Configuración y Optimización de Sistemas de Cómputo

Master Universitario en Ingeniería Informática
Depto. de Informática de Sistemas y Computadores (DISCA)
Universidad Politécnica de Valencia

Aceleradores Hardware

- Tipos de Aceleración
- Aceleradores Hardware
  - Co-procesadores
  - GPUs
  - FPGA
  - ASICs
- Programación de aceleradores
  - OpenCL/Cuda/SyCL
- Infraestructuras Heterogeneas

Aceleradores Hardware

- Los procesadores de propósito general tienen un rendimiento aceptable para un gran número de aplicaciones
  - Son capaces de ejecutar "cualquier" tipo de problema
  - No están optimizados para resolver un tipo de operaciones en concreto
    - No son la solución más eficiente
      - Rendimiento de pico
      - Consumo

Aceleradores Hardware

- Los aceleradores surgen con el propósito de optimizar la ejecución de ciertas tareas
  - Tareas frecuentes
  - Tareas costosas
- Su optimización repercute en una mejoría significativa de la eficiencia del sistema
  - Menor consumo (a igual servicio)
  - Mejor servicio
    - Menores latencias
    - Mayor capacidad

Tipos de aceleradores

- Integrados en el procesador (e.g co-procesadores)
- Integrados en el SoC
  - Coherentes / No coherentes
- Integrados en el Nodo
  - Chips diferentes

Co-procesadores

- Las CPU actuales incluyen co-procesadores o unidades de aritmética "especial" para acelerar la ejecución de determinadas tareas
- Normalmente los co-procesadores aceleran códigos pequeños, sensibles a la latencia
- Ejemplos: Instruciones SIMD, Vector units, cripto-extensions, …

Co-procesadores

- Expanden el juego de instrucciones del procesador y se integran en el pipeline de ejecución
  - Comparten la jerarquía de memoria con el núcleo de propósito general
  - Necesitan de soporte del compilador
  - Facilitan su uso al programador

Co-procesadores

- Unidades Vectoriales
  - En los 80/90 los procesadores vectoriales eran comunes para HPC
  - El avance de tecnología y la aparición de las GPUs hizo que quedaran en desuso
  - Operaciones vectoriales fueron incorporadas en el juego de instrucciones de CPUs
    - MMX Instrucciones SIMD para multimedia
    - SIMD → operaciones vectoriales tamaño fijo
  - Procesadores Actuales incorporan capacidades vectoriales/SIMD

Co-procesadores

- Unidades Vectoriales /SIMD
  - Extienden el ISA de la arquitectura
  - Requieren nuevas instrucciones y registros
  - Implementan operaciones vectoriales de tamaño fijo
    - 512 → 8 operaciones de 64 bits / 16 operaciones de 32
  - AVX (x86) (AVX-512, AVX-256, …)
    - AVX-512 extensions multiples que pueden ser implementadas de forma independiente
    - Soportado en procesadores Intel
      - Algunos cores AMD también implementan AVX
    - Códigos no compatibles entre versiones

Co-procesadores

- SVE (ARM) Scalable Vector Extensions
  - Implementacion de operaciones de ARM
  - El código es agnóstico al tamaño de la unidad vectorial
    - Codigo reusable entre SVE128 y SVE256
- ¿Como aprovecho con mi código la unidad vectorial?
  - Autovectorización (cuando esta disponible)
    - El compilador se encarga de aprovechar los recursos HW
  - Instrinsics
    - Primitivas implementadas en C que se insertan como llamadas al funciones (inline)
  - Ensamblador
    - El programador tiene alto conocimiento de la arquitectura e implementa las funciones a bajo nivel

Co-procesadores

- Diseño modular de Procesadores (RISC-V)
  - RISC-V es un juego de instrucciones abierto
  - Su juego de instrucciones tipo RISC (similar a ARM) esta formado por:
    - Juego básico de instrucciones
    - Módulos de expansión
      - Manipulaciones de bits
      - Vectoriales

