

Proyecto	3	Página	1/25
Trabajo	Control y programación RTC con Nexys 4	Actualizado en:	30/09/2016
Grupo	8	Revisado en:	30/09/2016
Revisado por:	Alfonso Chacón Rodríguez	Diseñadores	Luis Leon Vega Luis Merayo Gatica Keylor Mena Venegas

1. Resumen

En este proyecto se diseña un control para el RTC (Real-Time Controller) mediante lógica combinatorial y secuencial. Además, se implementa este diseño en una placa FPGA Nexys 4, para lo cual, se desarrolla la lógica en el lenguaje de descripción de hardware (HDL) denominado Verilog.

Asimismo, se desarrolla un circuito de despliegue de datos utilizando un monitor comunicado mediante el protocolo VGA (Video Array Graphics), el cual recibirá los datos de una memoria que contiene los parámetros del RTC y los colocará en una interfaz desarrollada en Inkscape y trasladada a la Nexys 4 mediante una ROM.

2. Introducción

En la mayoría de circuitos digitales, se emplean varios módulos que permiten la integración de un sistema completo. En algunos, se aprovecha el procesamiento interno y, en otros, se colocan otros módulos para liberar el uso del procesador. En este caso, para manejar el reloj y el cronómetro se ha desarrollado un módulo llamado RTC [AGREGAR REFERENCIA]. Este facilita aislar el procesamiento del reloj y volverlo independiente mediante su propio sistema de control.

Para efectos del proyecto, se empleará el circuito integrado V3023 que contiene la funcionalidad del RTC e independiza el control de los tiempos de cualquier circuito de procesamiento moderno. Para adentrar del funcionamiento de este IC (Circuito integrado por sus siglas en inglés), este cuenta con un bus de direcciones y datos multiplexado, es decir, cuenta con un solo bus que cumple la función de direccionamiento y de datos mediante tres entradas de control, que en este caso, se denominan *AD*, *RD* y *WR*. Estas permiten cambiar el modo del bus, la lectura y la escritura dentro del mismo. Asimismo, tiene una salida de interrupción que permite llamar alguna subrutina respecto a la finalización del timer incorporado en el RTC o, bien, la activación de la alarma [AGREGAR REFERENCIA].

Por otro lado, para el desarrollo de una interfaz de usuario amigable es posible implementar imágenes en una ROM mediante un algoritmo de procesamiento de imágenes que las convierte en una matriz tridimensional, que ubica posición horizontal, posición vertical y profundidad de colores, subdivididos en grupos R, G y B, siendo RGB los colores básicos para formar imágenes con luminiscencia.

Para escribir dentro de una interfaz de este tipo, se debe cargar la tipografía en otra ROM y emplear punteros de memoria según la posición del cursor de pantalla (Posición en X y en Y). Esto permite un desarrollo de la interfaz ágil y rápido.

3. Objetivos

- Aprender el lenguaje ensamblador, utilizado para programar el picoblaze.
- Controlar el circuito integrado V3023, usado para generar el RTC, mediante el uso del Picoblaze.
- Utilizar la programación desarrollada en el proyecto anterior como driver para el control del RTC.

Proyecto	3	Página	2/25
Trabajo	Control y programación RTC con Nexys 4	Actualizado en:	30/09/2016
Grupo	8	Revisado en:	30/09/2016
Revisado por:	Alfonso Chacón Rodríguez	Diseñadores	Luis Leon Vega Luis Merayo Gatica Keylor Mena Venegas

- Diseñar controlador para la pantalla VGA, utilizando imágenes.
- Manejar un teclado mediante el protocolo PS2.
- Facilitar al usuario la manipulación de los datos del reloj mediante la implementación de un teclado.
- Generar una interfaz en una pantalla, que sea amigable al usuario y fácil de interpretar.
- Generar un sonido de alerta cuando el cronómetro llegue al final de su cuenta.

4. Descripción del sistema

El sistema se puede dividir en cuatro subsistemas, el controlador de teclado, el controlador de la pantalla, el controlador para el RTC y el microcontrolador Picoblaze con su respectiva ROM de instrucciones. Éstos subsistemas, pueden ser desarrollados de manera separada siempre que se tenga el cuidado necesario con los datos que comparten entre los bloques. En la Fig. 1 se puede observar la composición general del sistema.

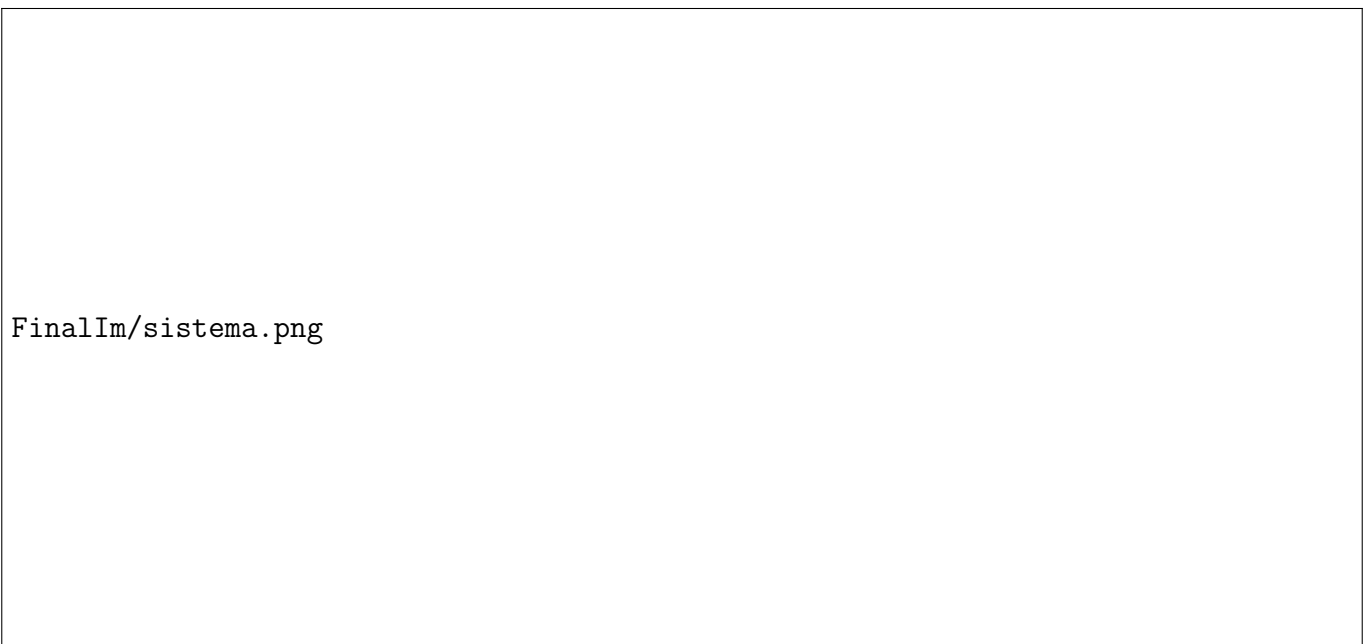


Figura 1: Diagrama de módulos principales del sistema.

4.1. Controlador de la pantalla

4.1.1. Diagrama de primer nivel

El diagrama general del controlador de la pantalla se muestra en la Fig. 2. En ésta podemos observar que el controlador tiene 5 entradas, el reloj que nos proporciona la Nexys 3 el cual no va a ser utilizado directamente, el reset es el que coloca todos los valores internos en 0, de modo que se dé un buen funcionamiento del bloque; y las entradas llamadas id_port, write_strobe “Data”son enviadas por el picoblaze. La entradas write_strobe es la indicadora de que el picoblaze está escribiendo en el VGA,

Proyecto	3	Página	3/25
Trabajo	Control y programación RTC con Nexys 4	Actualizado en:	30/09/2016
Grupo	8	Revisado en:	30/09/2016
Revisado por:	Alfonso Chacón Rodríguez	Diseñadores	Luis Leon Vega Luis Merayo Gatica Keylor Mena Venegas

mientras que id_port direcciona el dato al registro correspondiente, pues todos los datos que necesita el VGA son enviados por un mismo byte, por lo cual el id_port nos indica a cual dato corresponde “Data”.

