NT1004

3.4 INTERFASE A SENSORES Y MOTORES LEGO CON MC68HC08

PARA MICROCONTROLADORES HC08, ELEMENTOS DEL "KIT MINDSTORMS"

Preparado por: Rangel Alvarado
Estudiante Graduando de Lic. en Ing. Electromecánica
Panamá, Panamá
"e-mail": issaiass@cwpanama.net
"web site": http://www.geocities.com/issaiass/

ÍNDICE

3.4.1 Introducción

Hasta el momento hemos utilizado equipo didáctico de la compañía Jameco para trabajar experimentos básicos, pero también se puede trabajar con el "Kit" de Robótica LEGO – Mindstorms[1] el cual trae consigo instructivos multimedia de cómo iniciarse en la robótica, además la mecánica de los elementos encaja perfectamente lo que reduce el tiempo de implementación de "hardware", esto favorece al momento de construir un pequeño prototipo de evaluación.

El tutorial abarca como construir un "explorador" utilizando piezas del "Kit Mindstorms" más el "kit" del microcontrolador 68HC908JK3/JL3 como:

- Motores DC del "Kit Mindstorms": Motores de corriente continua de nueve voltios (9V) que proporcionan la fuerza motriz del "explorador"
- Sensores de Tacto del "Kit Mindstors": Se utilizan como simples contactos abiertos para determinar.
- La tarjeta de desarrollo JK3 ó JL3: Es el elemento de control o cerebro del experimento, esta envía señales al circuito de control del motor para la dirección dependiendo del estado de los sensores de tacto.

Nota: Ver en las Referencias, Sección de "Vínculos LEGO" para información sobre programación en Visual Basic, C++ y Java, además de sitios de interés del "Kit Mindstorms"; además ver Apéndice P, sección de Proyecto – Fase LEGO.

[1] http.//www.legomindstorms.com/

3.4.2 Equipo y Materiales

1. Pinza de Corte Diagonal
2. Cautín de 25 Watts con punta fina de 1/16”
3. Estaño 60/40 con resina en el centro - para trabajo electrónico
4. Alambre AWG 20
5. Pelador de Alambres
6. Un (1) pedazo de papel aluminio
7. Seis (6) baterías AA de 1.5 V cada una
8. Gutapercha (cinta adhesiva aislante)
9. Una (1) fuente de poder
10. Tarjeta de desarrollo TD68HC908JK3
11. Dos (2) Motores DC del “Kit LEGO Mindstorms”
12. Dos (2) Sensores de Toque del “Kit LEGO Mindstorms”
13. Cuatro (4) Conectores adicionales del “Kit de LEGO Mindstorms”[2]
14. Resistores: 4 × 3.6kΩ y 4 × 56Ω (*Nota: el dibujo no corresponde al listado*)
15. Transistores: 8 × 2n2222 y 4 × 2n2907
16. Diodos: 4 × 1n4001
17. “Constructopedia” y piezas del “Kit LEGO Mindstorms”
18. Dos (2) “Protoboards” pequeños – Jameco No. 20600.

Figura 174. Equipo para Construcción del “Explorador”

