FPGA

Quien sabe programa

Por Paul Goossens

En el aspecto tecnológico los FPGA han evolucionado mucho en poco tiempo. Ahora los FPGA son aptos para las aplicaciones más dispares. Sin embargo, la mayor parte de aparatos nuevos todavía no utilizan las posibilidades que ofrecen estos chips. Y la razón principal es que todavía hay relativamente pocos expertos en FPGA. El más reciente desarrollo en el campo EDA podría eliminar en gran parte esta limitación.

De origen, la mayoría de aparatos y aparatejos están provistos de un microcontrolador. Incluso los proyectos más sencillos suelen contener un microcontrolador. Pero no siempre ha sido así. En los inicios del microcontrolador el uso de estos componentes estaba limitado a un selecto grupo de ingenieros. Los aparatos electrónicos habituales se realizaban al 100% con electrónica analógica.

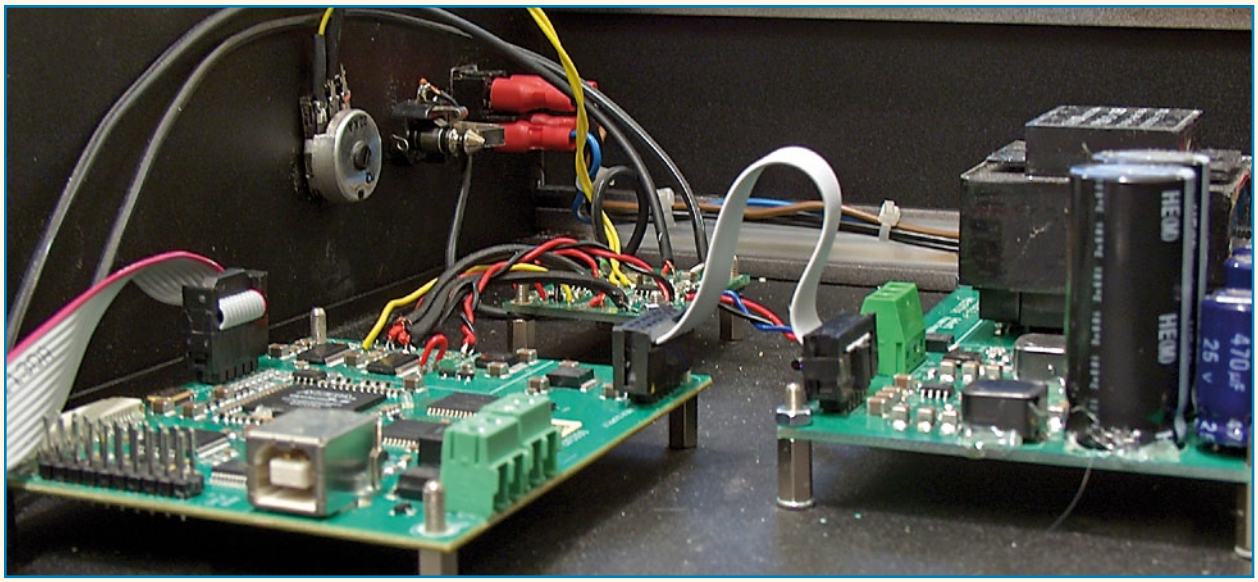
El misterioso FPGA

Hasta hace poco, los microprocesadores sólo se usaban en los ordenadores. Los ingenieros que trabajaban con estos chips, eran considerados expertos, un área aparte de la electrónica común. La programación del firmware correspondiente se realizaba mayormente en código ensamblador. Los diseñadores más iniciados en analógica (léase la gran mayoría de los diseñadores) consideraban que los microprocesadores y los controladores eran componentes misteriosos.

Y ahora ocurre una situación similar en torno a los FPGA. La mayoría de diseñadores ha oído hablar de los FPGA y están interesados en estos prometedores chips, pero en sus proyectos siguen aferrándose a sus conocidos microcontroladores. En muchos casos un microcontrolador también es la elección más lógica, pero cada vez más a menudo el FPGA constituirá un buen sustituto del microcontrolador.

Avance

Con respecto a los microcontroladores, los FPGA son conocidos como chips caros que consumen mucha energía. Los fabricantes de FPGA han reaccionado al respecto redu-



‘made easy’

r, también sabe como manejar FPGA

ciendo el hambre de energía de sus chips y desarrollando FPGA económicos. En este campo ya se ha conseguido bastante y las especificaciones no harán más que mejorar en el futuro.

