**UNIVERSIDAD NACIONAL DE INGENIERÍA**

**FACULTAD DE ELECTROTECNIA Y COMPUTACIÓN**



Trabajo monográfico para la optar al

Título en Ingeniería en Computación

**Herramienta de generación de código para definición y gestión de lenguaje ensamblador como recurso didáctico para la asignatura de Máquinas Computadoras II de la Universidad Nacional de Ingeniería (UNI)**

**AUTOR:**

**Br. Harvin Manuel Toledo Polanco**

**Br. Jorge Manuel Potosme Álvarez**

**TUTOR:**

**Ing. José Díaz Chow**

**Managua, Nicaragua**

**2016**

Contenido

**No se encontraron entradas de tabla de contenido.**

## Introducción al lenguaje ensamblador

### Lenguaje de Bajo Nivel

Los lenguajes de bajo nivel, también llamados lenguajes de ensamblador, permiten al programador escribir los programas usando palabras nemotécnicas que ejercen control completo sobre el hardware, razón por la cual son muy dependientes del hardware donde se trabaje, no es lo mismo el lenguaje ensamblador para procesadores Intel que el lenguaje ensamblador para procesadores PowerPC(IBM) o procesadores ARM etc,

El uso de la palabra bajo en su denominación no implica que el lenguaje sea menos potente que un lenguaje de alto nivel, si no que se refiere a la reducida abstracción entre el lenguaje y el hardware. Por ejemplo, se utiliza este tipo de lenguajes para programar tareas críticas de los sistemas operativos, de aplicaciones en tiempo real o controladores de dispositivos.

### Ventajas y desventajas de los lenguajes ensambladores

Los lenguajes ensambladores tienen algunas ventajas sobre los lenguajes de alto nivel, por ejemplo:

Velocidad: debido a que el lenguaje ensamblador trabaja directamente con el hardware permite más rápido acceso a los recursos de la computadora por esta razón la velocidad, no obstante esto también puede depender mucho de la calidad del código fuente.

Flexibilidad: el lenguaje ensamblador es flexible ya que puede utilizar el hardware de en su totalidad, sin restricciones de ningún tipo, también puede trabajar independiente del sistema operativo por ejemplo un programa en ensamblador puede correr tanto en sistemas Unix como en sistemas Windows,

Desventajas

Portabilidad mínima: los programas creados en ensamblador al depender tanto de la arquitectura del procesador, no pueden ejecutarse en computadoras que posean una arquitectura diferente de donde fueron creados.

Mantenimiento de código es muy complicado: modificar, agregar o eliminar alguna instrucción y/o función resulta mucho más complicado debido a que el lenguaje ensamblador no permite mucha abstracción por esta razón el código fuente es más grande.

## El procesador

## Unidades Funcionales Básicas

### Unidad Aritmética Lógica (ALU)

La ALU es la encargada de realizar operaciones aritméticas sobre los datos de la memoria tales como suma, resta, multiplicación y división, también se encarga de realizar las operaciones lógicas, tales como AND,OR, NOT etc.

Unidad Aritmética: todas las operaciones pueden realizarse mediante tres sistema:

* Sistema serie: se procesa bit a bit en forma secuencial
* Sistema paralelo: se procesan simultáneamente todos los bits.
* Sistema paralelo-serie: se procesa por grupos. Los grupos se procesan en serie y los bits que componen los grupos en paralelo.

Unidad Lógica: establece comparaciones para facilitar la toma de decisiones. Estas comparaciones son mucho más sencillas que las operaciones aritméticas, puesto que no necesitan considerar el resultado de la operación realizada con los bits anteriores, pudiéndose efectuar en paralelo, El resultado de las operaciones puede ser vedadero o falso, que en representación binaria es 1 o 0.

### Unidad de Control (CU)

La unidad de control se encarga de buscar las instrucciones en la memoria principal, interpretarlas y ejecutarlas, empleando para ello la unidad empleando para ello la unidad de procesos,

Coordina todos los componentes del computador de modo que los eventos tomen lugar en la secuencia apropiada en el momento correcto. Además de realizar esta labor de sincronización, la unidad de control, es decir entiende las instrucciones del programa que obtiene de la memoria y dirige la acción para realizarlas.

La secuencia lógica que la unidad de control debe realizar para ejecutar una instrucción es la siguiente:

1. Localizar y extraer de la memoria principal la instrucción correspondiente.
2. Transferir la instrucción de la memoria a la Unidad de Control.
3. Determinar qué tipo de operación se debe ejecutar.
4. Ejecutar la instrucción, enviando las señales de control u órdenes a los elementos pertinentes.
5. Supervisar la operación anterior para determinar si ha finalizado correctamente.
6. Localizar la siguiente instrucción a ejecutar.

Elementos de la Unidad de Control

El reloj: consiste en un circuito eléctrico capaz de generar una sucesión de pulsos a intervalos de tiempo constantes. El intervalo entre dos puntos de reloj se denomina *ciclo*.

Contador de programa (PC): también denominado registro contador de instrucción, (RCI). Su misión es controlar el orden de ejecución de las instrucciones del programa, de acuerdo con su contenido. Un programa no siempre ejecuta las instrucciones secuencialmente. Puede haber instrucciones de salto o bifurcación.

Registro de Instrucción: es una unidad de almacenamiento temporal, este registro guarda la instrucción cuando se extrae de la memoria principal y se mantiene mientras se realiza la decodificación o interpretación.

Decodificador: habitualmente, toda instrucción contiene un campo conocido como código de operación, que indica el tipo de operación que hay que realizar; el decodificador es el elemento encargado de realizar el análisis del código de operación.

Secuenciador: Es un generador de ordenes simples, denominadas microordenes que sincronizadas con el reloj y distribuidas a los elementos necesarios permiten la ejecución de la instrucción.

1. Arquitectura de registros dos direcciones

Esta arquitectura se caracteriza por solo definir 2 operandos, uno corresponde a fuente y destino a la vez por tanto se da una lectura destructiva.

1. Código de instrucciones
   1. Instrucciones de transferencia de datos

MOV

Copia el operando 2 al operando 1

Operandos posibles:

Registro, Memoria

Memoria, Registro

Registro, Registro

Memoria, Inmediato

PUSH

Almacena un valor de 16 bits en la pila

Operandos posibles

Registro

-sreg

Memoria

Inmediato

POP

Obtiene un valor de 16 bits de la pila

Operandos posibles:

Registro

Memoria

-sreg

* 1. Instrucciones Lógicas

AND

Instrucción lógica AND entre todos los bits de 2 operandos, el resultado se almacena en el operando 1.

Operandos posibles

Registro, Memoria,

Memoria, Registro

Registro, Registro

Memoria, Inmediato

Registro, Inmediato

OR

Operación lógica OR entre todos los bits de 2 operandos, el resultado es almacenado en el primer operando.

Operandos posibles:

Registro, Memoria

Memoria, Registro

Registro, Registro

Memoria, Inmediato

Registro, Inmediato

XOR

Operación lógica XOR (OR exclusivo) entre todos los bits de un operando, el resultado es almacenado en el primer operando.

Operandos posibles:

Registro, Memoria

Memoria, Registro

Registro, Registro

Memoria, Inmediato

Registro, Inmediato

NOT

Invierte cada bit del operando.

Operandos posibles:

Registro

Memoria

* 1. Instrucciones Aritméticas

ADD

Suma los operandos

Operandos posibles:

Registro, Memoria,

Memoria, Registro

Registro, Registro

Memoria, Inmediato

Registro, Inmediato