Diseño Combinacional

Nunca desistas de un sueño.
Sólo trata de ver las señales que te lleven a él.
Paulo Coelho

Diseño de Sistemas Combinacionales

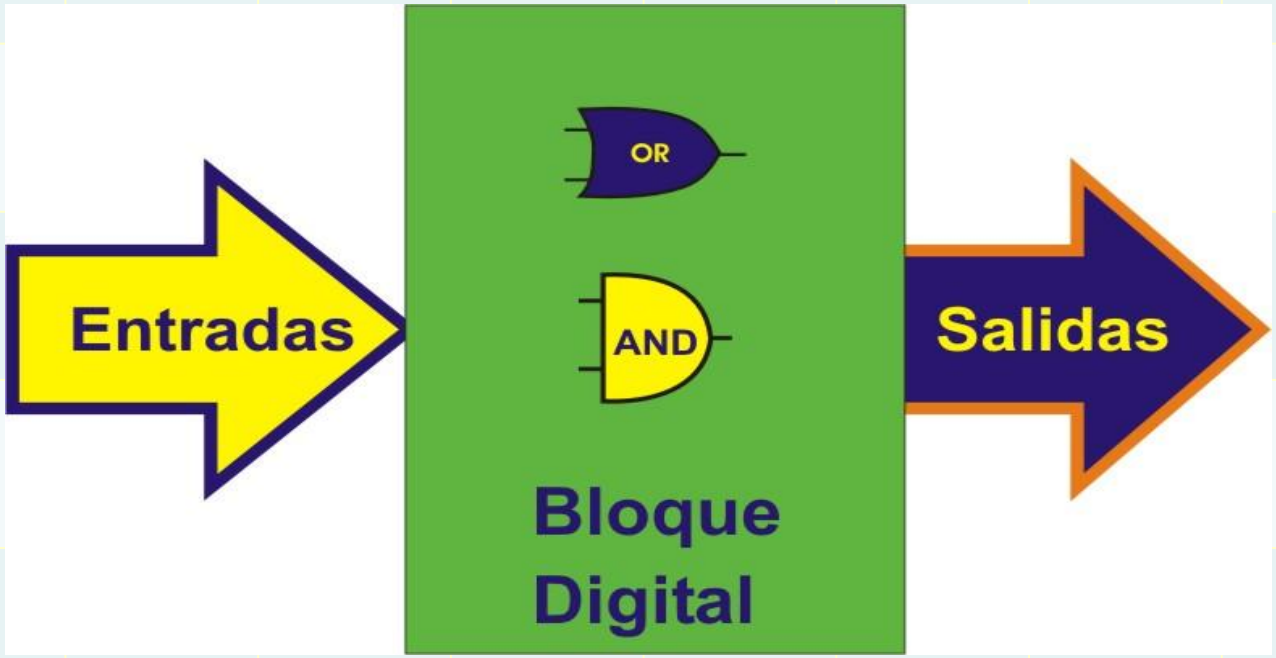
Diseño en Ingeniería:

Es la creación y desarrollo de un producto, proceso o sistema económicamente viable para satisfacer necesidades definidas por un cliente o proceso.

Andrew McLaren, Approaches to the Teaching of Design, Engineering Subject Centre, The Higher Education Academy, University of Sheffiled UK, 2008, ISBN 978-1-904804-802

Sistema Combinacional

Es aquel bloque digital en donde los valores de salida dependen únicamente de las combinaciones de entrada.

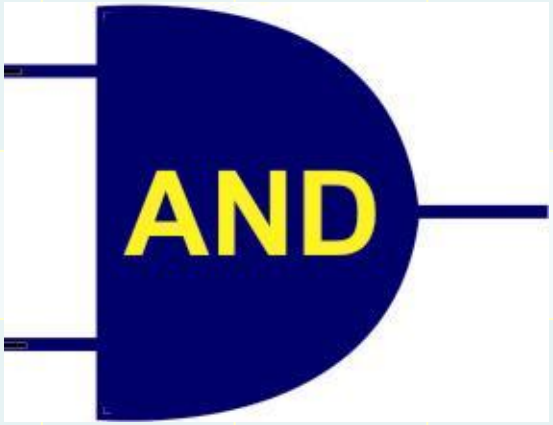


Un sistema combinacional puede estar compuesto de una sola operación.

¿De que depende que la salida de la operación AND valga uno?

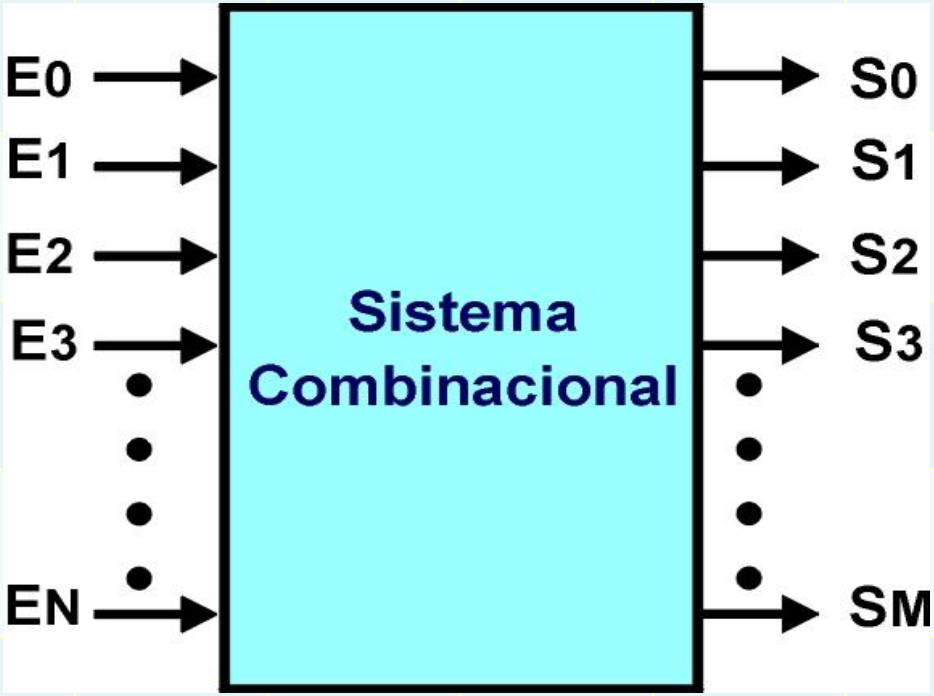
De que sus entradas tengan el valor de uno

La salida solo depende de las combinaciones de entrada.

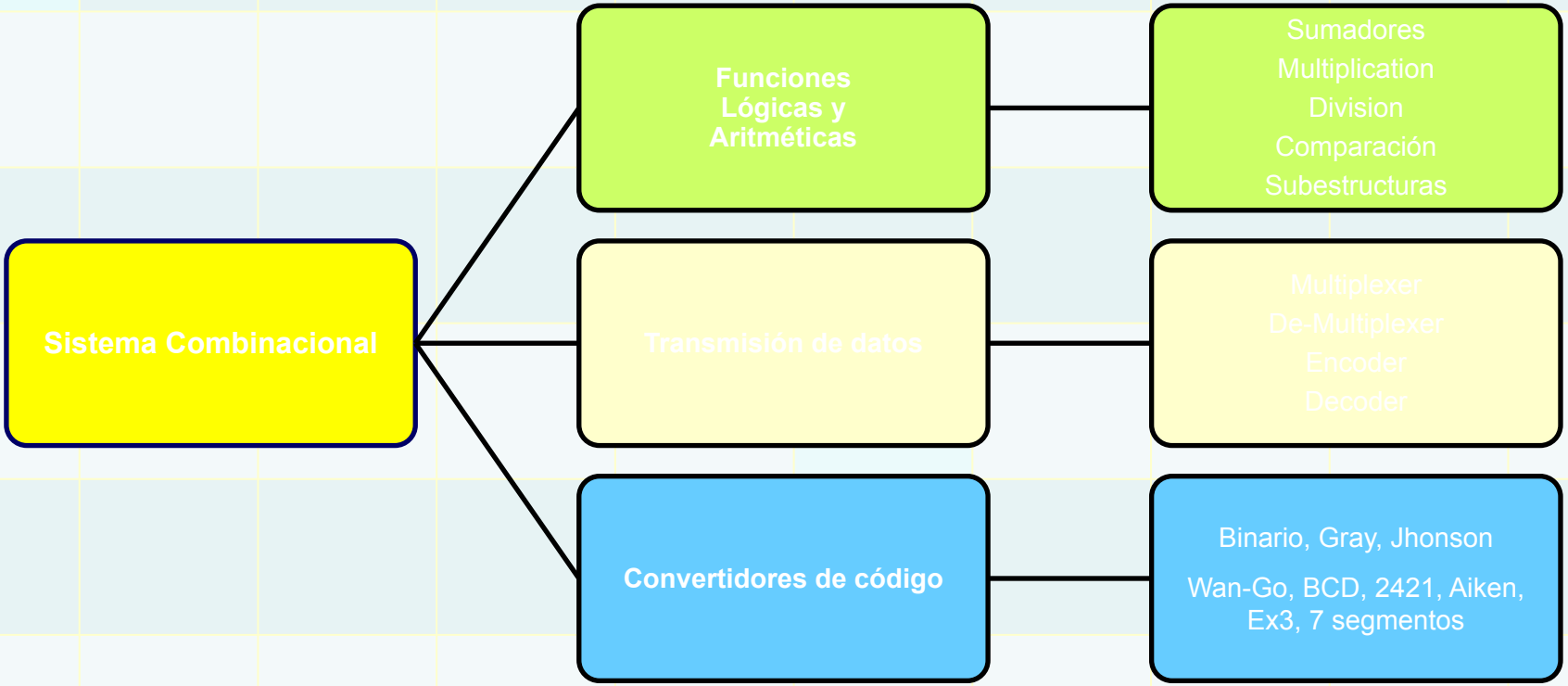


?

Un sistema combinacional puede tener una o más entradas y/o una o mas salidas y el número de entradas puede ser mayor, menor o igual al número de salidas.



Aplicaciones de los sistemas Combinacionales



Método del Diseño Combinacional

1.- Especificar el Sistema

2.- Determinar entradas y salidas

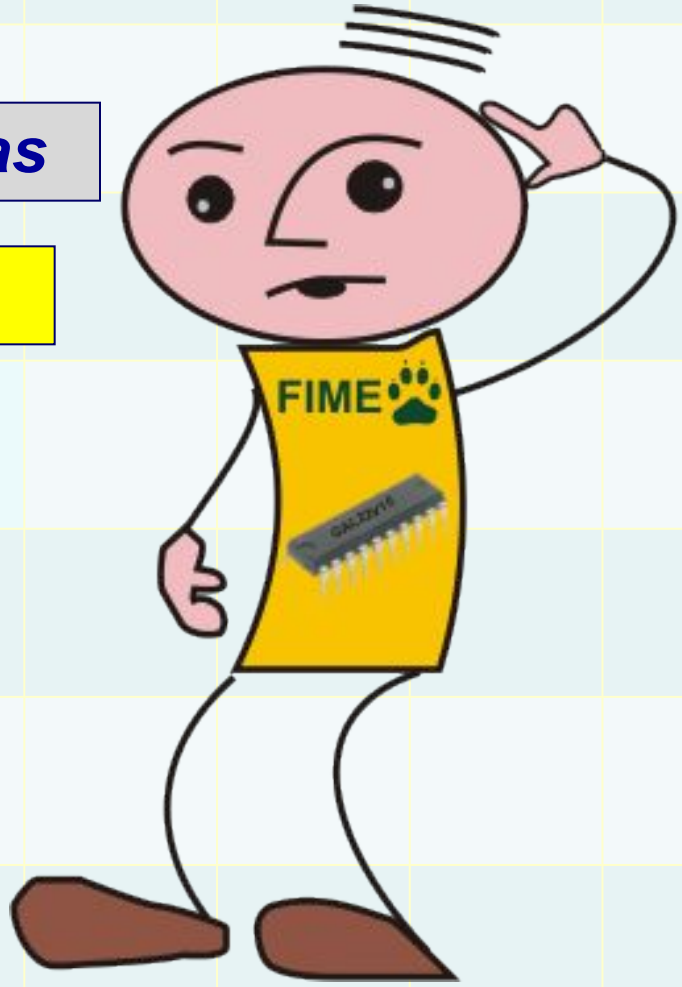
3.- Construir la Tabla de Verdad

4.- Ecuaciones Mínimas

5.- Diagrama Esquemático

6.- Simulación

7.- Construir un Prototipo

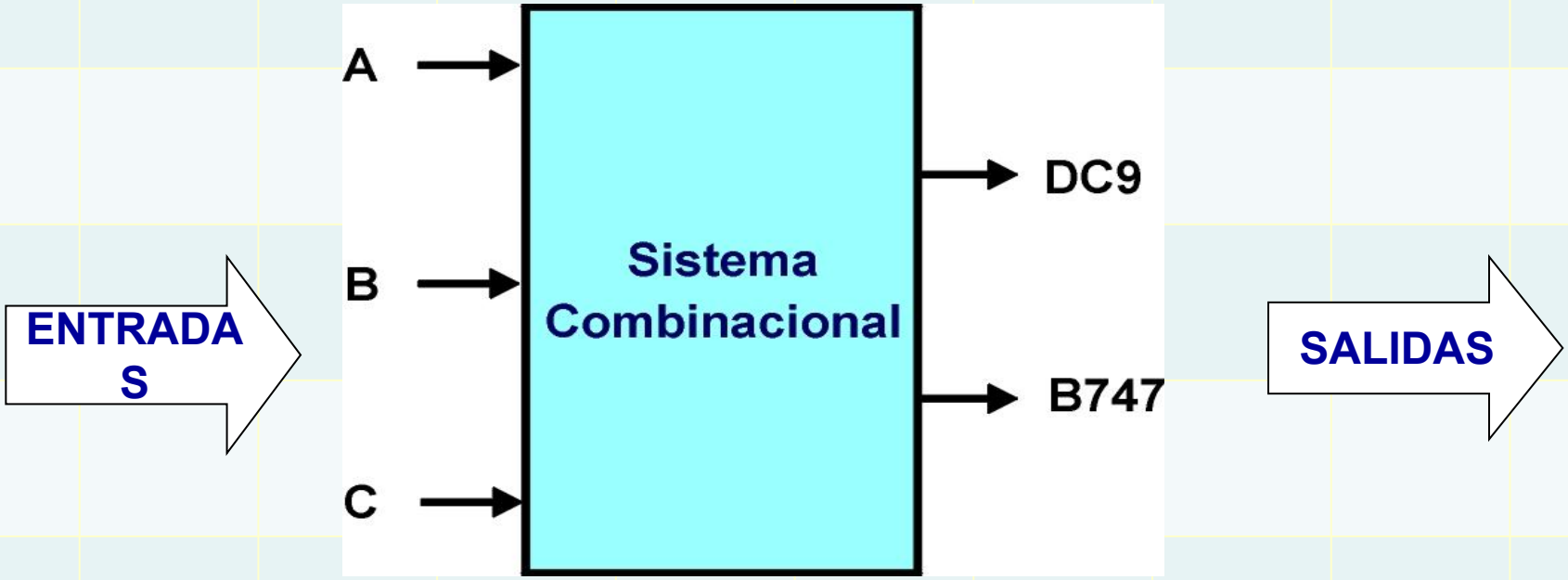


1.- Especificar el Sistema

En esta parte se detalla el propósito del diseño

2.- Determinar entradas y salidas

De las variables que intervienen en el problema hay que identificar cuales y cuantas son de entrada y de salida.



