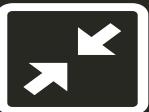
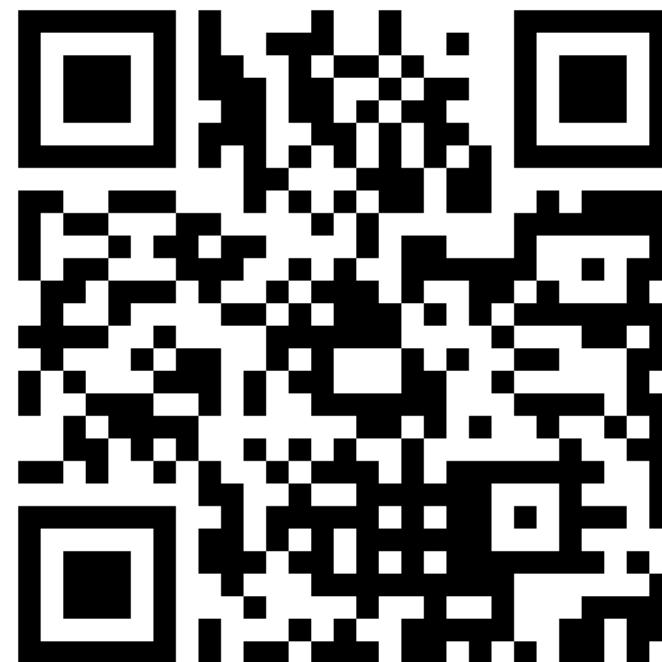


Informática I

Claudio Paz

Marzo 2023



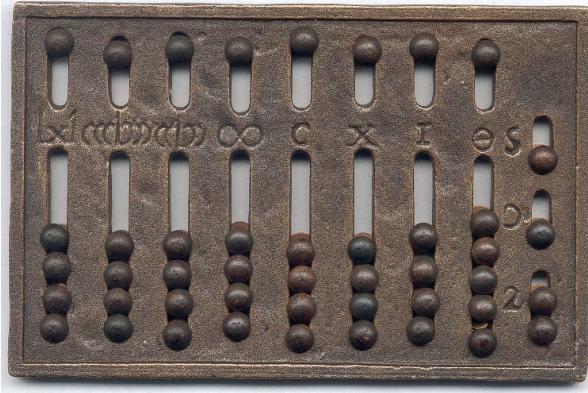
Unidad 1

Estructura de una computadora

Antecedentes históricos

Antecedentes históricos

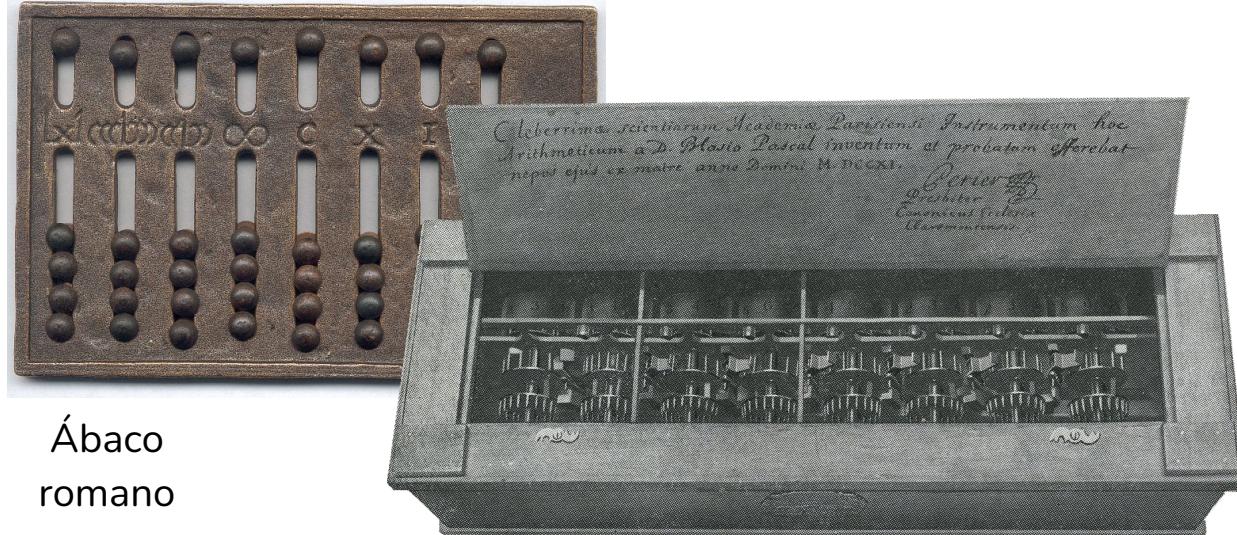
En la antigüedad



Ábaco
romano

Antecedentes históricos

En la antigüedad

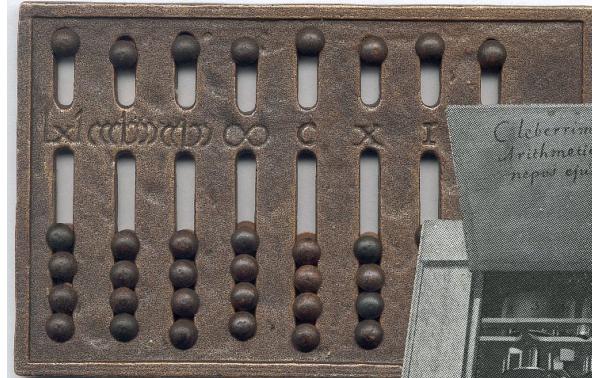


Ábaco
romano

Pascalina

Antecedentes históricos

En la antigüedad



Ábaco
romano



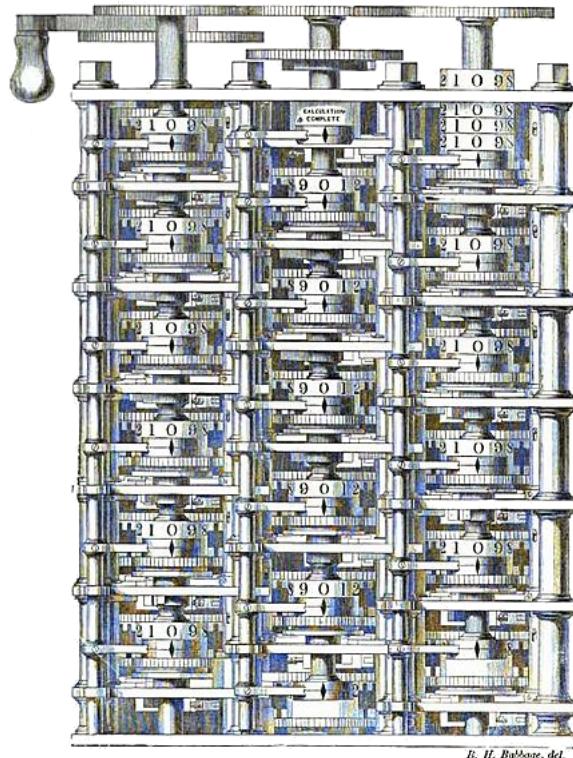
Pascalina



Ábaco
Neperiano

Antecedentes históricos

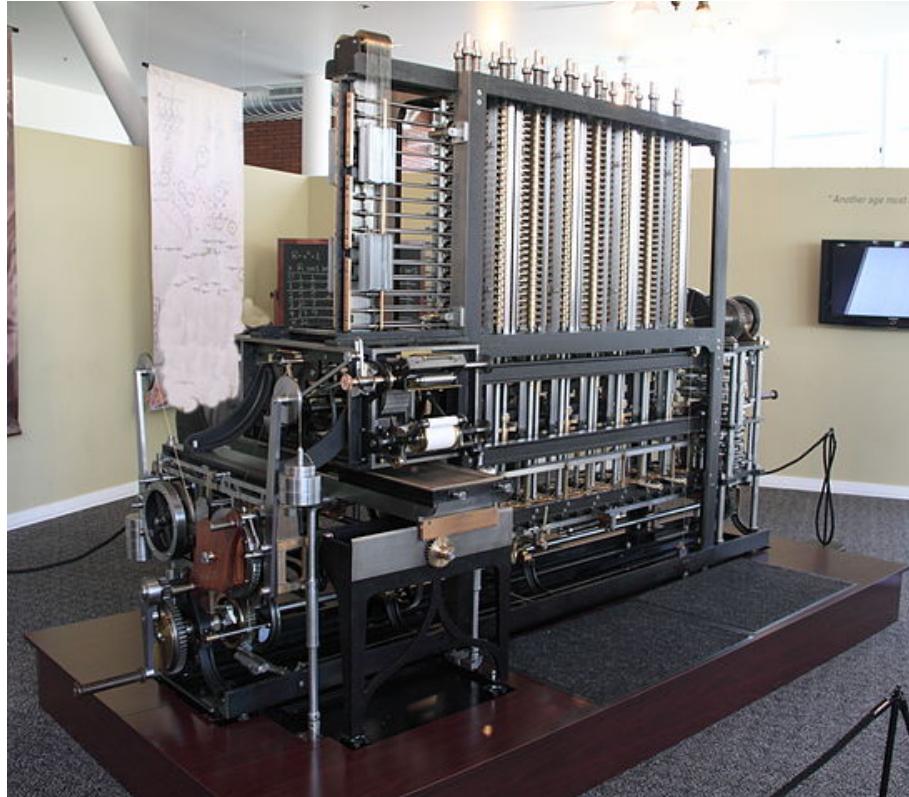
Primer dispositivo de cómputo de propósitos generales (Teórico)



Difference Engine, *Charles Babbage*. 1822.

Antecedentes históricos

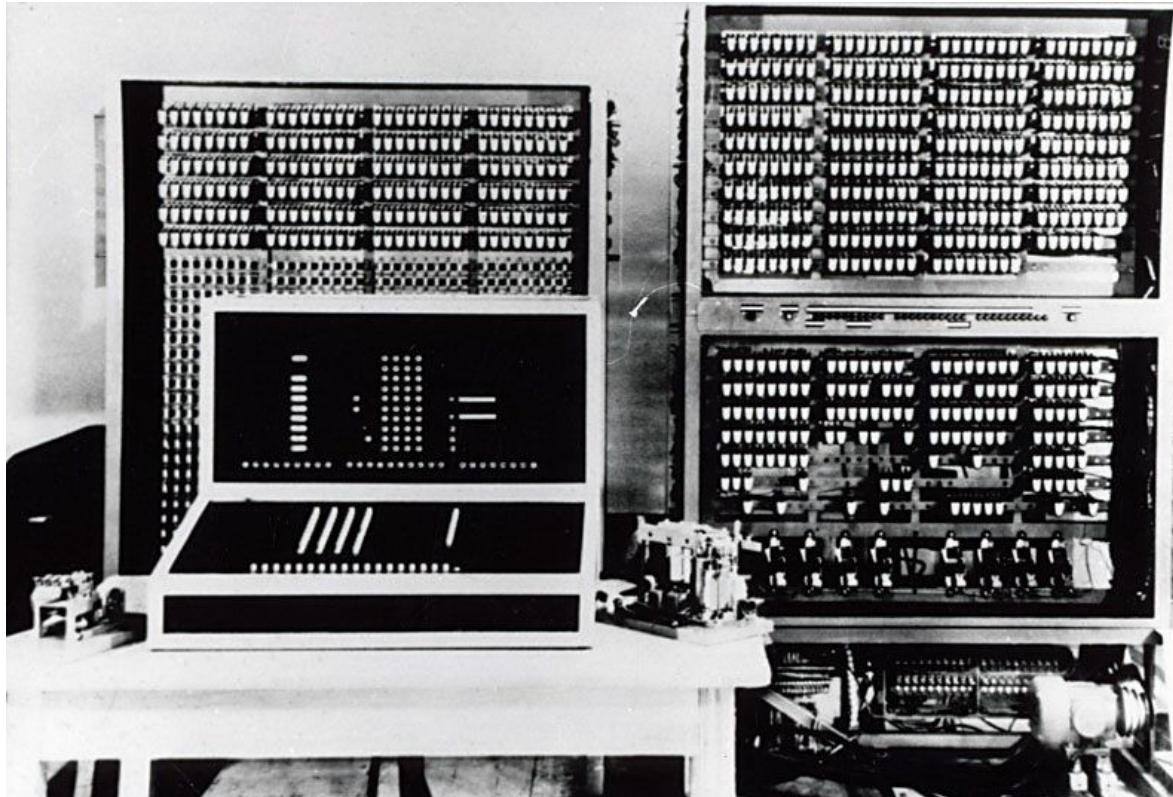
Primer dispositivo de cómputo de propósitos generales (teórico)



Difference Engine, *Charles Babbage*. 1822.

Antecedentes históricos

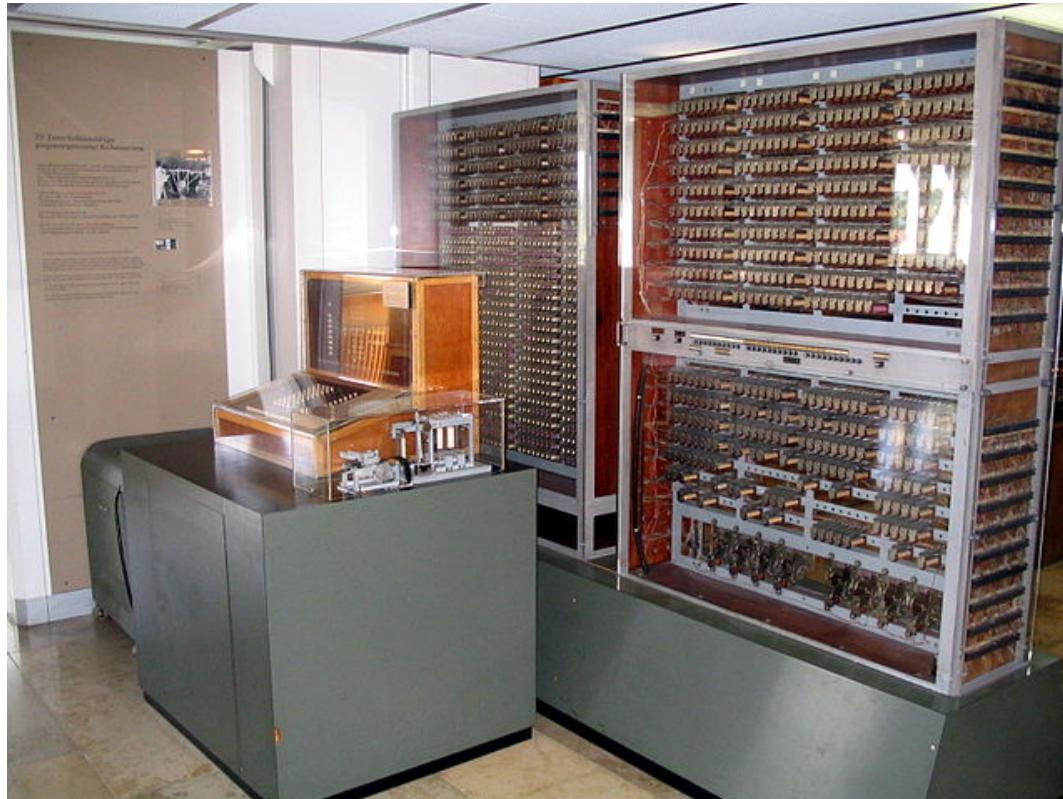
Computadoras electromecánicas



Z3, Konrad Zuse. 1941.

Antecedentes históricos

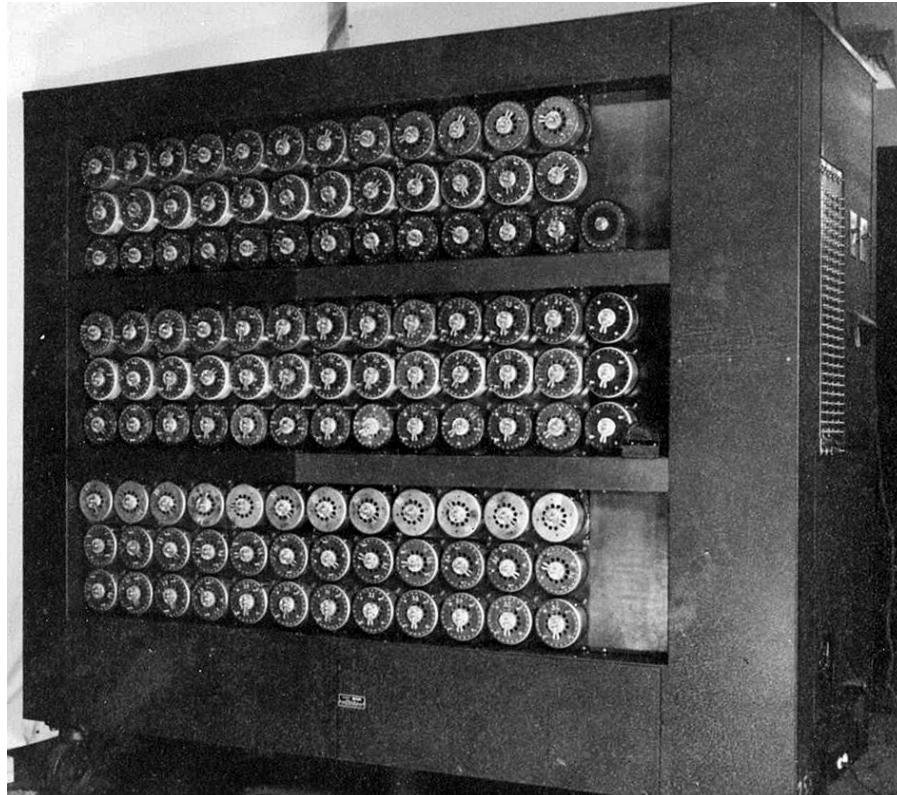
Computadoras electromecánicas



Z3, Konrad Zuse. 1941.

Antecedentes históricos

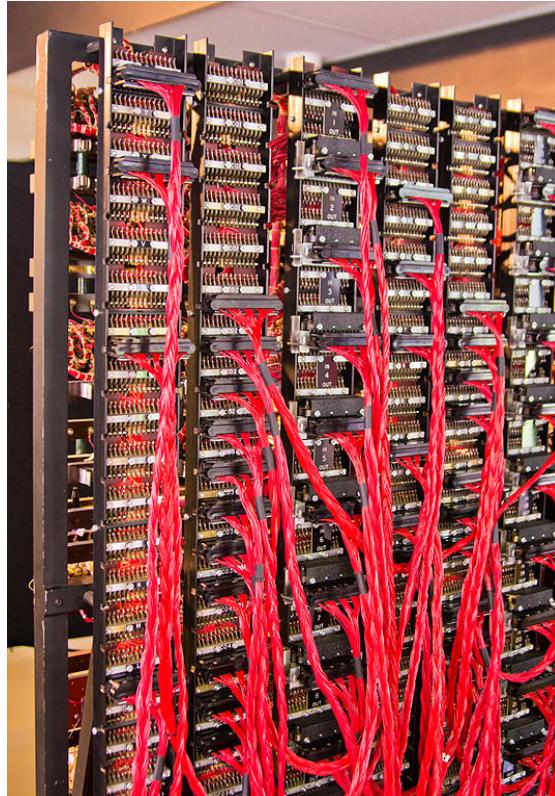
Computadoras electromecánicas



Bombe, Marian Rejewski y Alan Turing. 1941.

Antecedentes históricos

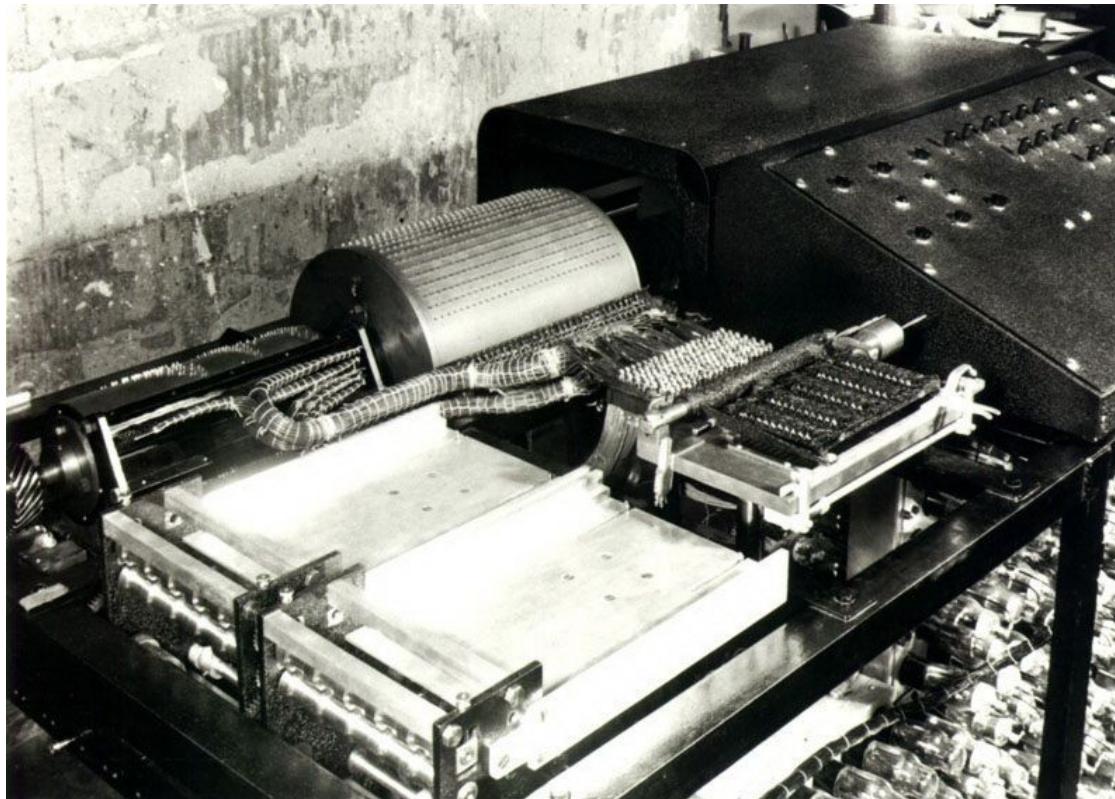
Computadoras electromecánicas



Bombe, Marian Rejewski y Alan Turing. 1941.

Antecedentes históricos

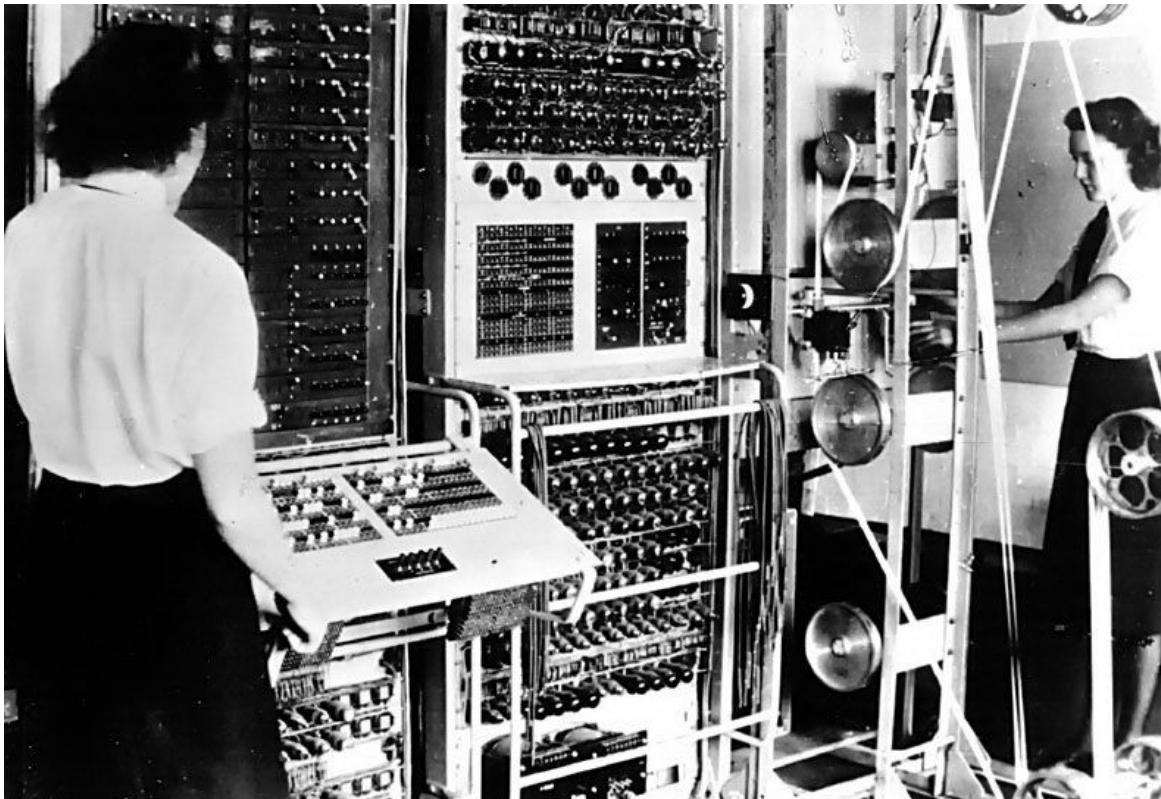
Computadoras electrónicas



ABC, *John Vincent Atanasoff y Clifford Berry*. 1942.

Antecedentes históricos

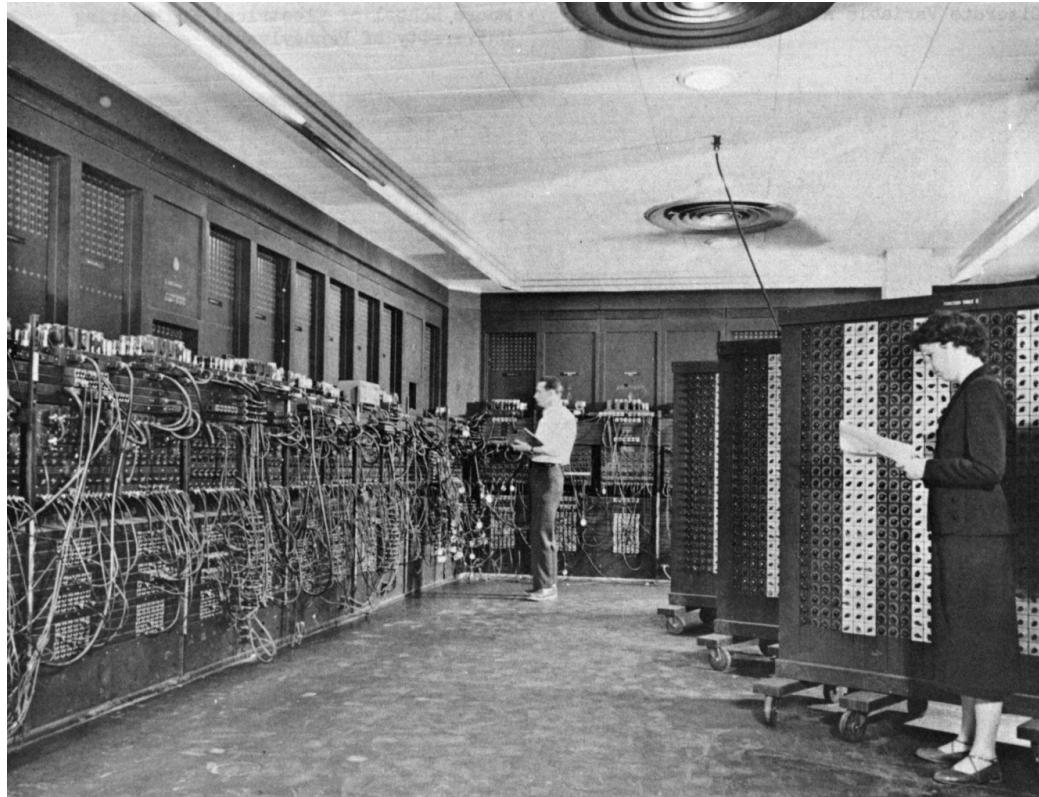
Computadoras electrónicas



Colossus, *Thomas Harold Flowers*. 1944.

Antecedentes históricos

Computadoras electrónicas



ENIAC, *Universidad de Pennsylvania y US Army*. 1944.

Antecedentes históricos

Computadoras electrónicas (Primer programa almacenado en memoria)



Manchester Baby, Universidad de Manchester. 1948.

