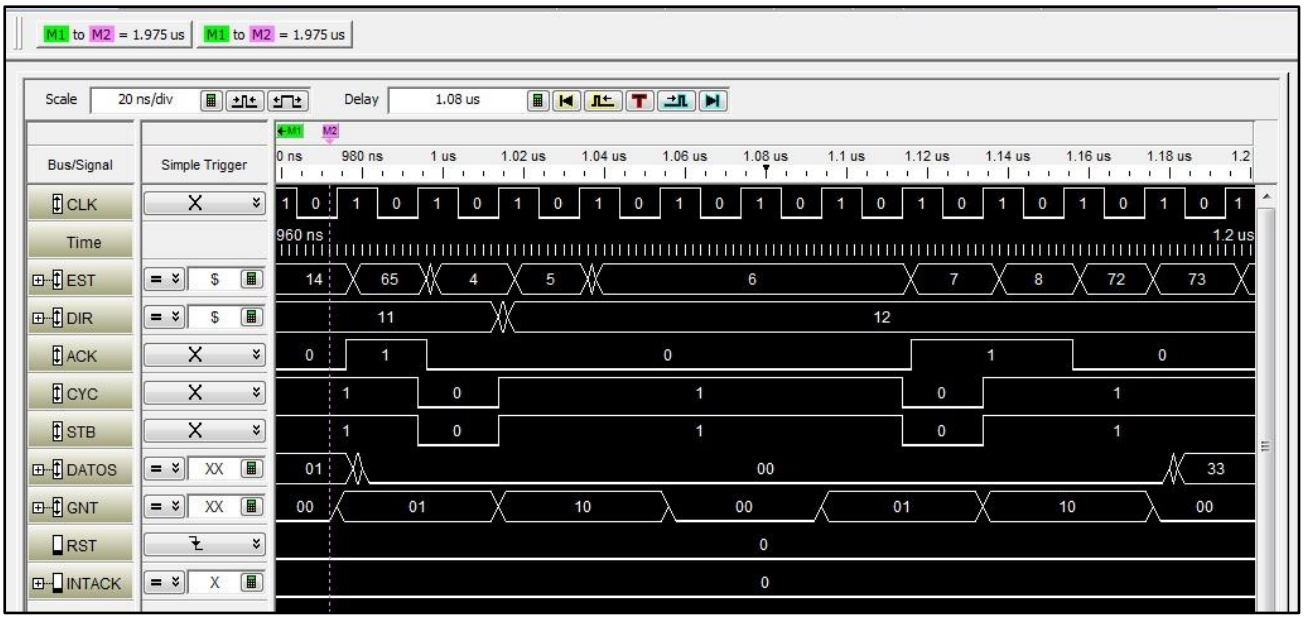

Figura 28. Implementación del procesador 1 parte 5.

En la figura 24 se observa que cuando la señal de reset toma el valor de 0 el procesador inicia su funcionamiento, pasa por los estados 0 al 6 en ese momento recibe la señal de ACK, pasa a los estados 7 y 8 y decodifica la instrucción que debe realizar y por esto pasa al estado 50. También se observa que el procesador inicia en la posición 0 donde debe buscar la posición de memoria donde se encuentra la primera instrucción del programa que debe ejecutar, la cual es la posición 10. A los 200 ns se observa que el bus de datos toma el valor de 20 confirmando que la instrucción se ejecutó de manera satisfactoria.

En la figura 25 se evidencia que el procesador pasa al estado 65 para verificar si se generó una interrupción, lo cual no ocurre por lo que regresa al estado 4 y nuevamente se queda en el estado 6 hasta que recibe la señal de ACK, y continua a los estados 7 y 8 donde decodifica la nueva instrucción a ejecutar, el bus de direcciones pasa a la posición 20 para ejecutar la segunda instrucción del programa establecido.

A los 240 ns se observa que el bus de datos toma el valor de 20, la primera instrucción a ejecutar es un salto a la posición de memoria 20, y teniendo en cuenta que la memoria esta divida por bytes el valor que se esperaría en el bus de datos es el número 80 que indica la posición de memoria 20,

Sin embargo, debido a la limitante de los pines de la FPGA solo se sacaron los primeros 5 bits de la señal de datos, los primeros 5 bits del número 80 corresponden a 10100 el cual corresponde al número 20 en decimal, esto comprueba que la instrucción se ejecutó correctamente.

Se observa que el procesador regresa al estado 4, va al estado 6 recibe la señal de ACK y luego pasa a los estados 7 y 8 y decodifica la instrucción por lo que pasa al estado12 para ejecutar la segunda instrucción, a los 440 ns se observa que el bus de datos toma el valor de 16 lo cual permite afirmar que la instrucción se ejecutó de manera correcta.

