SEP
SECRETARÍA DE EDUCACIÓN PÚBLICA
SUBSECRETARÍA DE EDUCACIÓN SUPERIOR
TECNOLÓGICO NACIONAL DE MÉXICO

TECNM
TECNOLÓGICO NACIONAL DE MÉXICO

TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE APIZACO

DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

T E S I S

Monitoreo remoto de dispositivos conectados a internet mediante FPGA con sistema operativo embebido Linux usando sensores Bus SPI

PARA OBTENER EL GRADO DE:

MAESTRO EN INGENIERÍA MECATRÓNICA

PRESENTA:

ALFREDO JESÚS PÉREZ CASTILLO

ASESOR:

DR. ROBERTO MORALES CAPORAL

Instituto Tecnológico de Apizaco

APIZACO, TLAXCALA AGOSTO 2018

TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE APIZACO

DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

T E S I S

“Monitoreo remoto de dispositivos conectados a internet mediante FPGA con sistema operativo embebido Linux usando sensores Bus SPI”

PARA OBTENER EL GRADO DE:

MAESTRO EN INGENIERÍA MECATRÓNICA

PRESENTA:

ALFREDO JESÚS PÉREZ CASTILLO

ASESOR:

DR. ROBERTO MORALES CAPORAL

APIZACO, TLAXCALA SEPTIEMBRE 2018

AGRADECIMIENTOS

A mi familia por el apoyo y amor incondicional en todo momento.

A mi asesor Dr. Roberto Morales Caporal por su confianza en mi persona y por la oportunidad de trabajar con él, así como a todo mi comité tutorial.

Al INAOE (Instituto Nacional de Astrofísica, Óptica y Electrónica) por permitir realizar mis estancias técnicas en sus instalaciones, así como a el Dr. José de Jesús Rangel Magdaleno.

A CONACYT (Consejo Nacional de Ciencia y Tecnología) por la beca otorgada.

DEDICATORIAS

A mi esposa Erandi y mi hija Natalia por ser mi fuente de inspiración y fuerza.

RESUMEN

Este trabajo de tesis presenta el uso de una FPGA con arquitectura Zynq-7000 y Linux embebido, como una herramienta digital para realizar monitoreo remoto de dispositivos conectados a internet, haciendo uso de sensores que admitan la comunicación SPI, la ventaja de la implementación del protocolo SPI en el FPGA es que se puede usar una gran cantidad de sensores que admiten este protocolo incluso uniendo varios sensores al mismo tiempo. La incorporación de un sistema operativo Linux embebido abre las posibilidades de comunicación y ejecución de software como si fuera una computadora personal, en esta tesis, se utiliza un acelerómetro de 3 ejes que servirá para el monitoreo de vibraciones en un motor de inducción para detectar fallas. Integrando los elementos anteriores, se genera una herramienta digital capaz de cubrir la necesidad de monitoreo remoto en tiempo real usando un servidor web y conectando la FPGA a través del puerto Ethernet utilizando el protocolo TCP / IP.

ABSTRACT

This work presents the use of an FPGA with Zynq-7000 architecture and embedded Linux, as a digital tool to perform remote monitoring of internet-connected devices, making use of sensors that support SPI communication, the advantage of the implementation of the SPI protocol in the FPGA, is that you can use a large number of sensors that support this protocol even joining several sensors at the same time. The incorporation of an embedded Linux operating system opens the possibilities of communication and execution of software as if it were a personal computer, in this thesis, a 3-axis accelerometer is used that will serve for the monitoring of vibrations in an induction motor to detect faults. By integrating the above elements, a digital tool capable of covering the need for remote monitoring in real time is generated using a web server and connecting the FPGA through the Ethernet port using the TCP / IP protocol.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

LISTA DE ACRÓNIMOS

ADC	*analog-to-digital converter.*
ANSI	*American National Standards Institute.*
API	*application programming interface.*
ASIC	*application-specific integrated circuit.*
ASP	*active server pages.*
AXI	*advanced extensible interface.*
BSD	*berkeley software distribution.*
CS	*chip select.*
CSS	*cascading style sheets.*
CSV	*comma-separated values.*
DAC	*digital-to-analog converter.*
DMA	*direct memory Access.*
DNS	*domain name system.*
EPROM	*erasable programmable read-only Memory.*
EXT4	*fourth extended filesystem.*
FAT32	*file allocation table 32.*
FIFO	*first in, first out.*
FPGA	*field programmable gate array.*
FTP	*file transfer protocol.*
Gbps	*giga bits per second.*
GNU	*gnu is not unix.*
GPIO	*general purpose input/output.*
GPL	*general public license.*
HDL	*hardware description language.*
HTML	*hypertext markup language.*

HTTP *hypertext transfer protocol.*

IEEE *institute of electrical and electronics engineers.*

IoT *internet of things.*

JTAG *joint test action group.*

LUT *look-up table.*

LVDS *low-voltage differential signaling.*

MEMS *microelectromechanical systems.*

MI *motor de inducción.*

MISO *master input slave output.*

MOSI *master output slave input.*

NCSA *national center for supercomputing applications.*

NFS *network file system.*

OS *operating system.*

PC *personal computer.*

PCB *printed circuit board.*

PCI *peripheral component interconnect.*

PDA *personal digital assistant.*

PHP *hypertext preprocessor.*

PL *programmable_logic.*

PLD *programmable logic device.*

PMOD *peripheral module.*

PS *processing system.*

RAM *random access memory.*

ROM *read-only memory.*

RTOS *real-time operating system.*

SCL *serial clock.*

SDK *software development kit.*

SoC	*system on a chip.*
SPI	*serial peripheral interface.*
SQL	*structured query language.*
SSH	*secure shell.*
TCP/IP	*transmission control protocol / internet protocol.*
UDP	*user datagram protocol.*
VHDL	*very high speed integrated circuit-hardware description language.*

Capítulo 1 INTRODUCCIÓN

1.1. INTRODUCCIÓN

El ser humano se encuentra actualmente en constante movimiento las actividades así lo exigen, por lo tanto, la interacción local y remota con los dispositivos eléctricos y electrónicos es relevante para minimizar el tiempo de respuesta. Además de eso, a través de numerosos avances tecnológicos la sociedad se está moviendo hacia un paradigma "siempre conectado" [1]. Para lograr esta interacción, se requiere una interfaz entre el dispositivo y el usuario que permita el monitoreo y, de acuerdo con las necesidades, obtener valores específicos. Esto es muy importante para la cuarta revolución industrial, conocida como Industria 4.0, trae Sistemas Ciber-físicos e Internet de las cosas en la industria (IIoT) a los sistemas de fabricación industrial [2], [3]. Además, la cantidad de dispositivos físicos interconectados aumentará drásticamente e interactuarán continuamente con los servicios locales en la nube para actuar de forma inteligente y flexible [4].

1.2. PLANTEAMIENTO DEL PROBLEMA

El presente trabajo responder a las necesidades actuales que propone la industria 4.0 que junto con el internet de las cosas se plantea que todos los dispositivos eléctricos/electrónicos deben estar conectados a la nube, con lo cual, al estar todo interconectado, se generan múltiples opciones de control, monitoreo, planes de mantenimiento, supervisión, etc. Actualmente se cuenta con los sistemas micro electromecánicos que son elementos fundamentales debido a que son la conexión entre un sistema de procesamiento en general con el dispositivo a conectar, un ejemplo de los sistemas micro electromecánicos son los sensores de vibración(acelerómetros), de corriente, de presión, etc. El sistema de procesamiento y el sistema micro electromecánico al acoplarlos generan una herramienta digital poderosa que permite la interconexión y monitoreo de dispositivos.

1.3. JUSTIFICACIÓN

En los tiempos modernos la capacidad de conectarse para interactuar con los dispositivos electrónicos vía remota hace más eficiente los tiempos de respuesta ya que en tiempo real podemos estar analizando valores determinados y tomar decisiones de forma más eficaz, el acceso a la tecnología de las tarjetas de desarrollo FPGA que pueden manejar sistemas operativos en tiempo real (RTOS) permite tener una interfaz hecha a la medida y cambiarla solo modificando su programación lógica así como disponer de todos los servicios que proporciona un sistema operativo en general como por ejemplo una comunicación por puerto Ethernet con protocolo TPC/IP y con servicios de SSH server y FTP server.

Por lo tanto, haciendo uso de los elementos mencionados e integrándolos se genera una herramienta digital capaz de realizar el monitoreo por ejemplo de señales de vibración de un motor de inducción por medio de la lectura de un sensor MEMS acelerómetro de 3 ejes que nos servirá para detectar fallas de manera oportuna al detectar ciertos patrones indicadores.

1.4. OBJETIVOS

1.4.1. OBJETIVO GENERAL

Monitorear de forma local y remota en tiempo real las señales provenientes de sensores que utilicen el protocolo de comunicación SPI por Ethernet haciendo uso de una tarjeta de

desarrollo FPGA (Zynq-7000 Xilinx, “Zedboard”) con sistema operativo Linux embebido por Bus SPI y Servidor web.

1.4.2. OBJETIVOS ESPECÍFICOS

- Revisar literatura de tecnología FPGA y analizar la tarjeta de desarrollo Xilinx Zynq-7000 modelo “Zedboard”.
- Instalación y puesta en marcha de una estación de trabajo con una distribución de Linux con sus respectivos controladores para la comunicación con la FPGA, así como la instalación del SDK de Xilinx y Vivado.
- Crear un diseño de hardware en Vivado.
- Generar sistema embebido Linux para ejecutarse sobre la FPGA.
- Anexar protocolo SPI al diseño de hardware.
- Estudio del sensor acelerómetro de 3 ejes ADXL345 de Analog Devices.
- Desarrollar aplicación en lenguaje de alto nivel para la comunicación del sensor ADXL345 con el puerto PMOD de la FPGA.
- Instalar Servidor Web para atender las peticiones remotas.
- Desarrollar portal Web y aplicaciones secundarias que ejecuten aplicaciones requeridas por el usuario.

1.5. METODOLOGÍA

La metodología propuesta para el monitoreo de señales de corriente y vibración de un motor de inducción mediante el FPGA Xilinx Zynq con sistema operativo Linux usando bus SPI es la conformada por los siguientes pasos:

- Analizar la arquitectura de la FPGA.
- Instalar el software y hardware requerido.
- Diseñar por medio de VHDL Vivado el hardware compatible necesario para el uso de un core de Linux.
- Ejecutar la Síntesis, implementación y generar el bitstream para así exportar el diseño del hardware al SDK de Xilinx.

- Programar Software (C, Python, Shell script, etc.) para leer y escribir en los diferentes dispositivos tanto del FPGA como de los sensores.
- Instalar y configurar un servidor web que realice transacciones con los valores obtenidos.
- Realizar las pruebas correspondientes.

1.6. ALCANCES Y LIMITACIONES

Con el presente trabajo de tesis se generará una herramienta digital que será capaz de controlar dispositivos que acepten el protocolo de comunicación SPI además podrá ejecutar software en la misma tarjeta FPGA, las limitaciones son que no se podrá procesar las señales adquiridas de los dispositivos ya que los algoritmos necesarios implican un desarrollo específico para cada aplicación la cual forma parte de los trabajos futuros.

1.7. ESTADO DE LA TÉCNICA

En trabajos anteriores [5] [6] [7] [8], se implementó la solución de monitoreo remoto. Sin embargo, se ha llevado a cabo utilizando microcontroladores e incluso FPGA, pero sin la arquitectura capaz de instalar un sistema operativo o sin la incorporación de comunicación Serial Peripheral Interface Bus (SPI). SPI es un estándar de enlace de datos en serie síncrono nombrado por Motorola que opera en modo dúplex completo. A veces SPI se llama un bus serie de "cuatro cables". Hay un maestro y uno o más dispositivos esclavos en la comunicación [9]. Además, la ventaja de este bus en comparación con la comunicación I2C es que ofrece una mayor velocidad de transmisión.

Trabajos previos han implementado el protocolo de comunicación SPI y se han hecho usando microcontroladores como LM358962 [10], ATmega32 [11], ESP32 [12], y también usando FPGA por ejemplo un Spartan 2 [13], por parte de los microcontroladores son de arquitectura fija y no se puede modificar el hardware y en los casos de FPGA sin arquitectura SoC se limitan a un diseño Bare Metal, la contribución de este trabajo es que al usar el chip Zynq \ textregistered-7000 se ofrece la posibilidad de crear un propio diseño de hardware e instalación de un sistema operativo integrado (SO) y con lo anterior crear un entorno similar al de una computadora personal, tener un sistema operativo puede crear y ejecutar programas creados en lenguajes de programación de alto o bajo nivel. Por lo tanto, este trabajo propone el uso de una placa de desarrollo Xilinx FPGA Zedboard con arquitectura Zynq-7000, todos

los SoC programables [14] (ver Fig.1), esta arquitectura incluye un Sistema de Procesamiento (PS) y una Lógica Programable (PL) que incorpora un ARM Cortex-A9 de doble núcleo a 866 Mhz y un FPGA Artix-7 respectivamente, lo que permite instalar un sistema operativo (SO) y en la parte lógica el co-diseño de un hardware específico para la interfaz requerida. En este caso, se trata de una comunicación SPI de 4 hilos para recibir datos del acelerómetro triaxial modelo ADXL345 [15]. La aparición de procesadores y FPGA de bajo costo y eficiencia energética ha hecho posible la realización de estas aplicaciones a través del diseño conjunto de hardware / software [16].

Año de Publicación	Autor	Articulo/Tesis	Hardware	Software
2017	Manuel Quiñones-Cuenca, Victor Gonzalez-Jaramillo, Rommel Torres, Miguel Jumbo	Monitoring System of Environmental Variables Using a Wireless Sensor Network and Platforms of Internet of Things	Microcontrolador Atmega328	Arduino SDK
2017	Jude Adekunle Adeleke	Integrating Statistical Machine Learning in a Semantic Sensor Web for Proactive Monitoring and Control	Raspberry Pi	Phyton
2017	Alamsyah, .; Amir, Ardi; Subito, Mery	Electronic Device Web-Based Monitoring System Using Atmega8535 Microcontroller	Microcontrolador Atmega3585	PHP,C++
2017	ArdhenduSaha[a]SampaDas[a]SureshM.[b] Raj KiranV.[c]NaiwritaDey[d]	FPGA based self-vibration compensated two dimensional non-contact vibration measurement using 2D position sensitive detector with remote monitoring	cRIO-9074 FPGA	Software de National Instruments
2017	Widianto, Eko Didik; Waskitaningrum, Khoirunisa; Isnanto, Rizal	A Motorcycle Monitor and Control System for Teenager Riders	Arduino Mega 2560	Arduino SDK
2017	Vivekanand Jha, Nupur Prakash, Sweta Sagar	Wearable anger-monitoring system	Arduino UNO	Arduino SDK
2017	BillelBengherbia[ab]MohamedOuld ZmirliAbdelmoghniToubal[ab]Abderrezak Guessoum[b]	FPGA-based wireless sensor nodes for vibration monitoring system and fault diagnosis	Xilinx Artix-7 XC7A35T FPGA	Vivado VHDL

Tabla 1 Estado del arte.

1.8. PREGUNTA DE INVESTIGACIÓN

¿Es posible monitorear local, así como remotamente un dispositivo electrónico usando tecnología de FPGA con sistema operativo incluido?

1.9. CONTENIDO DE LA TESIS

Capítulo 1.- Introducción.

Se describe el problema a tratar, así como los objetivos, la justificación, la metodología, la pregunta de investigación y el contenido de la tesis.

Capítulo 2.- La Herramienta Digital

Se realiza la investigación sobre FPGA, tipos de arquitectura en FPGA, Sistemas Operativos que se pueden incorporar a los FPGA, sensores y sus protocolos de comunicación, Integración de la Herramienta digital FPGA con OS y sensor propiamente comunicado.

Capítulo 3.- Aplicación de la Herramienta Digital a la detección de fallas en motores.

Se utiliza la herramienta digital previamente integrada a la aplicación de detección de fallas en motores de inducción.

Capítulo 4.-Resultados en la aplicación y conclusiones.

Se obtienen y analizan los resultados obtenidos en la aplicación de la herramienta digital creada y conclusiones.

Capítulo 2 HERRAMIENTA DIGITAL

Para construir la herramienta digital que se utilizara, se requiere contemplar los conceptos que se utilizaron para comprender como integrar la herramienta digital así como el porqué de la elección de los componentes, se comenzará por los conceptos de FPGA, los lenguajes de descripción de hardware, la arquitectura Zynq, Zynq-7000, sistemas embebidos o empotrados, Linux, Xillinux, PMOD, SPI, Acelerómetro ADXL345, lenguajes de alto nivel, servidor web, portal web, código PHP incrustado en HTML así como el protocolo de comunicación TCP/IP.

2.1. FPGA

La matriz de puertas programables (del inglés *Field programable gate array*) es un dispositivo lógico que contiene una matriz bidimensional de células lógicas genéricas y conmutadores programables. La estructura conceptual de un dispositivo FPGA se muestra en la Figura 1. Una célula lógica puede configurarse (es decir, programarse) para realizar una función simple, y un conmutador programable puede personalizarse para proporcionar interconexiones entre las células lógicas. Se puede implementar un diseño personalizado especificando la función de cada celda lógica y configurando selectivamente la conexión de cada interruptor programable. Una vez que se completa el diseño y la síntesis, podemos usar un cable adaptador simple para descargar la celda lógica deseada y cambiar la configuración al dispositivo FPGA y obtener el circuito personalizado. Dado que este proceso se puede realizar "en el campo" en lugar de "en una instalación de fabricación", el dispositivo se conoce como campo programable [24].

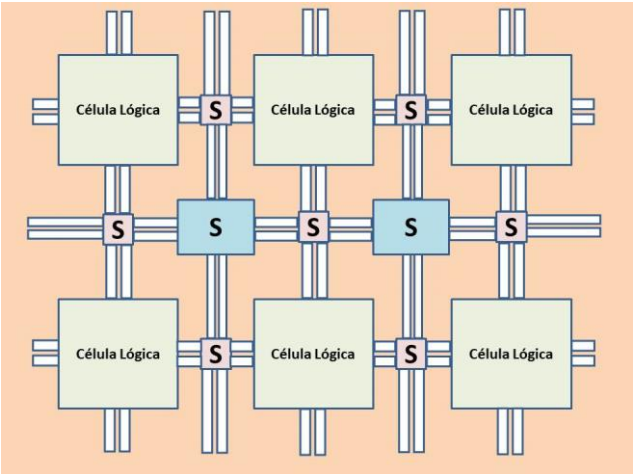
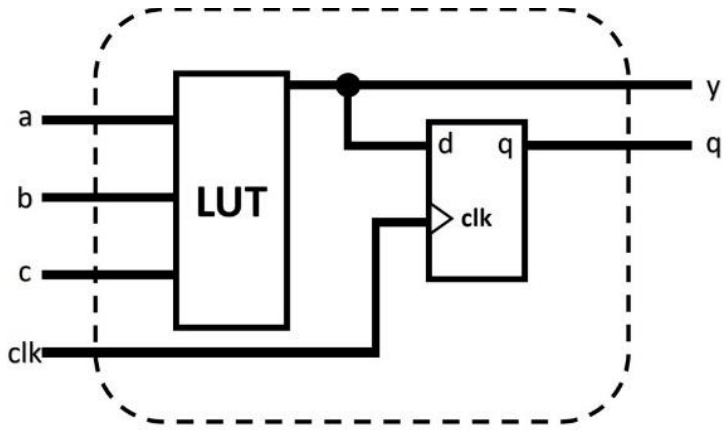


Figura 2-1Concepto de la estructura de una FPGA.



a	b	c	y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Figura 2-2 Diagrama conceptual y tabla ejemplo.

Celda lógica basada en LUT: Una celda lógica generalmente contiene un pequeño circuito combinacional configurable con un flip-flop de tipo D (D FF). El método más común para implementar un circuito combinacional configurable es una tabla de búsqueda (LUT). Un LUT de entrada n puede considerarse como una pequeña memoria de 2 "por 1. Al escribir correctamente el contenido de la memoria, podemos usar el LUT para implementar cualquier función combinacional de entrada n. El diagrama conceptual de una lógica basada en LUT de tres entradas La Figura 2.2 (a) muestra una celda. En la Figura 2.2 (b) se muestra un ejemplo de implementación de LUT de tres entradas de un @ bc. Tenga en cuenta que la salida de la LUT puede usarse directamente o almacenarse en el D FF. este último se puede usar para implementar circuitos secuenciales [24].

Macro Cell La mayoría de los dispositivos FPGA también incorporan ciertas macro células o macro bloques. Estos están diseñados y fabricados a nivel de transistor, y sus funcionalidades complementan las celdas lógicas generales. Las macro células comúnmente utilizadas incluyen bloques de memoria, multiplicadores combinatorios, circuitos de gestión de reloj y circuitos de interfaz de E / S. Los dispositivos FPGA avanzados incluso pueden contener uno o más núcleos de procesador prefabricados [24].

En el año 1984 la compañía Xilinx, Inc. Introdujo el dispositivo FPGA como un dispositivo lógico avanzado programable. Hoy en día forma parte de un mercado

multimillonario el cual es utilizado en diversos productos como cámaras digitales, en automóviles, interruptores de red, etc. Las características mencionadas más otras como múltiples procesadores, grandes cantidades de memoria, cientos de multiplicadores, entradas y salidas de alta velocidad en conjunto hacen que sea más atractivo el uso de los FPGA. Las principales compañías que fabrican FPGA son 2 (ver Fig. 2.3), Xilinx y Altera la cual que fue adquirida por la compañía Intel en 2015 por 16700 millones de dólares.

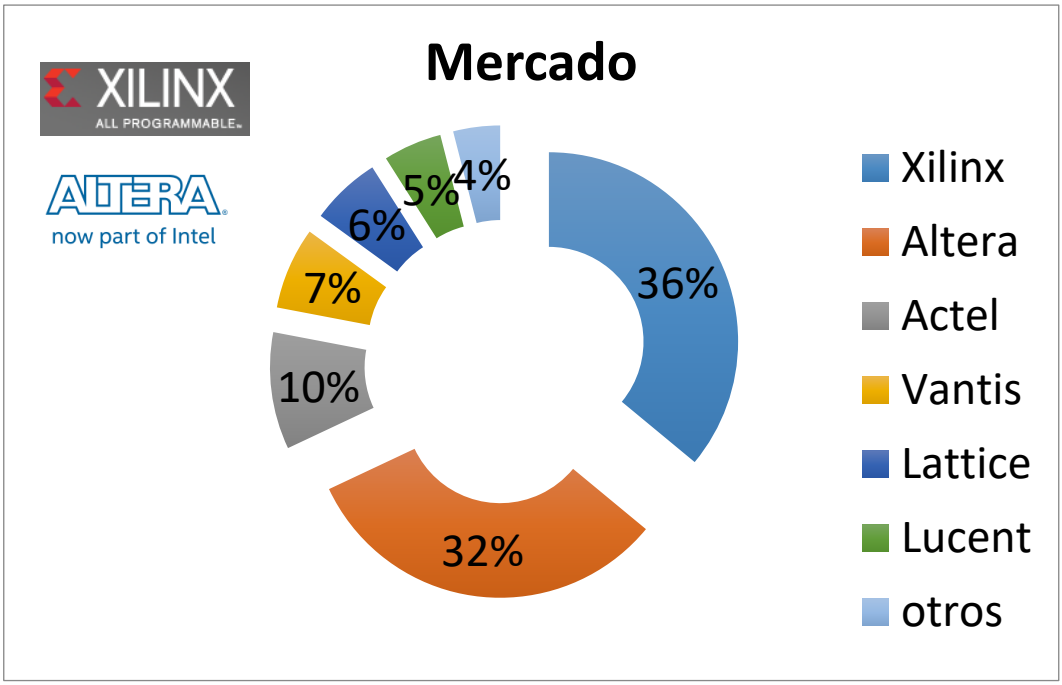


Figura 2-3 Principales marcas de FPGA.

2.2. Lenguajes de Descripción de Hardware

Para utilizar el dispositivo FPGA se requiere de un lenguaje de descripción de hardware (HDL) el cual es un lenguaje de alto nivel para describir el circuito a ser implementado en el FPGA. El origen de la descripción del hardware inicia por la necesidad de documentar el comportamiento del hardware, el proceso de traducir una fuente HDL a una forma adecuado para un procesador de uso general para imitar el hardware descrito se llama simulación. La simulación ha demostrado ser una herramienta extremadamente útil para desarrollar hardware y verificar antes de fabricar físicamente el hardware. Los lenguajes de descripción de hardware son VHDL y Verilog. Se puede decir que son los más utilizados para el diseño FPGA y depende del usuario cual el más adecuado para sus necesidades.

2.2.1. VHDL

VHDL es un lenguaje de especificación definido por el IEEE (Institute of Electrical and Electronics Engineers) (ANSI/IEEE 1076-1993) utilizado para describir circuitos digitales y para la automatización de diseño electrónico. VHDL es acrónimo proveniente de la combinación de dos acrónimos: VHSIC (Very High Speed Integrated Circuit) y HDL (Hardware Description Language). Aunque puede ser usado de forma general para describir cualquier circuito digital se usa principalmente para programar PLD (Programable Logic Device - Dispositivo Lógico Programable), FPGA (Field Programmable Gate Array), ASIC y similares.

2.2.2. VERILOG

Verilog es un lenguaje de descripción de hardware (HDL, del Inglés Hardware Description Language) usado para modelar sistemas electrónicos. El lenguaje, algunas veces llamado Verilog HDL, soporta el diseño, prueba e implementación de circuitos analógicos, digitales y de señal mixta a diferentes niveles de abstracción.

2.3. ARQUITECTURA ZYNQ

Los dispositivos Zynq están diseñados para ser flexibles y formar una plataforma atractiva para una amplia variedad de aplicaciones, así como el zinc metálico se puede mezclar con otros metales para formar aleaciones con diferentes propiedades deseables [25].

La característica definitoria de Zynq es que combina un procesador ARM Cortex-A9 de doble núcleo con el tejido lógico de Arreglo de puerta programable de campo (FPGA) tradicional. Aunque los procesadores dedicados se han acoplado con FPGA antes, nunca ha sido la misma proposición. En Zynq, el ARM Cortex-A9 es un procesador de grado de aplicación, capaz de ejecutar sistemas operativos completos como Linux, mientras que la lógica programable se basa en la arquitectura FPGA de la serie 7 de Xilinx. La arquitectura se completa con interfaces AXI estándar de la industria, que proporcionan conexiones de alta latencia y ancho de banda alto entre las dos partes del dispositivo. Esto significa que el procesador y la lógica se pueden usar para lo que mejor hacen, sin la sobrecarga de la interfaz entre dos dispositivos físicamente separados. Mientras tanto, los beneficios derivados de la simplificación del sistema a un solo chip incluyen reducciones en el tamaño físico y el costo general [25]. Zynq es más que solo el silicio. Es convincente porque las herramientas de

desarrollo de software, el flujo de diseño y los métodos de integración centrados en estándares se adaptan a los requisitos del diseño del sistema basado en Zynq [26].

Dado que ya hemos descrito a Zynq como System-on-Chip, una primera pregunta obvia sería "¿Qué es un SoC?". El concepto ha existido por un tiempo; la implicación es que un único chip de silicio se puede usar para implementar la funcionalidad de un sistema completo, en lugar de requerir varios chips físicos diferentes. En el pasado, el término SoC usualmente se refería a un Circuito Integrado Específico de Aplicación (ASIC), que puede incluir componentes de frecuencia digital, analógica y de radio, junto con bloques de señal mixta para implementar convertidores analógicos a digital y digital a analógico. (ADCs y DAC). Centrándonos en el aspecto digital, un SoC puede combinar todos los aspectos de un sistema digital: procesamiento, lógica de alta velocidad, interconexión, memoria, etc. Todas estas funciones de otro modo podrían realizarse utilizando dispositivos separados físicamente, y combinarse juntas en un sistema en el nivel de la Placa de Circuito Impreso (PCB). La solución SoC tiene un costo menor, permite transferencias de datos más rápidas y seguras entre los diversos elementos del sistema, tiene una mayor velocidad general del sistema, un menor consumo de energía, un tamaño físico más pequeño y una mayor confiabilidad. De hecho, hay una serie de razones convincentes para preferir SoC sobre sistemas equivalentes de componentes discretos. Para una comparación gráfica simple del sistema-en-una-placa y el sistema-en-chip, considere la figura 2.4.

Las principales desventajas de los SoC basados en ASIC son (i) tiempo y costo de desarrollo, y (ii) falta de flexibilidad. El esfuerzo de ingeniería no recurrente (y el costo) del desarrollo de un ASIC son significativos, lo que hace que este tipo de SoC sea adecuado solo para mercados de gran volumen en los que no se requieren actualizaciones futuras. Ejemplos representativos de SoC basados en ASIC son los procesadores integrados que se encuentran en PC, tabletas y teléfonos inteligentes; estos típicamente comprenden al menos dos núcleos de procesador, memoria, gráficos, interconexión y otras funciones [27], y se fabrican en grandes volúmenes para productos con una vida útil limitada.