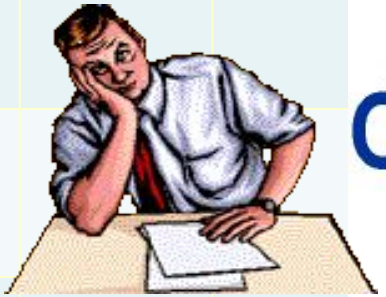
3.- Construir la Tabla de Verdad

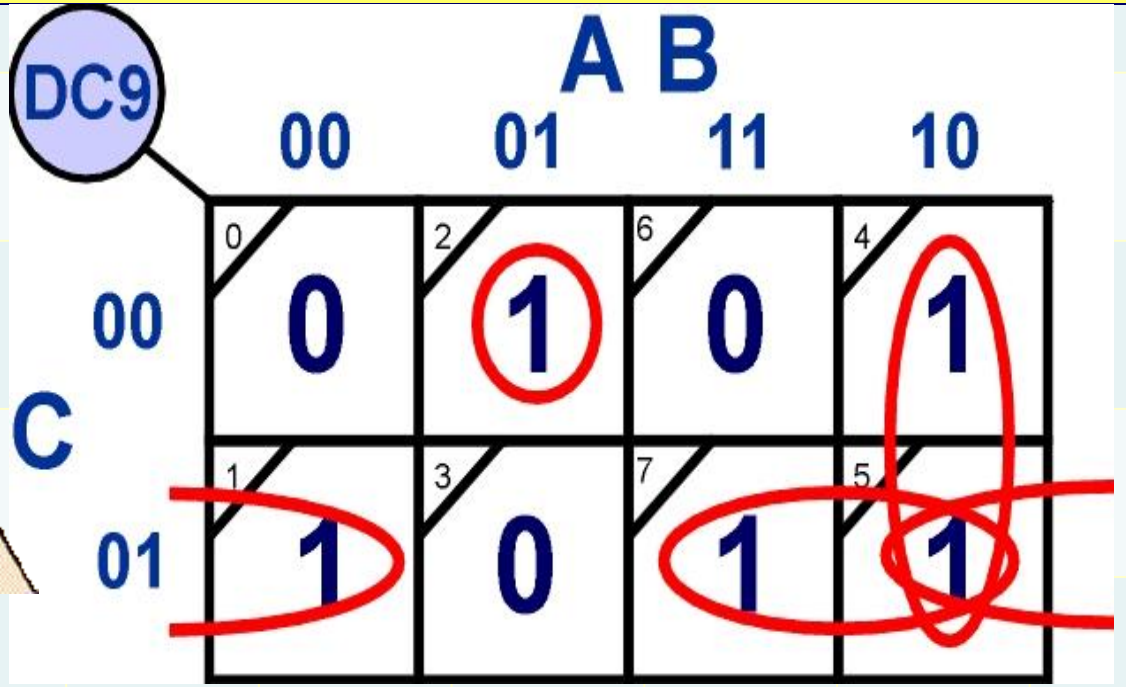
Trasladar el Comportamiento del sistema a una tabla de verdad, indicando para cada combinación de entrada la salida o salidas mas convenientes para el diseño

m	A B C	F3	F4
0	0 0 0		
1	0 0 1		
2	0 1 0		
3	0 1 1		
4	1 0 0		
5	1 0 1		
6	1 1 0		
7	1 1 1		

4.- Ecuaciones Mínimas

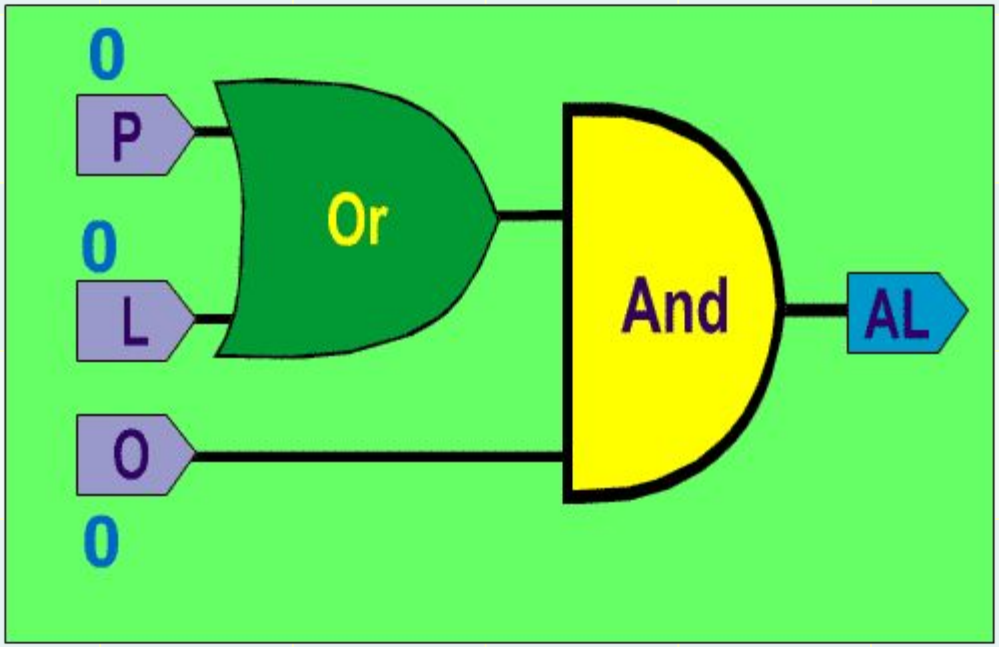
Para obtener las ecuaciones mínimas se puede utilizar algún método de simplificación como LogicAid, manipulación algebraica, mapas de Karnaugh, etc





5.- Diagrama Esquemático

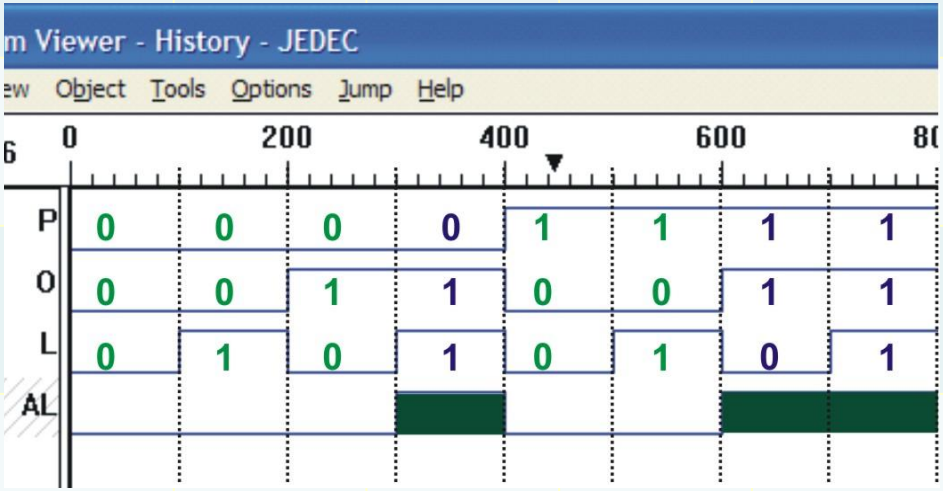
Después de haber obtenido las ecuaciones mínimas se representa en forma de símbolos para su análisis y comprensión.



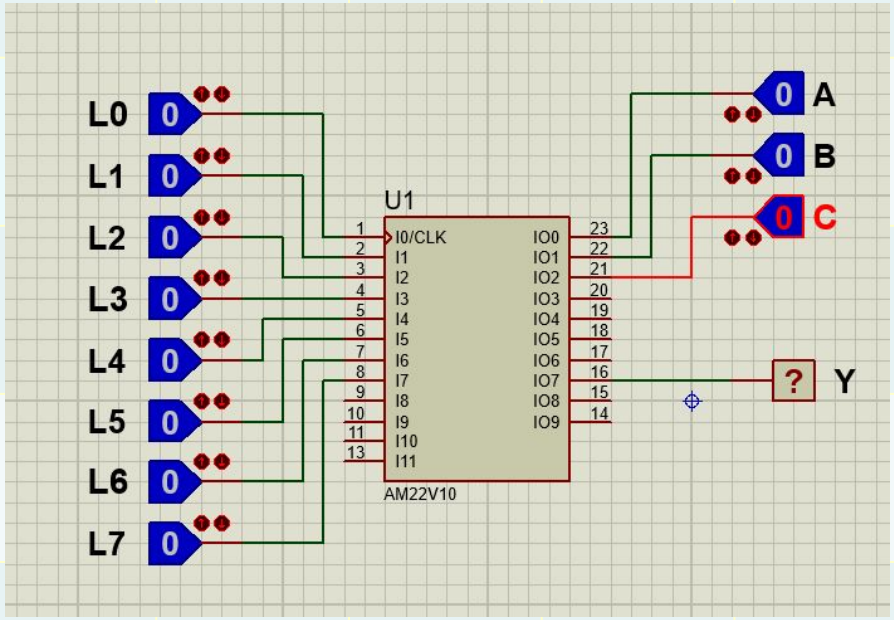
6.- Simulación

En el software utilizado para el diseño ya se captura esquemática o HDL nos permite hacer una simulación y comprobar su funcionamiento antes de implementarlo.

Test Vectors



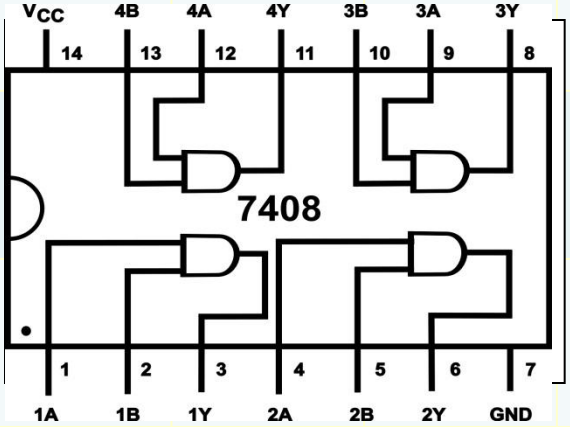
Proteus



7.- Construir un Prototipo

Se tienen dos opciones para la implementación

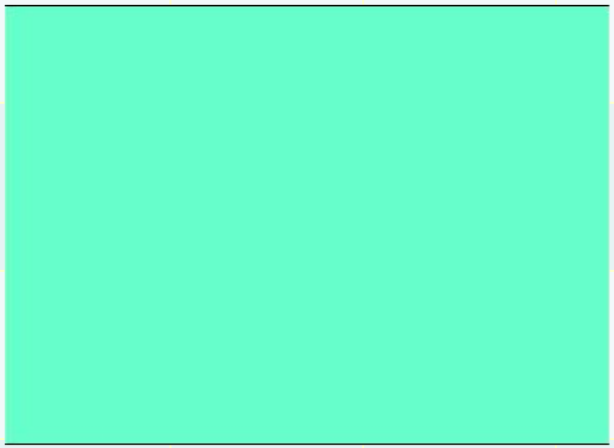
- Circuitos Integrados de función fija (TTL o CMOS)
- Dispositivos Lógicos Programables (PLD's)



Diseño con PLD

En los Dispositivos Lógicos Programables (PLD's) se puede diseñar mediante :

- Captura esquemática
- Un Lenguaje de Descripción de Hardware (HDL)



Text Editor - [comp2n1b.abl]

File Edit View Templates Tools Options Window Help

```
MODULE comp
X=.x.;" igualdad
"Entradas
A,B pin 1,2;
"En Salidas los signos de = > <
"estan reservados para los
"operadores relacionales usaremos
" AeqB en lugar de A=B
" AmyB en lugar de A>B
" AmeB en lugar de A<B
AeqB,AmyB,AmeB pin 19..17 istype 'com';
equations
AeqB= !(A$B);
AmyB= A&!B;
AmeB= !A&B;
test_vectors
([A,B]->[AeqB, AmyB, AmeB])
[0,0]->[X,X,X];
[0,1]->[X,X,X];
[1,0]->[X,X,X];
[1,1]->[X,X,X];
END
```

Diseño con PLD

Lenguaje de Descripción de Hardware (HDL)

a) Las Ecuaciones

b) La Tabla De Verdad

c) La Descripción del Problema.

$FAL_{(P, O, L)} = O(P + L)$

m	P	O	L	AL
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	0
6	1	1	0	1
7	1	1	1	1

```
WHEN !A&!B&!C THEN Y=L0;
WHEN !A&!B&C THEN Y=L1;
WHEN !A&B&!C THEN Y=L2;
```

Método del Diseño Combinacional

1.- Especificar el Sistema

2.- Determinar entradas y salidas

3.- Construir la Tabla de Verdad

4.- Ecuaciones Mínimas

HDL

~~5.- Diagrama Esquemático~~

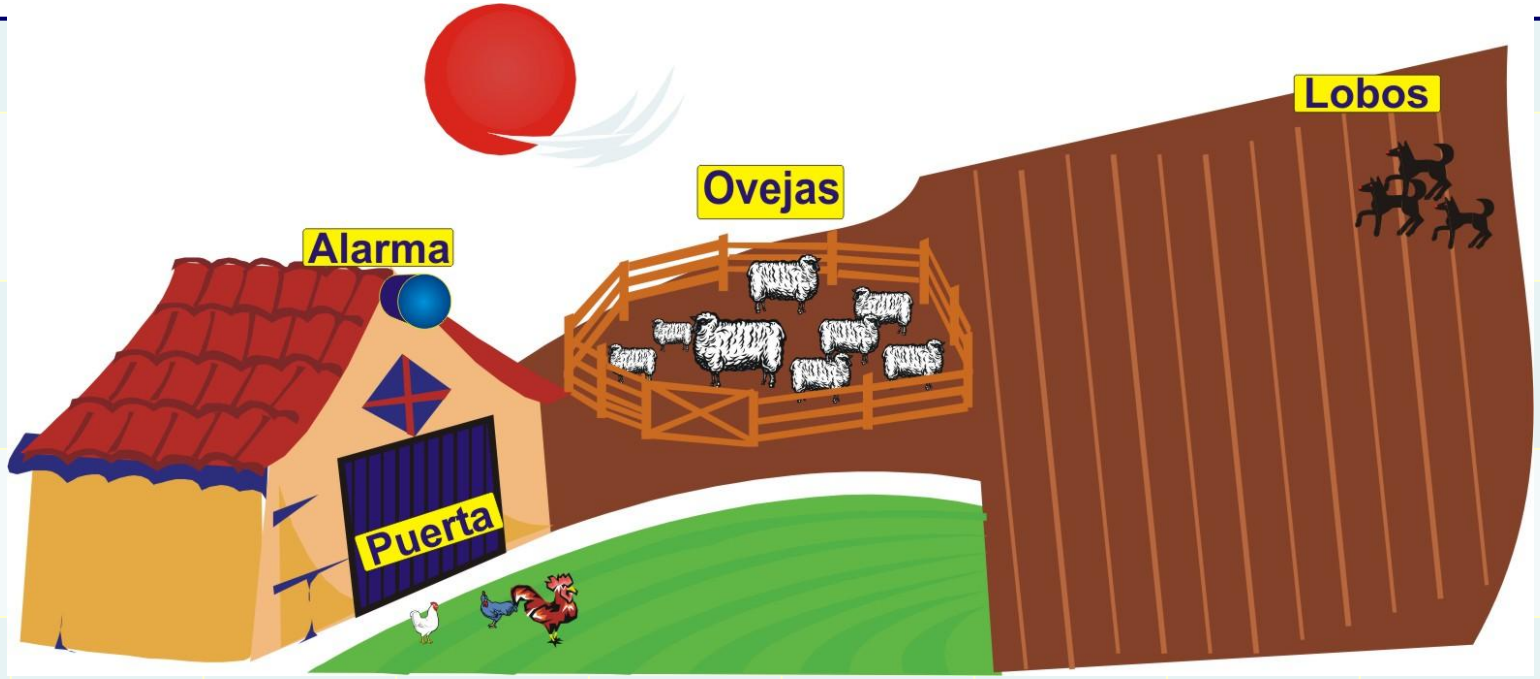
6.- Simulación

7.- Construir un Prototipo

Ejemplo 1

En una granja se tiene:

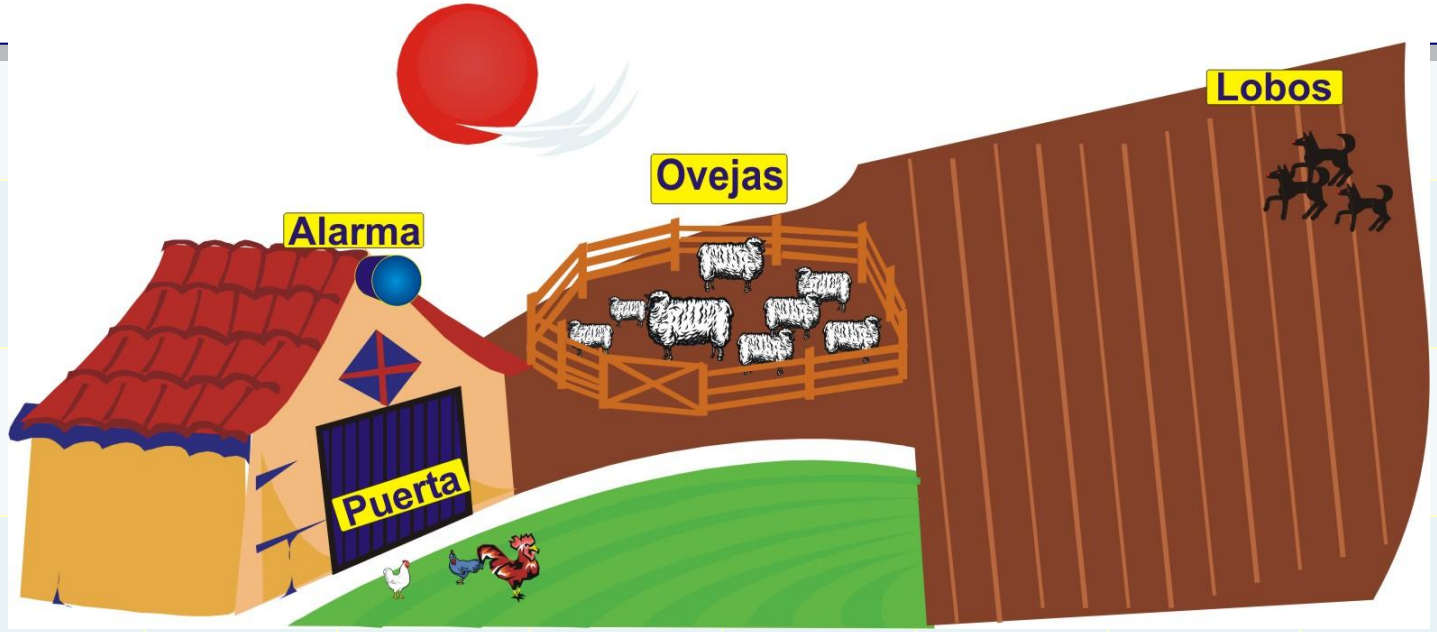
- Un granero con una puerta muy grande y pesada en donde se requiere de varias personas para abrirla o cerrarla
- Un corral de ovejas
- Además ocasionalmente llegan lobos



El granjero necesita el diseño de un sistema de alarma de modo que:

1.- Se active cuando las ovejas estén fuera del corral y la puerta abierta, para hacer una acción correctiva ya sea cerrar la puerta del granero o poner las ovejas en su corral.

2.-También deberá de activarse la alarma cuando estén los lobos próximos y las ovejas fuera del corral, para hacer la acción correctiva de ahuyentar a los lobos.



1.-Especificar el Sistema

Las variables que intervienen son Puerta, Ovejas, Lobos y la Alarma para las primeras tres se tienen sensores de detección de modo que:

Puerta

Si esta abierta = *1*, Si esta cerrada =*0*

Ovejas

Si están fuera del corral =*1*, Si están dentro del corral =*0*

Lobos

Si están próximos = *1*, Si están lejos =*0*

Para el dispositivo de alarma se considera que:

Alarma

Se activa con un *1*,Se desactiva con un *0*

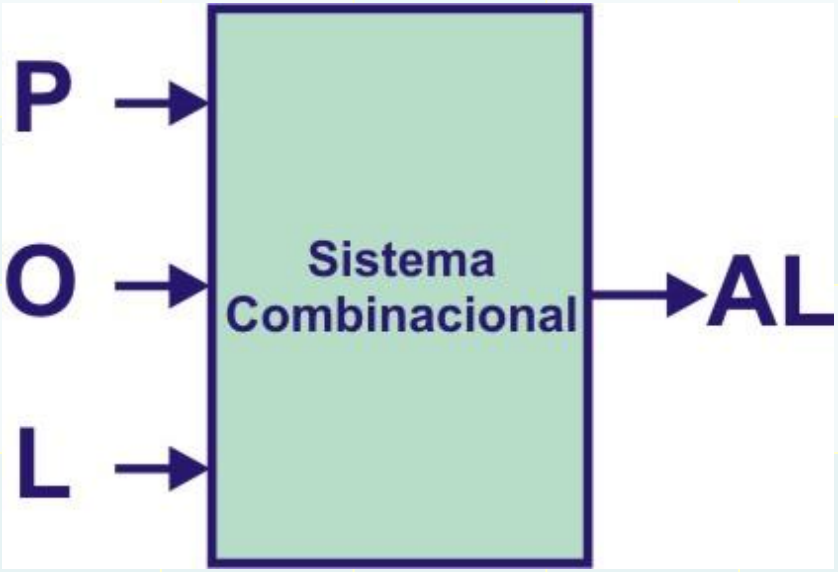
2.- Determinar entradas y salidas.

Podemos decir que:

La puerta, Ovejas y Lobos (*P, O y L)* son las entradas del sistema.

Mientras que la alarma (*AL*) es la salida.

Representada a continuación en un diagrama de bloques.



3.- Trasladar el comportamiento a una tabla de verdad.

En este paso hay que decidir el valor de la salida (0 o 1) para cada una de las posibles combinaciones de entrada:

Puerta
Si esta abierta = *1*
Si esta cerrada =*0*

Ovejas
Si están fuera del corral =*1*
Si están dentro =*0*

Lobos
Si están próximos = *1,*
Si están lejos =*0*

Alarma
Se activa con un *1,*
Se desactiva con un *0*

m	P	O	L	AL
0	0	0	0	
1	0	0	1	
2	0	1	0	
3	0	1	1	
4	1	0	0	
5	1	0	1	
6	1	1	0	
7	1	1	1	

3.- Trasladar el comportamiento a una tabla de verdad.

En este paso hay que decidir el valor de la salida (0 o 1) para cada una de las posibles combinaciones de entrada:

Puerta
Si esta abierta = *1*
Si esta cerrada =*0*

Ovejas
Si están fuera del corral =*1*
Si están dentro =*0*

Lobos
Si están próximos = *1,*
Si están lejos =*0*

Alarma
Se activa con un *1,*
Se desactiva con un *0*

m	P	O	L	AL
0	0	0	0	0
1	0	0	1	
2	0	1	0	
3	0	1	1	
4	1	0	0	
5	1	0	1	
6	1	1	0	
7	1	1	1	

las ovejas estén fuera del corral y la puerta abierta
o
los lobos próximos y las ovejas fuera del corral

3.- Trasladar el comportamiento a una tabla de verdad.

En este paso hay que decidir el valor de la salida (0 o 1) para cada una de las posibles combinaciones de entrada:

Puerta
Si esta abierta = *1*
Si esta cerrada =*0*

Ovejas
Si están fuera del corral =*1*
Si están dentro =*0*

Lobos
Si están próximos = *1,*
Si están lejos =*0*

Alarma
Se activa con un *1,*
Se desactiva con un *0*

m	P	O	L	AL
0	0	0	0	0
1	0	0	1	0
2	0	1	0	
3	0	1	1	
4	1	0	0	
5	1	0	1	
6	1	1	0	
7	1	1	1	

las ovejas estén fuera del corral y la puerta abierta o los lobos próximos y las ovejas fuera del corral

3.- Trasladar el comportamiento a una tabla de verdad.

En este paso hay que decidir el valor de la salida (0 o 1) para cada una de las posibles combinaciones de entrada:

m	P	O	L	AL
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	
4	1	0	0	
5	1	0	1	
6	1	1	0	
7	1	1	1	

las ovejas estén fuera del corral y la puerta abierta o los lobos próximos y las ovejas fuera del corral

3.- Trasladar el comportamiento a una tabla de verdad.

En este paso hay que decidir el valor de la salida (0 o 1) para cada una de las posibles combinaciones de entrada:

m	P	O	L	AL
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	
5	1	0	1	
6	1	1	0	
7	1	1	1	

las ovejas estén fuera del corral y la puerta abierta o los lobos próximos y las ovejas fuera del corral

3.- Trasladar el comportamiento a una tabla de verdad.

En este paso hay que decidir el valor de la salida (0 o 1) para cada una de las posibles combinaciones de entrada:

m	P	O	L	AL
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	
6	1	1	0	
7	1	1	1	

las ovejas estén fuera del corral y la puerta abierta o los lobos próximos y las ovejas fuera del corral

3.- Trasladar el comportamiento a una tabla de verdad.

En este paso hay que decidir el valor de la salida (0 o 1) para cada una de las posibles combinaciones de entrada:

m	P	O	L	AL
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	0
6	1	1	0	
7	1	1	1	

las ovejas estén fuera del corral y la puerta abierta o los lobos próximos y las ovejas fuera del corral

3.- Trasladar el comportamiento a una tabla de verdad.

En este paso hay que decidir el valor de la salida (0 o 1) para cada una de las posibles combinaciones de entrada:

m	P	O	L	AL
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	0
6	1	1	0	1
7	1	1	1	

las ovejas estén fuera del corral y la puerta abierta
o
los lobos próximos y las ovejas fuera del corral

3.- Trasladar el comportamiento a una tabla de verdad.

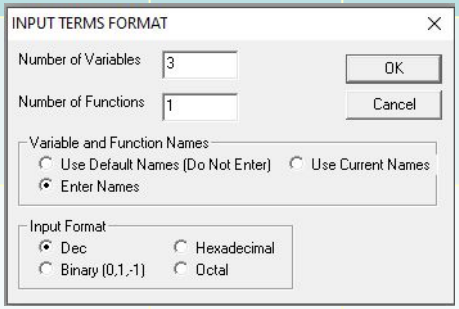
En este paso hay que decidir el valor de la salida (0 o 1) para cada una de las posibles combinaciones de entrada:

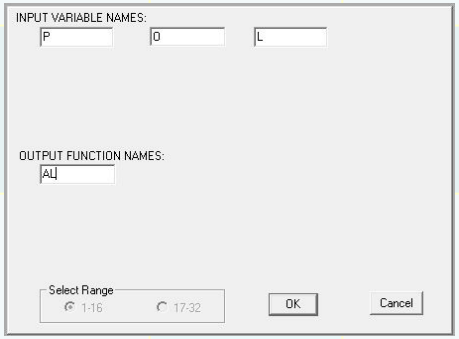
m	P	O	L	AL
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	0
6	1	1	0	1
7	1	1	1	1

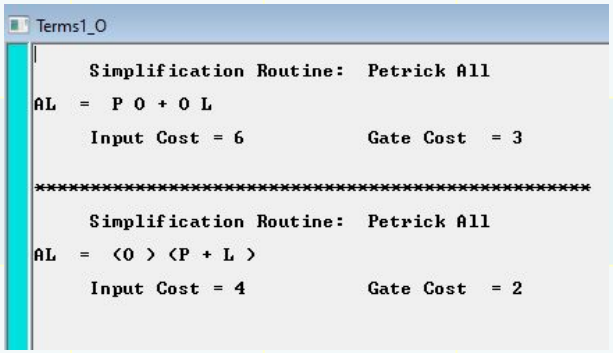
las ovejas estén fuera del corral y la puerta abierta o los lobos próximos y las ovejas fuera del corral

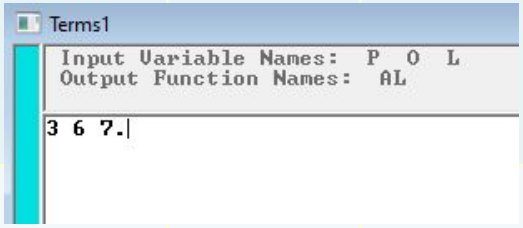
4.- Minimizar Para efectuar la simplificación podemos hacer uso de **LogicAid**

m	P	O	L	AL
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	0
6	1	1	0	1
7	1	1	1	1









$$AL_{(P,O,L)} = \Sigma\ (3, 6, 7)$$

$$FAL_{(P,\ O,\ L)} = O(P + L)$$

6.- Implementar

Para programar un DLP con la función deseada puede ser a través de :

1.- Captura esquemática

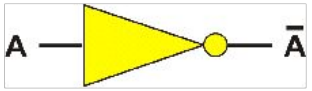
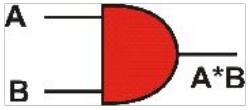
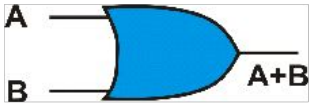
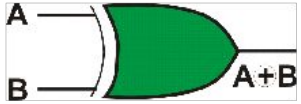
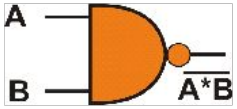
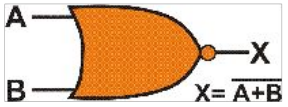
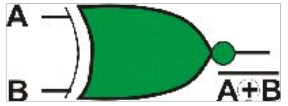
2.- Lenguaje de Descripción de Hardware (HDL)

a) Las Ecuaciones

b) La Tabla De Verdad

c) La Descripción Del Problema.

