# CMPE283 : Virtualization
# Assignment 1: Discovering VMX Features
**Version 2**

**Due: Mar 18 2019 Before Midnight**

In this assignment you will learn how to discover VMX features present in your processor by writing a Linux kernel module that queries these features. This lab assignment is worth up to 30 points and may be done in groups of up to **two people max.** Each team member can receive up to 30 points. It is expected that groups of more than one student will find an equitable way to distribute the work outlined in this assignment.

## Prerequisites

- You will need a machine capable of running Linux, with VMX virtualization features exposed. You may be able to do this inside a VM, or maybe not, depending on your hardware and software configuration.

## The Assignment

Your assignment is to create a Linux kernel module that will query various MSRs to determine virtualization features available in your CPU. This module will report (via the system message log) the features it discovers.

At a high level, you will need to perform the following:

- Configure a Linux machine, either VM based or on real hardware. You may use any Linux distribution you wish.
- Download and build the Linux kernel source code
- Create a new kernel module with the assignment functionality
- Load (insert) the new module
- Verify proper output in the system message log.

I will be describing which MSRs to read and how to interpret the answers in class, so make sure to take good notes (or you can figure it out yourself via Google and reading the SDM, it's not that hard).

On or before the due date, turn a code listing and answers to the questions below via E-Mail. **Make sure to follow the instructions on diff/code format below.**

**Note:** The instructions below are specific to Intel brand CPUs. If you are using an AMD brand CPU, you should do the lab work in a Google Cloud Platform VM. This restriction applies only to this first assignment.

## Functionality to Implement

You will need to perform the following in your module's main code:

- Determine if your CPU supports VMX true controls
- Based on the above, read various MSRs to ascertain support capabilities/features
  - Entry / Exit / Procbased / Secondary Procbased / Pinbased controls
- For each group of controls above, interpret and output the values read from the MSR to the system via printk(..), including if the value can be set or cleared.

Part of a sample output might look like the output below. Note that this is just a representative sample of what an output could look like, other output formats are accepted provided they are clearly readable and easily understood.

```
true procbased ctls: 0xfff9fffe04006172
    INTERRUPT_WINDOW_EXITING: Can set:Yes Can clear:Yes
    USE_TSC_OFFSETTING: Can set:Yes Can clear:Yes
    HLT_EXITING: Can set:Yes Can clear:Yes
    INVLPG_EXITING: Can set:Yes Can clear:Yes
    MWAIT_EXITING: Can set:Yes Can clear:Yes
    RDPMC_EXITING: Can set:Yes Can clear:Yes
    RDTSC_EXITING: Can set:Yes Can clear:Yes
    CR3_LOAD_EXITING: Can set:Yes Can clear:Yes
    CR3_STORE_EXITING: Can set:Yes Can clear:Yes
    CR8_LOAD_EXITING: Can set:Yes Can clear:Yes
    CR8_STORE_EXITING: Can set:Yes Can clear:Yes
    USE_TPR_SHADOW: Can set:Yes Can clear:Yes
    NMI_WINDOW_EXITING: Can set:Yes Can clear:Yes
    MOV_DR_EXITING: Can set:Yes Can clear:Yes
    UNCONDITIONAL_IO_EXITING: Can set:Yes Can clear:Yes
    USE_IO_BITMAPS: Can set:Yes Can clear:Yes
    MONITOR_TRAP_FLAG: Can set:Yes Can clear:Yes
    USE_MSR_BITMAPS: Can set:Yes Can clear:Yes
    MONITOR_EXITING: Can set:Yes Can clear:Yes
    PAUSE_EXITING: Can set:Yes Can clear:Yes
```

To determine if true controls are available:

- Read the IA32_VMX_BASIC MSR
- Check bit 55 – if set, true controls are available.

To determine if secondary procbased controls are available:

- Check the ability to set “Activate Secondary Controls” control in the primary procbased controls
  - There are no “true secondary procbased controls”, so there is only one MSR to read if the CPU supports secondary controls.

