Artículo **Revista de Sistemas y Gestión Educativa**
Diciembre 2015 Vol.2 No.5 1003-1009

Metodología para la enseñanza de sistemas digitales mediante lenguaje ensamblador

BAUTISTA, Jorge*†, ROJAS, Carlos y LÓPEZ, Asdrubal

Recibido 19 de Octubre, 2015; Aceptado 10 de Diciembre, 2015

Resumen

El presente trabajo propone la metodología para impartir las Unidades de Aprendizaje Ensambladores y Lenguaje ensamblador en el Programa Educativo de Ingeniero en Computación del Centro Universitario UAEM Zumpango de la Universidad Autónoma del Estado de México, destacando que para este propósito se emplea el lenguaje ensamblador de los microcontroladores (uC) PIC de la familia 16, en específico se emplean el PIC 16F628 y 16F887 con el software del fabricante Microchuip[MR] (MPLAB). Se destaca que los programas para ambos dispositivos son similares teniendo especial cuidado en la configuración de los puertos y considerando las capacidades de pines de entrada y salida.

Domótica, Microcontrolador, Control, Remoto

Abstract

This paper proposes the methodology to teach the Assemblers and Assembly Language subjets on the Education Program in Computer Engineering from the UAEM Zumpango Campus of the Autonomous University of the State of Mexico, noting that for this purpose the assembly language is used microcontrollers (uC) PIC 16 family, specifically the PIC 16F628 and 16F887 are used with the software manufacturer Microchuip[MR] (MPLAB). It is emphasized that the programs are similar for both devices being careful configuration of ports and considering the capabilities of input and output pins.

Home Automation, Microcontroller, Control, Remote

Citación: BAUTISTA, Jorge, ROJAS, Carlos y LÓPEZ, Asdrubal. Metodología para la enseñanza de sistemas digitales mediante lenguaje ensamblador. Revista de Sistemas y Gestíon Educativa 2015. 2-5: 1003-1009

* Correspondencia al Autor (Correo Electrónico: jbautistal@uaemex.mx)
† Investigador contribuyendo como primer autor

Introducción

Desde hace un par de décadas la mayoría de los sistemas digitales pasaron a formar parte de los sistemas embebidos, mediante el empleo de circuitos contenidos en una solo pastilla ya sea mediante PLD´s (Dispositivo Lógico Programable) o microcontroladores. Los primeros se programan mediante lenguajes de descripción de hardware y los segundos mediante lenguajes de bajo y alto nivel. El lenguaje empleado para programar los uC en la institución mencionada es el lenguaje ensamblador, ya que forma parte del plan de estudios de la Licenciatura en Ingeniero en Computación.

Como se mencionó una de las opciones para el diseño de los sistemas embebidos es mediante los Lenguajes de Descripción de Hardware (HDL) empleando PLD´s, CPLD´s (Dispositivo Lógico Programable Complejo) y FPGA´s (del inglés Field Programmable Gate Array). La otra forma de diseñar los sistemas embebidos es mediante el lenguaje de bajo nivel conocido como Lenguaje Ensamblador la cual es una Unidad de Aprendizaje que junto con Ensambladores son impartidas en el programa de Ingeniero en Computación de la UAEM.

Cabe mencionar que existe una gran gama de lenguajes que dependerán del microprocesador o microcontrolador empleado, además del fabricante y de la arquitectura de cada uno de ellos pudiendo ser CISC (Complex Instruction Set Computer) o RISC (Reduced instruction set computing).

La evolución de la implementación de los Sistemas Digitales y embebidos se debió gracias al enorme crecimiento de la tecnología de circuitos integrados.

La cual hasta nuestros días sigue cumpliendo con la ley de Moore, mencionada por Gordon Moore gerente de Intel Corporation en 1965 que planteaba que el número de transistores en los circuitos integrados se duplicaba cada 2 años [1].

A continuación de muestran los tamaños comparativos de los CI (Tabla 1) para los años 2001 y 2012 según la SIA [1].