Lenguaje de Descripción de Hardware (HDL)

Operador	Descripción	Ecuación en ABEL-HDL	Símbolo
!	NOT	!A	
&	AND	A&B	
#	OR	A#B	
$	EXOR	A$B	
!&	NAND	!(A&B)	
!#	NOR	!(A#B)	
!$	EXNOR	!(A$B)	

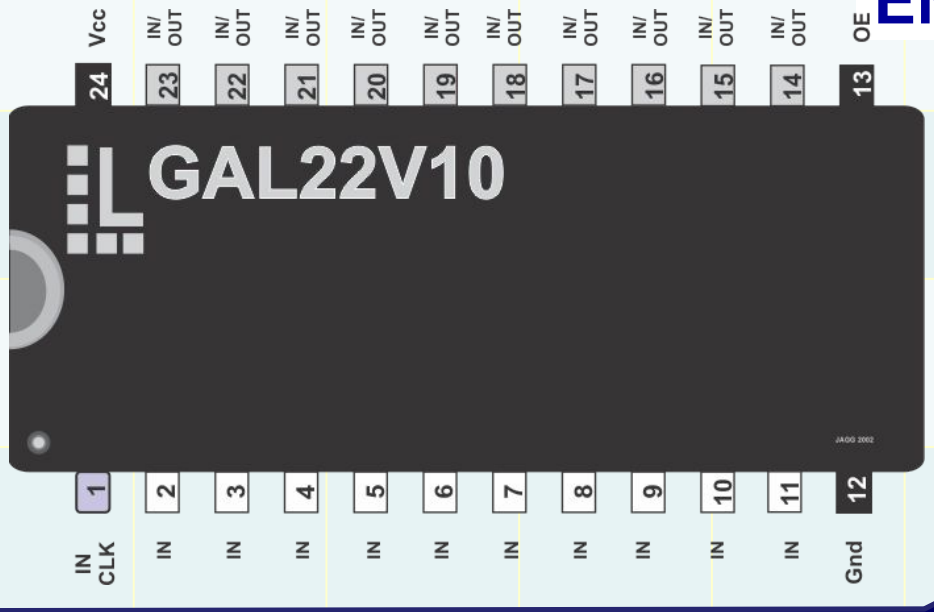
Ecuación

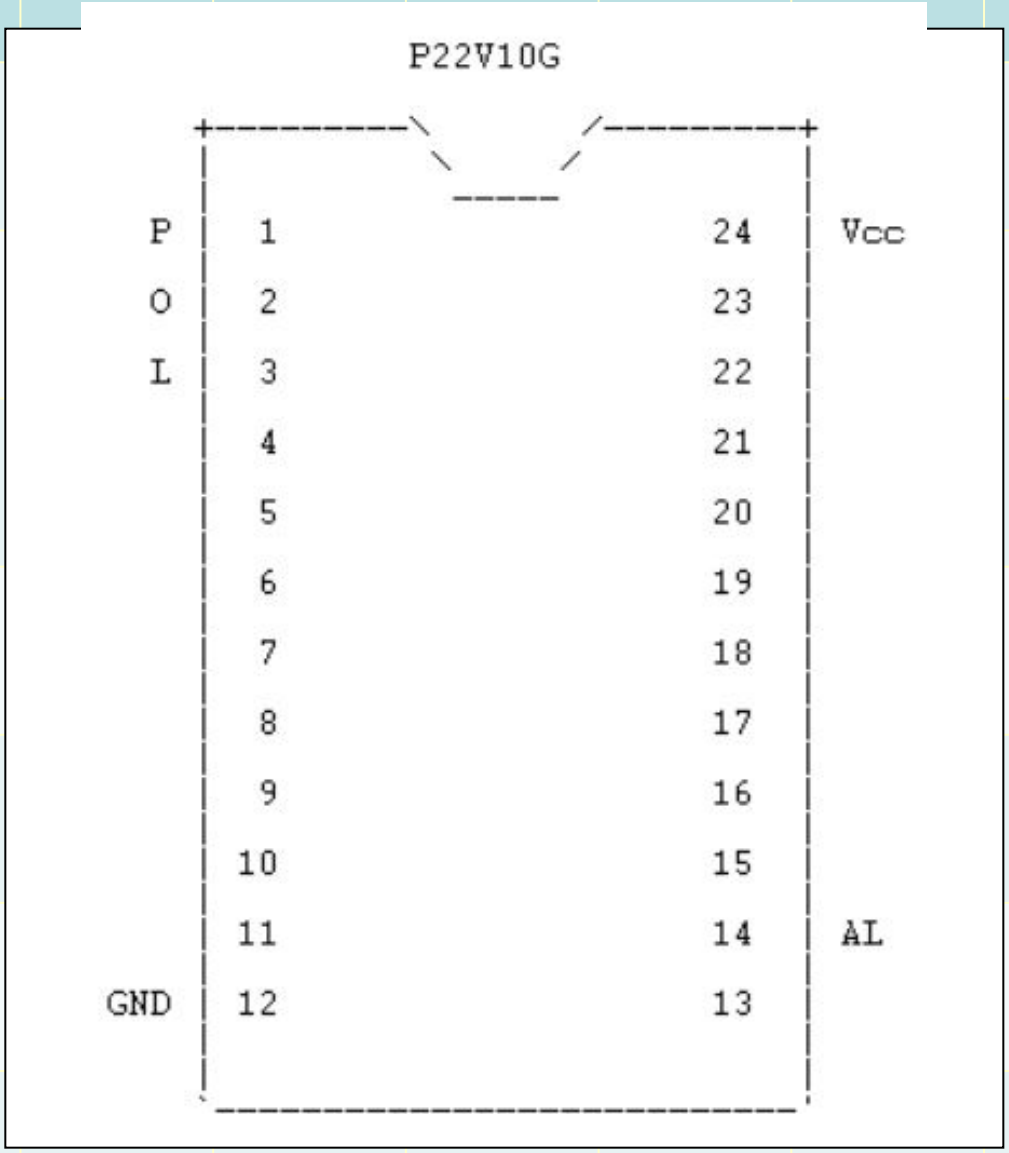
AL= O(P+L)

Formato Abel-HDL

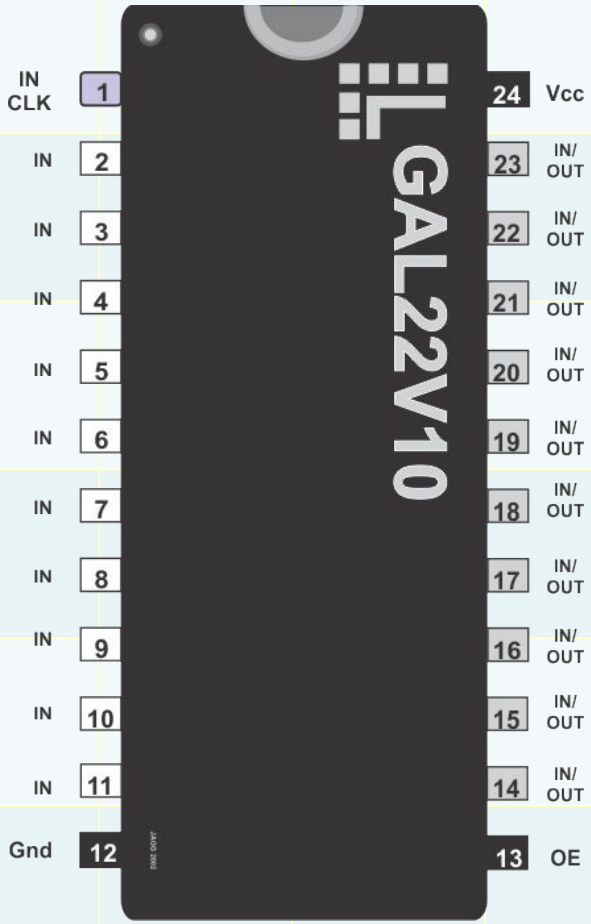
Al=O&(P#L);

```
MODULE ovejas
“19 SEP 2018
“Entradas
P,O,L pin 1,2,3;
“Salida
Al pin 14 istype 'com';
equations
Al=O&(P#L);
END
```



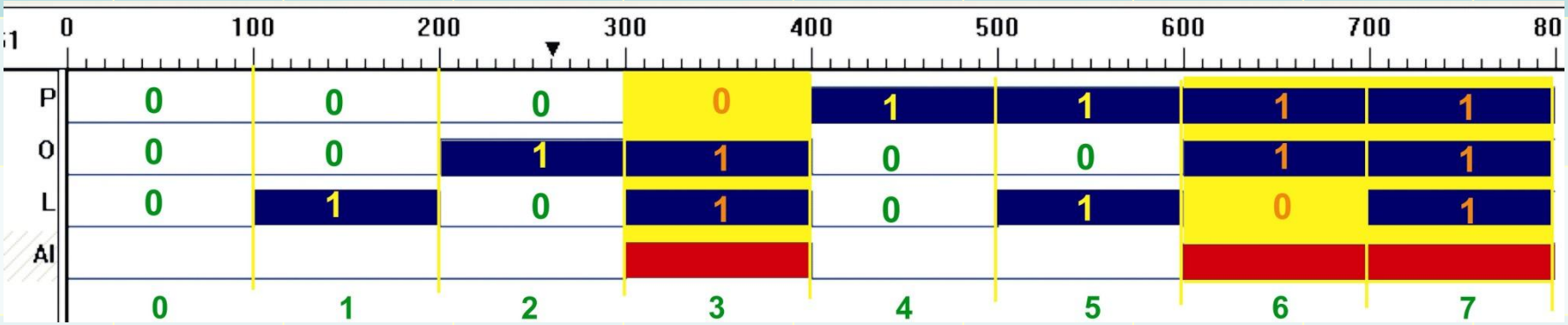

P22V10G
P
O
L
GND
1
2
3
4
5
6
7
8
9
10
11
12
24
23
22
21
20
19
18
17
16
15
14
13
Vcc
AL

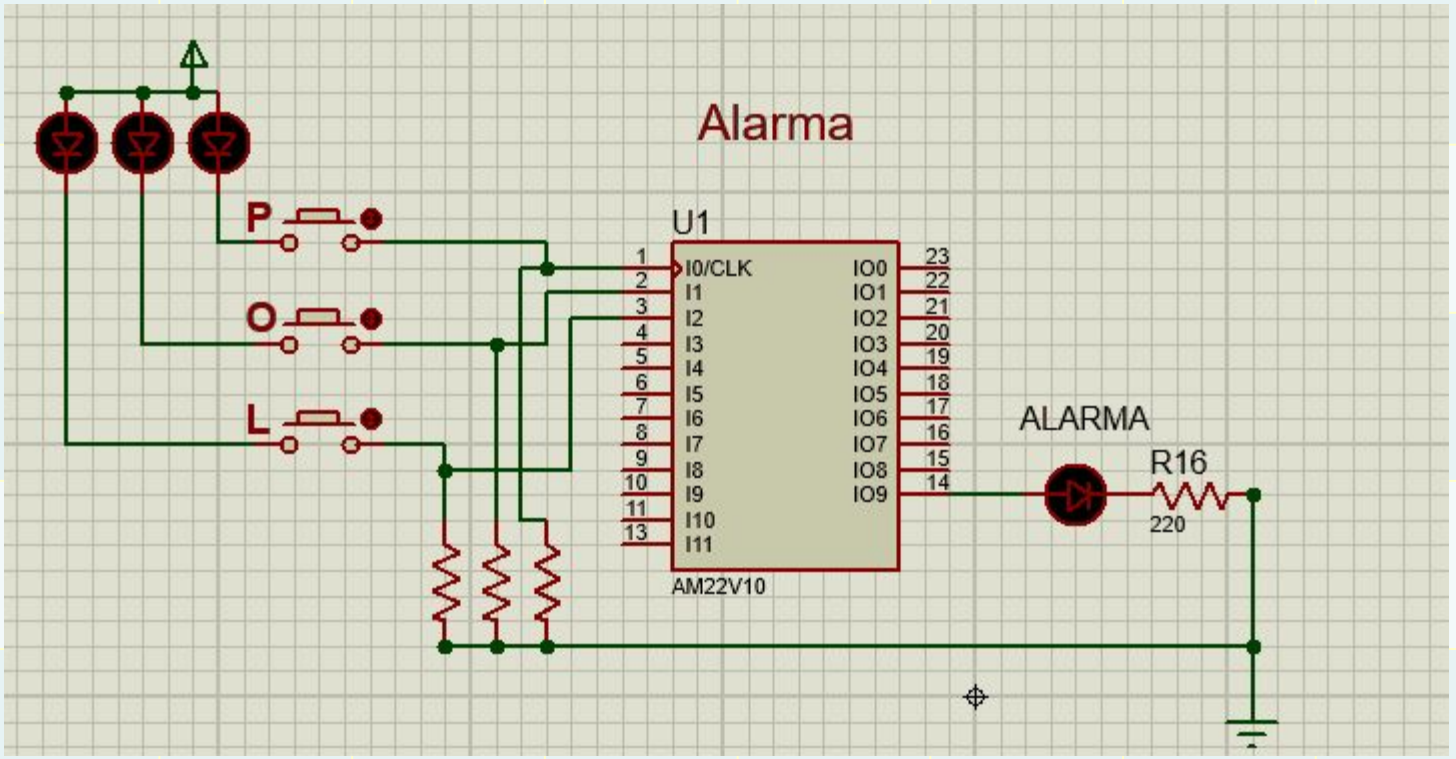
Archivo en formato ABEL-HDL incluyendo la simulación




```
MODULE ovejas
"entradas
P,O,L pin 1,2,3;
"salida
Al pin 14 istype 'com';
equations
Al=O&(P#L);
test_vectors
([P,O,L]->Al)
[0,0,0]->.x.;
[0,0,1]->.x.;
[0,1,0]->.x.;
[0,1,1]->.x.;
[1,0,0]->.x.;
[1,0,1]->.x.;
[1,1,0]->.x.;
[1,1,1]->.x.;
END
```

Simulación


0
100
200
300
400
500
600
700
P
O
L
Al
0 0 0 0 1 1 1 1
0 0 1 1 0 0 1 1
0 1 0 1 0 1 0 1
0 1 2 3 4 5 6 7


Alarma
P
O
L
U1
I0/CLK
I1
I2
I3
I4
I5
I6
I7
I8
I9
I10
I11
IO0
IO1
IO2
IO3
IO4
IO5
IO6
IO7
IO8
IO9
AM22V10
ALARMA
R16
220

Lenguaje de Descripción de Hardware (HDL)

