Estimación de Consumo de Potencia en FPGA a través de un Servicio Web

E. Todorovich[1], A. Holderbeke[2] y E. Boemo[1]

[1] Escuela Politécnica Superior, Universidad Autónoma de Madrid, España
{elias.todorovich, eduardo.boemo}@uam.es

[2] KATHO - Departement VHTI, Belgium
arne.holderbeke@pandora.be

Resumen. En este artículo se describe un servicio Web sencillo para una herramienta de estimación de consumo en circuitos combinacionales implementados en FPGA. El servicio Web se provee mediante el protocolo SOAP, ampliamente aceptado. Una de las ventajas de esta tecnología es que los usuarios remotos pueden acceder a la herramienta en su versión más actualizada sin hacer ningún esfuerzo de instalación o mantenimiento. Si bien un usuario remoto puede usar este servicio en sus aplicaciones mediante invocaciones estilo RPC o mensaje a servidor, también se desarrolló una interfaz Web que sirve para acceder al servicio y como demostración de su uso. En la versión actual se estima la potencia consumida por el circuito y los nodos individuales mediante estadísticas. Los dispositivos soportados son los de Xilinx. En esta técnica el usuario especifica el nivel de confianza para el error tolerado en la estimación. También se genera información para construir mapas de consumo en el chip.

1 Introducción

Si bien una herramienta de estimación de consumo medio desarrollada se viene mejorando y probando en el laboratorio [1-3], está claro que para conseguir versiones estables y depuradas, debe ser evaluada por el mayor número de usuarios que sea posible. Para que esta herramienta EDA (*Electronic Design Automation*) esté disponible para los usuarios en Internet, se desarrolló una API (*Application Program Interface*) simple que encapsula su funcionalidad: La API usa SOAP como mecanismo de comunicación debido a su amplia aceptación en Internet. Prácticamente cualquier lenguaje de programación tiene librerías para usar SOAP sin dificultad. De esta manera la herramienta se pone a disposición de los usuarios de todo el mundo como un servicio Web.

Los servicios Web comienzan a difundirse masivamente a partir de cuando compañías como Google o Amazon los adoptan para poner sus servicios disponibles a programadores de aplicaciones en Internet. Los servicios Web son aplicaciones accesibles desde Internet desarrolladas según estándares abiertos. Se puede construir un servicio Web desde cualquier plataforma y usando el lenguaje de programación preferido. El protocolo SOAP para el intercambio de mensajes (*Simple Object Access Protocol*), es un protocolo basado en XML (*eXtended Markup Language*) para invocar procedimientos remotos e intercambiar información de manera descentralizada y distribuida. Los mensajes SOAP se transportan sobre HTTP y sus capas inferiores por Internet. En la Fig. 1 se da una representación de esta pila de protocolos.

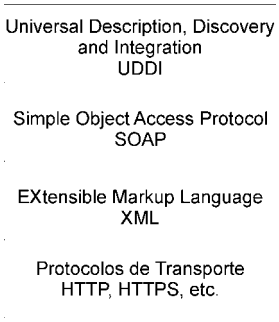

Fig. 1. Pila de protocolos de los servicios Web

Mientras que las páginas Web son para los humanos, los servicios Web son para los ordenadores. Las páginas Web contienen cierta información o realizan cierta tarea y se puede acceder desde un navegador. Los servicios Web en cambio permiten el intercambio de información y la realización de tareas entre ordenadores mediante SOAP/XML. Del mismo modo, cuando se navega por un portal, normalmente se cuenta con un índice, o mapa del sitio. Este índice permite conocer las partes accesibles del portal. Con los servicios Web, este "índice" es el archivo WSDL (*Web Services Description Language*). Este archivo WSDL, también en formato XML, indica al ordenador que lo consulta, qué servicios puede requerir con una referencia precisa sobre ellos, para poder invocarlos usando los parámetros adecuados.

Siguiendo con el paralelismo anterior, al igual que en la Web existen buscadores como Google, el concepto equivalente a nivel de servicios Web es UDDI (*Universal Description, Discovery and Integration*). UDDI es, a su vez, un servicio Web que se puede utilizar desde las aplicaciones cliente para descubrir de forma dinámica otros servicios. Este servicio se puede considerar las páginas amarillas de los servicios Web.

Un escenario más dinámico donde un servicio Web utiliza estos protocolos se ve en la Fig. 2.

