**Índice**

[Introducción 2](#_TOC_250012)

[Camino de Datos 3](#_TOC_250011)

[Registros 4](#_TOC_250010)

[UAL (Unidad Aritmética-Lógica) 5](#_TOC_250009)

[Desplazador 9](#_TOC_250008)

[Bus de Datos 10](#_TOC_250007)

[Unidad de Control 11](#_TOC_250006)

[Almacén de Control 12](#_TOC_250005)

[MPC 13](#_TOC_250004)

[MIR 14](#_TOC_250003)

[Microprograma 16](#_TOC_250002)

[Decodificador 16](#_TOC_250001)

[Conclusión 17](#_TOC_250000)

**La Microarquitectura MIC-1**

# Introducción

Este proyecto final para la materia Arquitectura y Organización de la Computadora 1 se

centra en la implementación y simulación de la MIC-1, una microarquitectura diseñada por

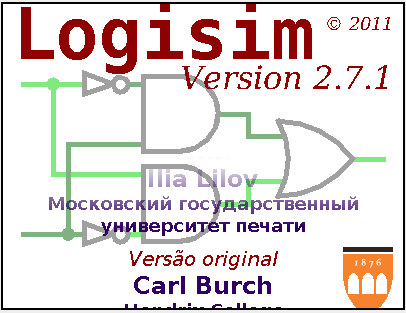
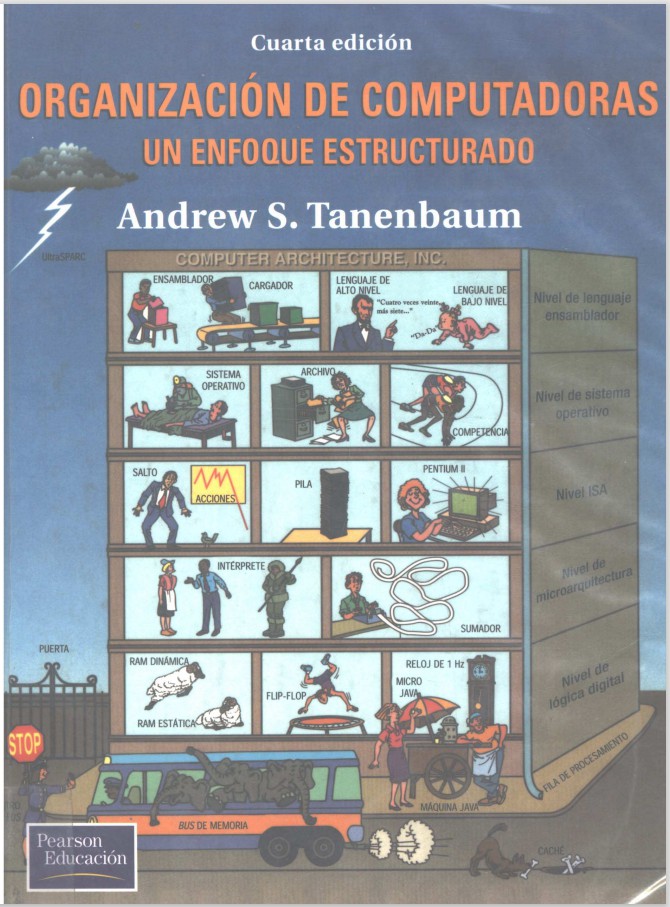
Andrew S. Tanenbaum con propósitos educativos, enseñando los conceptos fundamentales de arquitectura de computadoras y diseño de procesadores, presentada en su libro “Organización de Computadoras: Un Enfoque Estructurado”.

La MIC-1, posee una ISA que consiste en una versión simplificada de la *Java Virtual*

*Machine* (JVM), enfocada específicamente en números enteros, conocida como *Integer Java Virtual Machine* (IJVM).

Esta microarquitectura, como se detalla más adelante en este documento, puede separarse en dos grandes partes conceptuales: el Camino de Datos, donde profundizaremos en sus 4 componentes, sus funcionamientos y cómo están conectados; y la Unidad de Control, donde veremos qué rol cumple dentro de la MIC-1.

En este caso para realizar los circuitos de la MIC-1 se utilizó Logisim, un programa desarrollado por Carl Burch, el cual nos permite diseñar y simular circuitos lógicos digitales con una interfaz gráfica sencilla, pudiendo construir y probar el modelo de microarquitectura de una manera interactiva.

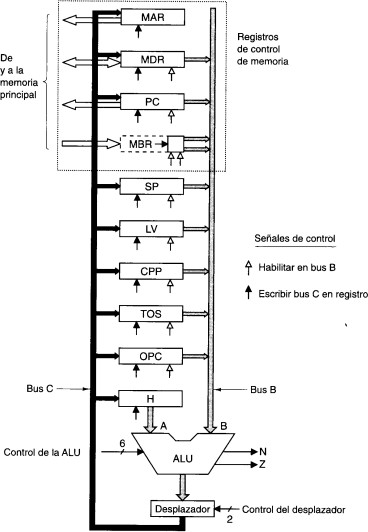


# Camino de Datos

Podemos decir que la Trayectoria o Camino de Datos, es el núcleo de la MIC-1, está diseñado para permitir la manipulación eficiente de datos y la ejecución de operaciones aritméticas y

lógicas necesarias para la ejecución de programas, y como se puede observar en la Fig. 1 se compone de:

* Varios registros, donde algunos de ellos tienen un uso general y otros más específicos.
* Una Unidad Aritmética Lógica.
* Un Desplazador.
* Buses de datos.



**Figura 1.** Camino de datos de la MIC-1.

## Registros:

Son los encargados de almacenar datos y operandos para ser utilizados por las instrucciones. Cada registro de 32 *bits* es seleccionado mediante dos líneas de control (una que habilita el bus B y otra el C) e implementan la utilización de *D flip-flops* (*Delay*) para almacenar datos binarios y direcciones de memoria. Los mismos se encuentran conectados a través de buses de datos para permitir transferencias entre registros y la UAL.