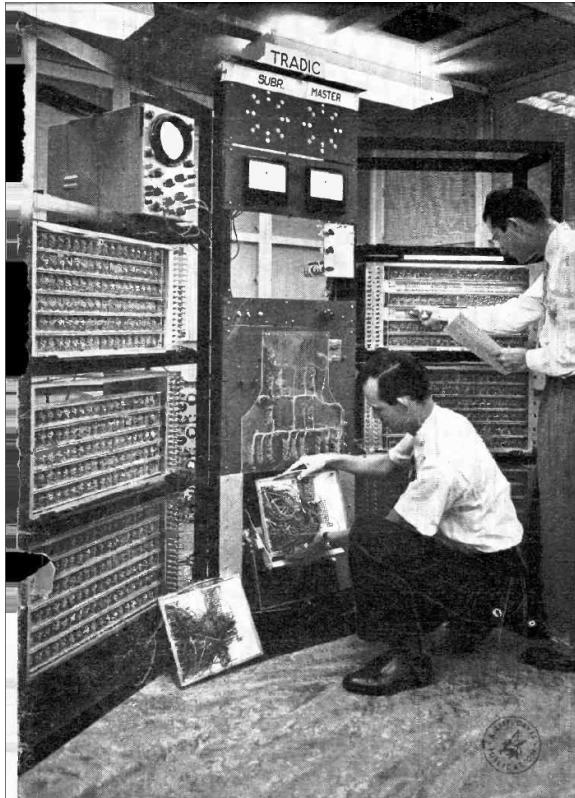
Antecedentes históricos

Computadoras electrónicas (Primera computadora con transistores)

Transistor Computer (prototype), *Universidad de Manchester*. 1953.

Antecedentes históricos

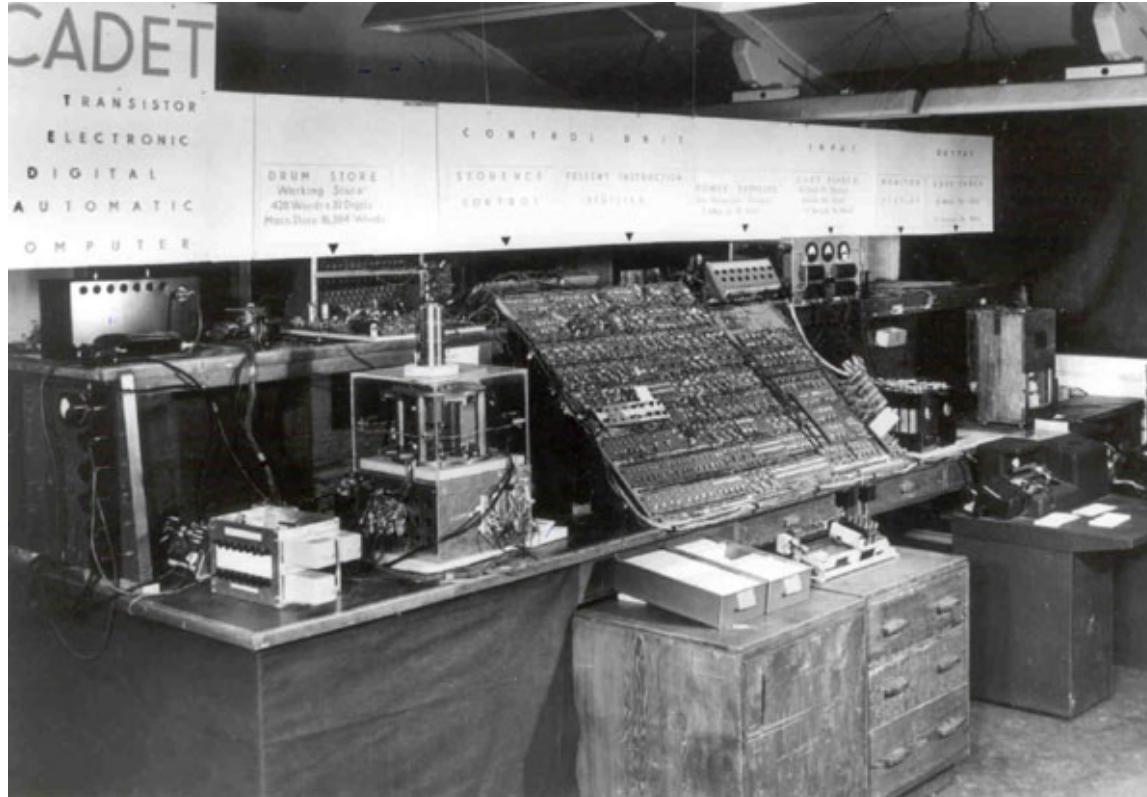
Computadoras electrónicas (Primera computadora con transistores en América)



TRADIC, *Bell Labs*. 1954.

Antecedentes históricos

Computadoras electrónicas (Primera computadora totalmente transistorizada)



Harwell CADET, *Harwell Laboratory*. 1955.

Antecedentes históricos

Antecedentes históricos

Hasta aquí los desarrollos eran principalmente llevados a cabo por universidades, laboratorios y entes gubernamentales de distintos países.

Antecedentes históricos

Hasta aquí los desarrollos eran principalmente llevados a cabo por universidades, laboratorios y entes gubernamentales de distintos países.

A partir de aquí comenzaron los desarrollos comerciales, que rápidamente llegaron a tener una arquitectura similar a las computadoras modernas.

Computadora moderna

Computadora moderna

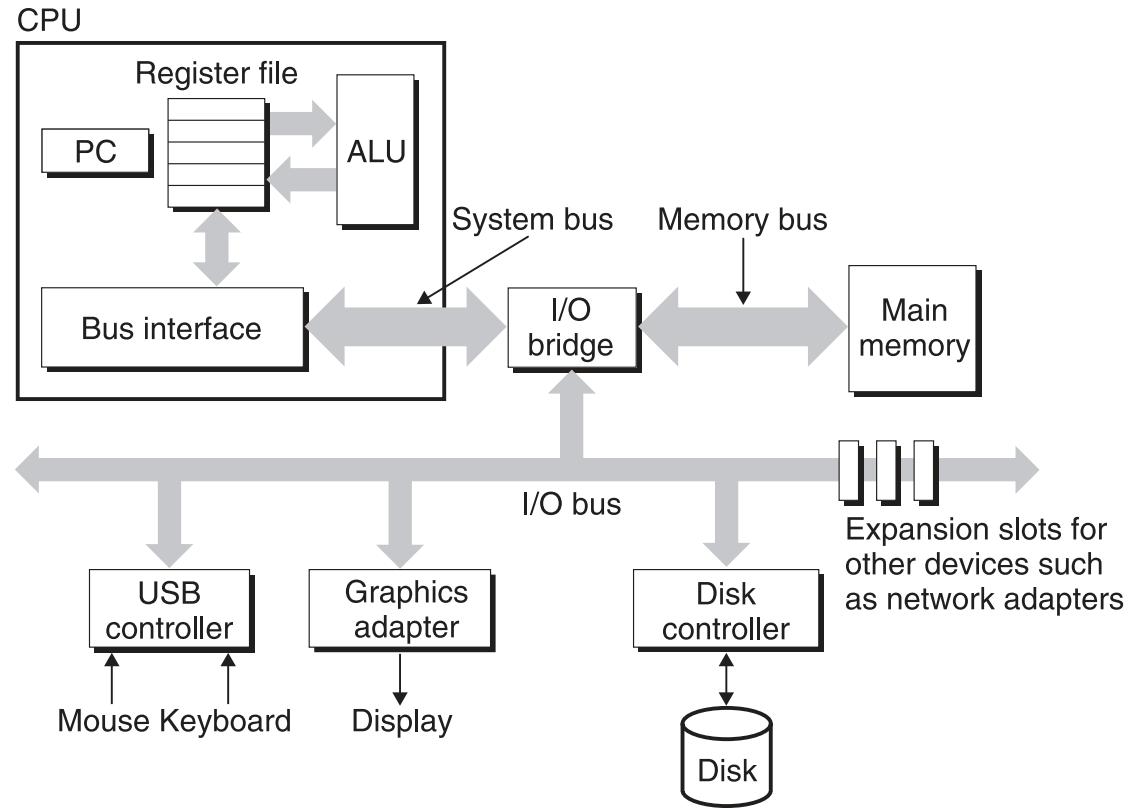
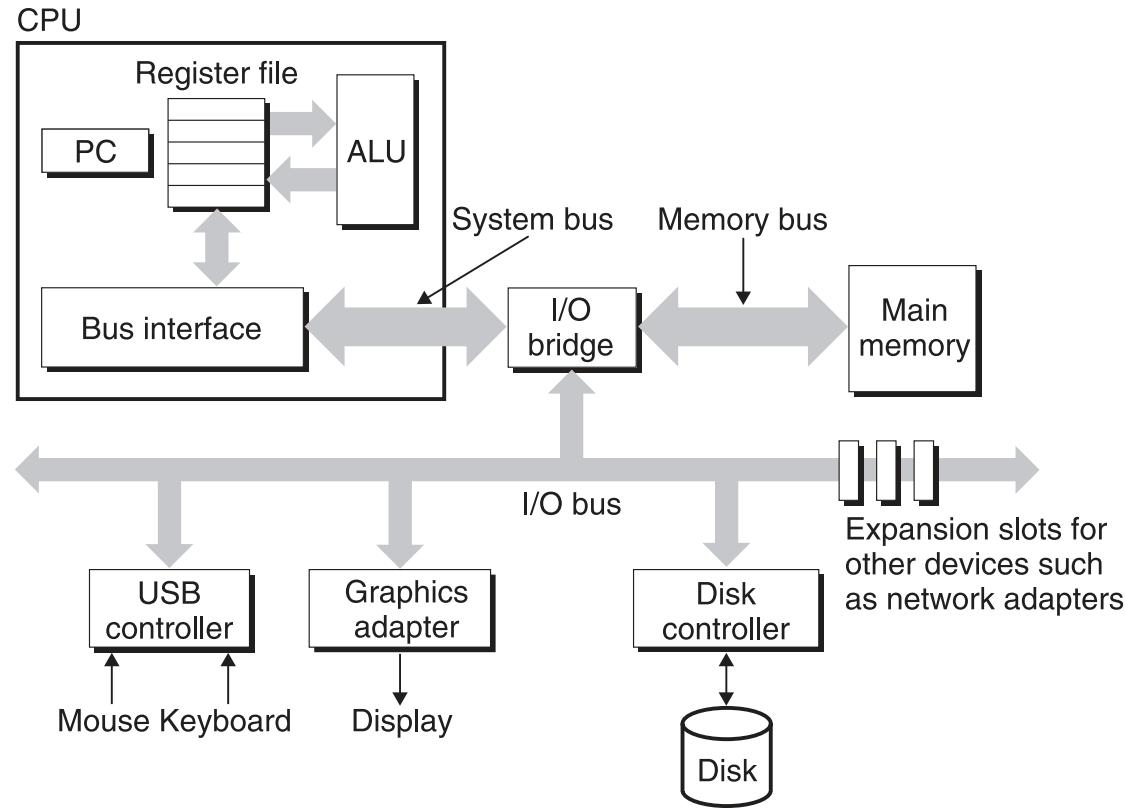


Ilustración tomada de *Computer Systems*, Bryant-O'Hallaron, Ed. 2015, Cap.1, pág.44

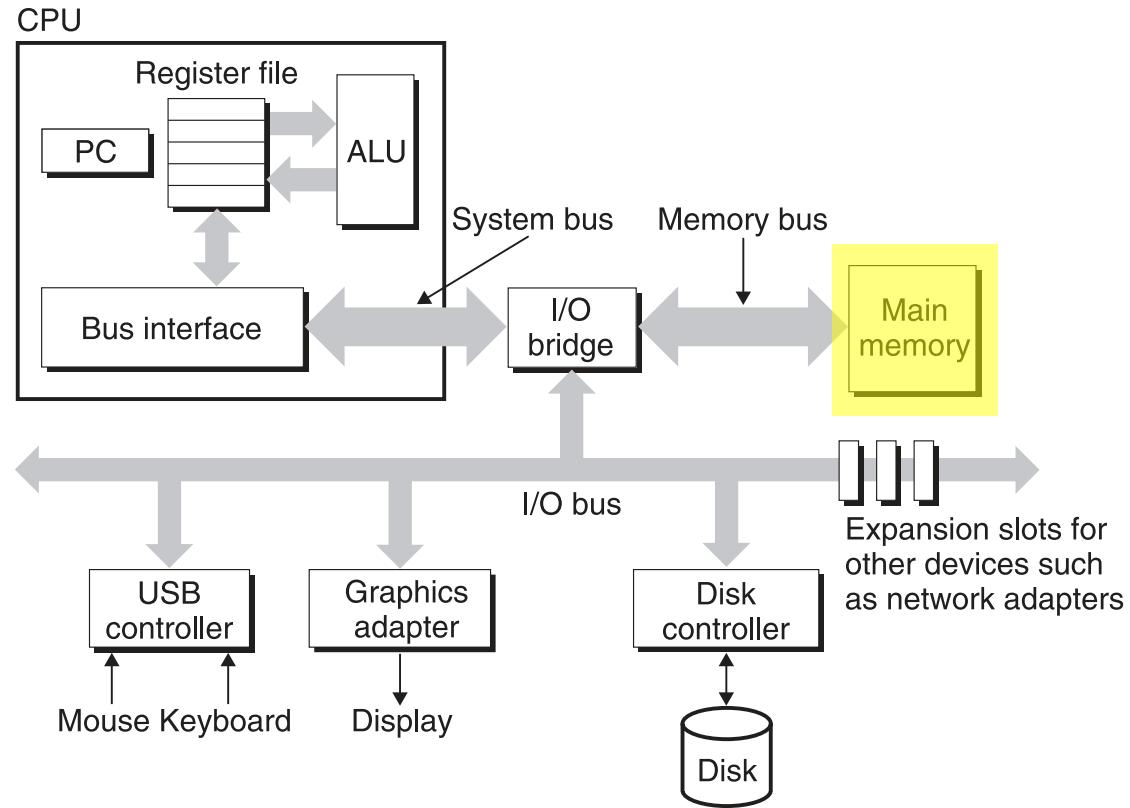
Computadora moderna

Memoria Principal

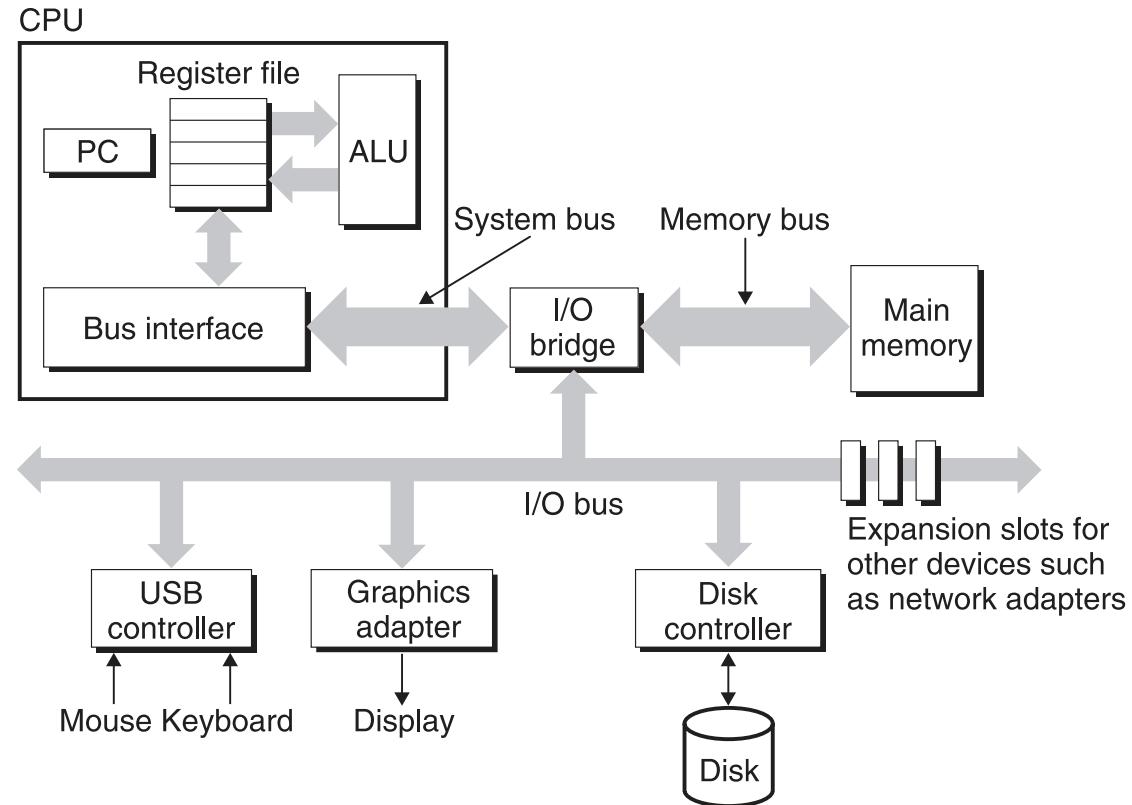


Computadora moderna

Memoria Principal

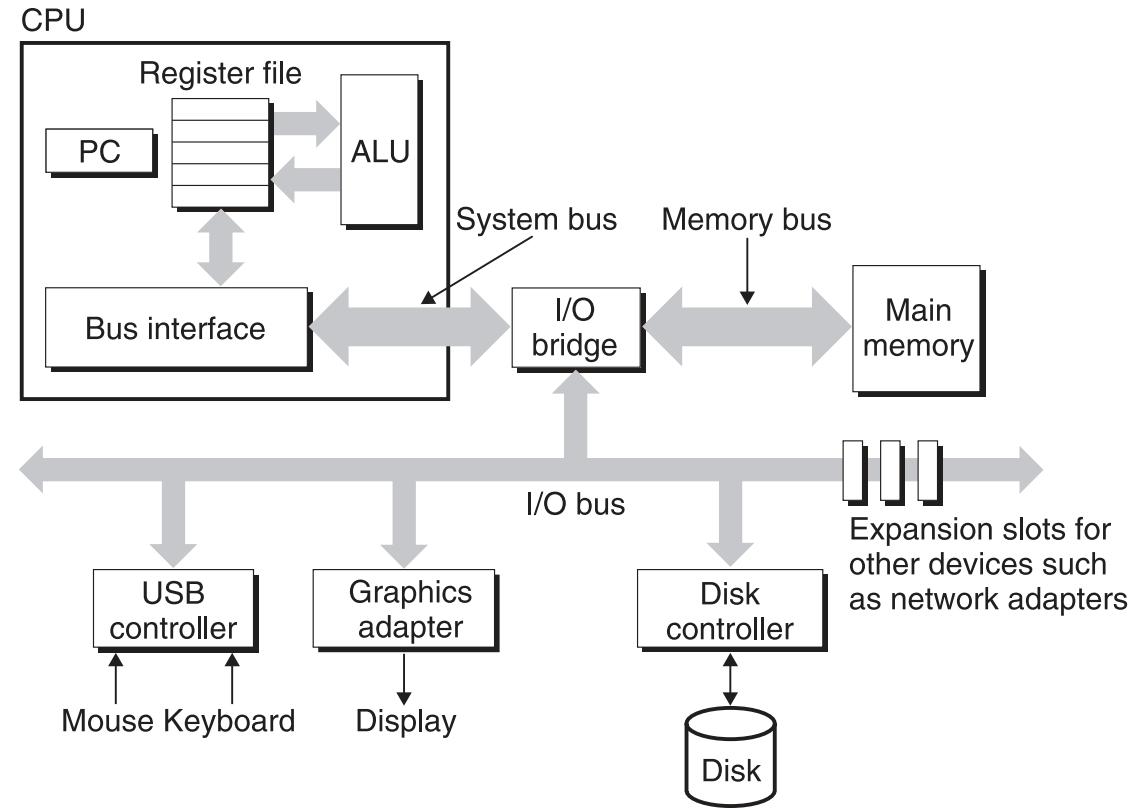


Computadora moderna



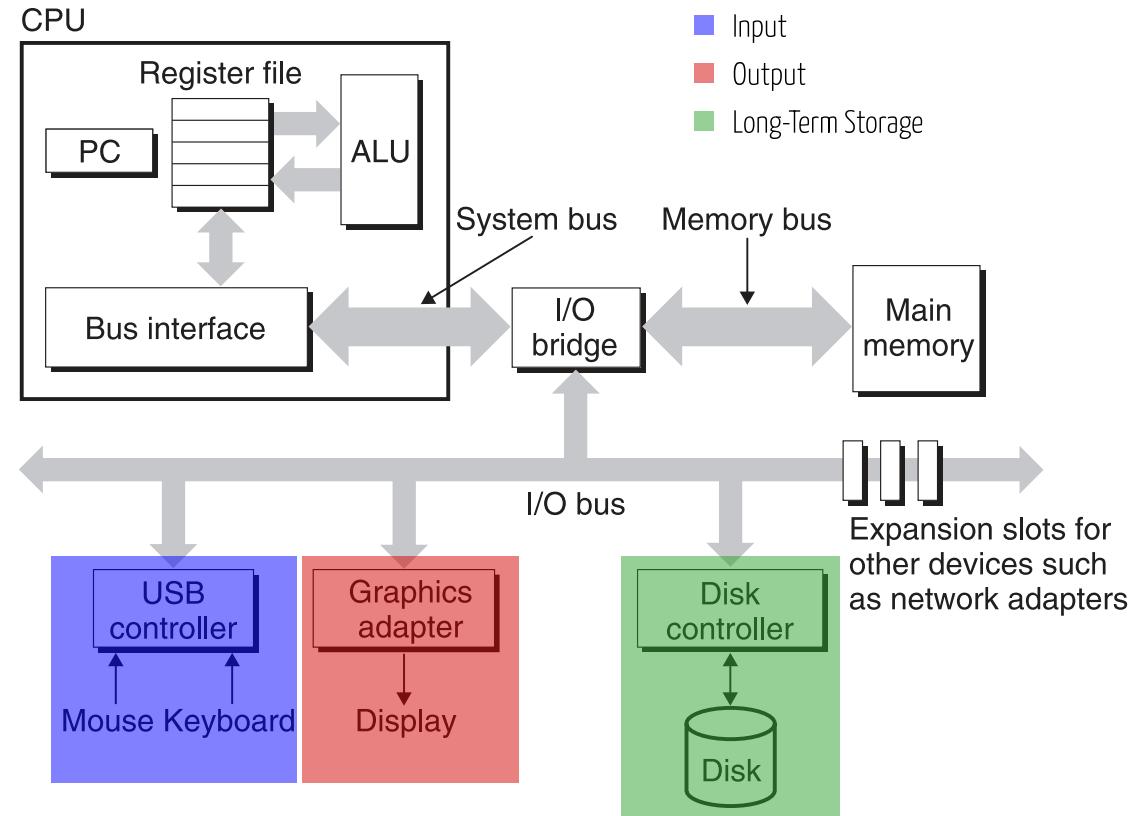
Computadora moderna

Dispositivos de Entrada/Salida



Computadora moderna

Dispositivos de Entrada/Salida



Computadora moderna

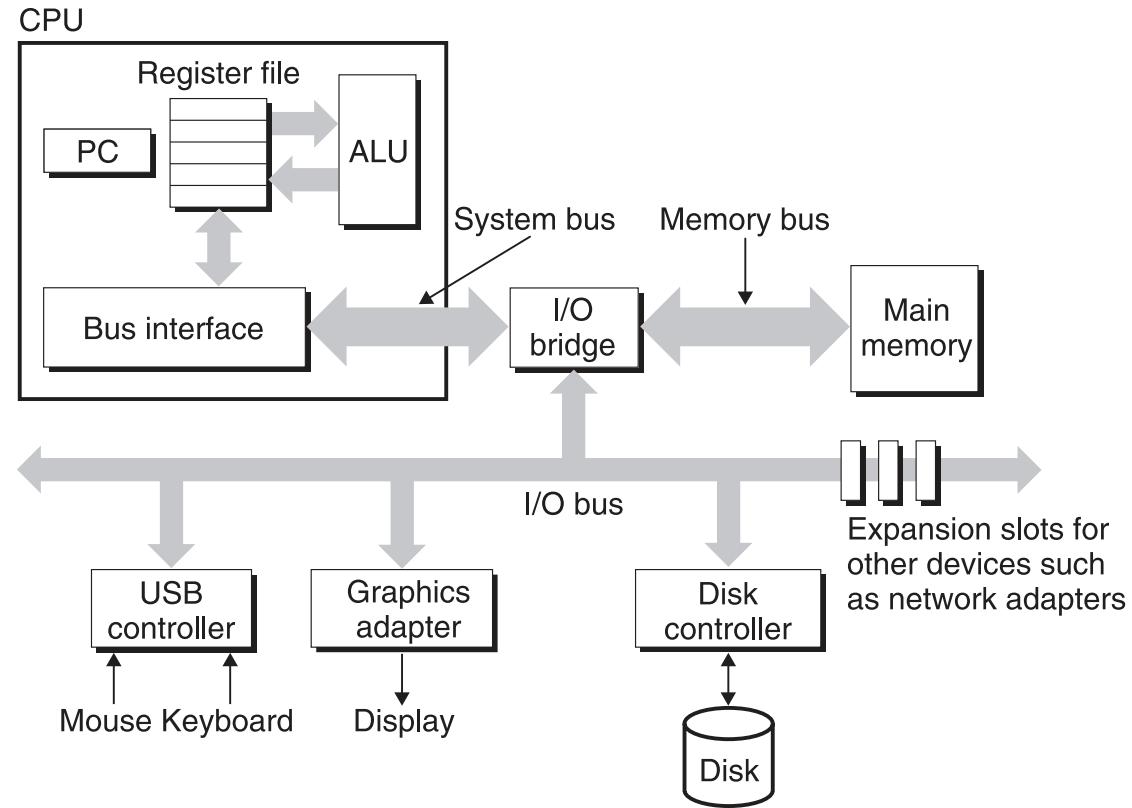
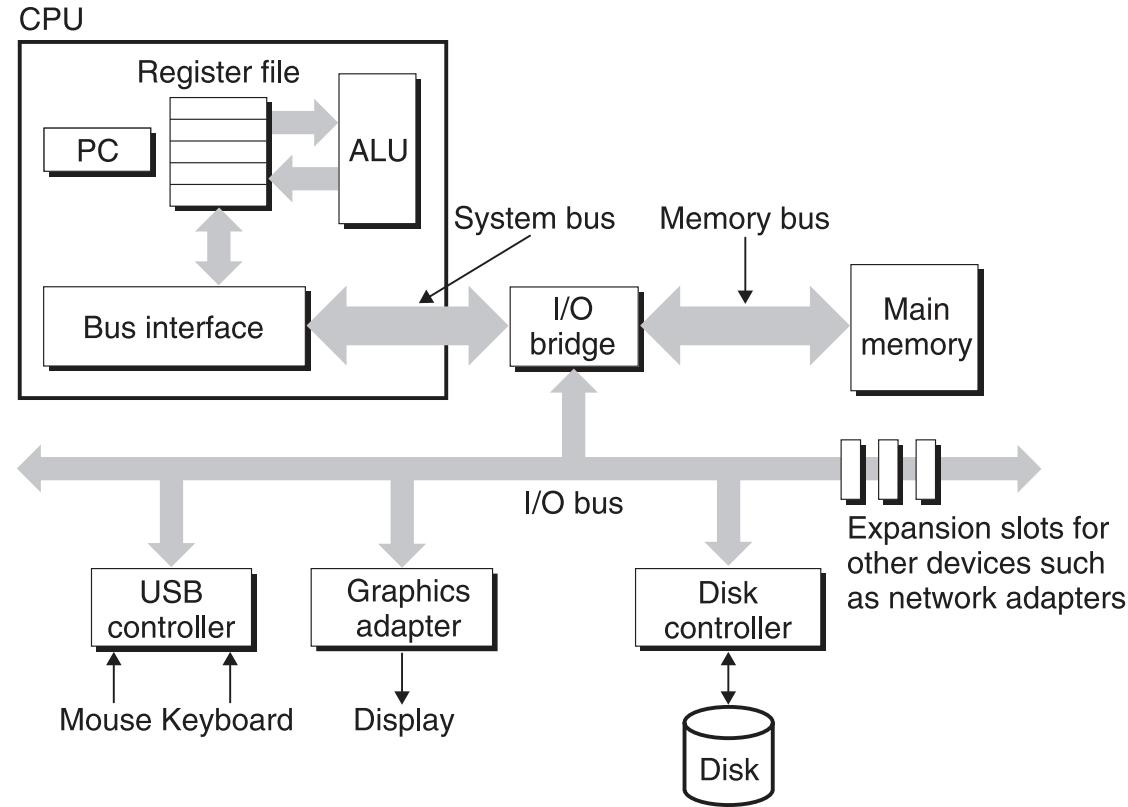