En las salidas son las necesarias para que la pantalla funcione de la manera adecuada y además la denominada “audio” genera la señal para que funcione el sonido requerido. El Hsync es el encargado de generar la señal de sincronización horizontal, que va a estar activa durante el tiempo necesario para que la pantalla despliegue la información de cada pixel en una columna y Vsync va a estar activa mientras pasamos por todas las filas. La información de la que se está hablando va a estar contenida en la salida llamada “Color”, pues en Color se encuentran los 8 bits de información del color exacto que se debe desplegar en la dirección indicada por Vsync y Hsync. La salida “audio” se genera en este bloque ya que como debe ser generada cuando se llegue el final del cronómetro se considera innecesario hacer un módulo aparte para realizar esta simple acción.

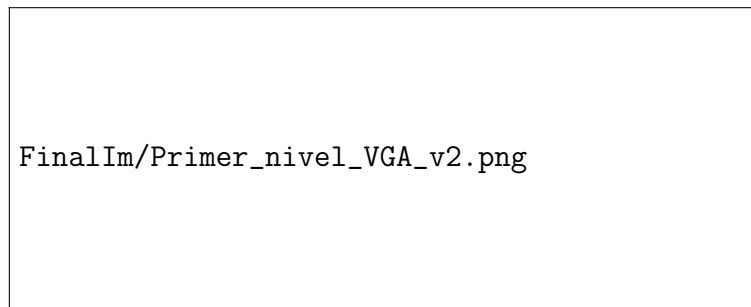


Figura 2: Diagrama de primer nivel del VGA.

4.1.2. Diagrama de segundo nivel

Una descripción más detallada del controlador VGA se encuentra en la Fig. 3, en la que podemos observar 3 bloques: el bloque sincronizador, el generador de texto y figuras, y el rgb mux. Estas división de los bloques se realiza basándonos en el funcionamiento del controlador.

El bloque sincronizador es el encargado de generar la secuencia con la cual nos vamos a ir desplazando pixel por pixel, Hsync y Vsync. Además nos indica la dirección de pixel en la cual nos encontramos (señales pixel_x y pixel_y), se pueden tratar análogamente como un par ordenado.

El bloque generador de texto, figuras e imágenes es el cual, por medio de las coordenadas, va indicando que ?color? se debe poner en esa dirección, de modo que la combinación va generando la interfaz. Dependiendo de la dirección en la que se encuentre se va seleccionar una imagen; del mismo modo, dependiente de la posición, pero también dependiente de la información proveniente de la “Data” y posicionada en el registro correcto mediante la señal id_port si se está escribiendo en el VGA (lo cual se indica por medio de write_strobe), se van a generar los cambios en la interfaz modificados por el paso del tiempo (cambio en la numeración generada al usar una ROM), por el cambio en la programación de tanto la fecha, el reloj o el cronómetro, o por la finalización del cronómetro. Si se da la finalización del cronómetro, también se genera la señal para que se active el audio.

El bloque llamado “rgb mux” se encarga de colocar el color si estamos direccionados en la parte de la pantalla visible (640x480) de no ser así coloca el color negro.

Proyecto	3	Página	4/25
Trabajo	Control y programación RTC con Nexys 4	Actualizado en:	30/09/2016
Grupo	8	Revisado en:	30/09/2016
Revisado por:	Alfonso Chacón Rodríguez	Diseñadores	Luis Leon Vega Luis Merayo Gatica Keylor Mena Venegas

FinalIm/Segundo_nivel_VGA_v2.png

Figura 3: Diagrama de segundo nivel del VGA.

4.1.3. Diagrama de tercer nivel

Como se observa en la Fig. 3 el controlador de VGA posee cinco entradas las cuales son tratadas y utilizadas de modo que se generen las cuatro salidas. Sin embargo los bloques que se muestran en esta figura son bastante amplios, por lo cual en la Fig. 4 se muestra un diagrama más específico.

Como se puede observar en la Fig. 4 se conservan los bloques generales mencionados en la sección 4.1.2 pero se muestra lo que ellos contienen.

Dentro del bloque sincronizador se encuentra: el contador y el sincronizador. El contador realiza la función de reducir el reloj, proporcionado por la Nexys 3, de 100 MHz a 25 MHz; pues necesitamos 25 MHz para el ritmo de pixel a modo de poder correr los 800 pixeles por línea, 525 líneas por pantalla y unas 60 pantallas por segundo. Mientras que el sincronizador realiza la función de ir recorriendo, pixel por pixel, la pantalla, permitiéndonos ubicar cada pixel que “señala” y poder generar imágenes con esa información. De la misma forma, las señales de sincronización, tanto vertical como horizontal, permiten a la pantalla, conectada por VGA, ir ubicando cada pixel y desplegando en él la información de color que le indiquemos.

En el bloque generador de texto, figuras e imágenes se tienen varios módulos: registros y handshake, decodificador de caracteres e imágenes, audio y la memoria de fuente ROM.

Los registros y handshake se encargan de colocar el “Data”, el cual es un byte, en el registro provisional correspondiente siempre y cuando se indique que se está escribiendo, por medio de la entrada write_strobe, y la dirección de escritura es proporcionada por la entrada id_port corresponda a las seleccionadas y especificadas, estas dos últimas entradas son generadas en el picoblaze; además este bloque se encarga de guardar en los registros los datos provisionales si se indica un handshake, lo cual significa que se ha terminado de escribir y todos los datos provisionales son los actuales.

El decodificador de caracteres e imágenes se encarga de colocar la información correspondiente en la pantalla, dependiendo tanto de la dirección del pixel como de la información recolectada del picoblaze. En este módulo se combina tanto la parte de la generación de las imágenes como la parte del texto ambas generadas por procesos diferentes y combinadas mediante el uso de un tipo de multiplexor. Las direcciones que envía el picoblaze por medio del id_port y su correspondiente registro se describen en

Proyecto	3	Página	5/25
Trabajo	Control y programación RTC con Nexys 4	Actualizado en:	30/09/2016
Grupo	8	Revisado en:	30/09/2016
Revisado por:	Alfonso Chacón Rodríguez	Diseñadores	Luis Leon Vega Luis Merayo Gatica Keylor Mena Venegas

la Tabla 1

El audio es un pequeño módulo que se encarga de generar la frecuencia con la cual suena un buzzer cuando el cronómetro ha llegado a su fin, lo que implica que hay un tipo de sincronización con el “Data” visual mostrado en pantalla; éste se estima de manera tal que tarde aproximadamente 8 segundos sonando que el mismo tiempo que el “ring” visual se va a mostrar.

La memoria de fuente ROM es la que contiene los números y unas pocas letras que van a ser utilizadas para exponer los datos del reloj, esta memoria se encuentra diseñada para trabajar con un tamaño de letra de 16x32, de trabajarse con un mayor tamaño, escalarlo, la numeración pierde detalle.

Con todos los bloques descritos anteriormente se pretende lograr a formar una interfaz como la mostrada en la Fig. 5, pero que además de ello incluya los números provenientes de la ROM. Este fondo se considera apropiado y amigable para el usuario.

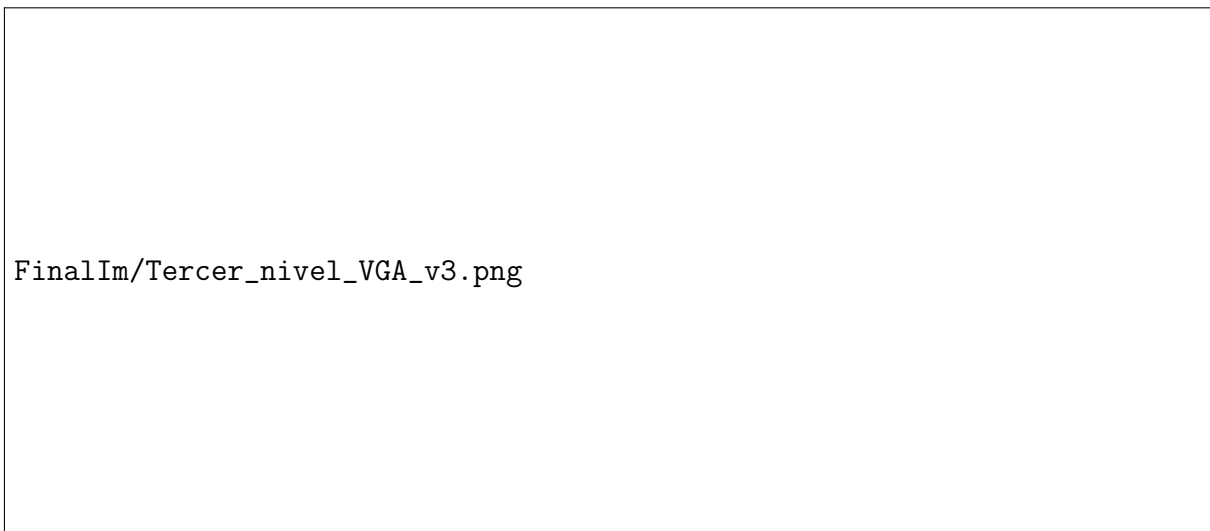


Figura 4: Diagrama de tercer nivel del VGA.

