ISSN 1870-4069

Procesador dedicado para el tratamiento de datos en formato de punto flotante

Úrsula Samantha Morales-Rodríguez[1], Gelacio Castillo-Cabrera[2], Herón Molina-Lozano[1]

[1] Instituto Politécnico Nacional, Centro de Investigación en Computación, Ciudad de México, México
[2] Instituto Politécnico Nacional, Escuela Superior de Cómputo, Ciudad de México, México
smoralesr-ipn@hotmail.com, hmolina@cic.ipn.mx, gcastilloc@ipn.mx

Resumen. El desarrollo de hardware en el sector mercantil es considerablemente bajo respecto al desarrollo de software a nivel mundial, cabe mencionar que México no cuenta con el impulso adecuado para implementar nuevas arquitecturas dedicadas a la resolución de problemas específicos en Hardware para el perfeccionamiento de procesos o análisis de datos de una manera óptima. Consecuentemente y en pro de mejorar las estadísticas, el presente artículo muestra el desarrollo y la implementación de un procesador dedicado basado en la arquitectura RISC que permite el procesamiento de datos en formato de punto flotante consumiendo un solo ciclo de reloj por instrucción, ya que cuenta con una Unidad de Punto Flotante (UPF) dedicada que permite la ejecución de operaciones entre números enteros y decimales siguiendo el estándar IEEE-754 para su procesamiento.

Palabras clave: procesador dedicado. IEEE-754. UPF. punto flotante.

Dedicated Processor for Data Processing in Floating Point Format

Abstract. The development of Hardware in the commercial sector is considerably low with respect to the development of Software worldwide, it is worth mentioning that Mexico does not have the appropriate impulse to implement new architectures dedicated to the resolution of specific problems in Hardware for the improvement of processes or analysis of data in an optimal way. Consequently, and in order to improve statistics, this article shows the development and implementation of a dedicated processor based on the RISC architecture that allows data processing in floating point format consuming a single clock cycle per instruction, since it counts with a dedicated Floating Point Unit (UPF) that allows the execution of operations between whole numbers and decimals following the IEEE-754 standard for processing.

Keywords: dedicated processor. IEEE-754. UPF floating point.

1. Introducción

El desarrollo de nuevas tecnologías en el ámbito computacional ha incrementado debido al impulso de dos grandes actividades; el desarrollo de *Hardware* y el desarrollo de *Software*. En el sector industrial, el desarrollo de hardware es considerablemente bajo siendo ca. 19%, en comparación al desarrollo de software que abarca ca. 81% del mercado a nivel mundial [1], dichos porcentajes muestran limitaciones para implementar nuevas tecnologías por la falta de hardware. Estadísticamente, el continente europeo cuenta con un mayor desempeño desarrollando hardware en comparación con el continente americano [2], consecuentemente México carece del impulso adecuado y necesario para implementar nuevas arquitecturas dedicadas a la resolución de problemas específicos en Hardware que ayuden al perfeccionamiento de procesos o análisis de datos de una manera óptima y precisa.

Es necesario que desde el nivel académico se desarrolle hardware para el funcionamiento eficiente de diversas aplicaciones de impacto e interés en el sector industrial, significando que las tareas se realicen de una manera más rápida, con un menor consumo de potencia y/o minimizando el espacio en cuanto a recursos, entendiéndose por hardware a la arquitectura física para el procesamiento de la información, es decir, un procesador [3].

El presente trabajo comprende el desarrollo, desde el diseño hasta la simulación de un procesador dedicado para la ejecución de tareas específicas realizando operaciones en punto flotante y pretende impulsar un amplio panorama de nuevas tecnologías en diseño y desarrollo de hardware, y por consecuencia, abrir camino hacia una ilimitada fuente de ideas para el desarrollo de software.

