

Modelado de protocolo MESI y MOESI (App)

v. 2023-1

Yordi Brenes, Fátima Leiva, Brian Wagemans,
Andrey Zúñiga

October 7, 2023



Introduction

La app de modelado de protocolos de coherencia de caché MESI y MOESI, es una aplicación web que simula el comportamiento de un sistema multiprocesador de tres unidades de procesamiento (PE) con caché privada que a su vez, se conectan a una memoria compartida. El modelo simula el funcionamiento del protocolo de coherencia de caché MESI y MOESI, utiliza una política de escritura write-back.

Contents

1	Instalar el simulador	3
1.1	Software necesario	3
1.2	Clonar repositorio	3
1.3	Correr backend	3
1.4	Correr frontend	4
2	Sobre el simulador	5
2.0.1	Interfaz general	5
2.1	Instrucciones de uso	5
2.1.1	Selección de protocolo	5
2.1.2	Generar el código	6
2.1.3	Correr Simulación	6
2.1.4	Ver reporte	6
3	Información de contacto	6

1 Instalar el simulador

1.1 Software necesario

Para correr el simulador deberá tener los siguiente programas instalados. Si no tiene algún programa, siéntase libre de descargarlos utilizando los hipervínculos.

1. dotnet v6
2. node.js
3. angular
4. git

1.2 Clonar repositorio

1. Antes de comenzar, asegúrese de que tiene Git instalado en tu sistema.
2. Abra su terminal o línea de comandos.
3. Navegue al directorio en su sistema donde desee clonar el repositorio.
4. Ejecute el comando para clonar el repositorio utilizando el url del repositorio (https://github.com/Yorbre25/Proyecto_I_Arqui_II.git)

```
> git clone https://github.com/Yorbre25/Proyecto_I_Arqui_II.git
```

1.3 Correr backend

1. Navegue hasta el directorio raíz del proyecto que clonó en el paso anterior.
2. Desde la raíz del proyecto, diríjase a *Backend > Proyecto_Arqui*.
3. Una vez que se encuentre en el directorio adecuado, ejecute el siguiente comando para compilar el proyecto.

```
C:\Backend\Proyecto_Arqui> dotnet build
```

4. Después de que la compilación sea exitosa, puede ejecutar el servidor utilizando el siguiente comando. Esto iniciará el servidor y mostrará la salida en la terminal, incluyendo información sobre el puerto en el que está escuchando el servidor.

```
C:\Backend\Proyecto_Arqui> dotnet run
```

- Si el servidor se inicia correctamente, debería ver un mensaje que indica que el servidor está en funcionamiento y escuchando en un puerto específico.

```
PS C:\Users\yraul\OneDrive\Documentos\Github\Proyecto_I_Arqui_II\Backend\Proyecto_Arqui> dotnet run
Compilando...
info: Microsoft.Hosting.Lifetime[14]
      Now listening on: https://localhost:7158
info: Microsoft.Hosting.Lifetime[14]
      Now listening on: http://localhost:5158
info: Microsoft.Hosting.Lifetime[0]
      Application started. Press Ctrl+C to shut down.
info: Microsoft.Hosting.Lifetime[0]
      Hosting environment: Development
info: Microsoft.Hosting.Lifetime[0]
      Content root path: C:\Users\yraul\OneDrive\Documentos\Github\Proyecto_I_Arqui_II\Backend\Proyecto_Arqui\
```

- Para detener el servidor, simplemente presione Ctrl + C en la terminal donde se está ejecutando el servidor.

1.4 Correr frontend

- Navegue hasta el directorio raíz del proyecto.
- Desde la raíz del proyecto, diríjase a *Frontend* > *Protocol*.
- Una vez que se encuentre en el directorio adecuado, ejecute el siguiente comando para instalar las dependencias del proyecto. Esto descargará todas las dependencias necesarias para su aplicación Angular.

```
I\Frontend> npm install
```

- Después de que la instalación de las dependencias sea exitosa, puede iniciar el servidor de desarrollo de Angular con el siguiente comando.

```
\Frontend\protocols> ng serve --open
```

