Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC2343 – Arquitectura de Computadores

Comunicación de CPU y Memoria con I/O: Comunicación e Interacción - Parte 1

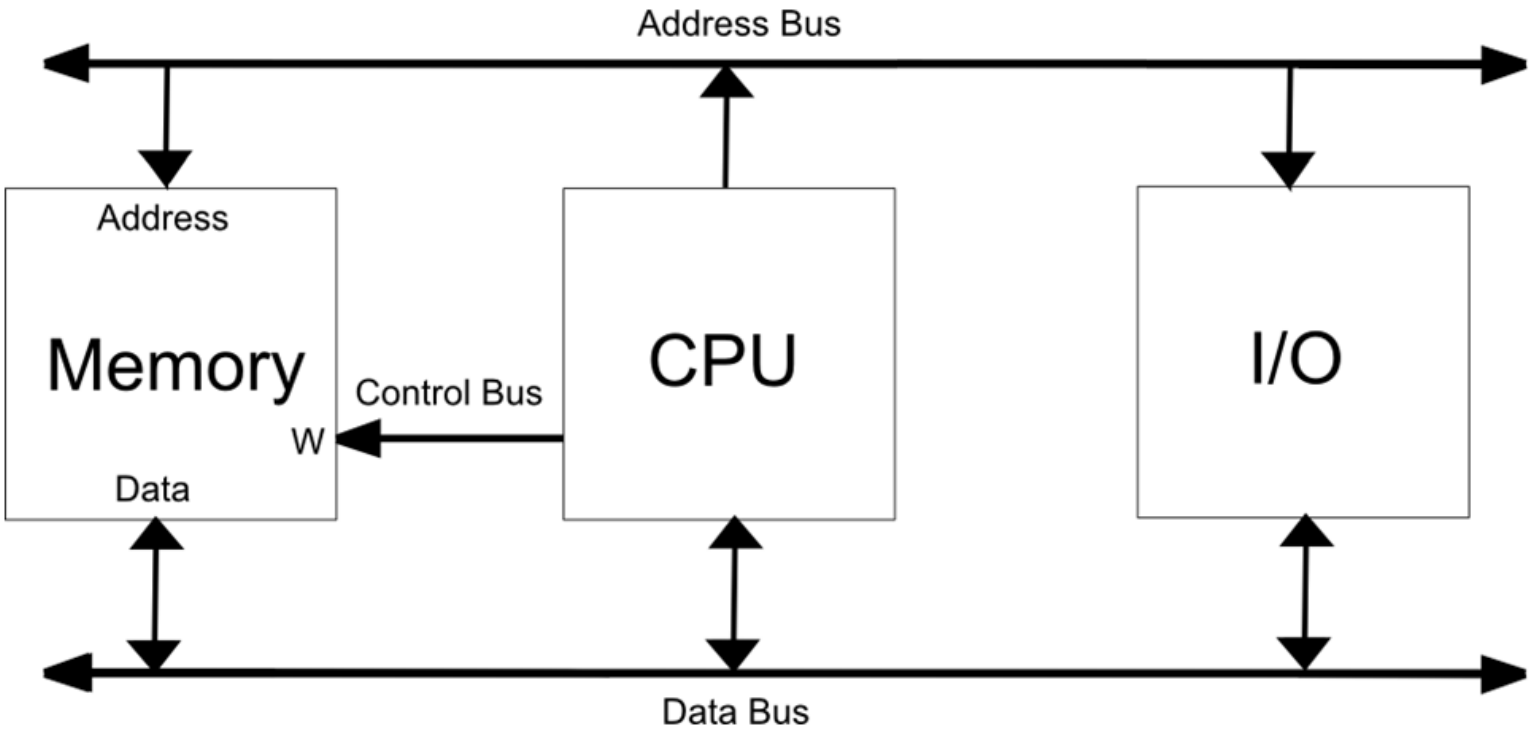
Profesor: Hans Löbel

Dispositivos de I/O se comunican de manera distinta al resto de los elementos de un computador