Al navegar por los sitios web de los distintos fabricantes de FPGA, se observa que cada fabricante ofrece una familia de FPGA “de bajo coste” y “baja potencia”. Con ello, la elección entre un microcontrolador y un FPGA empieza a no ser tan sencilla si sólo se toman en cuenta los de bajo coste y baja potencia.

Otro gran inconveniente ante el uso de FPGA es que todavía hay relativamente pocos diseñadores que estén familiarizados con el desarrollo de diseños basados en FPGA.

El lenguaje de programación más utilizado para programar controladores es el lenguaje C, que funciona totalmente según el principio secuencial. El contenido de los FPGA, por el contrario, se diseña principalmente en VHDL o en Verilog. Estos dos lenguajes son independientes del pensamiento secuencial. El paso de C a uno de estos dos lenguajes requiere una forma de pensar totalmente distinta.

Para los diseñadores esto constituye un reto bastante considerable. Alguien que quiera defenderse más o menos con VHDL, por ejemplo, entonces seguro que tendrá que invertir tiempo en la adquisición de conocimientos y experiencia. Esto entraña que el primer par de diseños basados en un FPGA, requerirán mucho más tiempo en comparación con un proyecto igual basado en un microcontrolador. Para muchas empresas esto es motivo suficiente para realizar normalmente un microcontrolador.

Dos mundos

Sin embargo, en cuanto se decide emplear un FPGA, a menudo también se usa un microcontrolador. Ya sea en forma de un controlador externo, o se elija colocar un controlador en el FPGA.

De esta forma se puede combinar lo mejor de ambos mundos. Así, gran parte de la aplicación se puede seguir diseñando de la forma tradicional. El FPGA se utiliza entonces para una parte específica del proyecto, con lo cual el resto se soluciona de la forma conocida con firmware.

Entonces se añaden bloques IP listos para su uso para obtener una base funcional. A veces aún se añade a esta mezcla una parte de hardware desarrollada por uno mismo (en VHDL, verilog, etc.). Este ensamblaje se puede introducir gráficamente en muchos software EDA, de manera que tampoco hace falta adaptar ninguna letra de VHDL o Verilog. Gracias a estos métodos los diseñadores no tienen que dedicar mucho tiempo al proyecto VHDL.

De C a H

El desarrollo más novedoso, que para mayor comodidad denominaremos de “C a H”, es el paso siguiente en el desarrollo sin VHDL de los circuitos FPGA.

Varios desarrolladores de EDA han desarrollado métodos que permiten desarrollar un circuito digital sin VHDL. El método más notable contiene la conversión directa de código fuente ANSI-C estándar en un proyecto digital. Ya existían compiladores que podían convertir un sub-set propio de lenguaje C en un circuito digital. Los compiladores más modernos pueden defenderse con el C estándar (ANSI-C).

