**CENTRO REGIONAL UNIVERSITARIO CÓRDOBA IUA**



## “Cyclone IV”

**ARQ. DE COMPUTADORAS I - INGENIERÍA EN INFORMÁTICA PROYECTO INTEGRADOR**

**PROFESOR**

Toledo, Luis Eduardo

## INTEGRANTES

Cardetti Facchin, Agustin (acardetti298@alumnos.iua.edu.ar) Doffo, Camila([cdoffo674@alumnos.iua.edu.ar](mailto:cdoffo674@alumnos.iua.edu.ar))

Gomez Bonelli, Carlos Emanuel (cgomez453@alumnos.iua.edu.ar)

## 17 de noviembre del año 2023

|  | **Índice General** |  |
| --- | --- | --- |
| **Índice General** |  | **1** |
| **Introducción** |  | **2** |
| **Objetivos** |  | **3** |
| **Desarrollo** |  | **4** |
| Observaciones |  | 4 |
| Funcionamiento |  | 6 |
| **Instrucciones** |  | **10** |
| Menú |  | 10 |
| Selección |  | 10 |
| Secuencias |  | 12 |
| **Conclusiones** |  | **16** |
| **Bibliografía** |  | **17** |

**Introducción**

El siguiente trabajo consiste en la implementación de conceptos trabajados en la materia Arquitectura de Computadoras 1. La finalidad del proyecto es manipular una placa DE0-Nano con funcionalidades que permitan la interacción del usuario con secuencias de luces. Toda la codificación del proyecto se encuentra dada en el lenguaje de programación *Assembly*.

# Objetivos

Este proyecto tiene como finalidad los siguientes objetivos:

* Realizar un menú, el cual dé a elegir cuatro secuencias de luces diferentes, dos de ellas tienen que ser el “Auto fantástico”, ejecutada a través de un algoritmo, y el “La Carrera” ejecutada mediante tabla. Las otras dos, deberán crearlas los alumnos teniendo el mismo formato que las anteriores.
* Implementar el menú y las cuatro secuencias en el simulador del IDE Quartus 1.19.
* Cargar el código en hexadecimal a la placa “De0-nano” y ejecutarlo.
* Realizar una opción para cambiar las velocidades de las secuencias.

# Desarrollo

A continuación, vamos a explicar la operatoria que seguimos para llevar a cabo el proyecto

## Observaciones

* Distribución de pines
  + INport: A continuación, se detalla los pines correspondientes a la señal de entrada *INport* del módulo *top*

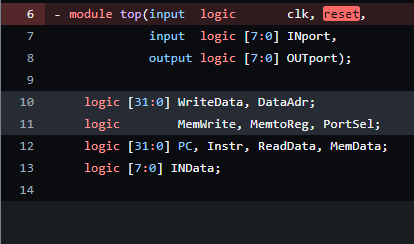
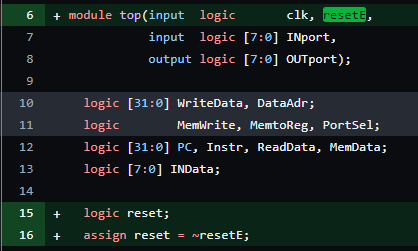
| 1 | 1 | 1 | J15 | M15 | B9 | T8 | M1 |
| --- | --- | --- | --- | --- | --- | --- | --- |

Observar que, los tres bits más significativos no han sido asignados a ningún pin, lo que implica que, por defecto, su valor será 1.

* + OUTport: A continuación, se detalla los pines correspondientes a la señal de salida *OUTport* del módulo *top*

| L3 | B1 | F3 | D1 | A11 | B13 | A13 | A15 |
| --- | --- | --- | --- | --- | --- | --- | --- |

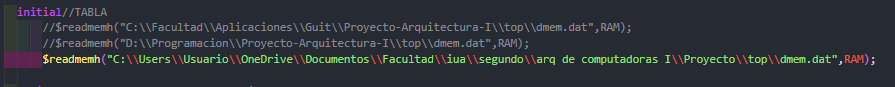
* + resetE: Para esta señal utilizamos el botón key\_0 asignado al pin E1.
  + clk: Para esta señal utilizamos el pin R8 asignado al clock de 50 MHz propio de la placa.
* Funcionamiento de la placa

Dado que los pines E1 y J15, correspondientes a los botones key\_0 y key\_1 tienen funcionamiento por bajo (es decir, mientras no están presionados, la señal que emiten corresponde a un uno lógico, mientras que cuando son presionados, emiten un cero lógico) realizamos la siguiente corrección al código del ARM reducido implementado para llevar a cabo el proyecto.

La corrección consiste en la negación de la señal externa del clock (resetE en la imagen de la derecha) proveniente de la placa, valor que luego asignamos a una señal interna (reset) para sustituir la señal original.

