eman ta zabal zazu
Universidad del País Vasco
Euskal Herriko Unibertsitatea
BILBOKO INGENIARITZA ESKOLA
ESCUELA DE INGENIERÍA DE BILBAO

Arquitectura de Computadores

Práctica Final de laboratorio - DRON

2023-2024

...

Titulación:

Grado en Informática de Gestión y Sistemas de Información

...

2º Curso (1º Cuatrimestre)

...

Unai Bermúdez Osaba

17 de diciembre de 2023

Índice

Índice de figuras

Índice de cuadros

1. Introducción

En este informe se recoge el trabajo desarrollado relativo a la **Practica Final** de la asignatura de **Arquitectura de Computadores.**

El objetivo principal ha sido el **desarrollo de software** para **microcontroladores**. Por ello, se ha utilizado el microcontrolador **80C552** del fabricante **Philips Semiconductors.** Se ha llevado a cabo en el entorno de simulación **Keil µVision2.**

Se han trabajado los conceptos de *Interrupciones*, *ADC (Analogic-Digital Converter)* para los sensores de altura, distancia frontal y nivel de batería, *PWM (Pulse Width Modulation)* para en control de los motores, *Timers* para controlar los tiempos, Pulsadores, Leds y Display.

La temática de la practica ha sido la realización de un drone ideal. Su descripción se encuentra en el enunciado facilitado por el profesorado.

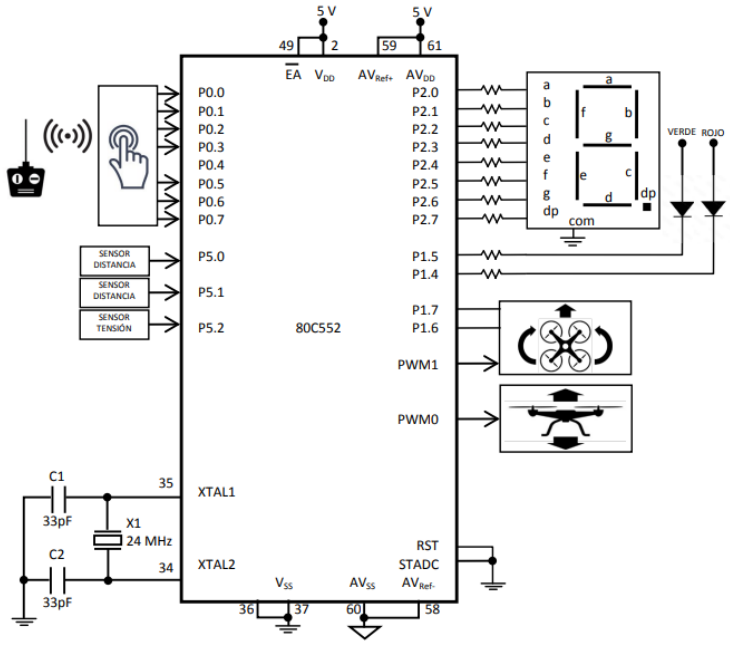

Figura 1: Esquema electrónico del sistema de control indicado.

Con el fin de lograr el completo desarrollo de la aplicación, en primer lugar se han diseñado el **diagrama de estados, eventos y acciones** y los **diagramas de flujo** correspondientes.

En cuanto a las *Interrupciones*, se han programado las interrupciones de *over-flow* del **Timer 0** y por **fin de conversión** del **ADC**.

El botón de **Despegue/Aterrizaje (P0.7)**, el botón de **Paro/Avanza (P0.6)** y los selectores del control manual **g.izda, g.dcha, sube, baja.(P0.3, P0.2, P0.1, P0.0, respectivamente)** se controlan mediante **flanco**.

Se ha utilizado el mismo ADC para el **sensor de altura (canal 0)**, **sensor de distancia frontal (canal 1)** y **sensor de tensión de la batería (canal 3)**, con este conversor se han convertido los valores analógicos leídos a valores digitales.

También de ha utilizado un *DAC (Digital-Analogic Converter)*, en concreto un **PWM** para controlar la potencia que entregan los motores y de que modo lo hacen. El **PWM0** es el encargado de controlar la altura del drone, mediante el *Duty Cicle* se elige si queremos que el drone este parado, baje, mantenga altura o suba. Por otra parte el modulo de **PWM1** controla si el drone está parado o gira/avanza también mediante el *Duty Cicle*. Si el PWM1 esta en modo gira/avanza, los **bits 6 y 7 del puerto 1** controlan si esta parado, avanza, gira a la derecha o gira a la izquierda.

Para mostrar por el ***Display*** el número deseado, se han codificado los valores mediante la instrucción MOVC. Se ha seguido el mismo procedimiento para el PWM, y se explica en sus respectivos apartados de esta memoria

2. Diagrama de estados, eventos y acciones

Se ha desarrollado una solución con 9 estados, 12 eventos (incluyendo el evento 0) y 16 acciones.

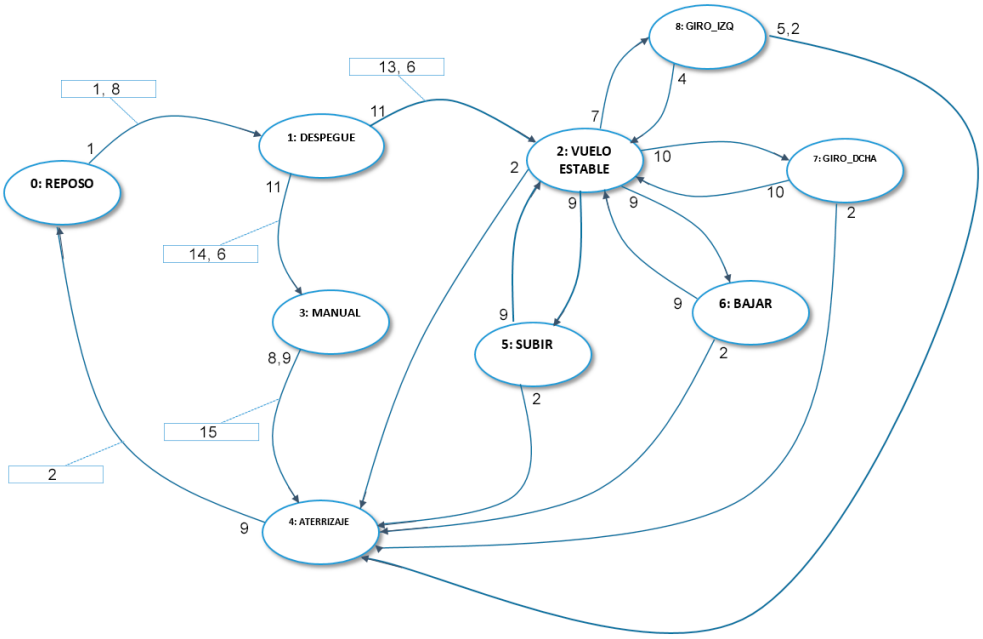

Figura 2: Diagrama de estados, eventos y acciones

2.1. Estados

Se han implementado los estados de **Reposo** (estado 0), **Despegue** (estado 1), **Vuelo Estable** (estado 2), **Manual** (estado 3), **Aterrizaje** (estado 4), **Sube** (estado 5), **Baja** (estado 6), **Giro_dcha** (estado 7) y **Giro_izq** (estado 8).

Para su programación en el microprocesador, a la hora de elegir el estado en la máquina de estados se guarda el valor del estado en el acumulador y se usa el comando *RL A* para multiplicar por dos el valor, ya que el comando *JMP DPTR* hace saltos de dos en dos.

Estado	Número
Reposo	00
Despegue	01
Vuelo estable	02
Manual	03
Aterrizaje	04
Sube	05
Baja	06
Giro_dcha	07
Giro_izq	08

Cuadro 1: Tabla de estados

2.1.1. Reposo

Este estado es el inicial y el que se escoge en las inicializaciones.

Cada 10s se inicia la conversión del ADC en el canal de la batería y se muestra el valor correspondiente en el **display**.

Si hay un cambio ascendente en el botón de Despegue/Aterrizaje se genera el evento de *encendido* y se cambia a estado **Despegue**

2.1.2. Despegue

En este estado los **PWM** están haciendo que el drone suba.

En caso de que haya un cambio desdendente en el boton de Despegue/Aterrizaje, pasara el evento de **apagado** y cambiara a estado **Aterrizaje**.

Cada 10s se inicia la conversión del ADC en el canal de la batería y se muestra el valor correspondiente en el **display**. Si la medición resulta en que la batería es menor que 5 %, cambiará al estado de **Aterrizaje**

Cada 10ms se inicia la conversión del ADC en el canal de altura y si la altura es igual o mayor a 1m activa el *flag* de **tick_automan**.

Si **tick_automan** esta activo significa que ya ha llegado a un metro, por lo que mira el selector **"Auto/Manual"(P0.5)** y cambia a estado.

2.1.3. Vuelo estable

En este estado los **PWM** están manteniendo la altura del drone.

En caso de que haya un cambio desdendente en el boton de Despegue/Aterrizaje, pasara el evento de **apagado** y cambiara a estado **Aterrizaje**.

Cada 10s se inicia la conversión del ADC en el canal de la batería y se muestra el valor correspondiente en el **display**. Si la medición resulta en que la batería es menor que 5 %, cambiará al estado de **Aterrizaje**

Cada 10ms se inicia la conversión del ADC en el canal de altura y si la altura es menor de 80cm cambia al estado **Subir**, en cambio si la altura es mayor a 120cm cambia al estado **Bajar**.

A su vez, también cada 10ms, se inicial la conversión del ADC en el canal de distancia frontal, si la distancia frontal es menor a 40cm cambia al estado **Giro_Dcha**. Por último si pasan 25 segundos en este estado sin ningún obstáculo cambiara al estado **Giro_Izq**.

2.1.4. Manual

En este estado las acciones del drone estaran relacionadas con los inputs del usuario en los botones de **Paro/Avanza (P0.6)**, **g_izq (P0.3)**, **g_dcha (P0.2)**, **sube P0.1** y **baja (P0.0)**.

En caso de que haya un cambio desdendente en el boton de Despegue/Ate-

rrizaje, pasara el evento de **apagado** y cambiara a estado **Aterrizaje**.

Cada 10s se inicia la conversión del ADC en el canal de la batería y se muestra el valor correspondiente en el **display**. Si la medición resulta en que la batería es menor que 5 %, cambiará al estado de **Aterrizaje**

Las acciones asociadas a cada una de las opciones de botones están descritas en la siguiente tabla:

P0.6	P0.3	P0.2	P0.1	P0.0	
Paro/Avanza	g_izq	g_dcha	sube	baja	**Acción**
0	0	0	0	0	Parado
0	0	0	0	1	Baja
0	0	0	1	0	Sube
0	0	0	1	1	-
0	0	1	0	0	Gira dcha
0	0	1	0	1	Gira dcha + baja
0	0	1	1	0	Gira dcha + sube
0	0	1	1	1	-
0	1	0	0	0	Gira izda
0	1	0	0	1	Gira izda + baja
0	1	0	1	0	Gira izda + sube
0	1	0	1	1	-
0	1	1	0	0	-
0	1	1	0	1	-
0	1	1	1	0	-
0	1	1	1	1	-
1	0	0	0	0	Avanza
1	0	0	0	1	Baja + avanza
1	0	0	1	0	Sube + avanza
1	0	0	1	1	-
1	0	1	0	0	Gira dcha
1	0	1	0	1	Gira dcha + baja
1	0	1	1	0	Gira dcha + sube
1	0	1	1	1	-
1	1	0	0	0	Gira izda
1	1	0	0	1	Gira izda + baja
1	1	0	1	0	Gira izda + sube
1	1	0	1	1	-
1	1	1	0	0	-
1	1	1	0	1	-
1	1	1	1	0	-
1	1	1	1	1	-

Cuadro 2: Comportamiento del drone en modo manual

Cada una de las acciones posibles descritas en la tabla anterior tendrán un resultado en los **PWM0** , **PWM1** y los pines **1.6, 1.7**. Sus valores en cada accion estan descritas en la siguiente tabla:

Acción	PWM0	PWM1	P1.7	P1.6
Parado	80	FF	0	0
Baja	B3	FF	0	0
Sube	4D	FF	0	0
Gira dcha	80	80	1	0
Gira dcha + baja	B3	80	1	0
Gira dcha + sube	4D	80	1	0
Gira izda	80	80	0	1
Gira izda + baja	B3	80	0	1
Gira izda + sube	4D	80	0	1
Avanza	80	80	1	1
Baja + avanza	B3	80	1	1
Sube + avanza	4D	80	1	1
Gira dcha	80	80	1	0
Gira dcha + baja	B3	80	1	0
Gira dcha + sube	4D	80	1	0
Gira izda	80	80	0	1
Gira izda + baja	B3	80	0	1
Gira izda + sube	4D	80	0	1

Cuadro 3: Valores PWM0,PWM1 P1.7 y P1.6 en modo manual

2.1.5. Aterrizaje

En este estado los **PWM** estarán bajando.

Cada 10s se inicia la conversión del ADC en el canal de la batería y se muestra el valor correspondiente en el **display**.

Cada 10ms se inicia la conversión del ADC en el canal de altura y si la altura es menor de 1cm cambia al estado **Reposo**.

