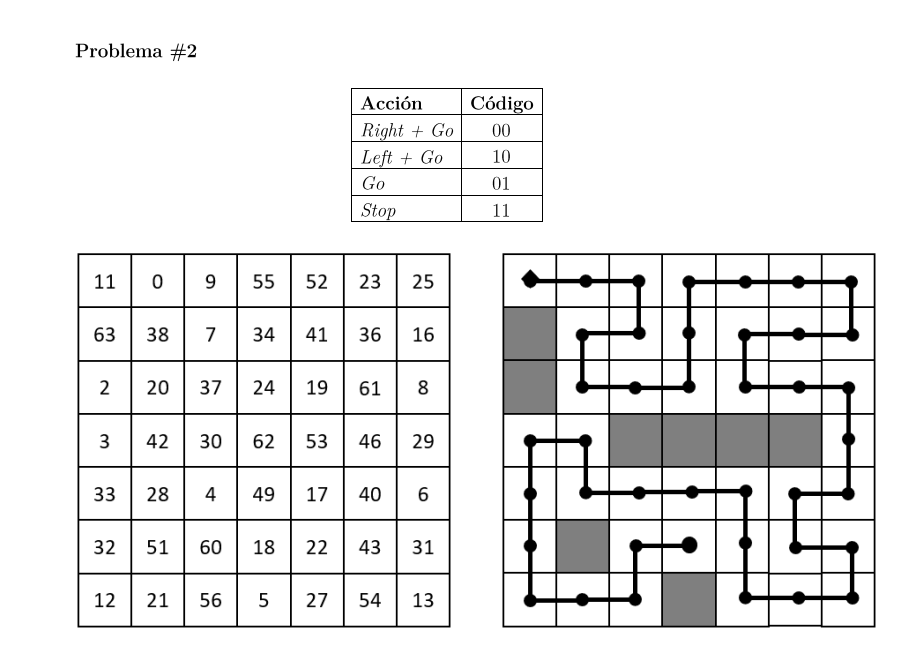
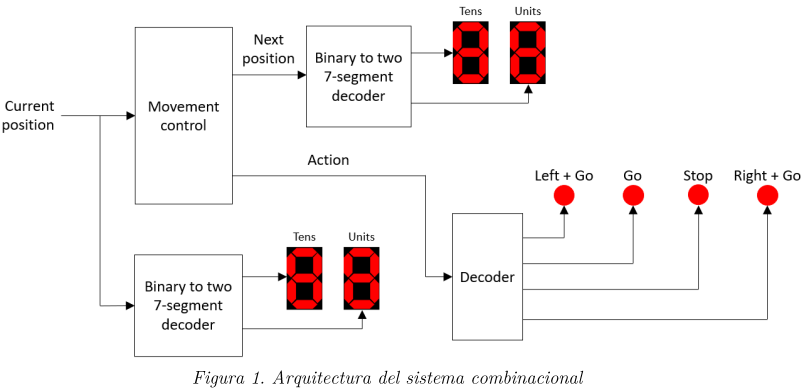
# Problema Asignado

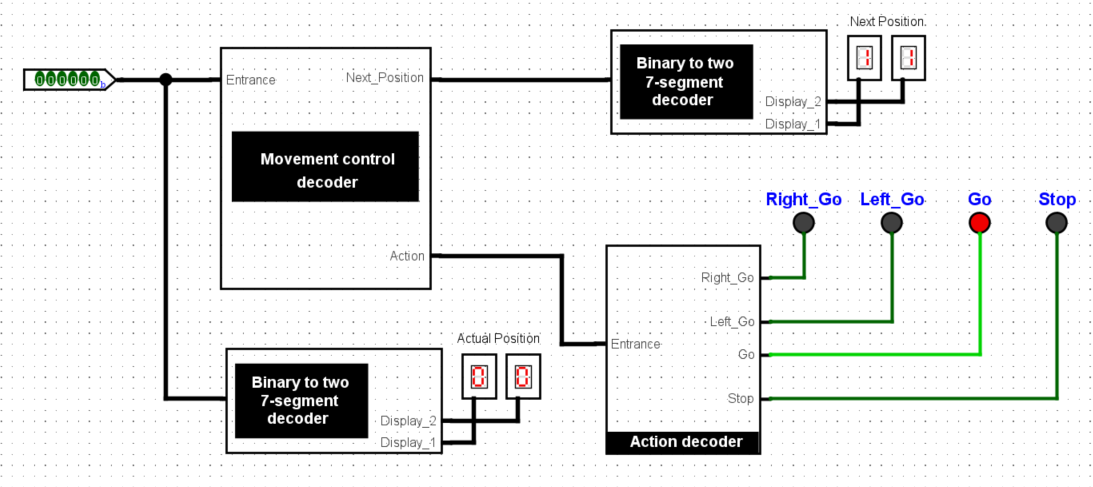


# Arquitecturas

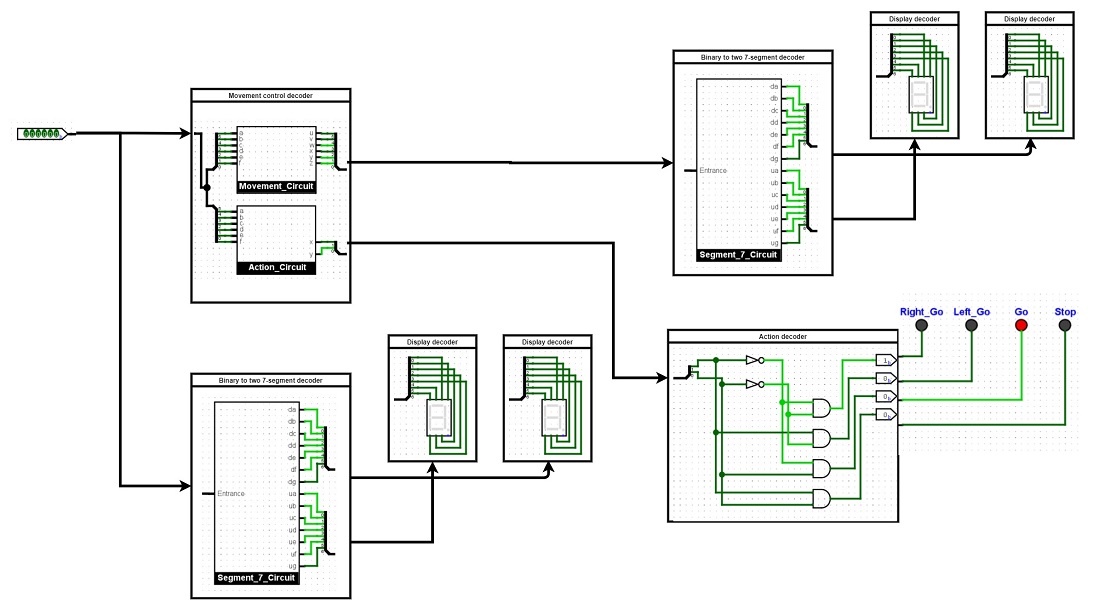
## Arquitectura Propuesta



## Arquitectura de la solución



## Arquitectura del Diseño de Circuito



# Decisiones de Diseño

Se ha tomado diferentes decisiones con respecto al diseño del circuito para llegar a la arquitectura propuesta y requerida.