The table below provides some MSRs that may be of interest to you.

| MSR Name | MSR Index | Notes |
|---|---|---|
| IA32_VMX_BASIC | 0x480 | Use this to determine true controls capability (bit 55) |
| IA32_VMX_PINBASED_CTLS | 0x481 | Use this MSR for pinbased controls if no true controls capability |
| IA32_VMX_PROCBASED_CTLS | 0x482 | Use this MSR for procbased controls if no true controls capability |
| IA32_VMX_PROCBASED_CTLS2 | 0x48B | Use this MSR for secondary procbased controls, if available |
| IA32_VMX_EXIT_CTLS | 0x483 | Use this MSR for exit controls if no true controls capability |
| IA32_VMX_ENTRY_CTLS | 0x484 | Use this MSR for entry controls if no true controls capability |
| IA32_VMX_TRUE_PINBASED_CTLS | 0x48D | Use this MSR for pinbased controls if CPU supports true controls |
| IA32_VMX_TRUE_PROCBASED_CTLS | 0x48E | Use this MSR for procbased controls if CPU supports true controls |

| | | |
|---|---|---|
| IA32_VMX_TRUE_EXIT_CTLS | 0x48F | Use this MSR for exit controls if CPU supports true controls |
| IA32_TRUE_ENTRY_CTLS | 0x490 | Use this MSR for entry controls if CPU supports true controls |

## Grading

This assignment will be graded and points awarded based on the following:

- 25 points for the implementation and code producing the output above
- 5 points for the answers to the questions below

Submissions shall be made via email to the email addresses listed in the class syllabus/green sheet. DO NOT WAIT UNTIL LATE ON THE DUE DATE, as email server lags or delays may result in a late submission. Since you have three weeks to complete this assignment, I will not accept "email server outage or delay" as an excuse for late submissions. If you are concerned about this, submit your assignment early. This is one area that I am extremely picky with – even 1 second late will result in a zero score.

**I will be comparing all submissions to ensure no collaboration has taken place. Make sure you do not copy another group's work. If you copy another group's work, members of both groups will receive an F in the class and be reported to the department chair for disciplinary action. If you are working in a group, make sure your partners do not copy another group's work without your knowledge, as all group members will be penalized if cheating is found.**

## Special Notes

I am implementing an automated framework to test these submissions. Therefore, you **must** follow the subsequent submission rules precisely. I will be using a script that will automatically process my mailspool to extract your submissions, and the script will expect the submission email to formatted a specific way, as described below:

- Use a kernel built from the master Linux git repository
  - https://github.com/torvalds/linux.git
  - Record the head commit ID of your tree:
    - For example, if the output of "git log" shows the following:

      ```
      commit 89970a04d70c6c9e5e4492fd4096c0b5630a478c
      Merge: 806276b7f07a 3ea3217cf918
      Author: Linus Torvalds <torvalds@linux-foundation.org>
      Date:   Wed Mar 29 19:59:49 2017 -0700
      ```
    - .. you would record "commit 89970a04d70c6c9e5e4492fd4096c0b5630a478c"
- Use 'git add' and 'git commit' to add your module source file(s) and changes to your local copy of the Linux repository.
- Submit a plain text file containing a unified diff ("git diff" format) diff file containing your entire submission. Name the diff file "cmpe283-1.diff" in your submission when attaching to the email.
- When submitting, submit **only**:
  - A plain text email (no HTML)
  - A single PDF file attachment containing the answers to the questions (the PDF can have whatever name you want)
  - The diff file in plain text format as described above.
  - Your commit ID as calculated above, by including a line starting with "commit" followed by a space, followed by the SHA value of the commit ID, with no other text on that line.
  - A list of student IDs and names for members of the group, in the following format, one per line:
    - ID <id> <name>
      - (example)   ID 0123456789 Larkin, Michael
- The subject line of the email must be "CMPE283 Assignment 1 Submission"  (no quotes)
- Do not submit .zip, .tar, .rar, etc. Send me an email with two attachments, a PDF and a .diff file, as described above.

- Make sure to follow all other instructions in this assignment precisely.
- **Failure to follow the instructions above may result in a zero score for the assignment.**

## Questions

1. For each member in your team, provide 1 paragraph detailing what parts of the lab that member implemented / researched. (You may skip this question if you are doing the lab by yourself).
2. Describe in detail the steps you used to complete the assignment. Consider your reader to be someone skilled in software development but otherwise unfamiliar with the assignment. Good answers to this question will be recipes that someone can follow to reproduce your development steps.
   **Note**: I may decide to follow these instructions for random assignments, so you should make sure they are accurate.