	2001	2012
Longitud de compuerta de transistor	0.12 µm	35 nm
Transistores por cm^2	16 millones	100 millones
Tamaño de chip	850 mm^2	1300 mm^2

Tabla 1 Muestra de la guía SIA

El impacto de los sistemas digitales es tal que los encontramos en cualquier parte de nuestro quehacer cotidiano por ejemplo: electrodomésticos, control electrónico de un automóvil, instrumentación electrónico, redes de sensores para monitoreo y vigilancia, dispositivos portátiles como lo son: teléfonos celulares y PDA´s (Asistente Digital Personal).

El propósito del presente trabajo es dar a conocer la metodología empleada en las unidades de aprendizaje de Lenguaje ensamblador y Ensambladores en el Centro Universitarios Zumpango de la Universidad Autónoma del Estado de México.

Fundamentos

El diseño del sistema embebido se ocupa del diseño de los sistemas electrónicos digitales tales como: circuitos integrados (CI), microcontroladores, procesadores digitales de señales, computadoras, sistemas de comunicaciones, entre otros, que en esencia conforman el hardware digital.

ISSN-2410-3977

Los sistemas digitales se diseñan haciendo uso de la lógica programable o lenguaje ensamblador, siendo una forma más rápida y directa de integrar aplicaciones, permitiendo independizar el proceso de fabricación del proceso de diseño fuera de la fábrica de semiconductores, además de integrar aplicaciones y desarrollos lógicos mediante el empaquetamiento de soluciones en un CI.

Como se mencionó para la enseñanza del lenguaje ensamblador se emplean los microcontroladores el cual es un sistema cerrado que contiene una computadora completa y de prestaciones limitadas que no se pueden modificar, capaz de ejecutar las órdenes grabadas en su memoria.

Un uC incluye en su interior tres principales unidades (Figura 1) funcionales: unidad central de procesamiento, memoria, y periféricos.

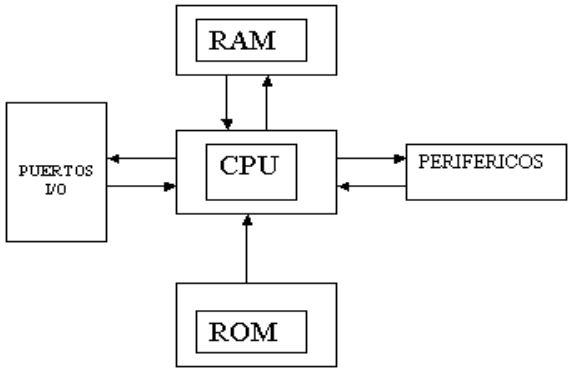

Figura 1 Bloques de un Microcontrolador

Los fabricantes principales se encuentran: Atmel, Freescale (antes motorola), Holtek, Intel, National semiconductor, Microchip, NXP semiconductor, Renesas (Antes HITACHI, Mitsubishi, NEC), STMicroelectronics, Texas Intruments, Zilog, entre otros.

Lenguaje ensamblador

Fue desarrollado en los años de 1950, cuando fueron referidos como lenguajes de programación de 2ª generación. Los lenguajes ensambladores están basados en los mnemónicos que simbolizan los pasos de procesamiento de los registros de un procesador, que fueron utilizados por los sistemas operativos IBM PC DOS [5].

Los lenguajes de bajo nivel son lenguajes totalmente dependientes de la máquina, formado por abreviaturas de letras y números llamadas mnemotécnicos. Los mnemónicos son un lenguaje en el que cada enunciado produce exactamente una instrucción máquina y tienen acceso a todas las características e instrucciones disponibles en la máquina, ya sea para computadoras, microprocesadores o microcontroladores. Además es una representación simbólica de los códigos de máquinas binarias, en otras palabras es la primera abstracción del lenguaje máquina, que consiste en asociar los OPCODE (códigos de operación) con palabras clave que sean fáciles de recordar para el programador (Tabla 2).