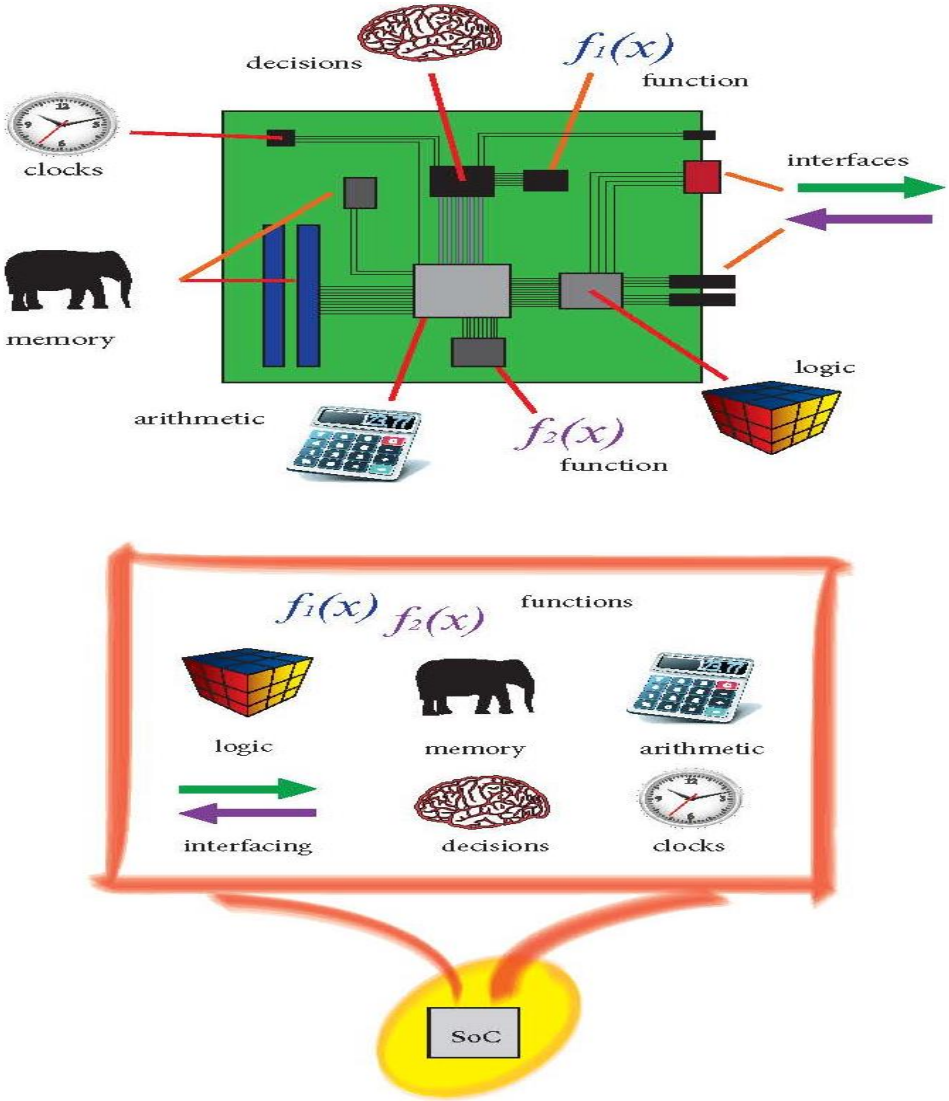


Figura 2-4 Comparativa entre un sistema en una placa (arriba)y el SoC (abajo) [25].

Las limitaciones de los ASC SoC los hacen incompatibles con un número significativo de aplicaciones, particularmente donde el tiempo de comercialización, la flexibilidad y la capacidad de actualización son de vital importancia. También constituyen una solución pobre para mercados de volumen bajo o mediano. Existe una clara necesidad de una solución más flexible, y esto es lo que motiva el Sistema en Programmable-Chip, un sabor específico de SoC implementado en un dispositivo programable y reconfigurable. La solución natural ha sido durante mucho tiempo FPGA. Los FPGA son dispositivos intrínsecamente flexibles que se pueden configurar para implementar cualquier sistema arbitrario, incluidos los procesadores integrados si es necesario. Los FPGA también se pueden reconfigurar tantas veces como se desee, ofreciendo así una plataforma más flexible que los ASIC para la

implementación de SoC. Prácticamente no existe riesgo en la implementación de un FPGA en aplicaciones donde se requieren actualizaciones del sistema. Ahora, Zynq proporciona una plataforma aún más ideal para implementar SoC flexibles: Xilinx comercializa el dispositivo como un 'SoC totalmente programable (AP SoC), que captura perfectamente sus capacidades. Es útil introducir un modelo de alto nivel de su arquitectura (Figura 2.5). Tenga en cuenta que Zynq consta de dos partes principales: un Sistema de Procesamiento (PS) formado alrededor de un procesador ARM Cortex-A9 de doble núcleo, y una Lógica Programable (PL), que es equivalente al de un FPGA. También presenta memoria integrada, una variedad de periféricos e interfaces de comunicaciones de alta velocidad.

La sección PL es ideal para implementar subsistemas lógicos, aritméticos y de flujo de datos de alta velocidad, mientras que el PS admite rutinas de software y / o sistemas operativos, lo que significa que la funcionalidad general de cualquier sistema diseñado puede dividirse adecuadamente entre hardware y software. Los enlaces entre el PL y el PS se realizan usando conexiones de Interfaz ampliable extensible (AXI) estándar de la industria [28].

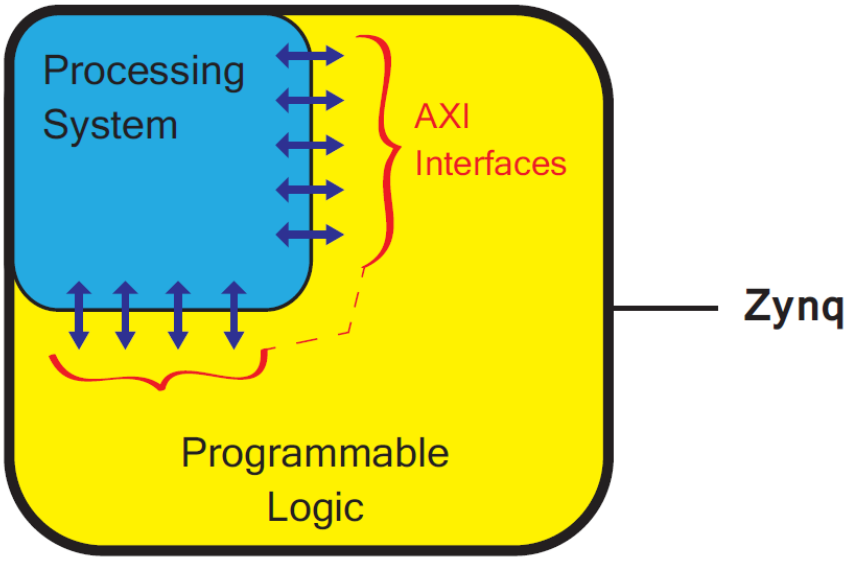


Figura 2-5Modelo simplificado de la arquitectura Zynq [25].

2.3.1. ZYNQ-7000

La gama de productos Zynq comprende seis dispositivos Zynq-7000 de propósito general diferentes, todos con características y tamaños ligeramente diferentes. Las características destacadas de estos se resumen en la Tabla 2 (detalles se pueden encontrar en [29]).

	Z-7010	Z-7015	Z-7020	Z-7030	Z-7045	Z-7100
Processor	Dual core ARM Cortex-A9 with NEON and FPU extensions					
Max. processor clock frequency	866MHz			1GHz		
Programmable Logic	Artix-7			Kintex-7		
No. of FlipFlops	35,200	96,400	106,400	157,200	437,200	554,800
No. of 6-input LUTs	17,600	46,200	53,200	78,600	218,600	277,400
No. of 36Kb Block RAMs	60	95	140	265	545	755
No. of DSP48 slices (18x25 bit)	80	160	220	400	900	2020
No. of SelectIO Input/output Blocks[a]	HR: 100 HP: 0	HR: 150 HP: 0	HR: 200 HP: 0	HR: 100 HP: 150	HR: 212 HP: 150	HR: 250 HP: 150
No. of PCI Express Blocks	-	4	-	4	8	8
No. of serial transceivers	-	4	-	4	8 or 16	16
Serial transceivers maximum rate	-	6.25Gbps	-	6.6Gbps / 12.5Gbps [c]	6.6Gbps / 12.5Gbps [b]	10.3Gbps

Tabla 2 Modelos de la familia Zynq-7000[25].

Como se muestra en la tabla, el principal diferenciador entre los dispositivos específicos dentro de la familia Zynq es el tipo y la cantidad (o 'densidad') de la lógica programable. De los miembros de la familia Zynq de uso general, los dispositivos más pequeños se basan en Xilinx Artix-7 Tejido lógico FPGA y los dispositivos más grandes en el tejido lógico Kintex-7. Cada uno de los seis miembros de la familia proporciona una cantidad diferente de lógica de propósito general, Block RAM, y DSP48E1, y, naturalmente, la capacidad de procesamiento general de la sección PL aumenta en proporción a sus recursos. El PS es estándar en todos los miembros de la familia, la única diferencia es que la frecuencia máxima del núcleo ARM difiere: el PS en el Artix- 7 dispositivos basados pueden sincronizarse a hasta 866GHz, los dispositivos basados en Kintex hasta 1GHz. También hay algunos otros

factores diferenciadores. En particular, los bloques PCI Express y las interfaces de comunicaciones de alta velocidad se integran dentro de la sección PL de dispositivos seleccionados, proporcionando transceptores capaces de operar a velocidades de varios Gbps [30].

Además de los dispositivos Zynq de propósito general, también hay dos Zynq de grado automotriz (basados en Z-7010 y Z-7020) y tres Zynq de grado de defensa (basados en Z-7020, Z-7030 y Z-7045). Tanto los dispositivos de grado automotriz como de defensa tienen un rango de temperatura extendido en comparación con los equivalentes de la corriente principal, mientras que el último también tiene un embalaje robusto y características de seguridad mejoradas. Teniendo en cuenta todos estos factores, hay una amplia variedad de opciones de dispositivo para elegir, asegurando que Zynq sea adecuado para una serie de aplicaciones.

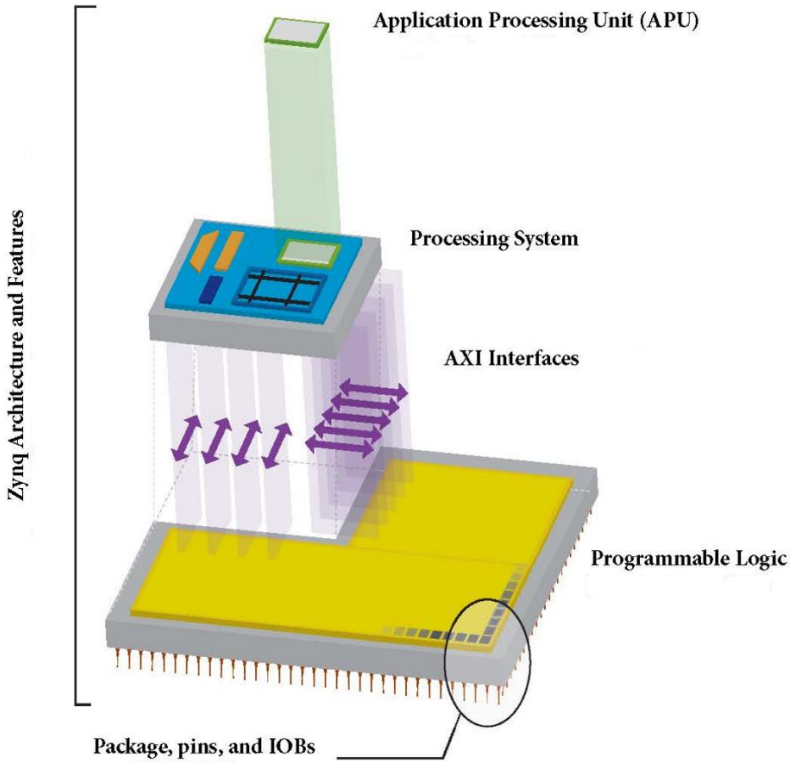


Figura 2-6Arquitectura de Zynq-7000[25].

2.3.1.1. Zedboard

Zedboard es una placa de evaluación y desarrollo basada en Xilinx Zynq TM -7000 Todos los SoC programables (AP SoC). Combinando un Dual Corex-A9 Processing

System (PS) con 85,000 células Serie-7 Programmable Logic (PL), el Zynq-7000 AP SoC se puede orientar para un uso amplio en muchas aplicaciones. La sólida combinación de periféricos incorporados y capacidades de expansión lo convierten en una plataforma ideal tanto para diseñadores principiantes como experimentados. Las características de Zedboard son las siguientes:

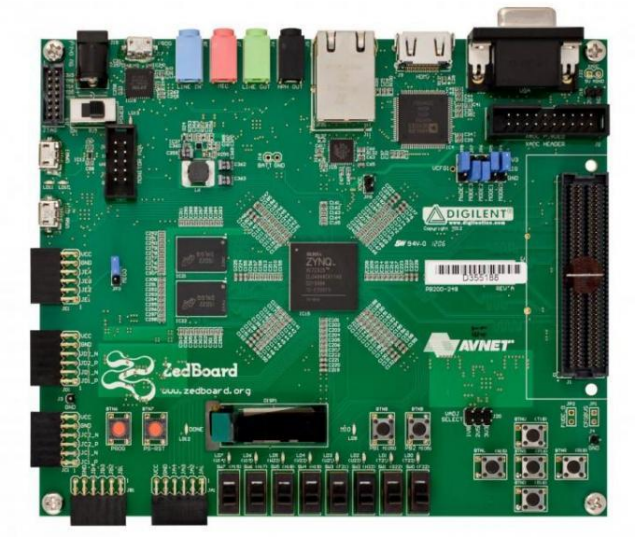


Figura 2-7. Zedboard [33].

• Xilinx ® XC7Z020-1CLG484C Zynq-7000 AP SoC
- O Primary configuration = Q SPI Flash
- O Auxiliary configuration options
  - ♣ Cascaded JTAG
  - ♣ SD Card

• Memory
- O 512 MB DDR3 (128M x 32)
- O 256 Mb QSPI Flash

•Interfaces
- O USB-JTAG Programming using Digilent SMT1-equivalent circuit
  - ♣ Accesses PL JTAG
  - ♣ PS JTAG pins connected through PS Pmod
- O 10/100/1G Ethernet
- o USB OTG 2.0
- o SD Card
- o USB 2.0 FS USB-UART bridge
- o Five Digilent Pmod™ compatible headers (2x6) (1 PS, 4 PL)
- o One LPC FMC
- o One AMS Header
- o Two Reset Buttons (1 PS, 1 PL)
- o Seven Push Buttons (2 PS, 5 PL)

o Eight dip/slide switches (PL)
o Nine User LEDs (1 PS, 8 PL)
o DONE LED (PL)

•On-board Oscillators
O 33.333 MHz (PS)
o 100 MHz (PL)

•Display /Audio
O HDMI Output
O VGA (12-bit Color)
O 128x32 OLED Display
O Audio Line-in, Line-out, headphone, microphone

• Power
O On/Off Switch
O 12V @ 5A AC/DC regulator

•Software
O ISE® WebPACK Design Software
O License voucher for ChipScope™ Pro locked to XC7Z020

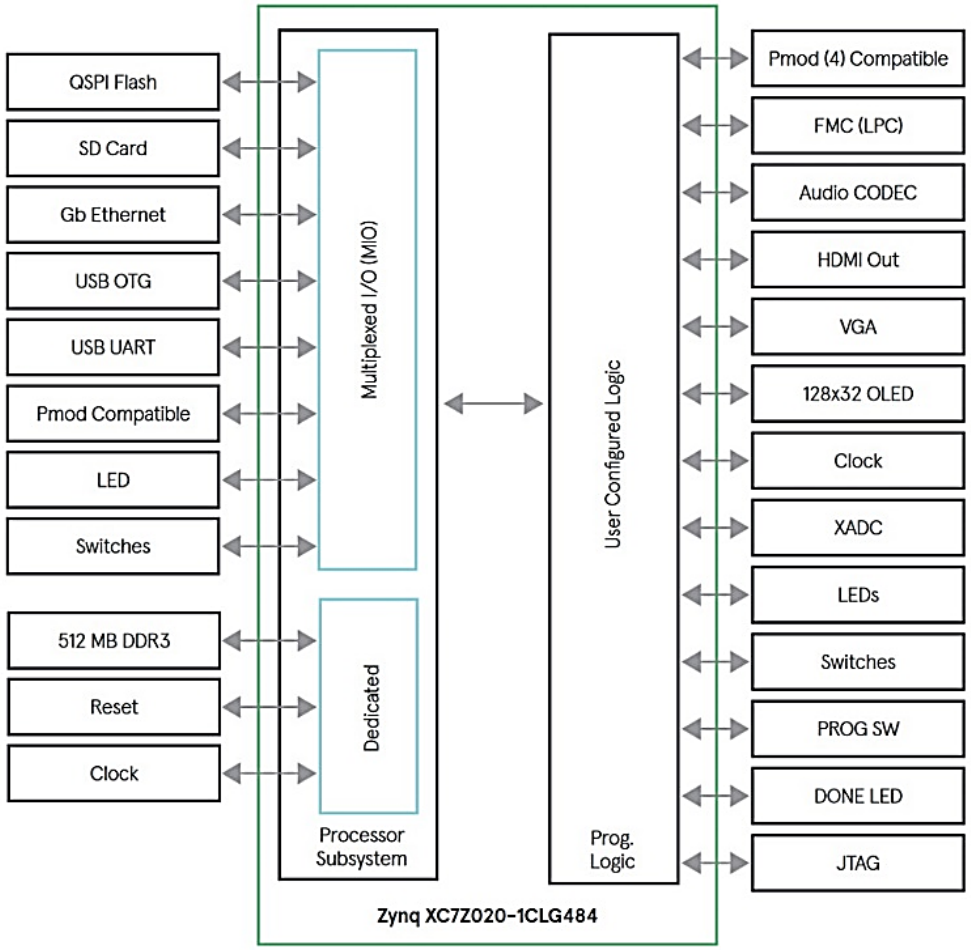


Figura 2-8 Diagrama interno de Zedboard.

2.4. SISTEMAS OPERATIVOS EMBEBIDOS

Existen numerosas definiciones de sistemas embebidos, pero en esencia es un sistema que como su nombre lo indica se encuentra embebido o incrustado dentro del sistema. Una de las principales preguntas a las que se enfrenta un diseñador cuando desarrolla un sistema integrado es si incluye o no un sistema operativo incorporado [25]. Aunque un sistema operativo integrado no es necesario para todas las aplicaciones de sistemas integrados, su uso tiene ventajas, por ejemplo, las siguientes:

- Reducir el tiempo de comercialización.
- Hacer uso de las características existentes
- Reducir los costos de mantenimiento y desarrollo.

2.4.1. REDUCIR EL TIEMPO DE COMERCIALIZACIÓN

Al desarrollar un sistema integrado, hay varias áreas clave en las que un SO incorporado puede reducir el tiempo de desarrollo. Los proveedores de sistemas operativos brindan soporte para una amplia gama de arquitecturas y plataformas; esto tiene la ventaja si llega un punto durante el desarrollo del producto que se considera necesario pasar a una nueva plataforma de procesamiento. Como el software que ya se ha desarrollado está dirigido a un sistema operativo, en lugar de a un dispositivo específico, el proceso de pasar a una nueva arquitectura o dispositivo no debería ser problemático [25].

2.4.2. HACER USO DE LAS CARACTERÍSTICAS EXISTENTES

Los sistemas operativos integrados ofrecen soporte para muchas características que de otro modo tendrían que ser desarrolladas por el diseñador del sistema. Al incorporar una pantalla a una plataforma integrada, ya sea una resolución baja, un panel LCD interno o una salida de pantalla externa a través de HDMI, el sistema debe ser capaz de soportarlo. El grado de soporte que es necesario depende de la aplicación específica, pero generalmente se puede dividir en dos niveles: compatibilidad con el controlador y compatibilidad con la interfaz gráfica. Al elegir utilizar un sistema operativo establecido, generalmente tendrá acceso tanto a nivel de controlador como a nivel de interfaz [25].

2.4.3. REDUCIR LOS COSTOS DE MANTENIMIENTO Y DESARROLLO

Al utilizar un sistema operativo integrado, se reduce la cantidad de códigos personalizados que se deben desarrollar y probar. En realidad, esto equivale a muchos menos errores potenciales que se introducen en el software, lo que a su vez reduce el tiempo y el costo de probar el sistema para encontrarlos y eliminarlos. Un SO proporcionará una plataforma estable, que ha sido ampliamente probada, para que pueda concentrarse en desarrollar las características personalizadas de su sistema y no depurar el código de bajo nivel. Al hacer uso de la suite de herramientas proporcionada por el proveedor del sistema operativo, tendrá acceso a las herramientas de creación de perfiles y eliminación de fallas que acelerará el desarrollo y ayudará a encontrar los cuellos de botella de rendimiento. El conjunto de herramientas también será esencial si se desarrolla software para sistemas de procesamiento de subprocesos múltiples [25].

2.4.4. LINUX

GNU/Linux, es un sistema operativo libre tipo Unix; multiplataforma, multi-usuario y multi-tarea. El sistema es la combinación de varios proyectos, entre los cuales destacan GNU (encabezado por Richard Stallman y la Free Software Foundation) y el núcleo Linux (encabezado por Linus Torvalds). Su desarrollo es uno de los ejemplos más prominentes de software libre: todo su código fuente puede ser utilizado, modificado y redistribuido libremente por cualquiera, bajo los términos de la GPL (Licencia Pública General de GNU) y otra serie de licencias libres.

Linux embebido o **empotrado** (*Embedded Linux*) se refiere al uso del núcleo Linux en un sistema embebido, como por ejemplo PDA, teléfonos móviles, robots, enrutadores, servidores, dispositivos electrónicos y aplicaciones industriales con microcontroladores y microprocesadores.

2.4.5. XILLYBUS

Xillybus se diseñó con el entendimiento de que el manejo del flujo de datos es donde la mayoría de los ingenieros de FPGA tienen dificultades, y también es una razón común para los errores. Las fluctuaciones en el suministro y la demanda de datos de aplicación tienden a generar estados raramente alcanzados en la lógica, a menudo revelando errores ocultos que son extremadamente difíciles de abordar. En consecuencia, Xillybus no solo suministra un

envoltorio para el transporte subyacente (por ejemplo, un motor DMA PCIe), sino que ofrece varios conductos de flujo extremo a extremo para el transporte de datos de la aplicación. Esta es una solución de "una vez y para todos", que ha sido sometida a fuertes pruebas de estrés en numerosas plataformas FPGA y configuraciones de núcleo IP. La robustez y la confiabilidad no se pueden especificar en una hoja de datos, pero se pueden contar fácilmente durante la prueba. Afortunadamente, la integración de Xillybus con una aplicación de destino para pruebas de estrés de la vida real es una tarea relativamente simple, por lo que evaluar su valía es una asignación inmediata y de bajo riesgo [31].

2.4.5.1. Funcionalidad del núcleo IP

Considere el siguiente diagrama de bloques simplificado, que muestra la aplicación de un flujo de datos en cada dirección (se pueden conectar muchos más).

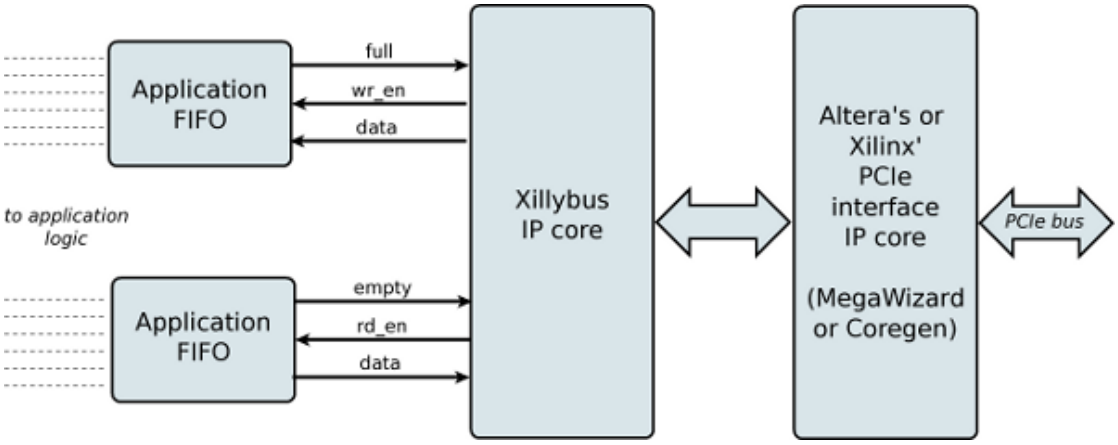


Figura 2-9 Diagrama a bloques de flujo de datos de Xillybus [31].

Como se muestra en la figura 2.9, el núcleo Xillybus IP (el bloque en el medio) comunica datos con la lógica del usuario a través de un FIFO estándar ("FIFO de aplicación" más arriba), que es suministrado por el usuario del núcleo IP. Esto le da al diseñador de FPGA la libertad de decidir la profundidad de FIFO y su interfaz con la lógica de la aplicación [31].

Esta configuración alivia completamente al diseñador de FPGA de administrar el tráfico de datos con el host. Más bien, el núcleo de Xillybus comprueba las señales FIFO "vacía" y "completa" de forma circular e inicia las transferencias de datos hasta que las FIFO se vacíen o llenen nuevamente (dependiendo de la dirección de la secuencia). En su otro extremo, el núcleo Xillybus IP está conectado al núcleo PCIe suministrado por Xilinx o Altera, como se

ve arriba. Para mantener las cosas simples, el núcleo de Xillybus IP no tiene conocimiento sobre la velocidad de datos esperada, y cuándo la lógica del usuario va a suministrarla o buscarla. Tampoco sabe sobre el estado de FIFO, espera cuando está vacío o lleno. Esto hace que la comunicación sea algo ineficaz cuando la velocidad de datos es baja, ya que se envían pequeños fragmentos a través del bus tan pronto como alcanzan el FIFO, en lugar de esperar a que llegue un fragmento de información específico de la aplicación. La importancia de esta ineficiencia es bastante mínima, porque no se aplica cuando la velocidad de datos aumenta [31].

Así que, en general, Xillybus es eficiente cuando debe serlo, y desperdicia recursos de transporte de bus cuando no se utilizan de todos modos. A pesar del deseo típico del ingeniero de FPGA de tomar el control de las transacciones de DMA y utilizar los recursos de manera económica, hay poca o ninguna ventaja al hacerlo. Por otro lado, el diseño del tráfico DMA de acuerdo con una aplicación específica hace que el diseño general sea rígido para modificaciones en el futuro, que no es el caso para un drenaje o fuente FIFO simple.

2.4.5.2. Funcionalidad del controlador anfitrión

El controlador de host genera archivos de dispositivo que se comportan como conductos con nombre: se abren, se leen y se escriben como cualquier otro archivo, pero se comportan de manera similar a las canalizaciones entre procesos o flujos TCP / IP. Para el programa que se ejecuta en el host, la diferencia es que el otro lado de la transmisión no es otro proceso (a través de la red o en la misma computadora), sino un FIFO en la FPGA. Al igual que una transmisión TCP / IP, la transmisión Xillybus está diseñada para funcionar bien con transferencias de datos de alta velocidad y bytes únicos que llegan o se envían ocasionalmente.

Un controlador binario admite cualquier configuración de núcleo Xillybus IP: el controlador detecta automáticamente las secuencias y sus atributos a medida que se cargan en el sistema operativo del host, y los archivos del dispositivo se crean en consecuencia. En máquinas con Windows, se accede a estos archivos de dispositivo como \\. \ Xillybus_something. En Linux, aparecen como / dev / xillybus_something. También al cargar el controlador, los buffers DMA se asignan en el espacio de memoria del host, y el FPGA recibe información sobre sus direcciones. La cantidad de almacenamientos intermedios DMA y su tamaño son parámetros separados para cada flujo. Estos parámetros

están codificados en el núcleo IP de FPGA para una configuración dada, y son recuperados por el host durante el proceso de descubrimiento. Un protocolo de handshake entre el FPGA y el host hace una ilusión de un flujo continuo de datos. Detrás de escena, los buffers de DMA se llenan, se entregan al otro lado y se reconocen. Se utilizan técnicas similares a las utilizadas para la transmisión TCP / IP para garantizar una utilización eficiente de los búferes DMA, a la vez que se mantiene la capacidad de respuesta para pequeños fragmentos de datos [31].

Xillybus ofrece tanto al ingeniero de FPGA como a los programadores de aplicaciones host una interfaz de abstracción con la que probablemente se sienta más cómodo. Si bien en ocasiones se desperdicia cierto ancho de banda PCIe, la eficiencia aumenta naturalmente con la demanda de ancho de banda, lo que hace que estas ocasiones de ancho de banda desperdiciado sean prácticamente imperceptibles. Para aquellos interesados en cómo se implementa Xillybus internamente, se puede encontrar una breve explicación en el Apéndice A de las guías de programación para Linux o Windows [31].

2.5. PMOD/GPIO

Este es un tipo de interfaz simple para agregar pequeños módulos periféricos (de ahí el nombre Pmod = módulo periférico), utilizando interfaces de 6 o 12 pines. El nombre fue estandarizado por, y es una marca registrada de, Digilent Inc., pero otras compañías como Maxim Integrated también producen Pmod [32]. Los Pmod típicos son sensores, actuadores, convertidores de datos y dispositivos de E / S del usuario. También hay algunos transceptores de comunicaciones disponibles. Los Pmod también pueden facilitar las conexiones directas de cables.

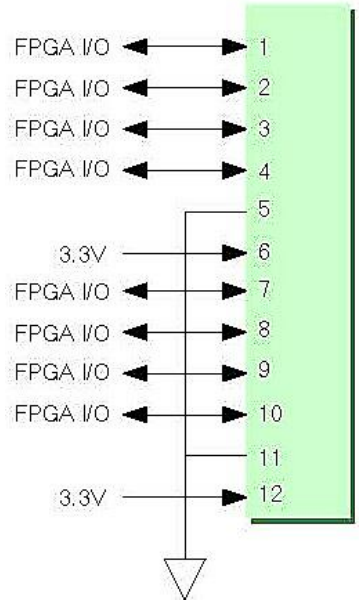


Figura 2-10 Conexiones del PMOD[33].

2.5.1. PMOD CON LVDS

La tarjeta FPGA Z7020 "Zedboard" incluye 5 PMODS, solo 2 de ellos son de tipo LVDS, en la Tabla 3 se muestran los nombres de señal, así como los pines correspondientes en Zynq de los PMOD "C" y "D" respectivamente, en este trabajo de tesis se hace uso de uno de ellos, la elección es indistinta (ver Fig. 2.10).

Pmod	Signal Name	Zynq EPP pin	Pmod	Signal Name	Zynq EPP pin
JC1 Differential	JC1_N	AB6	JD1 Differential	JD1_N	W7
	JC1_P	AB7		JD1_P	V7
	JC2_N	AA4		JD2_N	V4
	JC2_P	Y4		JD2_P	V5
	JC3_N	T6		JD3_N	W5
	JC3_P	R6		JD3_P	W6
	JC4_N	U4		JD4_N	U5
	JC4_P	T4		JD4_P	U6

Tabla 3 Pmod pin configuration [33].

