

## Prueba 4

**1.1** Hay 4 zonas:

1<sup>a</sup> zona: registros de propósito general (R0, R1... R32)  
ocupan las direcciones (0x000 - 0x01F)

2<sup>a</sup> zona: registros de I/O, hay 64  
ocupan las direcciones (0x020 - 0x05F)

3<sup>a</sup> zona: registros de I/O extendida, hay 160  
ocupan las direcciones (0x060 - 0x0FF)

4<sup>a</sup> zona: memoria de datos (SRAM), las hay  
de distinto tamaño según modelo  
ocupan las direcciones (0x100 - RAMEND)  
en el ATmega328P (0x100 - 0x8FF)

**1.2**

(a) MOV R0, R4      ó    LDS R0, \$4

(b) IN R0, 4        ó    LDS R0, \$24

(c) LDS R0, \$64

(d) LDS R0, \$104

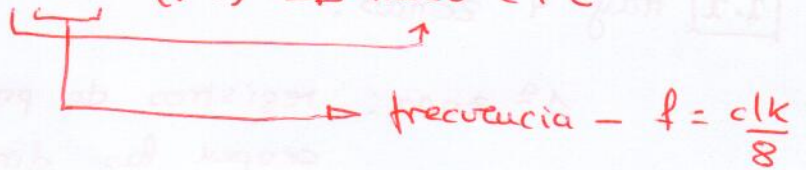
**1.3**

MULTIPLICA:  
PUSH R1  
PUSH R0  
LD R4, X  
LD R5, Y  
MULS R5, R4  
MOV R5, R1  
MOV R4, R0  
POP R0  
POP R1  
RET

1.4

Si  $OCR1A = \$0064 \Rightarrow V_{MAX} = 100$

Si  $TCCR1B = 00001010$  (BA)  $\Rightarrow$  modo CTC



como  $f = 8\text{MHz} \Rightarrow$  la frecuencia del temporizador con  $\frac{clk}{8}$  seña  $1\text{MHz}$

Entonces en  $1s$   $\times$   $\frac{1000000}{100}$  ciclos, para

llegar solo a  $V_{MAX} = 100 \Rightarrow$  empleará  $\frac{1s}{10.000} = 0.1\text{ms}$

Se activa el bauderín cada  $0.1\text{ms}$

1.5

1.6

2

```

• include "m328Pdef.inc"
• cseg
• org $0
  jmp start
• org $16
  jmp refrigera
• org $40

```

f = 1 MHz

1 s	—	1.000.000
30 s	—	30.000.000
30 s	—	29296

f/1024

OK

```

start: rcall iniciapuestos
       rcall iniciatimer
       ldi r16, 0b1101
       sts bccr1b, r16
       sei

```

→ puede ir dentro de la subrutina

```
fin: rjmp fin
```

```

iniciapuestos: clr r16
               out ddrb, r16
               sbr ddrd, 0
               ret

```

} puede obviarse (por defecto ya están a 0)

```

iniciatimer:  clr r16
               sts tenth, r16
               sts tentl, r16
               ldi r16, high(29296)
               sts ocr1ah, r16
               ldi r16, low(29296)
               sts ocr1al, r16
               ldi r16, 2
               sts tmskr, 2
               ret

```

} puede obviarse (por defecto ya está a 0)

refrigerera: in r16, sreg → no es obligatorio preservar sreg  
                  in r17, pin b        pues el programa principal  
  está en el bucle fin: rjmp fin

cpi r17, 25        T ≥ 25

brcc enciende    T ≥ 25

cpi r17, 23        T < 23

brcs apaga        T < 23

vuelve:        out sreg, r16   /\* si  $23 \leq T < 25$  no hago nada \*/  
                  reti

enciende:      sbi portb, 0  
                  rjmp vuelve

apaga:         cbi portb, 0  
                  rjmp vuelve