DISEÑO DE SISTEMAS DIGITALES
Curso Propedéutico
Tópico Práctico. No. 2
"Diseño de contadores con ABEL-HDL"

Dr. Juan Carlos Herrera Lozada
M. en C. Juan Carlos González Robles
{jlozada; jgrobles}@ipn.mx

Instituto Politécnico Nacional
Centro de Innovación y Desarrollo Tecnológico en Cómputo

Abril 2013

Campo 1: **Datos Personales.**

Campo 2: **Objetivos**.

- Diseño de circuitos de lógica secuencial.
- Descripción de contadores con ABEL – HDL, utilizando operadores aritméticos y tabla de transiciones.
- Simulación de circuitos secuenciales e implementación en PLDs de arquitectura simple.

Campo 3: **Desarrollo de la Práctica**.

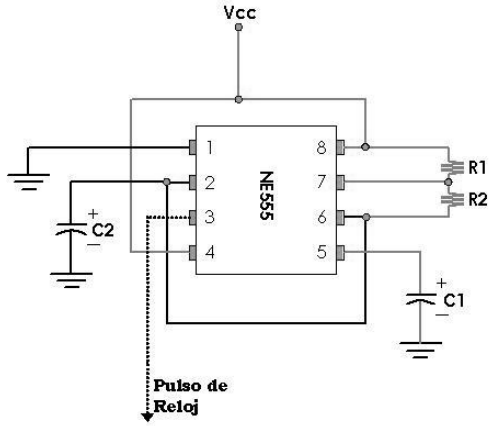
Nota: Para los diseños, anexar los respectivos códigos y simulaciones según el caso.

Preparación: Para el diseño secuencial será necesario que utilices una señal de reloj. Para generar este pulso de reloj, refiérete a la siguiente nota.

Configuración del Temporizador 555.

Realiza las conexiones sobre un temporizador 555, para implementar un multivibrador astable (conmuta entre un estado alto y uno bajo continuamente) que sea el reloj de tu contador. El pulso de reloj debe tener una duración aproximada de 1 segundo (frecuencia de 1 Hz).

Para las conexiones, auxíliate del siguiente diagrama. Observa que entre el los pines 2 y 6 existe un puente que los une; los pines 4 y 8 se conectan a Vcc de la fuente de 5 Volts. Comprueba el funcionamiento de tu temporizador (oscilador) conectando en el pin 3 un Led.



R1 = R2 = 4.7 KOhms

C1 = 0.01µF, puede ser un capacitor cerámico

C2= 100 µF, a 25 Volts, preferentemente electrolítico (de botecito). Recuerda que para los capacitores electrolíticos se debe respetar su polaridad, la cual está marcada en el mismo cuerpo del capacitor. En algunos casos, el capacitor explota si no se tiene en consideración la polaridad.
Los de cerámica no tienen una polaridad definida.

Si en vez de utilizar una resistencia de un valor fijo en R2 (para nuestro caso, de 4.7 KOhms) conectas un preset vertical de engrane (resistencia variable, también conocida como potenciómetro vertical) de 4.7 KOhms, podrás cambiar la duración de tu pulso para que éste sea más rápido o más lento. El potenciómetro vertical tiene 3 pines de los cuales sólo utilizarás dos: el pin central y cualquiera de los extremos. Observa la siguiente figura.

Nota 1: *Dependiendo de la fuente de alimentación que utilices y considerando que las conexiones son correctas (existe conteo y los decodificadores están bien asignados), en ocasiones los contadores realizan conteos extraños, que por lo general se presentan "saltándose" estados. Esto se debe al ruido eléctrico causado por diferentes factores. Puedes optar por colocar un capacitor electrolítico de 330 µF a 25 Volts, entre positivo y negativo (respeta la polaridad del capacitor) en cualquier parte de la protoboard, como si se tratara de un LED con el que probarías si existe alimentación en tu tablilla.*

Nota 2: *El diseño del multivibrador astable con el 555 obedece a una relación matemática para calcular la duración del pulso de reloj combinando el valor de R1, R2 y C2. La más directa de las relaciones es f = 1 / T = 1.44/(R1 + 2R2) C. Te recomiendo profundices en el tema consultando alguna bibliografía de diseño digital o bien, a través de Internet. También existen multivibradores monoestables y biestables (flip – flops y latches).*

1. (4 puntos) Diseño de contadores utilizando operadores aritméticos.

