

### Test de Teoría (3.0p)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
d	c	c	b	c	c	c	d	d	a	a	b	d	b	a	a	c	d	d	a	c	a	a	b	b	c	b	a	d	c

### Test de Prácticas (4.0p)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
b	c	b	c	c	b	d	d	a	d	a	d	c	b	d	c	a	d	d	b

Las preguntas 16 y 18 tenían lapsus en su redacción que no pudimos detectar a tiempo: kit por entorno, y 2520 por 2025. A todos los que entregaron se les puntuó como acertadas, independientemente de lo que contestaran, independientemente de si contestaron.

### Examen de Problemas (3.0p)

La pregunta 1.a tenía un lapsus en su redacción que no pudimos detectar a tiempo: utilizar las dimensiones del array para nombrar el campo, surgiendo la ambigüedad de si se pide un desplazamiento “ficticio” a un elemento inexistente, o desplazamiento “real” al principio del campo de dicho nombre (equivalente al índice 0 del array). Ambas alternativas puntúan como válidas, siempre que no se mezclen respuestas de una y otra alternativas.

#### 1. Estructuras (0.6 puntos).

Se puntuó 0.06p por cada número (total 0.06 x 10 = 0.60p).

a) Alternativas: la que se pretendía redactar, y la que se acepta por el lapsus.

Tipo	Campo	Desplazamiento	Desplazamiento
long	Courses[0].CrsCod;	0	0
char	Courses[0].InstitutionalCrsCod[0/MAX...];	8	263
long	Courses[0].DegCod;	264	264
unsigned	Courses[0].Year;	272	272
unsigned	Courses[0].Status;	276	276
long	Courses[0].RequesterUsrCod;	280	280
unsigned	Courses[0].NumUsrs[0/10];	288	328
char	Courses[0].ShrtName[0/MAX...];	328	583
char	Courses[0].FullName[0/MAX...];	583	838
long	Courses[1].CrsCod;	840	840

b) 256

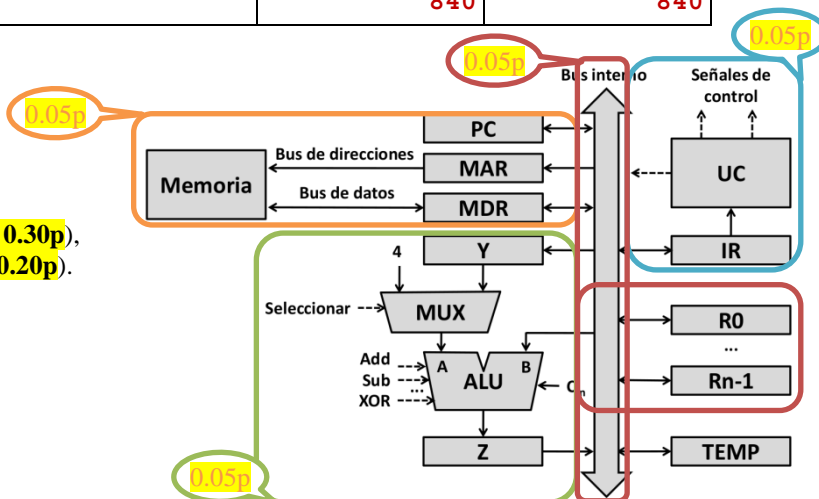
#### 2. Unidad de Control (0.5 puntos).

Se puntuó 0.05p por pseudo-instrucc. (total 0.05 x 6 = 0.30p),  
0.05p por cada zona dibujo (total 0.05 x 4 = 0.20p).

```

FETCH: MAR := PC; Z := PC+4
      MDR := M[MAR]; PC := Z
      IR := MBR
      goto f(IR)

