**Entrada/Salida (E/S) en Computadoras**

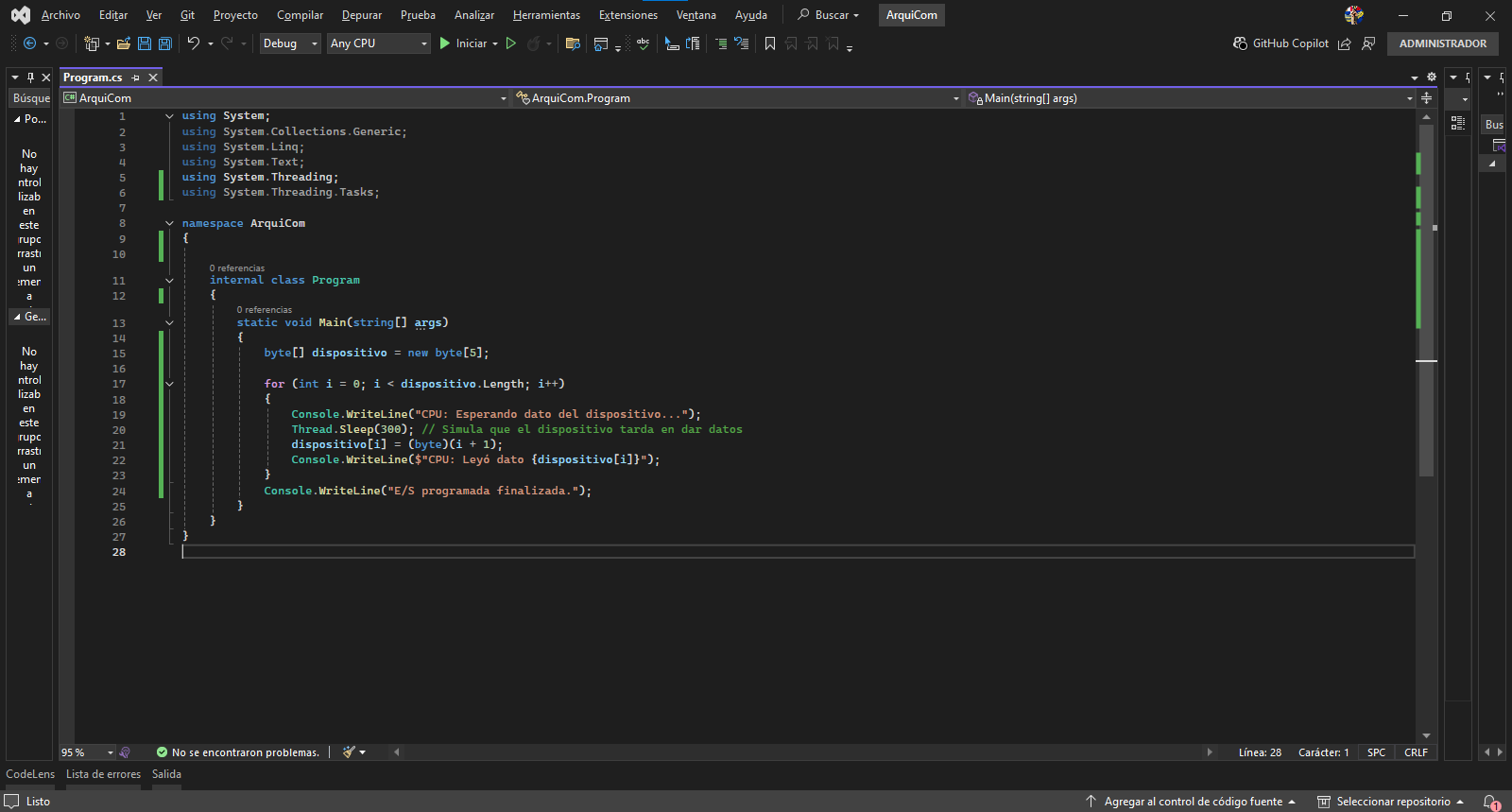