Cuando se requiere diseñar un contador con comportamiento regular, sin cambios abruptos de estado, es posible recurrir a una descripción comportamental utilizando operadores aritméticos que permitirán incrementar o decrementar, según el caso, para que el contador funcione de manera ascendente o descendente. Observa la siguiente descripción para un contador de 4 bits, ascendente – descendente, con una variable para controlar la dirección del conteo (*dir*) y un *reset*:

```
MODULE conta
TITLE 'Este es un contador sencillo'
reloj, reset, dir pin;
q3, q2, q1, q0 pin 19, 18, 17, 16 istype 'reg';
Q = [q3, q2, q1, q0];

equations
@carry 2;
Q.clk = reloj;
when reset == 0 then Q := 0
else when dir == 0 then Q := Q + 1
else Q := Q - 1;

test_vectors
([reloj, reset, dir] -> [Q])
 [.C., 0, 0] -> [.X.];
@repeat 20 {[.C., 1, 0] -> [.X.];}; "Ascendente
@repeat 20 {[.C., 1, 1] -> [.X.];}; "Descendente
END
```

La directiva punto, *Q.clk = reloj*, nos indica que la variable *Q* de 4 bits, estará supeditada por un pulso de reloj, es decir, que sus datos de tipo registrado (istype 'reg') estarán controlados por una señal de reloj que se introduce al dispositivo a través de la variable nombrada *reloj*.
En la descripción, *Q := Q + 1*, nos indica que *Q* cambiará a *Q + 1*, después de un pulso de reloj.
La simulación para los diseños secuenciales tiene una sintaxis característica que incluye un pulso de reloj anotando *.C.*, así mismo, puedes observar que para hacer repeticiones es posible utilizar la directiva *@repeat* (número de repeticiones).

1.a Para cumplir con este numeral de la práctica, incluye una variable de *stop* que obligue al contador a Q := Q cuando ésta se active. Realiza la simulación e incluye una pantalla de ésta.

1.b Programa, el contador diseñado en ***1.a***, en un GAL22V10. Diseña un decodificador hexadecimal (*0 hasta F*) para un display a siete segmentos y prográmalo en un GAL22V10. Para validar este numeral de la práctica deberás mostrar funcionando tu contador ascendente-descendente conectado al decodificador hexadecimal y al display a siete segmentos.

2. (6 puntos) Diseño de contadores utilizando tablas de transiciones (tablas de verdad).

Cuando se requiere diseñar un contador con un comportamiento irregular, que presenta cambios abruptos de estado y que estos cambios no pueden ser atendidos con un incremento o decremento fijo a través de un operador aritmético, es posible recurrir a una descripción comportamental utilizando una tabla de verdad o tabla de transiciones. Observa la siguiente descripción. Se trata de un contador ascendente de 4 bits, que incluye una salida adicional (*primo*) que es combinatoria y que permite la detección de estados primos.

```
MODULE contador_tt
"entradas
 reloj, reset pin 1, 2;
"Salidas
 Q3,Q2,Q1,Q0 pin 19,18,17,16 istype 'reg';
 primo pin 15 istype 'com';
 Y =[Q3,Q2,Q1,Q0];

equations
Y.clk = reloj;
Y.ar = reset;

truth_table
([Y]:>[Y]->primo)
[0]:>[1]->0;
[1]:>[2]->1;
[2]:>[3]->1;
[3]:>[4]->1;
[4]:>[5]->0;
[5]:>[6]->1;
[6]:>[7]->0;
[7]:>[8]->1;
[8]:>[9]->0;
[9]:>[10]->0;
[10]:>[11]->0;
[11]:>[12]->1;
[12]:>[13]->0;
[13]:>[14]->1;
[14]:>[15]->0;
[15]:>[0]->0;

test_vectors
([reloj, reset]->[Y, primo])
[.C., 1]->[.X., .X.];
@repeat 20 {[.C., 0] -> [.X., .X.];};
END
```

