Aquí mis soluciones para el semáforo doble. Véanlos funcionando en simulación, sólo copien los códigos en un nuevo proyecto y lleven el archivo JEDEC a ISIS de Proteus.

La primera es muy similar al código que vimos en clase para el semáforo individual, utilizando una máquina de estados; la diferencia radica en que en vez de utilizar un vector de 3 salidas (V, A, R), ahora se utiliza un vector de 6 salidas para manejar los dos semáforos al mismo tiempo.

```
module sema_doble
title 'Controlador de Semáforo'
reloj, reset pin 1, 2;
verde1, ambar1, rojo1, verde2, ambar2, rojo2 pin 14,15,16,17,18,19 istype 'reg,dc';
"Las salidas del contador se declaran como nodos
Count3..Count0 node istype 'reg,dc';
Counter = [Count3..Count0];
"Se definen las condiciones de los estados
Luz = [verde1, ambar1, rojo1, verde2, ambar2, rojo2];
GO = [ 1, 0, 0, 0, 0, 1 ];
CAUTION1 = [ 1, 1, 0, 0, 0, 1 ];
CAUTION2 = [ 0, 0, 1, 1, 1, 0 ];
STOP = [ 0, 0, 1, 1, 0, 0 ];
INI= [0, 0, 0, 0, 0, 0];
"Monitoreo de señales para el reloj y el reset
Equations
Counter.ar= reset;
Counter.clk = reloj;
Luz.clk = reloj;
"Comienza diagrama de estados, nótese que el diagrama se nombra igual que el set de salida
state_Diagram Luz
"Condición inicial, necesaria para el arranque

state INI:
IF reset then INI
ELSE goto GO with Counter:=0;
"Estados del semáforo

state GO:
IF (Counter < 15) then GO with
Counter := Counter + 1;
ELSE goto CAUTION1 with
Counter := 0;

state CAUTION1:
IF (Counter != 3) then CAUTION1 with
Counter := Counter + 1;
ELSE goto STOP with
Counter := 0;

state STOP:
IF (Counter < 15) then STOP with
Counter := Counter + 1;
ELSE goto CAUTION2 with
Counter := 0;

state CAUTION2:
IF (Counter != 3) then CAUTION2 with
Counter := Counter + 1;
ELSE goto GO with
```


Dr. Juan C. Herrera-Lozada.
IPN - CIDETEC, Mayo 2013.

```
Counter := 0;


"Vectores de prueba, con la directiva @repeat
TEST_VECTORS
([reloj, reset]->[Luz])
@repeat 2 {[.c., 1]->[.x.];};   "Se propone un reset
@repeat 50 {[.c., 0]->[.x.];};  "Trabajo normal
end
```

La segunda solución muestra una aproximación para lograr el parpadeo. En esta ocasión utilicé una descripción a través de una tabla de transiciones. Quedaría muy bien con más estados, aunque estamos limitados a sólo 16, a razón de 4 bits para el contador (que aunque q3, q2, q1 y q0 están declarados como nodos, ISP Lever los asigna como salidas físicas) y las 6 salidas para los dos semáforos, teniendo el total de las 10 salidas del GAL22V10.

```
Module sema_parp
Title 'Semaforo doble con parpadeo'

"Pines de etrada
        reloj, rst pin 1,2;
"Pines de salida
        v1,a1,r1,v2,a2,r2 pin 14..19 istype'com,dc';
        q3,q2,q1,q0 node istype 'reg,dc';
"Set
        Q=[q3,q2,q1,q0];
x=.x.;
c=.c.;
Equations
"Descripción de control
        Q.clk=reloj;
        Q.ar=rst;

Truth_table
([Q]:>[Q]->[v1,a1,r1,v2,a2,r2]);
 [0]:>[1]->[1,0,0,0,0,1];
 [1]:>[2]->[1,0,0,0,0,1];
 [2]:>[3]->[1,0,0,0,0,1];
 [3]:>[4]->[0,0,0,0,0,1];
 [4]:>[5]->[1,0,0,0,0,1];
 [5]:>[6]->[0,0,0,0,0,1];
 [6]:>[7]->[1,0,0,0,0,1];
 [7]:>[8]->[0,1,0,0,0,1];
 [8]:>[9]->[0,0,1,1,0,0];
 [9]:>[10]->[0,0,1,1,0,0];
 [10]:>[11]->[0,0,1,1,0,0];
 [11]:>[12]->[0,0,1,0,0,0];
 [12]:>[13]->[0,0,1,1,0,0];
 [13]:>[14]->[0,0,1,0,0,0];
 [14]:>[15]->[0,0,1,1,0,0];
 [15]:>[0]->[0,0,1,0,1,0];

Test_vectors
([reloj, rst] -> [Q,v1,a1,r1,v2,a2,r2]);
[ c, 1] -> [x,x,x,x,x,x,x] ;
@repeat 50 {[ c, 0] -> [x,x,x,x,x,x,x] ;};
End
```


Dr. Juan C. Herrera-Lozada.
IPN - CIDETEC, Mayo 2013.