La unidad lógica aritmética del procesador diseñado realiza operaciones en formato de punto flotante, siendo ésta su principal característica, cuya ventaja es la precisión de los resultados en comparación de la unidad lógica aritmética simple. Actualmente, las operaciones con punto flotante son implementadas en distintas aplicaciones; de telecomunicaciones, control industrial, procesamiento de señales, entre otras. A pesar de la incorporación de las computadoras en los campos mencionados, la falta de sistemas dedicados que permitan la realización de diversas tareas específicas sin desperdicio de recursos y eficiencia en su desempeño prevalece.

En la actualidad existen diversos procesadores dedicados desarrollados a nivel académico; por citar dos de ellos se aprecia un proyecto publicado en 2013 a nivel internacional en el congreso "International Conference on Advanced Electronic Systems" (ICAES) desarrollado en la universidad VIT en India [4] el cual presenta un co-procesador con una Unidad Lógica Aritmética de arquitectura segmentada de punto flotante para minimizar la potencia e incrementar la frecuencia de operación con la intención de mejorar el rendimiento del diseño usando Verilog HDL, sin embargo utiliza más de un ciclo de reloj para poder ejecutar las operaciones básicas, mientras que el segundo se trata de un proyecto académico desarrollado en la Escuela Superior de Computo del Instituto Politécnico Nacional [5] donde se implementa un microprocesador en VHDL de 16 bits de arquitectura RISC tipo MIPS con arquitectura Harvard y una Unidad Lógica aritmética simple con esquema de acarreo anticipado por propagación que no permite el procesamiento de datos en punto flotante, no obstante cada instrucción ejecutada se lleva a cabo en un ciclo de reloj.

El presente artículo muestra el diseño y el desempeño de un procesador dedicado de 32 bits de arquitectura tipo Harvard con una Unidad de Punto Flotante (UPF) diseñada únicamente para permitir el procesamiento de datos en formato punto flotante de precisión simple de acuerdo al estándar IEEE-754 [6]; se realiza el diseño implementando una filosofía orientada a procesadores RISC basados en MIPS, modelando la arquitectura mediante lenguaje de descripción de Hardware VHDL y mostrando como característica principal la ejecución de una instrucción por cada ciclo de reloj, es decir, posee una arquitectura monociclo.

2. Diseño del procesador

Se diseñó un procesador dedicado de 32 bits basado en la arquitectura RISC tipo MIPS, cuyas principales características son: los datos procesados únicamente serán del tipo Punto Flotante, únicamente se realizaran operaciones sobre registros, cuenta con una memoria de programa y una memoria para datos, así como con un banco de registros, solo contiene la Unidad de Punto Flotante (UPF) para realizar operaciones, un Contador de Direcciones, el cual indica el conteo en la dirección al ejecutarse un programa y un Contador de Programa (CP) para indicar direcciones de memoria. La unidad de control es la encargada de controlar las señales destinadas al buen funcionamiento de los componentes de acuerdo al tipo de instrucción implementada. La Fig. 1 Muestra el diagrama a bloques del procesador dedicado, donde se puede apreciar la arquitectura y la interconexión de cada componente.

