DISEÑO DE PROCESADORES DEDICADOS

Práctica. No. 8

"Circuitos de Barrido y Escaneo: Teclado Matricial de Propósito General y Matriz de LEDs"

DR. JUAN CARLOS HERRERA LOZADA
jlozada@ipn.mx

Instituto Politécnico Nacional

Centro de Innovación y Desarrollo Tecnológico en Cómputo

CIDETEC

Mayo 2013

Campo 1: **Datos Personales.**

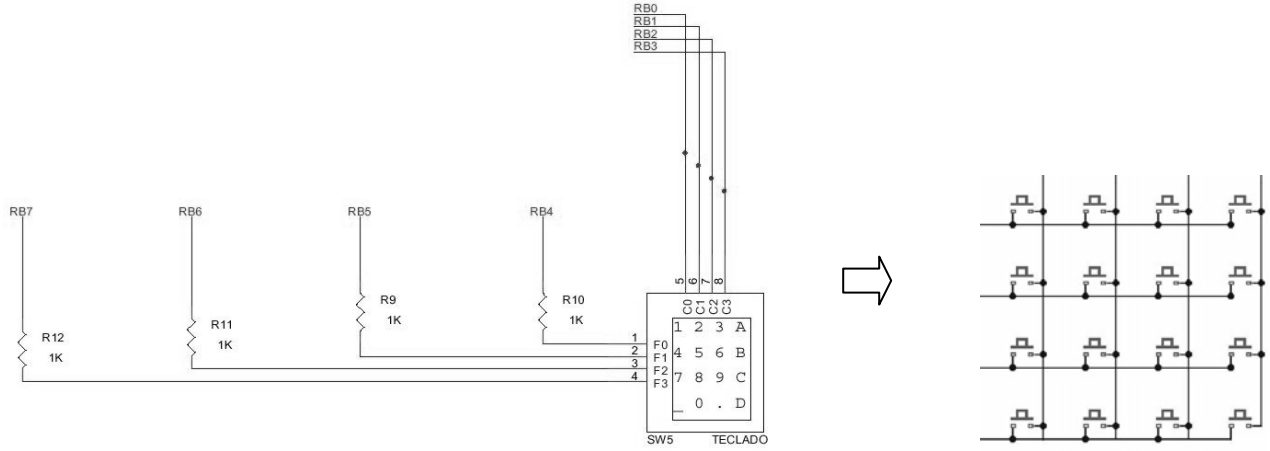
Campo 2: **Objetivos**.

- Diseñar interfaces sencillas para controlar circuitos de barrido y escaneo, en lo particular un teclado matricial y una matriz de LEDs.
- Crear macros reutilizables.
- Síntesis lógica en la tarjeta Spartan 3.

Campo 3: **Desarrollo de la Práctica**.

1. (2 puntos) Realiza una investigación básica sobre los tipos de teclados simples: *Punto Común y Matricial*. Consulta cómo funciona y cómo se construye un circuito antirrebotes y qué alternativas existen para ello (alternativa software y alternativa hardware). Recuerda que anteriormente ya se diseñó un antirrebotes para cumplir con la práctica de los motores a pasos, por lo que utilizarás este código para completar los diseños de esta práctica, donde aplique.

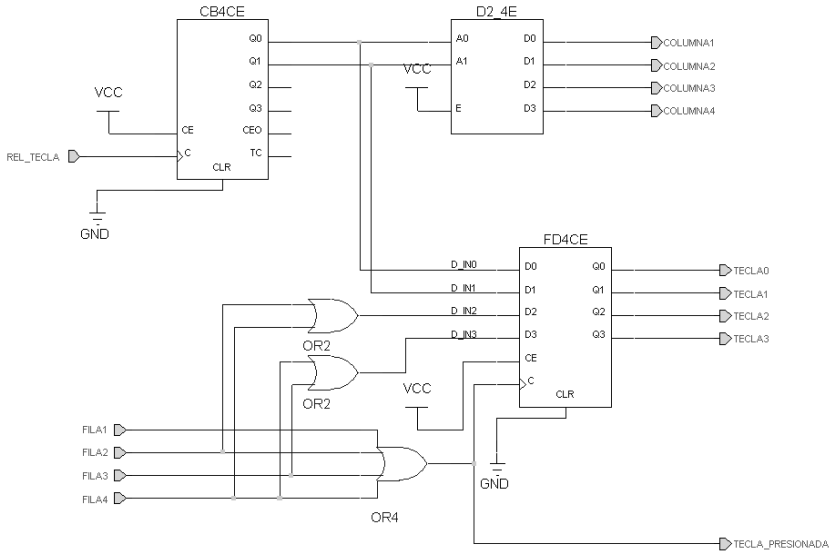
2. (4 puntos). Un teclado matricial, independientemente del número de teclas, se divide en filas y en columnas. Cuando una tecla específica se presiona, se une la columna con la fila respectiva en una coordenada única. El teclado más común es el de 4 x 4 (16 teclas) con 8 pines base: 4 para las columnas y 4 para las filas (en ocasiones, se encuentran teclados matriciales con más pines redundantes a un mismo pin base).



