# *Práctica 1*

# Arquitectura de Computadoras
# Diseño de una computadora basada en un acumulador

Fernanda Mora, ITAM 9/02/2016

## 1. Objetivos

Los objetivos de esta práctica son los siguientes:

- Dada la especificación de una computadora, implementarla en logisim y comprobar su funcionamiento ejecutando programas.
- Comprender claramente la integración de un computador sencillo. En particular, entender la forma en que la funciones de control gobiernan los demás componentes del sistema.

## 2. Implementación

### Especificaciones generales

- Se trata de una computadora basada en acumulador
- La longitud de palabra será de cuatro bits
- Se usará como memoria principal una RAM de 16*4
- Existen cuatro códigos de operación diferentes
- Se tiene un total de 20 instrucciones, de las cuales 16 son aritméticas y lógicas
- Un programa finaliza con una instrucción “VETE A” esa misma dirección. El proceso se podrá detener por medio de una interrupción externa
- El inicio de un programa se realiza por medio de una interrupción externa
- Las instrucciones que no son aritméticas ni lógicas son CICLO DE BÚSQUEDA, LEE, GUARDA, VETE A Y OPERA.

A continuación se muestra la especificación de las instrucciones de la computadora:

| Nombre | CodOp | Señal | Pal.1 | Pal.2 | Pal.3 |
|---|---|---|---|---|---|
| LEE | 0001 | $q_0$ | 0001 | dir | ... |
| GUARDA | 0010 | $q_1$ | 0010 | dir | ... |
| VETE A | 0011 | $q_2$ | 0011 | dir | ... |
| OPERA | 1000 | $q_3$ | 1000 | tipo | ... |

Cuadro 1: Especificación de las instrucciones

- dir: es la dirección de memoria RAM
- tipo: es el tipo de operación para programar la ALU
- Op.2: es el segundo operando para la ALU
- LEE: A <- M[dir]
- GUARDA: M[dir] <- A
- VETE A: PC<-dir
- OPERA: A<-A tipo Op.2

Y los ciclos de operación son los mismos que se vieron en clase, por lo que ya no se vuelven a mencionar en el reporte.

### Esqueleto

Lo primero que se hizo fue pintar el esqueleto de la computadora en papel. Para eso se dibujaron la unidad de control, de procesamiento y el output (no el input en este caso).

Posteriormente se dibujaron los registros básicos necesarios: PC, MAR MBR, Acc, IR, ROP, Salida y por otro lado la Memoria y la ALU.

Posteriormente se usarion lod cilos de operación sin tomar en cuenta los tiempos para hacer las conexiones entre dichos elementos.

### Señales de Control

Usando los ciclos de operación se obtuvieron las señales de control:

- $MAR \leftarrow PC, X_1 = t_0 + q_3t_4 + t_2$
- $MAR \leftarrow MBR, X_2 = q_0t_4 + q_1t_4$
- $PC \leftarrow PC + 1, X_3 = t_1 + q_0t_3 + q_1t_3 + q_2t_3 + q_3t_3 + q_3t_5$
- $MBR \leftarrow M, X_4 = t_1 + q_0t_5 + q_0t_3 + q_1t_3 + q_2t_3 + q_3t_3 + q_3t_5$
- $PC \leftarrow MBR, X_5 = q_2t_4$
- $A \leftarrow MBR, X_6 = q_0t_6$

- $MBR \leftarrow A, X_7 = q_1 t_5$
- $M \leftarrow MBR, X_8 = q_1 t_6$
- $ROP \leftarrow MBR, X_9 = q_3 t_4$
- $A \leftarrow AtipoMBR, X_{10} = q_3 t_6$
- $t \leftarrow 0, X_1 1 = q_0 t_6 + q_9 t_6 + q_2 t_4 + q_3 t_6$
- $IR \leftarrow MBR, X_1 2 = t_2$

Y entonces se añadió la parte de Control a la estructura usando un Decodificador para instrucciones y otro Decodificador para los tiempos que mediara la parte de Control y las instrucciones y decidiera qué se ejecuta y cuándo se ejecuta.

## Diagrama en Logisim

A continuación se muestra el diagrama de la computadora especificada implementado en Logisim. Cabe mencionar que una posible mejora es diseñarla modularmente para facilitar su lectura y comprensión.