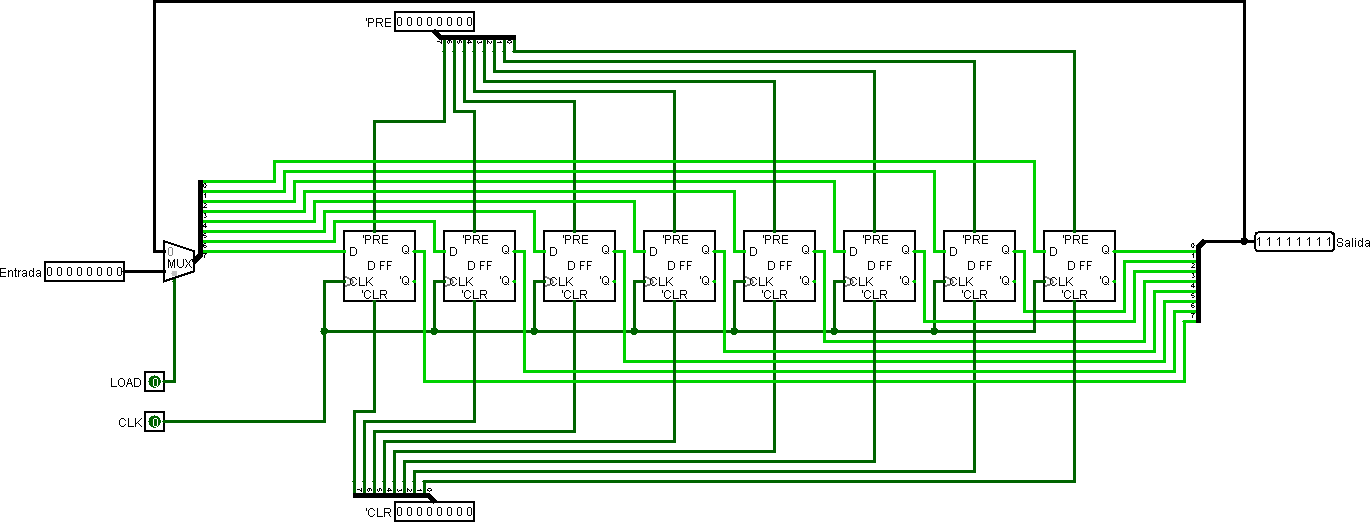
Dentro de los tantos registros, podemos destacar varios que son de uso específico: el *PC*, el cual contiene la dirección de la próxima instrucción a ser ejecutada; el *IR* o *MBR*, que se encarga de retener la instrucción que está siendo ejecutada; y unos registros de entrada/salida, el *MAR* y el *MDR*, los cuales generalmente se usan para guardar temporalmente datos que fueron leídos de memoria o que tienen que ser escritos en memoria junto a sus direcciones.

Casi todos los registros pueden colocar su contenido en el bus B.

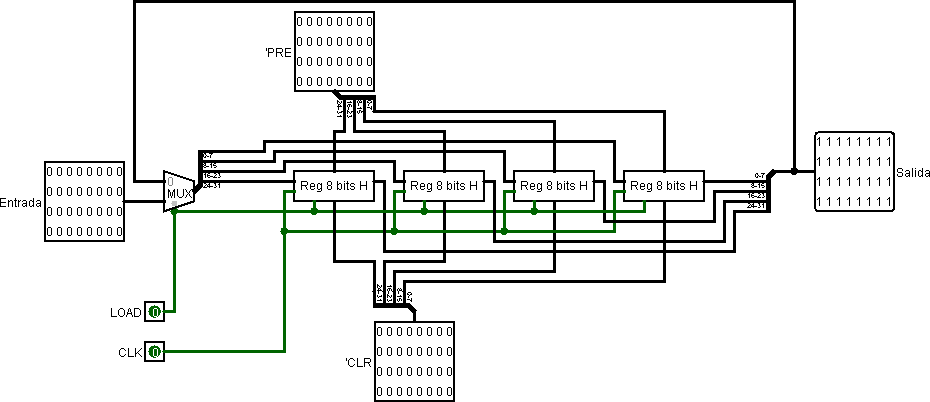
Además los registros implementados en Logisim, tienen la particularidad de contar con dos funciones las cuales se acceden a través de *pin* como se observa en la Fig. 2-a y 2-b. Estas

son capaces de mantener la información usando su método *Hold* o de cargar la misma usando

*Load*.



**(a)**



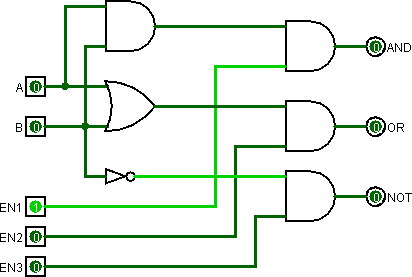
**(b)**

**Figura 2.** (a) Registro de 8 *bits* con funciones *Hold* y *Parallel Load.* (b) Registro de 32 *bits* con funciones *Hold* y *Parallel Load.*

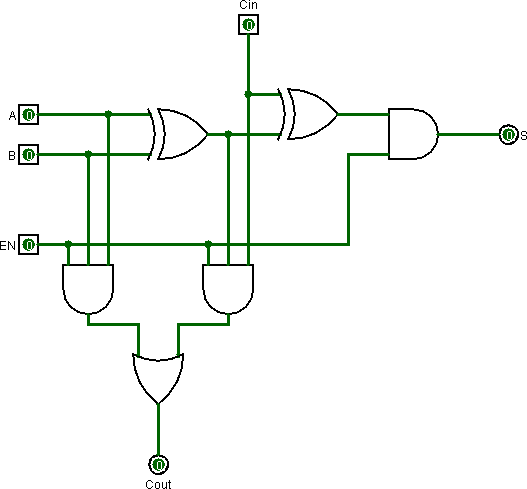
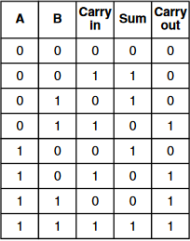
## UAL (Unidad Aritmética-Lógica):

La UAL (o ALU por su traducción al inglés - *Arithmetic-Logic Unit*) es un componente fundamental en las CPU, y como su nombre lo indica, su principal función dentro del Camino de Datos es la de realizar operaciones aritméticas y lógicas sobre los datos que le van llegando.

