	Universidad de Costa Rica Escuela de ingeniería Eléctrica Experimento 5	EIE Escuela de Ingeniería Eléctrica
IE0424: Laboratorio de Circuitos Digitales II-2020		

Objetivo General

Aplicar los conceptos de arquitectura de computadoras para el diseño de jerarquías de memoria.

Objetivos específicos

- Analizar interfaces de memoria de un CPU e implementar un cache sencillo.
- Recolectar y analizar datos de rendimiento de un cache.

Preguntas a responder en el anteproyecto.

- Documente qué significa cada una de las señales que forman la interfaz de memoria de picorv32.
- ¿Qué es el miss/hit rate de un cache?
- ¿Qué es un bloque? ¿Cómo se identifica un bloque dentro de un cache?
- ¿Cuál es la diferencia entre write-back y write-through?

Propuesta del problema.

Proceda ahora a obtener el código inicial del proyecto. Para ello, siga el siguiente enlace:

```
https://classroom.github.com/g/30mql6Bg
```

De la misma forma que en los Laboratorios anteriores, autorice la aplicación, busque su nombre (si su nombre no aparece, contacte al profesor) y seleccione un equipo (si su equipo no aparece listado, proceda a crear uno). Una vez finalizado un repositorio debió haber sido creado. Este repositorio va a ser el repositorio con el que va a trabajar y va a ser uno de los entregables de este laboratorio.

Proceda a clonar el repositorio que ha sido creado:

```
$ git clone https://github.com/ucr-ie-0424/...
```

La dirección completa de su repositorio debe aparecer seleccionando el botón verde que dice *Code.*

El código que se provee es el mismo código que se usó como punto de inicio de los Laboratorios anteriores.

El objetivo de este experimento es diseñar un cache de mapeo directo y estrategia de escritura de writeback.

Ejercicio 1 (Obligatorio):

Incremente el tamaño de la memoria en el módulo *system.v* a 64 kB.

Escriba un firmware que utilice todos los últimos 48 kB de la memoria para crear una lista enlazada. Use la lista enlazada para guardar números enteros consecutivos a partir de 0, es decir: 0, 1, 2, 3..

La lista enlazada tendrá un formato particular y se almacenará en la memoria del CPU. Para formar la lista, se usarán elementos que contendrán el dato y un puntero al siguiente elemento.

Para ello, las direcciones terminadas en 0x0 o 0x8 serán los punteros a los siguientes elementos, mientras que las direcciones terminadas en 0x4 o 0xC serán los datos de los elementos de la lista. Por ejemplo, los primeros elementos de la lista deberían ser:

Dirección de memoria	**Contenido**	**Elemento**	**Tipo de dato**
0x4000	0x4008	0	Puntero (apunta al elemento 1)
0x4004	0	0	Dato
0x4008	0x4010	1	Puntero (apunta al elemento 2)
0x400C	1	1	Dato
0x4010	0x4018	2	Puntero (apunta al elemento 3)
0x1014	2	2	Dato

El *firmware* debe de crear la lista enlazada de forma tal que cada elemento de la lista tenga como dato el índice del elemento. Por ejemplo, usando el ejemplo de la

tabla anterior, 0x4004 debe contener 0x0 como dato y la dirección 0x400C debe de contener 0x1 como dato.

El firmware debe de ser capaz de recorrer la lista enlazada leyendo los punteros y debe obtener los últimos 5 datos que son impares. Estos datos luego se deben desplegar en los LEDs de 7 segmentos. Llame a este firmware *firmware_lab5_part1.c*.

Agregue en el reporte la cantidad de slices que el diseño consume y documente los últimos 5 datos impares que su firmware encontró.