2.1.6. Sube

En este estado los **PWM** estarán haciendo que el drone suba.

Cada 10s se inicia la conversión del ADC en el canal de la batería y se muestra el valor correspondiente en el **display**. Si la medición resulta en que la batería es menor que 5 %, cambiará al estado de **Aterrizaje**

Cada 10ms se inicia la conversión del ADC en el canal de altura y si la altura es igual a 1m volverá al estado de **Vuelo Estable**.

2.1.7. Baja

En este estado los **PWM** estarán haciendo que el drone baje.

Cada 10s se inicia la conversión del ADC en el canal de la batería y se muestra el valor correspondiente en el **display**. Si la medición resulta en que la batería es menor que 5 %, cambiará al estado de **Aterrizaje**

Cada 10ms se inicia la conversión del ADC en el canal de altura y si la altura es igual a 1m volverá al estado de **Vuelo Estable**.

2.1.8. Giro_dcha (por obstaculo)

En este estado los **PWM** estarán haciendo que el drone gire a la derecha mientras mantiene la altura.

Cada 10s se inicia la conversión del ADC en el canal de la batería y se

muestra el valor correspondiente en el **display**. Si la medición resulta en que la batería es menor que 5 %, cambiará al estado de **Aterrizaje**

Cada 10ms se inicia la conversión del ADC en el canal de distancia frontal y si la distancia frontal es mayor a 80cm se estima que el obstáculo de ha esquivado y volverá el estado de **Vuelo estable**, sin embargo si pasan 8s (el tiempo que se estima que el drone tarda en dar dos vueltas) y no se ha cumplido este evento, resulta que el drone no es capaz de esquivar el obstaculo por lo que saltará a estado **Aterrizaje**.

2.1.9. Giro_izq (por timeout)

En este estado los **PWM** estarán haciendo que el drone gire a la izquierda mientras mantiene la altura.

Cada 10s se inicia la conversión del ADC en el canal de la batería y se muestra el valor correspondiente en el **display**. Si la medición resulta en que la batería es menor que 5 %, cambiará al estado de **Aterrizaje**

Cuando pasen 0,5 segundos (500ms) se estima que supondría un giro de 45º, por lo que el drone volverá al estado **Vuelo Estable**.

2.2. Eventos y Acciones

A continuación se muestran las tablas de eventos y acciones:

Evento	Flag	Descripción
0	-	-
1	encendido	cambio ascendente del boton despegue/aterrizaje
2	Apagado	cambio ascendente del boton despegue/aterrizaje
3	tick_10ms	10ms
4	tick_500ms	500ms
5	tick_8s	8s
6	tick_10s	10s
7	tick_25s	25s
8	tick_ADC_bat	fin de conversión batería
9	tick_ADC_alt	fin de conversión altura
10	tick_ADC_frnt	fin de conversión frente
11	tick_automan	en estado despegue ha llegado a 1m

Cuadro 4: Tabla de eventos

Acción	Descripción
1	encender
2	apagar
3	iniciar conversión altura
4	iniciar conversión batería
5	iniciar conversión frente
6	reiniciar timer
7	elegir automático/manual
8	cambiar a despegue
9	cambiar a gir_oizq
10	cambiar a giro_dcha
11	cambiar a baja
12	cambiar a sube
13	cambiar a vuelo estable
14	cambiar manual
15	cambiar a aterrizaje
16	comprobar botones

Cuadro 5: Tabla de acciones

3. Diagramas de Flujo

En este apartado e exponen los diagramas de flujo solicitados en el enunciado de la actividad.