Ilustración tomada de *Computer Systems*, Bryant-O'Hallaron, Ed. 2015, Cap.1, pág.44

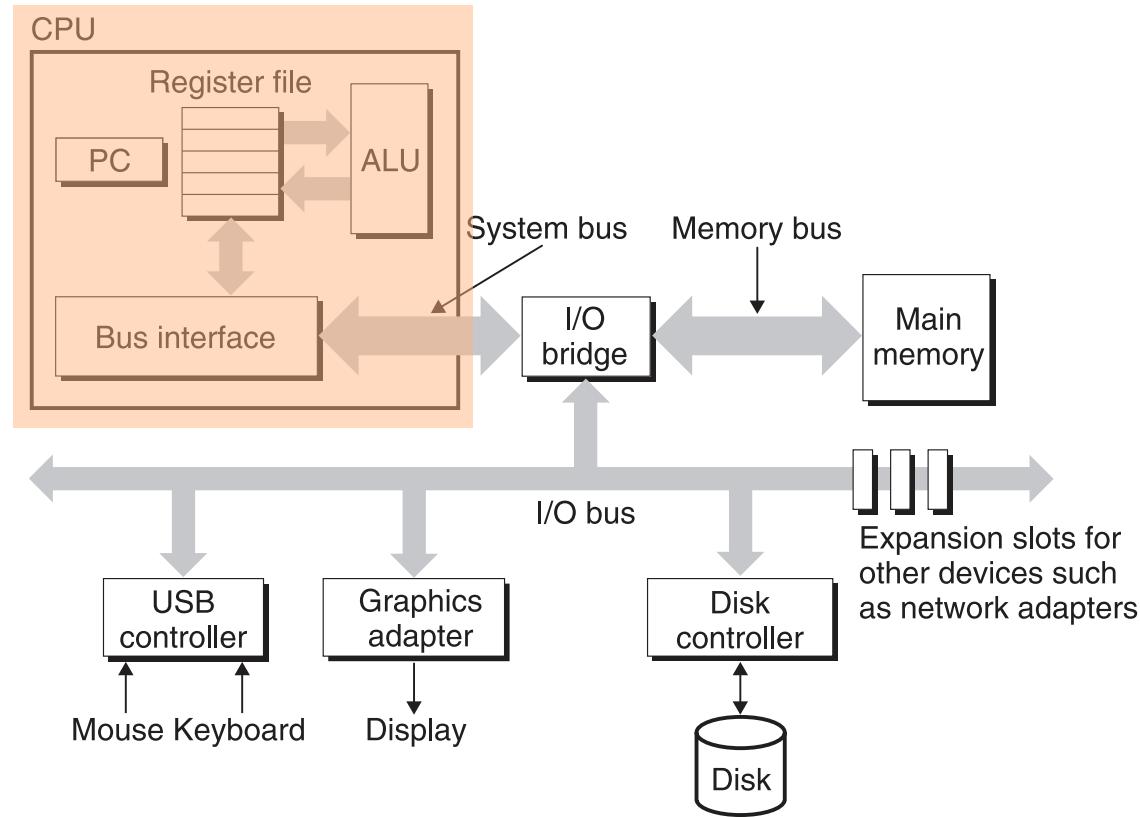
Computadora moderna

Unidad Central de Procesamiento

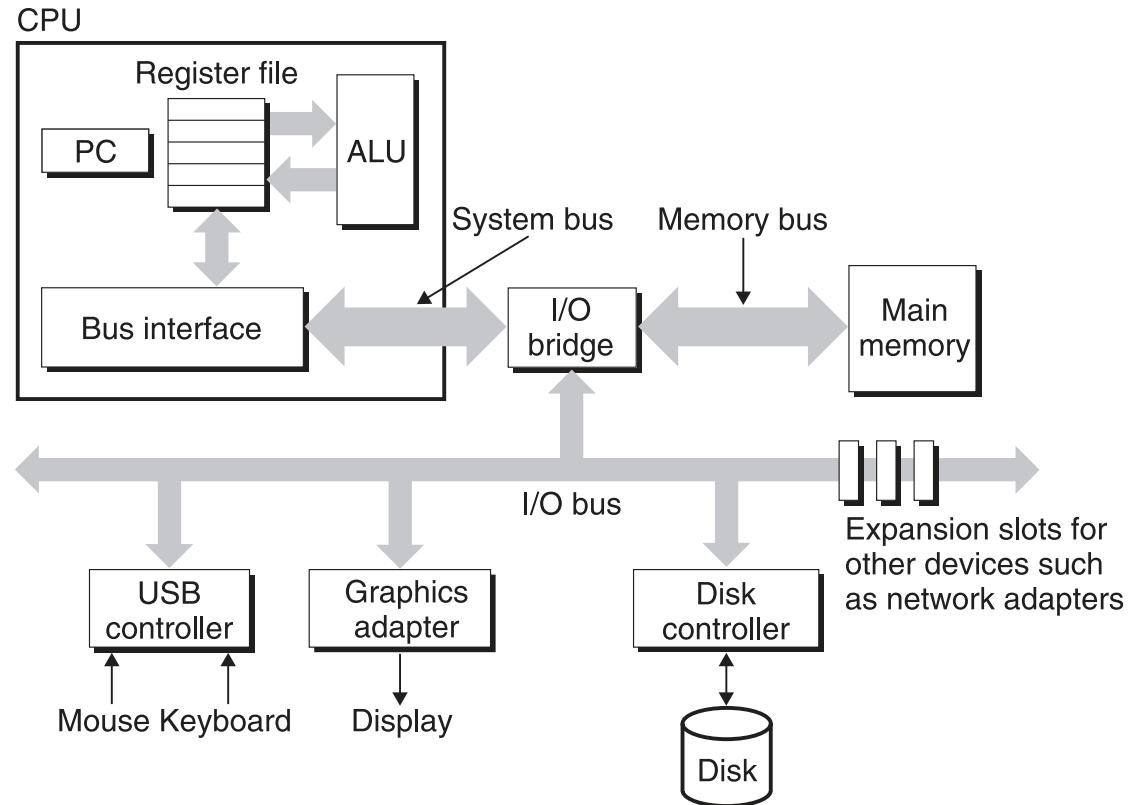


Computadora moderna

Unidad Central de Procesamiento

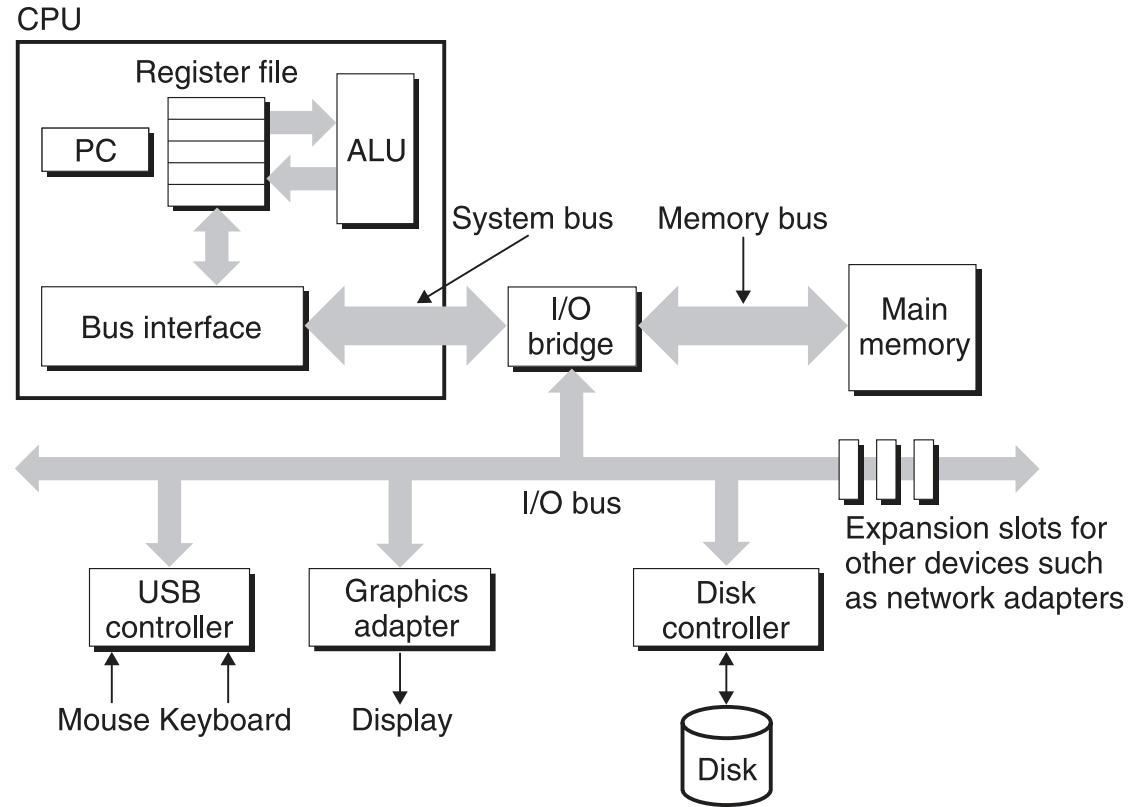


Computadora moderna



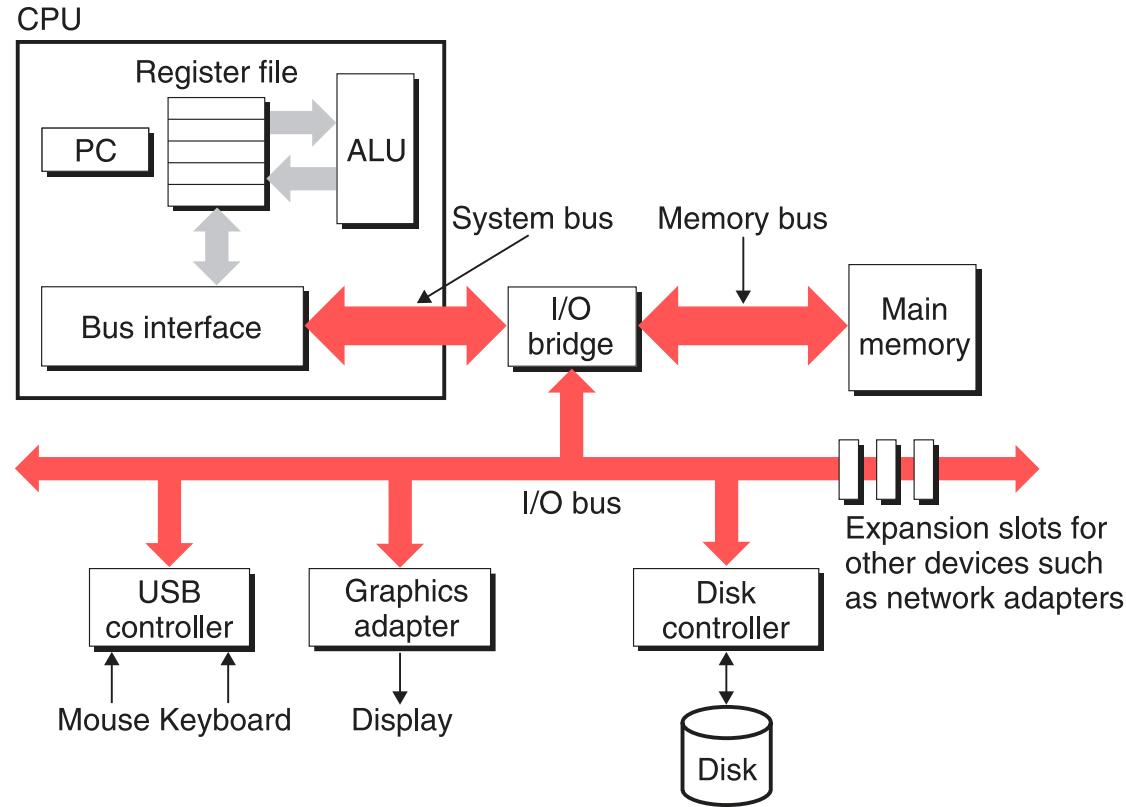
Computadora moderna

Buses



Computadora moderna

Buses



Unidades fundamentales

Unidades fundamentales

El *bit* es el elemento más pequeño que reconoce la computadora. Tiene solo dos valores posibles, 0 y 1.

Unidades fundamentales

El *bit* es el elemento más pequeño que reconoce la computadora. Tiene solo dos valores posibles, 0 y 1.

El *byte* está formado por 8 bits. Todas las combinaciones posibles de 8 bits prendidos y apagados son 256.

Unidades fundamentales

El *bit* es el elemento más pequeño que reconoce la computadora. Tiene solo dos valores posibles, 0 y 1.

El *byte* está formado por 8 bits. Todas las combinaciones posibles de 8 bits prendidos y apagados son 256.

Una *palabra* (o *word* en inglés) está formada por 2 o 4 bytes (esto depende del sistema).

Unidades fundamentales

Unidades fundamentales

bit
□

Unidades fundamentales

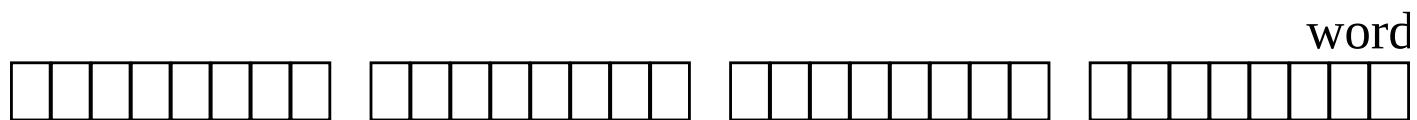
bit
□

byte
□□□□□□□□

Unidades fundamentales

bit
□

byte


word


Unidades fundamentales

bit
□

byte
□ □ □ □ □ □ □
7 6 5 4 3 2 1 0

word
□ □ □ □ □ □ □ □ □ □

Unidades fundamentales

bit
□

byte
□□□□□□□□
7 6 5 4 3 2 1 0

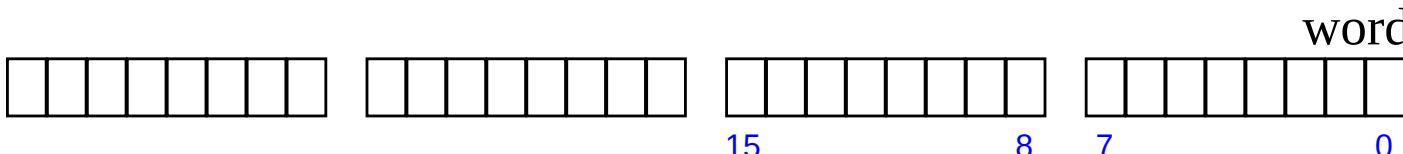
word
□□□□□□□□□□
7 0

Unidades fundamentales

bit
□

byte

7 6 5 4 3 2 1 0

word

15 8 7 0

Unidades fundamentales

bit
□

byte
□□□□□□□□
7 6 5 4 3 2 1 0

word
□□□□□□□□
23 16 15 8 7 0

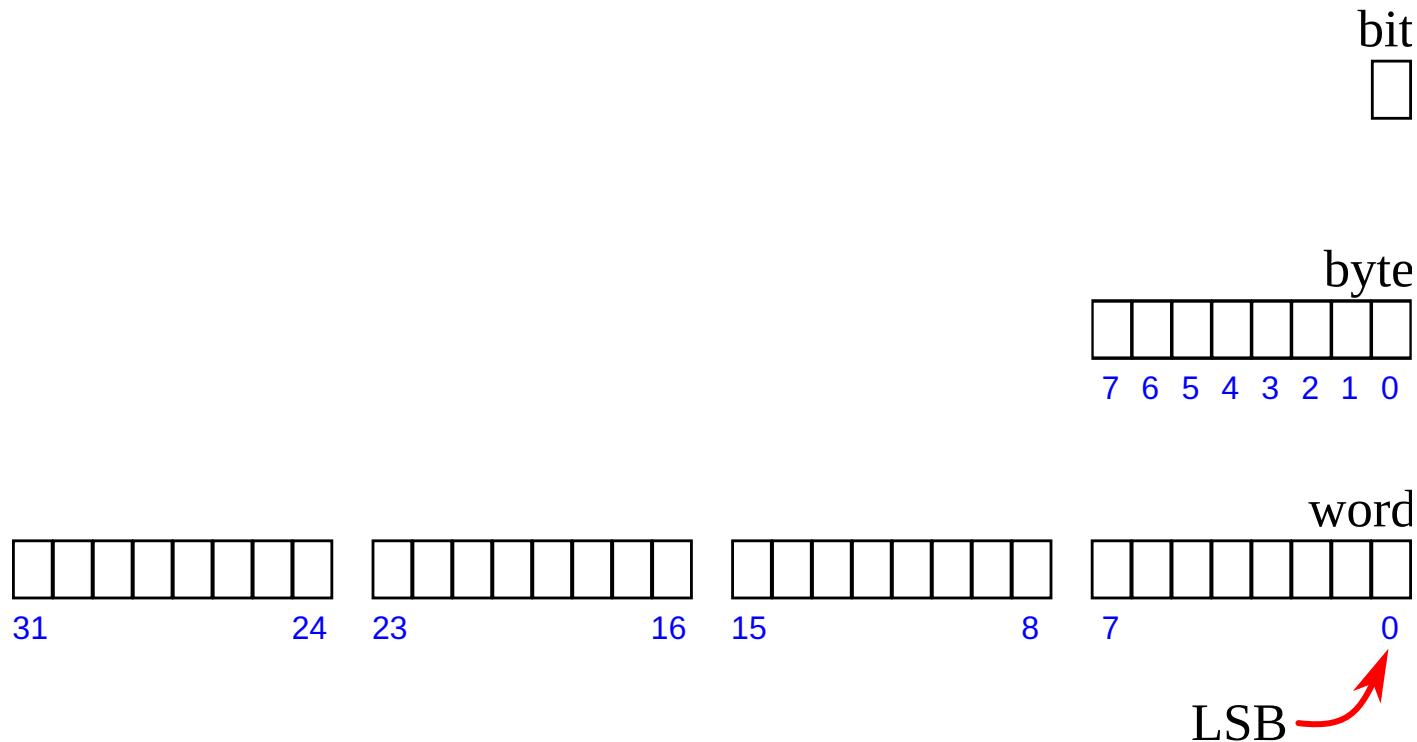
Unidades fundamentales

bit
□

byte
□□□□□□□□
7 6 5 4 3 2 1 0

word
□□□□□□□□
31 24 23 16 15 8 7 0

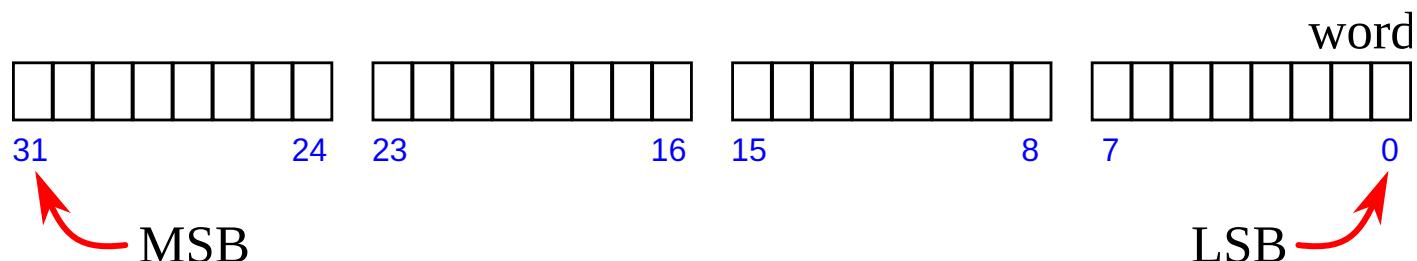
Unidades fundamentales



Unidades fundamentales

bit
□

byte
□□□□□□□□
7 6 5 4 3 2 1 0



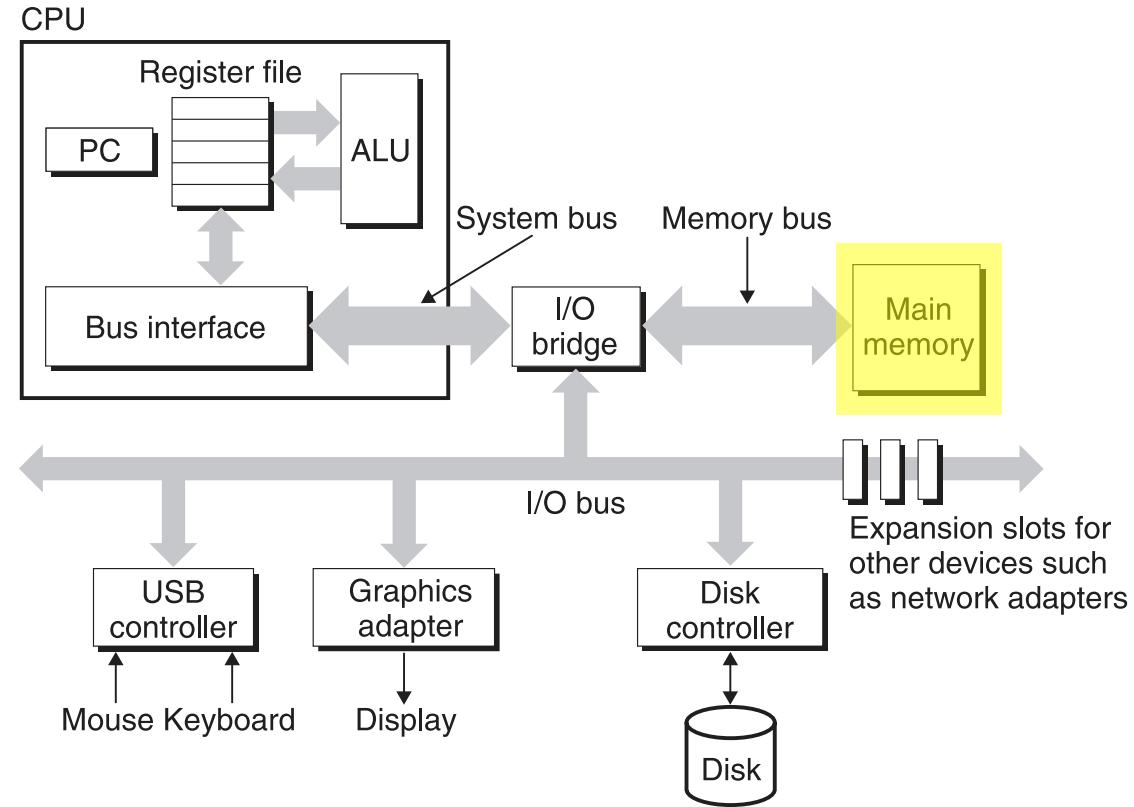
Múltiplos

Múltiplos

Unidad	bytes	Aproximadamente
1 kilobyte (kB)	1024 bytes	1000 bytes
1 megabyte (MB)	1024 kilobytes	$10^6 = 1000000$ bytes
1 gigabyte (GB)	1024 megabytes	$10^9 = 1000000000$ bytes
1 terabyte (TB)	1024 gigabytes	$10^{12} = 1000000000000$ bytes
1 petabyte (PB)	1024 terabytes	$10^{15} = 1000000000000000$ bytes
1 exabyte (EB)	1024 petabytes	$10^{18} = 1000000000000000000$ bytes
1 zetabyte (ZB)	1024 exabytes	$10^{21} = 1000000000000000000000$ bytes

Memoria Principal

Memoria Principal



Memoria Principal

Memoria Principal

Dispositivo que retiene o almacena datos informáticos durante algún periodo de tiempo.

Memoria Principal

Dispositivo que retiene o almacena datos informáticos durante algún periodo de tiempo.

Retiene datos provenientes de los dispositivos de entrada que serán utilizados por programas.

Memoria Principal

Dispositivo que retiene o almacena datos informáticos durante algún periodo de tiempo.

Retiene datos provenientes de los dispositivos de entrada que serán utilizados por programas.

Retiene datos procesados hasta que puedan ser puestos en los dispositivos de salida.

Memoria Principal

Dispositivo que retiene o almacena datos informáticos durante algún periodo de tiempo.

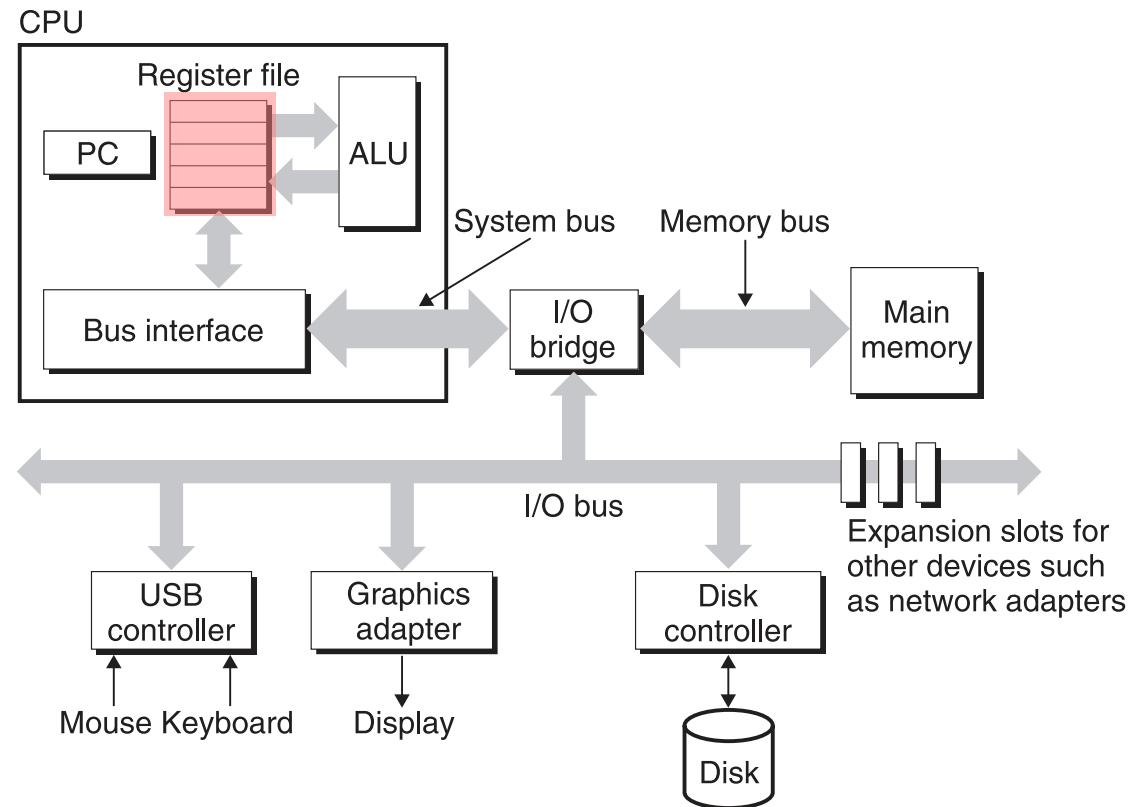
Retiene datos provenientes de los dispositivos de entrada que serán utilizados por programas.

Retiene datos procesados hasta que puedan ser puestos en los dispositivos de salida.

Incluso almacena los programas que están en ejecución.

Registros

Registros



Registros

Registros

Son pequeños espacios de almacenamiento del tamaño de una *palabra*

Registros

Son pequeños espacios de almacenamiento del tamaño de una *palabra*

Cada registro tiene su propio nombre.

Registros

Son pequeños espacios de almacenamiento del tamaño de una *palabra*

Cada registro tiene su propio nombre.

La colección de todos los registros se conoce como *Archivo de registros*

Registros

Son pequeños espacios de almacenamiento del tamaño de una *palabra*

Cada registro tiene su propio nombre.

La colección de todos los registros se conoce como *Archivo de registros*

Dependiendo de la arquitectura, cuantos registros tendrá el CPU.

Registros Vs. Memoria

Registros Vs. Memoria

Los registros están dentro del *chip* del procesador.

Registros Vs. Memoria

Los registros están dentro del *chip* del procesador.

La memoria principal está en peines conectada al procesador mediante *buses*.

Registros Vs. Memoria

Los registros están dentro del *chip* del procesador.

La memoria principal está en peines conectada al procesador mediante *buses*.

