

Universidad de Costa Rica
Escuela de Ingeniería Eléctrica
IE0623 - Microprocesadores. IS 2024
Informe del Proyecto final
Radar623

Juan Ignacio Hernández Zamora (B93826)

15 de julio de 2024

Índice

1. Resumen	2
2. Diseño de la aplicación	3
2.1. Comportamiento general del programa	3
2.2. Tarea LeerDS	5
2.3. Tarea Modo Inactivo	7
2.4. Tarea Configurar	10
2.5. Tarea EnServicio	13
2.6. Tarea DsplzLeds	18
2.7. Tarea Brillo	21
2.8. Tarea Teclado	24
2.9. Tarea Led Testigo	29
2.10. Tarea Leer PB0	31
2.11. Tarea Leer PB1	35
2.12. Tarea PantallaMUX	39
2.13. Tarea LCD	42
2.14. Tarea Send LCD	44
2.15. Subrutina BCD BIN	47
2.16. Subrutina Calcula	49
2.17. Subrutina BIN BCD MUXP	52
2.18. Subrutina BCD 7SEG	54
2.19. Subrutina Borrar NumArray	56
2.20. Subrutina Leer Teclado	58
2.21. Máquina de tiempos	60
2.22. Rutina de inicialización de la pantalla LCD	63
3. Conclusiones	65
4. Comentarios personales	66
5. Recomendaciones	67
6. Bibliografía	68

1. Resumen

El objetivo del proyecto es implementar, mediante una parte del *hardware* de la tarjeta *Dragon12+*, el sistema de control de un radar vehicular que mide la velocidad promedio de los automóviles que circulan por una carretera y la compara con un valor límite que se puede configurar en tiempo de ejecución. También se despliegan resultados de manera concordante a la situación que suceda: exceso de velocidad y conformidad con el tope dado.

El sistema está conformado por una pantalla LCD, dos *displays* de 7 segmentos, 5 leds de alarma, 3 leds de modo de operación, 2 *dipswitches* para seleccionar cómo se opera y un teclado multiplexado para seleccionar los valores límites de velocidad. Además se tienen dos botones de pulsación que emulan la activación de los sensores ultrasónicos que detectan el paso de un automotor.

En cuanto a la topología del sistema a implementar, los sensores están separados por 40 metros y la pantalla de información se encuentra a 300 metros del segundo sensor, siguiendo el sentido de circulación de la carretera. El *hardware* de configuración se ubica en un panel de control

El principio de funcionamiento del sistema está dado por la relación lineal entre la velocidad promedio, la distancia y el tiempo de un objeto modelado como un cuerpo libre:

$$v = \frac{d}{t} \quad (1)$$

Así es posible, mediante el cálculo de fracciones, estimar la velocidad del automóvil.

Dadas estas situaciones, se identifica el sistema a desarrollar como un *sistema operativo en tiempo real*¹ según la definición dada por [1], donde los definen como: “aquellos sistemas operativos que requieren manejar múltiples actividades de manera temporizada con recursos limitados” (p. 511).

El sistema logrado cumple con todos los requisitos solicitados dados que logra emular efectivamente el funcionamiento del radar propuesto. Se logró calcular las velocidades en el procesador de punto fijo, como lo es el *9S12* del que se dispone, mediante la computación de fracciones al multiplicar y dividir los valores ingresados por constantes definidas en tiempo de compilación.

¹RTOS, por sus siglas en inglés.

2. Diseño de la aplicación

Dada la complejidad del problema propuesto, se detalla primeramente el funcionamiento a nivel general para luego estudiar con detalle las subrutinas y tareas desarrolladas.

2.1. Comportamiento general del programa

Ante el inicio de un ciclo de *power-on reset*, se empieza por relocalizar el vector de interrupciones de *output compare* del contador en el canal 5 a la máquina de tiempos desarrollada en el curso, que se estudia más adelante.

Después de esto establece las etiquetas de estructuras de datos necesarias, configura los periféricos operados mediante *memory mapped I/O*, inicializa las estructuras de datos que lo requieren así como setea los estados iniciales de las máquinas de estado en el primero.

Terminado esto, se inicializa la pantalla LCD dado que esta requiere un proceso especial, detallado en el estudio de la rutina *Init_LCD*.

Finalmente, se mantiene ejecutando un despachador de tareas que inicia la ejecución de las máquinas de estado y subrutinas siguiendo el orden de primero las tareas generadoras de datos y luego las consumidoras. Además, se prioriza la impresión de mensajes en la LCD luego de terminar un estado de cualquier máquina, cuando la pantalla está en estado ocioso no se actualiza.

El diagrama de flujo del sistema se encuentra en la siguiente página, las estructuras de datos se detallan individualmente por tarea, rutina o subrutina y luego de la figura 1 se estudian una por una en el orden dado en el enunciado del proyecto.

