CS 620: Homework 10

Carmen St. Jean

November 15, 2012

1. *(5) Consider the following segment table:*

Segment	Base	Length
0	219	600
1	2300	14
2	90	100
3	1327	580
4	1952	96

What are the physical addresses for the following logical addresses?

(a) *0, 430*
$0, 430 + 219 \rightarrow 0, 649$

(b) *1, 10*
$1, 10 + 2300 \rightarrow 1, 2310$

(c) *2, 500*
$2, 500 + 90 \rightarrow 2, 590 \rightarrow$ out of bounds

(d) *3, 400*
$3, 400 + 1327 \rightarrow 3, 1727$

(e) *4, 112*
$4, 112 + 1952 \rightarrow 4, 2064 \rightarrow$ out of bounds

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

(a) *(1.5) If a memory reference takes 100 nanoseconds, how long does a paged memory reference take?*
There are two memory accesses for every logical address look-up with paging. Therefore, a paged memory reference takes 200 nanoseconds.

(b) *(2.5) If we add associative registers, and 75 percent of all page-table references are found in the associative registers, what is the effective memory reference time? (Assume that finding a page-table entry in the associative registers takes zero time, if the entry is there.)*
75% of references will take 100 nanoseconds since only one memory access is necessary for them, while the remaining 25% of references will take 200 nanoseconds. This means that the effective look-up time will be 125 nanoseconds because:
$200 * 0.25 + 100 * 0.75 = 50 + 75 = 125$

3. *Given memory partitions of 200K (Hole 1), 80K (Hole 2), 100K (Hole 3), 600K (Hole 4), 300K (Hole 5) and 700K (Hole 6) (in order), how would each of the* ***First-fit***, ***Next-fit***, ***Best-fit*** *and* ***Worst-fit*** *algorithms place processes of 153K (Process A), 280K (Process B), 85K (Process C), 310K (Process D), 650K (Process E) (in order)? For first-fit algorithm, searching starts at the beginning of the set of holes every time. For next-fit algorithm, searching starts at the beginning of the set of holes the first time.*

First-fit

Process A	153 K
Hole 1	47 K
Hole 2	80 K
Process C	85 K
Hole 3	15 K
Process B	280 K
Process D	310 K
Hole 4	10 K
Hole 5	300 K
Process E	650 K
Hole 6	50 K

Next-fit

Process A	153 K
Hole 1	47 K
Hole 2	80 K
Hole 3	100 K
Process B	280 K
Process C	85 K
Hole 4	235 K
Hole 5	300 K
Process D	310 K
Hole 6	290 K

*

Best-fit

Process A	153 K
Hole 1	47 K
Hole 2	80 K
Process C	85 K
Hole 3	15 K
Process D	310 K
Hole 4	290 K
Process B	280 K
Hole 5	20 K
Process E	650 K
Hole 6	50 K

Worst-fit

Hole 1	200 K
Hole 2	80 K
Hole 3	100 K
Process B	280 K
Hole 4	320 K
Hole 5	300 K
Process A	153 K
Process C	85 K
Process D	310 K
Hole 6	152 K

*

* No hole large enough remains so Process E will have to wait.

4. *In the following problem, main memory consists of 64 10-bit words. The contents of main memory are as follows:*

Address	Contents	Address	Contents	Address	Contents	Address	Contents
0	823	16	876	32	317	48	55
1	578	17	617	33	734	49	268
2	30	18	157	34	703	50	518
3	64	19	612	35	439	51	170
4	521	20	509	36	33	52	10
5	568	21	353	37	667	53	920
6	112	22	785	38	391	54	867
7	583	23	264	39	606	55	25
8	371	24	124	40	16	56	912
9	293	25	228	41	986	57	405
10	20	26	315	42	539	58	19
11	168	27	693	43	613	59	25
12	570	28	829	44	182	60	108
13	827	29	182	45	78	61	258
14	15	30	611	46	943	62	624
15	157	31	45	47	512	63	217

Note: the addresses in the problems below are virtual addresses. The normal convention is to place the most significant portion of the address in the most significant bits of the address. Thus, the virtual addresses below will be formatted as follows:

page # | word #

Consider an operating system using paging. Main memory is divided into 4-word page frames. In a page table descriptor, the low order bits contain the frame in which the page resides. To the left of the frame number is the residency bit and the remaining bits are used by the operating system (protection, whether the page has been modified, etc.). Thus the page descriptor looks like:

OS ResidencyBit Frame#

The page table pointer for a process points to address 18. Give the results of the following memory references by that process:

(a) *69*

$69_{10} = 1000101_2$

Page $10001_2 = 17_{10}$, Word $01_2 = 1_{10}$

Page table pointer + Page number = $18 + 17 = 35$

Address(35) $\rightarrow$ 439

$439_{10} = 110110111_2$

Frame: 0111 (7)

Residency bit: 1

OS: 1101

Address = frame number * frame size + word number = $7 * 4 + 1 = 29$

Address(29) = 182

(b) *51*

$51_{10} = 110011_2$

Page $1100_2 = 12_{10}$, Word $11_2 = 3_{10}$

Page table pointer + Page number = $18 + 12 = 30$

Address(30) $\rightarrow$ 611

$611_{10} = 1001100011_2$

Frame: 0011 (3)

Residency bit: 0

OS: 10011

(c) *36*

$36_{10} = 100100_2$

Page $1001_2 = 9_{10}$, Word $00_2 = 0_{10}$

Page table pointer + Page number = $18 + 9 = 27$

Address(27) $\rightarrow$ 693

$693_{10} = 1010110101_2$

Frame: 0101 (5)

Residency bit: 1

OS: 10101

Address = frame number * frame size + word number = $5 * 4 + 0 = 20$

Address(20) = 509

(d) *86*

$86_{10} = 1010110_2$

Page $10101_2 = 21_{10}$, Word $10_2 = 2_{10}$

Page table pointer + Page number = $18 + 21 = 39$

Address(39) $\rightarrow$ 606

$606_{10} = 1001011110_2$

Frame: 1110 (14)

Residency bit: 1

OS: 10010

Address = frame number * frame size + word number = $14 * 4 + 2 = 58$

Address(58) = 19