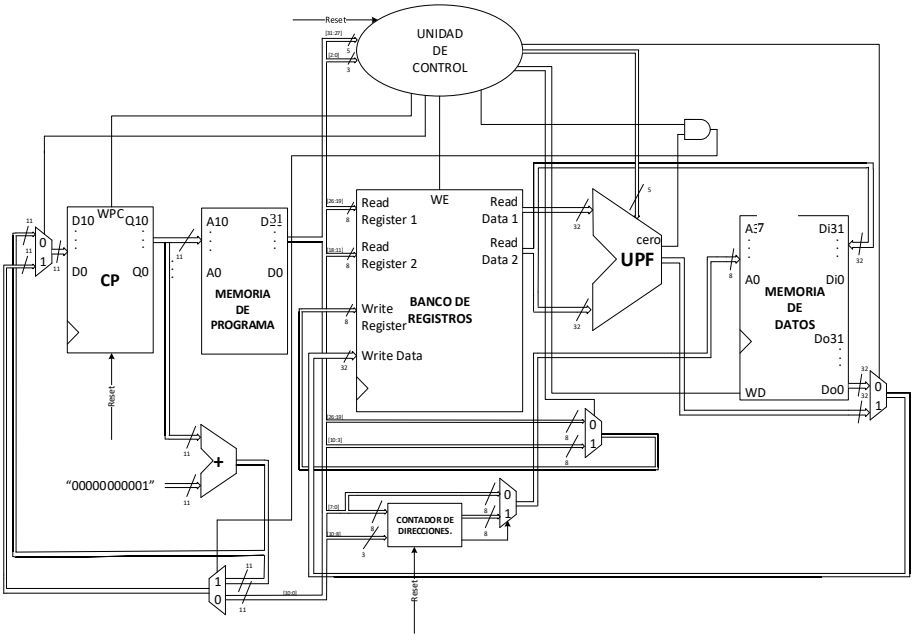

Fig. 1. Diagrama general a bloques del procesador dedicado de 32 bits en punto flotante.

2.1. Formato de instrucciones

Para la definición del formato de instrucciones, se considera la representación de las instrucciones del procesador formadas por una cadena de 32 bits cada una y el procesamiento del tipo de dato en formato de punto flotante según el estándar IEEE-754 de precisión sencilla, estableciendo campos para cada tipo de instrucción de acuerdo con la Tabla 1.

Las instrucciones tipo R son instrucciones de tipo aritmético-lógicas; mediante el código de operación (op) indicado realizan una operación en la UPF sobre los contenidos en los registros fuentes 1 y 2 (rs, rt) escribiendo el resultado sobre el registro final (rd) apoyándose del campo resultante (cr) para la definición del tipo de operación. Las instrucciones tipo Jump y Branch son instrucciones de salto; de tipo incondicional, con 5 bits para el código de operación y los 11 bits menos significativos son para el campo de dirección (dir), o de tipo condicional, donde se debe especificar dos campos en los operandos (rs, rt) para realizar la diferencia, además de la dirección de salto, respectivamente. Las instrucciones de transferencia de datos pueden ser de carga o almacenamiento; el formato de la instrucción carga/almacenamiento cuenta con su respectivo código de operación (op), un registro fuente o destino (rd, rs) y la dirección de memoria (dir).

Tabla 1. Descripción de los tipos de formato de instrucción con sus respectivas definiciones.

Tipo de instrucción		**Formato de instrucción**				
Sobre Registro		**op**	**rs**	**rt**	**rd**	**cr**
		5 bits	8 bits	8 bits	8 bits	3 bits
Salto	Condicional	**op**	**rs**	**rt**	**dir**	
		5 bits	8 bits	8 bits	11 bits	
	Incondicional	**op**	**s/u**	**s/u**	**dir**	
		5 bits	8 bits	8 bits	11 bits	
Transf. de datos	Carga	**op**	**rd**	**s/u**	**dir**	
		5 bits	8 bits	8 bits	11 bits	
	Almacenamiento	**op**	**s/u**	**rs**	**dirección**	
		5 bits	8 bits	8 bits	11 its	

Definiciones	
op	Código de operación
rs	Registro 1/ almacenamiento
rt	Registro 2
rd	Registro final / carga
cr	Campo resultante
s/u	Sin uso
dir	Dirección de memoria

2.2. Set de instrucciones

El procesador dedicado maneja un total de 9 operaciones en el set de instrucciones de tipo salto, aritmético-lógicas y transferencia de datos; se pueden consultar en la Tabla 2. Cabe mencionar que cada instrucción consume un ciclo de reloj en el procesador y todos los operandos utilizarán su respectiva representación en formato punto flotante para ser procesados de acuerdo al estándar IEEE-754.

Tabla 2. Set de instrucciones del procesador dedicado.