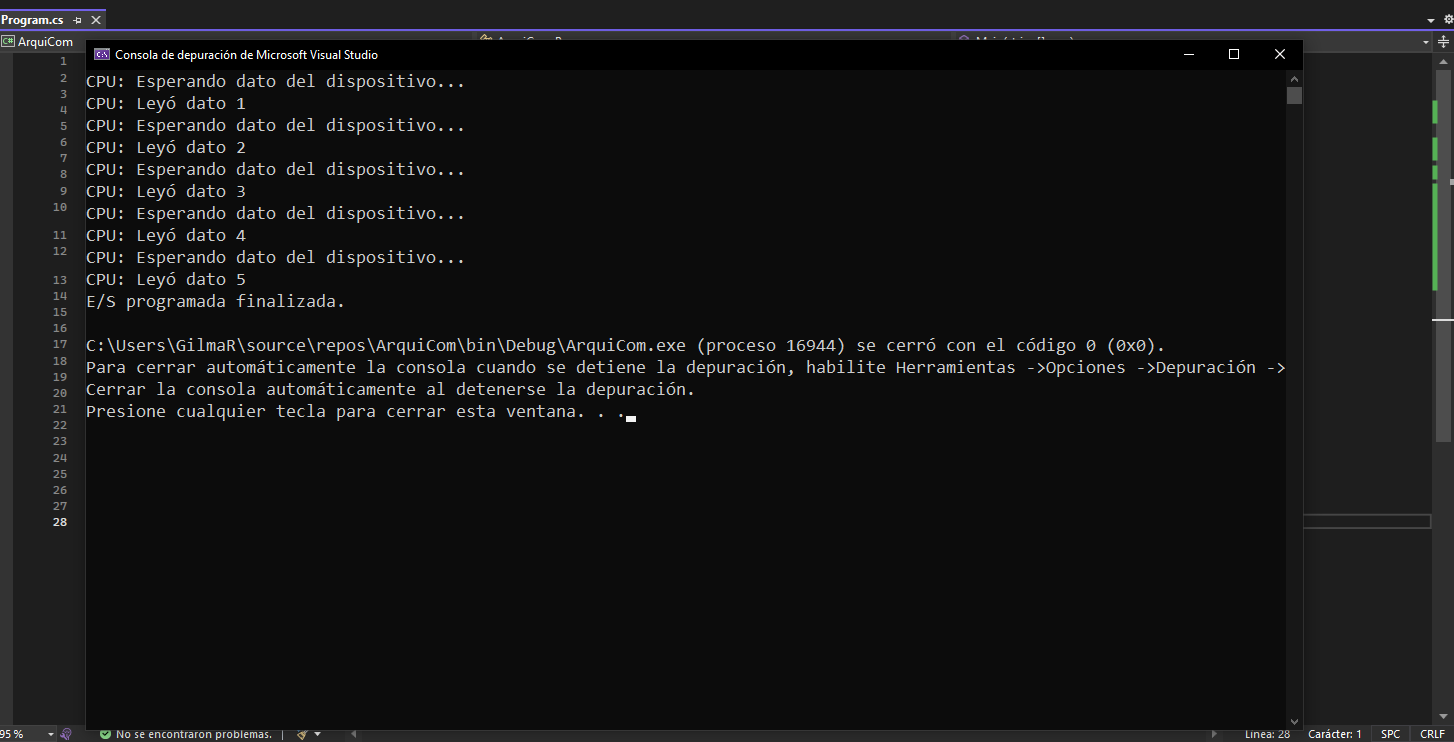
* PROCEDIMIENTOS

