# IMPLEMENTACIÓN DE UNA ARQUITECTURA RISC CON PERIFÉRICOS

A. Vargas Vélez, P. Mejía Romo, D. Uribe Hernández, L. Ortiz Camacho, L. Rodríguez Palacios, I. Ramírez Jiménez, J. Hernández Reyes, J. Colín Ramiro, E. Mendoza García
Departamento de ingeniería en sistemas computacionales, ESCOM IPN
avargasv1601@alumno.ipn.mx, pmejiar1900@alumno.ipn.mx, curibeh1900@alumno.ipn.mx , lortizc1901@alumno.ipn.mx , orodriguezp1802@alumno.ipn.mx, itzelraji12@gmail.com, jhernandezr1923@alumno.ipn.mx, joelcolinr@gmail.com, emendozag1602@alumno.ipn.mx

***Resumen*— La arquitectura RISC, es una de las más conocidas hoy en día en el mundo de las ciencias de la computación, más comúnmente en la rama de la arquitectura de computadoras. En esta práctica desarrollamos un sistema que pudiese resolver ecuaciones lineales en un FPGA, esto como el nombre de la práctica lo menciona se desarrolló basándonos en una arquitectura de tipo RISC. Para esto se implementó el código en VHDL y para obtener una salida más gráfica se utilizó la tarjeta gráfica ALTERA DE-215, está por supuesto proporcionada por el docente.**

***Palabras Clave* —arquitectura, display, ecuación, incógnita**

***Abstract*— The RISC architecture is one of the best known nowadays in the world of computer science most commonly in the branch of computer architecture. In this very work we developed a computer system that could solve linear equations in an FPGA, this, as the name of the practice itself mentions, was developed based on a RISC-type architecture. For this, the code was implemented in VHDL and to obtain a more graphic output the ALTERA DE-215 graphic card was implemented this last one of course provided by the teacher to make the tests and to compare the results**

***Keywords* —architecture, display, equation, unknown**

## I. INTRODUCCIÓN

Como bien lo estudiamos y analizamos en la práctica, una arquitectura de tipo RISC, puede llegar a ser robusta y muy común de ver en las nuevas computadoras. Este mercado puede extenderse inclusive a hablar de procesadores para teléfonos celulares y tablets, esto debido a su bajo consumo de recursos y su buen rendimiento

Entre los productores de arquitecturas RISC, representados en gran número en el mercado, existe una fuerte lucha por la supervivencia entre cada uno de ellos.

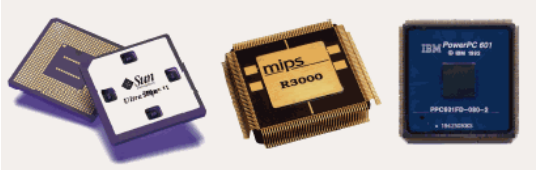



*Figura 1. Procesadores con Arquitectura RISC*

No obstante, no es la calidad de las características técnicas, sino la cantidad de programas de aplicación disponibles lo que decide sobre la permanencia de las diversas arquitecturas existentes. Los esfuerzos de mercadotecnia de los productores de arquitecturas RISC se dirigen hacia la obtención de una cantidad lo mayor posible de aplicaciones. Asi entonces, para casi todas las arquitecturas RISC se pueden encontrar foros o asociaciones de usuarios (fomentadas por los fabricantes) en las que se agrupan los usuarios, los productores de software de sistemas comercial, etc., dando lugar a un amplio espectro de organizaciones.

En esta ocasión y para fines del curso de Arquitectura de Computadoras, se realizó la implementación de una Arquitectura de tipo RISC, la cual en puntos posteriores se explicarán las instrucciones de la actividad, asi como la metodología implementada para la realización de la misma.
Asi mismo se adjuntan las fotografías que reportan el éxito del programa junto con el código del mismo, para asi demostrar el trabajo realizado por todos los integrantes del equipo.

## II. METODOLOGÍA Y DESARROLLO

Para la realización de esta práctica nos basamos en la práctica anterior, la cuál se trataba de una aplicación con una arquitectura de tipo RISC.
En primera instancia se declararon las librerías necesarias para el programa tal y como se observa en la Figura 2, estas se describen a continuación:

- **ieee.std_logic_1164.-** Define funciones para operadores lógicos.
- **ieee.numeric.std .-** Proporciona funciones aritméticas para vectores.
- **ieee.std_logic_arith .-** Especifica tipos de datos con y sin signo, operaciones aritméticas y de comparación numérica
- **ieee.std_logic_unsigned.-** Permite operaciones sin signo.