3.1. Programa principal

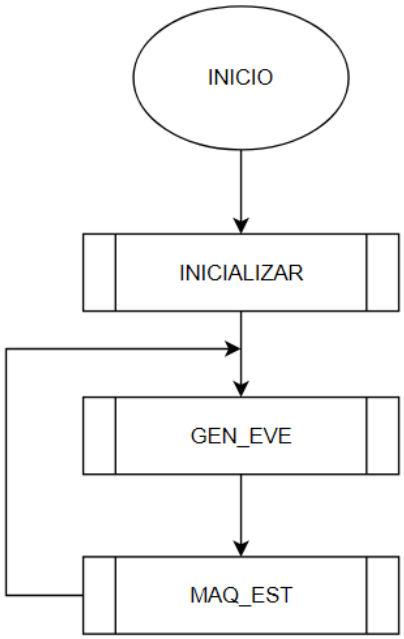

Figura 3: Diagrama de flujo programa principal

3.2. Inicializaciones

Figura 4: Diagrama de flujo inicializaciones

3.3. Generador de eventos

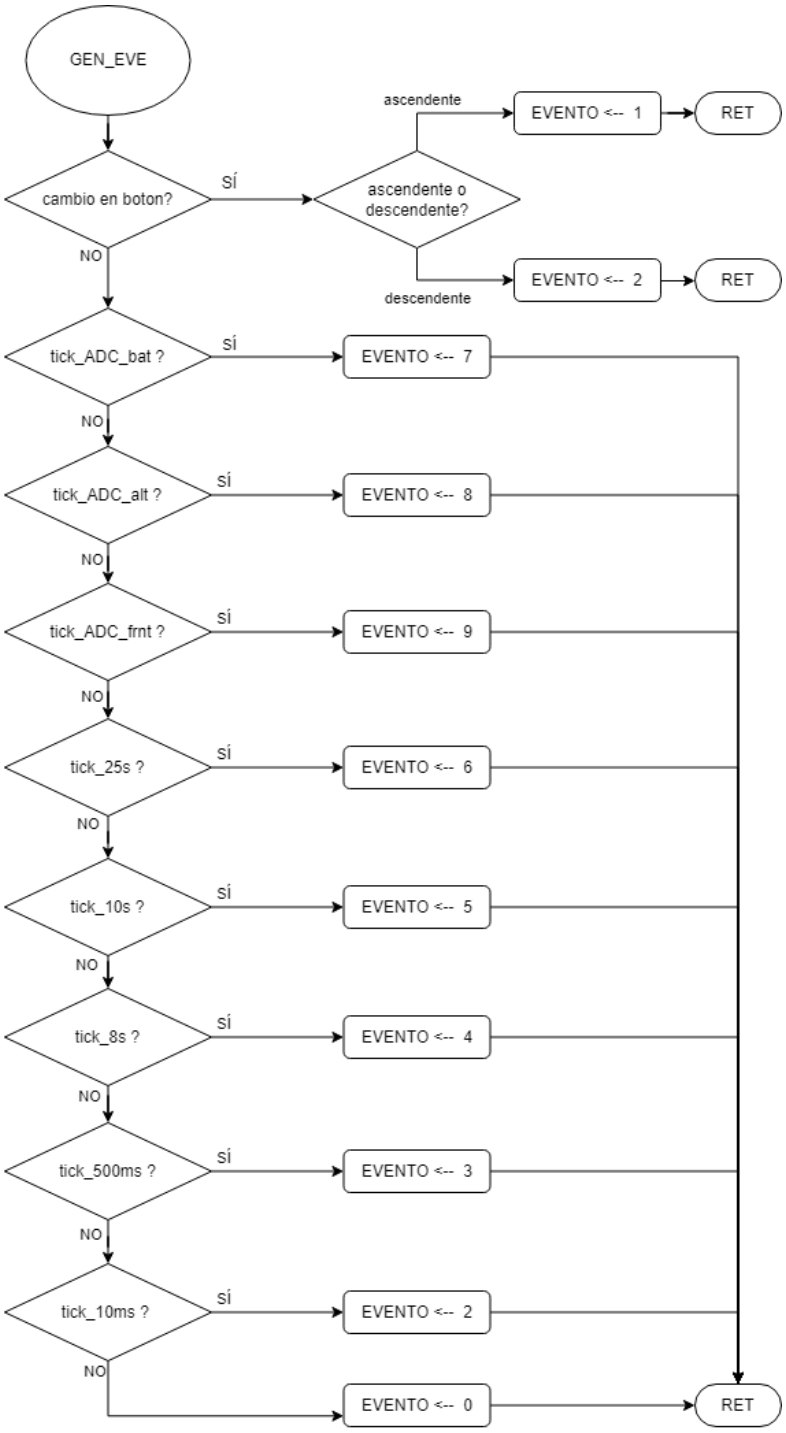

Figura 5: Diagrama de flujo generador de eventos

3.4. Subrutinas de atención a la interrupción

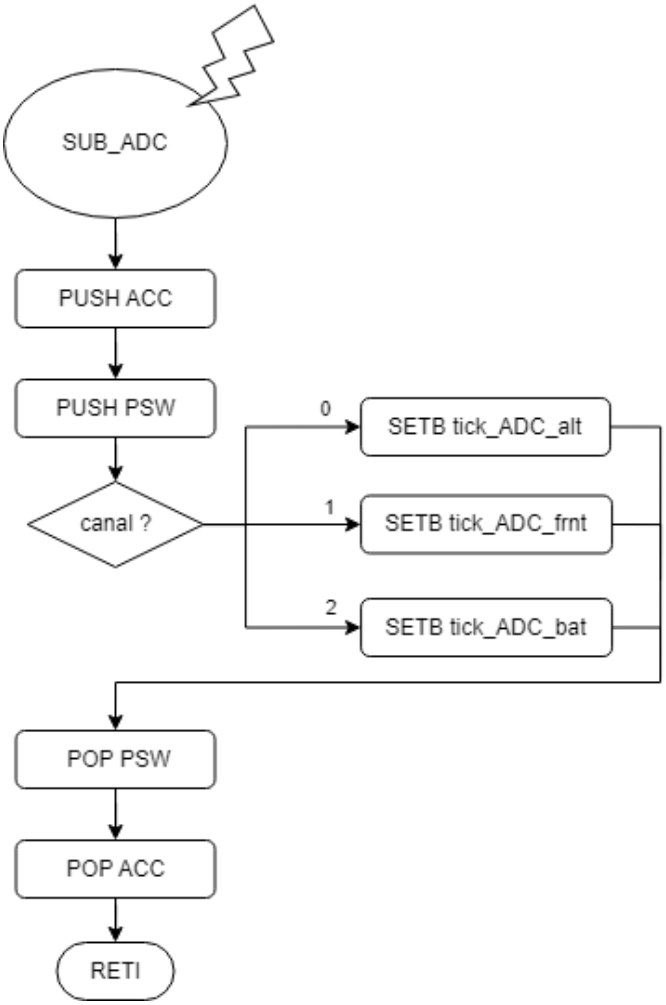

Figura 6: Diagrama de flujo subrutina ADC

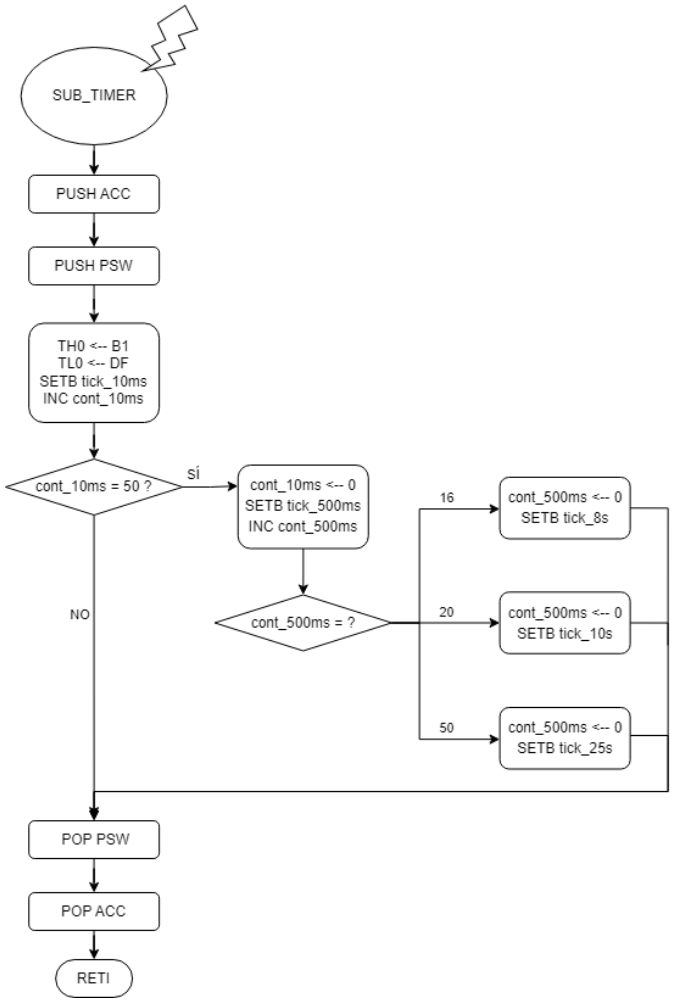

Figura 7: Diagrama de flujo subrutina del Timer

3.5. Maquina de estados

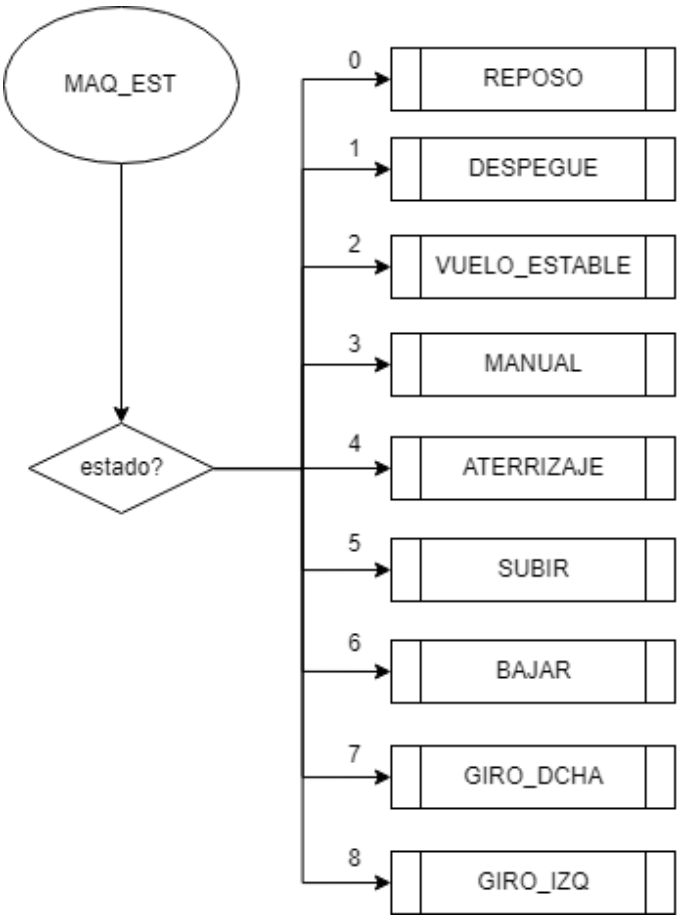

Figura 8: Diagrama de flujo maquina de estados

3.6. Maquinas de eventos

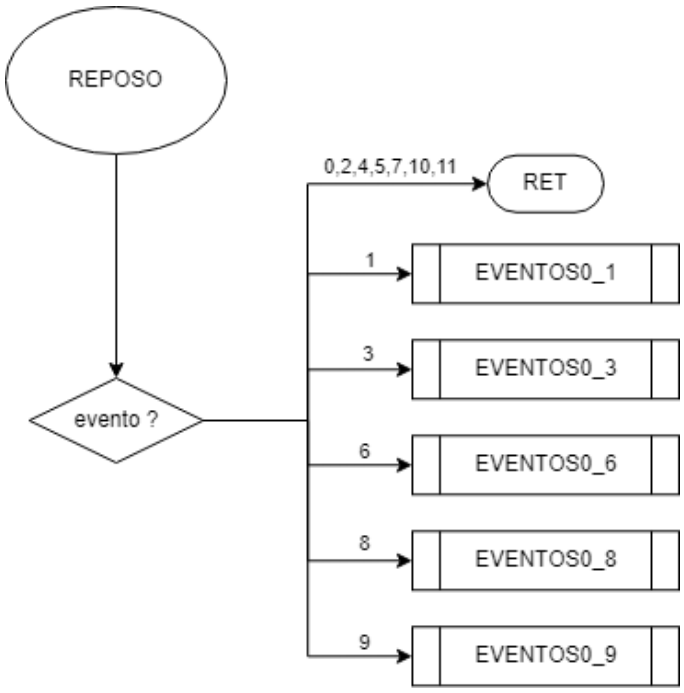

Figura 9: Diagrama de flujo estado reposo

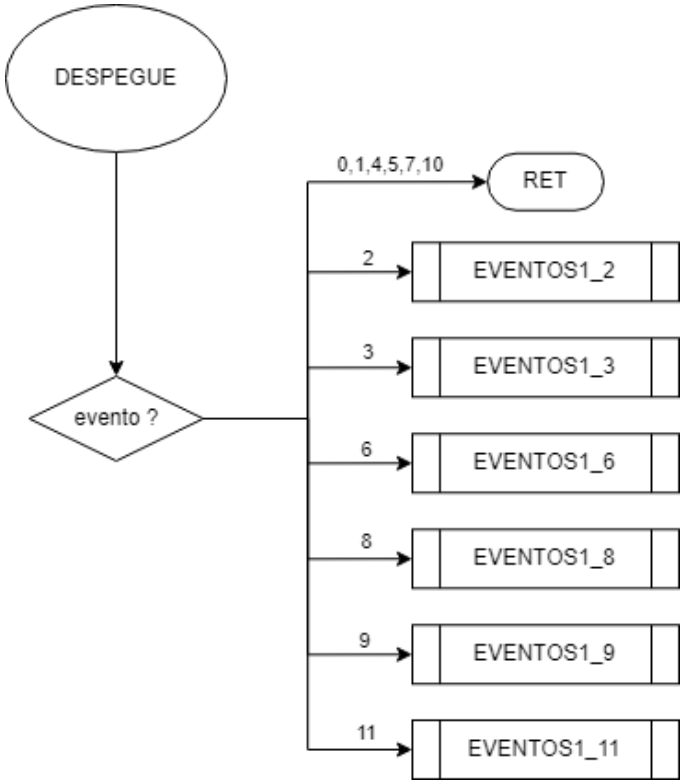

Figura 10: Diagrama de flujo estado despegue

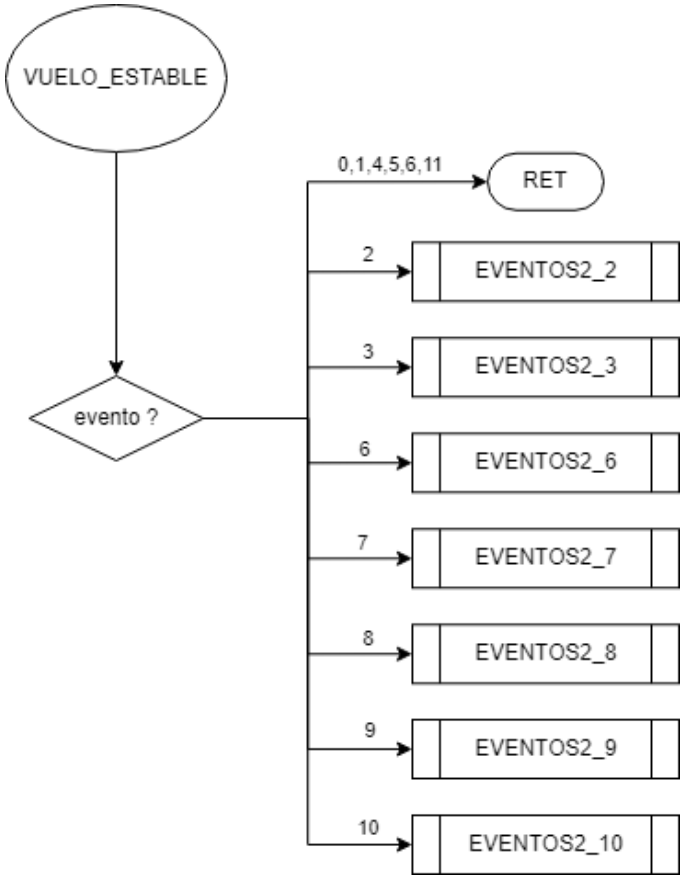

Figura 11: Diagrama de flujo estado vuelo estable

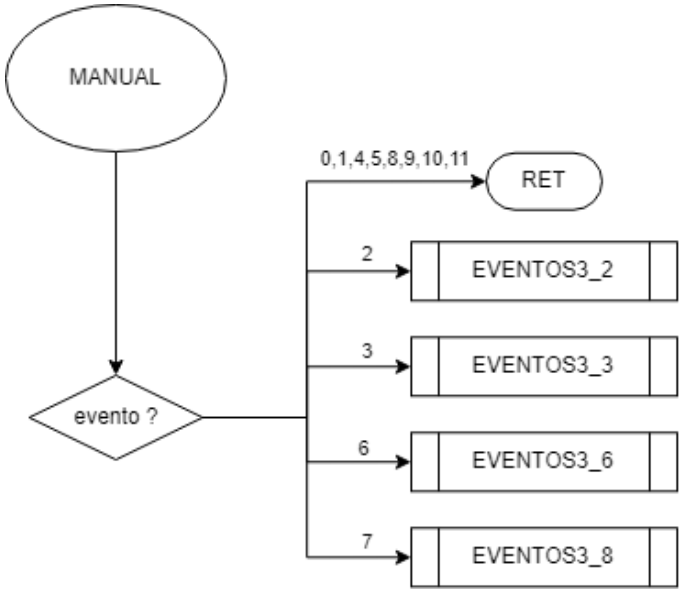

Figura 12: Diagrama de flujo estado manual

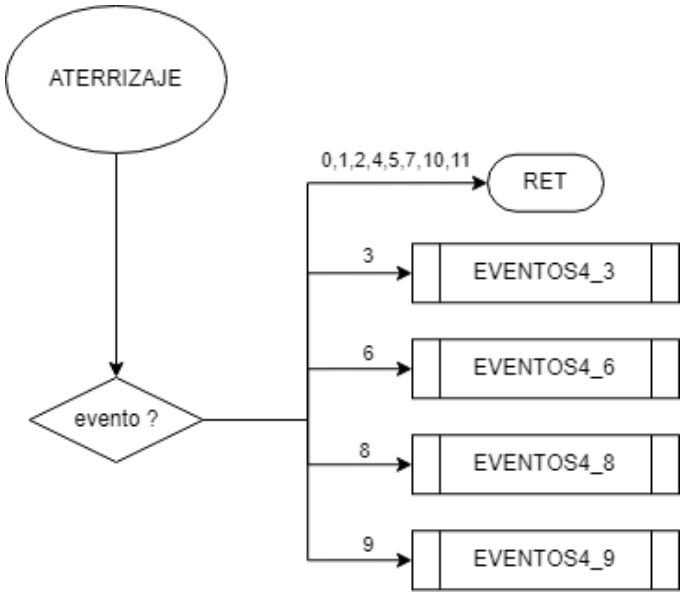

Figura 13: Diagrama de flujo estado aterrizaje

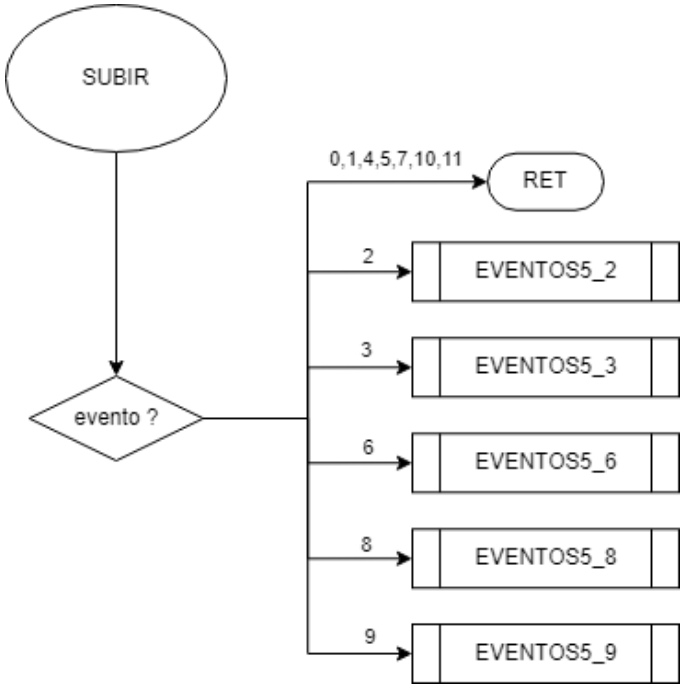

Figura 14: Diagrama de flujo estado subir

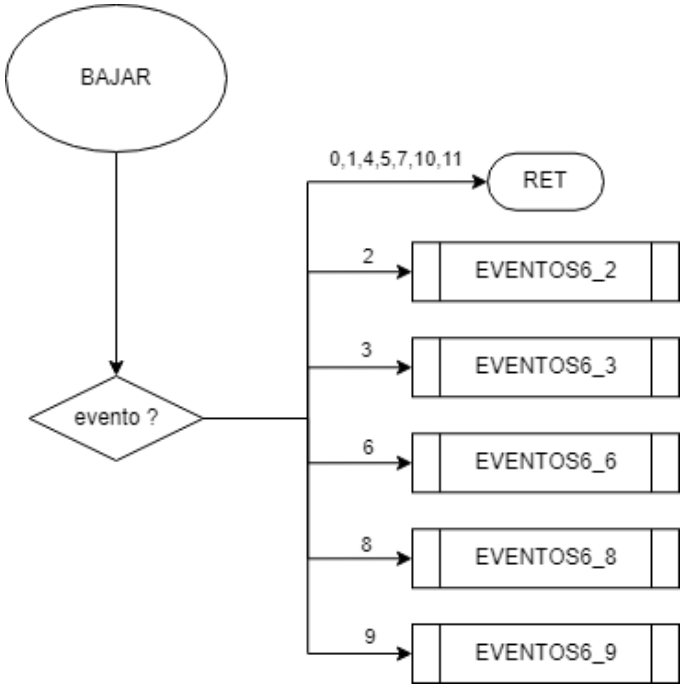

Figura 15: Diagrama de flujo estado bajar

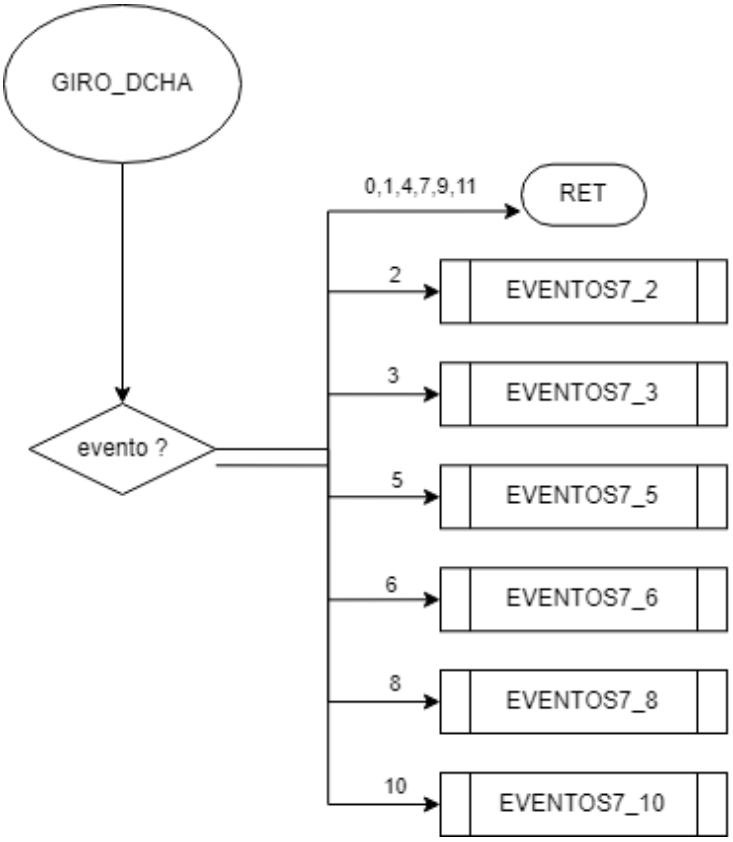

Figura 16: Diagrama de flujo estado giro derecha

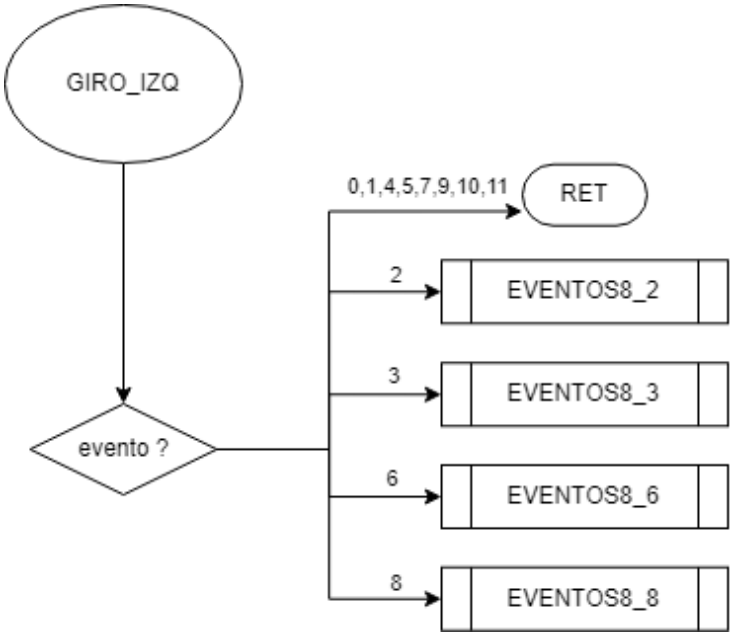

Figura 17: Diagrama de flujo estado giro izquierda

3.7. Subrutinas

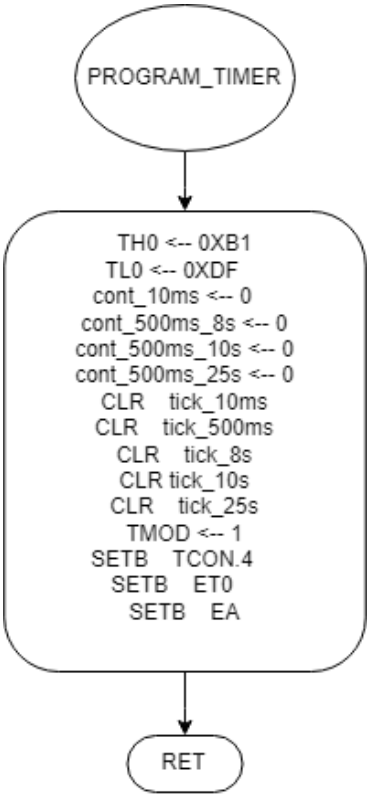

Figura 18: Diagrama de flujo subrutina program_timer

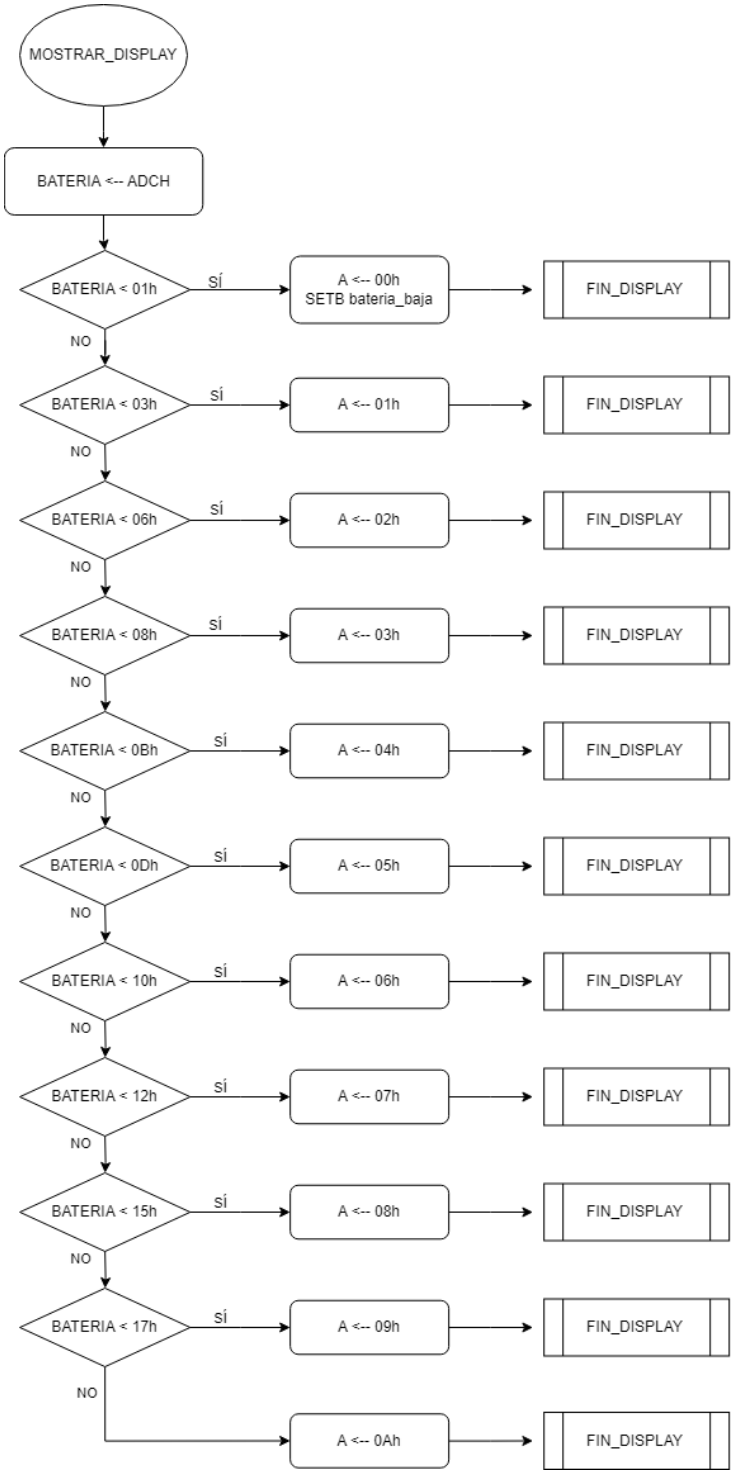

Figura 19: Diagrama de flujo subrutina mostrar_display

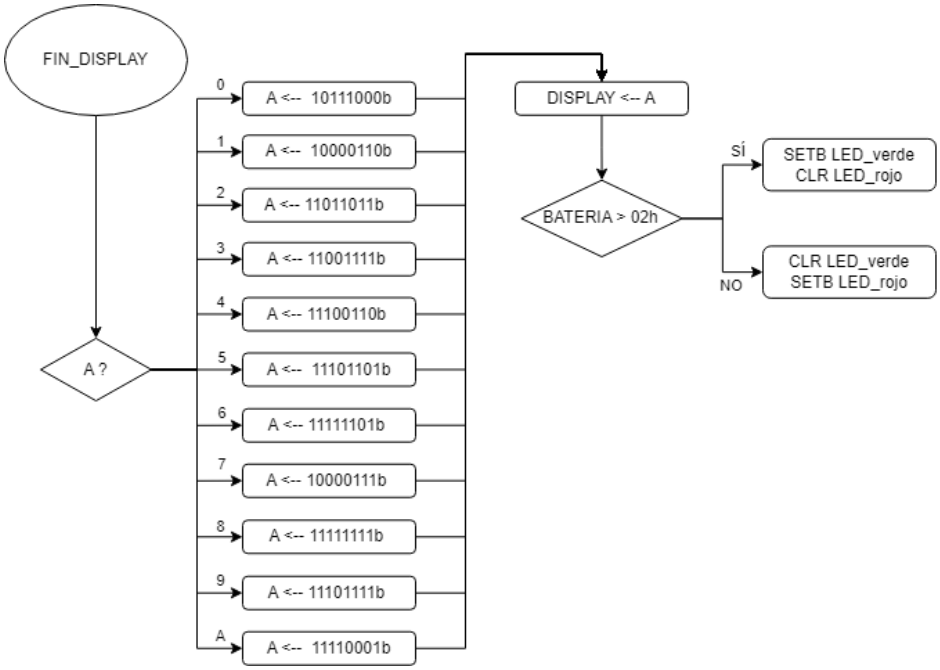

Figura 20: Diagrama de flujo subrutina fin_display

4. Explicación y cálculos

4.1. EQUs

Para facilitar la programación de la solución, se utiliza la directiva EQU, que asigna un nombre a una dirección de memoria. Este nombre sustituirá a dicha dirección, de forma que la asignación de un nombre identificativo nos dará una mayor idea sobre qué se está realizando.

Por ejemplo, se ha utilizado la dirección 0x30 como contador de 10ms de la siguiente manera:

```
1 cont_10ms      EQU     0x30
```

4.2. Directivas ORG

Esta otra directiva, por su parte, nos permite escribir de manera ordenada las instrucciones en la memoria de programa.

```
1 ORG      0x80
```

El código que se escriba por debajo de esta línea, empezará en la dirección de memoria 0x80.