1. Se usa separadores de bits para enlazar por medio de un único cable los diferentes componentes o decoders tal y como la arquitectura propuesta lo exige.
2. Se crea un componente Display\_decoder para instanciar los 4 displays requeridos en la solución y este se crea también teniendo en cuenta el punto 1 anterior.
3. Se crean componentes principales: **Movement control decoder**, **Binary to two 7-segment decoder**, **Action decoder.** Los cuales cada uno instancia sub componentes o circuitos combinacionales previamente diseñados.
4. De acuerdo al problema #2 que me correspondió, se toma el número mayor usado en este problema el cual es el: **63** que en binario corresponde a **111111** igual a **6 bits.** Partiendo de esto, decido que la entrada de datos del robot será de 6 bits.
5. Para tener una consistencia de datos, las entradas de los componentes: **Binary to two 7-segment decoder** y **Movement control decoder** corresponden a 6 bits incluyendo la entrada de los sub circuitos combinacionales que contienen estos.
6. El diseño de los sub circuitos combinacionales se hacen partiendo de tablas de verdad con entradas de 6 bits para todos sus circuitos, excepto para el circuito **Circuit\_Action\_Decoder** (que es el mismo componente **Action Decoder**) debido a que la entrada de este será en 2 bits.

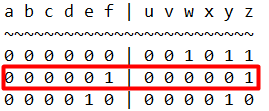
# Proceso de Diseño

Se inicia diseñando las tablas de verdad para cada sub circuito combinacional.

## - Circuit\_Movement

### Análisis

* Se crean entradas de 6 bits para los 64 números posibles a usar por el robot.
* Las entradas de los 64 números se crean en orden de 0 a 63, siendo cada entrada el número correspondiente en binario y la salida la posición siguiente que debería tomar el robot.
* Para los números que no están mapeados en la matriz del problema, como los son: (1,2,5,10,14,15,26,30,35,39,44,45,46,47,48,50,51,53,57,58,59,62,63,64) tendrán como salida el mismo número. Ejemplo del número 1:

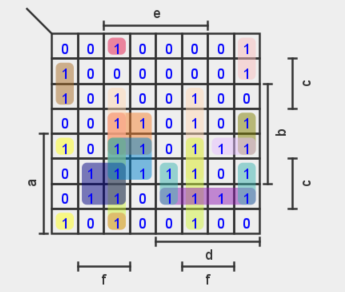


* De este circuito la entrada proviene del usuario y la salida va directa a los display que mostrarán la posición siguiente

### Tabla de Verdad

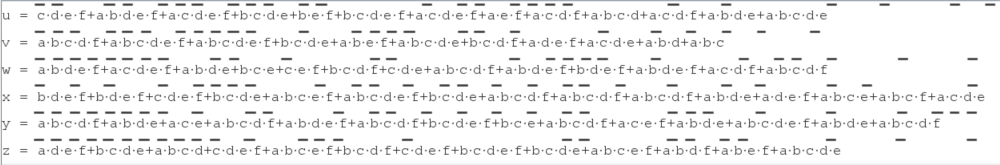
|  |
| --- |
| a b c d e f | u v w x y z  ~~~~~~~~~~~~~~~~~~~~~~~~~  0 0 0 0 0 0 | 0 0 1 0 1 1  0 0 0 0 0 1 | 0 0 0 0 0 1  0 0 0 0 1 0 | 0 0 0 0 1 0  0 0 0 0 1 1 | 1 0 1 0 1 0  0 0 0 1 0 0 | 1 1 0 0 0 1  0 0 0 1 0 1 | 0 0 0 1 0 1  0 0 0 1 1 0 | 0 1 1 1 0 1  0 0 0 1 1 1 | 0 0 1 0 0 1  0 0 1 0 0 0 | 1 1 1 1 0 1  0 0 1 0 0 1 | 0 0 0 0 0 0  0 0 1 0 1 0 | 0 0 1 0 1 0  0 0 1 0 1 1 | 0 0 1 0 1 1  0 0 1 1 0 0 | 1 0 0 0 0 0  0 0 1 1 0 1 | 0 1 1 1 1 1  0 0 1 1 1 0 | 0 0 1 1 1 0  0 0 1 1 1 1 | 0 0 1 1 1 1  0 1 0 0 0 0 | 0 1 1 0 0 1  0 1 0 0 0 1 | 0 1 0 1 1 0  0 1 0 0 1 0 | 1 1 1 1 0 0  0 1 0 0 1 1 | 1 0 1 0 0 1  0 1 0 1 0 0 | 1 0 0 1 1 0  0 1 0 1 0 1 | 0 0 1 1 0 0  0 1 0 1 1 0 | 0 1 1 0 1 1  0 1 0 1 1 1 | 1 1 0 1 0 0  0 1 1 0 0 0 | 1 0 0 1 0 1  0 1 1 0 0 1 | 0 1 0 1 1 1  0 1 1 0 1 0 | 0 1 1 0 1 0  0 1 1 0 1 1 | 1 1 0 1 1 0  0 1 1 1 0 0 | 0 0 0 1 0 0  0 1 1 1 0 1 | 0 0 1 0 0 0  0 1 1 1 1 0 | 0 1 1 1 1 0  0 1 1 1 1 1 | 1 0 1 0 1 1  1 0 0 0 0 0 | 1 0 0 0 0 1  1 0 0 0 0 1 | 0 0 0 0 1 1  1 0 0 0 1 0 | 0 1 1 0 0 0  1 0 0 0 1 1 | 1 0 0 0 1 1  1 0 0 1 0 0 | 0 1 0 0 0 0  1 0 0 1 0 1 | 0 1 0 1 0 0  1 0 0 1 1 0 | 0 0 0 1 1 1  1 0 0 1 1 1 | 1 0 0 1 1 1  1 0 1 0 0 0 | 0 0 0 1 1 0  1 0 1 0 0 1 | 1 0 0 1 0 0  1 0 1 0 1 0 | 0 1 1 1 0 0  1 0 1 0 1 1 | 1 0 1 0 0 0  1 0 1 1 0 0 | 1 0 1 1 0 0  1 0 1 1 0 1 | 1 0 1 1 0 1  1 0 1 1 1 0 | 1 0 1 1 1 0  1 0 1 1 1 1 | 1 0 1 1 1 1  1 1 0 0 0 0 | 1 1 0 0 0 0  1 1 0 0 0 1 | 0 1 0 0 0 1  1 1 0 0 1 0 | 1 1 0 0 1 0  1 1 0 0 1 1 | 1 1 0 0 1 1  1 1 0 1 0 0 | 1 1 0 1 1 1  1 1 0 1 0 1 | 1 1 0 1 0 1  1 1 0 1 1 0 | 0 0 1 1 0 1  1 1 0 1 1 1 | 1 0 0 0 1 0  1 1 1 0 0 0 | 0 1 0 1 0 1  1 1 1 0 0 1 | 1 1 1 0 0 1  1 1 1 0 1 0 | 1 1 1 0 1 0  1 1 1 0 1 1 | 1 1 1 0 1 1  1 1 1 1 0 0 | 1 1 1 0 0 0  1 1 1 1 0 1 | 0 1 0 0 1 1  1 1 1 1 1 0 | 1 1 1 1 1 0  1 1 1 1 1 1 | 1 1 1 1 1 1 |