En la figura 26 se puede ver que al finalizar la instrucción el bus de estados pasa luego al estado 65 a verificar si se produjo una interrupción, lo cual no ocurre por lo que regresa al estado 4 y va al estado 6 nuevamente a esperar la señal de ACK, el bus de direcciones pasa a la posición de memoria 21 en donde se encuentra la tercera instrucción del programa a ejecutar, en el estado 8 decodifica nuevamente la instrucción y por esto va al estado 50.

A los 680 ns se observa que el bus de datos toma el valor de 14, la tercera instrucción a ejecutar es un salto a la posición de memoria 11, y teniendo en cuenta que la memoria esta divida por bytes el valor que se esperaría en el bus de datos es el número 44 que indica la posición de memoria 11.

Sin embargo, debido a la limitante de los pines de la FPGA solo se sacaron los primeros 5 bits del bus de datos, los primeros 5 bits del número 44 corresponden a 01110 el cual corresponde al número 14 en decimal, esto comprueba que la instrucción se ejecutó correctamente.

En la figura 27 se puede ver que al finalizar la instrucción el bus de estados pasa luego al estado 65 a verificar si se produjo una interrupción, lo cual no ocurre por lo que regresa al estado 4 y va al estado 6 nuevamente a esperar la señal de ACK, el bus de direcciones pasa a la posición de memoria 11 en donde se encuentra la tercera instrucción del programa a ejecutar, en el estado 8 decodifica nuevamente la instrucción y por esto va al estado 12. A los 920 ns se observa que el bus de datos toma el valor de 1 lo cual permite afirmar que la instrucción se ejecutó de manera correcta.

En la figura 28 se puede ver que al finalizar la instrucción el bus de estados pasa luego al estado 65 a verificar si se produjo una interrupción, lo cual no ocurre por lo que regresa al estado 4 y va al estado 6. Se observa que el procesador recibe nuevamente la señal de ACK.

En el estado 8 decodifica la instrucción que debe ejecutar, pero debido a que la instrucción es errónea el estado al que pasa es al 72 que ejecuta una rutina de excepción, es por esto que se observa que el bus de direcciones se va a la posición 1022 (se observa el valor de 30 debido a que por la limitación del número de pines no fue posible observar todos los bits del bus de direcciones).

El resultado de esta prueba demostró que el procesador 1 ejecuta correctamente el programa designado, y que por medio de la arquitectura el tiempo de ejecución del programa fue de aproximadamente 980 ns. La siguiente prueba correspondía a verificar el funcionamiento del procesador 2, en las siguientes figuras se observa el resultado de esta prueba.