El uso de esta directiva es imprescindible para gestionar las interrupciones. Cuando estas suceden, el microprocesador salta al vector de interrupción correspondiente, y si se quiere dar respuesta a dicha interrupción, se deberá escribir la ISR (*Interrupt Service Request*, por sus siglas en inglés -Rutina de Atenci´on a la Interrupci´on-) en ese vector. Para ello se utilizará la directiva ORG.

En este caso se han utilizado las siguientes directivas ORG para las ISR:

```
1 ORG     0x0B          ;INT. TIMER0
2
3 ORG     0x53          ;INT. POR FIN DE CONVERSION ADC
```

4.3. Inicializaciones

Si bien el microprocesador nos garantiza unos valores de reset para los registros, inicializarlos a los valores deseados (aunque estos coincidan con los valores garantizados) constituye una buena práctica de desarrollo de software. De esta manera, evitamos cualquier fallo potencial o desestabilización de software debido a la lectura o al uso de un valor inapropiado. Por este motivo, se dedica una primera parte del programa a la inicializaciones.

Asimismo, en esta subrutina se configuran todos aquellos registros del microprocesador (y en caso de ser necesario, también del microcontrolador) que nos permitirán ejecutar el código programado de forma adecuada. Se inicializan los registros, los puertos de salida y las flags, se configuran los periféricos (Timer 0, ADC, señal PWM...) y se habilitan las interrupciones.

Destacar que para que la potencia entregada por la **señales PWM**, su duty-cycle deberán ser inicializados al máximo (FFh).

4.4. Generados de Eventos

La solución diseñada consta de un **único** generador de eventos, que independientemente del estado en el que se encuentre la maquina mirará el estado de los flags y/o periféricos generar unos eventos u otros. Esta decisión de evento se almacenará en la variable **EVENTO** ubicada en el registro 1 (R1)

4.5. Máquina de Estados

La máquina de estados redirigir´a la ejecución del programa al código correspondiente al estado actual de la máquina. Lee el valor del registro **ESTADO** ubicada en el registro 0 (R0) y saltar´a a la subrutina del código que gestiona el estado actual.

4.6. Interrupciones

Una interrupción es un evento impredecible que detiene (o interrumpe) el flujo normal del programa. Es por esto por lo que ante una situación de

interrupción, esta deber´a ser atendida mediante una **rutina de atención a la interrupción (ISR).**

Cuando se produzca una interrupción, el microprocesador detendrá la ejecución del programa y saltar´a al denominado ***vector de interrupción***. Este es una posición de memoria a la que se salta cuando se produce una interrupción. Por consiguiente, la ISR de una interrupción tendrá que ser escrita en su vector de interrupción.