### Expresión Minimizada

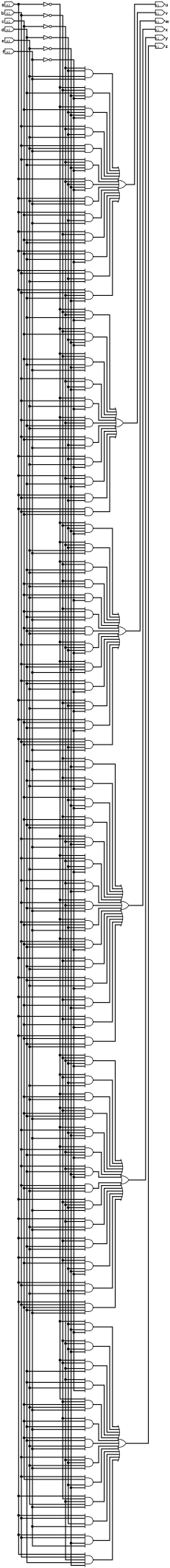




### Expresión de Salida



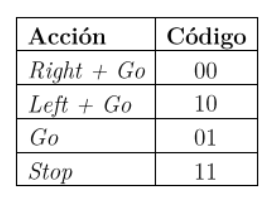
### Circuito



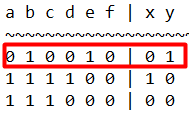
## - Circuit\_Action

### Análisis

* Se crean entradas de 6 bits para los 19 números que usará el robot para hacer el recorrido hasta la posición final.
* Las entradas de los 19 números se crean en orden desde 18 a 11 tal y como indica el camino del problema #2, siendo cada entrada el número correspondiente en binario y la salida el código correspondiente a la acción que ejecutará el robot de acuerdo a la posición donde se encuentra. Código de la acción:



Ejemplo de la primera acción a ejecutar por el robot (Go):

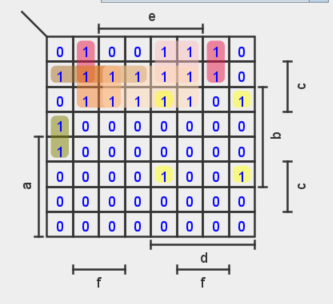


Desde el 18 ejecuta la acción con código 01 para ir al 60.

### Tabla de Verdad

|  |
| --- |
| a b c d e f | x y  ~~~~~~~~~~~~~  0 1 0 0 1 0 | 0 1  1 1 1 1 0 0 | 1 0  1 1 1 0 0 0 | 0 0  0 1 0 1 0 1 | 0 1  0 0 1 1 0 0 | 0 0  1 0 0 0 0 0 | 0 1  1 0 0 0 0 1 | 0 1  0 0 0 0 1 1 | 0 0  1 0 1 0 1 0 | 0 0  0 1 1 1 0 0 | 1 0  0 0 0 1 0 0 | 0 1  1 1 0 0 0 1 | 0 1  0 1 0 0 0 1 | 0 0  0 1 0 1 1 0 | 0 1  0 1 1 0 1 1 | 1 0  1 1 0 1 1 0 | 0 1  0 0 1 1 0 1 | 1 0  0 1 1 1 1 1 | 1 0  1 0 1 0 1 1 | 0 0  1 0 1 0 0 0 | 0 0  0 0 0 1 1 0 | 1 0  0 1 1 1 0 1 | 0 1  0 0 1 0 0 0 | 1 0  1 1 1 1 0 1 | 0 1  0 1 0 0 1 1 | 0 0  1 0 1 0 0 1 | 0 0  1 0 0 1 0 0 | 0 1  0 1 0 0 0 0 | 1 0  0 1 1 0 0 1 | 1 0  0 1 0 1 1 1 | 0 1  1 1 0 1 0 0 | 0 1  1 1 0 1 1 1 | 0 0  1 0 0 0 1 0 | 0 1  0 1 1 0 0 0 | 0 0  1 0 0 1 0 1 | 0 1  0 1 0 1 0 0 | 0 0  1 0 0 1 1 0 | 0 0  0 0 0 1 1 1 | 1 0  0 0 1 0 0 1 | 1 0  0 0 0 0 0 0 | 0 1  0 0 1 0 1 1 | 1 1 |

### Expresión Minimizada

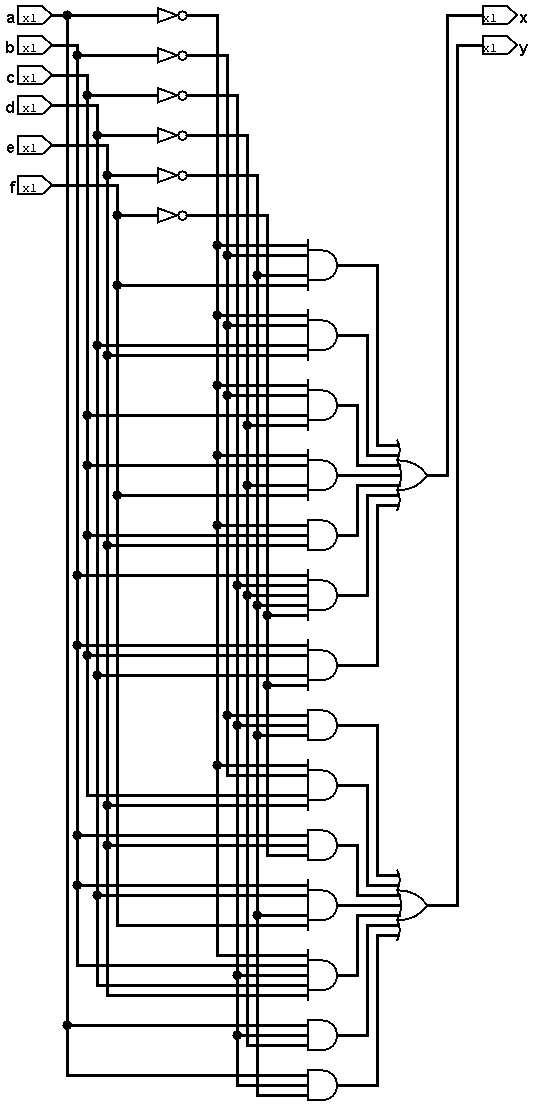




### Expresión de Salida



### Circuito



## - Action\_Decoder

### Análisis

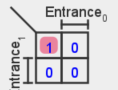
* Se crean entradas de 2 bits para los 4 códigos que usará el robot al ejecutar una acción.
* Las entradas son el código de la acción que ejecutará el robot y la salida es de 1 bit correspondiente al cable que llevará la energía al LED de acuerdo al código de la acción