a) Las Ecuaciones

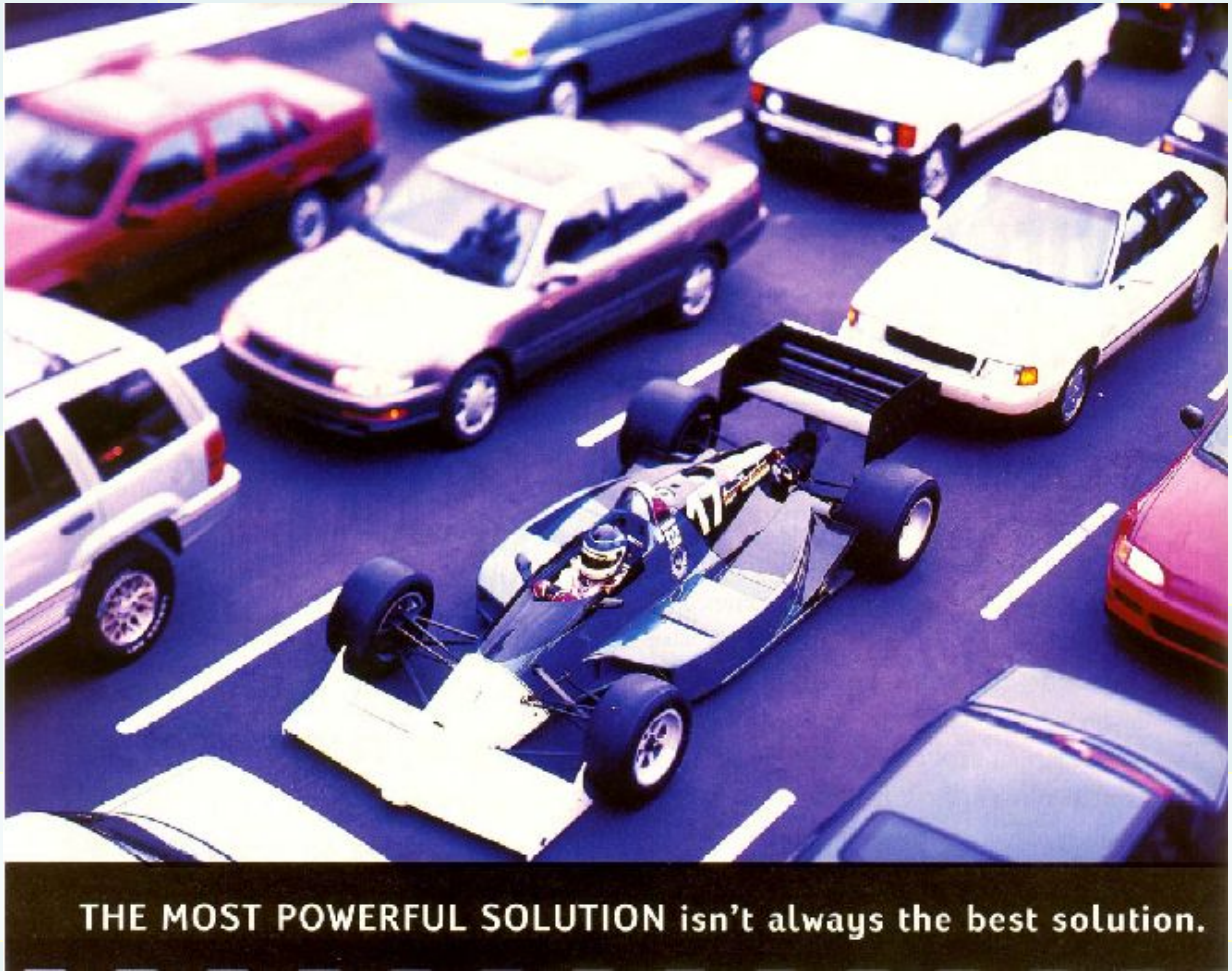
b) La Tabla De Verdad

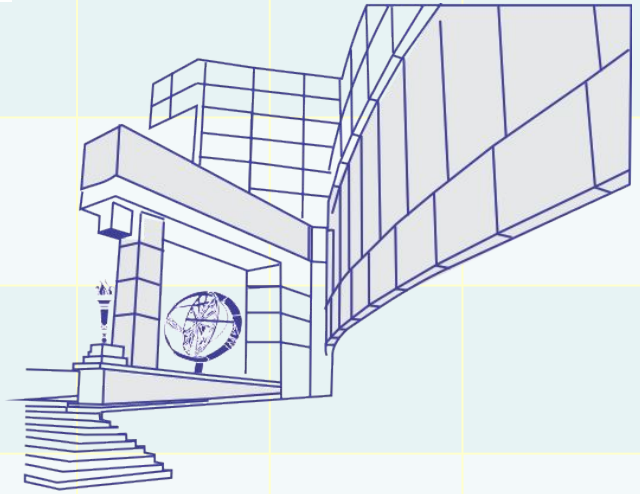
c) La Descripción Del Problema.

Partes de un programa en ABEL-HDL

1	**Module** inicio del programa máximo 8 caracteres no números
2	“ Comentarios opcional
3	**Declaration** asignación de terminales de entrada y salida
4	**Descripción lógica** (ecuaciones, Tabla de verdad etc.)
5	**Test_vectors** (vectores de prueba opcional)
6	**End** fin del programa

Always consider the whole system....
Siempre considere en todo el diseño de un sistema





Prof. Dr. H. Kirrmann
ABB Research Center, Baden, Switzerland

Que la solución más poderosa, a veces no es la mejor solución

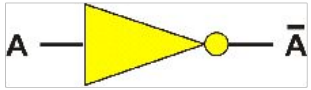
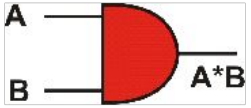
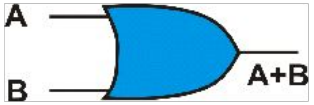
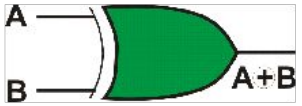
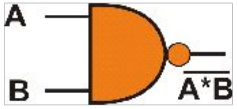
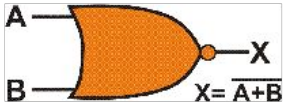
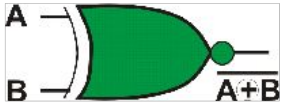
Actividades y proyectos en proceso

	Actividad	puntos	Fecha límite
PF5	Solución del examen	F	Martes 8 de octubre
PF6	Diseño Combinacional con HDL	F	Viernes 11 de octubre
PF7	Multiplexor	5	jueves 17 de octubre
AF3	Decodificador con Display	10	Martes 29 de octubre
PF8	Flip Flops	F	Jueves 31 de octubre
	Pulsos de sincronía	5	Martes 5 de noviembre
AF4	Diseño Secuencial	10	Miércoles 13 de noviembre
AF5	PIA	40	Día del examen

Lenguaje de Descripción de Hardware ABEL-HDL

Advanced Boolean Expression Language

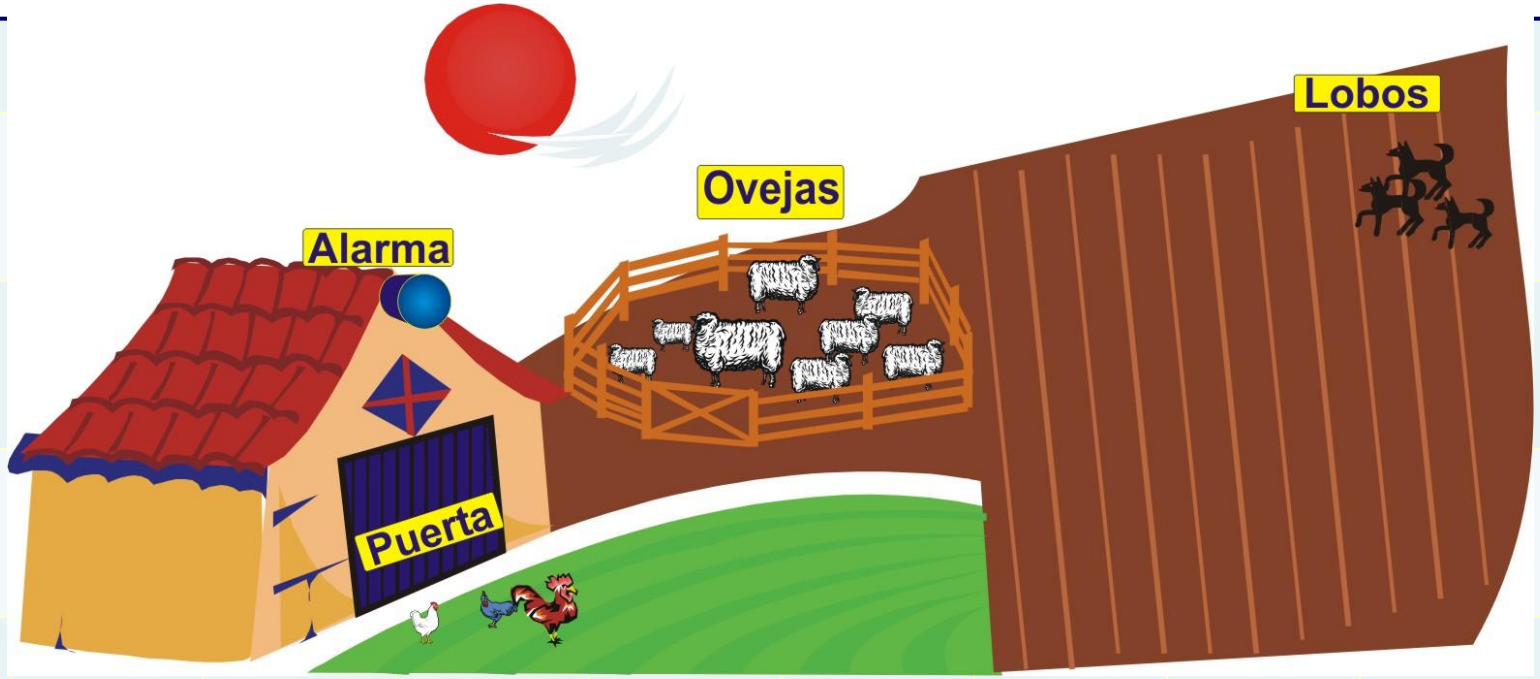
equations

Operador	Descripción	Ecuación en ABEL-HDL	Símbolo
!	NOT	!A	
&	AND	A&B	
#	OR	A#B	
$	EXOR	A$B	
!&	NAND	!(A&B)	
!#	NOR	!(A#B)	
!$	EXNOR	!(A$B)	

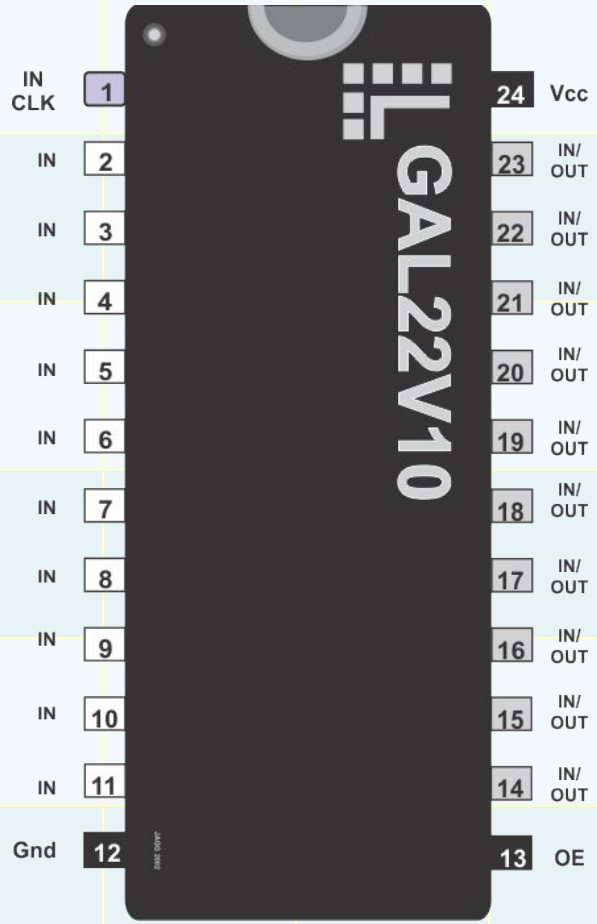
Ejemplo 1

En una granja se tiene:

- Un granero con una puerta muy grande y pesada en donde se requiere de varias personas para abrirla o cerrarla
- Un corral de ovejas
- Además ocasionalmente llegan lobos

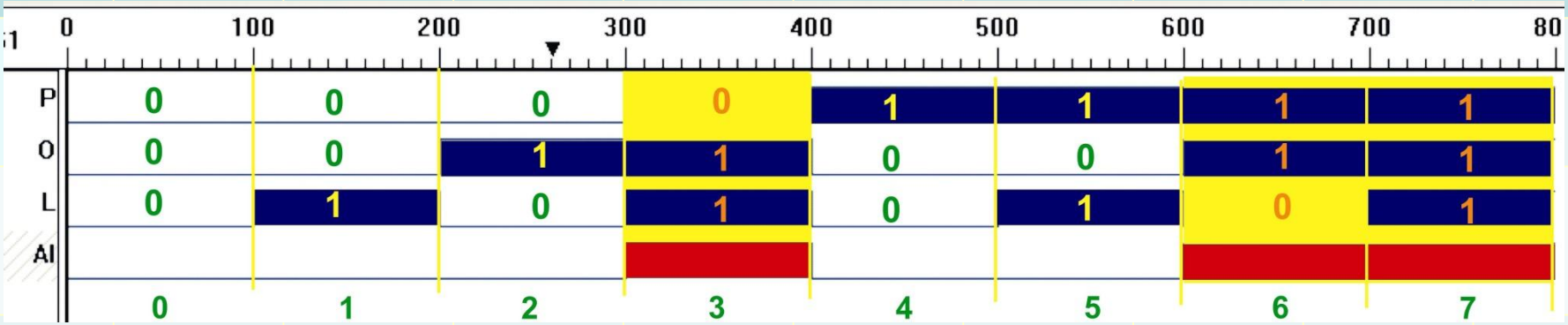


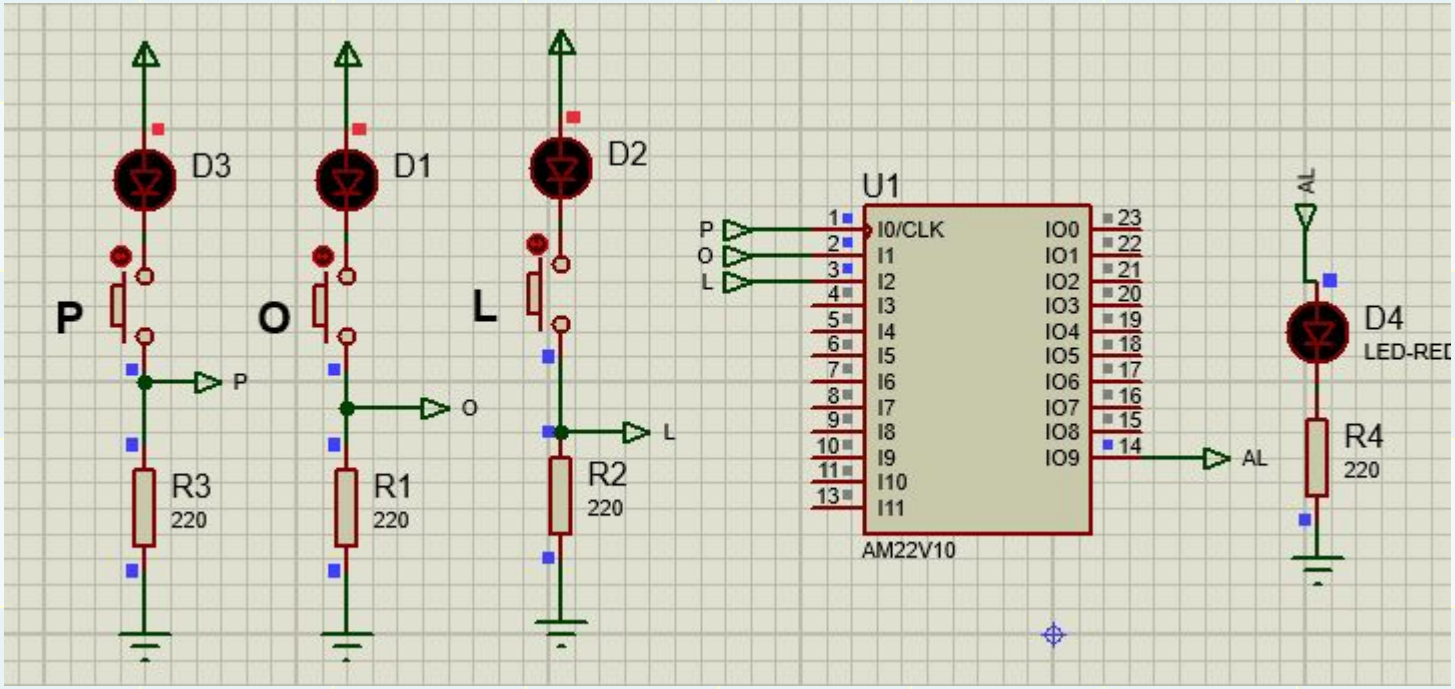
Archivo en formato ABEL-HDL incluyendo la simulación




```
MODULE ovejas
"entradas
P,O,L pin 1,2,3;
"salida
AI pin 14 istype 'com';
equations
AI=O&(P#L);
test_vectors
([P,O,L]->AI)
[0,0,0]->.x.;
[0,0,1]->.x.;
[0,1,0]->.x.;
[0,1,1]->.x.;
[1,0,0]->.x.;
[1,0,1]->.x.;
[1,1,0]->.x.;
[1,1,1]->.x.;
END
```