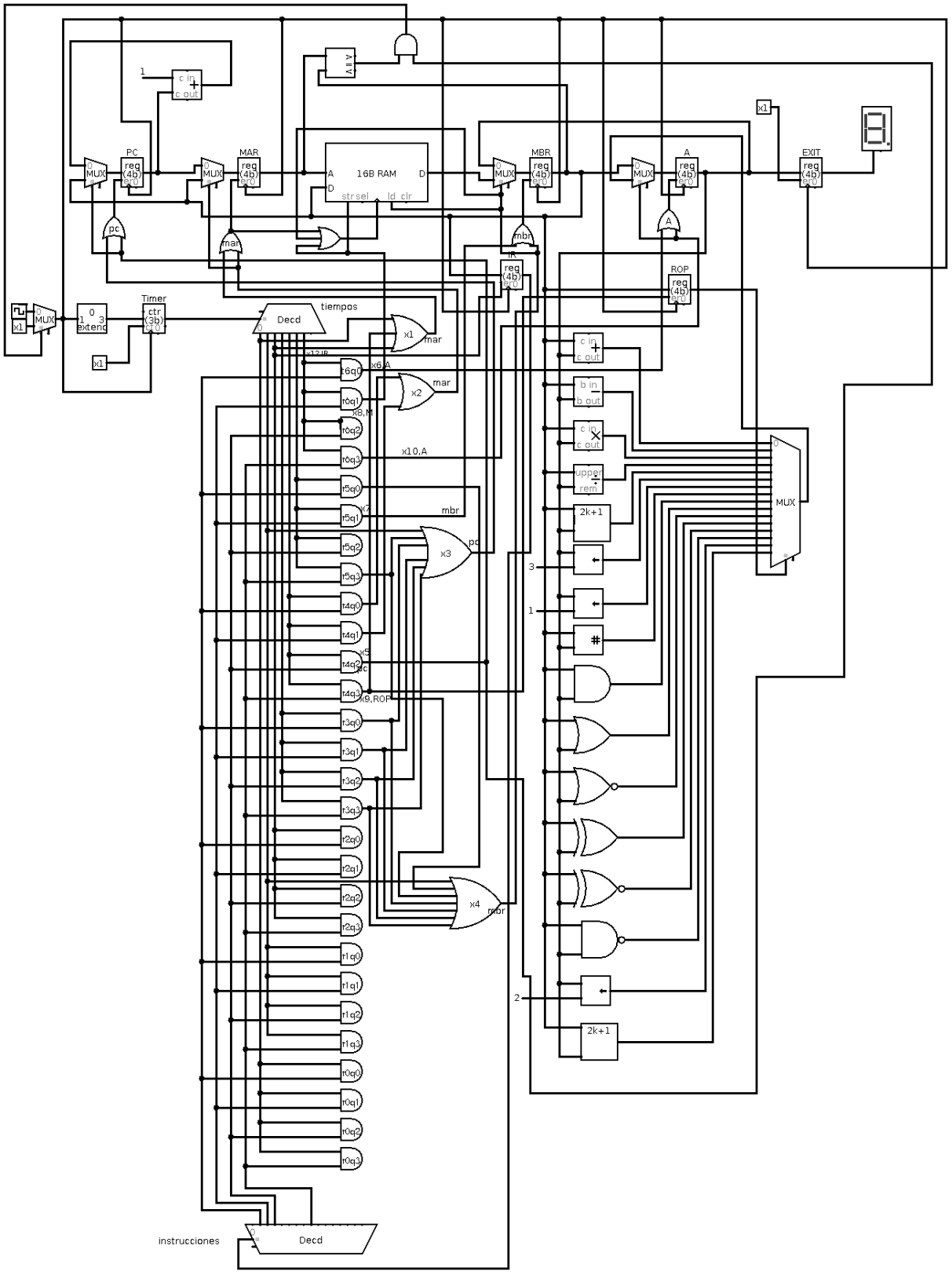


Figura 1: Diseño de la computadora en Logisim

### Pruebas

Para probar nuestra computadora se pidió un programa en pseudo-lenguaje de alto nivel para mostrar que se ejecuta correctamente.

Ese programa consistió por simplicidad en probar las 4 instrucciones requeridas: LEE, GUARDA, VETE A, OPERA, las cuales se ejecutaron correctamente después hacer un par de modificaciones a la computadora.

Programas más sofisticados podrían ser probados pero con las instrucciones consideramos que era suficiente para los fines del ejercicio, que era probar el programa.

Para probar el código asociado a la computadora se puede consultar el siguiente link y abrir el diseño en Logisim: Código en Github.

## 3. Preguntas adicionales

- ¿Qué opciones tiene para diseñar la unidad de control?

  Existen dos opciones: *microprogramado*. y *control alambrado*.

