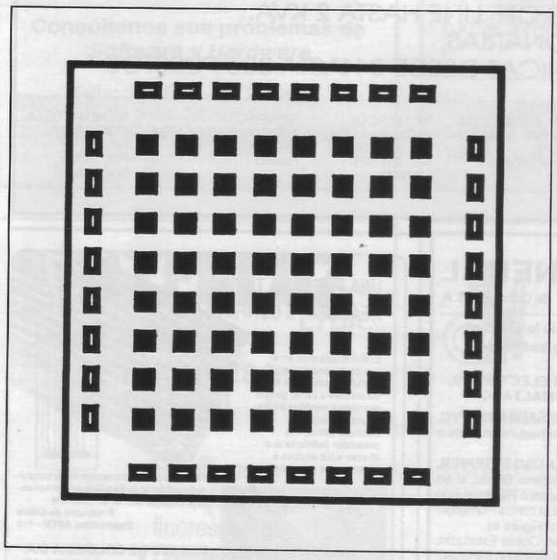
Generador de funciones usando los nuevos dispositivos lógicos programables



Angel García Baños
Ingeniero de Telecomunicación
Profesor Instructor
Departamento de Electricidad
Universidad del Valle

RESUMEN

Se introduce un nuevo concepto en hardware digital: las matrices lógicas con interconexiones programables por el usuario (FPGA). Se explica para qué sirven, cuáles son sus características, y sus ventajas sobre los chips convencionales. Además, se muestra una aplicación realizada en España por el autor del artículo, consistente en un generador de funciones programable.

ABSTRACT

A new concept on digital hardware is presented: field programmable gate arrays (FPGA). Uses, characteristics and advantages over conventional chips are explained. Furthermore an application using this device is presented: a programmable function generator designed by the author of this paper at Spain.

INTRODUCCION A LAS FPGA

Tradicionalmente, cualquier diseño realizado en el área de la lógica digital, se implementaba en la práctica a través de circuitos SSI y MSI (baja y media escala de integración), como por ejemplo, los de la familia lógica TTL (74xxx).

Conforme aumentó la complejidad de los diseños, el número de circuitos fue creciendo, y con ello el área de circuito impreso en donde montarlos.

Piénsese que, incluso en sistemas basados en microprocesador, la lógica de interconexión (en inglés, "glue logic") de éste con sus periféricos se ha hecho más compleja, día a día.

Para contrarrestar esta tendencia indeseable del aumento del tamaño de los equipos, surgen 2 nuevas tecnologías: los encapsulados de menor tamaño (montaje de superficie, pin grid array, etcétera),

que no son el tema de este artículo, y los nuevos dispositivos de puertas lógicas programables.
Estos últimos dispositivos albergan en su interior muchas puertas e incluso flip-flops, que pueden encontrarse bien aisladas o bien previamente interconectadas según ciertos patrones (lo cual determina la arquitectura interna del circuito).
Además existe la posibilidad de interconectar unas puertas con otras a voluntad del usuario.
Dependiendo de la arquitectura interna y de la manera de programar las conexiones, se distinguen varias tecnologías, de las cuales cabe destacar:

• Memorias ROM, PROM, EPROM, EEPROM: Son dispositivos sobradamente conocidos en su aplicación habitual, es decir, para guardar información de programas y datos. Pero también sirven para realizar funciones lógicas combinacionales. Para ello, simplemente se graba en su interior las tablas de verdad de las funciones a realizar. Como entradas se usan las líneas del bus de direcciones y como salidas las del bus de datos.
En las memorias ROM la información no la puede grabar el usuario, sino que éste debe remitirla al fabricante del dispositivo para que la incorpore durante el proceso de fabricación.
Las memorias PROM se basan en unos fusibles internos, que pueden ser quemados selectivamente por el usuario final para grabar así la información deseada. Obviamente, un fusible quemado no se puede regenerar, por lo que no se pueden reprogramar.
Las memorias EPROM si pueden ser reprogramadas, pues en vez de fusibles tienen unas estructuras flotantes aisladas donde la información se graba, almacenando carga eléctrica en ellas. Para borrarlas se las irradia con luz ultravioleta.
Las memorias EEPROM son similares a las anteriores, pero para borrarlas se emplea un voltaje especial, que hace que las cargas atrapadas salten el aislante por efecto túnel.