Base ISA	Instructions	Description
RV32I	47	32-bit address space and integer instructions
RV32E	47	Subset of RV32I, restricted to 16 registers
RV64I	59	64-bit address space and integer instructions, along with several 32-bit integer instructions
RV128I	71	128-bit address space and integer instructions, along with several 64- and 32-bit instructions
Extension	**Instructions**	**Description**
M	8	Integer multiply and divide
A	11	Atomic memory operations, load-reserve/store conditional
F	26	Single-precision (32 bit) floating point
D	26	Double-precision (64 bit) floating point; requires F extension
Q	26	Quad-precision (128 bit) floating point; requires F and D extensions
C	46	Compressed integer instructions; reduces size to 16 bits

[2]

Co-procesadores

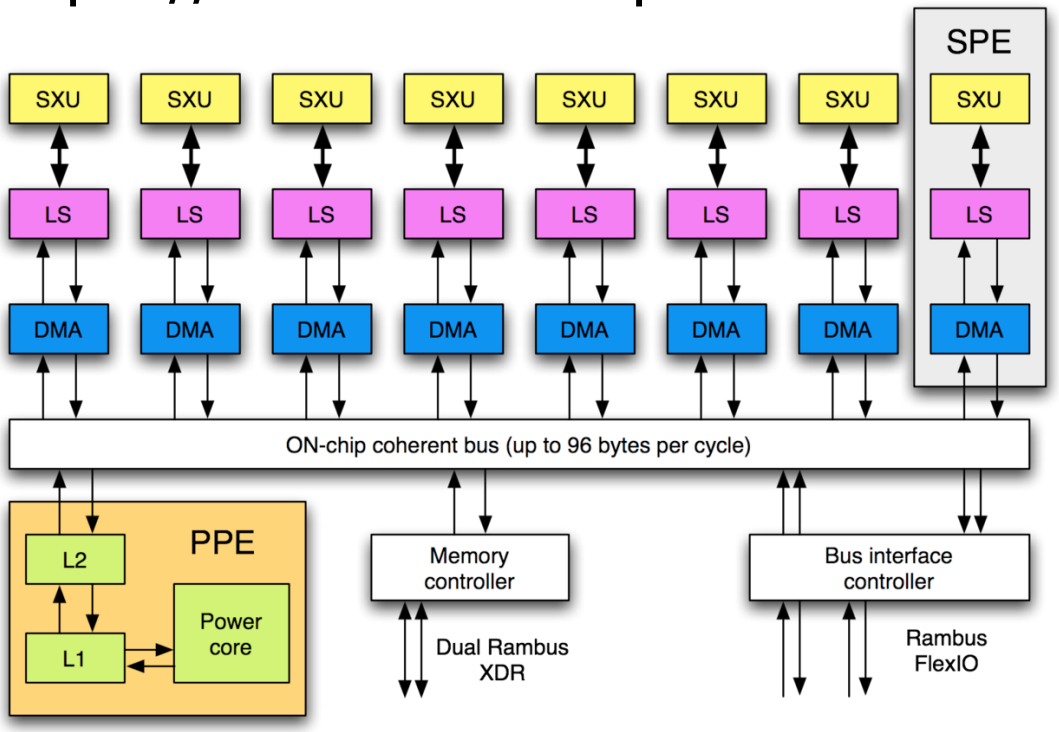
- Ventajas
  - Baja latencia para obtener resultados
  - Programación transparente al usuario
  - Menor coste hardware
- Desventajas
  - Necesitan soporte del compilador
  - Complican el diseño del pipeline del core

Aceleradores en el SoC

- El System-on-chip (SoC) de una CPU incluye los núcleos de computo, las interfaces de memoria, y otro periféricos
- En el caso de sistemas heterogéneos el SoC puede incluir aceleradores específicos
  - Los procesadores ARM para móviles incluyen una GPU Mali como aceleador
  - El procesador Cell de IBM (PS3) fue unos de los primeros procesadores en incorporar co-procesadores a nivel de SoC