Si fuera el caso que no requieres más que el contador, es decir, sin salidas combinatorias, sólo las registradas, puedes utilizar la tabla como se muestra a continuación con el ejemplo del diseño de un contador ascendente de números primos de 4 bits. Observa que se requiere el estado *0* para considerar un posible arranque a través de un *reset*, sin embargo, ya estando en funcionamiento el contador no debe pasar por el *0* pues éste no es primo, en otras palabras y refiriéndonos al último estado de la tabla de transición, podrás advertir que del estado *13* pasaremos directamente al estado *1* para cumplir el conteo de sólo primos de 4 bits.

```
MODULE contador_primo
"entradas
 reloj, reset pin 1, 2;
"Salidas
 Q3,Q2,Q1,Q0 pin 19,18,17,16 istype 'reg';
 Y =[Q3,Q2,Q1,Q0];

equations
Y.clk = reloj;
Y.ar = reset;

truth_table
([Y]:>[Y])
[0]:>[1];
[1]:>[2];
[2]:>[3];
```

```
[3]:>[5];
[5]:>[7];
[7]:>[11];
[11]:>[13];
[13]:>[1];

test_vectors
([reloj, reset]->[Y])
[.C., 1]->[.X.];
@repeat 20 {[.C., 0] -> [.X.];};
END
```

2.a Esta descripción es muy útil cuando se desea conectar contadores en cascada, debido a que la salida combinatoria la puedes utilizar como señal de reloj para otro módulo contador. Diseña un contador ascendente 00-99, utilizando dos contadores conectados en cascada, un contador para las unidades y otro para las decenas. Recuerda que el contador de unidades debe proporcionar el pulso de reloj para el de las decenas. Es claro que el reloj proveniente del c. i. 555 debe estar conectado al contador de unidades. Incluye la simulación de tu diseño (para cada módulo contador) e implementa el contador 00-99, fisicamente en dos dispositivos GAL22V10. Opcionalmente puedes conectar dos decodificadores BCD (0-9), programados en otros dos dispositivos GAL, para poder visualizar los estados en dos displays a siete segmentos.

2.b Seguramente te habrás dado cuenta que en ocasiones es posible utilizar de mejor forma los términos producto que se implementarán dentro del GAL. El siguiente código infiere un contador con tabla de transición pero que sus salidas se dirigen a un decodificador para un display a siete segmentos. Tanto el contador como el display estarán en el mismo dispositivo GAL.
¿El GAL22V10 tendrá suficiente capacidad para soportar un contador que incluya su propio decodificador de forma embebida? Completa la tabla de verdad del diseño siguiente para que el contador sea (0 - 7). ¿Cómo incluirías una variable para un *stop* que implique que el estado siguiente se mantenga? Simula y muestra funcionando fisicamente tu diseño.

```
MODULE contt_display
"entradas
Reloj, reset pin 1, 2;
"Salidas
Q2,Q1,Q0 node istype 'reg';
a, b, c, d, e, f, g pin istype 'com';
Y =[Q2,Q1,Q0];
Display=[a, b, c, d, e, f, g];

equations
Y.clk = reloj;
Y.ar = reset;

truth_table
([Y]:>[Y]->Display)
[0]:>[1]->254;
[1]:>[2]->48;
completar tabla
[7]:>[0]->113;

test_vectors
([reloj, reset]->[Display])
[.C., 1]->[.X.];
@repeat 20 {[.C., 0] -> [.X.];};
END
```

Campo 4: **Conclusiones individuales.**