### Tabla de Verdad

Entrada r: right\_go, Entrada l: left\_go, Entrada g: go, Entrada s: stop

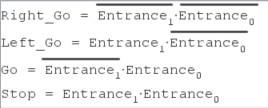
|  |
| --- |
| a b | r l g s  ~~~~~~~~~~  0 0 | 1 0 0 0  1 0 | 0 1 0 0  0 1 | 0 0 1 0  1 1 | 0 0 0 1 |

### Expresión Minimizada

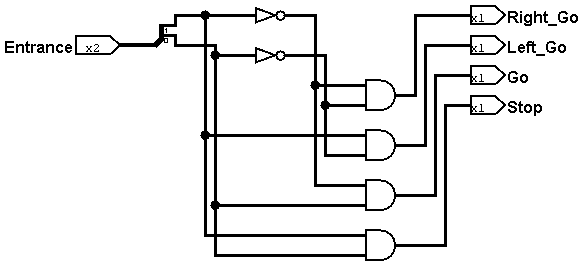




### Expresión de Salida



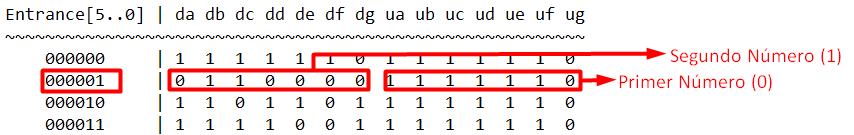
### Circuito



## - Circuit\_two\_7\_Segment

### Análisis

* Se crean entradas de 6 bits para los 64 números posibles a usar por el robot.
* Las entradas de los 64 números se crean en orden de 0 a 63, siendo cada entrada el número correspondiente en binario y las salidas son los bits correspondiente a cada segmento de los dos display.
* Un ejemplo para el número 1, habrán 2 display y los 7 primeros bits corresponden al segundo número y los 7 bits restantes al primer número, para mostrar en ambos display el 01



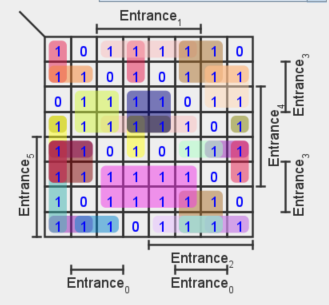
### Tabla de Verdad

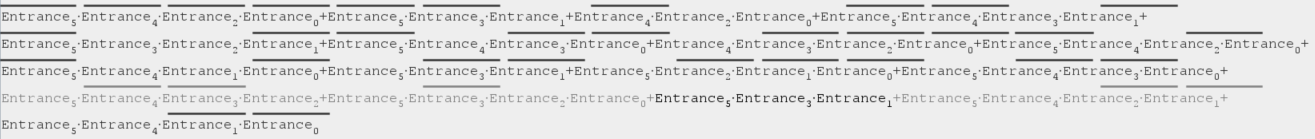
Las salidas (da,db,dc,dd,de,df,dg) corresponden a los bits del segundo 7 segmentos

Las salidas (ua,ub,uc,ud,ue,uf,ug) corresponden a los bits del primer 7 segmentos

