EdC - Prueba2 - 2018/19

Problema 1.

1. JMP es un salto incondicional, el contador de programa toma un nuevo valor y la ejecución continua por el punto del programa a donde se ha saltado

CALL es un salto de características parecidas al anterior pero la diferencia radica en que se salta a un segmento de código llamado subrutina y tras ejecutar dicho segmento el programa ha de seguir ejecutándose por la instrucción siguiente a CALL, se ha de volver al sitio donde se saltó.

2. El ciclo de búsqueda es común para ambas:

1. IR ← MEMPROG(PC), PC ← PC + 1
   WIR, IPC

El ciclo de ejecución de JMP:

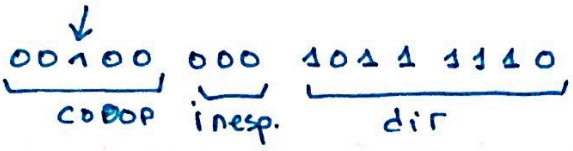
2. AC ← IR7:0 — WAC, INM, TB
3. PC ← AC — WPC, RAC

..

El ciclo de ejecución de CALL

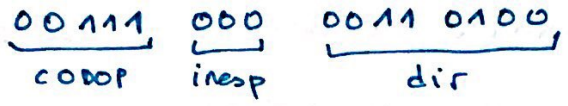
2. AC ← IR7:0 — WAC, Inm, TB
   MAR ← SP — Wmar, RSP
3. MEMDAT(MAR) ← PC — Wmem, RPC
   SP ← SP − 1 — DSP
4. PC ← AC — WPC, RAC

3. CALL $BE



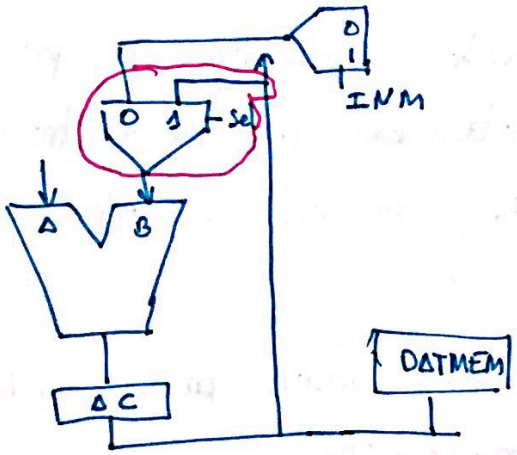
$\longleftrightarrow$ $20BE

JMP 52 — 52 $\xrightarrow{hex}$ 48 + 4 = 3·16 + 4 → 34_{16}



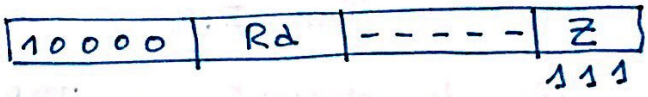
$\longleftrightarrow$ $3834

4. Hay que hacer el siguiente cambio



establecemos un camino desde el bus de datos a la ALU, usamos un MUX.

Formato:A:



1. AC ← R(IR2:0) W_{AC}, T_B
2. MAR ← AC W_{MAR}, R_{AC}
3. AC ← MEM DAT(MAR) + R(IR10:8) W_{ac}, R_{mem}, sel, S
   SREG ← CNZV(MEMDAT(MAR) + R(IR10:8)) W_{SR}
4. R(IR10:8) ← AC R_{ac}, W_{reg}

Problema 2

(a)
```
inicializapuertos:  sbi ddrc, 2
                    cbi ddrc, 4
                    sbi portc, 4
                    ret
```

(b) $f = 4$ Mhz $\longrightarrow$ 1 s — 4.000.000

0.5 s — 2.000.000

$f/64 \longrightarrow$ 0.5 s — $\frac{2.000.000}{64} = 31250 = VMAX$

```
inicializatimer:  ldi  r16, high(31250)
                  sts  ocr1ah, r16
                  ldi  r16, low
                  sts  ocr1al, r16
                  ldi  r16, 0
                  sts  tcht1h, r16
                  sts  tcnt1l, r16
                  ldi  r16, 2
                  sts  timsk1, r16
                  ret
```

(c)
```
generarack:  in   r16, sreg
             sbic pinc, 2
             rjmp ponero
             rjmp poner1
ponero:      cbi  portc, 2
             reti
poner1:      sbi  portc, 2
             reti
```

(d)

```
.include "m328Pdef.inc"
.cseg
.org 0
jmp main
.org 0x16
jmp generack
.org 0x40
main:   call inicializapuertos
        call inicializatimer
        sei
        cbi portc, 2
espera1: sbic pinc, 4
         rjmp espera1
suelta1: sbis pinc, 4
         rjmp suelta1
espera2: sbic pinc, 4
         rjmp espera2
         ldi r16, 0b00001011
         sts tccr1b, r16
suelta2: sbis pinc, 4
         rjmp suelta2
espera3: sbic pinc, 4
         rjmp espera3
         ldi r16,0
         sts tccr1b, r16
         cbi portc, 2
         clr r16
         sts tcnt1h, r16
         sts tcnt1l, r16
suelta3: sbis pinc, 4
         rjmp suelta3
         rjmp espera1
```

Problema 3

1. STS $109, R10

2. LDS R20, $200

3. LDS R0, $104
   OUT 2, R0

4. ldi XH, 2
   ldi XL, 0xaf
   o bieu: ldi r27, 2
   ldi r26, 0xaf
   o bieu: ldi r27, high (0x2af)
   ldi r26, low (0x2af)

5. ldd R2, Y+9

6. IN R16, DDRC
   OUT DDRB, R16

7. clv

8. CP R3, R2
   BRLO etis1

9. ICALL