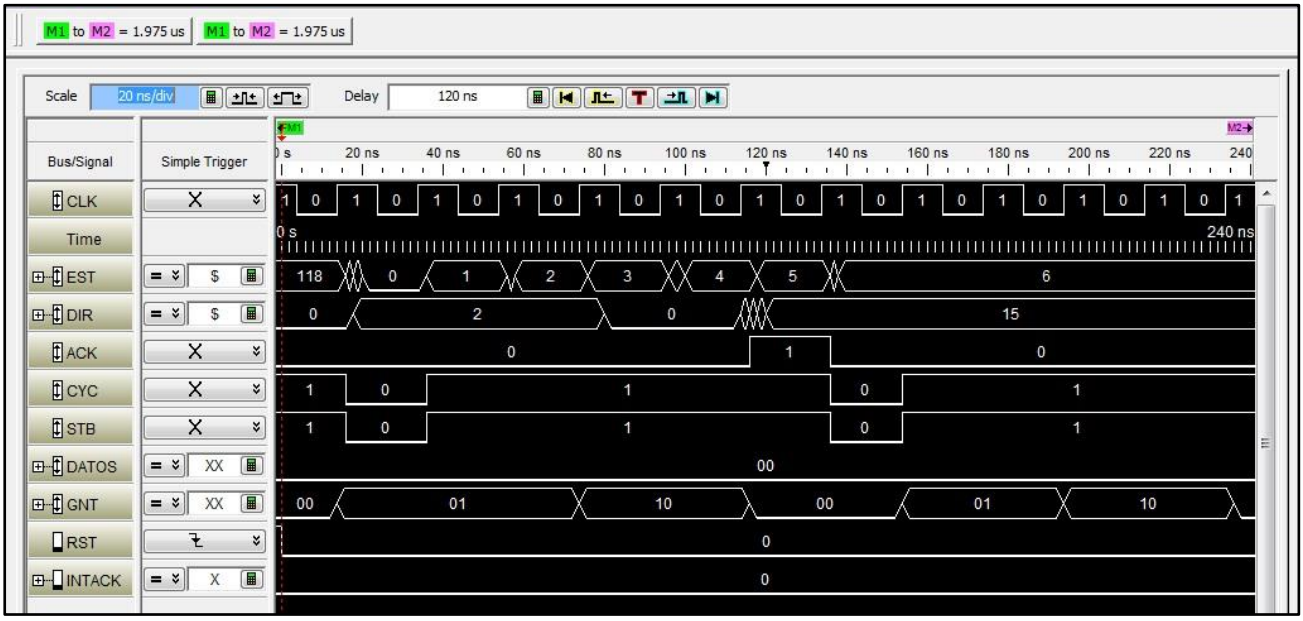

Figura 29. Implementación del procesador 2 parte 1.

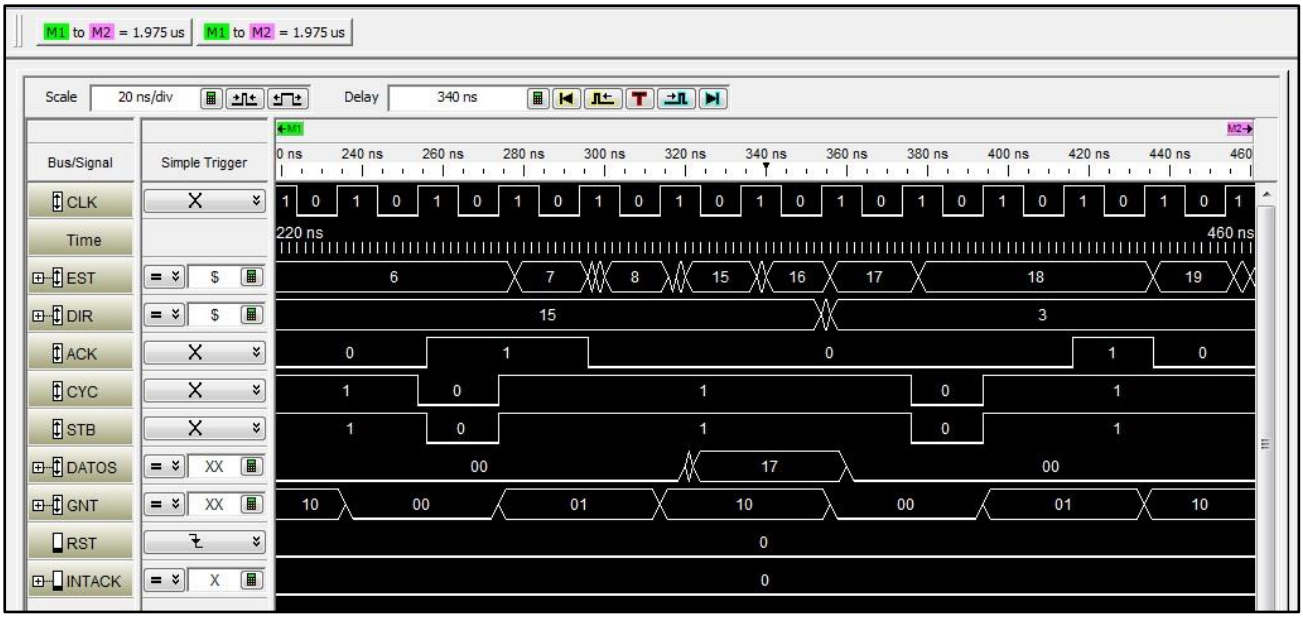

Figura 30. Implementación del procesador 2 parte 2.