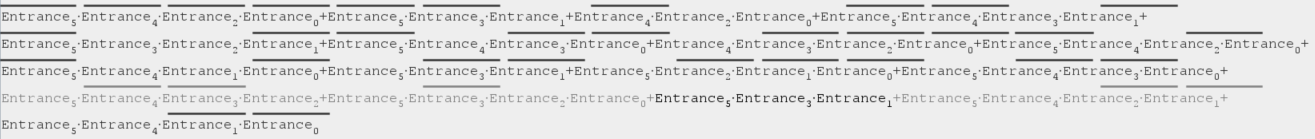
|  |
| --- |
| Entrance[5..0] | da db dc dd de df dg ua ub uc ud ue uf ug  ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~  000000 | 1 1 1 1 1 1 0 1 1 1 1 1 1 0  000001 | 0 1 1 0 0 0 0 1 1 1 1 1 1 0  000010 | 1 1 0 1 1 0 1 1 1 1 1 1 1 0  000011 | 1 1 1 1 0 0 1 1 1 1 1 1 1 0  000100 | 0 1 1 0 0 1 1 1 1 1 1 1 1 0  000101 | 1 0 1 1 0 1 1 1 1 1 1 1 1 0  000110 | 1 0 1 1 1 1 1 1 1 1 1 1 1 0  000111 | 1 1 1 0 0 0 0 1 1 1 1 1 1 0  001000 | 1 1 1 1 1 1 1 1 1 1 1 1 1 0  001001 | 1 1 1 0 0 1 1 1 1 1 1 1 1 0  001010 | 1 1 1 1 1 1 0 0 1 1 0 0 0 0  001011 | 0 1 1 0 0 0 0 0 1 1 0 0 0 0  001100 | 1 1 0 1 1 0 1 0 1 1 0 0 0 0  001101 | 1 1 1 1 0 0 1 0 1 1 0 0 0 0  001110 | 0 1 1 0 0 1 1 0 1 1 0 0 0 0  001111 | 1 0 1 1 0 1 1 0 1 1 0 0 0 0  010000 | 1 0 1 1 1 1 1 0 1 1 0 0 0 0  010001 | 1 1 1 0 0 0 0 0 1 1 0 0 0 0  010010 | 1 1 1 1 1 1 1 0 1 1 0 0 0 0  010011 | 1 1 1 0 0 1 1 0 1 1 0 0 0 0  010100 | 1 1 1 1 1 1 0 1 1 0 1 1 0 1  010101 | 0 1 1 0 0 0 0 1 1 0 1 1 0 1  010110 | 1 1 0 1 1 0 1 1 1 0 1 1 0 1  010111 | 1 1 1 1 0 0 1 1 1 0 1 1 0 1  011000 | 0 1 1 0 0 1 1 1 1 0 1 1 0 1  011001 | 1 0 1 1 0 1 1 1 1 0 1 1 0 1  011010 | 1 0 1 1 1 1 1 1 1 0 1 1 0 1  011011 | 1 1 1 0 0 0 0 1 1 0 1 1 0 1  011100 | 1 1 1 1 1 1 1 1 1 0 1 1 0 1  011101 | 1 1 1 0 0 1 1 1 1 0 1 1 0 1  011110 | 1 1 1 1 1 1 0 1 1 1 1 0 0 1  011111 | 0 1 1 0 0 0 0 1 1 1 1 0 0 1  100000 | 1 1 0 1 1 0 1 1 1 1 1 0 0 1  100001 | 1 1 1 1 0 0 1 1 1 1 1 0 0 1  100010 | 0 1 1 0 0 1 1 1 1 1 1 0 0 1  100011 | 1 0 1 1 0 1 1 1 1 1 1 0 0 1  100100 | 1 0 1 1 1 1 1 1 1 1 1 0 0 1  100101 | 1 1 1 0 0 0 0 1 1 1 1 0 0 1  100110 | 1 1 1 1 1 1 1 1 1 1 1 0 0 1  100111 | 1 1 1 0 0 1 1 1 1 1 1 0 0 1  101000 | 1 1 1 1 1 1 0 0 1 1 0 0 1 1  101001 | 0 1 1 0 0 0 0 0 1 1 0 0 1 1  101010 | 1 1 1 0 1 1 1 0 1 1 0 0 1 1  101011 | 1 1 1 1 0 0 1 0 1 1 0 0 1 1  101100 | 0 1 1 0 0 1 1 0 1 1 0 0 1 1  101101 | 1 0 1 1 0 1 1 0 1 1 0 0 1 1  101110 | 1 0 1 1 1 1 1 0 1 1 0 0 1 1  101111 | 1 1 1 0 0 0 0 0 1 1 0 0 1 1  110000 | 1 1 1 1 1 1 1 0 1 1 0 0 1 1  110001 | 1 1 1 0 0 1 1 0 1 1 0 0 1 1  110010 | 1 1 1 1 1 1 0 1 0 1 1 0 1 1  110011 | 0 1 1 0 0 0 0 1 0 1 1 0 1 1  110100 | 1 1 0 1 1 0 1 1 0 1 1 0 1 1  110101 | 1 1 1 1 0 0 1 1 0 1 1 0 1 1  110110 | 0 1 1 0 0 1 1 1 0 1 1 0 1 1  110111 | 1 0 1 1 0 1 1 1 0 1 1 0 1 1  111000 | 1 0 1 1 1 1 1 1 0 1 1 0 1 1  111001 | 1 1 1 0 0 0 0 1 0 1 1 0 1 1  111010 | 1 1 1 1 1 1 1 1 0 1 1 0 1 1  111011 | 1 1 1 0 0 1 1 1 0 1 1 0 1 1  111100 | 1 1 1 1 1 1 0 1 0 1 1 1 1 1  111101 | 0 1 1 0 0 0 0 1 0 1 1 1 1 1  111110 | 1 1 0 1 1 0 1 1 0 1 1 1 1 1  111111 | 1 1 1 1 0 0 1 1 0 1 1 1 1 1 |

### Expresión Minimizada

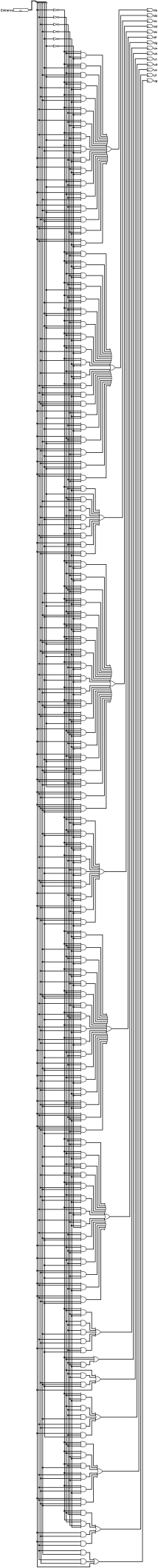




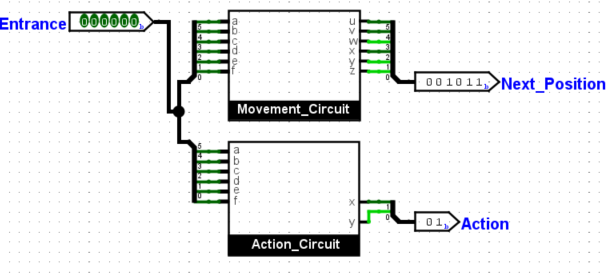
### Expresión de Salida



### Circuito



## Componente Movement\_Control\_Decoder

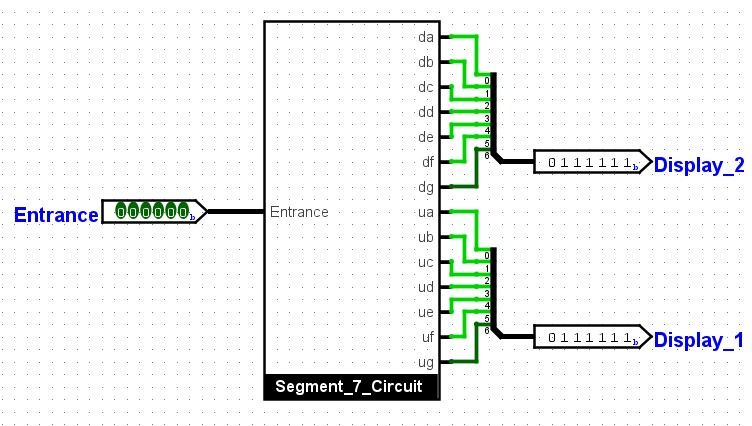


Este componente es creado partiendo de los circuitos **Action\_Circuit** y **Movement\_Circuit** Los cuales ambos tienen como entradas un número en base binaria de 6 bits.

**Action\_Circuit** El tiene una salida de 2 bits (como describo anteriormente) y este conecta directamente al componente Action\_Decoder.

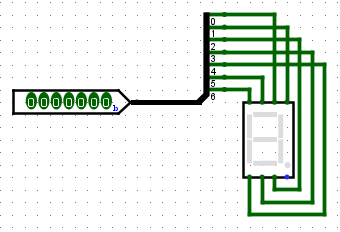
**Movement\_Circuit** El tiene una salida de 6 bits (como describo anteriormente) y este conecta directamente al componente Binary\_to\_two\_7\_Segment\_Decoder.

