Correcto uso de Reset en FPGAs y su Codificación en VHDL

Nota Técnica 10

Cristian Sisterna

Introducción

En esta nota técnica se describirán con bastante detalle los distintos tipos de reset que se pueden usar en un sistema digital, sus ventajas y desventajas, y cual de ellos es el más aconsejable a usar para tener un sistema más confiable.

A pesar de que el reset de un sistema es un tema crítico, pocas veces se le da la importancia que tiene y es usualmente uno de los aspectos ignorados en un diseño con FPGA. Un circuito de reset que no se comporte correctamente resulta directamente en un mal funcionamiento aleatorio del sistema. Y como ya sabemos, los problemas aleatorios, no repetitivos, de un sistema son los más difíciles de depurar.

Un diseño puede tener reset sincrónico o asincrónico. Normalmente la señal de reset es generada externamente al FPGA, por lo que es una señal asincrónica. Básicamente la señal de reset es una entrada al sistema que posibilita la correcta inicialización del mismo. Como así también, pueda forzar al sistema a ese estado inicial cuando haga falta (por ejemplo, cuando el sistema se 'cuelga', o se va de su normal funcionamiento). Esta señal asincrónica puede sincronizarse a través de un circuito sincronizador, para de este modo crear una señal de reset sin glitches y por lo menos de un ciclo de reloj de ancho. Sin embargo, como se detallará en los próximos puntos el reset totalmente sincrónico puede ocasionar alguna fallas aleatorias, por lo que se propone otro circuito a fin de hacer el sistema más confiable.

Hardware del Reset

Como ya se dijo anteriormente el reset de un sistema no algo a lo que se le presta mucha atención ni se le dedica un tiempo de estudio para ver cual sería la mejor manera de resetear el sistema, así por lo general en un board de un sistema más o menos complejo encontramos solo un pulsador (tipo push-on) para llevar a cabo la tarea de resetear el sistema. Este pulsador puede ser de utilidad una vez que el sistema está andando y que por una razón u otra se fue de secuencia. Presionando el pulsador se resetea el sistema al obligarlo a ir la inicialización del mismo. Pero que ocurre por ejemplo durante el tiempo en que el voltaje de alimentación al FPGA se establece y hasta que se presiona el pulsador de generación de reset?. Teniendo sobre todo en cuenta que por mas rápido se presione el pulsador pueden haber pasado varios, muchos, suficientes como para hacer un desastre, ciclos de reloj. Otro punto también en que el pulsador no es la solución adecuada, se ve reflejada en situaciones en que por ejemplo por alguna razón el voltaje del core del FPGA bajo por unas milésimas de segundos pero suficiente para resetear al FPGA; pero la lógica del FPGA volvió a su estado inicial?. Es un riesgo que no

debería ser asumido en caso de querer diseñar un sistema confiable. Por los problemas mencionados a causa de usar un simple pulsador, es que se necesita una lógica más elaborada que soluciones los problemas detallados y generar un reset más confiable a partir no solo de la señal del pulsador sino también de las variaciones de la tensión de alimentación, sobre todo durante la inicialización del sistema (power-up). Esta lógica se encuentra encapsulada en un circuito integrado normalmente llamado controlador de reset o supervisor de voltaje de alimentación.

Controlador de Reset

Un controlador de reset está diseñado exclusivamente para ser usado como generador de reset en sistemas basados en microcontroladores, microprocesadores, FPGAs, etc. Un ejemplo de este controlado es el TL 77xx de Texas Instruments o el ICL6455 de Intersil. El funcionamiento de estos controladores se basa en estar permanentemente monitoreando ya sea la entrada de reset al mismo (proveniente de un pulsador por ejemplo) y monitoreando los niveles de voltaje de interés, por ejemplo el voltaje de alimentación del core del FPGA, a través de un pin dedicado a tal efecto. Durante el proceso de inicialización o power-up, es decir mientras la tensión de alimentación no se estabilice en el valor especifico, el controlador de reset mantiene activa la salida de reset (el TL 77xx provee dos salidas de reset una activa en bajo y otra activa en alto) hasta que el voltaje de alimentación haya alcanzado un cierto nivel de disparo (threshold) de la tensión especifica. Recordar que usando una escala de tiempo del orden de los milisegundos, la tensión de alimentación es una rampa inclinada cuya pendiente depende de la carga y de la fuente de alimentación, pero en todos los casos es una rampa, con mas o menos pendiente. Así, durante el crecimiento de los valores de la tensión de alimentación, una vez alcanzado ese nivel de disparo, el controlador mantiene la salida reset en su estado activo por un cierto tiempo programable por el valor de un capacitor que se añade al circuito.

Este controlador se comporta del mismo modo durante el proceso de apagado del sistema (power-down) o durante un proceso de caída temporario de voltaje (Brown-out), en ambas situaciones el nivel de disparo es el mismo que durante la inicialización, activando la salida reset cuando el voltaje cae por debajo del nivel de disparo.