Simulación


0
100
200
300
400
500
600
700
P
O
L
Al
0 0 0 0 1 1 1 1
0 0 1 1 0 0 1 1
0 1 0 1 0 1 0 1
0 1 2 3 4 5 6 7


D3
D1
D2
P
O
L
R3
220
R1
220
R2
220
U1
I0/CLK
I1
I2
I3
I4
I5
I6
I7
I8
I9
I10
I11
IO0
IO1
IO2
IO3
IO4
IO5
IO6
IO7
IO8
IO9
AM22V10
AL
D4
LED-RED
R4
220

Método del Diseño Combinacional

1.- Especificar el Sistema

2.- Determinar entradas y salidas

3.- Construir la Tabla de Verdad

~~4.- Ecuaciones Mínimas~~

~~5.- Diagrama Esquemático~~

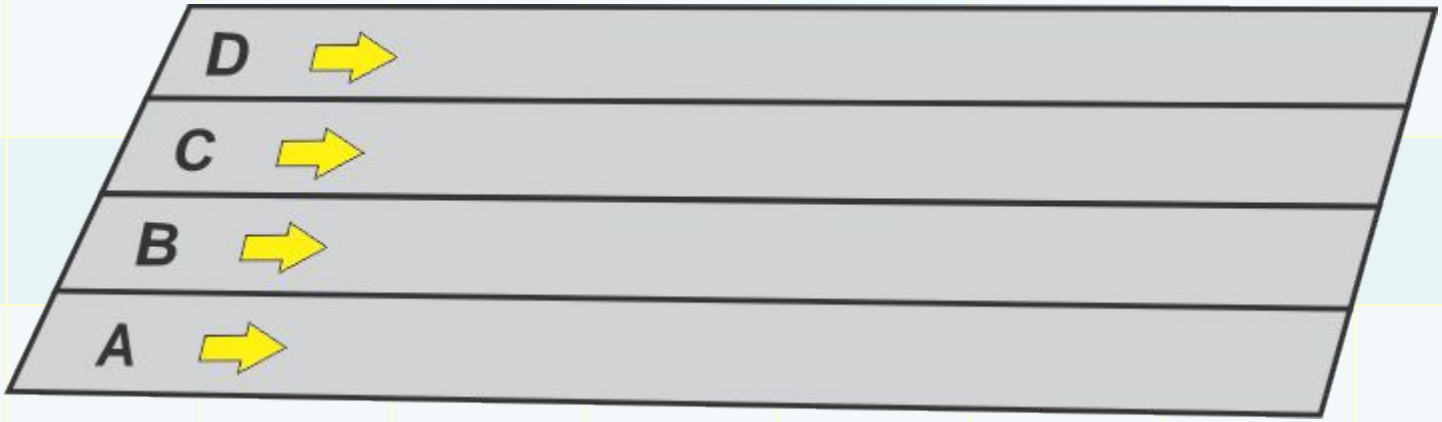
6.- Simulación

7.- Construir un Prototipo

ABEL-HDL
Truth_TABLE

Ejemplo 2

En un aeropuerto para aviones de carga que consta de solo de cuatro pistas (A, B, C y D), aterrizan dos tipos de aviones:



Ejemplo 2

El primero Antonov 225 Mriya, que por su tamaño requiere de tres pistas para aterrizar.

El segundo Airbus 300-600 ST que requiere de solo dos pistas.

Se solicita diseñar, simular y construir un prototipo de un sistema combinacional, que determine que tipo de avión puede aterrizar, teniendo en cuenta que:

1.-Especificar el Sistema

Las variables que intervienen son:

PISTAS ***A, B, C y D***

Disponible = *1*

No disponible =*0*

Aviones Antonov y Airbus

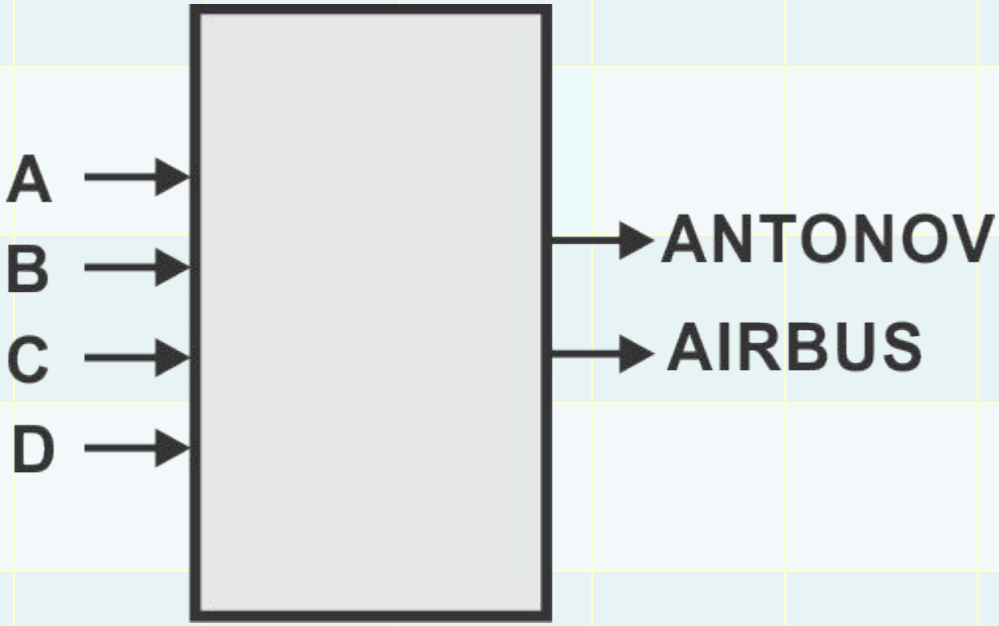
Permiso para aterrizar =*1*

No permiso para aterrizar =*0*

2.- Determinar las entradas y salidas

Las pistas A, B C y D son las entradas del sistema.

Mientras que permiso para aterrizar para el Antonov o Airbus, son las salidas que representamos a continuación en un diagrama de bloques.



3.- Trasladar el comportamiento a una tabla de verdad

hay que decidir el valor más conveniente de las salidas (0 o 1) para cada una de las combinaciones de entrada:

m	A	B	C	D	ANTONOV	AIRBUS
0	0	0	0	0		
1	0	0	0	1		
2	0	0	1	0		
3	0	0	1	1		
4	0	1	0	0		
5	0	1	0	1		
6	0	1	1	0		
7	0	1	1	1		
8	1	0	0	0		
9	1	0	0	1		
10	1	0	1	0		
11	1	0	1	1		
12	1	1	0	0		
13	1	1	0	1		
14	1	1	1	0		
15	1	1	1	1		

3.- Trasladar el comportamiento a una tabla de verdad

El Antonov 225 Mriya, requiere

de tres pistas para aterrizar.

m	A	B	C	D	ANTONOV	AIRBUS
0	0	0	0	0		
1	0	0	0	1		
2	0	0	1	0		
3	0	0	1	1		
4	0	1	0	0		
5	0	1	0	1		
6	0	1	1	0		
7	0	1	1	1	1	
8	1	0	0	0		
9	1	0	0	1		
10	1	0	1	0		
11	1	0	1	1		
12	1	1	0	0		
13	1	1	0	1		
14	1	1	1	0		
15	1	1	1	1		

3.- Trasladar el comportamiento a una tabla de verdad

El Antonov 225 Mriya, requiere

de tres pistas para aterrizar.

m	A	B	C	D	ANTONOV	AIRBUS
0	0	0	0	0		
1	0	0	0	1		
2	0	0	1	0		
3	0	0	1	1		
4	0	1	0	0		
5	0	1	0	1		
6	0	1	1	0		
7	0	1	1	1	1	
8	1	0	0	0		
9	1	0	0	1		
10	1	0	1	0		
11	1	0	1	1		
12	1	1	0	0		
13	1	1	0	1		
14	1	1	1	0	1	
15	1	1	1	1		

3.- Trasladar el comportamiento a una tabla de verdad

El Antonov 225 Mriya, requiere

de tres pistas para aterrizar.

m	A	B	C	D	ANTONOV	AIRBUS
0	0	0	0	0		
1	0	0	0	1		
2	0	0	1	0		
3	0	0	1	1		
4	0	1	0	0		
5	0	1	0	1		
6	0	1	1	0		
7	0	1	1	1	1	
8	1	0	0	0		
9	1	0	0	1		
10	1	0	1	0		
11	1	0	1	1		
12	1	1	0	0		
13	1	1	0	1		
14	1	1	1	0	1	
15	1	1	1	1	1	

3.- Trasladar el comportamiento a una tabla de verdad

El Antonov 225 Mriya, requiere

de tres pistas para aterrizar.

$$F\,An_{(A,\,B,\,C,\,D)} = \Sigma m\,(7, 14, 15)$$

m	A	B	C	D	ANTONOV	AIRBUS
0	0	0	0	0	0	
1	0	0	0	1	0	
2	0	0	1	0	0	
3	0	0	1	1	0	
4	0	1	0	0	0	
5	0	1	0	1	0	
6	0	1	1	0	0	
7	0	1	1	1	1	
8	1	0	0	0	0	
9	1	0	0	1	0	
10	1	0	1	0	0	
11	1	0	1	1	0	
12	1	1	0	0	0	
13	1	1	0	1	0	
14	1	1	1	0	1	
15	1	1	1	1	1	

3.- Trasladar el comportamiento a una tabla de verdad

El Airbus 300-600 ST requiere de

solo dos pistas

m	A	B	C	D	ANTONOV	AIRBUS
0	0	0	0	0	0	
1	0	0	0	1	0	
2	0	0	1	0	0	
3	0	0	1	1	0	
4	0	1	0	0	0	
5	0	1	0	1	0	
6	0	1	1	0	0	
7	0	1	1	1	1	
8	1	0	0	0	0	
9	1	0	0	1	0	
10	1	0	1	0	0	
11	1	0	1	1	0	
12	1	1	0	0	0	
13	1	1	0	1	0	
14	1	1	1	0	1	
15	1	1	1	1	1	

3.- Trasladar el comportamiento a una tabla de verdad

El Airbus 300-600 ST requiere de

solo dos pistas

m	A	B	C	D	ANTONOV	AIRBUS
0	0	0	0	0	0	
1	0	0	0	1	0	
2	0	0	1	0	0	
3	0	0	1	1	0	1
4	0	1	0	0	0	
5	0	1	0	1	0	
6	0	1	1	0	0	
7	0	1	1	1	1	
8	1	0	0	0	0	
9	1	0	0	1	0	
10	1	0	1	0	0	
11	1	0	1	1	0	
12	1	1	0	0	0	
13	1	1	0	1	0	
14	1	1	1	0	1	
15	1	1	1	1	1	

3.- Trasladar el comportamiento a una tabla de verdad

El Airbus 300-600 ST requiere de

solo dos pistas

m	A	B	C	D	ANTONOV	AIRBUS
0	0	0	0	0	0	
1	0	0	0	1	0	
2	0	0	1	0	0	
3	0	0	1	1	0	1
4	0	1	0	0	0	
5	0	1	0	1	0	
6	0	1	1	0	0	1
7	0	1	1	1	1	
8	1	0	0	0	0	
9	1	0	0	1	0	
10	1	0	1	0	0	
11	1	0	1	1	0	
12	1	1	0	0	0	
13	1	1	0	1	0	
14	1	1	1	0	1	
15	1	1	1	1	1	

3.- Trasladar el comportamiento a una tabla de verdad

El Airbus 300-600 ST requiere de

solo dos pistas

m	A	B	C	D	ANTONOV	AIRBUS
0	0	0	0	0	0	
1	0	0	0	1	0	
2	0	0	1	0	0	
3	0	0	1	1	0	1
4	0	1	0	0	0	
5	0	1	0	1	0	
6	0	1	1	0	0	1
7	0	1	1	1	1	
8	1	0	0	0	0	
9	1	0	0	1	0	
10	1	0	1	0	0	
11	1	0	1	1	0	1
12	1	1	0	0	0	
13	1	1	0	1	0	
14	1	1	1	0	1	
15	1	1	1	1	1	

3.- Trasladar el comportamiento a una tabla de verdad

El Airbus 300-600 ST requiere de

solo dos pistas

m	A	B	C	D	ANTONOV	AIRBUS
0	0	0	0	0	0	
1	0	0	0	1	0	
2	0	0	1	0	0	
3	0	0	1	1	0	1
4	0	1	0	0	0	
5	0	1	0	1	0	
6	0	1	1	0	0	1
7	0	1	1	1	1	
8	1	0	0	0	0	
9	1	0	0	1	0	
10	1	0	1	0	0	
11	1	0	1	1	0	1
12	1	1	0	0	0	1
13	1	1	0	1	0	
14	1	1	1	0	1	
15	1	1	1	1	1	

3.- Trasladar el comportamiento a una tabla de verdad

El Airbus 300-600 ST requiere de

solo dos pistas

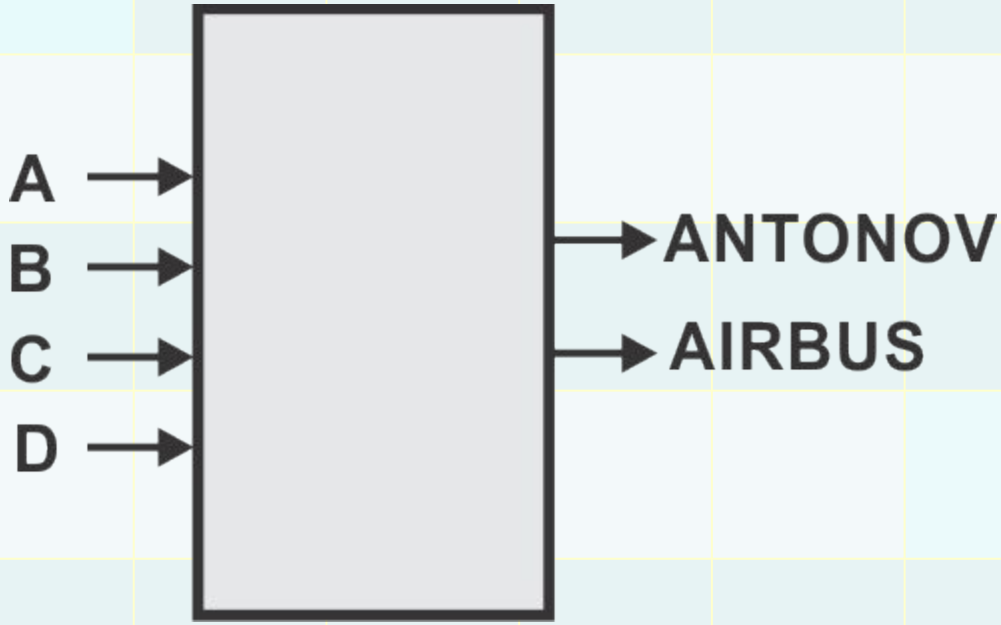
m	A	B	C	D	ANTONOV	AIRBUS
0	0	0	0	0	0	
1	0	0	0	1	0	
2	0	0	1	0	0	
3	0	0	1	1	0	1
4	0	1	0	0	0	
5	0	1	0	1	0	
6	0	1	1	0	0	1
7	0	1	1	1	1	
8	1	0	0	0	0	
9	1	0	0	1	0	
10	1	0	1	0	0	
11	1	0	1	1	0	1
12	1	1	0	0	0	1
13	1	1	0	1	0	1
14	1	1	1	0	1	
15	1	1	1	1	1	

