1 Memory Management

1.1 Segmentation

Consider the following segment table:

<table>
<thead>
<tr>
<th>Segment</th>
<th>Base</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>219</td>
<td>600</td>
</tr>
<tr>
<td>1</td>
<td>2300</td>
<td>14</td>
</tr>
<tr>
<td>2</td>
<td>90</td>
<td>100</td>
</tr>
<tr>
<td>3</td>
<td>1327</td>
<td>580</td>
</tr>
<tr>
<td>4</td>
<td>1952</td>
<td>96</td>
</tr>
</tbody>
</table>

What are the physical addresses for the following logical addresses?

a) 0,430: 219 + 430 = 649
b) 1,10: 2300 + 10 = 2310
c) 2,500: illegal reference; traps to OS
d) 3,400: 1327 + 400 = 1727
e) 4,112 : illegal reference; traps to OS

1.2 Paging

Consider a paging system with the page table stored in memory.

a) If a memory reference takes 200 nanoseconds, how long does a paged memory reference take if there is no TLB?
  400 nanoseconds: 200 ns to access the page table in memory plus 200 ns to access the word in memory.

b) If we add a TLB and 75% of all page-table references are found in the TLB, what is the average memory reference time when a TLB access takes 2 nanoseconds?
  Caution: the TLB always has to be accessed, not only for misses. Hence, $2 + (0.75 \times 200) + (0.25 \times 400) = 252\,\text{ns}$
c) A typical program has 20% memory instructions. Each instruction requires 1 cycle to execute and each memory operation in the cache takes 1 cycle. Assume there are 5% data TLB misses, each requiring 100 cycles (1 cycle access + 99 cycles overhead) to handle. Also, 10% of the data accesses are cache misses each of which takes 15 cycles (1 cycle access + 14 cycles overhead). How long would it take to execute 1000 instructions?

The overhead on cache-misses is 14 cycles, the overhead on a TLB miss is 99 cycles.

\[
\begin{align*}
\text{# cache misses} &= 1000 \times 20\% \times 10\% = 20 \\
\text{# TLB misses} &= 1000 \times 20\% \times 5\% = 10 \\
\text{total cycles} &= 1000 \text{ cycles} + \text{(cache miss overhead)} + \text{(TLB miss overhead)} \\
&= 1000 + (20 \times 14) + (10 \times 99) = 1000 + 280 + 990 = 2270 \text{ cycles}
\end{align*}
\]

### 1.3 Virtual Memory

Consider a paged virtual address space composed of 32 pages of 2 KB each which is mapped into a 1 MB physical memory space.

a) **What is the format of the logical address; i.e., which bits are the offset bits and which are the page number bits? Explain.**

Each page size is 2 KB = 2^{11} Bytes. Hence, we need 11 bits in order to represent the offset and to be able to access each byte in the page. Since we have 32 pages we need 5 bits to represent all the page numbers and to be able to access each page. Then, the format of the logical address is:

- 5 bits – page number
- 11 bits – offset

b) **What is the length and width of the page table, disregarding the access right bits?**

Page table length = 32 rows since each table can have at most 32 pages.

Page table width: physical addresses are 1 MB addresses = 2^{20} Bytes. Hence, we need 20 bits to represent all the addresses at the physical memory. We need 11 bits as offset we saw before, the other 9 bits represent the page number.

### 1.4 Page Replacement

Consider the following page access pattern:

1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6

**How many page faults would occur for the following replacement algorithms, assuming one, two, three, four, five, six, or seven frames?**

4 Frames (example):

**LRU :**

\[
1 2 3 4 2 1 5 6 2 1 2 3 7 6 3 2 1 2 3 6
\]

x x x x x x x x x x

10 faults

**FIFO :**

\[
1 2 3 4 2 1 5 6 2 1 2 3 7 6 3 2 1 2 3 6
\]

x x x x x x x x x x x

14 faults

**Optimal :**

\[
1 2 3 4 2 1 5 6 2 1 2 3 7 6 3 2 1 2 3 6
\]

x x x x x x x x

8 faults
<table>
<thead>
<tr>
<th># Frames</th>
<th>LRU</th>
<th>FIFO</th>
<th>Optimal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>20</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>2</td>
<td>18</td>
<td>18</td>
<td>15</td>
</tr>
<tr>
<td>3</td>
<td>15</td>
<td>16</td>
<td>11</td>
</tr>
<tr>
<td>4</td>
<td>10</td>
<td>14</td>
<td>8</td>
</tr>
<tr>
<td>5</td>
<td>8</td>
<td>10</td>
<td>7</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>10</td>
<td>7</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
</tr>
</tbody>
</table>