***Arquitectura de Computadoras***

******

**TEMA**

Trabajo Práctico N°3 BIP (Basic Instruction-set Processor)

**Profesor:**

Ing. Rodriguez Santiago

<[santiagoarodriguez@gmail.com](mailto:santiagoarodriguez@gmail.com)>

**INTEGRANTES:**

Tissot, Esteban

<[egtissot@gmail.com](mailto:egtissot@gmail.com)>

Manero, Matias

<[matiasmanero@gmail.com](mailto:matiasmanero@gmail.com)>

**CARRERA:**

Ingeniería en Computación

**AÑO:**

2018

**INDICE**

[**Introducción**](#_din1d9z8cpxm) **4**

[**Diseño**](#_i0xbm99ddqw8) **5**

[Reglas](#_hy8gbzh1xxbg) 5

[Arquitectura](#_tvthl22nppkg) 6

[Atributos](#_3diizs4g21lb) 6

[Instrucción](#_rx88p3hdri3q) 6

[Registros](#_80lf66w3rgws) 6

[Direccionamiento](#_kq2pxvgoh8ux) 6

[Set de Instrucciones](#_xvdsnnmywez9) 7

[Programación](#_nyzzq6d67eme) 7

[Organización](#_vkvw40yjx3ya) 8

[**Implementación**](#_a1hscosgcrx7) **10**

[**Conclusión**](#_e68wa9y5ninf) **11**

[**Referencias**](#_ugigsb3k844f) **12**

[**Anexo**](#_qtxigmlagb8) **13**

# 

# 

# Introducción

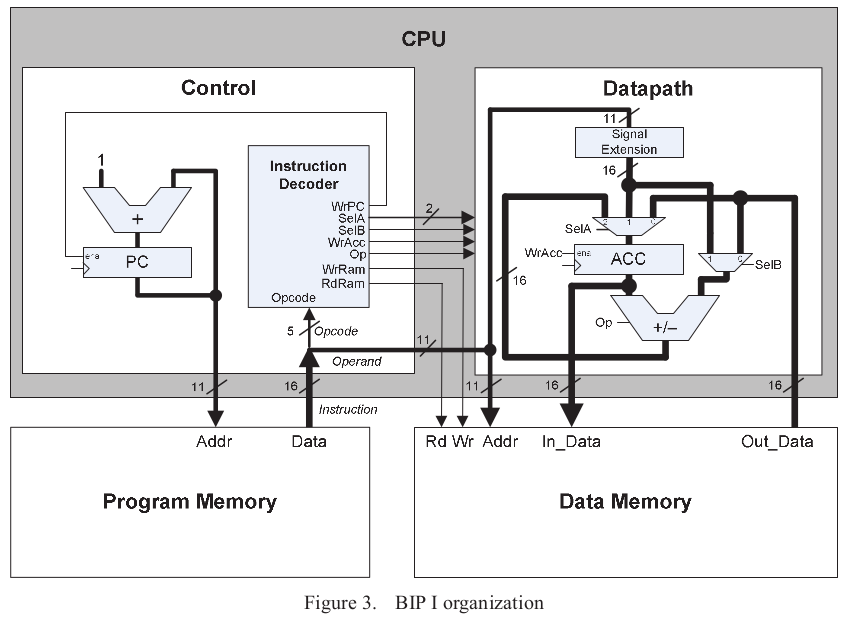
El **microprocesador** (o simplemente **procesador**) es el circuito integrado central más complejo de un sistema informático. Es el encargado de ejecutar los programas, desde el sistema operativo hasta las aplicaciones de usuario; sólo ejecuta instrucciones programadas en lenguaje de bajo nivel, realizando operaciones aritméticas y lógicas simples, tales como sumar, restar, multiplicar, dividir, las lógicas binarias y accesos a memoria. Su arquitectura puede contener una o más unidades centrales de procesamiento (CPU) constituidas, esencialmente, por registros, una unidad de control, una unidad aritmético lógica (*ALU*) y una unidad de cálculo en coma flotante (conocida antiguamente como «coprocesador matemático»).

Los componentes típicos de una CPU son la unidad aritmético lógica (ALU), que realiza operaciones aritméticas y lógicas, y la unidad de control (CU), que extrae instrucciones de la memoria, las decodifica y las ejecuta, llamando a la ALU cuando sea necesario.