## Componente Binary\_to\_two\_7\_Segment\_Decoder



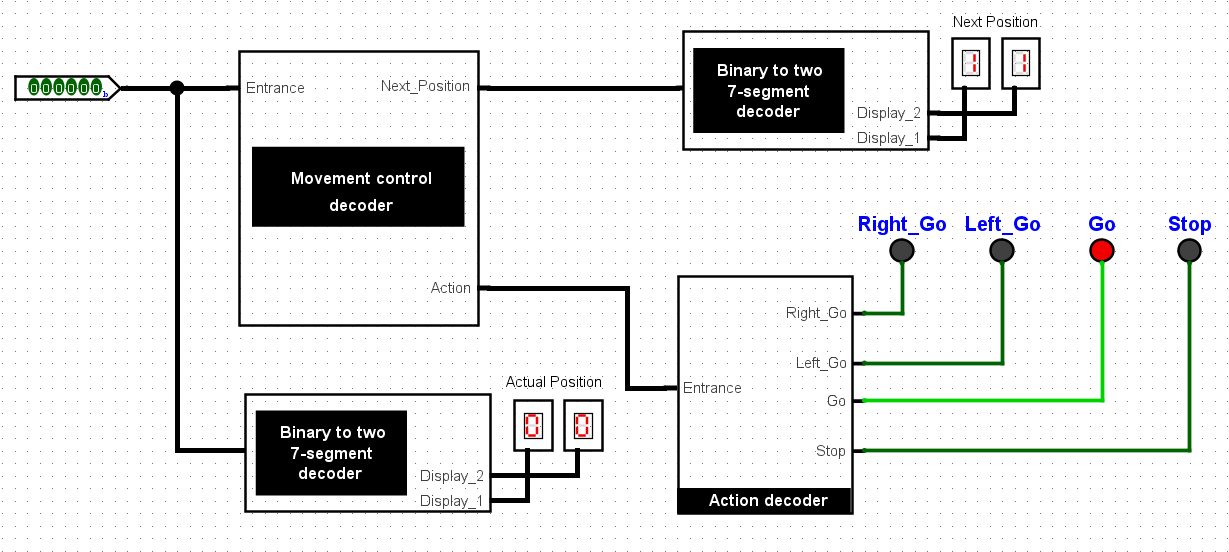
Este componente es creado partiendo del circuito **Circuit\_two\_7\_Segment** El cual tiene como entrada un número en base binaria de 6 bits y como salida 14 bits, de los cuales los 7 primeros corresponden al número binario expresado en decimal del número que va en el display 2 y los 7 bits restantes corresponden al número binario expresado en decimal del número que va en el display 1.

## Componente Display\_Decoder



Este componente es creado con el fin de incrustar el 7 segmentos de forma ordenada con un separador de bits de 7 y una entrada de 7bits para que posteriormente se conecte al decoder del número a mostrar.

## Componente Main



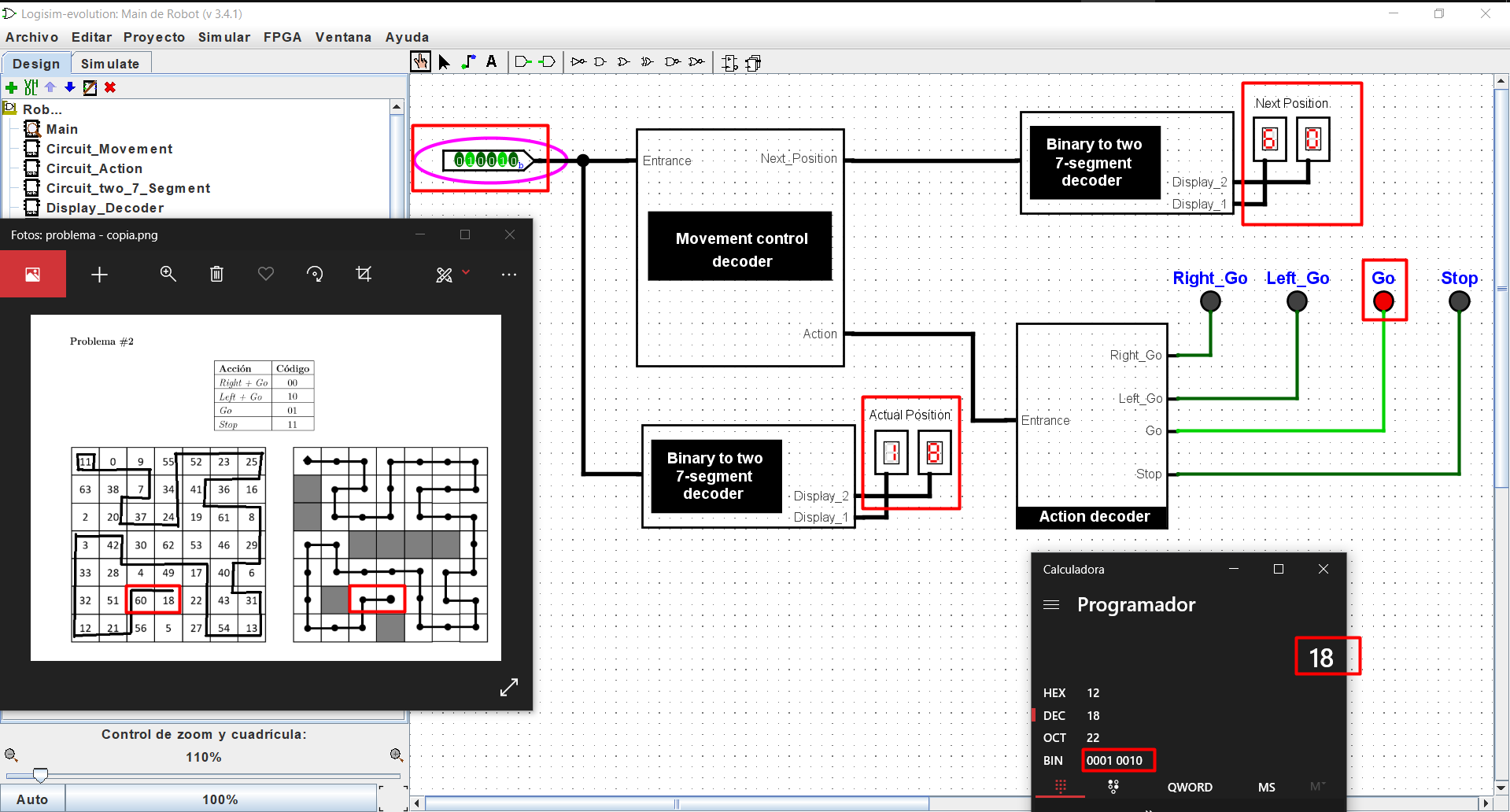
En este componente integro instanciando los demás componentes con el fin de que todos trabajen mancomunadamente para satisfacer las necesidades y problematicas propuestas, cumpliendo con la arquitectura exigida.