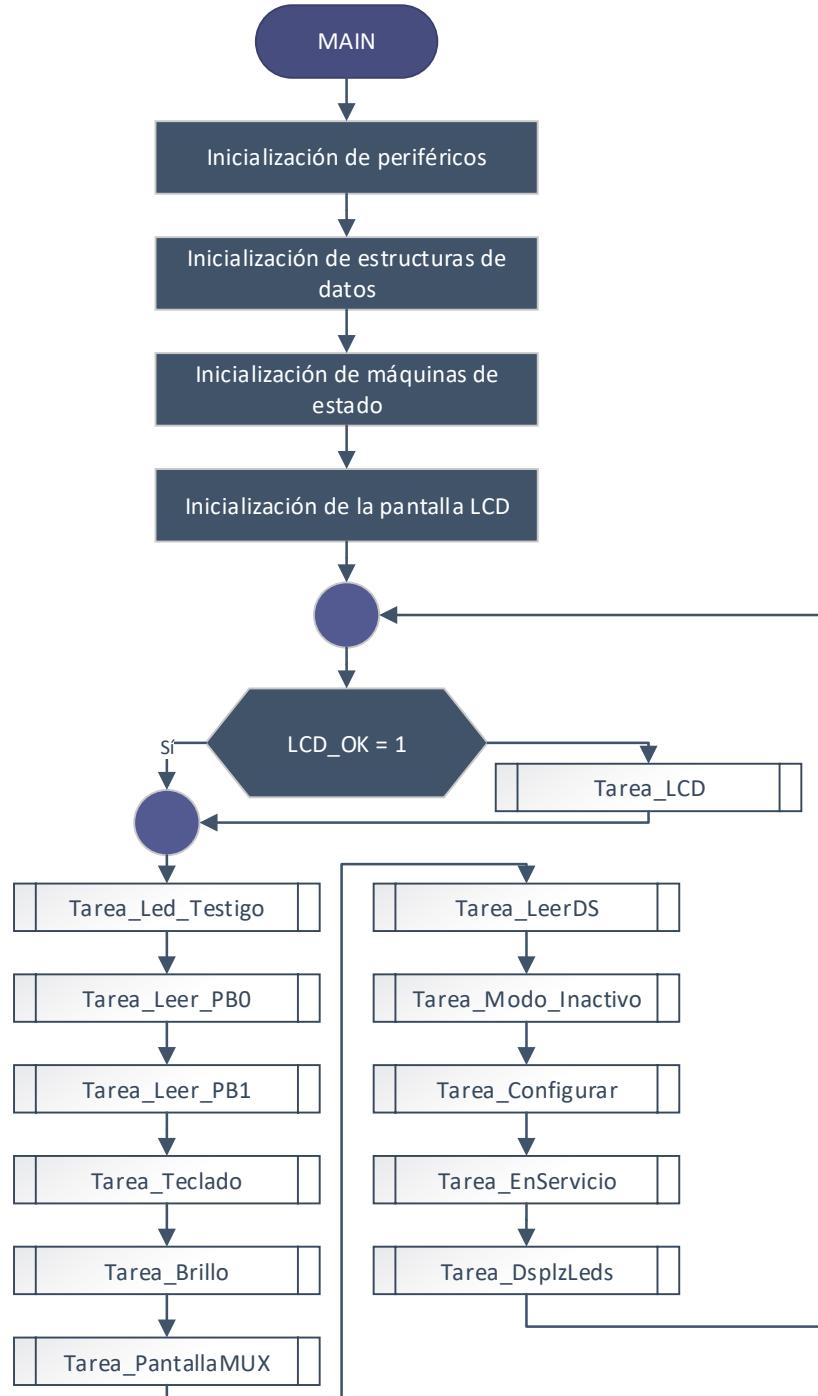


Figura 1: Diagrama de flujo del sistema operativo en tiempo real.

2.2. Tarea LeerDS

Esta tarea lee el estado de los *dipswitches* del puerto *H* con el fin de poder seleccionar el modo de operación del radar.

Se utilizaron las estructuras de datos y valores de la tabla 1.

Cuadro 1: Valores y estructuras de datos utilizados en Tarea LeerDS.

Nombre	Tipo	Magnitud o tamaño	Descripción
tTimerRebDS	Valor	10	Constante del timer de rechazo de rebotes para DS.
PortDS	Valor	PTIH	Puerto para leer los DS.
Mask_Mode_Sel	Valor	\$C0	Constante para detectar qué dipswitches activan cuáles modos.
Est_Pres_LeerDS	Variable	Word	Estado actual de la máquina que lee dipswitches
Temp_DS	Variable	Byte	Valor temporal de DS mientras se rechaza los rebotes.
Valor_DS	Variable	Byte	Valor final de los dipswitches luego de rechazar los rebotes.
Timer_Reb	Variable timer	Byte	Timer de rechazo de rebote de DS, base 1mS

Seguidamente se presentan el diagrama de flujo de la tarea en 2:

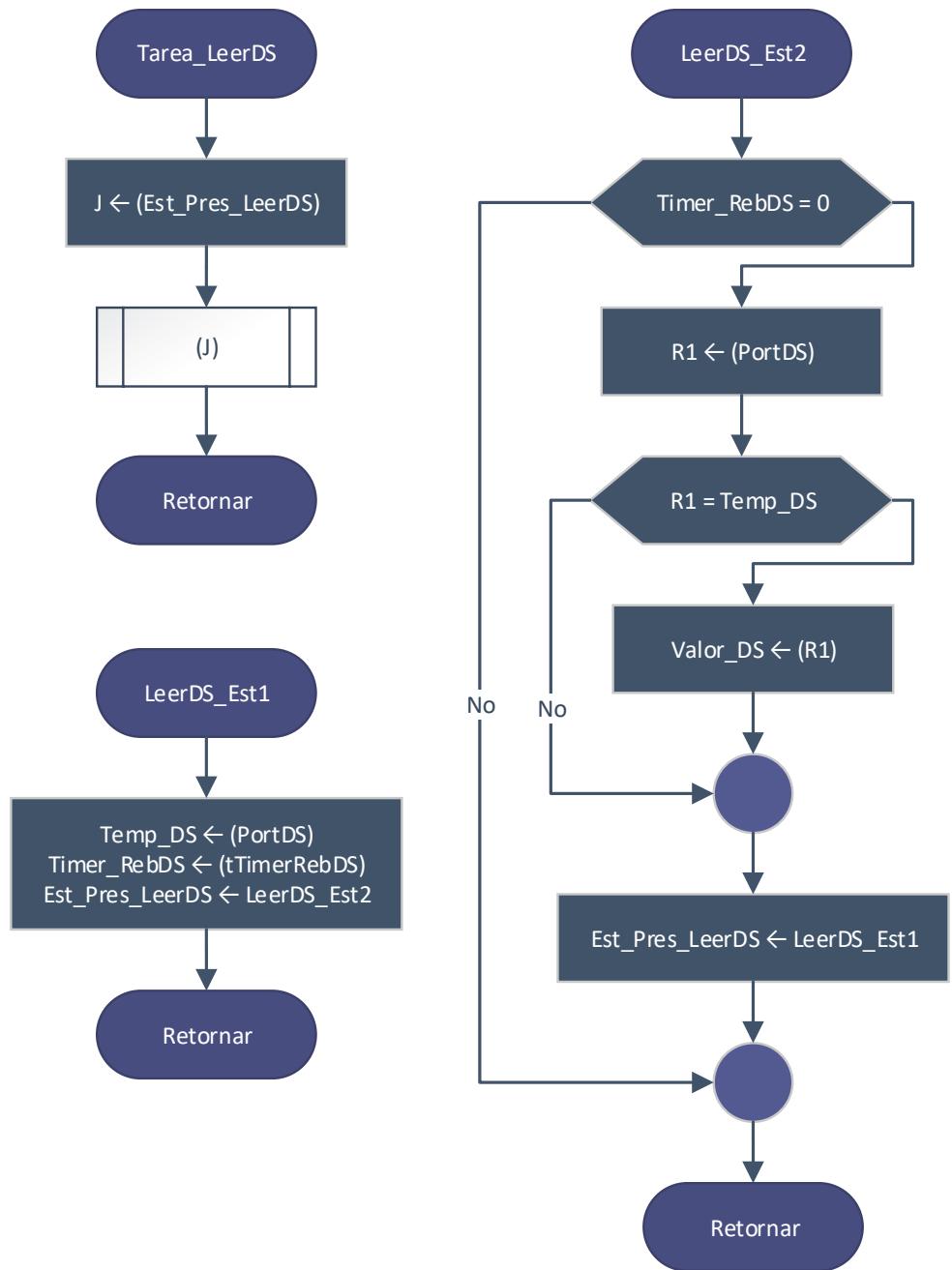


Figura 2: Diagrama de flujo de la tarea Leer DS.

2.3. Tarea Modo Inactivo

Esta tarea representa el estado ocioso del radar: no se está esperando ningún vehículo y no se va a cambiar su configuración. En él se puede consultar el valor actual de la velocidad límite al ingresar un *long press* al botón *PH3* o *PH0*.

En el enunciado se permitió elegir un *timer* para el tiempo que se requiere esperar, se tomó *TimerPant* dado que la ejecución de los modos de operación es mutuamente excluyente.

Las estructuras de datos se encuentran en la tabla 2.

Cuadro 2: Valores y estructuras de datos utilizados en la tarea del Modo Inactivo.