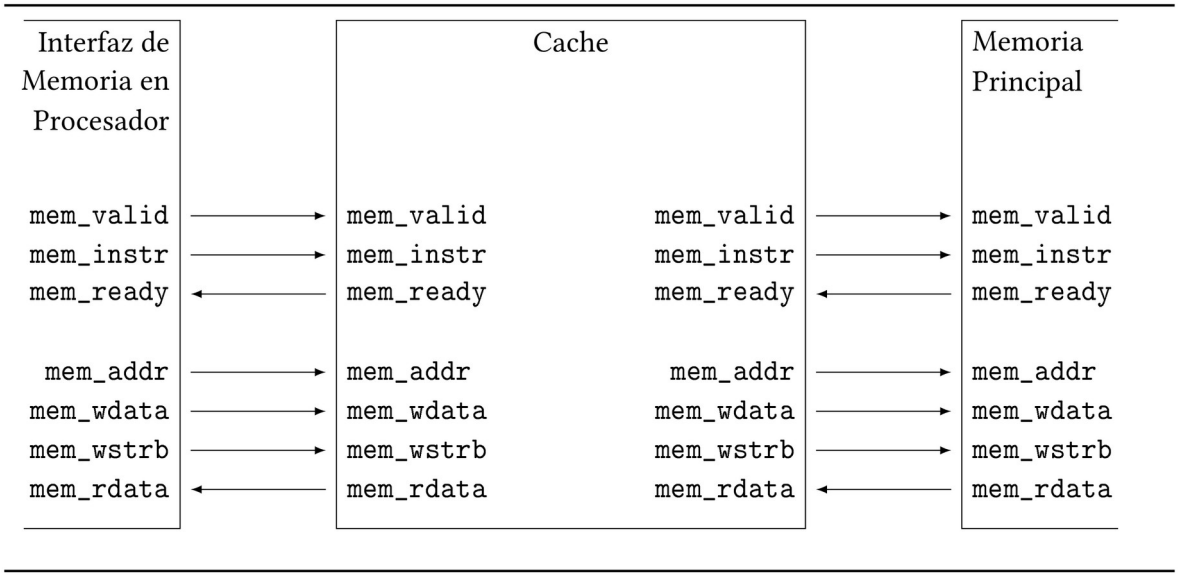
Ejercicio 2 (Obligatorio):

Implemente un cache de mapeo directo y estrategia de escritura write-back.

El diseño de su cache deberá soportar parámetros que definan el tamaño del bloque y el tamaño del cache. Inicialmente utilice un tamaño de bloque de 2 palabras y un tamaño de cache de 1 kB.

Asimismo, deberá tener contadores de hits y misses de forma tal que se pueda luego documentar los rates.

Utilice el siguiente diagrama como referencia para la conexión del cache:



El cache a implementar, debe de estar instanciado en system.v y el módulo debe de tener el nombre *cache_direct*. Asimismo debe de contener las entradas y salidas como se documentaron en el diagrama anterior.

Para este experimento, defina a 0 el parámetro de *FAST_MEMORY* de *system.v*.

Una vez implementado, utilice el firmware del Ejercicio 1 y documente el hit y miss rate para el recorrido completo de la lista enlazada para detectar los últimos 5 datos impares.

Al igual que en el ejercicio anterior, agregue en el reporte la cantidad de slices que el diseño consume y documente los últimos 5 datos impares que su firmware encontró.

Ejercicio 3 (Obligatorio):

Utilice los siguientes valores de tamaño de bloque y tamaño de cache (cambiando los parámetros del diseño del cache del Ejercicio 2):

Tamaño de bloque	Tamaño del cache
2 palabras	2 kB
4 palabras	2 kB
2 palabras	4 kB
4 palabras	4 kB

Asimismo, para cada uno de las entradas de la tabla anterior, utilice el firmware y recorra la lista para encontrar los últimos 5 datos impares. Documente el hit/miss rate y compare los valores obtenidos.

Anote la cantidad de slices que consume el diseño con cada uno de los diferentes parámetros con los que sintetizó su cache.

Ejercicio 4 (Opcional):

Reemplace la memoria instanciada en *system.v* por la memoria DDR2 que la tarjeta Nexys 4 DDR contiene. Para ello consulte el capítulo 4.1 del Manual de Referencia de la tarjeta.