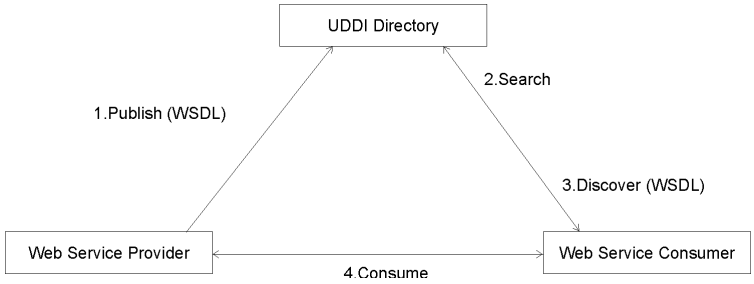

Fig. 2. Uso de los protocolos WSDI y UDDI

Para proveer y usar un servicio Web en Internet, el proveedor del servicio lo publica mediante WSDL en un directorio de servicios que usa UDDI. Los clientes buscan el servicio en el directorio y obtienen la manera de comunicarse con el proveedor de servicio usando parte de la especificación WSDL. El cliente, finalmente, invoca procedimientos en el servidor. El servidor devuelve una respuesta en el caso que la comunicación sea del tipo

RPC o nada si se le envió un mensaje. Para conocer mejor esta tecnología se puede buscar información en Internet o en revistas especializadas [4]. Aquí se va a mostrar como se usa para poner una herramienta EDA a disposición de todo usuario con conexión a Internet.

2 Implementación

En este trabajo solamente se implementó una primera capa del esquema anterior, que provee invocación de procedimientos en el servidor mediante mensajes SOAP como se ve en la Fig. 3. Esto quiere decir que cualquier cliente tiene que conocer la dirección IP o URL (*Uniform Resource Locator*) donde está el servidor, los nombres de las funciones que provee la API implementada y los parámetros de estos métodos. En esta etapa del desarrollo, el servicio propuesto no está en las "páginas amarillas". Por ahora sólo puede usarse si un "amigo" nos da la dirección del servidor.

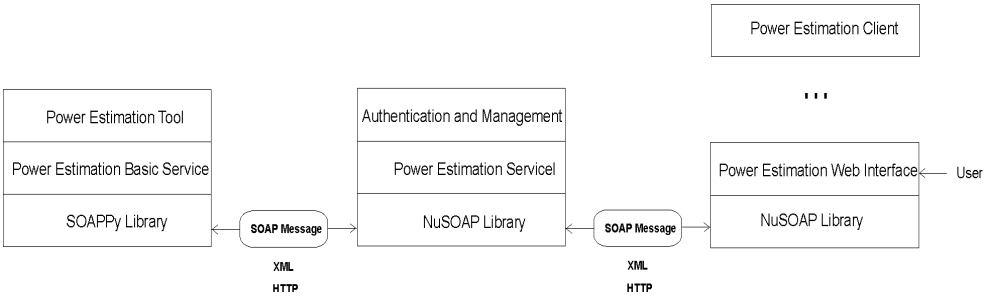

Fig. 3. Servicio *web* de estimación de consumo en tiempo de ejecución

En la Fig. 3 se tiene, en el centro, el servicio Web principal que usan los clientes externos. Provee estimación de consumo de potencia de diseños en FPGA, para usuarios registrados que deben autenticarse. Esto es necesario para tener una mínima privacidad sobre los trabajos que cada usuario requiere. El servicio también administra las tareas de los diferentes usuarios. Como el proceso de estimación de consumo demora como mínimo unos cuantos minutos, las tareas requeridas por los usuarios se encolan por orden de llegada. Si la estimación se pide con un alto nivel de confianza que el error sea muy pequeño, el software puede necesitar incluso horas para completar la tarea.

A la izquierda, en la Fig. 3, está el servicio Web de Estimación de Consumo básico. Este servicio solamente encapsula la funcionalidad de la herramienta EDA de Estimación de Consumo. Cada vez que termina una tarea de estimación, el servicio básico coordina con el servicio principal para procesar la siguiente tarea en espera.

