REVISTA COLOMBIANA DE COMPUTACIÓN
Volumen 17, número 1
Págs. 122 - 144

Estimación de ancho de banda disponible por generación de paquetes de prueba a través de NetFPGA

Diego A. Reyes*, César D. Guerrero*

Fecha de recibido: 20/01/2016 Fecha de aprobación: 29/03/2016

Resumen

Estimar el ancho de banda disponible requiere mediciones y análisis adecuados de los flujos de paquetes producidos donde, errores asociados a la generación y recepción de paquetes de prueba son complejos de controlar y en algunos casos están fuera del alcance de las herramientas de estimación. La marcación incorrecta de los paquetes, el uso de tarjetas de red estándar, el no uso de la capacidad real de transmisión, la recepción de los paquetes fuera de orden, la replicación, la manipulación inadecuada de las herramientas de estimación, entre otros, obstaculizan la correcta medición del ancho de banda disponible. A su vez, el rendimiento y precisión impuestos por el sistema operativo, el cual no permite dar prioridad al proceso a la generación, envío y recepción de los paquetes de prueba, son procesos complejos de controlar y en algunos casos están fuera del alcance de las herramientas de estimación. A continuación, se presenta el diseño de una herramienta que permite la generación de paquetes de prueba a nivel de hardware utilizando NetFPGA. Esta plataforma permite modificar su comportamiento por los arreglos lógicos programables que posee para generar y transmitir paquetes a velocidad del enlace e interactúa con la herramienta software de estimación de ancho de banda disponible llamada Traceband. Su diseño contiene diversos módulos que realizan la generación de los paquetes de prueba, la marcación del timestamp de envío de los paquetes de prueba y describe la forma en que la NetFPGA y Traceband se comunican a través del módulo Traceband Register.

Palabras clave: *Ancho de Banda Disponible, Generación de paquetes de prueba, NetFPGA, Traceband.*

Abstract

Estimate the available bandwidth required of appropriate measurements and analysis of packet flows produced and the measures contained in them, errors associated with generating and receiving test packets cannot be controlled and are outside the scope of tools estimation. Incorrect timestamp of test packets,

* Universidad Autónoma de Bucaramanga. Grupo de Investigación en Tecnologías de Información – GTI. Avenida 42 No 48 – 11 Bucaramanga Colombia. {dreyes507, cguerrer}@unab.edu.co.

using standard network card, the not using the real capacity of transmission of the network interfaces, receiving packets out of order, replication, packet corruption, among others, hinder the correct measurement of available bandwidth. The interchange of process in the operating system ago to be added unnecessary time between sending test packets and therefore are added errors in estimating the available bandwidth. The purpose of this article is to design a mechanism that allows for the generation of test packets at the hardware level using a technology called NetFPGA. This platform allows you to modify their behavior by having programmable logic arrays to generate and transmit packets to the link speed and in turn, interact with the software tool called Traceband estimate. The results raised the various modules for generating test packets and dialing the timestamp sent to the RAM of the NetFPGA; In addition to describing how the NetFPGA and Traceband must communicate it is through Traceband Register module.

Keywords: *Available Bandwidth, Generation of test packets, NetFPGA, Traceband..*

1. Introducción

La creciente demanda de aplicaciones, productos y servicios en la red ha llevado a las empresas a proveer productos y servicios que no se vean limitados por velocidad de acceso y la capacidad de los enlaces de una red. Además, el crecimiento continuo a la par de la implementación de nuevas tecnologías ha vuelto complejas las redes, tanto en la configuración como en la distribución del tráfico. La anterior problemática, ha despertado un campo de investigación en el cual los investigadores en el área de telemática, pueden desarrollar e implementar herramientas que permitan la medición de parámetros que al final pueden ser utilizados para optimizar una conexión de red.

Métricas como el ancho de banda disponible, latencia, máximo rendimiento de una transmisión TCP, *One-Way Delay,* entre otros, son parámetros que pueden ser medidos y son de gran importancia a la hora de optimizar una conexión de red. Dentro de las métricas presentadas, la medición del ancho de banda disponible de extremo a extremo es de gran importancia, debido a que el ancho de banda es finito, por lo tanto es limitado y la demanda por su uso crece cada vez más. En este trabajo de maestría, se enfocará al estudio de la estimación de ancho de banda disponible, métrica que permite obtener la capacidad no utilizada de un canal, el cual puede utilizarse para mejorar la utilización del mismo en servicios como telefonía, *streaming* entre otros.

Las herramientas desarrolladas en la actualidad que permiten estimar el ancho de banda disponible de extremo a extremo, cuentan con la capacidad de adaptarse a las condiciones cambiantes de la red. Aunque las técnicas desarrolladas han logrado una buena adaptación a los

diferentes escenarios donde fueron aplicadas. Las herramientas desarrolladas a nivel de software evidencian una gran desventaja debido a la dependencia de la velocidad y restricciones del sistema operativo de la máquina el cual es incapaz de priorizar los procesos de generación y recepción de los paquetes de prueba, por lo que realiza la conmutación entre procesos que impide que los tiempos de envío y recepción de los paquetes de prueba sean los planeados por la herramienta de estimación. Sobre lo anterior, errores asociados a la generación y recepción de paquetes de prueba son complejos de controlar y en algunos casos están fuera del alcance de cualquier herramienta de estimación.

NetFPGA es una plataforma de hardware y software abierto de red orientado al diseño y construcción sistemas de redes de alto rendimiento. En NetFPGA se han desarrollado proyectos relacionados a la generación y recepción de paquetes de prueba e incluso se han desarrollado aplicaciones que permiten desarrollar la comunicación entre hardware y software lo que le permite complemento ideal para hacer los estimadores de ancho de banda disponible. Llevar el proceso de generación y recepción de los paquetes de prueba de un estimador de ancho de banda disponible a velocidad de línea evitaría las restricciones impuestas por el sistema operativo mencionadas previamente. Realizar un envío preciso de paquetes de prueba a través de NetFPGA según lo determine un estimador de ancho de banda disponible, es el tema a tratar en el presente artículo.

2. Trabajos relacionados

Los estimadores de ancho de banda disponible (en adelante ABET), son aplicaciones de tipo software que se han desarrollado implementando los enfoques *Probe Gap Model* (en adelante PGM) y *Probe Rate Model* (en adelante PRM). Para comprender cómo un ABET llega a los resultados obtenidos, requiere de mediciones y análisis adecuados de los flujos de paquetes producidos por el ABET y las medidas recogidas [1]. Llegar a una correcta estimación requiere la comprobación en entornos de prueba pertinentes para evaluar la precisión de los ABETs en un rango de condiciones controladas y correctamente caracterizadas. Dentro de las principales limitaciones encontradas tras analizar diferentes ABETs presentados en la Tabla 1, las cuales comprometen el rendimiento, el desempeño y la precisión en la estimación están:

- Rendimiento y precisión impuestos por el sistema operativo: estos problemas están asociados a la imposibilidad para dar prioridad al proceso de envío de paquetes, lo que ocasiona que los tiempos de transmisión no correspondan a los tiempos solicitados especificados por el ABET.

