**Report on Operating System Project:**

**Virtual Memory - Two Level paging and TLB in Pintos**

**Group Members:**

**Harshal Daglia (y13uc109)**

**Parth Bhatnagar (y13uc188)**

**Shubham Gupta (y13uc276)**

**Project related Details about Pintos:**

Pintos’ uses simple paging method which creates Main Memory shortage and increases the lookup time, As OS brings whole Page Table into main memory at same time and Due to which it constrains the Number of Processes running simultaneously.

Our project is to understand the working of pintos and extend the functionality by implementing Two-level Paging and TLB [Translate Lookaside Buffer] in Pintos.

**Main objectives of project:**

1.) Extending the functionality of Paging using Two-Level Paging in Multi- Processes.

2.) Implementing TLB [Translate Lookaside Buffer].

3.) Dealing with the implementation of Binary Semaphores, locks.

4.) Calculating Effective Access Time.

* **Extending the functionality of Paging using Two-Level Paging in Multi Processes :**
* Suppose a logical address (on 32-bit machine with 4K page size) is divided into:

1. A page number consisting of 20 bits.

2. A page offset consisting of 12 bits.

Since the page table is paged, the page number is further  divided into:

3. A 10-bit page number.

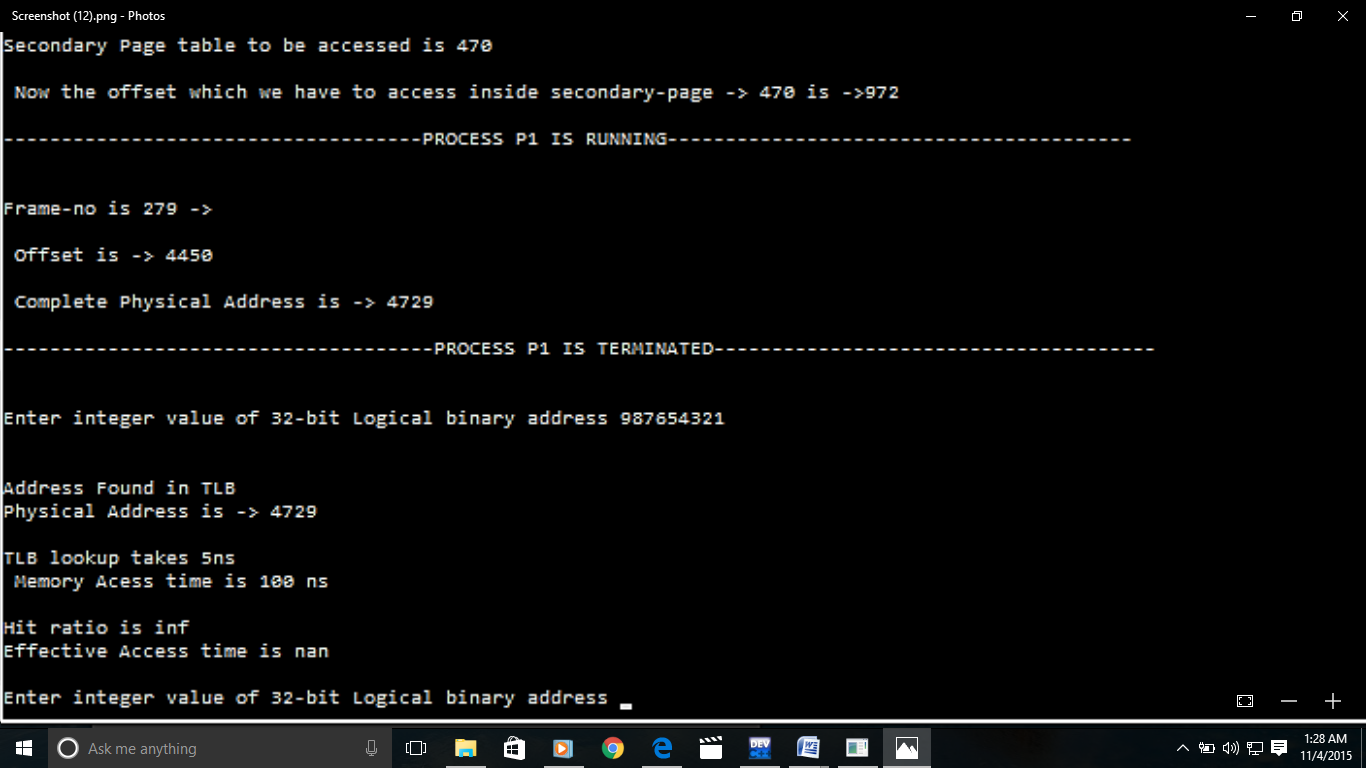
4. A 10-bit page offset



* **Implementing TLB [Translate Lookaside Buffer] :**

Extra memory references to access translation tables can slow programs down by a factor of two. So we are using translation look-aside buffers (TLB) to speed up address translation