Otra característica de un controlador de este tipo es que permite la entrada de un pulsador de reset para, de este modo, activar la salida reset del mismo. Asi, si en cualquier momento se desea resetear el FPGA usando el pulsador, se puede hacer sin ningún problema. En este caso, también el reset es mantenido activo por la constante de tiempo fijada por el capacitor usado en la entrada destinada a tal fin.

Y una característica más de estos controladores es el hecho que pueden colocarse en el board tantos controladores de reset como tensiones de alimentación se quieran monitorear. Asi, por ejemplo en un FPGA donde se tienen diferentes tensiones, todas se pueden monitorear al usar un controlador por cada

tensión, y se genera un solo reset. De este modo se pueden monitorear tantas tensiones de alimentación como se deseen.

Figura 1 muestra la implementación de un sistema con un FPGA el cual tiene un controlador de reset en el board, con entrada también del pulsador. Dentro del FPGA la líneas de conexión del reloj no se han dibujado todas para evitar perder de vista el propósito de la figura, pero en realidad todas la entradas de reloj de los flip-flops van conectadas a la señal de reloj.

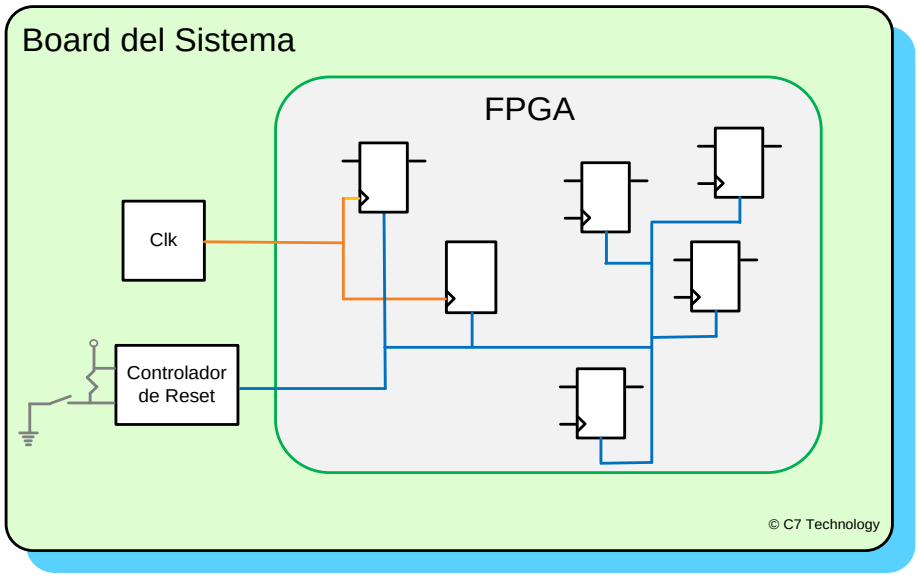

Figura 1- Esquema general de un sistema con controlador de reset y pulsador generador de reset

Como puede verse en Figura 1, la salida de reset del controlador, NO tiene ninguna relación con el reloj del sistema. Este asincronismo entre el reset y el reloj puede ocasionar problemas ya que existen parámetros de tiempo, no muy conocidos pero realmente importantes, que deben ser tenidos en cuenta para una señal como reset, para evitar llevar la salida del flip-flop a un valor metaestable. Estos parámetros de tiempo son conocidos como tiempo de recuperación y tiempo de remoción, detallándose los mismos a continuación.

Tiempo de Recuperación y Tiempo de Remoción de Entradas Asincrónicas

Los flip-flops tienen dos parámetros de tiempo relacionados con las entradas asincrónicas reset/preset, el tiempo de recuperación (recovery time) y el tiempo de remoción (removal time).

El tiempo de recuperación específica el *tiempo mínimo* entre la desactivación del reset y el próximo flanco activo del reloj. Durante ese tiempo la señal de reset debe permanecer estable para que el flip-flop no entre en metaestabilidad.

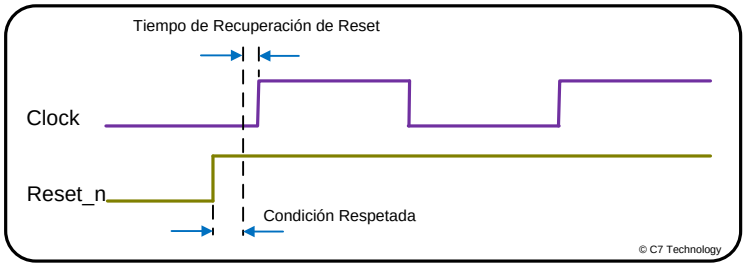

Figura 2 - Tiempo de recuperación del reset de un flip-flop

El tiempo de remoción especifica el tiempo mínimo que la señal de reset debe permanecer estable antes de desactivarse y después del flanco activo del reloj. Sí este tiempo es violado la salida del flip-flop es incierta, con una gran probabilidad de que vaya a un estado metaestable.

