FACULTAD DE CIENCIAS DE LA COMPUTACIÓN Y DISEÑO DIGITAL

INTEGRANTES

HERRERA MORAN GILMAR JAVIER

MENDOZA PARRAGA ANDY JOHEL

ZAMORA AGUILAR RONALDO WILFRIDO

CURSO

2DO SOFTWARE “B”

GRUPO

F

MATERIA

ARQUITECTURA DE COMPUTADORAS

TEMA DE LA EXPOSICIÓN

MICROARQUITECTURA DE PROCESADORES

ÍNDICE

1. Microarquitectura 3

1.1. Datos históricos 3

2. Diseño 3

3. Técnicas de organización y optimización en la microarquitectura 4

3.1. Pipeline 4

3.2. Predicción de saltos 5

3.3. Ejecución especulativa 6

4. Microarquitecturas modernas: Intel SkyLake vs AMD Zen 2 6

4.1. AMD Zen 2 6

4.1.1. Cachés en Zen 2 7

4.2. Intel Skylake 8

BIBLIOGRAFÍA 9

ANEXOS 11

MICROARQUITECTURA DE PROCESADORES

1. Microarquitectura

La microarquitectura define la implementación de una ISA en un procesador. Debido a las diferentes organizaciones y combinaciones de componentes dentro de un procesador, los diseños de microarquitectura bajo un proceso tecnológico específico pueden afectar la disipación de energía, el rendimiento, el área de la matriz, etc., de un núcleo [1].

La arquitectura del conjunto de instrucciones del procesador (ISA) es un modelo abstracto de la computadora que describe qué operaciones puede realizar directamente el procesador, qué datos, cómo y cuánto puede procesar. En ocasiones, la arquitectura del conjunto de instrucciones puede incluir múltiples sistemas de comandos [2].

1.1.Datos históricos

En 1971, Intel lanzó el chip Intel 4004 junto con su chipset, y nació el primer microprocesador comercial y con este la microarquitectura [3].

A mediados de la década de los 80, los investigadores intentaron aumentar aún más el ancho de banda de procesamiento, pero la compleja microarquitectura se convirtió en un gran obstáculo para la decodificación del conjunto de instrucciones. Esto condujo a la idea de la arquitectura informática de conjunto de instrucciones reducido (RISC) [3].

2. Diseño

La configuración de la microarquitectura consta de varios parámetros de diseño del procesador, cada uno de los cuales debe ajustarse en función de su impacto en la potencia y el rendimiento general del procesador. Seleccionar una configuración de microarquitectura implica una rigurosa exploración del espacio de diseño en un espacio de búsqueda que incluye todas las configuraciones posibles para los parámetros de diseño del procesador ajustables. Hay dos desafíos principales que deben abordarse en este proceso: [4].

En primer lugar, la metodología de exploración del espacio de diseño, empleada para seleccionar las configuraciones de microarquitectura de los procesadores, debe ser temporalmente eficiente. En segundo lugar, la metodología de exploración del espacio de diseño debe equilibrar el consumo de energía del procesador con el rendimiento, que son métricas de diseño contradictorias. No es posible obtener soluciones óptimas para problemas de optimización con métricas de diseño contradictorias [4].

3. Técnicas de organización y optimización en la microarquitectura

3.1. Pipeline

Las Pipeline de las CPU modernas son básicamente canalizaciones de ejecución superescalar fuera de orden, lo que puede mejorar considerablemente el rendimiento de la canalización. Por ejemplo, una instrucción no relacionada podría ejecutarse fuera de orden porque la instrucción anterior podría necesitar esperar para leer la memoria, lo que requiere muchos ciclos de instrucción [5].

Un motor de ejecución fuera de orden consta de varias partes: renombramiento de registros en orden, ejecución fuera de orden de instrucciones o microoperaciones, y retiro en orden. Para facilitar la ejecución fuera de orden y el envío secuencial de microoperaciones, el motor de ejecución fuera de orden incluye unidades de hardware como el búfer de reorden (ROB), el búfer de carga, el búfer de almacenamiento y la estación de reserva (RS). Cuando el motor de ejecución fuera de orden recibe microoperaciones del frontend, asigna recursos a estas microoperaciones. El orden de programación original se conserva en el búfer de reorden (ROB) para retirar las microoperaciones en orden [5].

El predecodificador extrae bloques de 16 bytes de la caché de instrucciones y detecta el inicio de cada instrucción. Las instrucciones predecodificadas se insertan en la Instruction Queue (IQ). El decodificador puede decodificar hasta cuatro instrucciones por ciclo desde la IQ, usando un decodificador complejo y tres simples. Instrucciones con más de cuatro µops son manejadas por el Microcode Sequencer (MS). Los µops decodificados también se almacenan en el Decoded Stream Buffer (DSB) para aumentar el throughput en loops donde la decodificación es un cuello de botella [6].