- Uso de interfaces de red estándar no diseñadas para generar y enviar paquetes de prueba de forma precisa, lo que produce registros incorrectos en los tiempos de salida o llegada de los paquetes y en algunos casos no permiten usar la capacidad real de transmisión de las interfaces de red.
- Recepción fuera de orden, replicación, corrupción de paquetes, el comportamiento de los servicios de colas FIFO o no-FIFO de los routers.
- Sobrecarga de la red por la transmisión de los paquetes de prueba.
- El comportamiento del tráfico cruzado y la especificación de un conjunto de parámetros antes de ejecutar la aplicación de estimación, afectan la precisión de los ABETs.

Tabla 1. Resumen de estimadores de ancho de banda disponible.

Nombre	Autores	Año	Técnica		Tipo		Metodología
			PRM	PGM	SW	HW	
Cprobe [2]	Carter, Crovella	1996		X	X		Trenes de paquetes
Delphi [3]	Ribiero, Riedi, Sarvotham, Coates, Brent, Baraniuk	2000		X	X		Trenes de paquetes
TOPP [4]	Melander, Bjorkman, Gunningberg	2000	X		X		Trenes de paquetes
Pathload [5]	Jain, Dovrolis	2002	X		X		Self-Loading Periodic Streams
IGI/PTR [6]	Hu, Steenkiste	2002		X	X		Self-Loading Periodic Streams
Spruce [7]	Strauss, Katabi, Kaashoek	2003		X	X		Trenes de paquetes
Abing [8]	Navratil, Cottrell	2003		X	X		Trenes de paquetes
pathChirp [3]	Ribeiro,Riedi,Baraniuk, Navratil,Cottrell	2003	X		X		Self-Loading Packet Chirps
Netest [9]	Jin, Tierney	2003	X		X		Trenes de paquetes
ProbeGap [10]	Lakshminarayanan, Padmanabhan, Padhye	2004		X	X		Trenes de paquetes
DietTOPP [11]	Johnsson, Melander, Bjŏrkman	2004	X		X		Trenes de paquetes
PathMon [12]	Kiwior, Kingston,Spratt	2004	X		X		Self-Loading Packet Chirps
PoissonProb [13]	Xin	2005	X		X		Trenes de paquetes
Yaz [1]	Sommers, Barford, Willinger	2006	X		X		Trenes de paquetes
BART [14]	Ekelin, Nilsson, Hartikainen, Johnsson	2006	X			X	Trenes de paquetes
MoSeab [15]	Zhang, Luo, Li	2006		X	X		Trenes de paquetes
IGMPS [16]	Ali, Lepage	2007		X	X		Trenes de paquetes
FEAT [17]	Wang, Cheng	2008	X		X		Trenes de paquetes
Assolo [18]	Goldoni, Rossi, Torelli	2009	X		X		Reflected Exponential Chirp
PathAB [19]	Roy	2009	X	X	X		Trenes de paquetes
Traceband [20]	Guerrero, Labrador	2010		X	X		Trenes de paquetes

Un ejemplo del funcionamiento de un ABET se describe en la Fig. 1. En la Fig. 1, el ABET envía un par de paquetes (o tren de paquetes) de prueba con una separación entre inicial entre paquetes Δ_{in} desde el host abet_snd al host abet_rcv. A continuación, los paquetes de prueba interactúan con el tráfico cruzado representado por el envío de paquetes desde el host ct_snd al host ct_rcv, pasan por el cuello de botella (que es a la vez el vínculo estrecho de menor capacidad), hasta que abet_rcv recibe los paquetes de prueba con una separación entre paquetes Δ_{out} mientras que ct_rcv recibe el tráfico cruzado. Sin embargo, para comprender cómo un ABET llega a los resultados obtenidos, se requieren de mediciones y análisis de los flujos de paquetes producidos por el ABET y las medidas recogidas [21].

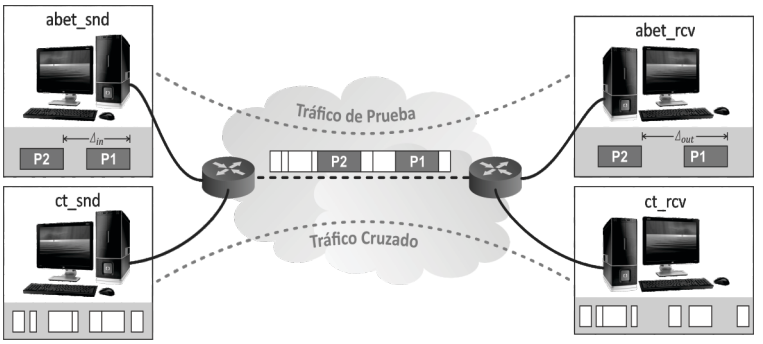

Fig. 1. Funcionamiento de un estimador de ancho de banda disponible.

2.1 Comparación de las herramientas de estimación de ancho de banda disponible

Los ABETs analizados en la Tabla 1, evidencian que PRM es una metodología cuyas estimaciones de ABw son más precisas que PGM pero, debido a que utilizan el principio de congestión autoinducida, el tiempo para lograr una estimación es considerablemente más alto en comparación con PGM, así como la posibilidad de causar una interrupción en los espaciamientos de los paquetes de prueba e invalidar la medición. PGM por otro lado, es menos intrusivo que PRM, cada par de paquetes de prueba logra una estimación y los tiempos de iteración de la herramienta son cortos. Aunque Traceband [20] logra estimaciones similares a las de Pathload [5], en general, las herramientas basadas en PGM son menos precisas, tienen problemas para calcular la dispersión de los paquetes y requieren el conocimiento previo de la capacidad del *tight link* para lograr una adecuada estimación de Abw.

2.2 *Traceband*

Traceband [20], es una herramienta cliente-servidor (emisor-receptor) escrito en ANSI C que utiliza modelos ocultos de Markov (del inglés *Hidden Markov Model*, en adelante HMM) para proporcionar estimaciones de ABw de forma rápida, continua y precisa, bajo el modelo PGM descrito en las técnicas para la estimación del ancho de banda disponible. En Traceband, un tren de pares de paquetes de prueba se envía desde la aplicación cliente al servidor a la velocidad del *tight link*. Después de interactuar con el tráfico cruzado en el *tight link,* cada par de paquetes en el tren proporcionará un único valor de dispersión que se constituye en una observación en la secuencia del HMM. En el servidor, la aplicación procesa la secuencia de observación, estima la secuencia de estados que genera las observaciones y proporciona una única estimación promediada del ancho de banda disponible. Según [20], los resultados experimentales demuestran que el implementar HMM permite a *Traceband* proporcionar estimaciones rápidas y precisas con una tasa de tráfico de prueba baja, con la capacidad de reaccionar con rapidez y precisión a las variaciones del tráfico cruzado, como los presentes en enlaces cargados con tráfico a ráfagas.