En la parte derecha de la figura están los clientes del servicio de Estimación de Consumo. En particular, se desarrolló una interfaz Web que sirve de demostración en vivo de cómo se usa el servicio Web de Estimación de Consumo. Los usuarios, una vez autenticados, pueden requerir una nueva tarea de estimación de consumo para uno de sus diseños implementados en FPGAs de Xilinx. También pueden ver cuales de las tareas previamente requeridas están terminadas o pendientes, y bajar los resultados de la estimación de sus tareas terminadas a su ordenador remoto.

En cada una de las sub secciones siguientes se presentan los detalles de implementación del servicio principal, el servicio básico de Estimación de Consumo y el cliente basado en la interfaz Web.

3.1 Servidor Básico de Estimación de Consumo

La herramienta EDA de estimación de consumo medio, se basa en la técnica estadística. Permite obtener estimaciones tanto del circuito completo como de los nodos individuales. Además, al acceder a archivos del fabricante con información luego de haber hecho el emplazamiento y el rutado, se puede hacer la correspondencia entre las posiciones físicas de las pistas y sus consumos individuales de potencia. Con esta información se pueden hacer mapas de consumo en la FPGA.

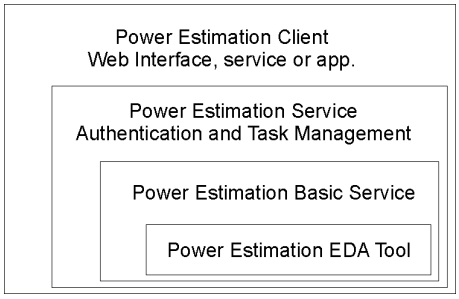

Fig. 4. Capas de software sobre la herramienta EDA de Estimación de Consumo

El servicio *web* básico que encapsula esta funcionalidad (Ver Fig. 4) provee 5 métodos públicos:

```
exec_task_a_dyp (task, subtask, ncd, pcf, ini)
exec_suk_task_a_dyp (task, subtask, ini)
is_busy_a_dyp ()
get_powrep_a_dyp (task, subtask)
get_powmap_a_dyp (task, subtask, resolution)
```

exec_task_a_dyp inicia la ejecución de una tarea de estimación de potencia, dados el archivo NCD post emplazamiento y rutado, el archivo PCF de restricciones físicas (por ejemplo, posiciones de los pines de entrada y salida) y el archivo INI con los parámetros definidos por el usuario. En el archivo INI se especifican tanto los parámetros de la técnica estadística (error admisible en la estimación, nivel de confianza, etc.), como otros propios de esta herramienta (nivel en la cantidad de información que se debe almacenar, tiempo mínimo de un *glitch*, etc.).

La estimación de consumo se realiza mediante la estimación de actividad de los nodos según los parámetros definidos en el archivo INI y la obtención de las capacidades de los nodos individuales según los archivos NCD y PCF. Pero por ejemplo, si se quiere estimar la potencia de un diseño para diferentes especificaciones de actividad a las entradas del circuito, se tendrán nuevos archivos INI siendo la obtención de capacidades idéntica. La tarea de obtención de capacidades tiene un tiempo de ejecución que no es para nada

despreciable. Por lo tanto, se tiene el método *exec_sub_task_a_dyp* para estimar el consumo de potencia de circuitos para los cuales ya se han obtenido las capacidades de los nodos.

El método *is_busy_a_dyp* indica al cliente que lo invoca, si la herramienta de estimación está en ejecución u ociosa. Finalmente, *get_powrep_a_dyp* y *get_powmap_a_dyp* permiten recuperar los resultados del proceso de estimación: un reporte con el consumo del circuito y de todos sus nodos, y archivos con información para construir mapas de consumo medio para diferentes resoluciones respectivamente. En estos mapas la FPGA se divide en rectángulos de una cantidad de CLBs de lado (resolución) y la potencia de todas las pistas con origen en cada zona se suman.