Un programa en lenguaje ensamblador traduce el o convierte el código fuente (ensamblador) a código objeto (lenguaje máquina).

Dirección de memoria	Código Máquina	Lenguaje ensamblador (mnemonicos)	
0CFD:0100	BA0B01	MOV	DX,010B
0CFD:0103	B409	MOV	AH,09
0CFD:0105	CD21	INT	21
0CFD:0107	B400	MOV	AH,00
0CFD:0109	CD21	INT	21

Tabla 2 Ejemplo de código máquina y mnemónicos

ISSN-2410-3977

En la actualidad manejamos lenguajes de alto nivel que son relativamente sencillos en comparación con el lenguaje máquina. Empero el lenguaje ensamblador es importante porque es considerado de primera generación y a partir de él se derivaron todos los demás lenguajes hasta llegar a los de alto nivel. Es por ello que el lenguaje ensamblador:

- Es directamente traducible al Lenguaje de máquina, y viceversa.
- La computadora no entiende directamente al Lenguaje Ensamblador; es necesario traducirle a Lenguaje de Máquina.
- Se utilizan traductores que convierten el código fuente (en Lenguaje Ensamblador) a código objeto, con el fin de facilitar la programación y tener el control del Hardware.

Dentro de las características de este lenguaje se encuentran [3]:

1. Velocidad
   Un intérprete es siempre más lento que realizar la misma acción en Lenguaje Ensamblador.
   Los compiladores son mucho más rápidos que los intérpretes, pues hacen la traducción una vez y dejan el código objeto.
2. Tamaño
   Existen programas donde el uso de la memoria es crítico, para esos casos es eficiente el lenguaje ensamblador por la mínima cantidad de recursos de los que dispone.
3. Flexibilidad
   Los lenguajes de alto nivel tienen limitantes en el control; al hacer abstracciones, limitan su propia capacidad.
   En cambio, en ensamblador es sumamente sencillo, pues tenemos el acceso directo al hardware del monitor.
4. Tiempo de programación
   Requiere más instrucciones para realizar el mismo proceso.
5. Programas fuente grandes
   Requerimos más instrucciones primitivas para describir procesos equivalentes. Esto es una desventaja porque dificulta el mantenimiento de los programas.
6. Peligro de afectar recursos
   El problema es que todo error que podamos cometer, o todo riesgo que podamos tener, podemos tenerlo también en este Lenguaje. Dicho de otra forma, tener mucho poder es útil pero también es peligroso.
7. Falta de portabilidad
   Existe un lenguaje ensamblador para cada máquina; por ello, evidentemente no es una selección apropiada de lenguaje cuando deseamos codificar en una máquina y luego llevar los programas a otros SO.

El código interno para cada instrucción puede ser enlistado en:

- Binario
- Octal
- Hexadecimal

Cuando se escribe un programa para una computadora, microprocesador o microcontrolador se asigna a cada instrucción un nombre simbólico para identificarlo. Las instrucciones del microprocesador pueden clasificarse en tres tipos[5]:

- **Instrucciones de Transferencia**
  Mueven datos entre registros, palabras de memoria, y registros sin cambiar el contenido de la información binaria.
- **Instrucciones de Operación**
  Realizan operaciones con los datos almacenados en los registros o palabras de memoria.
- **Instrucciones de Control:**
  Prueban el estado de las codificaciones en los registros y causar un cambio en la secuencia del programa dependiendo de los resultados.

ISSN-2410-3977

Por lo tanto el conjunto de instrucciones de un microprocesador en particular especifica las operaciones de transferencia entre registros y decisiones de control que están presentes en el sistema del microcomputador. Un programa específico para un microcomputador es equivalente a especificar la secuencia de operaciones para un sistema digital particular.

Metodología

Dentro del proceso de diseño para cualquier dispositivo se puede considerar varias etapas, las cuales pueden variar dependiendo del individuo y de su experiencia, por lo que podemos generalizar mediante un diagrama de flujo dicho procedimiento (Figura 2).

