UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Implementación del algoritmo Differential Evolution en OpenCL

Autor
Mauricio Raúl, Palavecino Nicotra

Director
Rafael Gadea Gironés

Trabajo de fin de máster para optar por el título de Máster Universitario en Ingeniería de Sistemas Electrónicos

Valencia, miércoles 9 de noviembre de 2016

Índice

Objetivos

- Objetivo principal:

Implementar el algoritmo DE para evolucionar los pesos de un conjunto de redes neuronales de tipo Perceptrón Multicapa, acelerando la computación por medio de kernels descritos en OpenCL e implementados en hardware sobre un dispositivo acelerador FPGA.

- Objetivo secundario:

Incorporar el módulo IP que computa la tangente hiperbólica, descrito en HDLs, como función auxiliar en la descripción de los kernels de OpenCL.

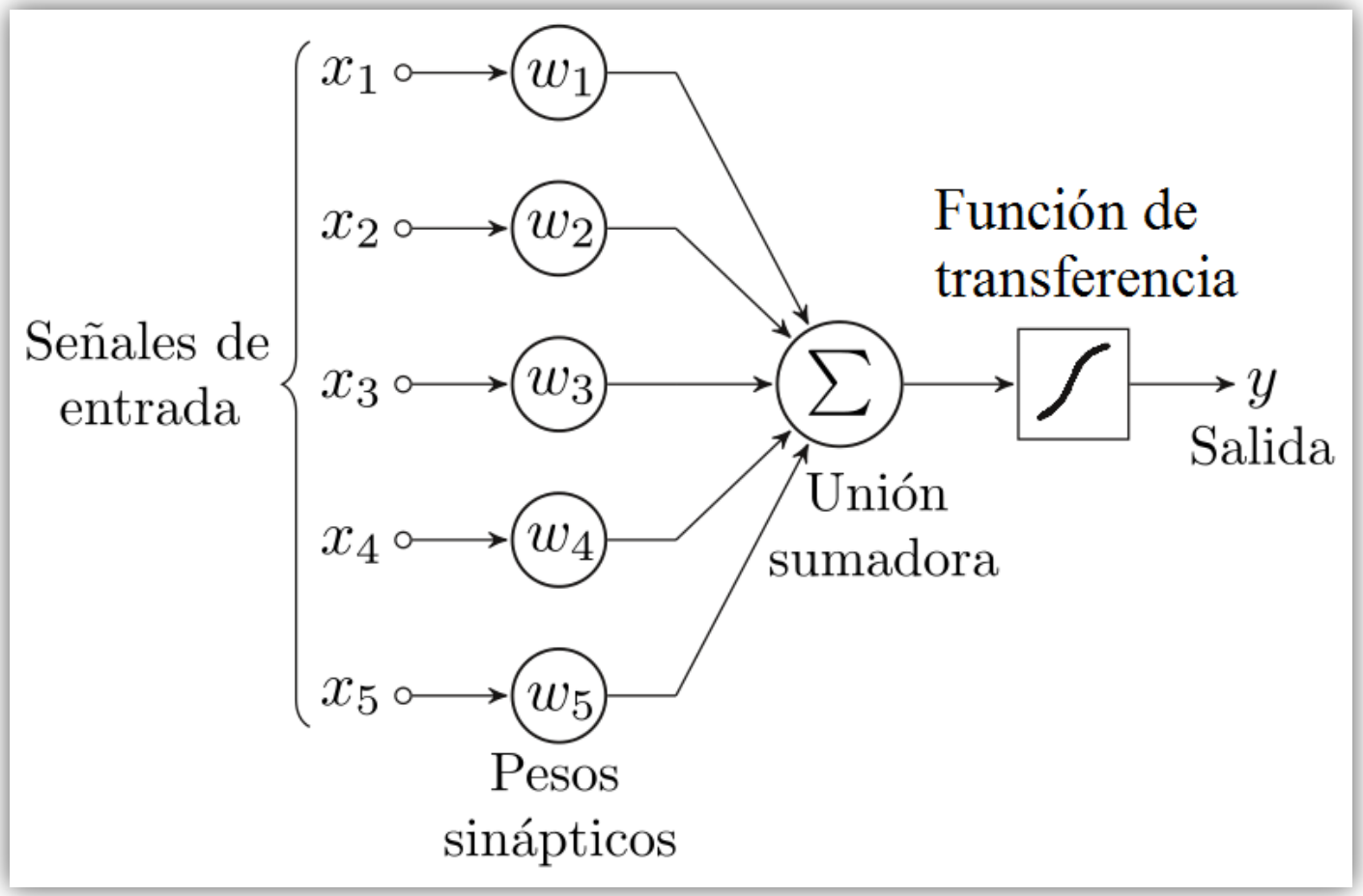
Redes Neuronales Artificiales

Concepto

Las RNA son una clase de modelos computacionales que intentan emular el comportamiento del cerebro humano, utilizando una estructura en red en la cual los nodos, denominados neuronas, son procesos numéricos que involucran el estado de otros nodos