El circuito que controla el teclado matricial se divide en dos partes: el *circuito de escaneo*, que permite identificar qué tecla se presionó y el *circuito del encoder*, cuya función es codificar el valor binario de una tecla hacia una conversión particular (por ejemplo, un display a 7 segmentos). El algoritmo de escaneo más simple es utilizar un contador de anillo con un demultiplexor que "barre" cada una de las columnas (o filas) en un tiempo. El valor binario correspondiente a una tecla presionada se almacena en un registro de datos con salida hacia el módulo del encoder.

A continuación se anexa un diagrama simple que implementa el circuito de escaneo para un teclado matricial de 4 x 4, implementado en un FPGA a través de *ISE* de Xilinx, en alguna versión anterior de este Software. Nótese en la conexión final de la macro generada (segundo esquemático) que las filas son conectadas a resistencias de *Pull Down* disponibles en la Symbols Toolbox (valor "0" fijo) para evitar el corto circuito al conectar la fila con la columna.

Por principio, se debe capturar el esquemático siguiente para generar una macro reutilizable. El demultiplexor (D2_4E) se encuentra en la biblioteca de símbolos dentro de *Decoders* y barre el teclado colocando un "1" en cada columna, por lo que cuando se presiona una tecla, el valor lógico "1" se lee en cada fila.
El contador CB4CE mantiene la frecuencia de barrido, pudiendo ser adaptado a dos bits en vez de 4. *Es importante que coloques un divisor de tiempo con una frecuencia de aproximadamente 15 Khz previo a la entrada REL_TECLA. Es posible que el bloque divisor de tiempo esté dentro de la misma macro.*
En el diagrama se observa un registro de datos (FD4CE, que se obtiene de *FlipFlops* dentro de la biblioteca de símbolos disponibles) para garantizar que cuando una tecla sea presionada se almacene el dato encodificado. En la salida de la OR de 4 entradas, que a su vez es la entrada de reloj del registro de datos, podrás notar la presencia de una salida hacia pin físico denominada *TECLA_PRESIONADA*. Esta señal infiere simplemente que se ha presionado cualquier tecla, y será útil para conectar tu circuito de teclado a alguna aplicación que así lo requiera e indicarle que es posible leer un dato que ha sido entrado.

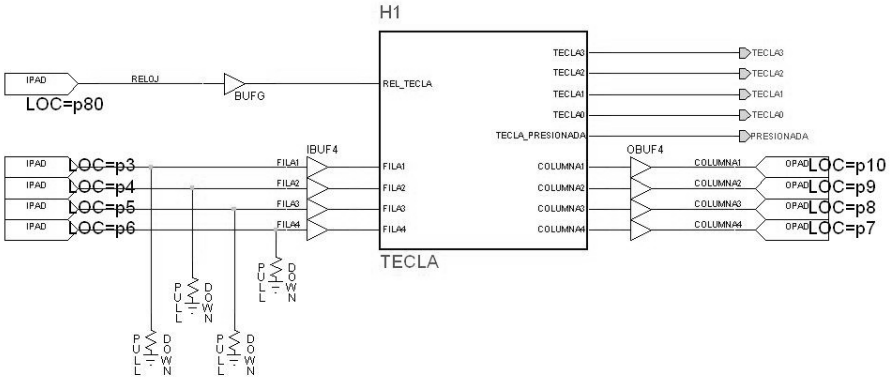


Una vez generada la macro, realiza las conexiones indicadas por el esquemático final. Las salidas TECLA3..TECLA0 deben entrar a un bloque decodificador que escriba en la LCD de la tarjeta. TECLA_PRESIONADA debe asignarse a un pin externo cualquiera, ya que sirve como salida de reloj de propósito general. Es necesario adecuar la asignación de pines, a la tarjeta que se esté utilizando en el curso.