El acceso a memoria principal es aproximadamente 100 veces más lenta.

Registros Vs. Memoria

Los registros están dentro del *chip* del procesador.

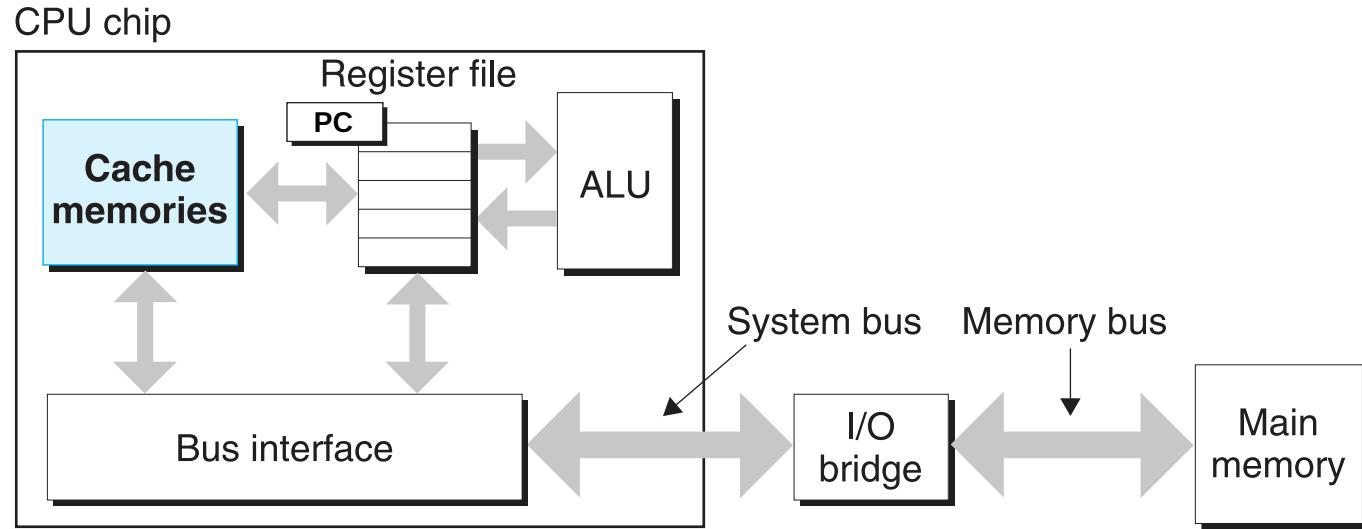
La memoria principal está en peines conectada al procesador mediante *buses*.

El acceso a memoria principal es aproximadamente 100 veces más lenta.

El tamaño del archivo de registros es muy pequeño, típicamente del orden del kB.

Memoria caché (en inglés se dice *cash*)

Memoria caché (en inglés se dice *cash*)



Memoria caché

Memoria caché

Es un espacio de almacenamiento dentro del *chip* donde está el procesador central.

Memoria caché

Es un espacio de almacenamiento dentro del *chip* donde está el procesador central.

Es más lento de accesar que los registros, pero mucho más rápido que la memoria RAM.

Memoria caché

Es un espacio de almacenamiento dentro del *chip* donde está el procesador central.

Es más lento de accesar que los registros, pero mucho más rápido que la memoria RAM.

La idea es que en la memoria *caché* se almacenen datos que el procesador *podría* necesitar.

Memoria caché

Memoria caché

Las computadoras modernas tienen 3 niveles de caché: L1, L2 y L3

Memoria caché

Las computadoras modernas tienen 3 niveles de caché: L1, L2 y L3

La L1 tiene una capacidad del orden de los cientos de kB. (Ej. en mi laptop L1i 128kB, L1d 128kB)

Memoria caché

Las computadoras modernas tienen 3 niveles de caché: L1, L2 y L3

La L1 tiene una capacidad del orden de los cientos de kB. (Ej. en mi laptop L1i 128kB, L1d 128kB)

La L2 tiene una capacidad que puede ir del orden de los 100kB hasta los MB. (Ej. en mi laptop L2 1MB)

Memoria caché

Las computadoras modernas tienen 3 niveles de caché: L1, L2 y L3

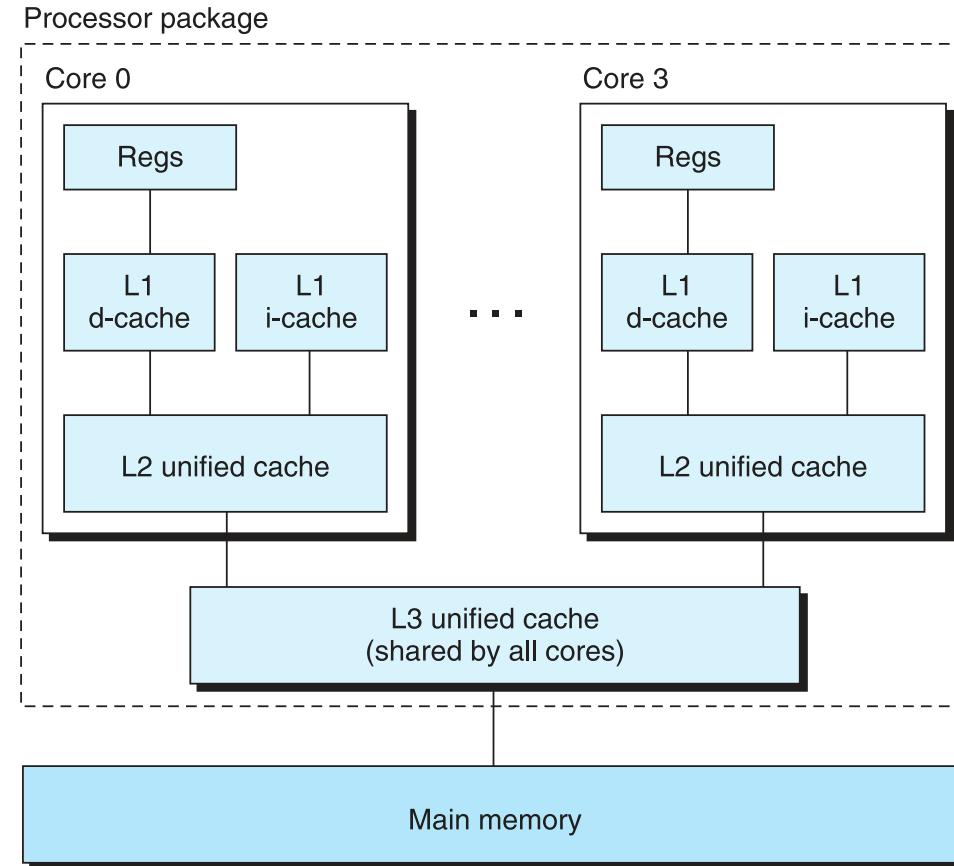
La L1 tiene una capacidad del orden de los cientos de kB. (Ej. en mi laptop L1i 128kB, L1d 128kB)

La L2 tiene una capacidad que puede ir del orden de los 100kB hasta los MB. (Ej. en mi laptop L2 1MB)

La caché L3 tiene una capacidad del orden de los MB. (Ej. en mi laptop L3 6MB)

Memoria caché

Memoria caché



Jerarquía de memoria.

Jerarquía de memoria.

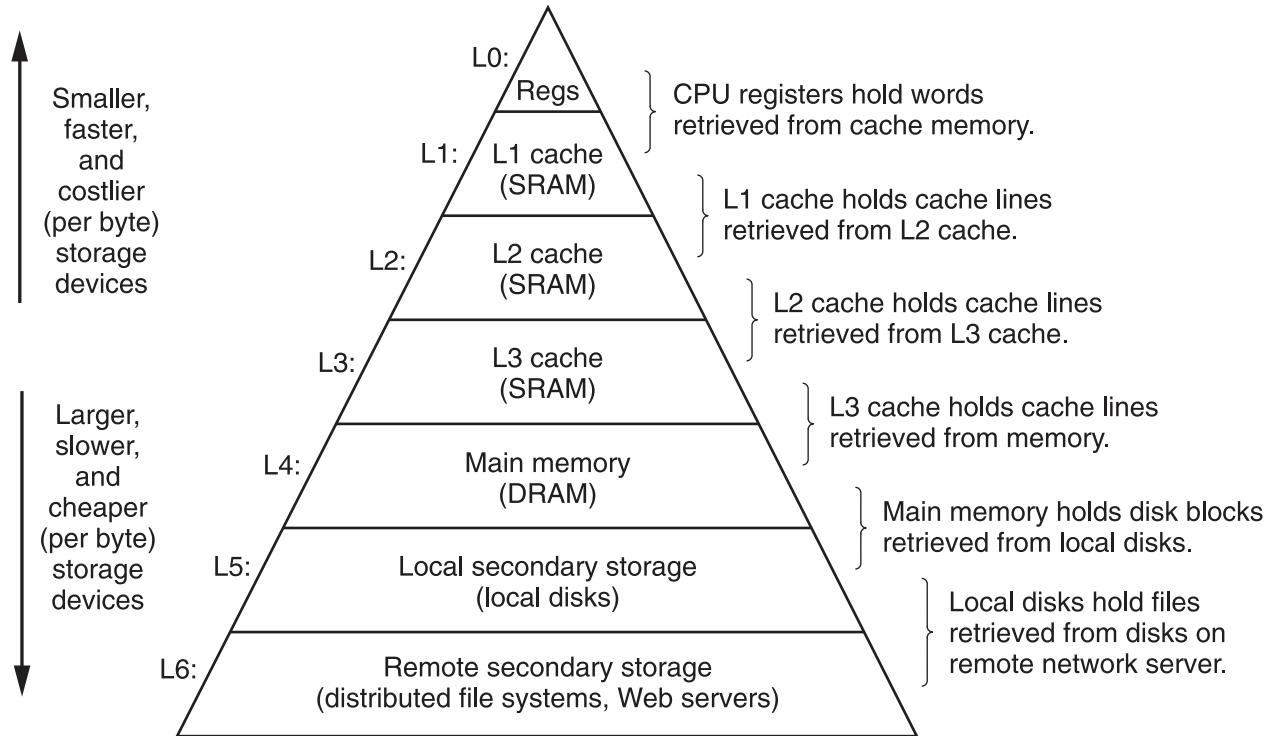
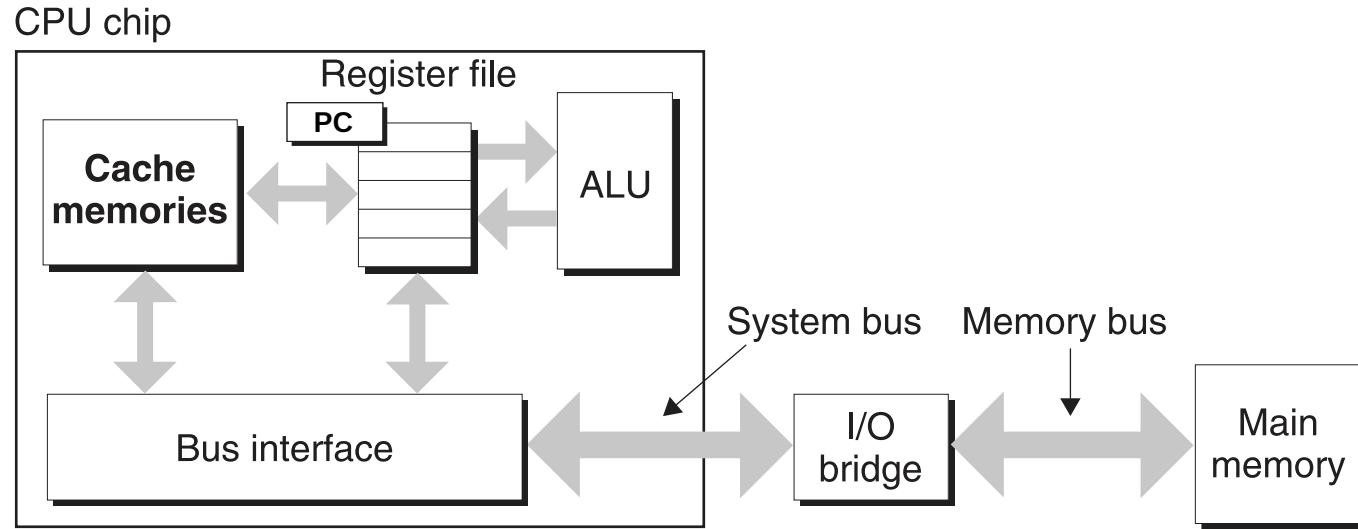


Figure 1.9 An example of a memory hierarchy.

Unidad Central de Procesamiento o CPU

Unidad Central de Procesamiento o CPU



Unidad Central de Procesamiento o CPU

Unidad Central de Procesamiento o CPU

Ejecuta las instrucciones almacenadas en la memoria.

Unidad Central de Procesamiento o CPU

Ejecuta las instrucciones almacenadas en la memoria.

Un registro llamado PC (Program Counter) apunta a la dirección donde está almacenada la instrucción que debe ejecutarse a continuación.

Unidad Central de Procesamiento o CPU

Ejecuta las instrucciones almacenadas en la memoria.

Un registro llamado PC (Program Counter) apunta a la dirección donde está almacenada la instrucción que debe ejecutarse a continuación.

El procesador lee la instrucción de la dirección de memoria *apuntada* por el PC...

Unidad Central de Procesamiento o CPU

Ejecuta las instrucciones almacenadas en la memoria.

Un registro llamado PC (Program Counter) apunta a la dirección donde está almacenada la instrucción que debe ejecutarse a continuación.

El procesador lee la instrucción de la dirección de memoria *apuntada* por el PC...

...interpreta la instrucción, realiza alguna operación sencilla indicada por la instrucción...

Unidad Central de Procesamiento o CPU

Ejecuta las instrucciones almacenadas en la memoria.

Un registro llamado PC (Program Counter) apunta a la dirección donde está almacenada la instrucción que debe ejecutarse a continuación.

El procesador lee la instrucción de la dirección de memoria *apuntada* por el PC...

...interpreta la instrucción, realiza alguna operación sencilla indicada por la instrucción...

... y actualiza el PC

Unidad Central de Procesamiento o CPU

Unidad Central de Procesamiento o CPU

Las operaciones que se pueden realizar son reducidas en número y simples en complejidad.

Unidad Central de Procesamiento o CPU

Las operaciones que se pueden realizar son reducidas en número y simples en complejidad.

Todas a cargo de la *Unidad Aritmética/Lógica* o por sus siglas en inglés (ALU).

Unidad Central de Procesamiento o CPU

Las operaciones que se pueden realizar son reducidas en número y simples en complejidad.

Todas a cargo de la *Unidad Aritmética/Lógica* o por sus siglas en inglés (ALU).

Alguna de las operaciones son: carga (*load*), almacenamiento (*store*), operación (*operate*), salto (*jump*)

Unidad Central de Procesamiento o CPU

Las operaciones que se pueden realizar son reducidas en número y simples en complejidad.

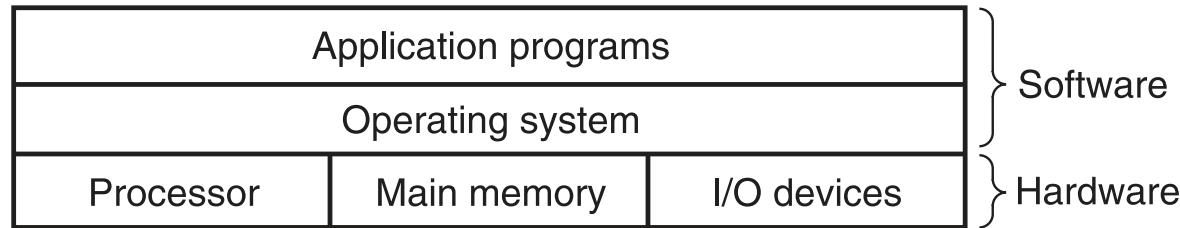
Todas a cargo de la *Unidad Aritmética/Lógica* o por sus siglas en inglés (ALU).

Alguna de las operaciones son: carga (*load*), almacenamiento (*store*), operación (*operate*), salto (*jump*)

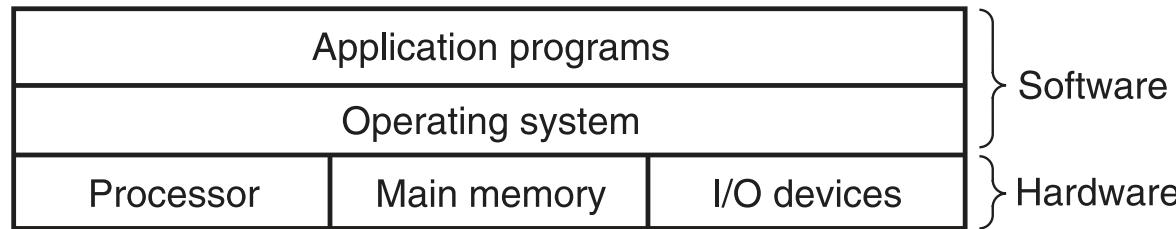
Todas las operaciones van de la memoria a los registros, la ALU y de vuelta, todo el tiempo.

Sistemas Operativos

Sistemas Operativos

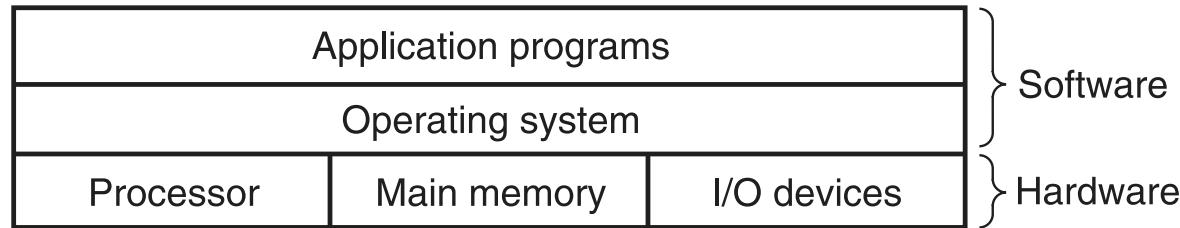


Sistemas Operativos

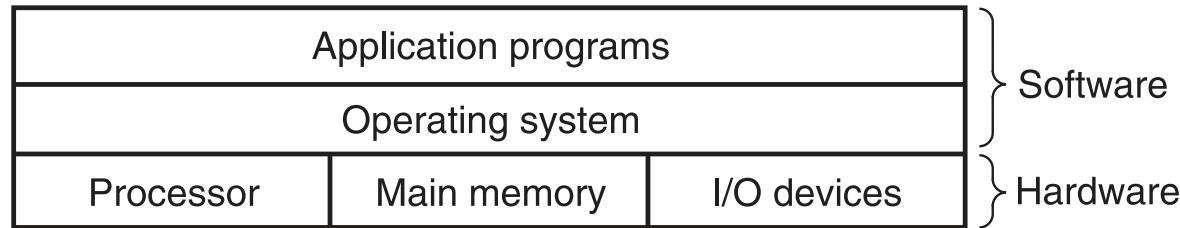


Los programas no reciben directamente la información ingresada desde el teclado ni de la memoria, ni escriben la memoria o imprimen en pantalla...

Sistemas Operativos



Sistemas Operativos



...por el contrario, utilizan *servicios* provistos por el *Sistema Operativo*

Sistemas Operativos

Sistemas Operativos

El SO tiene dos objetivos principales:

Sistemas Operativos

El SO tiene dos objetivos principales:

- Proteger el *hardware* de uso indebido por parte de las aplicaciones

Sistemas Operativos

El SO tiene dos objetivos principales:

- Proteger el *hardware* de uso indebido por parte de las aplicaciones
- Proveer un mecanismo unificado para manipular diferentes tipos de *hardware*

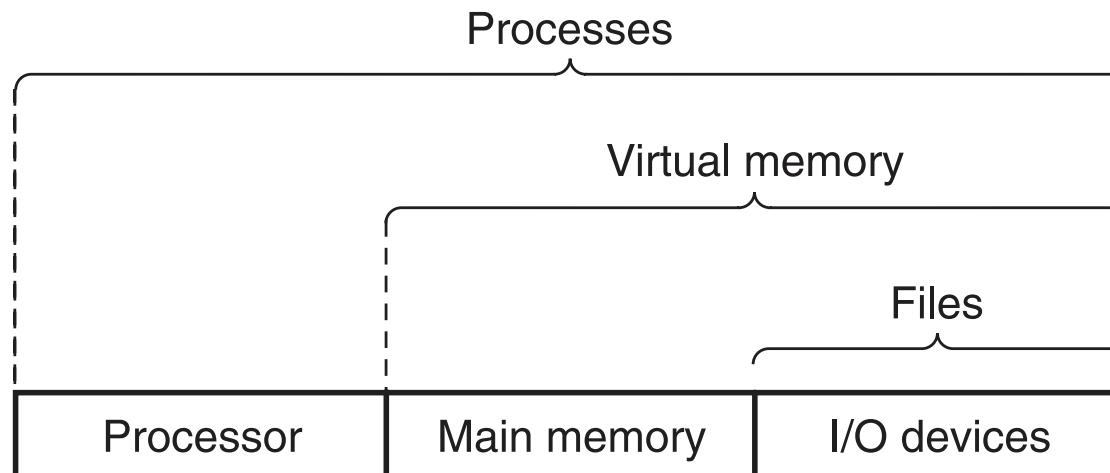
Sistemas Operativos

Sistemas Operativos

Para lograr estos objetivos el SO utiliza tres abstracciones:
procesos, memoria virtual y archivos

Sistemas Operativos

Para lograr estos objetivos el SO utiliza tres abstracciones:
procesos, memoria virtual y archivos



Sistemas Operativos

Sistemas Operativos

Un **proceso** es la abstracción que hace el sistema operativo de un programa corriendo proveyendo la ilusión de que es el único

Sistemas Operativos

Un **proceso** es la abstracción que hace el sistema operativo de un programa corriendo proveyendo la ilusión de que es el único

Muchos procesos pueden correr *concurrentemente* en el mismo sistema...

Sistemas Operativos

Un **proceso** es la abstracción que hace el sistema operativo de un programa corriendo proveyendo la ilusión de que es el único

Muchos procesos pueden correr *concurrentemente* en el mismo sistema...

...entendiéndose como concurrentes a los procesos que pueden ser interrumpidos momentáneamente por otros.

Sistemas Operativos

Un **proceso** es la abstracción que hace el sistema operativo de un programa corriendo proveyendo la ilusión de que es el único

Muchos procesos pueden correr *concurrentemente* en el mismo sistema...

...entendiéndose como concurrentes a los procesos que pueden ser interrumpidos momentáneamente por otros.

Normalmente, hay más procesos que CPUs.

Sistemas Operativos

Sistemas Operativos

La **memoria virtual** es una abstracción que provee a cada proceso la ilusión que tiene uso exclusivo de la memoria principal, conocida como *espacio virtual de memoria*

Sistemas Operativos

La **memoria virtual** es una abstracción que provee a cada proceso la ilusión que tiene uso exclusivo de la memoria principal, conocida como *espacio virtual de memoria*

El espacio de memoria virtual tiene áreas bien definidas cada una con su propósito específico

Sistemas Operativos

Sistemas Operativos

Todos los dispositivos de entrada/salida son modelados como **archivos**.

Sistemas Operativos

Todos los dispositivos de entrada/salida son modelados como **archivos**.

Un archivo es simplemente una secuencia de bits.

Sistemas Operativos

Todos los dispositivos de entrada/salida son modelados como **archivos**.

Un archivo es simplemente una secuencia de bits.

Desde el teclado, disco, pantalla, cámaras hasta las redes son modeladas como archivos.

Sistemas Operativos

Todos los dispositivos de entrada/salida son modelados como **archivos**.

Un archivo es simplemente una secuencia de bits.

Desde el teclado, disco, pantalla, cámaras hasta las redes son modeladas como archivos.

Para realizar accesos a los distintos dispositivos se utilizan herramientas de acceso a archivos (apertura, lectura, escritura).

Sistemas Operativos

Sistemas Operativos

- Windows

Sistemas Operativos

- Windows
- macOS (y derivados)

Sistemas Operativos

- Windows
- macOS (y derivados)
- GNU/Linux (y derivados)

Sistemas Operativos

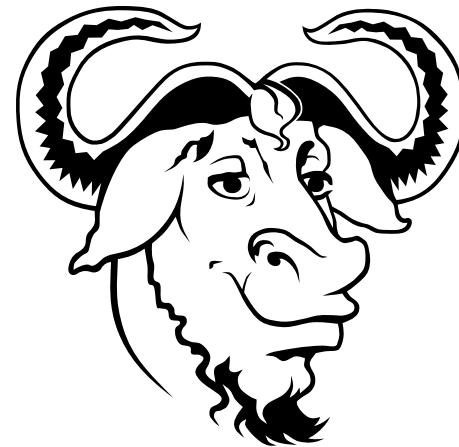
- Windows
- macOS (y derivados)
- GNU/Linux (y derivados)
- Android (pero es derivado de GNU/Linux)

Sistemas Operativos

- Windows
- macOS (y derivados)
- GNU/Linux (y derivados)
- Android (pero es derivado de GNU/Linux)
- otros

Sistemas Operativos

GNU/Linux



GNU's Not Unix

Sistemas Operativos

GNU/Linux

- No solo un sistema operativo, es también una colección de programas



GNU's Not Unix

Sistemas Operativos

GNU/Linux

- No solo un sistema operativo, es también una colección de programas
- gcc (GNU Compiler Collection)



GNU's Not Unix

Sistemas Operativos

GNU/Linux

- No solo un sistema operativo, es también una colección de programas
- gcc (GNU Compiler Collection)
- glibc



GNU's Not Unix

Sistemas Operativos

GNU/Linux

- No solo un sistema operativo, es también una colección de programas
- gcc (GNU Compiler Collection)
- glibc
- bash



GNU's Not Unix

Sistemas Operativos

GNU/Linux

- No solo un sistema operativo, es también una colección de programas
- gcc (GNU Compiler Collection)
- glibc
- bash
- GNU Core Utilities (cat, ls, rm, etc.)



GNU's Not Unix

Sistemas Operativos

GNU/Linux

- No solo un sistema operativo, es también una colección de programas
- gcc (GNU Compiler Collection)
- glibc
- bash
- GNU Core Utilities (cat, ls, rm, etc.)
- GNOME desktop environment



GNU's Not Unix

Sistemas Operativos

GNU/Linux

- No solo un sistema operativo, es también una colección de programas
- gcc (GNU Compiler Collection)
- glibc
- bash
- GNU Core Utilities (cat, ls, rm, etc.)
- GNOME desktop environment
- etc.