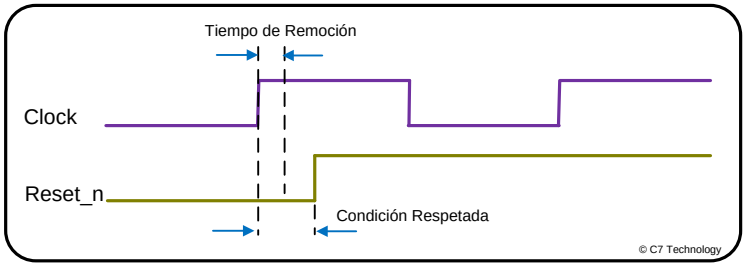

Figura 3 - Tiempo de remoción del reset de un flip-flop

Como se sabe el hecho que la salida de un flip-flop entre en metaestabilidad es una probabilidad. La mayoría de las veces la desactivación de la senal de reset no provocará problemas de funcionamiento en el flip-flop, tal como se muestra en Figuras 2 y Figura 3, en las cuales la desactivación asincrónica del reset se produce o mucho antes del tiempo de recuperación (Figura 2), o bastante después (Figura 3) por lo que no provocará ningún problema de funcionamiento. Sin embargo, y dado que la señal de reset es justamente una entrada asincrónica, el flanco de desactivación de reset puede quedar muy cerca del flanco activo del reloj provocando una violación de uno de los dos tiempos mencionados, provocando que la salida del flip-flop vaya a un estada metaestable, tal como se muestra en Figura 3.

Existen herramientas de los fabricantes de FPGA que pueden detectar violaciones de tiempo de recuperación y tiempo de remoción. Así por ejemplo la herramienta *TimeQuest Timing Analyzer de Altera*, lleva a cabo un análisis de los tiempos de recuperación y remoción.

Debido a que este tipo de fallas, que un flip-flop entre en metaestabilidad es una falla del sistema, se producen aleatoriamente en el tiempo es que son muy difícil de depurar (debug) por lo que el diseñador del sistema debe llevar a cabo el diseño de modo de evitar fallas aleatorias.

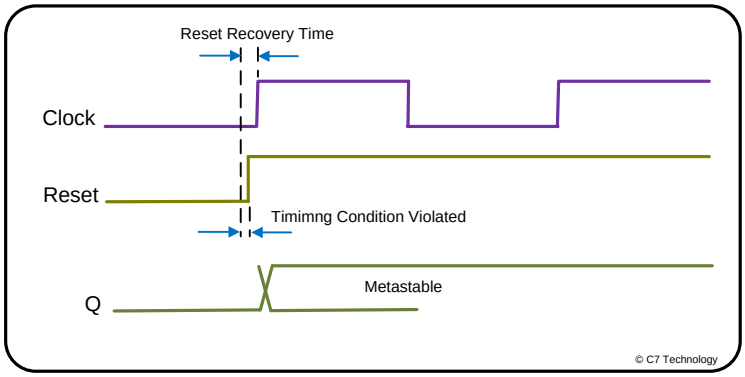

Figura 4 - Violación de tiempo de recuperación

Ahora bien, cómo se puede evitar este problema?, que consideraciones debe tener el diseñador a fin de que la desactivación de la señal de reset no provoque que el flip-flop o mejor dicho los flip-flops no entren en metaestabilidad. Los distintos esquemas de usar el reset en un sistema digital son descriptos a continuación.

Reset Totalmente Asincrónico

Cuando la señal que resetea el sistema se activa y desactiva asincrónicamente; es decir no tiene relación alguna con el reloj del flip-flop que está reseteando, se dice que es un reset totalmente asincrónico. Un ejemplo de este tipo de esquema y su implementación se detalla en Figura 1.

El código VHDL de un reset asincrónico es el siguiente:

```
library ieee;
use ieee.std_logic_1164.all;

entity a_rst is
port(
      d   : in  std_logic;
      clk: in  std_logic;
      rst: in  std_logic;
      q   : out std_logic
    );
end entity
```

```
architecture beh of a_rst is
begin
a_rst_proc: process(clk, rst)
begin
      if (rst= '1') then
            q <= '0';
      elsif(rising_edge(cLk)) then
            q <= d;
      end if;
end process a_rst_proc;
```

Aunque este sea el esquema más comúnmente utilizado, enseñado, y que funciona correctamente la mayoría de las veces, *no es en realidad el más aconsejado*. Pues, como ya se explicó anteriormente y se representó en Figura 4, puede ocurrir que el flanco de desactivación de reset esté muy cerca del flanco del reloj, violando el tiempo de recuperación o de remoción del flip-flop.

Reset Totalmente Sincrónico

La primera solución que se le ocurre a uno luego de leer el punto anterior es hacer el reset totalmente sincrónico. Es decir sincronizar la entrada asincrónica de reset para transformarla en un reset sincrónico con el reloj del sistema. Este esquema es el detallado en Figura 5.