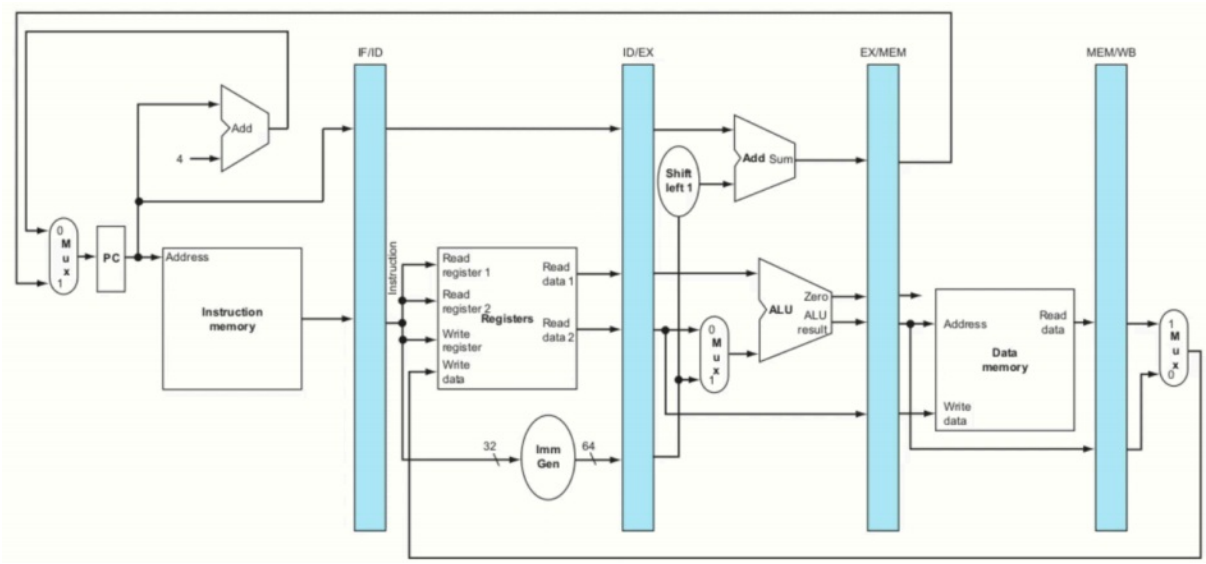


Figura 1. Pipeline

3.2. Predicción de saltos

Los procesadores modernos introducen técnicas de predicción de saltos para resolver los bloqueos en el pipeline debido a los peligros de control de flujo y mejorar el rendimiento de la ejecución de instrucciones. Sin embargo, debido a las características compartidas de las unidades predictoras de saltos y a los tiempos de ejecución retardados o estados microarquitectónicos que no se pueden revertir en caso de predicción incorrecta, los predictores de saltos están expuestos a amenazas significativas de ataques microarquitectónicos bien diseñados, que pueden ser lanzados desde software no privilegiado, resultando en consecuencias de seguridad más graves que los ataques de canal lateral tradicionales, como la filtración de claves criptográficas y la exfiltración de datos de enclaves SGX [7].

Los productores a menudo necesitan comunicar cambios en el flujo de control o condiciones excepcionales a los consumidores. Hacer esto mediante operaciones normales de encolado y desencolado sería ineficiente. En su lugar, Pipette proporciona valores de control (CVs). Los valores de control son similares a otros valores que se pasan a través de colas, excepto que transmiten cambios en el flujo de control en lugar de datos de aplicación. Antes de iniciar la ejecución, cada hilo registra un manejador de control de desencolado, similar a un manejador de excepciones. Un hilo que desencola o consulta una cola con un valor de control en la cabeza salta al manejador de control de desencolado (este salto ocurre completamente a nivel de usuario y no involucra al sistema operativo). El manejador recibe el valor de control y el ID de la cola que lo disparó, procesa el valor de control y luego vuelve al código principal de Pipette para continuar la computación [8].

3.3. Ejecución especulativa

La tecnología de ejecución especulativa reduce el bloqueo de la canalización al predecir la dirección de destino del salto de rama. Si la ejecución es incorrecta, se realiza una reversión de la canalización y se cancela la ejecución especulativa. Sin embargo, el contenido de la caché no se revierte [5].

Las predicciones de saltos precisas pueden ocultar una gran fracción de los stalls del frontend debido a la naturaleza desacoplada de los frontends de procesadores modernos. FDIP evita el acoplamiento entre la predicción de saltos y la recuperación de instrucciones, permitiendo que la prefetching guiada por predicción de ramas reduzca stalls del frontend. En las aplicaciones de centros de datos, eliminar predicciones erróneas de ramas no solo reduce los stalls del pipeline, sino que también disminuye los misses en la I-cache a través de FDIP, mostrando que la predicción de saltos es crítica para la eficiencia del pipeline y la ejecución especulativa [9].

Los procesadores modernos incorporan técnicas de ejecución especulativa para prevenir bloqueos en el pipeline. Este proceso implica predecir la siguiente instrucción a ejecutar basándose en el historial de ejecuciones previas, lo que permite que las instrucciones potenciales se ejecuten con anticipación. Sin embargo, esta técnica de optimización tiene serias fallas de diseño. Las predicciones incorrectas resultan en que las instrucciones especulativas y sus implicaciones arquitectónicas sean descartadas. No obstante, rastros de estas acciones se retienen en el estado microarquitectónico, como en la caché y el TLB, lo que representa riesgos de seguridad [10].

4. Microarquitecturas modernas: Intel SkyLake vs AMD Zen 2

4.1. AMD Zen 2

El procesador “Zen 2” está diseñado para satisfacer las necesidades de diversos mercados, como servidores, ordenadores de sobremesa, dispositivos móviles y estaciones de trabajo. El núcleo ofrece mejoras significativas de rendimiento y eficiencia energética respecto a Zen gracias a cambios en la microarquitectura, que incluyen un nuevo predictor de ramificación TAGE, una caché de operaciones de doble tamaño y una unidad de punto flotante de doble ancho. Basándose en el diseño del núcleo, un enfoque modular de chiplets proporciona flexibilidad y escalabilidad hasta 64 núcleos por zócalo con un total de 256 MB de caché L3 [11].