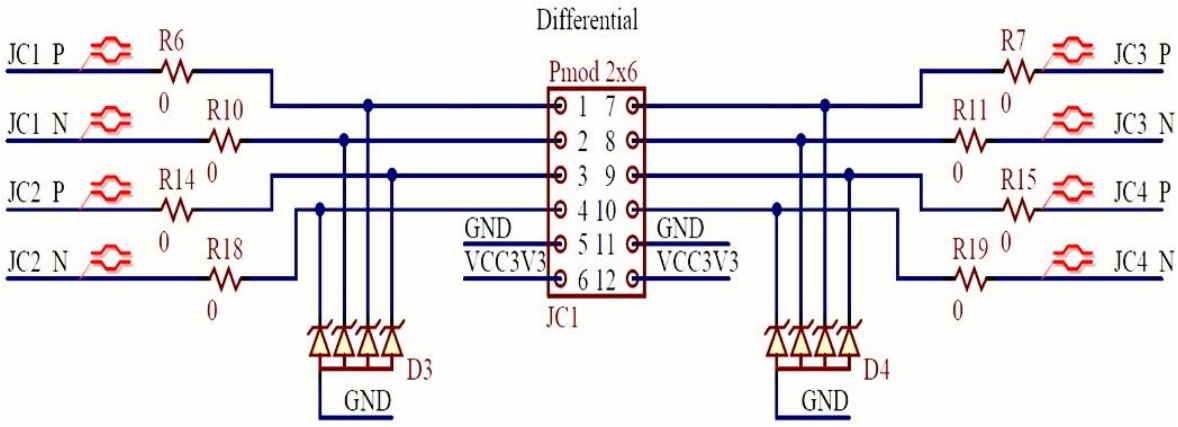


Figura 2-11 Esquema interno de un PMOD con señal LVDS [34].

2.5.1.1. LVDS

TIA / EIA-644, también conocido como LVDS, es un método de señalización utilizado para transmisión de alta velocidad y baja potencia de datos binarios en cobre. Esta señalización técnica utiliza niveles de voltaje de salida más bajos que los estándares diferenciales de 5-V (como TIA / EIA-422) para reducir el consumo de energía, aumentar la velocidad de conmutación, y permita el funcionamiento con un riel de suministro de 3.3 V Los controladores de modo actual de LVDS crea una tensión diferencial (247 mV a 454 mV) en una carga de 100 Ω. El LVDS los receptores detectan señales de hasta ± 100 mV con un ruido de tierra de hasta ± 1-V. TI ofrece receptores LVDS capaces de recuperar datos en un rango de modo común de -4 V a 5 V, que permite hasta 3 V de ruido de tierra. Estos receptores son designados SN65LVDS33 y SN65LVDS34 [35].

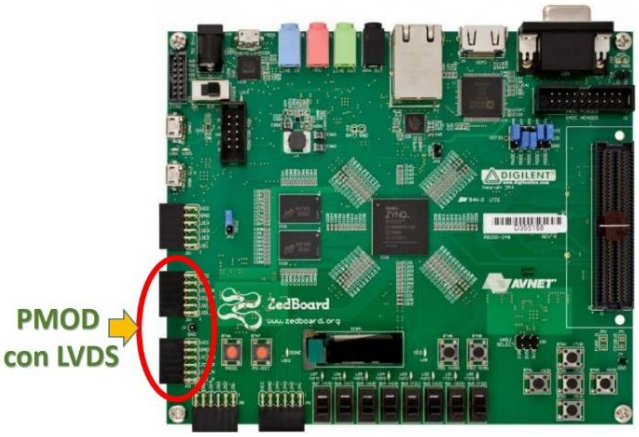


Figura 2-12 PMOD con señal LVDS [33].

El estándar especifica un máximo teórico de 1.923 Gbit / s. La aplicación prevista de esta técnica de señalización es para datos de banda base transmisión sobre medios de impedancia controlada de aproximadamente 100Ω, donde los medios de transmisión pueden ser trazas de placa de circuito impreso (PCB), placas posteriores, o cables La tasa y distancia máxima de transferencia de datos depende de la característica de atenuación de los medios y el acoplamiento de ruido del ambiente. La Figura 2.12 muestra una conexión típica con los controladores y receptores LVDS. Los datos las entradas al controlador cuádruple se reciben en la interfaz de las huellas de PCB del controlador de host. Las entradas de datos consisten en hasta n + 1 bits de información y un reloj de transmisión (Tx) [35].

Las señales de datos y reloj se transmiten de manera diferencial a la interfaz de las salidas del conductor cuádruple, a las huellas de interconexión, y a el conector PCB host. Las señales luego se propagan desde la interfaz del host Conector PCB al conector del cable a los medios de interconexión balanceados. A el enchufe en el otro extremo del cable, las señales pasan a través del enchufe del cable, la interfaz del conector de destino, y luego a las trazas del PCB objetivo. El LVDS la ruta de señal termina en la interfaz de las trazas de PCB objetivo y la terminación circuito. Una interfaz adicional se encuentra en los puntos donde la PCB se remonta a las cuatro entradas de receptor están conectadas. Las salidas de la interfaz del receptor al dirija las trazas de PCB y luego al controlador receptor [35].

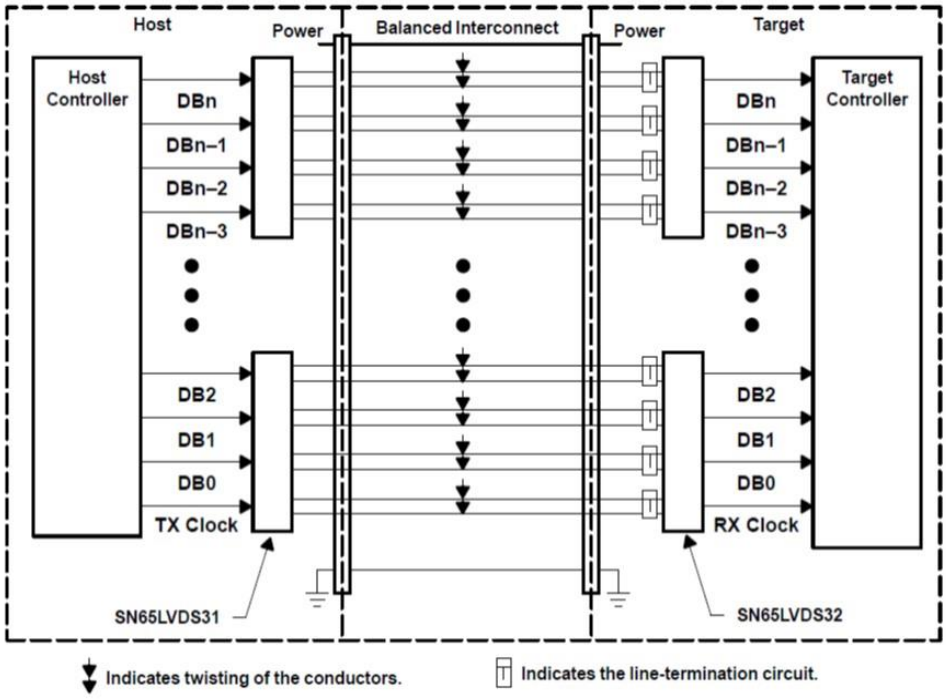


Figura 2-13 Conexión típica con controladores y receptores LVDS [35].

2.6. COMUNICACIÓN POR SPI

El SPI es un puerto de entrada / salida síncrona de alta velocidad que permite un bit de serie flujo de longitud programada (2 a 16 bits) para desplazarse dentro y fuera del dispositivo a una velocidad de transferencia de bits programada. El SPI se usa normalmente para la comunicación entre el dispositivo y los periféricos externos. Las aplicaciones típicas incluyen la interfaz a externa E / S o expansión de periféricos a través de dispositivos como registros de desplazamiento, controladores de pantalla, SPI EPROMS y convertidores analógicos a digital [36].

2.6.1. CARACTERÍSTICAS

El SPI tiene las siguientes características:

- 16-bit shift register
- 16-bit Receive buffer register (SPIBUF) and 16-bit Receive buffer emulation Alias register (SPIEMU)
- 16-bit Transmit data register (SPIDAT0) and 16-bit Transmit data and format selection register (SPIDAT1)

• 8-bit baud clock generator
• Serial clock (SPICLK) I/O pin
• Slave in, master out (SPISIMO) I/O pin
• Slave out, master in (SPISOMI) I/O pin
• Multiple slave chip select (SPISCS[n]) O pins (4 pin mode only)
• Programmable SPI clock frequency range
• Programmable character length (2 to 16 bits)
• Programmable clock phase (delay or no delay)
• Programmable clock polarity (high or low)
• Interrupt capability
• DMA support (read/write synchronization events)
• Up to 66 MHz

Además, SPI permite que el software programe las siguientes opciones:
•SPICLK frequency (SPI module clock/2 through SPI module clock/256)
•3-pin and 4-pin options
•Character length (2 to 16 bits) and shift direction (MSB/LSB first)
•Clock phase (delay or no delay) and polarity (high or low)
•Delay between transmissions in master mode.
•Chip select setup and hold times in master mode
•Chip select hold in master mode

2.6.1.1. Reloj

El reloj SPI (SPICLK) se deriva del reloj del módulo SPI. La frecuencia de bits máxima del reloj admitida es el reloj del módulo SPI / 2, según lo determinado por el campo PRESCALE en el registro de formato de datos SPI n (SPIFMTn). La frecuencia SPICLK se calcula como: Frecuencia SPICLK = [Reloj del módulo SPI] / [SPIFMTn.PRESCALE + 1] Cuando SPIFMTn.PRESCALE se borra a 0, la frecuencia SPICLK se establece por defecto en el reloj del módulo SPI / 2 [36].

2.6.1.2. Descripciones de Señales

La tabla siguiente muestra los pines SPI utilizados para interactuar con dispositivos externos.

Pin	Type	Function
SPISIMO	Output	Serial data output in master mode
SPISOMI	Input	Serial data input in master mode
SPICLK	Output	Serial clock output in master mode
SPISCS	Output	Slave chip select output in master mode

Tabla 4 Pines SPI [36].

2.6.1.3. Modos de Operación

El SPI funciona en modo maestro. El maestro de bus SPI es el dispositivo que controla SPICLK, SPISIMO y, opcionalmente, las señales SPISCS, y por lo tanto inicia SPI transferencias de autobús. Los bits CLKMOD y MASTER en el registro de control global SPI 1 (SPIGCR1) selecciona el modo maestro. En el modo maestro, el SPI admite dos opciones:

• Opción de 3 pines

• Opción de selección de 4 pines con chip

La opción de 3 pines es la interfaz SPI de reloj, datos y salida de datos básica y usa Pines SPICLK, SPISIMO y SPISOMI. La opción de selección de 4 pines con chip agrega PIN SPISCS que se usa para admitir múltiples dispositivos esclavos SPI en un solo bus SPI.

2.7. SENSOR ACELERÓMETRO DE 3 EJES ADXL345

El ADXL345 es un acelerómetro de 3 ejes pequeño, delgado, de potencia ultra-baja con medición de alta resolución (13 bits) de hasta ± 16 g. Los datos de salida digital están formateados como un complemento de dos dígitos (ver Figura 2.15) de 16 bits y son accesibles a través de una interfaz digital SPI (3 o 4 hilos) o I2C. El ADXL345 es ideal para aplicaciones de dispositivos móviles. Mide la aceleración estática de la gravedad en las aplicaciones de detección de inclinación, así como la aceleración dinámica resultante del movimiento o el impacto. Su alta resolución (3.9 mg / LSB) permite la medición de cambios de inclinación menores que 1.0 °. Se proporcionan varias funciones de detección especiales. La actividad y la detección de inactividad detectan la presencia o la falta de movimiento comparando la aceleración en cualquier eje con los umbrales establecidos por el usuario.

Tap Sensing detecta golpes simples y dobles en cualquier dirección. La detección de caída libre detecta si el dispositivo se está cayendo [37].

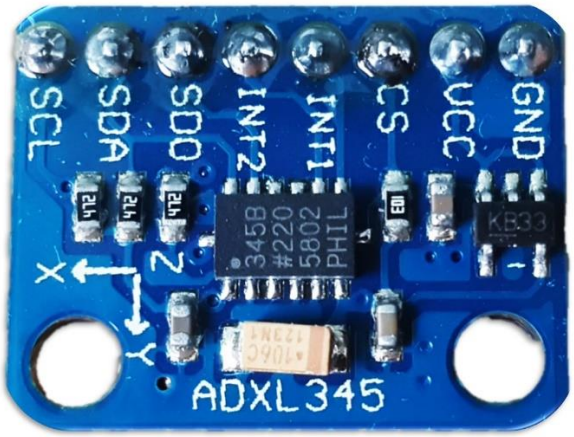


Figura 2-14Sensor acelerómetro 3 ejes ADXL345.

Estas funciones se pueden asignar individualmente a cualquiera de los dos pines de salida de interrupción. Se puede utilizar un sistema de gestión de memoria integrado con un búfer de entrada y salida inicial (FIFO) de 32 niveles para almacenar datos a fin de minimizar la actividad del procesador host y reducir el consumo total de energía del sistema. Los modos de baja potencia permiten una administración de energía inteligente basada en movimiento con detección de umbral y medición de aceleración activa con una disipación de potencia extremadamente baja. El ADXL345 se suministra en un paquete de plástico pequeño, delgado, de 3 mm × 5 mm × 1 mm, 14 conductores [37].

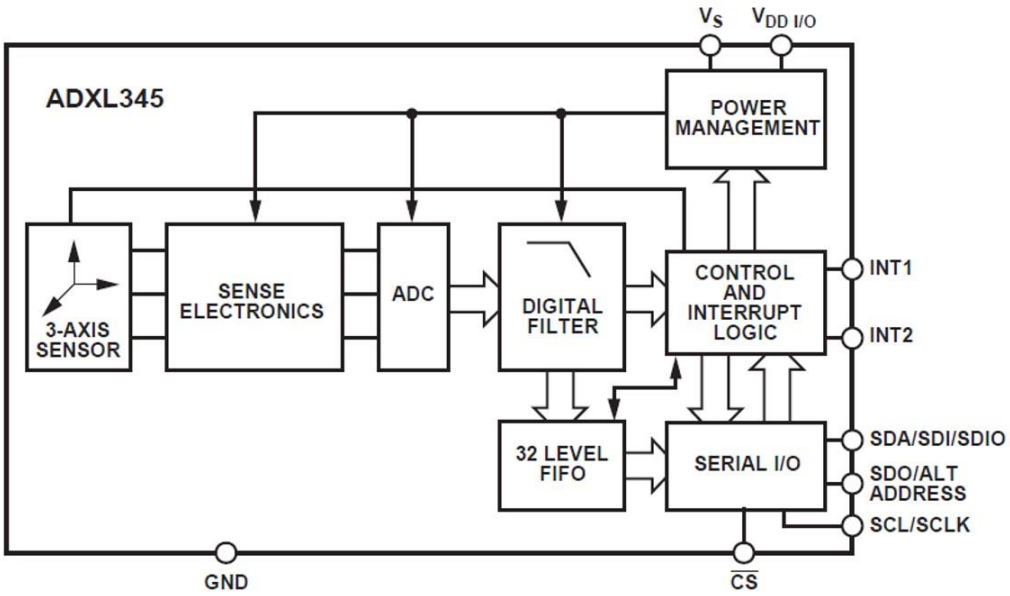


Figura 2-15Diagrama a bloques funcional del sensor ADXL345 [37].

Dirección	Nombre	Características	Tipo
0x00	DEVID	bit [1:0]	R
0x2D	POWER_CTL	bit[3]	R/W
0x31	DATA_FORMAT	bit [1:0]	R/W
0x32	DATAX0	Valor X0	R
0x33	DATAX1	Valor X1	R
0x34	DATAY0	Valor Y0	R
0x35	DATAY1	Valor Y1	R
0x36	DATAZ0	Valor Z0	R
0x37	DATAZ1	Valor Z1	R

Tabla 5 Mapa de Registros.

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
SIGN	SIGN	SIGN	SIGN	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
				DATAX1 DATAY1 DATAZ1							DATAX0 DATAY0 DATAZ0				

Figura 2-16 Construcción de datos del sensor ADXL345 [37].

2.8. LENGUAJES DE ALTO NIVEL

En informática, un lenguaje de programación de alto nivel es un lenguaje de programación con una fuerte abstracción de los detalles de la computadora. A diferencia de los lenguajes de programación de bajo nivel, puede usar elementos de lenguaje natural, ser más fácil de usar o puede automatizar (o incluso ocultar por completo) áreas importantes de sistemas informáticos (por ejemplo, gestión de memoria), simplificando el proceso de desarrollo de un programa. más comprensible que cuando se usa un lenguaje de nivel inferior. La cantidad de abstracción proporcionada define cuán "alto nivel" es un lenguaje de programación.

En la década de 1960, los lenguajes de programación de bajo nivel que utilizaban un compilador se denominaban códigos automáticos. Los ejemplos de códigos automáticos son COBOL y Fortran [38].

El primer lenguaje de programación de alto nivel diseñado para computadoras fue Plankalkül, creado por Konrad Zuse. [39] Sin embargo, no se implementó en su tiempo, y sus contribuciones originales se aislaron en gran parte de otros desarrollos debido a la Segunda Guerra Mundial, aparte de la influencia de la lengua en el lenguaje "Superplan" por Heinz Rutishauser y también en algún grado Algol. El primer lenguaje de alto nivel ampliamente difundido fue Fortran, un desarrollo independiente de la máquina de los primeros sistemas Autocode de IBM. Algol, definido en 1958 y 1960 por comités de científicos informáticos europeos y estadounidenses, introdujo la recursión, así como funciones anidadas bajo el alcance léxico.

"Lenguaje de alto nivel" se refiere al mayor nivel de abstracción del lenguaje de máquina. En lugar de tratar con registros, direcciones de memoria y pilas de llamadas, los lenguajes de alto nivel tratan con variables, matrices, objetos, expresiones aritméticas complejas o booleanas, subrutinas y funciones, bucles, subprocesos, bloqueos y otros conceptos abstractos de informática, con un enfoque en la usabilidad sobre la eficiencia óptima del programa. A diferencia de los lenguajes ensambladores de bajo nivel, los lenguajes de alto nivel tienen pocos elementos de lenguaje, si es que tienen alguno, que se traducen directamente en los códigos de operación nativos de una máquina.

También pueden estar presentes otras características, como las rutinas de manejo de cadenas, las características del lenguaje orientado a objetos y la entrada / salida de archivos.

Una cosa a tener en cuenta acerca de los lenguajes de programación de alto nivel es que estos lenguajes le permiten al programador separarse y separarse de la máquina. Es decir, a diferencia de los lenguajes de bajo nivel, como el ensamblaje o el lenguaje de máquina, la programación de alto nivel puede amplificar las instrucciones del programador y desencadenar una gran cantidad de movimientos de datos en el fondo sin su conocimiento. La responsabilidad y el poder de las instrucciones de ejecución han sido transferidas a la máquina por parte del programador. En este trabajo de Tesis se utilizará el lenguaje de alto nivel "C" debido a su gran portabilidad.

2.9. Programa/Aplicación para conexión y activación de sensor 3 ejes

La aplicación como se ha mencionado, se programó en lenguaje C, debido a su alta portabilidad y facilidad de interpretar. Como se observó en este capítulo, el sensor ADXL345 requiere de algunos parámetros específicos para su operación (ver Tabla 5). Los valores de los registros deben ir dentro de la aplicación principal que en este trabajo de tesis está programada en código C, por lo tanto, en la figura 2.16 se puede observar un fragmento de código que consta de 2 partes, en la primera parte se observa que la primera línea de código manda la instrucción de que las mediciones se efectuarán con el parámetro de "2g" y la segunda línea manda la instrucción de iniciar el sensor y comience a medir. Por otra parte, en la parte 2 las líneas de código contienen las instrucciones para obtener los valores de las variables e imprimirlas en este caso en su valor hexadecimal.

```
54  //part1
55  Write(0x31, 0x00); // 2g
56  Write(0x2d, 0x08); // Start Measure
57
58  //part2
59  xdata0 = SpiRead(0x32); // Read X0
60  printf("%x", xdata0);
61  xdata1 = SpiRead(0x33); // Read X1
62  printf("%x ", xdata1);
63  ydata0 = SpiRead(0x34); // Read Y0
64  printf("%x", ydata0);
65  ydata1 = SpiRead(0x35); // Read Y1
66  printf("%x ", ydata1);
67  zdata0 = SpiRead(0x36); // Read Z0
68  printf("%x", zdata0);
69  zdata1 = SpiRead(0x37); // Read Z1
70  printf("%x\n ", zdata1);
```

Figura 2-17 Código de aplicación valores RAW de X, Y y Z.

Des pues e compilar el código se genera una aplicación la cual se muestra en la Figura 2.17 en la que se puede observar que hace uso de la librería OpenCV [41] que sirve entre otras cosas a generar gráficos, por lo tanto, en la parte izquierda se observa el gráfico generado por OpenCV y del lado derecho los valores que se obtienen del sensor adxl345.

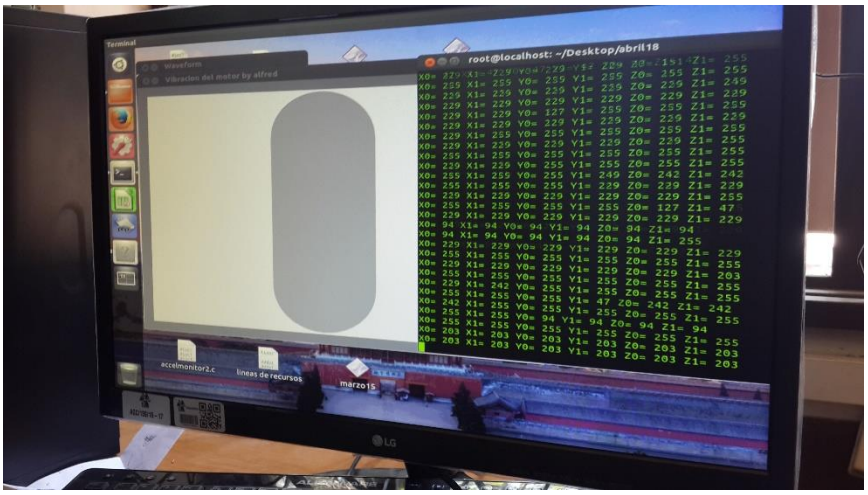


Figura 2-18 Aplicación en funcionamiento.

2.10. SERVIDOR WEB

En este trabajo de tesis el servidor web que se utilizara es Apache (ver instalación ANEXO 6). El Proyecto de Servidor Apache HTTP es un esfuerzo de desarrollo de software colaborativo destinado a crear una implementación de código fuente robusta, comercial, funcional y de libre disponibilidad de un servidor HTTP (Web). El proyecto es administrado conjuntamente por un grupo de voluntarios ubicados en todo el mundo, que utilizan Internet y la Web para comunicar, planificar y desarrollar el servidor y su documentación relacionada. Este proyecto es parte de la Apache Software Foundation. Además, cientos de usuarios han aportado ideas, códigos y documentación al proyecto [40].

En febrero de 1995, el software de servidor más popular en la Web era el daemon HTTP de dominio público desarrollado por Rob McCool en el Centro Nacional de Aplicaciones de Supercomputación, Universidad de Illinois, Urbana-Champaign. Sin embargo, el desarrollo de ese httpd se había estancado después de que Rob abandonó NCSA a mediados de 1994, y muchos webmasters habían desarrollado sus propias extensiones y correcciones de errores que necesitaban una distribución común. Un pequeño grupo de estos webmasters, contactados a través de correo electrónico privado, se reunieron con el fin de coordinar sus

cambios (en forma de "parches"). Brian Behlendorf y Cliff Skolnick crearon una lista de correo, compartieron espacio de información e inicios de sesión para los desarrolladores principales en una máquina en el Área de la Bahía de California, con ancho de banda donado por HotWired.

Utilizando NCSA httpd 1.3 como base, agregamos todas las correcciones de errores publicadas y las mejoras útiles que pudimos encontrar, probamos el resultado en nuestros propios servidores e hicimos el primer lanzamiento público oficial (0.6.2) del servidor Apache en abril de 1995. Por coincidencia, NCSA reinició su propio desarrollo durante el mismo período, y Brandon Long y Beth Frank del Equipo de Desarrollo de Servidores NCSA se unieron a la lista en marzo como miembros honorarios para que los dos proyectos pudieran compartir ideas y soluciones.

El primer servidor Apache fue un gran éxito, pero todos sabíamos que la base de código necesitaba una revisión general y rediseño. Durante mayo-junio de 1995, mientras Rob Hartill y el resto del grupo se centraron en implementar nuevas características para 0.7.x (como procesos secundarios pre-bifurcados) y apoyar a la creciente comunidad de usuarios Apache, Robert Thau diseñó una nueva arquitectura de servidor (código llamado Shambhala) que incluía una estructura modular y API para una mejor extensibilidad, asignación de memoria basada en grupos y un modelo de proceso de pre-bifurcación adaptable. El grupo cambió a esta nueva base de servidores en julio y agregó las características de 0.7.x, lo que resultó en Apache 0.8.8 (y sus hermanos) en agosto [40].

Después de extensas pruebas beta, muchos puertos para ocultar plataformas, un nuevo conjunto de documentación (por David Robinson) y la adición de muchas características en la forma de nuestros módulos estándar, Apache 1.0 fue lanzado el 1 de diciembre de 1995. Menos de un año después de que se formara el grupo, el servidor Apache pasó el httpd de NCSA como el servidor n. ° 1 en Internet y según la encuesta de Netcraft, hoy conserva esa posición. En 1999, los miembros del Grupo Apache formaron la Apache Software Foundation para proporcionar soporte organizacional, legal y financiero para el Servidor Apache HTTP. La fundación ha puesto el software en una base sólida para el desarrollo futuro, y ha ampliado en gran medida la cantidad de proyectos de software de código abierto, que caen bajo el paraguas de esta Fundación.

2.11. PORTAL WEB PÁGINA PRINCIPAL

Para este trabajo de tesis se desarrolló un portal web el cual es un conjunto de páginas web, orientado a interactuar con los usuarios que se conecten al servidor. Las páginas web que conforman al portal fueron desarrolladas en lenguaje de marcado HTML, en conjunto con CCS para lograr una mejor estética del portal. Por otro lado, se requirió introducir código PHP dentro del código HTML para la finalidad de tener la opción de ejecutar aplicaciones desde el navegador web del usuario.

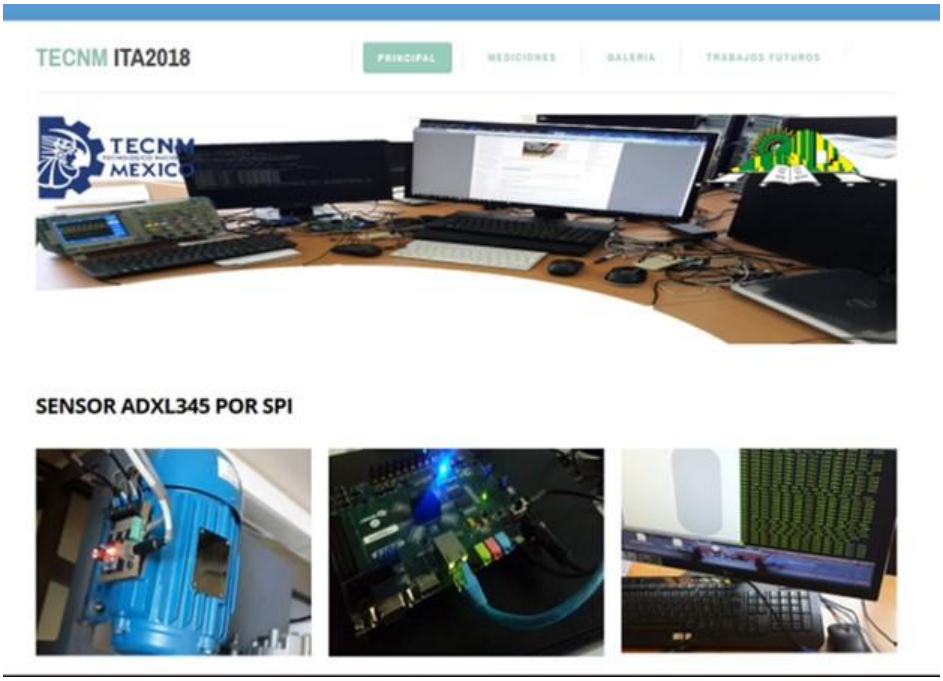


Figura 2-19 Página principal del portal web.

La figura 2.18 muestra la página principal del portal web, la estructura del portal web se compone de 4 secciones principales: PRINCIPAL, MEDICIONES, GALERIA Y TRABAJOS FUTUROS, dentro de mediciones, se encuentran las diferentes opciones para realizar las mediciones, las cuales contienen las diferentes combinaciones de los valores de X, Y o Z en formato decimal o hexadecimal respectivamente, además de un botón para descargar el archivo de la última medición que se haya realizado, el archivo se descargará en formato CSV el cual es un archivo que contiene valores separados por comas (ver figura 2.19).

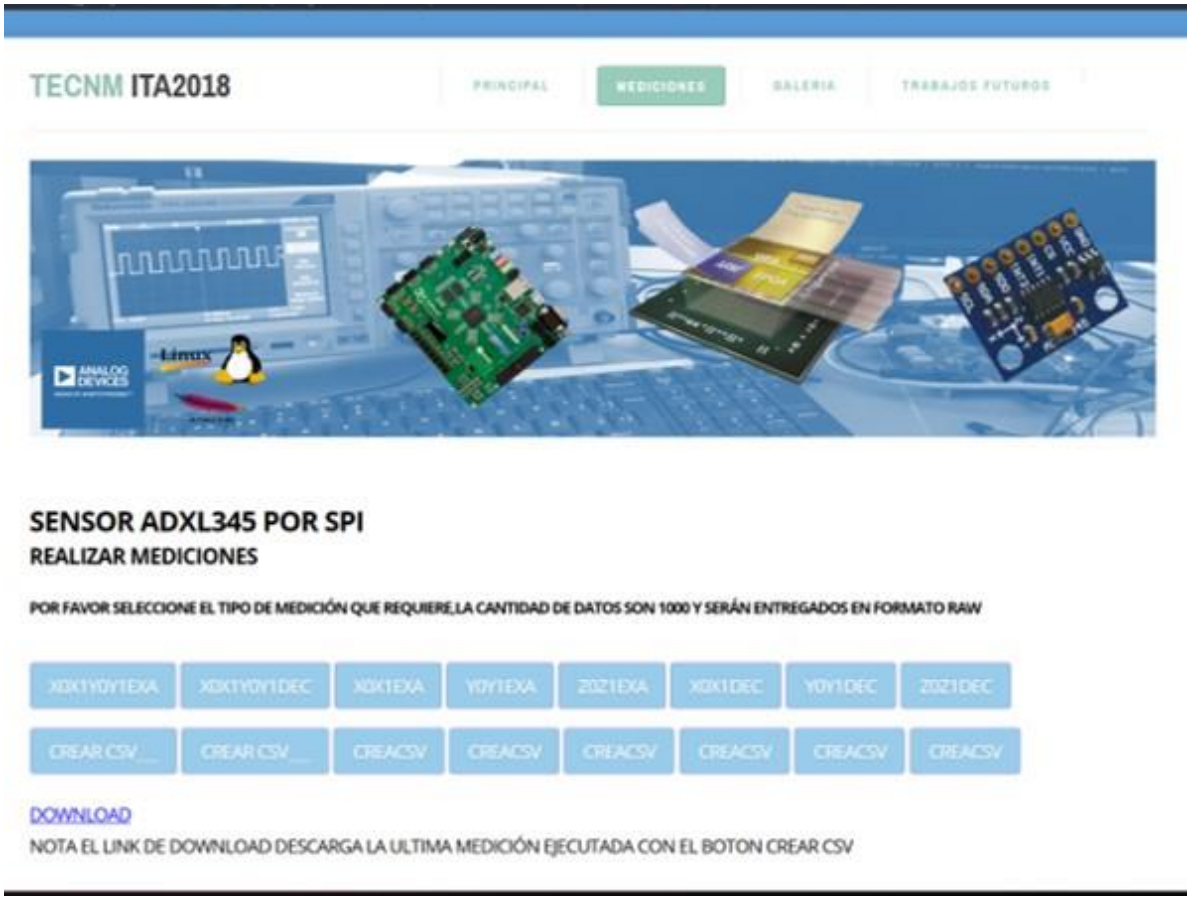


Figura 2-20 Página de la sección de mediciones.