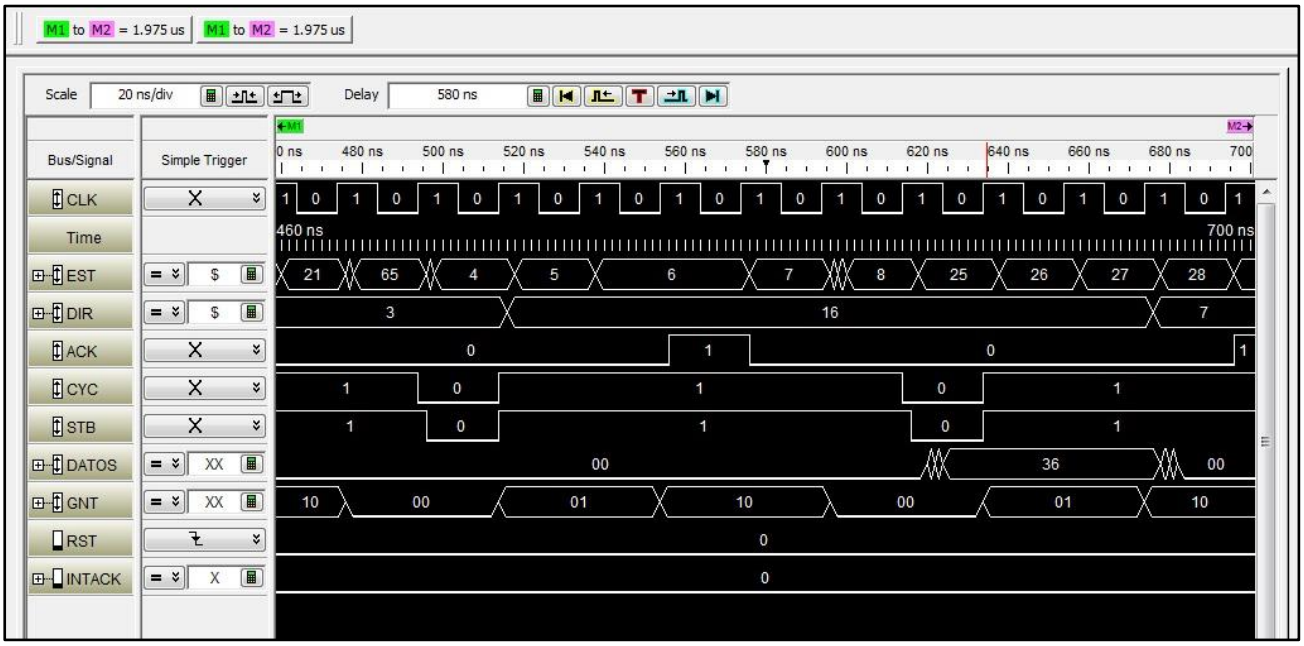

Figura 31. Implementación del procesador 2 parte 3.

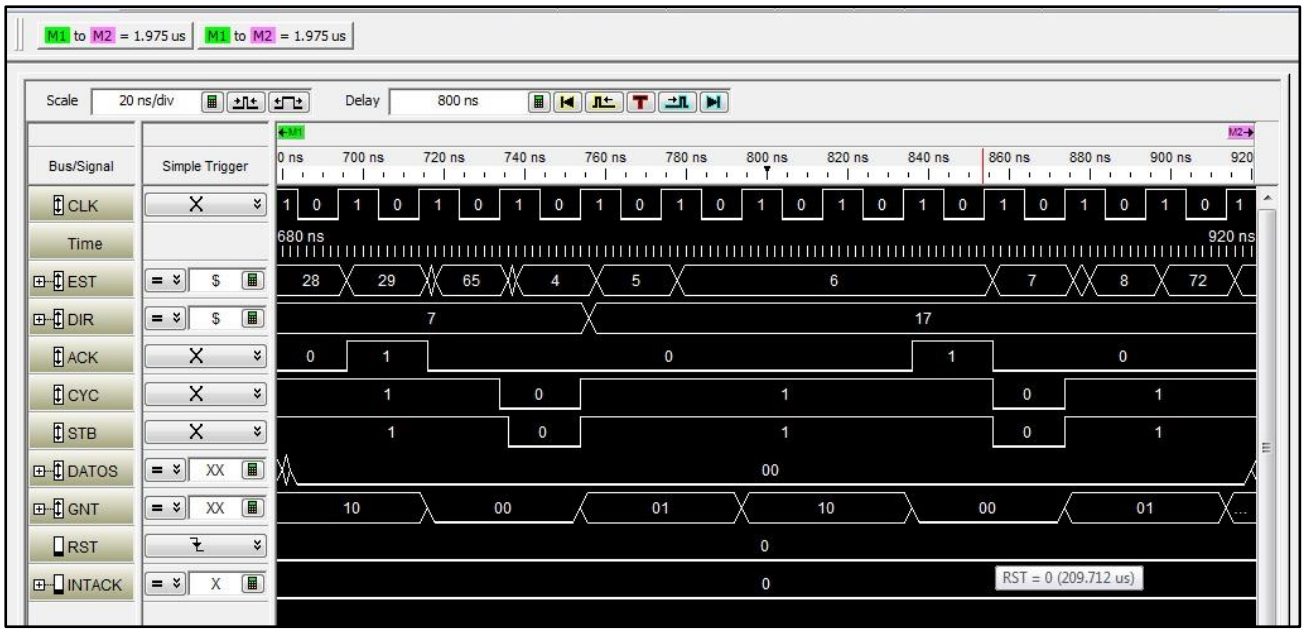

Figura 32. Implementación del procesador 2 parte 4.