3.1.1 Detalles de codificación

Este servicio está programado en Python [5] por la relativa sencillez con que se puede desarrollar. Hay varias librerías que proveen una API para abstraer SOAP en Python pero se seleccionó SOAPpy por su facilidad de uso y difusión [6]. A continuación se presenta una versión simplificada de *exec_task_a_dyp*:

```
1   ...
2   from SOAPpy import *
3   ...
4   def exec_task_a_dyp (task, subtask, ncd_filename, ncd_file,
    ini_filename, ini_file, pcf_filename, pcf_file):
5     # PE working directory
6     PE_dir = "C:\\exec-PE\\"
7
8     # Download the files in the PE working directory
9     # Save ncd_file in the PE working directory
10    r = save_file (PE_dir+ncd_filename, ncd_file,".ncd")
11    if r=="error":
12            # Ends the function call with error
13            return "error"
14
15    # The same for the ini_file and pcf_file
16    ...
17    # Generates the Basic PE initialization script
18    PE_ini_file = file (PE_dir + "pe-ini.do", "w")
19    ...
20    # netgen generates the VHDL simulation model
21    # from the NCD and PCF files
22    if use_pcf:
23            netgen_command = "exec netgen -sim -ofmt vhdl -w -
    pcf " + pcf_file_name + " " + design_name + " " +
    vhdl_file_name
24    else:
25            netgen_command = "exec netgen -sim -ofmt vhdl -w "
    + design_name + " " + vhdl_file_name
26
27    PE_ini_file.write(netgen_command + "\n")
28
29    PE_ini_file.write("source estimate.tcl\n")
30    PE_ini_file.write("powerEstim\n")
31    ...
```

```
32    PE_ini_file.close()
33    ...
34    # Starts Modelsim to run the Power Estimation Task
35    vsim = 'vsim -c -do ' + PE_dir + 'pe-ini.do'
36    # Run the time-consuming PE in a separate thread while
    allowing returning this function
37    A = PE_Thread(vsim)
38
39    return 'ok'
40  ...
41  server = SOAPServer(("150.244.57.94", 8080))
42  server.registerFunction(exec_task_a_dyp, "PE")
43  ...
44  server.serve_forever()
```

Primero se copian los archivos de entrada en un directorio de trabajo. La estimación de consumo se hace simulando el circuito en un simulador comercial (véase [1]). Éste se invoca por línea de comandos con un *script* (.do) como parámetro (líneas 35 y 37). Dicho *script* hace procesamientos anteriores a la simulación y se genera dinámicamente con el código presentado a partir de la línea 18. Un punto importante es que debe generarse un modelo VHDL post emplazamiento y rutado a partir del archivo NCD (líneas 22 a 27) [7]. La línea 30 crea la línea de código que invoca el procedimiento que controla la estimación de consumo de potencia. Otros *scripts* auxiliares inicializan la simulación (compilación, elaboración, etc.). *pe-ini.do*, tiene un contenido como se muestra a continuación:

```
cd C:/exec-PE/

exec netgen -sim -ofmt vhdl -w -pcf topFIRDA.pcf topFIRDA
topFIRDA_timesim.vhd

source estimate.tcl

powerEstim

quit -f
```

En la línea 37 se invoca el simulador pero en un hilo de ejecución separado. Esto se hace para no bloquear al cliente del servicio Web que recibe inmediatamente la confirmación de la recepción de los archivos de entrada, mientras el servidor hace su trabajo.

3.2 Servicio Principal de Estimación de Consumo

El servicio principal de Estimación de Consumo invoca métodos del servicio básico y provee autenticación y administración de tareas. El servicio básico, es un servicio Web de uso privado y no puede accederse desde el exterior. El servicio principal provee 5 métodos a clientes externos:

```
req_a_dyp (username, password, ncd, pcf, ini)

sub_req_a_dyp (username, password, task, ini)

task_done_a_dyp (username, password, task, sub_task)

get_powrep_a_dyp (username, password, task, sub_task)

get_powmap_a_dyp (username, password, task, sub_task)
```

Mediante *req_a_dyp* se da de alta un requerimiento de estimación de consumo para un usuario registrado en el sistema. Este método devuelve el número de tareas y sub tareas pendientes que deben realizarse antes que la actualmente solicitada pueda iniciarse. También se devuelve un identificador de tarea.

sub_req_a_dyp sirve para dar de alta una sub tarea basada en otra previamente requerida. Para realizarla, se usan los resultados obtenidos a partir de los archivos NCD y PCF previamente dados.

task_done_a_dyp indica si la tarea especificada está completa o no. Si bien se provee esta función, el método preferido para que el cliente sepa que la tarea requerida se completó, es mediante una notificación por parte del servicio. Esta notificación puede ser un correo electrónico enviado a la dirección que el usuario especificó cuando se registró, o a un método que tendría el cliente también en un servicio Web. Como se ve, más que hablar de clientes y servidores, debería hablarse de servicios que colaboran entre sí.

get_powrep_a_dyp y *get_powmap_a_dyp* devuelven los archivos con los resultados de la estimación de consumo igual que en el servicio básico.