Tanto el fabricante de FPGA Altera (compilador C2H),

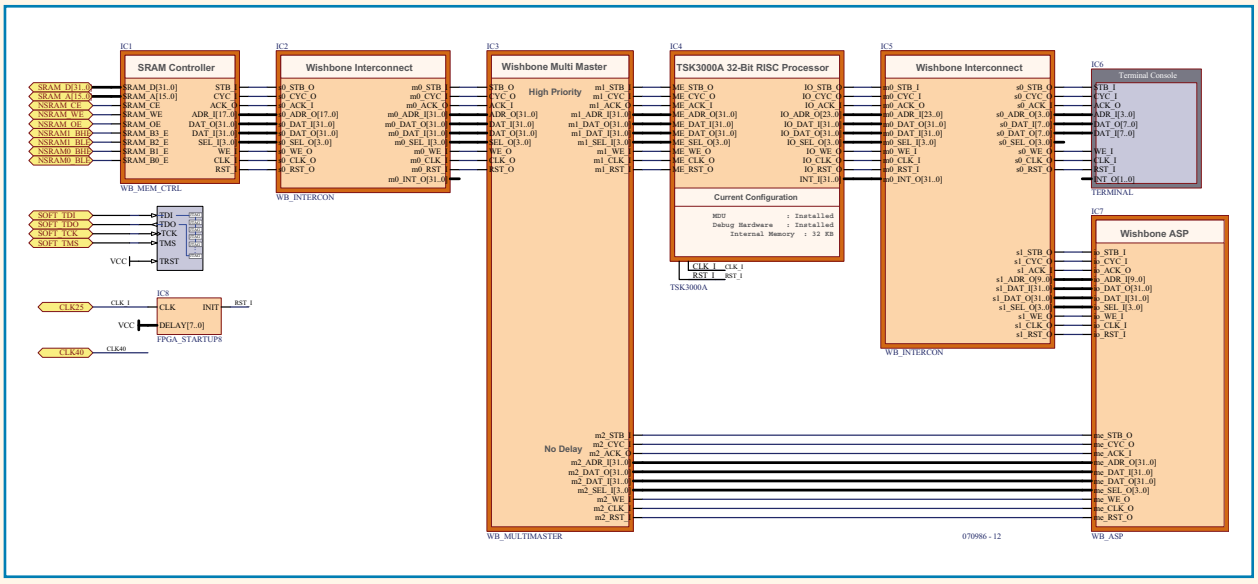
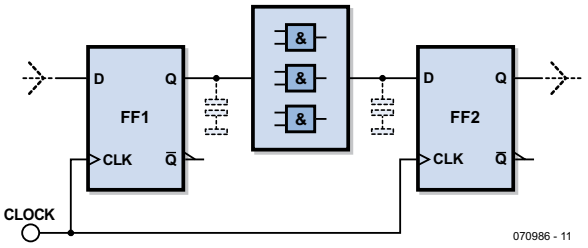


Figura 1.
Esquema de un diseño interno de FPGA

Consumo de corriente estática frente a dinámica

La corriente que consume un chip se puede subdividir en dos partes. Una parte de la corriente seguirá pasando siempre mientras haya tensión de alimentación. Esta corriente consta de distintas corrientes de dispersión, pero también de corrientes para seguir reteniendo, por ejemplo, el contenido de memorias estáticas y similares. Esta corriente recibe el nombre de corriente estática.

La segunda parte de la corriente es la llamada corriente dinámica. El origen de esta corriente se encuentra en el cambio de nivel de diversas señales internas del chip.



Como ya sabemos, un chip digital (y por tanto también un FPGA) consta de una colección de circuitos digitales sencillos. Todos estos circuitos tienen una salida. En la electrónica digital estas salidas pueden tener dos niveles, "1" y "0", o "alto" y "bajo". Estas salidas están conectadas con los conductores del chip a una o múltiples entradas de otros circuitos lógicos del chip.

La física nos enseña que cada conductor tiene una capacidad determinada en relación a otros conductores de sus alrededores. Estos condensadores parasitarios indeseados suelen causar problemas a los ingenieros electrónicos. Y también a los chips. En cuanto una salida cambia de nivel, cambia el nivel de tensión de la señal de salida.

Como esta señal tiene una capacidad determinada en relación a la masa, entre otros, tendrá que pasar corriente de o hacia esta capacidad para obtener un cambio en el nivel de tensión.

En caso de que una salida aumente de 0 a 5 voltios, la salida deberá suministrar una corriente para cargar la capacidad parasita hasta que alcanza este nivel de tensión de 5 voltios. En cuanto esta salida cambia de 5 a 0 voltios, la carga de esta capacidad debe transportarse a masa, antes de que el nivel de tensión vuelva a 0 voltios.

La cantidad de carga que se requiere para cargar esta capacidad depende de dos factores. El primero es la diferencia en el nivel de tensión entre "0" y "1". Al reducir la tensión de alimentación, la carga necesaria es menor. El segundo factor es la magnitud de esta capacidad parasitaria. Con la adaptación de la estructura interna del chip y la elección de los materiales adecuados, esta capacidad se puede reducir. Aquí también rige que una capacidad más pequeña hace que sea necesaria menos carga para cambiar de nivel.

SINCRÓNICO

En un diseño sincrónico el chip reacciona en la función de una señal de reloj. En el momento en que aparece un impulso de reloj, pueden cambiar las salidas de los biestables. Los posibles circuitos combinatorios detrás *de estos biestables recibirían entonces otra entrada. Con ello, eventualmente sus salidas también pueden cambiar de nivel.*

Tras un breve tiempo de demora todas las salidas de los biestables y los circuitos combinatorios habrán adoptado su nivel definitivo. Estas salidas sólo pueden volver a cambiar cuando llega un nuevo impulso de reloj.

