# Planificadores para sistemas heterogéneos formados por CPUs, GPUs y FPGAs

Mª Angélica Dávila-Guzmán, Rubén Gran[1], María Villarroya-Gaudó[1], Darío Súarez-Gracia[1]

[1] Grupo de Arquitectura de Computadores
Instituto de Investigación en Ingeniería de Aragón (I3A)
Universidad de Zaragoza, Mariano Esquillor s/n, 50018, Zaragoza, Spain.
Tel. +34-976762707, e-mail: angelicadg@unizar.es

## Resumen

En esta comunicación se presenta parte de trabajo que se está realizando dentro de la tesis doctoral de Mª Angélica Dávila-Guzmán. En concreto, se está desarrollando un planificador de carga para sistemas de cómputo heterogéneo compuestos por dispositivos como la CPU, GPU y FPGA. Este trabajo está en fase inicial y se presentan los objetivos y primeros resultados alcanzados.

## Introducción

La ley de Moore, enunciada en el año 65 y revisada en el 75, indicaba que el número de transistores que forman los circuitos integrados se iba a duplicar primero cada año y después cada 2 años se ha cumplido hasta hoy[1]. Pero ya no se ha podido seguir empleando el escalado de Dennard, que establecía como debe escalarse varias dimensiones físicas para mantener constante la densidad de potencia a la par que se obtenían transistores más pequeños, más rápidos y que consumieran menos. Al no poder mantener la densidad de potencia constante, ha surgido la tendencia a incrementar el número de procesadores en un chip como en los multiprocesadores y a combinar arquitecturas de procesamiento como en los sistemas heterogéneos.

En los sistemas heterogéneos se ha adicionado a las CPUs tradicionales procesadores especializados como las GPU (unidades de procesamiento gráfico), que han destacado por su alto rendimiento en aplicaciones de cómputo intensivo, pero su alto consumo energético ha derivado en problemas de disipación térmica limitando su uso.

Por ello se ha propuesto el empleo de otro tipo de aceleradores como la FPGA (matriz de puertas programables) que reconfiguran sus unidades funcionales internas para generar una arquitectura de procesamiento específica para cada programa. El mayor atractivo de las FPGA es el bajo consumo energético, comparado con una GPU. Su beneficiosa relación desempeño-potencia está siendo explotada en centros de cómputo y dispositivos donde el consumo de potencia es un factor determinante [2].

El uso de las FPGA se ha visto limitado por la dificultad en la programación debido a las diferencias con los paradigmas tradicionales de programación. Esta brecha con los programadores está siendo cerrada gracias a la introducción de la síntesis de alto nivel con lenguajes de programación como C, C++ o OpenCL[3] .

Con la necesidad de una unificación en la programación de sistemas heterogéneos, empresas interesadas en conectar el hardware con el software desarrollaron la especificación OpenCL[4], dando soporte a un modelo de la programación paralela sin ocultar la arquitectura del dispositivo. Aunque OpenCL permite la concurrencia entre múltiples dispositivos es el programador el encargado de decidir donde ejecutar la aplicación y buscar el óptimo para distribuir la carga de cómputo, lo que requiere un proceso sintonización de cada dispositivo para minimizar el tiempo de ejecución y el consumo energético del sistema.

La distribución de carga ha sido estudiada en sistemas conformados por pares de dispositivos siendo uno de ellos una CPU más una GPU o un Xeon Phi o una FPGA [5,6], pero sobre los 3 dispositivos a la vez casi no existen trabajos.

En este trabajo se busca que más de dos dispositivos como la CPU, GPU y FPGA realicen una ejecución cooperativa buscando maximizar el desempeño de todo el conjunto. Inicialmente se analizará las herramientas existentes, para posteriormente conseguir un reparto de carga óptimo en tiempo de ejecución y además facilite el trabajo de programación.

## Resultados Preliminares

Para analizar el comportamiento del sistema heterogéneo se ha adaptado el *framework*

engineCL[7], que permite el uso paralelo de los dispositivos de un sistema heterogéneo y se le ha adicionada soporte para la FPGA. Se evalúa el tiempo de ejecución en dos aplicaciones: Multiplicación de Matrices y Mersenne, optimizándolos para los tres dispositivos: CPU Intel core i7-6700k, GPU NVIDIA GeForce GTX TITAN X y la FPGA Stratix V GX.

