Software 3

SW 3: Subroutines

Summer 2021

Thiha Myint

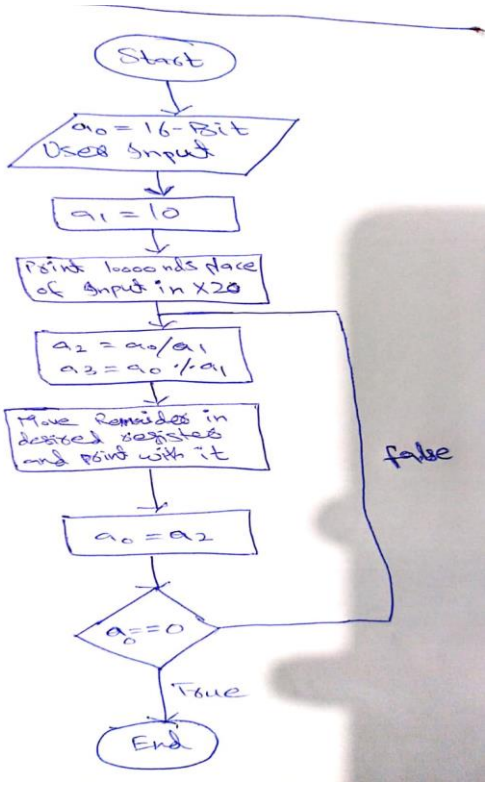
Description

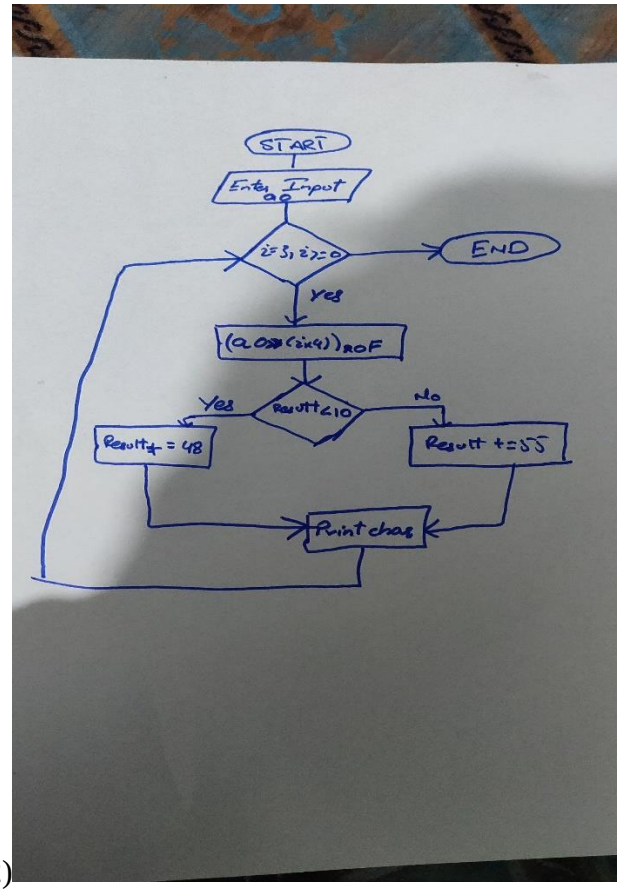
This program has two parts. First one is to converts an unsigned 16-bit binary value into BCD values. Assuming the lower halfword of register x10 has the binary value and then placing the 10000nds place is register x20, the 1000nds place in register x21, the 100eds place in register x22, the 10s place in register x23 and the 1s place in x24.

The second part is as follow: to read a program that converts a 4-digit binary coded decimal value to a binary value. Assume the lower halfword of register x10 stores the value. Place the final binary number in register x20. Assume all BCD inputs are valid. The binary to decimal is done using the shift and add 3 method. The BCD to binary is done using multiplication.