[2] Comprar en http://www.pldstore.com/pld/finditem.cfm?itemid=1120

3.4.3 Prototipo

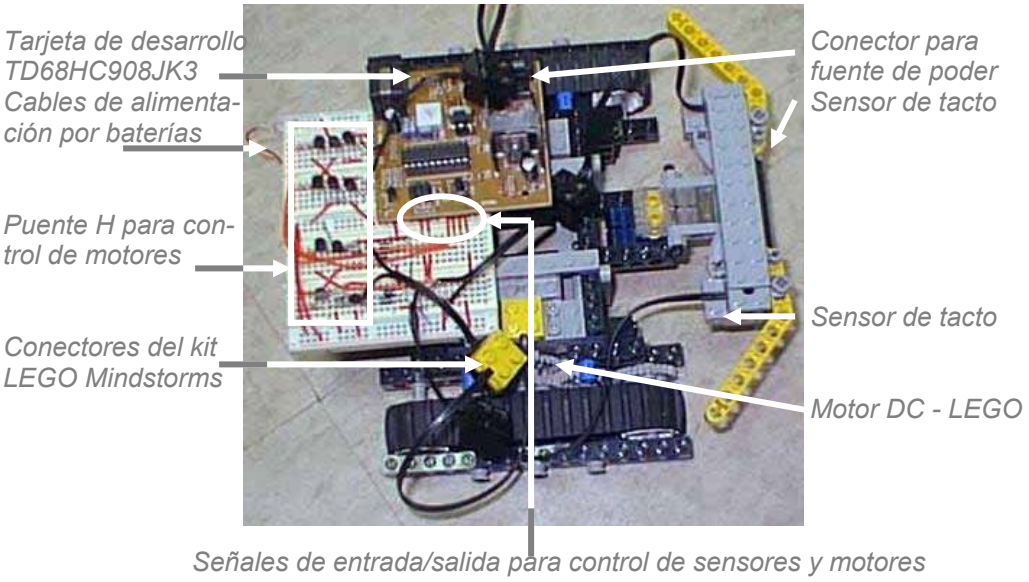

Figura 175. Interfase LEGO Mindstorms – TD68HC908JK3. El robot utiliza toda la mecánica del "Kit LEGO Mindstorms", de esta forma solo se enfoca en la electrónica, control y software del pequeño robot. Puede recorrer terrenos a nivel, y por medio de los sensores de toque evita obstáculos retrocediendo y dando un giro a la izquierda o a la derecha..

3.4.4 Construcción

En esta sección refiérase a las figuras 175 y 176.

Figura 176. Conector LEGO. (a) Cable del "Kit Mindstorms". Corte por la mitad para dos (2) cables. (b) Cable seccionado. Seccione, desnude y suelde un alambre a sus extremos para poder insertarlos en el "protoboard".

(a) Seccione en dos partes el conector de LEGO, obteniendo cuatro (4) mitades iguales.

(b) Corte por la mitad caca cable a ±1 cm, desnude sus puntas.

(c) Suelde un pedazo de cable desnudo a cada punta (8 en total).

(d) Arme el circuito de la figura 178(a) dos veces y conéctelo a la tarjeta de desarrollo según la figura 177.

(e) Asegúrese de haber conectado los sensores de tacto.

(f) Arme el circuito de batería según la figura 178.

(g) Arme el soporte de oruga que se encuentra en la "constructopedia" (libro de trabajo del "Kit Mindstorms").

(h) Consultar la NT1003 para bajar el programa al microcontrolador.