Tipo	**Codop**	**Instrucción**	**Nemónico**	**Ejemplo**	**Significado**
Store	00000	Carga	LW	LW rd, dir	rd = Mem [dir]
Store	10000	Almacenamiento	SW	SW rs, dir	Mem[dir] = rs
R	00001	Suma	ADD	add rd,rs,rt	rd = rs + rt
R	00010	Resta	SUB	sub rd,rs,rt	rd = rs - rt
R	00011	Multiplicación	MULT	Mult rd,rs,rt	rd = rs * rt
R	00100	División	DIV	div rd,rs,rt	rd = rs / rt
Branch	10010	Salto si es diferente	BNEQ	BNEQ dir	If(rs!=rt) go to Dir
Jump	10001	Salto	B	B dir	PC = Dir
	11111	NOP	NOP	NOP	--

2.3. Formato del tipo de dato

De acuerdo con la intención que el estándar IEEE-754 refiere, donde la implementación de cualquier sistema en punto flotante puede ser completamente desarrollada a nivel software, a nivel hardware o cualquier combinación entre estos dos niveles, el formato del tipo de dato a ser procesado es el formato de precisión simple que sigue el estándar, donde la representación de un número en coma flotante contempla una parte entera y una parte decimal. Siguiendo el estándar, una vez aplicada la conversión de dicha representación al formato de precisión simple [6], el cual posee un bus de 32 bits dividido en tres campos reservados, el primer campo cuenta con el bit más significativo (bit31) indicando el signo del número a ser representado, un segundo campo que reserva 8 bits para representar el exponente (bit30 … bit23) y un último campo con 23 bits para representar la mantisa (bit22 … bit0), el formato final puede ser apreciado en la Fig. 2.

$$(1.853515625 \times 2^{6}) \times (-1) = -118.625$$

Bit de Signo (S)	Exponente (E)	Mantisa (M)
1	**1000 0101**	**11011010100000000000000**

Fig. 2 Representación científica del número -118.625 y su respectiva representación en formato de precisión simple de punto flotante según el estándar IEEE-754 [7].

2.4. Componentes del procesador

El procesador cuenta con siete componentes fundamentales para su funcionamiento: el contador de programa, contador de direcciones, la memoria de programa, el banco de registros, la memoria de datos, la unidad de control y la unidad de punto flotante (UPF).

Contador de programa: registro que contiene la dirección de la consecuente instrucción a ejecutar obtenida de la memoria de programa, dicho registro será incrementado automáticamente por cada ciclo de instrucción.

Contador de direcciones: permite el incremento de la dirección contigua a ser leída por el contador de programa según se trate de la instrucción a ser ejecutada de acuerdo al formato de instrucciones.

Memoria de programa: memoria de solo lectura; las instrucciones del programa a procesar serán almacenadas en esta memoria. Su capacidad de memoria es de 2K x 4B, la instrucción leída de la memoria es dirigida hacia la unidad de control, el banco de registros y el contador de direcciones; cuenta con un bus de direcciones y un bus de datos.

Banco de registros: conformado por 256 registros de 32 bits cada uno; cuenta con dos puertos de lectura y un solo puerto de escritura, los puertos de lectura con 8 bits a la entrada, Read Register1 y Read Register2, haciendo posible direccionar hasta 256 localidades del respectivo banco de registros, cabe mencionar que siguiendo la implementación de una arquitectura tipo MIPS donde el banco de registros direcciona hasta 32 localidades, el presente diseño considera la implementación del total de localidades disponibles a partir de la capacidad de direccionamiento disponible sin descartar la implementación de una menor cantidad de registros basándonos en la arquitectura MIPS donde únicamente se podrían considerar los 4 bits menos

significativos de los buses de entrada en los puerto de lectura; dichos puertos son leídos a través de las salidas Read Data1 y Read Data2, respectivamente. El puerto de escritura toma 8 bits de dirección a la entrada de Write Register, así como 32 bits de datos para la entrada de Write Data. Posee un bit de control denominado WE, el cual, si se encuentra posicionado en alto, el banco de registros es capaz de escribir los datos recibidos (Write Data) en el registro especificado (Write Register), en caso contrario es de solo lectura sobre los registros fuente.