En la Fig. 1 se muestra el tiempo de ejecución normalizado, respecto a la peor ejecución individual, de los dispositivos solos y cooperando cada uno con un porcentaje de carga de trabajo equivalente a la capacidad de cómputo individual. Se evidencia la dependencia entre la aplicación y el dispositivo, dificultando la selección a priori del dispositivo que mejor desempeño puede generar. En cambio, la combinación de ellos acelera el tiempo de ejecución hasta un 37%, respecto al mejor dispositivo individual, incluso el dispositivo con peor desempeño aporta un 7.7%. Esta es solo una muestra de una posible solución de las infinitas combinaciones que se pueden generar, lograda gracias a la caracterización previa de los dispositivos.

# Conclusiones y Trabajo Futuro

En el escenario tecnológico actual, el desarrollo de herramientas que faciliten la programación de sistemas heterogéneos es clave para la programación de los futuros sistemas informáticos.

Los resultados preliminares muestran que el trabajo conjunto de dispositivos individuales en un sistema heterogéneo puede mejorar el desempeño de cada uno por separado. Pero realizar esta distribución requiere un trabajo arduo de caracterización de los dispositivos que varía con la carga de cómputo. Esto crea la necesidad de planificadores autónomos que reconozcan estas diferencias para lograr optimizar el uso de los recursos en tiempo de ejecución y eficiencia energética a la vez que se incremente la productividad del programador.

# Agradecimientos

Este ha sido financiado por TIN2016-76635-C2-1-R (AEI/FEDER, UE), gaZ: T48 grupo de investigación del (Gobierno de Aragón), HiPEAC4 (European H2020/687698), A. Dávila-Guzmán ha recibido una beca de doctorado de la Universidad de Zaragoza-Banco Santander.

## REFERENCIAS

[1]. M. Bohr and I. Young . “CMOS Scaling Trends and Beyond”, 2017. IEEE Micro, vol. 37, no. 6, 2017, pp. 20-29.

[2]. A. Dassatti and R. Rigamonti, “Heterogeneous Hardware from Homogeneous Software,” 2017 Int. Conf. High Perform. Comput. Simul., 2017, pp. 913–914.

[3]. F. B. Muslim, L. Ma, M. Roozmeh, and L. Lavagno, “Efficient FPGA Implementation of OpenCL High-Performance Computing Applications via High-Level Synthesis,” IEEE Access, vol. 5, no. 99, pp. 2747–2762, 2017.

[4]. B. J. E. Stone, D. Gohara, and G. Shi, “OpenCL: A Parallel Programming Standart For Heterogeneous Computing Systems,” 2010.

[5]. R. Pandit, Prasanna and Govindarajan, “Fluidic kernels: Cooperative execution of opencl programs on multiple heterogeneous devices,” Proc. Ann.. IEEE/ACM Int. Symp. Code Gener. Optim. , 2014, p. 273.

[6]. B. Pérez, E. Stafford, J. L. Bosque, and R. Beivide, “Energy efficiency of load balancing for data-parallel applications in heterogeneous systems,” J. Supercomput., vol. 73, no. 1, pp. , 2017, 330–342.

[7]. M. A. Dávila-Guzmán, R. Nozal, R. Gran, M. Villaroya-Gaudó, D. Suárez, and J. L. Bosque, “First Steps Towards CPU, GPU, and FPGA Parallel Execution with EngineCL,” Proc. 18th Int. Conf. Comput. Math. Method Sci. Eng. C., 2018.

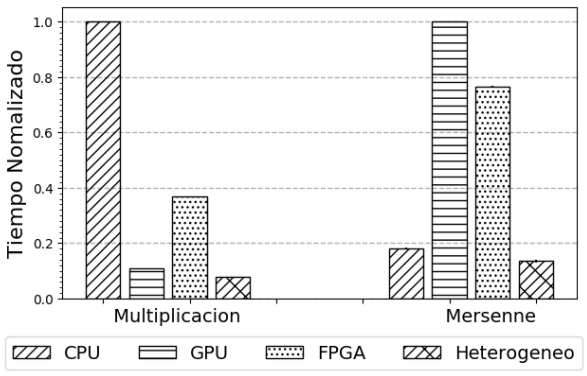


Fig 1. Tiempo de ejecución normalizado al dispositivo con peor tiempo de ejecución individual para la CPU, GPU y FPGA solos y cooperando