Flowchart

1)



2) 

RISC-V Program code

1)

```
#Program to convert 16-Bit BCD value to binary value

        la      a0, Input
        jal     ra Print
        jal     ra Read_Input
        mv      t0 a0
        la      a0, Output
        jal     ra Print
        mv      a0 t0
        jal     ra BCDtoBinary
        j end

BCDtoBinary:
        addi    sp, sp, -32
        sw      ra, 28(sp)
        sw      s0, 24(sp)
        addi    s0, sp, 32
```

```
        sw      a0, -12(s0)
        addi    a0, zero, 3
        sw      a0, -20(s0)
        j       FOr_i
FOr_i:
        lw      a0, -20(s0)
        mv      a1, zero
        blt     a0, a1, end_loop
        j       mult
mult:
        lw      a0, -12(s0)
        lw      a1, -20(s0)
        slli    a1, a1, 2
        srl     a0, a0, a1
        andi    a0, a0, 15
        sw      a0, -16(s0)
        lw      a0, -16(s0)
        addi    a1, zero, 9
        bltu    a1, a0, result_grtr_10
        j       result_less_10
result_less_10:
        lw      a0, -16(s0)
        addi    a0, a0, 48
        sw      a0, -16(s0)
        jal     ra, PrintB
        j       inc_i
result_grtr_10:
        lw      a0, -16(s0)
        addi    a0, a0, 55
        sw      a0, -16(s0)
        jal     ra, PrintB
        j       inc_i
inc_i:
        lw      a0, -20(s0)
        addi    a0, a0, -1
        sw      a0, -20(s0)
        j       FOr_i
end_loop:
        lw      s4, -12(s0)
        lw      s0, 24(sp)
        lw      ra, 28(sp)
        addi    sp, sp, 32
        ret

Read_Input:
        li a7, 5
        ecall
        jr ra
Print:
        li a7, 4
        ecall
        jr ra
PrintB:
        li a7, 11
        ecall
        jr ra
end:
```

```
.data
Input: .asciz "Input: "
Output: .asciz "Ouput: 0x"
Comma_Space: .asciz ", "
New_Line: .asciz "\n"
```

2)

```
# Program to convert 16-Bit binary value into BCD values

        la a0, Input
        jal ra Print
        jal ra Read_Input
        mv t0, a0
        la a0, New_Line
        li a7, 4
        ecall
        la a0, Output
        li a7, 4
        ecall
        mv a0, t0
        li a1, 10
        jal ra, Get_BCD_Value
        mv x24, a3
        mv a0, a2
        jal ra, Get_BCD_Value
        mv x23, a3
        mv a0, a2
        jal ra, Get_BCD_Value
        mv x22, a3
        mv a0, a2
        jal ra, Get_BCD_Value
        mv x21, a3

        li x20, 0
        la a0, X20
        jal ra, Print
        mv a0, x20
        jal ra, Print_BCD
        jal ra, Print_Comma_And_Space
        la a0, X21
        jal ra, Print
        mv a0, x21
        jal ra, Print_BCD
        jal ra, Print_Comma_And_Space
        la a0, X22
        jal ra, Print
        mv a0, x22
        jal ra, Print_BCD
        jal ra, Print_Comma_And_Space
        la a0, X23
        jal ra, Print
        mv a0, x23
```

```
        jal ra, Print_BCD
        jal ra, Print_Comma_And_Space
        la a0, X24
        jal ra, Print
        mv a0, x24
        jal ra, Print_BCD
        la a0, New_Line
        li a7, 4
        ecall
        j end

Get_BCD_Value:
        div a2, a0, a1
        rem a3, a0, a1
        jr ra

Read_Input:
        li a7, 5
        ecall
        jr ra
Print:
        li a7, 4
        ecall
        jr ra
Print_BCD:
        li a7, 1
        ecall
        jr ra

Print_Comma_And_Space:
        la a0, Comma_Space
        li a7, 4
        ecall
        jr ra
end:

.data
Input: .asciz "Input: "
Output: .asciz "Ouput: "
X20: .asciz "x20:"
X21: .asciz "x21:"
X22: .asciz "x22:"
X23: .asciz "x23:"
X24: .asciz "x24:"
Comma_Space: .asciz ", "
New_Line: .asciz "\n"
```