• PAL: Es una estructura formada por un plano de puertas AND y otro plano de puertas OR, configuradas de manera que se puedan realizar varias funciones lógicas, expresadas habitualmente como suma de productos. Las conexiones de las entradas a las puertas AND son programables por el usuario, mientras que las conexiones de las salidas de las AND a las entradas de las OR están prefijadas. Las salidas de las puertas OR son las salidas del dispositivo. Y las tecnologías empleadas en la fabricación de estos dispositivos son las mismas que en la memorias vistas en el párrafo anterior, dando lugar a equivalentes formas de realizar la programación y el borrado.

• Gate arrays: Consisten internamente de una gran cantidad de puertas cuyas entradas y salidas pueden ser conectadas según especifique el usuario. Sin embargo, de manera similar a las memorias ROM, el usuario, por sí mismo, no puede realizar la programación de las interconexiones, sino que debe solicitar esa labor al propio fabricante del dispositivo. Obviamente, ésto es sólo rentable para grandes series de fabricación, pues los costos fijos son muy altos. Además, cualquier error (o deseo de modificación) en el diseño de las conexiones internas es irreversible, una vez fabricado el circuito.

• FPGA: Su nombre, en inglés, es "Field Programmable Gate Array" (1,2,3). Hay dos variantes: la primera es básicamente similar a un "Gate Array" explicado en el párrafo anterior, pero donde las conexiones internas se realizan por medio de transistores trabajando como interruptores. Cada interruptor puede estar abierto o cerrado, dependiendo del estado (0 o 1) de un bit de programación, almacenado en una memoria RAM interna al dispositivo. Por tanto, la programación de las conexiones se hace de manera similar a como se escribe información en memorias RAM. Además es volátil, es decir, la programación se pierde al faltar la tensión de alimentación.
Como ventaja, al igual que las memorias RAM, estos dispositivos se pueden reprogramar tantas veces como se desee.
La segunda variante de este tipo de dispositivos consiste internamente de una matriz de bloques lógicos (Logic Cell Array), conformado cada uno de ellos básicamente por funciones combinacionales y flip-flops, cuyo comportamiento puede programarse, así como su interconexión con los demás bloques. La información que gobierna la configuración interna deseada por el usuario, se programa también en una memoria RAM interna al dispositivo.

Hay que mencionar que existen otras variantes de FPGA, donde la información del conexionado se graba en una memoria EPROM o EEPROM, pero este artículo se va a centrar en las de tipo RAM, y dentro de ellos, las fabricadas por XILINX.

Características de las FPGA de XILINX

Con el nombre de LCA ("Logic Cell Array"), XILINX (4) introdujo el primero de estos dispositivos, que fue el XC2064, en dos encapsulados posibles: PLCC y PGA, ambos de 68 patas. Internamente consta de 64 bloques lógicos internos, situados en 8 filas por 8 columnas. Dispone así mismo de 68 patas, 10 de las cuales tienen una misión asignada previamente, mientras que las 58 restantes son configurables (Figura 1).
Cada uno de los 64 bloques lógicos (Figura 2) consta de 4 entradas (A, B, C, D), una entrada de reloj (K) y dos salidas (X, Y). Con las cuatro entradas se puede implementar cualquier función de cuatro variables, o bien dos funciones (F, G) de tres variables cada una. Además, en cada bloque existe un flip-flop cuyo reloj se puede configurar por flanco o por nivel, positivo o negativo, y con entradas de SET y RESET. Las salidas (X, Y) se pueden tomar de las salidas de las funciones, o bien de la salida del flip-flop. (Figura 2)
Cada una de las 58 patas confi-

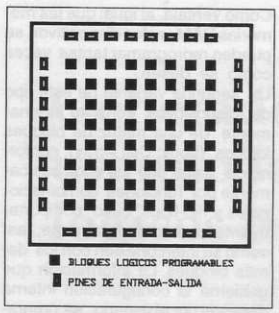


Figura 1. FPGA XC2064 DE XILINX

pueden configurar de tipo TTL o CMOS. (Figura 3)

La conexión entre las 58 patas y los 64 bloques configurables también es programable a través de líneas metálicas y puntos de conmutación internos al dispositivo.

Hay disponibles 3 versiones del dispositivo, con frecuencias de conmutación máxima de los flip-flops de 33MHz, 50MHz y 70MHz, respectivamente.