Memoria de datos: una memoria de lectura/escritura, su capacidad de memoria es de 256 x 32 bits; debido a su ancho de palabra permite almacenar datos en formato de punto flotante con precisión simple. Maneja dos buses de datos de 32 bits para entrada y salida, así como un bus de direccionamiento de 8 bits; considera una menor capacidad de almacenamiento en comparación con la memoria de programa, ya que el diseño del presente procesador prospecta el tratado de una gran cantidad de instrucciones debido a la naturaleza de su arquitectura.

Unidad de control: Para implementar la unidad de control como una máquina de estados finita, primero se asignó un número a cada uno de los 9 estados, en este caso, se usó un enumerado simple secuencial; el diagrama de la máquina de estados finita implementada se aprecia en la Fig. 3. conteniendo nueve estados con su debida representación en binario (4 bits) para codificar el número de estado representado.

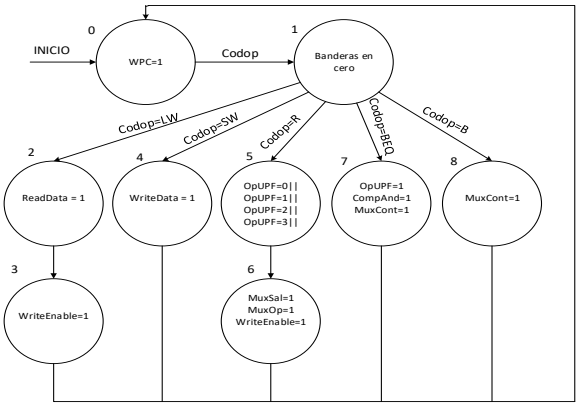

Fig. 3 Diagrama de la máquina de estados finita para la unidad de control.

La **Tabla 3** muestra de manera general las banderas de activación utilizadas por la unidad de control por cada camino de datos respectivo de cada instrucción implementada en el procesador.

Tabla 3. Bits de entradas y salidas manejados por la unidad de control.

Control	Nombre de la señal	Carga	Almacenamiento	Tipo R				Beq	B
				ADD	SUB	MULT	DIV		
Entradas	Codop(4)	0	1	0	0	0	0	1	1
	Codop(3)	0	0	0	0	0	0	0	0
	Codop(2)	0	0	0	0	0	1	0	0
	Codop(1)	0	0	0	1	1	0	1	0
	Codop(0)	0	0	1	0	1	0	0	1
Salidas	MuxCont	0	0	0	0	0	0	1	0
	WriteEnable	1	0	1	1	1	1	0	0

Control	Nombre de la señal	Carga	Almacenamiento	Tipo R				Beq	B
				ADD	SUB	MULT	DIV		
	MuxOp	1	0	1	1	1	1	0	0
	ReadData	0	0	0	0	0	0	0	0
	WriteData	1	1	0	0	0	0	0	0
	MuxSal	0	0	1	1	1	1	0	0
	CompAnd	0	0	0	0	0	0	1	0
	MuxBranch	0	0	0	0	0	0	1	1
	WPC	1	1	1	1	1	1	1	1
	OpUPF(2)	0	0	0	0	0	0	0	0
	OpUPF(1)	0	0	0	0	1	1	0	0
	OpUPF(0)	0	0	0	1	0	1	0	0

Unidad de Punto Flotante [7]: se implementó una unidad de punto flotante que permite el procesamiento únicamente de números con el formato de punto flotante de precisión simple del estándar IEEE-754. Se trata de una unidad dedicada que realiza el procesamiento de los operandos en un solo ciclo de reloj, sacrificando espacio debido a una mayor densidad en cuanto a recursos de hardware, pero considerando una mejora en el tiempo de respuesta, la arquitectura se observa en la **Fig.4**.