* Lectura de tabla de datos

Para poder implementar secuencias leídas desde tabla, tuvimos que insertar las siguientes líneas de código al ARM reducido.



permitiendo así la lectura de una tabla de datos.

### Funcionamiento

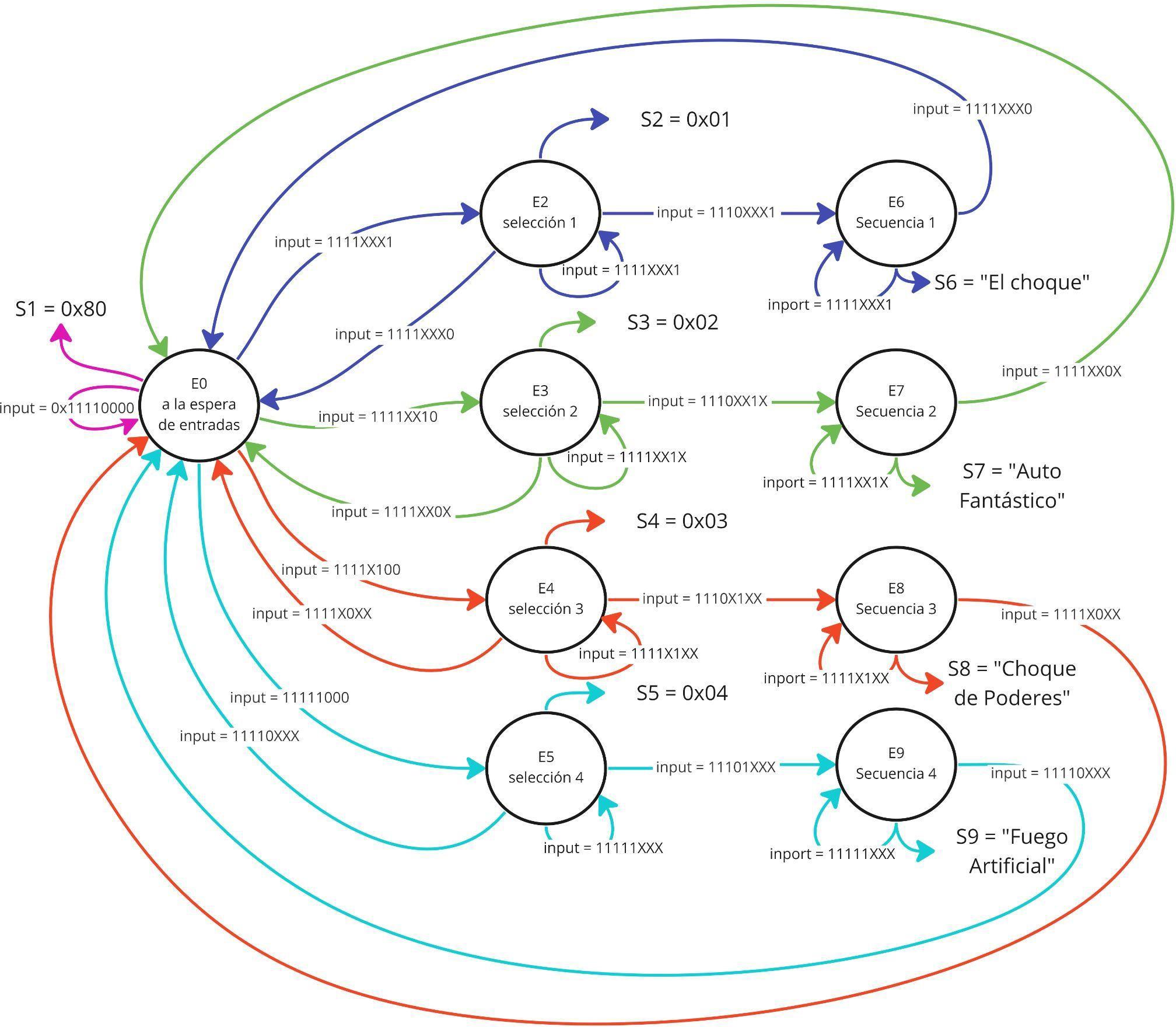
* Entradas

Mediante la siguiente tabla presentamos las distintas combinaciones de la señal

*INport* y su correspondencia a las distintas funcionalidades del programa.

| **Operación** | **Pin E1** | **Pin M15** | **Pin B9** | **Pin T8** | **Pin M1** |
| --- | --- | --- | --- | --- | --- |
| **Selección 1** | 1 | X | X | X | 1 |
| **Salir Sel. 1** | 1 | X | X | X | 0 |
| **Selección 2** | 1 | X | X | 1 | 0 |
| **Salir Sel. 2** | 1 | X | X | 0 | X |
| **Selección 3** | 1 | X | 1 | 0 | 0 |
| **Salir Sel. 3** | 1 | X | 0 | X | X |
| **Selección 4** | 1 | 1 | 0 | 0 | 0 |
| **Salir Sel. 4** | 1 | 0 | X | X | X |
| **Secuencia 1** | 0 | X | X | X | 1 |
| **Salir Sec. 1** | 1 | X | X | X | 0 |
| **Secuencia 2** | 0 | X | X | 1 | X |
| **Salir Sec. 2** | 1 | X | X | 0 | X |
| **Secuencia 3** | 0 | X | 1 | X | X |
| **Salir Sec. 3** | 1 | X | 0 | X | X |
| **Secuencia 4** | 0 | 1 | X | X | X |
| **Salir Sec. 4** | 0 | 0 | X | X | X |