4.2. Controlador del Teclado

Para poder controlar el teclado, que se comunica por medio del protocolo PS2, se necesita comprender el funcionamiento del mismo. Éste posee dos señales que envía: un reloj y los datos en serie. Cuando se envía el dato el reloj debe estar en alto por lo cual debemos realizar un “PULLUP” en el mismo y también en los datos, ya que se pueden enviar o recibir datos del mismo. En este caso sólo se reciben datos del mismo pero aún así se debe realizar este “PULLUP”. Comprendiendo mejor el funcionamiento del teclado se realizan los diagramas de bloques que se muestran a continuación.

4.2.1. Diagrama de primer nivel

El lector de teclado es el que se encarga de tomar el dato que viene en serie y convertirlo al byte necesario para ser manejado por el PicoBlaze y además decodificará el dato, el teclado es la entrada de este módulo, el cual genera por sí mismo dos señales, el reloj y el dato. A la salida del módulo encontramos una interrupción, la cual va a indicar al picoBlaze que hay una tecla que se presionó, es aquí donde se nota la importancia de la decodificación, ya que con ella evitamos estar enviando

Proyecto	3	Página	6/25
Trabajo	Control y programación RTC con Nexys 4	Actualizado en:	30/09/2016
Grupo	8	Revisado en:	30/09/2016
Revisado por:	Alfonso Chacón Rodríguez	Diseñadores	Luis Leon Vega Luis Merayo Gatica
		Keylor Mena Venegas	

Cuadro 1: Dirección de escritura y su correspondiente registro

Registro	Dirección
Decenas del año	0x04
Unidades del año	0x05
Decenas del mes	0x06
Unidades del mes	0x07
Decenas del día	0x08
Unidades del día	0x09
Decenas hora del reloj	0x0a
Unidades hora del reloj	0x0b
Decenas minuto del reloj	0x0c
Unidades minuto del reloj	0x0d
Decenas segundo del reloj	0x0e
Unidades segundo del reloj	0x0f
Decenas hora del cronómetro	0x10
Unidades hora del cronómetro	0x11
Decenas minuto del cronómetro	0x12
Unidades minuto del cronómetro	0x13
Decenas segundo del cronómetro	0x14
Unidades segundo del cronómetro	0x15
Formato hora y am_pm	0x16
Dirección del cursor	0x17
Dato a programar	0x18
Handshake	0x19
Decenas hora del cronómetro programadas	0x1a
Unidades hora del cronómetro programadas	0x1b
Decenas minuto del cronómetro programadas	0x1c
Unidades minuto del cronómetro programadas	0x1d
Decenas segundo del cronómetro programadas	0x1e
Unidades segundo del cronómetro programadas	0x1f
Fin del cronómetro	0x20

Proyecto	3	Página	7/25
Trabajo	Control y programación RTC con Nexys 4	Actualizado en:	30/09/2016
Grupo	8	Revisado en:	30/09/2016
Revisado por:	Alfonso Chacón Rodríguez	Diseñadores	Luis Leon Vega Luis Merayo Gatica Keylor Mena Venegas

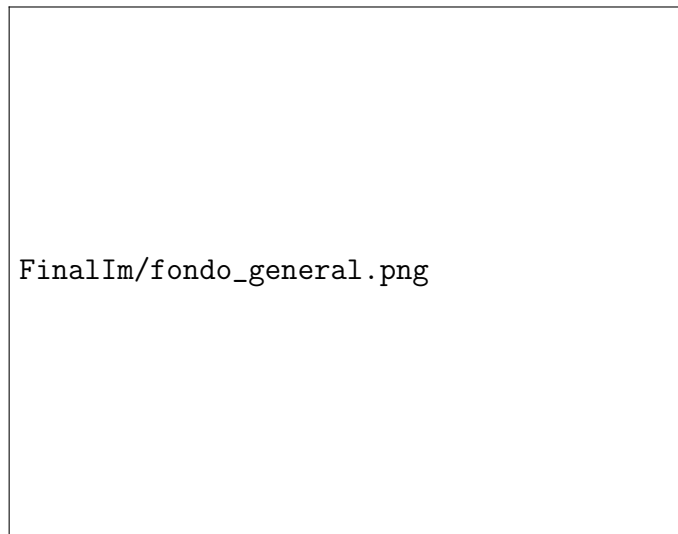


Figura 5: Interfaz del VGA.

interrupciones al picoblaze cuando cualquier tecla se presiona, sin importar que sea o no las teclas que se utilizan. Mientras que “Tecla” posee el código de la tecla que se ha presionado. En la Fig. 6 se muestra el diagrama general de lo anteriormente descrito.

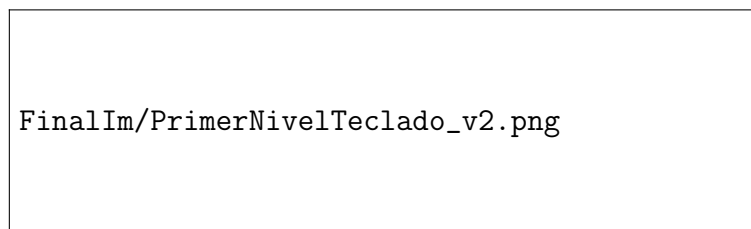


Figura 6: Diagrama de primer nivel del Teclado.

4.2.2. Diagrama de segundo nivel

En este diagrama mostrado en la Fig. 7 se muestra como se pretende realizar el lector de teclado, el cual consiste en tan solo tres bloques.

En el receptor de dato se toma el dato que viene en serie y se guarda en un registro, en donde ya se obtienen los 11 bits que nos envía el teclado. De este se obtiene el byte que contiene el dato y también una bandera que indica cuando se ha terminado de leer el dato.

En el detector de tecla se evalúa si el dato que se obtiene del teclado indica que una tecla ha sido presionada, y se obtiene el código de esa tecla. En el PS2 cuando se presiona y se suelta una tecla se obtiene el código en hexadecimal 1c f0 1c, en donde 0x1c denota el código de la tecla y 0xf0 indica que se ha dejado de presionar esa tecla, por lo cual se busca identificar el código 0xf0 para guardar el dato 0x1c que es la tecla en interés. La salida del bloque es la tecla que se acaba de presionar, pero antes de esto también se envía el código 0xf0 para que cada vez que se suelta una tecla el siguiente bloque conozca esta información, ésto por la lógica del bloque “decodificador de tecla”.

El decodificador de tecla va a asegurar que las teclas que se presionan son las que se van a utilizar, de

Proyecto	3	Página	8/25
Trabajo	Control y programación RTC con Nexys 4	Actualizado en:	30/09/2016
Grupo	8	Revisado en:	30/09/2016
Revisado por:	Alfonso Chacón Rodríguez	Diseñadores	Luis Leon Vega Luis Merayo Gatica Keylor Mena Venegas

modo que se asegure que cuando se manda la interrupción al picoblaze sea debido a una tecla de las que son utilizadas. En este bloque se encuentran unos flip-flop's tipo D, los cuales nos van a ayudar a generar la lógica, debido a que éstos van actualizando sus datos cada ciclo de reloj y se van pasando la información de uno a otro si tan solo se manda el dato se la tecla que se acaba de soltar y no se manda el 0xf0, de presionarse consecutivamente la misma tecla este bloque no envía la señal de interrupción al picoblaze, por lo cual no se genera un correcto funcionamiento. En el mapeo de memoria de lectura del Picoblaze el dato de Tecla es leído en la dirección 0x04.

FinalIm/SegundoNivelTeclado_v2.png

Figura 7: Diagrama de segundo nivel del Teclado.

4.3. Controlador del RTC

Para la implementación de esta interfaz que va a permitir la comunicación entre el micro y el RTC, se desarrolló un bloque que permite leer y escribir datos, basado en los tiempos ajustados en el proyecto anterior, de manera que el Picoblaze indique la dirección, el dato a escribir en caso de escritura y si se debe escribir o leer, este bloque a su vez va a indicar mediante una bandera que ha terminado de realizar el debido proceso para que se pueda leer el dato extraído y se pueda proseguir con la siguiente acción sobre el RTC. A continuación en la Fig. 8 se muestra un diagrama donde se especifican los componentes que componen este bloque.