En la figura 29 se observa que cuando la señal de reset toma el valor de 0 el procesador inicia su funcionamiento, pasa por los estados 0 al 6 y se queda en este a la espera de recibir la señal de ACK. También se observa que el procesador inicia en la posición de memoria 0 donde debe buscar la posición de memoria donde se encuentra la primera instrucción del programa que debe ejecutar, la cual es la posición 15.

En la figura 30 se evidencia que el procesador recibe la señal de ACK por lo que el bus de estados pasa al estado 7 y luego al 8 a decodificar la primera instrucción que debe ejecutar por esto pasa al estado 15, debido a que las instrucciones que ejecuta este procesador son para escribir datos en memoria se observa que el procesador nuevamente en el estado 18 espera a recibir la señal de ACK para continuar con su funcionamiento.

El procesador recibe nuevamente la señal de ACK para avanzar al estado 19 y concluir así con la ejecución de la instrucción, a los 320 ns se observa que el bus de datos toma el valor de 17 lo cual permite afirmar que la instrucción se ejecutó de manera correcta.

En la figura 31 se observa que el procesador pasa al estado 65 para verificar si se generó una interrupción, lo cual no ocurre por lo que regresa al estado 4 y nuevamente se queda en el estado 6 hasta que recibe la señal de ACK, y continua a los estados 7 y 8 donde decodifica la nueva instrucción a ejecutar, el bus de direcciones pasa a la posición de memoria 16 para ejecutar la segunda instrucción del programa establecido.

Luego de decodificar la instrucción pasa al estado 25 para ejecutar la segunda instrucción del programa establecido, a los 620 ns se observa que el bus de datos toma el valor de 36 lo cual permite afirmar que la instrucción se ejecutó de manera correcta.

En la figura 32 se evidencia que el procesador pasa al estado 65 para verificar si se generó una interrupción, lo cual no ocurre por lo que regresa al estado 4 y nuevamente se queda en el estado 6 hasta que recibe la señal de ACK, y continua a los estados 7 y 8 donde decodifica la nueva instrucción a ejecutar.

Sin embargo, debido a que la instrucción es errónea el estado al que pasa es al 72 que ejecuta una rutina de excepción, es por esto por lo que se observa que el bus de direcciones se va a la posición 1022 (se observa el valor de 30 debido a que por la limitación del número de pines no fue posible observar todos los bits del bus de direcciones).

El resultado de esta prueba demostró que el procesador 1 ejecuta correctamente el programa designado, y que por medio de la arquitectura el tiempo de ejecución del programa fue de aproximadamente 740 ns.

Flow Summary	
Flow Status	Successful - Wed Sep 15 12:07:12 2021
Quartus II 64-Bit Version	13.1.0 Build 162 10/23/2013 SJ Web Edition
Revision Name	PROCESADOR
Top-level Entity Name	PROCESADOR
Family	Cyclone IV E
Device	EP4CE115F29C7
Timing Models	Final
Total logic elements	12,149 / 114,480 (11 %)
Total combinational functions	10,979 / 114,480 (10 %)
Dedicated logic registers	4,672 / 114,480 (4 %)
Total registers	4672
Total pins	231 / 529 (44 %)
Total virtual pins	0
Total memory bits	32,768 / 3,981,312 (< 1 %)
Embedded Multiplier 9-bit elements	0 / 532 (0 %)
Total PLLs	0 / 4 (0 %)

Figura 33. Características finales del sistema en la FPGA 1.

Analysis & Synthesis Resource Utilization by Entity

	Compilation Hierarchy Node	LC Combinationals	LC Registers	Memory Bits
1	|ARBITRADOR:ARBITRADOR|	17 (17)	7 (7)	0
2	|COUNTER:COUNTERS|	8 (8)	4 (4)	0
3	|MASTER:MAESTRO0|	1 (1)	1 (1)	0
4	|MASTER:MAESTRO1|	1 (1)	1 (1)	0
5	|MASTER:MAESTRO2|	1 (1)	1 (1)	0
6	|MEMORIA:MEMORIA|	2 (2)	0 (0)	32768
7	|RV32I:PROCESADOR0|	3584 (24)	1556 (0)	0
8	|RV32I:PROCESADOR1|	3574 (23)	1551 (0)	0
9	|RV32I:PROCESADOR2|	3583 (25)	1551 (0)	0
10	|SELECTORACK:SELECTORDACK|	4 (4)	0 (0)	0
11	|SELECTORDAT:SELECTORDATI|	64 (64)	0 (0)	0
12	|SELECTORDATO:SELECTORDATO|	96 (96)	0 (0)	0
13	|SELECTORDIR:SELECTORDIRI|	30 (30)	0 (0)	0
14	|SELECTORESC:SELECTORESC|	8 (8)	0 (0)	0
15	|SELECTORLEC:SELECTORLEC|	4 (4)	0 (0)	0
16	|SELECTORSTB:SELECTORSTB|	2 (2)	0 (0)	0

Figura 34. Características finales del sistema en la FPGA 2.