Nombre	Tipo	Magnitud o tamaño	Descripción
LDInactivo	Valor	\$01	Led PB0. Led a encender en el modo inactivo.
tTimerVLim	Valor	30	Valor del timer para esperar 3 segundos, 30*100mS
Mask_DS_MD_Ina	Valor	0	Disposición de los dipswitches para el modo inactivo.
Est_Pres_TInac	Variable	Word	Estado actual de la máquina para tarea inactivo
TimerPant	Variable timer	Byte	Temporizador para esperar a mientras se muestra la velocidad, base 100mS.
Dsp1	Variable	Byte	Valor del dígito 1 codificado para 8 segmentos
Dsp2	Variable	Byte	Valor del dígito 2 codificado para 8 segmentos
Dsp3	Variable	Byte	Valor del dígito 3 codificado para 8 segmentos
Dsp4	Variable	Byte	Valor del dígito 4 codificado para 8 segmentos
Segment	Valor	\$1100	Tabla de equivalencias para códigos BCD-7seg
LongP0	Valor	\$02	Valor para identificar un pulso largo en PH.0.
LongP1	Valor	\$08	Valor para identificar un pulso largo en PH.0.

El diagrama de flujo de la tarea se muestra en las figuras 3 y 4:

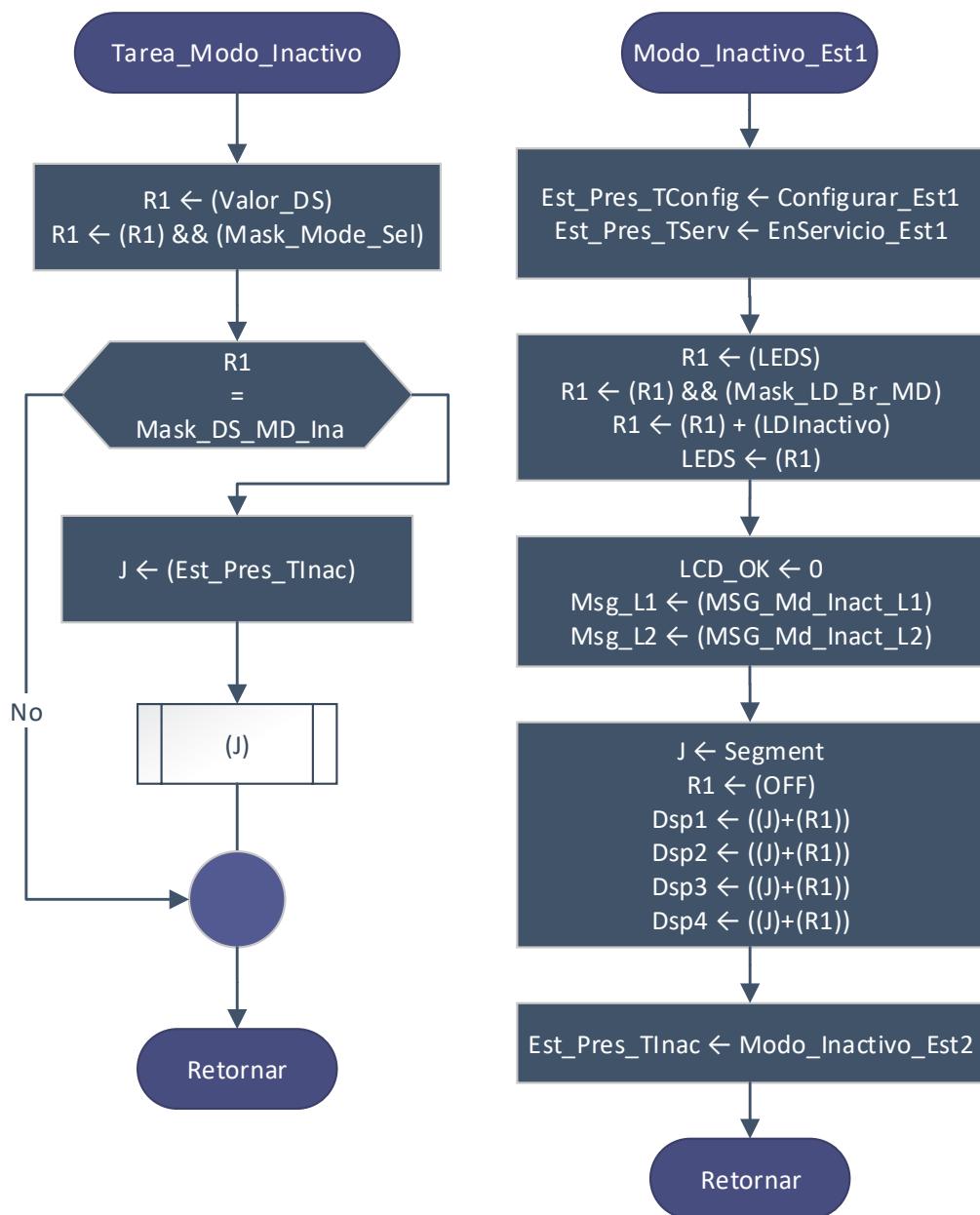


Figura 3: Diagrama de flujo de la tarea Modo Inactivo, parte 1.