Cuando el Picoblaze realiza una escritura, especifica la dirección con el id_port y con el write-f indica que se está escribiendo, además entrega el dato en el bus dpico, cuando se escribe la función correspondiente a lectura o escritura, se habilita el contador para que en conjunto con el decodificador se vayan alternando las señales de control, PUP que es la señal de alta impedancia y el dato de ready que indica al micro que puede proseguir con la siguiente acción. Con los registros se almacenan los datos de dirección, dato de escritura, función, dato extraído y un byte que indica el estado de la bandera de ampm y el formato del reloj al momento de la lectura. Con el deco se habilita la carga del registro del dato extraído, la cuenta y la salida del mux especificando si debe ser dirección, dato o unos. En la Tabla 2 se detalla el mapeo de memoria respectivo para el controlador del RTC.

Proyecto	3	Página	9/25
Trabajo	Control y programación RTC con Nexys 4	Actualizado en:	30/09/2016
Grupo	8	Revisado en:	30/09/2016
Revisado por:	Alfonso Chacón Rodríguez	Diseñadores	Luis Leon Vega Luis Merayo Gatica Keylor Mena Venegas

FinalIm/rtcp.png

Figura 8: Diagrama de tercer nivel del controlador del RTC.

4.4. ROM de Instrucciones

En el diseño de la ROM de instrucciones, deben quedar muy claras las rutinas a desarrollar para que estas sean llamadas en una rutina principal según las condiciones dadas por el usuario mediante el uso de teclas, por lo que también se debe desarrollar la programación que obtenga estas teclas utilizando la facilidad que nos brinda la rutina de interrupción. A continuación en la Fig. 9 se muestra el diagrama que explica el comportamiento de la rutina principal.

La Fig. 9 nos da una noción del proceso bajo el que va a operar el sistema, pero es necesario detallar las subrutinas llamadas en este programa principal por lo que seguidamente se va a explicar el funcionamiento de dichas subrutinas con sus respectivos diagramas.

Cuadro 2: Mapeo de memoria del controlador del RTC

Registro	Dirección
Escritura	
Dirección	0x00
Función	0x01
Decenas del DatoW	0x02
Unidades del DatoW	0x03
Lectura	
Decenas del DatoExt	0x00
Unidades del DatoExt	0x01
AmPmFormato	0x02
ListoRTC	0x03

Proyecto	3	Página	10/25
Trabajo	Control y programación RTC con Nexys 4	Actualizado en:	30/09/2016
Grupo	8	Revisado en:	30/09/2016
Revisado por:	Alfonso Chacón Rodríguez	Diseñadores	Luis Leon Vega Luis Merayo Gatica Keylor Mena Venegas



Figura 9: Diagrama de flujo de instrucciones principal.

Proyecto	3	Página	11/25
Trabajo	Control y programación RTC con Nexys 4	Actualizado en:	30/09/2016
Grupo	8	Revisado en:	30/09/2016
Revisado por:	Alfonso Chacón Rodríguez	Diseñadores	Luis Leon Vega Luis Merayo Gatica Keylor Mena Venegas

4.4.1. Ini_rtc

Esta rutina se encarga de inicializar el RTC bajo el respectivo proceso indicado en la hoja de datos que implica la escritura de ciertas variables por lo que se utiliza el controlador del RTC, esta rutina a su vez tiene incluida la rutina inicrRTC que se encarga de la escritura de ceros en los datos del cronómetro, para que este inicie en cero, en la Fig. 10 se puede ver que es un proceso sencillo.

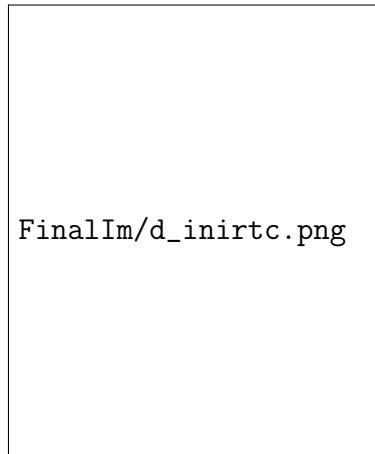


Figura 10: Diagrama de flujo de inicrtc.

4.4.2. Ini_variables

Esta rutina se encarga de escribir ceros en los espacios de memoria donde se almacenan las teclas y en el byte que indica la finalización del cronómetro, esto con tal de evitar que se evalúe basura almacenada en estos datos.

4.4.3. Leer_rtc

Con esta rutina se van a leer los datos de hora, fecha y cronómetro para que sean mostrados en la pantalla al usuario, también se van a extraer los datos referentes al formato y estado del cronómetro para que cuando se presione la tecla correspondiente se cambien dicho estado en base al actual. En la Fig. 11 se muestra el desarrollo de esta rutina.

4.4.4. Escribir datos en el VGA

Esta rutina toma los datos almacenados en la memoria del Picoblaze de la hora, fecha, cronómetro, un byte que define el formato y si la hora es am o pm, además de los datos que indican si se debe mostrar el cursor y en que posición. Básicamente carga dichos datos y los escribe en el modulo VGA basado en el mapeo de memoria de los periféricos.

4.4.5. Prog_hora, Prog_fecha y Prog_crono

Se encargan de evaluar si se presiona alguna flecha o numero para desplazar el cursor o modificar el dígito en el cual está ubicado el cursor en ese momento, además verifica constantemente si se ha presionado la tecla escape para dar por terminada la rutina, como son procesos similares, se va a mostrar solamente el diagrama de programación de la hora en la Fig. 12

Proyecto	3	Página	12/25
Trabajo	Control y programación RTC con Nexys 4	Actualizado en:	30/09/2016
Grupo	8	Revisado en:	30/09/2016
Revisado por:	Alfonso Chacón Rodríguez	Diseñadores	Luis Leon Vega Luis Merayo Gatica
			Keylor Mena Venegas



Figura 11: Diagrama de flujo de Leer_rtc.

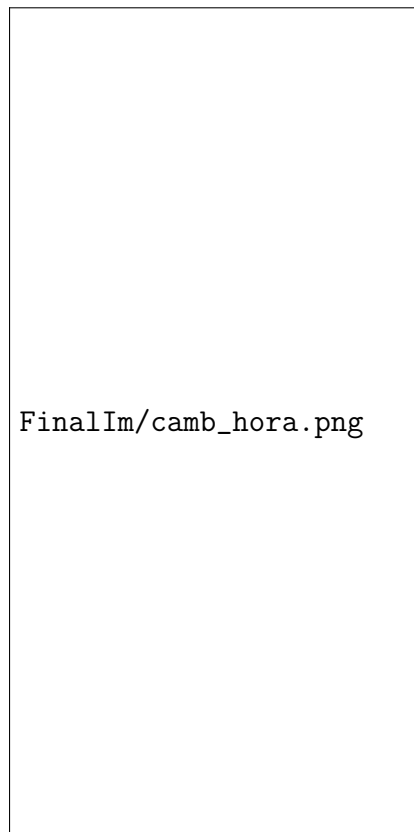


Figura 12: Diagrama de flujo de Prog_hora.

Proyecto	3	Página	13/25
Trabajo	Control y programación RTC con Nexys 4	Actualizado en:	30/09/2016
Grupo	8	Revisado en:	30/09/2016
Revisado por:	Alfonso Chacón Rodríguez	Diseñadores	Luis Leon Vega Luis Merayo Gatica Keylor Mena Venegas

4.4.6. Camb_formato y Camb_cronoini

Dado que los bits que definen si el formato de la hora es de 24 o 12 horas y si el cronometro debe contar o no, se encuentran en la misma posición de memoria de estatus del RTC, cuando se cambia alguno de los dos se debe escribir un byte que contiene ambos, por lo que las rutinas de cambiar formato y cronoini toman en cuenta el bit referente a la variable a modificar, lo niegan y escriben en el RTC dicho byte, en la Fig. 13 se muestra el proceso para cambiar el formato, similar al de cambiar cronoini.