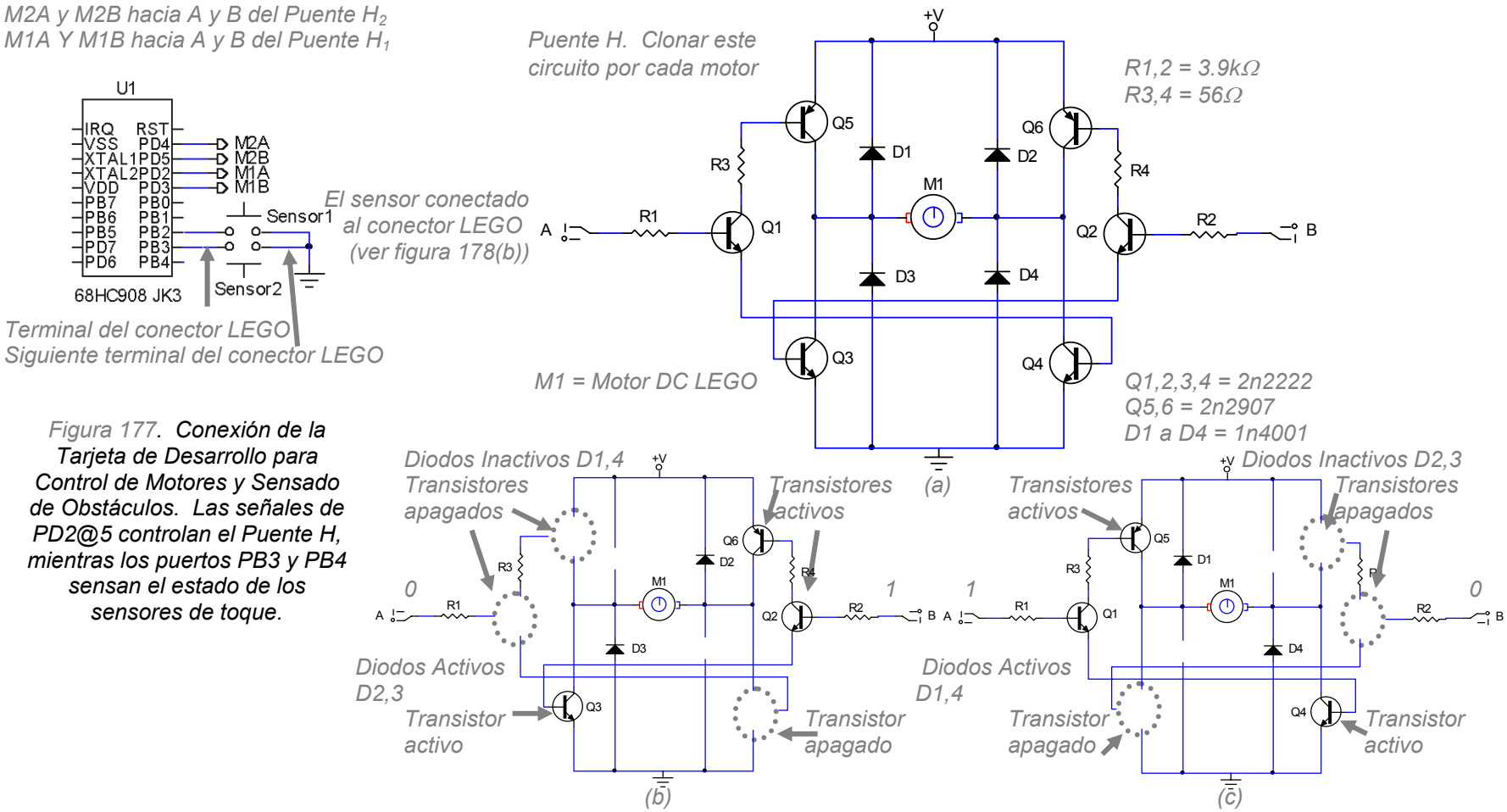

Figura 177. Conexión de la Tarjeta de Desarrollo para Control de Motores y Sensado de Obstáculos. Las señales de PD2@5 controlan el Puente H, mientras los puertos PB3 y PB4 sensan el estado de los sensores de toque.

Figura 178. Circuito de Control Bidireccional de Motores. (a) Puente H. Permite el control bidireccional. (b) Giro en sentido horario. Transistores Q2,3 y 6 activos. (c) Giro en sentido antihorario. Transistores Q1,4 y 5 activos.

(a)

(b)

(c)

Enrolle el resto de las baterías con gutapercha según los pasos (a), (b) y (c).

(e)

(f)

En la parte opuesta solo coloque papel aluminio en grupos de dos a las intermedias, y coloque cables en los extremos.

(g)

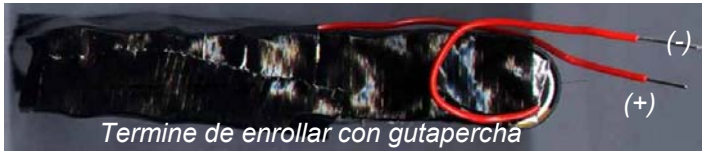

Figura 179. Batería de 9V a base de 1.5V. (a) Polos opuestos. Colóquelas con los polos opuestos, es más fácil unir para conseguir 9V. (b) Unión. Enrolle la primera batería y luego la siguiente. (c) Siguiente secuencia de unión. Enrolle las siguientes dos, luego las siguientes tres y así sucesivamente. (d) Asegure las seis (6) baterías. Enrolle el final con gutapercha hasta conseguir firmeza. (e) Una los polos opuestos. En un lado una los polos opuestos de las baterías. (f) Una los polos intermedios del lado contrario. Una los polos intermedios y en los extremos coloque cables que hagan contacto con las baterías., desnude sus extremos. (g) Batería de 9V. Enrolle con gutapercha para que el papel aluminio no se caiga de los polos de la batería.