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3  use work.ctrl.all;
4  use ieee.numeric_std.all;
5  use ieee.std_logic_arith.all;
6  use ieee.std_logic_unsigned.all;
```

*Figura 2. Librerías Utilizadas*

Posteriormente, se declaró la entidad, donde declaramos, entradas y salidas a implementar en el programa.

```
entity SYNC2 is
  port (
     CLK: in std_logic;                          -- Reloj
     U: IN integer;                              -- Primera variable
     V: IN integer;                              -- Segunda variable
     W: IN integer;                              -- Tercera variable
     CTRL: IN STD_LOGIC_VECTOR;                  -- Variable de control
     -- DATO: IN STD_LOGIC;
     HSYNC, VSYNC: out std_logic;                -- Sincronización
     R, G, B: out std_logic_vector(7 downto 0)   -- RGB
  ) ;
end SYNC2;
```

*Figura 3. Declaración de la Entidad*

Así entonces, se pasó a la declaración de la arquitectura del programa, como primer paso en esta parte, se declararon las señales que utilizamos a lo largo de la codificación (Figura 4).

```
architecture MAIN of SYNC2 is

    signal HPOS: integer Range 0 to 1688 := 0;
    signal VPOS: integer Range 0 to 1688 := 0;
    signal U1: integer Range -320 to 320  := 0; -- Variable auxiliar para el eje X1
    signal V1: integer Range -256 to 256  := 0; -- Variable auxiliar para el eje Y1

    signal data_frame: std_logic_vector(21 downto 0);
```

*Figura 4. Señales implementadas en el programa*

Dentro de la misma arquitectura se declaró un solo process en el cual se llevará a cabo toda la lógica funcional de la práctica. En dicho process, primeramente, se declaró una estructura if, la cual su única función es dibujar la gráfica, posteriormente se tiene el código para adaptar la línea de color, la cuál será capaz de rotar, según el cambio en los switches.

```
begin

    process(CLK)

        begin

            if (CLK'Event and CLK = '1') then

                if (HPOS  = 1048 OR VPOS = 554) then

                   R<=X"FF";
                   G<=X"FF";
                   B<=X"FF";

                else
                   R<=X"00";
                   G<=X"00";
                   B<=X"00";
                end if;
```

*Figura 5. Inicio del process*

Más adelante se declaró otra estructura if, la cual se encarga de dibujar la línea con la que se trabajará, esta es la que estará rotando según se presione el botón de la tarjeta Altera, esta proporcionada por el docente.

```
                --Dibuja la linea

            IF CTRL = "00" THEN
                    U1 <= U;
                    V1 <= V;
                END IF;

                if ( (554 - VPOS) = (U1)*( HPOS -1040 ) + (V1)*8) THEN

                   IF CTRL = "00" THEN
                      R<=X"33";
                      G<=X"FF";
                      B<=X"F6";
                   ELSE
                      R<=X"96";
                      G<=X"96";
                      B<=X"96";
                   END IF;

                end if;
```

*Figura 6. Estructura if para dibujar la línea que rota*

Finalmente, la parte restante de la arquitectura es para tener un control de la posición actual en todo momento, tanto horizontalmente como verticalmente. Teniendo un contador que si llega al límite del tamaño de la pantalla se reinician los valores de las posiciones para empezar una nueva iteración. Esto para poder trabajar con la línea de color y poder controlarla desde la tarjeta Altera.

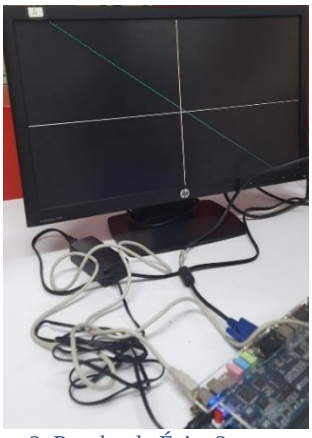
```
            if (HPOS < 1688) then
                HPOS <= HPOS + 1;
            else
                HPOS <= 0;                  -- Si HPOS llega al final de la line
                if(VPOS < 1066) then
                   VPOS <= VPOS + 1;