Los modelos de rendimiento son fundamentales para optimizar el código sensible al rendimiento. Al modelar el uso de unidades funcionales de CPU x86-64 fuera de servicio, la

disponibilidad de datos varía según el fabricante: existen asignaciones de instrucciones a puertos para los procesadores Intel, mientras que la información para los diseños de AMD es insuficiente. Esta disparidad se debe a que las técnicas estándar para inferir asignaciones de puertos exactas requieren contadores de rendimiento de hardware que AMD no proporciona [12].

El núcleo de CPU Zen 2, presenta dos mejoras principales con respecto a su predecesor, Zen. En primer lugar, la eficiencia energética se duplica gracias a una combinación de mejoras tecnológicas y de microarquitectura. En segundo lugar, el IPC aumenta aproximadamente un 15 % gracias a los cambios de microarquitectura en el front-end ordenado, la ejecución de enteros, la ejecución de coma flotante/vectorial, la carga/almacenamiento y la jerarquía de caché [11].

Zen 2 utiliza un diseño modular en múltiples niveles. Cuatro núcleos se agrupan en un Complejo de Núcleos (CCX, también Complejo de CPU). Un Matriz de Complejo de Núcleos (CCD) consta de dos CCX. Hasta ocho CCD se conectan a una matriz de E/S. (a) Matriz de Complejo de Núcleos (CCD) con Complejos de Núcleos (CCX) en procesadores de hasta 64 núcleos. Según el número de núcleos, dos o uno de los CCD se conectan al mismo conmutador dentro de la red de matrices de E/S. Cada conmutador de la matriz de E/S que conecta los CCD también conecta un controlador de memoria con dos canales de memoria, lo que puede generar cuatro nodos de acceso a memoria no uniforme (NUMA) [13].

4.1.1. Cachés en Zen 2

Cada núcleo tiene un front-end común que obtiene instrucciones para dos hilos de hardware independientes. La ventana de obtención tiene un ancho de 32B y alimenta a un decodificador de 4 vías. El back-end se divide en dos partes: una parte comprende cuatro Unidades Lógicas Aritméticas y tres Unidades de Generación de Direcciones (AGU), la otra contiene dos unidades de Multiplicación-Adición de Punto Flotante (FMA) de 256 bits de ancho y dos unidades de adición de Punto Flotante de 256 bits de ancho. Las AGU se pueden usar para dos cargas y un almacenamiento por ciclo, donde cada una de estas puede transferir hasta 32B de datos. Cada núcleo del procesador contiene una caché de operaciones para 4096 operaciones, cachés L1I y L1D de 32 KiB, y una caché L2 de 512 KiB, que se utiliza para instrucciones y datos. Además, cada CCX contiene 16 MiB de caché L3, distribuidos en cuatro porciones con 4 MiB cada una [13].

4.2. Intel Skylake

La microarquitectura Skylake incluye: (a) múltiples decodificadores 1:1 que pueden traducir macrooperaciones simples que solo se descomponen en una microoperación; (b) un decodificador 1:4 que puede traducir macrooperaciones complejas que pueden descomponerse en entre una y cuatro microoperaciones; y (c) una ROM de microsecuenciación (MSROM) que traduce instrucciones microcodificadas más complejas. Una sola macrooperación puede traducirse en más de cuatro microoperaciones, lo que potencialmente implica múltiples ramificaciones y bucles, ocupando varios ciclos de decodificación. El pipeline de decodificación puede proporcionar un ancho de banda máximo de 5 microoperaciones por ciclo [14].

En la Figura 2, se muestra una ilustración simplificada de un núcleo único de la microarquitectura Skylake de Intel. Las instrucciones se decodifican en µOP y se ejecutan desordenadamente en el motor de ejecución por unidades de ejecución individuales [15].

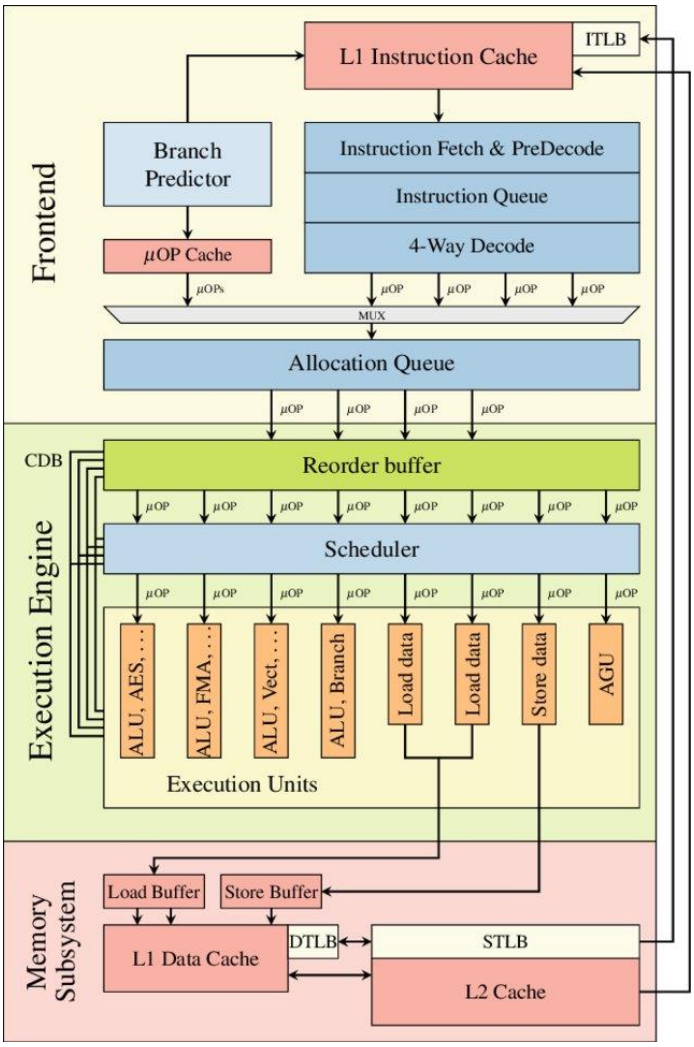


Figura 2. Diagrama de microarquitectura Intel Skylake [15]

BIBLIOGRAFÍA

[1] C. Bai, Q. Sun, J. Zhai, Y. Ma, B. Yu, and M. D. F. Wong, "BOOM-Explorer: RISC-V BOOM Microarchitecture Design Space Exploration," *ACM Transact Des Autom Electron Syst*, vol. 29, no. 1, pp. 1–23, Jan. 2024, doi: 10.1145/3630013.