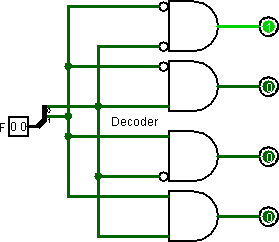
La UAL de 1 *bit* se encuentra implementada con la Unidad Lógica (Fig. 3), conformada por circuitos combinacionales que realizan operaciones como: Suma, Resta y *NOT*; un Sumador Completo (Fig. 4), puesto para resolver los problemas de límites que se daban con el medio sumador; y un Decodificador (Fig. 6), el cual funciona como un selector de función 2 a 4.



**Figura 3.** Unidad Lógica en Logisim.

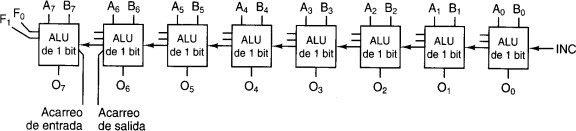
**Figura 4.** Sumador Completo en Logisim. **Figura 5.** Tabla de verdad sumador completo.



**Figura 6.** Decodificador en Logisim.

Para formar UALs más grandes, en nuestro caso hasta una de 32 *bits*, es tan simple como ir conectando las 1 *bit*, el acarreo de salida de una al acarreo de entrada de la otra e ir

anidándolas hasta llegar al números de *bits* deseado, tal y cómo se muestra en la Fig 7.



**Figura 7.** UALs de 1 *bit* conectadas para formar una de ocho *bits*. No se muestran las señales de habilitación ni de inversión.

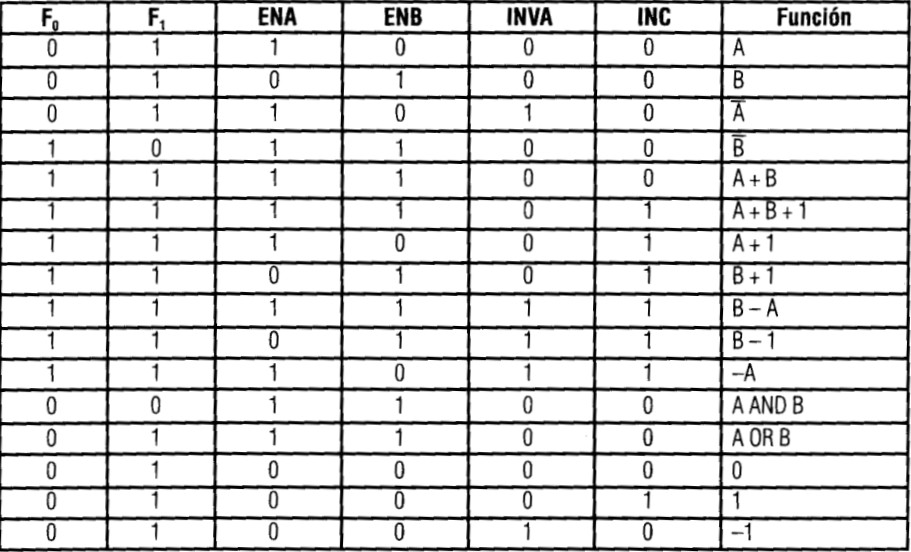
La UAL es controlada por la Unidad de Control, la cual genera señales de operación y selección de operandos, indicando así qué operaciones realizar y cuándo hacerlo. Más adelante se explica cómo funciona dicha Unidad. Si detallamos su estructura y

funcionamiento, podemos decir que cuenta con dos entradas A y B, la primera conectada al registro de retención H, y la segunda directamente al bus B. Y la salida de la UAL se encarga de alimentar la entrada del Desplazador.

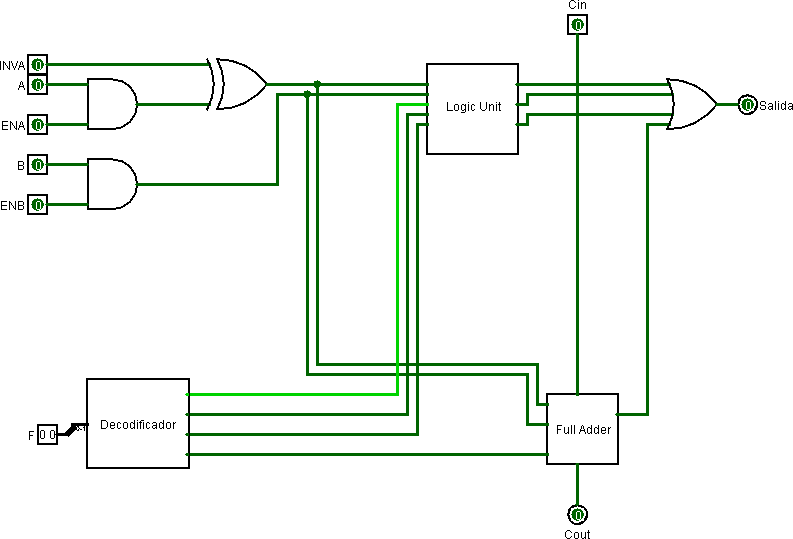
Además, cuenta con:

* Dos salidas para las banderas:
  + N (negativa)
  + Z (cero),
* Y seis líneas de control:
  + F0 y F1: señales de control que se encuentran conectadas al decodificador y de cuyos valores depende la selección de la operación a realizar.
  + ENA y ENB: las cuales sirven para habilitar individualmente las salidas.
  + INVA: para invertir la entrada A.
  + INC: para forzar un acarreo de entrada en el bit menos significativo.

Entonces, una vez que tenemos el registro de retención H como entrada A, a continuación se selecciona un registro como entrada B, su valor se coloca en el Bus B al principio del ciclo y se mantiene ahí durante todo el ciclo. Luego, la UAL efectúa su trabajo y produce un resultado que se pasa al Bus C luego de pasar a través del desplazador. Cerca del final del ciclo, cuando las salidas de la UAL y el desplazador se encuentran estables, una señal de reloj activa el almacenamiento del Bus C en uno más de los registros.