Para el diseño de los sistemas digitales y embebidos es de suma importancia los conceptos fundamentales tales como: lógica binaria, manejo de direccionamiento directo e indirecto, y de manera general lógica combinatoria y secuencial, sistemas digitales y arquitectura de computadoras. Es por ello que en la UAEM se tocan los aspectos básicos así como el diseño modular [2].

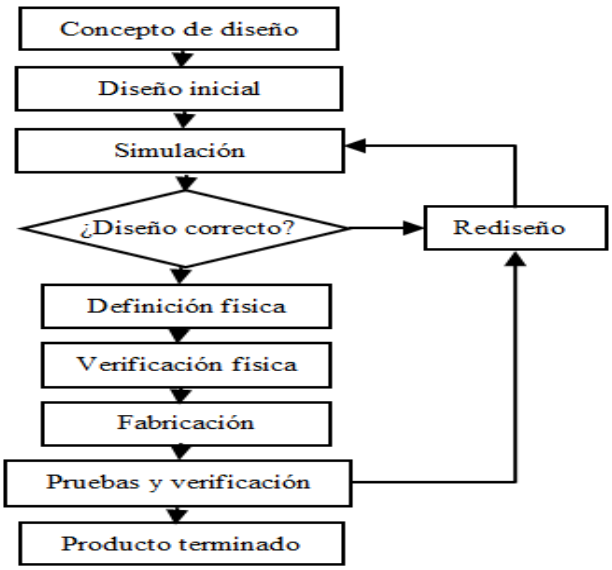

Figura 2 Proceso de diseño

Primeramente en el diseño del sistema digital se plantea la problemática (Figura 3a) con las características requeridas del sistema, posteriormente se lleva acabo el análisis del sistema (Figura 3b) mediante los conceptos fundamentales (solo en las primeras prácticas para conocer las bases), posteriormente se plantea un diagrama a bloques o funcional del sistema (Figura 3c), seguido de un diagrama de flujo (Figura3d) y simulación del sistema para finalmente grabar el archivo .HEX (Figura 3e) en el microcontrolador e implementación en protoboard (Figura 3f).

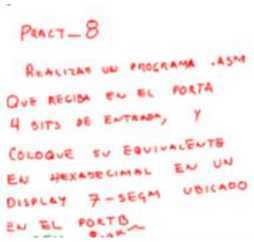

Figura 3a

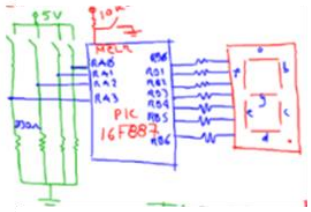

Figura 3b

Figura 3c

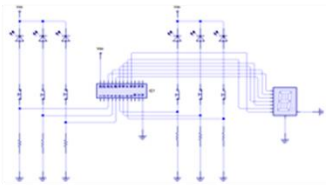

Figura 3d

Figura 3e

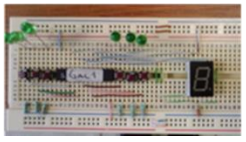

Figura 3f

Figura 3 Pasos para la implementación del sistema digital

Los pasos para programar el PIC en lenguaje ensamblador de manera simplificada son:

1. Creación del proyecto (Figura 4)
2. Selección del dispositivo (Figura5)
3. Programación (Figura 6)
4. Compilación
5. Simulación
6. Verificación de los pines asignados y porcentaje de utilización del dispositivo