Una aproximación al código en VHDL para lograr el controlador del teclado matricial, se muestra a continuación:

```
LIBRARY IEEE;
USE IEEE.std_logic_1164.ALL;
use IEEE.std_logic_arith.all;
use IEEE.std_logic_unsigned.all;
ENTITY teclas IS
  PORT(
         RELOJ, RESET: in STD_LOGIC;
         FILAS: in STD_LOGIC_VECTOR (3 downto 0);
         COLUMNAS: out STD_LOGIC_VECTOR (3 downto 0);
         TECLA_PRESIONADA: out STD_LOGIC;
         TECLAS_DECO:out STD_LOGIC_VECTOR (3 downto 0));
END teclas;

ARCHITECTURE funcionamiento_teclado OF teclas IS
signal COUNT: STD_LOGIC_VECTOR (1 downto 0);
signal AUX_TECLA_PRESS: STD_LOGIC;
signal DATOS_REG: STD_LOGIC_VECTOR (3 downto 0);
begin
contador: process (RELOJ, RESET)
begin
  if RESET='1' then
      COUNT <= "00";
   elsif RELOJ='1' and RELOJ'event then
         COUNT <= COUNT + 1;
  end if;
end process contador;
```

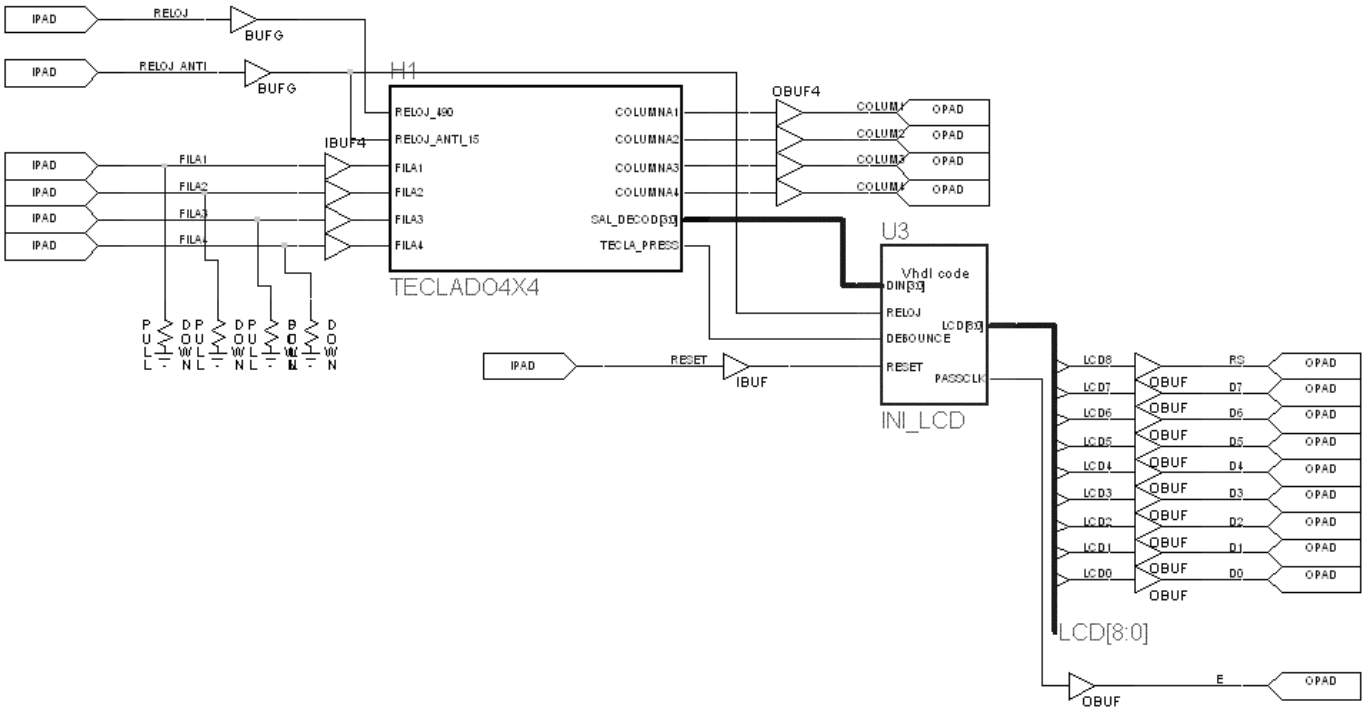
```
demux: process (COUNT)
begin
case COUNT is
when "00" => COLUMNAS <="1000"; --COLUMNA 1
when "01" => COLUMNAS <="0100"; --COLUMNA 2
when "10" => COLUMNAS <="0010"; --COLUMNA 3
when others => COLUMNAS <="0001"; --COLUMNA 4
end case;
end process demux;

tecla_press: process (FILAS)
begin
TECLA_PRESIONADA<=FILAS(3) OR FILAS(2) OR FILAS(1) OR FILAS(0);
AUX_TECLA_PRESS<=FILAS(3) OR FILAS(2) OR FILAS(1) OR FILAS(0);
end process tecla_press;

registro: process (FILAS,COUNT,AUX_TECLA_PRESS)
begin
case AUX_TECLA_PRESS is
when '1' =>
DATOS_REG(0)<=COUNT(0);
DATOS_REG(1)<=COUNT(1);
DATOS_REG(2)<=FILAS(1) OR FILAS(3);
DATOS_REG(3)<=FILAS(2) OR FILAS(3);
when others => DATOS_REG <= DATOS_REG;
end case;
TECLAS_DECO<=DATOS_REG;
end process registro;


END funcionamiento_teclado;
```