**Figura 8.** Tabla de verdad sobre las Funciones de la UAL.



**Figura 9.** UAL de 1 *bit* en Logisim.

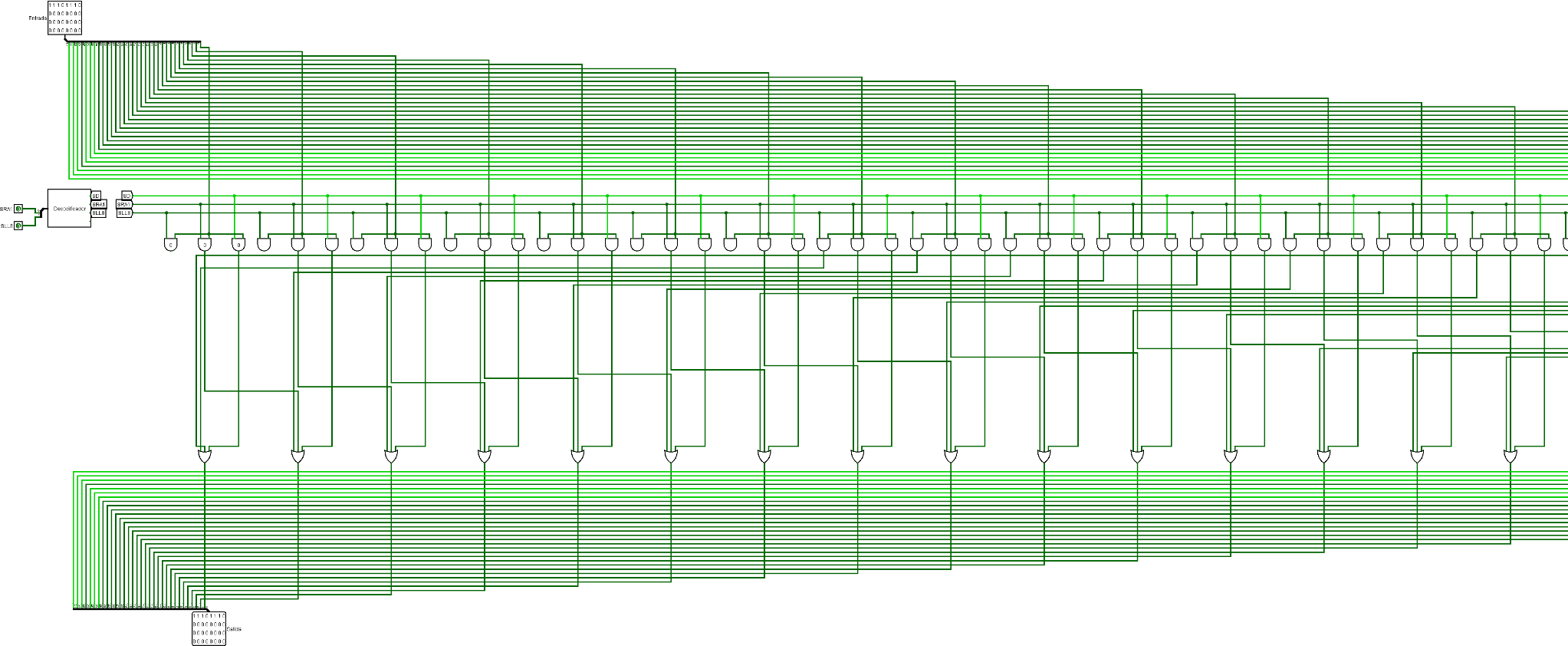
## Desplazador:

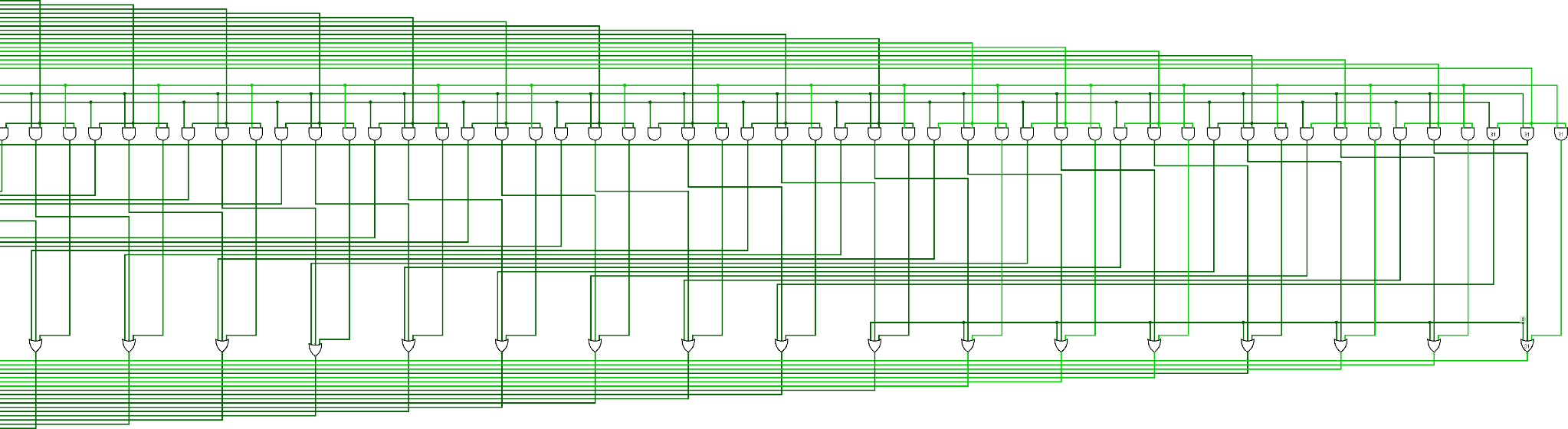
El desplazador (o *Shifter* por su traducción del inglés) es un componente especializado que contiene tanto una entrada como una salida de 32 *bits*. La entrada llega desde la UAL y la salida se conecta directamente al Bus C.

Su función es la de realizar operaciones de desplazamiento de bits en los datos almacenados en los registros. Dicho desplazamiento puede ser lógico o aritmético, para seleccionar cuál utilizar se necesita una de sus dos señales de control, llamadas SLL8 y SR1. La primera, (desplazamiento lógico a la izquierda) desplaza el contenido un *byte* a la izquierda y

convierte los ochos *bits* menos significativos en ceros. Mientras que la segunda, (desplazamiento aritmético a la derecha) desplaza el contenido un *bit* a la derecha sin modificar el *bit* más significativo.