Cada vez que se digite un número entre 0 y 63 en base binaria, ese número viaja a binary to tow 7-segment decoder para mostrar él número ingresado en base decimal en el display de 7 segmentos, ese mismo número digitado viaja a Movement control decoder, dónde se obtendrá la acción a realizar y el valor de la siguiente posición. Luego en el Action decoder se toma el código de la acción a realizar y se decodifica en un bit que se encarga de encender el LED correspondiente a dicha acción.

# Simulación

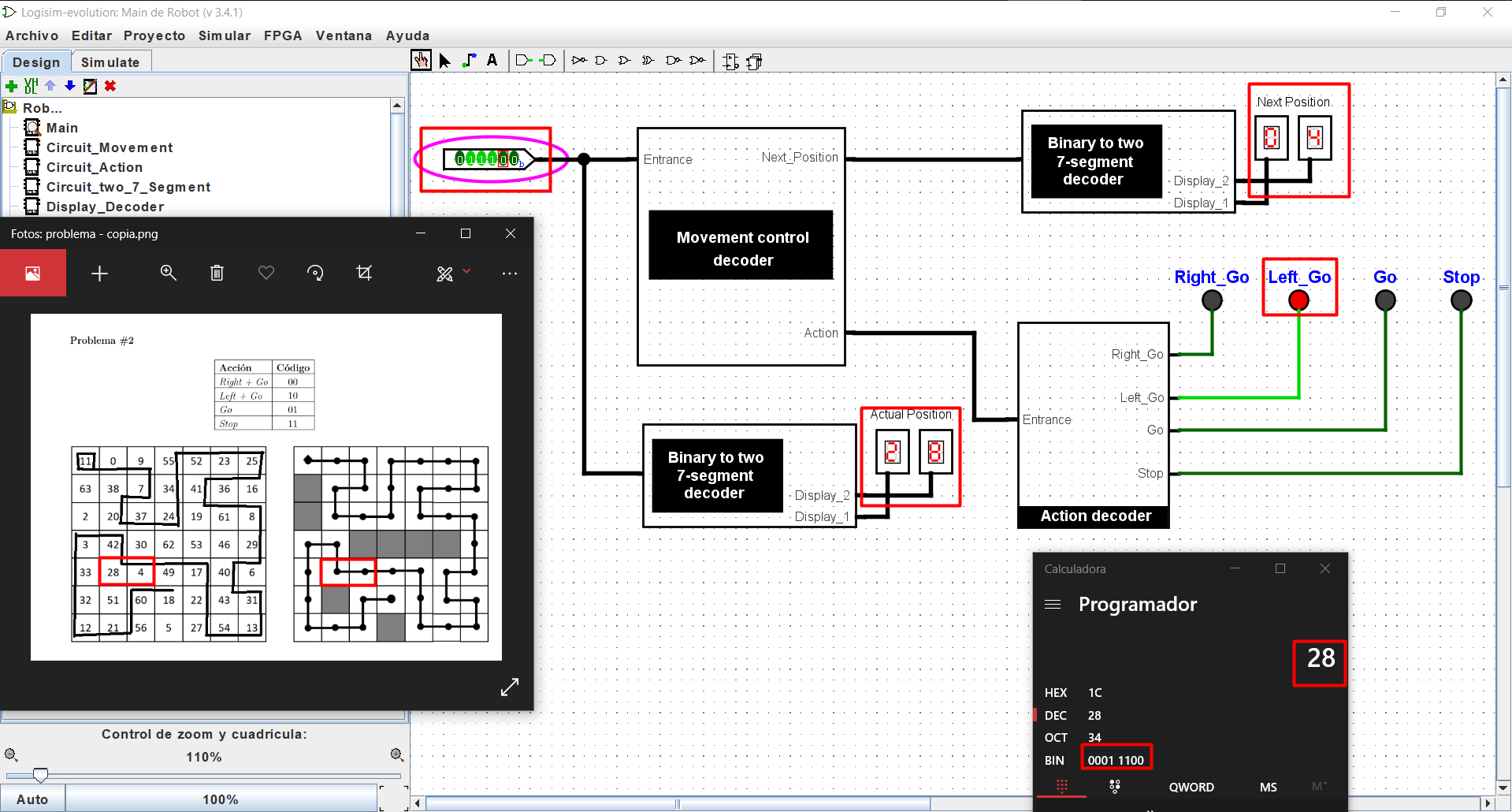
La simulación la haremos con base a distintos casos. Caso real, caso probable, caso extremo.

### Caso real 1:



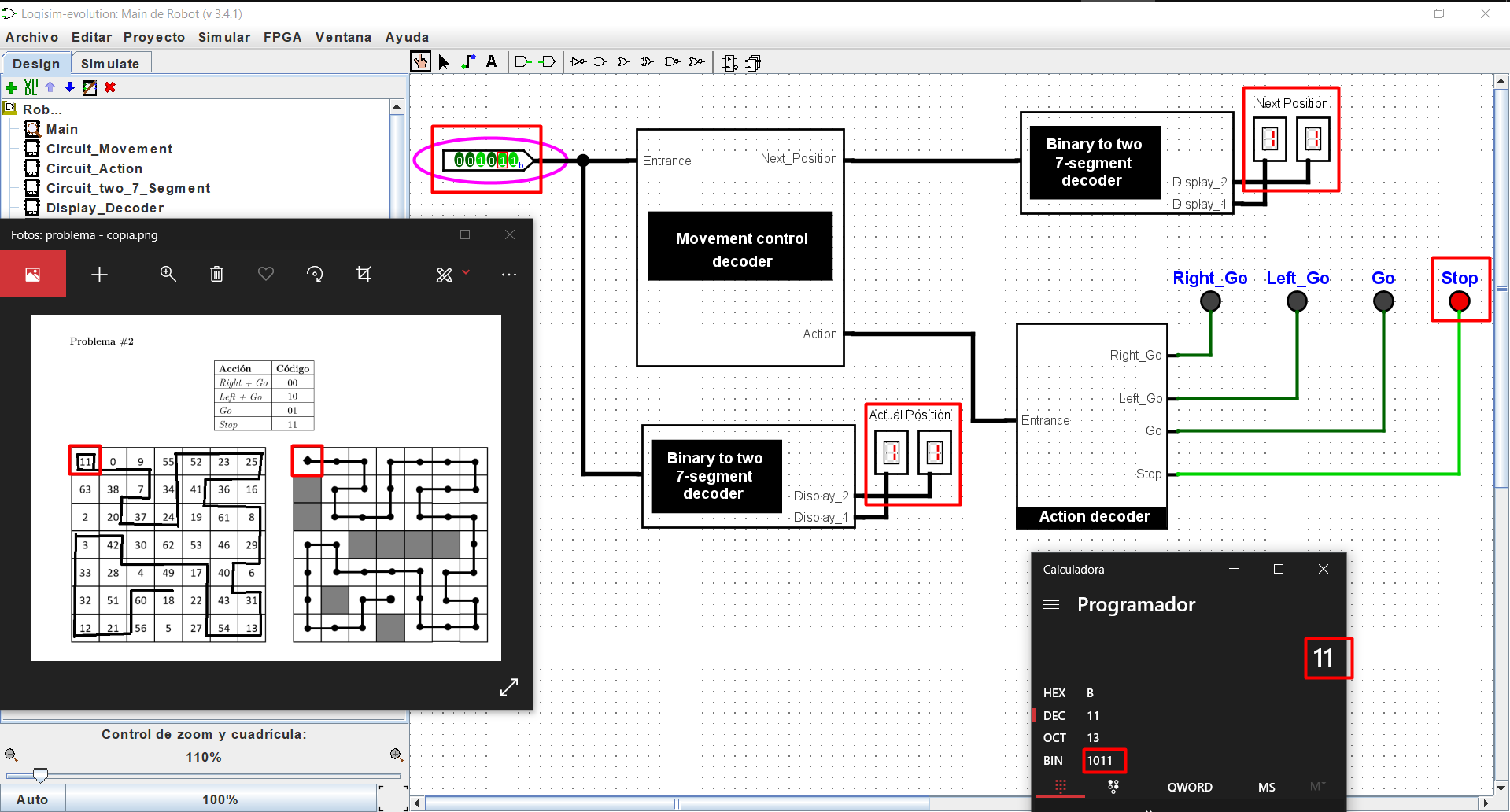
Visualizamos el número 18 en binario como: 010010 en la entrada y como posición actual, el LED indica que la acción es Go y que la siguiente posición es 60; de acuerdo al diagrama, es correcto.