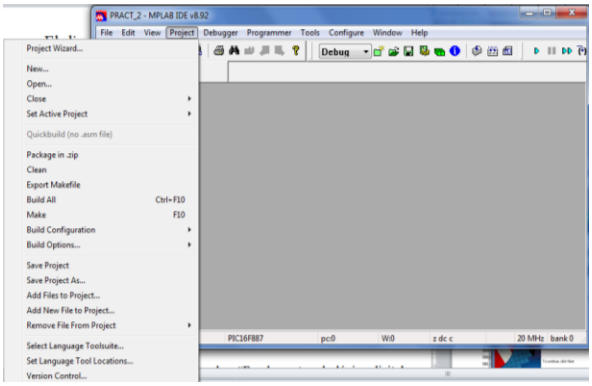

Figura 4 Creación del proyecto

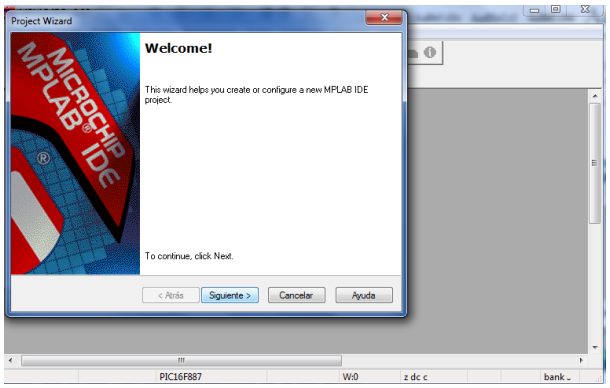

Figura 5 Selección del dispositivo

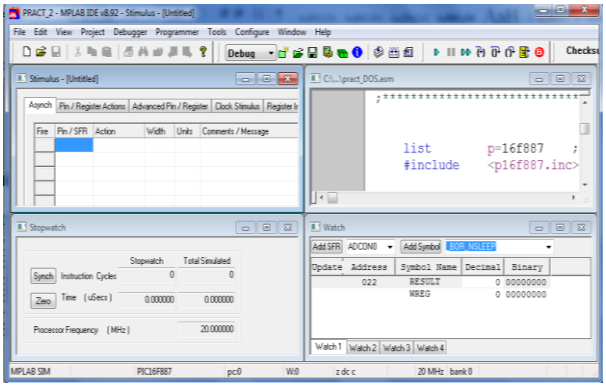

Figura 6 Programación, compilación y simulación

Conclusiones

El diseño de los sistemas digitales se ha modificado en las 2 últimas décadas por lo que es de suma importancia que en las instituciones de nivel superior donde se impartan las asignaturas o unidades de aprendizaje relacionadas a estos, se aborden con herramientas actuales sin olvidar la base de tal conocimiento.

En el Centro Universitarios de Zumpango la enseñanza del lenguaje ensamblador ha permitido comprender la forma de diseñar los sistemas embebidos que forman parte de los sistemas digitales. Por ello es indispensable actualizarse en el uso de las herramientas y aprovechar los beneficios que estas ofrecen para potencializar las aplicaciones.

Finalmente el lenguaje ensamblador cambio la forma de programar los micrpprocesadores y microcontroladores mediante el empleo de nemónicos los cuales tiene un código de operación (OPCODE) que representa cada una de las operaciones a realizar.

Referencias

[1] Stephen Brown, Zvonko Vranesic. (2006). “Fundamentos de lógica digital con diseño VHDL”. Ed. Mc Graw Hill, México,

[2] Tocci Ronald J. (2003). “Sistemas Digitales: principios y aplicaciones”. Editorial Pearson Educación. 6ta edición.

[3] Mano Morris. (2003) “Diseño Digital”. Ed. Prentice Hall. 3ra edición.

[4] Lattice Semiconductor Corporation, “GAL 22V10D”. December 2006.

ISSN-2410-3977

[5] Angulo Amusátegui J. M. (2009). "Microcontroladores PIC Diseño practico y aplicaciones". Ed. Mc Graw Hill. Primer parte.

[6] Histand & Alciatore, (1999) Introduction to Mechatronics and Measurement Systems. McGraw Hill.

[7] Barrett, M. "Managing the Invisible Assets" Engineering & Technology, Vol. 3, No. 12, pp. 50-52, Oct 2008.

[8] Domingo, J.; Gámiz, J.; Grau, A. and Martínez, H. Introducción a los Autómatas Programables, 1st published, VOC, 2003, pp. 124, 135.

ISSN-2410-3977