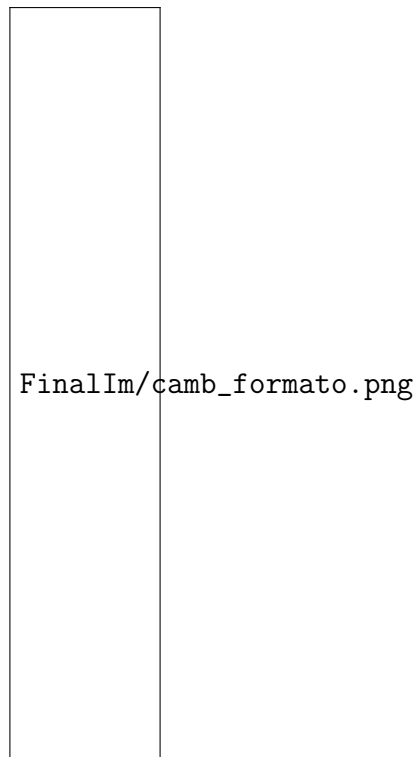


Figura 13: Diagrama de flujo de Camb_formato.

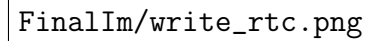
4.4.7. Escribir_RTC

En caso de que se hayan cambiado los datos de la hora o fecha, se deben actualizar los datos en el RTC por lo que con esta rutina se realiza esta acción, actualiza los datos de hora, fecha y cronómetro, de esta forma se implementa el efecto de pausar la cuenta del reloj cuando se programa alguna de estas variables. En la Fig. 14 se muestra este proceso de escritura.

4.4.8. Comp_crono

Esta rutina carga los datos programados por el usuario del cronómetro y los datos leídos del RTC y escribe en una posición de memoria 0XFF si son iguales y 0X00 si son distintos, de esta manera se determina cuando el cronómetro llega a su fin para que este sea detenido con la rutina camb_cronoini, reiniciado con inicrRTC y mostrado en pantalla con Escribir_Ring que escribe en un registro del módulo de VGA para que realice la respectiva alarma.

Proyecto	3	Página	14/25
Trabajo	Control y programación RTC con Nexys 4	Actualizado en:	30/09/2016
Grupo	8	Revisado en:	30/09/2016
Revisado por:	Alfonso Chacón Rodríguez	Diseñadores	Luis Leon Vega Luis Merayo Gatica Keylor Mena Venegas

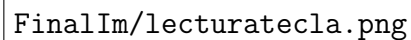


FinalIm/write_rtc.png

Figura 14: Diagrama de flujo de Escribir_RTC.

4.4.9. Lectura de tecla

Cuando una tecla es presionada, el controlador del teclado indica esta acción mediante la entrada de interrupción del Picoblaze, de esta forma el micro ingresa a esta rutina encargada de leer la tecla y clasificarla, de esta forma la tecla puede ser evaluada en las demás rutinas, en el diagrama de la Fig. 15 se muestra en que consiste este proceso.



FinalIm/lecturatecla.png

Figura 15: Diagrama de Lectura de tecla.

Proyecto	3	Página	15/25
Trabajo	Control y programación RTC con Nexys 4	Actualizado en:	30/09/2016
Grupo	8	Revisado en:	30/09/2016
Revisado por:	Alfonso Chacón Rodríguez	Diseñadores	Luis Leon Vega Luis Merayo Gatica Keylor Mena Venegas

5. Datos y resultados

5.1. Simulaciones

Se realizan varias simulaciones del control de VGA en donde se va a notar como sucede el proceso de guardar los datos y la función del handshake. Además, se encuentran simulaciones donde se muestra el correcto funcionamiento de la sincronización.

En la Fig. 16 se puede notar el cambio en algunas de las señales, esta figura tiene especial énfasis en que se note pix_x, el cual nos identifica en cual pixel de la coordenada x nos encontramos, y PT que es la señal de reloj para el VGA (25 MHz). En la Fig. 17 se puede apreciar el cambio en pix_y, que es el pixel en la coordenada y en el que nos encontramos, hsincro, que es la señal de sincronía horizontal, ON_VID, que nos indica cuando estamos dentro de los 640x480, y color_salida, que es el color que se coloca en la pantalla en cada pixel que nos encontramos (0x49 es un tipo de gris y 0x00 es negro). Si realizamos un acercamiento a esta imagen podemos verificar que cuando se den los cambios en las señales hsincro y ON_VID en los pixeles indicados, lo cual podemos ver en la Fig. 18 en la cual podemos notar en la parte izquierda unos valores que indican el valor de cada variable en donde está posicionado el cursor y con este valor podemos verificar algunos de las características. En la Fig. 19 se puede observar cuando vsincro se pone en bajo, esta señal corresponde a la señal de sincronización vertical, lo cual sucede durante un pequeño lapso casi al final de la cuenta vertical.

Luego nos aseguramos del funcionamiento del handshake y los registros. La primera de ellas es la



FinalIm/CambioSenialesVGA.PNG

Figura 16: Cambio en las señales del VGA, énfasis generación del pix_x y PT.



FinalIm/CambioSenialesVGA2.PNG

Figura 17: Cambio en las señales del VGA, énfasis generación del pix_y, hsincro, ON_VID y color_salida.

que se muestra en la Fig. 20 en la cual podemos observar como se van ingresando los datos en los correspondientes registros provisionales si el write_strobe se encuentra en alto; estos datos se guardan en las direcciones que indica el id_port las cuales corresponden a los registros o datos indicados en la

Proyecto	3	Página	16/25
Trabajo	Control y programación RTC con Nexys 4	Actualizado en:	30/09/2016
Grupo	8	Revisado en:	30/09/2016
Revisado por:	Alfonso Chacón Rodríguez	Diseñadores	Luis Leon Vega Luis Merayo Gatica Keylor Mena Venegas

FinalIm/CambioSenialesVGA2_closeup.PNG

Figura 18: Acercamiento de la Fig. 17

FinalIm/CambioSenialesVGA2_faraway.PNG

Figura 19: Cambio en las señales del VGA, énfasis generación del vsincro.

Tabla 1. Si alejamos un poco la simulación obtenemos la Fig. 21 en la cual podemos ver el cambio de todos los registros provisionales. En la Fig.22 se muestra que cuando hay un handshake en 0xf0 entonces los datos se pasan de los registros provisionales a los registros de los cuales los demás bloques toman la información para interpretarla.

Para asegurar el correcto funcionamiento del teclado se realizan varias simulaciones. Una de ellas es la simulación de detección de tecla, mostrada en la Fig. 23, en donde vemos que después de que se nota un 0xf0 se envía esta tecla seguida de la tecla que se soltó, y notamos que hasta que la bandera scan_done_tick indique que ya se han terminado de leer los datos no ocurre ningún proceso, no se pasa la tecla a la salida; también se puede observar que hasta que no exista un 0xf0 no se lee nada. En la Fig. 24 se muestra la simulación de la decodificación de la tecla, en donde podemos observar que si la tecla no es la correcta no se envía la interrupción aunque en los flip-flops se tengan datos distintos, cuando se indique que la tecla es la correcta (la tecla es una de las dispuestas a utilizar) y si en los flip-flops hay datos de teclas distintas se genera la interrupción la cual se va a mantener activa hasta que el pucoblaze envíe la señal de paro de interrupción; también se puede observar que si hay una interrupción y el picoblaze no ha enviado el paro suceda lo que suceda con los demás procesos (tecla correcta y flip-flops con distintos datos) no va a ocurrir nada.

Para corroborar el funcionamiento del Picoblaze se realizó la respectiva simulación contemplando las señales que interactúan con los periféricos, en la Fig. 25 se puede ver cómo se comportan las señales de read y write en conjunto con el in_port y out_port mientras transcurren las instrucciones brindadas por la ROM.