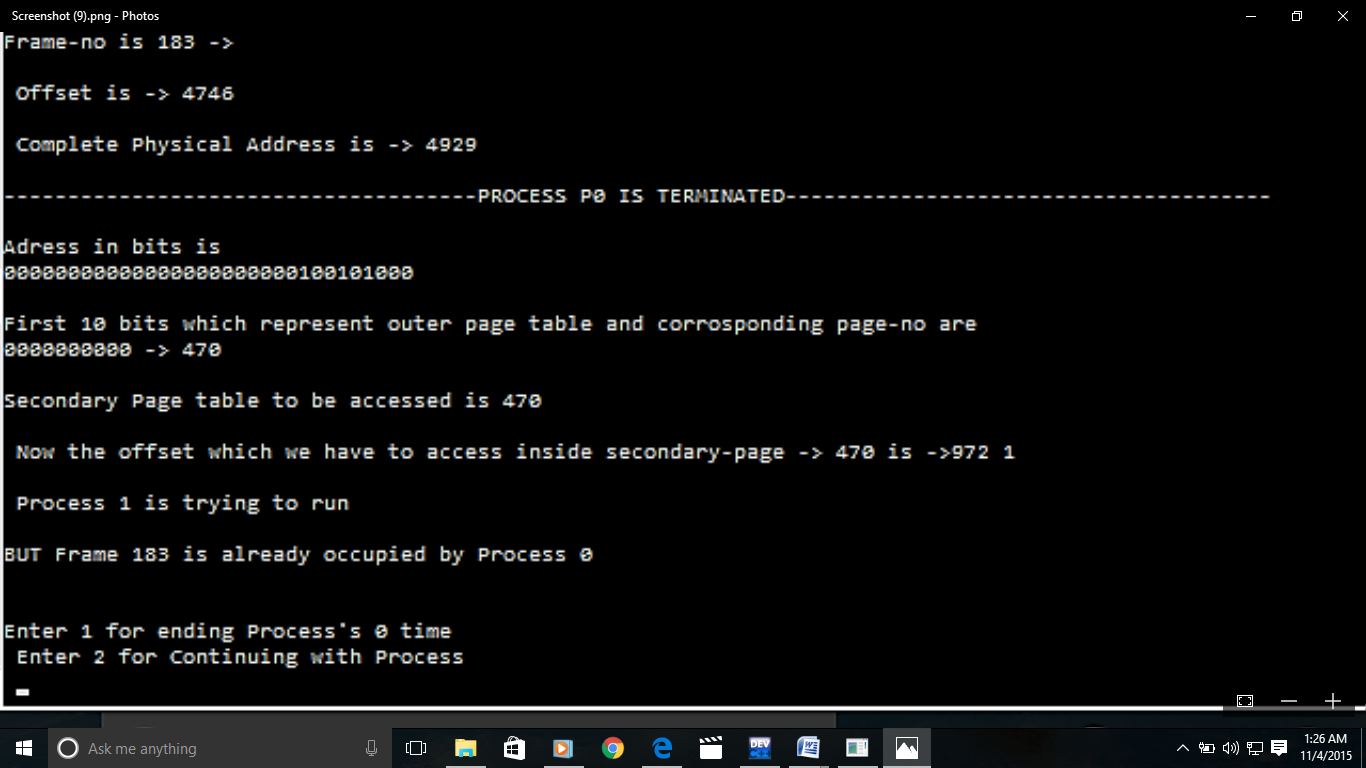
A **translation lookaside buffer** (**TLB**) is a cache that Memory Management Hardware uses to improve Virtual address translation speed. It is nearly always present in any hardware that utilizes Paged Virtual Memory.



* **Dealing with the implementation of Binary Semaphores, locks :**

Implementing Binary Semaphore assure the Lock on the Frame requested by a Process (suppose P1) which is in Use by some other Process (suppose P0) and the Frame can only be available if the process (P0) free the particular Frame for other Processes.

And it has been done by using Binary Semaphore, locking the particular Frame.



