Segmentation and Paging Combined

Introduction

This shows the function used to map a logical address to a physical address for schemes combining paging and segmentation. Throughout this handout, *page_size* is the size of the page (which is a multiple of 2), *seg_tbl_base_reg* contains the address of the base of the segment table, and *memory* is the main store of the computer. We will assume the entire program is in memory, so no error handling is given; were this assumption false, the situation where the requested address were not in memory would need to be handled (by generating a fault and loading the appropriate data structure).

Segmented Paging

In this algorithm, the page tables are segmented. The virtual address is represented as a pair (*logical_page*, *offset*), but the *logical_page* consists of a pair (*seg_number*, *seg_offset*) indicating which segment number *seg_number* of the page table the frame number *frame_no* is stored in, and the offset *seg_offset* from the base of that segment table. As usual, an associative memory is first checked; this will be represented by the funcetion *assoc_page_table*, which returns the frame number if that is in the table, and –1 if not:

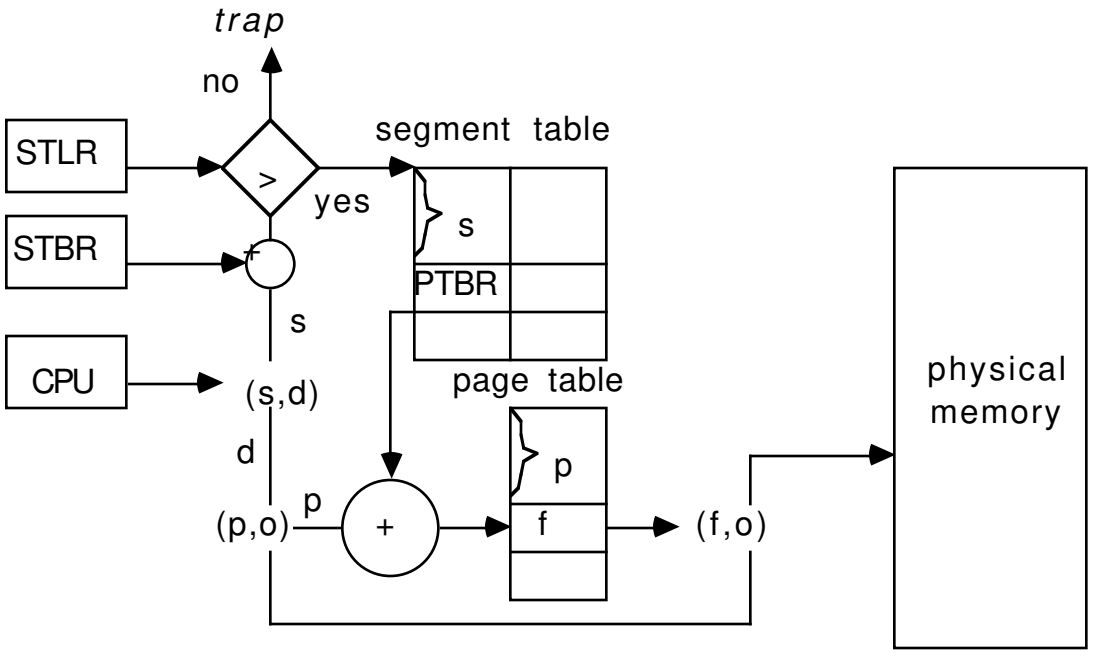
```
function NL_map((logical_page, offset)): physical_address;
var   frame_no: integer;            (* number of frame *)
      pg_tbl_base: integer;         (* addr. of page table segment *)
begin
      frame_no := assoc_page_table(logical_page);
      if frame_no = -1 then begin
            pg_tbl_base := memory[seg_tbl_base_reg + seg_number];
            frame_no := memory[pg_tbl_base + seg_offset];
      end;
      NL_map := frame_no * page_size * offset;
end (* NL_map *)
```

Paged Segmentation

In this algorithm, the segments are paged. The virtual address is represented as a pair (*seg_number*, *offset*), but the offset consists of a pair (*page_number*, *page_offset*), indicating which page number *page_number* of the segment *seg_number* the frame number *frame_no* is stored in, and the offset *page_offset* from the base of that page. As usual, an associative memory is first checked; this will be represented by the funcetion *assoc_page_table*, which returns the frame number if that is in the table, and –1 if not. Note it takes the segment number as an argument as well:

```
function NL_map((seg_number, offset)): physical_address;
var   frame_no: integer;            (* number of frame *)
      pg_tbl_base: integer;         (* addr. of page table segment *)
begin
      frame_no := assoc_page_table(seg_number, page_number);
      if frame_no = -1 then begin
            pg_tbl_base := memory[seg_tbl_base_reg + seg_number];
            frame_no := memory[pg_tbl_base + page_number];
      end;
      NL_map := frame_no * page_size * page_offset;
end (* NL_map *)
```

In pictures:


trap
no
STLR
STBR
CPU
>
+
yes
segment table
s
PTBR
s
(s,d)
page table
d
p
p
(p,o)
+
f
(f,o)
physical
memory