Un **conjunto de instrucciones**, **repertorio de instrucciones**, **juego de instrucciones** o **ISA** (del inglés *instruction set architecture*, «arquitectura del conjunto de instrucciones») es una especificación que detalla las instrucciones que una unidad central de procesamiento puede entender y ejecutar, o el conjunto de todos los comandos implementados por un diseño particular de una CPU. El término describe los aspectos del procesador generalmente visibles para un programador, incluyendo los tipos de datos nativos, las instrucciones, los registros, la arquitectura de memoria y las interrupciones, entre otros aspectos.

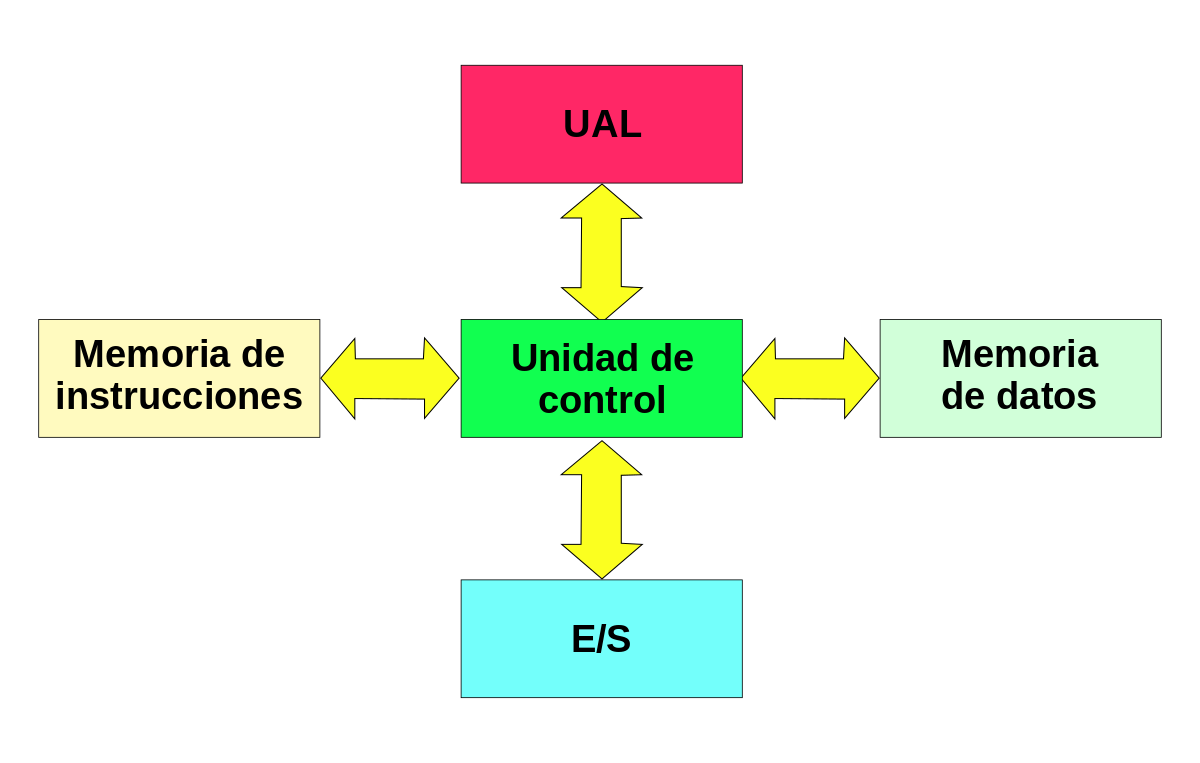
# Diseño

Para el presente trabajo desarrollaremos un procesador simple denominado BIP (Basic Instruction-set Processor) que es conocido por ser un procesador de ejemplo didáctico siendo muy sencillo aprender su arquitectura debido que el set de instrucciones que maneja es reducido. Luego de entender su funcionamiento procedemos a implementarlo en un lenguaje de descripción de hardware (HDL) como lo es verilog, para luego sintetizar en una FPGA.



## Requerimientos

## Un cpu es una unidad compleja que lleva mucho tiempo de desarrollo, para fines prácticos de este trabajo presentamos los requerimientos que iremos a seguir para reducir la complejidad del mismo.