3. Metodología

La metodología utilizada está compuesta de 5 fases las cuales se presentan en la Fig. 2. En la Sección 3.1, se dará a conocer la importancia de implementar la generación de paquetes de prueba en la NetFPGA. Se estudiarán los diferentes proyectos orientados a la generación tráfico y posterior a ello se seleccionará la herramienta que mejor se adapte a las necesidades del presente proyecto. En la Sección 3.2, se analizará cómo es el proceso de generación de paquetes de prueba en la herramienta de estimación de ancho de banda disponible, en este caso *Traceband*. Posteriormente en las Secciones 3.3, 3.4 y 3.5 se describirán las especificaciones que se le harán *Traceband* y a la NetFPGA, incluida la forma como se establecerá la comunicación entre ambas soluciones mediante la implementación de un register I/O que permita dicha tarea.

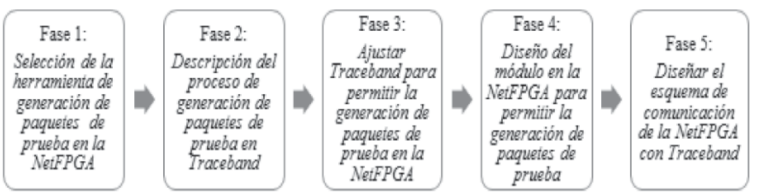

Fig. 2. Fases del proceso investigativo.

3.1 Selección de la herramienta de generación de paquetes de prueba en la NetFPGA

Dado que el rendimiento y precisión impuestos por el sistema operativo asociados a la imposibilidad para dar prioridad al proceso de envío de paquetes, ocasiona que los tiempos de transmisión no correspondan a los tiempos solicitados por las herramientas de estimación del ABw, además de que las interfaces de red estándar no están diseñadas para generar paquetes de prueba y enviarlos de forma precisa, estos problemas pueden solucionarse en parte implementando hardware de red reconfigurable, como es el caso de NetFPGA. A continuación, se presentan las diferentes herramientas desarrolladas para NetFPGA orientadas a la generación de paquetes de prueba:

3.1.1 NetFPGA-Packet Generator

El *NetFPGA-Packet Generator* (NPG) ofrece un generador de paquetes, así como una herramienta para captación de datos [22]. Está diseñado para ser utilizado por cualquier persona que quiera inyectar paquetes en una red y/u observar los paquetes que salen de una red. La arquitectura del NPG utiliza la NIC de referencia de la NetFPGA. Las principales modificaciones realizadas para el Generador de Paquetes se realizaron en el *User Data Path* y se presentan a continuación.

- Se añadió un módulo de *Time Stamp* (marcado de los paquetes) antes del MAC RxQ. Esto permite que los paquetes entrantes sean marcados a medida que se reciben por el hardware.
- Se agregó el módulo *Packet Capture* dentro del *User Data Path*, el cual realiza dos funciones: la compilación de estadísticas al momento de generación y captura de paquetes (como el número de paquetes recibidos y el tiempo total de captura) y capturar las marcas de tiempo de los paquetes cuando la función de generación/captura está desactivado.
- Se modificó el *Output Queues* para operar con doce colas de salida y no con ocho como en la NIC de referencia. Esto permite que las cuatro nuevas colas se utilicen para almacenar los datos PCAP, para luego ser transmitidos cuando el Generador de Paquetes está activado.
- Se añadió el módulo *Packer Generator Output Sel* el cual determina cuál de las 12 colas de salida debe ser conectado a las ocho colas de salida para el siguiente módulo.
- Cada una de las ocho colas de salida de referencia cuenta con un limitador de tasa y un módulo de retardo. Esto permite que al PTG agregar un retardo a la velocidad de cada una de las ocho colas de salida individualmente.

3.1.2 *NetThreads*

NetThreads [23] es una plataforma para el procesamiento de paquetes que permite ejecutar software en la NetFPGA escrito en C. Dentro de sus características está el procesamiento en 4 vías, dos procesadores con cuatro hilos cada uno (los cuales ejecutan una instrucción cada 4 ciclos de reloj) y un *pipeline* de 5 etapas. Los procesadores en NetThreads encajan en el *pipeline* utilizado en la NIC de referencia. En el *pipeline*, los procesadores reemplazan el módulo *output port lookup*, ubicándose entre los módulos *Input Arbiter* y *Output Queues* como se muestra en la Fig. 3. En la NIC de referencia, hay cuatro colas de entrada RxQ CPU para los paquetes copiados a través del bus PCI del host. Sin embargo, en *NetThreads*, sólo una de estas colas está conectado al *Input Arbiter* (por lo tanto, las tres colas restantes no están conectadas y paquetes enviados a ellos nunca serán leídos por *NetThreads*, mientras que los cuatro MAC RxQ si están conectados al árbitro de entrada). .

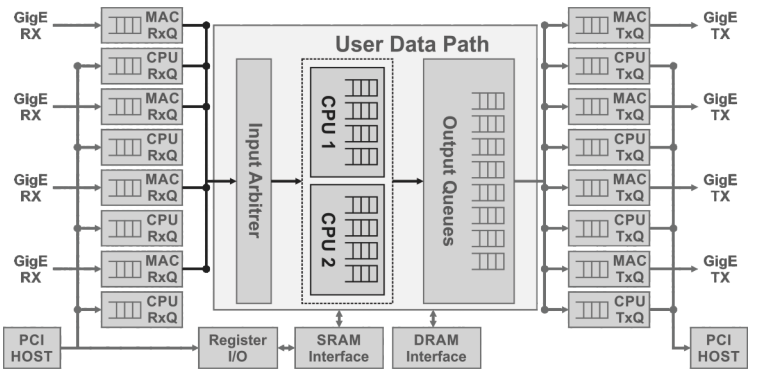

Fig. 3. *Pipeline* de NetThreads.

NetThreads, incluye un generador de paquetes llamado *pktgen hereafter* [23], el cual recibe paquetes enviados desde el controlador de la NetFPGA para su construcción y envío a través de los puertos Ethernet de manera precisa. A diferencia de NPG mencionados previamente, Caliper no es un diseño realizado directamente sobre la NetFPGA, sino que es una aplicación desarrollada para correr sobre NetThreads y está orientado a controlar con precisión los tiempos de transmisión de los paquetes que se crean en el equipo *host* a través de la NetFPGA[23]. Al ser una aplicación que funciona directamente sobre la NetFPGA, muchas de las limitaciones presentadas en la Sección 2 referentes a las aplicaciones de tipo software a través del sistema operativo se ven solventadas a llevarse en el nivel de hardware.

3.1.3 Selección de la herramienta a implementar en la NetFPGA

Dado que se requiere integrar una herramienta para la estimación del ancho de banda disponible escrita en C con el hardware de la NetFPGA, *NetThreads* resalta sobre el NPG debido a que:

- Puede ejecutar aplicaciones en C. En este caso, se utilizará la herramienta para la generación de ancho de banda disponible *Traceband*.
- Permite generar paquetes de prueba según los requerimientos de la aplicación escrita en C. En comparación, *NetFPGA-Packet Generator* no construye su propio tráfico; sólo replica paquetes previamente capturados en un archivo PCAP.