De igual forma se simuló el comportamiento del controlador del RTC en conjunto con el Picoblaze para observar el cumplimiento de tiempos en las señales de control, dirección y adquisición del dato, en las Fig. 26 y 27 se puede observar los procesos de lectura y escritura que cumplen con los tiempos

Proyecto	3	Página	17/25
Trabajo	Control y programación RTC con Nexys 4	Actualizado en:	30/09/2016
Grupo	2	Revisado en:	30/09/2016
Revisado por:	Alfonso Chacón Rodríguez	Diseñadores	Luis Leon Vega Luis Merayo Gatica Keylor Mena Venegas

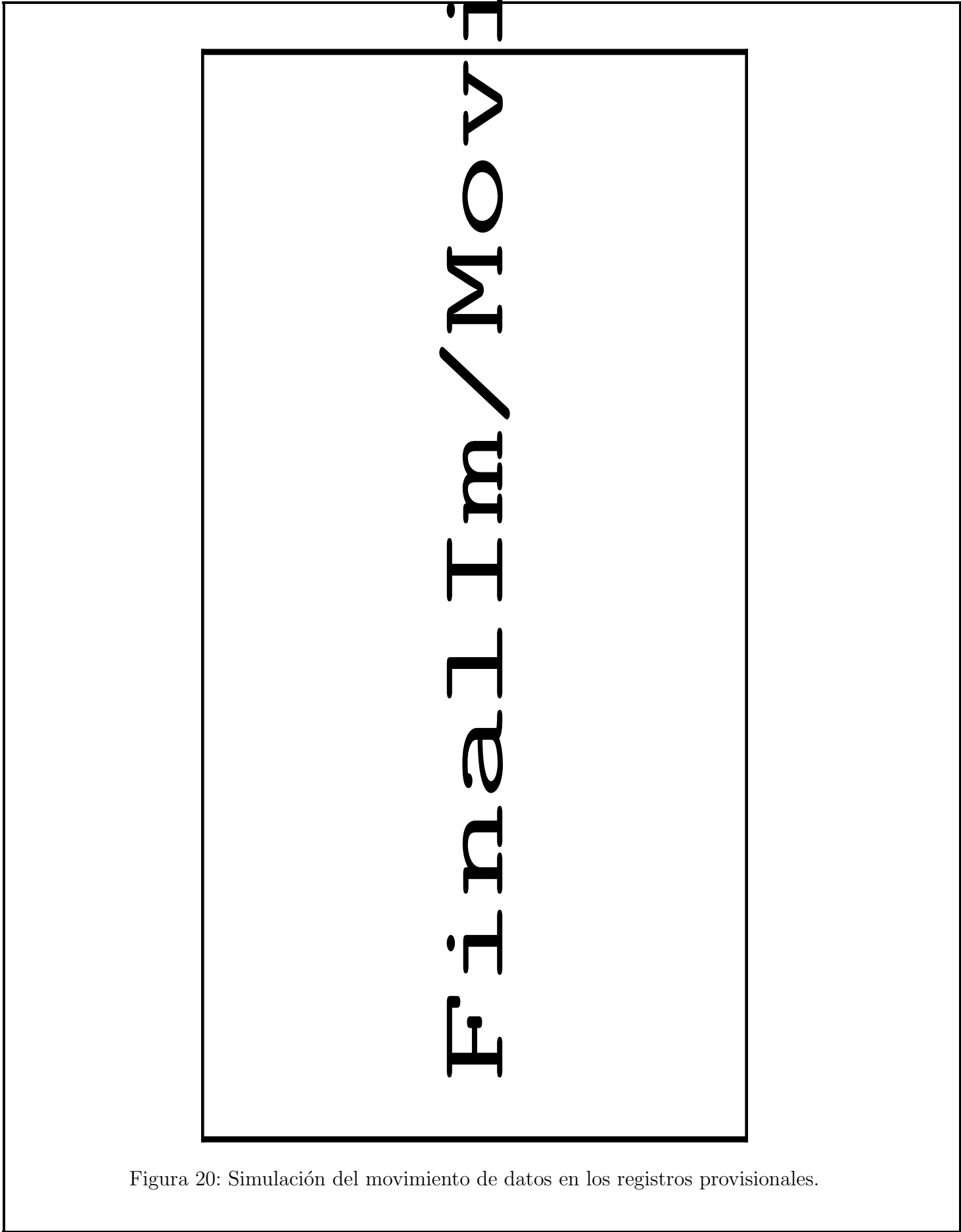


Figura 20: Simulación del movimiento de datos en los registros provisionales.

Proyecto	3	Página	18/25
Trabajo	Control y programación RTC con Nexys 4	Actualizado en:	30/09/2016
Grupo	2	Revisado en:	30/09/2016
Revisado por:	Alfonso Chacón Rodríguez	Diseñadores	Luis Leon Vega Luis Merayo Gatica Keylor Mena Venegas

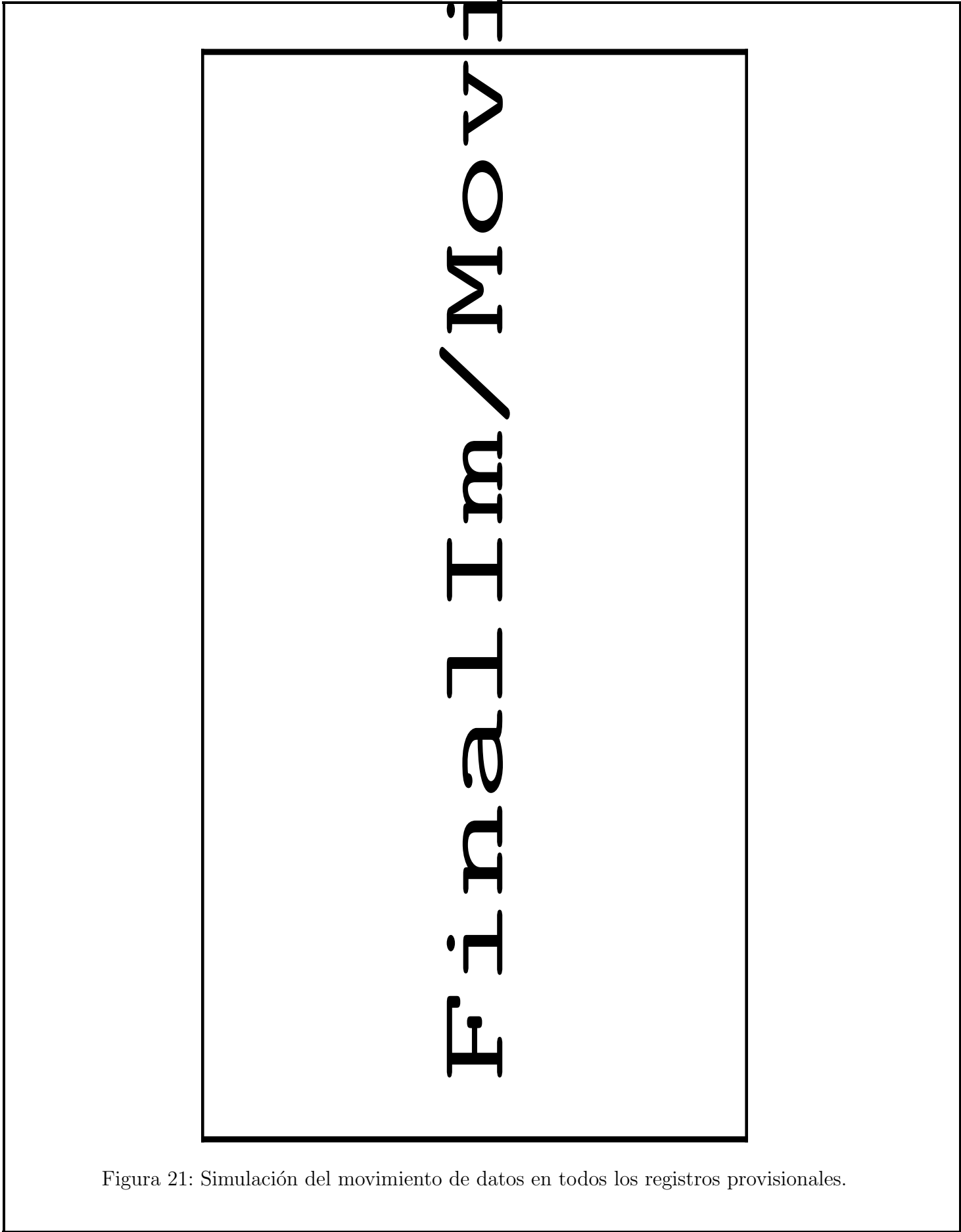


Figura 21: Simulación del movimiento de datos en todos los registros provisionales.