En las figuras 33 y 34 se muestran la cantidad de elementos lógicos de cada uno de los bloques de la arquitectura de interconexión WISHBONE, los cuales corresponden a una cantidad mínima 12149, lo que permitiría agregar más procesadores o memorias al sistema.

6. CONCLUSIONES Y RECOMENDACIONES

6.1 CONCLUSIONES

En este trabajo de grado se diseñó una arquitectura de interconexión WISHBONE, esta arquitectura permitió interconectar 3 procesadores con una memoria compartida, cada uno de ellos ejecutando un programa distinto, logrando una arquitectura de interconexión open source.

Se diseñaron cada uno de los bloques que componen la arquitectura de interconexión WISHBONE, y se comprobó que cada uno de estos bloques funcionara correctamente de manera individual y en conjunto con los demás bloques.

La última prueba que se realizó con el analizador de estados lógicos buscaba comprobar que con la arquitectura de interconexión WISHBONE el tiempo de ejecución de diferentes programas es más corto que si un solo procesador realizara todos los programas, es por esto que en el analizador de estados lógicos se verificó cuando tiempo tardaría un solo procesador en hacer los programas que ejecutan los 3 procesadores. En las siguientes figuras se observa el resultado obtenido.

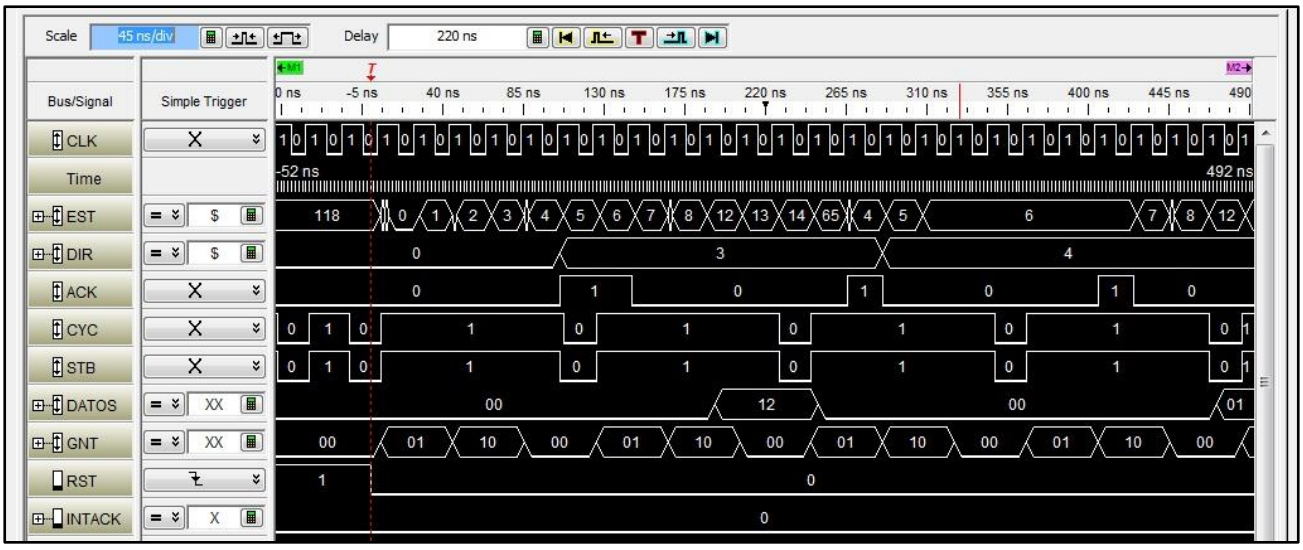

Figura 35. Implementación de un procesador ejecutando todos los programas parte 1.