La programación de los bloques lógicos, las patas configurables y las interconexiones se hace en un PC con un programa CAD gráfico interactivo, suministrado por XILINX. Este programa permite realizar simulaciones y también genera los códigos a cargar en la FPGA. Existen 5 maneras distintas de hacer esa programación: algunas enviando los códigos en serie (bit a bit), otras en paralelo (byte a byte), otras desde un computador o micro- procesador y otras siendo el propio FPGA quién gobierna la programación.

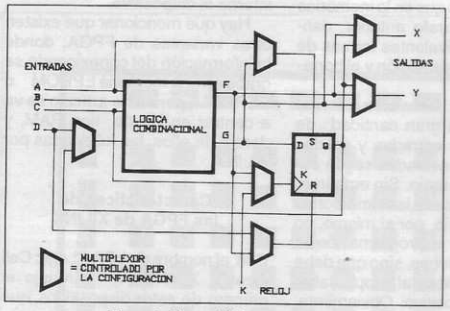


Figura 2. Bloque lógico programable

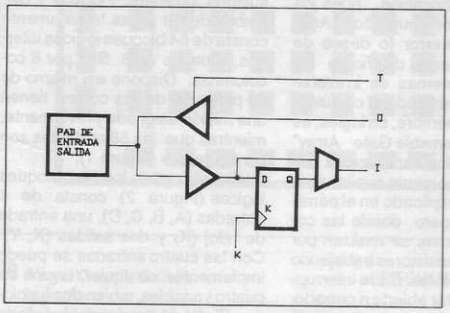


Figura 3. Pin de entrada - salida programable

Aunque es sólo un indicativo sin mayor utilidad práctica, la integración de este dispositivo equivale a 1.200 puertas convencionales, a un costo aproximado de US$20. Notese, que el costo de esas 1.200 puertas en lógica TTL alcanzaría los US$75.

Como cabía esperar, el éxito de estos dispositivos fue muy grande, por lo que XILINX amplió la oferta a nuevas familias de FPGA, siguiendo en todas ellas la misma filosofía ya explicada. Así apareció la XC2018 con 100 bloques lógicos y 74 patas con- figurables (equivalente a 1800 puertas convencionales). Y luego vino la familia XC3000, en la cual hay más posibilidades en cada bloque lógico y en cada pata configurable. La más potente de todas ellas es la XC3090, con 320 bloques lógicos, 144 patas configurables y una equivalencia a 9000 puertas convencionales.

Como es común en la mayoría de los circuitos integrados, existe una segunda fuente donde conseguir los mismos diseños de XILINX: se trata de AMD (Advanced Micro Devices). Existen además otros fabricantes que suministran FPGA con distintas funciones y capacidades, en versión RAM y EEPROM.

Aplicación de FPGA al diseño de un generador de funciones

La ventaja de usar una FPGA en sustitución de muchos circuitos integrados MSI es sustancial. Por un lado, el paquete de diseño de FPGA incluye un módulo de simulación, de modo que se puede probar todo el circuito antes de haberlo construido. Por otro lado, los errores de diseño son fácilmente subsanables y a un costo muy bajo, pues no hay que re- diseñar el circuito impreso, con los subsecuentes costos (como ocurría con anteriores tecnologías), sino que basta modificar por software la configuración de la FPGA. Y como si fuera poco, la ca- pacidad de cargar "sobre la marcha" en la FPGA configuraciones previamente diseñadas, permite que los equipos así construidos sean mucho más versátiles.

Descripción del generador de funciones

Es sobradamente conocida la utilidad de un generador de funciones en cualquier laboratorio de elec-

gurables (Figura 3) se puede programar para que sea entrada, salida, entrada/salida, salida triestado o entrada con "latch" de memorización. Los niveles lógicos de todas las patas del circuito se

trónica. Habitualmente, estos equipos disponen de varias formas de onda de salida (sinusoidal, triangular, diente de sierra, cuadrada...) cuya frecuencia se puede barrer manualmente desde casi 0 Hz hasta 1 MHz típicamente. Algunos traen una entrada especial de modulación de frecuencia, convirtiéndose en lo que se llama técnicamente VCO (Voltage Controlled Oscillator).

Estos generadores de función están basados en circuitería analógica, siguiendo algún montaje clásico. Usualmente generan una forma de onda básica usando algún montaje de oscilador (5), y a partir de ella se sacan las demás ondas. Tal como se explicará enseguida, se pueden mejorar mucho las prestaciones de un generador de funciones si la onda se sintetiza digitalmente.