Como se observa en la Fig. 10 las dos señales de control que corresponden a cada función se conectan a un decodificador con el observado en la Fig. 6, aunque en este caso solo se hace uso de tres salidas. La primera no produce ningún tipo de desplazamiento, por lo que la

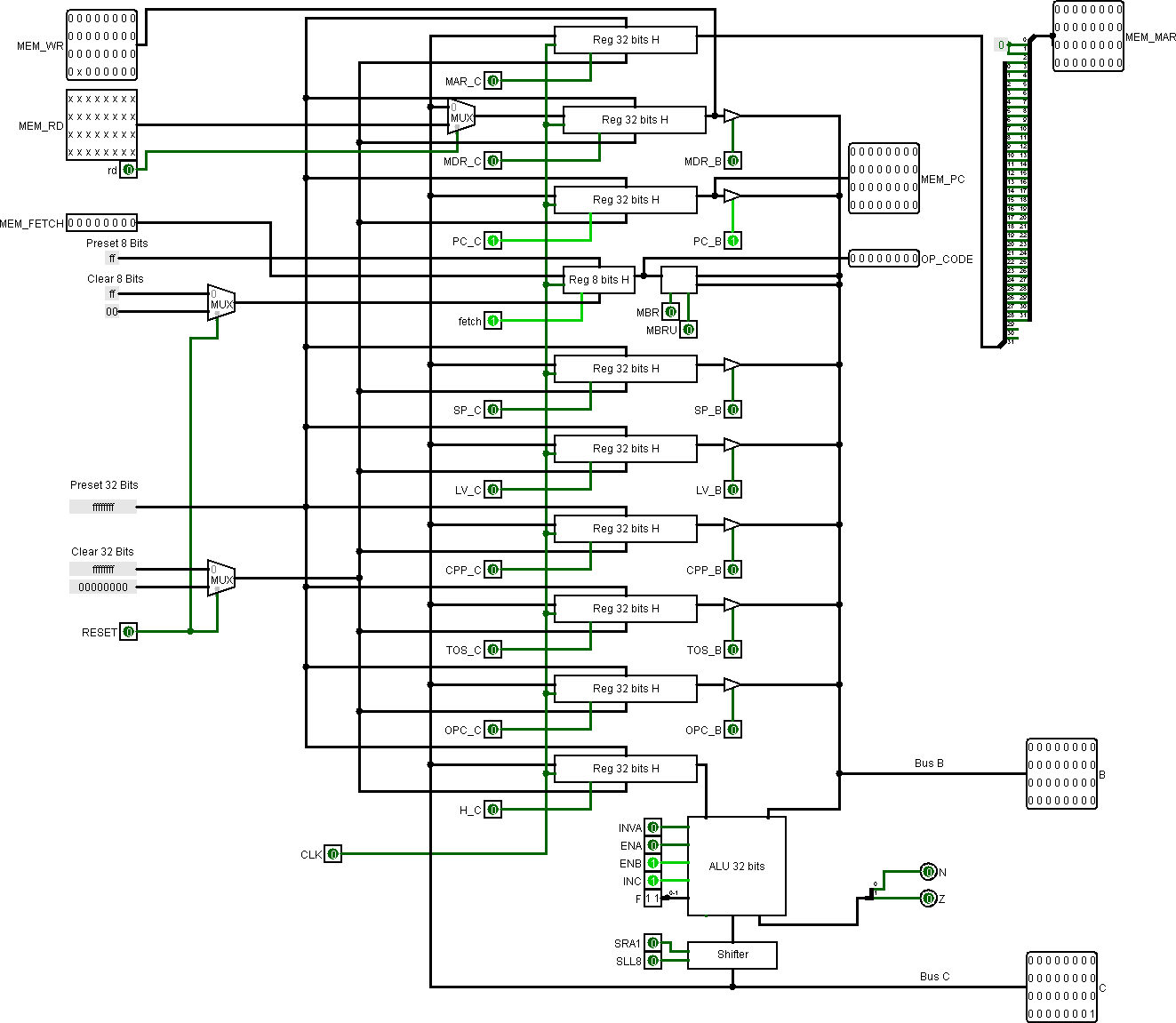
entrada sale del desplazador tal y cómo entró. La segunda salida fue asignada a la función SR1, y la tercera a SLL8.

**Figura 10.** Desplazador en Logisim.

## Bus de Datos:

Es un canal compartido que permite la transferencia de datos, durante la ejecución de instrucciones, entre todos los componentes del Camino de Datos.