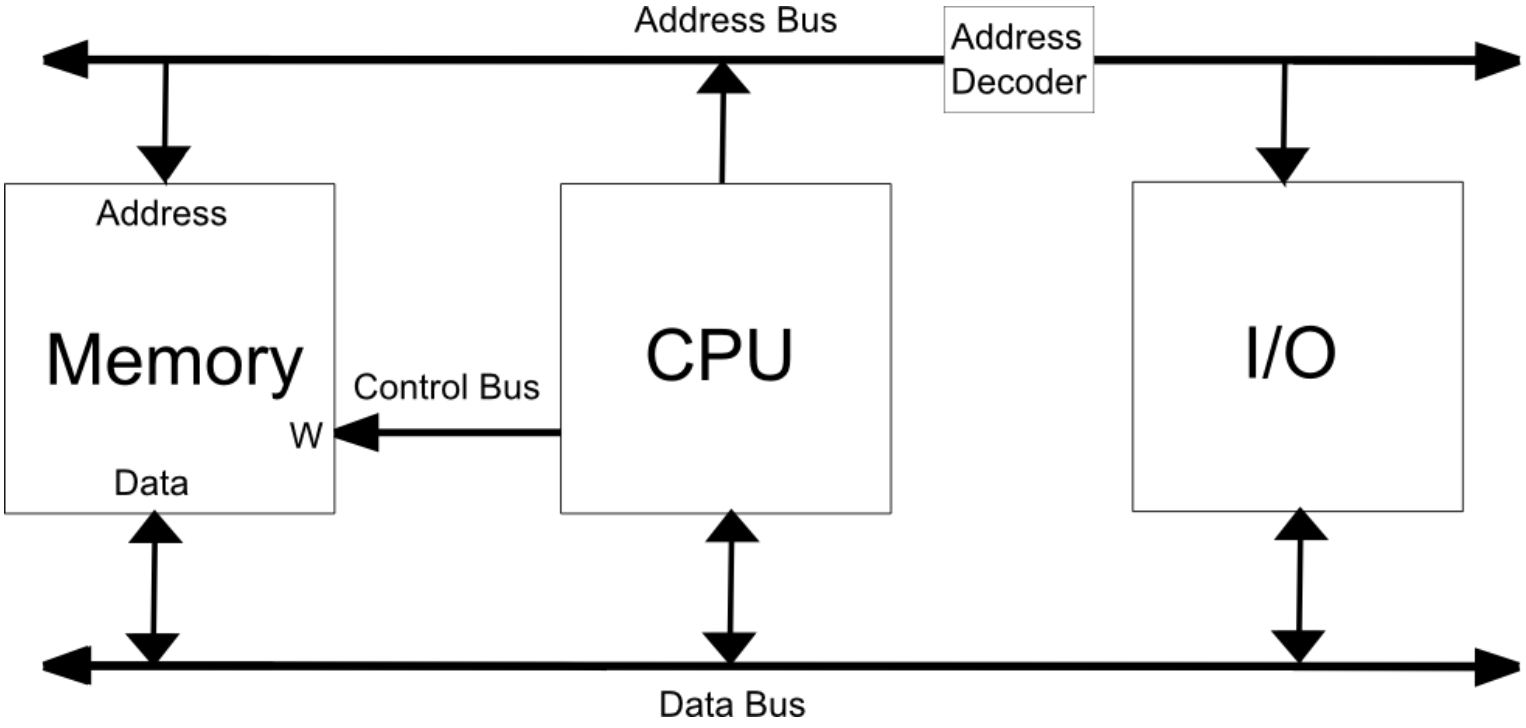
Al no existir señales de control explícitas para los dispositivos de I/O, debemos definir qué tipo de comunicación se llevará a cabo entre CPU, memoria y estos:

1. Comunicación de comandos: CPU -> I/O
2. Comunicación de estado: I/O -> CPU
3. Transferencia de datos: Memoria <-> I/O

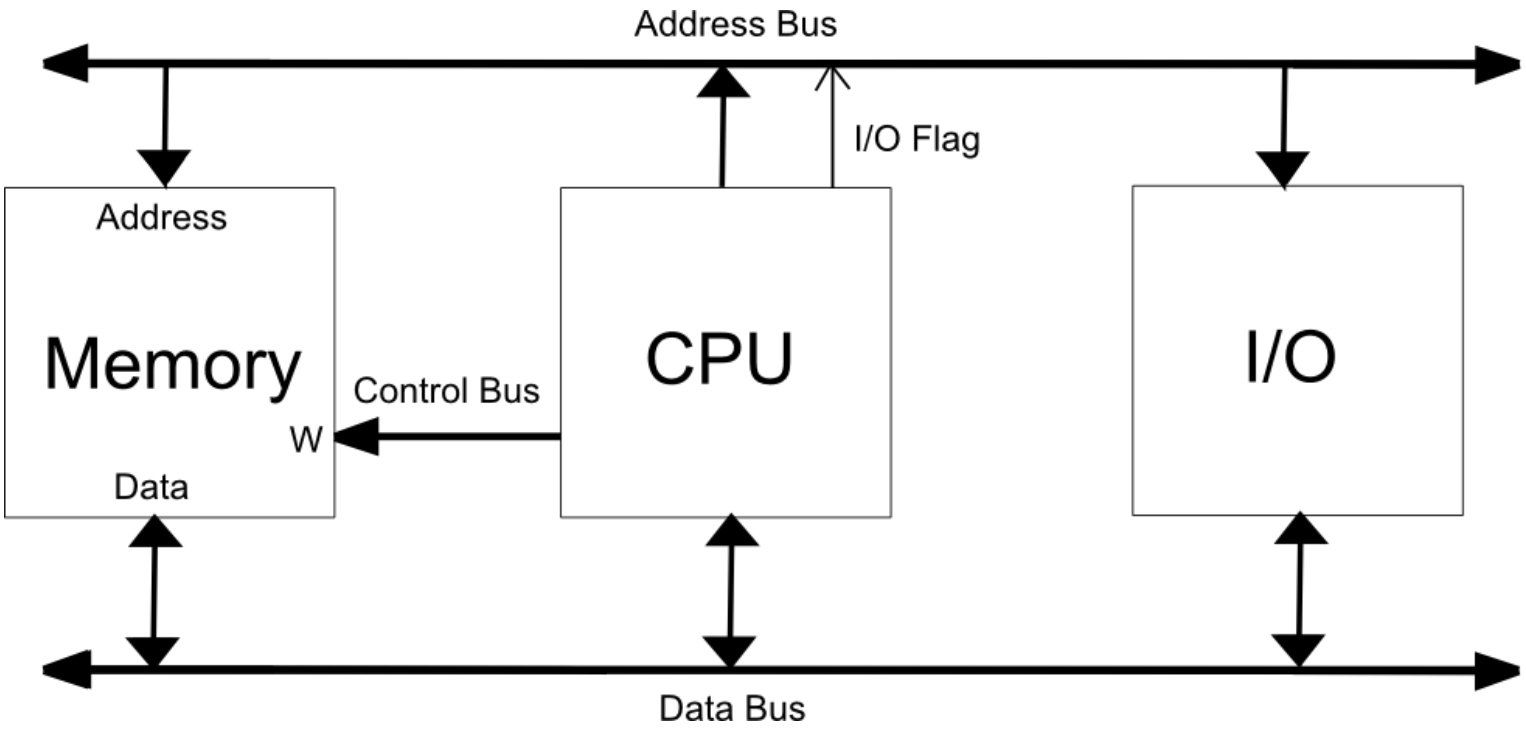
¿Cómo podemos hacer que un programa se comunique con un dispositivo de I/O?



Un programa puede comunicarse con un dispositivo de I/O mediante dos formas: i) memory mapped I/O



Un programa puede comunicarse con un dispositivo de I/O mediante dos formas: i) memory mapped I/O o ii) port I/O



Interacción entre CPU, memoria y dispositivos puede llevarse a situaciones reales

Usaremos como analogía a un curso haciendo una guía de ejercicios:

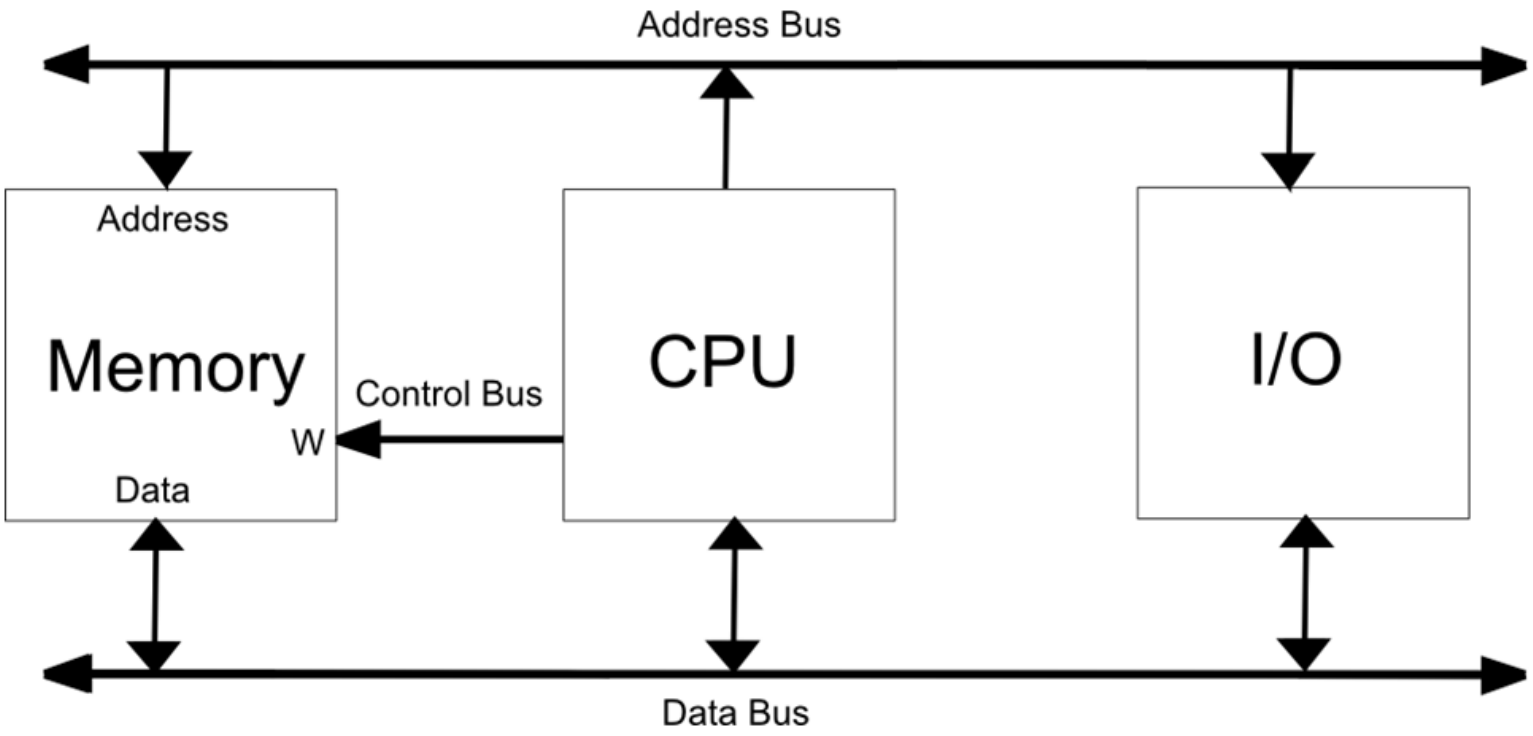
- Alumnos (I/O) hacen guía de ejercicios, si alguien termina, el profesor quiere guardar la respuesta.
- Profesor (CPU) está corrigiendo pruebas.
- Pizarrón o proyector (Memoria), donde se pueden ver los ejercicios y anotar las respuestas.
- Comandos: instrucciones de parte del profesor.
- Estado: alumnos trabajando, con dudas.
- Datos: preguntas y respuestas guía, dudas alumnos, respuestas profesor.
- Supuestos: alumnos no hablan entre ellos, se entregan varias guías durante la clase

Revisaremos tres modelos de interacción ente alumnos y profesor

1. Sin proyector ni pizarrón ni copias de las guías, alumnos tímidos:

   Polling

¿Necesitamos nuevo HW para el esquema de polling?
(sin contar Address Decoder e I/O Flag)