Operación de Entrada/Salida:

Implementar ejemplos de E/S programada y E/S mediante interrupciones usando simuladores.

Analizar el flujo de datos mediante DMA con un ejemplo práctico.

1. **Ejemplo de E/S Programada (Polling)**

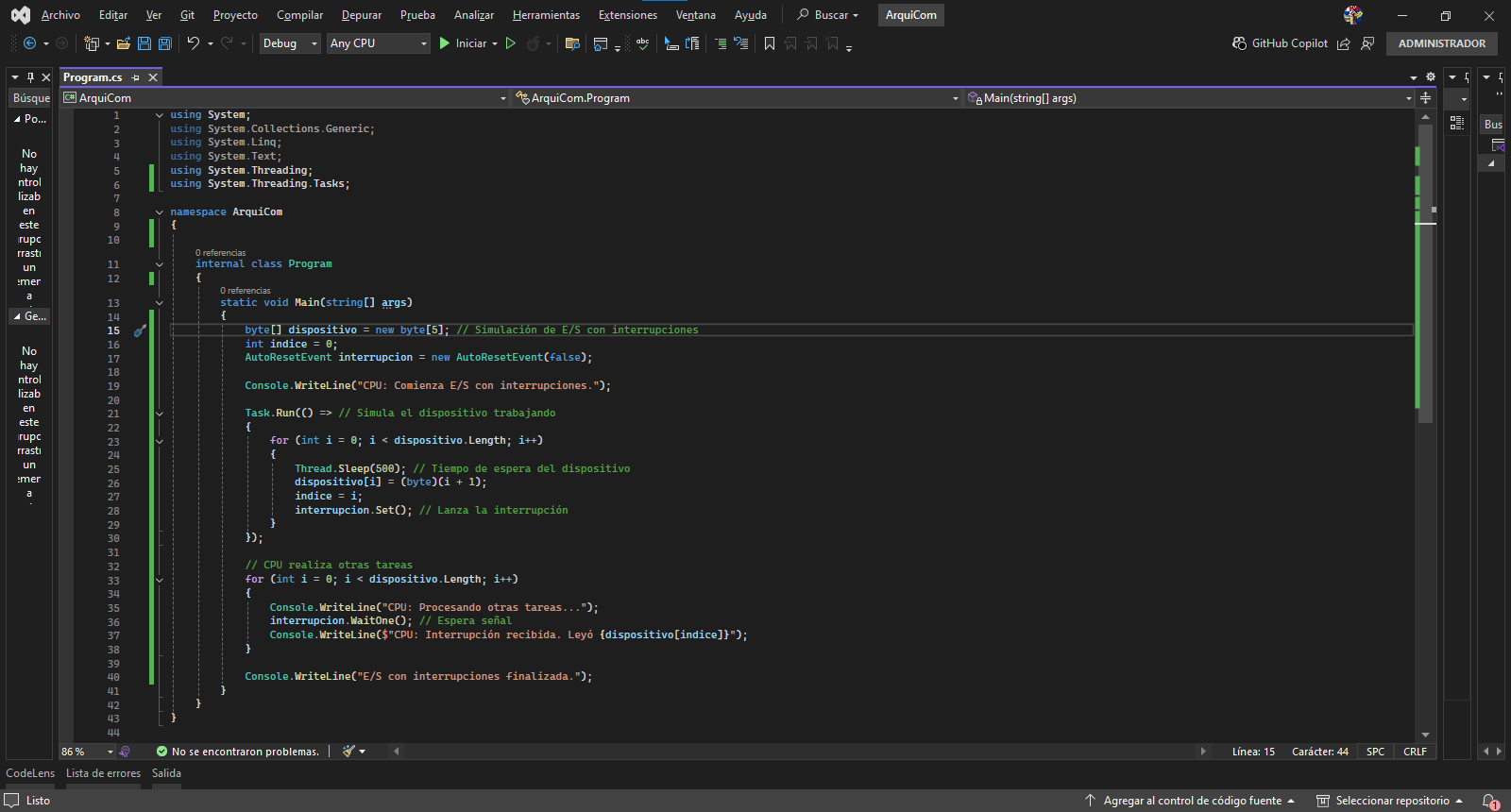
En este método, la CPU consulta repetidamente el estado del dispositivo hasta que haya datos listos. Esto consume tiempo de CPU.