3.4.5 Diagrama de Flujo

El siguiente programa hace que el "explorador" se mueva hacia delante indefinidamente hasta encontrar un obstáculo.

(a) Obstáculo del sensor 1: retrocede, gira a la izquierda y avanza nuevamente.
(b) Obstáculo del sensor 2: retrocede, gira a la derecha y avanza nuevamente.

Programa Principal de movimientos básicos

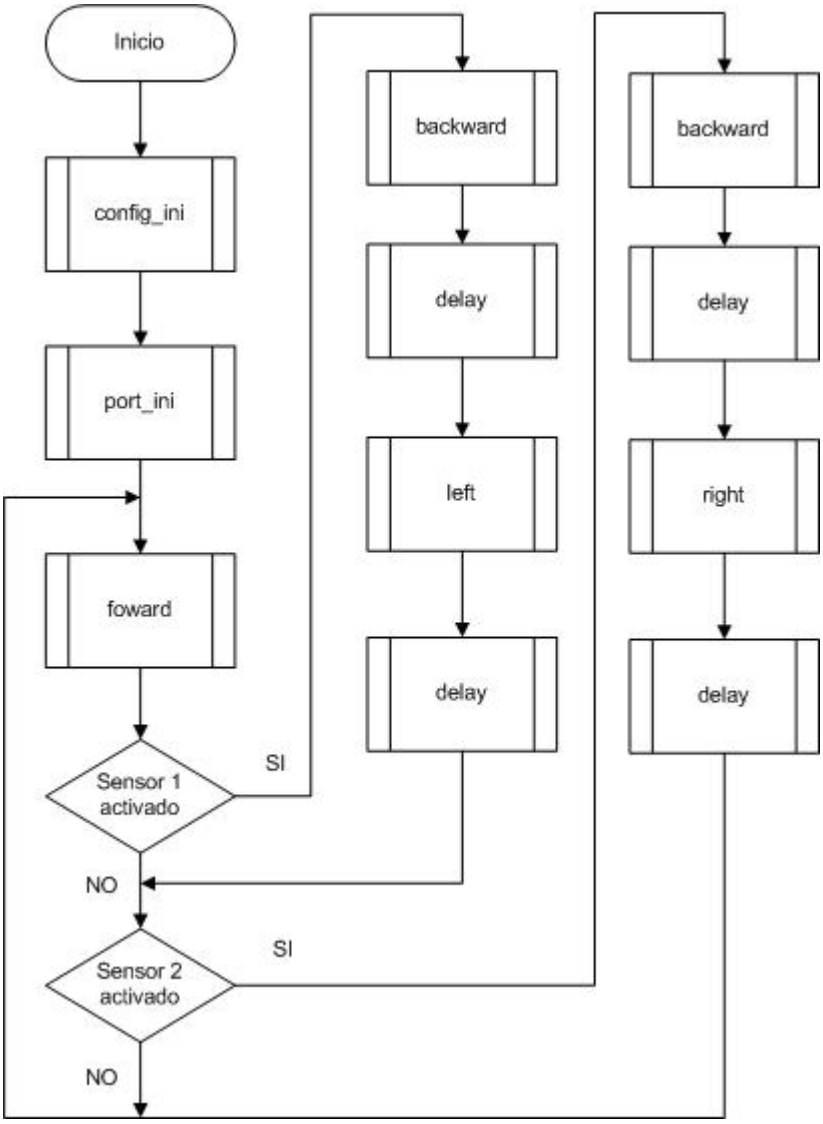

Figura 180. Diagrama de flujo de NT1004. Programa Principal, realiza movimientos básicos de avance, retroceso, giro a la izquierda y a la derecha con el "explorador".