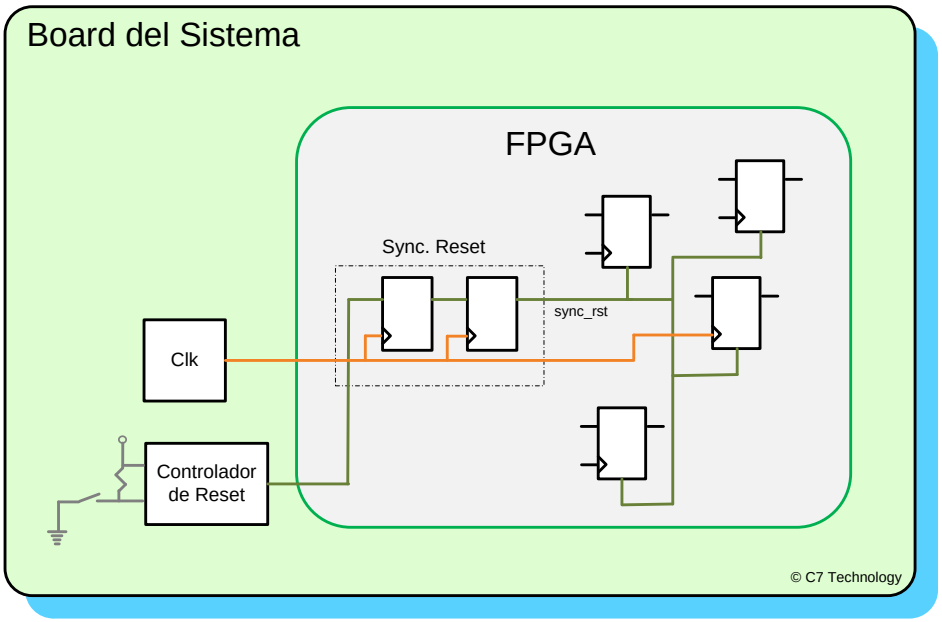

Figura 5 - Esquema de un reset sincrónico

El código VHDL que describe el sincronizador de la señal reset, que se muestra dentro del rectángulo de línea de puntos en Figura 5, es el siguiente:

```
library ieee;
use ieee.std_logic_1164.all;

entity sync_rst is
port(
      sys_clk : in  std_logic;
      sys_rst : in  std_logic;
      sync_rst: out std_logic
      );
end entity;

architecture beh of sync_rst is
signal rst1: std_logic;
begin
sync_rst_proc: process(sys_clk)
begin
      if (rising_edge(sys_clk)) then
            rst1     <= sys_rst;
            sync_rst <= rst1;
      end if;
end process sync_rst_proc;
end beh;
```

La siguiente figura representa la implementación del código escrito arriba.

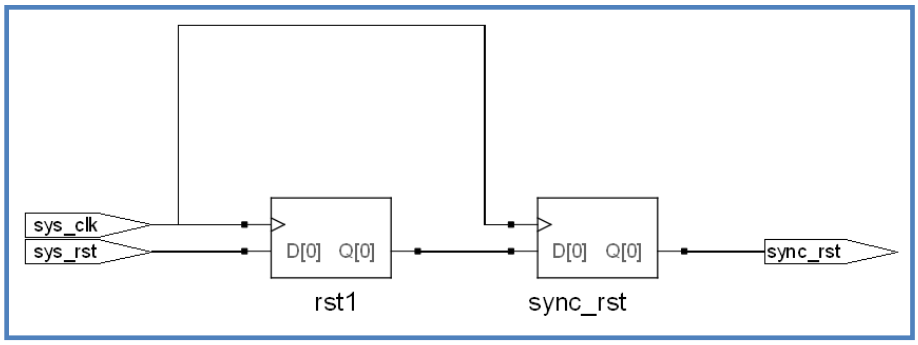

Figura 6 - Implementación de reset con activación y desactivación sincrónica

La ventaja de este tipo de reset es que el reset llega a todos los flip-flops sincrónicamente con el reloj de los flip-flops, asi NUNCA provocará una violación de tiempo de recuperación de reset del flip-flop (considerando que la señal de reset ha sido apropiadamente conducida a través de buffers para mejorar el fan-out de la misma). Sin embargo, este esquema presenta un problema distinto al explicado en el caso anterior. El problema que puede llegar a suceder está relacionado con la activación del reset, no con la desactivación del mismo. Supongamos un sistema con un reloj de alta frecuencia y otro de baja frecuencia. Para que el reset trabaje correctamente se debería sincronizar en ambas frecuencias, tal como muestra la siguiente figura.