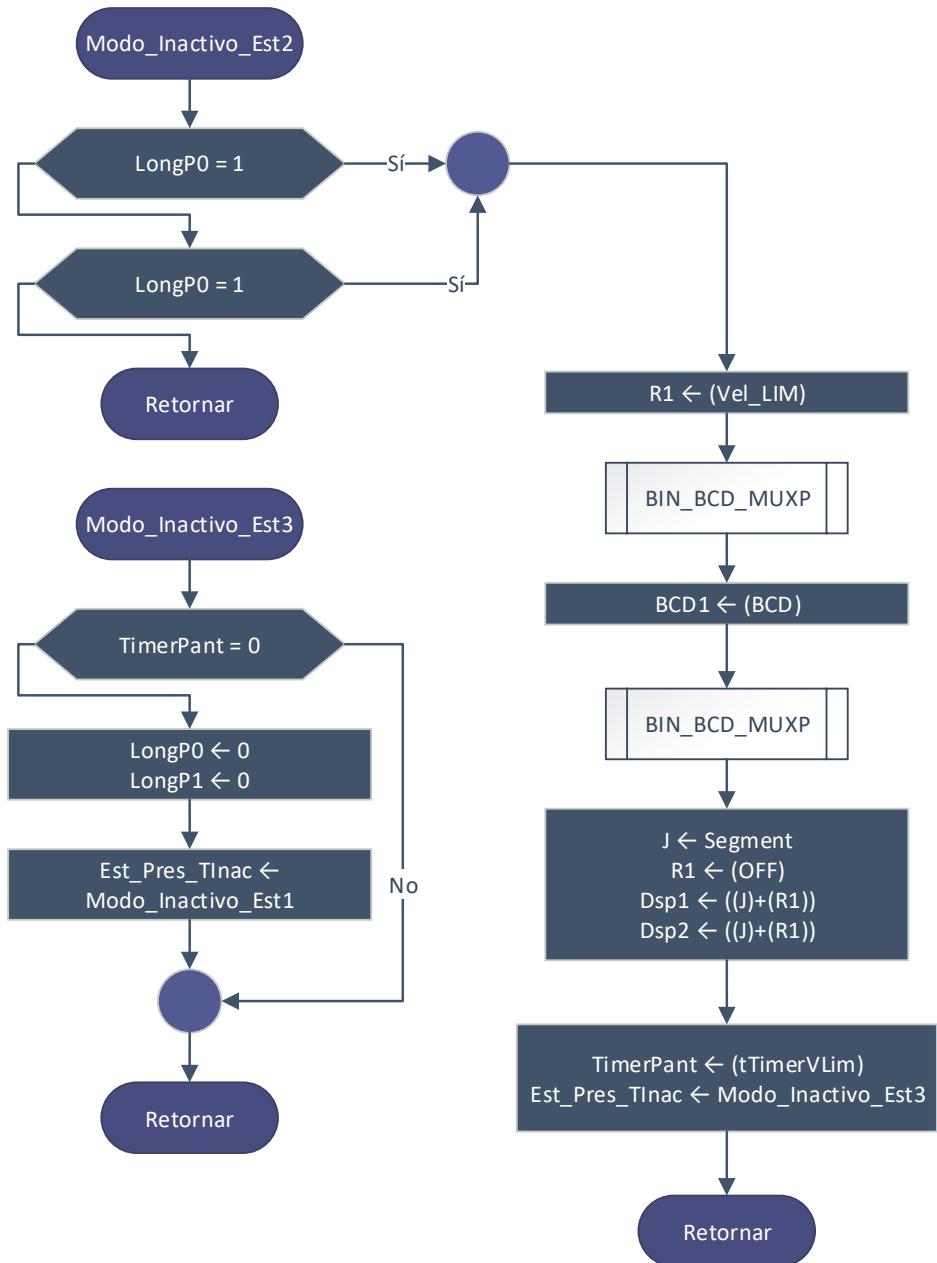


Figura 4: Diagrama de flujo de la tarea Modo Inactivo, parte 2.

2.4. Tarea Configurar

Esta tarea permite cambiar en tiempo de ejecución el valor de la velocidad límite que detecta el radar. La velocidad límite por defecto, luego de un ciclo de *power-on reset* se escogió para que fuera de 65 km/h.

Las estructuras de datos se encuentran en la tabla 3.

Cuadro 3: Valores y estructuras de datos utilizados en el modo Configurar.

Nombre	Tipo	Magnitud o tamaño	Descripción
LDConfig	Valor	\$02	Led PB1. Led a encender en el modo configurar.
LimMax	Valor	90	Límite de velocidad máxima.
LimMin	Valor	65	Límite mínimo de la velocidad.
Mask_DS_MD_Cfg	Valor	\$40	Disposición de los dipswitches para el modo configurar.
Def_Vel_LIM	Valor	65	Magnitud o tamaño (después del poweron-reset) para la variable Vel_LIM
Est_Pres_TConfig	Variable	Word	Estado actual de la máquina de la tarea config.
ValorLIM	Variable	Byte	Variable donde se almacena temporalmente la velocidad límite.
Vel_LIM	Variable	Byte	Valor actual de la velocidad límite.

Los diagramas de flujo de la tarea se presentan en las figuras 5 y 6:

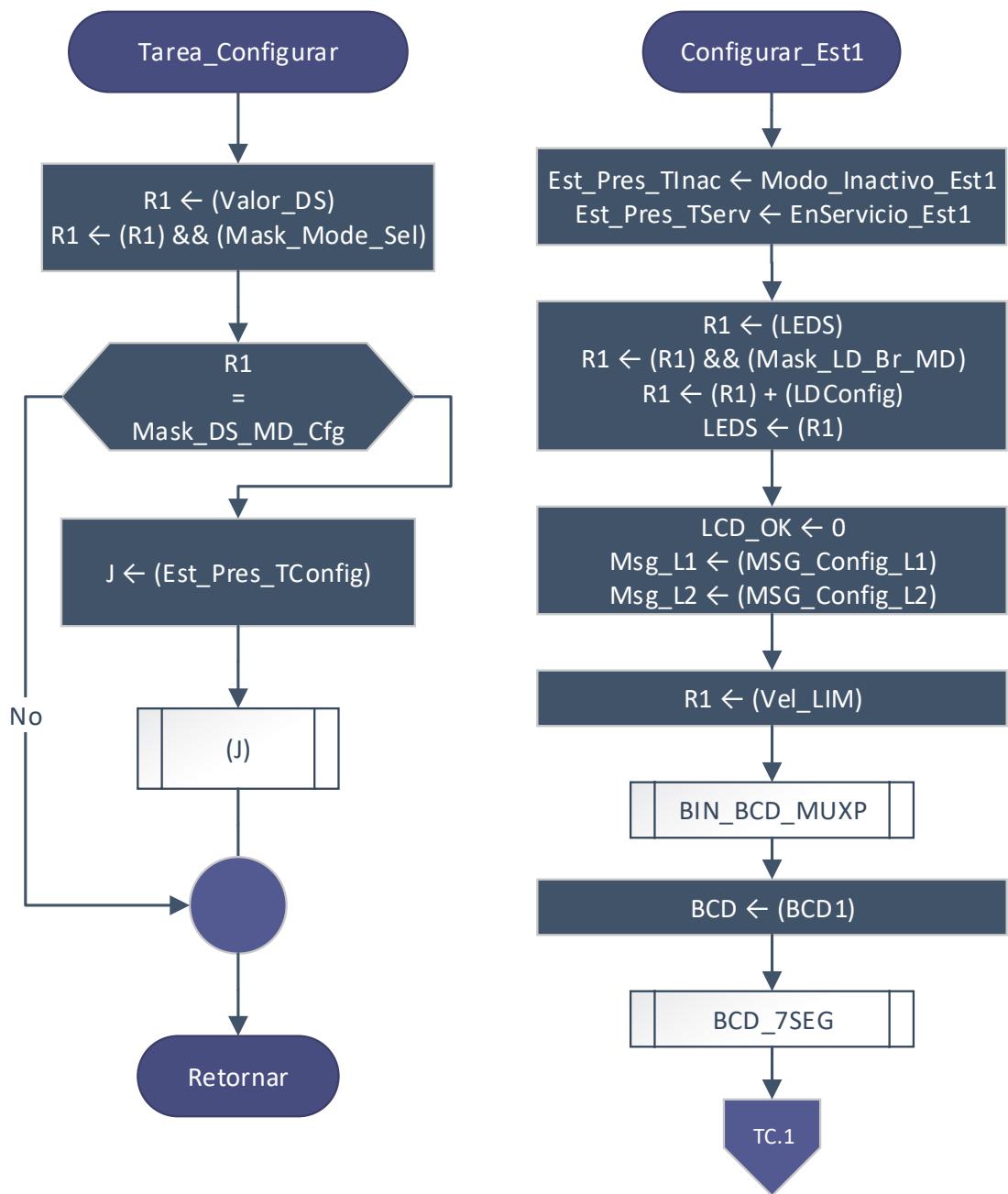


Figura 5: Diagrama de flujo de la tarea Modo Configurar, parte 1.