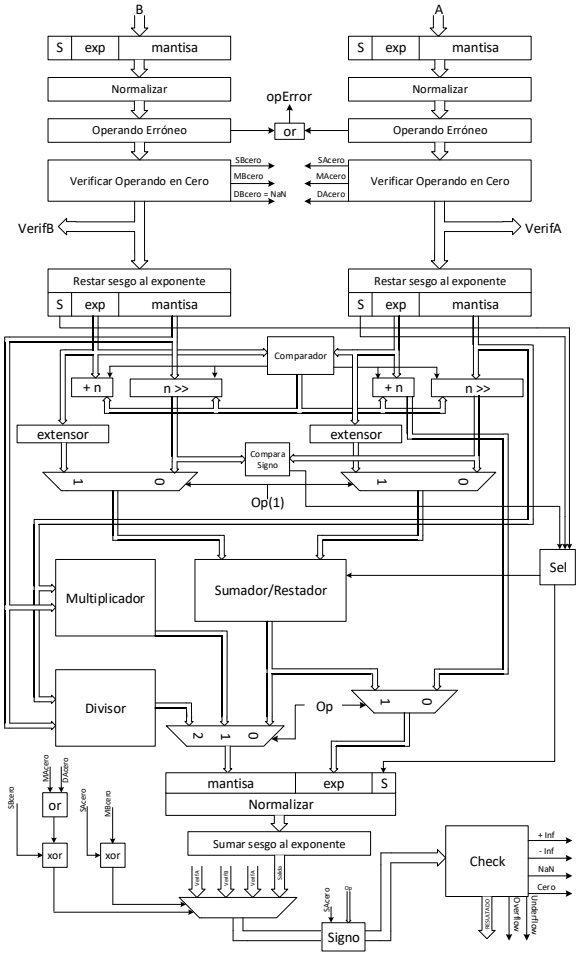

Fig. 4. Diagrama general de la unidad de punto flotante para el procesamiento de datos en formato de precisión simple de acuerdo al estándar IEEE-754.

3. Campo de aplicación

Para visualizar la operatividad del procesador, se incluyó en la memoria de programa una serie de instrucciones que permiten conocer el resultado de procesar parte del funcionamiento de la primera iteración en la etapa de entrenamiento de la red neuronal ADALINE.

Una neurona ADALINE es una neurona simple de una unidad, recibe entradas de varias unidades y una entrada de una "unidad" la cual siempre tiene el valor de +1 con el objetivo de entrenar el peso del sesgo por el mismo proceso mediante la regla delta, usado para poder actualizar los pesos de cada entrada ponderada [8].

El algoritmo de entrenamiento de una neurona ADALINE es el siguiente:

- Paso 0. Inicializar los pesos
- Paso 1. Mientras la condición de paro sea falsa, realiza los pasos 2-6.
- Paso 2. Colocar la activación de las unidades de entrada
- Paso 3. Calcular la entrada de la red hacia la salida de la unidad dada por:

$$y_{in} = b + \sum_{1}^{i} x_i w_i.$$

- Paso 4. Actualiza sesgo y pesos
- Paso 5. Verificar la condición de paro.

En términos generales, el procesador se encarga de ponderar los pesos por los valores de las señales, realizar la suma acumulatoria y finalmente agregar el sesgo de la neurona, es decir, realiza el paso 3 del algoritmo de entrenamiento de la red neuronal con los valores que se muestran en la **Fig. 5.**

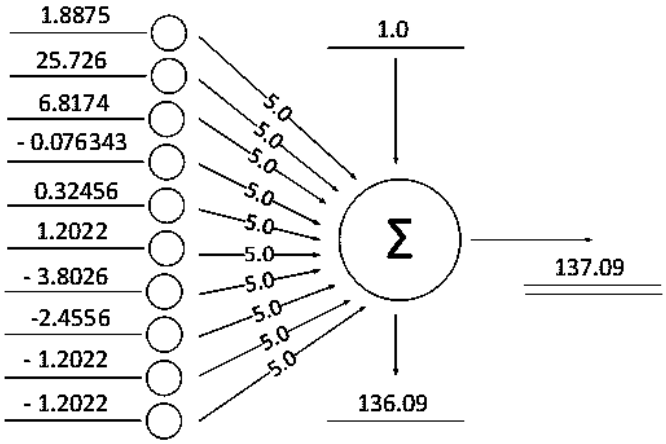

Fig. 5. Neurona ADALINE con valores numéricos.