[2] T. R. Zmyzgova, A. V. Solov'ev, E. N. Polyakova, and D. I. Dik, "The Impact of Hardware Features on the Processor Instruction Set Architecture," in *2020 International Multi-Conference on Industrial Engineering and Modern Technologies (FarEastCon)*, IEEE, Oct. 2020, pp. 1–5. doi: 10.1109/FarEastCon50210.2020.9271271.

[3] F. H. Khan, M. A. Pasha, and S. Masud, "Advancements in Microprocessor Architecture for Ubiquitous AI—An Overview on History, Evolution, and Upcoming Challenges in AI Implementation," *Micromachines (Basel)*, vol. 12, no. 6, p. 665, Jun. 2021, doi: 10.3390/mi12060665.

[4] P. Kansakar and A. Munir, "Selecting Microarchitecture Configuration of Processors for Internet of Things (IoT)," *IEEE Trans Emerg Top Comput*, pp. 1–1, 2018, doi: 10.1109/TETC.2018.2817923.

[5] Q. Ke, C. Wang, H. Wang, Y. Lyu, Z. Xu, and D. Wang, "Model Checking for Microarchitectural Data Sampling Security," in *2022 7th IEEE International Conference on Data Science in Cyberspace (DSC)*, IEEE, Jul. 2022, pp. 56–63. doi: 10.1109/DSC55868.2022.00015.

[6] A. Abel and J. Reineke, "uiCA," in *Proceedings of the 36th ACM International Conference on Supercomputing*, New York, NY, USA: ACM, Jun. 2022, pp. 1–14. doi: 10.1145/3524059.3532396.

[7] Q. Wang, M. Tang, K. Xu, and H. Wang, "Unveiling and Evaluating Vulnerabilities in Branch Predictors via a Three-Step Modeling Methodology," *ACM Transactions on Architecture and Code Optimization*, vol. 22, no. 1, pp. 1–26, Mar. 2025, doi: 10.1145/3711923.

[8] Q. M. Nguyen and D. Sanchez, "Pipette: Improving Core Utilization on Irregular Applications through Intra-Core Pipeline Parallelism," in *2020 53rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, IEEE, Oct. 2020, pp. 596–608. doi: 10.1109/MICRO50266.2020.00056.

[9] T. A. Khan *et al.*, “Whisper: Profile-Guided Branch Misprediction Elimination for Data Center Applications,” in *2022 55th IEEE/ACM International Symposium on Microarchitecture (MICRO)*, IEEE, Oct. 2022, pp. 19–34. doi: 10.1109/MICRO56248.2022.00017.

[10] H. Jang, T. Kim, and Y. Shin, “SysBumps: Exploiting Speculative Execution in System Calls for Breaking KASLR in macOS for Apple Silicon,” in *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, New York, NY, USA: ACM, Dec. 2024, pp. 64–78. doi: 10.1145/3658644.3690189.

[11] D. Suggs, M. Subramony, and D. Bouvier, “The AMD ‘Zen 2’ Processor,” *IEEE Micro*, vol. 40, no. 2, pp. 45–52, Mar. 2020, doi: 10.1109/MM.2020.2974217.

[12] F. Ritter and S. Hack, “Explainable Port Mapping Inference with Sparse Performance Counters for AMD’s Zen Architectures,” in *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3*, New York, NY, USA: ACM, Apr. 2024, pp. 317–330. doi: 10.1145/3620666.3651363.

[13] R. Schone, T. Ilsche, M. Bielert, M. Velten, M. Schmidl, and D. Hackenberg, “Energy Efficiency Aspects of the AMD Zen 2 Architecture,” in *2021 IEEE International Conference on Cluster Computing (CLUSTER)*, IEEE, Sep. 2021, pp. 562–571. doi: 10.1109/Cluster48925.2021.00087.

[14] X. Ren, L. Moody, M. Taram, M. Jordan, D. M. Tullsen, and A. Venkat, “I See Dead µops: Leaking Secrets via Intel/AMD Micro-Op Caches,” in *2021 ACM/IEEE 48th Annual International Symposium on Computer Architecture (ISCA)*, IEEE, Jun. 2021, pp. 361–374. doi: 10.1109/ISCA52012.2021.00036.

[15] M. Lipp *et al.*, “Meltdown,” *Cryptography and Security*, Jan. 2018, doi: 10.48550/arXiv.1801.01207.