HERCIOS POR CORRIENTE

El consumo de corriente dinámico podemos restringirlo reduciendo la frecuencia de reloj. Desciende a menos impulsos de reloj por segundo. Esto se traduce directamente en una reducción de la corriente dinámica.

Podemos conseguir otra restricción de la corriente dinámica procurando que cambien de nivel el mínimo posible de salidas por impulso de reloj. Esto se puede conseguir deteniendo partes del chip, que en un momento concreto no tengan que realizar un trabajo importante. De esta manera, se puede alcanzar con un diseño inteligente, un buen ahorro de corriente.

Glosario explicativo

ASP – Application Specific Processor. Circuito digital, adaptado a una tarea específica.

EDA – Electronic Design Automation. Nombre colectivo para todos los softwares destinados a facilitar el desarrollo de aparatos electrónicos.

FPGA – Field Programmable Gate Array. Chip que suele constar de decenas de miles de pequeños circuitos lógicos. Estos pueden acoplarse mediante un programador y formar así circuitos digitales complejos y grandes. Incluso permite "construir" microcontroladores completos, tarjetas de vídeo, etc.

IP – Intellectual Property. En relación con los FPGA, significa una parte de código ('IP-core') que se puede adquirir, en general con una licencia. Este código tiene una función compleja predefinida, que se puede usar fácilmente en un proyecto FPGA.

como el desarrollador de EDA Altium (compilador CHC) han desarrollado un compilador que lo consigue. Se espera que pronto otros proveedores lancen al mercado sus propios compiladores de "C a H".

FPGA para todos

La fuerza de estos compiladores es que las rutinas, desarrollas a la antigua manera conocida, ahora utilizan la fuerza del FPGA.
Un controlador se diseña originalmente para ser lo más flexible posible. Esto permite al controlador llevar a cabo las tareas más dispares. Este es también uno de los motivos del gran éxito de los microcontroladores. Pero al mismo tiempo es también su punto débil. El controlador es un factótum para todo, pero no es un especialista.
Un FPGA es precisamente como un microcontrolador, un factótum. Pero a otro nivel. En él se puede uno montar un circuito digital propio (por lo tanto también un controlador), pero también una parte de hardware especializado que sobresale en una tarea específica. Estas partes de hardware se pueden desarrollar en C con la ayuda de estos compiladores de "C a H". Esto significa que un programador sin experiencia en VHDL también puede aprovechar la fuerza de un FPGA.

Práctica

¿Cómo funciona en la práctica? En nuestro laboratorio contamos con Altium Designer, que desde la versión 6.8 se entrega con un compilador de "C a H", denominado CHC. Con la ayuda de este software hemos utilizado un sencillo ejemplo para ver como funciona todo esto en la práctica.
En nuestro ejemplo (**figura 1**) hemos usado exclusivamente los componentes IP estándar de Altium. Con ello se puede crear en un abrir y cerrar de ojos un sistema empotrado. El procesador (IC4) es un TSK3000, un procesador de 32 bits. A través de una serie de bloques este procesador se conecta entre otros a la memoria externa, un emulador de terminal y nuestro protagonista: el ASP. Finalmente, aparece en el bloque el resultado del compilador CHC.
Para más información sobre los bloques usados en este esquema, véase el sitio web de Altium.

Test

Para realizar el test, además del software y un proyecto FPGA también necesitamos hardware. Un futuro proyecto de Elektor resultó que era sumamente adecuado para ello. Este hardware contiene un FPGA Cyclone-III, más 256kByte de memoria externa.
En nuestro firmware de test dejamos que el procesador calculara 500.000 veces la raíz cuadrada de una cifra. Primero hicimos que el cálculo lo hiciera el ASP (es decir resultado del compilador CHC) y medimos el tiempo necesario para ello.
Seguidamente, pusimos a trabajar la versión de software de nuestra rutina de cálculo de raíz cuadrada 500.000 veces, en la que también medimos el tiempo empleado.
La rutina utilizada se ha copiado en el *listado 1*. La primera parte intenta con un método inteligente llegar al resultado. El algoritmo de este intento está escrito de tal manera que el resultado siempre es un poco demasiado elevado. Seguidamente se usa este valor para comprobar si el resultado es correcto. Si no es el caso, entonces el resultado pronosticado se rebaja de uno. Se repite esta receta hasta obtener el resultado correcto.
Lo bueno de este algoritmo es que se utilizan dos bucles: instrucciones de desplazamiento (shift) y la multiplicación. Así, una buena combinación de muchas operaciones frecuentes.