GNU's Not Unix

Sistemas Operativos

GNU/Linux

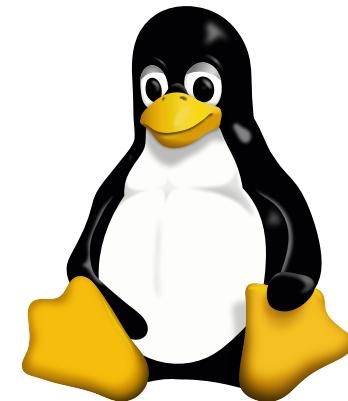


Linux

Sistemas Operativos

GNU/Linux

- Iniciado en 1991 por Linus Torvalds (~ 10000 lineas de código)

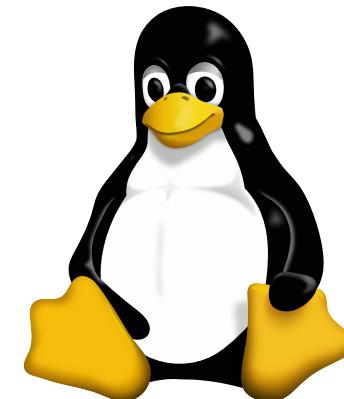


Linux

Sistemas Operativos

GNU/Linux

- Iniciado en 1991 por Linus Torvalds (~ 10000 lineas de código)
- Escrito completamente en C (y algo de ASM)

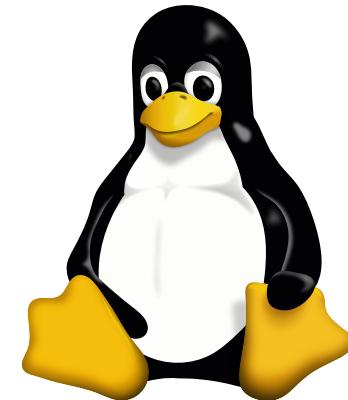


Linux

Sistemas Operativos

GNU/Linux

- Iniciado en 1991 por Linus Torvalds (~ 10000 líneas de código)
- Escrito completamente en C (y algo de ASM)
- Hoy tiene 25 millones de líneas de código

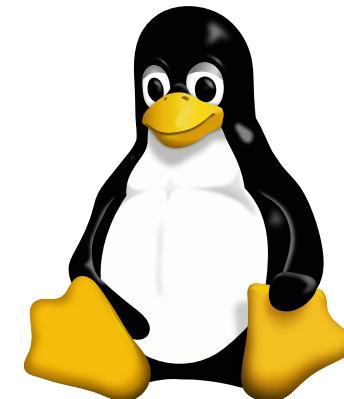


Linux

Sistemas Operativos

GNU/Linux

- Iniciado en 1991 por Linus Torvalds (~ 10000 líneas de código)
- Escrito completamente en C (y algo de ASM)
- Hoy tiene 25 millones de líneas de código
- y más de 19000 desarrolladores

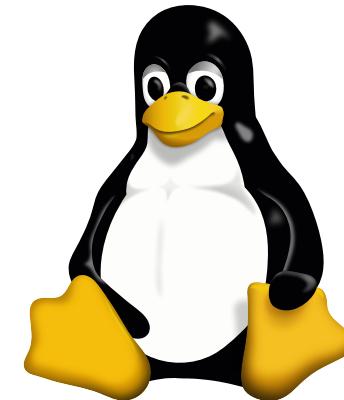


Linux

Sistemas Operativos

GNU/Linux

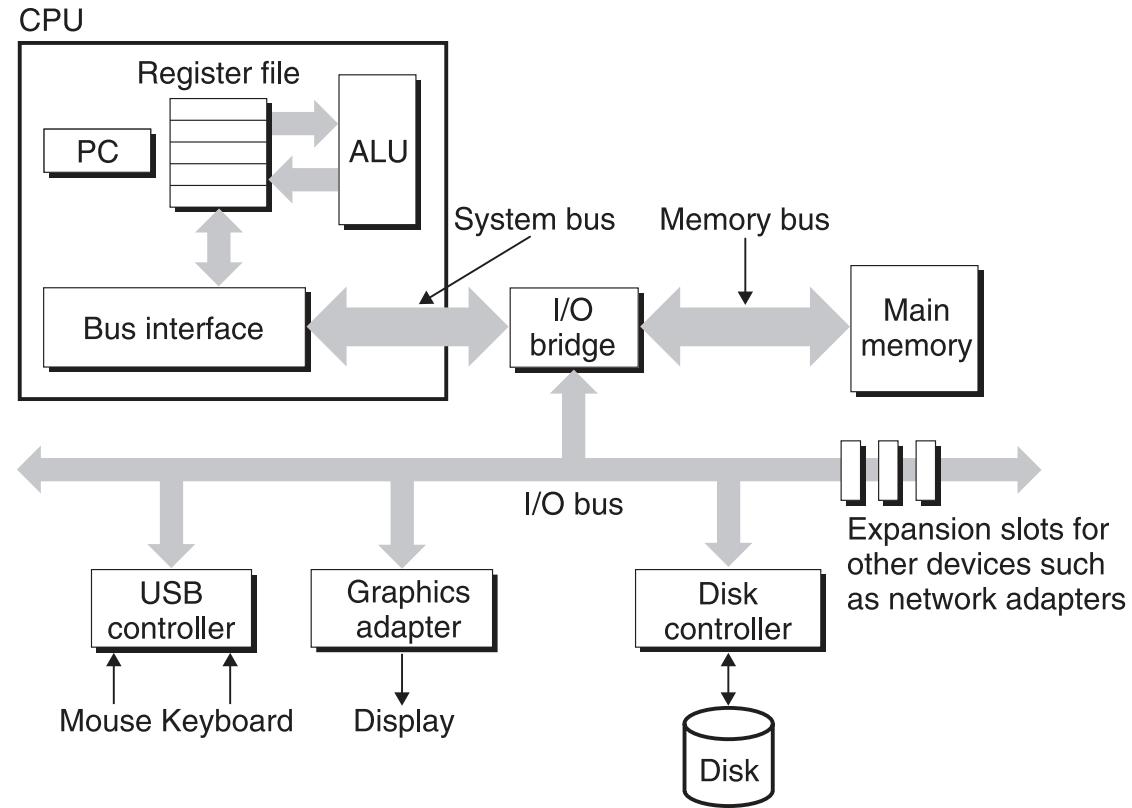
- Iniciado en 1991 por Linus Torvalds (~ 10000 lineas de código)
- Escrito completamente en C (y algo de ASM)
- Hoy tiene 25 millones de líneas de código
- y más de 19000 desarrolladores
- es el proyecto de software más grande de la historia