ANEXOS

Link diapositivas:

https://docs.google.com/presentation/d/13l34oLwepYnKM8D0ZW6k3041VqaWF0kq1SElKbsS58E/edit?usp=sharing

Link GitHub:

https://github.com/AndyMendoza0308/Exposicion-Grupo-F---Microarquitectura.git

MICROARQUITECTURA DE PROCESADORES

INTEGRANTES:

HERRERA MORAN GILMAR JAVIER

MENDOZA PARRAGA ANDY JOHEL

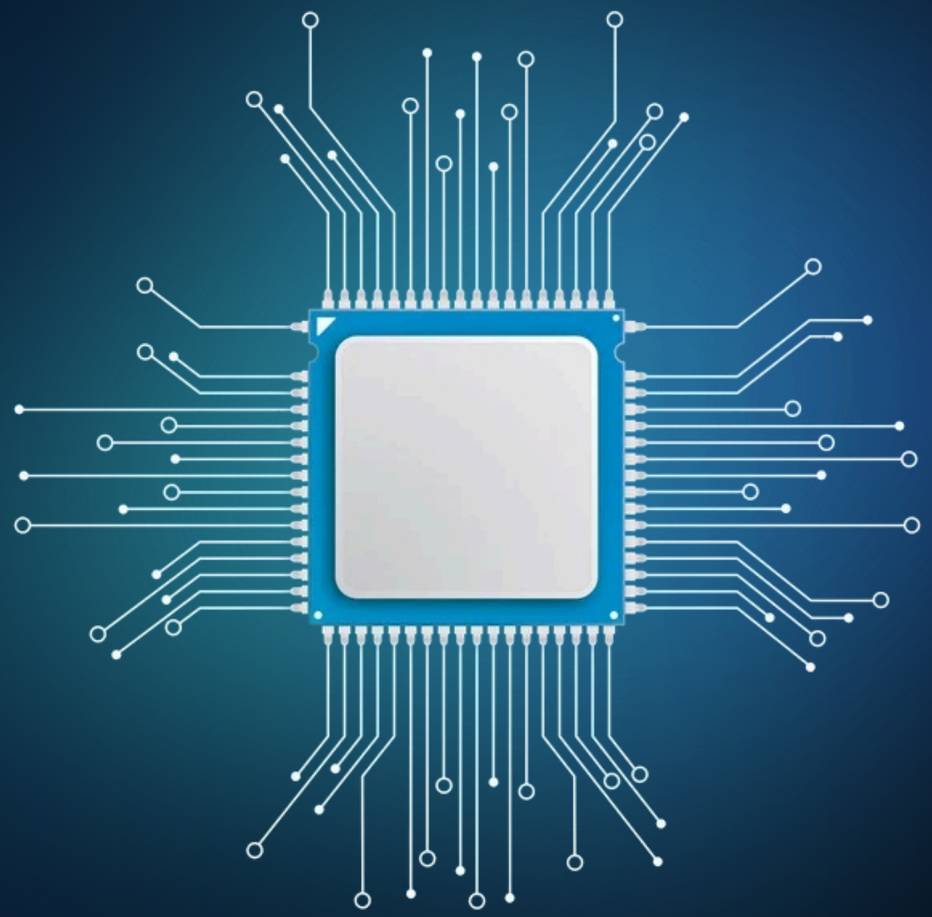
ZAMORA AGUILAR RONALDO WILFRIDO

CURSO:

Segundo Software “B”

GRUPO:

F



Índice

1 ¿Qué es la microarquitectura?

2 Datos históricos

3 Diseño de una microarquitectura

4 Fases de diseño

5 Técnicas de organización y optimización en la microarquitectura

6 Pipeline

7 Predicción de salto y ejecución especulativa

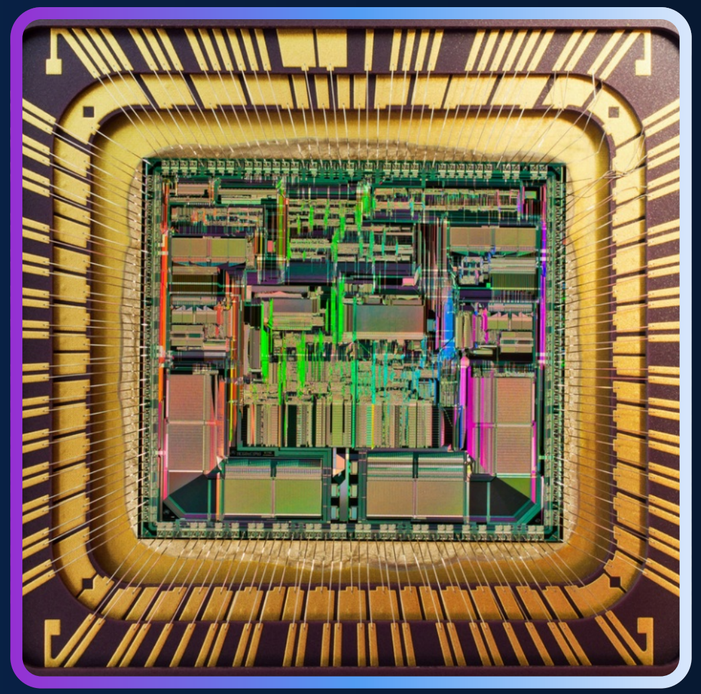
8 Microarquitecturas Modernas: AMD Zen 2 vs Intel Skylake

Awesome Web Browser

www.grupof_microarquitectura.com

¿Qué es la microarquitectura?

Es la implementación de una ISA (Arquitectura de Conjunto de Instrucciones) en un procesador.