El listado de instrucciones incluido en la memoria de programa del procesador es el mostrado en la Fig. 6.

```
    LW R0, x"00"          ; R0= MEM[0] – Carga del número de muestras al registro 0
    LW R1, x"01"          ; R1= MEM[1] – Carga el número uno al registro 1
    LW R2, x"02"          ; R2= MEM[2] – Carga el número cero al registro 2
    LW R6, x"02"          ; R6= MEM[2] – Carga el número cero al registro 6

CICLO:
        LW R4, x"0D"              ; R4= MEM[13] – Carga el valor del peso
        LW R5, x"03"              ; R5= MEM[3]  -- Carga el valor de la señal
        MULT R5, R4, R5           ; R5 = R4 * R5 – Multiplica el peso por la señal
        ADD R6, R5, R6            ; R6 = R5 + R6 – Realiza la suma acumulatoria.
        SUB R0, R1, R0            ; R0 = R1 – R0 -- Resta el número de muestras menos uno
        BNEQ R0, R2, x"04"                ; if (R2!=R0) go to CICLO – Si el número de iteraciones
                                          es distinto al número de muestras, salta a la etiqueta CICLO,
                                          en caso contrario, continúa.

    ADD R6, R1, R6        ; R6 = R1 + R6 – Suma el sesgo a la suma acumulatoria.
    SW R6, x"FF"  ; MEM[255] = R6 – Almacena en memoria el valor del registro 6.
    LW R7, x"FF"  ; R7 = MEM[255] – Carga el resultado final sobre el registro 7.
    B x"0C"               ; PC = 12 – Brinca a la dirección 12
```

Fig. 6 Programa detallado embebido en la memoria de programa para realizar el procesamiento del paso 3 del algoritmo de entrenamiento de la neurona ADALINE.

4. Resultados

Los resultados obtenidos en la simulación del funcionamiento del procesador con el programa ya embebido en la memoria utilizando el software ACTIVE HDL fueron los mismos que los resultados esperados.