Supongamos que queremos generar una onda en diente de sierra, como la de la figura 4, usando un microprocesador conectado a un conversor digital a analógico (DAC). El programa en ensamblador (usando un 6809 de Motorola) sería algo así como:

```
        LDA    #0
BUCLE   ADDA   DELTA;  4 ciclos
        STA    DAC;    4 ciclos
        BRA    BUCLE;  3 ciclos
```

En el registro A, que es de 8 bits, se mantiene el código binario a sacar hacia el conversor en cada instante. Ese código se irá incrementando en cada iteración del bucle, según el valor previamente cargado en la variable DELTA. El valor máximo que se puede mantener en el registro A es +127. Un incremento en una unidad lo hará saltar a -128 (estamos operando en complemento a dos). Por tanto, a la salida del conversor DAC tendremos un voltaje similar al de la figura 5. Basta con poner un filtro paso bajo que cumpla con el criterio de Nyquist para eliminar los armónicos de orden superior y obtener la onda deseada.

No obstante, debido a la sencillez de este método existe una limitación: la frecuencia máxima de la onda generada. Efectivamente, en cada línea del programa anterior aparece el número de ciclos que demora la ejecución de la correspondiente instrucción. El ciclo de reloj del 6809 es 1 microsegundo, y como cada iteración del bucle se efectúa en 11 ciclos, el total resulta ser de 11 microsegundos. Según Nyquist, un período de onda necesita al menos de dos puntos, por lo que la onda de frecuencia máxima tendrá un período de 22 microsegundos. Es decir, su frecuencia será de 45 KHz. Este valor es muy pobre, comparado con el de los generadores analógicos.

Como es sabido, todo lo que se hace por software, también se puede implementar por hardware, a un costo mucho mayor, pero obteniendo a cambio un incremento considerable de velocidad (éte es el fundamento del DMA, por ejemplo). Pues bien, tomando esta idea y observando el programa anterior se llega a la conclusión de que el circuito de la figura 6 realiza la misma función. Es decir, el contador software ha sido sustituido por un contador hardware. La velocidad ahora está limitada únicamente por la tecnología empleada. Por ejemplo, usando un contador 74 AS 163 (que puede trabajar con reloj de hasta 75 MHz), la frecuencia máxima de la onda será de 75MHz/ 2 puntos = 37.5 MHz, que representa un buen incremento. Estos valores son apenas indicativos de los límites máximos de trabajo, ya que obviamente, sacando sólo 2 puntos por cada período de la onda, se pierde toda posibilidad de controlar la forma (en definitiva, siempre saldrá sinusoidal).(Figura 6).

Además, no hay que olvidar que el reloj que gobierna al contador digital estará basado en un cristal de cuarzo, por tanto, su precisión y estabilidad será mucho mejor que la de los antiguos osciladores analógicos. Por si fuera poco, a partir de ahora quedan abiertas las puertas a dos mejoras sustanciales:

Primero: Intercalando una memoria (RAM o ROM) entre el contador y el conversor (Figura 7), se consigue que la forma de onda sea una cualesquiera. Si la memoria es ROM, la forma de onda queda fija, mientras que si es RAM, puede ser programada por el usuario. Para ello, basta grabar consecutivamente en cada dirección de la

