CS 537: Intro to Operating Systems
Worksheet 4 - Multi-level Page Tables
July 12th, 2017 (Wednesday)

Assume a system with a 2-level page table.

Parameters:
- page size = 32 bytes
- virtual address space size = 1024 pages
- physical memory size = 128 pages
- Size of one Page Directory Entry (PDE) = 1 byte
- Size of one Page Table Entry (PTE) = 1 byte
- Value of Page Directory Base Register (PDBR) = 108 (decimal) [This means the page directory is held in this page]

The format of the PDE and the PTE is simple. The high-order (left-most) bit is the VALID bit. If the bit is 1, the rest of the entry is thePFN. If the bit is 0, the page is not valid.

You are given two pieces of information to begin with. First, you are given the value of the page directory base register (PDBR), which tells you which page the page directory is located upon. Second, you are given a complete dump of each page of physical memory. A page dump looks like this:

```
page 0: 08 00 01 15 11 1d 1d 1c 01 17 15 14 16 1b 13 0b ...
page 1: 19 05 1e 13 02 16 1e 0c 15 09 06 16 00 19 10 03 ...
page 2: 1d 07 11 1b 12 05 07 1e 09 1a 18 17 16 18 1a 01 ...
...```

which shows the 32 bytes found on pages 0, 1, 2, and so forth. The first byte (0th byte) on page 0 has the value 0x08, the second is 0x00, the third 0x01, and so forth.

For each virtual address, write down the physical address it translates to AND the data value at this physical address OR write down that it is a segmentation fault (an out-of-bounds address). Write all answers in hexadecimal.

<table>
<thead>
<tr>
<th>Virtual Address</th>
<th>Physical Address OR Seg Fault</th>
<th>Data Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x611c</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x3da8</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1
CS 537: Intro to Operating Systems
Worksheet 4 - Multi-level Page Tables
July 12th, 2017 (Wednesday)

Assume a system with a 2-level page table.

Parameters:
- page size = 32 bytes
- virtual address space size = 1024 pages
- physical memory size = 128 pages
- Size of one Page Directory Entry (PDE) = 1 byte
- Size of one Page Table Entry (PTE) = 1 byte
- Value of Page Directory Base Register (PDBR) = 108 (decimal) [This means the page directory is held in this page]

The format of the PDE and the PTE is simple. The high-order (left-most) bit is the VALID bit. If the bit is 1, the rest of the entry is thePFN. If the bit is 0, the page is not valid.

You are given two pieces of information to begin with. First, you are given the value of the page directory base register (PDBR), which tells you which page directory is located upon. Second, you are given a complete dump of each page of physical memory. A page dump looks like this:

```
page 0: 08 00 01 15 11 1d 1d 1c 01 17 15 14 16 1b 13 0b ...
paging 1: 19 05 1e 13 02 16 1e 0c 15 09 06 16 00 19 10 03 ...
paging 2: 1d 07 11 1b 12 05 07 1e 09 1a 18 17 16 18 1a 01 ...
...```

which shows the 32 bytes found on pages 0, 1, 2, and so forth. The first byte (0th byte) on page 0 has the value 0x08, the second is 0x00, the third 0x01, and so forth.

For each virtual address, write down the physical address it translates to AND the data value at this physical address OR write down that it is a segmentation fault (an out-of-bounds address). Write all answers in hexadecimal.

<table>
<thead>
<tr>
<th>Virtual Address</th>
<th>Physical Address OR Seg Fault</th>
<th>Data Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x611c</td>
<td>0xF6BC</td>
<td>0x08</td>
</tr>
<tr>
<td>0x3da8</td>
<td>Seg Fault</td>
<td></td>
</tr>
</tbody>
</table>
V.A: 110000 010000 111000

# PTEs = 2

1 page = 2^5

How many pages for PTEs = \( \frac{2^{10}}{2^5} = 2^5 \)

PDBR

PDI = 24

PDE = 0x01

PTI

PEN of the PT

PEN

PT_31

Page 0

Page 31

32 pages for PTEs.
PTE = 0x6b5 = \[ \begin{array}{c}
1 \\
0 \\
1 \\
1 \\
0 \\
1 \\
\end{array} \]

PFN of the page

PFN = 53

\[ \frac{64}{32} 16 \frac{8}{4} 2 1 \]

P.A. = 0x6BC

\[ \frac{64}{32} 16 \frac{8}{4} 2 1 \]

Offset