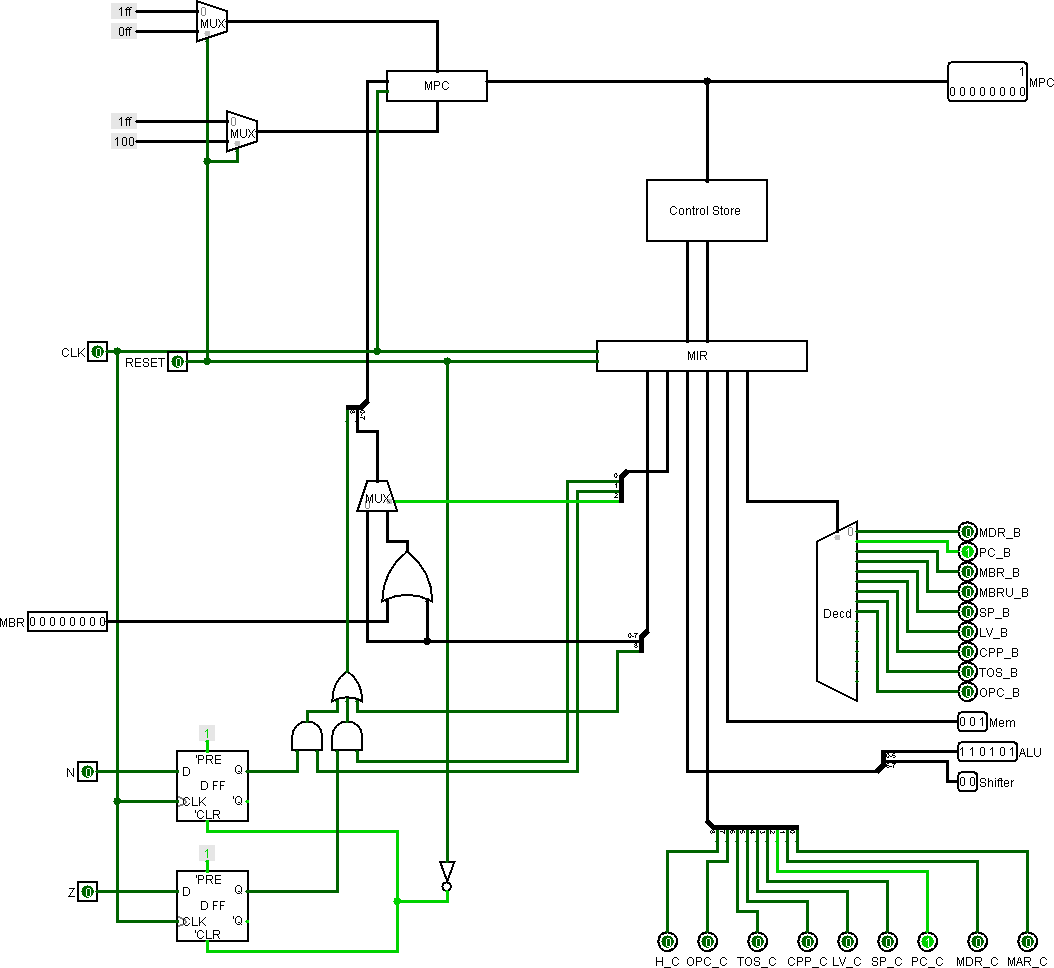
En la MIC-1, los buses multipuntos A, B y C, consisten en líneas de datos de 32 *bits* cada una, que conectan registros, la UAL y otros componentes. Se encuentran implementados como líneas físicas en el diseño del circuito y son controlados por señales de multiplexación, decodificadores y búferes controlados que seleccionan las fuentes y destinos de los datos según las instrucciones que llegan desde la Unidad de Control.



**Figura 11.** Camino de Datos en Logisim.

# Unidad de Control

La unidad de control es la responsable de gestionar y coordinar las operaciones que se van a realizar, la misma se encarga de generar las señales de control necesarias para ejecutar cada instrucción. Esto último lo hace decodificando instrucciones de la memoria de programa y emitiendo señales que controlan los diferentes componentes, tales como los registros, la UAL y la memoria.



**Figura 12.** Unidad de Control en Logisim.

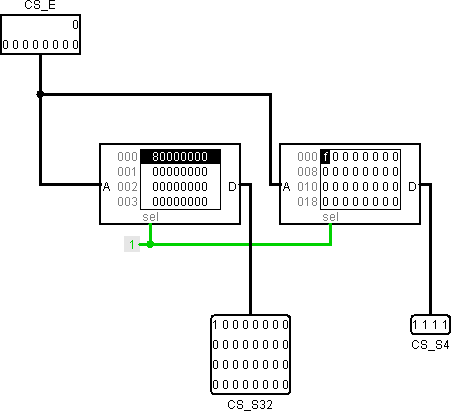
## Almacén de Control:

El almacén de control es el elemento más grande e importante de la porción de control de la máquina, se trata de una memoria que contiene microinstrucciones en lugar de instrucciones ISA. Este es muy diferente de la memoria principal, ya que las instrucciones de la memoria principal se ejecutan en orden según su dirección; mientras que las microinstrucciones especifican explícitamente su sucesora.

Puesto que el almacén de control es funcionalmente una memoria, necesita sus propios

registros de direcciones de memoria y de datos de memoria. Entonces, tenemos por un lado el

registro de dirección de memoria MPC (Contador de MicroPrograma), y por otro el registro de datos MIR (Registro de MicroInstrucción).



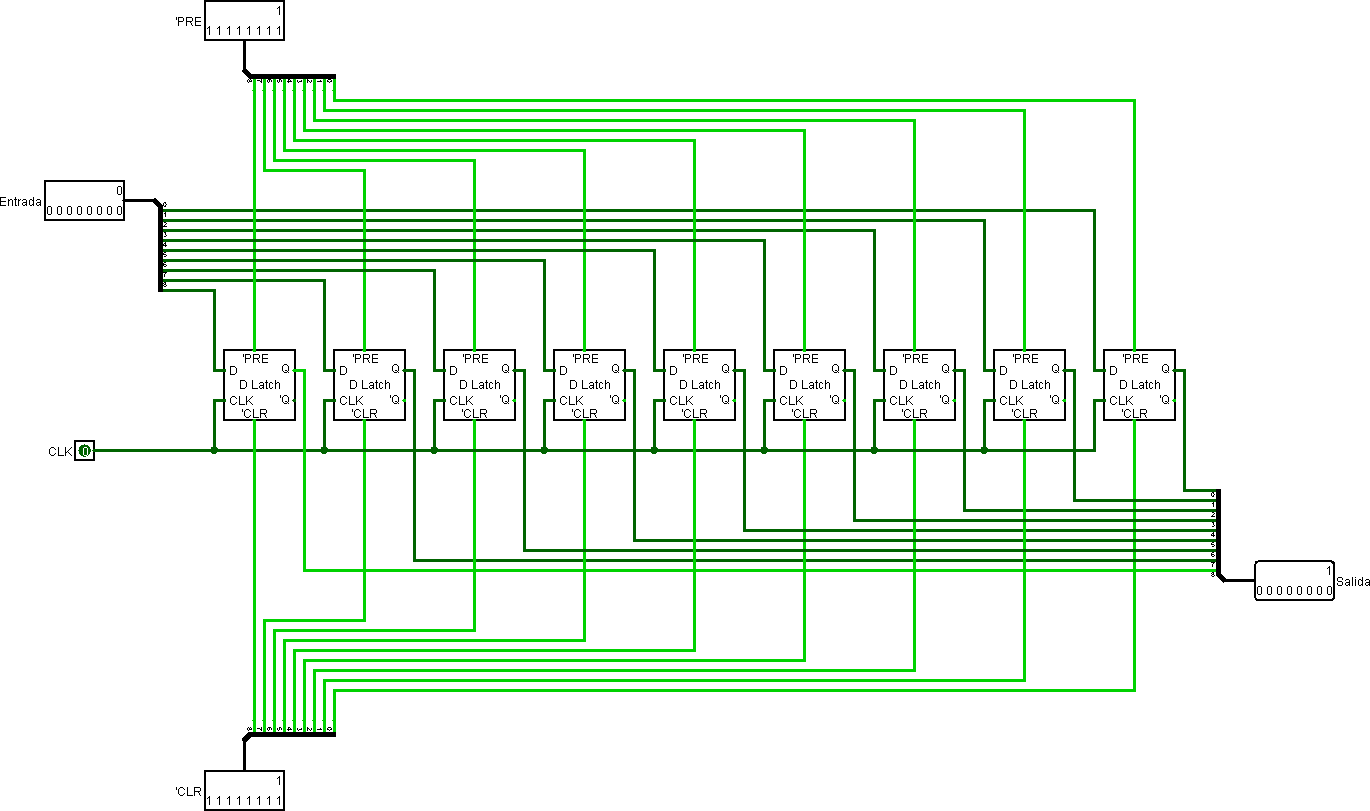
**Figura 13.** Almacén de Control en Logisim.