Listado 1: cálculo de raíces cuadradas

```
unsigned int isqrt_sw (int number)
{
  signed int n  = 1;
  signed int n1 = (((n) + (number)/
(n)) >> 1);

  while(((n1 - n) > 1)||((n-n1) >
1)) {
    n  = n1;
    n1 = (((n) + (number)/(n)) >>
1);
  }
  while((n1*n1) > number) {
    n1 -= 1;
  }
  return n1;
}
```

Exportación al hardware

Esta rutina, que se llama *isqrt_sw,* la utilizamos ahora para medir el tiempo que se necesita para realizar el cálculo. Para probar el compilador CHC, hacemos una copia exacta de esta rutina. Y le damos el nombre de *isqrt*.
Ahora tenemos que procurar que el entorno de desarrollo se encargue de que esta rutina se transporte al hardware a través del compilador CHC. Este proceso es sumamente simple. Primero debe colocarse el cursor en el caso de rutina. Al pulsar el botón derecho del ratón aparece un pequeño menú. En el menú elegimos la opción "Push and export hardware" (**figura 2**). Con ello hacemos que la

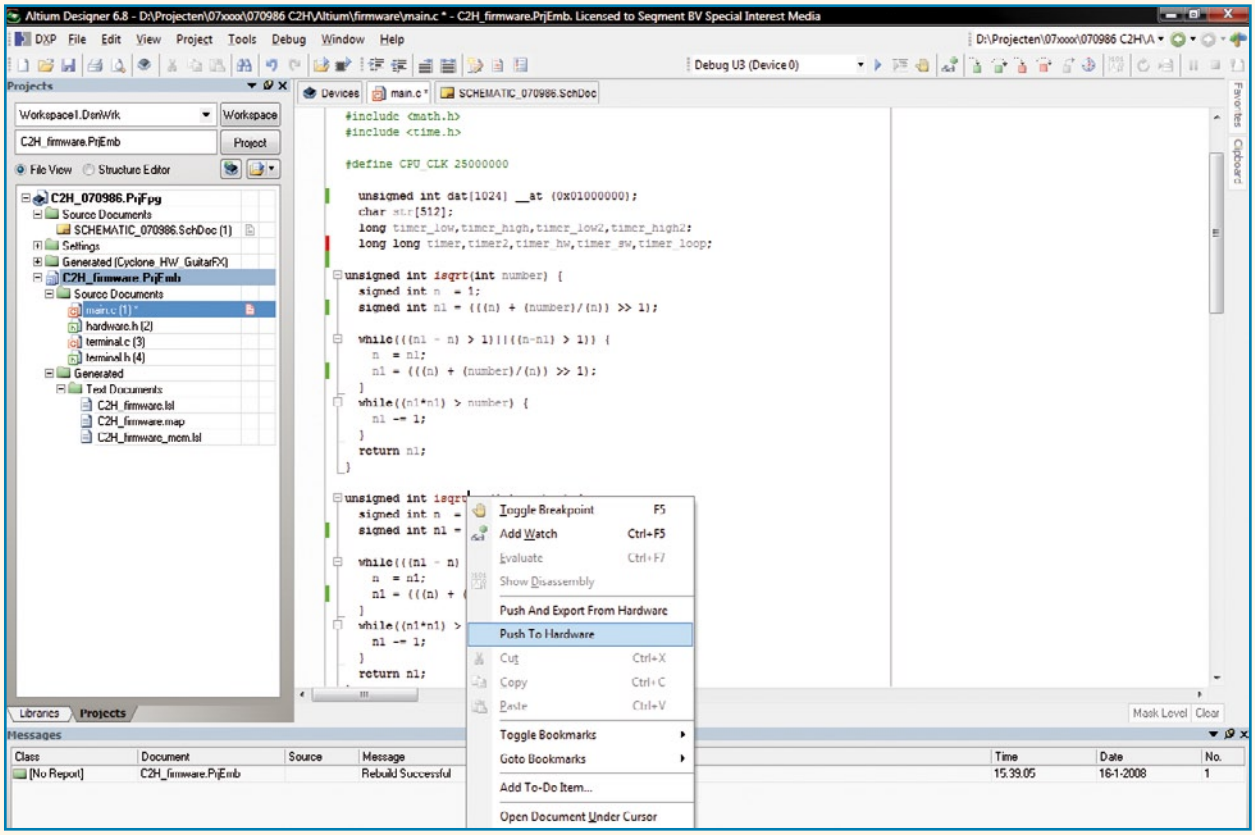


Figura 2.
La prisa, no puede ser más fácil

rutina correspondiente se realice en el hardware a través del compilador CHC. Además, esta acción se encarga de que cualquier llamada del firmware de esta rutina, se sustituya por una llamada del ASP. ¡Y todo ello con un sólo clic del ratón!

Resultado