Awesome Web Browser

Datos históricos

1971

Intel 4004, primer microprocesador

1980's

Complejidad impulsa la arquitectura RISC

Awesome Web Browser

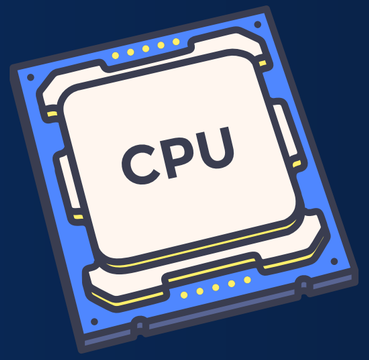
www.grupof_microarquitectura.com

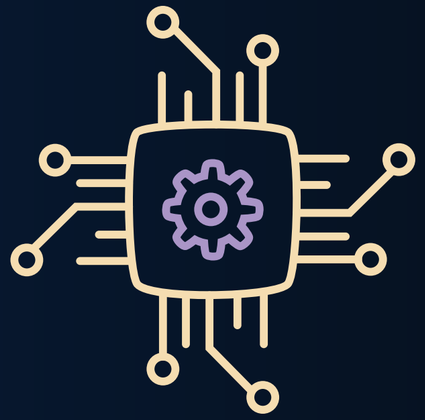
Diseño de una microarquitectura

Ajuste de parámetros de diseño

Exploración del espacio de diseño

Desafíos





Awesome Web Browser

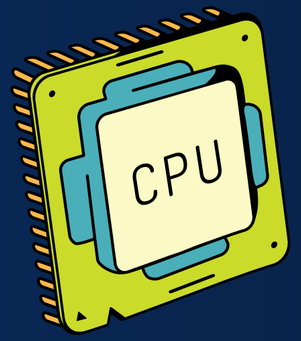
www.grupof_microarquitectura.com

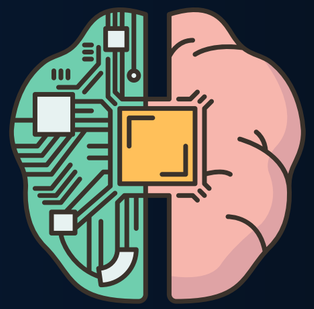
FASES DE DISEÑO

Selección del ISA → Definición de bloques funcionales → Organización del flujo de ejecución → Diseño modular y descripción abstracta → Simulación y ajuste de parámetros → Optimización final