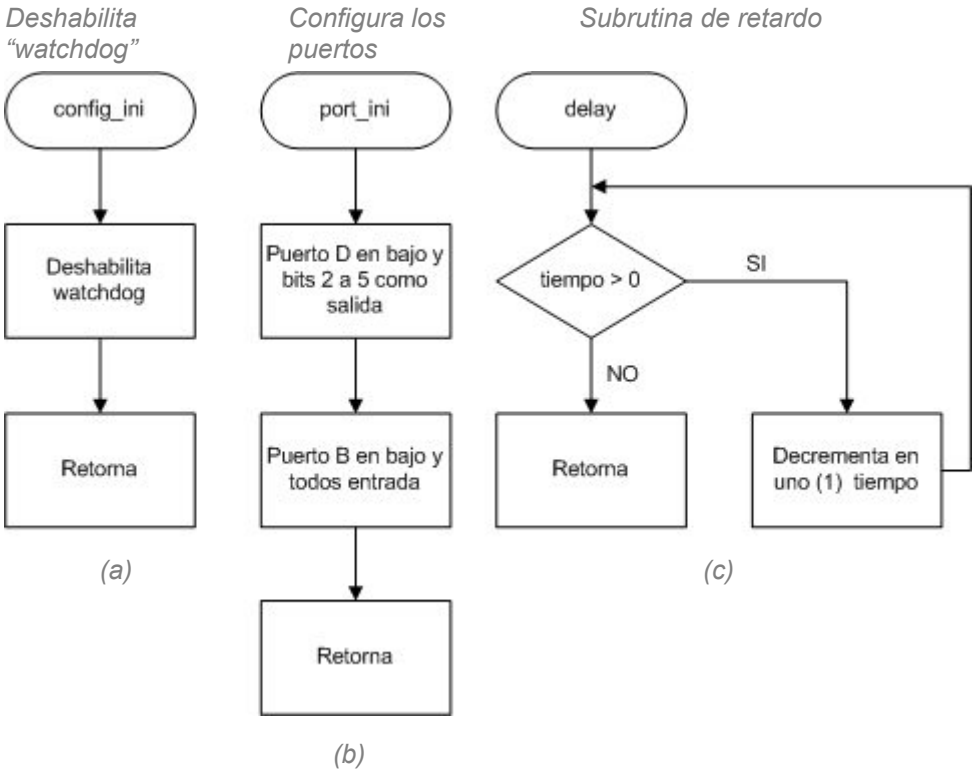

Figura 181. Subrutinas de Configuración. (a) Configuración del sistema. Deshabilita el "watchgdog". (b) Inicializa Puertos. Activa el Puerto D para señales de control de motores y el Puerto B como entrada para sensores voléanos de tacto. (c) Retardo. Demora de tiempo configurable de alrededor de un (1) segundo o menos.

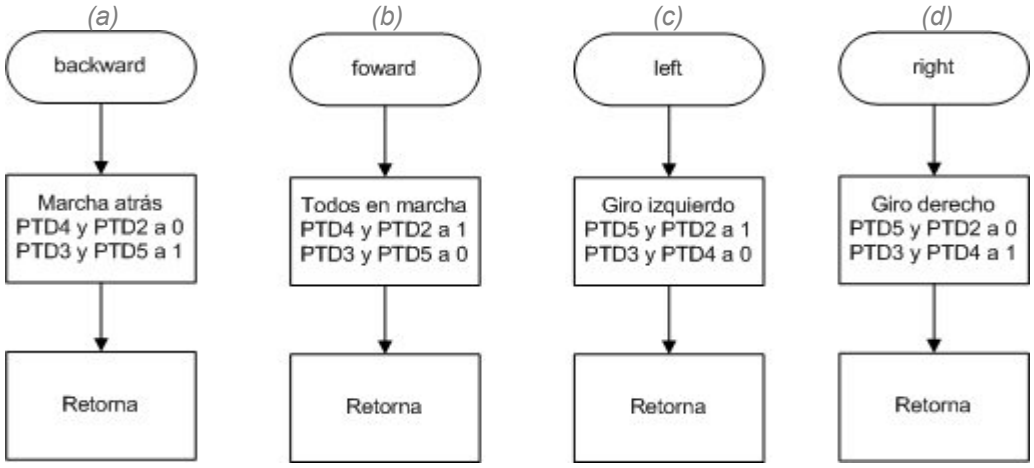

Figura 182. Funciones de Movimiento. (a) Movimiento hacia atrás. (b) Movimiento hacia delante. (c) Movimiento hacia la izquierda. (d) Movimiento hacia la derecha.