                else
                    VPOS <= 0;
                end if;
             end if;


             if(HPOS > 48 and HPOS < 160) then
                HSYNC <= '0';
             else
                HSYNC <= '1';
             end if;

             if(VPOS > 0 and VPOS < 4) then
                VSYNC <= '0';
             else
                VSYNC <= '1';
             end if;

          end if;
    end process;

end MAIN ; -- MAIN
```

*Figura 7. Control de posición Horizontal y Vertical*

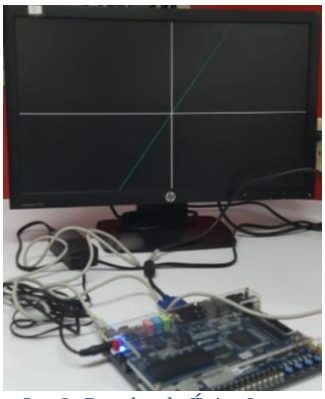
## III. RESULTADOS Y DISCUSIÓN

En esta sección se presentan las fotografías y evidencias del correcto funcionamiento del programa. Este elaborado por el equipo en conjunto.

*Img1. Prueba de Éxito 1*

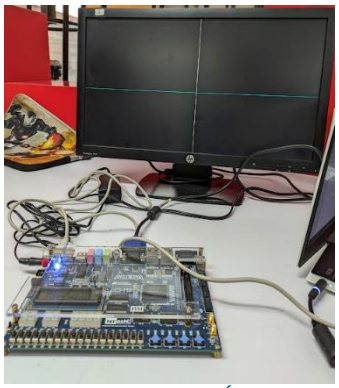


*Img2. Prueba de Éxito 2*



*Img3. Prueba de Éxito 3*

*Img4. Prueba de Éxito 4*



*Img5. Prueba de Éxito 5*

Como bien se puede observar en las imágenes, conforme se iba presionando el switch de la tarjeta Altera, la línea de color, rotaba en la pantalla. Lo cual comprueba lo solicitado en la práctica como exitoso.

## IV. CONCLUSIONES

Este trabajo nos permitió aplicar lo aprendido durante el curso como también aprender nuevos contenidos. Se implementó el procesador basado en la arquitectura RISC, su correcto funcionamiento se presentó con el analizador lógico.

Tal y como se pudo observar en las evidencias, el resultado obtenido es el esperado, sin embargo, su desarrollo represento cierto grado de dificultad para el equipo, ya que algunos medios dentro de la programación en VHDL eran desconocidos, por lo que se requirió de un arduo trabajo de investigación para resolver los errores que surgían durante la programación.

Para un correcto funcionamiento se requirió comprender bien como se debía implementar y asignar correctamente las salidas en la FPGA ALTERA DE2-115, debido que el asignar las salidas es un poco complicado si no se entiende como es que funciona la tarjeta por otro lado al tener que mostrarlo en un monitor se estudió los tiempos de sincronismo que necesita el monitor VGA para poder ver una imagen. Con este sistema se pudo enviar una imagen a color cada que las posiciones de las ecuaciones cambiaban.

Como equipo podemos concluir que fue una práctica un poco más compleja de comprender y de implementar que las anteriores.

## AGRADECIMIENTOS

Los autores de este trabajo agradecen a la Escuela Superior de Cómputo por el material de apoyo brindado para la realización del mismo.

## REFERENCIAS

[1] Luna, J. I. V., González, R. S., Guzmán, G. S., & González, L. A. S. Arquitectura RISC vs CISC.

[2] ROBLETO, I. M. Arquitecturas CISC y RISC.

[3] Martínez Belot, L. J. J., & Leyes, D. A. (2007). Descripción VHDL de una arquitectura RISC (Doctoral dissertation, Universidad Nacional de La Plata).

[4] Cifuentes, Á. P., Dimaté, L. E., Rincón, A. M., Velásquez, J. R., Villegas, M. P., & Flores, P. (2012). Ecuaciones lineales con una incógnita.

[7] R. Sóle, «RISC: La arquitectura de procesadores usada por ARM para cambiar el mercado,» Profesional Review, 17 07 2021. [En línea]. Available: https://www.profesionalreview.com/2021/07/17/que-es-risc/#Que_es_RISC. [Último acceso: 05 05 2022].