Revisaremos tres modelos de interacción ente alumnos y profesor

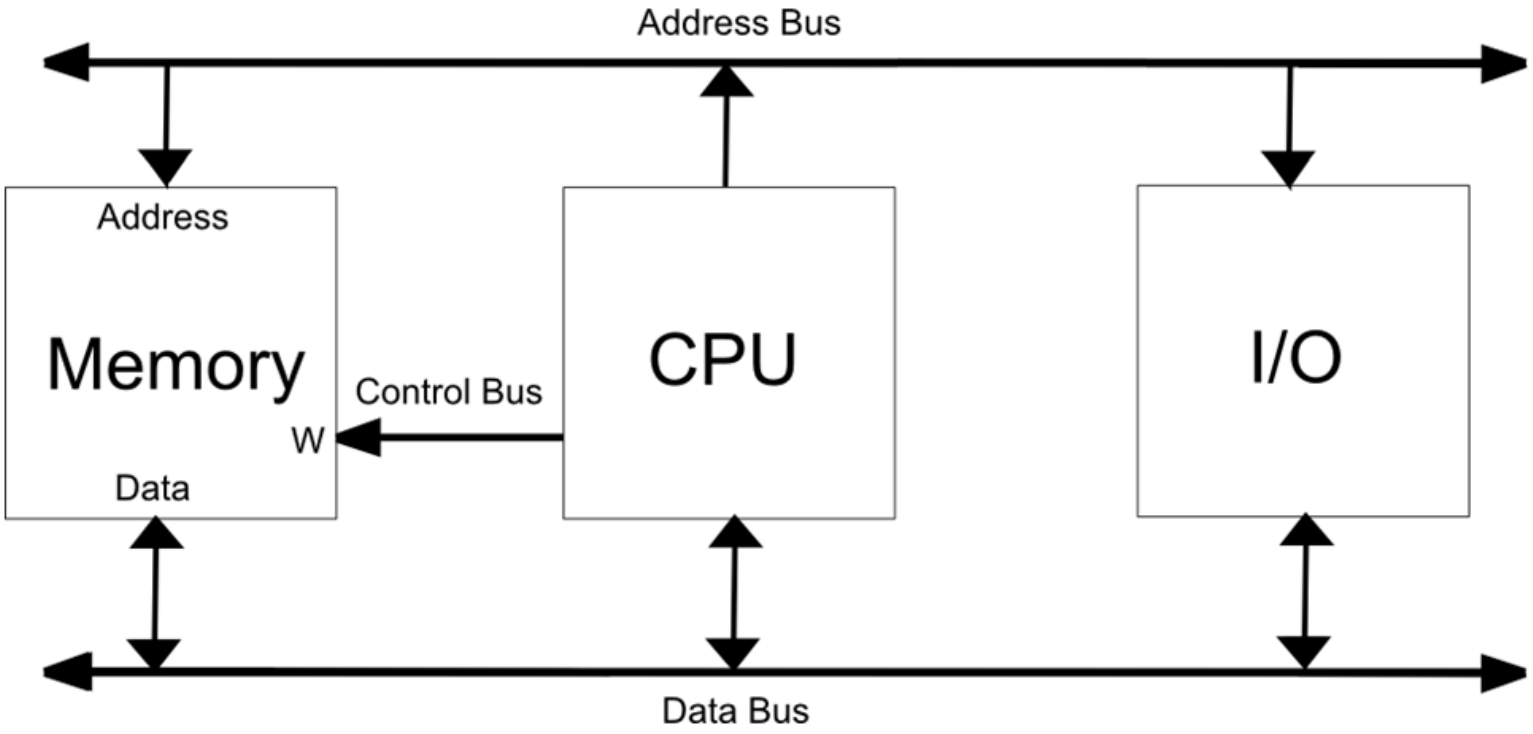
1. Sin proyector ni pizarrón ni copias de las guías, alumnos tímidos:

   Polling

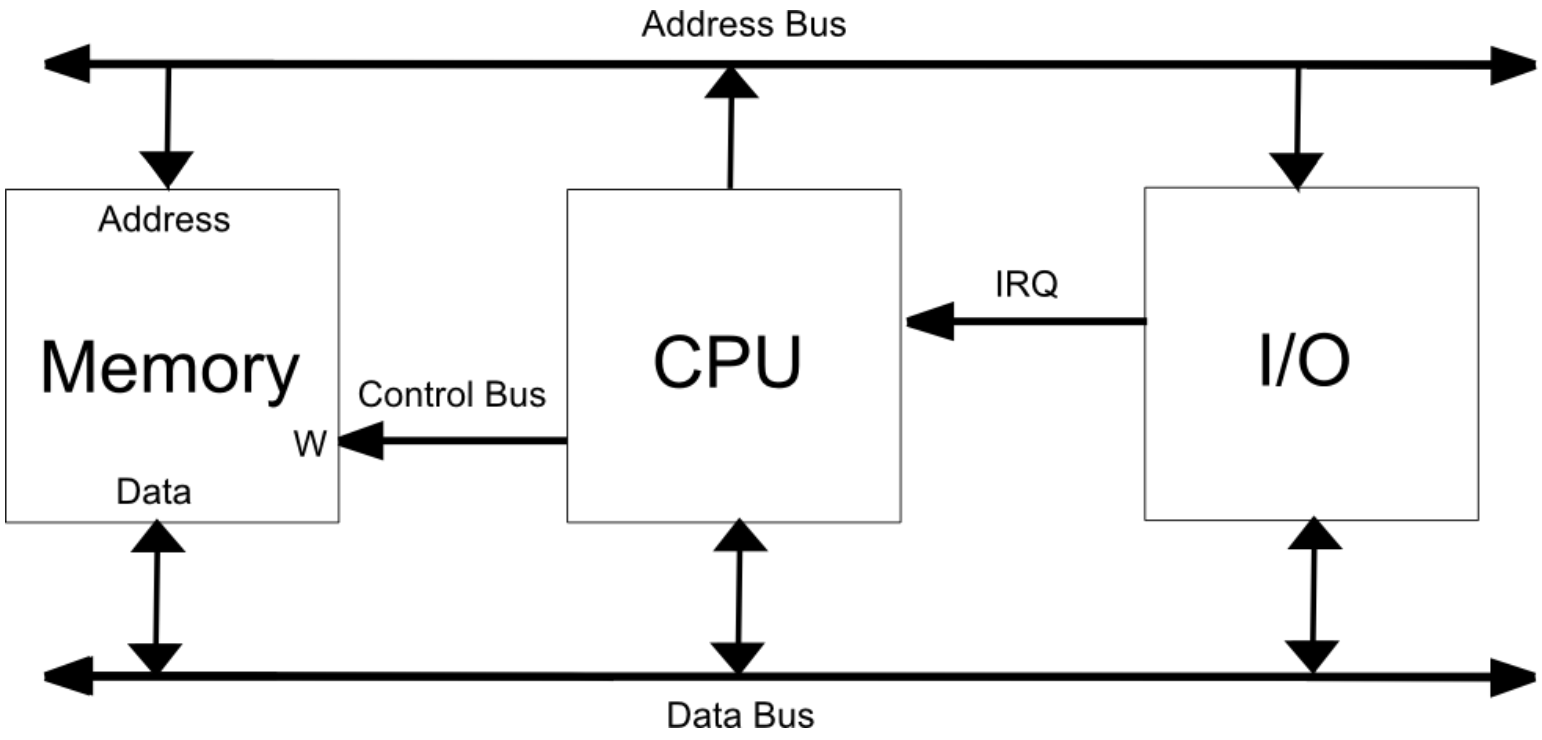
2. Sin proyector ni pizarrón ni copias de las guías, alumnos preguntones y participativos:

   Interrupción

¿Cómo podemos implementar el esquema de interrupción?



Agregamos posibilidad de que los dispositivos interrumpan a la CPU mediante una señal directa de solicitud de interrupción (IRQ)