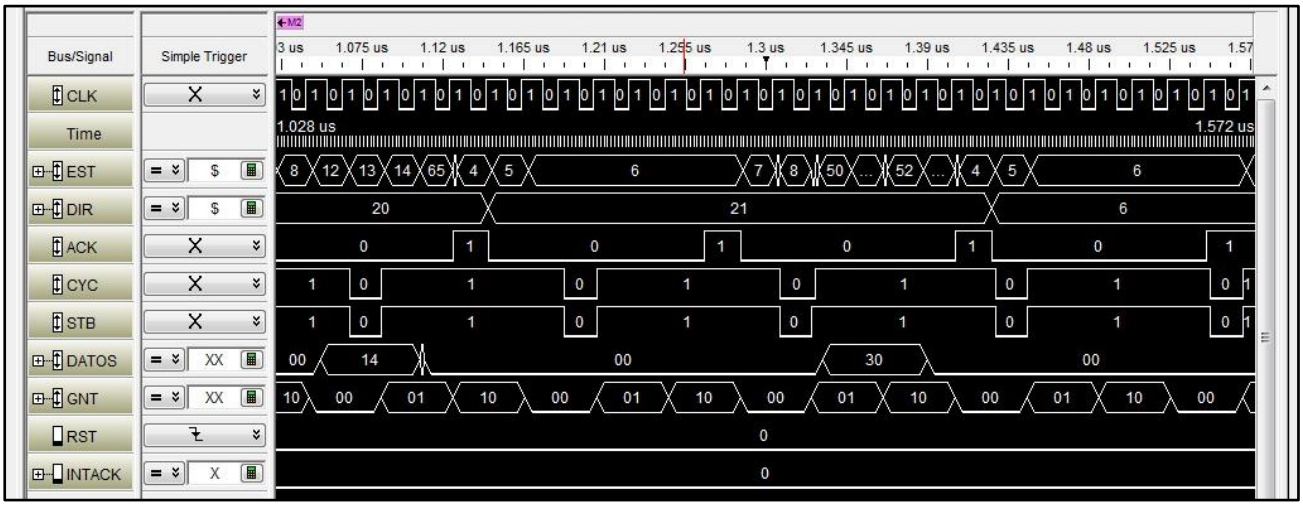

Figura 36. Implementación de un procesador ejecutando todos los programas parte 2.

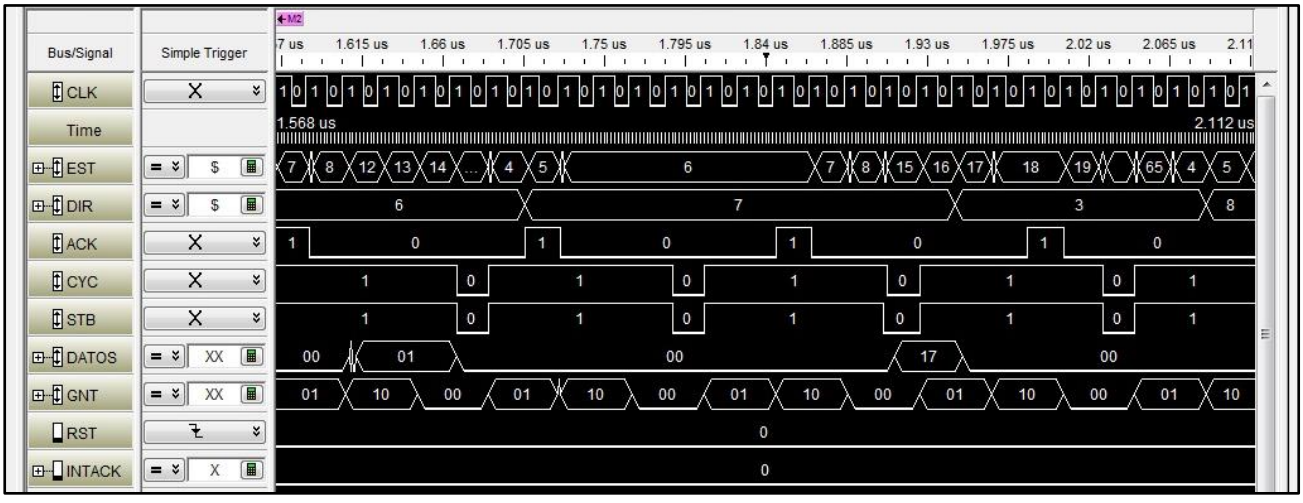

Figura 37. Implementación de un procesador ejecutando todos los programas parte 3.