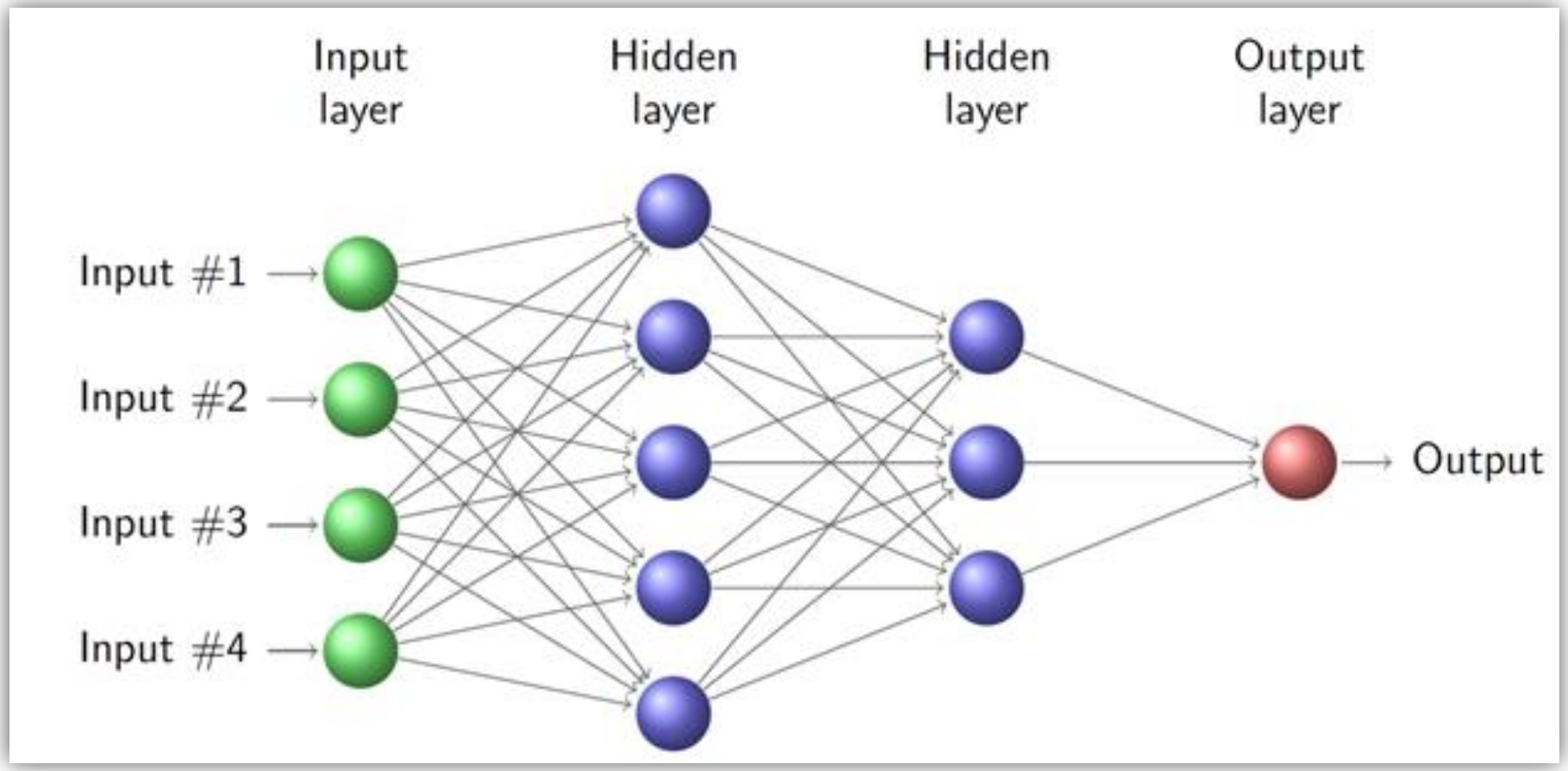
Redes Neuronales Artificiales

Modelo de la neurona



Redes Neuronales Artificiales

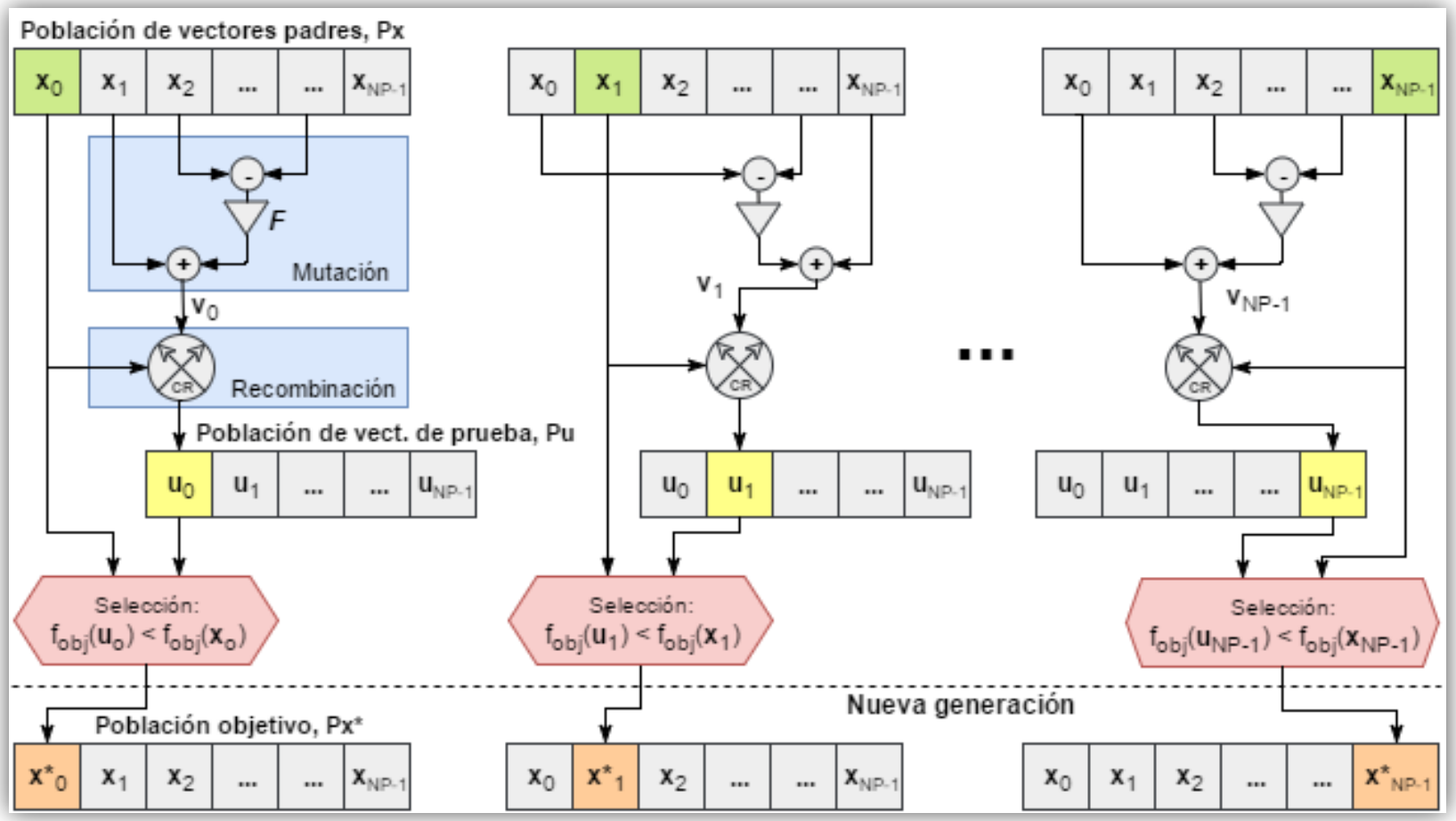
El Perceptrón Multicapa



Algoritmo de Evolución Diferencial, DE

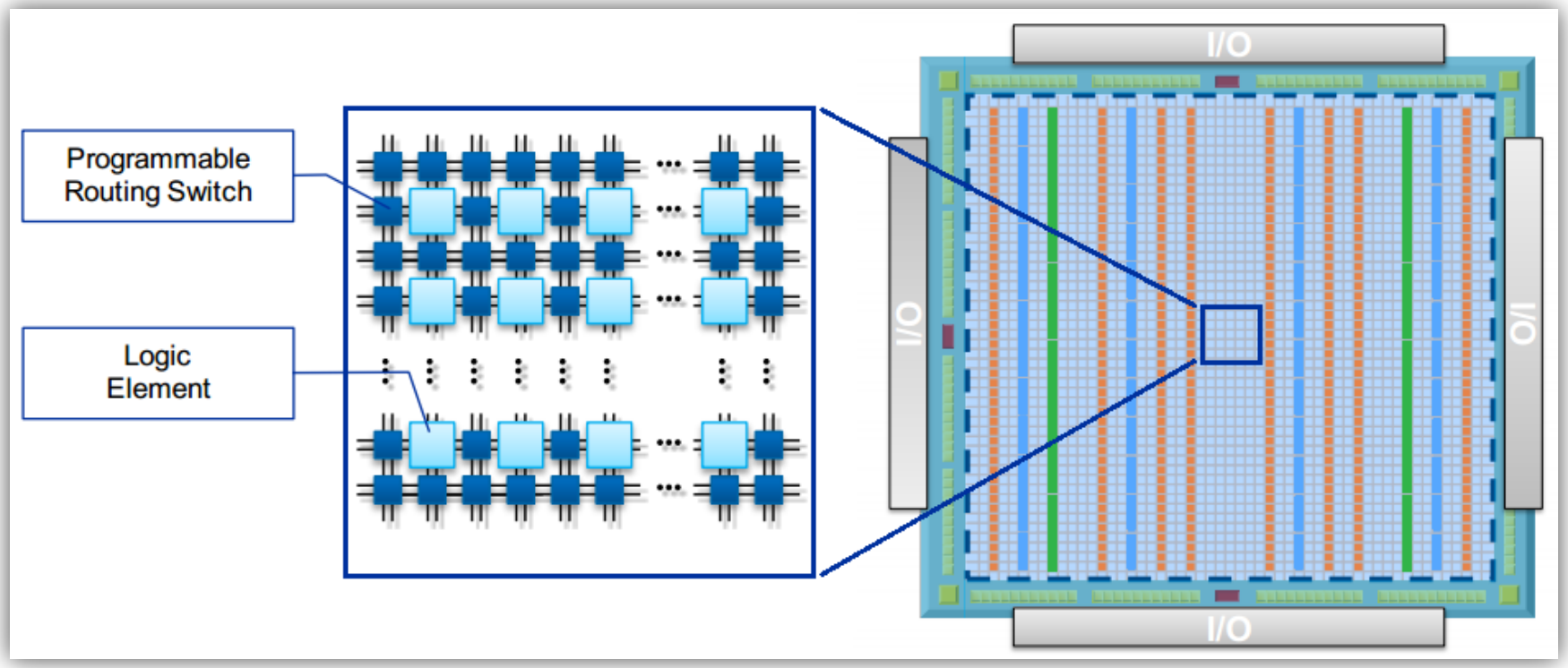
- Es un algoritmo de optimización de funciones reales evaluadas en un espacio multidimensional D…