## MPC:

Su función principal es mantener la dirección de la próxima microinstrucción a ejecutar en el almacén de control. Cada microinstrucción en el almacén de control tiene una dirección única, y el MPC se encarga de rastrear y seleccionar la siguiente microinstrucción que debe ser ejecutada.

Este puede ser actualizado de diferentes maneras:

* Incrementarse secuencialmente para ejecutar la siguiente microinstrucción en orden.
* Ser cargado con una nueva dirección basada en ciertos campos de la microinstrucción actual, permitiendo saltos dentro del programa de microinstrucción.



**Figura 14.** MPC en Logisim.

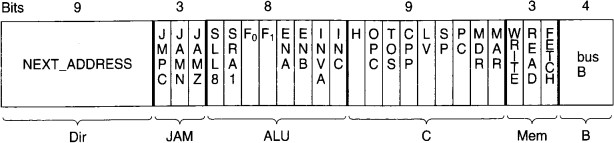
## MIR:

Su función es almacenar la microinstrucción que se encuentra actualmente en ejecución. Cuando una microinstrucción es leída desde el almacén de control, se carga en el MIR. El contenido de este es lo que la unidad de control utiliza para generar señales de control

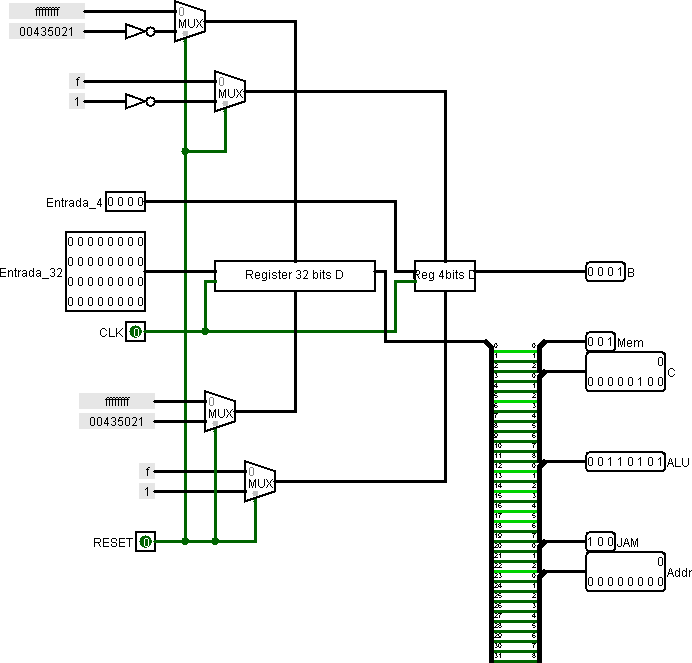
necesarias para la ejecución de la microinstrucción.

Cada microinstrucción en el MIR contiene campos que:

* Especifican las operaciones a realizar en la UAL.
* Determinan qué registros deben ser leídos o escritos.
* Controlan las señales de entrada y salida.
* Indican las condiciones para los saltos y la actualización del MPC.



**Figura 15.** Formato de Microinstrucciones.



**Figura 16.** MIR en Logisim.

## Microprograma:

El microprograma es una secuencia de microinstrucciones almacenadas en el almacén de control de la MIC-1. Cada una de ellas especifica una serie de operaciones de bajo nivel que se deben realizar para ejecutar una instrucción de máquina. Este programa desempeña un papel fundamental en la arquitectura al traducir las instrucciones de alto nivel del programa en una serie de pasos detallados que el hardware del procesador puede llevar a cabo.

El microprograma descompone las instrucciones de máquina en microinstrucciones

individuales que controlan las operaciones internas del procesador. Cada instrucción de máquina tiene una secuencia correspondiente en el microprograma. Además, el

microprograma incluye instrucciones para manejar el control de flujo, como saltos

condicionales e incondicionales, utilizando el MPC para seguir la secuencia correcta.

Cada microinstrucción contiene campos que son decodificados en señales de control por el decodificador. Estas señales dirigen las operaciones de los diferentes componentes del procesador, asegurando que las tareas se realicen en el orden correcto y que los datos se muevan adecuadamente entre los distintos elementos del sistema.

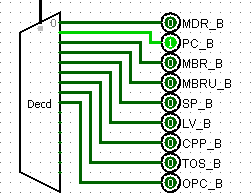
## Decodificador:

El decodificador es un componente crucial dentro de la MIC-1. Las microinstrucciones en el MIR contienen varios campos que representan diferentes operaciones necesarias para la ejecución de una instrucción a nivel de microarquitectura. El decodificador traduce estos

campos en señales de control específicas que pueden ser entendidas y ejecutadas por los componentes de la MIC-1. Estas señales de control se utilizan para activar y desactivar componentes como la UAL, los registros, la memoria y los buses.