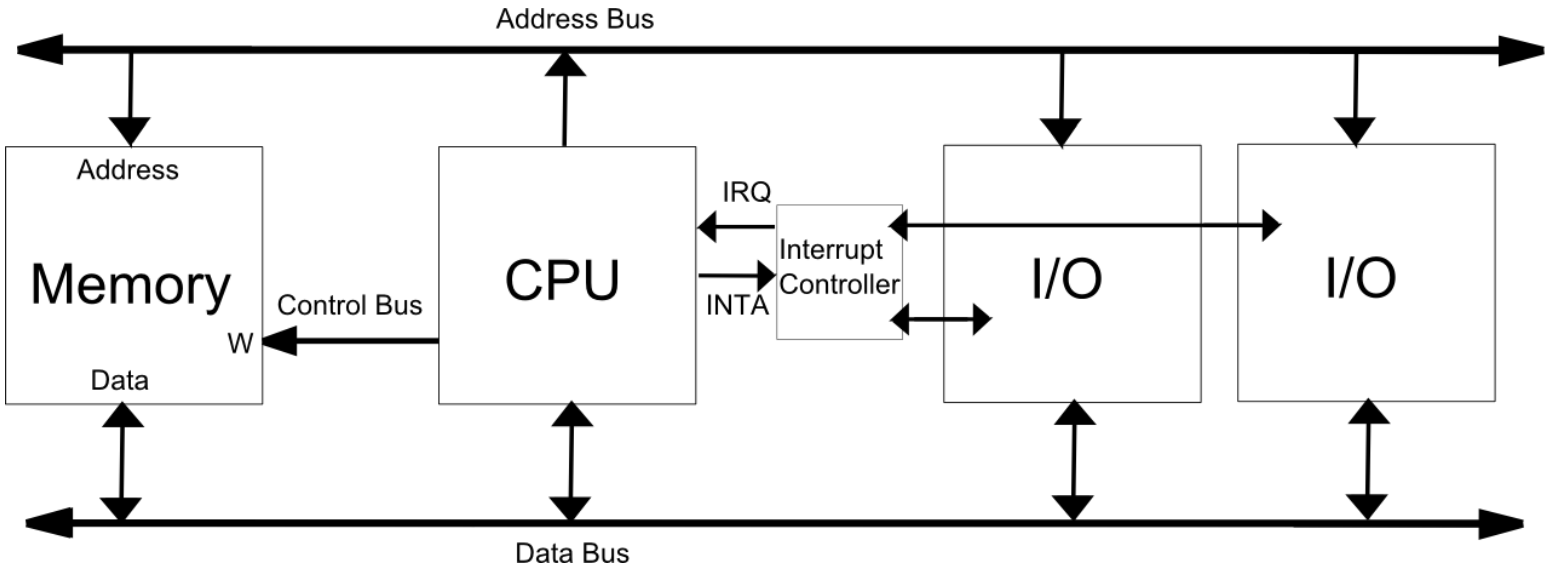


Dispositivo es atendido mediante una *Interrupt Service Routine* (ISR)

Revisemos qué pasa cuando se genera una solicitud de interrupción

1. Dispositivo solicita interrumpir, enviando señal IRQ
2. CPU termina de ejecutar la instrucción actual y guarda *condition codes* en el stack
3. CPU revisa si el *flag* de interrupciones está activo (IF = 1). En caso contrario, saltar al paso 11.
4. CPU deshabilita atención de más interrupciones (IF=0)
5. CPU llama a la ISR asociada al dispositivo
6. ISR respalda estado actual de la CPU
7. ISR es ejecutada
8. ISR restaura estado de la CPU
9. ISR retorna
10. CPU habilita la atención de interrupciones (IF=1)
11. CPU recupera *condition codes* desde el stack

Al usar varios dispositivos, es necesario incorporar un controlador de interrupciones y una tabla de vectores de interrupción



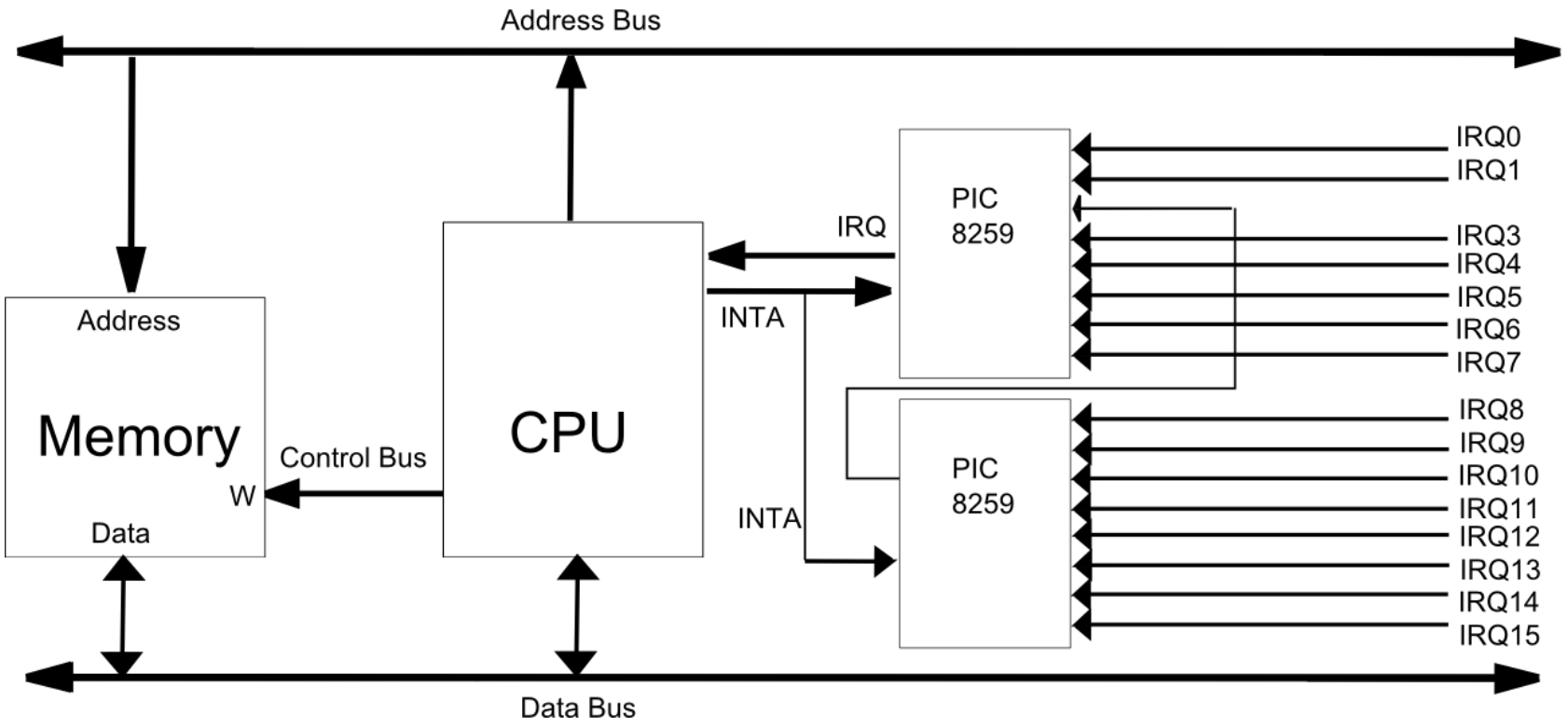
Al usar varios dispositivos, es necesario incorporar un controlador de interrupciones y una tabla de vectores de interrupción