3.4.6 Código

```
/*****************************************************************************
 FILE:          main.c
 PURPOSE:   Mueve motores hacia adelante
            - Si se presiona el sensor izquierdo...
                    -Retrocede, espera, gira a la derecha
                     y espera nuevamente
                    -Avanza
            - Si se presiona el sensor derecho...
                    -Retrocede, espera, gira a la izquierda
                     y espera nuevamente
 LANGUAGE:  ANSI-C
 ---------------------------------------------------------------------------
  HISTORY
  DD MM YY
  16  11  03   Created.
  29  01  04   Last Modification.
 *****************************************************************************/
#include "HC08.H"      /* Definición de los puertos y registros del Microcontrolador */
#include "FUNCT.H"     /* Funciones de Motores, sistema, retardo */
#pragma DATA_SEG SHORT _DATA_ZEROPAGE     /* Inicio de la Página 0 */
volatile unsigned int *time;                        /* Apuntador a -> tiempo */

void main (void)                    /* Programa Principal */
{
unsigned int time = 65535;          /* Variable Tiempo = 0xFFFF */
config_ini ( );                     /* Deshabilita el watchdog */
port_ini ( );               /* PTD[2:5] para motores, PTA como entradas (por defecto) */
for (;;)                    /* Repite indefinidamente */
        {foward( );         /* Motores en Marcha Hacia Adelante */
        if (SENSOR1==0)         /* Si Sensor de Toque 1 = 0 */
                {
                backward( );    /* Atrás */
                delay(time);    /* Espera un tiempo */
                left( );        /* Izquierda */
                delay(time);    /* Espera un tiempo */
                }
        if (SENSOR2==0)         /* Si Sensor de Toque 2 = 0 */
                {
                backward( );    /* Atrás */
                delay(time);    /* Espera un tiempo */
                right( );       /* Derecha */
                delay(time);    /* Espera un tiempo */
                }
        }
}
```

Listado 28. main.c – NT1004 – LEGOMCU. Archivo principal que realiza el movimiento de motores y giro si encuentra un obstáculo dependiendo del sensor que haya sido activado.

```
/*****************************************************************************
 FILE:          HC08.h
 PURPOSE:    Define macros de programa.
 LANGUAGE:  ANSI-C
 ----------------------------------------------------------------------------
  HISTORY
  DD MM YY
  16  11  03   Created.
  29  01  04   Last Modification.
*****************************************************************************/
/*********************************************************
 MACRO NAME:        PORTB
 PURPOSE:           Define dirección de Entra-
                    da/Salida del Puerto B
 INPUT:             None
 OUTPUT:            None
 UTILITY:           main.c
*********************************************************/
#define PORTB   *(volatile unsigned char *) 0x0001
        /* Definición del Registro del Puerto B */
/*********************************************************
 MACRO NAME:        PORTD
 PURPOSE:           Define dirección de Entra-
                    da/Salida del Puerto D
 INPUT:             None
 OUTPUT:            None
 UTILITY:           main.c
*********************************************************/
#define PORTD   *(volatile unsigned char *) 0x0003
        /* Definición del Registro del Puerto D */

/*********************************************************
 MACRO NAME:        DDRB
 PURPOSE:           Define registro de direc-
                    cionamiento del Puerto B
 INPUT:             None
 OUTPUT:            None
 UTILITY:           main.c
*********************************************************/
#define DDRB  *(volatile unsigned char *) 0x0005
        /* Definición del Registro de Direccionamiento del Puerto B */
```

```
/*************************************************************
  MACRO NAME:       DDRD
  PURPOSE:          Define dirección de Entra-
                    da/Salida del Puerto D
  INPUT:            None
  OUTPUT:           None
  UTILITY:          main.c
*************************************************************/
#define DDRD  *(volatile unsigned char *) 0x0007
        /* Definición del Registro de Direccionamiento del Puerto D */

/*************************************************************
  MACRO NAME:       CONFIG1
  PURPOSE:          Define dirección de registro
                    de configuración no. 1.
  INPUT:            None
  OUTPUT:           None
  UTILITY:          main.c
*************************************************************/
#define CONFIG1   *(volatile unsigned char *) 0x001F
        /* Definición del Registro de Configuraciones Número 1 */

/*************************************************************
  MACRO NAME:       SENSOR1
  PURPOSE:          Define el bit 3 del Puerto B
                    como elemento sensor de
                    toque no. 1.
  INPUT:            None
  OUTPUT:           None
  UTILITY:          main.c
*************************************************************/
#define SENSOR1 (PORTB & 0x04)
        /* Máscara de PTB3 */

/*************************************************************
  MACRO NAME:       DDRD
  PURPOSE:          Define el bit 4 del Puerto B
                    como elemento sensor de
                    toque no. 2.
  INPUT:            None
  OUTPUT:           None
  UTILITY:          main.c
*************************************************************/
#define SENSOR2 (PORTB & 0x08)
        /* Máscara de PTB4 */
```