Sin embargo, entre cada vector de interrupción solamente hay ocho posiciones de memoria, una capacidad reducida para cualquier ISR sofisticada. Por ello, se ha optado por escribir en el vector de interrupción un salto al inicio de la ISR.

El microcontrolador 80C552 tiene un total de quince fuentes de interrupción, por lo que tendrá quince vectores de interrupción. En la siguiente tabla se recogen aquellos que se utilizan en la solución de esta práctica. El resto pueden visualizarse en la *Tabla 9 de la página 45 del documento 80C51 Family Derivatives.*

Fuente de Interrupción	Vector
TF0 or T0 (Overflow Timer 0)	0x0B
ADC (Conversión finalizada)	0x53

Cuadro 6: Caption

Recordemos que para que estas situaciones interrumpan al microprocesador, tienen que estar habilitadas, al igual que el habilitador global de interrupciones.

4.7. Botón despegue/aterrizaje

Conectado al pin 0.7 (Puerto 0) se encuentra el botón de Despegue/Aterrizaje, para controlar los inputs en este botón se ha usado un enfoque mediante **flanco**.

Cada vez que se ejecuta el generador de eventos se compara el estado actual de puerto con el estado que tenemos guardado de la anterior medición, mediante un a operación lógico *XOR* se mira si ha habido un cambio en ese pin, en el

caso de que lo haya se mira si es un cambio ascendente o descendente para asignar el evento 1 (*encendido*) o evento 2 (*apagado*) respectivamente.

4.8. Botones modo manual

Conectados a los pines 0.6, 0.3, 0.2, 0.1 y 0.0 (Puerto 0) se encuentras los botones utilizados para el modo manual. Estos botones también se controlan por **flanco**, de forma que se guarda el estado anterior del puerto y cada 10ms se compara con el nuevo estado y se ejecutan las acciones pertinentes según que bits son ascendentes y cuales son descendentes. Todas las acciones de estos botones están descritas en la tabla *2 de la página 12*

4.9. ADC

En los pines 5.0, 5.1 y 5.2 (Puerto 5) se encuentran los convertidores Analógico-Digital (ADC), dejando los canales 0, 1 y 2 para el sensor de altura, el sensor de distancia frontal y sensor de batería respectivamente.

Para poder controlar estos 3 sensores al mismo tiempo, se selecciona el canal que nos interesa en cada momento manipulando los 3 bits menos significativos de registro ADCON ubicado en la dirección 0xC5. Una vez se termina de convertir el dato analógico y salta la interrupción de fin de conversión, no sabemos en cual de los 3 canales nos encontramos por lo que en la rutina de atención a la interrupción se hace una selección los bits anteriormente mencionados de ADCON, una vez sabemos en que canal nos encontramos se activara el flag correspondiente a ese canal (tick_ADC_alt, tick_ADC_frnt o tick_ADC_bat).

Por ultimo cuando se atiende el evento de fin de conversión con los flags anteriormente mencionados, el valor digital de la conversión se encuentra en el registro ADCH ubicado en la dirección 0xC6.

Para este proyecto se han utilizado mediciones en cm para los canales 0 y 1 (altura y distancia frontal), según el enunciado : *"Los sensores de distancia del sistema proporcionan una salida analógica de* ***16,667 mV/cm****, que nunca superará los 5 V. Cuando la medida es nula, la tensión de salida es 0V."*

Para poder calcular los voltajes correspondientes a cada medida se ha empleado la fórmula proporcionada por el ADC *(página 2, 80C51 Family Derivatives)*

$$ADCH = 256 \times \frac{V_{\mathbf{IN}} - AV_{\mathbf{ref-}}}{AV_{\mathbf{ref+}} - AV_{\mathbf{ref-}}}$$

Sabiendo que las tensiones de referencia son 0V y 5V, si despejamos esa formula sacamos la tensión en voltios que generarían cada medida en centímetros y que resultado hexadecimal tendrían en el ADCH, esos resultados están descritos en la siguiente tabla:

Altura (cm)	**Tensión de Entrada (V)**	**ADCH**
0	0.0000	0
20	0.3333	11
40	0.6667	22
60	1.0000	33
80	1.3334	44
90	1.5000	4C
100	1.6667	55
110	1.8334	5D
120	2.0000	66
130	2.1667	6E

Cuadro 7: Valores ADC Distancia

Por otra parte para los valores de tensión de la batería el enunciado no dice lo siguiente : "*El sensor de tensión de la batería proporciona 4,883 mV/V (% de carga, desde 0 % al 100 %) y también tiene una referencia de 0 V cuando la batería está descargada.*"

Usando la misma formula que antes podemos sacar estos valores de tensión para cada tramo de porcentaje de batería y que mostraran en el desplaya

Batería (%)	Tensión de Entrada (V)	ADCH	Display
0	0.0000	0.000	L
5	0.0244	1.250	1
10	0.0488	2.500	1
15	0.0732	3.750	2
20	0.0977	5.000	2
25	0.1221	6.250	3
30	0.1465	7.500	3
35	0.1709	8.750	4
40	0.1953	10.000	4
45	0.2197	11.250	5
50	0.2442	12.500	5
55	0.2686	13.751	6
60	0.2930	15.001	6
65	0.3174	16.251	7
70	0.3418	17.501	7
75	0.3662	18.751	8
80	0.3906	20.001	8
85	0.4151	21.251	9
90	0.4395	22.501	9
95	0.4639	23.751	F
100	0.4883	25.001	F

Cuadro 8: Valores ADC Batería

4.10. PWM

Una señal PWM (Pulse Width Modulation -Modulación por anchura de pulsos-) se utiliza como un conversor Digital-Analógico (DAC: Digital Analogic Converter).

En esta práctica esta señal es la encargada de controlar la potencia que se entrega a los motores del drone, para ello contamos con dos módulos: PWM0 y PWM1. En primer lugar el módulo PWM0 controla la altura con el *duty cycle*: 0 % parado, 50 % mantiene altura,70 % sube y 30 % baja.

Por otra parte el módulo PWM1 controla el "Giro/Avance" con el *duty cycle* asociado: 0 % parado y 50 %gira/avanza. En adición a estos *duty cycle*, los pines 1.7 y 1.6 controlan cómo se reparte el PWM1 a los rotores del drone

para conseguir el giro o el avance de la siguiente manera: Giro_Avance: “00” ‘Parado’, “11” ‘Avanza’, “10” ‘Giro_dcha.’, “01” ‘Giro_izda.’

Para configurar la señales PWM en el microcontrolador, el primer paso a realizar es fijar la frecuencia de conmutación.

El enunciado nos indica por una parte que $f_{PWM} = 500Hz$ y por otra parte de la frecuencia del oscilador del microcontrolador de de 24MHz, por lo que tendremos que calcular el valor del registro PWMP que fija la frecuencia de conmutación a 500Hz. Seguiremos los pasos indicados en la página 46 del documento *80C51 Family Derivatives:*

$$f_{\text{PWM}} = \frac{f_{\text{osc}}}{2 \times (1 + PWMP) \times 255}$$

Sustituyendo los valores:

$$500Hz = \frac{24Mhz}{2 \times (1 + PWMP) \times 255}$$

Despejamos PWMP y obtenemos:

$$PWMP = 93,11 \approx 93d = 0x5Dh$$

En segundo lugar, se procede a calcular el valor del duty-cycle para conseguir cada porcentaje. En total serán cuatro niveles: 0 %, 30 %, 50 % y 70 % Se ha utilizado la señal PWM0 y se hará de la siguiente manera:

$$PWM = 255 \times (1 - Duty)$$

Si calculamos los valores obtenemos los siguiente:

%	**PWM0 / PWM1**
0 %	FFh
30 %	B3h
50 %	80h
70 %	4Dh

Cuadro 9: Valores PWM

4.11. Timer

El *Timer 0* de ha empleado para controlar la cadencia de las mediciones de altura, frontal y la batería, además de contar los 25s y los 8s. Basandonos en nuestro enunciado, necesitaremos flags de tiempo de 10ms, 500ms, 8s, 10s y 25s; teniendo esto en cuenta hay que generar una interrupción del timer cada 10ms.

Para ello, se ha configurado en modo 1 (contador de 16 bits) y se ha programado una interrupción cada 10ms. Es necesario precargar el timer, y debido al modo seleccionado, se realiza de forma manual.

La precarga se ha calculado de la siguiente manera, siendo 'm' el módulo del Timer y 'n' el n´umero de ciclos:

$$m - n$$

$$2^{16} - ciclos$$

Y el número de ciclos es la relación entre el tiempo de salida (T_e) y el tiempo de entrada (T_e):

$$\frac{T_e}{T_o} = ciclos$$

Como sabemos que el tiempo de entrada es 0,5µ (24 MHz de reloj, 12 ciclos), y el tiempo de salida es 10ms:

$$\frac{10 \cdot 10^{-3}}{0,5 \cdot 10^{-6}} = 20000 ciclos$$

El valor hexadecimal calculado irá al registro *TH0* y *TL0* :

Precarga (hex)	Resgistro
B1	TH0
DF	TL0

Cuadro 10: Valores precarga del Timer

Al solo utilizar un Timer, necesitamos múltiples contadores. En la subrutina de atención a la interrupción del Timer, cada vez que se da se incremente en uno el contador cont_10ms y cuando este llega a 50 se incrementa en uno el contador cont_500ms y cont_10ms vuelve a 0. A su vez, cuando cont_500ms llega a 16, 20 o 50 se activan los flags de tiempo tick_8s, tick_10s y tick_25s respectivamente.

4.12. Display

Conectado al puerto 2 se encuentra un display siete segmentos en el que se va a mostrar el porcentaje de batería, además de los esos siete segmentos también tenemos un pin adicional para un segmento denominado ́dpque indica si el drone se encuentra en modo manual o automático, los segmentos están organizados de la siguiente manera:

Pin	**Segmento**
2.0	a
2.1	b
2.2	c
2.3	d
2.4	e
2.5	f
2.6	g
2.7	dp

Cuadro 11: Segmentos del display

Estos segmentos están dispuestos de la siguiente manera:

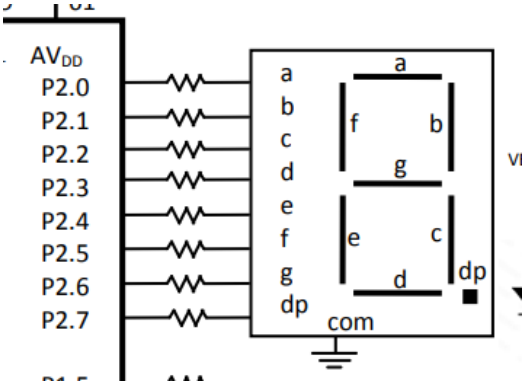

Figura 21: Disposición display

Para mostrar los valores se utiliza la instrucción MOVC, es decir, se codifican los valores. Posteriormente, es suficiente con mover este valor codificado al Puerto 2 para mostrar por el display la cifra o letra deseada.

En enunciado nos dice los siguiente : *Entre el 0 % y el 5 %, deberá marcar 'L' (low). Del 5 % al 15 %, '1'. Del 15 % al 25 %, '2', y así sucesivamente hasta '9'. Del 95 % al 100 %, debería indicar 'F' (full). El indicador 'dp' del display indicará el modo de funcionamiento. Si se encuentra encendido, el modo actual será Automático. Si se encuentra apagado, el modo será Manual.*

De este modo estos serán los valores del puerto dependiendo de el numero que queramos representar¿

Valor	P2.0	P2.1	P2.2	P2.3	P2.4	P2.5	P2.6
L	1	0	1	1	1	0	0
1	1	0	0	0	0	1	1
2	1	1	0	1	1	0	1
3	1	1	0	0	1	1	1
4	1	1	1	0	0	1	1
5	1	1	1	0	1	1	0
6	1	1	1	1	1	1	0
7	1	0	0	0	0	1	1
8	1	1	1	1	1	1	1
9	1	1	1	0	1	1	1
F	1	1	1	1	0	0	0

Cuadro 12: Tabla valores display

4.13. LEDs

Además del display también hay un LED rojo (P1.4) y un LED verde (P1.5) para indicar el estado de la bateria, si el valor de la misma es inferior al 10 % se encenderá el LED rojo y de lo contrario se encenderá el LED verde.

La actualización de estos LEDs se realiza justo después del display, al terminar la subrutina del display se mira el porcentaje de batería para cambiar los LEDs si es necesario.

4.14. ISR

Una *ISR (Interrupt Service Request* -rutina de atención a la interrupción-) es el proceso mediante el que se trata una interrupción.

En consecuencia, deberán ser escritas en el vector de interrupción, o ser llamadas desde el mismo.

Todas estas rutinas tendrán una parte común:

```
1 PUSH PSW
2 PUSH ACC
3 ;
4 POP ACC
5 POP PSW
6 RETI
```

Listing 1: Subrutinas

Se deberá guardar en la pila el valor de los registros, acumulador y estado de la palabra de control, así como SFRs.

Durante la ejecución de estas rutinas, los valores de dichos registros pueden verse alterados, y tras dar respuesta a la interrupción, se volver´a al mismo estado en que se encontraba el microprocesador cuando sucedi´o la interrupci´on.