Algoritmo de Evolución Diferencial, DE

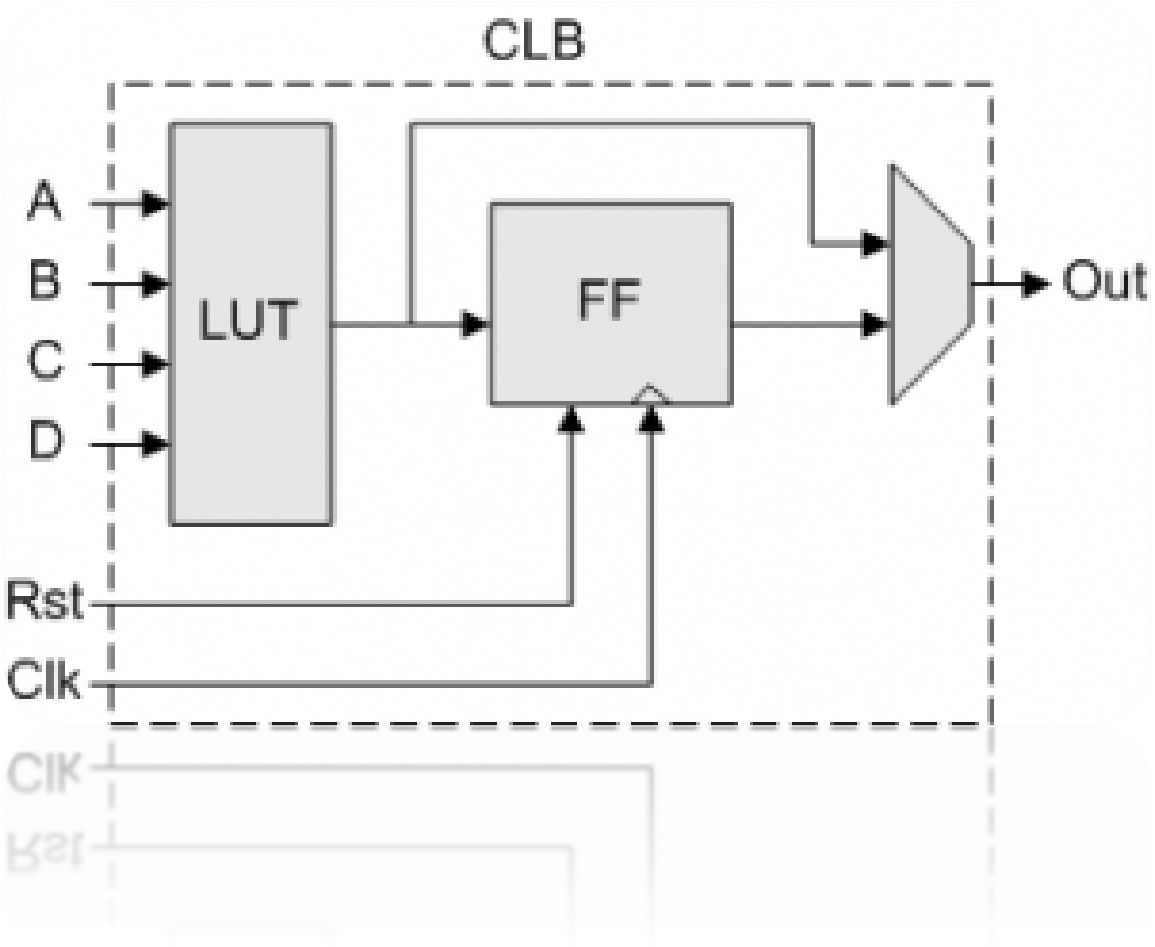


FPGAs

Field Programmable Gate Array



FPGAs



OpenCL

OpenCL v1.00

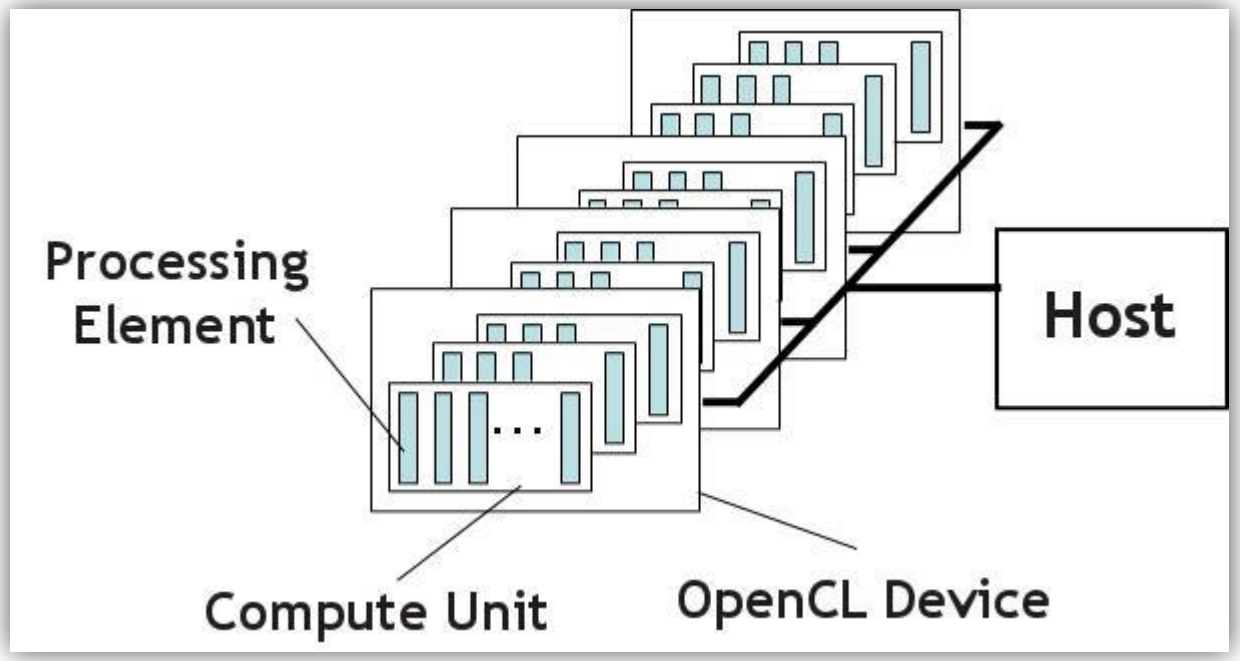
Modelo de programación en paralelo, abierto, libre de regalías (royalty-free) y unificado, orientado a la aceleración de algoritmos en plataformas heterogéneas:

Su arquitectura se basa en cuatro modelos:

- Modelo de plataforma
- Modelo de ejecución
- Modelo de memoria
- Modelo de programación