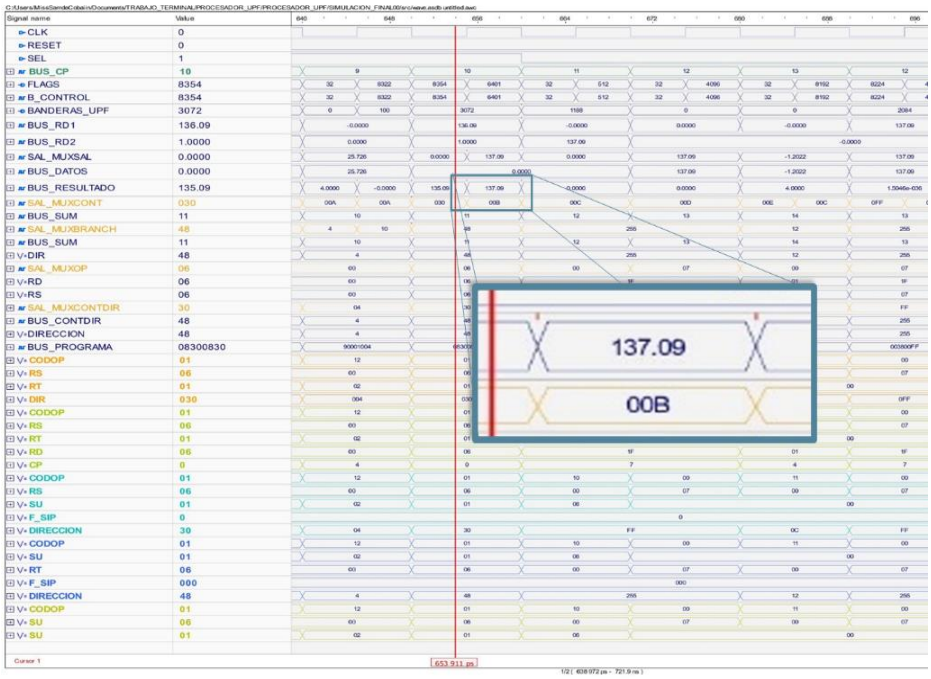

Fig. 7 Simulación del resultado final empleando el software ACTIVE HDL.

5. Conclusiones

Los resultados obtenidos en la simulación fueron los mismos resultados que los esperados, cabe mencionar que siguiendo el estándar IEEE-754 para tratar los números en formato de punto flotante, fue posible implementar en hardware una arquitectura dedicada capaz de procesar cualquier valor numérico existente en el formato presentado; estrictamente, el procesador únicamente maneja datos en punto flotante debido a que a los números enteros también los opera como datos en punto flotante.

El diseño del procesador dedicado de punto flotante fue basado en una arquitectura tipo MIPS, permitiendo ejecutar cada instrucción del mismo set de instrucciones en un solo ciclo de reloj procurando mejorar los tiempos de respuesta en comparación con otras arquitecturas, si bien, en ocasiones resulta complicado realizar comparativas entre arquitecturas dedicadas que sean capaces de procesar un mismo programa, no se descarta la elaboración de dicha comparativa como trabajo a futuro con la finalidad de evaluar el desempeño del presente procesador.

Como bien se sabe, la filosofía de la arquitectura de computadoras nos indica una proporcionalidad en relación densidad-tiempo, de modo que, al reducir el tiempo de procesamiento, se incrementa la densidad de nuestro diseño y ello se ve reflejado en la Unidad de Punto Flotante.

Hoy en día existen diversas arquitecturas dedicadas al procesamiento de datos y al análisis de señales, pero el procesador presentado es un procesador dedicado para diversas aplicaciones específicas, ya que podría ser implementado una en aparatos de tamaño mínimo que permitan la cancelación de ruido, el promedio de diversos valores para el diagnóstico y/o control de la hipertensión o la diabetes, la selección de una señal con mayor intensidad, entre otras.

Referencias

1. AESOFT, Asociación Ecuatoriana de Software: Estudio de mercado en el sector software y hardware en Ecuador. http://www.aesoft.com.ec/ (septiembre, 2011)
2. AESOFT, Asociación Ecuatoriana de Software: Estudio de mercado en el sector software y hardware en Ecuador http://www.aesoft.com.ec/ (septiembre, 2011)
3. Grediaga Olivo, A., Pérez Martínez, J.: Procesadores. Diseño de procesadores en VHDL. Publicaciones Universidad de Alicante Ed., Primera ed. España: Alicante. pp. 225 (2007)
4. Manisha S., Anita, A.: Design and Implementation of Single Precision Pipelined Floating Point Co-Processor. In: International Conference on Advanced Electronic Systems (ICAES), Nwe Delhi (2013)
5. García Ortega, V.H. et al.: Microprocesador didáctico de arquitectura RISC implementado en un FPGA. Universidad de Guadalajara, e-Gnosis, vol. Esp., pp 1–8 (2009)
6. Cornea: IEEE 754-2008 Decimal Floating-Point for Intel® Architecture Processors, In: 19th IEEE International Symposium on Computer Arithmetic (2009)
7. Morales Rodríguez, U.S.: Procesador de punto flotante para análisis de señales. Tesis (Ingeniería en Sistemas Computacionales). Ciudad de México, Instituto Politécnico Nacional, Escuela Superior de Cómputo, p. 202 (2016)
8. Fauset, L.: Fundamentals on Neural Networks, architectures, algorithms and applications. United States of America: Prentice Hall, Englewood Cliffs (1994)