Descarga tu diseño en la tarjeta para comprobar su funcionamiento. Además de las resistencias de Pull Down, es *altamente recomendable que conectes unas resistencias externas de 4.7 KOhms para reforzar las anteriores*.

De manera tentativa puedes utilizar la macro del teclado matricial o alguna otra que controle y decodifique un teclado externo, mínimo de 4 teclas. La salida *Tecla _Presionada* del control del teclado, debe pasar a ser la señal de *Enable* una vez que se inicializó correctamente la LCD, así cada vez que se presione una tecla la señal *Tecla_Presionada* validará que se escriba el caracter en la pantalla. .Idealmente se requiere un anti rebotes (*debounce circuit*) que debes conectar a la salida de la macro que controla tu teclado para que ésta quede completa.

El siguiente esquemático muestra la conexión final. Una vez inicializada la memoria es posible pasar al modo de escritura. Obsérvese que la macro del teclado adiciona el antirrebotes. La señal de reloj de la LCD es la misma que la del antirrebotes (aproximadamente 15 Hz) y la frecuencia del trabajo del teclado es de 500 Hz.

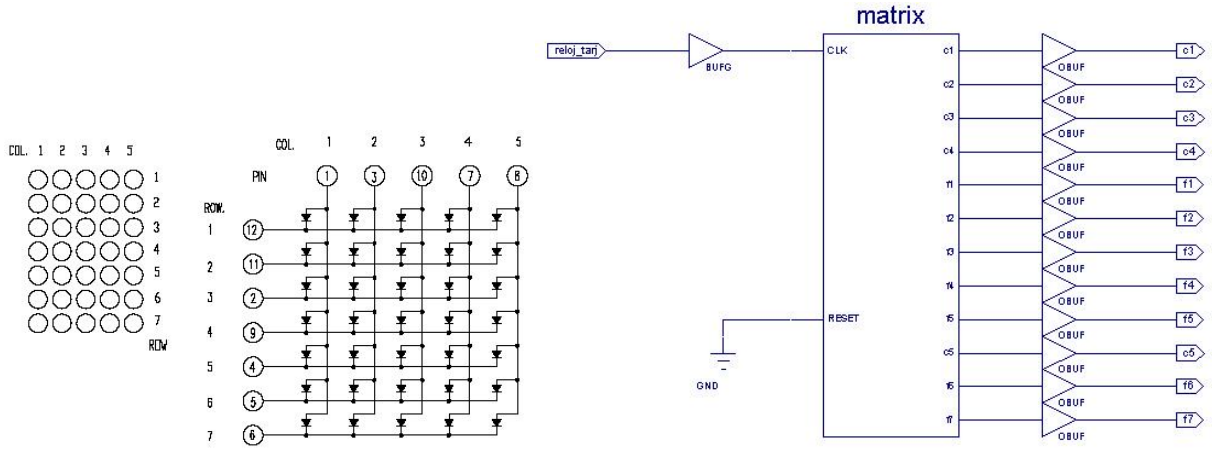


El siguiente código modela la macro en VHDL que inicializa y posteriormente acepta datos del teclado.

```
LIBRARY IEEE;
USE IEEE.std_logic_1164.ALL;
use IEEE.std_logic_arith.all;
use IEEE.std_logic_unsigned.all;
ENTITY ini_lcd IS
  PORT(
         RELOJ, RESET, DEBOUNCE: in STD_LOGIC; --PRUEBA
         DIN: in STD_LOGIC_VECTOR (3 downto 0);
         PASSCLK: out STD_LOGIC;       -- Salida del PIN Enable de la LCD
         lcd: out STD_LOGIC_VECTOR (8 downto 0));
  END ini_lcd;

ARCHITECTURE cont OF ini_lcd IS
signal CLK, hab: STD_LOGIC;
signal COUNT: STD_LOGIC_VECTOR (3 downto 0);
signal ESCRIBE: STD_LOGIC_VECTOR (3 downto 0);
signal COUNTT: STD_LOGIC_VECTOR (4 downto 0);

begin
contador: process (CLK, RESET)
begin
   if RESET='1' then
      COUNT <= "0000";
       elsif CLK='1' and CLK'event then
         if COUNT /= "1010" then
               COUNT <= COUNT + 1;
         else
               COUNT <= COUNT;
         end if;
    end if;
end process contador;

mux: process (COUNT, RELOJ, CLK, DEBOUNCE, DIN)
begin
-- Este MULTIPLEXOR permite deshabilitar el RELOJ para pasar a modo de espera de dato del teclado