2.12. CÓDIGO PHP ACTIVADOR

2.12.1. PHP

PHP, acrónimo recursivo en inglés de PHP Hypertext Preprocessor (procesador de hipertexto), es un lenguaje de programación de propósito general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico. Fue uno de los primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en un documento HTML en lugar, de llamar a un archivo externo que procese los datos. El código es interpretado por un servidor web con un módulo de procesador de PHP que genera el HTML resultante. PHP ha evolucionado por lo que ahora incluye también una interfaz de línea de comandos que puede ser usada en aplicaciones gráficas independientes. Puede ser usado en la mayoría de los servidores web al igual que en muchos sistemas operativos y plataformas sin ningún costo [42].

Fue creado originalmente por Rasmus Lerdorf en el año 1995. Actualmente el lenguaje sigue siendo desarrollado con nuevas funciones por el grupo PHP.2 Este lenguaje forma parte del software libre publicado bajo la licencia PHPv3_01, es una licencia Open Source validada por Open Source Initiative. La licencia de PHP es del estilo de licencias BSD, esta licencia no tiene restricciones de copyleft asociadas con GPL.

PHP puede ser desplegado en la mayoría de los servidores web y en casi todos los sistemas operativos y plataformas sin costo alguno. El lenguaje PHP se encuentra instalado en más de 20 millones de sitios web y en un millón de servidores. El número de sitios basados en PHP se ha visto reducido progresivamente en los últimos años, con la aparición de nuevas tecnologías como Node.JS, Golang, ASP.NET, etc. El sitio web de Wikipedia está desarrollado en PHP. Es también el módulo Apache más popular entre las computadoras que utilizan Apache como servidor web.

El gran parecido que posee PHP con los lenguajes más comunes de programación estructurada, como C y Perl, permiten a la mayoría de los programadores crear aplicaciones complejas con una curva de aprendizaje muy corta. También les permite involucrarse con aplicaciones de contenido dinámico sin tener que aprender todo un nuevo grupo de funciones. Aunque todo en su diseño está orientado a facilitar la creación de sitios webs, es posible crear aplicaciones con una interfaz gráfica para el usuario, utilizando alguna extensión como puede ser PHP-Qt, PHP-GTK,6 WxPHP, WinBinder, Roadsend PHP, Phalanger, Phc o HiP Hop VM. También puede ser usado desde la línea de comandos, de la misma manera como Perl o Python pueden hacerlo; a esta versión de PHP se la llama PHP-CLI (Command Line Interface).

Cuando el cliente hace una petición al servidor para que le envíe una página web, el servidor ejecuta el intérprete de PHP. Éste procesa el script solicitado que generará el contenido de manera dinámica (por ejemplo, obteniendo información de una base de datos). El resultado es enviado por el intérprete al servidor, quien a su vez se lo envía al cliente. Mediante extensiones es también posible la generación de archivos PDF,8 Flash, así como imágenes en diferentes formatos. Permite la conexión a diferentes tipos de servidores de bases de datos tanto SQL como MySQL, PostgreSQL, Oracle, ODBC, DB2, Microsoft SQL Server, Firebird, SQLite o MongoDB.9 PHP también tiene la capacidad de ser ejecutado en la mayoría de los sistemas operativos, tales como Unix (y de ese tipo, como Linux o Mac

OS X) y Microsoft Windows, y puede interactuar con los servidores de web más populares ya que existe en versión CGI, módulo para Apache, e ISAPI.

PHP es una alternativa a las tecnologías de Microsoft ASP y ASP.NET (que utiliza C# y Visual Basic .NET como lenguajes), a ColdFusion de la empresa Adobe, a JSP/Java, CGI/Perl y a Node.js/JavaScript. Aunque su creación y desarrollo se da en el ámbito de los sistemas libres, bajo la licencia GNU, existe además un entorno de desarrollo integrado comercial llamado Zend Studio. CodeGear (la división de lenguajes de programación de Borland) ha sacado al mercado un entorno de desarrollo integrado para PHP, denominado Delphi for PHP. También existen al menos un par de módulos para Eclipse, uno de los entornos más populares.

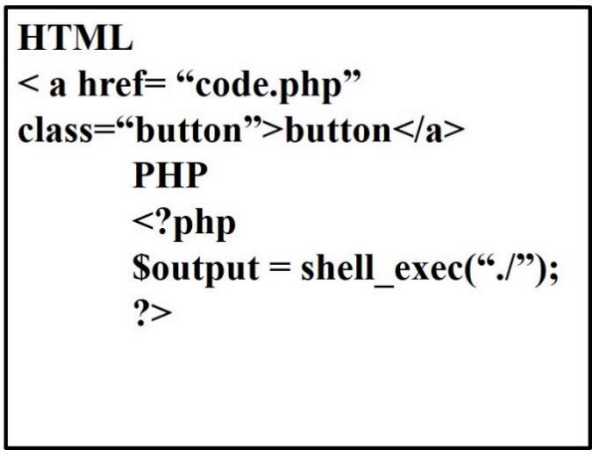


Figura 2-21 Código ejemplo que muestra fragmento de PHP dentro de HTML.

2.13. PROTOCOLOS DE COMUNICACIÓN TCP/IP Y ETHERNET

El 1968 la **Agencia de investigación de Proyectos Avanzados** del Departamento de Defensa de EE.UU. (**DARPA**) comienza un programa de desarrollo que permitiese la transmisión de información entre redes de distintos tipos y características. Se implementó una red punto a punto de líneas telefónicas denominada **ARPANET**, usando un conjunto de protocolos que posteriormente se denominarían **TCP/IP**. Esta red formada por organizaciones educativas, militares y de investigación se convirtió en el núcleo de Internet hacia 1980, y en 1983, todos los hosts de ARPANET utilizaban dicho conjunto de protocolos.

No existe un acuerdo general en como presentar el conjunto de protocolos **TCP/IP** con un modelo de capas. Generalmente se presentan como válidos entre tres y cinco niveles funcionales en la arquitectura del protocolo (ver Figura 2.21).

4 – Nivel de aplicación

3 – Nivel de transporte

2 – Nivel Internet

1 – Nivel de acceso a la red

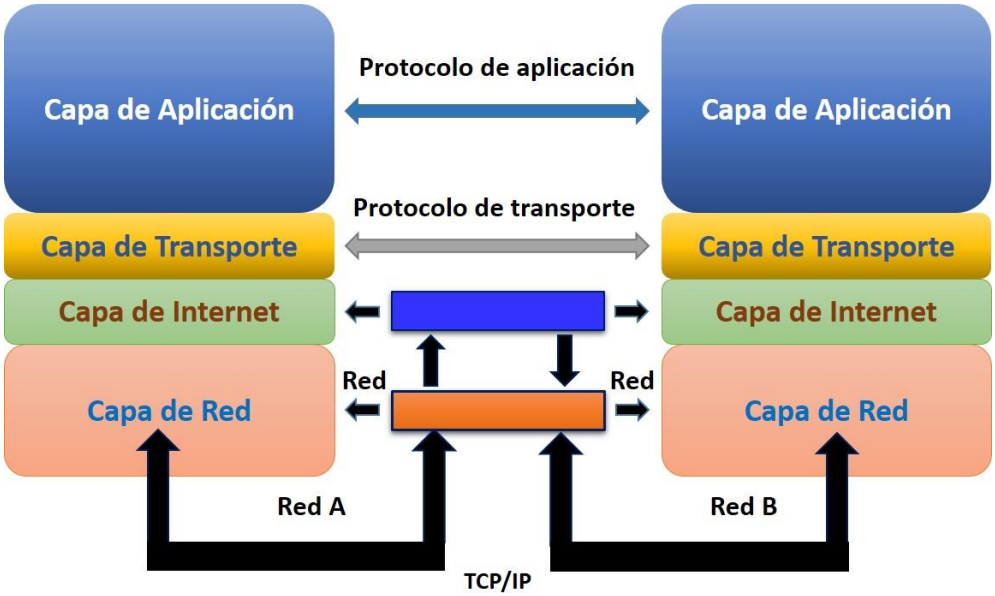


Figura 2-22 Diagrama TCP/IP.

Nivel de acceso a la red

Este este es el nivel inferior de la jerarquía de protocolos de TCP/IP. Los protocolos de esta capa proporcionan los medios para que el sistema entregue los datos a otros dispositivos directamente conectados a la red. Define cómo utilizar la red para transmitir un datagrama IP. En este nivel se encapsulan los datagramas IP formando frames o cuadros que se transmiten a la red, y transforman las direcciones IP a las direcciones físicas usadas en la red. Un ejemplo de protocolo de este nivel seria ARP (Address Resolution Protocol) en redes LAN y SLIP (Ip de Línea Serie) o PPP (Protocolo de Punto a Punto) en redes WAN.

Nivel Internet

Este nivel controla la comunicación entre equipos, eligiendo la ruta más adecuada que deben seguir los paquetes de datos para llegar a su destino. Crea el servicio básico de entrega de paquetes sobre el que se construye una red TCP/IP. El protocolo más importante de este nivel es **IP** (Internet Protocol).

Nivel de transporte

Facilita comunicación punto a punto desde un programa de aplicación a otro, asegurándose en caso de que sea necesario de que los datos llegan sin errores y en la secuencia correcta. Realiza un **checksum** para verificar también que la información no ha sido modificada durante la transmisión. **TCP** (Transmission Control Protocol) y **UDP** (User Datagram Protocol) serían los protocolos de ente nivel.

Nivel de aplicación

Encontramos en este nivel todos los procesos que hacen uso de los protocolos del nivel de transporte. Entre todos los protocolos existentes en este nivel, podemos indicar **FTP** (File transfer Protocol), **HTTP** (Hypertext Transfer Protocol), **SMTP** (Simple Mail Transfer Protocol), **DNS** (Domain Name Server), **NFS** (Network FileSystem), **telnet**, etc...

2.14. ETHERNET PHY 10/100/1000

El Zedboard implementa un puerto Ethernet 10/100/1000 para la conexión de red utilizando un dispositivo Marvell 88E1518 PHY. Esta parte opera a 1.8V. El PHY se conecta al MIO Bank 1/501 (1.8V) al Zynq-7000 AP SoC vía RGMII. El conector RJ-45 es un TE Connectivity 1840750-7 con magnetismo integrado. El RJ-45 tiene dos indicadores LED de estado que indican el tráfico y el estado del enlace válido.

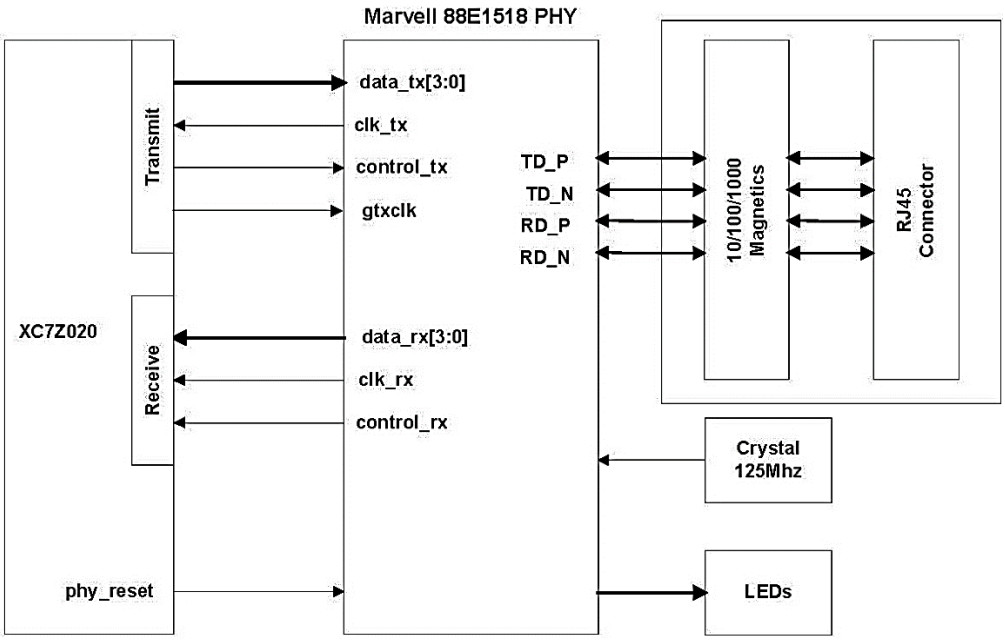


Figura 2-23 10/100/1000 Ethernet Interface [33].

En la figura anterior se muestra un diagrama de bloques de alto nivel de una interfaz Ethernet 10/100/1000 (Figura 2.22).

Capítulo 3 INTEGRANDO LA HERRAMIENTA DIGITAL

3.1. INTEGRACIÓN DE ELEMENTOS QUE CONFORMAN LA HERRAMIENTA DIGITAL

En el capítulo anterior, se desglosaron conceptos necesarios para entender los elementos que conforman la herramienta digital que se integró en este trabajo de tesis, por lo tanto, una vez hecho lo anterior, en este capítulo se muestra de forma integral y en funcionamiento, por lo cual se comienza por elegir la arquitectura Zinq-7000 por la ventaja de poder instalar un sistema operativo y la elección fue un sistema operativo basado en Linux Ubuntu, por consecuente se inicia diseñando el hardware base que requiere la arquitectura Zynq-7000 y para realizarlo se hizo uso de Vivado.

3.2. HARDWARE BASE EN VIVADO

En este parte cabe señalar que el sistema operativo que se seleccionó (Xillinux) proporciona un diseño base de hardware (ver Figura 3.1) al cual se le puede añadir nuevos bloques IP o incluso directamente archivos en lenguaje Verilog o VHDL.

Después de tener el diagrama a bloques en el mismo software Vivado se procede a realizar la síntesis e implementación y posteriormente si no existen errores se accede a generar el "Bitstream" y éste a su vez crea un archivo con extensión ".bit" que contiene el diseño encapsulado el cual es leído cuando se enciende la FPGA "Zedboard" y el iniciar busca ese archivo y genera el hardware base en el cual se montara el sistema operativo y cabe señalar que cambiando solo ese archivo, cambia el hardware base pero podemos seguir conservando en la tarjeta SD la instalación del sistema operativo al igual que las aplicaciones instaladas en dicho sistema operativo. Los reportes de consumo, de paquetes, así como de utilización de la FPGA después de la síntesis se muestran en las figuras 3.2, 3.3 y 3.4 respectivamente.

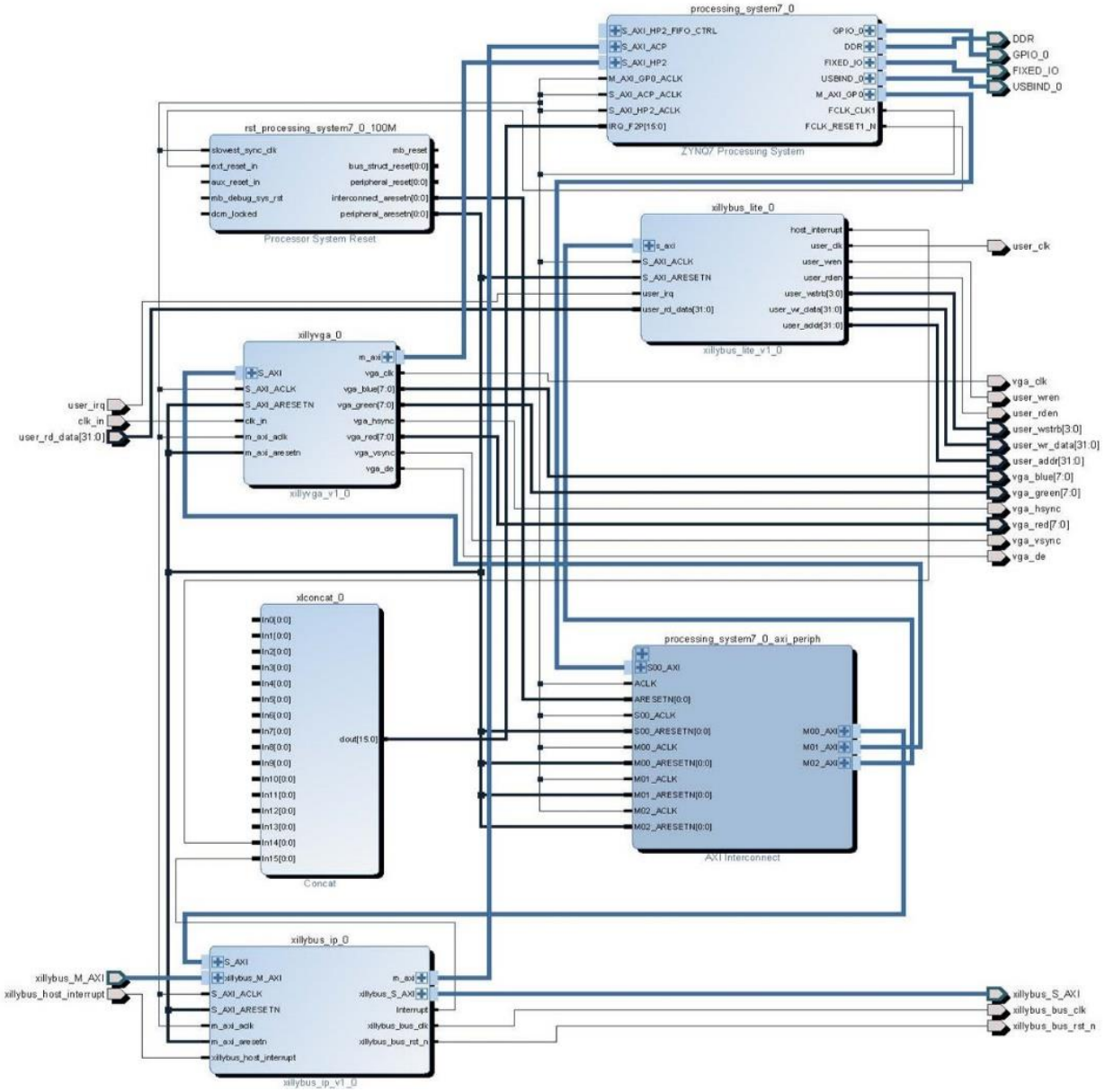


Figura 3-1 Diagrama a bloques del hardware base realizado en Vivado.

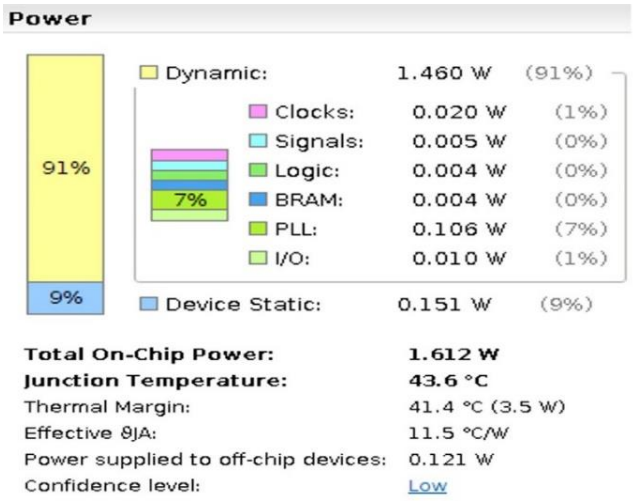


Figura 3-2 Reporte de consumo de FPGA después de la implementación.

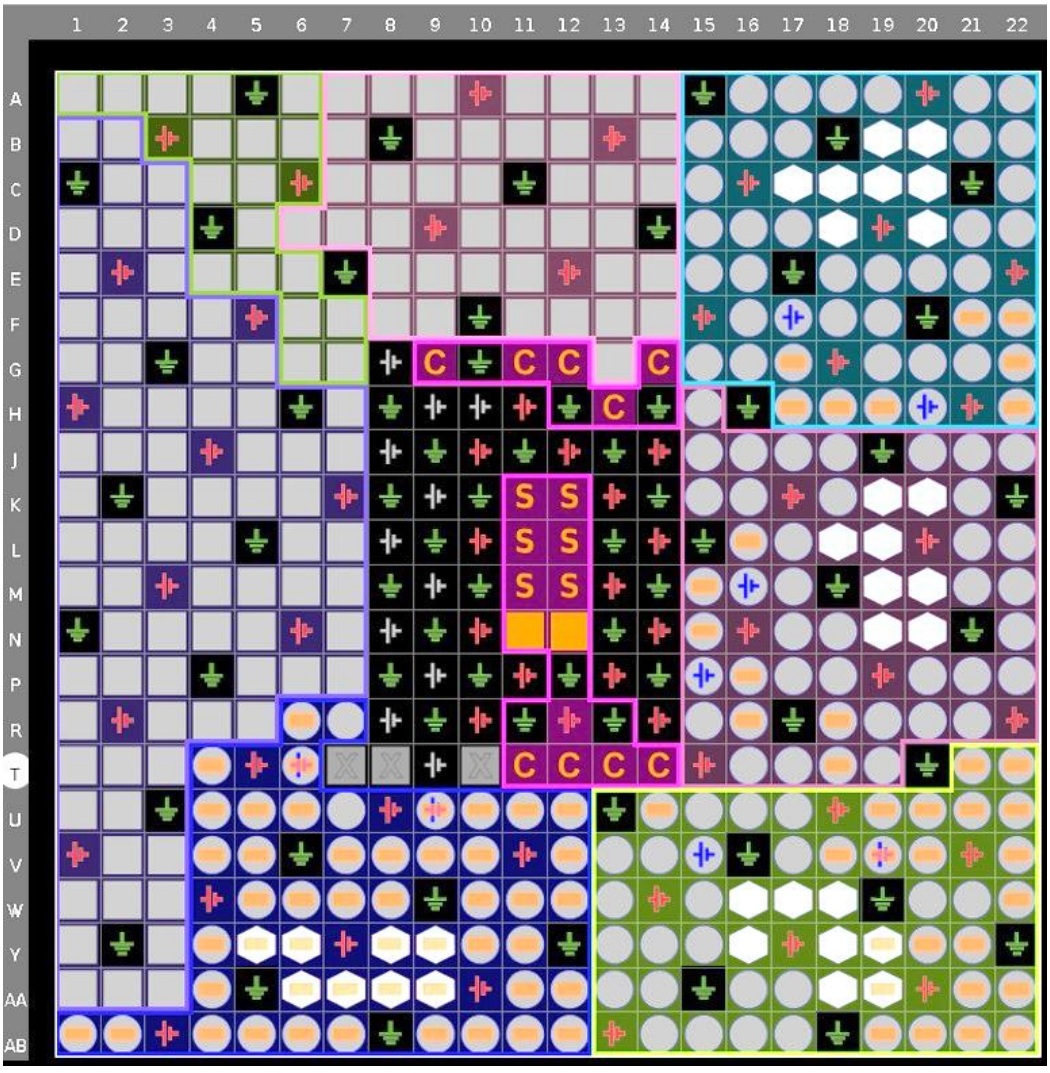
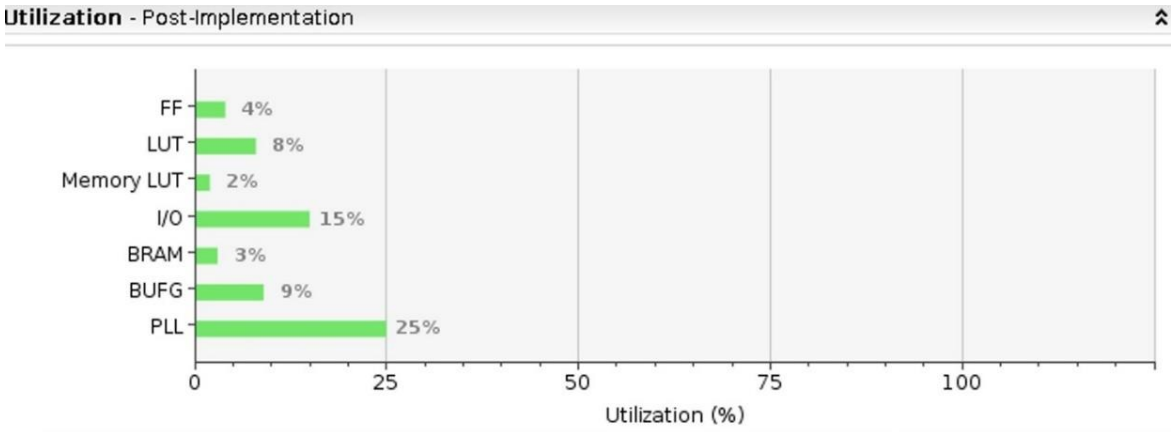


Figura 3-3 Bancos de entrada/salida asignados.



Resource	Utilization	Available	Utilization %
FF	4541.0	106400.0	4
LUT	4468.0	53200.0	8
Memory LUT	327.0	17400.0	2
I/O	29.0	200.0	15
BRAM	4.0	140.0	3
BUFG	3.0	32.0	9
PLL	1.0	4.0	25

Figura 3-4 Reporte de utilización de FPGA después de implementación.

3.3. PREPARACIÓN DE TARJETA SD

La FPGA dispone de un puerto lector de tarjetas SD el cual se utilizó en este trabajo de tesis, sin embargo, la tarjeta SD debe contener archivos específicos además de estar formateada de forma específica también (ver Anexo 4), se crean 2 particiones de diferentes formatos una en EXT4 la cual contendrá los archivos de la instalación del sistema operativo y otra en formato FAT32 la cual contendrá los archivos requeridos para el arranque correcto del sistema.

Una vez generado el archivo con extensión bit en Vivado, se agrega a otros tres archivos que deben incluirse en una de las 2 particiones creadas en la memoria SD específicamente en la partición FAT32, los archivos son los siguientes: boot.bin, devicetree.dtb, uImage y design.bit (ver Figura 3.4).

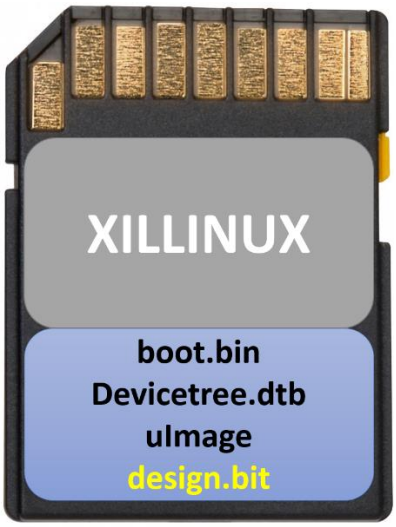


Figura 3-5 Tarjeta SD con archivos de arranque.

3.4. Conectando el Sensor ADXL345 al PMOD

El sensor ADXL345 se conecta por medio del PMOD"C" con señal LVDS, el diagrama de la conexión se muestra en la figura 3.5, los pines de "int1" e "int2" se utilizan si se requiere conectar más sensores. Los GPIO correspondientes al PMODC son del 40 al 47 sin embargo solo se utilizarán el 40,41,42,43 así como sus respectivos pines de voltaje y tierra. Los GPIO mencionados (40-43) pertenecen al Package_Pin AB7, AB6, Y4, y AA4 respectivamente.

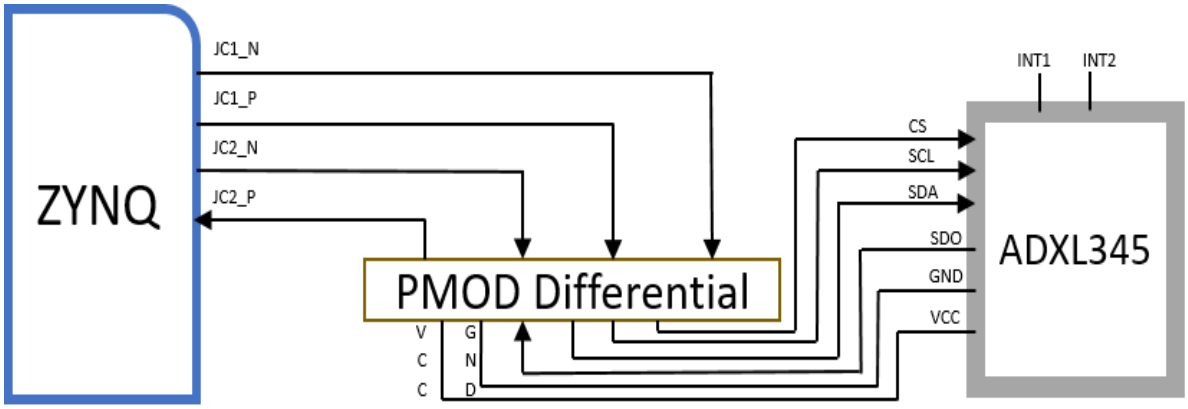


Figura 3-6 Conexión entre el Zynq, el PMOD con señal LVDS y el sensor.

Por otra parte, el esquema mostrado en la Figura 3.6 muestra la conexión interna de la Zedboard de las partes PS y PL, por lo tanto, se observa como en la parte PS se encuentra el sistema operativo Linux y dentro la aplicación desarrollada en "C" interactúa con

xillybus_mem y con el registro que fue descrito en la parte de PL, además se muestra el sensor ADXL345 conectado mediante SPI de 4 hilos (CS, SCL, MOSI y MISO).

