ARQUITECTURA DE COMPUTADORES 2

Estudo para o teste teórico 1

2/3 - INTEL/MIPS
-CISC
-Endereçamento real: 1MB (2^20)
-Endereçamento protegido: 4GB (2^32)
-Segmentos de 64K: (end=segment<<4 + offset)

4/5 - I/O
-STROBE (CPU internal)
-T.sincrona: 4T + WAIT STATES
-T.assincrona: AKNOWLEDGE (dados OK)
-T.semi-sincr: WAIT (indica ao CPU para esperar)
-R/W\ actua em flanco descendente

7 - INTERRUPÇÕES
-Interrupção por s/w: Ler status dos perifericos. Uma RSI geral
-Interrupção por vectores: Uma RSI para casa periférico. INT (perif) + INT_ACKNOWLEDGE (CPU) + VECTOR (perif)
-Daisy chain: ligação serie de interrupt aknowledge em periféricos

8 - INTERRUPÇÕES x86, TIMERS, COUNTERS
-Interrupções: 256 vectores (pos=4*VECTOR=1024B - 0x00000-0x003FF) [2B offset + 2B segmento]
-Timer impulso: Fo=Fi/K ; To('0')=K*Ti (Fi=1/Ti) [@ Intelx86: Fi=(1/4)*fCPU]
-Timer regulavel: Fo=Fi/(KA+KB) ; To=KB*Ti('1')+KA*Ti('0')

9/10 - DMA, 80188 GERAL
-DMA: fetch+deposit
-BUS_REQUEST (DMA) -> BUS_GRANT (CPU)
-Operação block: transfere todo o bloco
-Operação cycle stealing: DMA usa 1 bus-cycle para transferir 1 B/W
-Cycle-stealing: CPU pode ter de esperar (@ cycle 1, 3, 5)
-DMA partilhado: barramento usado 2x
-DMA dedicado: barramento usado 1x
-DMA x86 config: Transfer counter, Source, Dest, Channel control word.
-Peripherac Control Block: end=0xFFXX (XX=id)
-Selecção programável (CS): Mem (UCS, LCS, MCS0~MCS3) ; I/O,Mem (PCS0~PCS6)
-Configuração: PACS (end + Wait states 0~3) ; MPCS (MS + wait states 4~6)
-PCSX: PBA + 7 blocos (128 Bytes)
-Timers: [T0/T1, T2]: Control reg (op mode), MaxCount (const), Count reg
-Timer T2 [internal](1 const): T'0'=(1/4)*T = 1/(4*Fi)

11/12 - MEMÓRIA
-Matriz SRAM: ADD[H+L] + CE(EN)[@ row/add_H] -> CE[0]~CE[n]
-SRAM: +Memória: descodif[Addr[H]+CE(EN)] -> CE[0]~CE[n]
-DRAM: Read: bit=VDD/2 + select -> delta_U[bit] + write/refresh
-DRAM: WE\ (=R/W\) RAS\ (internal refresh), CAS\ (chip select)
-DRAM: +Memória: descodif[Addr[H]+CAS(EN)] -> CAS[0]~CAS[n]

13/14 - CACHE
-Localidade espacial: Se A acedido -> probabilidade de acesso a A-1,A+1,...
-Localidade temporal: Se zona A acedida -> provavel que seja novamente acedida.
-Dado no nivel superior: HIT, se nao: MISS (+transferencia)
-HIT_ratio=Nºhits/Nºacessos
-MISS_ratio=1-HIT_ratio
-Tempo de acesso: HIT TIME, tempo de transferencia: MISS PENALTY
-Tempo de acesso médio: Ta = HIT_ratio * HIT_time + (1-HIT_ratio) * PENALTY_time
-Mapeamento associativo: add[16] = (bloco/tag[13] + byte[3])
-Mapeamento directo: add[16] = (tag[5] + group[8] + byte[3])
 add_bloco = add_real/tamanho_bloco ; Linha = add_bloco % num_blocos
-Mapeamento parcialmente associativo: (Mapeam. directo) x 2+
-Substituição: LRU (least recently used), LFU (least frequently used), FIFO, Random
-Politicas de escrita:
1)Dado @ cache: WRITE-THROUGH (cache + mem write), WRITE-BACK (cache write -> mem write)
2)Dado not @ cache: WRITE-ALLOCATE (load to cache), WRITE NO-ALLOCATE (mem refresh)