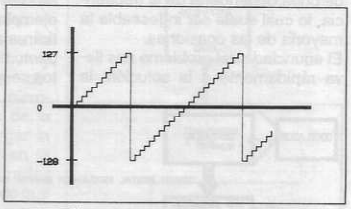


Figura 4. Onda en diente de sierra

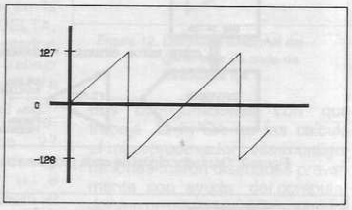


Figura 5. Onda antes del filtro

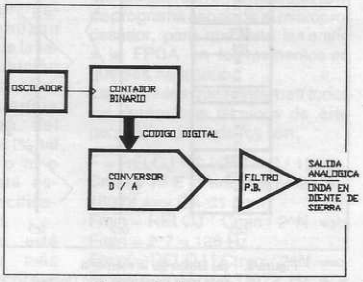


Figura 6. Oscilador digital

memoria el valor deseado para la amplitud de la onda de salida (Fig. 8-a), a lo largo de todo un período.
Segundo: En los montajes digitales vistos hasta ahora (Figuras 6 y 7), la manera de cambiar la frecuencia de la onda era cambiando la correspondiente frecuencia del reloj. Esta solución implicaría desgraciadamente que la frecuencia de corte del respectivo filtro paso bajo habría que variar la proporcionalmente, para seguir cumpliendo el criterio de Nyquist. Aunque hoy día existe la manera de implementar tales filtros, el problema final es que el ancho de banda (contenido armónico) de la salida del generador de onda dependería de la frecuencia, lo cual suele ser indeseable la mayoría de las ocasiones.
El enunciado del problema nos lleva rápidamente a la solución: la frecuencia del reloj del contador hardware hay que dejarla fija. Lo que hay que variar es el incremento del contador. (Obsérvese que el contador software está realizado así precisamente: el incremento en la cuenta no vale 1 sino que es la variable DELTA). Incrementos muy pequeños en el contador producirán ondas de período grande (de baja frecuencia), mientras que incrementos muy grandes en el contador producirán ondas de período corto (de alta frecuencia), según se puede ver en las figuras 8-b y 8-c.
Obsérvese que siguiendo este método, hay puntos en la tabla que no se llevan al conversor D/A. Por ejemplo, si DELTA=2, se convertirán a analógico un punto sí y un punto no. O sea, de cada dos puntos se ignora uno. Y no habrá problema mientras el número de puntos que se ignoren sea menor que la mitad del período del armónico más alto contenido en la onda tabulada. Pero obviamente, si ocurre lo contrario habrá distorsión, pues dejaría de cumplirse el criterio de Nyquist.
Por otro lado, si el incremento DELTA sólo pudiera tomar valores enteros la resolución en frecuencia sería muy pobre. Para mejorarla es obligatorio manejar los incrementos con una parte entera y otra fraccionaria. La suma se hace usando ambas partes, pero sólo la parte entera se lleva al bus de direcciones de la memoria.

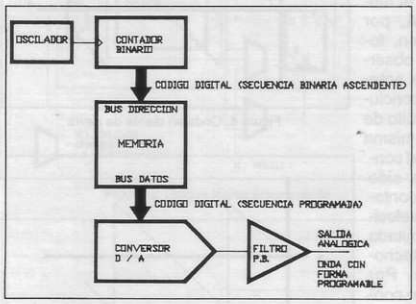


Figura 7. Oscilador digital de onda programable

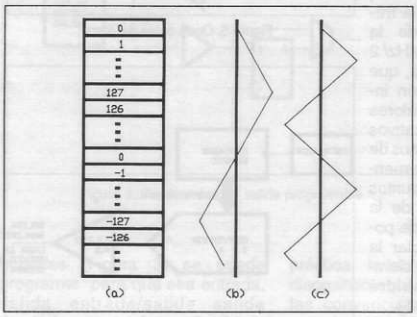


Figura 8. (a) Datos de la memoria
(b) Onda generada con Delta=1
(c) Onda generada con Delta=2

Especificaciones del generador de funciones diseñado

El generador de onda que se diseñó forma parte de un equipo de medida de parámetros de líneas telefónicas. La principal consideración en este caso no fue conseguir una alta frecuencia máxima de la onda de salida, sino:

- Forma de onda de salida: programable en 4096 puntos.
- Frecuencia de salida: programable de 128 Hz a 131 KHz, con tres posibilidades:
- Frecuencia fija
- Frecuencia variable según barridos lineales
- Resolución en frecuencia: 0.01%
- Relación señal/ruido: la de un conversor de 8 bits.

Implementación del generador de funciones con una FPGA deXILINX

Realizar un contador cuyo incremento sea variable no es algo que se pueda lograr eficiente y económicamente con circuitos MSI. Por eso se recurrió al empleo de una FPGA de XILIX, concretamente la XC2064.
El contador se implementó con un sumador-acumulador, cuyo diagrama de bloques lo podemos observar en la figura 9. Toda la circuitería diseñada en el interior de la FPGA, es de tipo bit-slice (rodaja de bit), es decir, se diseño un bloque que opera con un único bit, y luego se añadieron tantos bloques como fueron necesarios. En la figura 10 se puede observar la implementación de un bit del sumador-acumulador, usando los bloques lógicos de la FPGA. Cada bit-slice del sumador acumulador se puede realizar con un único bloque lógico programable implemen-tando las dos funciones lógicas F y G y usando el flip-flop. Dentro de cada bloque se expresan, usando ceros y unos, los miniterminos que activan la respectiva función. La función G implementa la suma y la F genera el acarreo a la siguiente etapa.
DELTA(i) es el bit i-ésimo de la