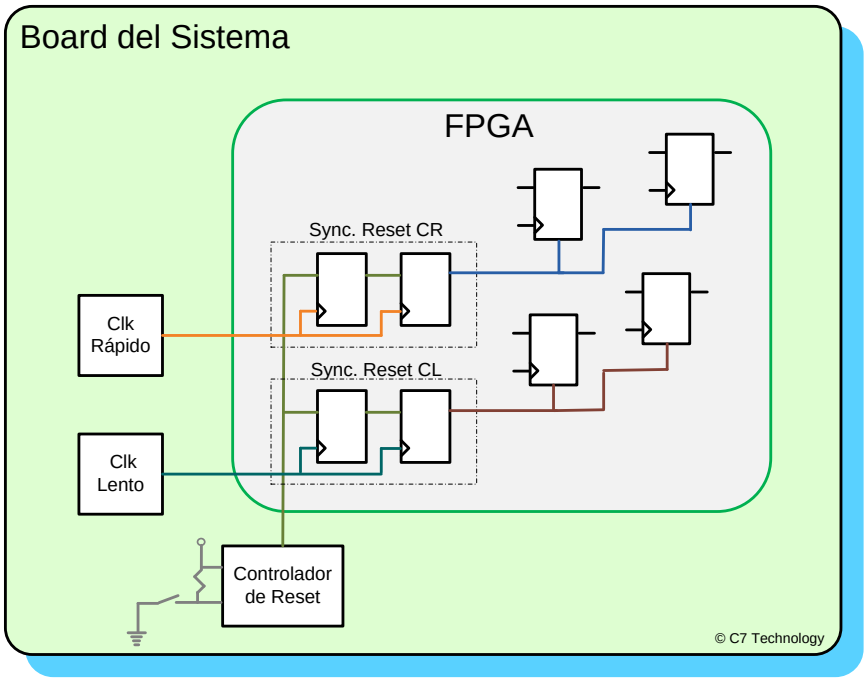

Figura 7 - Esquema de reset sincrónico con dos frecuencias distintas

El problema de este esquema es que puede suceder que la señal de reset NO sea capturada en el dominio del reloj más lento, debido a la probabilidad de que cuando el reset se activa o incluso durante todo el tiempo de activación, el flanco activo del reloj más lento nunca 'llega'. Así, el reset puede no ser capturado y los flip-flops del sistema controlados por este reset *no son nunca* reseteados.

Las siguientes formas de ondas detallan el comportamiento de la señal de reset en el dominio de alta frecuencia y de baja frecuencia. Notando la falla de activación de reset en el dominio de baja frecuencia.

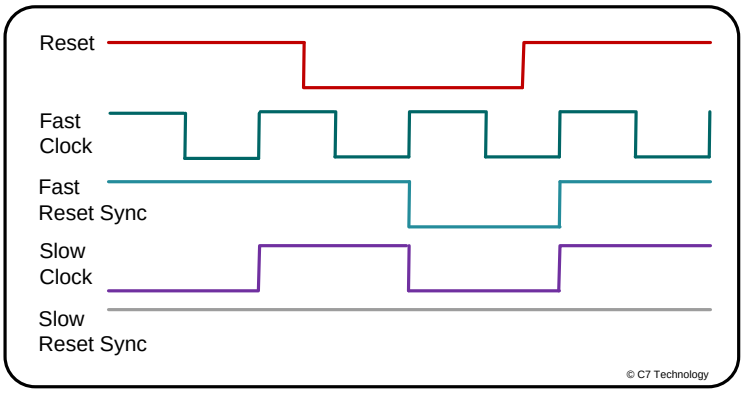

Figura 8 - Formas de ondas del esquema de reset con distintas frecuencias

Por lo que se demuestra en Figura 8, reset totalmente sincrónico es recomendado *siempre que la captura del reset pueda ser de algún modo garantizado por el diseño.*

Un punto importante a tener en cuenta con este esquema está relacionado con el hecho de que el reloj o los relojes deben ser osciladores que funcionen desde el momento en que se alimente el board, por lo que en los sistemas cuyos relojes son generados desde un DLL o PLL y estos mismos relojes son usados para sincronizar el reset, la señal de reset será retardada los dos ciclos de reloj de los sincronizadores más el tiempo que tarde el PLL o DLL en estabilizarse (lock time). Figura 9 muestra un esquema de este caso.

Debido al tiempo de retardo que sufre la señal de reset para llegar a los flip-flips, mientras a su vez el sistema ya ha estado en 'funcionamiento', puede suceder por ejemplo que una máquina de estado no vaya al estado inicial hasta que haya pasado este tiempo, tiempo durante el cual la máquina de estado se haya quedado en un estado inválido o con valores de las salidas que pongan en peligro la confiabilidad del sistema. Por lo que este esquema se desaconseja completamente cuando los relojes son generados por componentes tipo PLL o DLL.