* *Requerimientos de Desarrollo*
  + La arquitectura del procesador y el programa en assembler debe ser lo más simple posible.
  + El procesador debe ilustrar los conceptos básicos.
  + La implementación debe ser lo más intuitiva y simple posible.
* *Requerimientos de Arquitectura*
  + Todas las instrucciones del ISA (Instruction Set Architecture) están basadas en un solo formato.
  + Las instrucciones lógicas y aritméticas pueden acceder a la memoria de datos (no es una arquitectura Load/Store).
  + El set de instrucciones es escalable y compatible con las versiones anteriores.. Cada generación añade funcionalidades, y complejidad, al set.
* *Requerimientos de Organización*
  + El procesador debe implementar la organización monociclo/Harvard para simplificar la búsqueda de instrucciones y ejecución, y limitar la unidad de control a un circuito combinacional. 
  + Los registros deben estar limitados a la cantidad requerida por la arquitectura del procesador, es decir que no debe haber registros adicionales, como en organizaciones multiciclos o pipelines.
  + El procesador debe incluir una unidad de aritmética básica (no una ALU) para soportar operaciones de suma y resta.
  + El procesador no debe tener un archivo de registros.

## Arquitectura

Antes de empezar la implantación en Verilog necesitamos definir ciertos aspectos básicos que forman parte de un procesador. Entre ellos está la arquitectura que vamos a utilizar, el formato y tamaño de las instrucciones que acepta el mismo, el set de instrucciones capaz de procesar, etc.

### Atributos

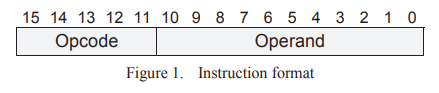
* Accumulator-oriented architecture.
* Bus de datos de 16-bit.
* Bus de Instrucciones de 16-bit.
* Un tipo de datos (enteros).
* Un solo formato de instrucciones.
* Dos tipos de direccionamiento (Directo e Inmediato).
* Set de Instrucciones reducido.
* Memory-mapped I/O.

Nota: No se debe considerar un procesador RISC (Reduced Instruction Set Computer) porque no implementa la arquitectura Load/Store y no contiene registros de propósito general.

### Instrucción

El formato de la instrucción tiene dos campos:

* *Opcode*: son los 5 bits más significativos que identifican el código de la operación.
* *Operand*: corresponde a los 11 bits restantes que identifican la operación de la instrucción. Puede representar un dato inmediato (constante), una dirección en la memoria de datos (una variable), o una dirección en la memoria del programa (un salto).

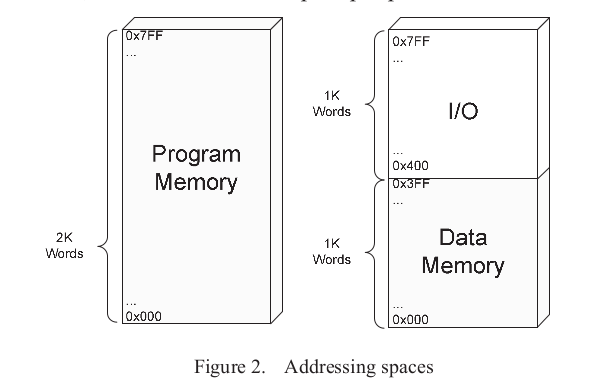


### Registros

* *PC* (Program Counter): almacena la dirección de la instrucción actual.
* *ACC* (Acumulator): funciona como un operando implícito en muchas instrucciones.