Tras compilar y programar el FPGA, el terminal se encarga de que la salida del programa sea visible en el PC (**figura 3**). Ahora resulta que la versión de hardware necesita un poco más de medio segundo para realizar un cálculo de raíz cuadrada 500.000 veces. En este mismo tiempo, la versión de software lo repite casi 13 veces.
Lo mejor de todo esto es que en nuestro ejemplo (deliberadamente) no hemos escrito una sola línea en VHDL. Así que esta tecnología permite a los programadores utilizar las posibilidades que ofrece un FPGA sin tener que abandonar el conocido lenguaje de programación C.

Aplicaciones

El ASP puede considerarse como un coprocesador de construcción propia. El ASP puede adaptarse según la aplicación, de manera que la aceleración del hardware puede utilizarse en casi cualquier aplicación imaginable.

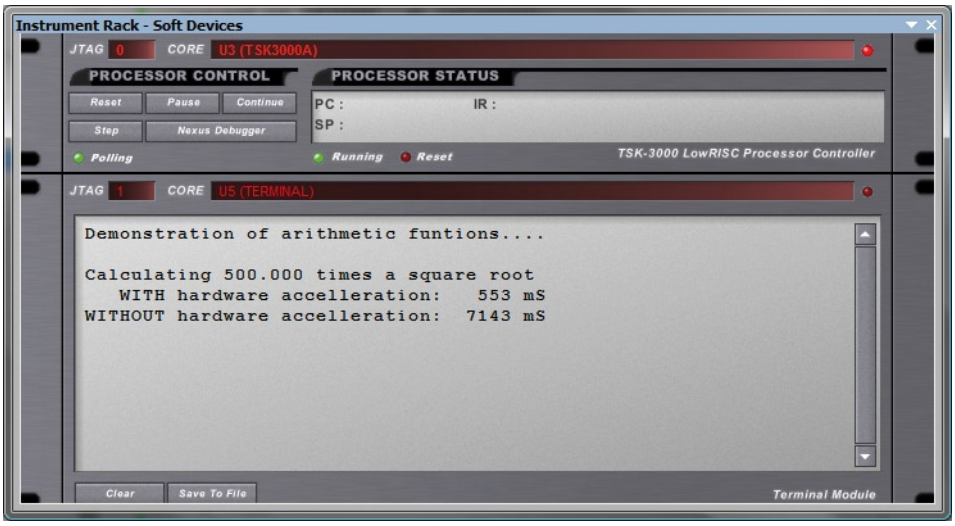


Figura 3.
Resultado del test

También es posible dirigirse a la memoria desde el ASP. Esto hace posible el diseño por ejemplo de un acelerador de vídeo. En este caso, las rutinas para dibujar líneas, círculos y planos pueden realizarse más rápidamente en un ASP.
Incluso se pueden realizar rutinas en el ASP paralelamente con rutinas en software. Evidentemente esto requiere una adaptación del software. La ventaja es que la potencia de cálculo es todavía mayor.
El ejemplo del acelerador de vídeo que acabamos de mencionar se presta a un enfoque de este tipo. El procesador no tiene que esperar hasta el que acelerador de vídeo haya dibujado una línea, un círculo u otro objeto y entretanto puede dedicar su tiempo a otras tareas.