Linux

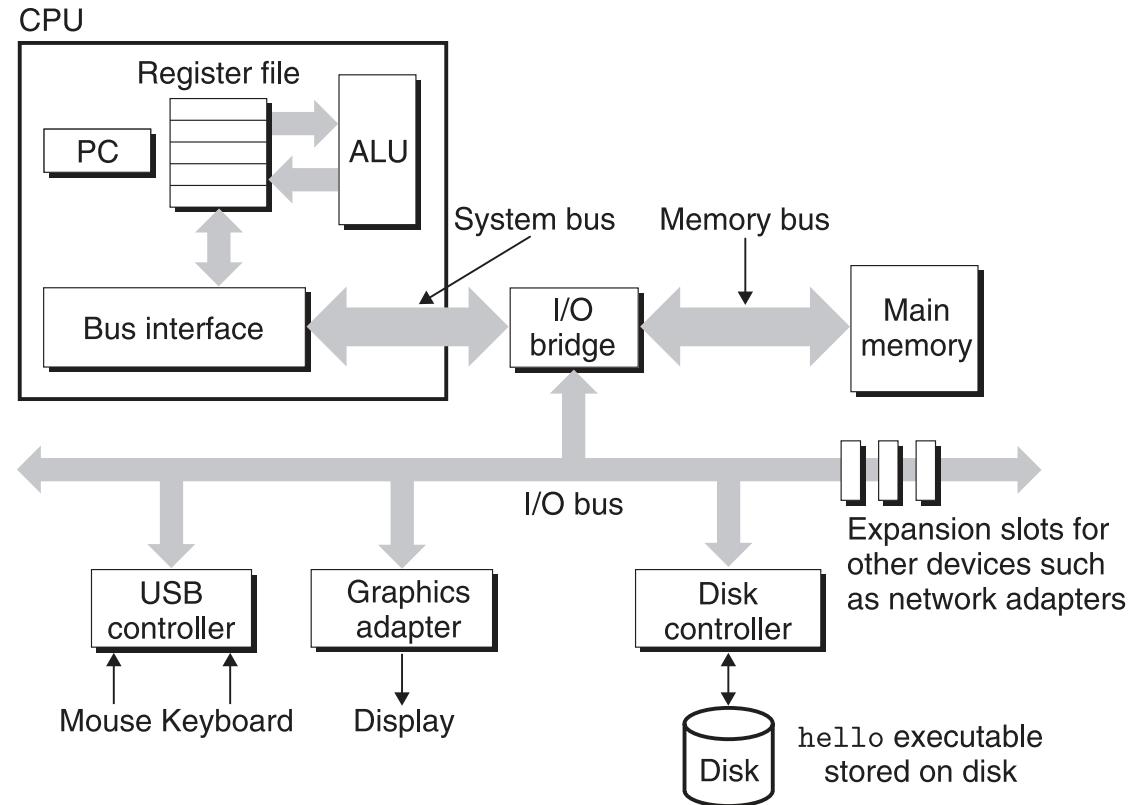
Computadora moderna

Corriendo un programa



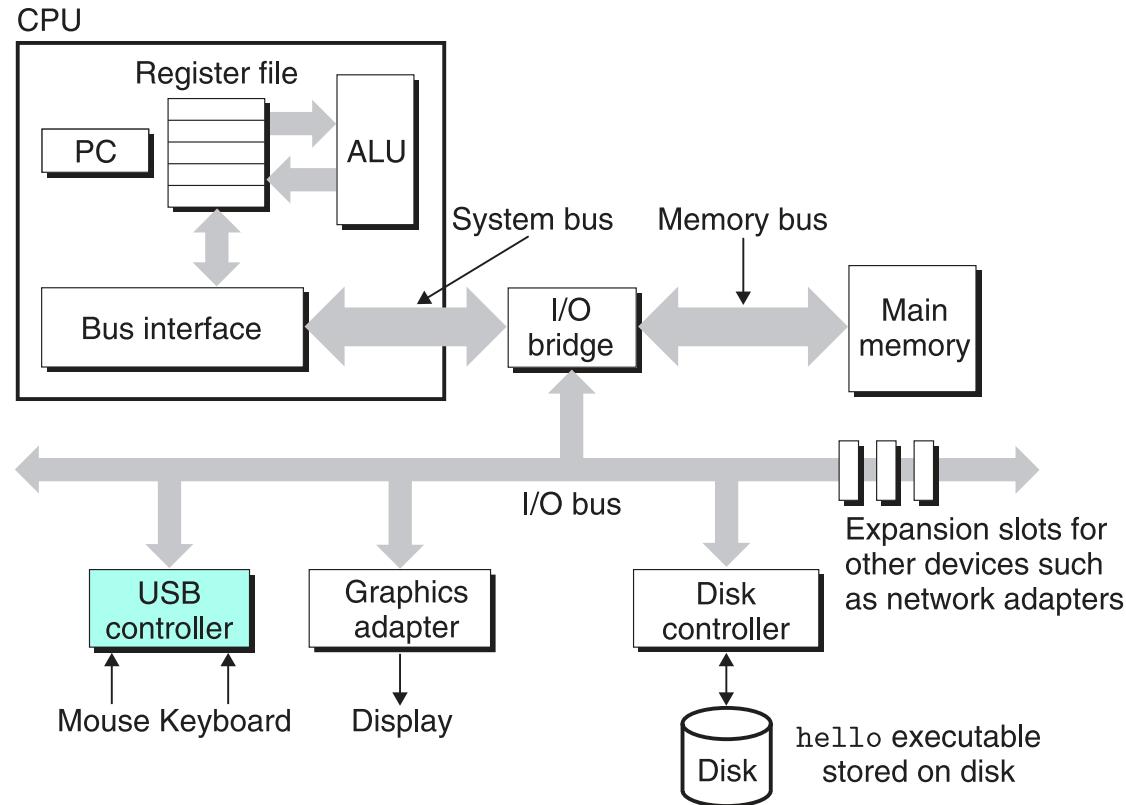
Computadora moderna

Corriendo un programa



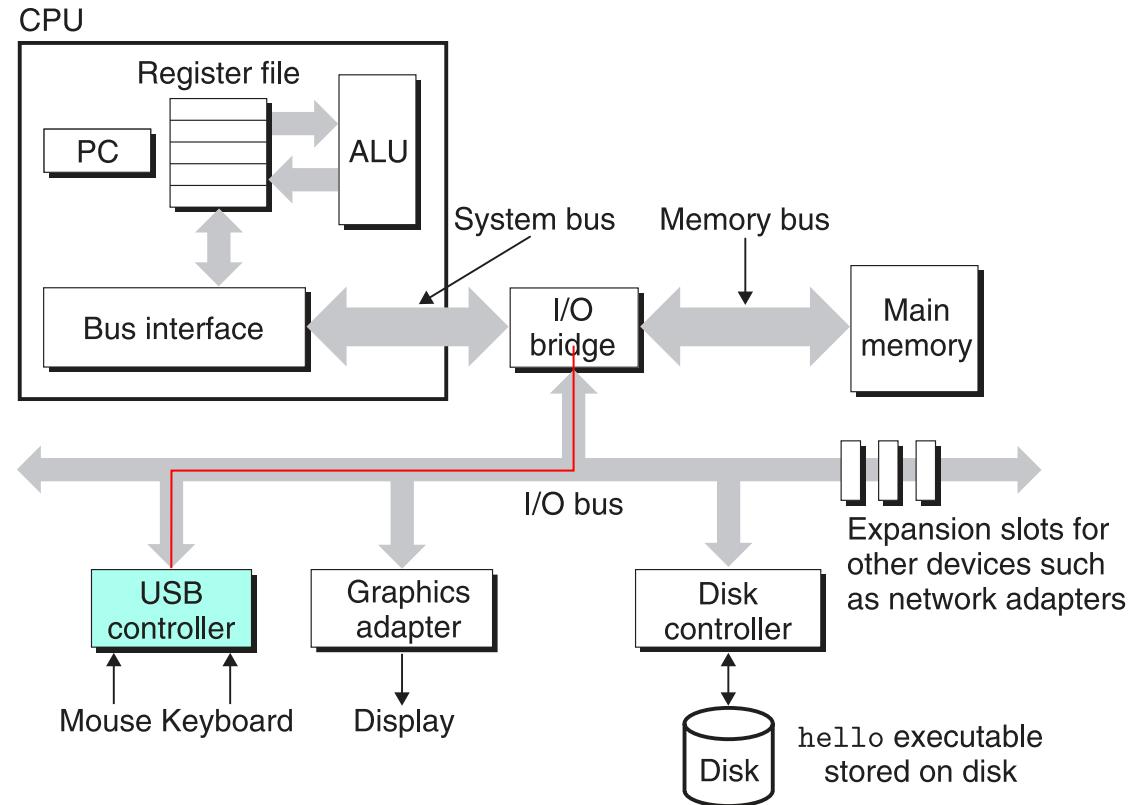
Computadora moderna

Corriendo un programa. Ingreso del programa solicitado



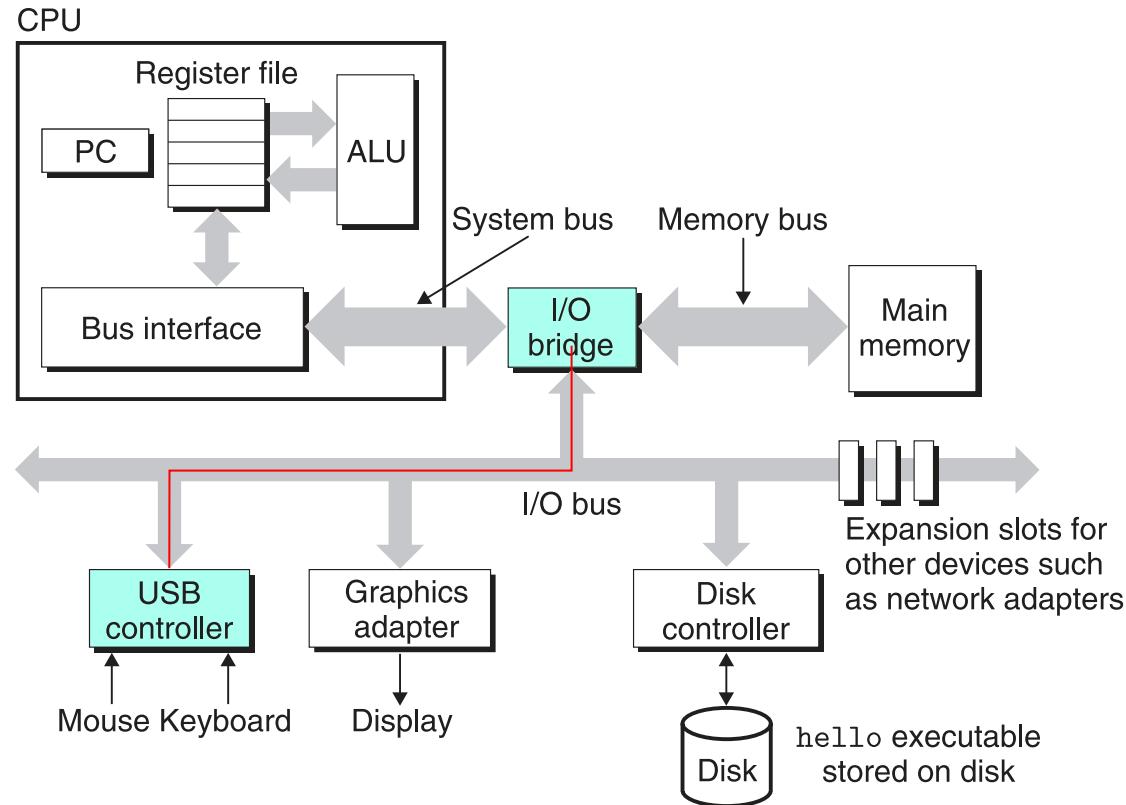
Computadora moderna

Corriendo un programa. Ingreso del programa solicitado



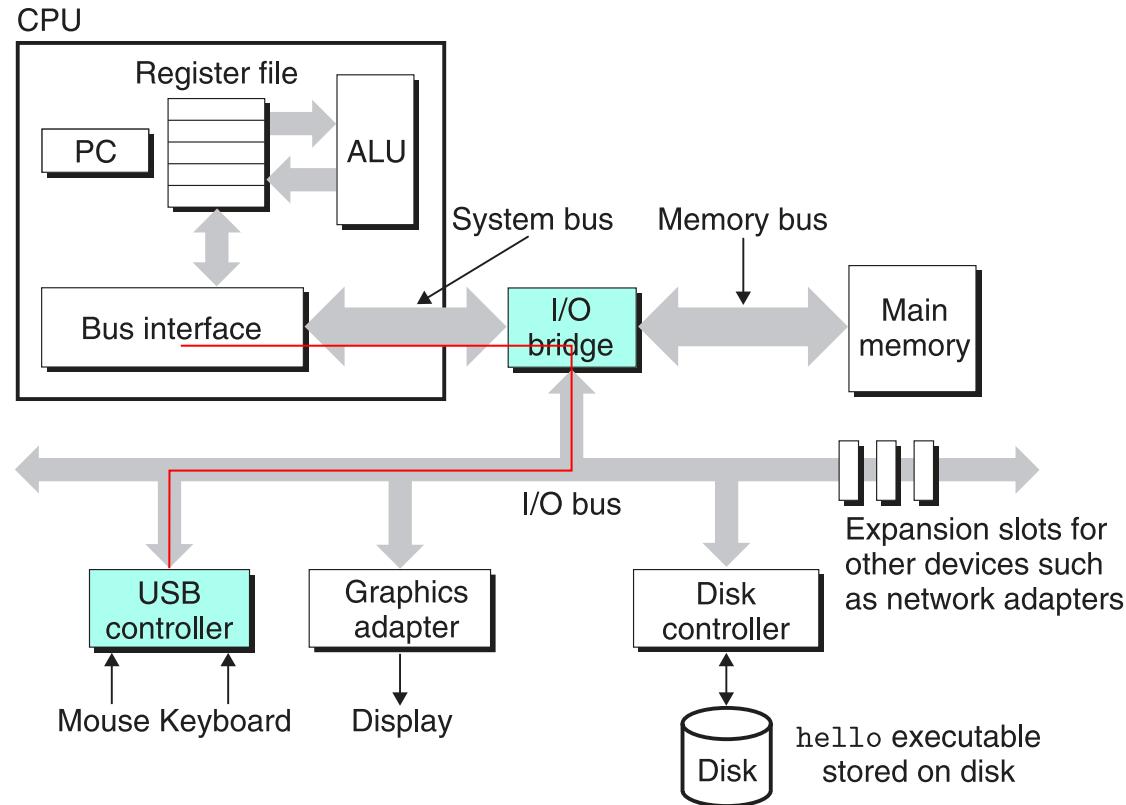
Computadora moderna

Corriendo un programa. Ingreso del programa solicitado



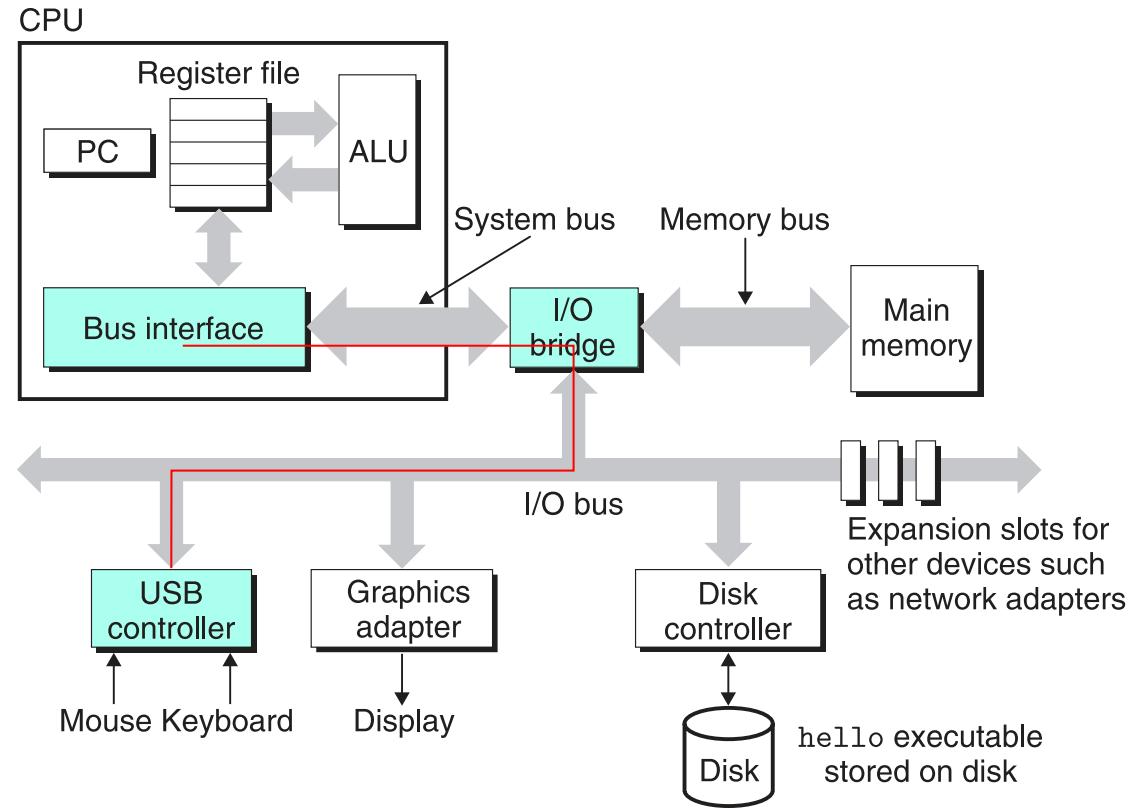
Computadora moderna

Corriendo un programa. Ingreso del programa solicitado



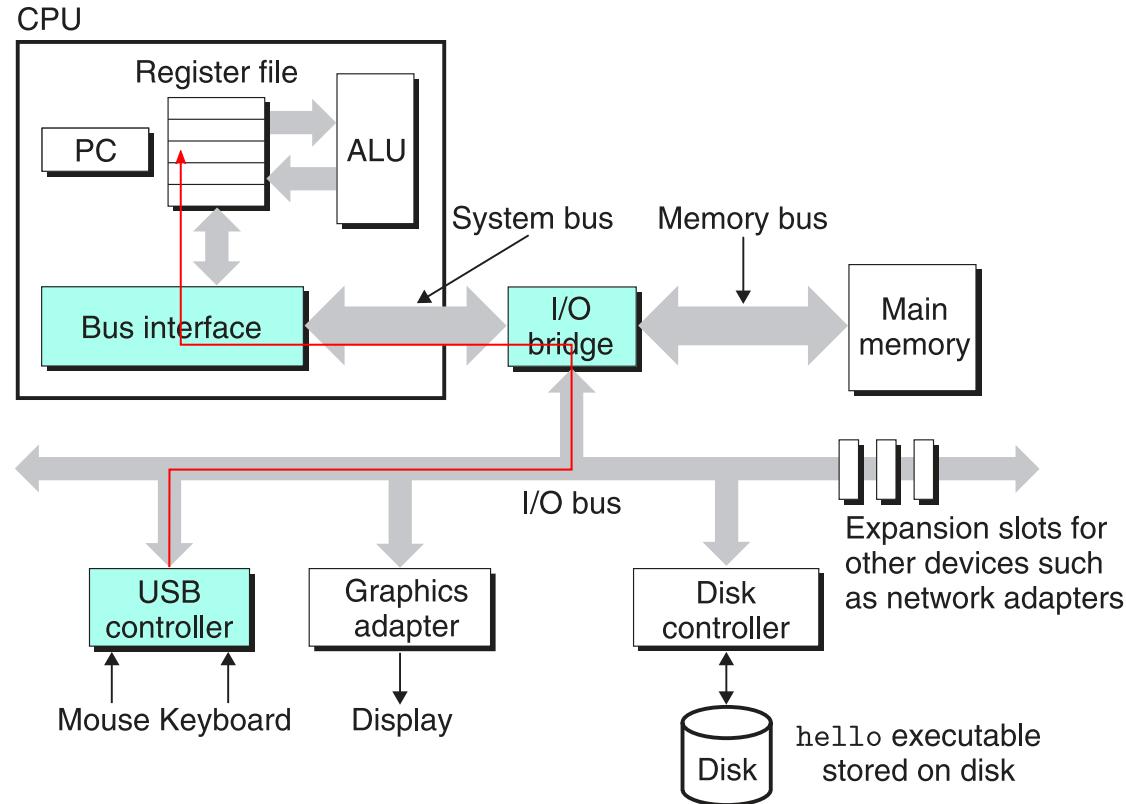
Computadora moderna

Corriendo un programa. Ingreso del programa solicitado



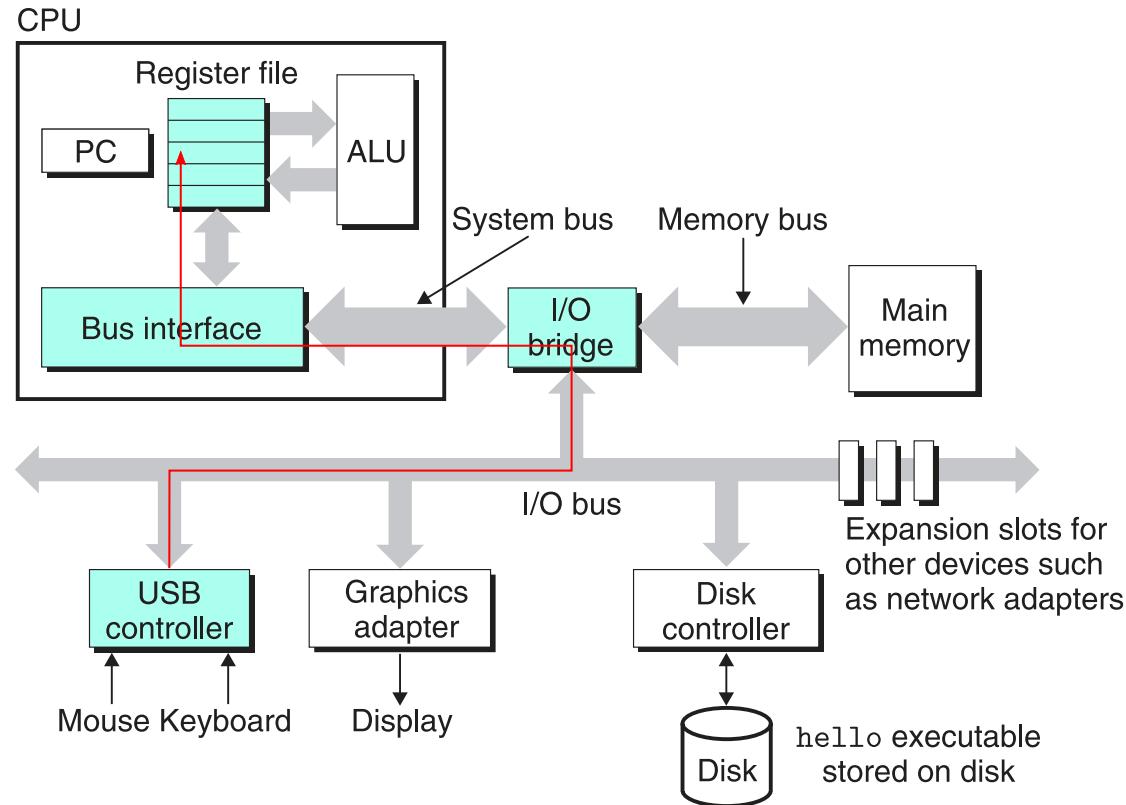
Computadora moderna

Corriendo un programa. Ingreso del programa solicitado



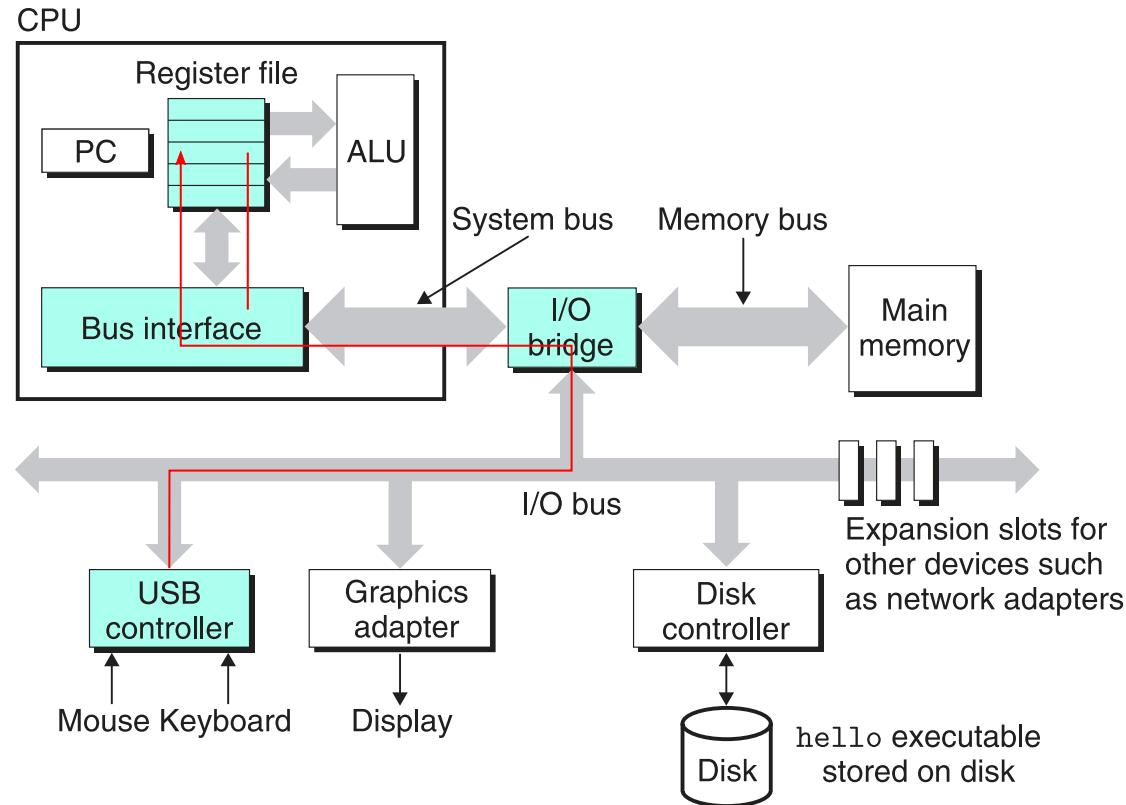
Computadora moderna

Corriendo un programa. Ingreso del programa solicitado



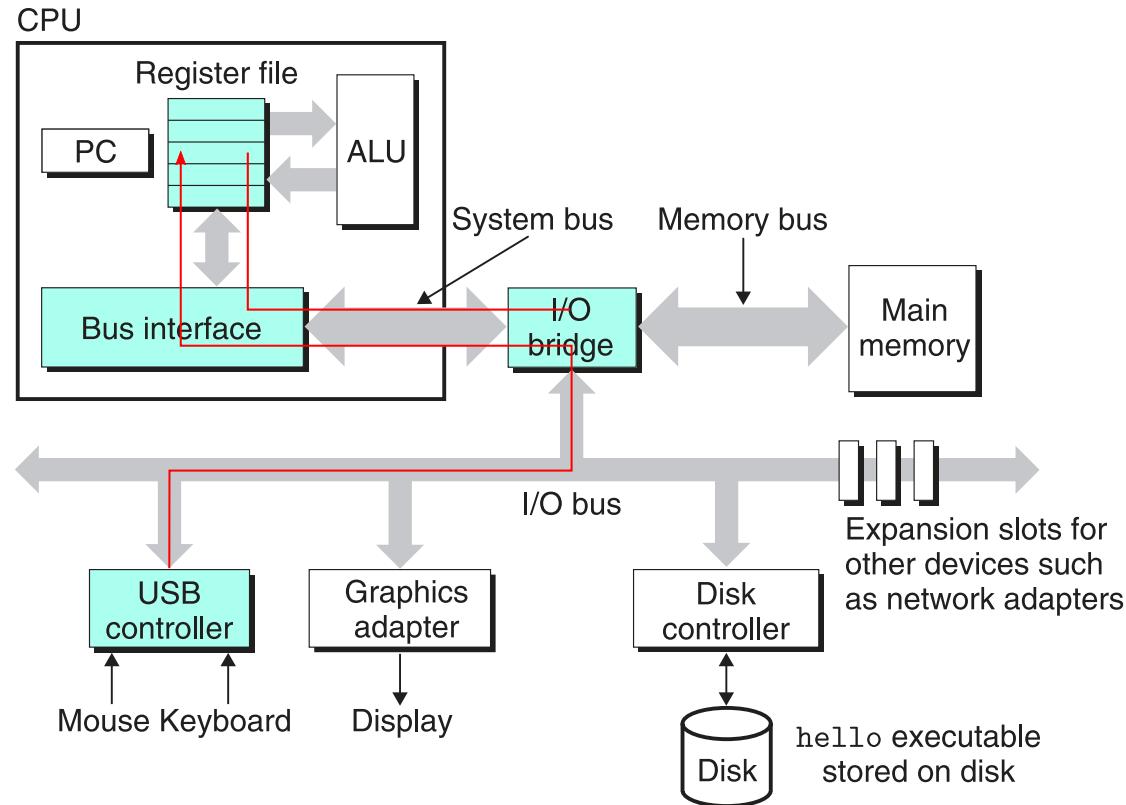
Computadora moderna

Corriendo un programa. Ingreso del programa solicitado



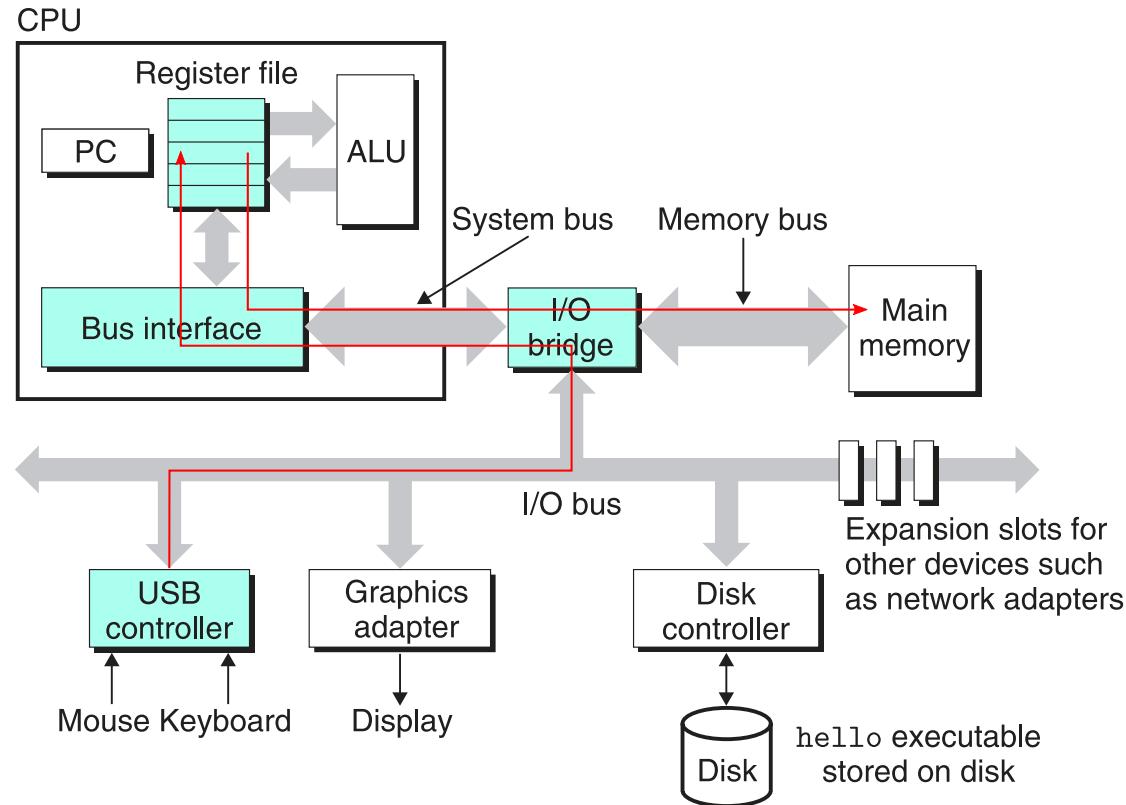
Computadora moderna

Corriendo un programa. Ingreso del programa solicitado



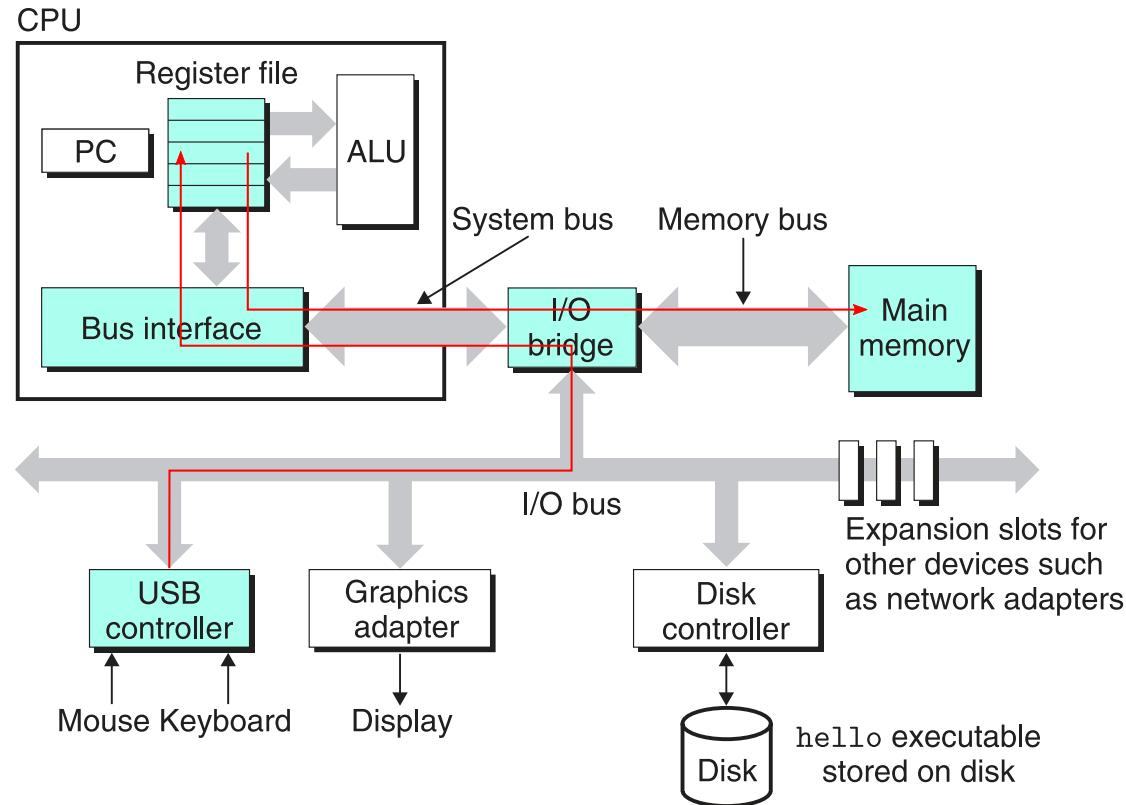
Computadora moderna

Corriendo un programa. Ingreso del programa solicitado



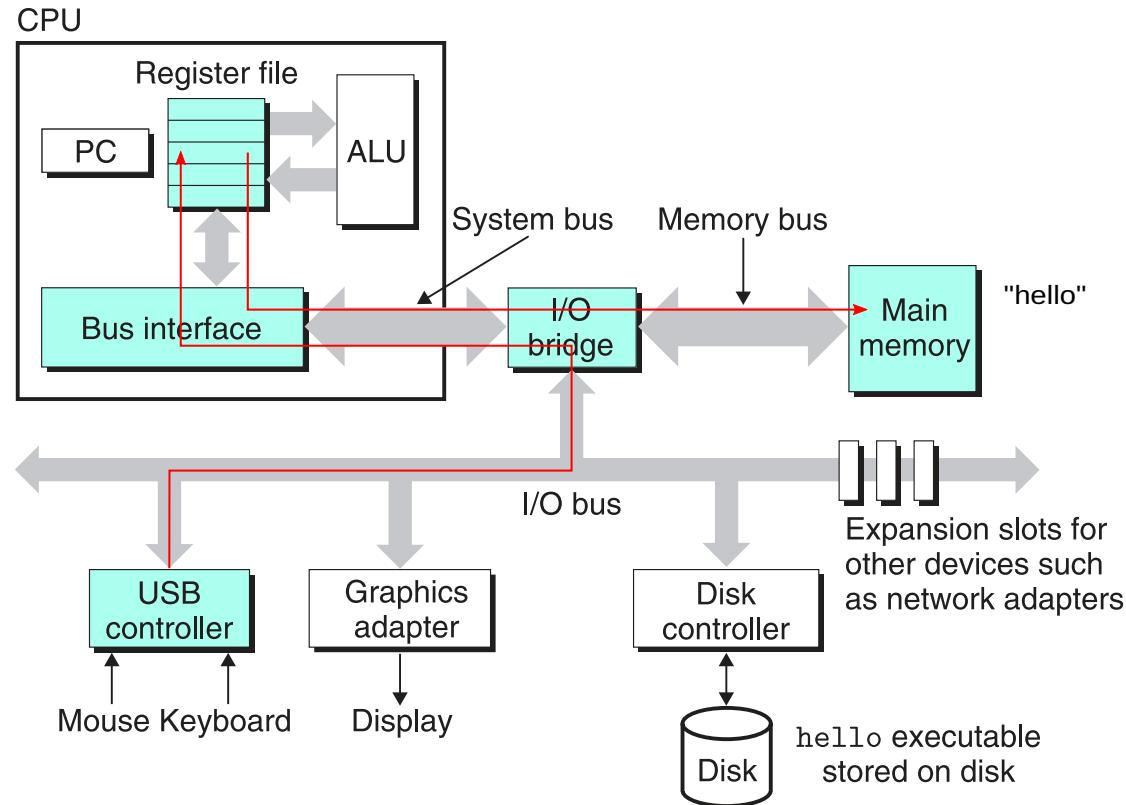
Computadora moderna

Corriendo un programa. Ingreso del programa solicitado



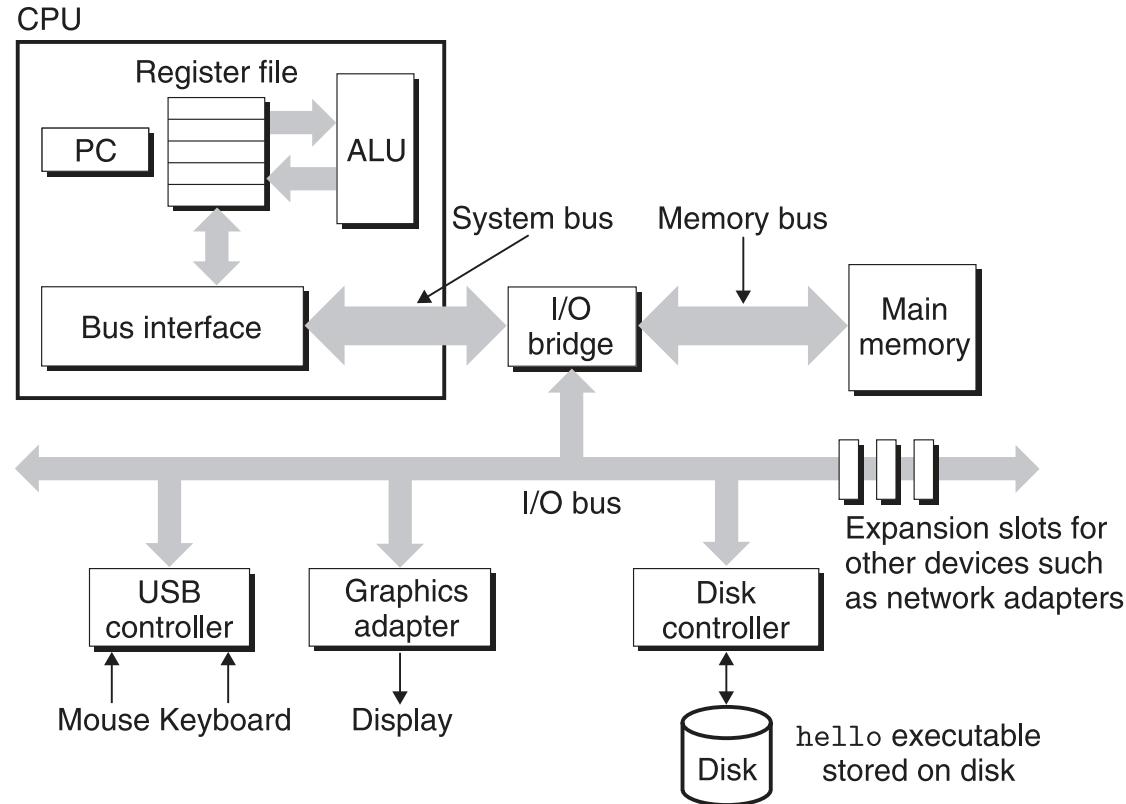
Computadora moderna

Corriendo un programa. Ingreso del programa solicitado



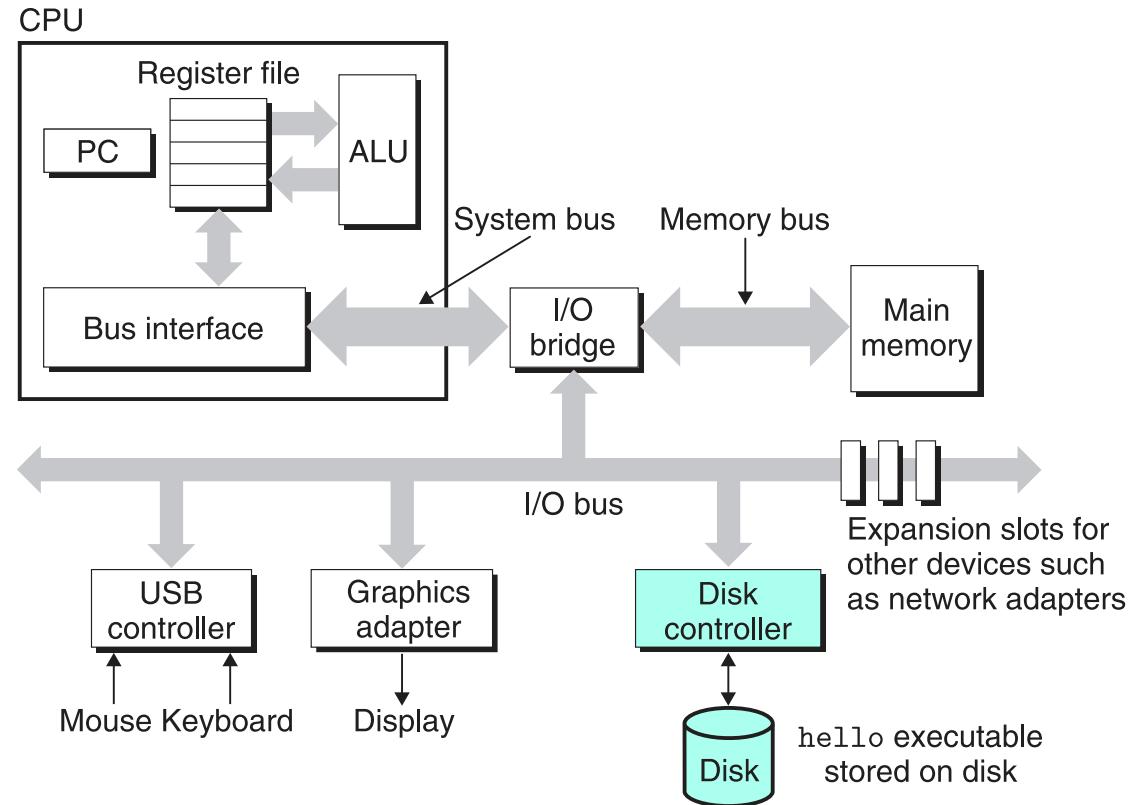
Computadora moderna

Corriendo un programa.