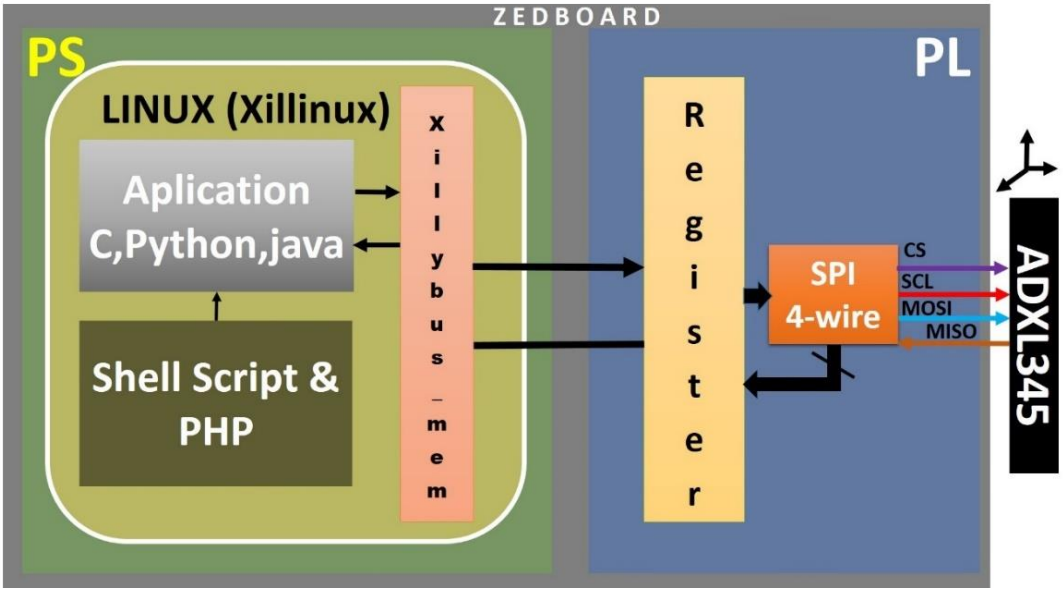


Figura 3-7 Esquema de conexión dentro de Zynq PS y PL.

Físicamente la FPGA Zedboard con el sensor conectado en el PMOD "C" se muestra en la figura 3.7.

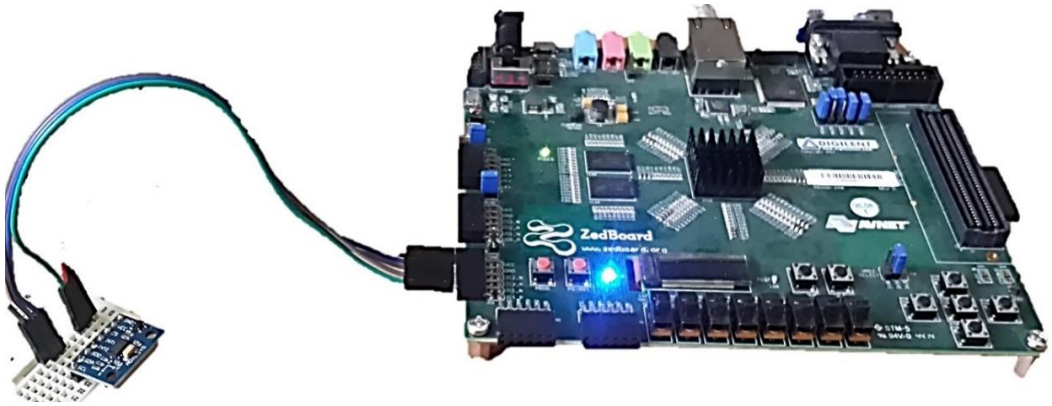


Figura 3-8 Sensor ADXL345 conectado a Zedboard en PMOD "C".

3.5. INICIANDO LA ZEDBOARD FPGA

Al iniciar la FPGA arrancara la distribución de Linux que se instaló en este caso es Xillinux, el cual iniciará en modo consola, y aunque se puede trabajar con comandos, existe la posibilidad de pasar a un entorno gráfico, el comando necesario para iniciar el entorno grafico es "**startx**" además de ofrecer mayor facilidad de uso el entorno grafico igual es necesario para poder ejecutar algunas librerías gráficas utilizadas en aplicaciones por ejemplo, la librería de OpenCV la cual si es llamada desde entorno de línea de comandos no podrá ejecutarse y saldrá un mensaje de error. Tanto en el entorno de línea de comandos, así como en el entorno gráfico se puede instalar software como si de cualquier PC se tratará, es importante señalar que no se puede actualizar la versión del sistema operativo con el comando update debido a que se crean múltiples errores en el sistema y deja de iniciar el sistema operativo, por lo que en consecuencia se tendría que volver a instalar todo lo de la partición ext4.

3.6. CONFIGURACIÓN, SERVIDOR Y APLICACIONES

Las aplicaciones que tienen la finalidad de activar al sensor con comunicación SPI como anteriormente se mencionó, se pueden crear en lenguajes de programación de alto o bajo nivel de acuerdo a las necesidades, una vez creada la aplicación se puede ejecutar para lograr el objetivo de realizar un monitoreo de forma local. En este trabajo de tesis para crear las aplicaciones necesarias que activan el sensor e inician las mediciones que arroja el sensor se utilizó lenguaje "C" debido a su alta portabilidad. Para lograr un monitoreo remoto, en primera instancia, se requiere hacer uso de un servidor web, que para este trabajo de tesis se ha seleccionado el servidor web Apache en el sistema Linux.

Posteriormente de haber instalado correctamente el servidor Apache, se creó un portal web y se alojó en la siguiente ruta /var/www/ la cual es la ruta por default del servidor web sin embargo podría alojarse en cualquier otra carpeta configurando previamente el servidor. El portal web se construyó con el lenguaje de marcado HTML en conjunción de CSS para proporcionar una mejor estética al portal. Dentro de la estructura de HTML se introdujo código en PHP el cual permite ejecutar aplicaciones desde la página web, por lo tanto, se utilizó ese recurso para iniciar aplicaciones específicas de acuerdo al botón que haya presionado el usuario.

La página que contiene los botones para realizar las mediciones que el usuario requiera es la de mediciones, así como la posibilidad de descargar los valores de las mediciones en un archivo separado por comas CSV (ver Figura 3.8) y con ello, la posibilidad de graficar los datos.

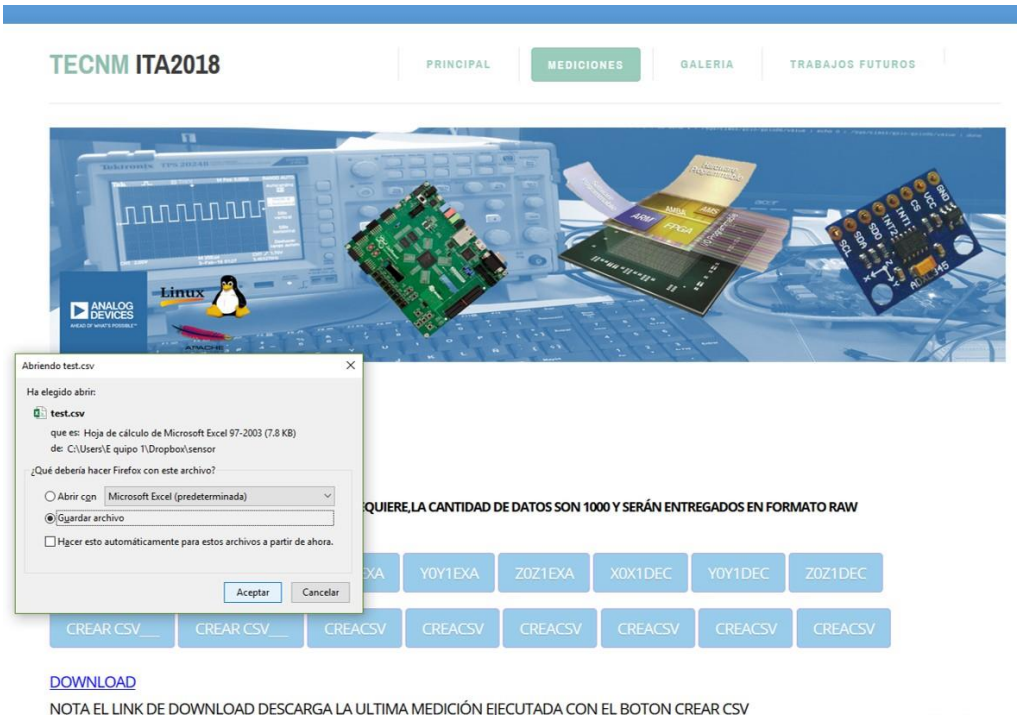


Figura 3-9 Página "Mediciones".

Para corroborar que las señales del protocolo SPI (SC, SCL, MOSI y MISO) se estén generando correctamente se realizaron mediciones con un osciloscopio de la marca Tektronix modelo TDS 3014C (ver Figuras 3.9 y 3.10).

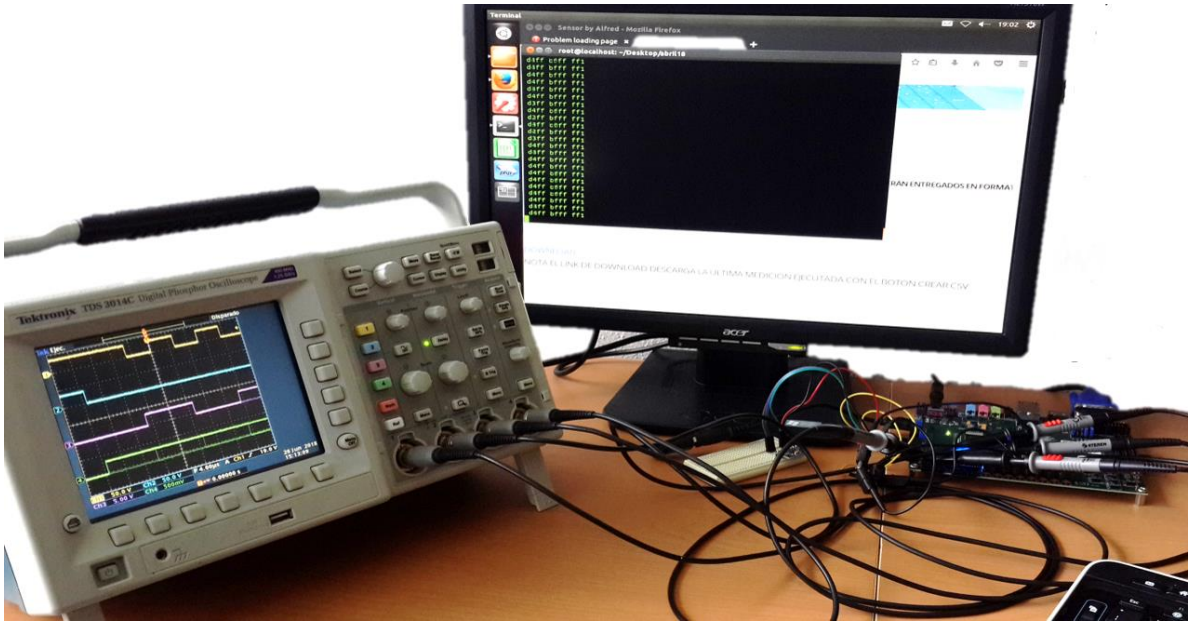


Figura 3-10 Medición de los 4 hilos de SPI.

Figura 3-11 Medición de los 4 hilos de SPI visto en un Osciloscopio.

3.7. PRUEBAS MODO LOCAL

Las primeras pruebas se realizaron de forma local, para lo cual se incluyen los siguientes elementos: Fpga, sensor, Monitor, Teclado/mouse y la aplicación que activa al sensor (ver Figura 3.11).

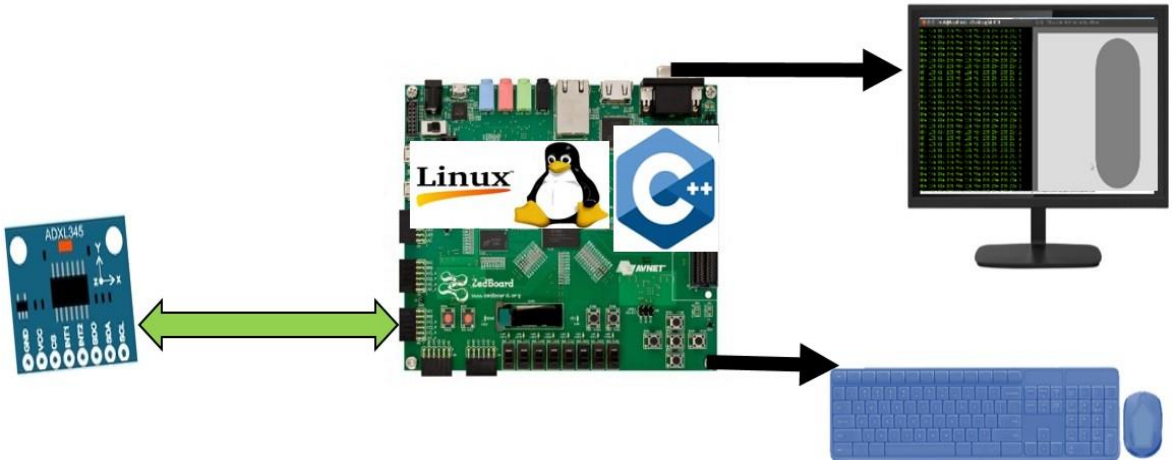


Figura 3-12 Esquema para medición local.

Realizando estas pruebas se hicieron ajustes a la aplicación en lenguaje C, ya que dependiendo de los requerimientos los valores se pueden mostrar en formato decimal, hexadecimal, binario, etc. Por lo tanto, se crearon varias aplicaciones que entregaban los valores en diferentes formatos, principalmente para realizar la página web e incluir las diferentes opciones que el usuario puede necesitar. Por otra parte, se incluyó una rutina de la librería OpenCV con la cual se logra un gráfico que responde a los diferentes valores del eje X, por ejemplo.

3.8. PRUEBAS MODO REMOTO

Para realizar las pruebas en forma remota, de forma esencial se requirió de instalar un servidor web capaz de atender las peticiones de los clientes, así mismo la creación de un portal web que se vio en el capítulo anterior y que incluye una página con las diferentes opciones de entrega de datos de la medición por ejemplo tiene un botón para solo obtener los valores de uno de los 3 ejes del sensor, además de los formatos decimal y hexadecimal, también existe la posibilidad de obtener los valores en un archivo separado por comas CSV y que tiene un botón de descarga destinado para dicha función (ver Figura 3.8)

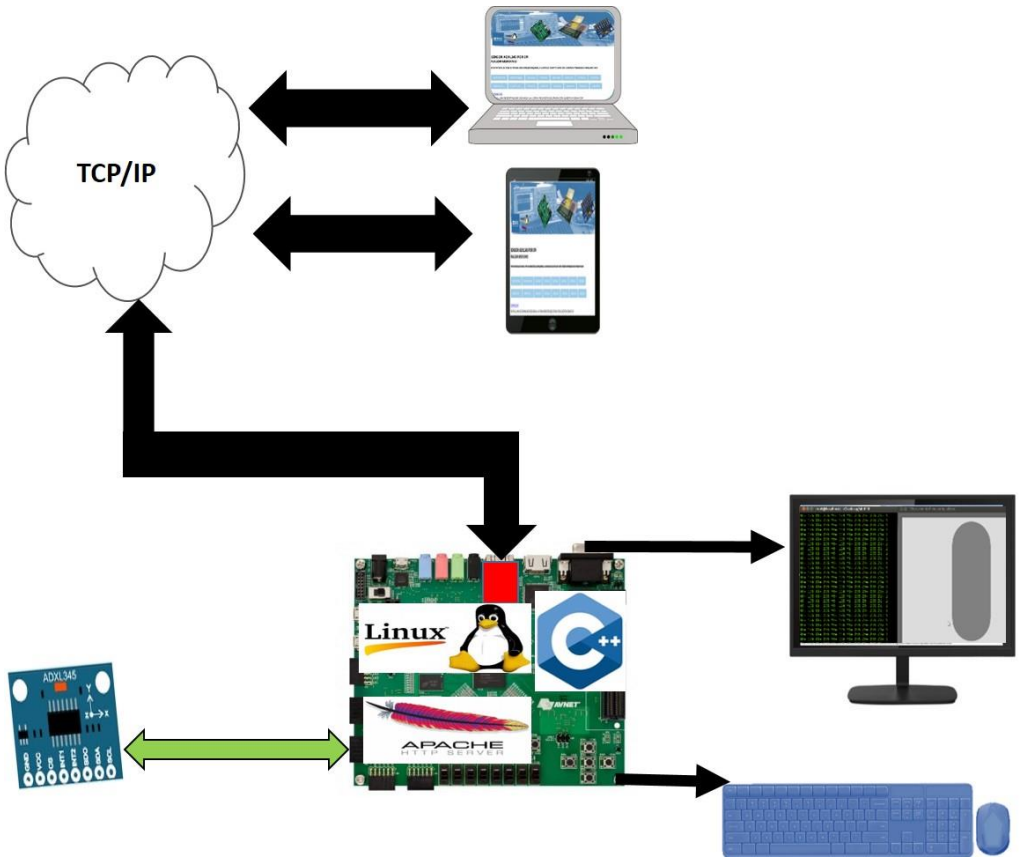


Figura 3-13 Esquema para medición Local y Remota.

3.9. PRUEBAS EN BANCO DE PRUEBAS CON MOTOR DE INDUCCIÓN

Posteriormente de las pruebas iniciales de forma local y remota se procedió a utilizar la herramienta digital a una aplicación la cual fue utilizarla acoplada a un motor de inducción el cual se encuentra con un daño en la barra del rotor, estas fallas son muy comunes y se han realizado trabajos específicos sobre la detección de fallas en los motores [43]. En este trabajo de tesis solo se realizarán pruebas de manera superficial, puesto que no se aplicará ningún algoritmo para la detección de falla ni se utilizará un diccionario de datos. Las pruebas consisten en obtener los datos del motor sano y motor dañado con una respectiva carga para posteriormente compararlos.

El banco de pruebas contiene el motor de inducción y un alternador que simula una carga y nos sirvió para realizar las mediciones agendadas (ver Figura 3.13).

Figura 3-14 Motor de inducción en el banco de pruebas.

De forma global todo el sistema capaz de realizar mediciones de forma local y remota lo vemos en la Figura 3.14.

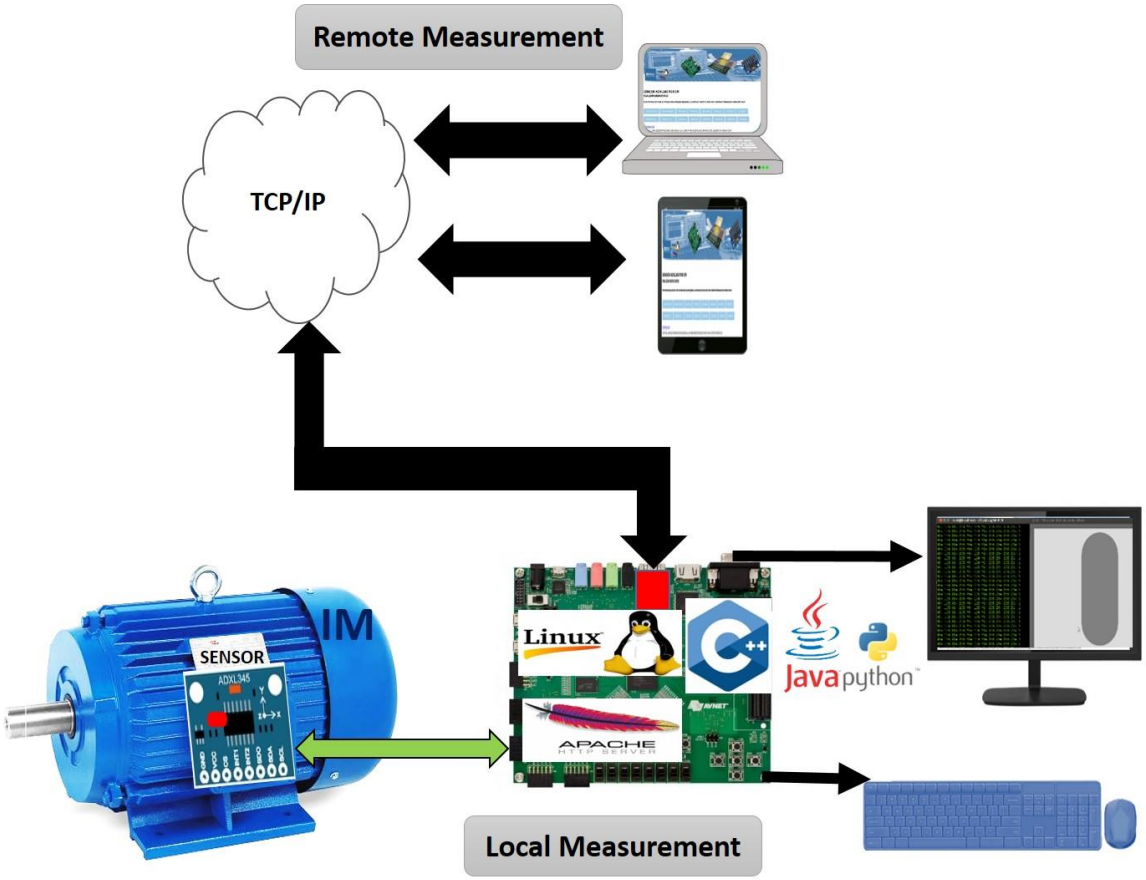


Figura 3-15 Sistema integral que muestra la herramienta digital y una aplicación.

En este capítulo se logró integrar por completo la herramienta digital que se propuso, dicha herramienta tiene la capacidad de realizar mediciones locales, así como remotas, además se incorporó a un motor de inducción mismo que en conjunto de la herramienta forma una aplicación que es la “Monitorización de las vibraciones en el motor” y que podrían servir para la detección de fallas en motores. Por consiguiente, en el capítulo 4 se realizan pruebas específicas con un motor sano y con un motor dañado y se añadirá la una carga.

Capítulo 4 APLICACIÓN

4.1. HERRAMIENTA DIGITAL EN UNA APLICACIÓN

En este trabajo de tesis se desarrolló una herramienta digital que puede ser utilizada en diversas áreas, sin embargo, para fines de ejemplificar la versatilidad e importancia de tener esta herramienta se ha seleccionado aplicarla a detectar las vibraciones en los motores de inducción, debido a que son los más utilizados en la industria.

Los motores de inducción (MI), son máquinas capaces de convertir la energía eléctrica en energía mecánica. Los primeros indicios de un motor eléctrico fueron mostrados en los experimentos de Anyos Jedlik en 1828, que consistían en un artefacto eléctrico de corriente directa formado por un estator, un rotor y un conmutador. Años más tarde en 1888, el célebre ingeniero de origen serbio Nikola Tesla, desarrolló y patentó el primer motor de corriente alterna. Con este último, se inició la era de desarrollo de los MMII, dando origen a la diversidad de diseños que se conocen hoy en día. La demanda de uso de máquinas rotatorias que existen actualmente en la industria hace de los MMII un gran candidato debido a su robustez, precio y simplicidad.

Por lo cual, es de gran importancia tener una vigilancia constante en estas máquinas, ya que las averías a nivel industrial pueden provocar pérdidas económicas. Aunque existen diversas fallas en los MI, una de gran importancia es la ruptura en barras del rotor. Aunque inicialmente el daño puede ser indetectable e inofensivo, este es paulatino y puede provocar daños en otras regiones como los rodamientos o el consumo excesivo de electricidad, hasta finalmente provocar un daño catastrófico si no es atendido.

En el trabajo ” FPGA-based broken bar detection on IM using OMP algorithm,”[43] se analizan las vibraciones obtenidas mediante el uso del acelerómetro triaxial MEM ADXL345 y le aplican un algoritmo usando el software Matlab para poder detectar fallas en el MI, la desventaja de este trabajo es que aparte de la FPGA se utilizan más recursos de hardware y software, la ventaja de adquirir los datos por la herramienta digital desarrollada en este trabajo de tesis, es que al incorporar un sistema operativo embebido se pueden aplicar procesos a las señales obtenidas de los sensores en la misma FPGA usando programas desarrollados en la misma tarjeta FPGA sin adicionar más hardware ni software. La

aplicación que se realizará con fines ilustrativos solo adquirirá la señal del sensor ADXL345 sin embargo no se le aplicará ningún algoritmo de procesamiento debido a que será implementado en trabajos futuros.

4.2. Banco de Pruebas

El banco de pruebas utilizado incorpora un motor de inducción marca WEG modelo 00118ET3EM143TW tipo jaula de ardilla construcción totalmente cerrado con ventilación exterior (TCCV-TEFC) con 1HP (ver Figura 4.1). Incorporado al motor como se muestra en la Figura 4.2 se encuentra el sensor ADXL345 el cual está cableado y en su extremo tiene un conector de 12 pines el cual se conectó a la FPGA en un puerto PMOD también de 12 pines.

Figura 4-1 MI Weg 1 HP

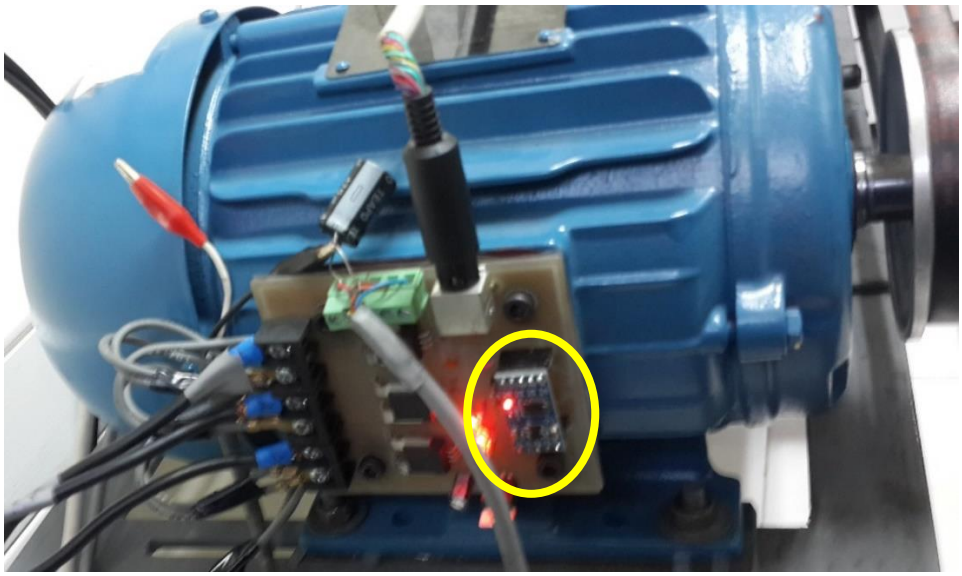


Figura 4-2 MI con sensor ADXL345 incorporado

El banco de prueba también cuenta con un alternado para simular una carga, la configuración completa para iniciar las mediciones se muestra en la Figura 4.3

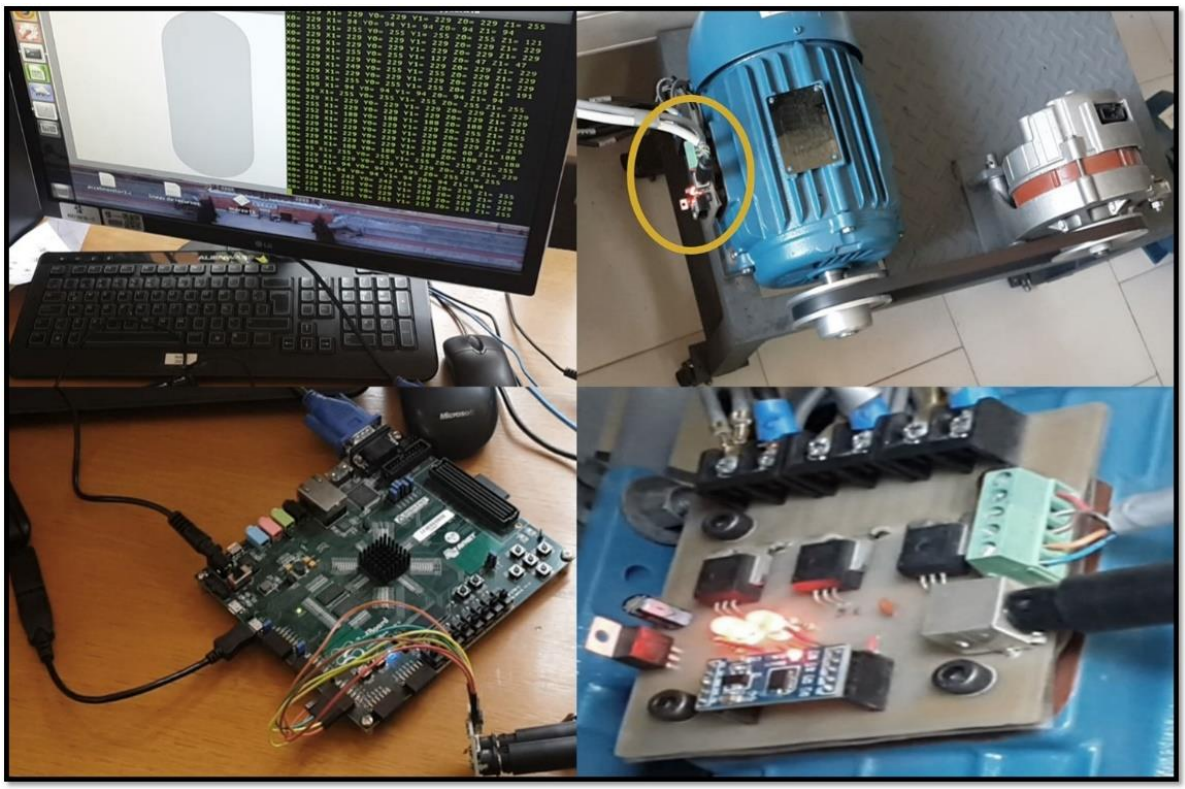


Figura 4-3 Configuración para iniciar mediciones

4.3. INICIANDO LAS MEDICIONES

Para las mediciones que se realizaron se utilizaron 2 motores, uno completamente sano y otro con un daño en el rodamiento delantero (ver Figura 4.4), lo anterior con la finalidad de poder realizar una comparativa de los tipos de valores obtenidos. Como anteriormente se indicó las señales utilizadas para la comunicación SPI se adecuaron a la configuración de los pines que tiene el banco de pruebas.

