Lab 1: Cache Simulation with Pin

Due Feb 11 by 12pm **Points** 1 **Available** after Feb 1 at 12am

Introduction

The purpose of this assignment is to give insight into:

- How a cache works.
- How different cache designs affect program execution.
- How a program can be tuned for a specific cache configuration.

You will extend a simple cache model and perform experiments with different programs and cache configurations.

You should team up and work in groups of two. Please sign up for a group under **People -> Groups -> Lab Groups.** **You need to be part of a group to be examined for the lab.** This lab assignment is examined in the computer lab slots (via Zoom, links will be published before the lab). During the examination, you will be asked to demonstrate and explain your solutions. Each group member should be able to answer questions for all parts of the lab. We might also ask you additional questions (relevant to the lab material) not found in this document.

There are two lab slots, Lab 1A and Lab 1B. You are only required to attend one of them. Please sign up for which slot you want to attend under **People -> Groups -> Lab 1**. Remember that it is mandatory to pass the lab in order to pass the course, with the exception of the bonus questions, which give you one bonus point.

Zoom link: **https://uu-se.zoom.us/j/69544346205** **(https://uu-se.zoom.us/j/69544346205)**

Lab queue (available during the lab): **https://forms.gle/1cFCgfvNLgQwXMi69** **(https://forms.gle/1cFCgfvNLgQwXMi69)**

TA email: it-avdark-ta@lists.uu.se

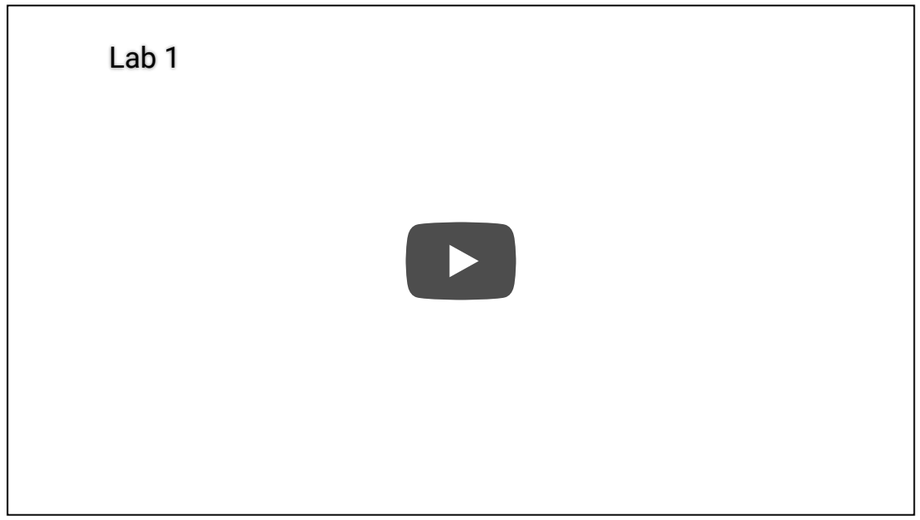
Brief overview of Pin

Pin **(http://en.wikipedia.org/wiki/Pin_(computer_program))** is a dynamic binary instrumentation framework. It allows you to easily insert code at specified parts of the program.

Lets consider the following example. A programmer would like to find out the total number of memory instructions that his application triggers. In this case, it is enough to write a simple C/C++ function that increments a variable and then set up Pin to execute it before or after every memory instruction it detects. Pin will then execute the application's binary and insert your code at the right moment, and by the end of the execution the variable will store the total number of memory instructions triggered. The tool is provided by **Intel** **(http://www.pintool.org/)** and is free to use.

Preparation video

At this point it can be a good idea to watch the preparation video for the lab:



Getting started

Linux servers

In this assignment we will be using the IT-departments linux servers. To login on, connect with *SSH* to *tussilago.it.uu.se* or *vitsippa.it.uu.se*, that is, using:

```
host$ ssh your-user-name@tussilago.it.uu.se.
```

You should now be connected to one of the linux servers.

Note: For compatibility reasons it is important to use the specified servers and not others.

Setting up the environment

First we need to download Pin and install the necessary lab files. To do so, get the `install_lab1.sh` file, and run the script by

```
host$ sh install_lab1.sh
```

It will install files in your home directory, `~/avdark`.

Modifying the cache model

Edit the file `avdark-cache.c` in the source directory (`~/avdark/lab1/avdark-cache`) to modify the cache model. To rebuild the cache model run `host$ make`. If you prefer not to change the working directory use `host$ make -C ~/avdark/lab1/avdark-cache`. Remember to use the test cases to check that your modifications work. There are separate test cases for direct mapped caches (these should *always* work) and associative caches.

The internals of the cache model are fairly simple, most of the code is just Pin *glue code*, and it is only necessary to set up Pin. Your assignment boils down to rearranging the data line array or/and fix the tag and index computations.

All important functions are explained in the source code. It is recommended that you spend some time with the original cache model to get a basic understanding of it before you start to modify it.

The Assignments

Simulating an associative cache

At the moment the cache model is only direct mapped. Modify the cache model so that it can be configured as both a direct mapped (i.e., 1-way) and a 2-way associative data cache. The 2-way associative cache should use the LRU replacement policy. Note that the cache model never handles actual data. The cache model only contains tags and valid bits.

To test the existing direct mapped cache you can use the `pin-avdc.sh` script to launch applications with the simulator. For example:

```
host$ ./pin-avdc.sh -a 2 -s 65536 -l 64 -- ls
```

The parameters before the double dashes (`--`) are passed to Pin and the simulator glue code. The simulator will output its results in `avdc.out` by default. The simulator glue code takes the following parameters:

- `-a ASSOC` Set associativity. Default: `1`
- `-s SIZE` Set cache size. Default: `8388608` B
- `-l BLOCK_SIZE` Set block size. Default: `64` B
- `-o FILE` Set output file. Default: `avdc.out`

Evaluation

- Describe the modifications made to the cache model (by detailing the source code).
- Run 2 or 3 examples with different cache parameters to show those modifications (i.e., run `host$ make test` and check that it works).

Miss Ratio measurements

Test the miss ratios for the RADIX program for the following cache settings:

Size (B)	Block Size (B)	Associativity
16384	32	1
16384	32	2
32768	16	1
32768	32	1
32768	64	1
32768	16	2
32768	32	2
32768	64	2
65536	32	1
65536	32	2

Use RADIX with the `-n 100000` option, that is, run

```
host$ ./pin-avdc.sh -s <SIZE> -l <BLOCK_SIZE> -a <ASSOC> -- ../radix/radix -n 100000
```

on the target machine. This makes RADIX sort 100000 keys. The RADIX binary is located in `~/avdark/lab1/radix`.

Evaluation

- Live-run some examples with different cache parameters.
- Examine and draw some conclusions from the cache-behavior of RADIX.
  Note: You don't need to test all configurations in table above, 6 different configurations should be enough to be able to draw some conclusion. Explain how you selected your cache configurations in that case.
  Tip: It can be easier to spot trends when plotting results in a diagram.