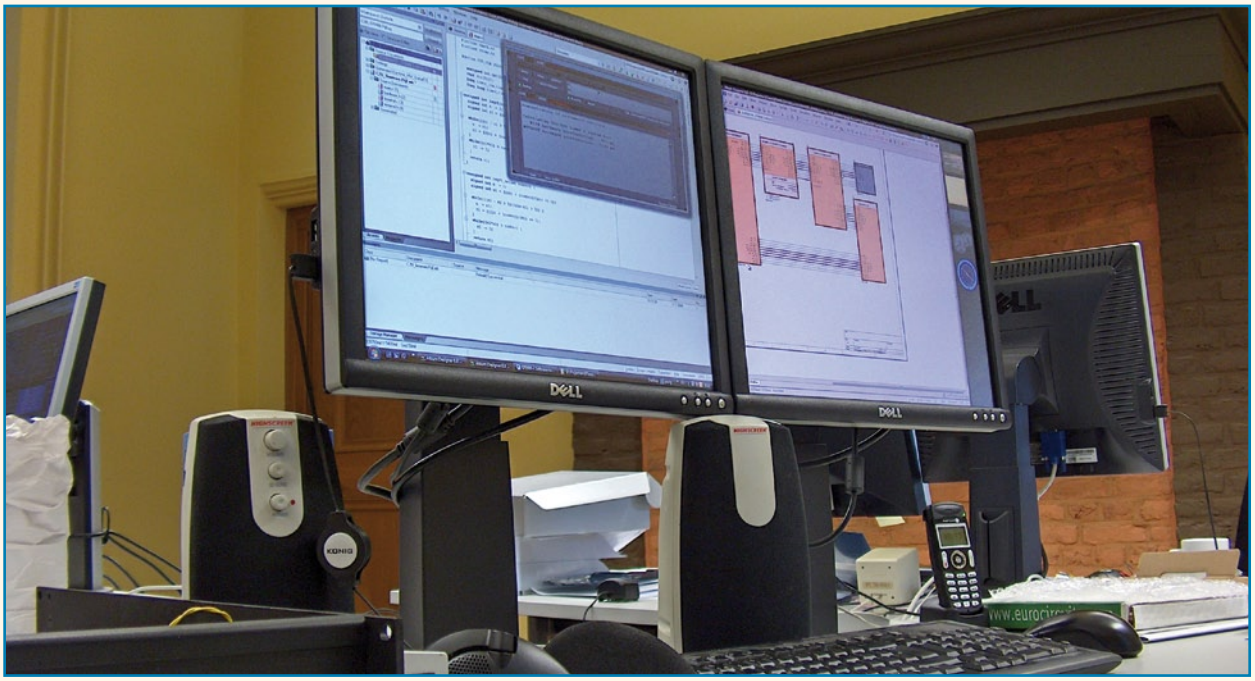
Baja potencia

Una ventaja menos obvia es que el uso de un ASP ofrece la posibilidad de ahorrar energía. Aumentando la potencia de cálculo con ayuda de un ASP, la frecuencia de reloj puede reducirse. La ganancia de velocidad, obtenida con el ASP, se puede cambiar total o parcialmente por una frecuencia de reloj más baja.
La toma de corriente dinámica del FPGA desciende proporcionalmente con la disminución de la frecuencia de reloj. Así, el resultado es un consumo de energía más bajo. Esto es especialmente importante en aparatos portátiles y las aplicaciones respetuosas con el medio ambiente.

Para terminar

Los nuevos desarrollos en torno a los compiladores de C a H se encargan de que los FPGA sean asequibles para programadores que no tienen experiencia con FPGA, VHDL y objetos afines. También el hecho de reducir el tiempo de desarrollo y la posibilidad de reutilizar las antiguas rutinas conocidas en FPGA, es un gran paso adelante en la aceptación de la tecnología de FPGA entre los ingenieros.
En combinación con el avance tecnológico entre los fabricantes de FPGA, se consigue que los FPGA sean cada vez más competidores directos de los controladores embebidos convencionales.

(070986)



¿De C a H para PC?

En algunos supercomputadores actualmente ya se instalan FPGA, para dar apoyo a los procesadores en tareas de aritmética penosas.

No es difícil imaginar que en el futuro estos FPGA también encontrarán su sitio en las placas madre de los PC estándar. Para ello evidentemente será necesario que haya un patrón, que especifique como debe comunicarse este procesador con el FPGA. Y aún más importante es que también se necesita una tecnología estándar para transportar segmentos de software hacia una versión de hardware adecuada para alojar un FPGA. Los nuevos compiladores de C a H ya son un paso en esta dirección. Quién sabe qué puede significar esta tecnología en el futuro para la velocidad de nuestros PCs.