Figura 4-4 MI Sano y MI con daño

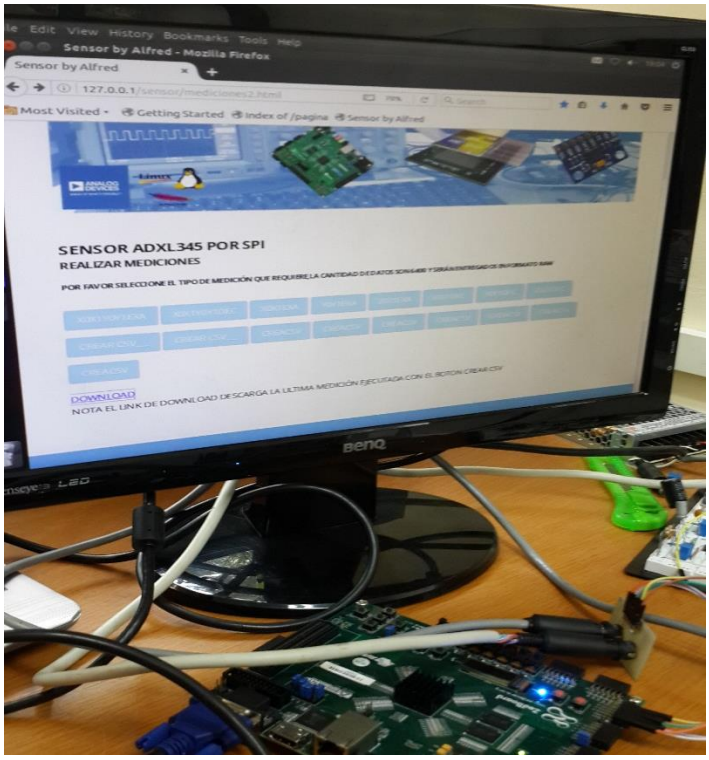


Figura 4-5 Portal Web en funcionamiento.

La construcción del portal web sirvió para realizar las mediciones con facilidad ya que se incorporó la funcionalidad de poder descargar los valores obtenidos en un archivo CSV y de esta forma hacer el proceso más eficiente (ver Figura 4.5).

En un inicio los valores que arroja el sensor ADXL345 son de aceleración con relación a la fuerza de gravedad, por lo cual se debe transformar dichas señales de aceleración, a señales de magnitud para percibir los valores que producen las vibraciones del motor.

4.3.1. MEDICIÓN DE MOTOR SANO Y CON DAÑO

Las mediciones que se obtuvieron del MI Sano y MI con daño fueron adquiridas de forma exitosa, sin embargo, al graficarlas, no existe una gran diferenciación debido a que como las señales son de tipo raw, es decir, que no se les ha aplicado algún algoritmo que permita observar la diferencia entre los dos motores como se puede observar en la Figura 4.6 y 4.7 en la que el resultado es muy similar. Por otro parte el objetivo de adquirir las señales de tipo RAW si fue exitosa.

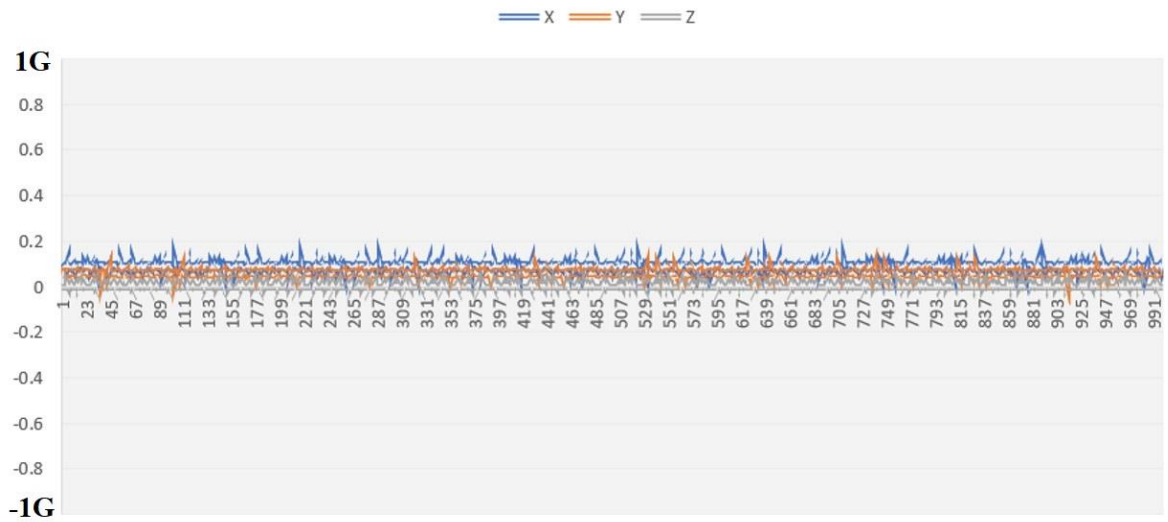


Figura 4-6 Gráfica aceleración MI sano.

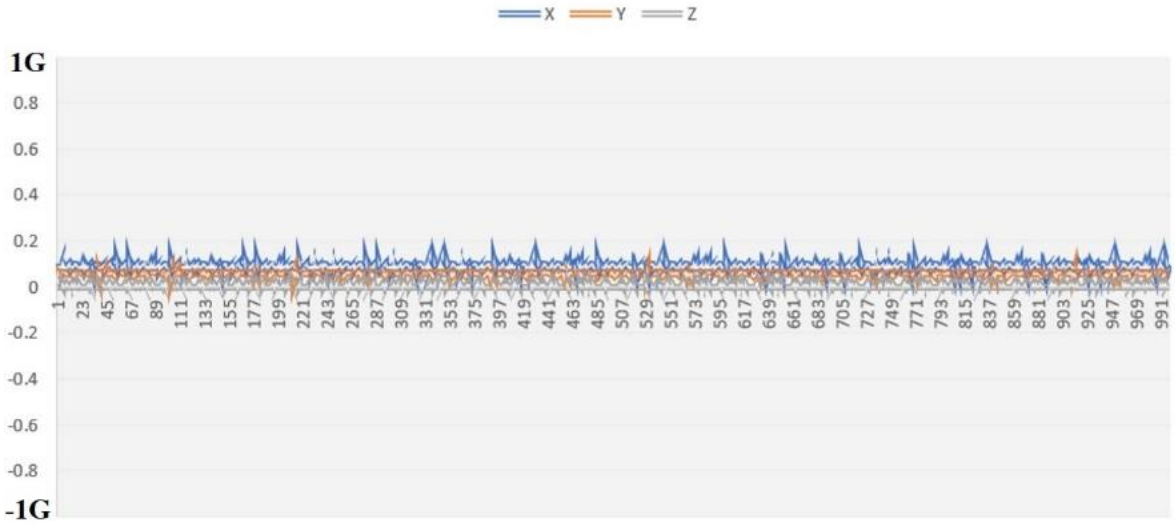


Figura 4-7 Gráfica aceleración MI con daño.

CONCLUSIONES Y TRABAJOS FUTUROS

CONCLUSIONES

La integración de la FPGA Zynq 7000 con un sistema embebido basado en Linux, así como la habilitación del PMOD LVDS con comunicación SPI dio como resultado una herramienta digital de gran relevancia porque además del sensor ADXL345 que se utilizó en este trabajo de tesis, se pueden utilizar más sensores que acepten la comunicación por SPI incluso tener 2 o más sensores conectados al mismo tiempo, además la ventaja de un sistema operativo embebido es que se puede crear software a la medida para manipular los datos recibidos mediante el PMOD que contiene la comunicación SPI usando aplicaciones creadas en lenguajes de alto nivel, de esta manera se puede reutilizar todo el diseño de hardware realizado en Vivado así como el portal web ya instalado en la FPGA, y solo se desarrollaría una nueva aplicación que tenga los parámetros necesarios para activar otro tipo de sensor y manipular los datos de acuerdo a las necesidades. Por lo tanto, se concluye respondiendo a la pregunta de investigación, que sí es posible el monitoreo remoto en una FPGA con sistema Operativo incorporado.

TRABAJOS FUTUROS

El trabajo futuro será agregar más dispositivos para tener 2 dispositivos o más conectados vía SPI al mismo tiempo aprovechando la posibilidad que ofrece el protocolo. Así como utilizar la herramienta digital a detección de fallas en motores de inducción, lo anterior conlleva a complementar con más elementos como por ejemplo la creación de un diccionario de datos y su respectivo entrenamiento, así como aplicar los diferentes algoritmos que se requieren para la detección de fallas.

REFERENCIAS

[1] Coetzee, Louis, and Johan Eksteen. 2011. “The Internet of Things – Promise for the Future? An Introduction.” Conference Proceedings:978–1.

[2] R. Drath and A. Horch, “Industrie 4.0: Hit or hype?” IEEE industrial electronics magazine, vol. 8, no. 2, pp. 56–58, 2014.

[3] N. Jazdi, “Cyber physical systems in the context of industry 4.0,” in IEEE International Conference on Automation, Quality and Testing, Robotics, 2014, pp. 1–4.

[4] A. E. Kalor, R. Guillaume, J. J. Nielsen, A. Mueller and P. Popovski, "Network Slicing in Industry 4.0 Applications: Abstraction Methods and End-to-End Analysis," in IEEE Transactions on Industrial Informatics.

[5] Y. Zeng, L. Liu and Y. Yu, ”Design of security positioning and remote network server monitoring terminal,” 2017 IEEE 17th International Conference on Communication Technology (ICCT), Chengdu, China, 2017, pp. 2026-2030.

[6] J. Saha et al., ”Advanced IOT based combined remote health monitoring, home automation and alarm system,” 2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, 2018, pp. 602-606.

[7] Z. Jinxiang and S. Chuanwei, ”Research and design of remote data acquisition system based on CC3200,” 2017 International Conference on Robotics and Automation Sciences (ICRAS), Hong Kong, 2017, pp. 158-161.

[8]] Z. Siyu, M. Jun, Z. Jianyong and M. Fei, ”Research on a Multilayer Distributed Online Monitoring System of 10kV High Voltage Switchgear,” 2013 Fourth International Conference on Digital Manufacturing Automation, Qingdao, 2013, pp. 1025-1028.

[9] A. Szekacs, T. Szakaill and Z. Hegykozi, ”Realising the SPI communication in a multiprocessor system,” 2007 5th International Symposium on Intelligent Systems and Informatics, Subotica, 2007, pp. 213-216.

[10] Zhao Ruimei andWang Mei, ”Design of ARM-based embedded Ethernet interface,” 2010 2nd International Conference on Computer Engineering and Technology, Chengdu, 2010, pp. V4-268-V4-270.

[11] Wei Zhang, ”The design of Atmega32 MCU SPI and MAX191 serial A/D communication,” 2011 International Conference on Business Management and Electronic Information, Guangzhou, 2011, pp. 526-528.

[12] I. Allafi and T. Iqbal, ”Design and implementation of a low cost web server using ESP32 for real-time photovoltaic system monitoring,” 2017 IEEE Electrical Power and Energy Conference (EPEC), Saskatoon, SK, 2017, pp. 1-5.

[13] S. Saha, M. A. Rahman and A. Thakur, ”Design and implementation of SPI bus protocol with Built-in-self-test capability over FPGA,” 2014 International Conference on

Electrical Engineering and Information Communication Technology, Dhaka, 2014, pp. 1-6.

[14] Xilinx inc. [Online]. Available: www.xilinx.com

[15] 3-Axis, 2 g/4 g/8 g/16 g Digital Accelerometer Data Sheet ADXL345. http://www.analog.com/media/en/technical-documentation/datasheets/ADXL345.pdf.

[16] G. De Micheli and Rajesh K. Gupta, Hardware/software co-design, in Proceedings of the IEEE 85.3, 1997, pp. 349-365.

[17] M. Quiñones-Cuenca, V. González-Jaramillo, R. Torres, y M. Jumbo, Sistema De Monitoreo de Variables Medioambientales Usando Una Red de Sensores Inalámbricos y Plataformas De Internet De Las Cosas, *Enfoque UTE*, vol. 8, n.º 1, pp. pp. 329 - 343, feb. 2017.

[18] Jude Adekunle Adeleke [1,2,4,*], Deshendran Moodley [2,3], Gavin Rens [1,2] and Aderemi Oluyinka Adewumi [1]," Integrating Statistical Machine Learning in a Semantic Sensor Web for Proactive Monitoring and Control",Russell Binions,2017.

[19] Alamsyah, Amir, Ardi; Subito, Mery,"Electronic Device Web-Based Monitoring System Using Atmega8535 Microcontroller"Advanced Science Letters, Volume 23, Number 3, March 2017, pp. 2220-2222(3)

[20] ArdhenduSaha[a]SampaDas[a]SureshM.[b]Raj KiranV.[c]NaiwritaDey[d] , FPGA based self-vibration compensated two dimensional non-contact vibration measurement using 2D position sensitive detector with remote monitoring, Agartala 799046, India, Elsevier Inc. 2017

[21] Widianto, Eko Didik; Waskitaningrum, Khoirunisa; Isnanto, Rizal, A Motorcycle Monitor and Control System for Teenager Riders, Semarang 50275, Indonesia, American Scientific Publishers

[22] Vivekanand Jha, Nupur Prakash, Sweta Sagar, Wearable anger-monitoring system, In ICT Express, 2017, ISSN 2405-9595.

[23] Billel Bengherbia, Mohamed Ould Zmirli, Abdelmoghni Toubal, Abderrezak Guessoum, FPGA-based wireless sensor nodes for vibration monitoring system and fault diagnosis, In Measurement, Volume 101, 2017, Pages 81-92.

[24] [5] Pong P. Chu, FPGA Prototyping by VHDL Examples. Hoboken, New Jersey: John Wiley & Sons 2008.

[25] L. Crockett, R. Elliot, M. A. Enderwitz, R. W. Stewart, The Zynq Book, Strathclyde Academic, 2014.

[26] M. Santarini, “Xilinx Unveils Vivado Design Suite for the Next Decade of ‘All Programmable’ Devices”, *Xcell Journal*, Second Quarter 2012, pp. 8 - 13.

[27] M. Dixon, P. Hammarlund, S. Jourdan and R. Singhal, “The Next Generation Intel Core Microarchitecture”, *Intel Technology Journal,* Vol. 14, Issue 3, 2010. pp. 8 - 29.

[28] Xilinx, Inc., “AXI Reference Guide”, UG761, v14.3, November 2012. Available: http://www.xilinx.com/support/documentation/ip_documentation/axi_ref_guide/latest/ug761_axi_reference_guide.pdf

[29] Xilinx, Inc., “Zynq-7000 All Programmable SoCs Product Table”, XMP087, v1.7. Available: http://www.xilinx.com/publications/prod_mktg/zynq7000/Zynq-7000-combined-product-table.pdf

[30] Xilinx, Inc., “Zynq-7000 All Programmable SoC Packaging and Pinout Product Specification”, UG865, v1.3, November 2013. Available: http://www.xilinx.com/support/documentation/user_guides/ug865-Zynq-7000-Pkg-Pinout.pdf

[31] Xillybus Ltd., https://xillybus.com

[32] Maxim Integrated, “Pmod-Compatible Plug-In Peripheral Modules” webpage.Available: http://www.maximintegrated.com/en/design/design-technology/fpga-design-resources/pmod-compatible-plug-in-peripheral-modules.html

[33] ZedBoard, Hardware Users Guide Online Available: http://zedboard.org

[34] ZedBoard, Schematic Prints(”All Documents”, Logical) Online Available: http://zedboard.org.

[35] Texas Instruments, “Interface Circuits for TIA/EIA-644 (LVDS)”, Design notes September 2002.

[36] Texas Instruments, “KeyStone Architecture Serial Peripheral Interface (SPI)”, User Guide March 2012.

[37] 3-Axis, 2 g/4 g/8 g/16 g Digital Accelerometer Data Sheet ADXL345.http://www.analog.com/media/en/technical-documentation/datasheets/ADXL345.pdf.

[38] London, Keith, "4, Programming". Introduction to Computers. 24 Russell Square London WC1: Faber and Faber Limited. 1968 p. 184.

[39] Giloi, Wolfgang, K, "Konrad Zuse's Plankalkül: The First High-Level "non von Neumann" Programming Language". IEEE Annals of the History of Computing, vol. 19, no. 2, pp. 17–24, April–June, 1997

[40] https://www.apache.org/

[41] https://opencv.org/

[42] http://www.php.net

[43] C. Morales-Perez, J. Rangel-Magdaleno, H. Peregrina-Barreto, J. Ramirez-Cortes and I. Cruz-Vega,” FPGA-based broken bar detection on IM using OMP algorithm,”

2017 IEEE International Instrumentation and Measurement Technology Conference (I2MTC), Turin, 2017, pp.1-6.

ANEXOS

ANEXO 1 ESTANCIAS TÉCNICAS

Carta liberación

Asunto: Carta de Liberación

Mtro. Felipe Pascual Rosario Aguirre
Director
Instituto Tecnológico de Apizaco
P r e s e n t e.

Por este medio me permito comunicarle que el **C. Pérez Castillo Alfredo Jesús**, con número de registro M16370020 de la Carrera de Maestría en Ingeniería Mecatrónica, ha finalizado su Estancias bajo la dirección del Dr. José de Jesús Rangel Magdaleno, en el departamento de Electrónica, en el proyecto "Activar un sistema controlador de motor modelo AD-FMCMOTCON2-EBZ de forma inalámbrica"

La colaboración comprendió del 01 de julio de 2017 al 31 de diciembre de 2017, cubriendo un total de 960 horas.

Se extiende la presente a petición del interesado y para los fines que a él convengan, en Tonantzintla, Puebla a 25 de abril de 2018.

Sin otro particular por el momento, hago propicia la ocasión para enviarle un cordial saludo.

Atentamente,

Mtra. Yenni María Carpinteyro Tlapanco
Jefa del Departamento de Servicios Escolares
Dirección de Formación Académica

YMCT*lcrm.

Instituto Nacional de Astrofísica, Óptica y Electrónica
Calle Luis Enrique Erro No.1 Santa María Tonantzintla, Puebla-México C.P. 72840 Conmutador 266 31 00
Dirección General Tel: 247 20 44 Fax 247 25 80
Dirección de Administración y Finanzas Tel: 247 43 21 Fax 247 01 81
Dirección de Formación Académica Tel: 247 27 42
Dirección de Investigación Tel: 247 43 06
Dirección de Desarrollo Tecnológico Tel: 247 43 14
Coordinación Astrofísica Tel: 247 22 31
Coordinación Óptica Tel: 247 29 40
Coordinación Electrónica Tel: 247 05 17
Coordinación Cs. Computacionales Tel: 266 31 00 Ext. 8302

Carta Satisfacción

Tonantzintla, Puebla 2018

Asunto: Constancia de Satisfacción de Estancias.

MTRO. FELIPE PASCUAL ROSARIO AGUIRRE
DIRECTOR DEL INSTITUTO TECNOLÓGICO DE APIZACO
P R E S E N T E

AT'N, DR. JOSÉ FEDERICO CASCO VASQUEZ
JEFE DE LA DIVISIÓN DE ESTUDIOS DE POSGRADO

Por medio del presente le envío un cordial saludo y aprovecho para hacerle CONSTAR que posterior a la recepción del proyecto "Comunicación remota con sistema controlador de motor AD-FMCMOTCON2-EBZ de la compañía Analog Devices", realizado por el Ing. Alfredo Jesús Pérez Castillo con número de control M16370020 y dirigido por el Dr. Roberto Morales Caporal, alumno y profesor respectivamente de la Maestría en Ingeniería Mecatrónica de la institución que usted destacadamente dirige , CUBRE Y SATISFACE LAS EXPECTATIVAS PLANEADAS AL INICIO DE LA FASE DE ESTE PROYECTO.

Agradeciendo sus atenciones quedo de ustedes.

A T E N T A M E N T E

DR. JOSÉ DE JESÚS RANGEL MAGDALENO
INVESTIGADOR TITULAR DEL AREA DE ELECTRÓNICA

ANEXO 2 ARTÍCULOS

2018 15th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE) Mexico City, Mexico. September 5-7, 2018

Real Time Monitoring of 3 Axis Accelerometer using an FPGA Zynq®-7000 and Embedded Linux through Ethernet

Alfredo Jesus Perez-Castillo
Roberto Morales-Caporal
Tecnológico Nacional de México
Instituto Tecnológico de Apizaco
Division de Estudios de Postgrado e Investigación
Email: alfredjpc@icloud.com, rmcaporal@hotmail.com

Jose de Jesus Rangel-Magdaleno
Carlos Javier Morales-Perez
Instituto Nacional de Astrofísica Óptica y Electrónica
Departamento de Electrónica
Luis Enrique Erro 1, Tonantzintla, Puebla, Mex. C.P. 72840
Email: {jrangel, carlosj.morales}@inaoep.mx

***Abstract*—This paper introduces the use of a Field Programmable Gate Array (FPGA) with Zynq-7000 architecture and embedded Linux, as a digital tool to perform real-time remote monitoring of Internet-connected devices, using sensors that support SPI communication, the advantage of implementing the SPI protocol in the FPGA is that it can use several sensors that support this protocol even joining them at the same time. The incorporation of an embedded Linux operating system opens the possibilities of communication and execution of software as if it were a personal computer, as an example in this work, a 3-axis accelerometer is used. By integrating the above elements, a digital tool capable of covering the need for remote monitoring in real time is accomplished with a web server and connecting the FPGA through the Ethernet port by the TCP / IP protocol.**

***Keywords*—FPGA, Xillinux, SPI, ADXL345, ZYNQ-7000, Zedboard™.**

I. Introduction

The human being is currently in constant movement, therefore, the local and remotely interaction with the electrical and electronic devices it is relevant to minimize response time. Besides that, through numerous technology advances, society is moving towards an always connected paradigm [1]. It requires an interface between the device and the user that allows monitoring and, according to the needs, obtains specific values. This is very important to the fourth industrial revolution, known as Industry 4.0, brings Cyber-physical systems and Internet of Things (IoT) to industrial manufacturing systems [2], [3]. Furthermore, the number of interconnected physical devices will increase drastically, and they will continuously interact with local cloud services in order to act intelligently and flexibly [4].

In previous works [5]–[8] the remote monitoring solution has been implemented. However, it has been done using micro-controllers and even FPGA but without the architecture capable of installing an operating system or without the incorporation of communication Serial Peripheral Interface Bus (SPI). SPI is a synchronous serial data link standard named by Motorola that operates in full duplex mode. Sometimes SPI is called a ”four wire” serial bus. There is one master and one or more slave devices in the communication [9]. In addition, the advantage of this bus compared to I2C communication is that it offers a higher transmission speed.

Fig. 1. ZedBoard™ FPGA [17].

Previous works have implemented the SPI communication protocol and have been done using microcontrollers such as LM358962 [12], ATmega32 [13], ESP32 [14], and also using FPGA for example a Spartan 2 [15], on the part of the microcontrollers are of fixed architecture and cannot be modify the hardware and in the cases of FPGA without architecture SoC are limited to a Bare Metal design, the contribution of this work is that by using the Zynq®-7000 chip offers the possibility of creating a own hardware design and install an embedded operating system (OS) and with the above create an environment similar to that of a personal computer, having an OS can create and execute programs made under high-level or low level programming languages, for example, in works

978-1-5386-7033-0/18/$31.00 ©2018 IEEE

2018 15th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE) Mexico City, Mexico. September 5-7, 2018

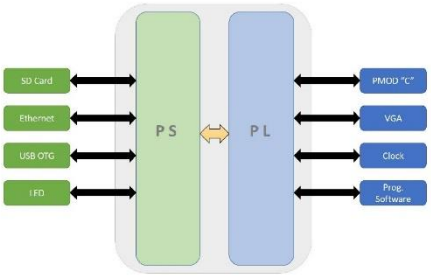


Fig. 2. PS and PL section used.

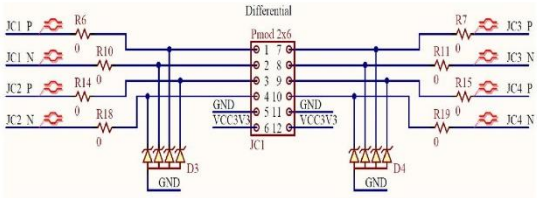


Fig. 3. PMOD "C" with LVDS [10]

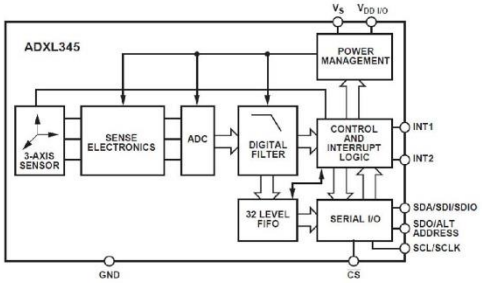


Fig. 4. Functional Block Diagram ADXL345 [11].

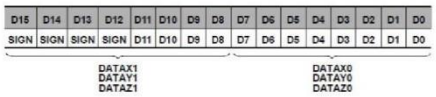


Fig. 5. Data Construction of ADXL345 [11].

related to the detection of motor failures [16], FPGA and adxl345 accelerometer have been used via SPI, however the processing of acquired signals is done offline using additional hardware and software, the main contribution of this work is that it can be performed the processing in the same FPGA and online with the operating system incorporated and SPI protocol. Therefore this work proposes the use of a Xilinx FPGA Zedboard™ development board with Zynq-7000 architecture, all programmable SoC [17] (see Fig.1), this architecture includes a Processing System (PS) and a Programmable Logic (PL) incorporating a Dual-Core ARM Cortex-A9 at 866 Mhz and an Artix-7 FPGA respectively, which allows installing an operating system (OS) and in the logical part the co-design of a specific hardware for the required interface. In this case is a 4-wire SPI communication to receive data from the triaxial accelerometer model ADXL345 [11]. The appearance of processors and FPGA of low cost and energy efficiency has made possible the realization of these applications through the joint design of hardware/software [18].

II. Hardware Co-design and OS

A. Hardware core Zedboard

The FPGA based on Zynq-7000 Zedboard model has two options of programming it, in a bare-metal way or using an operating system which uses the architecture that is defined in the design made in the proprietary software of Xilinx® (Vivado VHDL) (see Fig.2). By using the bare-metal type is limited to have only one specific application, however if an operating system is used it exists the possibility of install applications and run software as if it were a Personal Computer. In this case it was co-designed the base hardware already designed by the company Xillybus and add the SPI interface to one of the 2 available PMOD™ (GPIO) of differential type in the part of the programmable logic that includes the FPGA [19]. The main feature of having an FPGA and a processor on the same chip is that the hardware can be changed and adapted to specific needs just replacing a file inside SD card using by Zedboard.

The necessary OS to complement the interface should be capable of covering local and remote connection needs, beside a program that links the sensor with the FPGA. Therefore a Linux distribution is proposed as an operating system, thus allows to install a web server to be able to access remotely and makes requests to the server [20], also an HTML website was created that contains Hypertext Preprocessor code (PHP) [21]. The application which controls the sensor can be written in high-level language (C, C++, Java, python, etc.), it is proposed to program above application in "C" language due to its high portability. Finally the Ethernet port was used 10/100/1000 PHY of the FPGA to communicate through TCP/IP. With the aforementioned, a digital tool is obtained that incorporates the necessary interface for the interaction between the device and the user, creating the possibility of applying said tool to varied applications.

Figure 2 shows the elements of the processing part as well as programmable logic that were used mainly. The Zedboard has 5 PMOD available, nonetheless only 2 (PMOD "C" and PMOD "D") use the low-voltage differential signal system or LVDS. The LVDS is a technology that was developed [22], [23] to provide a low-power, low-voltage alternative for high-speed point-to-point data transmission. Figure 3 shows the "C" PMOD with LVDS signal.

B. Vibrations Sensor

Microelectromechanical systems (MEMS) digital sensor ADXL345 is a small, thin, ultra-low power, 3-axis accelerom-

2018 15th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE) Mexico City, Mexico. September 5-7, 2018

TABLE I
G RANGE SETTING

Setting		G Range
D1	D2	
0	0	±2g
0	1	±4g
1	0	±8g
1	1	±16g

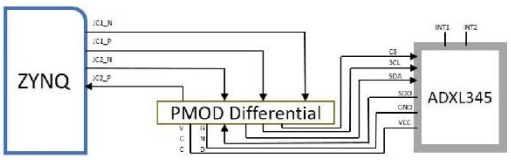


Fig. 6. SPI 4-wire to PMOD.

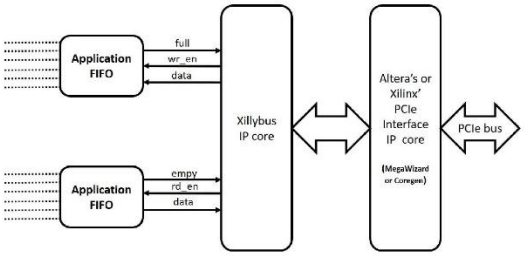


Fig. 7. Simplified FPGA block diagram of Xillybus using PCIe transport [23].

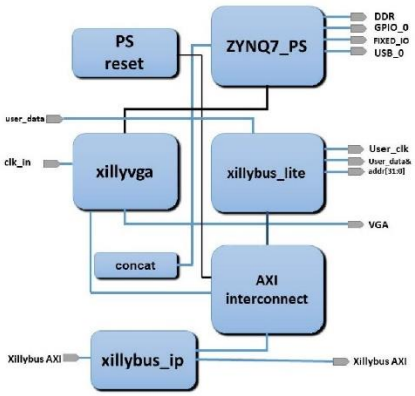


Fig. 8. Block diagram design in Vivado.

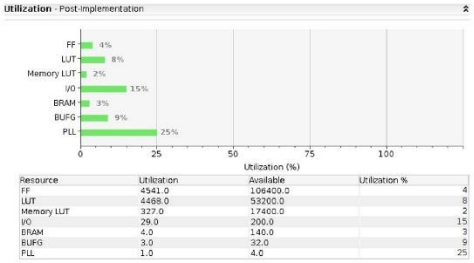


Fig. 9. Utilization post-implementation report.

eter with high resolution (13-bit) measurement at up to 16 g. Digital output data is formatted as 16-bit twos complement and is accessible through either a SPI (3- or 4-wire) or I2C digital interface. It measures the static acceleration of gravity in tilt-sensing applications, as well as dynamic acceleration resulting from motion or shock. Its high resolution (3.9 mg/LSB) enables measurement of inclination changes less than 1.0 degree, it has been used in works related to failure detection in induction motors [16], [24]. The range of the sensor is shown in table I, and functional block diagram in Figure 4.

The data format of the ADXL345 is 16 bits. Once acceleration data is acquired from data registers, the user must reconstruct the data. DATAX0 is the low byte register for X-axis acceleration and DATAX1 is the high byte register. In 13-bit mode, the upper 4 bits are sign bits (see Fig.5). When in 13-bit mode, 1 LSB represents about 3.9 mg.

Communication of the ADXL345 sensor with the PMOD "C" of the Zedboard FPGA was through 4-wire SPI CS, SCL, SDA and SDO (see Fig.6) and that connects to ZYNQ by AB7, AB6, Y4 and AA4 General Purpose Input/Output (GPIO).