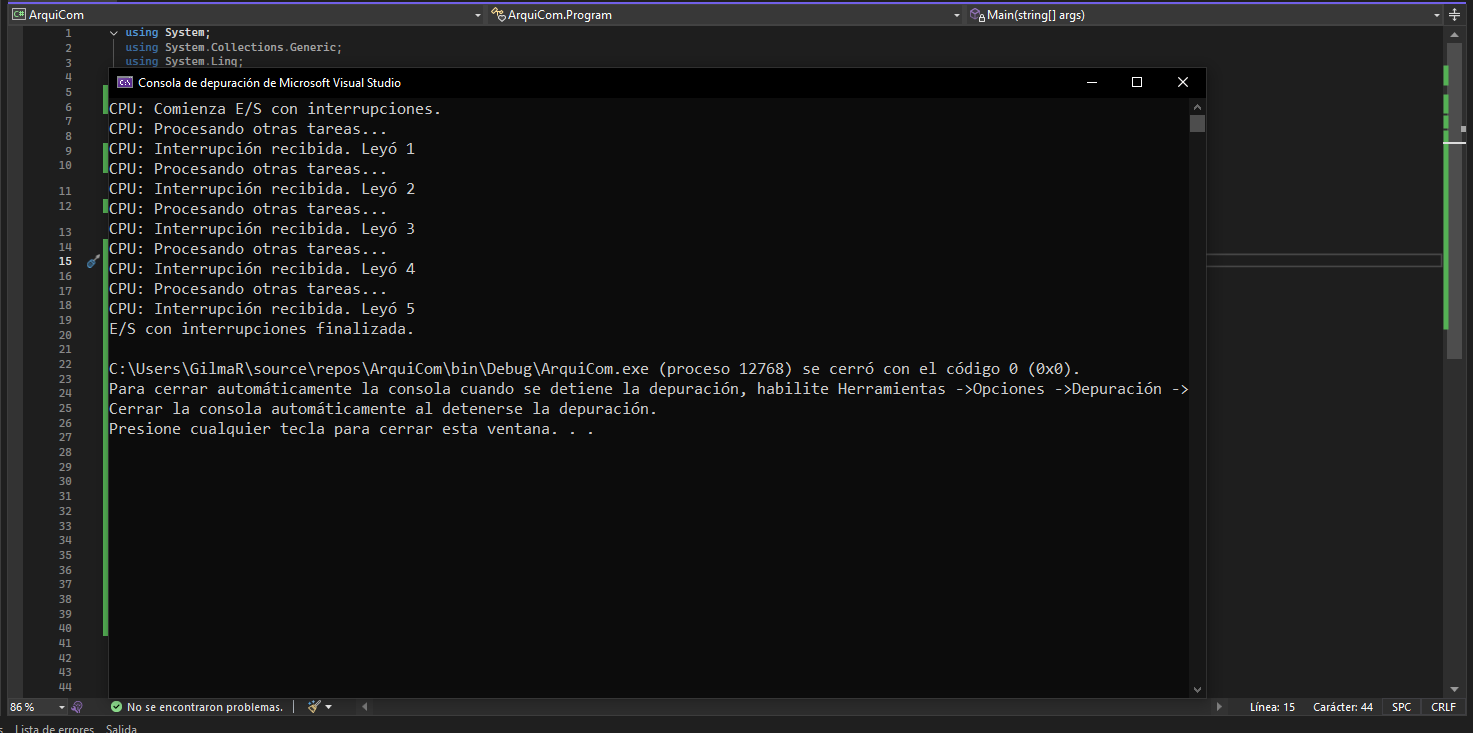
**Explicación:**

Este programa simula cómo la CPU lee datos de un dispositivo usando **E/S programada** (polling). Se crea un arreglo de 5 bytes que representa los datos que la CPU debe recibir. En un ciclo, la CPU espera activamente (con una pausa de 300 ms para simular la demora del dispositivo), luego lee un dato (que se asigna como i + 1) y lo muestra en pantalla. Este proceso se repite hasta leer los 5 datos. Al final, se indica que la operación de E/S programada terminó.

En resumen: el programa muestra cómo la CPU espera y obtiene datos secuencialmente de un dispositivo sin hacer otra cosa mientras espera.

1. **Ejemplo de E/S mediante Interrupciones**

Aquí la CPU puede realizar otras tareas mientras espera que el dispositivo “avise” que tiene datos listos.



**Explicación:**

Este programa simula la comunicación entre un dispositivo y la CPU usando interrupciones en vez de E/S programada. Aquí la CPU no está constantemente esperando, sino que realiza otras tareas hasta que el dispositivo le avisa que hay datos listos (interrupción).

1. Se crea un arreglo dispositivo de 5 bytes para almacenar datos que el dispositivo "envía".
2. Se usa un objeto AutoResetEvent llamado interrupcion, que sirve para que la CPU espere una señal
3. Se inicia una tarea en paralelo (Task.Run) que simula el trabajo del dispositivo:

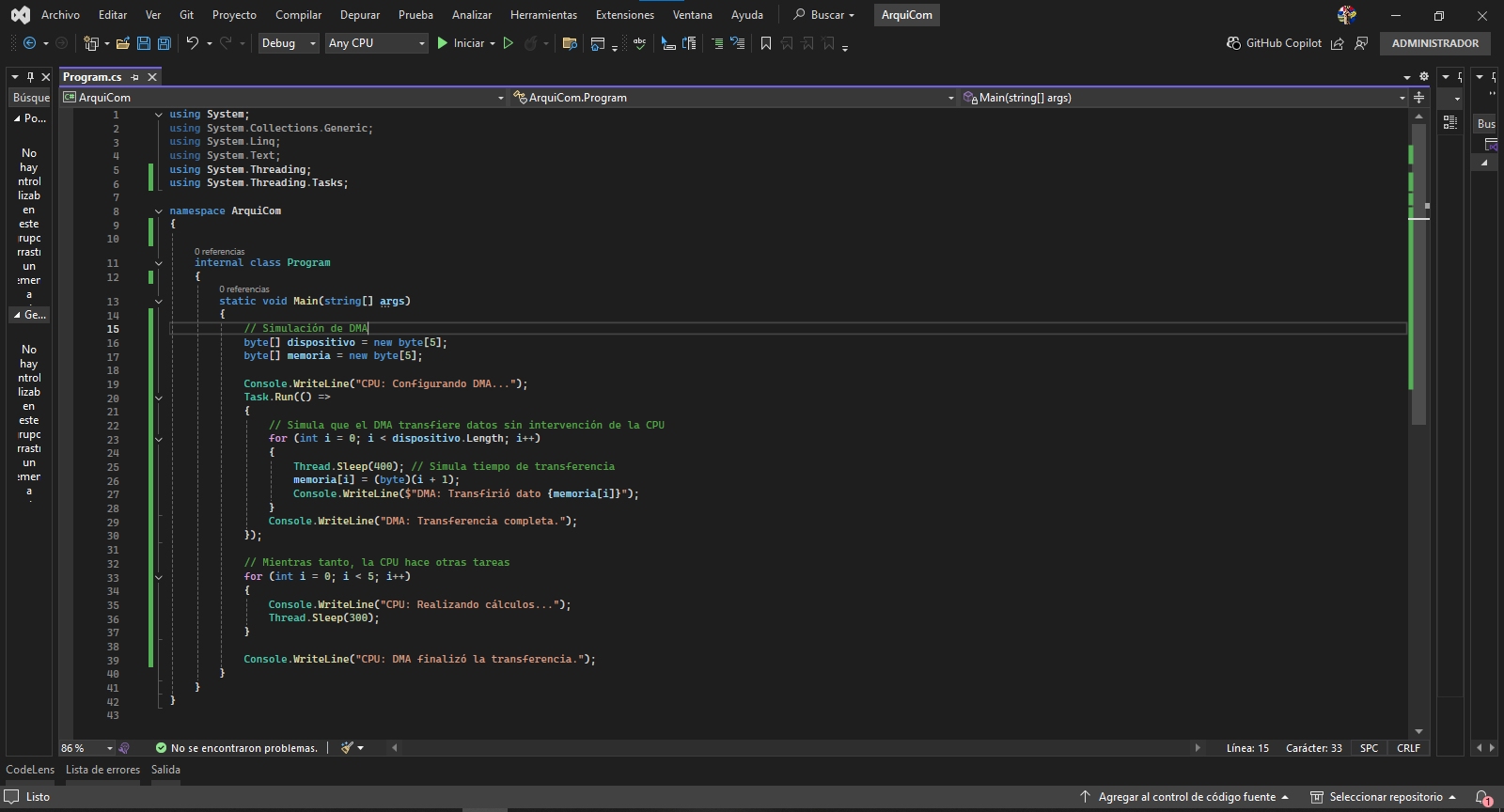
* Cada 500 ms, el dispositivo "genera" un dato (valores 1, 2, 3, 4 y 5).
* Cuando el dato está listo, actualiza la posición actual índice y llama a interrupcion.Set() para notificar a la CPU (simula la interrupción).