  La microprogramada genera señales de control usando una memoria llamada almacenamiento de control (CS, Control storage por sus siglas en inglés). A pesar de que el control microprogramado en teoría es ventajoso para las arquitecruas CISC (Complex Instruction Set Computers), dado que CISC requiere un sofisticado desarrollo de las señales de control, entonces no hay una diferencia intrínseca real entre estos dos mecanismos.

  El control alambrado genera las señales de control usando una máquina de estado finito. El registro de microconstrucción y el registro de control de almacenamiento de memoria se pueden considerar como un *estado de registro* para el control de alambrado. El almacenamiento de control puede ser considerado como un tipo de circuito lógico combinatorio.

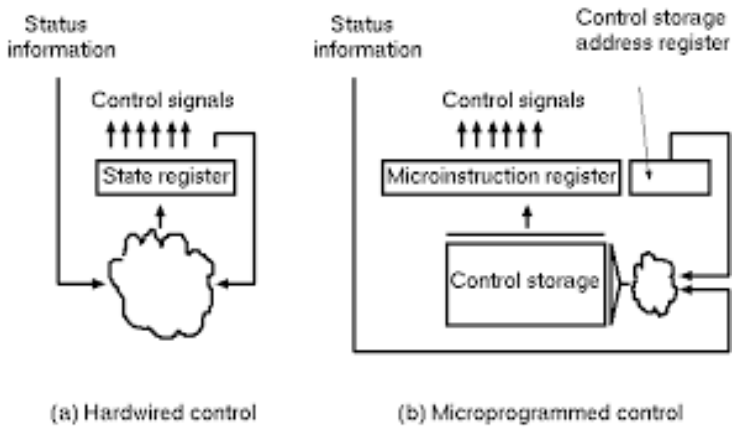


Figura 2: Control alambrado vs microprogramado

- Si la ALU permite realizar 48 operaciones (P.Ej. CI 74181), ¿qué cambios deberían hacerse en la computadora? Necesitamos poder codificar la selección de la operación de entrada de la ALU, entonces teóricamente necesitamos 6 bits, pues $2^5 = 32 < 48 <$

$64 = 2^6$.Pero para lograr la consistencia en nuestra computadora, sería necesario también cambiar los demás componentes a 6 bits: MBR, ROP, IR, *word size* de la Memoria, etc.

Sin embargo, como mencionamos arriba, lo anterior fue *teórico*. Si se quisiera de hecho construir dicha computadora entonces seria necesario usar 8 bits en vez de 6 porque no hay piezas físicas de 6 bits.

- ¿Qué modificaciones deben hacerse para convertir esta computadora en una de tres direcciones? ¿La arquitectura se modificaría?

  Hay que cambiar algunas cosas, pero otras no. Por ejemplo. dado que la computadora diseñada está basada en un acumulador, es decir , sólo tiene una dirección de memoria. Para expandirla a tres direcciones hay que añadir dos registros. El primer registro seria para el segundo operando y el segundo registro para la salida de la ALU.

  Por otro lado la ALU no cambiaría porque su tarea, como su nombre lo indica, es únicamente realizar operaciones aritméticas y lógicas.

  Hasta aquí parece que la *arquitectura* no cambia y notar que la funcionalidad definitivamente no cambia, es decir, nuestra computadora *puede* hacer lo mismo que antes. Sin embargo, hay que observar que la definición de ciclos de operación sí cambian, tal como lo vimos en clase con las 4 computadoras. Esto modificará a su vez la unidad de control. Por lo tanto, la *arquitectura* sí cambia.

## 4. Conclusiones

Diseñar una computadora en papel no es fácil. Mucho menos lo es hacerla en Logisim. Sin embargo, esta práctica ayudó a practicar los conceptos vistos en clase y a reforzarlos vía papel. Los retos aquí fueron teóricos y conceptuales pricipalmente. Posteriormente se implementaron en Logisim y se encontraron nuevas dificultades: de software. Tomó un rato entender el programa pero después de algunos intentos se empezó a tomar confianza de las herramientas disponibles. Esto me dio un mejor *grasp* de lo que es construir una computadora sencilla. Algunas mejoras como trabajo modular en Logisim podrían ayudar a la simplicidad del diseño.

## 5. Referencias

[1] Patterson, David A., and John L. Hennessy. *Computer organization and design: the hardware/software interface*. Newnes, 2013.

[2] Patt, Yale N., Sanjay J. Patel, and J. Patel. *Introduction to computing systems: from bits and gates to C and beyond*. 2004.