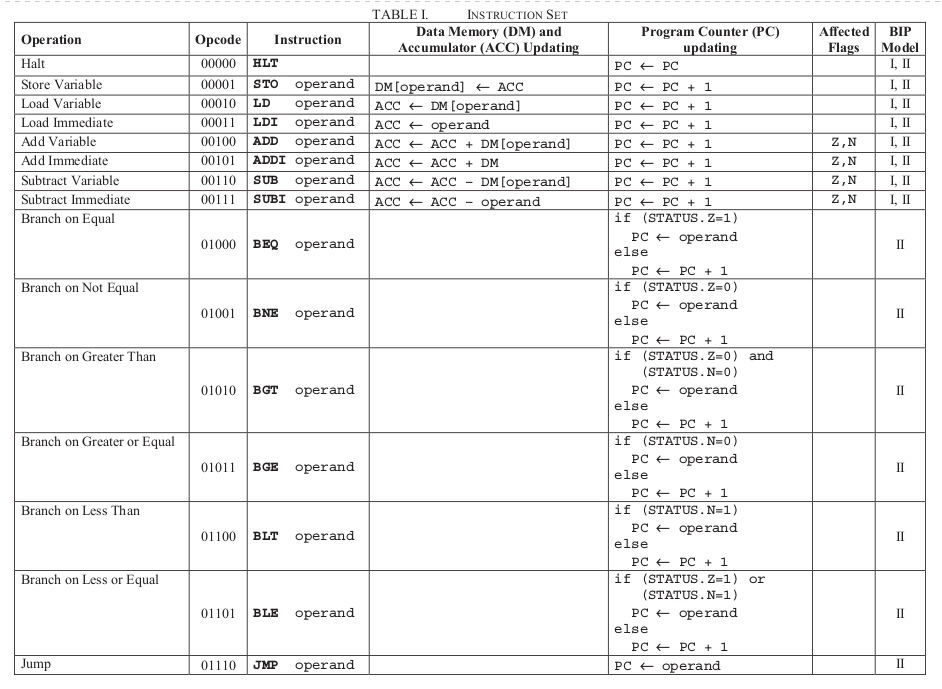
### Direccionamiento

El espacio de direccionamiento está organizado en 2-K palabras para el programa y 2-K palabras para datos. A su vez, el espacio para datos está subdividido en dos regiones iguales, una para “data memory” y la otra para “memory-mapped I/O”. Esta última puede utilizarse para representar el acceso a los dispositivos básicos como leds, switches y otros periféricos.



### Set de Instrucciones

En la siguiente imagen solo se muestran las instrucciones que soporta el BIP I, la tabla completa con las instrucciones del BIP I y II se encuentra en la sección ANEXO.



Las últimas cuatro instrucciones afectan las banderas Z y N

## 

## 

## 

## 

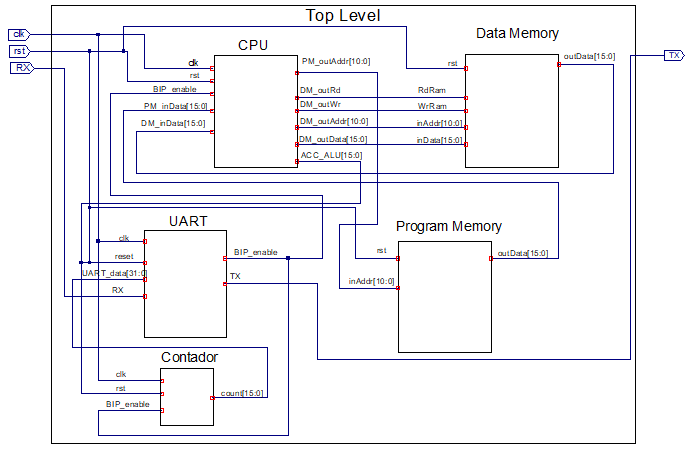
## 

# 

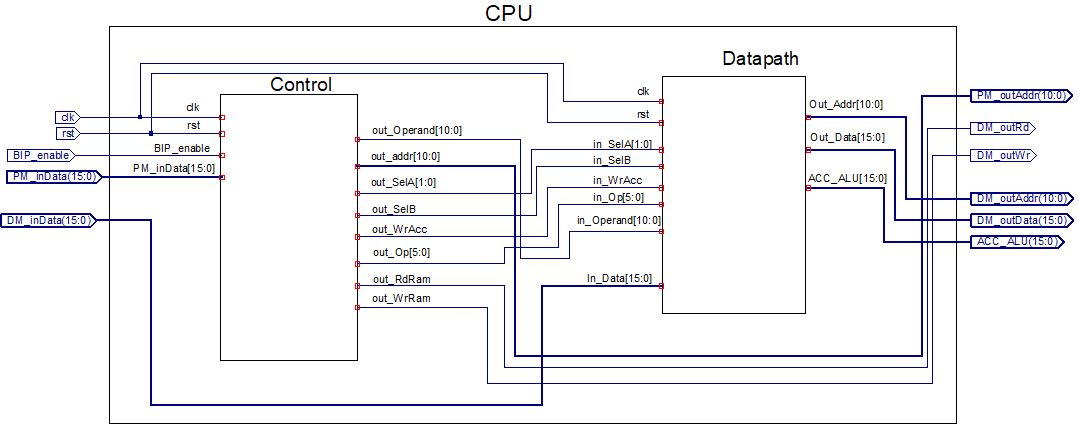
# Implementación

Se implementaron cinco módulos principales:

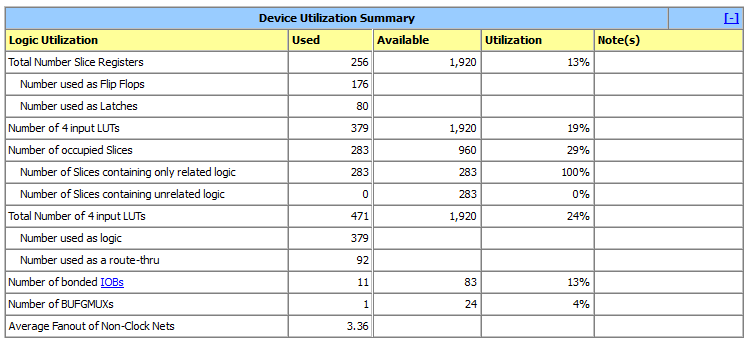
* Data Memory: Es una memoria que contiene registros con determinados valores seteados por el usuario.
* Program Memory: Es una memoria donde se almacenan las 9 instrucciones del programa a ejecutar.
* UART: Es un módulo encargado de recibir la señal enviada desde la PC para habilitar el BIP y, al finalizar el programa, envía a la PC el valor del registro ACC junto con la cantidad de instrucciones ejecutadas.
* Contador: Es un módulo encargado de contar los ciclos de clock.
* CPU: es la unidad encargada de la decodificación y ejecución de la instrucción, en este módulo se incrementa el PC en cada ciclo de clock.



A su vez el módulo de CPU contiene dos sub-modulos, uno encargado de manipular las instrucciones(Control) y el otro de realizar las operaciones correspondientes(Datapath).

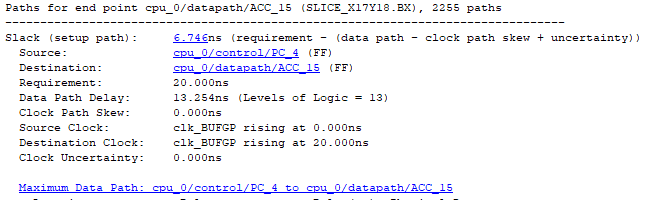


## Componentes físicos utilizados (**Basys II)**

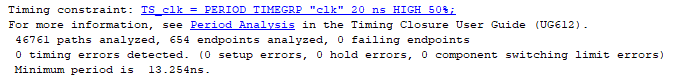


## Reporte de Timing

En la sección de reporte de timing del software se detalla el camino crítico, que es el camino donde la señal tiene mayor delay, en este caso desde el pin 4 del módulo PC hasta el pin 15 del módulo ACC. Este tiempo es de 13.254ns y determina la máxima frecuencia a la que puede trabajar (74MHz). Se debe tener en cuenta que este es un cálculo aproximado y que si es necesario trabajar a frecuencias cercanas a la máxima de deben hacer análisis más detallados y precisos.



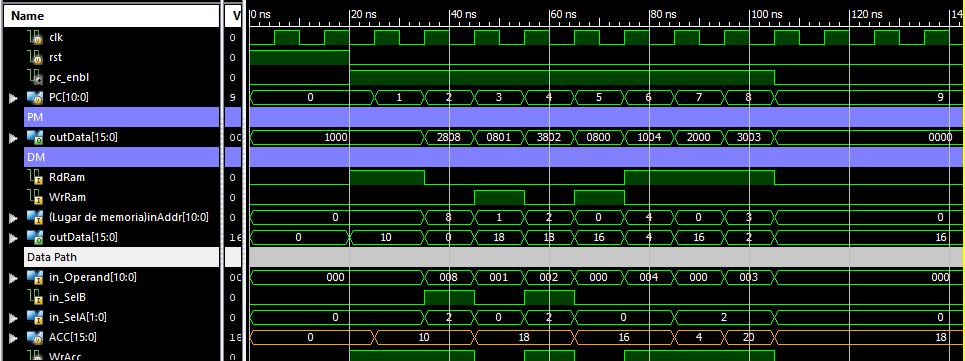
De acuerdo al reporte de timing generado por el software ISE obtenemos el resultado mostrado a continuacion



De la imagen de arriba podemos decir que el periodo mínimo al que debe trabajar el cpu desarrollado es de 13,252ns que convertido a frecuencia es igual 74MHz.