### Caso Real 2:



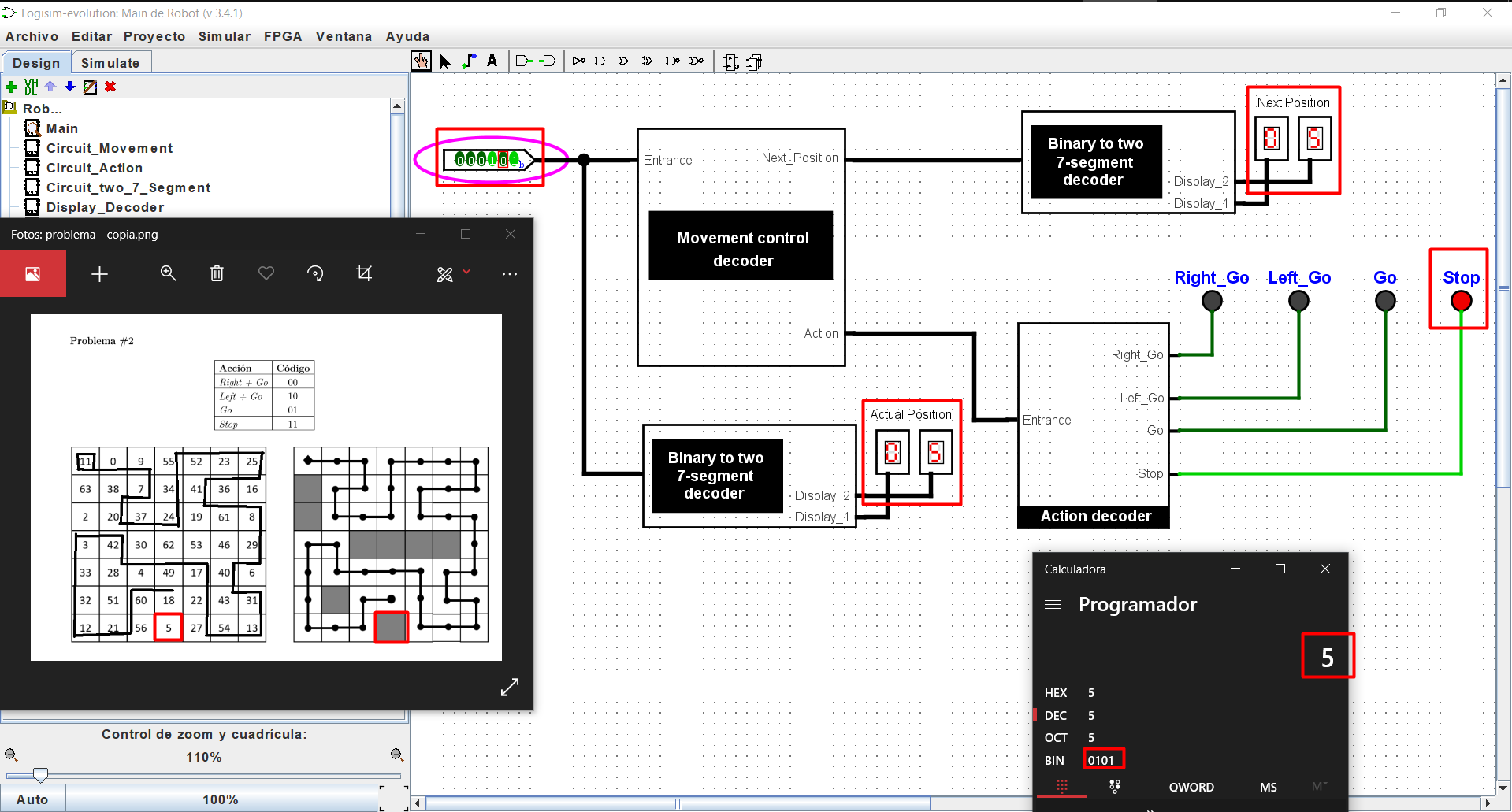
En este caso vemos que la posición actual es 28 denotada en binario como: 011100 de entrada, la acción a ejercer sería ir Left\_Go para ir a la posición siguiente que es 04, la cual se compara con el diagrama y muestra que es correcto.

### Caso Probable 1:



Este caso probable corresponde a que digiten como posición actual el 11. si vemos es el valor donde se quiere llegar y por consiguiente la acción a tomar es Stop y la siguiente posición es el mismo número 11 porque no puede avanzar.

### Caso Extremo 1:



Este caso sucede cuando se pone una posición actual que no exista en la tabla o que este sombreada, mostrará el valor de entrada que se dio y la acción será Stop, adicional la siguiente posición es el mismo valor de entrada porque no puede avanzar.

# Análisis de Resultados

Para un mejor entendimiento plantee el análisis de resultados en forma de video para que vean la interacción de una forma dinámica y orientada a un caso de uso real.

Clic aquí para ver el vídeo:

# Observaciones

# Conclusión