Extra Question

- Implement the memory copy function

```
memcopy( int[] A, int[] B, int n ) {
    for ( int i = 0; i < n; i++ ) A[i] = B[i];
}
```

  - Assume a MIPS machine with 1 instruction per clock cycle, delayed branching, a 5 stage pipeline, forwarding, and interlock on unresolved load hazards
  - Respect register conventions
  - Use only true assembly language
  - Use careful instruction ordering to make a loop that takes the shortest possible number of cycles to complete

Answer

```
memcopy:
beq $a2, $zero, exit # check n==0
sll $t1, $a2, 2          # end address at +4n
add $t1, $t1, $a0
loop: lw $t0, 0($a0)   # no interlock
addi $a0, $a0, 4        # load delay slot
sw $t0, 0($a1)
bne $a0, $t1, loop     # 5 clock cycles per loop iteration ( including the branch delay slot)
addi $a1, $a1, 4        # branch delay slot
exit: jr $ra
nop                      # must include nop in delay slot for “jr”
```