# Simulación

|  |  |  |  |
| --- | --- | --- | --- |
| Program Memory | | | ACC |
| LoadV  AddI  Store  SubI  Store  LoadV  AddV  SubV  HALT | 11'h001:Data <= 16'b00010\_00000000000; 11'h002:Data <= 16'b00101\_00000001000;  11'h003:Data <= 16'b00001\_00000000001;  11'h004:Data <= 16'b00111\_00000000010;  11'h005:Data <= 16'b00001\_00000000000;  11'h006:Data <= 16'b00010\_00000000100;  11'h007:Data <= 16'b00100\_00000000000;  11'h008:Data <= 16'b00110\_00000000011;  11'h009:Data <= 16'b00000\_00000000000; | Se carga en ACC el valor de la posición 0 de mem.  Se suma en valor de ACC y 8.  Se guarda ACC en la posición 1 de mem.  Se resta ACC y 2.  Se guarda ACC en la posición 0 de mem.  Se carga en ACC el valor de la posición 4 de mem.  Se suma ACC y el valor de la posición 0 de mem.  Se resta ACC y el valor de la posición 3 de mem.  Finaliza el programa. | 10  18  18  16  16  12  28  26  26 |



# Conclusión

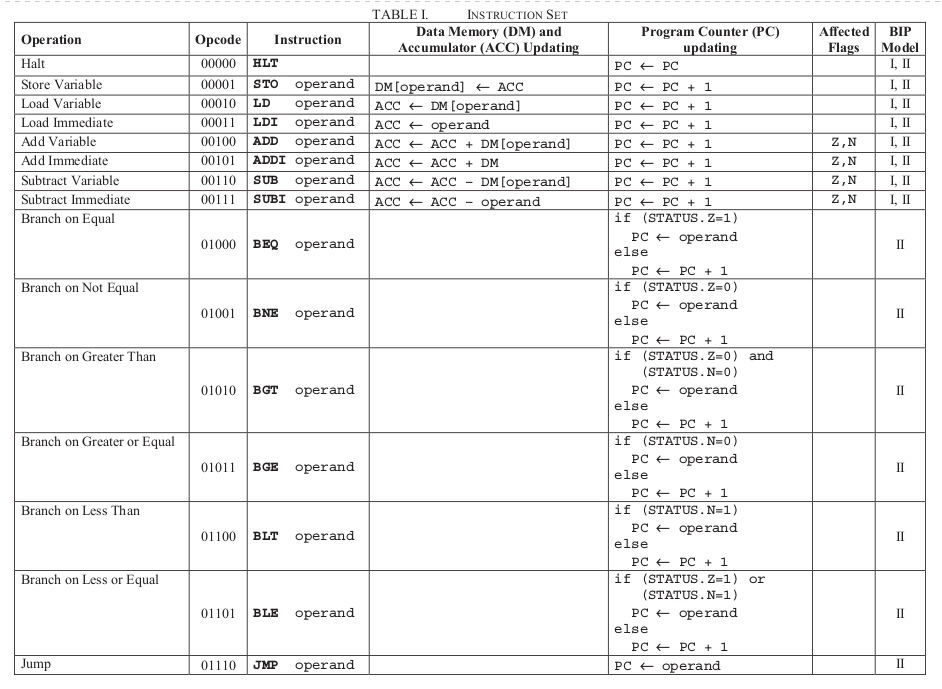
En el presente informe se logró implementar un procesador básico, el cual toma una instrucción de una memoria, la decodifica y setea las señales correspondientes para obtener un resultado esperado. En cada iteración se incrementa en uno un registro denominado PC que se utiliza para buscar la siguiente instrucción, de esta forma el sistema se vuelve autónomo y solo le enviamos una señal mediante la comunicación UART para que comience la ejecución.

Con este trabajo práctico se entendió la importancia de las señales de control y del sincronismo entre ellas, lo que nos brinda un mejor entendimiento de la arquitectura de un procesador complejo y la posibilidad de implementar un pipeline en el siguiente trabajo.

# 

# Anexo

Set de Instrucciones del BIP I y II



Instrucciones en assembler