Esta tabla también puede ser representada mediante el siguiente diagrama de estados



* Ilustración de secuencias

Nota: El carácter “-” representa un led apagado mientras que el carácter “\*” representa un led prendido.

* La carrera:

-------\*

-------\*

------\*-

------\*-

----\*---

----\*---

---\*---\*

---\*--\*-

--\*--\*--

--\*-\*---

-\*-\*----

-\*\*-----

\*\*------

\*-------

* El Auto Fantástico:

\*-------

-\*------

--\*-----

---\*----

----\*---

-----\*--

------\*-

-------\*

------\*-

-----\*--

----\*---

---\*----

--\*-----

-\*------

\*-------

* La Sirena:

\*\*\*\*----

----\*\*\*\*

* Encierro:

--------

\*-------

-\*------

--\*-----

---\*----

----\*---

-----\*--

------\*-

-------\*

------\*\*

-----\*-\*

----\*--\*

---\*---\*

--\*----\*

-\*-----\*

\*------\*

\*\*-----\*

\*-\*----\*

\*--\*---\*

\*---\*--\*

\*----\*-\*

\*-----\*\*

\*----\*\*\*

\*---\*-\*\*

\*--\*--\*\*

\*-\*---\*\*

\*\*----\*\*

\*\*\*---\*\*

\*\*-\*--\*\*

\*\*--\*-\*\*

\*\*---\*\*\*

\*\*--\*\*\*\*

\*\*-\*-\*\*\*

\*\*\*--\*\*\*

\*\*\*\*-\*\*\*

\*\*\*\*\*\*\*\*

Computadoras I





# Instrucciones

### Menú

sub r0, pc, pc //=> E04F000F

add r1, r0, #1 //=> E2801001

ldr r6, [r0, #8] //=> E5906008

add r2, r0, r0 //=> E0802000

add r3, r0, r0 //=> E0803000

add sb, r0, #0x80 //=> E2809080

str sb, [r0, #0x800] //=> E5809800

ldr r4, [r0, #0x800] //=> E5904800

add r2, r2, #1 //=> E2822001

ands r7, r4, #0x10 //=> E2147010

bne #0x1c //=> 1AFFFFFB

subs r5, r6, r2 //=> E0565002

addlt r3, r3, #1 //=> B2833001

add sb, sb, sb //=> E0899009

str sb, [r0, #0x800] //=> E5809800

bge #0x44 //=> AA000000

blt #0x1c //=> BAFFFFF5

subs r8, r3, #1 //=> E2538001

beq #0x64 //=> 0A000005

subs r8, r3, #2 //=> E2538002

beq #0x78 //=> 0A000008

subs r8, r3, #3 //=> E2538003

beq #0x7c //=> 0A000007

subs r8, r3, #4 //=> E2538004

beq #0x88 //=> 0A000008

add sb, r0, #0x81 //=> E2809081

str sb, [r0, #0x800] //=> E5809800

b #0x94 //=> EA000008

add sb, r0, #0x41 //=> E2809041

str sb, [r0, #0x800] //=> E5809800

b #0x94 //=> EA000005

add sb, r0, #0x21 //=> E2809021

str sb, [r0, #0x800] //=> E5809800

b #0x94 //=> EA000002

add sb, r0, #0x11 //=> E2809011

str sb, [r0, #0x800] //=> E5809800

b #0x94 //=> EAFFFFFF

b #0x94 //=> EAFFFFFE

**Selección**

### Secuencias

# Conclusiones

El trabajar con un lenguaje de bajo nivel como *Assembly* nos ha permitido observar más de cerca cómo funcionan los microprocesadores y qué operaciones llevan a cabo. Al realizar las distintas operaciones que nos permitían introducir secuencias a la placa, pudimos familiarizarnos un poco más con dicha complejidad de un lenguaje de bajo nivel. A su vez, dicha complejidad se vió reforzada por el uso de una ARM con operaciones reducidas, pero que al final logramos entender y encontrar las soluciones para sus distintas limitaciones, lo que nos permitió cumplir, en su gran mayoría, con los objetivos planteados al comienzo del trabajo.

# Bibliografía

* Harris, S. L., & Harris, D. (2015). *Digital design and computer architecture*. Morgan Kaufmann.
* Terasic Technologies (2003). *DE0-Nano User Manual.*