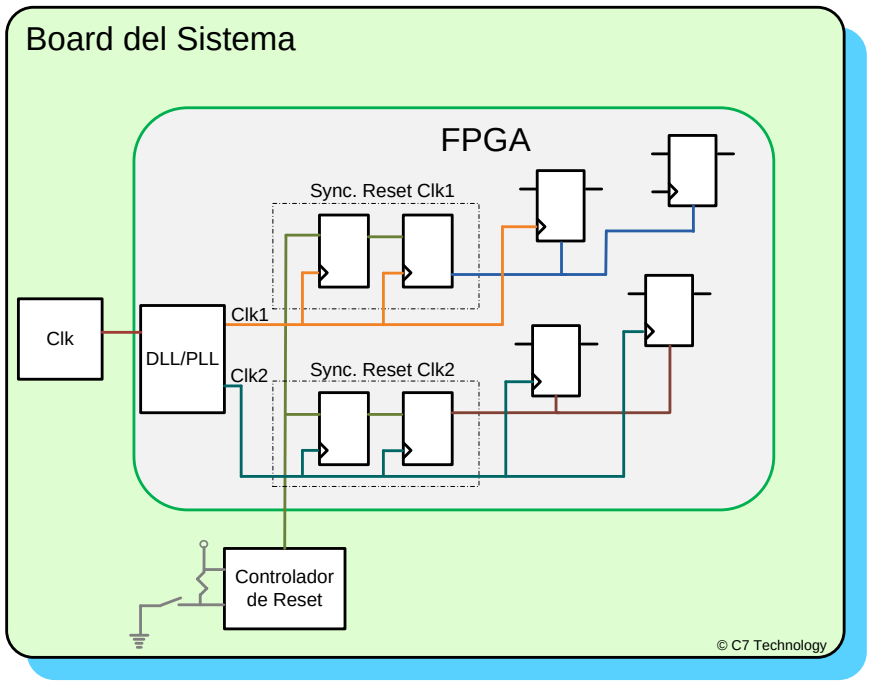

Figura 9 - Reset totalmente sincrónico con relojes generados por DLL o PLL

Una combinación de los esquemas de reset detallados en los puntos anteriores, provee una solución más confiable tal como se detalla a continuación.

Reset Asincrónicamente Activado, Sincrónicamente Desactivado

La mejor solución a los problemas planteados es activar el reset asincrónicamente y desactivarlo sincrónicamente, tal como se muestra en Figura 10.

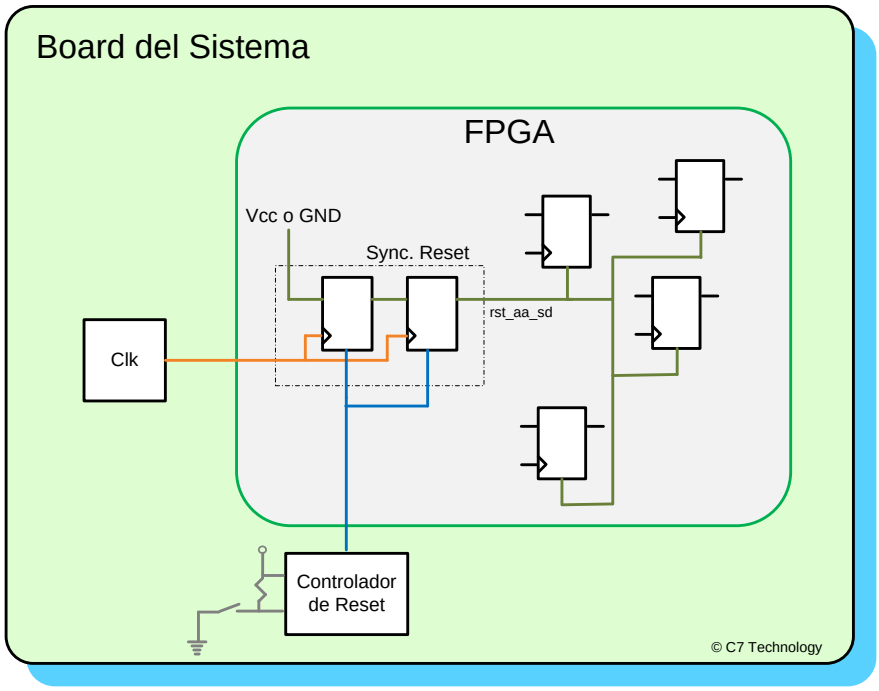

Figura 10 - Esquema de reset activado asincrónicamente y desactivado sincrónicamente

Las siguientes líneas de código describen el circuito encerrado en el cuadrado de línea de puntos en Figura 10.

```
library ieee;
use ieee.std_logic_1164.all;

entity rst_async_ass_synch_deass is
generic(rst_width:integer:=2;
        rst_active_value: std_logic:='0');
port(
    sys_clk  : in std_logic;
    sys_rst  : in std_logic;
    rst_aa_sd: out std_logic);
end entity rst_async_ass_synch_deass;
```

```
architecture rst_async_ass_synch_deass_beh of rst_async_ass_synch_deass is

signal sys_rst_i: std_logic_vector(rst_width downto 0);

begin

asyn_ass: process(sys_rst, sys_clk)
begin
  if (sys_rst= rst_active_value) then
    sys_rst_i <= (others => rst_active_value);
    rst_aa_sd <= '0';
  elsif (rising_edge(sys_clk)) then
      sys_rst_i(0) <= not rst_active_value;
      for i in 0 to rst_width-1 loop
      sys_rst_i(i+1) <= sys_rst_i(i);
      end loop;
  end if;
  rst_aa_sd <= sys_rst_i(rst_width-1);
end process asyn_ass;
end rst_async_ass_synch_deass_beh;
```

La siguiente figura muestra la implementación del código VHDL del esquema de reset descripto.