La instrucción **RETI**, además de cumplir con las mismas funciones de una **RET**, desactiva el indicador de que una interrupción está siendo atendida para que se puedan atender nuevas interrupciones.

5. Código

5.1. EQUs

```
;#########################################################
;                      ETIQUETAS
;#########################################################
;----variables----
ESTADO                  EQU     R0
EVENTO                  EQU     R1
BATERIA                 EQU     R2

;----puertos----
desp_ater               EQU     P0.7
par_avan                EQU     P0.6
auto_man                EQU     P0.5
g_izda                  EQU     P0.3
g_dcha                  EQU     P0.2
sube                    EQU     P0.1
baja                    EQU     P0.0
LED_rojo                EQU     P1.4
LED_verde               EQU     P1.5
DISPLAY                 EQU     P2

;----variables----
tick_10ms               EQU     0X20.0
tick_500ms              EQU     0X20.1
tick_8s                 EQU     0X20.2
tick_10s                EQU     0X20.3
tick_25s                EQU     0X20.4
tick_ADC_alt            EQU     0x20.5
tick_ADC_frnt           EQU     0x20.6
tick_ADC_bat            EQU     0x20.7
tick_automan            EQU     0x21.1
bateria_baja            EQU     0x21.2
bit_aux_avanza          EQU    0x21.3

;---- manual ----
est_ahora               EQU     0x22
est_antes               EQU     0x23
est_cambio              EQU     0x24
est_cambio_sube         EQU     0x25
```

```
39 est_cambio_baja       EQU     0x26
40 est_aux               EQU     0x27
41
42 :--- timers----
43 cont_10ms             EQU     0x30
44 cont_500ms_8s         EQU     0x31
45 cont_500ms_10s        EQU    0x32
46 cont_500ms_25s        EQU    0x33
47
48 ;---- boton desp/ater ----
49 boton_antes           EQU     0x34
50 boton_ahora           EQU     0x35
51 boton_sube            EQU     0x28.0
52 boton_baja            EQU     0x28.1
53
54 ;----PWM----
55 PWMP                  EQU     0xFE
56 PWM0                  EQU     0xFC
57 PWM1                  EQU     0xFD
58
59 ;----ADC----
60 ADCON                 EQU     0xC5
61 ADCH                  EQU     0xC6
```

Listing 2: EQUs

5.2. Interrupciones

```
;#########################################################
;                     INTERRUPCIONES
;#########################################################


ORG      0x0B                          ;TIMER0
         ACALL        SUB_TIMER        ; 10ms
         RETI

ORG      0x53                          ;FIN DE CONVERSION ADC
         PUSH         PSW
         PUSH         ACC
         ACALL        SUB_ADC
         POP          ACC
         POP          PSW
         RETI

;****************    ADC   *****************

SUB_ADC:
         ANL          ADCON, #11101111b
         MOV          A,ADCON
         ANL          A,#0x03
         RL           A
         RL           A
         MOV          DPTR,#ADC_TIPO
         JMP          @A+DPTR
ADC_TIPO:
         SETB         tick_ADC_alt
         RETI
         NOP
         SETB         tick_ADC_frnt
         RETI
         NOP
         SETB         tick_ADC_bat
         RETI


;****************    TIMER   *****************
SUB_TIMER:
        PUSH          ACC
```

```
42          PUSH            PSW
43          MOV             TH0,#0XB1
44          MOV             TL0,#0XDF
45          SETB            tick_10ms
46          INC             cont_10ms
47          MOV             A,cont_10ms
48          CLR             C
49          SUBB            A,#0x32
50          JC              FIN_SUBTIMER
51          MOV             cont_10ms,#0
52          SETB            tick_500ms
53          INC             cont_500ms_8s
54          INC             cont_500ms_10S
55          INC             cont_500ms_25S
56          MOV             A,cont_500ms_8s
57          CJNE            A,#0x10,SUB_TIMER_CONTINUA1
58          MOV             cont_500ms_8s,#0
59          SETB            tick_8s
60
61  SUB_TIMER_CONTINUA1:
62          MOV             A,cont_500ms_10s
63          CJNE            A,#0x14,SUB_TIMER_CONTINUA2
64          MOV             cont_500ms_10s,#0
65          SETB            tick_10s
66
67  SUB_TIMER_CONTINUA2:
68          MOV             A,cont_500ms_25s
69          CJNE            A,#0x32,FIN_SUBTIMER
70          MOV             cont_500ms_25s,#0
71          SETB            tick_25s
72
73  FIN_SUBTIMER:
74          POP             PSW
75          POP             ACC
76          RET
```

Listing 3: Interrupciones

5.3. Programa principal

```
;#########################################################
;                     PROGRAMA PRINCIPAL
;#########################################################
ORG     0x80

INICIO:
        ACALL       INICIALIZAR
BUCLE:
        ACALL       GEN_EVE
        ACALL       MAQ_ESTADOS
        AJMP        BUCLE


;#########################################################
;                     INICIALIZACIONES
;#########################################################

INICIALIZAR:
        MOV         ESTADO,#0X00
        MOV         EVENTO,#0X00
        MOV         BATERIA,#0X00
        MOV         PWM0, #0xFF
        MOV         PWM1, #0xFF
        MOV         PWMP, #0x5D
        CLR         P1.6
        CLR         P1.7
        CLR         par_avan
        CLR         auto_man
        CLR         g_izda
        CLR         g_dcha
        CLR         sube
        CLR         baja
        CLR         LED_rojo
        CLR         LED_verde
        CLR         desp_ater
        ANL         ADCON, #11011111b
        CLR         tick_10ms
        CLR         tick_500ms
        CLR         tick_8s
        CLR         tick_10s
        CLR         tick_25s
```

```
42          CLR         tick_ADC_alt
43          CLR         tick_ADC_frnt
44          CLR         tick_ADC_bat
45          CLR         tick_automan
46          CLR         bateria_baja
47          MOV         boton_antes,#0
48          MOV         est_antes,#0
49          MOV         est_cambio,#0
50          MOV         est_cambio_sube,#0
51          MOV         est_cambio_baja,#0
52          MOV         est_aux,#0
53          SETB        IE.5
54          SETB        IE.6
55          SETB        IE.7
56          ACALL       PROGRAM_TIMER
57          RET
58
59  ;#########################################################
60  ;                   GENERADOR DE EVENTOS
61  ;#########################################################
62
63  GEN_EVE:
64          MOV     boton_ahora, P0
65          ANL     boton_ahora, #10000000b
66          MOV     A,boton_ahora
67          XRL     A,boton_antes
68          JNZ     CAMBIO_EN_BOTON
69          JB      tick_ADC_bat, EV_7
70          JB      tick_ADC_alt, EV_8
71          JB      tick_ADC_frnt, EV_9
72          JB      tick_automan, EV_10
73          JB      tick_25s, EV_6
74          JB      tick_10s, EV_5
75          JB      tick_8s, EV_4
76          JB      tick_500ms, EV_3
77          JB      tick_10ms, EV_2
78          MOV     EVENTO,#0X00
79          RET
80
81  CAMBIO_EN_BOTON:
82          MOV     est_cambio,A
83          ANL     A,boton_ahora
84          JNZ     ENCIENDE
```

```
85          MOV     boton_antes,boton_ahora
86          MOV     EVENTO,#0X02
87          RET
88
89  ENCIENDE:
90          MOV     boton_antes,boton_ahora
91          MOV     EVENTO,#0X01
92          RET
93
94  EV_2:
95          CLR     tick_10ms
96          MOV     EVENTO,#0X03
97          RET
98  EV_3:
99          CLR     tick_500ms
100         MOV     EVENTO,#0X04
101         RET
102 EV_4:
103         CLR     tick_8s
104         MOV     EVENTO,#0X05
105         RET
106 EV_5:
107         CLR     tick_10s
108         MOV     EVENTO,#0X06
109         RET
110 EV_6:
111         CLR     tick_25s
112         MOV     EVENTO,#0X07
113         RET
114 EV_7:
115         CLR     tick_ADC_bat
116         MOV     EVENTO,#0X08
117         RET
118 EV_8:
119         CLR     tick_ADC_alt
120         MOV     EVENTO,#0X09
121         RET
122 EV_9:
123         CLR     tick_ADC_frnt
124         MOV     EVENTO,#0X0A
125         RET
126 EV_10:
127         CLR     tick_automan
```

```
128         MOV     EVENTO,#0X0B
129         RET
130 ;#######################################################
131 ;               MAQUINA DE ESTADOS
132 ;#######################################################
133
134 MAQ_ESTADOS:
135         MOV     A,ESTADO
136         RL      A
137         MOV     DPTR,#LISTA_EST
138         JMP     @A+DPTR
139
140 LISTA_EST:
141         AJMP    REPOSO
142         AJMP    DESPEGUE
143         AJMP    VUELO_ESTABLE
144         AJMP    MANUAL
145         AJMP    ATERRIZAJE
146         AJMP    SUBIR
147         AJMP    BAJAR
148         AJMP    GIRO_DCHA
149         AJMP    GIRO_IZQ
```

Listing 4: Programa principal

5.4. Estado 0

```
;#########################################################
;                Estado 0: REPOSO
;#########################################################

REPOSO:
        ACALL   MAQ_EVENTOS_0
        RET

MAQ_EVENTOS_0:
        MOV     A,EVENTO
        RL      A
        MOV     DPTR,#LISTA_EV_0
        JMP     @A+DPTR

LISTA_EV_0:
        RET
        NOP
        AJMP    EVENTOS0_1
        RET
        NOP
        AJMP    EVENTOS0_3
        RET
        NOP
        RET
        NOP
        AJMP    EVENTOS0_6
        RET
        NOP
        AJMP    EVENTOS0_8
        AJMP    EVENTOS0_9
        RET
        NOP
        RET
        NOP
        RET

EVENTOS0_1:
        MOV     PWM0,#0x4D
        MOV     ESTADO,#0x01
        RET

```

```
42 EVENTOS0_3:
43         CLR     tick_10ms
44         ANL     ADCON,#11111000b
45         ORL     ADCON,#08h
46         RET
47
48
49 EVENTOS0_6:
50         CLR     tick_10s
51         ANL     ADCON,#11111000b
52         ORL     ADCON,#0x02
53         ORL     ADCON,#08h
54         RET
55
56 EVENTOS0_8:
57         CLR     tick_ADC_bat
58         ACALL   MOSTRAR_DISPLAY
59         RET
60
61 EVENTOS0_9:
62         MOV     A,ADCH
63         JNZ     EVENTOS0_9_FIN
64         MOV     PWM0,#0xFF
65         MOV     PWM1,#0xFF
66         CLR     P1.7
67         CLR     P1.6
68
69 EVENTOS0_9_FIN:
70         RET
```

Listing 5: Estado 0

5.5. Estado 1

```
;#########################################################
;                Estado 1: DESPEGUE
;#########################################################


DESPEGUE:
        ACALL   MAQ_EVENTOS_1
        RET

MAQ_EVENTOS_1:
        MOV     A,EVENTO
        RL      A
        MOV     DPTR,#LISTA_EV_1
        JMP     @A+DPTR

LISTA_EV_1:
        RET
        NOP
        RET
        NOP
        AJMP    EVENTOS1_2
        AJMP    EVENTOS1_3
        RET
        NOP
        RET
        NOP
        AJMP    EVENTOS1_6
        RET
        NOP
        AJMP    EVENTOS1_8
        AJMP    EVENTOS1_9
        RET
        NOP
        AJMP    EVENTOS1_11
        RET

EVENTOS1_2:
        MOV     PWM0, #0xB3
        MOV     PWM1, #0xFF
        MOV     ESTADO, #0x04
        RET
```

```
42
43  EVENTOS1_3:
44          CLR     tick_10ms
45          ANL     ADCON,#11111000b
46          ORL     ADCON,#08h
47          RET
48
49
50  EVENTOS1_6:
51          CLR     tick_10s
52          ANL     ADCON,#11111000b
53          ORL     ADCON,#0x02
54          ORL     ADCON,#08h
55          RET
56
57  EVENTOS1_8:
58          CLR     tick_ADC_bat
59          ACALL   MOSTRAR_DISPLAY
60          JB      bateria_baja,EVENTOS1_8_CONTINUA
61          RET
62
63  EVENTOS1_8_CONTINUA:
64          CLR     bateria_baja
65          MOV     PWM0, #0xB3
66          MOV     PWM1, #0xFF
67          MOV     ESTADO, #0x04
68          RET
69
70  EVENTOS1_9:
71          CLR     tick_ADC_alt
72          MOV     A,ADCH
73          SUBB    A,#0x55
74          JNC     EVENTOS1_9_CONTINUA
75          RET
76
77  EVENTOS1_9_CONTINUA:
78          SETB    tick_automan
79          MOV     PWM0,#0x80
80          RET
81
82  EVENTOS1_11:
83          CLR     tick_automan
84          JB      auto_man,MODO_MANUAL
```

```
85          MOV     PWM1,#0x80
86          SETB    P1.7
87          SETB    P1.6
88          ACALL   PROGRAM_TIMER
89          MOV     ESTADO,#0x02
90          SETB    DISPLAY.7
91          RET
92
93  MODO_MANUAL:
94          CLR     DISPLAY.7
95          MOV     ESTADO,#0x03
96          RET
```