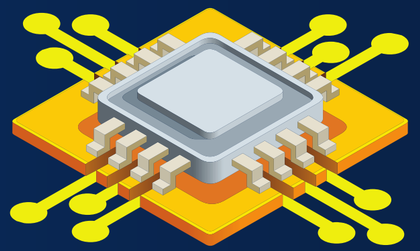
Awesome Web Browser

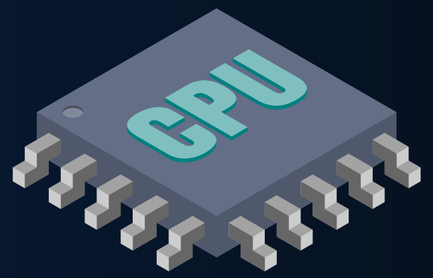
www.grupof_microarquitectura.com





Técnicas de organización y optimización en la microarquitectura



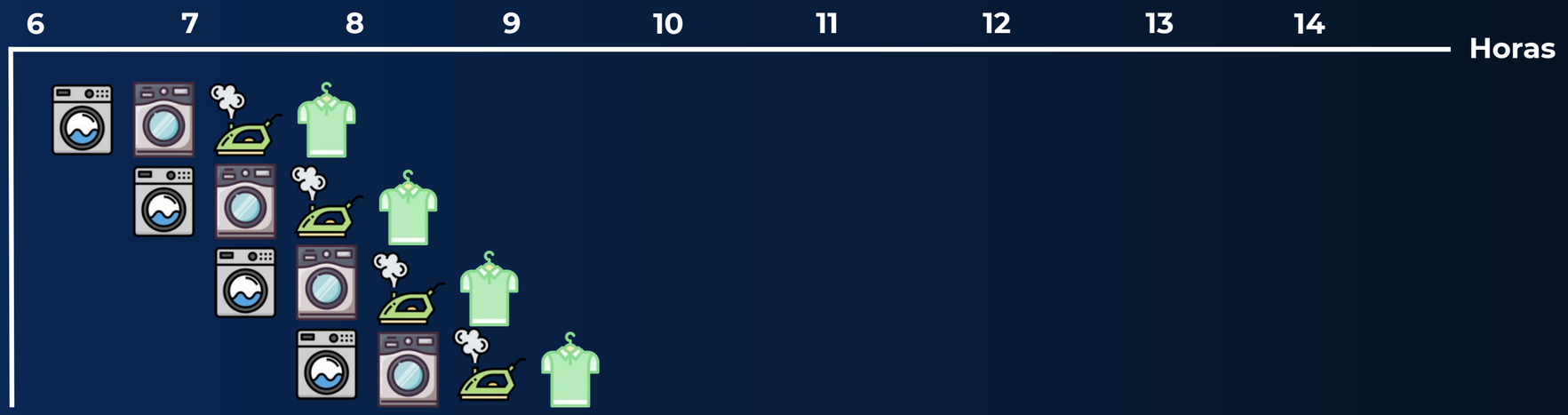


Awesome Web Browser

www.grupof_microarquitectura.com

Pipeline

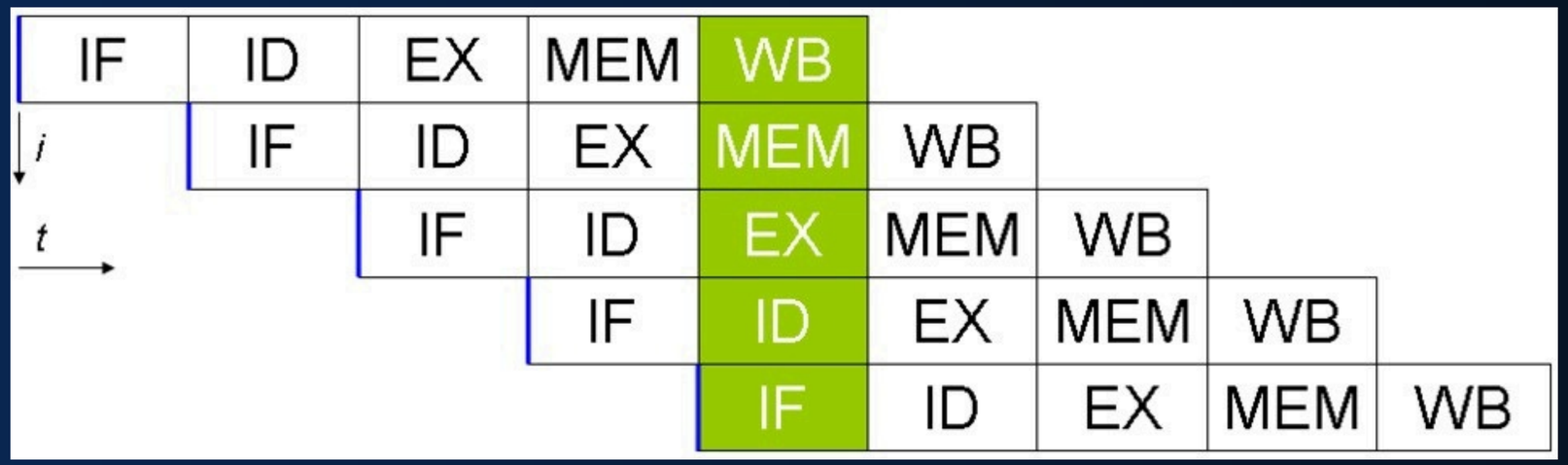
Es una técnica para mejorar el rendimiento de un procesador.



Pipeline

Ciclo de 5 estados:

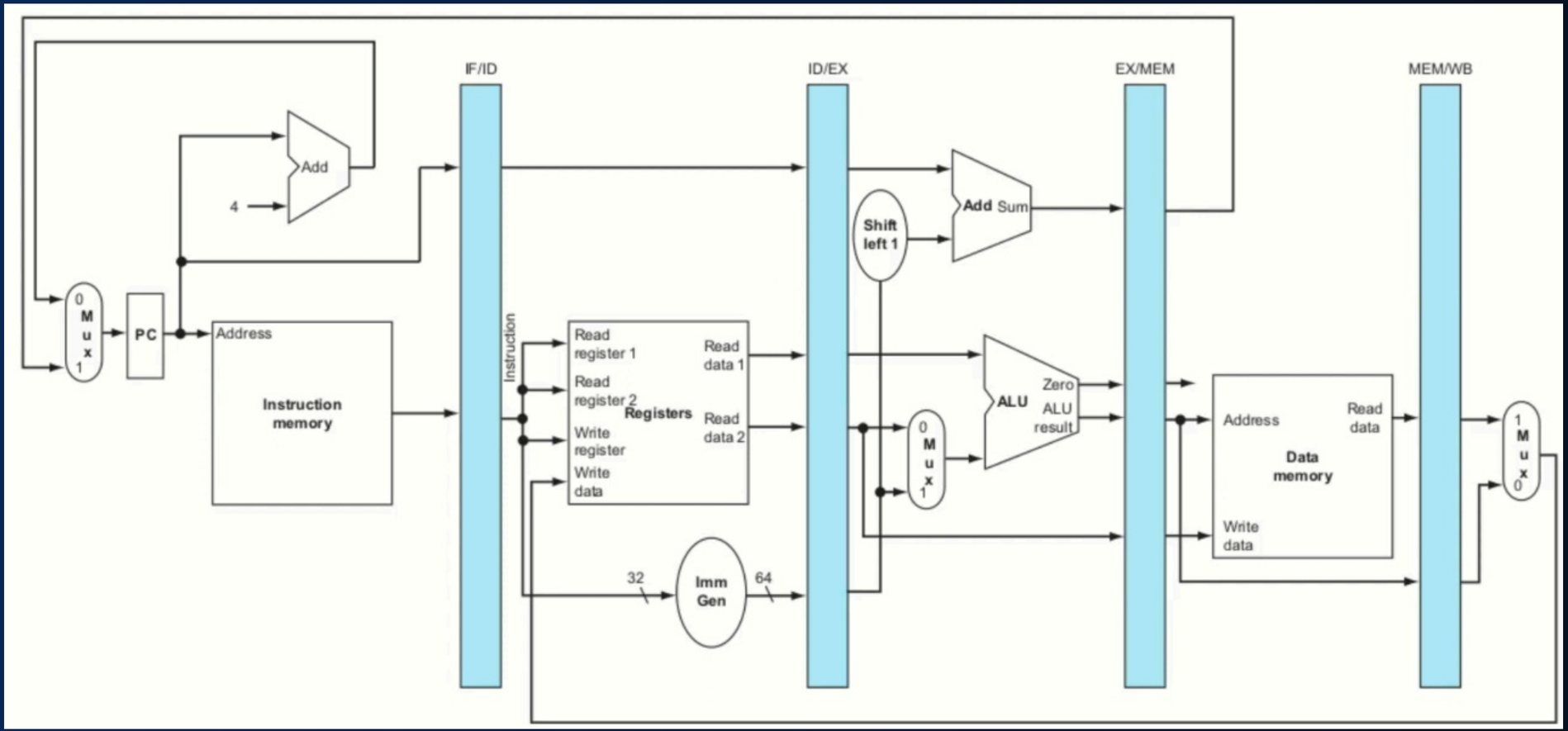
- IF._ Instruction Fetch
- ID._ Instruction Decode
- EX._ Execute
- MEM._ Memory Access
- WB._ Write Back



Awesome Web Browser

www.grupof_microarquitectura.com

Pipeline



Awesome Web Browser

www.grupof_microarquitectura.com

Predicción de saltos

- Técnica para evitar bloqueos por cambios de flujo de control.
- Mejora el rendimiento del pipeline al predecir con antelación las instrucciones siguientes.
- Evita stalls del pipeline por saltos condicionales.
- Crucial para ejecución fuera de orden y especulativa.