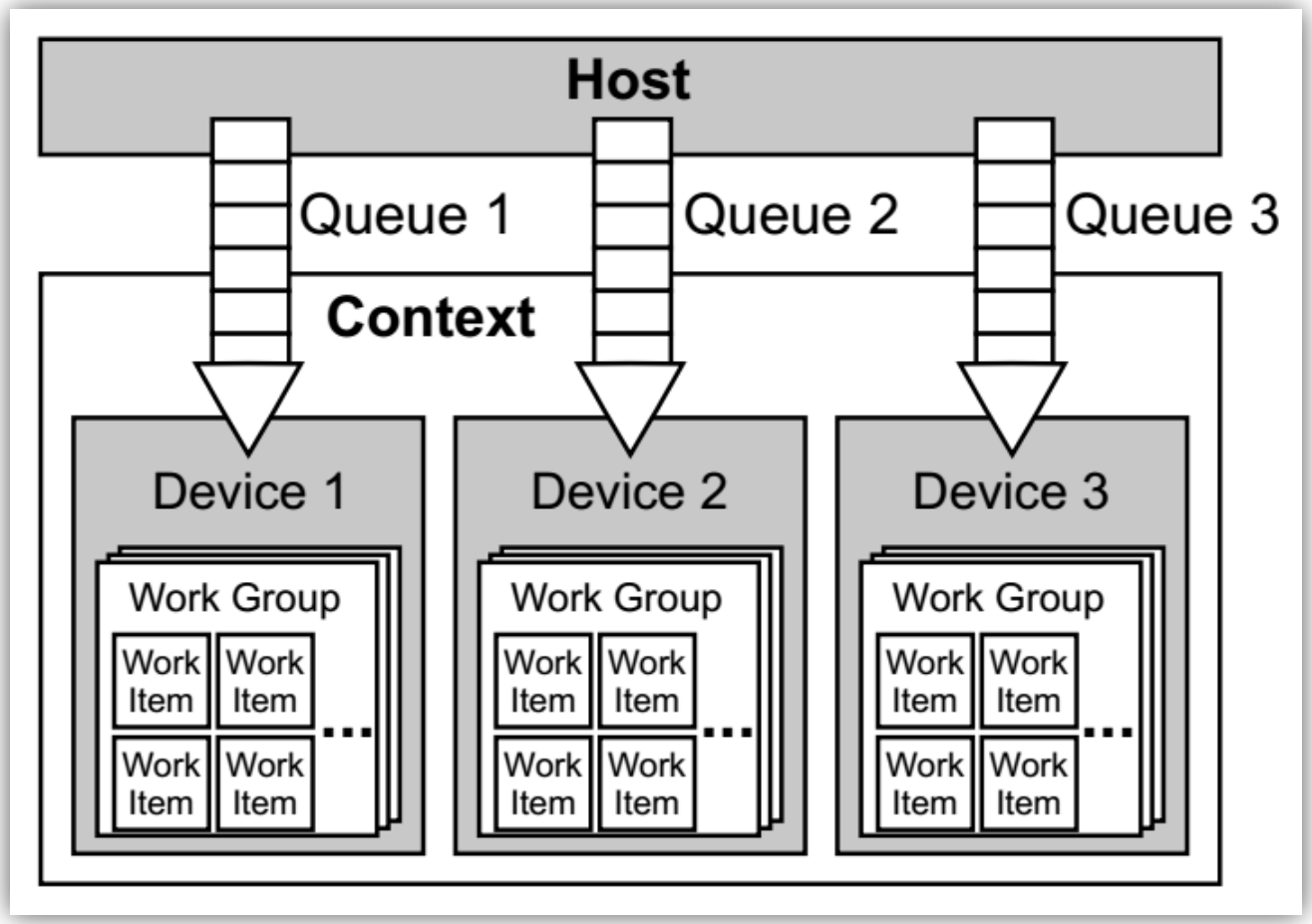
OpenCL

Modelo de plataforma



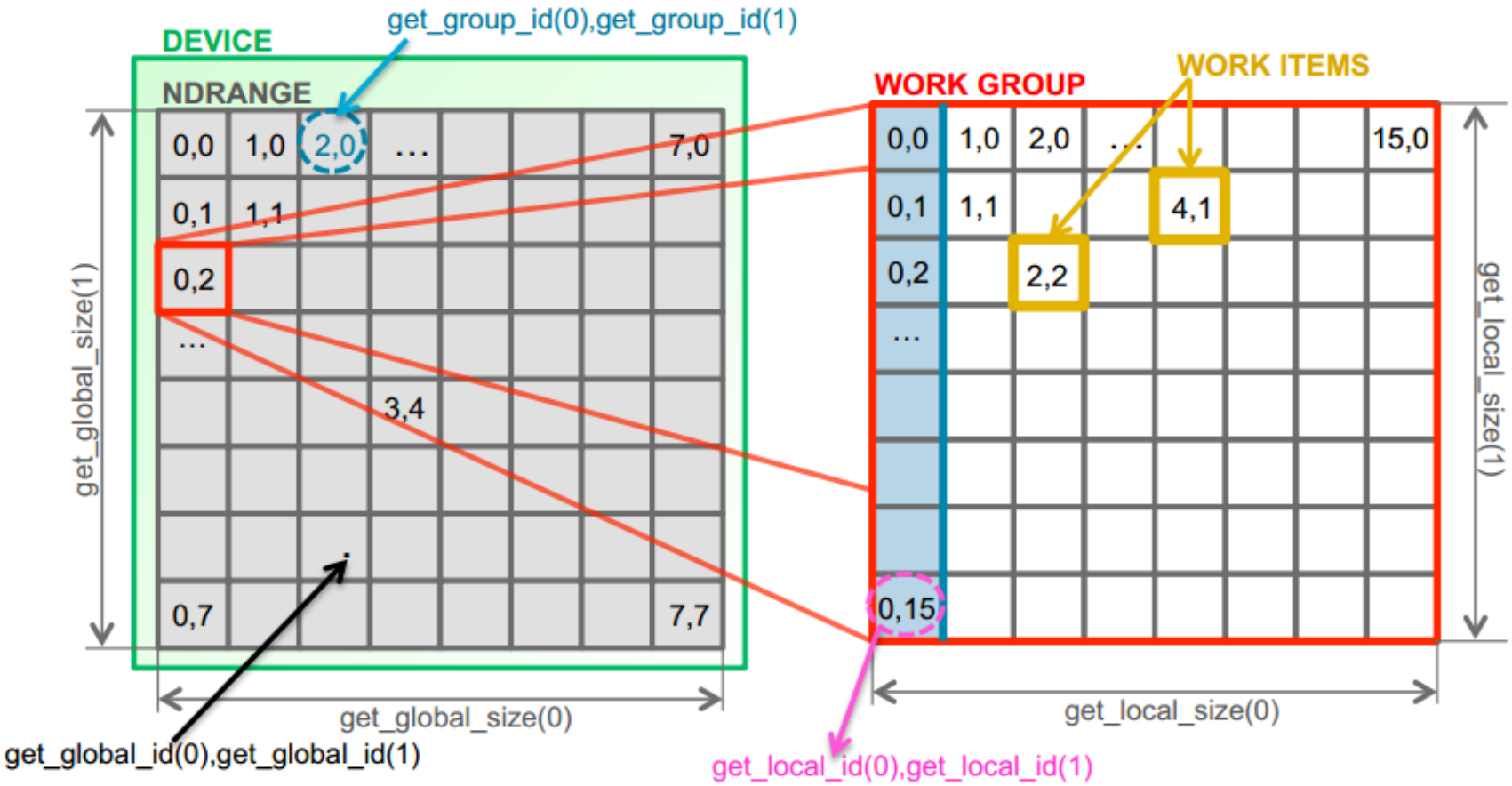
OpenCL

Modelo de ejecución



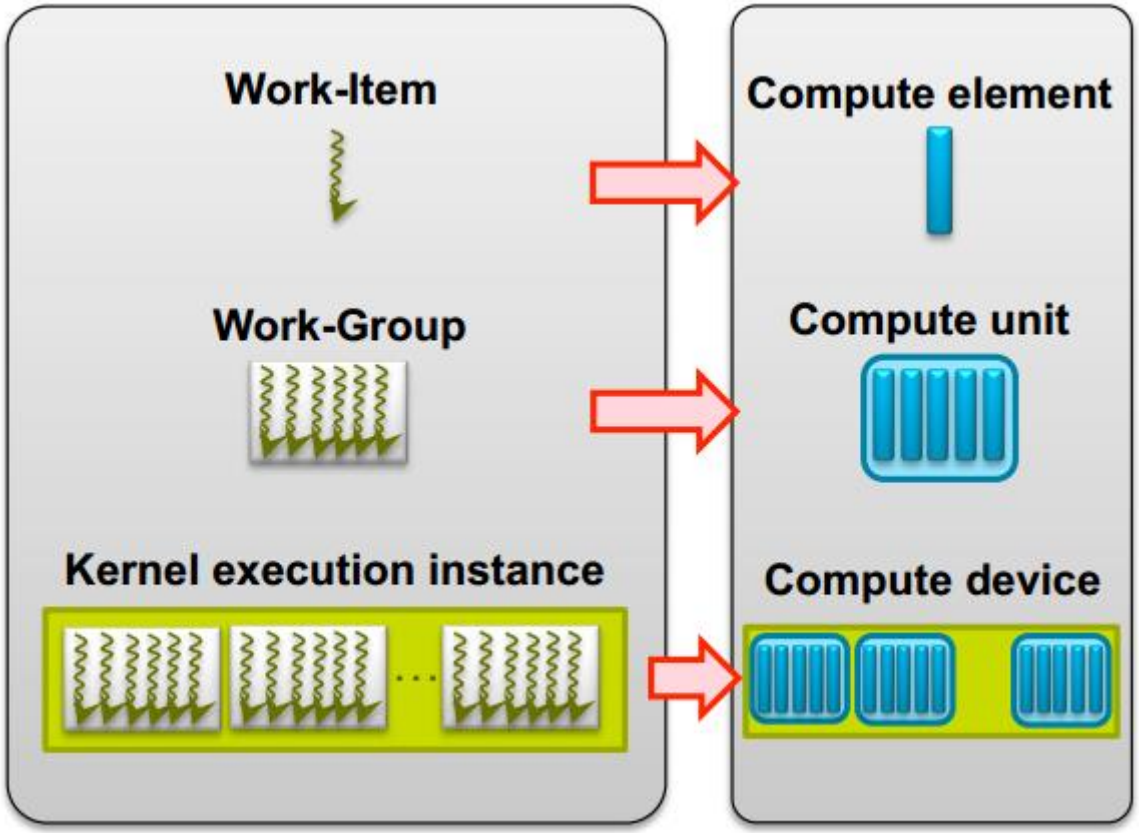
OpenCL

Modelo de ejecución



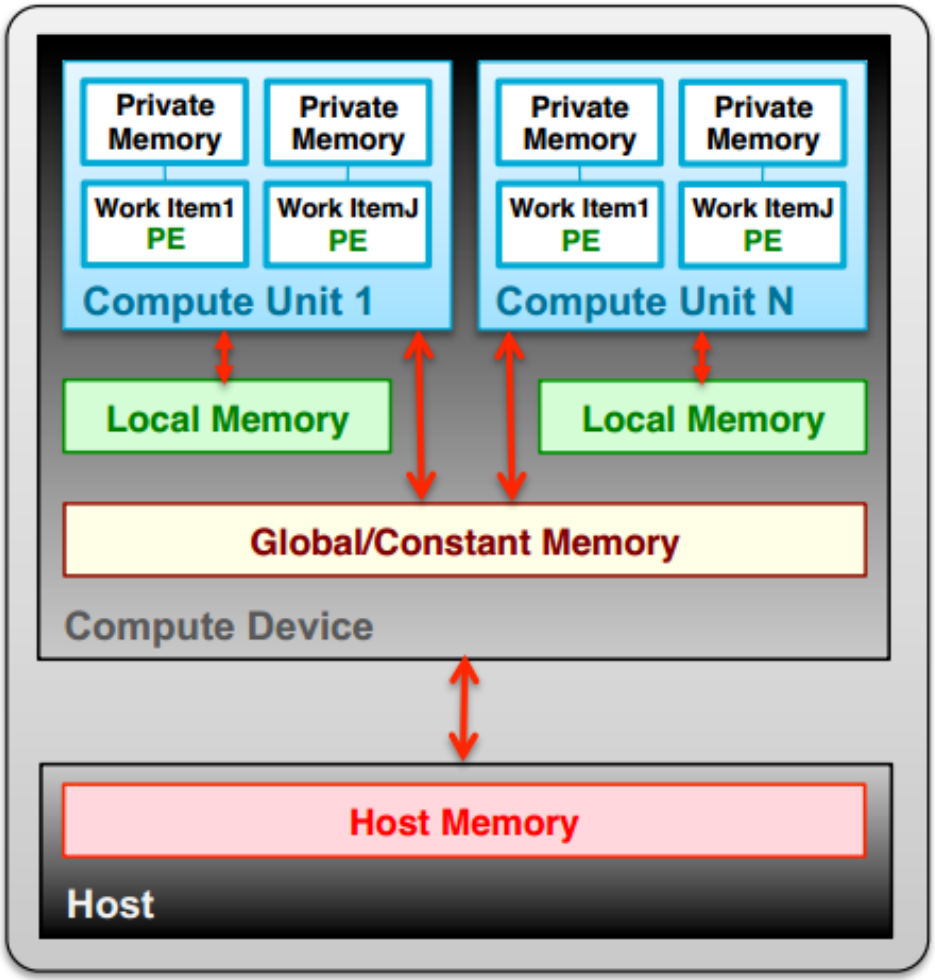
OpenCL

Modelo de ejecución



OpenCL

Modelo de memoria



OpenCL

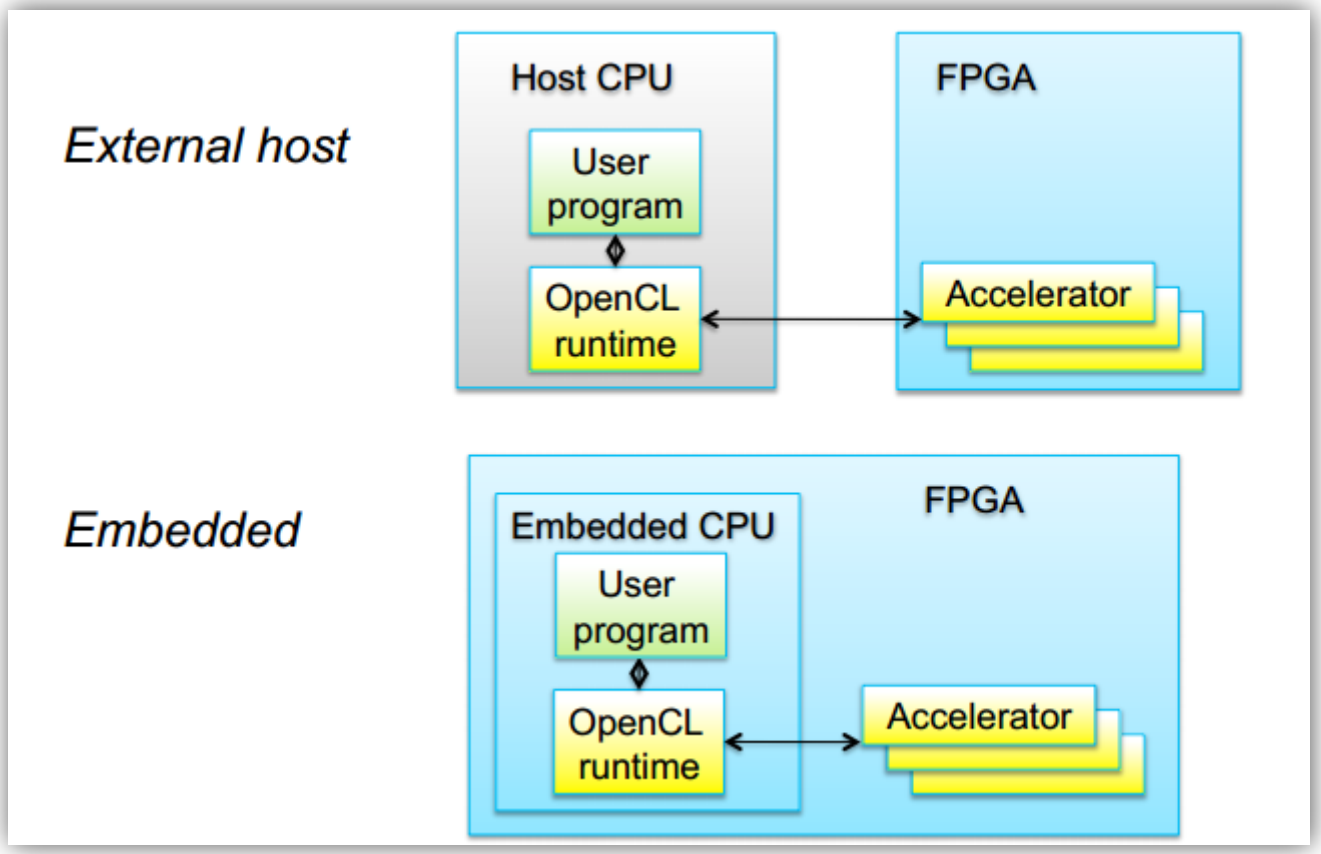
Modelo de programación

- Modelos de programación en paralelo:
  - Modelo de programación de paralelismo de los datos
  - Modelo de programación de paralelismo de las tareas
- Métodos de sincronización
  - Entre work-items en un mismo work-group
  - Por medio de comandos

NOTA: En las nuevas versiones de OpenCL el modelo de programación está '*incluido*' en el modelo de ejecución.

