Micro-program example from Last lecture

− Timing error: need two cycles to start an instruction execution
  1. instruction fetch (load IR)
  2. EXO (load CAR)
− ... then the micro-program at 0||IR can start

Micro-programming
− goal: write micro program for each instruction that implement the correct behavior
− what if we want to do an operation like register indirect addition?
  o R[DR] ← M[R[SA]]+R[SB]
− First, load m[R[SA]] into register:
  o R[?] ← M[R[SA]]
− Second cycle: do the addition
  o R[?] + R[SB]
− What do we use for R[?]?
  o R[0]-R[7] are used by the programmer and shouldn’t be changed
  o A register is created for this temporary storage R[8]
− R[8] will be used only by micro programs, we need to access it
  o Increase register address from 3 to 4 bits
  o Eg. DA will be 4 bits:
    ▪ TA||DR
    ▪ TD||DR (1 bit from microinstruction:3 bits from instruction)
    ▪ If the address starts with a 1, it refers to R[8]
    ▪ Any address 0abc refer to register abc
    ▪ Basically, TD=0: use programmers DR register TD=1 use private register 8
− So to do the register indirect:
  o Cycle1: R[8] ← M[R[SA]]
    ▪ TD=1
    ▪ TA=0
    ▪ M[0] (take address from t)
    ▪ MW=0 (don’t write to mem)
    ▪ MD=1 (take result from memory, not function unit)
    ▪ RW=1 (write to DA=R[8])
    ▪ NA= (control address of cycle 2 micro-op)
    ▪ MS=001 (jump to NA)
    ▪ MC=0 (use NA, not 011 opcode)
    ▪ IL=0
    ▪ PI=0
    ▪ PL=0
- TA=1 (use R8)
- TB=0 (use SB)
- MB=00010 (add)
- MD=0
- RW=1 (write to DA)
- TD=0 (use DR)

Other choices
- Some other choice we could have made
- Hardwired multiple cycle control
- We could have create an ASM diagram for the decoder & implemented that
- … instead of the control ROM & microprogramming
- eg.

MID TERM = June 25