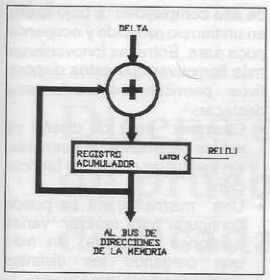


Figura 9. Sumador - acumulador

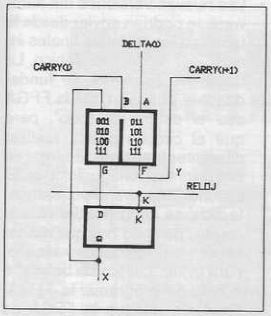


Figura 10. Bit - Slice del sumador acumulador para frecuencia fija

constante DELTA, mientras que CARRY(i) es el acarreo de entrada y CARRY (i+1) es el de salida. El diagrama de bloques de la Figura 11 nos muestra la interconexión del microprocesador con la FPGA, la memoria RAM estática (SRAM) donde se mantiene la forma de onda, y el conversor DAC, así como el filtro paso bajo de salida. El microprocesador se encarga de las tareas rutinarias de entrada/salida (teclado, display y RS-232). También se encarga de la programación adecuada de la FPGA. Obsérvese que el microprocesador no puede acceder directamente a la SRAM. Por ello, cada vez que el usuario decide cambiar la forma de onda, el microprocesador reprograma la FPGA con una configuración muy sencilla, que simplemente conecta el bus del propio microprocesador con los buses de la memoria. Después de cargar la forma de onda deseada en la SRAM, el microprocesador reprograma la FPGA de modo que funcione como sumador-acumulador (Figura 12). El propio microprocesador escribe en la FPGA el valor adecuado DELTA, para incrementar el contador, según la frecuencia de trabajo pedida por el usuario. Los 27 bits de que consta el sumador-acumulador garantizan que la resolución en frecuencia sea del 0.01%, tal como está especificado.

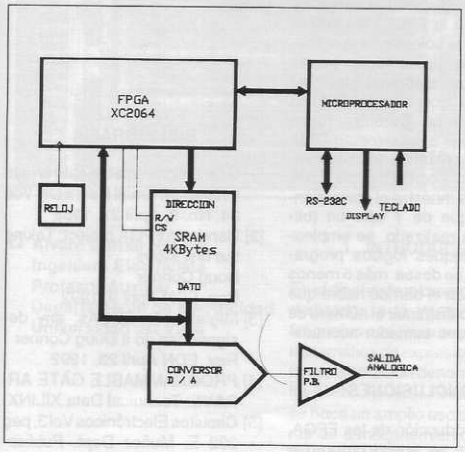


Figura 11. Diagrama de bloques del esquema eléctrico

No está de más recordar que las dos configuraciones con que trabaja la FPGA no las calcula el microprocesador: esas configuraciones fueron diseñadas previamente con ayuda del computador, en un paquete CAD proporcionado por XILINX. Una vez diseñadas, se guardan en la habitual ROM de programa asociada al microprocesador, para que éste las envíe a la FPGA en los momentos en que sea necesario.

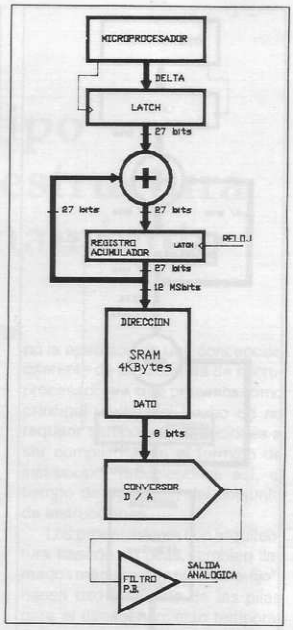


Figura 12. Diagrama funcional del generador de onda de frecuencia fija

Las fórmulas que relacionan todos los parámetros técnicos de este generador de onda fija son:

F = RELOJ * C / 2^N = C / 128
Cmin = 1 / E ; Para Cmin = 2^14 = 16378 ==> E = 61 ppm
Fmin = RELOJ * Cmin / 2^N ==> Fmin = 2^7 = 128 Hz
Fmax = RELOJ * Cmax / 2^N ==> Fmax = 2^17 = 131072 Hz A = RELOJ / (2 * Fmax) = 4.