3.2 Descripción del proceso de generación de paquetes de prueba en Traceband

El presente artículo trata sobre la generación de paquetes de prueba. Por lo tanto, requiere una mayor profundización en el funcionamiento de la herramienta de estimación del ABw. La aplicación *Traceband Sender* se encarga de generar los paquetes de prueba y visualizar la estimación del ancho de banda disponible calculada en el servidor *Traceband receiver*, ejecuta ciclos de diez estimaciones. En la primera estimación la herramienta envía 50 pares de paquetes UDP 1498 bytes de longitud por el puerto 3504. Las nueve estimaciones restantes se realizan con 30 pares de paquetes UDP cada uno. Esta reducción es posible ya que HMM tiene la capacidad de aprender la dinámica del ABw con una muestra inicial y mantener el modelo actualizado con muestras de tamaño reducido[20].

Traceband utiliza diferentes valores para los tiempos de *intra-gap* e *inter-gap* de los pares de paquetes. *Intra-gap* hace referencia al tiempo entre los dos paquetes de cada par de paquetes. El *intra-gap* o Δ_{in} se establece igual al tiempo de transmisión de un único paquete de prueba en el *light link.* De esa manera, el par de paquetes de prueba será capaz de captar el tráfico transversal en la cola, si lo hubiera. *Inter-gap* se refiere al tiempo entre los pares de paquetes, es decir, el tiempo entre el segundo par de paquetes de prueba $i-1$ y el primer par paquete de prueba i. Estos tiempos se obtienen usando la función *gettimeofday*( ), y sus valores se envían al servidor en la carga útil del paquete.

3.3 Especificaciones para que Traceband permita la generación de paquetes de prueba en la NetFPGA

Dado que *Traceband* es una herramienta de tipo software y *NetThreads* es de tipo hardware que permite ejecutar software en la NetFPGA, es necesario que se realicen modificaciones para lograr una adecuada comunicación entre ellas. En esta sección, se darán a conocer las especificaciones que tendrá *Traceband Sender* para que pueda generar paquetes de prueba a través de NetFPGA de manera precisa.

Las especificaciones se clasificarán en dos categorías: fuera de la NetFPGA y dentro de la NetFPGA, fundamentado en que parte de las tareas que realiza Traceband no necesariamente tienen que ser realizadas directamente dentro de la NetFPGA. Por ejemplo, parámetros de configuración, calcular el tiempo que tarda *Traceband Sender* en lograr una estimación, calcular la sobrecarga del enlace, entre otros, le restarían capacidad de procesamiento a la NetFPGA debido a que se subutilizaría el procesamiento multi-hilo de la tarjeta en tareas diferentes a la generación de paquetes de prueba.

3.3.1 Especificaciones de *Traceband* Sender para ser ejecutadas fuera de la NetFPGA

Dentro de las actividades que *Traceband Sender* debe realizar fuera de la NetFPGA, están:

1. Establecer parámetros de configuración de la herramienta tales como: capacidad del *tigth link,* tamaño de los paquetes de prueba, número de trenes de prueba, número de pares de paquetes por tren, espacio entre paquetes, espacio entre pares de paquetes y espacio entre trenes de paquetes.
2. Dado que los paquetes de prueba no se generarán desde la aplicación, *Traceband Sender* deberá indicarle a la NetFPGA la forma como debe crearlos. La mejor forma de lograrlo es por medio de una tabla que contenga la información necesaria que le permita a la NetFPGA construir y enviar los paquetes de prueba en los tiempos establecidos por la aplicación de lo más exacto posible.
3. Establecer la comunicación con la NetFPGA.
4. Enviar la tabla para la generación y envío de los paquetes de prueba.
5. Recibir la estimación del ancho de banda disponible proveniente del host del *Traceband Receiver*.
6. Visualizar la estimación del ancho de banda disponible, calcular el tiempo que tardó la estimación y la sobrecarga del enlace.

3.3.2 Especificaciones de *Traceband Sender* para ser ejecutadas dentro de la NetFPGA

En la sección anterior, se planteó la necesidad de generar los paquetes de prueba dentro de la NetFPGA. En la Sección 3.1.2, se encontró que NetThreads tiene la capacidad de ejecutar código en C dentro de la NetFPGA utilizando la NIC de referencia como base e implementando dos procesadores de cuatro hilos cada uno. Además, en dicha sección, fue presentado *pktgen hereafte*, el generador de paquetes de *NetThreads*, el cual está escrito en C y permite generar paquetes de forma precisa.

Para que *Traceband* pueda ejecutarse dentro de la NetFPGA utilizando *NetThreads* y establecer una correcta comunicación con la parte de *Traceband Sender* que se encuentra fuera de la NetFPGA, es necesario establecer las siguientes especificaciones:

1. Cargar la tabla creada por *Traceband* fuera de la NetFPGA con las especificaciones de generación de los paquetes de prueba.
2. Aprovechar los dos procesadores de cuatro hilos de NetThreads, para implementar la parte del código en C de *Traceband* que genera los paquetes de prueba.
3. Enviar los paquetes de prueba en los tiempos establecidos por *Traceband Sender.*

3.4 Especificaciones para los módulos de la NetFPGA que permiten la generación y envío de paquetes de prueba

Así como fue necesario modificar *Traceband*, fue necesario realizar modificaciones al *Pipeline* de NetThreads presentado en la Sección 3.1.2, para que, en conjunto, ambas herramientas puedan interactuar entre sí para lograr una correcta generación y envío de los paquetes de prueba. Dentro de las modificaciones que se necesita hacer en el *pipeline* de NetThreads para que interactúe con *Traceband Sender,* además de generar y enviar los paquetes de prueba, están:

1. Crear un registro de entrada y salida en la NetFPGA que permita la comunicación entre el *sender host.*
2. Utilizar los dos procesadores de 4 hilos de NetThreads, para cargar la tabla de generación de paquetes de prueba, para su posterior generación y envío.
3. Marcar el *timestamp* de salida de los paquetes de prueba cuando estos van a salir por cada uno de los puertos *Ethernet* hacia el *receiver host.*

4. Recibir la estimación del ABw y marcar el *timestamp* de llegada de dicha información.
5. Enviar la estimación y los *timestamp* de salida y de llegada a la parte de *Traceband Sender* ubicada fuera de la NetFPGA para que visualice el ABw, además de calcular y visualizar el tiempo que tardó la estimación y el *overhead.*

3.5 Especificaciones para la comunicación entre *Traceband Sender* y la NetFPGA

Mencionado en la sección anterior, la comunicación entre *Traceband Sender* y la NetFPGA utilizando NetThreads, será realizada implementando un registro de entrada y salida o register I/O. Dentro de las funciones del register I/O que permite una interacción entre *Traceband* y la NetFPGA se describen a continuación:

1. Permitir el ingreso de tabla con la información necesaria para generar los paquetes de prueba y enviarlos en tiempos requeridos por *Traceband Sender.*
2. Enviar a la aplicación fuera de la NetFPGA los *timestamps* obtenidos a la hora de generar los paquetes de prueba con el fin de calcular el overhead generado por *Traceband.*
3. Enviar a la aplicación fuera de la NetFPGA la estimación del ancho de banda disponible proveniente del *receiver host* de *Traceband.*

4. Resultados

Con base en la metodología propuesta en la Sección3, a continuación, se desarrollará cada una de las fases de proceso investigativo propuestos en la Fig. 3, así la solución implementada que permite integrar una herramienta para la estimación del ancho de banda disponible, en este caso *Traceband*, con el hardware de red NetFPGA, utilizando para ello NetThreads.

