DYNAMIC PAGE RETIREMENT

vR331 | March 2014

TABLE OF CONTENTS

Chapter 1. IMPLEMENTATION

The NVIDIA® driver will decide to retire a page once it has experienced a single Double Bit ECC Error (DBE) or 2 Single Bit ECC Errors (SBE). These addresses are stored in the InfoROM. When the driver loads it will retrieve these addresses from the InfoROM, then have the framebuffer manager set these pages aside, such that they cannot be used by the driver or user applications.

Ideally, the NVIDIA® driver will catch weakening cells at the 2 SBE point and retire the page, before the cell degrades to the point of a DBE and disrupts an application.

A retired page will be stored in the InfoROM for persistence for the life of the board. However, the driver will need to be reloaded for the retirement to take effect. See Caveats for more information.

Chapter 2. AVAILABILITY

Dynamic page retirement is supported on the following products and environments:

- Drivers: R319 and higher
- OSes: All standard driver-supported Linux, Windows and Mac platforms
- GPUs:
  - Tesla K20 and higher products; ECC must be enabled
  - No Quadro, GRID or Geforce products are currently supported

Chapter 3. VISIBILITY

Three main mechanisms provide visibility into page retirement: XID errors in system logs, the NVML API and the nvidia-smi command line tool.

3.1. XIDs

XID errors are driver errors that are logged to the system error log. Please see the XID Whitepaper for general info on XIDs.

There are three main XIDs related to dynamic page retirement:

- XID 48: A DBE has occurred.
- XID 63: A page has successfully been retired.
- XID 64: A page has failed retirement due to an error.

In the system log these XIDs show up in the following forms:

- XID 48: "XID 48 An uncorrectable double bit error (DBE) has been detected on GPU (<id>)"
- XID 63: "XID 63 Dynamic Page Retirement: New retired page, reload the driver to activate. (<address>)"
- XID 64: "XID 64 Dynamic Page Retirement: Fatal error, unable to retire page (<address>)"

3.2. NVML

The NVIDIA® Management Library (NVML) is a public C-based library for GPU monitoring and management. It includes APIs that report the status and count of retired pages. Please see the NVML API docs for general info on the library.

The set of currently retired pages, and their addresses, can be retireved using:

```
nvmlReturn_t nvmlDeviceGetRetiredPages (nvmlDevice_t device,
 nvmlPageRetirementCause_t cause, unsigned int* pageCount, unsigned long long*
 addresses)
```

Where cause is one of NVML_PAGE_RETIREMENT_CAUSE_MULTIPLE_SINGLE_BIT_ECC_ERRORS or NVML_PAGE_RETIREMENT_CAUSE_DOUBLE_BIT_ECC_ERROR

The current state of the driver (whether any pages are pending retirement) can be retrieved using:

```
nvmlReturn_t nvmlDeviceGetRetiredPagesPendingStatus (nvmlDevice_t device,
 nvmlEnableState_t* isPending)
```

3.3. Nvidia-smi

Nvidia-smi is a public command line interface for GPU monitoring and management. It implements most of the NVML APIs and supports reporting the status and count of retired pages. Please see the Nvidia-smi man page for general info on the tool.

To view the number of retired pages and the page retirement state of the driver in human readable form:

```
$ nvidia-smi -i <target gpu> -q -d  PAGE_RETIREMENT
...
Retired pages
              Single Bit ECC             : 2
              Double Bit ECC             : 0
              Pending                    : No
...
```

If pages have been retired the affected addresses can be viewed through nvidia-smi's scriptable outputs, either XML:

```
      $ nvidia-smi -i <target gpu> -q -x
      ...
                <retired_pages>
                        <multiple_single_bit_retirement>
                         <retired_count>2</retired_count>
                                <retired_page_addresses>
                                 <retired_page_addresse>0xABC123</
retired_page_addresse>
                                 <retired_page_addresse>0xDEF456</
retired_page_addresse>
    </retired_page_addresses>
                        </multiple_single_bit_retirement>
                        <double_bit_retirement>
                                <retired_count>0</retired_count>
                                <retired_page_addresses></
retired_page_addresses>
                        </double_bit_retirement>
                        <pending_retirement>No</pending_retirement>
                </retired_pages>
      ...
```

or CSV:

```
      $ nvidia-smi -i <target gpu> --query-retired-
pages=gpu_uuid,retired_pages.address,retired_pages.cause --format=csv
```

```
...
gpu_uuid, retired_pages.address, retired_pages.cause
GPU-d73c8888-9482-7d65-c95c-4b58c7d9eb4c, 0xABC123, Double Bit ECC
GPU-d73c8888-9482-7d65-c95c-4b58c7d9eb4c, 0xDEF456, Double Bit ECC
GPU-d73c8888-9482-7d65-c95c-4b58c7d9eb4c, 0x123ABC, Single Bit ECC
...
```

Chapter 4. CAVEATS

There are currently several page retirement behaviors that should be noted:

- As mentioned above, the driver requires a reload before a page is actively retired. The page will be written to the InfoROM, but cannot be retired until the driver is reloaded. The driver can be reloaded in multiple ways:
  - reboot the system
  - temporarily exit persistence mode and all driver applications (X, NVML, etc)
  - reset the GPU
  - unload and reload the kernel module
- There exists a race condition between logging errors to the InfoROM and ending a CUDA™ job while in persistence mode. This race condition is most often hit when shutting down in response to a DBE. The effect of this condition is that a page may fail to retire in certain corner cases.

  Exiting persistence mode before rebooting the system will forcibly flush any pending writes to the InfoROM. If XID 48 is seen and XID 63 is not seen, it is recommended to exit persistence mode via the command:

  ```
  % nvidia-smi -i <target GPU> -pm 0
  ```

  At this point, the XID 63 should be seen and the NVML query can be used to verify the page was written to the InfoROM.