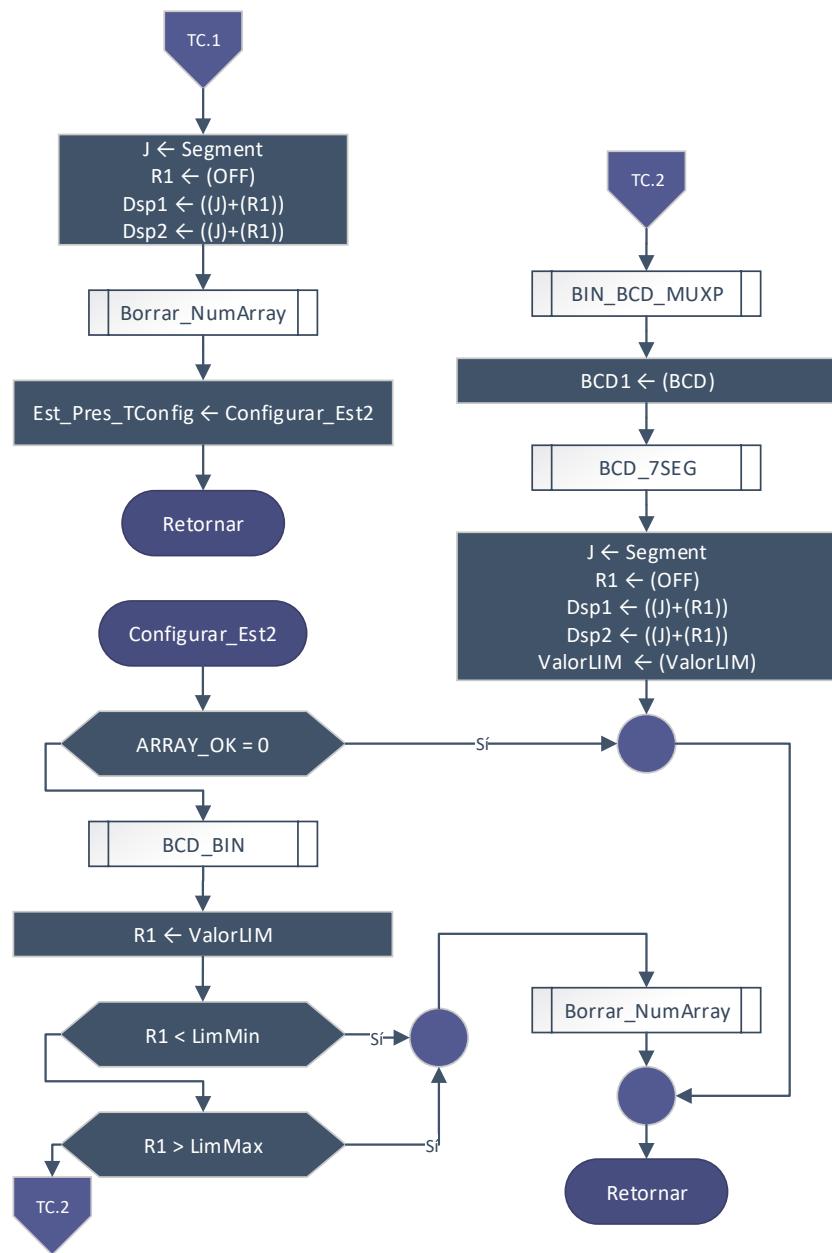


Figura 6: Diagrama de flujo de la tarea Modo Configurar, parte 2.

2.5. Tarea EnServicio

Cuando el radar está en servicio, espera a que un vehículo sea detectado por el primer sensor, en ese momento activa un timer para determinar cuánto tarda en pasar por el segundo, y una vez que esto sucede calcula la velocidad media que tuvo el automotor en los 40 metros de recorrido.

Con esto hecho, si la velocidad es inválida lo indica, y si no lo es espera a que pase el tiempo para activar la pantalla, muestra el mensaje e indica si se superó o no la el valor permitido para finalmente apagar el mensaje cuando el vehículo pierde vista con el sistema.

Los valores y estructuras de datos utilizadas se muestran en la tabla 4.

Cuadro 4: Valores y estructuras de datos utilizados en Modo En Servicio.

Nombre	Tipo	Magnitud o tamaño	Descripción
LDEnServ	Valor	\$04	Led PB2.
tTimerVel	Valor	100	Tiempo para esperar a que el vehículo pase por los sensores.
tTimerError	Valor	30	Timer de 3 segundos en base 100mS
VelocMax	Valor	99	Límite de velocidad máxima.
VelocMin	Valor	45	Límite mínimo de la velocidad.
Mask_DS_MD_Srv	Valor	\$80	Disposición de los dipswitches para el modo en servicio.
Est_Pres_TServ	Variable	Word	Estado actual de la máquina en servicio.
Vel_Calc	Variable	Byte	Valor calculado de velocidad
DeltaT	Variable	Byte	Diferencia de tiempo, en ticks, entre la activación de los sensores.
TimerPant	Variable timer	Byte	Timer para esperar a encender la pantalla que ve el vehículo.
TimerFinPant	Variable timer	Byte	Timer para esperar a apagar la pantalla que ve el vehículo.

Los diagramas de flujo se presentan en las figuras 7, 8, 9 y 10:

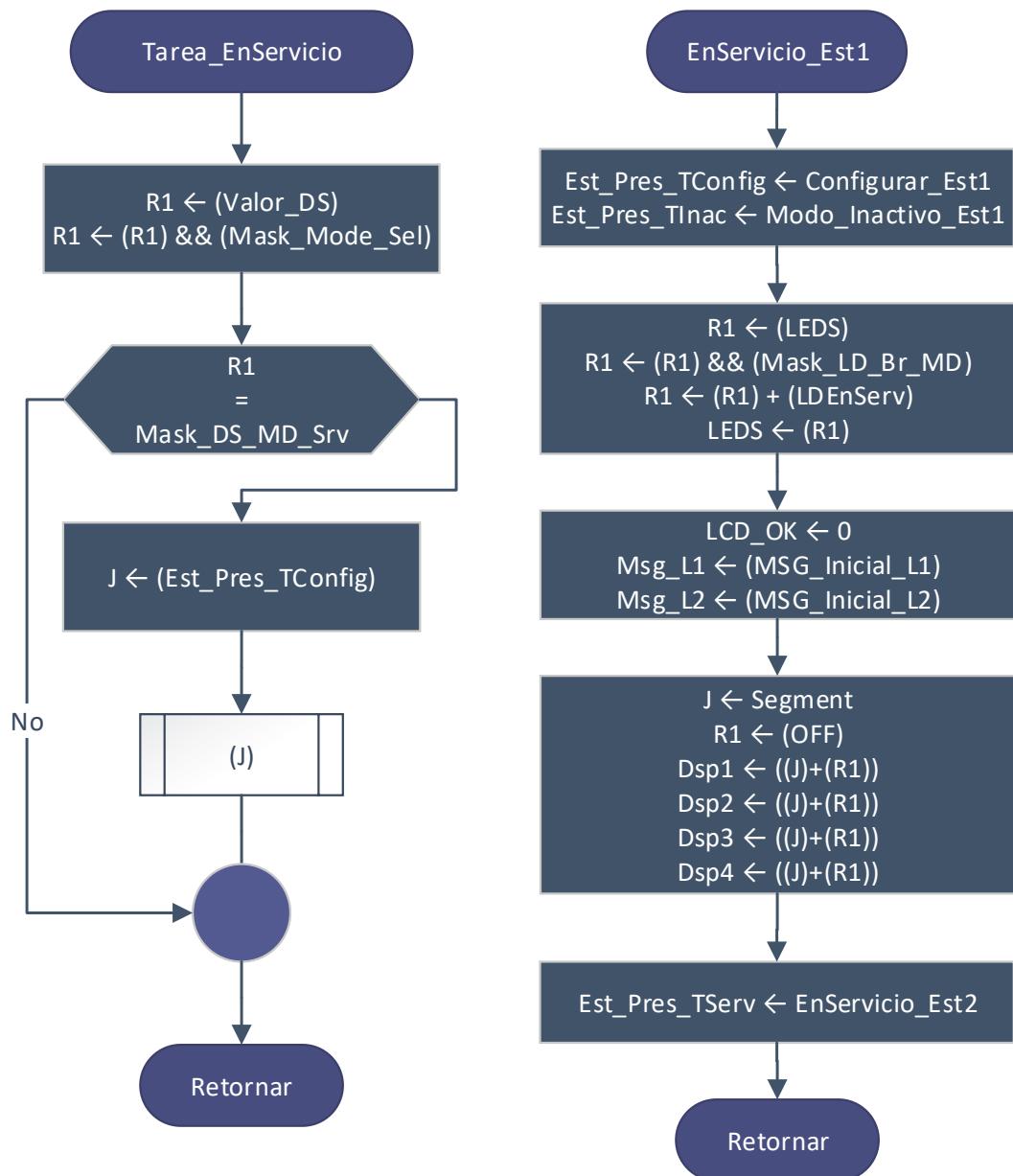


Figura 7: Diagrama de flujo de la tarea Modo En Servicio, parte 1.

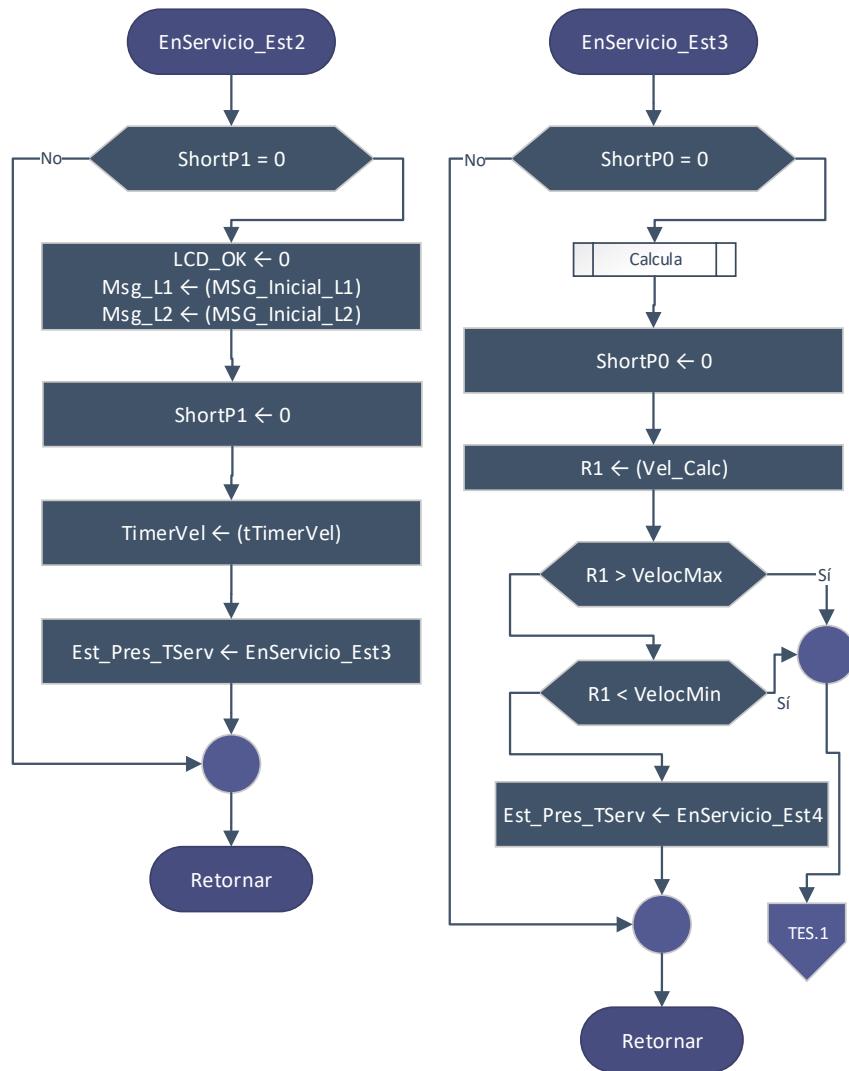


Figura 8: Diagrama de flujo de la tarea Modo En Servicio, parte 2.