4.1 Análisis de *Traceband*

Previamente en la Sección 3.3, se establecieron los parámetros de configuración que se requieren realizar tanto a *Traceband* como a NetThreads. A continuación, se presenta *Traceband_NF Sender* (entiéndase NF como NetFPGA). El diseño propuesto se encuentra en el *Host* y está compuesto de cuatro módulos como se muestra en la Fig. 4. Cada uno de los módulos se describirá en las secciones siguientes.

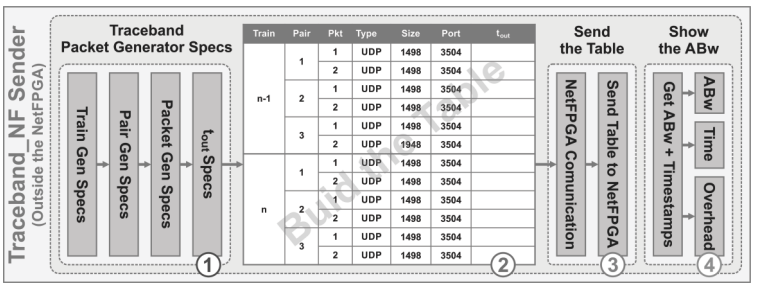

Fig. 4. Diseño de *Traceband_NF Sender*.

4.1.1 *Traceband Packet Generator Specs*

Traceband Packet Generator Specs, hace referencia a los datos de configuración inicial que requiere la versión original de *Traceband* para comenzar con la generación de los paquetes de prueba. En este caso, estos valores serán tomados para construir la tabla que posteriormente será enviada a la NetFPGA. Dentro de la información contenida en este módulo, está la capacidad del *tigth link,* el tamaño de los paquetes de prueba, el número de trenes y pares de paquetes por tren, el espacio entre paquetes, entre pares y entre trenes de paquetes, y *My Sleep*. *My Sleep* [20], hace referencia a una función utilizada por *Traceband* encargada de generar un espacio exponencial de tipo *Poisson* entre pares de paquetes (en milisegundos). Lo anterior es una característica de los estimadores de ancho de banda disponibles basados en PGM, que les permite lograr mejores estimaciones con poca sobrecarga del enlace.

4.1.2 *Build the Table*

Build the Table hace referencia al proceso que realizará *Traceband_NF Sender* para crear la tabla con la información necesaria que le permita a la NetFPGA construir y enviar los paquetes de prueba en los tiempos establecidos lo más exactos posibles. Aprovechando que los generadores de paquetes descritos en la Sección 3.1, los cuales tienen la capacidad de reproducir tráfico previamente capturado en un archivo PCAP, la tarea de *Build the Table* consiste en adaptar el código de *Traceband* para que, en lugar de crear y enviar los paquetes de prueba a través de una interface de red, los exporte a un archivo PCAP.

4.1.3 *Send the Table*

Send the Table hace referencia al proceso que realizará *Traceband_NF Sender* para enviar el archivo PCAP directamente a la NetFPGA. Lo

anterior se realizará creando un registro en el Pipeline de la NetThreads llamado *Traceband Register*. Las especificaciones del registro y la forma en que interactúan en la NetFPGA se presentarán posteriormente en la Sección 4.2.2.

4.1.4 *Show the ABw*

Show the ABw hace referencia al proceso que realizará *Traceband_NF Sender* para visualizar el ancho de banda disponible estimado en *Traceband Reveicer*. A su vez, calcular la sobrecarga y el tiempo de estimación con base en los *timestamps* (marcas de tiempo) calculados en la NetFPGA.

4.2 *NetTCB + Caliper*

NetTCB + Caliper, corresponde a una versión modificada de NetThreads orientada a interactuar con Traceband_NF Sender. Como se puede observar en la Fig. 5, se presentan cuatro modificaciones (una dentro del U*ser Data Path* y tres fuera de este) más la inserción de *Caliper* al *Pipeline* de NetThreads las cuales serán descritas a continuación.

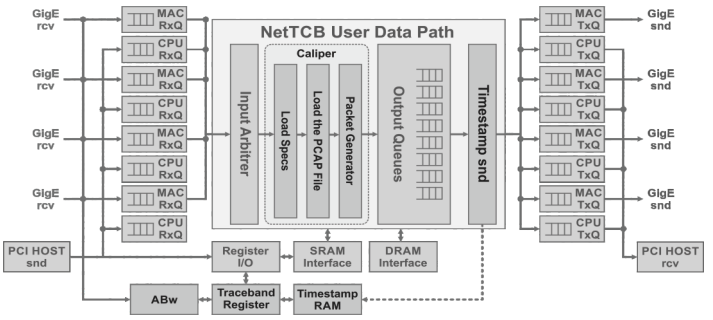

Fig. 5. *Pipeline* de *NetTCB*.**.**

4.2.1 Modificaciones realizadas dentro del User Data Path

La primera modificación realizada, se encuentra dentro del *User Data Path* y consiste en la inserción del módulo *Timestamp*_snd. Aunque genera un retardo por la creación de un módulo en la ruta de datos de la NetFPGA, se requiere agregar la información del *timestamp* a la carga útil de todos los paquetes UDP que salen hacia *Traceband Receiver* (*Traceband Receiver* requiere dos *timestamps* para lograr una estimación: el que se envía dentro de la carga útil del paquete UDP y el que se captura cuando el paquete llega al *receiver*). Para poder realizar la inserción entre el módulo, es necesario modificar el archivo user_data_path.v e incluir el módulo *Timestamp_snd* mediante la reconexión de "*wires*" entre los módulos de Verilog. Su resultado se presenta en la Fig. 6.

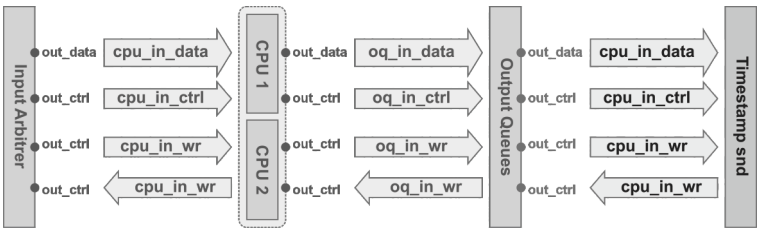

Fig. 6. Esquema de conexión del módulo *Timestamp_snd* de NetTCB.

En la Fig. 7, se describe el comportamiento del módulo *Timestamp_snd*, en el cual se puede observar los datos que serán almacenados en el módulo *Timestamp_RAM*.