3.- Trasladar el comportamiento a una tabla de verdad

El Airbus 300-600 ST requiere de solo dos pistas

$$F\,Ai_{(A,\,B,\,C,\,D)} = \Sigma m\,(3, 6, 11, 12, 13)$$

m	A	B	C	D	ANTONOV	AIRBUS
0	0	0	0	0	0	0
1	0	0	0	1	0	0
2	0	0	1	0	0	0
3	0	0	1	1	0	1
4	0	1	0	0	0	0
5	0	1	0	1	0	0
6	0	1	1	0	0	1
7	0	1	1	1	1	0
8	1	0	0	0	0	0
9	1	0	0	1	0	0
10	1	0	1	0	0	0
11	1	0	1	1	0	1
12	1	1	0	0	0	1
13	1	1	0	1	0	1
14	1	1	1	0	1	0
15	1	1	1	1	1	0



m	A	B	C	D	ANTONOV	AIRBUS
0	0	0	0	0	0	0
1	0	0	0	1	0	0
2	0	0	1	0	0	0
3	0	0	1	1	0	1
4	0	1	0	0	0	0
5	0	1	0	1	0	0
6	0	1	1	0	0	1
7	0	1	1	1	1	0
8	1	0	0	0	0	0
9	1	0	0	1	0	0
10	1	0	1	0	0	0
11	1	0	1	1	0	1
12	1	1	0	0	0	1
13	1	1	0	1	0	1
14	1	1	1	0	1	0
15	1	1	1	1	1	0

Programación en ABEL-HDL

```
MODULE cpistas
"21 sep 2022 Diseño Aeropuerto de 4 pistas
A,B,C,D pin 1..4;
ANTONOV,AIRBUS pin 15,14 istype 'com';

Truth_Table
([A,B,C,D]->[ANTONOV,AIRBUS])
[0,0,0,0]->[0,0];
[0,0,0,1]->[0,0];
[0,0,1,0]->[0,0];
[0,0,1,1]->[0,1];
[0,1,0,0]->[0,0];
[0,1,0,1]->[0,0];
[0,1,1,0]->[0,1];
[0,1,1,1]->[1,0];
[1,0,0,0]->[0,0];
[1,0,0,1]->[0,0];
[1,0,1,0]->[0,0];
[1,0,1,1]->[0,1];
[1,1,0,0]->[0,1];
[1,1,0,1]->[0,1];
[1,1,1,0]->[1,0];
[1,1,1,1]->[1,0];
```

m	A	B	C	D	ANTONOV	AIRBUS
0	0	0	0	0	0	0
1	0	0	0	1	0	0
2	0	0	1	0	0	0
3	0	0	1	1	0	1
4	0	1	0	0	0	0
5	0	1	0	1	0	0
6	0	1	1	0	0	1
7	0	1	1	1	1	0
8	1	0	0	0	0	0
9	1	0	0	1	0	0
10	1	0	1	0	0	0
11	1	0	1	1	0	1
12	1	1	0	0	0	1
13	1	1	0	1	0	1
14	1	1	1	0	1	0
15	1	1	1	1	1	0

```
MODULE cpistas
"21 sep 2022
" Diseño combinacional
"Aeropuerto de 4 pistas
A,B,C,D pin 1..4;
ANTONOV,AIRBUS pin 15,14 istype 'com';
```

```
Truth_Table
([A,B,C,D]->[ANTONOV,AIRBUS])
[0,0,1,1]->[0,1];
[0,1,1,0]->[0,1];
[0,1,1,1]->[1,0];
[1,0,1,1]->[0,1];
[1,1,0,0]->[0,1];
[1,1,0,1]->[0,1];
[1,1,1,0]->[1,0];
[1,1,1,1]->[1,0];
```

Las salidas de las combinaciones no listadas en la tabla las toma como cero

Programación en ABEL-HDL

m	A	B	C	D	ANTONOV	AIRBUS
0	0	0	0	0	0	0
1	0	0	0	1	0	0
2	0	0	1	0	0	0
3	0	0	1	1	0	1
4	0	1	0	0	0	0
5	0	1	0	1	0	0
6	0	1	1	0	0	1
7	0	1	1	1	1	0
8	1	0	0	0	0	0
9	1	0	0	1	0	0
10	1	0	1	0	0	0
11	1	0	1	1	0	1
12	1	1	0	0	0	1
13	1	1	0	1	0	1
14	1	1	1	0	1	0
15	1	1	1	1	1	0

```
MODULE cpistas
" 21 sep 2022
" Diseño combinacional
"Aeropuerto de 4 pistas
A,B,C,D pin 1..4;
ANTONOV,AIRBUS pin 15,14 istype 'com';
```

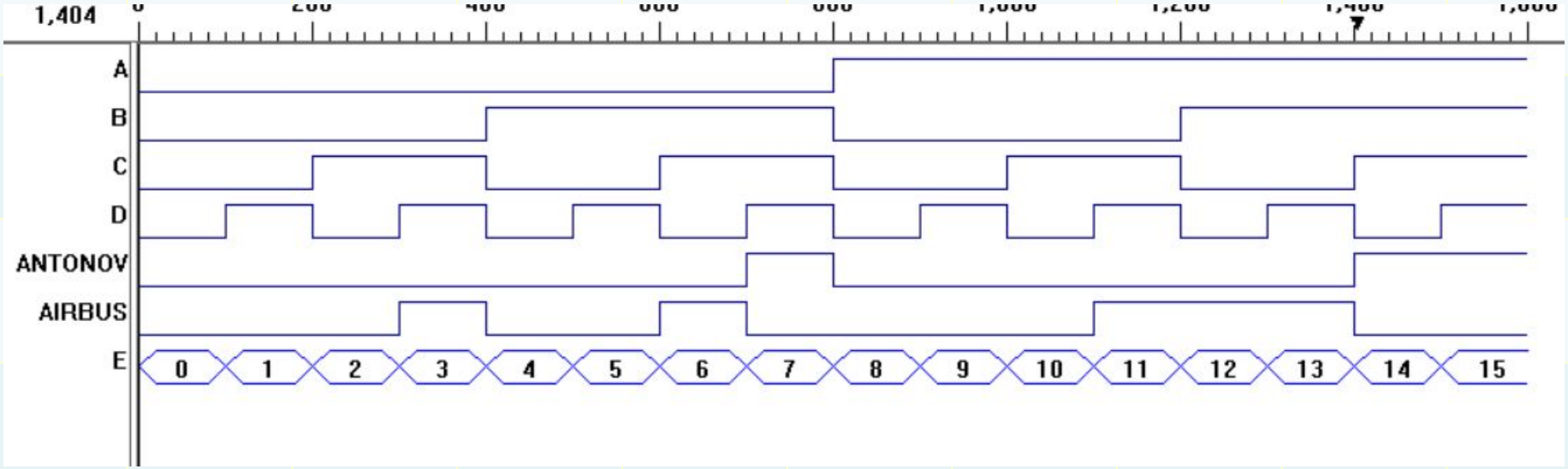
```
Truth_Table
([A,B,C,D]->[ANTONOV,AIRBUS])
[0,0,1,1]->[0,1];
[0,1,1,0]->[0,1];
[0,1,1,1]->[1,0];
[1,0,1,1]->[0,1];
[1,1,0,0]->[0,1];
[1,1,0,1]->[0,1];
[1,1,1,0]->[1,0];
[1,1,1,1]->[1,0];
```

```
Test_vectors
([A,B,C,D]->[ANTONOV,AIRBUS])
[0,0,0,0]->[0,0];
[0,0,0,1]->[0,0];
[0,0,1,0]->[0,0];
[0,0,1,1]->[0,1];
[0,1,0,0]->[0,0];
[0,1,0,1]->[0,0];
[0,1,1,0]->[0,1];
[0,1,1,1]->[1,0];
[1,0,0,0]->[0,0];
[1,0,0,1]->[0,0];
[1,0,1,0]->[0,0];
[1,0,1,1]->[0,1];
[1,1,0,0]->[0,1];
[1,1,0,1]->[0,1];
[1,1,1,0]->[1,0];
[1,1,1,1]->[1,0];
END
```

Las salidas de las combinaciones no listadas en la tabla las toma como cero

6.-Simulación

Test_vectors

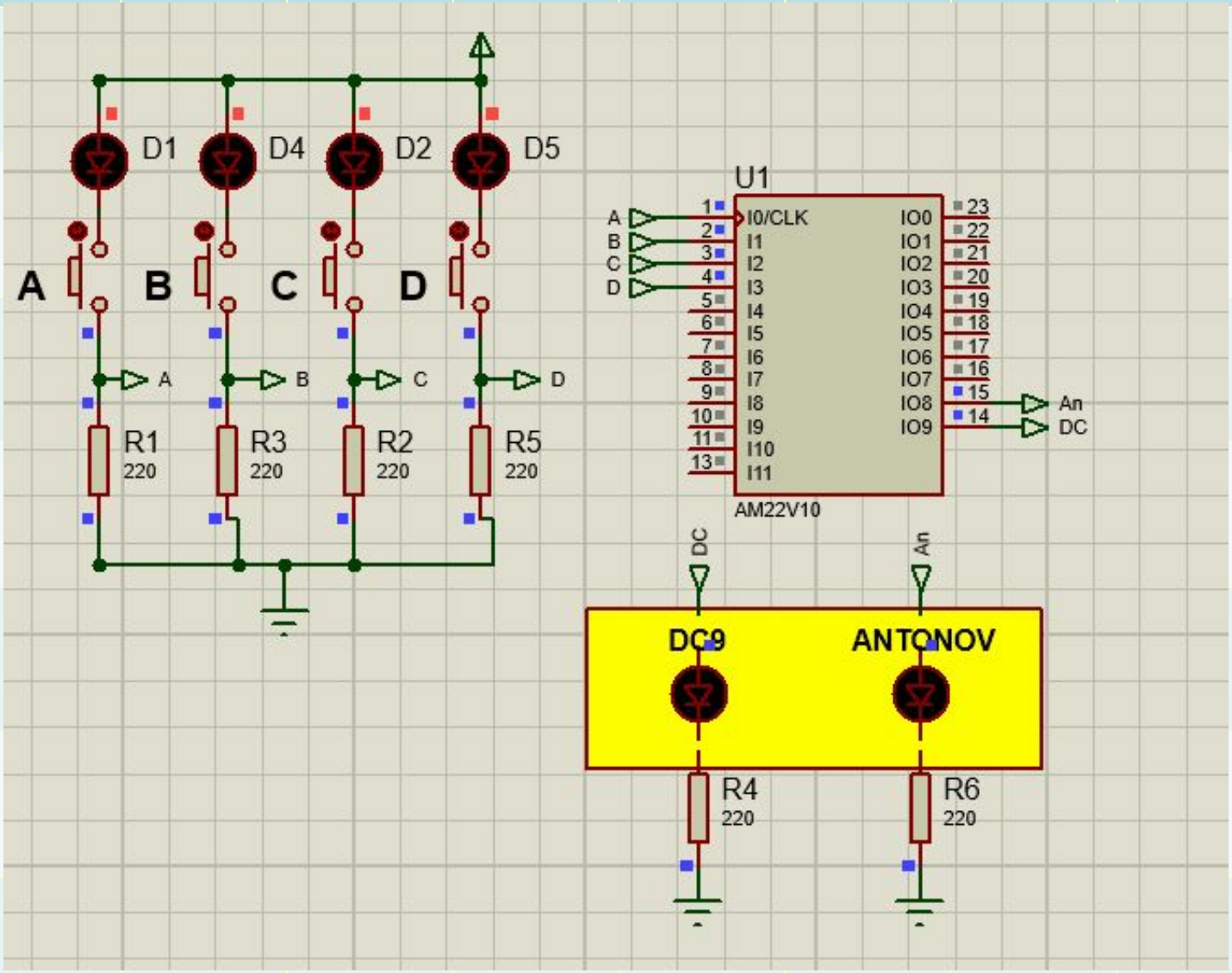


$$F\,An_{(A,\,B,\,C,\,D)} = \Sigma m\,(7,\,14,\,15)$$

$$F\,Ai_{(A,\,B,\,C,\,D)} = \Sigma m\,(3,\,6,\,11,\,12,\,13)$$

6.-Simulación

PROTEUS



5 Pistas

A
B
C
D
E

$F\ An_{(A, B, C, D)} = \Sigma m\ (7, 14, 15, 23, 28, 29, 30, 31)$

$F\ Ai_{(A, B, C, D)} = \Sigma m\ (3, 6, 11, 12, 13, 19, 22, 24, 25, 26, 27, 31)$

m	A	B	C	D	E	ANTONOV	AIRBUS	N(10)
3	0	0	0	1	1	0	1	1
6	0	0	1	1	0	0	1	1
7	0	0	1	1	1	1	0	2
11	0	1	0	1	1	0	1	1
12	0	1	1	0	0	0	1	1
13	0	1	1	0	1	0	1	1
14	0	1	1	1	0	1	0	2
15	0	1	1	1	1	1	0	2
19	1	0	0	1	1	0	1	1
22	1	0	1	1	0	0	1	1
23	1	0	1	1	1	1	0	2
24	1	1	0	0	0	0	1	1
25	1	1	0	0	1	0	1	1
26	1	1	0	1	0	0	1	1
27	1	1	0	1	1	0	1	1
28	1	1	1	0	0	1	0	2
29	1	1	1	0	1	1	0	2
30	1	1	1	1	0	1	0	2
31	1	1	1	1	1	1	1	3

En **ABEL-HDL** se pueden definir **conjuntos de valores binarios** para agrupar varias señales en una sola variable compuesta.
Por ejemplo: ***P=[A,B,C,D]; S=[AN,AI];***
Esto permite representar un grupo de bits como una variable única, similar a un valor decimal

P=[A,B,C,D];
S=[AN,AI];

Truth_Table
(P->S)
3->1;
6->1;
7->2;
11->1;
12->1;
13->1;
14->2;
15->2;

m	A	B	C	D	ANTONOV	AIRBUS	N(10)
0	0	0	0	0	0	0	
1	0	0	0	1	0	0	
2	0	0	1	0	0	0	
3	0	0	1	1	0	1	1
4	0	1	0	0	0	0	
5	0	1	0	1	0	0	
6	0	1	1	0	0	1	1
7	0	1	1	1	1	0	2
8	1	0	0	0	0	0	
9	1	0	0	1	0	0	
10	1	0	1	0	0	0	
11	1	0	1	1	0	1	1
12	1	1	0	0	0	1	1
13	1	1	0	1	0	1	1
14	1	1	1	0	1	0	2
15	1	1	1	1	1	0	2

5 Pistas

```
MODULE CPISTAS
A,B,C,D,E PIN 1..5;
AN,AB PIN 15,14 ISTYPE 'COM';
" 21 sep 2022
"JAGG
P=[A,B,C,D,E];
S=[AN,AB];
```

$$F\,An_{(A,\,B,\,C,\,D)} = \Sigma m\,(7, 14, 15)$$

$$F\,Ai_{(A,\,B,\,C,\,D)} = \Sigma m\,(3, 6, 11, 12, 13)$$

```
TRUTH_TABLE
(P->S)
3->1;
6->1;
7->2;
11->1;
12->1;
13->1;
14->2;
15->2;
19->1;
22->1;
23->2;
24->1;
25->1;
26->1;
27->1;
28->2;
29->2;
30->2;
31->3;
```