Proyecto	3	Página	19/25
Trabajo	Control y programación RTC con Nexos 4	Actualizado en:	30/09/2016
Grupo	2	Revisado en:	30/09/2016
Revisado por:	Alfonso Chacón Rodríguez	Diseñadores	Luis Leon Vega Luis Merayo Gatica Keylor Mena Venegas

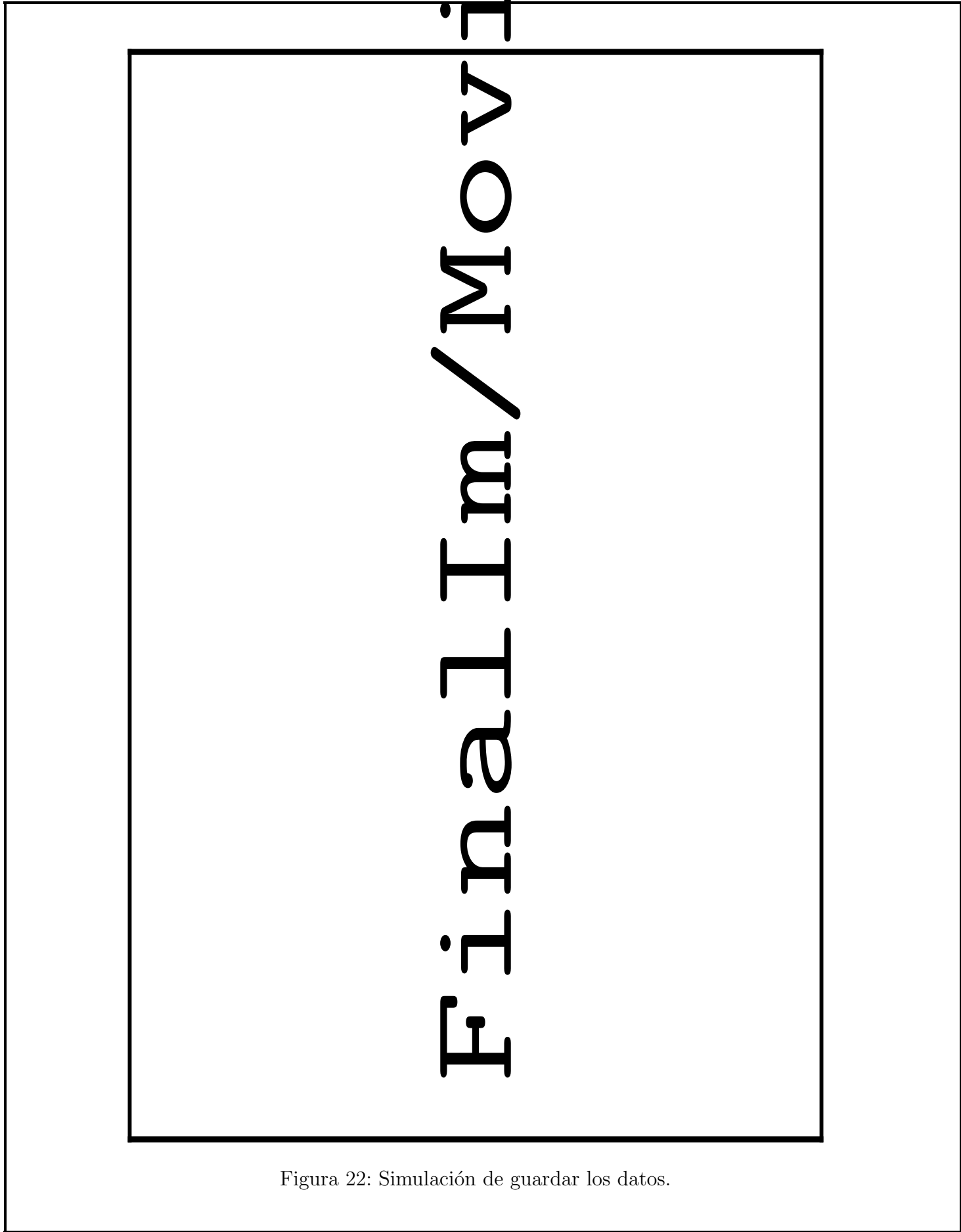


Figura 22: Simulación de guardar los datos.

Proyecto	3	Página	20/25
Trabajo	Control y programación RTC con Nexys 4	Actualizado en:	30/09/2016
Grupo	8	Revisado en:	30/09/2016
Revisado por:	Alfonso Chacón Rodríguez	Diseñadores	Luis Leon Vega Luis Merayo Gatica
			Keylor Mena Venegas

FinalIm/DeteccionTecla.PNG

Figura 23: Simulación de la detección de una tecla.

FinalIm/DecodificacionTecla.PNG

Figura 24: Simulación de la decodificación de una tecla.

FinalIm/simpico.png

Figura 25: Simulación de señales del Picoblaze.

Proyecto	3	Página	21/25
Trabajo	Control y programación RTC con Nexys 4	Actualizado en:	30/09/2016
Grupo	8	Revisado en:	30/09/2016
Revisado por:	Alfonso Chacón Rodríguez	Diseñadores	Luis Leon Vega Luis Merayo Gatica Keylor Mena Venegas

ajustados según el proyecto anterior.




Figura 26: Simulación de lectura del RTC.

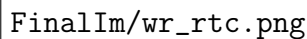


Figura 27: Simulación de escritura del RTC.

5.2. Mediciones

Para este apartado se van a tomar en cuenta el resultado del análisis del consumo de potencia del dispositivo y consumo de recursos dado que en el proyecto anterior quedo claro con las mediciones realizadas que se cumplía con los tiempos necesarios para controlar el RTC. En la Tabla 3 se pueden ver los datos de consumo del dispositivo, los de potencia se obtienen al utilizar la herramienta llamada XPower Analyzer.

6. Análisis de datos y resultados

En el desarrollo de este proyecto se contempló primeramente integrar el manejo del RTC por completo en el microcontrolador, pero dado que los tiempos de reloj no coinciden y para poder hacer algo funcional se requiere de una gran cantidad de instrucciones, además de que se menciona que el picoblaze no puede realizar control sobre tiempos críticos, se optó por desarrollar el módulo controlador del RTC que permite leer y escribir y de esta forma se simplifico el diseño y la cantidad de instrucciones en la ROM destinadas a controlar este dispositivo.

Proyecto	3	Página	22/25
Trabajo	Control y programación RTC con Nexys 4	Actualizado en:	30/09/2016
Grupo	8	Revisado en:	30/09/2016
Revisado por:	Alfonso Chacón Rodríguez	Diseñadores	Luis Leon Vega
			Luis Merayo Gatica
			Keylor Mena Venegas

Cuadro 3: Consumo de recursos de la FPGA

Recurso	Consumo
Potencia	28mW
slices ocupadas	98 %
Registros	3 %
MUX	3 %
IO	14 %
LUTs lógicos	81 %
LUT FF	7501
DSP48A1	56 %

De igual forma no se implementó el cronómetro de forma descendente porque esto requiere de un módulo más para realizar una resta BCD, debido a que este es el formato en que entrega los datos el RTC, pero la forma en que se desarrolló deja claro al usuario la cuenta y el límite respectivo.

Se presentaron algunos otros problemas relacionados con pulgas en la programación, algunas de las cuales eran generadas por faltas de consideraciones, en lo que concierne a la programación en ensamblador se dieron pocos problemas dato que la herramienta de simulación facilitó y agilizó el proceso de diseño.

Debido a los tiempos entre instrucciones que maneja el picoblaze, se tiene que realizar un handshake de forma diferente a lo realizado en el proyecto anterior, además de la forma en que se implementó el ring; esto genera un cambio en la lógica que ocasiona algunas pulgas, que se logran solucionar.

En el caso del controlador de la pantalla VGA, parte de la programación se basa en lo desarrollado en el libro de Pong Chu [2], la parte de los números, es decir la parte de usar la ROM además de la sincronización. Mientras que la parte de las imágenes se realiza utilizando información encontrada en la web. Al generar el controlador se observa la gran diferencia respecto al proyecto anterior. Los problemas que se presentaron con este controlador no tuvieron que ver con la programación sino con la estética, pues se sufrió con los colores, puesto que la programación de Matlab que convierte las imágenes en archivos que pueden ser interpretados por la FPGA tenía un error, el cual al detectarlo dejó de afectar en la modificación de los colores.

Sin embargo, por la cantidad de espacio con la que cuenta la Nexys 3 como memoria RAM se tiene que reducir el tamaño de las imágenes, lo cual es un problema menor que no afecta visualmente si se vuelven a dimensionar las imágenes de forma que se continuen viendo agradables. Como se muestra en la Fig. 28 se obtiene una interfaz muy similar a lo planeado y esperado (Fig. 5)

De esta parte lo que causó más problema fue el ring, pues se tuvo que cambiar parte de la lógica cuando ya se estaba finalizando el proyecto lo que cambiaba a su vez la forma en que se podía generar el ring, sin embargo, se logró generar. Al tener ya listos el ring generar el sonido fue una cuestión nada más de generar el reloj que lo controlaba.