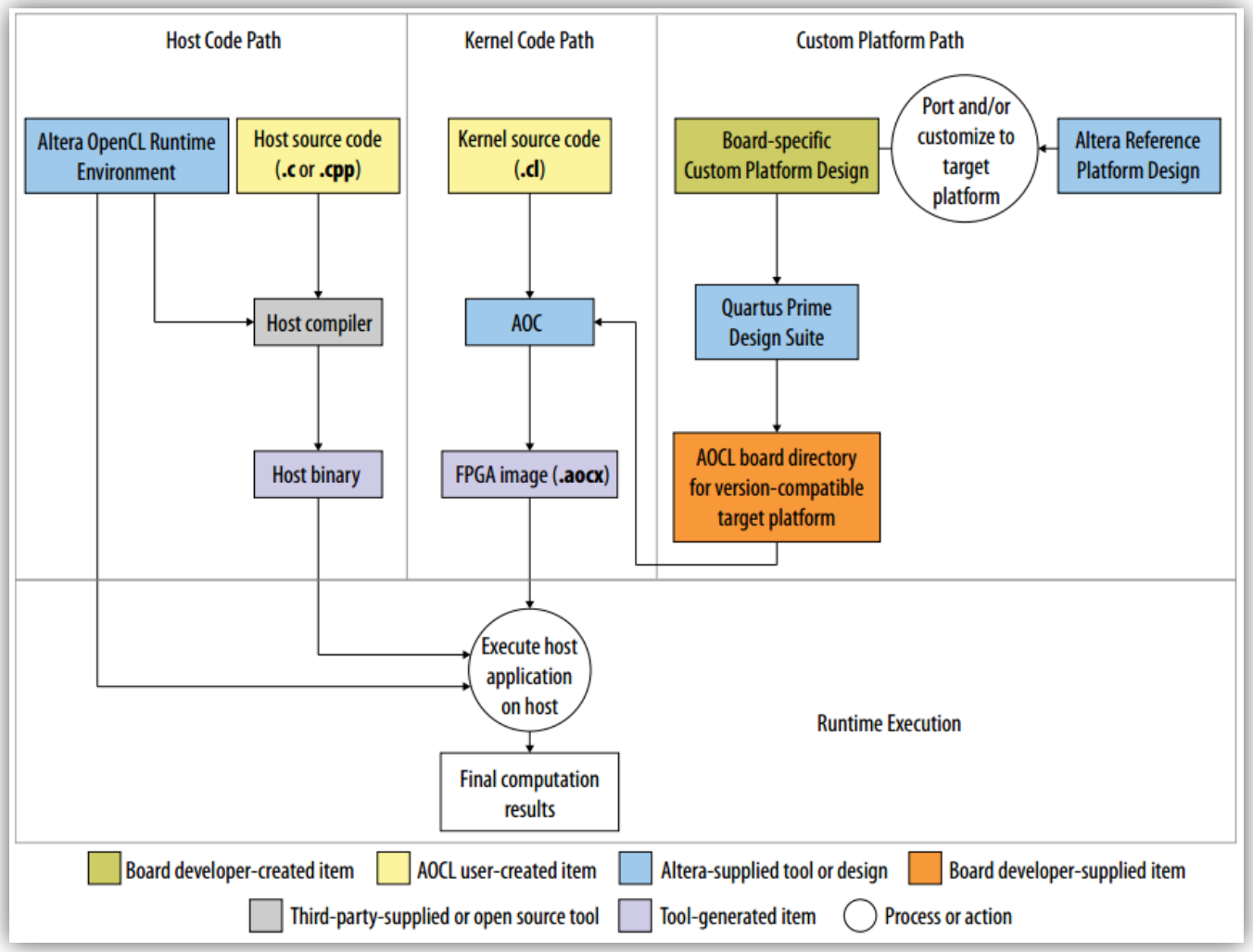
OpenCL & FPGAs

Tipos de aceleración con FPGAs



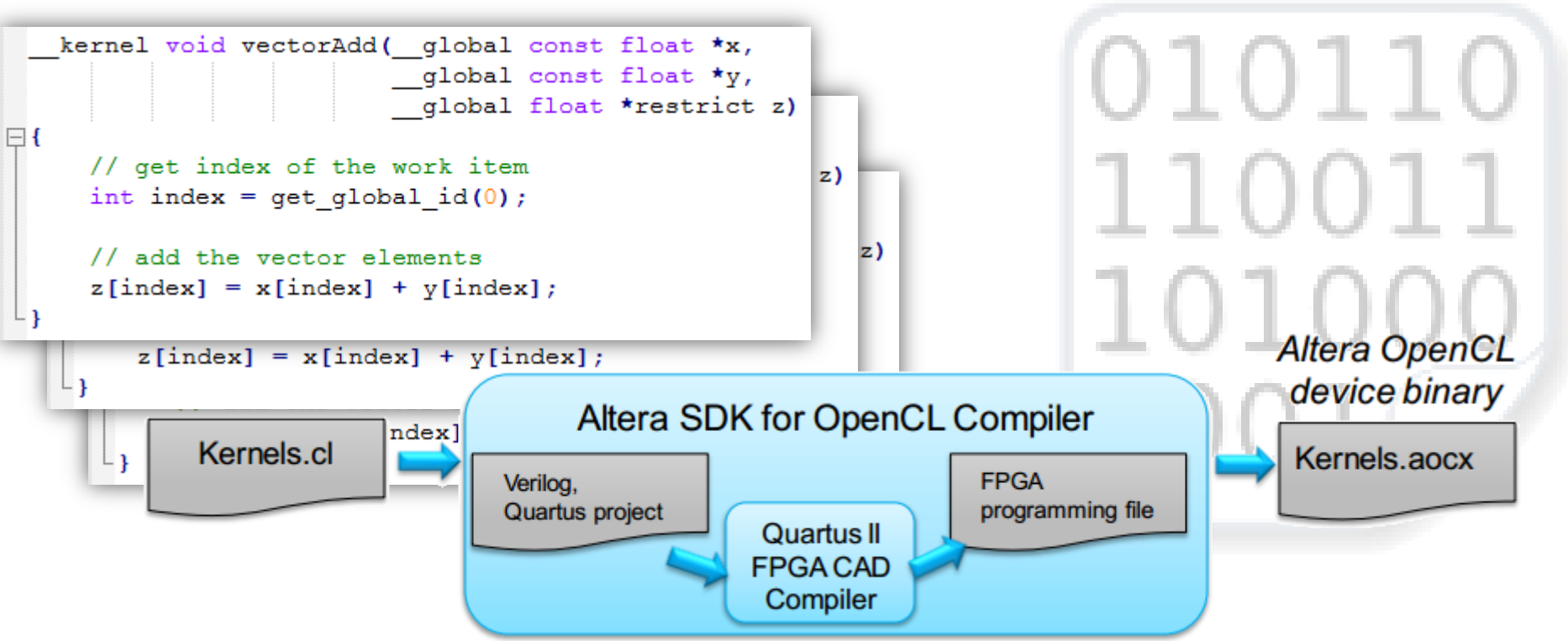
OpenCL & FPGAs

Flujo de diseño con Altera



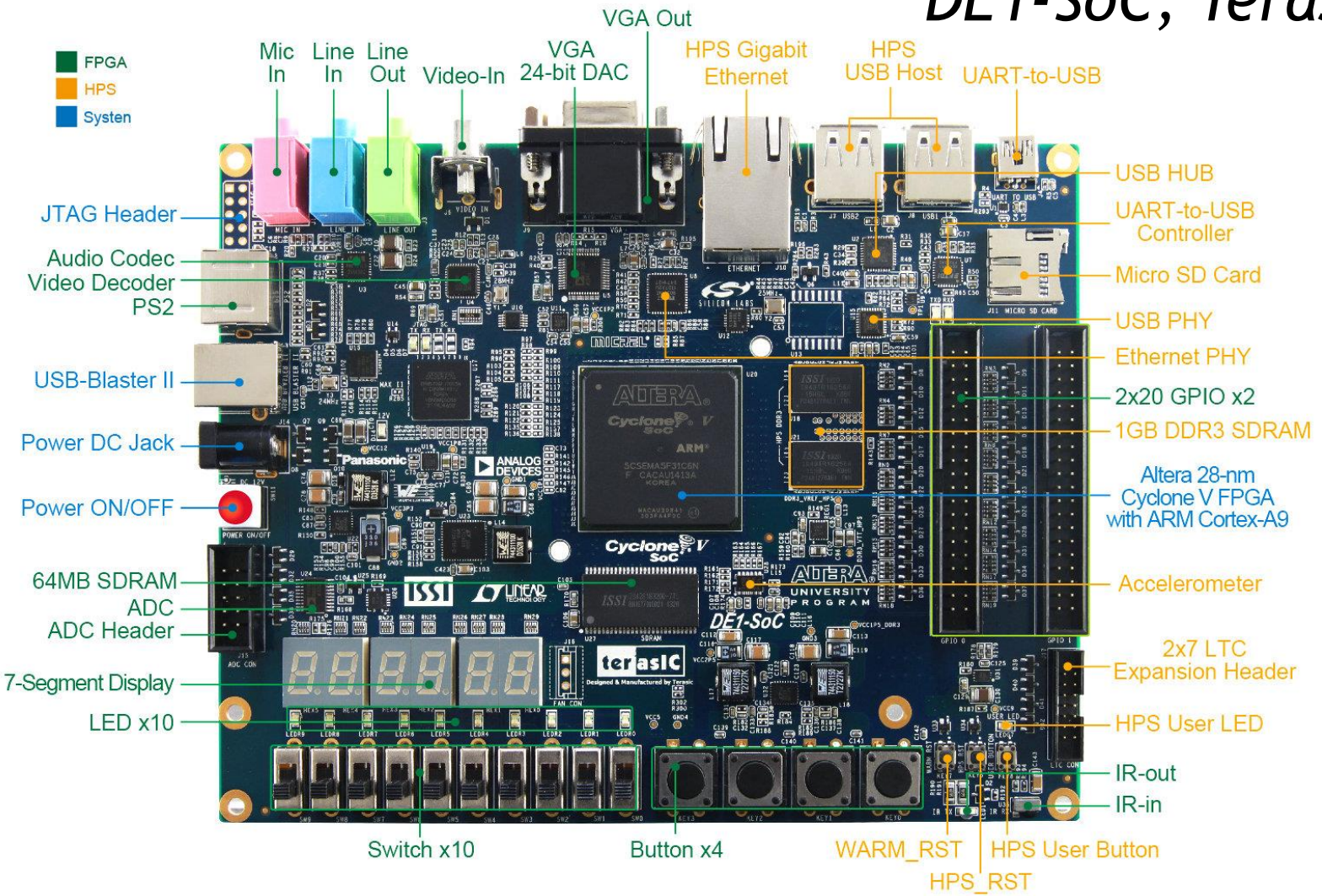
OpenCL & FPGAs

Flujo de compilación con Altera



Plataforma de desarrollo

DE1-SoC, Terasic



Justificación

Objetivos

El Perceptrón Multicapa

- Objetivo principal:

Implementar el algoritmo DE para evolucionar los pesos de un conjunto de redes neuronales de tipo Perceptrón Multicapa, acelerando la computación por medio de kernels descritos en OpenCL e implementados en hardware sobre un dispositivo acelerador FPGA.

- Objetivo secundario:

Incorporar el módulo IP que computa la tangente hiperbólica, descrito en HDLs, como función auxiliar en la descripción de los kernels de OpenCL.

Desarrollo

Temario

- DE para la optimización de los pesos de las redes neuronales
- Aceleración del DE con OpenCL
- Aceleración del cálculo de las ANNs
  - Paralelismo de en el cómputo de las neuronas
  - Paralelismo en el cómputo de diferentes redes
- Kernel de acumulación del error
- Programa del Host

Resultados

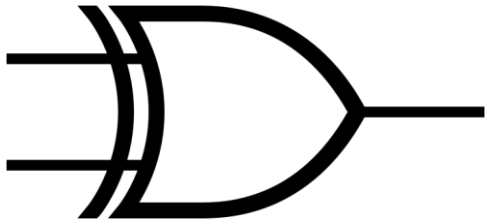
Temario

- Problema modelo a resolver
- Resultados de entrenamiento
- Resultados temporales y de aceleración
  - Mediciones para número variable de generaciones, tamaños de poblaciones, de neuronas en las capas ocultas y de muestras de entrenamiento
- Resultados de implementación sobre la plataforma

Resultados

Problema modelo

Función lógica XOR



Identificación de estímulo patrón	Entrada #1	Entrada #2	Valor de salida
#0	+0.95	-0.95	+0.95
#1	-0.95	+0.95	+0.95
#2	+0.95	+0.95	-0.95
#3	-0.95	-0.95	-0.95

Resultados

Resultados de entrenamiento

Los errores de salida son menores que los obtenidos para con el entrenamiento realizado con el *Resilent Back-Propagation* (RBP), un método común en el entrenamiento de MLPs, basado en la corrección de los pesos individuales de la red en base a la propagación hacia atrás del error de salida.