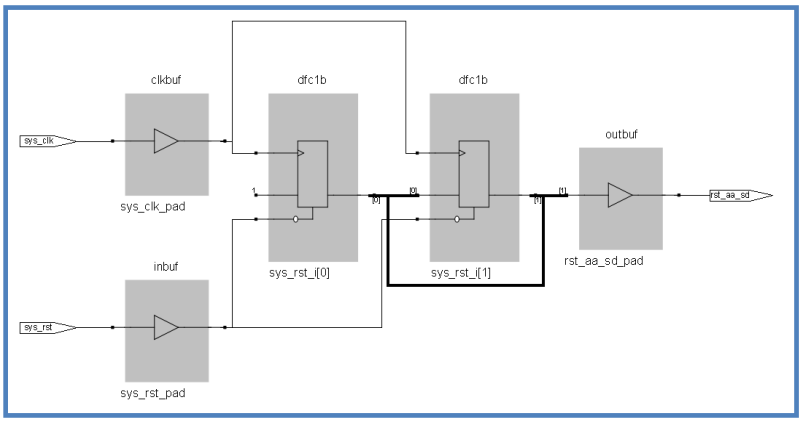

Figura 11 - Implementación de reset asincrónicamente activado, sincrónicamente desactivado

De este modo los flip-flops controlados con la salida de este circuito son asincrónicamente reseteados y salen sincrónicamente del estado de reset. Una gran ventaja del reset asincrónico es que no necesita que el reloj este corriendo para resetear los flip-flops, de este modo por ejemplo se puede tener el sistema bajo reset hasta que el DLL o PLL se estabilice.

Comportamiento de los Flip-Flops de un FPGA durante Power-Up

Es sabido que un FPGA tiene distintas tensiones de alimentación. También es sabido, o lo van a saber ahora, que un FPGA tiene un circuito interno que monitorea las distintas tensiones de alimentación durante el proceso de encendido de las alimentaciones del FPGA, proceso conocido como power-up. Este circuito interno genera una señal de reset interna del FPGA que es ruteada a *todos* los flip-flops del FPGA para inicializarlos, este senal de reset es comúnmente conocida como reset durante inicialización o power-on reset (POR). Así, todos los flip-flops son asincrónicamente reseteados, es decir las salidas son forzadas a un valor lógico. Qué valor lógico, '0' o '1'?. El valor lógico de una salida durante el proceso de inicialización es el mismo que el valor lógico que se le asigna en el código VHDL a esa salida cuando se activa el reset. Básicamente la salida de un FPGA toma los siguientes valores durante el proceso de inicialización:

1- Un estado de alta impedancia hasta que el reset interno genere la señal de reset interna una vez que las tensiones de alimentación han alcanzado un valor determinado
2- Los flip-flops son reseteados asincrónicamente forzando las salidas al valor asignado en el código. Sin embargo este valor puede ser cambiado tal como se explica más abajo
3- Cuando se activa el reset generado por el controlador externo de reset, el flip-flop tomará el valor asignado en el código VHDL para el caso en que se active la señal reset

Explicaremos ahora lo escrito en el punto 2. Los flip-flops de un FPGA pueden ser forzados durante el proceso de inicialización a un valor independientemente del valor que vaya tomar cuando se active la señal de reset del flip-flop (punto 3 detallado arriba). Esto se detalla en la siguiente figura.

```
architecture beh of test_srst is
 signal q_i: std_logic:= '0';
begin
 process (clk)
 begin
  if (rising_edge(clk)) then
     if rst='1' then
          q_i <= '1';
     else
        q_i <= d;
     end if;
  end if;
 end process;
 q<=q_i;
end beh;
```

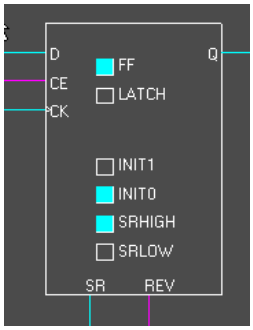

Figura 12 - Valor de inicialización de un flip-flop distinto al valor de reset

Primero que todo, este es uno de los muy pocos casos en que un valor inicial de una señal tiene un hardware asociado. Explico ahora la figura, lo que se ve del lado derecho es el flip-flop en que es implementado el código de la parte izquierda. El flip-flip permite un valor de inicialización que puede ser '0', INIT0, o '1', INIT1. Este es el valor que el sintetizador el asigna de acuerdo al valor inicial asignado a

la señal durante su declaración. En este caso se le asigna un valir inicial de '0', y es por ello que vemos que el valor inicial del flip-flop este marcado como INIT0. Por otro lado la señal de reset resetea el flip-flop a un valor '1' de acuerdo a lo escrito en el código. Por ello, el valor del flip-flop durante el proceso de Set/Reset es '1', SRHIGH. Difícilmente se necesite hacer algo similar a lo descripto, pues en este caso la salida, según los puntos detallados anteriormente, iría tomando los siguientes valores en el tiempo: 'Z', '0' y '1'.

Consideraciones de Implementación

Los circuitos de generación de reset propuestos generan la señal de reset internamente, por ejemplo la salida de los flip-flops de sincronización, y debido a que ésta es una señal que se usa para controlar todos o casi todos los flip-flops usados en el sistema basado en el FPGA, la señal de reset tiene una gran cargabilidad. Por ello hay que revisar el informa de la herramienta de colocación y ruteo (place and route) a fin de conocer la carga (fanout) de la señal reset, y el ruteo de la misma. De modo que si el fanout es un número mayor de 250-300, y no se está utilizando un ruteo de bajo retardo y alta cargabilidad, es aconsejable forzar a la herramienta de place and route a usar un componente que garantice la integridad de la señal reset. En caso de los FPGAs de Xilinx se puede usar la restricción (constraint) BUFG, para Altera usar GLOBAL, para que la señal interna reset sea ruteada usando ese tipo de buffer, que a su vez tiene asociado un ruteo dedicado de bajo skew y gran cargabilidad.