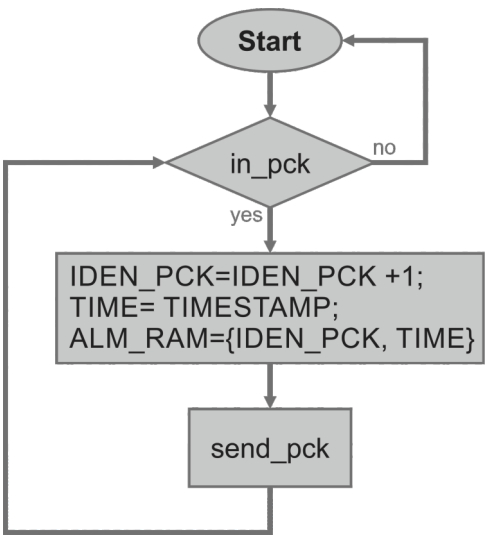

Fig. 7. Diagrama de flujo encargado del módulo *Timestamp_snd*.

4.2.2 Modificaciones realizadas fuera del *User Data Path*

Previamente, en la Fig. 5 se han presentado tres registros fuera del *User Data Path*. Su ubicación fue seleccionada fuera del *User Data Path* dado que no se generan retardos adicionales debidos al procesamiento de los paquetes de prueba. El primero de los registros creados se llama ABw y su objetivo es reconocer cuáles de los paquetes que ingresan a la ruta de datos de la NetFPGA, es un paquete UDP que contiene la estimación del ancho de banda disponible calculado por *Traceband Receiver*. La máquina de estados del módulo ABw, está representada en la Fig. 8, y contiene cinco estados. La función de *Next Packet* se encarga de anunciar la llegada de un nuevo paquete. Una vez el servicio de colas FIFO

contiene los datos y encabezados del paquete, le avisa al módulo anterior que está en proceso de lectura. En el estado *Extraction,* se encarga de guardar los valores de los campos como direcciones IP de origen, IP destino, puertos origen, puerto destino y el tipo de protocolo del paquete. En el estado *identification*, se verifica que el tipo de protocolo sea UDP, y que el puerto destino sea 3504. Si esto es verdadero, se pasa al módulo Get ABw para obtener el ancho de banda disponible.

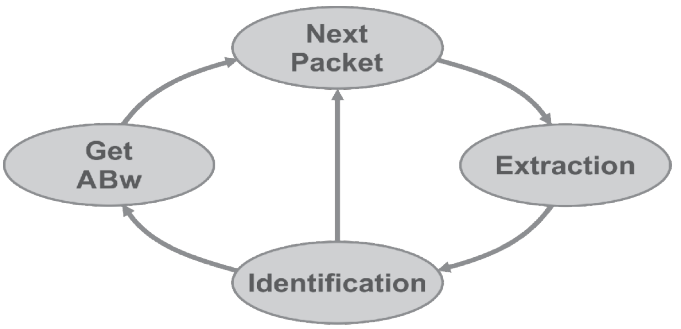

Fig. 8. Máquina de estados para el registro de ABw

El segundo de los registros creado fuera del *User Data Path,* se llama *Timestamp_Ram*, es de 64 bits, almacena 24575 datos y su objetivo es almacenar en la memoria el Timestamp de cada paquete de prueba generado. Dado que el máximo número de paquetes utilizado por *Traceband_snd* es de 320, no se sobrepasa la capacidad de la *Timestamp_Ram*. Una vez se haya terminado de marcar los paquetes y estén registrando los tiempos en que llegaron dichos paquetes, la aplicación debe solicitar el envío de la información y se realizará el barrido de la *Timestamp_Ram*. La forma como se almacenan los *Timestamp* en el registro, fue presentada previamente en la Sección4.2.1. Debido a que se requiere el uso de un registro de entrada y salida para poder realizar la comunicación entre *Traceband*, NetThreads y *Caliper*, fue necesaria la creación de un tercer registro llamado *Traceband Register*, el cual realiza las siguientes tareas:

1. Cuando la NetFPGA lo requiera, entregarle a Register I/O el archivo PCAP que permite generar los paquetes de prueba y enviarlos en tiempos requeridos por Traceband_NF Sender, así como los parámetros de configuración de Caliper.
2. Cuando Traceband_NF Sender lo requiera, acceder al módulo *Timestamp RAM* para obtener los valores de *timestamps* obtenidos dentro del *User Data Path.* Cuando los valores son obtenidos, los entrega a Register I/O para calcular el overhead generado por *Traceband*.
3. Cada vez que llegue una estimación del ABw proveniente de *Traceband Receiver,* enviarle a Register I/O dicha estimación para que Traceband_NF Sender la visualice.

4.3 Comunicación entre *Traceband_NF Sender* y NetTCB

Para dar respuesta a los objetivos planteados y a cada una de las fases de proceso investigativo propuestos en la Fig. 2, se hizo necesaria la integración de un estimador de ancho de banda disponible de tipo software (*Traceband*), de una herramienta de hardware que permite ejecutar aplicaciones escritas en C en la NetFPGA (*NetThreads*) y un generador de paquetes escrito en C para la NetFPGA (*Caliper*). A continuación, en la Fig. 9 se presenta el diagrama de bloques que muestra el ciclo de vida del proceso de generación de los paquetes de prueba, que va desde la generación del archivo PCAP, pasando por la generación de los paquetes de prueba hasta la visualización del ancho de banda disponible:

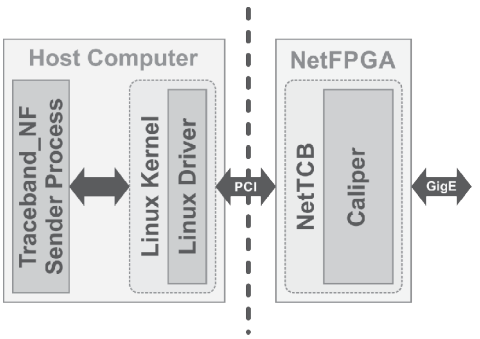

Fig. 9. Diagrama de bloques del funcionamiento de generación de paquetes de prueba en *Traceband_NF Sender* + NetTCB + *Caliper*.

En primer lugar, un proceso en el espacio de usuario o módulo del *kernel* en el *host* determina cuándo debe comenzar la estimación. Luego, una descripción de los trenes de paquetes de prueba, que contiene los tiempos de transmisión y toda la información necesaria. Para ensamblar el paquete se envía al controlador de la NetFPGA en un archivo PCAP. En el controlador, múltiples descripciones de los paquetes de prueba se combinan y se copian en la tarjeta NetFPGA. Combinando las descripciones se reduce el número de transferencias separadas requeridas, lo que permitirá posteriormente el envío de paquetes a la tasa de línea de 1 Gbps. A partir de ahí, las descripciones de los paquetes de prueba son procesados (en software) por los dos procesadores de NetTCB. Cada hilo de software ensambla los paquetes en la memoria de salida de NetTCB. A continuación, un hilo seleccionado inserta el *Timestamp* en cada paquete y los envía en el orden correcto en los tiempos de transmisión solicitados por *Traceband_NF Sender*. Posteriormente, el *pipeline* de la NetFPGA transmite los paquetes sobre el alambre.