En general, un controlador de interrupciones tendrá al menos los siguientes componentes:

- Registro de comandos y estado
- Registro de interrupciones en espera de atención
- Registro de interrupciones en atención
- Registro de enmascaramiento de interrupciones
- Circuito para manejar prioridades de interrupciones

¿Cómo cambian con esto los pasos para atender una interrupción?

La arquitectura x86 de 16 bits tiene 2 controladores de interrupciones llamados PIC, los cuales se conectan en cascada



A cada señal de IRQ se le asocia un dispositivo específico y una posición en la tabla de vectores de interrupción

IRQ	Dispositivo	Vector de interrupción
IRQ0	Timer del sistema	08
IRQ1	Puerto PS/2: Teclado	09
IRQ2	Conectada al PIC esclavo	0A
IRQ3	Puerto serial	0B
IRQ4	Puerto serial	0C
IRQ5	Puerto paralelo	0D
IRQ6	Floppy disk	0E
IRQ7	Puerto paralelo	0F
IRQ8	Real time clock (RTC)	70
IRQ9-11	No tienen asociación estándar, libre uso.	71-73
IRQ12	Puerto PS/2: Mouse	74
IRQ13	Coprocesador matemático	75
IRQ14	Controlador de disco 1	76
IRQ15	Controlador de disco 2	77

Además de **interrupciones de hardware**, existen las excepciones y las interrupciones de software (traps)

Dirección del vector	Tipo	Función asociada
00-01	Excepción	Excpetion handlers
02	Excepción	Usada para errores críticos del sistema, no enmascara
03-07	Excepción	Excpetion handlers
08	IRQ0	Timer del sistema
09	IRQ1	Puerto PS/2: Teclado
0A	IRQ2	Conectada al PIC esclavo
0B	IRQ3	Puerto serial
0C	IRQ4	Puerto serial
0D	IRQ5	Puerto paralelo
0E	IRQ6	Floppy disk
0F	IRQ7	Puerto paralelo
10	Int. de Software	Funciones de video
11-6F	Int. de Software	Funciones varias
70	IRQ8	Real time clock (RTC)
71 - 73	IRQ9-11	No tienen asociación estándar, libre uso
74	IRQ12	Puerto PS/2: Mouse
75	IRQ13	Coprocesador matemático
76	IRQ14	Controlador de disco 1
77	IRQ15	Controlador de disco 2
78-FF	Int. de Software	Funciones varias

Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC2343 – Arquitectura de Computadores

Comunicación de CPU y Memoria con I/O: Comunicación e Interacción - Parte 1

Profesor: Hans Löbel