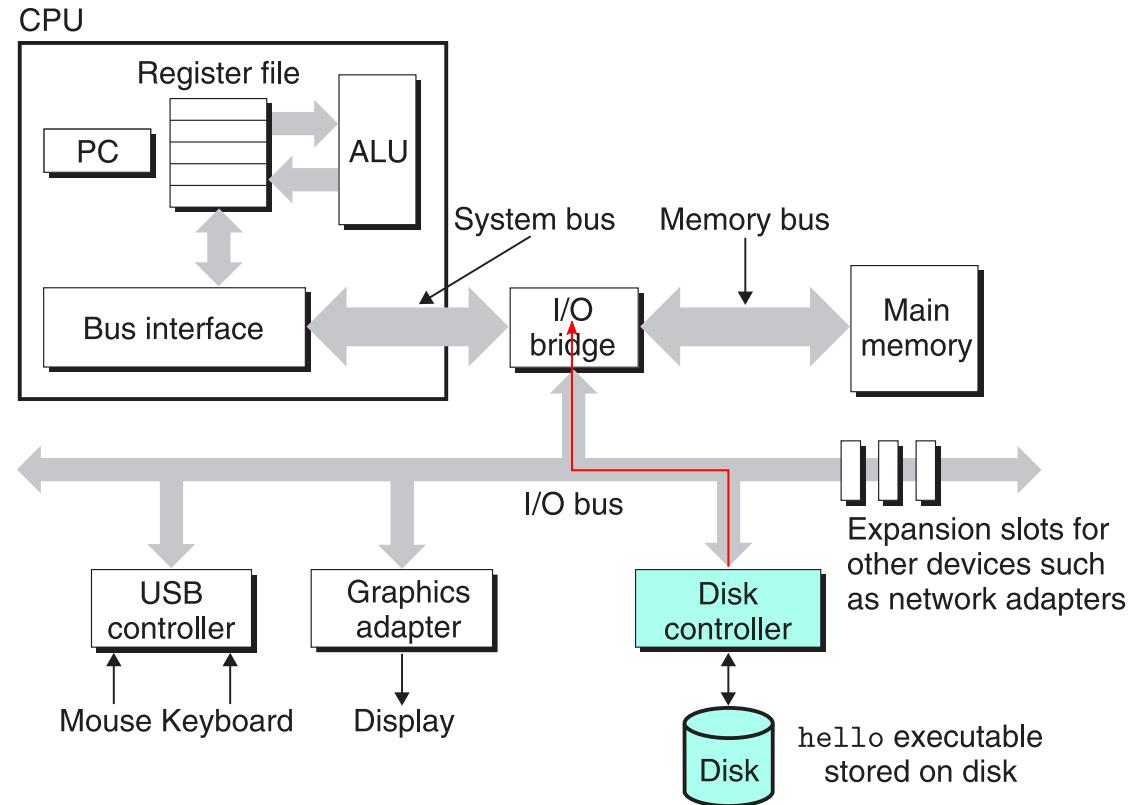
Computadora moderna

Corriendo un programa. Carga a memoria del programa



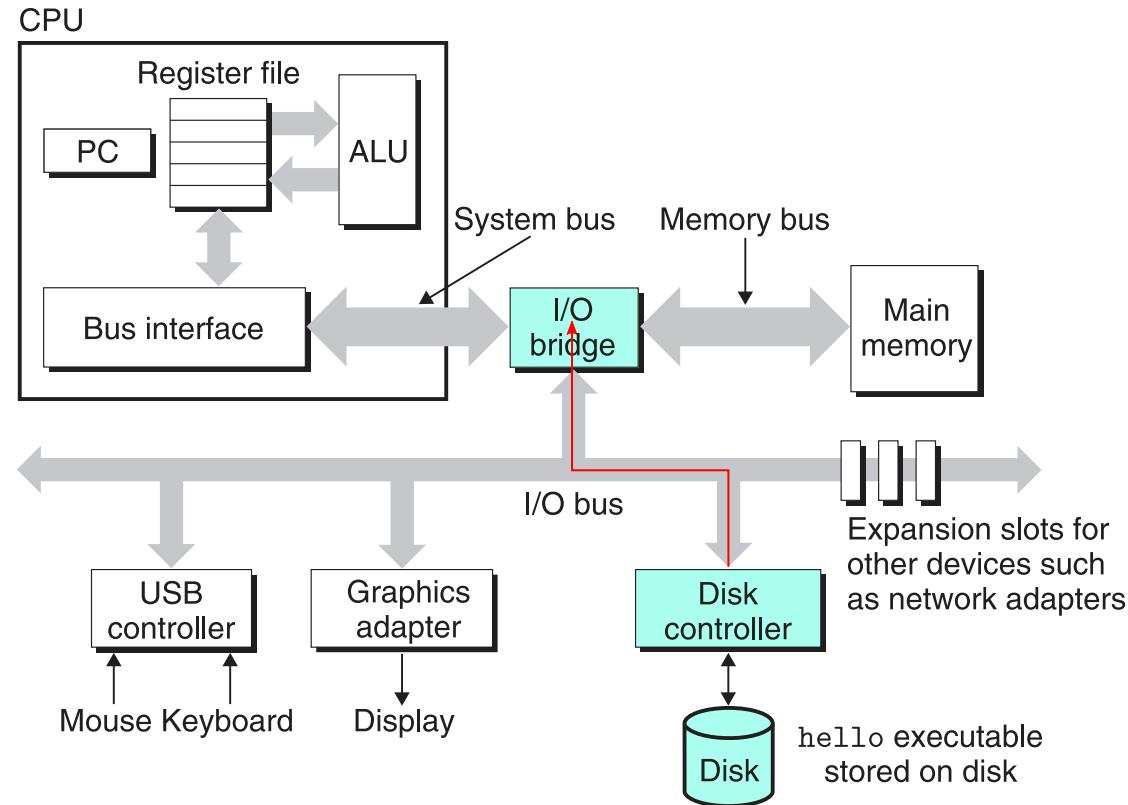
Computadora moderna

Corriendo un programa. Carga a memoria del programa



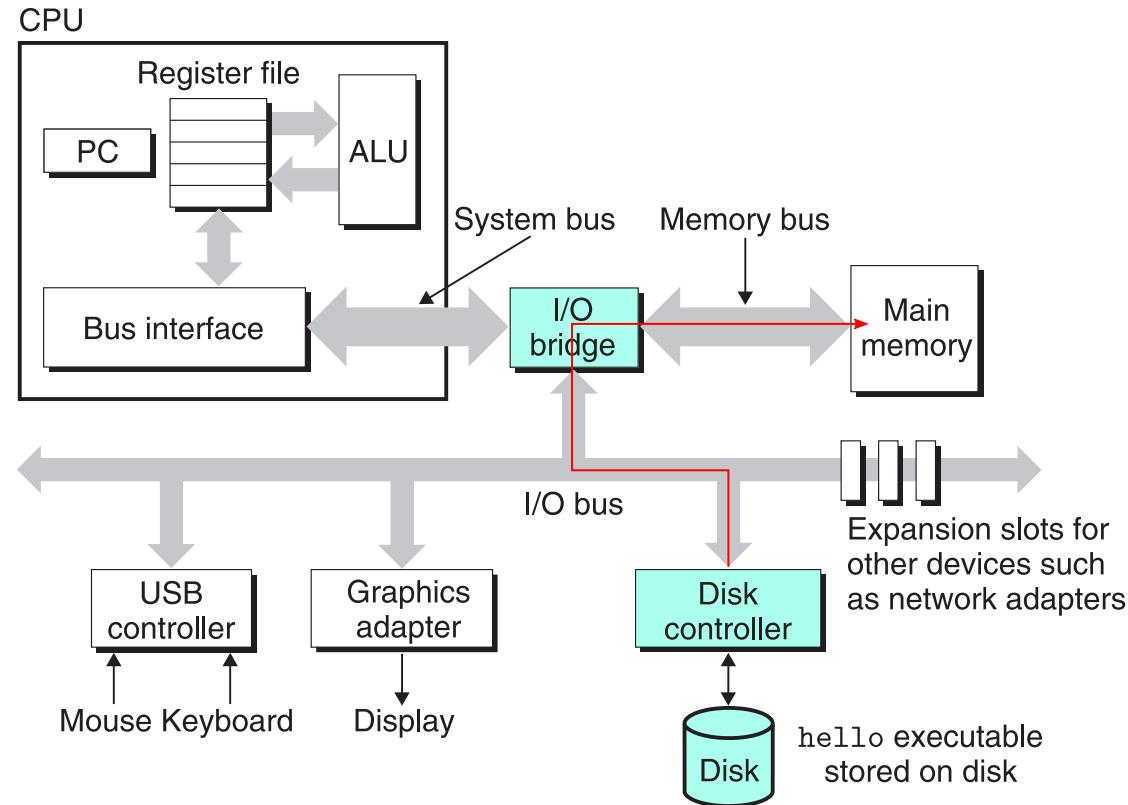
Computadora moderna

Corriendo un programa. Carga a memoria del programa



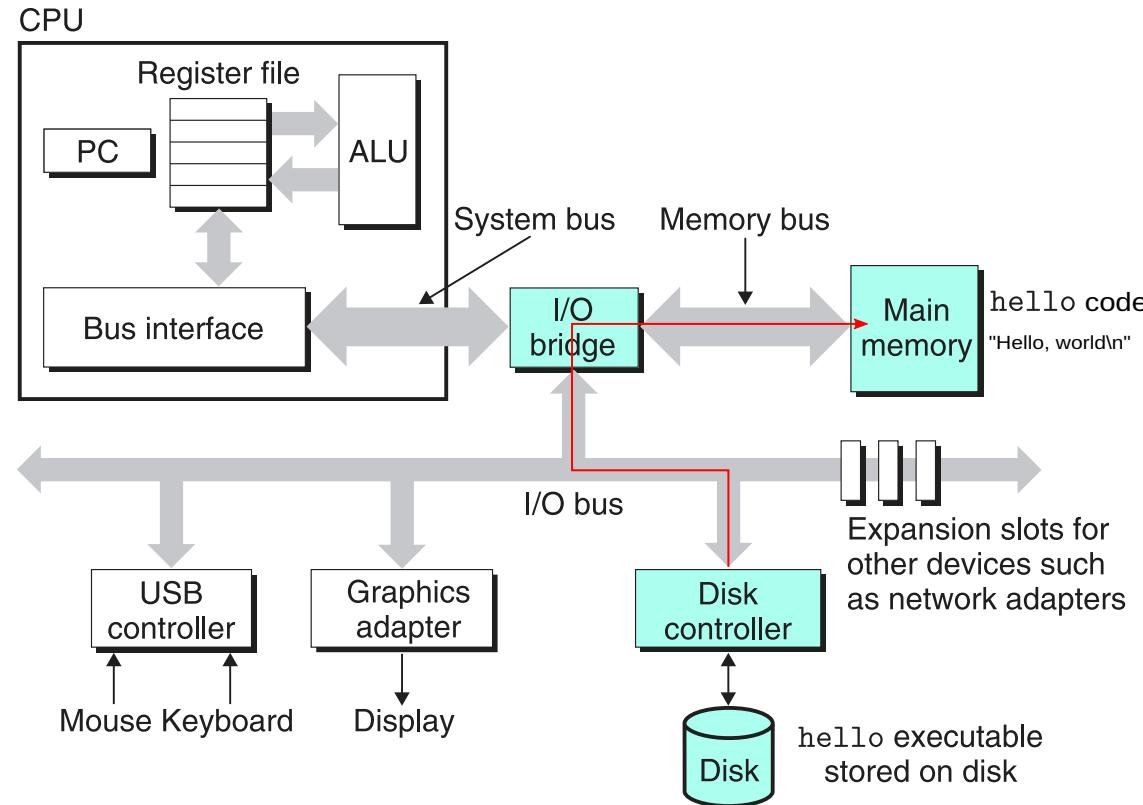
Computadora moderna

Corriendo un programa. Carga a memoria del programa



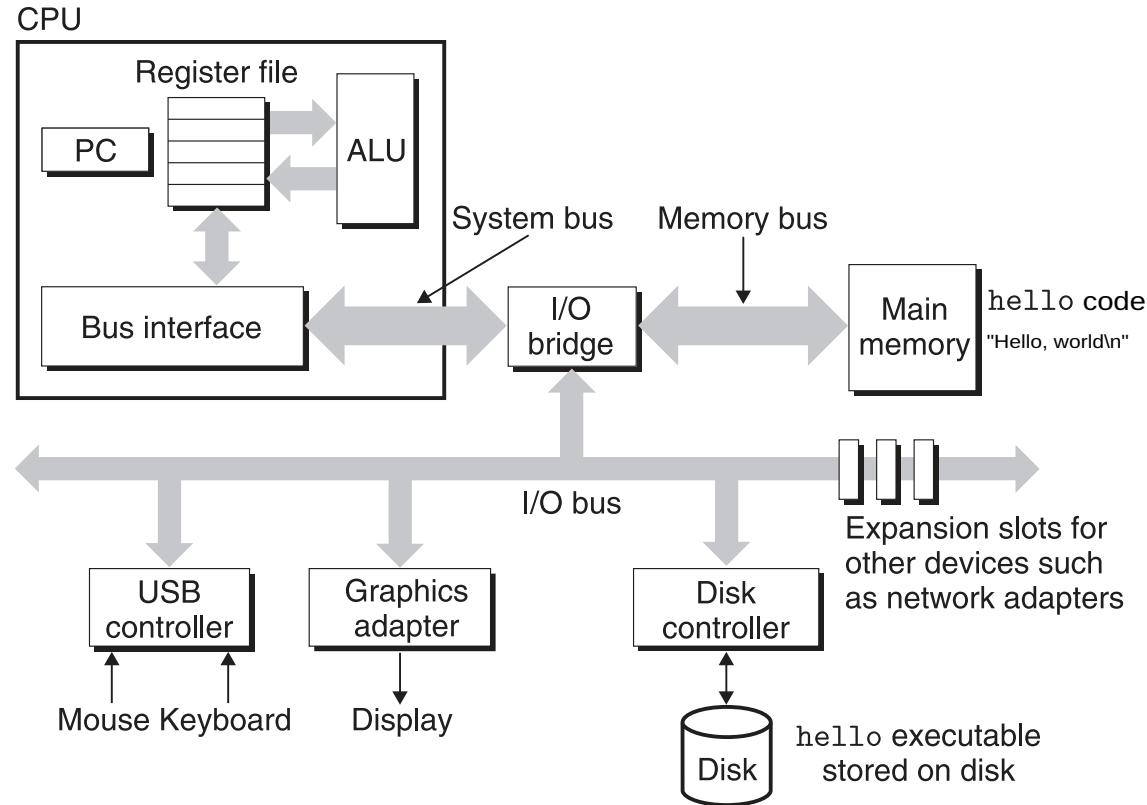
Computadora moderna

Corriendo un programa. Carga a memoria del programa



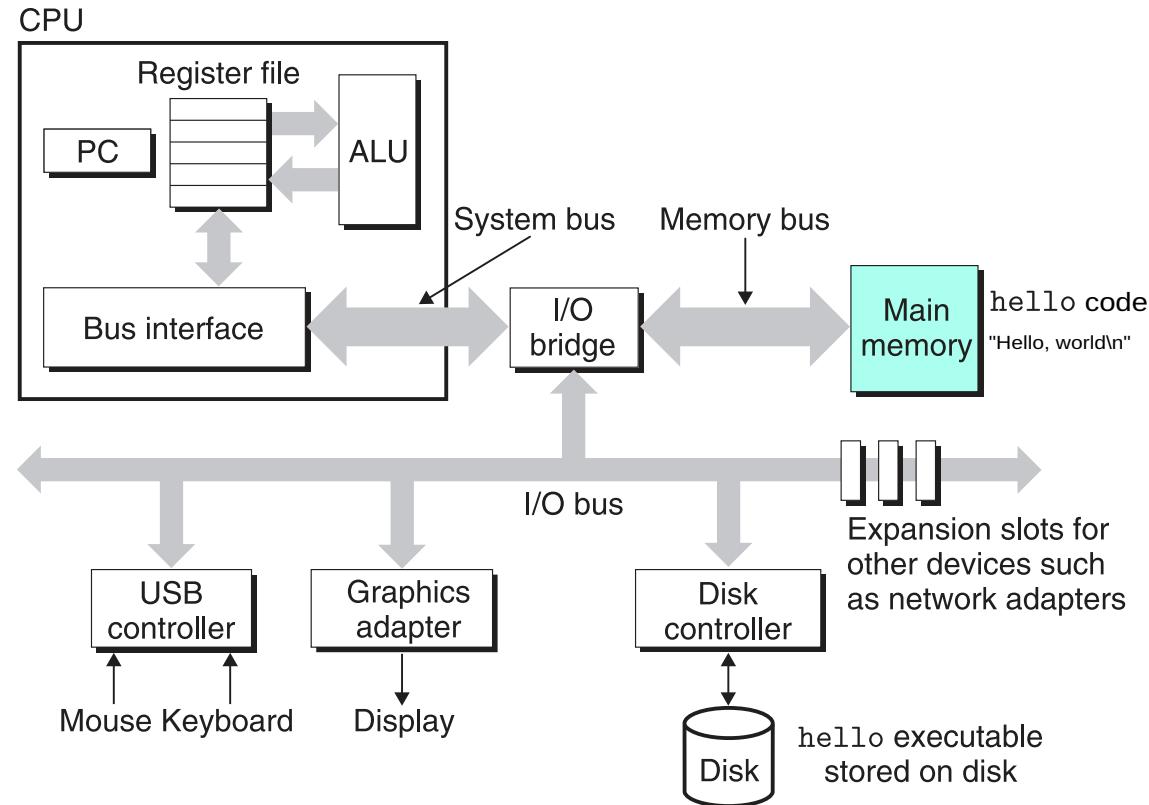
Computadora moderna

Corriendo un programa.