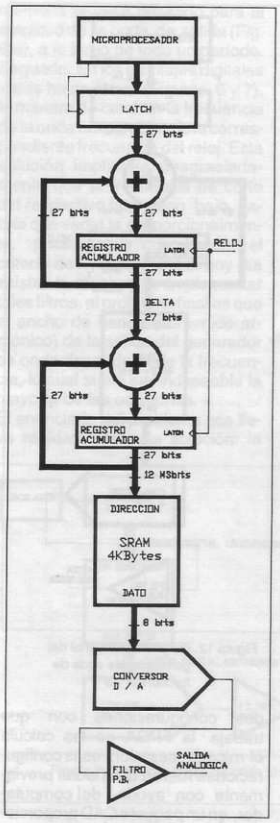


Figura 13. Diagrama funcional del generador de onda con barrido lineal en frecuencia

siendo:

F = frecuencia de la onda de salida
C = código digital (DELTA) a introducir como incremento
RELOJ = frecuencia de muestreo = 2^20 = 1.048576 MHz
N = número de bits en el sumador-acumulador = 27
E = resolución deseada en frecuencia < 1e-4
A = contenido armónico en el caso peor (debe ser mayor que 1 para que cumpla con el criterio de Nyquist).

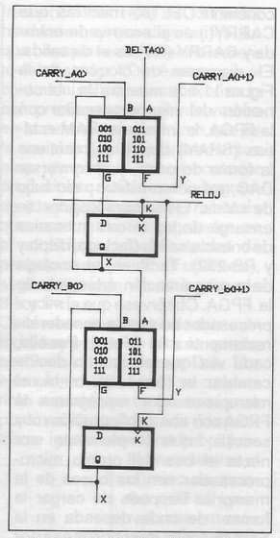


Figura 14. 1 Bit - slice del sumador acumulador para barridas lineales

Y obviamente, teniendo tanta potencia en un dispositivo así (estoy hablando de la FPGA) los límites vienen dados sólo por la imaginación. En la figura 13 se puede observar un diagrama funcional que permite hacer barridos lineales en frecuencia. Esto se logra haciendo que el incremento DELTA no sea constante, sino aumente linealmente con el tiempo. En la figura 14 se muestra el correspondiente bloque de 1 sólo bit (bit-slice). Para realizarlo se emplearon dos bloques lógicos programables. Si se desea más o menos resolución en el barrido habrá que aumentar o disminuir el número de bits del nuevo sumador-acumulador.

CONCLUSIONES

Con la introducción de las FPGA, el hardware se vuelve programable, permitiendo diseñar circuitos de alta complejidad a bajo costo, en un tiempo reducido y ocupando poca área. Entre las innovaciones más llamativas que estos dispositivos permiten, merece la pena destacar:

- Cualquier error de diseño se vuelve fácilmente subsanable, sin involucrar costos de fabricación.
- Una misma tarjeta se puede configurar para realizar varias funciones hardware, sin más que cargando las distintas configuraciones previamente diseñadas.
- Las nuevas versiones del hardware se podrían enviar desde la fábrica a los usuarios finales incluso por modem telefónico. La tarea del diseñador se fundamentará ahora en que la FPGA sea "el centro de todo", para que el circuito pueda realizar diferentes funciones al cargar en ella distintas configuraciones. El circuito "hardware" propiamente dicho se simplifica dramáticamente, pero no hay que olvidar hacer un cuidadoso diseño, para evitar "cuellos de botella" a la hora de programar la FPGA. Es decir, aunque la FPGA es grande, hay un límite a lo que uno puede configurar en su interior.

BIBLIOGRAFIA

[1] User programmable gate arrays Charles H. Small Rev. EDN Vol. 34, No. 9, April 27, 1989
[2] Hands on FPGA project: Taking the first steps Doug Conner Rev. EDN April 9, 1992
[3] Migration to FPGAs: any designer can do it Doug Conner Rev. EDN April 23, 1992
[4] PROGRAMMABLE GATE ARRAYS. Technical Data XILINX
[5] Circuitos Electrónicos Vol 3, pag 232 E. Muñoz Dept. Publicaciones ETSIT Madrid