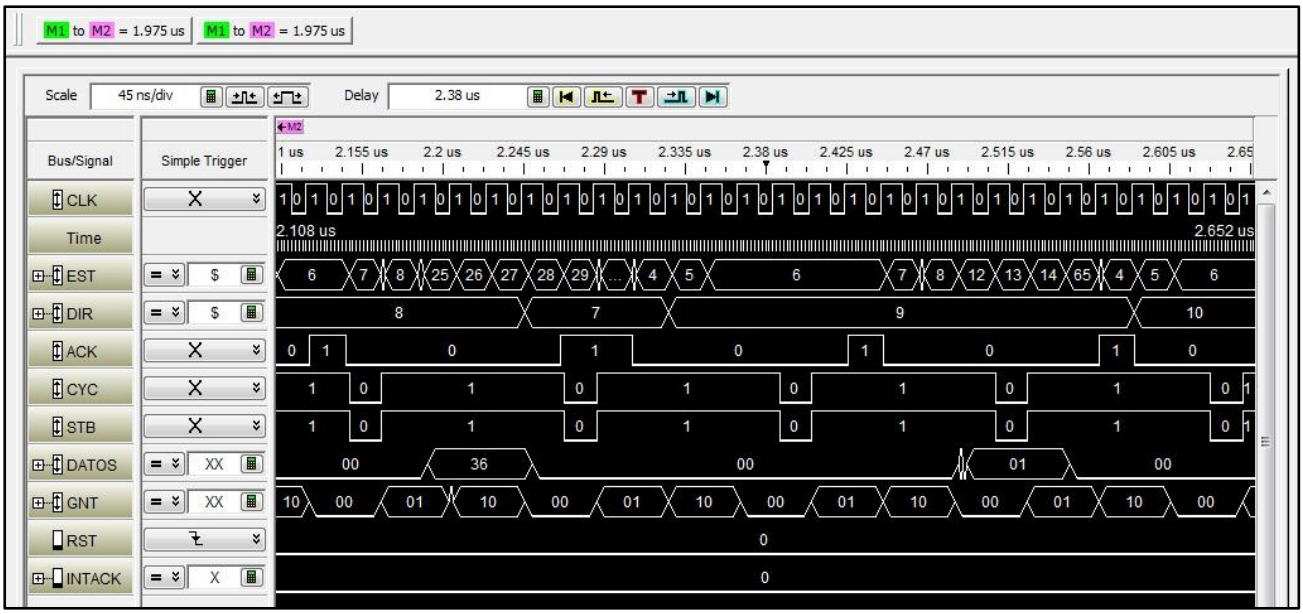

Figura 38. Implementación de un procesador ejecutando todos los programas parte 4.

El resultado de esta prueba arrojo que un solo procesador ejecutando los 3 programas, tarda aproximadamente 2650 ns mientras que con la arquitectura de interconexión WISHBONE, los procesadores tardan entre 740 ns y 980 ns lo que permite concluir que, el diseño de la arquitectura WISHBONE, funciona de manera eficiente y reduce el tiempo de ejecución de los programas a más de la mitad.

Es posible concluir que el diseño y funcionamiento de la arquitectura de interconexión WISHBONE fue satisfactorio, todos los bloques funcionan según lo esperado, y los procesadores ejecutan correctamente los programas que incorporan diferentes tipos de instrucciones del procesador seleccionado. [7]

6.2 TRABAJOS A FUTURO

En un trabajo posterior se podría incluir un número mayor de procesadores y memorias, esto con el fin de ejecutar más programas y verificar que la arquitectura de interconexión WISHBONE no tiene un número definido de procesadores y memorias las cuales puede manejar.

7. BIBLIOGRAFÍA

[1] O. Brenda, "¿Qué es la tecnología open source?," *Stratus media solutions*, 2020.

[2] A. Fran, "¿Qué es el Hardware Libre?," *El Diario*, 2013.

[3] IBM, "CoreConnect." http://www.chips.ibm.com/products/coreconnect/ (accessed Sep. 10, 2020).

[4] ARM, "CUSTOM SYSTEM-ON-CHIP (SOC)." https://www.arm.com/why-arm/custom-socs (accessed Sep. 10, 2020).

[5] Merrian Webster, "Wishbone." https://www.merriam-webster.com/dictionary/wishbone (accessed Nov. 17, 2020).

[6] W. D. Peterson and R. Herveille, "Wishbone B4 specification WISHBONE System-on-Chip (SoC) Interconnection Architecture for Portable IP Cores," p. 128, 2010.

[7] E. Barbosa, "DISEÑO DE UN PROCESADOR CON ARQUITECTURA RISC-V PARA APLICACIONES DE DOMÓTICA.," 2020.

[8] "RISC-V", [Online]. Available: https://riscv.org/

8. ANEXOS

En el siguiente enlace se encuentra el repositorio con todos los documentos anexados

https://livejaverianaedu-my.sharepoint.com/:f:/g/personal/juan_bernal_javeriana_edu_co/EifvYY0PV4NNnVbedbei0GEBIsuy1BlDu7WYKJpiw6G9Jw?e=twcAjW

ANEXO A: AHPL

ANEXO B: ESQUEMÁTICOS

ANEXO C: VHDL (PROYECTO HECHO EN QUARTUS II)