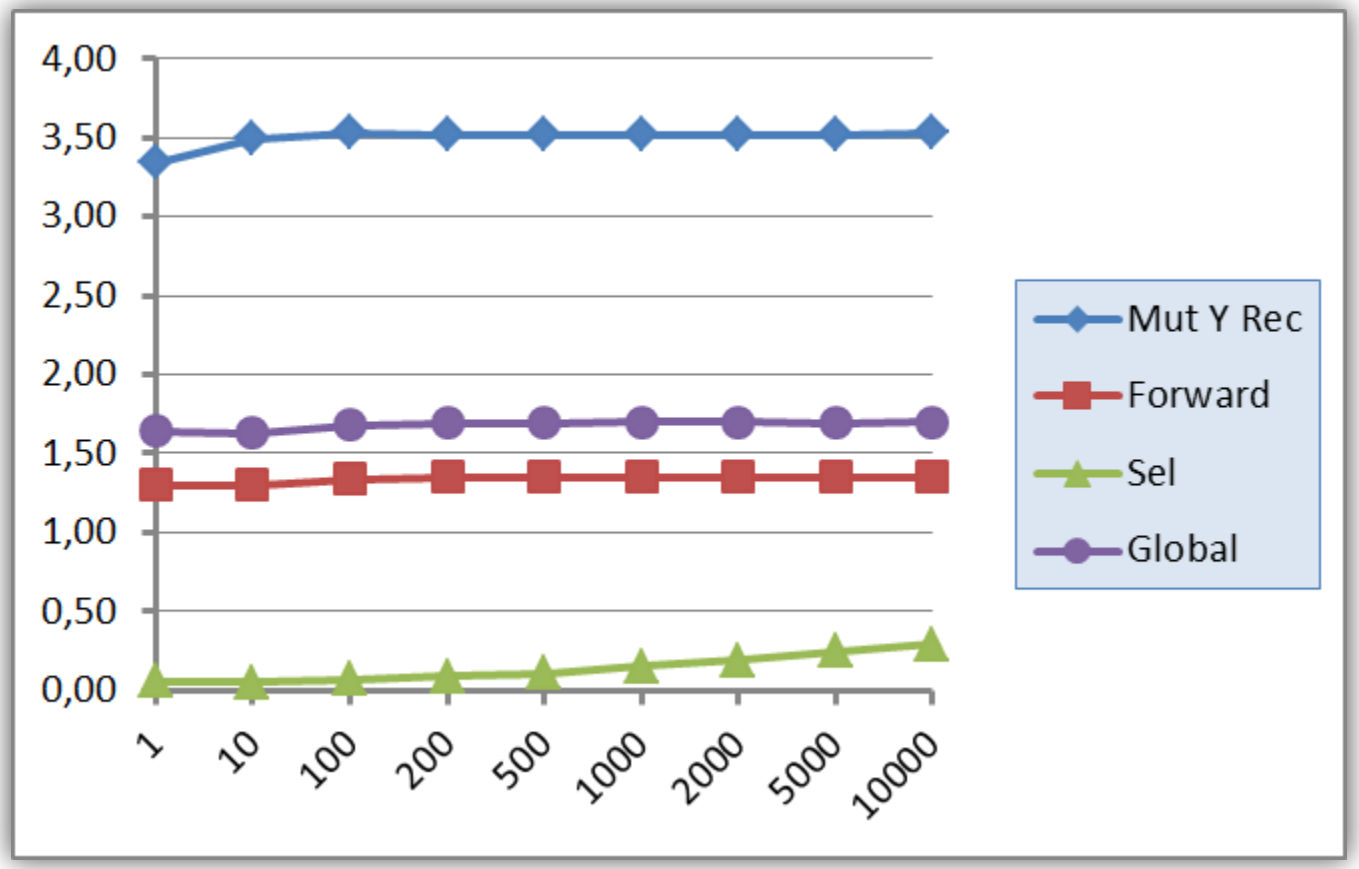
Resultados

Mediciones temporales y de aceleración

- Parámetros de los que depende el tiempo de ejecución:
  - Número de generaciones
  - Tamaño de las poblaciones
  - Número de neuronas
    - En la capa oculta 1
    - En la capa oculta 2
    - En ambas capas
  - Tamaño del conjunto de datos de entrenamiento (número de muestras de estímulos patrón)