Aceleradores en el SoC

- CELL IBM SoC
  - PPE core principal // SPE cores simples FPU

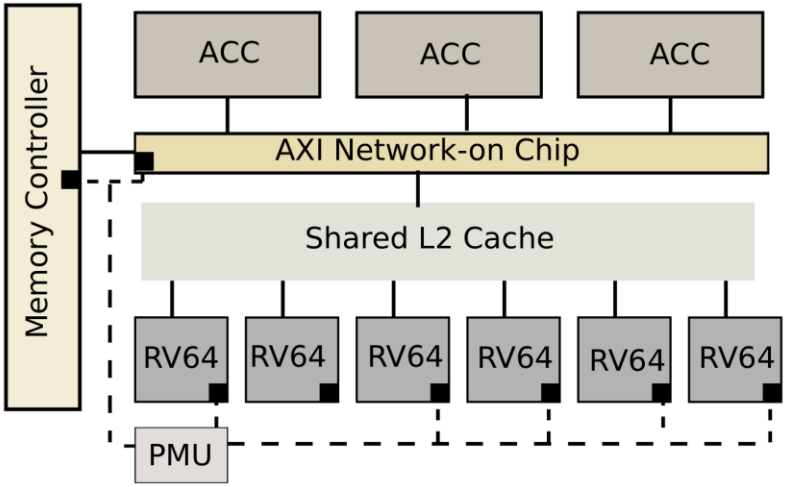


Aceleradores en el SoC

- Los aceleradores que se integran en el SoC comparten la memoria con la CPU
  - Opcionalmente puede compartir algún nivel de la memoria cache
- Coherencia de Cache
  - Es necesario aunque los aceleradores no gasten la cache
    - El espacio de memoria es compartido como saber que los datos en Memoria están actualizados

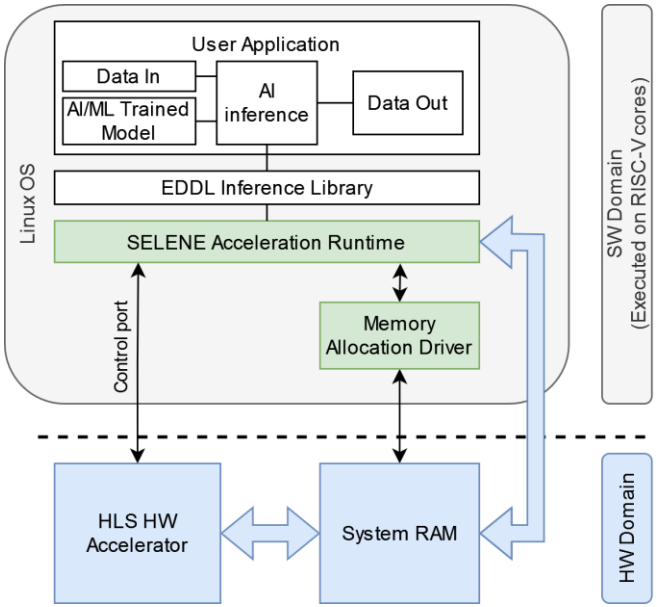
Aceleradores en el SoC

- Coherencia
  - Por Hardware → generalment cuando los acceleradores acceden a la cache
  - Por Software → los aceleradores no acceden a la cache
    - Al reservar memoria forzar flush/invalidate



Aceleradores en el SoC

- ¿Cómo se programan estos aceleradores?
  - El procesador hace una descarga (offloading) de una tarea al acelerador
    - Comunicación core → Acelerador
      - Paso de parámetros
      - Instrucciones
    - Los datos tienen que moverse
  - De esto se encarga el runtime
    - Lenguaje/API en el core
      - OpenCL, OpenACC, …
    - El código del acelerador
      - OpenCL, OpenGL, OpenACC, nada, …



Aceleradores en el SoC

- Ventajas
  - No necesitan instrucciones especificas
  - El diseño del core es independiente del acelerador
    - No hay que re-verificar otros componentes
- Inconvenientes
  - Gestión de la coherencia de cache
  - Comunicación/sincronización de aceleradores/core
    - El runtime tiene que ser eficiente
  - Mayor latencia en la obtención de resultados

Aceleradores Desacoplados

- Para operaciones muy "costosas" se utilizan aceleradores desacoplados
  - Chips diferentes para la CPU y el acelerador
  - Los aceleradores disponen de su propia memoria
  - Generalmente conectados a través de tarjetas de expansión conectados al bus PCI de la placa base
    - O otros enlaces propietarios
      - NVLINK, ARM CoreLink