Ejecución Especulativa

- Técnica que predice y ejecuta instrucciones antes de confirmar su necesidad.
- Si la predicción es incorrecta, se descarta la ejecución y se revierte la canalización.
- Reduce bloqueos en el pipeline.
- Mejora el rendimiento en procesadores modernos con pipelines profundos.

Awesome Web Browser

www.grupof_microarquitectura.com

Microarquitecturas Modernas

Awesome Web Browser

www.grupof_microarquitectura.com

AMD ZEN 2

Diseñado para servidores, PCs, móviles y estaciones de trabajo

Diseñado para servidores, PCs, móviles y estaciones de trabajo

Microarquitectura Mejorada

Predictor de ramificación TAGE

Caché de operaciones de doble tamaño

Unidad de punto flotante de doble ancho

Awesome Web Browser

www.grupof_microarquitectura.com

Intel Skylake

Incluye:

1:1: macrooperaciones simples
↓
1 microoperación

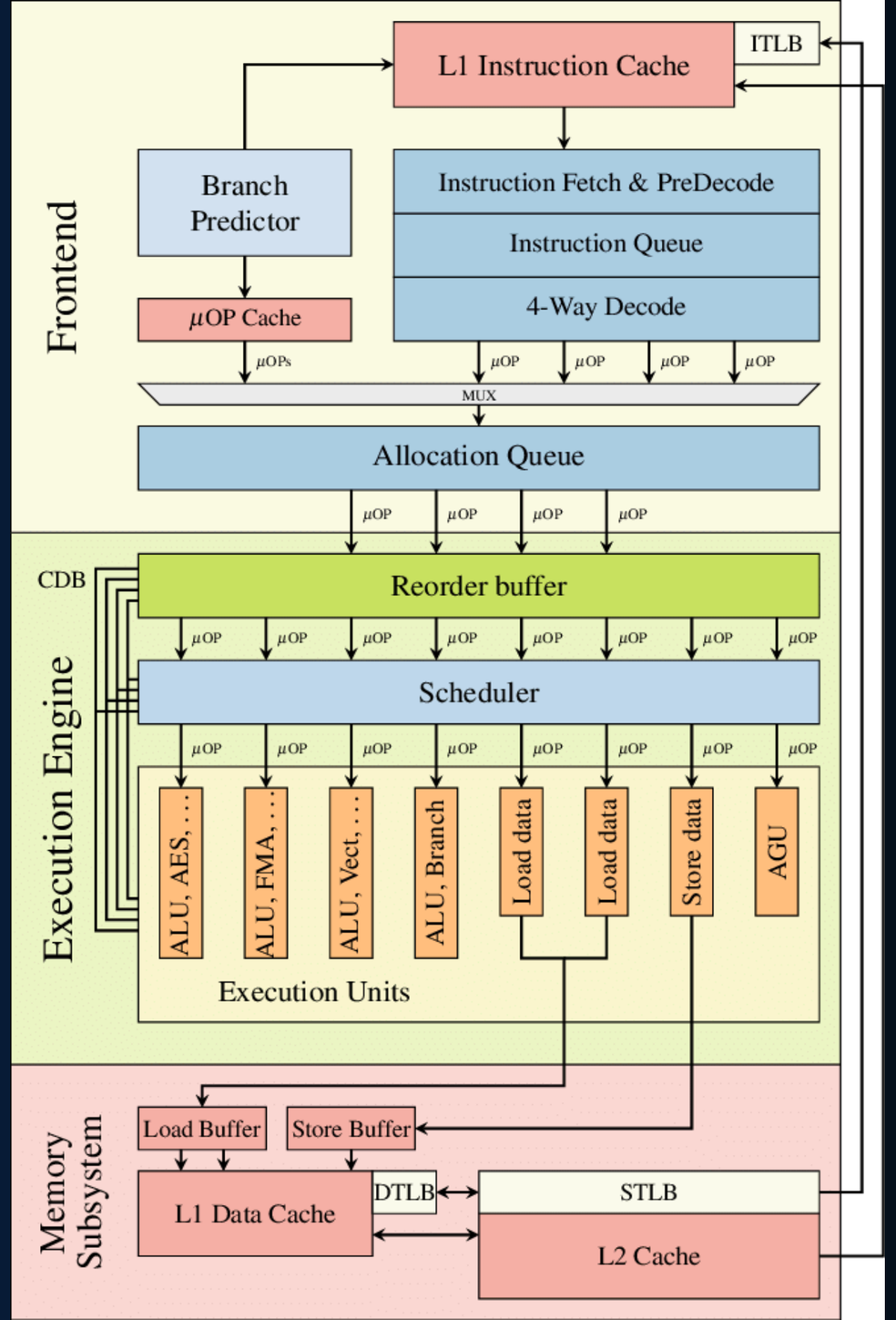
1:4: macrooperaciones complejas
↓
1–4 microops

MSROM
instrucciones microcodificadas complejas

Macrooperaciones >4 microops
↓
Múltiples ciclos/ramificaciones

Ancho de banda máximo
↓
5 microops por ciclo

Diagrama de microarquitectura Intel Skylake



GRACIAS POR SU ATENCIÓN