if COUNT /= "1010" then
CLK <= RELOJ;
hab <= '0';
ESCRIBE <= COUNT;
else
CLK <= DEBOUNCE;
```

```
hab <= '1';
ESCRIBE <= DIN;
end if;
PASSCLK <= CLK;
end process mux;


pantalla: process (COUNTT, COUNT, hab, ESCRIBE)
begin
COUNTT(4) <= hab;
COUNTT(3 downto 0) <= ESCRIBE;
--      RS|D7|D6|D5|D4|D3|D2|D1|D0
--      4 |14|13|12|11|10|9 |8 |7     "Pines de la LCD"
--      RS es el MSB de la palabra y se conecta al pin 4 de la LCD.
--      D7 es el MSB del dato y se conecta en el pin 14 de la LCD, por lo que el LSB
--      D0 se conecta al pin 7 de la LCD.
case COUNTT is
when "00000" => lcd <="000111000"; --038
when "00001" => lcd <="000000110"; --006
when "00010" => lcd <="000001110"; --00E
when "00011" => lcd <="000000001"; --001     Clear Screen
when "00100" => lcd <="010000011"; --083     80 es la primera. Cuarta posicion
when "00101" => lcd <="101001000"; --148     "H"
when "00110" => lcd <="101101111"; --16F     "o"
when "00111" => lcd <="101101100"; --16C     "l"
when "01000" => lcd <="101100001"; --161     "a"
when "01001" => lcd <="100100001"; --121     "!"
when "01010" => lcd <="100100001"; --121     "!"
when "01011" => lcd <="100100001"; --121     "!"
when "01100" => lcd <="100100001"; --121     "!"
when "01101" => lcd <="100100001"; --121     "!"
when "01110" => lcd <="100010000"; --110     " "
when "01111" => lcd <="100010000"; --110     " "
when "10000" => lcd <="100110000"; --130     "0"
when "10001" => lcd <="100110001"; --131     "1"
when "10010" => lcd <="100110010"; --132     "2"
when "10011" => lcd <="100110011"; --133     "3"
when "10100" => lcd <="100110100"; --134     "4"
when "10101" => lcd <="100110101"; --135     "5"
when "10110" => lcd <="100110110"; --136     "6"
when "10111" => lcd <="100110111"; --137     "7"
when "11000" => lcd <="100111000"; --138     "8"
when "11001" => lcd <="100111001"; --139     "9"
when "11010" => lcd <="101000001"; --141     "A"
when "11011" => lcd <="101000010"; --142     "B"
when "11100" => lcd <="101000011"; --143     "C"
when "11101" => lcd <="101000100"; --144     "D"
when "11110" => lcd <="101000101"; --145     "E"
when "11111" => lcd <="101000110"; --146     "F"
when others => lcd <="100010000";  --110     " "
end case;
end process pantalla;
END cont;
```