Otro punto de implementación a considerar es el hecho de realizar el diseño de modo de tener un sincronizador de reset por dominio de reloj. Tal como se explicó anteriormente es necesario que la desactivación sincrónica del reset sea sincrónica con el reloj del flip-flop que está reseteando, para evitar los problemas antes mencionados. Por ello si se tienen distintos relojes, por cada reloj debe haber un sincronizador de reset para ese dominio. Figura 13 detalla un esquema genérico con tres relojes diferentes, incluso uno generado por un bloque DLL o PLL. Convenientemente, para hacer el sistema más confiable, la señal de reset debería ser generada por el controlador de reset, sino se dispone del controlador se puede generar el reset usando un simple pulsador. En este último caso, no será posible hacer pasar la salida del pulsador por un sistema de antirebotes, porque justamente se desea que la activación del reset sea asincrónica, y casi todos los sistemas antirebotes funcionan en base a un reloj.

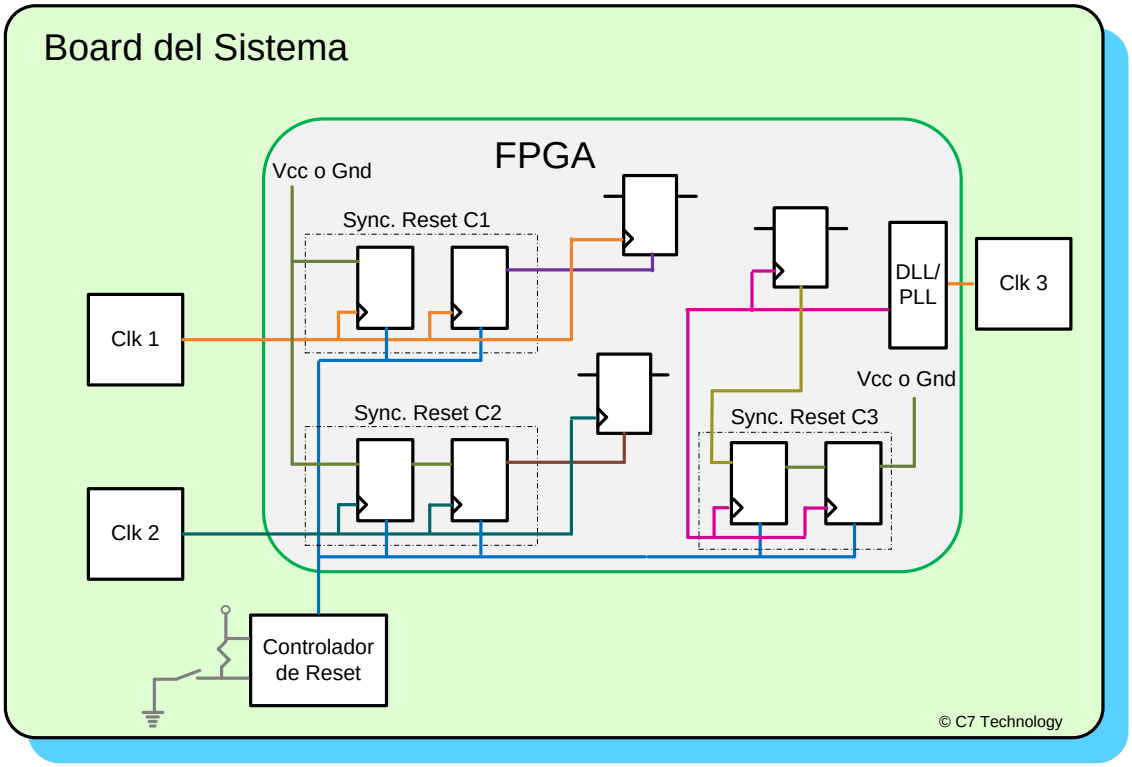

Figura 13 - Esquema de reset de mayor confiabilidad

Conclusión

Como se dijo al comienzo de este artículo, el reset de un sistema es algo que no es muy tenido en cuenta, sin embargo como se puede apreciar con lo descripto en estas páginas, es necesario tener muy en cuenta como se genera el reset y la manera de usarlo dentro del FPGA para aumentar la confiabilidad del sistema.

Desde el punto de vista práctico, el método de activación de reset asincrónicamente y su desactivación sincrónicamente es el más aconsejado.

C7 Technology www.c7t-hdl.com
Copyright © 2012.
All rights reserved.

Este documente puede ser copiado y distribuido libremente.
Si es copiado por favor mantenga el logo e información de la compañía.
Si parte del documento es cortado y pegado, por favor mencione C7 Technology como fuente de información. Gracias.