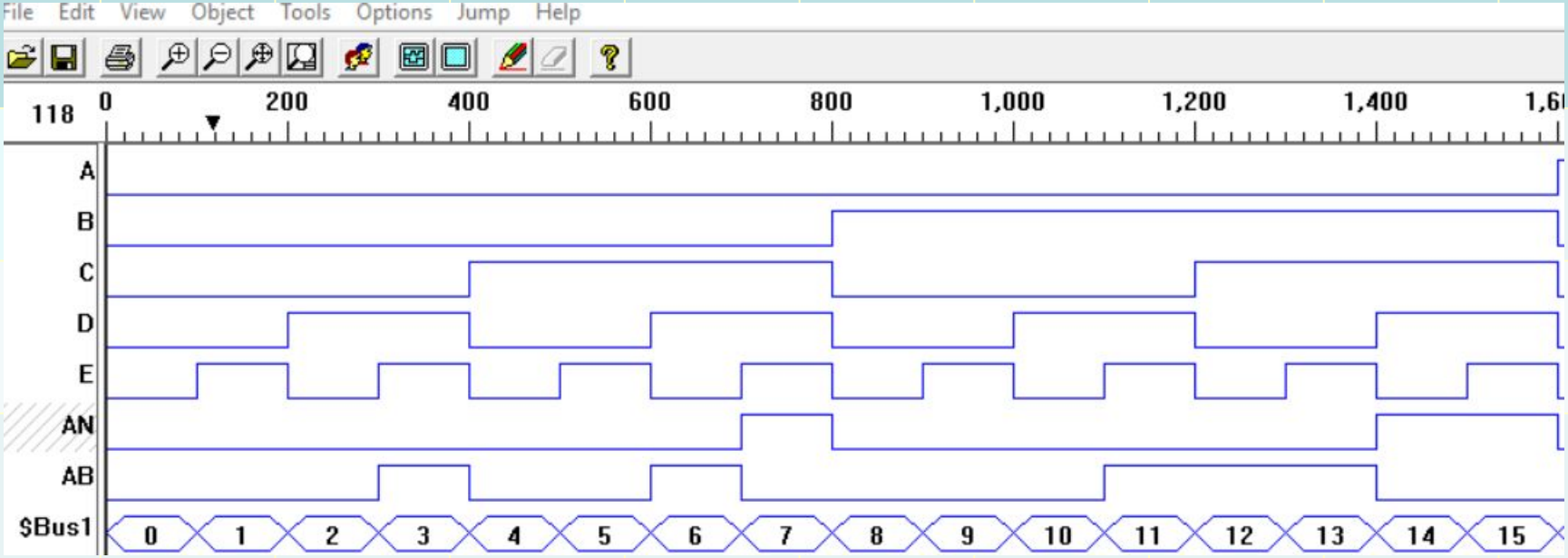
```
TEST_vECTORS
(P->S)
0->.X.;
1->.X.;
2->.X.;
3->.X.;
4->.X.;
5->.X.;
6->.X.;
7->.X.;
8->.X.;
9->.X.;
10->.X.;
11->.X.;
12->.X.;
13->.X.;
14->.X.;
15->.X.;
16->.X.;
17->.X.;
18->.X.;
19->.X.;
20->.X.;
21->.X.;
22->.X.;
23->.X.;
24->.X.;
25->.X.;
26->.X.;
27->.X.;
28->.X.;
29->.X.;
30->.X.;
31->.X.;
```

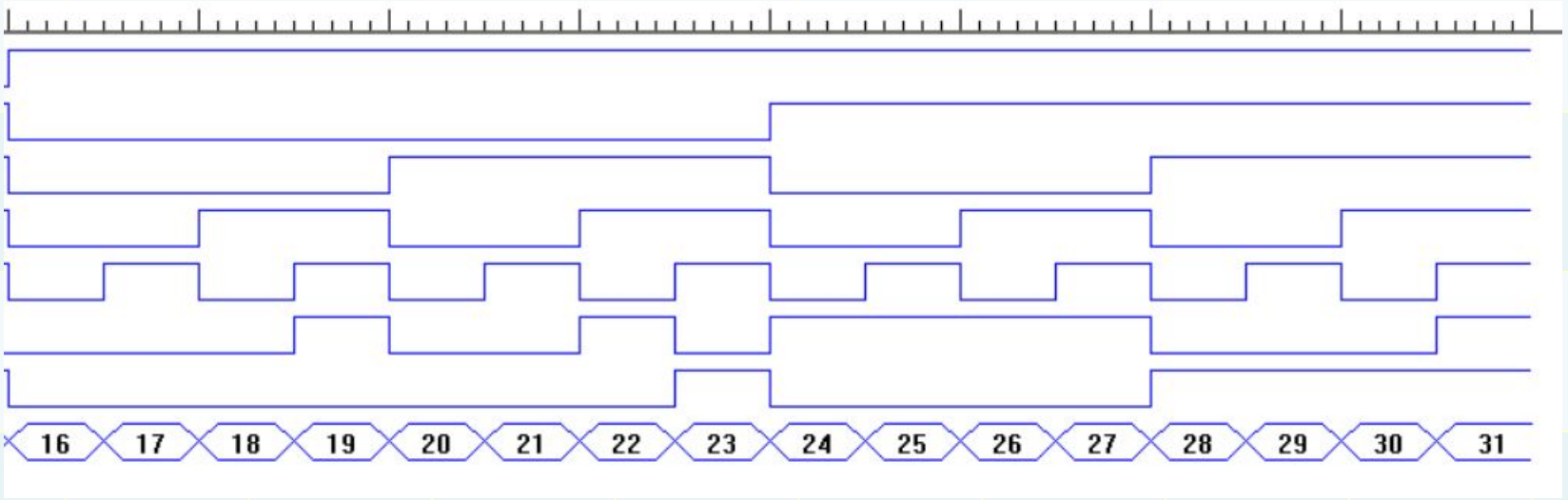
```
MODULE cpistas
" 21 sep 2022
" Diseño combinacional
"Aeropuerto de 4 pistas
A,B,C,D pin 1..4;
ANTONOV,AIRBUS pin 15,14 istype 'com';
```

```
Truth_Table
(P->S)
3->1;
6->1;
7->2;
11->1;
12->1;
13->1;
14->2;
15->2;
```

```
Test_vectors
(P->S)
0->0;
1->0;
2->0;
3->1;
4->0;
5->0;
6->0;
7->2;
8->0;
9->0;
10->0;
11->1;
12->1;
13->1;
14->0;
15->2;
END
```

Las salidas de las combinaciones no listadas en la tabla las toma como cero


File Edit View Object Tools Options Jump Help
118
0
200
400
600
800
1,000
1,200
1,400
A
B
C
D
E
AN
AB
$Bus1
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15


16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Método del diseño Combinacional HDL y PLD

Circuito		Captura esquemática
Ecuación	Equations	ABEL-HDL
Tabla de verdad	Truth_table	
Descripción del comportamiento	When, Then, Else	

Lenguaje de Descripción de Hardware (HDL)
Ecuaciones

Text Editor - [al.abl]

File Edit View Templates Tools Options Window Help

```
MODULE al
"este es un archivo en formato ABEL-HDL
"para implementar en un GAL16V8
"el problema de la alarma
" Resuelto por la ecuacion AL=O(P+L)
"Entradas con asignación de terminales
P,O,L pin  1,2,3;
" Salida asignada a la terminal 19
AL pin 19 istype 'com';
equations
AL= O&(P#L);
"se incluye la simulación
Test_vectors
([P,O,L]->AL)
[0,0,0]->.x.;
[0,0,1]->.x.;
[0,1,0]->.x.;
[0,1,1]->.x.;
[1,0,0]->.x.;
[1,0,1]->.x.;
[1,1,0]->.x.;
[1,1,1]->.x.;
END
```

Método del Diseño Combinacional

1.- Especificar el Sistema

2.- Determinar entradas y salidas

3.- Construir la Tabla de Verdad

4.- Ecuaciones Mínimas

~~5.- Diagrama Esquemático~~

6.- Simulación

7.- Construir un Prototipo

HDL

Método del diseño Combinacional HDL y PLD

Circuito		Captura esquemática
Ecuación	Equations	ABEL-HDL
Tabla de verdad	Truth_table	
Descripción del comportamiento	When, Then, Else	

Proyecto Formativo 7

Diseño de sistemas combinacionales

Objetivo

En el AF2 resolviste el Problema 4 mediante captura esquemática. Ahora, en el PF7, deberás reimplantar el mismo problema utilizando un Lenguaje de Descripción de Hardware (HDL), específicamente ABEL-HDL, siguiendo un flujo de diseño estructurado.

Competencia específica

Aplicar un método sistemático de diseño digital para trasladar un requerimiento funcional (Problema 4 del AF2) a una implementación en HDL sobre un PLD, realizando la simulación para comprobar su funcionamiento correcto, y validando la solución mediante la implementación en una tablilla de conexiones.

Proyecto Formativo 7

Universidad Autónoma de Nuevo León
Facultad de Ingeniería Mecánica y Eléctrica
Coordinación General de Ingeniería Electrónica

Electrónica Digital
Proyecto Formativo 6
Diseño de sistemas combinacionales

Comprensión, análisis, aplicación del Diseño y la simulación de sistemas combinacionales usando como ejercicio el problema 4 propuesto en tu examen de medio curso, mediante el uso de Lenguaje de Descripción de Hardware HDL y su simulación en PROTEUS usando un Dispositivo Lógico Programable PLD en su implementación del prototipo.

Método propuesto para el diseño combinacional con HDL (Flujo de diseño)

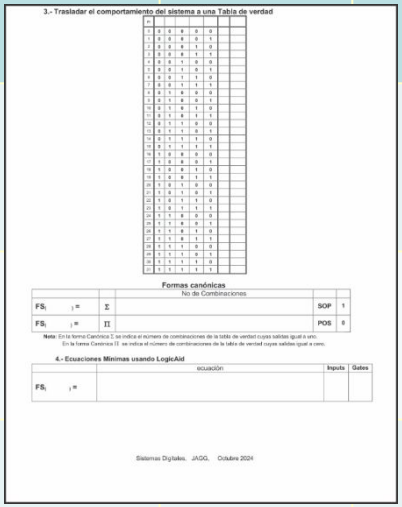
1.- Especificar el sistema.
2.-Determinar entradas y salidas (Diagrama de Bloques).
3.- Trasladar el comportamiento a una tabla de verdad. Representar la ecuación en sus formas canonicas SOP Σ y POS Π — Código ABEL-HDL
4.- Ecuaciones Mínimas
5.-Simulación.
6.- Construcción del prototipo

1.- Especificar el sistema: Problema No 4 propuesto en tu examen Redacción:

2.- Determinar entradas y salidas (Dibujo del Diagrama de Bloques).

Sistemas Digitales, JAGG, Octubre 2024

Diagrama de Bloques



3.- Trasladar el comportamiento del sistema a una Tabla de verdad

Formas canónicas — No de Combinaciones

$FS_{\_\_}$ =	Σ	SOP	1
$FS_{\_\_}$ =	Π	POS	0

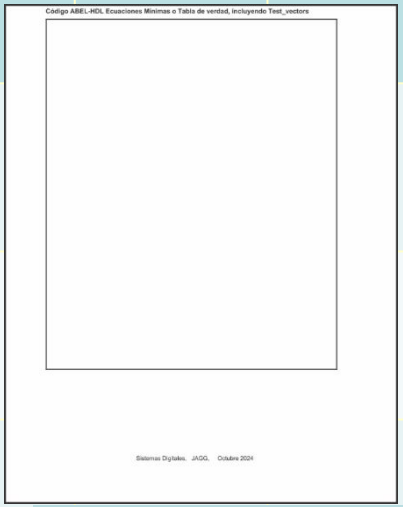
Nota: En la forma Canónica Σ se indica el número de combinaciones de la tabla de verdad cuyas salidas igualan a uno. En la forma Canónica Π se indica el número de combinaciones de la tabla de verdad cuyas salidas igualan a cero.

4.- Ecuaciones Mínimas usando LogicAid

	ecuación	Inputs	Gates
$FS_{\_\_}$ =			

Sistemas Digitales, JAGG, Octubre 2024

Tabla de verdad formas canónicas



Código ABEL-HDL Ecuaciones Mínimas o Tabla de verdad, incluyendo Test_vectors

Sistemas Digitales, JAGG, Octubre 2024

Código ABEL-HDL Test_Vectors

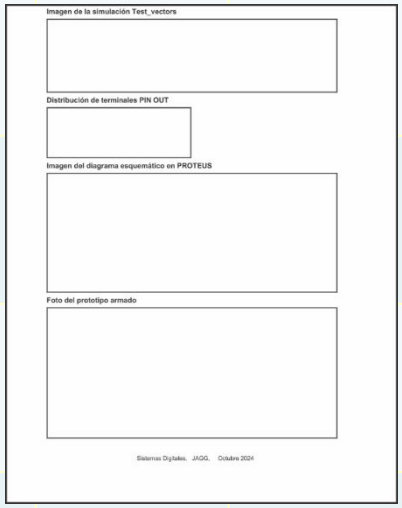


Imagen de la simulación Test_vectors

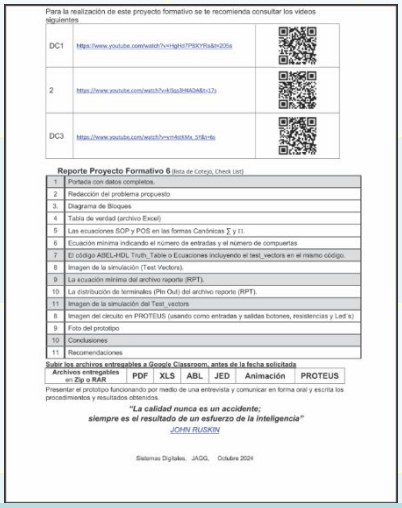
Distribución de terminales PIN OUT

Imagen del diagrama esquemático en PROTEUS

Foto del prototipo armado

Sistemas Digitales, JAGG, Octubre 2024

Imagen de la simulación, PIN OUT Diagrama esquemático en PROTEUS



Para la realización de este proyecto formativo se te recomienda consultar los videos siguientes

DC1	https://www.youtube.com/watch?v=HgHd7P8KYRs&t=205s	
2	https://www.youtube.com/watch?v=k9qo3HMA2MM&t=17s	
DC3	https://www.youtube.com/watch?v=n4ottMYa_3Y&t=4s	

Reporte Proyecto Formativo 6 (Lista de Cotejo, Check List)

1	Portada con datos completos.
2	Redacción del problema propuesto
3.	Diagrama de Bloques
4	Tabla de verdad (archivo Excel)
5	Las ecuaciones SOP y POS en las formas Canónicas ∑ y Π.
6	Ecuación mínima indicando el número de entradas y el número de compuertas
7	El código ABEL-HDL Truth_Table o Ecuaciones incluyendo el test_vectors en el mismo código.
8	Imagen de la simulación (Test Vectors).
9	La ecuación mínima del archivo reporte (RPT).
10	La distribución de terminales (Pin Out) del archivo reporte (RPT).
11	Imagen de la simulación del Test_vectors
8	Imagen del circuito en PROTEUS (usando como entradas y salidas botones, resistencias y Led´s)
9	Foto del prototipo
10	Conclusiones
11	Recomendaciones

Subir los archivos entregables a Google Classroom, antes de la fecha solicitada

Archivos entregables en Zip o RAR	PDF	XLS	ABL	JED	Animación	PROTEUS

Presentar el prototipo funcionando por medio de una entrevista y comunicar en forma oral y escrita los procedimientos y resultados obtenidos.

"La calidad nunca es un accidente;
siempre es el resultado de un esfuerzo de la inteligencia"
JOHN RUSKIN

Sistemas Digitales, JAGG, Octubre 2024

Reporte, Archivos entregables

Con que se quedan de esta clase ?