Por ejemplo, las señales de control pueden indicar qué operación debe realizar la UAL, qué registros deben ser leídos o escritos, o cómo deben ser direccionadas las operaciones de memoria.

El decodificador también ayuda a controlar el flujo de datos dentro de la microarquitectura, asegurando que las operaciones se realicen en el orden correcto y que los datos se muevan adecuadamente entre los diferentes componentes.



**Figura 17.** Decodificador dentro de la Unidad de Control en Logisim.

# Conclusión

Por último y para dar cierre, consideramos que es necesario resumir todo lo que hemos aprendido en este proyecto, donde exploramos a detalle todos los componentes y sus

funciones dentro de la microarquitectura MIC-1.

Como dijimos al comienzo, la MIC-1 se divide conceptualmente en dos grandes partes: El Camino de Datos, el núcleo, donde se producen las operaciones aritméticas-lógicas a través de la UAL, y se guardan los datos mediante los diversos Registros con funciones como *Load* y *Hold*, pasando todo esto a través del Desplazador pudiendo hacer uso de su capacidad para realizar un desplazamiento lógico o aritmético, llevando así la información a su salida para uno de los dos Buses, los cuales se encargan de transportar datos binarios y direcciones de memoria. Y la segunda gran parte conceptual, la Unidad de Control, se encarga de marcar un ritmo para coordinar y gestionar el cuándo y en qué orden se van a realizar las distintas

operaciones almacenadas como secuencias de microinstrucciones en el Microprograma, que a su vez se encuentra en el Almacén de Control, el cual es funcionalmente una memoria y por lo mismo hace uso de dos registros especializados: el MPC, encargado de contener la dirección de memoria de la próxima microinstrucción a ejecutar; y el MIR, el cual almacena

la microinstrucciones que se encuentra en ejecución, en donde el contenido funciona como la alimentación del Decodificador, el cual traduce el contenido en señales de control.

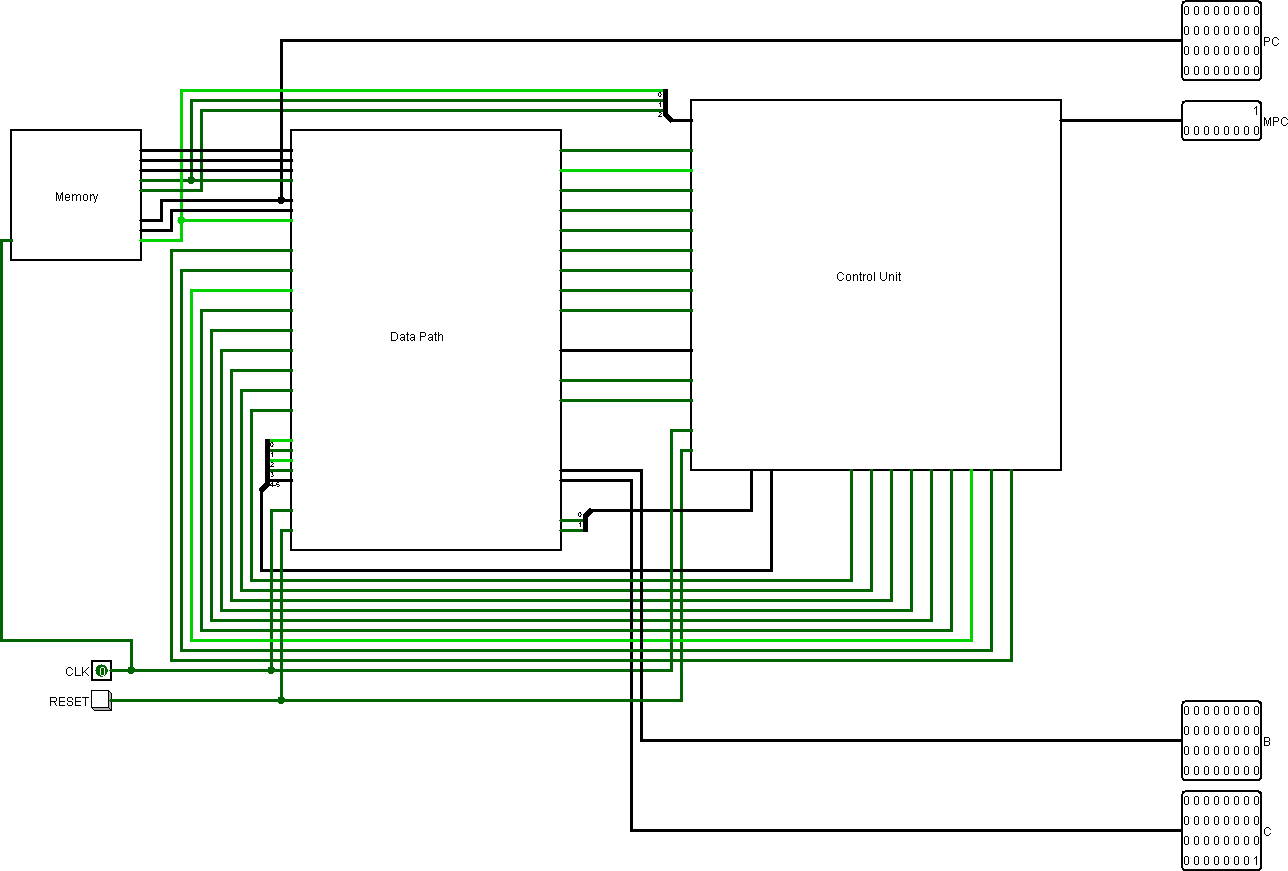
Además, a lo largo del proyecto entendimos cómo las instrucciones de alto nivel son desglosadas en operaciones relativamente más sencillas, lo que permite descomponer

operaciones complejas en pasos más simples que una CPU puede ejecutar, y en conjunto con el uso del programa Logisim y su interfaz sencilla, fuimos capaces de comprender lo visto

durante lo largo de la asignatura, de una manera completa, didáctica, interactiva y visual.

En resumen, este proyecto nos ha permitido desglosar y comprender los componentes y procesos esenciales de la microarquitectura MIC-1, proporcionando así una valiosa

experiencia sobre la Arquitectura y Organización de Computadoras.



**Figura 18.** MIC-1 en Logisim.