Para lograr que NetTCB permita la carga de los paquetes de prueba en la NetFPGA, los autores de NetThreads modifican el controlador suministrado con la NetFPGA para soportar *Caliper*. Su tarea principal del controlador consiste en copiar las descripciones de paquetes usando DMA a través del bus PCI. También reúne las cabeceras de los paquetes y calcula las sumas de comprobación a través de un bus de 32 bits a 33 MHz con una velocidad de transferencia teórica superior de 1056 Mbps, aunque el número de transferencias de DMA entre el conductor y la NetFPGA está limitado 260 Mbps[23].

Como el proceso de generación y envío de los paquetes de prueba es realizado por *Caliper* (el cual se ejecuta como software en la plataforma NetTCB dentro de la NetFPGA), el controlador envía información que contienen las cabeceras de los paquetes de prueba con sus respectivos tiempos de transmisión. Posteriormente, *Caliper* prepara estos paquetes para la transmisión y los envía en los momentos apropiados. Para lograr un alto rendimiento de NetTCB, se aprovecha el procesamiento multihilo para maximizar el paralelismo. Una vez los paquetes de prueba son enviados por NetTCB, *Traceband Receiver* realiza el proceso de estimación y envío del ancho de banda disponible de regreso a la NetTCB. Cuando esta información llega de regreso, la información es almacenada por NetTCB y es enviada a través del controlador de la NetFPGA a Traceband_NF Sender para ser visualizada. En la Fig. 10, se presenta la integración entre Hardware y Software propuesta:

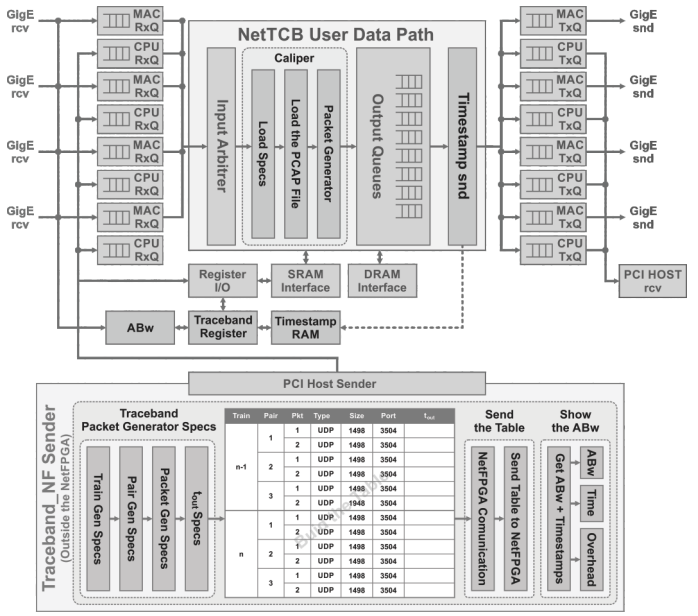

Fig. 10. Solución propuesta – Integración de *Traceband_NF Sender* + NetTCB + *Caliper*.

En la Fig. 11, se puede observar cómo la integración de la *Traceband_NF Sender* + NetTCB + *Caliper* atienden a cada una de las limitantes encontradas en los estimadores de ancho de banda disponible presentados en la Tabla 1. En la Figura, se puede observar que NetTCB está orientado a solucionar las limitantes respecto al sistema operativo y el uso de tarjetas de red estándar, y *Traceband_NF Sender* + NetTCB + *Caliper* están orientadas a dar solución a la transmisión y recepción de los paquetes de prueba.

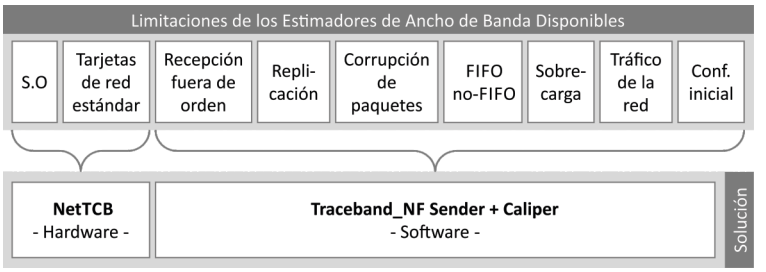

Fig. 11. Solución propuesta vs. Limitaciones de los estimadores de ancho de banda disponible.

5. Conclusiones

Dentro del proceso investigativo propuesto, la Fase 1 parte de la selección de la herramienta de generación de paquetes de prueba en NetFPGA. La herramienta seleccionada es NetThreads + Caliper, ya que cuenta con una precisión similar a NetFPGA Packet Generator, además de que NetThreads permite correr aplicaciones escritas en C dentro de la NetFPGA, lo que facilita la integración con *Traceband*. *Caliper* fue seleccionada, debido a que aprovecha los dos procesadores de NetThreads para generar paquetes de prueba de manera precisa con base en los tiempos establecidos en un archivo PCAP. Para poder implementar *Traceband* con NetThreads + Caliper, es necesario modificar *Traceband* para que en vez de generar y enviar los paquetes de prueba, los exporte en un archivo PCAP para que *Caliper* los genere y sean enviados a través de la NetFPGA. En la Fase 2 del proceso investigativo se propone la forma en que debe realizarse y en la Sección 4.1 se da solución a la Fase 3, presentando Traceband_NF Sender, la versión modificada de *Traceband* la cual consta de cuatro partes: *Traceband Packet Generator Specs* hace referencia a los datos de configuración inicial que requiere la versión original de *Traceband* para comenzar con la generación de los paquetes de prueba, *Build the Table* hace referencia al proceso que realizará Traceband_NF Sender para crear el archivo PCAP con la información necesaria que le permita a la NetThreads y *Caliper* construir y enviar los paquetes de prueba en los tiempo establecidos lo más exacto posible, *Send the Table* hace referencia al proceso que realizará Traceband_NF Sender para enviar el archivo PCAP directamente a la NetFPGA y *Show the ABw*,

el cual hace referencia al proceso que realizará Traceband_NF Sender para visualizar el ancho de banda disponible estimado en *Traceband Reveicer.*

La Fase 4 del proceso investigativo propone el diseño del módulo en la NetFPGA que permite la generación de paquetes de prueba. Para dar solución a ello, en la Sección 4.2, se proponen las modificaciones que deben realizarse a NetThreads. La primera modificación realizada, se encuentra dentro del *User Data Path* y consiste en la inserción del módulo Timestamp_snd. Aunque genera un retardo por la creación de un módulo en la ruta de datos de la NetFPGA, se requiere agregar la información del *timestamp* a la carga útil de todos los paquetes UDP que salen hacia *Traceband Receiver*, debido a que *Traceband Receiver* requiere dos *timestamp* para lograr una estimación: el que se envía dentro de la carga útil del paquete UDP y el que se captura cuando el paquete llega al *receiver.* Las tres modificaciones siguientes, son registros creados fuera del *User Data Path.*

El primero de los registros creados se llama ABw y su objetivo es reconocer cuáles de los paquetes que ingresan a la ruta de datos de la NetFPGA, es un paquete UDP que contiene la estimación del ancho de banda disponible calculado por *Traceband Receiver*. El segundo de los registros se llama Timestamp_Ram, es de 64 bits, almacena 24575 datos y su objetivo es almacenar en la memoria el *Timestamp* de cada paquete de prueba generado. Una vez se haya terminado de marcar los paquetes y estén registrados los tiempos en que llegaron dichos paquetes, la aplicación debe solicitar el envío de la información y se realizará el barrido de la Timestamp_Ram. Debido a que se requiere el uso de un registro de entrada y salida para poder realizar la comunicación entre *Traceband*, NetThreads y *Caliper*, fue necesaria la creación de un tercer registro llamado *Traceband Register,* el cual realiza las siguientes tareas:

- Entregarle a Register I/O el archivo PCAP que permite generar los paquetes de prueba.
- Acceder al módulo *Timestamp RAM* para obtener los valores de *timestamps* obtenidos dentro del User Data Path.
- Enviarle a Register I/O el valor de ancho de banda disponible para que Traceband_NF Sender la visualice.