Listado 29. HC08.h – NT100 – LEGOMCU. Define macros que se utilizan en el programa main.c, p.e., SENSOR 2 como entrada booleana de sensado de obstáculo presente.

```
/*****************************************************************************
 FILE:            FUNCT.h
 PURPOSE:    Declaración de funciones.
 FUNCTIONS: config_ini, port_ini, foward, backward, left, right, delay
 LANGUAGE:  ANSI-C
 ---------------------------------------------------------------------------
  HISTORY
  DD MM YY
  16   11   03   Created.
  29   01   04   Last Modification.
*****************************************************************************/
/*****************************************************
 FUNCTION NAME:     config_ini
 PURPOSE:           Inicializa el registro
                    CONFIG1.
 INPUT:             None
 OTHER VARIABLES:  None
 OUTPUT:            None
 UTILITY:           main.c
*****************************************************/
void config_ini (void);    /* Inicialización de registro de configuración */

/*****************************************************
 FUNCTION NAME:     port_ini
 PURPOSE:           Inicializa puertos de
                    entrada y de salida.
 INPUT:             None
 OTHER VARIABLES:  None
 OUTPUT:            None
 UTILITY:           main.c
*****************************************************/
void port_ini (void);      /* Inicialización de puertos de entrada/salida */

/*****************************************************
 FUNCTION NAME:     foward
 PURPOSE:           Motor hacia adelante.
                    PTD5 = PTD3 = 0
                    PTD4 = PTD2 = 1
 INPUT:             None
 OTHER VARIABLES:  None
 OUTPUT:            None
 UTILITY:           main.c
*****************************************************/
void foward (void);        /* Hacia adelante */
```

```
/*****************************************************
  FUNCTION NAME:    backward
  PURPOSE:          Motor hacia atrás.
                    PTD4 = PTD2 = 0
                    PTD5 = PTD3 = 1
  INPUT:            None
  OTHER VARIABLES:  None
  OUTPUT:           None
  UTILITY:          main.c
*****************************************************/
void backward (void);   /* Hacia atrás */

/*****************************************************
  FUNCTION NAME:    left
  PURPOSE:          Motor hacia la
                    izquierda.
                    PTD5 = PTD2 = 1
                    PTD4 = PTD3 = 0
  INPUT:            None
  OTHER VARIABLES:  None
  OUTPUT:           None
  UTILITY:          main.c
*****************************************************/
void left (void);       /* A la izquierda */

/*****************************************************
  FUNCTION NAME:    right
  PURPOSE:          Motor hacia la
                    derecha.
                    PTD3 = PTD4 = 1
                    PTD5 = PTD2 = 0
  INPUT:            None
  OTHER VARIABLES:  None
  OUTPUT:           None
  UTILITY:          main.c
*****************************************************/
void right (void);      /* A la derecha */

/*****************************************************
  FUNCTION NAME:    delay
  PURPOSE:          Retardo de tiempo
  INPUT:            (unsigned int) time
  OTHER VARIABLES:  None
  OUTPUT:           None
  UTILITY:          main.c
*****************************************************/
void delay (unsigned int time);
```