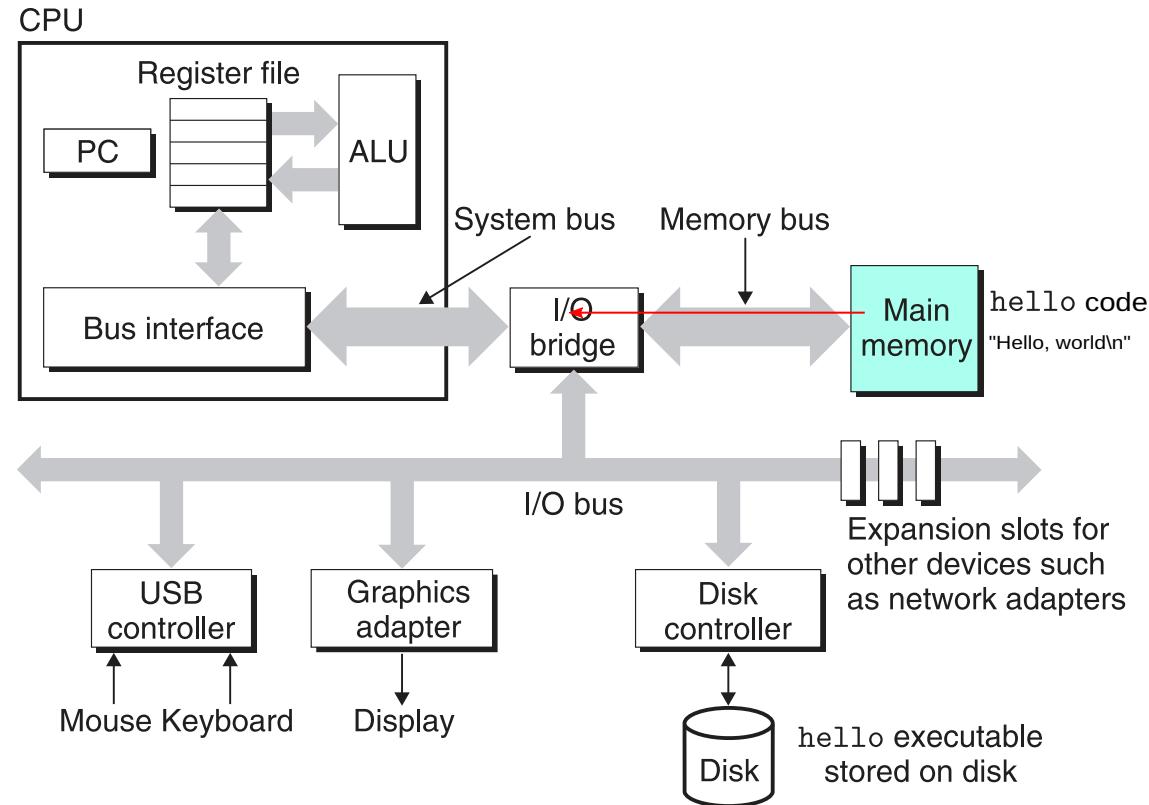
Computadora moderna

Corriendo un programa. Ejecución: Muestra en pantalla



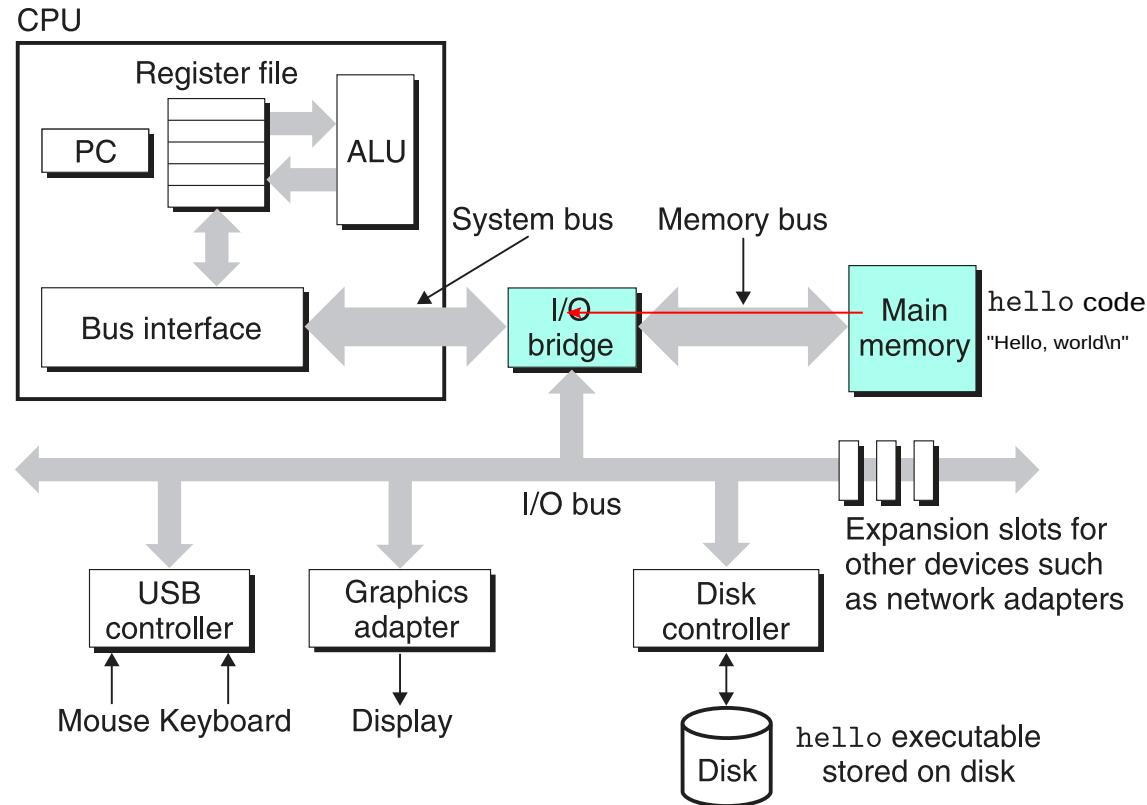
Computadora moderna

Corriendo un programa. Ejecución: Muestra en pantalla



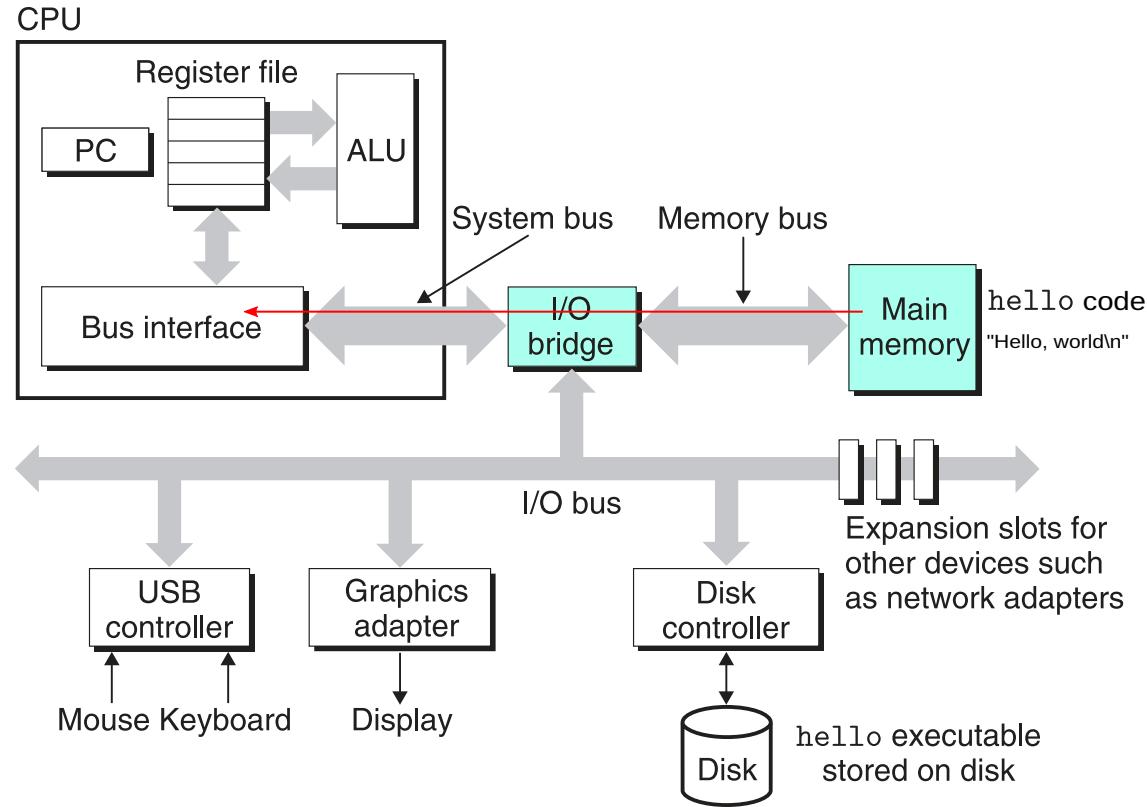
Computadora moderna

Corriendo un programa. Ejecución: Muestra en pantalla



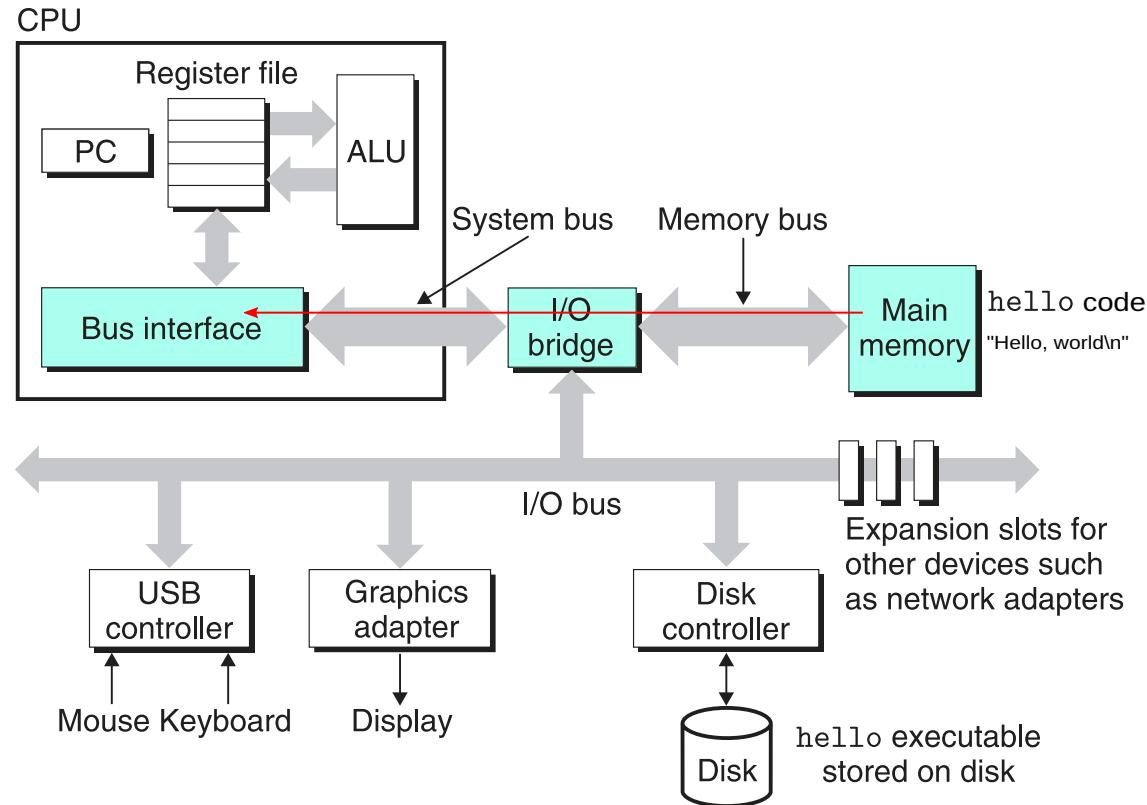
Computadora moderna

Corriendo un programa. Ejecución: Muestra en pantalla



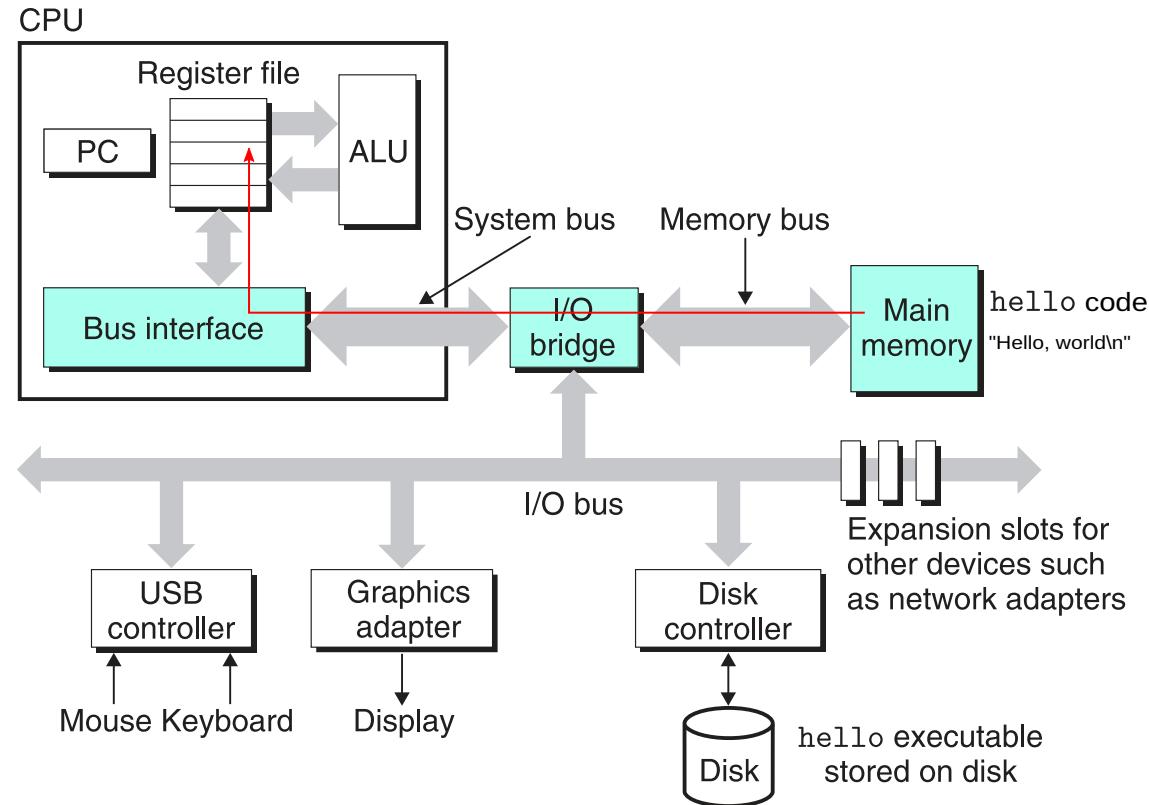
Computadora moderna

Corriendo un programa. Ejecución: Muestra en pantalla



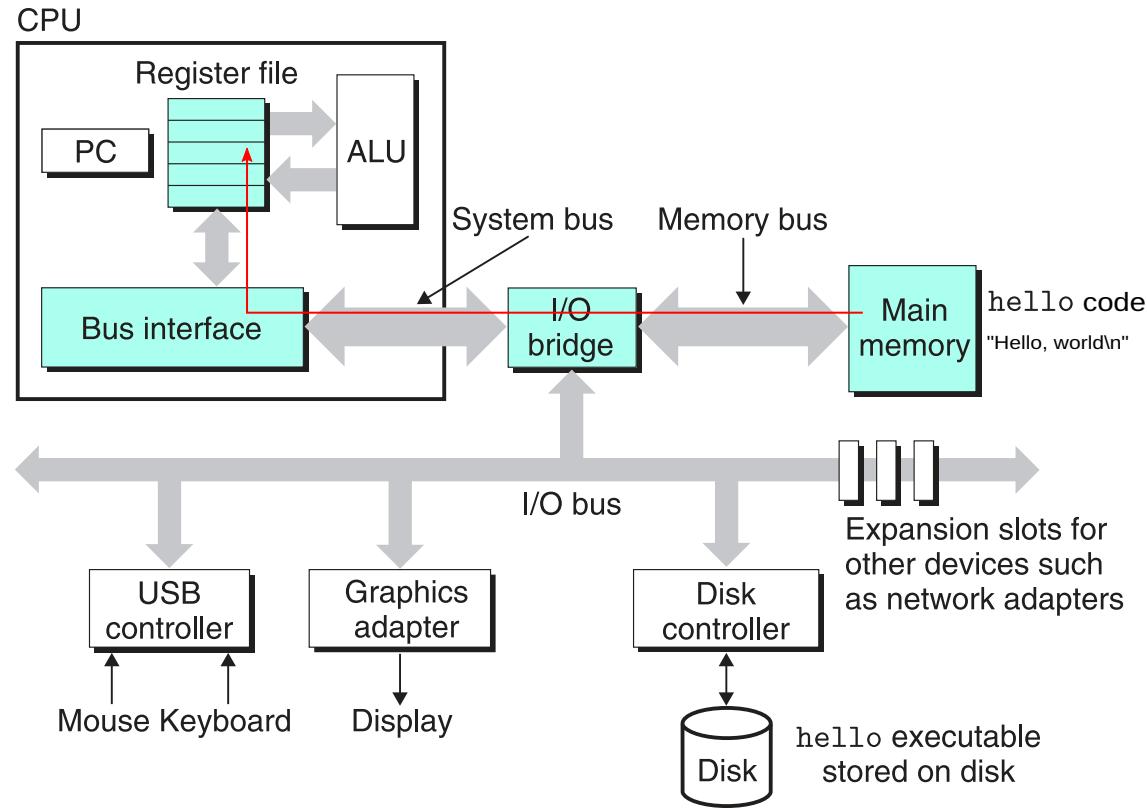
Computadora moderna

Corriendo un programa. Ejecución: Muestra en pantalla



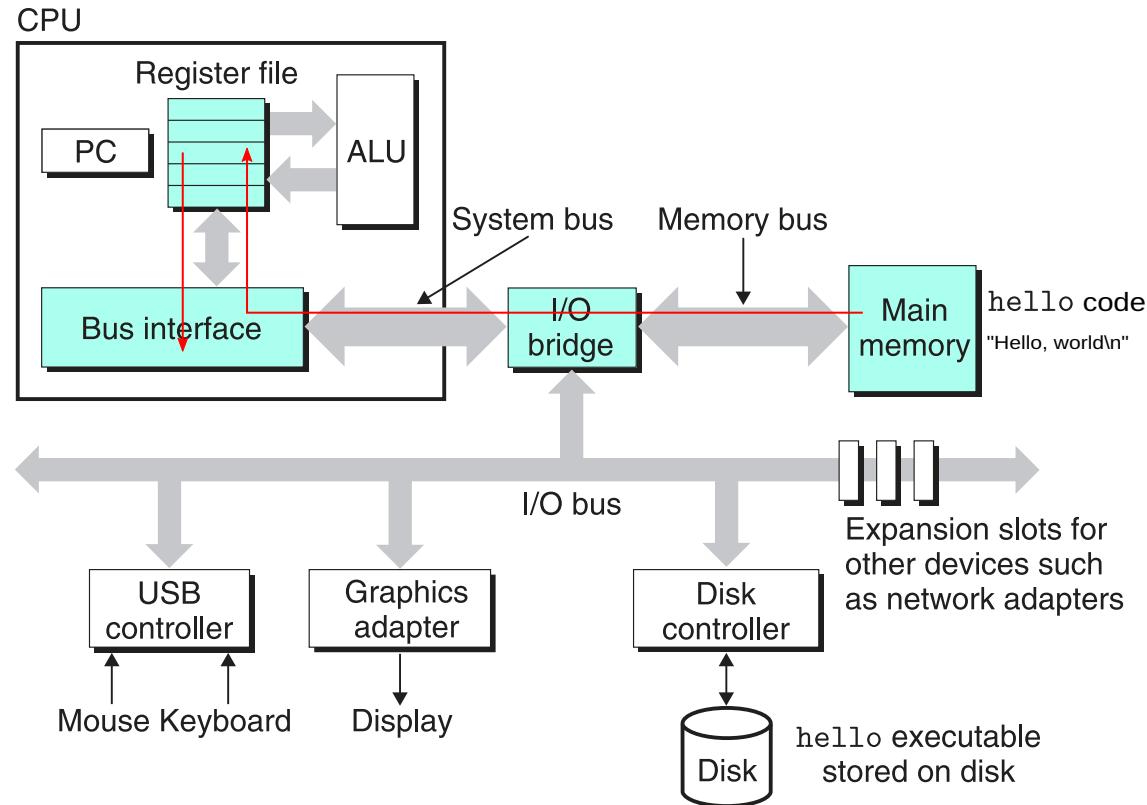
Computadora moderna

Corriendo un programa. Ejecución: Muestra en pantalla



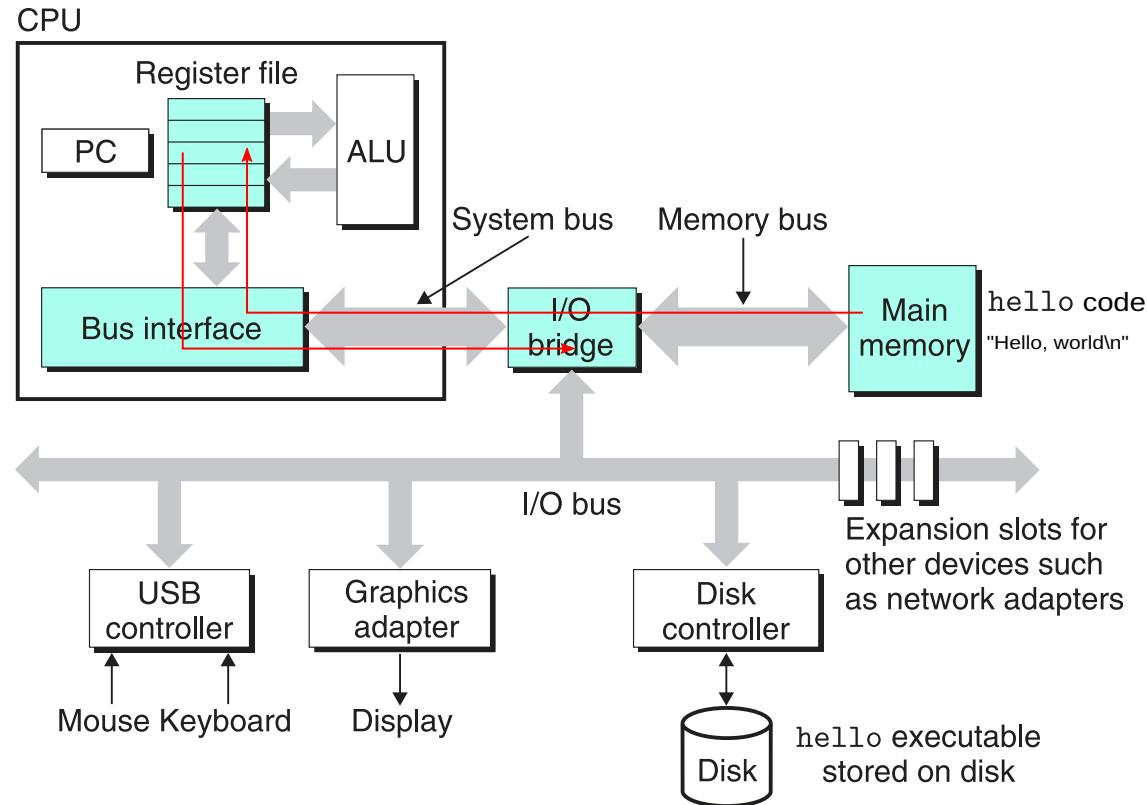
Computadora moderna

Corriendo un programa. Ejecución: Muestra en pantalla



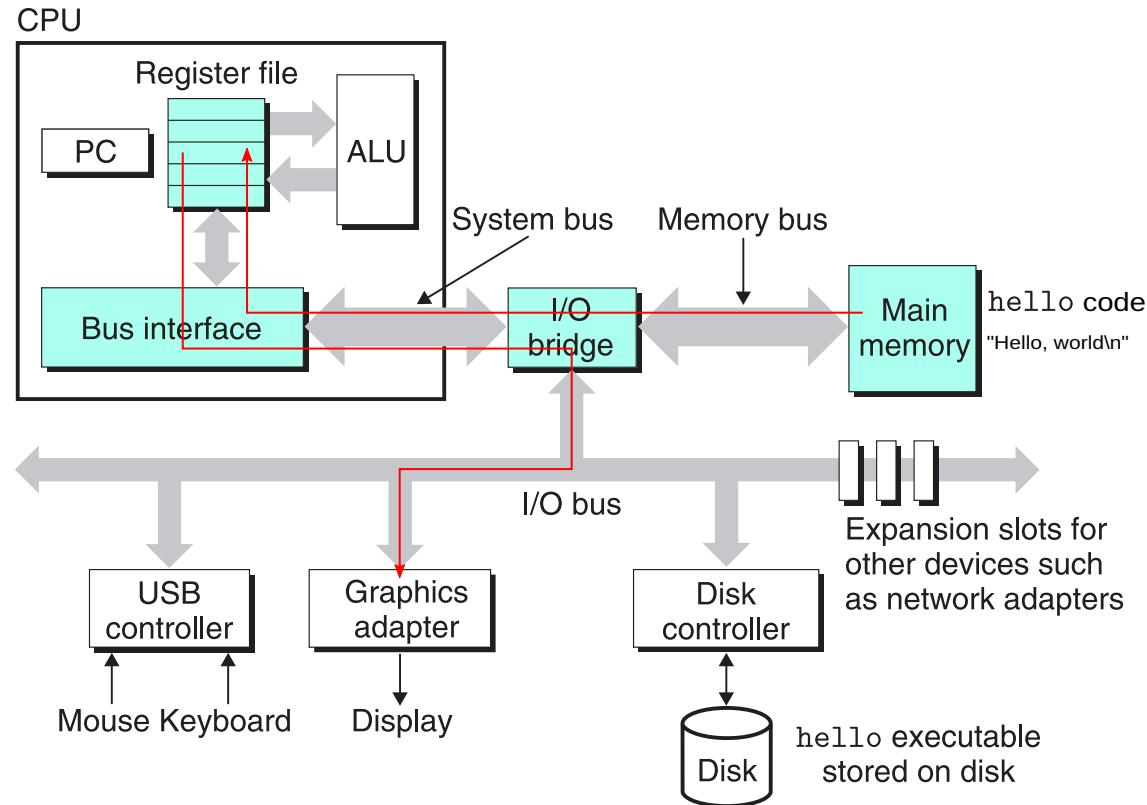
Computadora moderna

Corriendo un programa. Ejecución: Muestra en pantalla



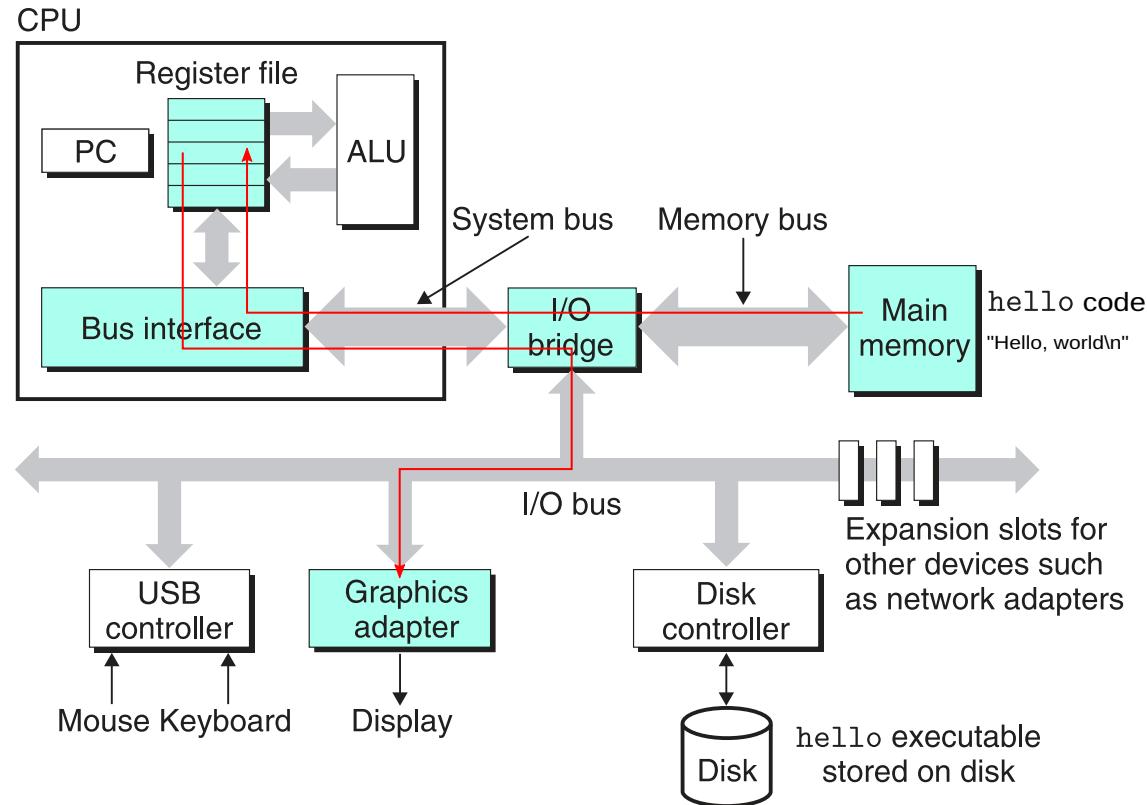
Computadora moderna

Corriendo un programa. Ejecución: Muestra en pantalla



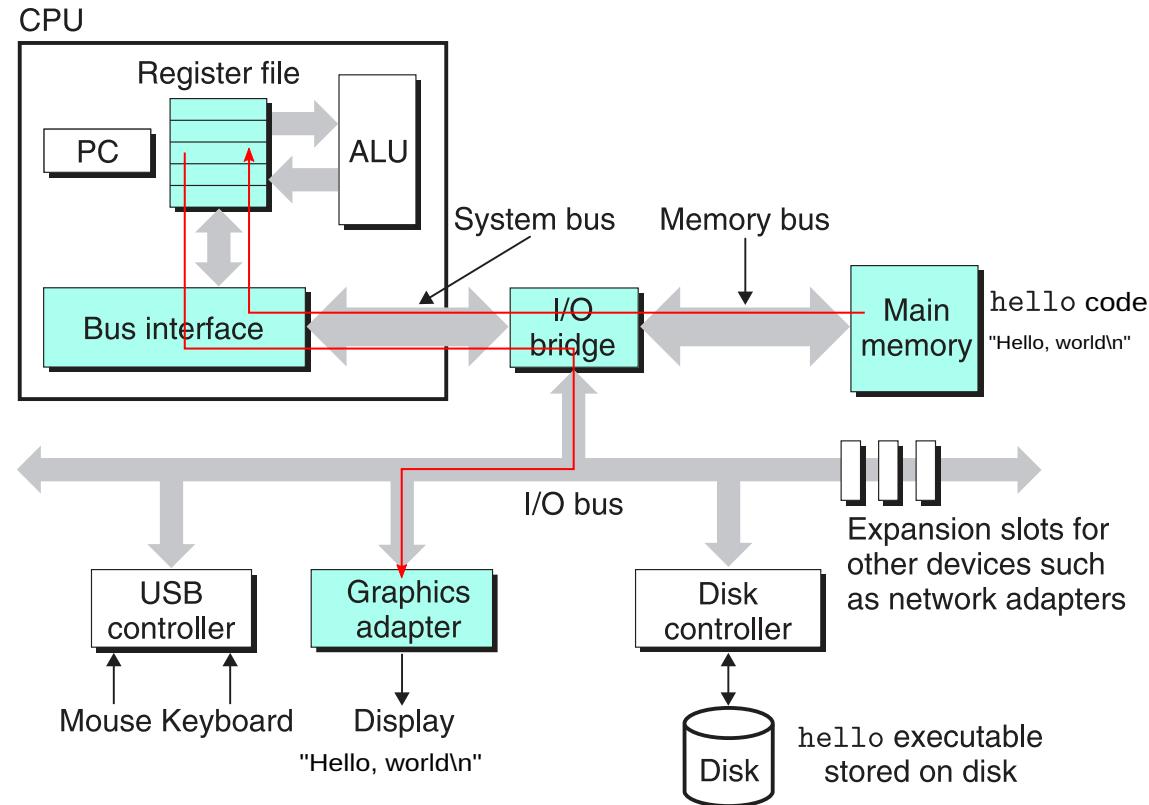
Computadora moderna

Corriendo un programa. Ejecución: Muestra en pantalla



Computadora moderna

Corriendo un programa. Ejecución: Muestra en pantalla



Fases de compilación

Fases de compilación

Suponga un programa que deje un saludo escrito en lenguaje C, llamado hello.c

Fases de compilación

Suponga un programa que deje un saludo escrito en lenguaje C, llamado hello.c

```
#include <stdio.h>

int main (void)
{
    printf("Hola, mundo!\n");
    return 0;
}
```

Fases de compilación

Suponga un programa que deje un saludo escrito en lenguaje C, llamado hello.c

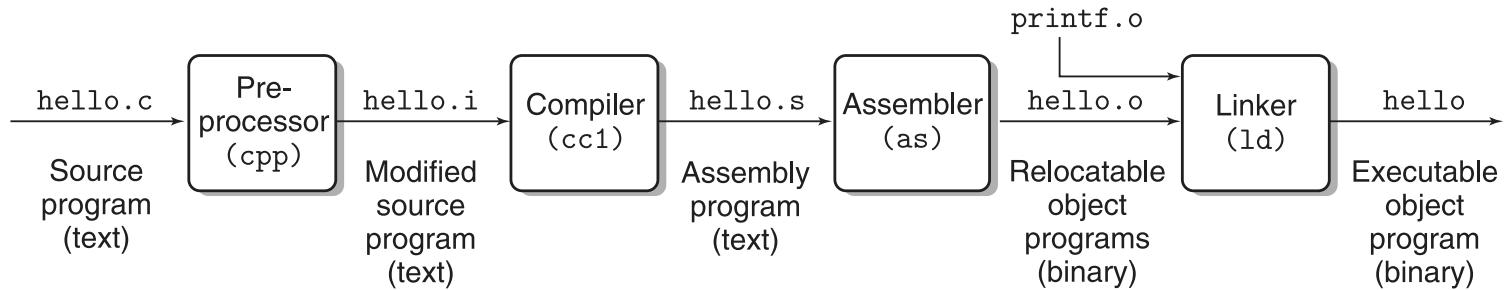
```
#include <stdio.h>

int main (void)
{
    printf("Hola, mundo!\n");
    return 0;
}
```

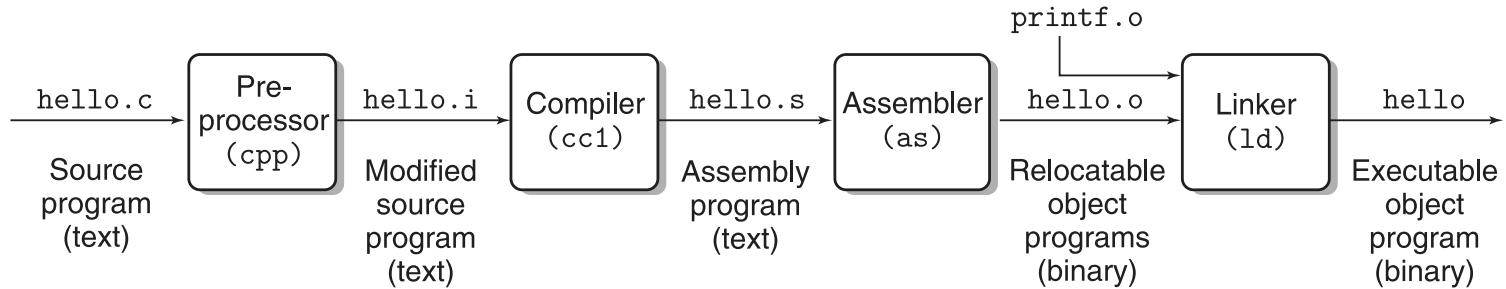
el cual se verá en detalle en la Unidad 3

Fases de compilación

Fases de compilación

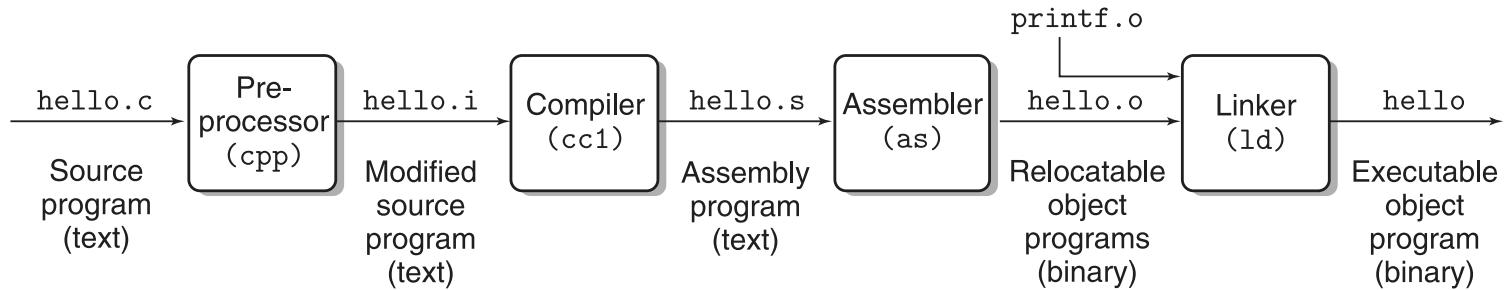


Fases de compilación

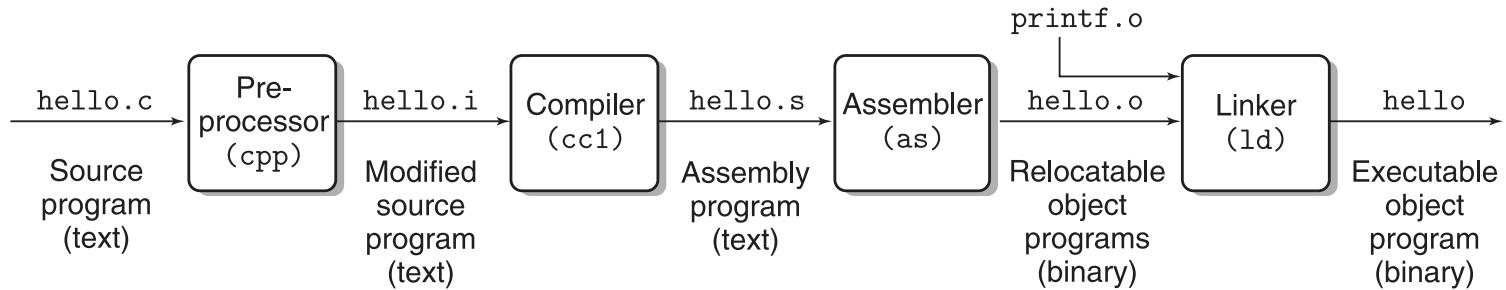


El **preprocesador** modifica el programa original agregando y quitando texto para dejar preparado el archivo `.i` para el compilador.

Fases de compilación

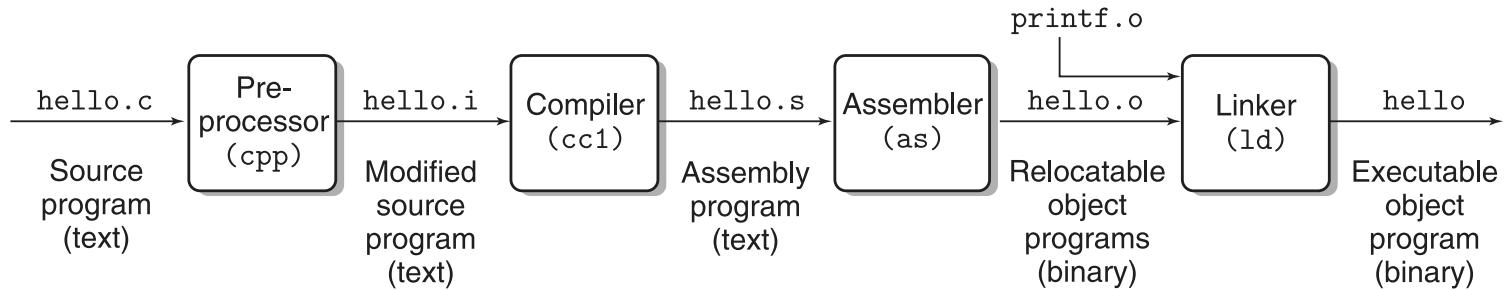


Fases de compilación

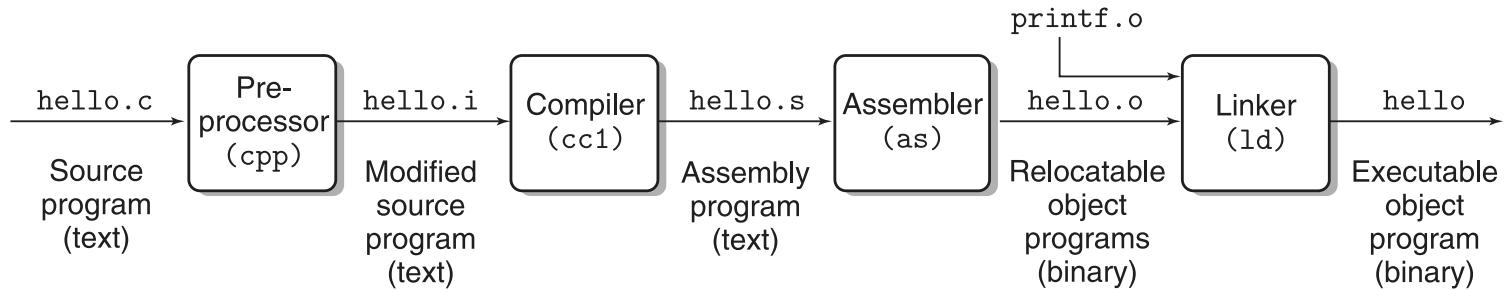


El **compilador** traduce el programa del archivo .i a lenguaje *assembly*, dejando el archivo .s

Fases de compilación

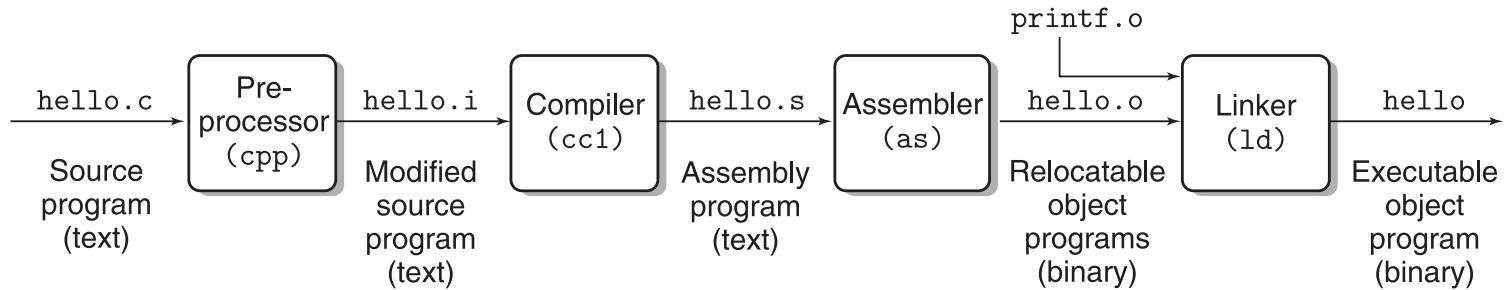


Fases de compilación

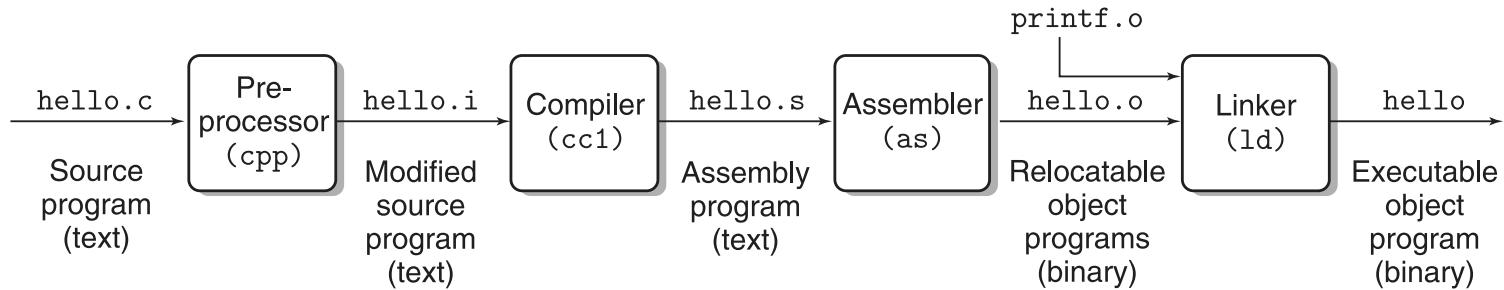


El **ensamblador** traduce el programa escrito en *assembly* a código objeto, comprensible por la computadora. Queda un archivo `.o`

Fases de compilación



Fases de compilación



Finalmente, el **linqueador** transforma compone un único archivo ejecutable con el .o que se creo a partir del código del usuario, junto con los .o de las funciones de las bibliotecas elegidas

Compilación y Ejecución

Compilación y Ejecución

En la terminal de linux...

```
$
```

Compilación y Ejecución

En la terminal de linux...

```
$ gcc hello.c -o hello
```

Compilación y Ejecución

En la terminal de linux...

```
$ gcc hello.c -o hello  
$
```

Compilación y Ejecución

En la terminal de linux...

```
$ gcc hello.c -o hello  
$ ./hello
```

Compilación y Ejecución

En la terminal de linux...

```
$ gcc hello.c -o hello
$ ./hello
Hola, mundo!
$
```

Compilación y Ejecución

En la terminal de linux...

```
$ gcc hello.c -o hello
$ ./hello
Hola, mundo!
$
```

Por suerte, todo el preprocessamiento, compilación, ensamblaje y linqueo se hace con el mismo comando, al mismo tiempo... con gcc

Compilación y Ejecución

En la terminal de linux...

```
$ gcc hello.c -o hello
$ ./hello
Hola, mundo!
$
```

Por suerte, todo el preprocessamiento, compilación, ensamblaje y linqueo se hace con el mismo comando, al mismo tiempo... con gcc

...aunque se puede hacer paso por paso si se quiere

Bibliografía

Computer Systems A Programmer's Perspective

Randal E. Bryant and David R. O'Hallaron

2015 Third edition.

Cap. 1