3.3 Cliente Web

El cliente Web implementado en este trabajo es una demostración en vivo del uso del servicio de estimación de potencia. Cualquier aplicación externa puede usar el servicio independientemente del cliente Web. Sin embargo todo usuario debe registrarse llenando el formulario correspondiente. En el caso de un usuario humano, debe decir una dirección de correo electrónico para que se le notifique cuando sus tareas se completan. Si se trata de una aplicación externa, puede especificar su dirección para que este servicio Web le invoque el método *task_done_a_dyp*.

3.3.1 Detalles de codificación

Para el cliente se usó PHP[8] y la librería NuSOAP[9] para abstraer la comunicación mediante SOAP. El siguiente fragmento de código muestra lo esencial del envío de un mensaje SOAP al servidor.

```
<?php
 ...
 require_once('nusoap.php');
 $parameters=array(
        "task"=> $TaskNumber,
        "subtask"=> 0,
        "ncd_filename" => $Ncd_File_Name,
        "ncd_file" => $Coded_Ncd_Contents,
        "ini_filename"=> $Ini_File_Name,
        "ini_file"=> $Coded_Ini_Contents,
        "pcf_filename"=> $Pcf_File_Name,
        "pcf_file"=> $Coded_Pcf_Contents);
 $cliente=new soapclient('http://150.244.57.94:8080/');
 $out=$cliente->call('exec_task_a_dyp', $parameters, "PE");
 ...
?>
```

Básicamente, se crea un *array* de parámetros (líneas 4 a 12) y se crea una instancia del cliente para realizar la comunicación vía SOAP (Línea 13). Luego se hacen llamadas al servidor (línea 14).

4 Conclusiones

Se tiene la primera versión de un servicio Web que permite interactuar con una herramienta EDA de estimación de potencia media para FPGA. Se puede interactuar con él remotamente, desde clientes escritos en una amplia variedad de lenguajes como PHP, Python, C++, Java, Delphi, etc.

El servicio se está probando en el laboratorio y está resultando de utilidad para automatizar las tareas de estimación. Se espera que también sea útil para otros usuarios externos. En ese caso se puede acceder al servicio en la dirección que aparece en las líneas 41 y 13 de los fragmentos de código en Python y PHP respectivamente. También se puede acceder a la página Web del servicio en http://150.244.57.94/PE-web/.

Este trabajo también puede ser útil como ejemplo para investigadores que tengan herramientas software o hardware y las quieran poner accesibles en Internet.

Agradecimientos

Este trabajo ha sido financiado mediante el Proyecto TIC2001-2688-C03-03 del Ministerio de Ciencia y Tecnología de España. También Se obtuvieron fondos adicionales del Proyecto 658001 07T/0052/2003-3 de la Consejería de Educación de la Comunidad de Madrid, España.

Referencias

1. Todorovich E., M. Gilabert, G. Sutter, S. Lopez-Buedo, and E. Boemo: A Tool for Activity Estimation in FPGAs. Lecture Notes in Computer Science, Vol. 2438. Springer-Verlag, Berlin Heidelberg 2002, pp. 340-349
2. Todorovich, E., G. Sutter y E. Boemo: Estimación de Actividad para FPGA Basada en una Técnica Estadística. JCRA 2003, Madrid, España, 10-12 de octubre de 2003. Pág. 217-224
3. Todorovich E., E. Boemo, F. Cardells, J. Valls: Power Analysis and Estimation Tool integrated with XPOWER. Poster at Twelfth ACM FPGA 2004
4. Curbera F., M. Duftler, R. Khalaf, W. Nagy, N. Mukhi, S. Weerawarana: Unraveling the Web Services Web: An Introduction to SOAP, WSDL, and UDDI. IEEE Internet Computing, March-April 2002, pp. 86-93
5. Lutz M. and D. Ascher: Learning Python. 2nd Edition. O'Reilly & Assosiates, 2004
6. Ullman C. et al: SOAPpy. http://pywebsvcs.sourceforge.net/
7. Xilinx Inc.: Development System Reference Guide, Chapter 23. 2003
8. Ratschiller T., T. Gerken: Web Application Development with PHP 4.0. SAMS, 2000
9. Ayala D., S. Nichol et al: NuSOAP. http://dietrich.ganx4.com/nusoap/