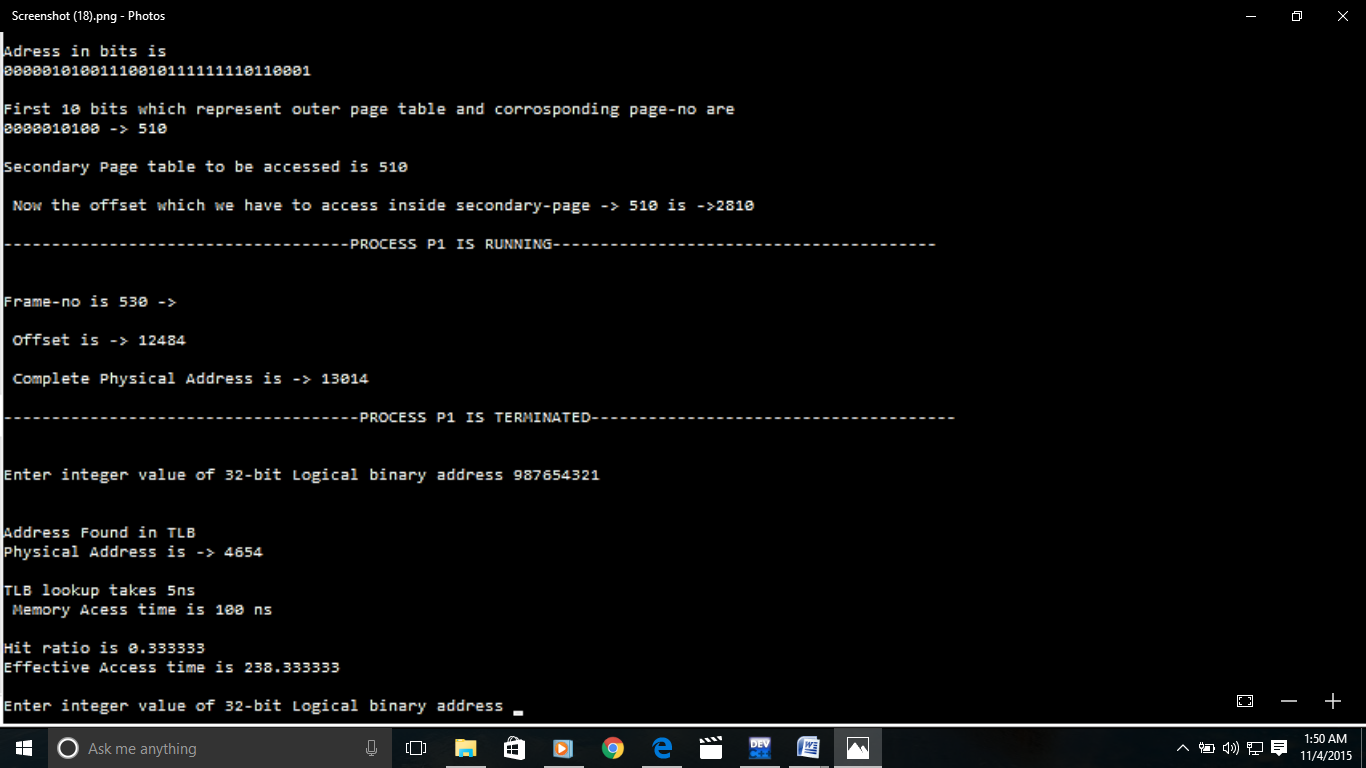
* **Calculating Effective Access Time :**

**Suppose:**

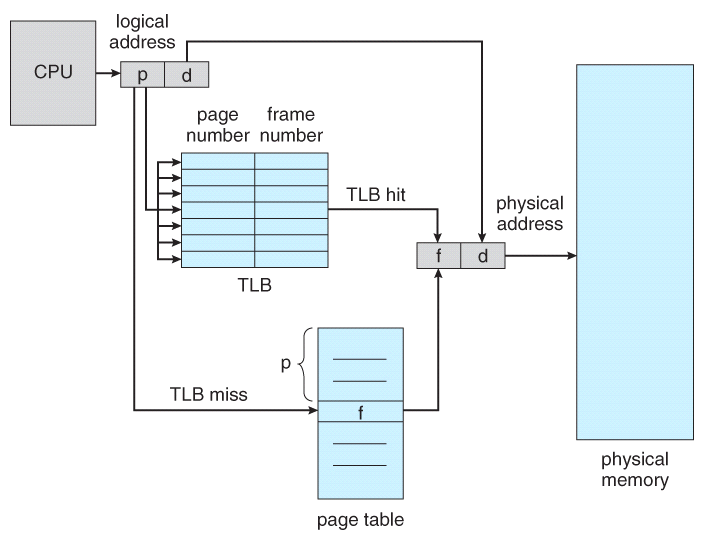
* + TLB lookup takes 5 nano sec.
  + Memory access time is 100 nano sec.
  + Hit ratio (probability to find page number in TLB) is ‘?’.
* Effective Access Time = (5+100)\*? + (5+100+100+100)\*(1- ?)   
   Memory Access Primary Secondary Memory  
   Page Table Page Table Access

Suppose ‘?’ = 80%

**EAT = 105\*.8 + 205\*.2 = 125 nano sec.**

****

* **Full Working of Two-Level Paging Using TLB Is Shown by the Diagram Below:**

****