III. INSTALLING OS AND CREATING APPLICATIONS

A. Selecting OS

The FPGA Zedboard allows for its architecture installs an operating system, the selection will depend on the needs, there are several options such as Linux™, Windows® and even Android™. The distributions are specific depending on the board, are made to order, for the purpose Xillinux was used [25] which is a Linux distribution based on Ubuntu 12.04 version, the distribution includes a Zedboard base design (in Verilog or VHDL code) in addition to the opportunity to create a specific IP from Xillybus web portal, the IP created is appended to the IP core in Vivado, the main feature of Xillinux is that it works through FIFOS using the Peripheral Component Interconnect Express bus (PCIe), the developed applications are communicated through specific FIFO that can be of different sizes (8, 16 and 32 bits) the overall operation is shown in Figure 7.

B. Vivado Software Design

After hardware and OS selection, it proceeded to hardware design in Vivado [17] provided by Xillybus, which was attached to a shift register to achieve SPI communication and

```
//part1
Write(0x31, 0x00); // 2g
Write(0x2d, 0x08); // Start Measure

//part2
xdata0 = SpiRead(0x32); // Read X0
printf("%x", xdata0);
xdata1 = SpiRead(0x33); // Read X1
printf("%x ", xdata1);
ydata0 = SpiRead(0x34); // Read Y0
printf("%x", ydata0);
ydata1 = SpiRead(0x35); // Read Y1
printf("%x ", ydata1);
zdata0 = SpiRead(0x36); // Read Z0
printf("%x", zdata0);
zdata1 = SpiRead(0x37); // Read Z1
printf("%x\n ", zdata1);
```

Fig. 10. Extract program code.

TABLE II
ADXL345

Address	*Name*	*Features*	*Type*
0x00	DEV ID	bit[1:0]	R
0x2D	POWER CTL	bit[3]	RW
0x31	DATA FORMAT	bit[1:0]	RW
0x32	DATAX0	X0 value	R
0x33	DATAX1	X1 value	R
0x34	DATAY0	Y0 value	R
0x35	DATAY1	Y1 value	R
0x36	DATAZ0	Z0 value	R
0x37	DATAZ1	Z1 value	R

Register Map.

performed the synthesis and implementation thereof with the aim of generating the file necessary to start the FPGA with the Linux distribution. The block diagram design of the project in the Vivado software is shown in Figure 8. With the above, a file with a .bit extension has been created, which contains the design of the hardware for the programming of the FPGA when starting up. It should be noted that the SD card contains 2 partitions, one with Xillinux in EXT4 format and the other one in FAT format with the necessary files to start up the FPGA, the four necessary files are boot.bin, Devicetree.dtb, uImage and own design.bit, the use of the FPGA after the synthesis and implementation shown in Figure 9.

C. Server and Application to manage sensor

To achieve the remote monitoring of the sensor, it is proposed to install a Web Server in the FPGA to attend the remote requests of the clients, for this Apache Web Server was installed in the Ubuntu distribution making possible a web portal in HTML that executes scrips in PHP and these at the same time execute applications. The advantage of embedded operating system is that it can develop applications in the high or low level language that is required externally or directly in FPGA, in this case the application was made in C language because it is a transportable and transparent language, Shell Script to execute commands automatically and PHP that is included within the HTML markup language. The application to activate the ADXL345 sensor must send, according to the needs, the hexadecimal values that are defined in the ADXL345 sensor shown in table II. On the other hand, it can see in Figure 10 in the code extract in the language "C" in part 1 the instructions are sent for the ADXL345 sensor to start the measurement and also to be set to 2G (it can be another range), in part 2 of the code we observe where the values 0 and 1 of XYZ are requested according to Table II.

The overall interaction of the ADXL345 sensor with Xillybus and the FPGA is described in the diagram of Figure 11.

D. Web page executing PHP

The main web page hosted on the server contains PHP code within its structure, which has the purpose of executing specific applications (see Fig.12). The above is very useful because the user can executes applications or even scripts by clicking on the button inside the web page.

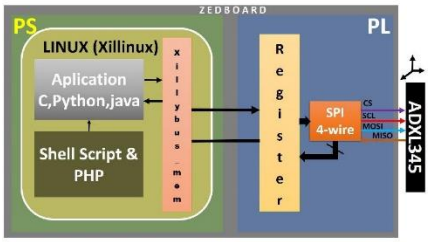


Fig. 11. Interaction between Zedboard and Xillybus.

IV. Measurements

A. Local

Locally to the Zedboard card the user can connect a monitor and keyboard/mouse devices through the USB On-The-Go port (OTG) to interact as if it were a Personal Computer, in this way the user can run the scheduled applications and visualize in real time how the sensor values go printed on the screen. On the other hand, if it is required, the user could use graphic libraries such as Opencv (Open Computer Vision) [26] (see Fig.13).

B. Remote

Access remotely is the most useful since there is a possibility of accessing the system from any site through the Internet, as mentioned in the previous section a web portal was built that hosted on a web server that offers the possibility of interact remotely performing the measurements and obtaining them in RAW form or if the user prefers also download it in a comma separated file format (CSV) to graph or manipulate it more easily. The diagram in Figure 14 shows graphically the operation and interaction of the FPGA card with the installed server and interacting with the clients, and in the Figure 15 shown the measurement of SPI signals in oscilloscope. The integration of the system and its operation is shown in the Figure 16, where it is observed that the FPGA is making measurements locally with directly connected peripherals and remotely through the TCP / IP protocol, following the requests of the web clients.

```
HTML
< a href= "code.php"
class="button">button</a>
        PHP
        <?php
        $output = shell_exec("./");
        ?>
```

Fig. 12. HTML code with PHP inside.

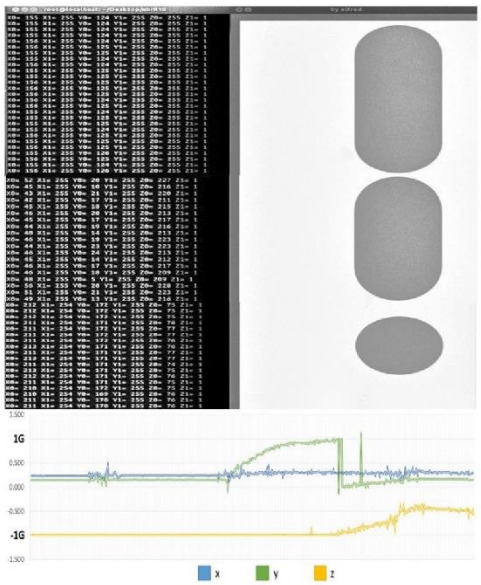


Fig. 13. Sensor values on the left and on the right side of a graph generated by the OpenCV library reacted to X values, and the corresponding graph

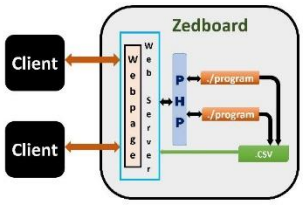


Fig. 14. Client-server model implement.

V. Conclusion

The integration of the Zynq 7000 FPGA with an embedded Linux-based system, as well as the enabling of a PMOD with LVDS and SPI communication, resulted in a powerful digital tool of great relevance due to besides of ADXL345 sensor used in this work. It can be used more sensors that accept SPI communication even having two or more sensors connected at the same time, plus the advantage of an embedded operating system is that the user can creates custom software

Fig. 15. Measurement of CS, SCL, SDO and SDA signals of the SPI in an oscilloscope.

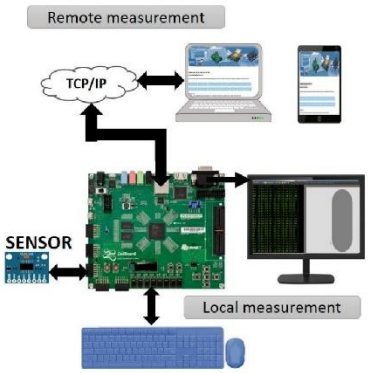


Fig. 16. Integrated system and running.

to manipulate the data received through the PMOD containing the SPI communication using applications made in high level languages. Furthermore it can be installed a web server and be able to perform remote monitoring, thus, it can reuse all the hardware design or even add additional hardware made in Vivado, as well as the web portal already installed in the FPGA, and it would only be necessary a new application that has the necessary parameters to activate another type of sensor and manipulate the data according to the needs. In future work this digital tool will be integrated to the monitoring of vibration signals in induction motors.

Acknowledgment

The first author wants to thanks to Consejo Nacional de Ciencia y Tecnología (CONACYT, México) for the support to the master degree studies in the Technological Institute of Apizaco and to National Institute for Astrophysics, Optics and Electronics (INAOE, México) for allow the technical residence and for all the support provided.

References

[1] L. Coetzee and J. Eksteen, "The internet of things - promise for the future? an introduction," in *2011 IST-Africa Conference Proceedings*, May 2011, pp. 1–9.

[2] R. Drath and A. Horch, "Industrie 4.0: Hit or hype? [industry forum]," *IEEE Industrial Electronics Magazine*, vol. 8, no. 2, pp. 56–58, June 2014.

[3] N. Jazdi, "Cyber physical systems in the context of industry 4.0," in *2014 IEEE International Conference on Automation, Quality and Testing, Robotics*, May 2014, pp. 1–4.

[4] A. E. Kalor, R. Guillaume, J. J. Nielsen, A. Mueller, and P. Popovski, "Network slicing in industry 4.0 applications: Abstraction methods and end-to-end analysis," *IEEE Transactions on Industrial Informatics*, pp. 1–1, 2018.

[5] Y. Zeng, L. Liu, and Y. Yu, "Design of security positioning and remote network server monitoring terminal," in *2017 IEEE 17th International Conference on Communication Technology (ICCT)*, Oct 2017, pp. 2026–2030.

[6] J. Saha, A. K. Saha, A. Chatterjee, S. Agrawal, A. Saha, A. Kar, and H. N. Saha, "Advanced iot based combined remote health monitoring, home automation and alarm system," in *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*, Jan 2018, pp. 602–606.

[7] Z. Jinxiang and S. Chuanwei, "Research and design of remote data acquisition system based on cc3200," in *2017 International Conference on Robotics and Automation Sciences (ICRAS)*, Aug 2017, pp. 158–161.

[8] Z. Siyu, M. Jun, Z. Jianyong, and M. Fei, "Research on a multilayer distributed online monitoring system of 10kv high voltage switchgear," in *2013 Fourth International Conference on Digital Manufacturing Automation*, June 2013, pp. 1025–1028.

[9] A. Szekacs, T. Szakaill, and Z. Hegykozi, "Realising the spi communication in a multiprocessor system," in *2007 5th International Symposium on Intelligent Systems and Informatics*, Aug 2007, pp. 213–216.

[10] Zedboard, "Schematic prints("all documents", logical)," urlhttp://zedboard.org, 2018.

[11] A. Devices, "3-axis, 2 g/4 g/8 g/16 g digital accelerometer data sheet adxl345." urlhttp://www.analog.com/media/en/technical-documentation/data-sheets/ADXL345.pdf, 2018.

[12] Z. Ruimei and W. Mei, "Design of arm-based embedded ethernet interface," in *2010 2nd International Conference on Computer Engineering and Technology*, vol. 4, April 2010, pp. V4–268–V4–270.

[13] W. Zhang, "The design of atmega32 mcu spi and max191 serial a/d communication," in *2011 International Conference on Business Management and Electronic Information*, vol. 2, May 2011, pp. 526–528.

[14] I. Allafi and T. Iqbal, "Design and implementation of a low cost web server using esp32 for real-time photovoltaic system monitoring," in *2017 IEEE Electrical Power and Energy Conference (EPEC)*, Oct 2017, pp. 1–5.

[15] S. Saha, M. A. Rahman, and A. Thakur, "Design and implementation of spi bus protocol with built-in-self-test capability over fpga," in *2014 International Conference on Electrical Engineering and Information Communication Technology*, April 2014, pp. 1–6.

[16] C. Morales-Perez, J. Rangel-Magdaleno, H. Peregrina-Barreto, J. Ramirez-Cortes, and I. Cruz-Vega, "Fpga-based broken bar detection on im using omp algorithm," in *2017 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, May 2017, pp. 1–6.

[17] X. inc., "Xilinx - adaptable, inteligent." urlhttp://www.xilinx.com, 2018.

[18] G. D. Michell and R. K. Gupta, "Hardware/software co-design," *Proceedings of the IEEE*, vol. 85, no. 3, pp. 349–365, Mar 1997.

[19] AVNET, "Hardware users guide," urlhttp://zedboard.org /sites/default/files/documentations/ZedBoard_HW_UG_v2_2.pdf, 2014.

[20] Apache, "The apache software fundation," urlhttps://www.apache.org/, 2018.

[21] T. P. Group, "The php group," urlhttp://www.php.net/, 2018.

[22] Q. Cabanes and B. Senouci, "Objects detection and recognition in smart vehicle applications: Point cloud based approach," in *2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN)*, July 2017, pp. 287–289.

[23] Xilinx, "Zynq-7000 soc technical reference manual," urlhttps://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf, 2018.

[24] C. Morales-Perez, J. Rangel-Magdaleno, H. Peregrina-Barreto, and J. Ramirez-Cortes, "Half-broken rotor bar detection on im by using sparse representation under different load conditions," in *2017 IEEE International Autumn Meeting on Power, Electronics and Computing (ROPEC)*, Nov 2017, pp. 1–5.

[25] Xillybus, "Xillybus. ip cores and desing service," urlhttp://xillybus.com/, 2018.

[26] O. team, "Opencv library," urlhttps://opencv.org/, 2018.

Proyectos Académicos de Robótica y Mecatrónica.
ISBN: 978-607-9394-XX-X, Asociación Mexicana de Mecatrónica A.C. – Instituto Tecnológico Superior de Poza Rica, 2017

Simulación de Convertidores CD/CD Como Cargadores de Baterías para una Bicicleta Eléctrica.

Cervantes Hernández David [1], Morales Caporal Roberto [1], Pérez López Allan [1], Pérez Castillo Alfredo Jesús [1], Montiel Gómez German [1]

[1] Tecnológico Nacional De México/Instituto Tecnológico De Apizaco.
División de Estudios de Posgrado e Investigación.
Carretera Apizaco-Tzompantepec, Esquina Con Av. Instituto Tecnológico S/N Conurbado Apizaco-Tzonpantepec, Tlaxcala, Mex.
C.P. 90300, Apizaco, Tlax. Tels: (01241) 417 20 10, Ext. 145

Resumen

En éste artículo se presenta la simulación de un cargador para baterías conectado a la red eléctrica de 120Vca$_{rms}$ para aplicación en bicicletas eléctricas, ésta propuesta es desarrollada mediante la simulación de un convertidor boost utilizado como corrector de factor de potencia y de un convertidor buck, el cual es utilizado para reducir el voltaje del convertidor boost al voltaje necesario para cargar la batería de la bicicleta eléctrica. En la simulación de estos convertidores, son analizadas diferentes técnicas de control, cada convertidor cuenta con un controlador diferente, en el convertidor boost es implementado un control PI digital para el control de voltaje conectado en cascada con otro PI para la corriente, y en el convertidor buck se implanta un doble lazo de control, un controlador PI digital para el voltaje, conectado en casaca con otro PI para la corriente, utilizando una saturación para limitar la corriente en la salida del convertidor.
En la simulación de los convertidores se implementada una técnica de carga de corriente constante-voltaje constante (CC-VC), para su posterior implementación.

Palabras clave: convertidor boost, convertidor buck, cargador para baterías, técnicas de carga, bicicletas eléctricas, factor de potencia.

1. Introducción

Actualmente las baterías con la que cuentan diferentes tipos de vehículos eléctricos tienen una capacidad limitada, por lo que deben de ser recargadas después de haber sido utilizadas. Para hacer viable esta tecnología, la recarga de las baterías debe de ser lo más rápido posible, sin causar ningún tipo de deterioro a la batería y así aumentar su vida útil. Generalmente el proceso de carga cuenta con dos procesos, al primer proceso de recarga se denomina corriente constante, y al segundo se le llama voltaje constante, por lo que a esta técnica de carga se le llama carga a corriente constante-voltaje constante (CC-VC), la mayor parte de la transferencia de energía a la batería ocurre en el periodo de carga a corriente constante, por otro parte la corriente de carga está limitada por la batería, que relativamente acepta pequeños niveles de corriente, dependiendo del tipo de batería que se desee recargar, también, de esto depende el tiempo de carga. En la tabla 1 se muestran algunos parámetros importantes de diferentes tipos de baterías [1] [2] [3].

En esencia un cargador para baterías será dimensionado con relación al tipo, potencia y al tiempo de carga que acepte la batería [1].

Proyectos Académicos de Robótica y Mecatrónica.
ISBN: 978-607-9394-XX-X, Asociación Mexicana de Mecatrónica A.C. – Instituto Tecnológico Superior de Poza Rica, 2017

Tabla 1. Características típicas de baterías recargables [1]

Características	Plomo-Acido	Níquel-cadmio	Níquel-metal hidruro	Li-ion		
				Co	Mn	PO4
Energía especifica (Wh/Kg)	30-50	45-80	60-120	150-190	100-135	90-120
Resistencia interna (mΩ)	<100 en baterías de 12V	100-200 en baterías de 6V	200-300 en baterías de 6V	150-300 baterías de 7.2V	25-75 Por celda	25-50 Por celda
Ciclo de vida (80% DoD)	200-300	1000	300-500	500-1000	500-1000	1000-2000
Tiempo de carga rápida	8-16h	1h típico	2-4h	2-4h	1h o menos	1h o menos
Tolerancia de sobrecarga	Alto	Moderado	Baja	Baja, no puede tolerar carga por goteo		
Autodescarga por mes	5%	20%	30%	<10%		
Voltaje por celda (nominal)	2V	1.2V	1.2V	3.6V	3.8V	3.3V
Corriente de carga pico Mejor resultado	5C 0.2C	20C 1C	5C- 0.5C	<3C <1C	<30C <10C	>30C <10C
Temperatura de operación en descarga	-20 a 60°C	-40 a 60°C	60-90 días	-20 a 60°C		
Mantenimiento	3-6 meses	30-60 días		No requiere		
Requerimientos de seguridad	Térmicamente estable	Térmicamente estable Comúnmente protegidas por fusibles		Protección contra cortocircuitó obligatoria		
Usada desde	1800	1950	1990	1991	1996	2006
Toxicidad	Muy alta	Muy alta	Baja	Baja		

Actualmente existen diferentes tipos de cargadores para baterías, dentro de los cuales encontramos topologías como el convertidor flyback, el cual cuenta con un aislamiento galvánico entre la entrada y salida, debido a sus altas frecuencias de conmutación es posible reducir el tamaño de los dispositivos pasivos, generalmente este tipo de convertidores es utilizado en muchos dispositivos electrónicos suministrados por la red eléctrica, este tipo de convertidores son utilizados para aplicaciones de hasta 250W. Otra de las topologías que es muy utilizada es el convertidor de medio puente, al igual que el convertidor flayback, presenta un aislamiento galvánico entre la entrada y salida, generalmente este tipo de convertidor se utiliza para aplicaciones con una salida de hasta 1kW, este tipo de topologías también se puede encontrar con técnicas de conmutación suave, esto para reducir las pérdidas por conmutación, con esta técnica de conmutación , el semiconductor conmuta cuando el voltaje o la corriente van cruzando por cero es decir, existe la conmutación a corriente cero (ZCS) o a voltaje cero (ZVS), así la disipación de potencia en el semiconductor se ve reducida, alcanzando eficiencias de hasta un 90 %. Otro tipo de convertidor utilizado en fuentes de alimentación en el convertidor forward, este convertidor también presenta una aislación galvánica entre la entrada y la salida, es adecuado para potencias de hasta varios cientos de Watts. Sin embargo, utilizando un convertidor buck, podemos disminuir los dispositivos semiconductores, debido a que solo se necesita dos semiconductores, un diodo y un transistor que cumpla la función de interruptor, también se puede trabajar a altas frecuencias de conmutación para reducir tamaño en los dispositivos pasivos, además de tener una fácil implementación y control.

Proyectos Académicos de Robótica y Mecatrónica.
ISBN: 978-607-9394-XX-X, Asociación Mexicana de Mecatrónica A.C. – Instituto Tecnológico Superior de Poza Rica, 2017

Los cargadores para baterías, por una parte, están dimensionados con base en la capacidad de la fuente principal, es decir, si el voltaje de red es de 120Vca$_{rms}$, 220Vca$_{rms}$ o si son fuentes de energía renovables como la energía fotovoltaica. El tamaño de los dispositivos también va con relación el voltaje y corriente máxima que puedan conducir, así como con la temperatura máxima de operación, En [4], se discute el efecto de las pérdidas en la inductancia y reducción de la fiabilidad de las fuentes de alimentación conmutadas. En [5] se calcula la fiabilidad de los convertidores CD-CD y se presenta una topología óptima desde el punto de vista de la confiabilidad, sin embargo, para efectos de simulación, todos los dispositivos que se utilizan son de forma ideal.

En figura 1 se observa el conjunto de los convertidores CD/CD, el convertidor boost es conectado en seguida del rectificador y el convertidor buck regula el voltaje de carga, el voltaje de carga es para una batería de una bicicleta eléctrica de 37V nominales. La simulación de estas topologías contempla un corrector de factor de potencia (convertidor boost), el cual es utilizado para aplicaciones de medianas potencias y un convertidor buck que provee un voltaje menor al de la entrada.

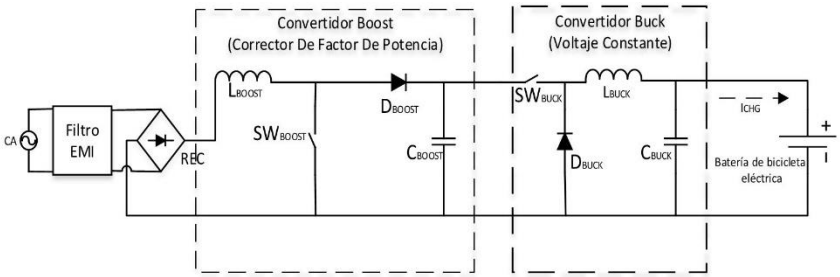


Figura 1. Esquema del cargador para baterías.

Se utilizan diferentes técnicas de control para cada convertidor. Para el control de convertidor boost en modo corrector de factor de potencia, la corriente de entrada es forzada a seguir la forma de onda del voltaje rectificado a la salida del rectificador de puente completo, con lo cual se logra mantener en fase la corriente y voltaje de línea. En el control del convertidor buck se utilizan dos controladores PI digitales conectados en casada con una saturación, el primer controlador PI digital tiene como referencia el voltaje de carga máximo, la saturación y segundo controlador PI digital se utilizan para limitar la corriente de carga.

En la simulación, los convertidores se dimensionan para entregar una potencia máxima 264W, siendo 6A y 44V los parámetros máximos de carga que soporta el paquete de baterías de 270W. En el caso de la batería de la bicicleta eléctrica, no es necesario contar con un algoritmo de carga, debido que dicha bicicleta cuenta con un gestor de batería (BMS, Battery Managmet System), el cual cumple con la función de proteger a la batería de sobre voltajes, sobre corrientes, bajos voltajes, altas temperaturas de operación y balanceo de voltaje en las celdas de la batería [5], por lo que la principal aportación es la integración de los convertidores para aumentar la corriente de carga y la diminución del tiempo que tarde en recargarse la batería.

El documento está dividido en 4 secciones, la primera sección se encuentra la introducción, la segunda sección se describe el funcionamiento del sistema y se las técnicas de control utilizadas en el convertidor boost y buck, así como las ventajas y desventajas que se tiene en los diferentes modos de conducción. En la tercera sección se muestran los resultados de las simulaciones de los convertidores boost y buck, y en la cuarta sección se presentan las conclusiones obtenías a partir de las correspondientes simulaciones.

Proyectos Académicos de Robótica y Mecatrónica.
ISBN: 978-607-9394-XX-X, Asociación Mexicana de Mecatrónica A.C. – Instituto Tecnológico Superior de Poza Rica, 2017

2. Descripción general del sistema

El cargador para baterías está integrado principalmente por un convertidor boost y un convertidor buck. Las normas europeas EN61000-3-2 definen límites para los armónicos de la corriente de línea. Se trata de aparatos que pueden ser vendidos al público en general y tienen una potencia de entrada >75W, algunos valores límite de esta norma se muestran en la tabla 2 [6] [7]. En aplicaciones monofásicas, el convertidor boost es una alternativa para obtener un factor de potencia muy cerca a la unidad, esto debido a sus características naturales donde la corriente del inductor sigue al voltaje de línea [9]. Típicamente en el control del convertidor boost utilizado como corrector de factor de potencia utiliza dos lazos de control conectados en cascada, el primer lazo de control tiene como retroalimentación el voltaje de salida del convertidor boost, este se compara con una referencia externa, en el segundo lazo de control, la salida del primer controlador se multiplica por una componente sinusoidal y se compara con la corriente de la bobina del convertidor boost, esto se hace para forzar a la corriente de línea a entrar en fase con el voltaje. El convertidor buck cumple la función de disminuir el voltaje en la salida del convertidor boost al voltaje de carga máximo y limitar la corriente de carga según tabla 3. Se utiliza el convertidor buck debido a su fácil implementación y bajo costo, comparado con otro tipo de convertidor como el convertidor flyback, el convertidor de medio puente o puente completo, además, el voltaje de salida en el convertidor buck puede variar con solo cambiar el valor de referencia y no es necesario hacer algún cabio en los dispositivos pasivos del convertidor.

Tabla 2. Valores límites de la norma EN61000-3-2

Harmonic-order n	Input Power 75 to 600W Allowable maximum value of harmonic current per Watt (mA/W) / maximum (A)	Input power > 600W maximum value of harmonic current (A)
3	3.4 / 2.30	2.30
5	1.9 / 1.14	1.14
7	1.0 / 0.77	0.77
9	0.5 / 0.4	0.40
11	0.35 / 0.33	0.33

Tabla 3. Rangos de operación del gestor de batería [4] [6] [7]

Tipo de protección	Rangos programados en el circuito integrado según la aplicación	Valores típicos por celda
Detección de alto voltaje	44V	4.4V
Detección de bajo voltaje	23V	2.3V
Detección de altas corrientes de carga, (será limitada según el fabricante de la batería)	6.6 A	2.2 A
Temperatura de operación	Descarga -20 a 75°C Carga <60°C	Descarga -20 a 75°C Carga <60°C
Balanceo de voltaje	N/A	NA

Proyectos Académicos de Robótica y Mecatrónica.
ISBN: 978-607-9394-XX-X, Asociación Mexicana de Mecatrónica A.C. – Instituto Tecnológico Superior de Poza Rica, 2017

Uno de los principales objetivos del cargador para baterías es disminuir el tiempo de carga, esto se logra llevando al limite la corriente de carga que soporta la batería según el fabricante. Aplicando el método de carga a corriente constante-voltaje constante (CC-VC) [12] [13] (Figura 2 c), la corriente de carga no está limitada por la batería en el periodo de corriente constante, por lo que es necesario limitar esta corriente por medio del cargador para que el gestor de batería se mantenga operando en condiciones seguras, es decir, no proteja a la batería de sobre corrientes y el circuito de carga se mantenga cerrado, en la figura 3 se muestra el circuito de protección con el que cuenta el gestor de batería, en la tabla 3 se mencionan las principales características de un gestor de batería.

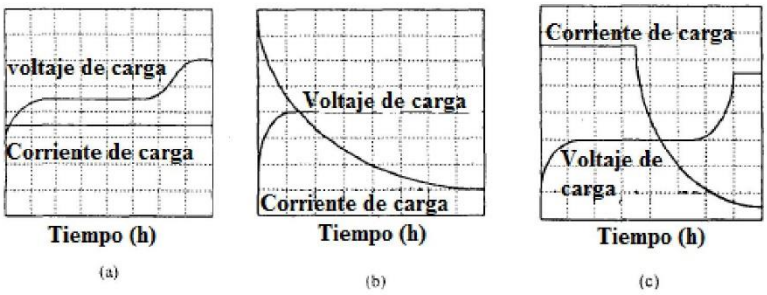


Figura 2. Formas de onda de las diferentes técnicas de carga. (a) corriente constante-voltaje variable (CC-VC), (b) voltaje constante-corriente variable (VC-CV),(c) Corriente constante-voltaje constante (CC-VC) [9]

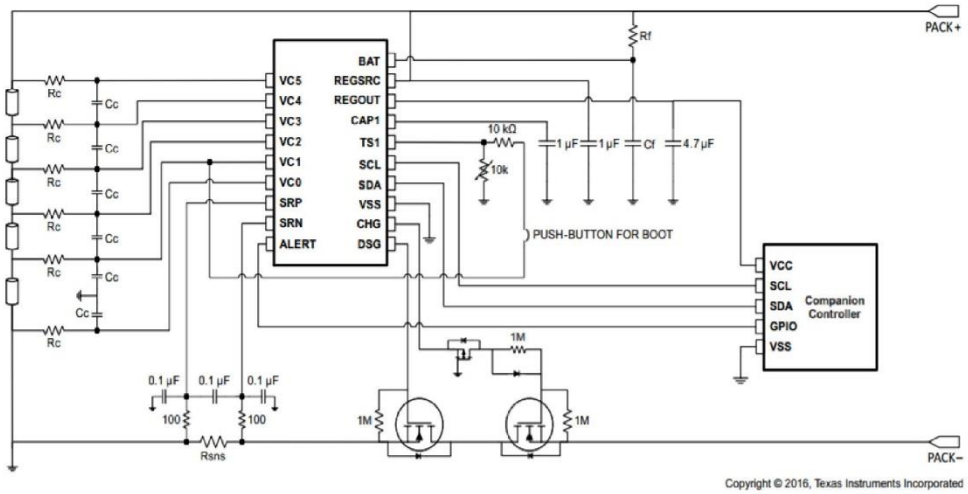


Figura 3. Esquema general de un gestor de batería

2.1 Convertidor buck

Para el diseño del convertidor buck se utilizarán los parámetros de la tabla 3. Este convertidor se puede diseñar para trabajar en modo de conducción continua (MCC) o en modo de conducción discontinua (MCD), en el modo de conducción continua la corriente que fluye por el inductor oscila entre los valores máximos y mínimos, pero nunca llega a cero. Esto se debe a la relación entre el

Proyectos Académicos de Robótica y Mecatrónica.
ISBN: 978-607-9394-XX-X, Asociación Mexicana de Mecatrónica A.C. – Instituto Tecnológico Superior de Poza Rica, 2017

tiempo en el que el interruptor se encuentra cerrado y el tiempo en el que el interruptor necesita estar abierto para que la bobina descargue la energía almacenada durante el tiempo en que el interruptor estuvo cerrado.