Listado 30. FUNCT.h – NT1004 – LEGOMCU. Declara las funciones que se llaman en el transcurso de la ejecución del programa principal, p.e. el retardo de tiempo de giro "delay".

```
/*****************************************************************************
 FILE:            FUNCT.c
 PURPOSE:    Definición de funciones.
 FUNCTIONS: delay
 LANGUAGE:  ANSI-C
 ----------------------------------------------------------------------------
 HISTORY
  DD MM YY
  16   11   03   Created
  29    01  04   Last Modification.
*****************************************************************************/
#include <HC08.H>      /* Definición de los puertos y registros del Microcontrolador */

/*****************************************************
 FUNCTION NAME:    config_ini
 PURPOSE:             Inicializa el registro
                           CONFIG1.
 INPUT:                 None
 OTHER VARIABLES:  None
 OUTPUT:              None
 UTILITY:              main.c
*****************************************************/
void config_ini (void)    /* Inicializa el registro de configuración número 1 */
{
CONFIG1 = 0x01;       /* Deshabilita Watchdog */
}

/*****************************************************
 FUNCTION NAME:    port_ini
 PURPOSE:             Inicializa puertos de
                           entrada y de salida.
 INPUT:                 None
 OTHER VARIABLES:  None
 OUTPUT:              None
 UTILITY:              main.c
*****************************************************/
void port_ini (void)      /* Inicializa puertos */
{
PORTD = 0;            /* Inicializa el Puerto D */
DDRD = 0x3C;          /* Puertos D[2 a 5] como salida */
PORTB = 0;            /* Inicializa el Puerto B */
DDRB = 0;             /* Entradas */
}
```

```
/*****************************************************
  FUNCTION NAME:    foward
  PURPOSE:          Motor hacia adelante.
                    PTD5 = PTD3 = 0
                    PTD4 = PTD2 = 1
  INPUT:            None
  OTHER VARIABLES:  None
  OUTPUT:           None
  UTILITY:                  main.c
*****************************************************/
void foward (void)      /* Motores hacia adelante */
{
PORTD = 0x14;           /* Todos los motores hacia adelante */
}

/*****************************************************
  FUNCTION NAME:    backward
  PURPOSE:          Motor hacia atrás.
                    PTD4 = PTD2 = 0
                    PTD5 = PTD3 = 1
  INPUT:            None
  OTHER VARIABLES:  None
  OUTPUT:           None
  UTILITY:          main.c
*****************************************************/
void backward (void)    /* Motores hacia atrás */
{
PORTD = 0x28;           /* Todos los motores hacia atrás */
}

/*****************************************************
  FUNCTION NAME:    left
  PURPOSE:          Motor hacia la izquier-
                    da.
                    PTD5 = PTD2 = 1
                    PTD4 = PTD3 = 0
  INPUT:            None
  OTHER VARIABLES:  None
  OUTPUT:           None
  UTILITY:          main.c
*****************************************************/
void left (void)        /* Motor 1 hacia adelante, Motor 2 hacia atrás */
{
PORTD = 0x24;           /* Giro a la izquierda */
}
```

```
/*****************************************************
  FUNCTION NAME:     right
  PURPOSE:           Motor hacia la dere-
                     cha.
                     PTD3 = PTD4 = 1
                     PTD5 = PTD2 = 0
  INPUT:             None
  OTHER VARIABLES:   None
  OUTPUT:            None
  UTILITY:           main.c
*****************************************************/
void right (void)          /* Motor 1 hacia atrás, Motor 2 hacia adelante */
{
PORTD = 0x18;              /* Giro a la derecha */
}

/*****************************************************
  FUNCTION NAME:     delay
  PURPOSE:           Retardo de tiempo

  INPUT:             (unsigned int) time
  OTHER VARIABLES:   None
  OUTPUT:            None
  UTILITY:           main.c
*****************************************************/
void delay (unsigned int time)
{
while ((time)>0)           /* Retardo de tiempo */
        {
        time--;            /* Decrementa la variable */
        }
}
```

Listado 31. FUNCT.c – NT1004 – LEGOMCU. Declara las funciones que se llaman en el transcurso de main.c, p.e., right ( ) es un giro a la izquierda invirtiendo el sentido de los motores.