La Fase 5 del proceso investigado propone el diseño esquema de comunicación de la NetFPGA con *Traceband*. En la Sección 3.4 se da solución a ello, haciendo necesaria la integración de Traceband con NetThreads y Caliper. Ello se logra de la siguiente manera:

- En primer lugar, un proceso en el espacio de usuario o módulo del *kernel* en el *host* determina cuándo debe comenzar la estimación.

- Luego, una descripción de los trenes de paquetes de prueba, que contiene los tiempos de transmisión y toda la información necesaria para ensamblar el paquete se envía al controlador de la NetFPGA en un archivo PCAP.
- En el controlador, múltiples descripciones de los paquetes de prueba se combinan y se copian en la tarjeta NetFPGA. Combinando las descripciones se reduce el número de transferencias separadas requeridas, lo que permitirá posteriormente el envío de paquetes a la tasa de línea de 1 Gbps.
- Posteriormente, las descripciones de los paquetes de prueba son procesados (en software) por los dos procesadores de NetTCB. Cada hilo de software ensambla los paquetes en la memoria de salida de NetTCB. A continuación, un hilo seleccionado inserta el *Timestamp* en cada paquete y los envía en el orden correcto en los tiempos de transmisión solicitados por Traceband_NF Sender.
- Finalmente, el *pipeline* de la NetFPGA transmite los paquetes sobre el alambre.

Referencias

[1] J. Sommers, P. Barford y W. Willinger, «A proposed framework for calibration of available bandwidth estimation tools,» de *Computers and Communications, 2006. ISCC'06. Proceedings. 11th IEEE Symposium on*, 2006.

[2] R. L. Carter y M. E. Crovelia, «Measuring bottleneck link speed in packet-switched networks,» *Performance evaluation,* vol. 27, pp. 297-318, 1996.

[3] V. Ribeiro, R. Riedi, R. Baraniuk, J. Navratil y L. Cottrell, «pathchirp: Efficient available bandwidth estimation for network paths,» de *Passive and active measurement workshop*, 2003.

[4] B. Melander, M. Bjorkman y P. Gunniengberg, «A new ent to end probing and analysis method for estimating bandwidth bottlenecks,» *Global Telecommunications Conference IEEE,* pp. 415- 420, 2000.

[5] M. Jain y C. Dovrolis, «Pathload: A measurement tool for end-to-end available bandwidth,» de *In Proceedings of Passive and Active Measurements (PAM) Workshop*, 2002.

[6] N. Hu y P. Steenkiste, «Estimating Available Bandwidth Using Packet Pair Probing,» *IEEE,* p. 27, 2002.

[7] J. Strauss, D. Katabi, F. Kaashoek y B. Prabhakar, «Spruce: A lightweight end-to-end tool for measuring available bandwidth,» de *Proc. of the Internet Measurement Conference (IMC)*, 2003.

[8] J. Navratil y R. Cottrell, «ABING: A practical approach to available bandwidth estimation,» *Passive and active measurement workshop IEEE,* 2003.

[9] G. Jin y B. Tierney, «Nestet: A tool to measure the maximum burts size, available bandwidth and achievable throughput,» *Proceedings In information Technology: Research and Education ,* pp. 578-582, 2003.

[10] K. Lakshminarayanan, V. N. Padmanabhan y J. Padhye, «Bandwidth estimation in broadband access networks,» *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement,* p. 314–321, 2004.

[11] A. Johnsson, B. Melander y M. Björkman, «Diettopp: A first implementation and evaluation of a simplified bandwidth measurement method,» de *Proceedings of the 2nd Swedish national computer networking workshop*, 2004.

[12] D. Kiwior, J. Kingston y A. Spratt, «PathMon, a methodology for determining available bandwidth over an unknown network,» de *Advances in Wired and Wireless Communication, 2004 IEEE/Sarnoff Symposium on*, 2004.

[13] L. Xin, «PoissonProb: A new rate-based available bandwidth measurement algorithm,» M. Sc. Thesis, University of Windsor, Windsor, 2005.

[14] S. Ekelin, M. Nilsson, E. Hartikainen, A. Johnsson, J.-E. Mangs, B. Melander y M. Bjorkman, «Real-time measurement of end-to-end available bandwidth using kalman filtering,» de *Network Operations and Management Symposium, 2006. NOMS 2006. 10th IEEE/IFIP*, 2006.

[15] M. Zhang, C. Luo y J. Li, «Estimating Available Bandwidth Using Multiple Overloading Streams,» 2006.

[16] A. A. Ali y F. Lepage, «IGMPS, a new tool for estimating end-to-end available bandwidth in IP network paths,» de *Networking and Services, 2007. ICNS. Third International Conference on*, 2007.

[17] Q. Wang y L. Cheng, «FEAT: Improving accuracy in end to end available bandwidth measurement,» *Global Telecommunications Conference IEEE,* pp. 1-4, 2006.

[18] T. Goldoni y E. Rossi, «Assolo, a new method for available bandwidth estimation,» *Internet and protection IEEE,* pp. 130-136, 2009.

[19] D. Roy, «PathAB: A New Method to Estimate End-to-End Available Bandwidth of Network Path,» 2009.

[20] C. Guerrero y M. Labrador, «Traceband: A fast, low overhead and accurate tool for available bandwidth estimation and monitoring,» *Computer Networks,* p. 977–990, 2010.

[21] J. Sommers, P. Barford y W. Willinger, «A proposed framework for calibration of available bandwidth estimation tools,» de *Computers and Communications, 2006. ISCC'06. Proceedings. 11th IEEE Symposium on*, 2006.

[22] A. Covington, G. Gibb, J. Lockwood y N. Mckeown, «A packet generator on the NetFPGA platform,» de *Field Programmable Custom Computing Machines, 2009. FCCM'09. 17th IEEE Symposium on*, 2009.

[23] M. Labrecque, J. G. Steffan, G. Salmon, M. Ghobadi y Y. Ganjali, «NetFPGA-based Precise Traffic Generation,» de *NetFPGA Dev. Workshop'09*, 2009.