COMP2120 Tutorial Exercise

YUAN Wenxuan (UID: 3036292740)

May 5, 2025

1. Programmed I/O
   An intelligent thermometer is installed in a room to detect the current room temperature. The thermometer is connected to the onboard computer with a Control and Status Register `TCSR` and a Buffer Register `TBR`. The current room temperature is displayed on the panel of the air conditioner on two 7-segment LEDs, with 2 buffer register `LEDBR1` and `LEDBR2` for display.
   The `TCSR` has the following format:

   Bit 0 Thermometer Ready bit, the thermometer is ready

   Bit 1 Set this bit to start reading the temperature

   Bit 2 Reading is available in `TBR`, automatically cleared after data read

   Bit 3 Set this bit to 1 to turn on the compressor, 0 to turn off

   For `LEDBR1`, `LEDBR2`, just write the corresponding digit (in integer) to the buffer register.

   Write an assembly language program using *Programmed I/O* to read the room temperature and display on the panel. (You may invent your own instruction set as long as it is reasonable. Comment your program so that it can be understood.)

   Solution:

```
L1:     LD    TCSR,R2
        AND     R2,#0x1,R3  # check bit 0 thermometer ready
        BZ      L1          # if not ready, loop and wait
        MOV     #0x2,R1     # set bit 1 to 1, start reading
        ST      R1,TCSR     # start reading
L2:     LD      TCSR,R2
        AND     R2,#0x4,R3  # check bit 2 reading available
        BZ      L2          # if not available, loop and wait
        LD      TBR,R0
        CALL    DIV         # input R0, output R1,R2 (R1 remainder, R2 quotient)
        ST      R1,LEDBR1   # R1 remainder, the second digit
        ST      R2,LEDBR2   # R2 quotient, the first digit
        HLT
DIV:    PUSH    R3          # R3 is modified in DIV, thus first push it
        SUB     R1,R1,R1    # R1 = 0
        MOV     R0,R2       # R2 = R0
L3:     SUB     R0,#0xa,R0  # R0 -= 10
        BLZ     L4          # goto L4 if result < 0
        ADD     R1,#0x1,R1  # R1 += 1
        MOV     R0,R2       # R2 = R0
        BR      L3
L4:     POP     R3          # restore R3
        RET
```

2. **Addressing Modes**
Consider implementing a displacement mode in the machine in Asg2/4, with `MBR` connecting to both `S1-Bus` and `S2-Bus`.

```
ST  DISP1(R1), DISP2(R3)
```

ST	DISP(R1)	-	DISP2(R3)
DISP1			
DISP2			

Solution:

```
Fetch:
MAR <- PC
IR <- mem[MAR]
PC <- PC + 4

Execute:
RFOUT1 <- R1
MAR <- PC
MBR <- mem[MAR] # MBR = DISP1
PC <- PC + 4
A <- RFOUT1
B <- MBR
C <- A + B
MAR <- C
TEMP <- MAR     # TEMP = DISP1 + (R1)

RFOUT1 <- R3
MAR <- PC
MBR <- mem[MAR]
PC <- PC + 4
A <- RFOUT1
B <- MBR
C <- A + B      # C = DISP2 + (R3)
MAR <- TEMP     # MAR = DISP1 + (R1)
MBR <- mem[MAR] # MBR = (DISP1 + R1)
MAR <- C        # MAR = DISP2 + (R3)
mem[MAR] <- MBR # mem[DISP2 + R3] = MBR = mem[DISP1 + R1]
```