En el modo de conducción discontinua (MCD) la intensidad en la carga se hace nula en un momento determinado a lo largo de un intervalo de tiempo T_{off} durante el cual el interruptor está abierto. El tiempo que permanece abierto el interruptor es mayor que el tiempo que puede estar la bobina cediendo energía, con lo que al iniciarse el periodo la intensidad en la carga partirá desde cero.

El convertidor trabajando en modo de conducción discontinua es más propenso a la inestabilidad causado por las cargas, mientras en modo de conducción continua es más sensible a cargas capacitivas [14] [15].

En el diseño del convertidor buck se deben de tener en cuenta las siguientes recomendaciones reconvenciones:

- Cuanto mayor sea el valor elegido del inductor L, menor será el rizo de corriente ΔI_L. Sin embargo, esto resulta en un inductor físicamente más grande y más pesado.
- Tomar en cuenta ΔI_L para que no sea demasiado grande. Con una ondulación de corriente mayor, la ondulación de voltaje de la tensión de salida V_{out} se hace claramente más grande mientras que el tamaño físico del inductor disminuye marginalmente.
- Cuanto mayor sea el valor elegido de la frecuencia de conmutación *f*, menor será el tamaño del inductor. Sin embargo, las pérdidas de conmutación del transistor también aumentan a medida que *f* aumenta.
- El menor tamaño físico posible para el inductor se logra cuando $\Delta I_L = 2I_{out}$ en V_{in_max}. Sin embargo, las pérdidas de conmutación en los transistores están en su nivel más alto en este estado.

El convertidor buck es diseñado con base en los parámetros de carga de la batería de la bicicleta eléctrica, tomando en cuenta lo presentado en las referencias [14] y [15], se hace un análisis del convertidor en modo de conducción continua, teniendo como parámetros de entrada y salida los datos de la tabla 4, donde se obtiene el valor del inductor L=2.86 µH y del capacitor de C=47µH [16] [17] [18].

Tabla 4. Parámetros de entrada y salida del convertidor buck.

Parámetro	Magnitud
Voltaje de entrada	200 VCD
Voltaje de salida	44 VCD
Corriente máxima	6A
Frecuencia de trabajo	20 kHz
Rizo de corriente	10%

2.2 Convertidor boost

El convertidor boost es uno de las más utilizados para operar en modo de corrector de factor de potencia, debido a que requiere menos filtros que le convertidor buck-boost. El convertidor boost como corrector de factor de potencia, al igual que el convertidor buck, puede trabajar en modo de conducción discontinuo o en modo de conducción continua, en modo de conducción discontinuo es

Proyectos Académicos de Robótica y Mecatrónica.
ISBN: 978-607-9394-XX-X, Asociación Mexicana de Mecatrónica A.C. – Instituto Tecnológico Superior de Poza Rica, 2017

uno de los más usados debido a su simplicidad en la implementación y diseño del controlador, pero en aplicaciones de media y alta potencia los componentes de conmutación presentan desgastes. En el modo de conducción continua los algoritmos de control son más complejos y debe ser agregado un lazo de control de corriente para la corrección del factor de potencia, sin embargo, pueden ser alcanzados factores de potencia cercanos a la unidad [19] [20].

Con base en los valores del convertidor buck, podemos calcular los parámetros de diseño para el convertidor boost. Tomando en cuenta lo mencionado en [19] y [120] el convertidor boost es diseñado en el modo de conducción continua, con lo cual se busca obtener un factor de potencia cercano a la unidad. A partir de la ecuación 1, podemos calcular la resistencia de carga que ve el convertidor boost, como la resistencia de carga que ve el convertidor buck es de 7.33 ohm y el ciclo de trabajo es del 22%, sustituyendo estos valores en (1) encontramos el valor de R_i=33.31 ohm, a partir del valor de la carga resistiva, podemos calcular lo valores del convertidor boost, estos los encontramos en la tabla 5, los cálculos del inductor y del capacitor se hacen con base en las referencias [17] [18] [19]. En este tipo de convertidor se hacen las mismas recomendaciones que en convertidor buck.

$$R_i = \frac{V_s}{I_a} = \frac{V_s}{\frac{kV_s}{R}} = \frac{R}{k} \tag{1}$$

Dónde:
R_i= Resistencia equivalente
V_s= Voltaje de entrada
I_a= Corriente Promedio
k= Ciclo de trabajo
R= Resistencia de carga en el convertidor buck

Tabla 5. Parámetros de diseño del convertidor boost

Parámetro	Magnitud
Voltaje de entrada	169.7 VCD
Voltaje de salida	200 VCD
Corriente máxima	6.006A
Frecuencia de trabajo	20 kHz
Rizo de corriente	10%
Inductor	1.84 mH
Capacitor	220 µF

2.3 Control de convertidores

Las topologías utilizadas en el control del convertidor boost y convertidor buck, se puede observar en las figuras 4 y 5 respectivamente. En fuentes de alimentación conmutadas, normalmente el circuito de corrección de factor de potencia se coloca enseguida del puente rectificador, y puede trabajar como una unidad independiente al resto del sistema de la fuente de alimentación, pero es una parte del sistema, es decir, siempre se encuentra conectado físicamente. Para corregir el factor de potencia el método más simple es utilizar circuitos pasivos, los cuales, trabajan bien en potencias

Proyectos Académicos de Robótica y Mecatrónica.
ISBN: 978-607-9394-XX-X, Asociación Mexicana de Mecatrónica A.C. – Instituto Tecnológico Superior de Poza Rica, 2017

bajas. Sin embargo, los circuitos de una sola etapa como el convertidor boost, solucionan el problema del factor de potencia para niveles de potencia medios.

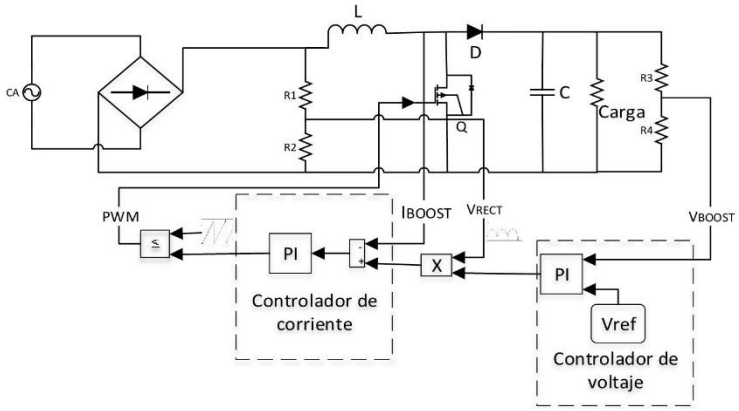


Figura 4. Control del convertidor boost; corrector de factor de potencia en modo de conducción continua.

Este tipo de convertidor CD/CD, como se dijo en la subsección 2.2, es diseñado en modo de conducción continua, debido a las ventajas que ya se mencionaron, en la topología de control de la figura 4 se puede observar un controlador PI digital para el voltaje, en cascada con otro controlador PI para la corriente, la salida del primer controlador debe de ser multiplicado por una componente sinusoidal, siendo el productor de estas dos señales la referencia de controlador de la corriente, esto para forzar a la corriente de línea a entrar en fase con el voltaje de línea, con esto se busca tener un factor de potencia cercano a la unidad.

En la figura 5 se puede ver la topología de control del convertidor buck, que al igual que el convertidor boost, está diseñado para trabajar en modo de conducción continua, La función básica de conversión de CD a CD se logra mediante el control de la fracción del tiempo de activación del transistor, o con relación al ciclo de trabajo k (0 <k <1) con una frecuencia de conmutación constante, en la imagen se puede apreciar dos controladores PI digital, el primero es el del control de voltaje, conectado en cascada con una saturación, la salida de la saturación se usa como referencia para el controlador de la corriente, con lo cual se limita la corriente que demanda la batería en el proceso de carga.

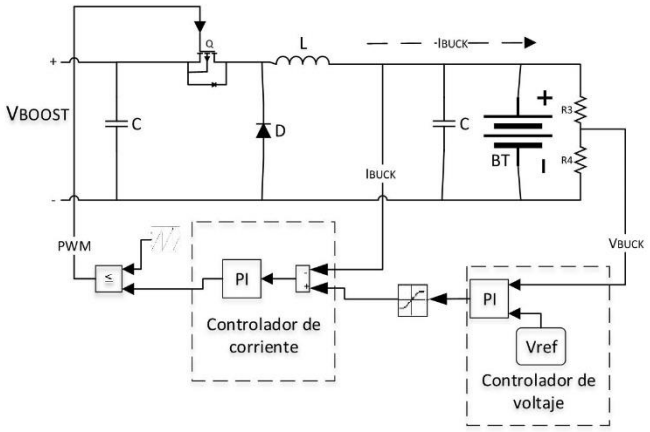


Figura 5. Esquema de control del convertidor reductor

Proyectos Académicos de Robótica y Mecatrónica.
ISBN: 978-607-9394-XX-X, Asociación Mexicana de Mecatrónica A.C. – Instituto Tecnológico Superior de Poza Rica, 2017

3 Resultados

En la simulación del cargador para baterías se supone que la batería de la bicicleta eléctrica tiene 0% de carga, según la tabla 3, el voltaje mínimo de la batería cuando el gestor de carga reconoce que la batería está descargada al 100% es de 23V. La simulación de la batería se hace mediante un componente capacitivo, el cual almacena la energía, y con una componente resistiva conectada en serie que es la resistencia interna de la batería, en otros modelos eléctricos, se le integran redes RC, conectadas en serie, las cuales simulan las componentes exponenciales en las gráficas de carga y descarga de la batería, generalmente estos se utilizan para aplicaciones de precisión, donde se desea saber el estado de carga real en la batería [21] [12] .

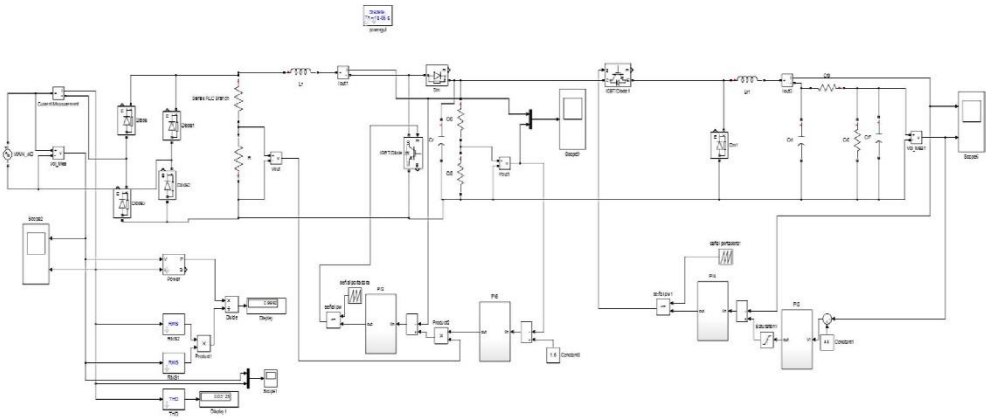


Figura 6. Simulación de topologías utilizadas en cargadores para baterías.

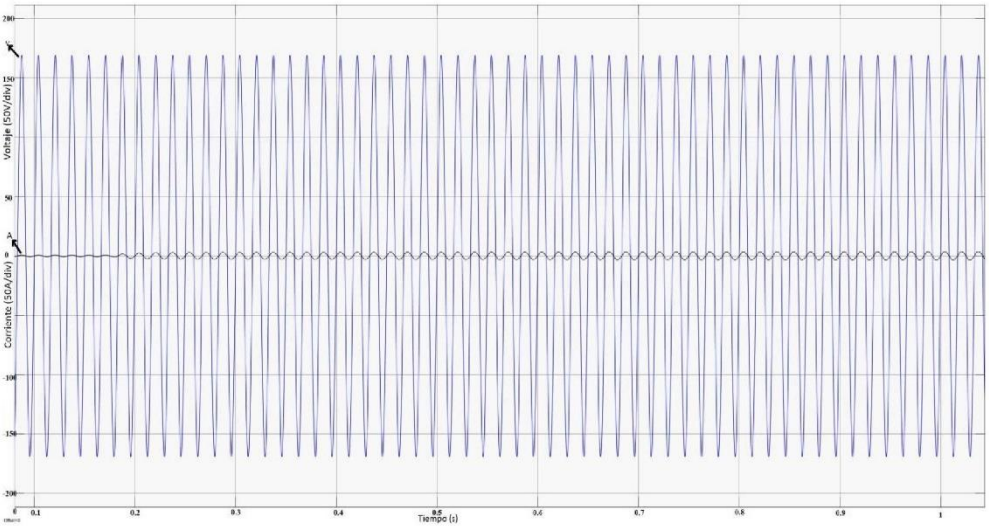


Figura 7. Formas de onda de la corriente y voltaje de línea

Proyectos Académicos de Robótica y Mecatrónica.
ISBN: 978-607-9394-XX-X, Asociación Mexicana de Mecatrónica A.C. – Instituto Tecnológico Superior de Poza Rica, 2017

En la figura 7 se puede observar las formas de onda de las señales sinusoidales, como se puede ver el nivel de la amplitud de la corriente tiene diferentes niveles, esto se debe a las variaciones de la carga, la forma de onda va aumentando gradualmente, con respecto al consumo de energía de la batería, si bien, al inicio se encuentra descargada al 100% debería de consumir más energía, pero como la corriente de carga es limitada por el convertidor buck y el voltaje en la batería es el mínimo, entonces la potencia que consume es mayor conforme aumenta el voltaje en la batería. Estas variaciones de corriente también se pueden observar en la figura 9, en donde se presentan las formas de onda en la salida del convertidor boost, la forma de onda de color azul representa el voltaje y en negro la corriente de línea.

En la figura 6 se muestran los valores del factor de potencia y de distorsión armónica total, como se mostró en la figura 7 la corriente de línea es forzada a entrar en fase con el voltaje de línea, como se puede ver el valor del factor de potencia es de 0.9997 lo cual es encuentra muy cerca a la unidad, y tiene una baja distorsión armónica, que se encuentra en los valores de 0.7 %.

Las formas de onda de la figura 8 corresponden a la corriente en la bobina del convertidor boost (negro), como se puede ver es una onda rectificada, la cual está en fase con el voltaje rectificado, pero con diferente amplitud, la línea azul corresponde al voltaje en el divisor resistivo, el cual representa el voltaje en el capacitor del convertidor boost.

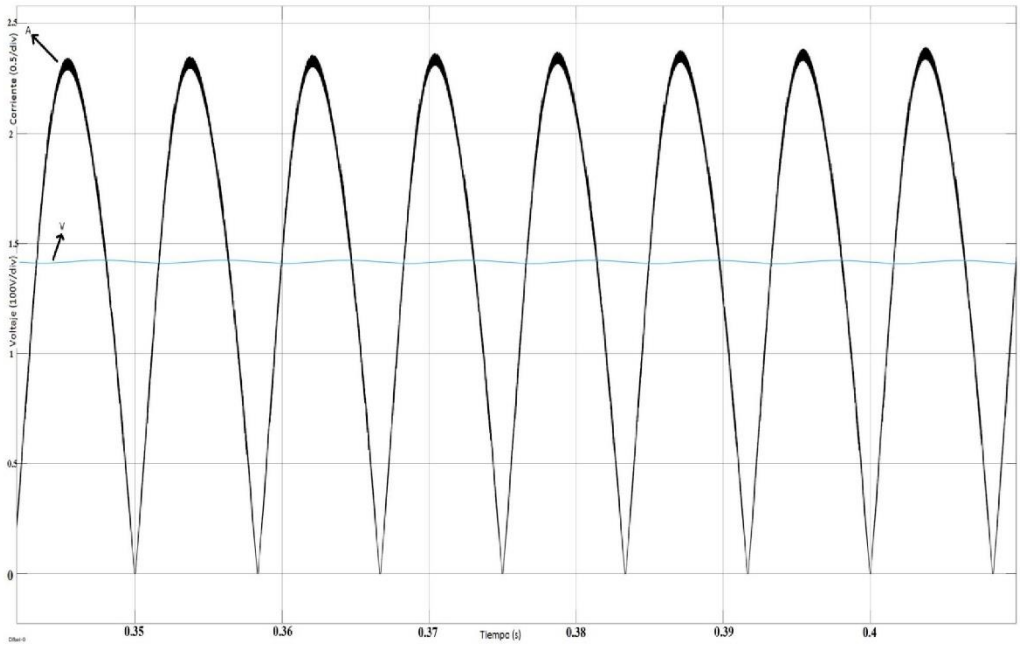


Figura 8. Formas de onda del voltaje y la corriente en la salida del convertidor boost

En la figura 9 se muestran las formas de onda del voltaje y la corriente en el convertidor buck, en esta imagen se puede observar el primer periodo de carga, donde la corriente se mantiene constante y el voltaje en la batería va aumentando gradualmente, el segundo periodo de carga es cuando el voltaje en la batería se mantiene constante y la corriente de carga comienza a disminuir, hasta que llega a un valor donde solamente consume la corriente que suple su autodescarga. En el caso de al batería de la bicicleta eléctrica, si la corriente de carga es mayor a 6.6A o el voltaje mayor a 44V, el circuito de la figura 3 deja de conducir, con el fin de proteger a la batería de sobre corrientes.

ANEXO 3 CÓDIGO PAGINA WEB

```
<DOCTYPE>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Sensor by Alfred</title>
<meta name="keywords" content="" />
<meta name="description" content="" />
<link href="http://fonts.googleapis.com/css?family=Open+Sans:400,300,600,700|Archivo+Narrow:400,700" rel="stylesheet" type="text/css">
<link href="default.css" rel="stylesheet" type="text/css" media="all" />
</head>
<body>
<div id="wrapper-bg">
        <div id="wrapper">
                <div id="header" class="container">
                        <div id="logo">
                                <h1><a href="index.html"><span>TECNM</span> ITA2018</a></h1>                        </div>
                        <div id="menu">
                                <ul>
                                        <li class="active"><a href="index.html" accesskey="1" title="">Principal</a></li>
                                        <li><a href="mediciones.html" accesskey="2" title="">Mediciones</a></li>
                                        <li><a href="galeria.html" accesskey="3" title="">Galeria</a></li>
                                        <li><a href="trabajosfuturos.html" accesskey="4" title="">Trabajos Futuros</a></li>
                                        <li><a href="#" accesskey="5" title=""></a></li>
                                </ul>
                        </div>
                </div>
                <div id="banner" class="container"> <img src="images/header-photo.jpg" width="1200" height="320" alt="" /> </div>
                <div id="three-column" class="container">
```

```
                <header>
                    <h2>Sensor adxl345 por SPI</h2>
                </header>
                <div class="tbox1">
                    <div class="box-style box-style01">
                        <div class="content"> <a href="#" class="image image-full"><img src="images/img01.jpg" alt=""></a>
        <h2>MI</h2>
                            <p>Los motores de induccion (MMII), son maquinas capaces de convertir la energia electrica en energia mecánica.</p>
                            <a href="#" class="button-style">Saber mas</a> </div>
                    </div>
                </div>
                <div class="tbox2">
                    <div class="box-style box-style02">
                        <div class="content"> <a href="#" class="image image-full"><img src="images/img02.jpg" alt=""></a>
                            <h2>Zedboard</h2>
                            <p>Tarjeta de desarrollo FPGA </p>
                            <a href="#" class="button-style">Saber mas</a> </div>
                    </div>
                </div>
                <div class="tbox3">
                    <div class="box-style box-style03">
                        <div class="content"> <a href="#" class="image image-full"><img src="images/img03.jpg" alt=""></a>
                            <h2>Software</h2>
                            <p>Tarjeta Zedboard ejecutando Linux embebido</p>
                            <a href="#" class="button-style">Saber mas</a> </div>
                    </div>
                </div>
            </div>
```

```
            <div id="page">
                <div id="content"></div>
                <div id="sidebar"></div>
            </div>
        </div>
        <div id="footer-content" class="container">
            <div id="fbox1">

            </div>
            <div id="fbox2">
                <h2>Acerca de</h2>
                <p><strong></strong>“Monitoreo de señales de vibración de un motor de inducción mediante FPGA con sistema operativo embebido Linux usando el Bus SPI”</p>
                <a href="#" class="button-style">Saber mas</a> </div>
            <div id="fbox3">
                <h2>Contacto</h2>
                <p></p>
                <ul class="style5">
                    <li class="first"><span class="address"></span> <span class="address-01"><br /> </span> </li>
                    <li> <span class="mail">Correo</span> <span class="mail-01"><a href="#">alfredki3@hotmail.com</a></span> </li>
                    <li> <span class="phone">Phone</span> <span class="phone-01">(55) 41768274</span> </li>
                </ul>
            </div>
        </div>
    </div>
    <div id="footer">
        <p>&copy; Creado por <a href="http://templated.co" rel="nofollow">Alfredo</a>
    </div>
    </body>
    </html>
```

ANEXO 4 PROGRAMA EN "C" MAIN.C

```
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <errno.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <stdint.h>
int GetAccelData();
int fdw, fdr;
int main (int argc, char **argv)
{
  int i;
  int j; j=1000;
  int val;
  int x, y;
  float theta;
   fdw = open ("/dev/xillybus_mem_8", O_WRONLY);
  fdr = open ("/dev/xillybus_mem_8", O_RDONLY);
  SpiWrite(0x31, 0x00); // 2g
  SpiWrite(0x2d, 0x08); // StartM
  printf("Iniciando mediciones del sensor\n");
  printf("X0X1 Y0Y1 Z0Z1\n");
  while(j--)
{
    val = GetAccelData();
                }
  close(fdw);
  close(fdr);
  return 0;
}
```

```
void SpiWrite(unsigned char addr, unsigned char data) {
  WritePL(fdw, 0x00, addr); // SPI address
  WritePL(fdw, 0x01, data); // SPI write data
  WritePL(fdw, 0x02, 0x01); // SPI start flag set
  _wait(30000);
  WritePL(fdw, 0x02, 0x00); // SPI start flag clear
  _wait(30000);
}
void WritePL(int fd, unsigned char addr, unsigned char data) {
  if (lseek(fd, addr, SEEK_SET) < 0) {
    perror("Failed to seek");
    exit(1);
  }
  allwrite(fd, &data, 1);
}
unsigned char SpiRead(unsigned char addr) {
  unsigned char res;
  unsigned char readaddr;

  readaddr = 0x80 + addr;
  WritePL(fdw, 0x00, readaddr); // SPI address
  WritePL(fdw, 0x01, 0xFF); // SPI dummy write data
  WritePL(fdw, 0x02, 0x01); // SPI start flag set
  _wait(30000);
  WritePL(fdw, 0x02, 0x00); // SPI start flag clear
  res = ReadPL(fdr, 0x03);
  _wait(30000);
  return res;
}
unsigned char ReadPL(int fd, unsigned char addr) {
  unsigned char data;
```

```
  if (lseek(fd, addr, SEEK_SET) < 0) {
    perror("Error solicitud ");
    exit (1);
  }
  allread(fd, &data, 1);
  return data;
}
int GetAccelData() {
  int res;
  devid = SpiRead(0x00); // R Device ID
  zdata_l = SpiRead(0x32); // Read XL
  printf("%d,",zdata_l);
  zdata_h = SpiRead(0x33); // Read XH
  printf("%d ",zdata_h);
  xdata_l = SpiRead(0x34); // Read yL
  printf("%d,",xdata_l);
  xdata_h = SpiRead(0x35); // Read yH
  printf("%d ",xdata_h);
  ydata_l = SpiRead(0x36); // Read zL
  printf("%d,",ydata_l);
  ydata_h = SpiRead(0x37); // Read zH
  printf("%d\n",ydata_h);
  tmpshort = ((unsigned short)(zdata_h) << 8) + (unsigned short)(zdata_l);
  if(tmpshort < 32768) res = (int)(tmpshort);
  else res = (int)(tmpshort) - 65536;
  return res;
}
void allwrite(int fd, unsigned char *buf, int len) {
  int sent = 0;
  int rc;
  while (sent < len) {
```

```
    rc = write(fd, buf + sent, len - sent);
    if ((rc < 0) && (errno == EINTR))
      continue;
    if (rc < 0) {
      perror("errorW");
      exit(1);
    }
    if (rc == 0) {
      fprintf(stderr, "errorW\n");
      exit(1);
    }
    sent += rc;
  }}
int _wait(int loop_count)
{
  volatile int sum, data;
  sum = 0;
  for(data = 0; data < loop_count; data++) {
    sum = (data << 8);
  }
  return sum;
}
void allread(int fd, unsigned char *buf, int len) {
  int received = 0;
  int rc;
  while (received < len) {
    rc = read(fd, buf + received, len - received);
    if ((rc < 0) && (errno == EINTR))
      continue;
    if (rc < 0) {
      perror("error");
```

```
    exit(1);
  }
  if (rc == 0) {
    fprintf(stderr, "EOF (?!)\n");
    exit(1);
  }
  received += rc;
}}
```

ANEXO 5 INSTALACIÓN DE WEB SERVER APACHE

En este anexo se muestran los pasos a seguir para conseguir instalar el servidor web Apache en un sistema operativo Linux, específicamente una distribución Ubuntu, los pasos son los siguientes:

Abrir una terminal y teclear lo siguiente:

```
alfredell@alfredell-VirtualBox:/$ sudo apt-get update
```

Con esto se realiza una actualización de los paquetes que están disponibles.

Después se procede a teclear el siguiente comando para instalar apache.

```
alfredell@alfredell-VirtualBox:/$ sudo apt-get install apache2
```

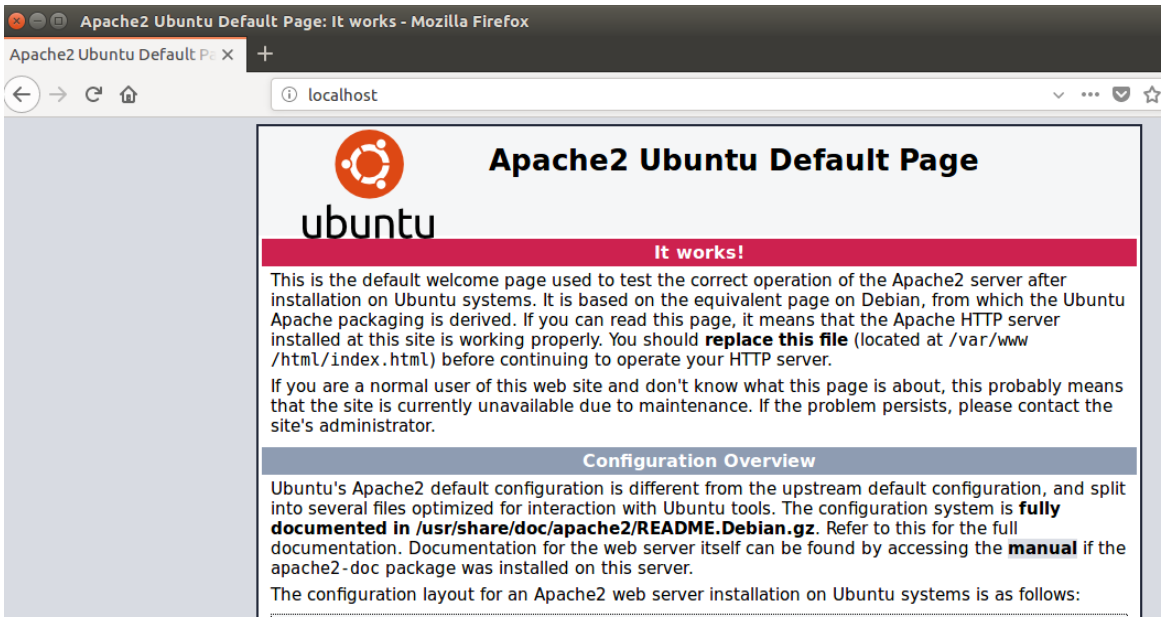
Posteriormente se despliega la siguiente pantalla para la cual hay que teclear la letra “Y”

```
alfredell@alfredell-VirtualBox:/$ sudo apt-get install apache2
[sudo] password for alfredell:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  linux-headers-4.13.0-36 linux-headers-4.13.0-36-generic linux-image-4.13.0-36-generic linux-image-extra-4.13.0-36-generic
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils libapr1 libaprutil1 libaprutil1-dbd-sqlite3 libaprutil1-ldap liblua5.1-0
Suggested packages:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom
The following NEW packages will be installed:
  apache2 apache2-bin apache2-data apache2-utils libapr1 libaprutil1 libaprutil1-dbd-sqlite3 libaprutil1-ldap liblua5.1-0
0 upgraded, 9 newly installed, 0 to remove and 188 not upgraded.
Need to get 1 542 kB of archives.
After this operation, 6 373 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

Iniciará la descarga e instalación:

```
Setting up libapr1:amd64 (1.5.2-3) ...
Setting up libaprutil1:amd64 (1.5.4-1build1) ...
Setting up libaprutil1-dbd-sqlite3:amd64 (1.5.4-1build1) ...
Setting up libaprutil1-ldap:amd64 (1.5.4-1build1) ...
Setting up liblua5.1-0:amd64 (5.1.5-8ubuntu1) ...
Setting up apache2-bin (2.4.18-2ubuntu3.8) ...
Setting up apache2-utils (2.4.18-2ubuntu3.8) ...
Setting up apache2-data (2.4.18-2ubuntu3.8) ...
Setting up apache2 (2.4.18-2ubuntu3.8) ...
Enabling module mpm_event.
Enabling module authz_core.
Enabling module authz_host.
Enabling module authn_core.
Enabling module auth_basic.
Enabling module access_compat.
Enabling module authn_file.
Enabling module authz_user.
Enabling module alias.
Enabling module dir.
Enabling module autoindex.
Enabling module env.
Enabling module mime.
Enabling module negotiation.
Enabling module setenvif.
Enabling module filter.
Enabling module deflate.
Enabling module status.
Enabling conf charset.
Enabling conf localized-error-pages.
Enabling conf other-vhosts-access-log.
Enabling conf security.
Enabling conf serve-cgi-bin.
Enabling site 000-default.
Processing triggers for libc-bin (2.23-0ubuntu10) ...
Processing triggers for systemd (229-4ubuntu21.1) ...
Processing triggers for ureadahead (0.100.0-19) ...
Processing triggers for ufw (0.35-0ubuntu2) ...
alfredell@alfredell-VirtualBox:/$
```

Después de un par de minutos terminará la instalación y se procede a corroborar que esté funcionando correctamente, para lo cual, se abre una ventana de algún explorador web y se teclea en la barra de direcciones el comando “localhost” y si todo está funcionando correctamente saldrá lo siguiente en la página del explorador:



Para configurar las opciones del servidor se tiene que editar un archivo el cual se encuentra en la siguiente ruta:

`/etc/apache2/apache2.conf`

Cabe señalar que los portales que se desarrollen deben ir dentro de la siguiente ruta.

`/var/www/`