GPUs

- Graphic Processor Units
  - Número muy elevado de unidades de coma flotante
  - Muy eficientes en computo matricial
  - Orientadas a capacidad no a latencia
  - Cores simples, en orden sin especulación, predicated instructions
- Los aceleradores más utilizados y los que a día de hoy consiguen mejor rendimiento en FLOPS/s
  - Gran eficiencia en la realización de operaciones de coma flotante
  - Para computo matricial más eficientes energéticamente que las CPUs

GPUs

- Lenguajes de programación específicos
  - Cuda
- Lenguajes multiplataforma
  - OpenCL
  - OpenACC
  - OpenVino (Intel)
- Runtime
  - Se encarga de el "offloading" y el movimiento de datos
  - Comunicación CPU / GPU
    - Polling o interrupciones

GPUs

- NVIDIA y AMD son los principales fabricantes

https://www.amd.com/es/graphics/instinct-server-accelerators

https://developer.nvidia.com/blog/nvidia-ampere-architecture-in-depth/

GPUs

- Ventajas
  - Mejoran sustancialmente la capacidad de computo de las CPUs o aceleradores en el SoC
  - No necesitan modificar/adecuar el procesador
  - Extienden las capacidades de nuestro sistema de forma compatible
- Desventajas
  - Necesitan código/compilador especifico para el acelerador
  - No útiles para acelerar kernels pequeños
  - Aunque son eficientes energeticamente contribuyen significativamente al consumo del sistema

ASICs

- Para funciones muy frecuentes y costosas → Chips específicos
  - Permiten reducir el consumo energético
  - Aumentan las prestaciones
- Ejemplo: Inferencia de redes neuronales
  - Google Tensor Processor Unit (TPU)
  - Mejora del proceso de inferencia
    - Arquitectura especifica (array sistólico)
    - Tipos de datos de baja precisión
  - Optimizada para su uso con TensorFlow

ASICs

- Ventajas
  - Siempre proporcionaran el mejor rendimiento para un problema concreto
- Desventajas
  - Poco flexibles
  - Elevado coste de diseño/fabricación

FPGAs

- Las FPGAs son dispositivos hardware programables
  - Pueden implementar cualquier circuito digital utilizando tablas (LUTs) y registros (flip-flops)
  - Las arquitecturas más modernas incorporan otros tipos de celdas
    - Unidades de coma flotante, DSPs, …
  - Los elementos de la FPGA se disponen en una cuadricula de manera que se pueden componer circuitos complejos

FPGAs

- Las FPGAs se pueden utilizar como acelerador HW desacoplado
  - Permiten implementar como un circuito funcionalidades que el SW no hace de forma eficiente
    - Cálculos criptográficos (operaciones a nivel de bit)
  - Tarjetas con conexión PCI express
  - La comunicación CPU y la FPGA es facilitada por el entorno de desarrollo SW
    - Runtime, API, compilador

FPGAs

- Programación FPGAs
  - Lenguajes de descripción de HW
    - Verilog, VHDL
  - Lenguajes de Alto Nivel utilizando herramientas de synthesis adecuadas (High-Level Synthesis)
    - C/C++ más pragmas para HLS
    - OpenCL
- El runtime se encarga de realizar el movimiento de datos y la sincronización con la CPU

FPGAs

- Ventajas
  - Reducido consumo respecto CPU/GPU
  - Flexibles
    - Se adaptan a cambios en las aplicaciones
  - Prestaciones
    - Eficientes cuando no es computo de coma flotante
- Desventajas
  - Difíciles de programar

Clusters Heterogeneos

- IBM RoadRunner
  - Cluster de procesadores Cell
  - Llego a ser el n1 del mundo
- Cloud TPU
  - Cluster de google que ofrece a clientes acceso a las TPUs
- Frontier (HPC)
  - Usa 4GPUs por CPU
- Fugaku (HPC)
  - Utiliza SVE de ARM

Bibliografia

1. N. Stephens *et al.*, "The ARM Scalable Vector Extension," in *IEEE Micro*, vol. 37, no. 2, pp. 26-39, Mar.-Apr. 2017, doi: 10.1109/MM.2017.35.
2. RISC-V OFFERS SIMPLE, MODULAR ISA New CPU Instruction Set Is Open and Extensible By David Kanter, Linley Group