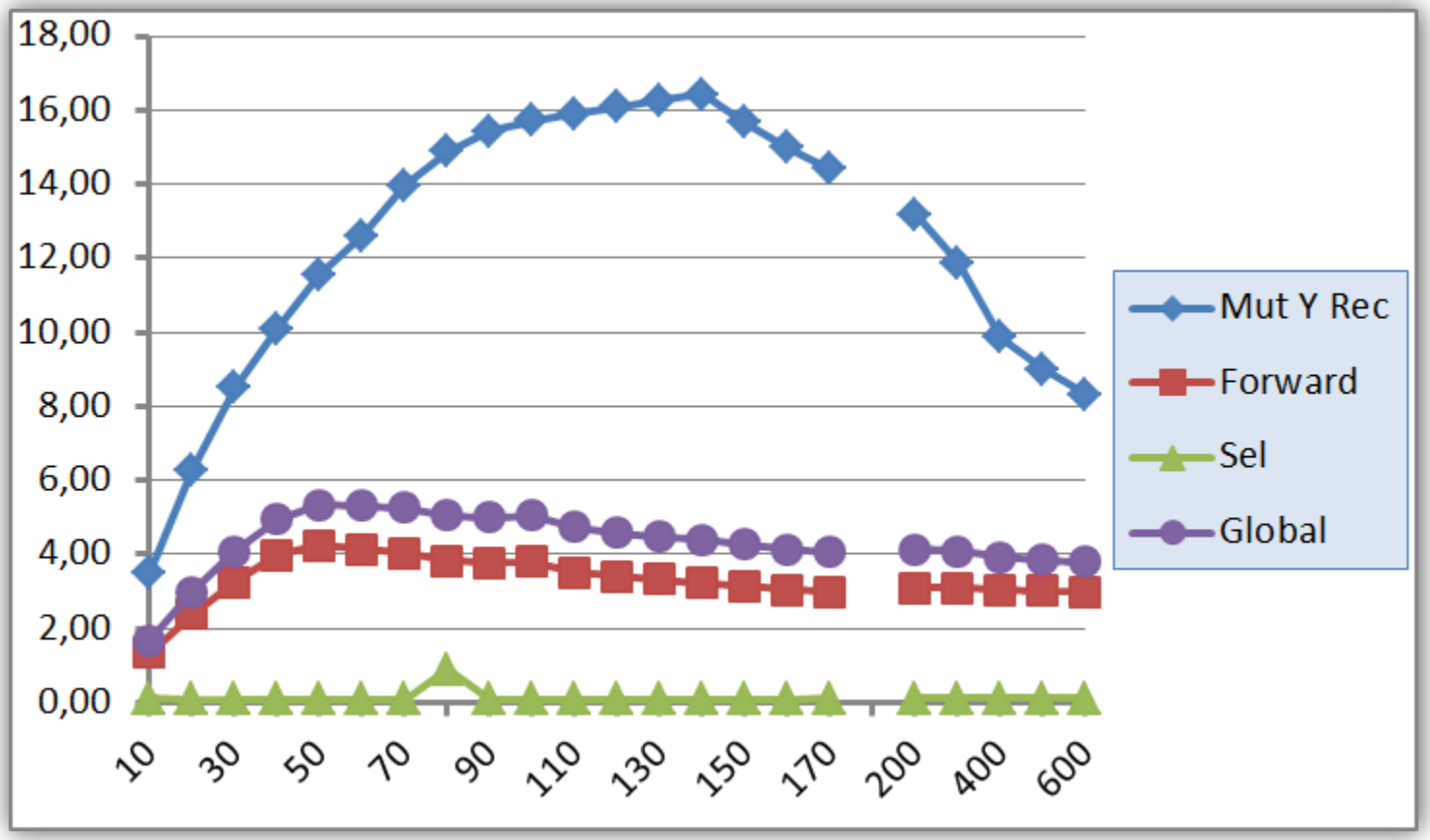
Resultados

Número variable de generaciones, G



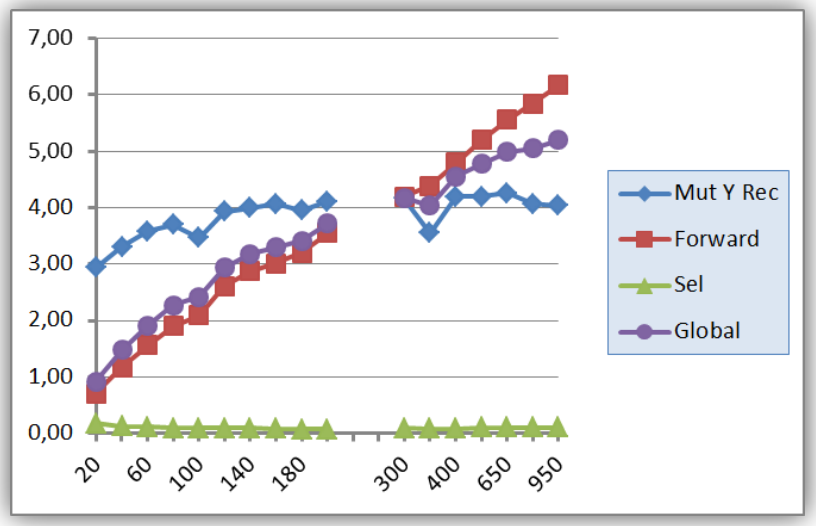
Resultados

Tamaño variable de las poblaciones, NP

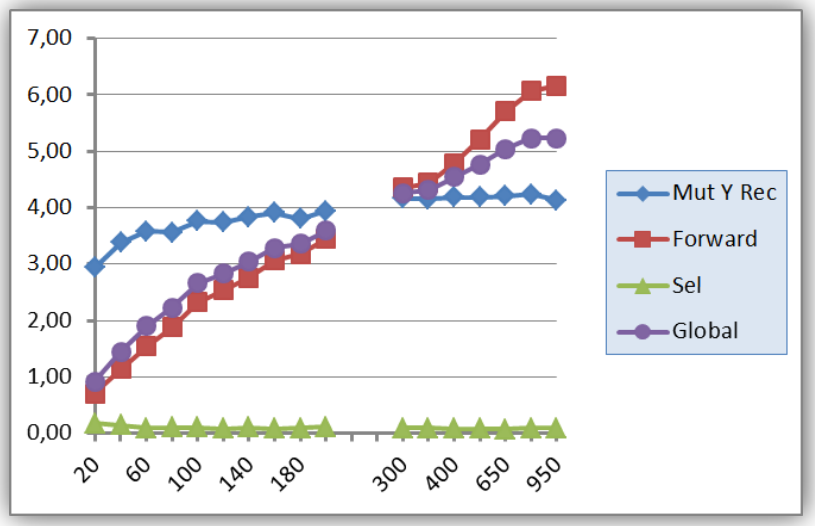


Resultados

Número variable de neuronas en cada capa oculta



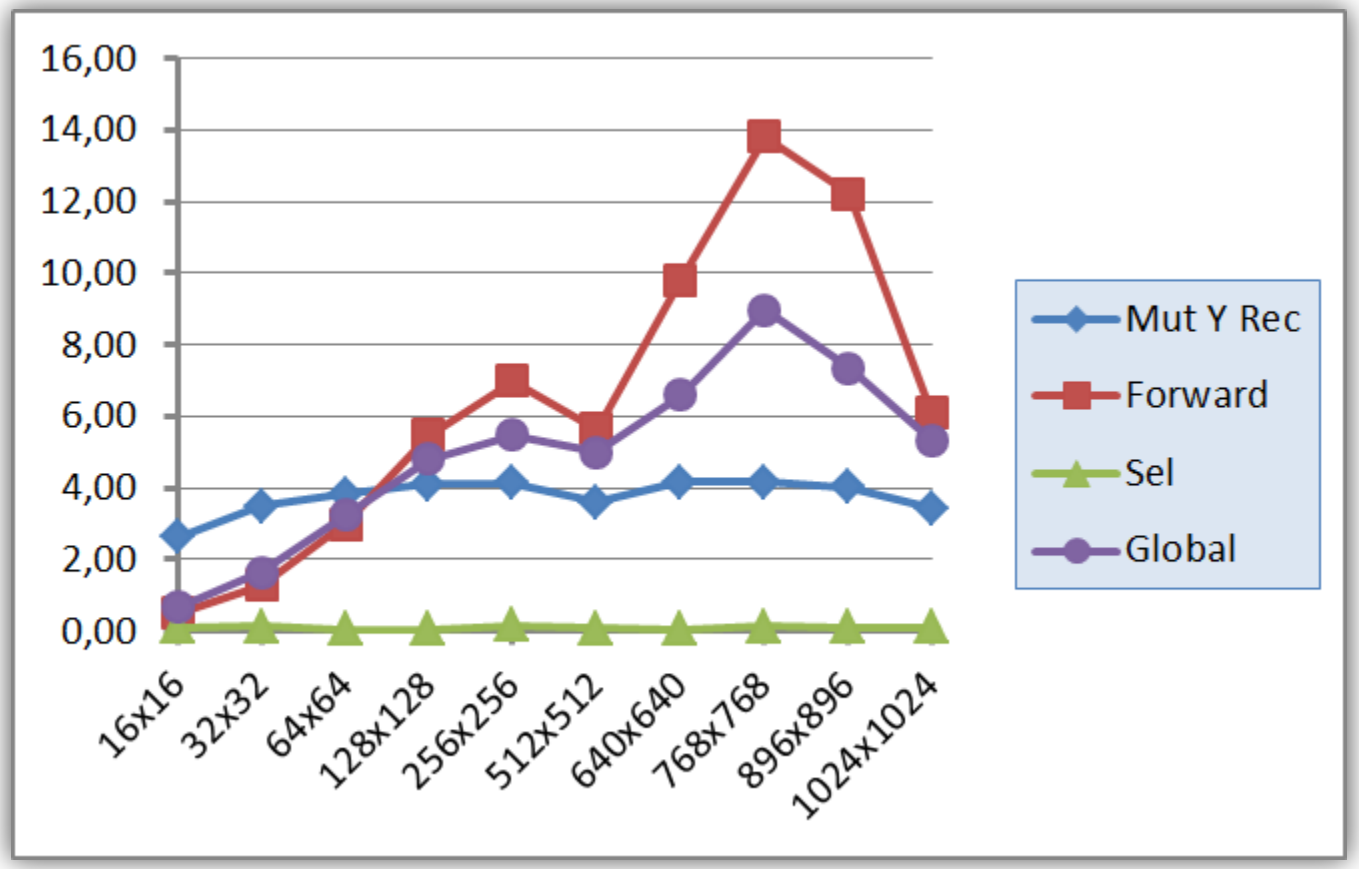
numH1



numH2

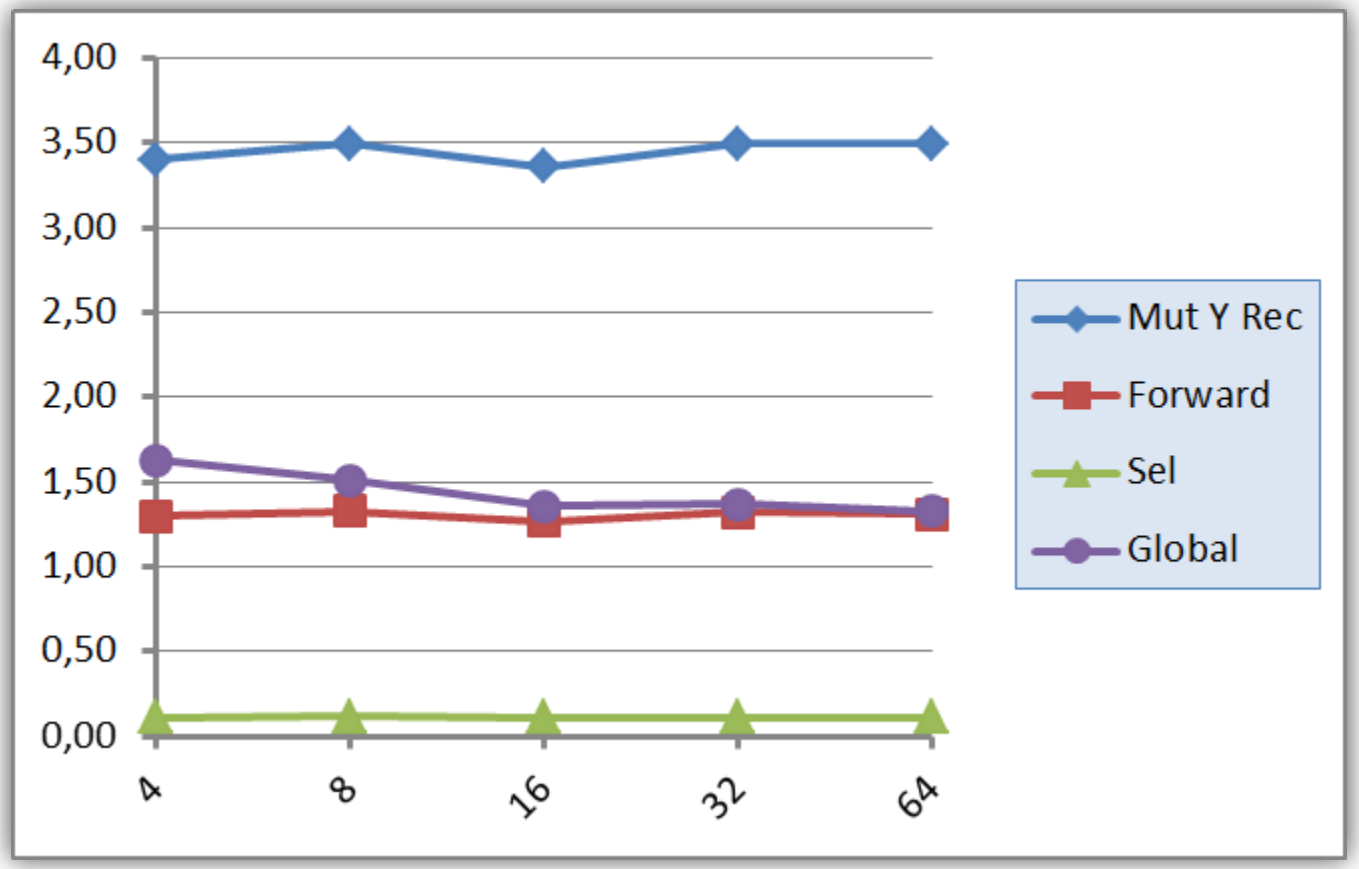
Resultados

Número variable de neuronas en ambas capas



Resultados

Número variable de muestras de entrenamiento



Resultados

Implementación sobre la plataforma

Recurso	Utilización (%)
Utilización lógica	86 %
Registros lógicos dedicados	39 %
Bloques de memoria	92 %
Bloques DSP	22 %

Frecuencia máxima de trabajo	130.75 MHz

Conclusiones

Sobre los objetivos

- ✓ Se ha conseguido implementar el algoritmo DE como método de evolución de los pesos de un conjunto de redes neuronales tipo MLP, utilizando kernels OpenCL como aceleradores de cómputo implementados sobre la FPGA, consiguiendo resultados de aceleración considerables frente a la implementación pura en el lado del host.

- ✓ Se consiguió implementar el módulo IP descrito en HDL de la tangente hiperbólica como función auxiliar en OpenCL. Sin embargo, no en un proceso estándar.

- ✓ En cuanto a lo personal, el trabajo a supuesto superar un desafío importante, con temas de gran interés personal y de los cuales casi no poseía experiencia previa.

Conclusiones

Trabajo futuro

- Multiplicación del camino de datos (SIMD):
  - Implementación del DE en el host.
  - Utilización de otras plataformas
- Integrar el módulo IP del RNG como función en OpenCL
- Evaluación de las capacidades de generalización de las redes entrenadas con el DE
- Análisis de similitud entre las diferentes redes evolucionadas con el DE
- Mejoras en los accesos a memoria
- Modificaciones del algoritmo DE
- Integrar la plataforma sobre el proyecto de investigación

Muchas gracias por su tiempo

Preguntas