El controlador del teclado también se basó en lo que nos muestra el Pong Chu [2], pero se le agregaron los bloques que son propios de la forma de la lógica del resto del circuito. Como se puede ver en las simulaciones funciona a la perfección y en la práctica el teclado no tiene ningún problema.

Proyecto	3	Página	23/25
Trabajo	Control y programación RTC con Nexys 4	Actualizado en:	30/09/2016
Grupo	8	Revisado en:	30/09/2016
Revisado por:	Alfonso Chacón Rodríguez	Diseñadores	Luis Leon Vega Luis Merayo Gatica Keylor Mena Venegas

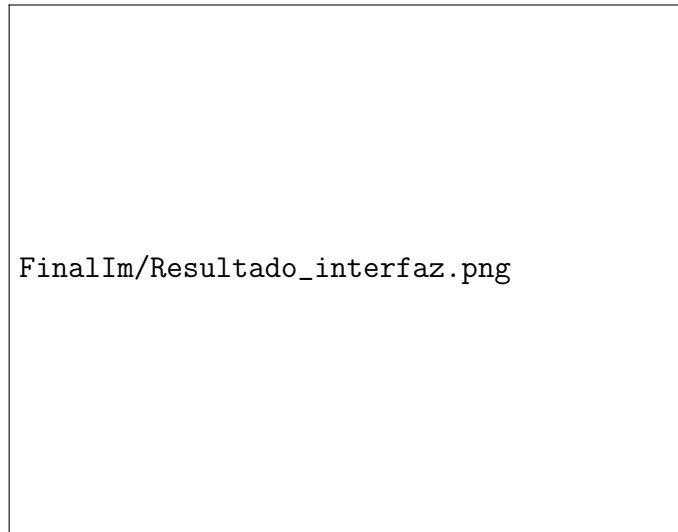


Figura 28: Resultado de la interfaz del VGA.

7. Hoja de datos de unidades desarrolladas

Dentro de las especificaciones finales se puede mencionar el uso del clock de 100 MHz de la Nexys 3, toda la lógica implementada opera bajo lógica positiva. En la tabla 4 se pueden observar los pines utilizados en la Nexys 3 para realizar las conexiones con el V3023 y parte de la interfaz con el usuario (VGA y teclado)

8. Conclusiones y recomendaciones

8.1. Conclusiones

- El uso de un microcontrolador de softcore como el Picoblaze facilita en gran medida el manejo de sistemas que hacen uso de más de un periférico y permiten que el sistema completo sea más organizado, además de facilitar el diseño en sí.
- Utilizar software libre para el desarrollo de ciertas aplicaciones, como el controlador VGA y parte del controlador de teclado simplifica el diseño, ahorra tiempo y permite utilizar ese tiempo en la optimización del diseño, lo cual es muy importante en todos los ámbitos, pues nos va a permitir tener un mejor diseño e inclusive uno más funcional.
- El buen diseño de las distintas interfaces para los periféricos es la clave para una correcta comunicación entre dispositivos y una programación en ensamblador simplificada.
- El uso del conteo ascendente en el cronómetro redujo el hardware utilizado y para el usuario es comprensible por completo la mecánica de esta funcionalidad.
- Utilizar imágenes para la generación de la interfaz visual hace más amena la experiencia del usuario con el dispositivo, a diferencia de utilizar un despliegue de pixeles escalado, que hace que se vea más cuadriculada la imagen y, por ende, antiestética.

Trabajo Control y programación RTC con Nexys 4

Grupo 8

Revisado por: Alfonso Chacón Rodríguez

Actualizado en: 30/09/2016

Revisado en: 30/09/2016

Diseñadores Luis Leon Vega

Luis Merayo Gatica

Keylor Mena Venegas

Cuadro 4: Asignación de pines de las entradas y salidas

Señal	Designación	Pin	Función
CLOCK_NEXYS	input	V10	Señal de reloj de la Nexys 4, es de 100MHz.
PS2C	input	L12	Señal de reloj del teclado
PS2D	input	J13	Datos en serie del teclado
DATA_IN[7]	input	F2	Bit de datos y direcciones de entrada.
DATA_IN[6]	input	J6	Bit de datos y direcciones de entrada.
DATA_IN[5]	input	J7	Bit de datos y direcciones de entrada.
DATA_IN[4]	input	G1	Bit de datos y direcciones de entrada.
DATA_IN[3]	input	G3	Bit de datos y direcciones de entrada.
DATA_IN[2]	input	K6	Bit de datos y direcciones de entrada.
DATA_IN[1]	input	L7	Bit de datos y direcciones de entrada.
DATA_IN[0]	input	H3	Bit de datos y direcciones de entrada.
DATA_OUT[7]	output	K5	Bit de datos y direcciones de salida.
DATA_OUT[6]	output	K3	Bit de datos y direcciones de salida.
DATA_OUT[5]	output	J1	Bit de datos y direcciones de salida.
DATA_OUT[4]	output	J3	Bit de datos y direcciones de salida.
DATA_OUT[3]	output	L3	Bit de datos y direcciones de salida.
DATA_OUT[2]	output	L4	Bit de datos y direcciones de salida.
DATA_OUT[1]	output	K1	Bit de datos y direcciones de salida.
DATA_OUT[0]	output	K2	Bit de datos y direcciones de salida.
COLORES[0]	output	R7	Bit para color azul.
COLORES[1]	output	T7	Bit para color azul.
COLORES[2]	output	P8	Bit para color verde.
COLORES[3]	output	T6	Bit para color verde.
COLORES[4]	output	V6	Bit para color verde.
COLORES[5]	output	U7	Bit para color rojo.
COLORES[6]	output	V7	Bit para color rojo.
COLORES[7]	output	N7	Bit para color rojo.
H_syncro	output	N6	Sincronización horizontal.
V_syncro	output	P7	Sincronización vertical.
R_D	output	V12	Señal para leer.
C_S	output	P11	Señal de selección.
W_R	output	T12	Señal para escribir.
A_D	output	N10	Selecciona entre dirección y dato.
sound	output	G11	Sonido.

Proyecto	3	Página	25/25
Trabajo	Control y programación RTC con Nexys 4	Actualizado en:	30/09/2016
Grupo	8	Revisado en:	30/09/2016
Revisado por:	Alfonso Chacón Rodríguez	Diseñadores	Luis Leon Vega Luis Merayo Gatica Keylor Mena Venegas

- Generar la manipulación de los datos del reloj por medio de un teclado hace más agradable la experiencia del usuario, además de que evita errores por ruido, como se experimentó en proyectos pasados.

8.2. Recomendaciones

- En caso de utilizar un microcontrolador como el Picoblaze es necesario dedicar un tiempo considerable a la lectura y comprensión de todos los aspectos a considerar, incluidos en el manual y la forma de programar e implementar la ROM de instrucciones, todo esto para minimizar errores futuros.
- Utilizar software libre para realizar el diseño del controlador de la pantalla, pues es software libre que funciona de forma correcta y que nos funciona para la función que necesitamos.
- Utilizar métodos de apoyo para la programación, como lo son los diagramas de estados, de modo que se simplifique la generación de código cuando ya se tenga bien claro el funcionamiento de los componentes y como se deben comportar según lo seleccionado en estos diagramas.
- Utilizar herramientas alternativas como FIDEX en el caso de la programación en ensamblador, ya que aunque maneja una sintaxis distinta, permite simular las diferentes rutinas de manera muy sencilla.
- Tener una buena comunicación entre los miembros del equipo, para evitar errores en la programación, los cuales causan atrasos que se pueden haber evitado si se comunica de forma efectiva la forma de manejar los datos.
- Conocer las capacidades de la plataforma de trabajo para evitar que después de haber realizado un diseño en la interfaz visual en la que se requerían muchas imágenes, se tenga que cambiar debido a que la plataforma de trabajo no tiene la capacidad para trabajar con tantas imágenes.
- Generar simulaciones minuciosas para corroborar el funcionamiento del software o hardware programado, pues al realizar esto se ahorra considerable tiempo en el momento de encontrar errores o detalles de funcionamiento.

Referencias

- [1] Chapman, K. (2014) *PicoBlaze for Spartan-6, Virtex-6, 7-Series, Zynq and UltraScale Devices (KCPSM6)*, Xilinx, Inc.
- [2] Chu, Pong P. (2008) *FPGA prototyping by Verilog examples. Xilinx SpartanTM-3 Version*, John Wiley & Sons, Inc.