```



### 3. Entrada/Salida (0.8 puntos).

Se puntúa si el programa funciona, o alternatively **0.1p** por cada sentencia (total **0.1 x 8 = 0.8p**).

```
#define SENSOR      A0
#define RED         6
#define GREEN       5
#define BLUE        3

void setup()
{
  pinMode(RED, OUTPUT);
  pinMode(GREEN, OUTPUT);
  pinMode(BLUE, OUTPUT);
}

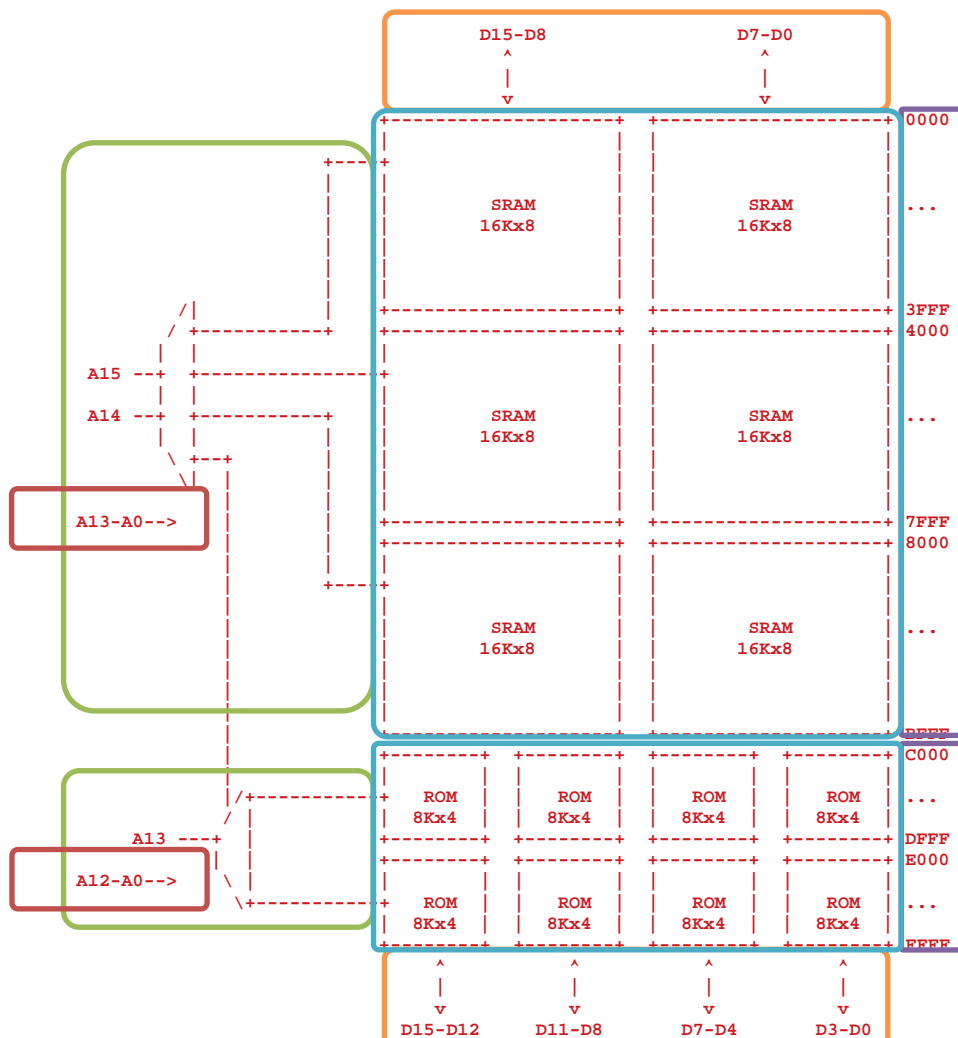
void loop()
{
  // leer el valor del sensor
  int sensorValue = analogRead(SENSOR);

  // mapear el valor del sensor (de 0 a 1023) a un valor de luz (de 0 a 255)
  int light = sensorValue / 4;

  // escribir el valor de luz en el LED RGB
  analogWrite(RED, light);
  analogWrite(GREEN, light);
  analogWrite(BLUE, light);
}
```

### 4. Diseño del sistema de memoria (0.5 puntos).

Se puntúa **0.05p** por cada zona dibujo (total **0.05 x 10 = 0.50p**).



## 5. Memoria cache (0.6 puntos).

Se puntúa **0.1p** por cada número (total **0.1 x 6 = 0.6p**).

MP: 1 TB =  **$2^{40}$  B**

L1-instrucciones: 64 KB =  $2^{16}$  B, 64 B/línea =  $2^6$  B/línea, 4 vías =  $2^2$  líneas/conjunto

L1-instrucciones:  $2^{16}$  B /  $2^6$  B/línea =  $2^{10}$  líneas,  $2^{10}$  líneas /  $2^2$  líneas/conjunto =  $2^8$  conjuntos

Dirección física de memoria principal desde el punto de vista de L1 (instrucciones):

etiqueta (26)	conjunto (8)	byte (6)
---------------	--------------	----------

Tamaño total en bits ocupado por las etiquetas en el directorio L1-instrucciones:

$2^{10}$  líneas · 26 bits/etiqueta =  $2^{10} \times 26$  bits = **26 624 bits**

Tamaño total en bits ocupado por las instrucciones en L1-instrucciones:

64KB/cache · 8 bits/B =  $2^{16} \times 2^3$  bits =  $2^{19}$  bits = **524 288 bits**

Etiquetas / Instrucciones =  $2^{10} \times 26 / 2^{19} = 26 / 2^9 =$  **5,078%**