Listing 6: Estado 1

5.6. Estado 2

```
;#########################################################
;                Estado 2: VUELO ESTABLE
;#########################################################

VUELO_ESTABLE:
        ACALL   MAQ_EVENTOS_2
        RET

MAQ_EVENTOS_2:
        MOV     A,EVENTO
        RL      A
        MOV     DPTR,#LISTA_EV_2
        JMP     @A+DPTR

LISTA_EV_2:
        RET
        NOP
        RET
        NOP
        AJMP    EVENTOS2_2
        AJMP    EVENTOS2_3
        RET
        NOP
        RET
        NOP
        AJMP    EVENTOS2_6
        AJMP    EVENTOS2_7
        AJMP    EVENTOS2_8
        AJMP    EVENTOS2_9
        AJMP    EVENTOS2_10
        RET
        NOP
        RET

EVENTOS2_2:
        MOV     PWM0, #0xB3
        MOV     PWM1, #0xFF
        MOV     ESTADO, #0x04
        RET

EVENTOS2_3:
```

```
42          CLR     tick_10ms
43          ANL     ADCON,#11111000b
44          ORL     ADCON,#08h
45          RET
46
47  EVENTOS2_6:
48          CLR     tick_10s
49          ANL     ADCON,#11111000b
50          ORL     ADCON,#0x02
51          ORL     ADCON,#08h
52          RET
53
54  EVENTOS2_7:
55          CLR     tick_25s
56          CLR     P1.7
57          MOV     ESTADO,#0x08
58          RET
59
60  EVENTOS2_8:
61          CLR     tick_ADC_bat
62          ACALL   MOSTRAR_DISPLAY
63          JB      bateria_baja,EVENTOS2_8_CONTINUA
64          RET
65
66  EVENTOS2_8_CONTINUA:
67          CLR     bateria_baja
68          MOV     PWM0, #0xB3
69          MOV     PWM1, #0xFF
70          MOV     ESTADO, #0x04
71          RET
72
73  EVENTOS2_9:
74          CLR     tick_ADC_alt
75          ANL     ADCON, #11111000b
76          ORL     ADCON,#0x01
77          ORL     ADCON,#08h
78          MOV     A,ADCH
79          SUBB    A,#0x44
80          JNC     EVENTOS2_9_CONTINUA_1
81          MOV     PWM0,#0x4D
82          CLR     P1.7
83          CLR     P1.6
84          MOV     ESTADO,#0x05
```

```
85          RET
86
87  EVENTOS2_9_CONTINUA_1:
88          MOV     A,ADCH
89          SUBB    A,#0x66
90          JC      EVENTOS2_9_CONTINUA_2
91          MOV     PWM0,#0x4D
92          CLR     P1.7
93          CLR     P1.6
94          MOV     ESTADO,#0x06
95          RET
96
97  EVENTOS2_9_CONTINUA_2:
98          RET
99
100 EVENTOS2_10:
101         CLR     tick_ADC_frnt
102         MOV     A,ADCH
103         SUBB    A,#0x22
104         JNC     EVENTOS2_10_CONTINUA
105         SETB    P1.7
106         CLR     P1.6
107         ACALL   PROGRAM_TIMER
108         MOV     ESTADO,#0x07
109         RET
110
111 EVENTOS2_10_CONTINUA:
112         RET
```

Listing 7: Estado 2

5.7. Estado 3

```
;#########################################################
                Estado 3: MANUAL
;#########################################################


MANUAL:
        ACALL   MAQ_EVENTOS_3
        RET

MAQ_EVENTOS_3:
        MOV         A,EVENTO
        RL          A
        MOV         DPTR,#LISTA_EV_3
        JMP         @A+DPTR

LISTA_EV_3:
        RET
        NOP
        RET
        NOP
        AJMP    EVENTOS3_2
        AJMP    EVENTOS3_3
        RET
        NOP
        RET
        NOP
        AJMP    EVENTOS3_6
        RET
        NOP
        AJMP    EVENTOS3_8
        RET
        NOP
        RET
        NOP
        RET
        NOP
        RET

EVENTOS3_2:
        MOV    PWM0, #0xB3
        MOV    PWM1, #0xFF
```

```
42          MOV     ESTADO, #0x04
43          RET
44
45  EVENTOS3_3:
46          CLR     tick_10ms
47          MOV     est_ahora,P0
48          MOV     A, est_ahora
49          ANL     A,#01001111b
50          MOV     est_ahora,A
51          MOV     A,est_ahora
52          XRL     A,est_antes
53          JZ      EVENTOS3_3_SIN_CAMBIO
54          MOV     est_cambio,A
55          ANL     A,est_ahora
56          MOV     est_cambio_sube,A
57          MOV     A,est_cambio
58          ANL     A,est_antes
59          MOV     est_cambio_baja,A
60          JNB     est_cambio_sube.6, EVENTOS3_3_CONTINUA
61          SETB    est_antes.6
62          SETB    bit_aux_avanza
63          AJMP    EVENTOS3_3_FIN
64
65
66  EVENTOS3_3_SIN_CAMBIO:
67          MOV     est_antes,est_ahora
68          RET
69
70
71  EVENTOS3_3_CONTINUA:
72          JNB     est_cambio_baja.6,EVENTOS3_3_FIN
73          CLR     bit_aux_avanza
74          AJMP    EVENTOS3_3_FIN
75
76
77  EVENTOS3_3_FIN:
78          MOV     est_antes,est_ahora
79          JB      bit_aux_avanza, EVENTOS3_3_AUX
80          CLR     est_aux.4
81          CLR     est_aux.6
82          CLR     est_aux.7
83          CLR     est_aux.5
84          ANL     est_cambio_sube,#00001111b
```

```
85          ANL     est_cambio_baja,#00001111b
86          MOV     A, est_aux
87          XRL     A,est_cambio_sube
88          XRL     A,est_cambio_baja
89          MOV     est_aux,A
90          ACALL   TABLA_PWM0
91          MOV     PWM0,A
92          ACALL   TABLA1_PWM1
93          MOV     PWM1,A
94          ACALL   TABLA2_PWM1
95          ACALL   PWM1_AUX
96          RET
97
98  EVENTOS3_3_AUX:
99          SETB    est_aux.4
100         CLR     est_aux.6
101         CLR     est_aux.7
102         CLR     est_aux.5
103         ANL     est_cambio_sube,#00001111b
104         ANL     est_cambio_baja,#00001111b
105         MOV     A, est_aux
106         XRL     A,est_cambio_sube
107         XRL     A,est_cambio_baja
108         MOV     est_aux,A
109         ACALL   TABLA_PWM0
110         MOV     PWM0,A
111         ACALL   TABLA1_PWM1
112         MOV     PWM1,A
113         ACALL   TABLA2_PWM1
114         ACALL   PWM1_AUX
115         RET
116
117 PWM1_AUX:
118          CJNE    A, #0x00, PWM1_AUX1
119          CLR     P1.7
120          CLR     P1.6         ; PWM1 parado
121          RET
122
123 PWM1_AUX1:
124     CJNE    A, #0x01, PWM1_AUX2
125     CLR     P1.7
126     SETB    P1.6         ; PWM1 izquierda
127     RET
```

```
128
129 PWM1_AUX2:
130     CJNE    A, #0x02, PWM1_AUX3
131     SETB    P1.7
132     CLR     P1.6        ; PWM1 derecha
133     RET
134
135 PWM1_AUX3:
136     SETB    P1.7
137     SETB    P1.6        ; PWM1 avanza
138     RET
139
140 TABLA_PWM0:
141         ; 1. TAULA - PWM0
142         MOV A,est_aux
143         INC A
144         MOVC A,@A+PC
145         RET
146         DB 0xFF ; 0%
147         DB 0xB3 ; 30%
148         DB 0x4D ; 70%
149         NOP ; [-]
150         DB 0x80 ; %50
151         DB 0xB3 ; %30
152         DB 0x4D ; %70
153         NOP ; [-]
154         DB 0x80 ; %50
155         DB 0xB3 ; %30
156         DB 0x4D ; %70
157         NOP ; [-]
158         NOP ; [-]
159         NOP ; [-]
160         NOP ; [-]
161         NOP ; [-]
162         DB 0x80 ; %50
163         DB 0xB3 ; %30
164         DB 0x4D ; %70
165         NOP ; [-]
166         DB 0x80 ; %50
167         DB 0xB3 ; %30
168         DB 0x4D ; %70
169         NOP ; [-]
170         DB 0x80 ; %50
```

```
171         DB 0xB3 ; %30
172         DB 0x4D ; %70
173         NOP ; [-]
174         NOP ; [-]
175         NOP ; [-]
176         NOP ; [-]
177         NOP ; [-]
178
179 TABLA1_PWM1:
180         ; 2. TAULA - PWM1
181         MOV A,est_aux
182         INC A
183         MOVC A,@A+PC
184         RET
185         DB 0xFF ; 0%
186         DB 0xFF ; 0%
187         DB 0xFF ; 0%
188         NOP ; [-]
189         DB 0x80 ; %50
190         DB 0x80 ; %50
191         DB 0x80 ; %50
192         NOP ; [-]
193         DB 0x80 ; %50
194         DB 0x80 ; %50
195         DB 0x80 ; %50
196         NOP ; [-]
197         NOP ; [-]
198         NOP ; [-]
199         NOP ; [-]
200         NOP ; [-]
201         DB 0x80 ; %50
202         DB 0x80 ; %50
203         DB 0x80 ; %50
204         NOP ; [-]
205         DB 0x80 ; %50
206         DB 0x80 ; %50
207         DB 0x80 ; %50
208         NOP ; [-]
209         DB 0x80 ; %50
210         DB 0x80 ; %50
211         DB 0x80 ; %50
212         NOP ; [-]
213         NOP ; [-]
```

```
214         NOP ; [-]
215         NOP ; [-]
216         NOP ; [-]
217
218 TABLA2_PWM1:
219         ; 3. TAULA - PWM1 CONTROL
220         MOV A,est_aux
221            INC A
222         MOVC A,@A+PC
223         RET
224         DB 0x00 ; 00 parado
225         DB 0x00 ; 00
226         DB 0x00 ; 00
227         NOP ; [-]
228         DB 0x02 ; 10 g_dcha
229         DB 0x02 ; 10
230         DB 0x02 ; 10
231         NOP ; [-]
232         DB 0x01 ; 01 g_izq
233         DB 0x01 ; 01
234         DB 0x01 ; 01
235         NOP ; [-]
236         NOP ; [-]
237         NOP ; [-]
238         NOP ; [-]
239         NOP ; [-]
240         DB 0x03 ; 11 avanza
241         DB 0x03 ; 11
242         DB 0x03 ; 11
243         NOP ; [-]
244         DB 0x02 ; 10 g_dcha
245         DB 0x02 ; 10
246         DB 0x02 ; 10
247         NOP ; [-]
248         DB 0x01 ; 01 g_izq
249         DB 0x01 ; 01
250         DB 0x01 ; 01
251         NOP ; [-]
252         NOP ; [-]
253         NOP ; [-]
254         NOP ; [-]
255         NOP ; [-]
256
```

```
257 EVENTOS3_6:
258         CLR     tick_10s
259         ANL     ADCON,#11111000b
260         ORL     ADCON,#0x02
261         ORL     ADCON,#08h
262         RET
263
264 EVENTOS3_8:
265         CLR     tick_ADC_bat
266         ACALL   MOSTRAR_DISPLAY
267         JB      bateria_baja,EVENTOS3_8_CONTINUA
268         RET
269
270 EVENTOS3_8_CONTINUA:
271         CLR     bateria_baja
272         MOV     PWM0, #0xB3
273         MOV     PWM1, #0xFF
274         MOV     ESTADO, #0x04
275         RET
```

Listing 8: Estado 3

5.8. Estado 4

```
;#########################################################
;                Estado 4: ATERRIZAJE
;#########################################################

ATERRIZAJE:
        ACALL   MAQ_EVENTOS_4
        RET

MAQ_EVENTOS_4:
        MOV         A,EVENTO
        RL          A
        MOV         DPTR,#LISTA_EV_4
        JMP         @A+DPTR

LISTA_EV_4:
        RET
        NOP
        RET
        NOP
        RET
        NOP
        AJMP    EVENTOS4_3
        RET
        NOP
        RET
        NOP
        AJMP    EVENTOS4_6
        RET
        NOP
        AJMP    EVENTOS4_8
        AJMP    EVENTOS4_9
        RET
        NOP
        RET
        NOP
        RET

EVENTOS4_3:
        ANL     ADCON,#11111000b
        ORL     ADCON,#08h
        RET
```

```
42
43 EVENTOS4_6:
44         CLR     tick_10s
45         ANL     ADCON,#11111000b
46         ORL     ADCON,#0x02
47         ORL     ADCON,#08h
48         RET
49
50 EVENTOS4_8:
51         CLR     tick_ADC_bat
52         ACALL   MOSTRAR_DISPLAY
53         RET
54
55 EVENTOS4_9:
56         CLR     tick_ADC_alt
57         MOV     A,ADCH
58         SUBB    A,#0x01
59         JC      EVENTOS4_9_CONTINUA
60         RET
61
62 EVENTOS4_9_CONTINUA:
63         MOV     PWM0,#0xFF
64         MOV     PWM1,#0xFF
65         CLR     P1.7
66         CLR     P1.6
67         CLR     desp_ater
68         MOV     ESTADO, #0x00
69         RET
```