3. **(4 puntos)** Refiriéndonos al caso de la matriz de LEDs de 35 puntos a razón de 5 columnas y 7 filas, y considerando el diagrama interno que se muestra en la figura siguiente (será muy importante realizar las adecuaciones necesarias acorde a la hoja de especificaciones del dispositivo que utilices para cumplir este numeral de la práctica), el código listado a continuación despliega una flecha vertical apuntando al norte y está descrito para un display que tiene el cátodo de los LEDs en las columnas (lógica contraria a la mostrada en el diagrama del arreglo de LEDs de la figura siguiente).
Observaciones importantes: para que el diseño funcione correctamente, se recomienda utilizar un Latch 74LS244 para sostener los datos enviados en filas y columnas del display; este c.i. requiere una alimentación nominal de 5V; podría usar transistores, para sustituir el Latch, en cada columna o fila.




```
-- Dot Matrix 5x7, despliega una flechita apuntado
hacia el norte
-- Editar UCF para ver pinout del diseño

LIBRARY IEEE;
USE IEEE.std_logic_1164.ALL;
use IEEE.std_logic_arith.all;
use IEEE.std_logic_unsigned.all;

ENTITY matrix IS
  PORT(
         CLK, RESET: in STD_LOGIC;
         c1, c2, c3, c4, c5, f1, f2, f3, f4,
f5,f6,f7: out STD_LOGIC
          );
  END matrix;

ARCHITECTURE archmatrix OF matrix IS
signal frec_alta, frec_lenta: STD_LOGIC;
signal COUNT: STD_LOGIC_VECTOR (23 downto 0);
signal COUNTB: STD_LOGIC_VECTOR (2 downto 0);
signal matrix_led: STD_LOGIC_VECTOR (11 downto 0);
begin
divisor: process (CLK, RESET)
begin
   if RESET='1' then
      COUNT <= "000000000000000000000000";
   elsif CLK='1' and CLK'event then
              COUNT <= COUNT + 1;
        else
              COUNT <= COUNT;
   end if;
       frec_alta <= COUNT(19);
       frec_lenta <= COUNT(23);
end process divisor;

contabin: process (frec_alta, RESET)
begin
   if (RESET='1' or COUNTB= 5) then
      COUNTB <= "000";
   elsif frec_alta='1' and frec_alta'event then
              COUNTB <= COUNTB + 1;
                              else   COUNTB <=
COUNTB;
   end if;
end process contabin;

decomatrix: process (COUNTB)
begin
--  CONTB(2:0) -> c1|c2|c3|c4|c5|f1|f2|f3|f4|f5
case COUNTB is
when "000" => matrix_led <="011110010000"; --
barrido c1
when "001" => matrix_led <="101110111111"; --c2
when "010" => matrix_led <="110111111111"; --c3
when "011" => matrix_led <="111010111111"; --c4
when "100" => matrix_led <="111100010000"; --c5
when others => matrix_led <="111110000000"; --
apagado
end case;
end process decomatrix;

salida_individual: process (matrix_led)
begin
c1 <= matrix_led(11);
c2 <= matrix_led(10);
c3 <= matrix_led(9);
c4 <= matrix_led(8);
c5 <= matrix_led(7);
f1 <= matrix_led(6);
f2 <= matrix_led(5);
f3 <= matrix_led(4);
f4 <= matrix_led(3);
f5 <= matrix_led(2);
f6 <= matrix_led(1);
f7 <= matrix_led(0);
end process salida_individual;

END archmatrix;
```

Modifica el código VHDL para realizar una animación en un display de matriz de LEDs. Podría ser que la flecha se desplace de abajo hacia arriba, o bien, podría definirse otra figura a criterio del diseñador que muestre un movimiento. Recuerda que para formar el movimiento de un solo caracter, requieres realizar varios caracteres que en conjunto simulen el movimiento sincronizado del primero.

Campo 4: **Conclusiones individuales.**