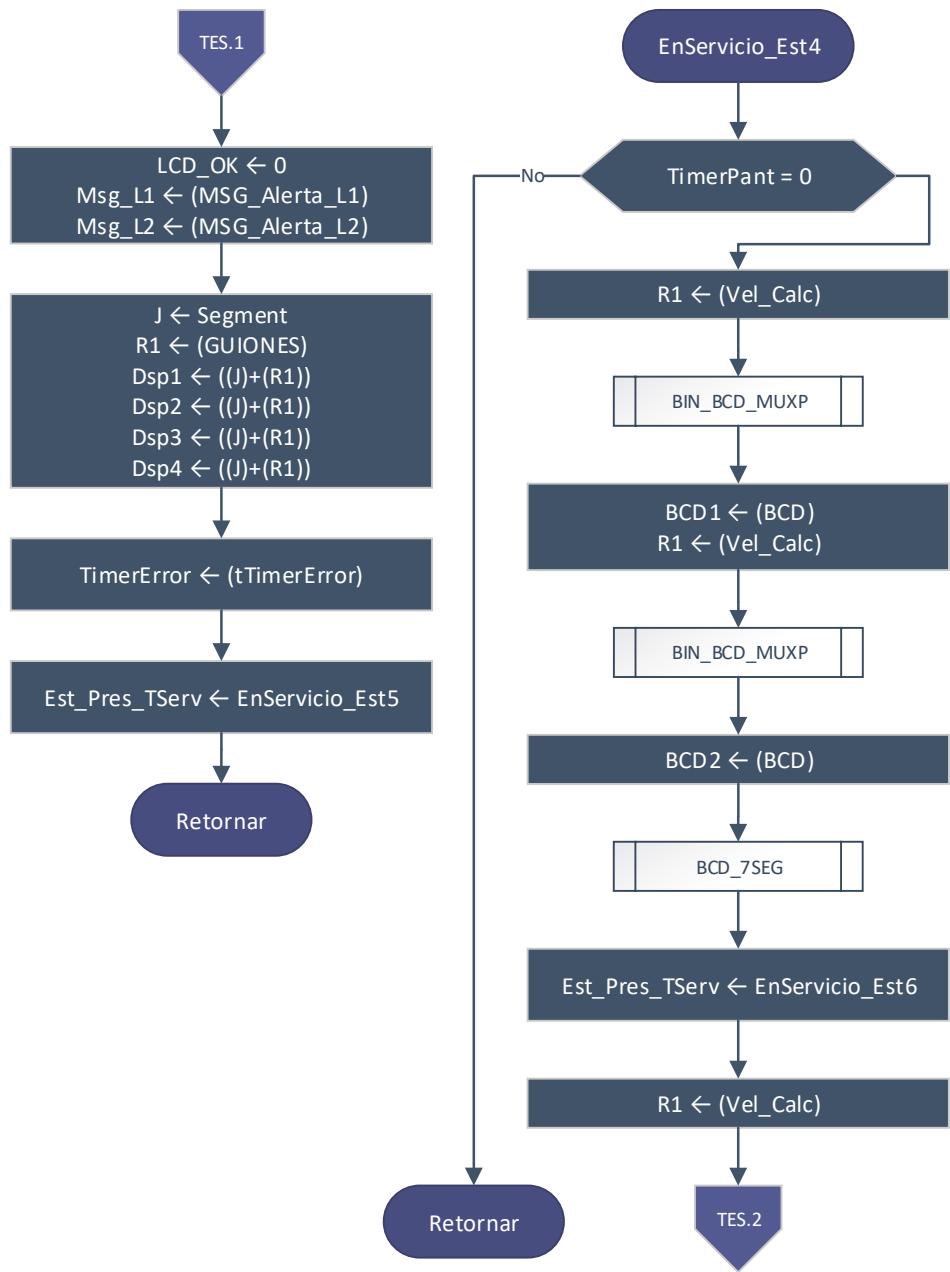


Figura 9: Diagrama de flujo de la tarea Modo En Servicio, parte 3.

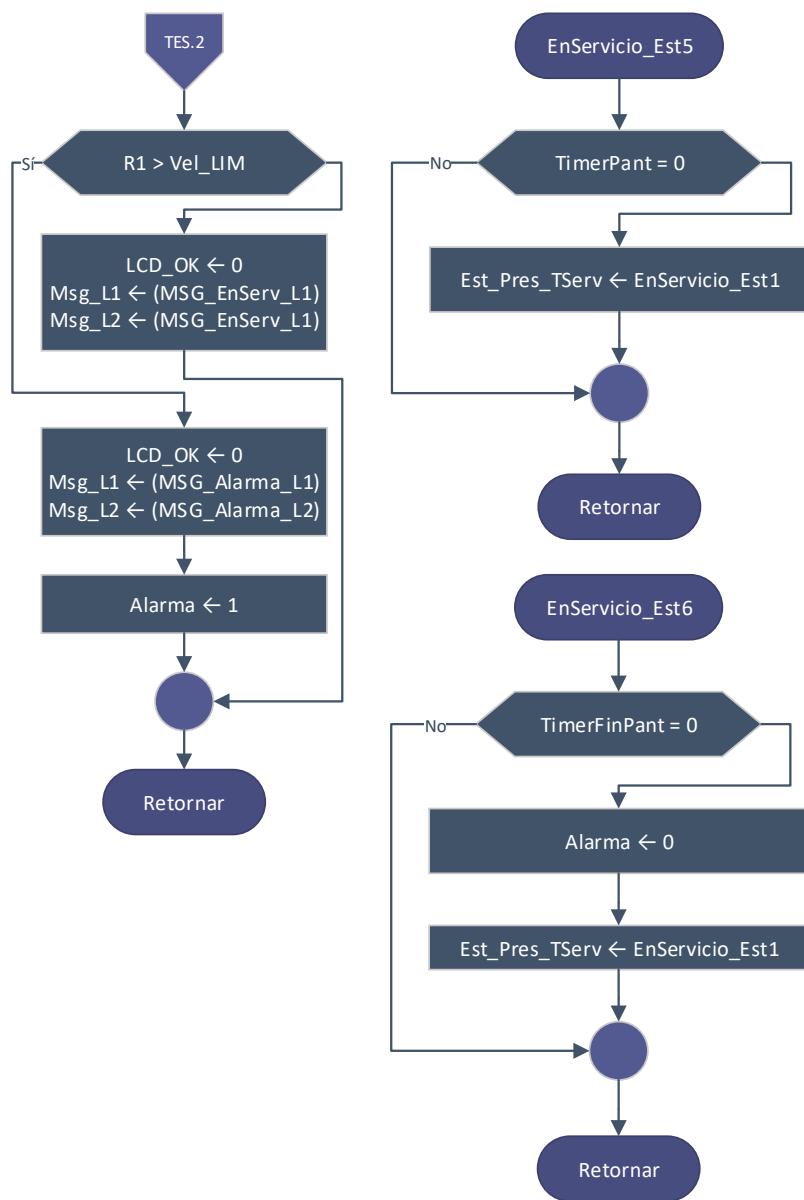


Figura 10: Diagrama de flujo de la tarea Modo En Servicio, parte 4.

2.6. Tarea DsplzLeds

La tarea se encarga de desplazar los leds de alarma cuando el vehículo excede la velocidad permitida a una razón de 100mS por led. Se mantiene esperando a que se active la señal Alarma del registro Banderas_2.

Se muestran en la tabla 5 los valores y estructuras de datos utilizados.

Cuadro 5: Valores y estructuras de datos utilizados en la tarea de desplazamiento de leds de alarma.

Nombre	Tipo	Magnitud o tamaño	Descripción
tTimerDplzLeds	Valor	Byte	Variable actual de la máquina de DsplzLeds.
LED_Sup	Valor	%10000000	Valor del led superior.
LED_Inf	Valor	%00001000	Valor del led inferior.
Mask_Dspz_LEDS	Valor	\$07	Máscara para borrar los leds a desplazar y dejar los de operación.
Est_Pres_DsplzLeds	Variable	Word	Estado actual de la máquina que desplaza los cinco led más significativos del puerto B.
DplzLeds	Variable	Byte	Variable con el estado actual de los leds de alarma.
Timer_DplzLeds	Variable timer	Byte	Timer para esperar a desplazar un led, base 100mS.
LEDS	Variable	Byte	Variable que almacena el estado actual de leds en PortB

Seguidamente, en las figuras 11 y 12 se encuentran los diagramas de flujo de la tarea.