Listing 9: Estado 4

5.9. Estado 5

```
;#########################################################
;                Estado 5: SUBIR
;#########################################################

SUBIR:
        ACALL   MAQ_EVENTOS_5
        RET

MAQ_EVENTOS_5:
        MOV         A,EVENTO
        RL          A
        MOV         DPTR,#LISTA_EV_5
        JMP         @A+DPTR

LISTA_EV_5:
        RET
        NOP
        RET
        NOP
        AJMP    EVENTOS5_2
        AJMP    EVENTOS5_3
        RET
        NOP
        RET
        NOP
        AJMP    EVENTOS5_6
        RET
        NOP
        AJMP    EVENTOS5_8
        AJMP    EVENTOS5_9
        RET
        NOP
        RET
        NOP
        RET


EVENTOS5_2:
        MOV     PWM0, #0xB3
        MOV     PWM1, #0xFF
        MOV     ESTADO, #0x04
```

```
42          RET
43
44  EVENTOS5_3:
45          CLR     tick_10ms
46          ANL     ADCON,#11111000b
47          ORL     ADCON,#08h
48          RET
49
50  EVENTOS5_6:
51          CLR     tick_10s
52          ANL     ADCON,#11111000b
53          ORL     ADCON,#0x02
54          ORL     ADCON,#08h
55          RET
56
57  EVENTOS5_8:
58          CLR     tick_ADC_bat
59          ACALL   MOSTRAR_DISPLAY
60          JB      bateria_baja,EVENTOS5_8_CONTINUA
61          RET
62
63  EVENTOS5_8_CONTINUA:
64          CLR     bateria_baja
65          MOV     PWM0, #0xB3
66          MOV     PWM1, #0xFF
67          MOV     ESTADO, #0x04
68          RET
69
70  EVENTOS5_9:
71          CLR     tick_ADC_alt
72          MOV     A,ADCH
73          SUBB    A,#0x55
74          JC      EVENTOS5_9_CONTINUA
75          MOV     PWM1,#0x80
76          SETB    P1.7
77          SETB    P1.6
78          MOV     PWM0,#0x80
79          MOV     ESTADO,#0x02
80          RET
81
82  EVENTOS5_9_CONTINUA:
83         RET
```

Listing 10: Estado 5

5.10. Estado 6

```
;#########################################################
;                Estado 6: BAJAR
;#########################################################

BAJAR:
        ACALL   MAQ_EVENTOS_6
        RET

MAQ_EVENTOS_6:
        MOV     A,EVENTO
        RL          A
        MOV         DPTR,#LISTA_EV_6
        JMP         @A+DPTR

LISTA_EV_6:
        RET
        NOP
        RET
        NOP
        AJMP    EVENTOS6_2
        AJMP    EVENTOS6_3
        RET
        NOP
        RET
        NOP
        AJMP    EVENTOS6_6
        RET
        NOP
        AJMP    EVENTOS6_8
        AJMP    EVENTOS6_9
        RET
        NOP
        RET
        NOP
        RET


EVENTOS6_2:
        MOV     PWM0, #0xB3
        MOV     PWM1, #0xFF
        MOV     ESTADO, #0x04
```

```
42          RET
43
44  EVENTOS6_3:
45          CLR     tick_10ms
46          ANL     ADCON,#11111000b
47          ORL     ADCON,#08h
48          RET
49
50  EVENTOS6_6:
51          CLR     tick_10s
52          ANL     ADCON,#11111000b
53          ORL     ADCON,#0x02
54          ORL     ADCON,#08h
55          RET
56
57  EVENTOS6_8:
58          CLR     tick_ADC_bat
59          ACALL   MOSTRAR_DISPLAY
60          JB      bateria_baja,EVENTOS6_8_CONTINUA
61          RET
62
63  EVENTOS6_8_CONTINUA:
64          CLR     bateria_baja
65          MOV     PWM0, #0xB3
66          MOV     PWM1, #0xFF
67          MOV     ESTADO, #0x04
68          RET
69
70  EVENTOS6_9:
71          ;tick_ADC_alt
72          CLR     tick_ADC_alt
73          MOV     A,ADCH
74          SUBB    A,#0x55
75          JC      EVENTOS6_9_CONTINUA
76          MOV     PWM1,#0x80
77          SETB    P1.7
78          SETB    P1.6
79          MOV     PWM0,#0x80
80          MOV     ESTADO,#0x02
81          RET
82
83  EVENTOS6_9_CONTINUA:
84          RET
```

Listing 11: Estado 6

5.11. Estado 7

```
;#########################################################
;                Estado 7: GIRO_DCHA (por obstaculo)
;#########################################################

GIRO_DCHA:
        ACALL   MAQ_EVENTOS_7
        RET

MAQ_EVENTOS_7:
        MOV         A,EVENTO
        RL          A
        MOV         DPTR,#LISTA_EV_7
        JMP         @A+DPTR

LISTA_EV_7:
        RET
        NOP
        RET
        NOP
        AJMP    EVENTOS7_2
        AJMP    EVENTOS7_3
        RET
        NOP
        AJMP    EVENTOS7_5
        AJMP    EVENTOS7_6
        RET
        NOP
        AJMP    EVENTOS7_8
        RET
        NOP
        AJMP    EVENTOS7_10
        RET
        NOP
        RET


EVENTOS7_2:
        MOV     PWM0, #0xB3
        MOV     PWM1, #0xFF
        MOV     ESTADO, #0x04
        RET
```

```
42
43  EVENTOS7_3:
44          CLR     tick_10ms
45          ANL     ADCON,#11111000b
46          ORL     ADCON,#0x01
47          ORL     ADCON,#08h
48          RET
49
50  EVENTOS7_5:
51          CLR     tick_8s
52          MOV     PWM0, #0xB3
53          MOV     PWM1, #0xFF
54          MOV     ESTADO, #0x04
55          RET
56
57  EVENTOS7_6:
58          CLR     tick_10s
59          ANL     ADCON,#11111000b
60          ORL     ADCON,#0x02
61          ORL     ADCON,#08h
62          RET
63
64  EVENTOS7_8:
65          CLR     tick_ADC_bat
66          ACALL   MOSTRAR_DISPLAY
67          JB      bateria_baja,EVENTOS7_8_CONTINUA
68          RET
69
70  EVENTOS7_8_CONTINUA:
71          CLR     bateria_baja
72          MOV     PWM0, #0xB3
73          MOV     PWM1, #0xFF
74          MOV     ESTADO, #0x04
75          RET
76
77  EVENTOS7_10:
78          MOV     A,ADCH
79          SUBB    A,#0x44
80          JNC     EVENTOS7_10_CONTINUA
81          RET
82
83  EVENTOS7_10_CONTINUA:
84          SETB    P1.7
```

```
85          SETB    P1.6
86          MOV     ESTADO,#0x02
87          RET
```

Listing 12: Estado 7

5.12. Estado 8

```
;#########################################################
;                Estado 8: GIRO_IZQ (por timeout)
;#########################################################

GIRO_IZQ:
        ACALL   MAQ_EVENTOS_8
        RET

MAQ_EVENTOS_8:
        MOV         A,EVENTO
        RL          A
        MOV         DPTR,#LISTA_EV_8
        JMP         @A+DPTR

LISTA_EV_8:
        RET
        NOP
        RET
        NOP
        AJMP    EVENTOS8_2
        RET
        NOP
        AJMP    EVENTOS8_4
        RET
        NOP
        AJMP    EVENTOS8_6
        RET
        NOP
        AJMP    EVENTOS8_8
        RET
        NOP
        RET
        NOP
        RET
        NOP
        RET


EVENTOS8_2:
        MOV     PWM0, #0xB3
        MOV     PWM1, #0xFF
```

```
42          MOV     ESTADO, #0x04
43          RET
44
45  EVENTOS8_4:
46          CLR     tick_500ms
47          SETB    P1.7
48          SETB    P1.6
49          MOV     ESTADO,#0x02
50          RET
51
52  EVENTOS8_6:
53          CLR     tick_10s
54          ANL     ADCON,#11111000b
55          ORL     ADCON,#0x02
56          ORL     ADCON,#08h
57          RET
58
59  EVENTOS8_8:
60          CLR     tick_ADC_bat
61          ACALL   MOSTRAR_DISPLAY
62          JB      bateria_baja,EVENTOS8_8_CONTINUA
63          RET
64
65  EVENTOS8_8_CONTINUA:
66          CLR     bateria_baja
67          MOV     PWM0, #0xB3
68          MOV     PWM1, #0xFF
69          MOV     ESTADO, #0x04
70          RET
```

Listing 13: Estado 8

5.13. Subrutinas

```
;#########################################################
;                 SUBRRUTINAS
;#########################################################

;************  PROGRAMAR TIMER  *****************

PROGRAM_TIMER:
        MOV         TH0,#0XB1
        MOV         TL0,#0XDF
        MOV         cont_10ms,#0
        MOV     cont_500ms_8s,#0
        MOV     cont_500ms_10s,#0
        MOV     cont_500ms_25s,#0
        CLR         tick_10ms
        CLR         tick_500ms
        CLR         tick_8s
        CLR     tick_10s
        CLR         tick_25s
        MOV         TMOD,#00000001
        SETB    TCON.4
        SETB    ET0
        SETB    EA
        RET

;************  VISUALIZAR DISPLAY   **************

MOSTRAR_DISPLAY:
        MOV     BATERIA,ADCH

DISPLAY_L:
        MOV     A,BATERIA
        SUBB    A,#0x01
        JNC     DISPLAY_1
        MOV     A,#0x00
        SETB    bateria_baja
        AJMP    FIN_DISPLAY
DISPLAY_1:
        MOV     A,BATERIA
        SUBB    A,#0x03
        JNC     DISPLAY_2
        MOV     A,#0x01
```

```
42          AJMP    FIN_DISPLAY
43  DISPLAY_2:
44          MOV     A,BATERIA
45          SUBB    A,#0x06
46          JNC     DISPLAY_3
47          MOV     A,#0x02
48          AJMP    FIN_DISPLAY
49  DISPLAY_3:
50          MOV     A,BATERIA
51          SUBB    A,#0x08
52          JNC     DISPLAY_4
53          MOV     A,#0x03
54          AJMP    FIN_DISPLAY
55  DISPLAY_4:
56          MOV     A,BATERIA
57          SUBB    A,#0x0B
58          JNC     DISPLAY_5
59          MOV     A,#0x04
60          AJMP    FIN_DISPLAY
61  DISPLAY_5:
62          MOV     A,BATERIA
63          SUBB    A,#0x0D
64          JNC     DISPLAY_6
65          MOV     A,#0x05
66          AJMP    FIN_DISPLAY
67  DISPLAY_6:
68          MOV     A,BATERIA
69          SUBB    A,#0x10
70          JNC     DISPLAY_7
71          MOV     A,#0x06
72          AJMP    FIN_DISPLAY
73  DISPLAY_7:
74          MOV     A,BATERIA
75          SUBB    A,#0x12
76          JNC     DISPLAY_8
77          MOV     A,#0x07
78          AJMP    FIN_DISPLAY
79  DISPLAY_8:
80          MOV     A,BATERIA
81          SUBB    A,#0x15
82          JNC     DISPLAY_9
83          MOV     A,#0x08
84          AJMP    FIN_DISPLAY
```

```
85  DISPLAY_9:
86          MOV     A,BATERIA
87          SUBB    A,#0x17
88          JNC     DISPLAY_F
89          MOV     A,#0x09
90          AJMP    FIN_DISPLAY
91  DISPLAY_F:
92          MOV     A,#0x0A
93
94  FIN_DISPLAY:
95          ACALL   TABLA_DISPLAY
96          MOV     C,DISPLAY.7
97          MOV     DISPLAY,A
98          MOV     DISPLAY.7,C
99          ACALL   SUB_LED
100         RET
101
102 TABLA_DISPLAY:
103         INC     A
104         MOVC    A,@A+PC
105         RET
106         DB      10111000b ; (L)
107         DB      10000110b ; 1
108         DB      11011011b ; 2
109         DB      11001111b ; 3
110         DB      11100110b ; 4
111         DB      11101101b ; 5
112         DB      11111101b ; 6
113         DB      10000111b ; 7
114         DB      11111111b ; 8
115         DB      11101111b ; 9
116         DB      11110001b ; F
117         RET
118
119 SUB_LED:
120         CLR     C
121         MOV     A,BATERIA
122         SUBB    A,#0x02
123         JC      SUB_LED_BATERIA_BAJA
124         SETB    LED_verde
125         CLR     LED_rojo
126         RET
127
```

```
128 SUB_LED_BATERIA_BAJA:
129         CLR     LED_verde
130         SETB    LED_rojo
131         RET
```

Listing 14: Subrutinas

6. Observaciones

- Para la creación del diagrama de estados, eventos y acciones se ha utilizado **Microsoft Word** y **Adobe Photoshop.**
- Para la creación de los diagramas de flujo se ha utilizado **Draw.io**

7. Bibliografía

- Plataforma docente eGela
- 80C51 Family Program Guide
- 80C51 Family Derivatives
- 80C552 Chip Datasheet
- Libro ***Sistemas de Procesamiento Digital*** escrito por Aitzol Zuloaga Izaguirre y Armando Astarloa Cuéllar