- Esto iniciará el servidor de desarrollo y mostrará la salida en la terminal, incluyendo información sobre el puerto en el que está ejecutándose.

```
PS C:\TEC\Arqui2\Proyecto_I_Arqui_II\Frontend\protocols> ng serve --open
✓ Browser application bundle generation complete.

Initial Chunk Files | Names          | Raw Size
---
vendor.js           | vendor         | 3.71 MB
polyfills.js        | polyfills      | 333.17 kB
styles.css, styles.js | styles        | 314.83 kB
main.js             | main          | 54.43 kB
runtime.js          | runtime        | 6.52 kB
Initial Total       |                | 4.40 MB

Build at: 2023-10-07T00:57:43.278Z - Hash: faaeb2d1ec7735cd - Time: 5093ms

** Angular Live Development Server is listening on localhost:4200, open your browser on http://localhost:4200/ **

✓ Compiled successfully.
```

2 Sobre el simulador

2.0.1 Interfaz general

El simulador es una aplicación web en la que se puede generar código, correr el código paso a paso y observar el valor, el estado y las transiciones de los datos en caché y memoria. Al final de la simulación se verá un reporte de la cantidad de transacciones realizadas.

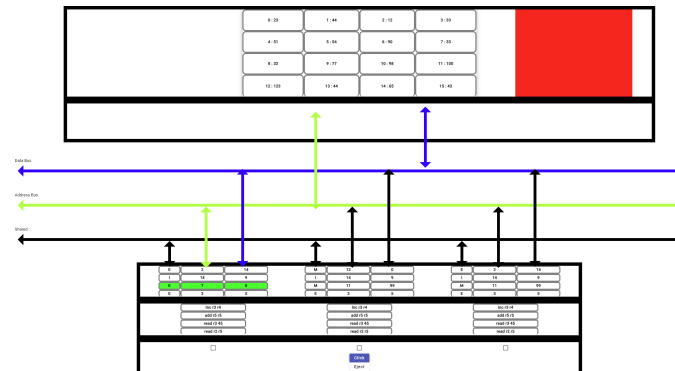


Figure 1: Simulador.

La sección inferior muestra las líneas de caché de cada procesador, mientras que los bloques muestran las instrucciones por procesador. La parte superior muestra la memoria principal. Las principales funciones del simulador son

1. Generar código
2. Correr código
3. Step código
4. Generar un reporte de la ejecución

2.1 Instrucciones de uso

El flujo normal del simulador consiste en

1. Seleccionar un tipo de protocolo
2. Generar el código
3. Correr simulación
4. Ver reporte

2.1.1 Selección de protocolo

Para seleccionar el protocolo marque:

2.1.2 Generar el código

Para generar el código presione:

El código por simular se verá en:

2.1.3 Correr Simulación

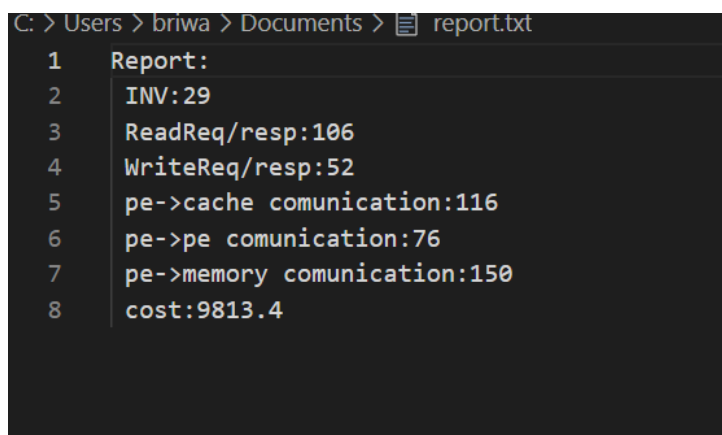
Existen dos formas de correr el código, paso a paso o corrido. La simulación paso a paso permite ir instrucción por instrucción. Para utilizarlo presione:

Por otro lado, la simulación corrida ejecuta todo el código de una vez. Para correr en ese modo presione:

2.1.4 Ver reporte

Al terminar de correr la simulación, un reporte formato .txt se

El reporte se puede ver en la siguiente figura



```
C: > Users > briwa > Documents > report.txt
1 Report:
2 INV:29
3 ReadReq/resp:106
4 WriteReq/resp:52
5 pe->cache communication:116
6 pe->pe communication:76
7 pe->memory communication:150
8 cost:9813.4
```

3 Información de contacto

Si tiene preguntas, o quiere reportar un bug contactet a:

1. Yordi Brenes: ybrenesr@estudiantec.cr
2. Fátima Leiva: 2019153089@estudiantec.cr
3. Brian Wagemans: briwag88@estudiantec.cr
4. Andrey Zúñiga: lazh@estudiantec.cr