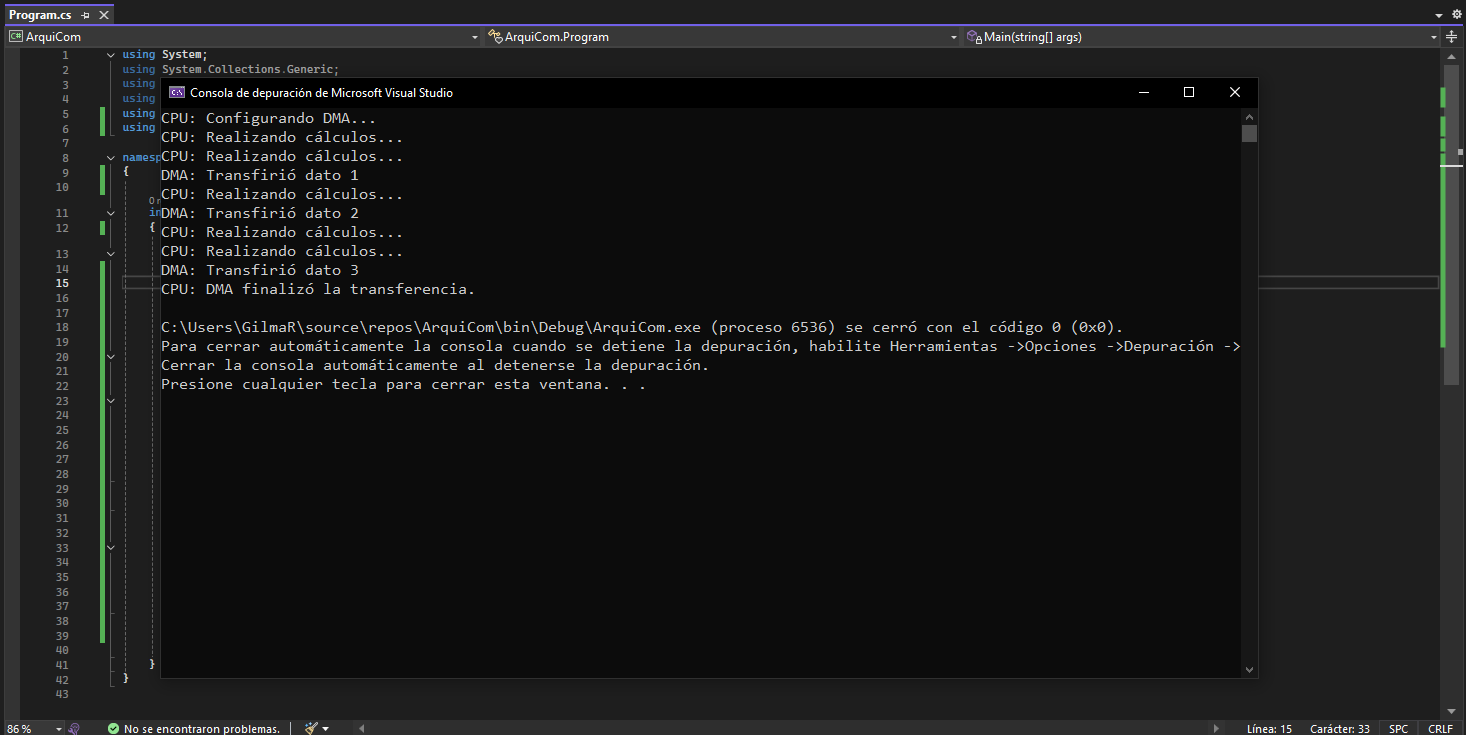
1. Mientras tanto, en el hilo principal (CPU), se ejecuta un ciclo donde:

* La CPU dice que está haciendo otras tareas (simulación).
* Luego llama a interrupcion.WaitOne(), que hace que espere hasta recibir la señal de interrupción.
* Cuando recibe la señal, lee el dato actual del arreglo y lo muestra.

1. Al final, se imprime que la operación de E/S con interrupciones ha terminado.
2. **Ejemplo de DMA (Acceso Directo a Memoria)**

En DMA, la CPU solo inicia la transferencia y un controlador especializado mueve los datos directamente a la memoria.





**Explicación:**

Este programa simula cómo un controlador DMA transfiere datos directamente desde un dispositivo a la memoria sin intervención activa de la CPU. Mientras el DMA transfiere, la CPU puede hacer otras tareas sin bloquearse.

1. Se crean dos arreglos de bytes:

* dispositivo (simula los datos que vienen del dispositivo),
* memoria (simula la memoria principal donde se copiarán los datos).

1. La CPU "configura" el DMA (simulado con un mensaje).
2. Se lanza una tarea en paralelo (Task.Run) que representa al controlador DMA:

* Recorre los 5 datos,
* Simula un tiempo de transferencia de 400 ms por dato,
* Copia cada dato al arreglo memoria (aquí simplemente asigna valores de 1 a 5),
* Muestra en consola qué dato transfirió.

1. Mientras el DMA está transfiriendo, en el hilo principal la CPU realiza otras tareas, simuladas con un ciclo que imprime "Realizando cálculos..." y espera 300 ms en cada iteración.
2. Al terminar la tarea DMA y las tareas de la CPU, se muestra un mensaje que la transferencia ha finalizado.