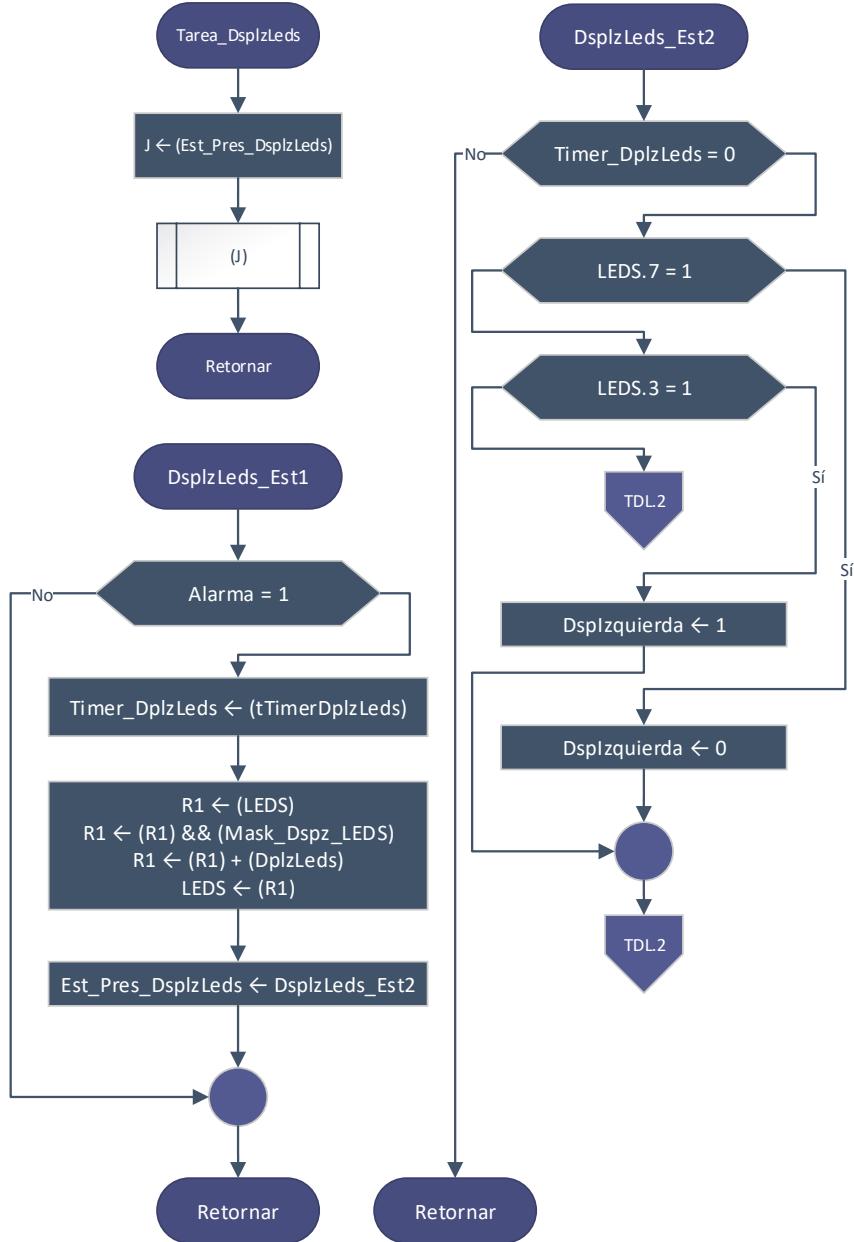


Figura 11: Diagrama de flujo de la tarea de desplazamiento de leds de alarma, parte 1.

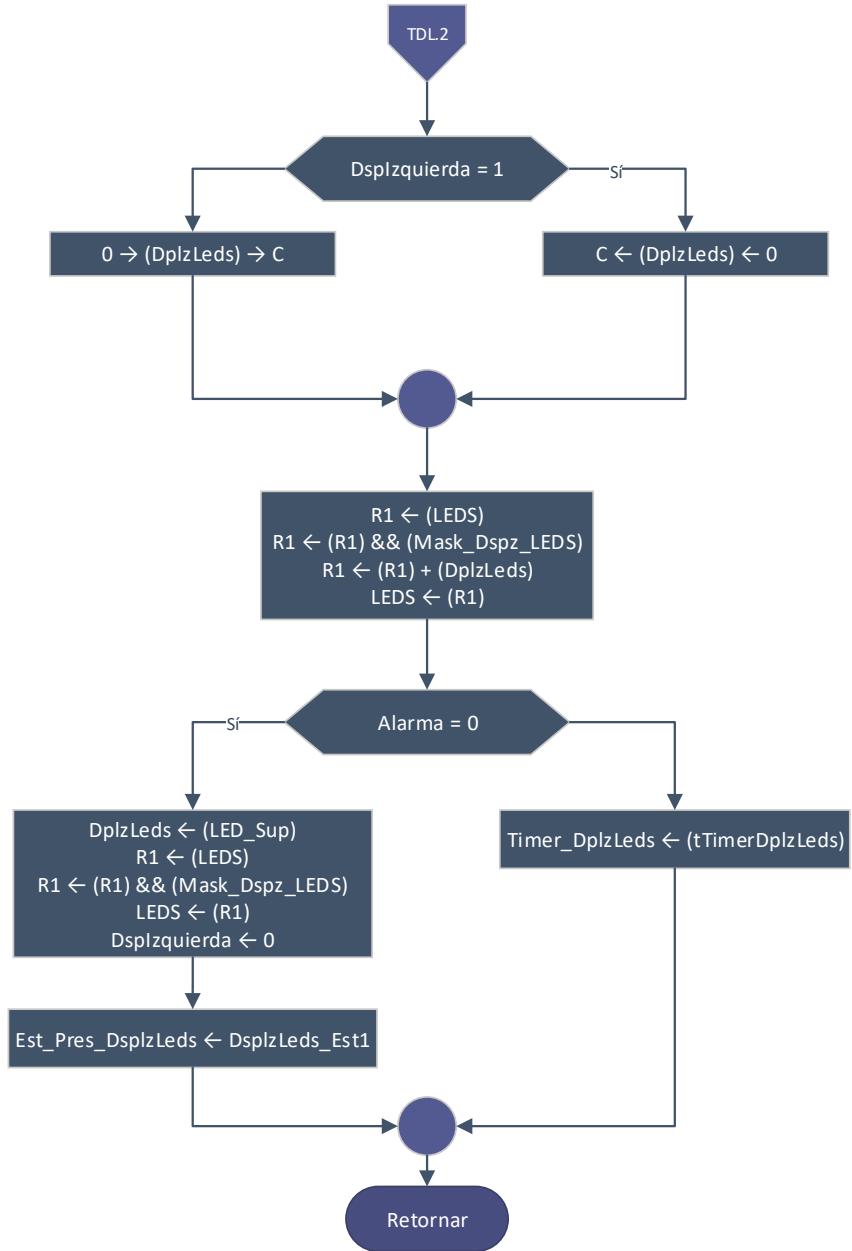


Figura 12: Diagrama de flujo de la tarea de desplazamiento de leds de alarma, parte 2.

2.7. Tarea Brillo

Esta tarea se encarga de leer el estado actual del potenciómetro del canal 7 del conversor analógico digital 0 y, mediante una regresión lineal obtener un valor de 0 a 100 que representa el tiempo de encendido de los leds y *displays* de 7 segmentos.

La regresión utilizada fue la siguiente:

$$Valor_{brillo} = Valor_{leido} \cdot \frac{100}{255} \quad (2)$$

En la tabla 6 se muestran los valores y estructuras de datos utilizados:

Cuadro 6: Valores y estructuras de datos utilizados en la tarea brillo.

Nombre	Tipo	Magnitud o tamaño	Descripción
tTimerBrillo	Valor	4	Valor de cuenta de 400 mS mientras se realza el ciclo de conversión. (4x100mS)
MaskSCF	Valor	\$80	Valor para determinar cuando se termina el ciclo de conversión. Corresponde con el registro ATD0STAT0
ATD_Read_CMD	Valor	\$87	Valor a escribir en el registro de control 5 del conversor analógico digital para iniciar la lectura. Análogo a un comando.
Cte_Div_V1_ATD	Valor	255	Constante de división para representar la pendiente de la recta de la regresión lineal para convertir el valor leído del ATD al rango de brillo: de [0,255] a [0,100].
Cte_Mul_V1_ATD	Valor	100	Constante de multiplicación para la regresión lineal.
Est_Pres_TBrillo	Variable	Word	Estado actual de la máquina que actualiza el valor de brillo.
TimerBrillo	Variable	Byte	Timer para esperar entre las conversiones del ATD.

Los diagramas de flujo se encuentran en las figuras 13 y 14:

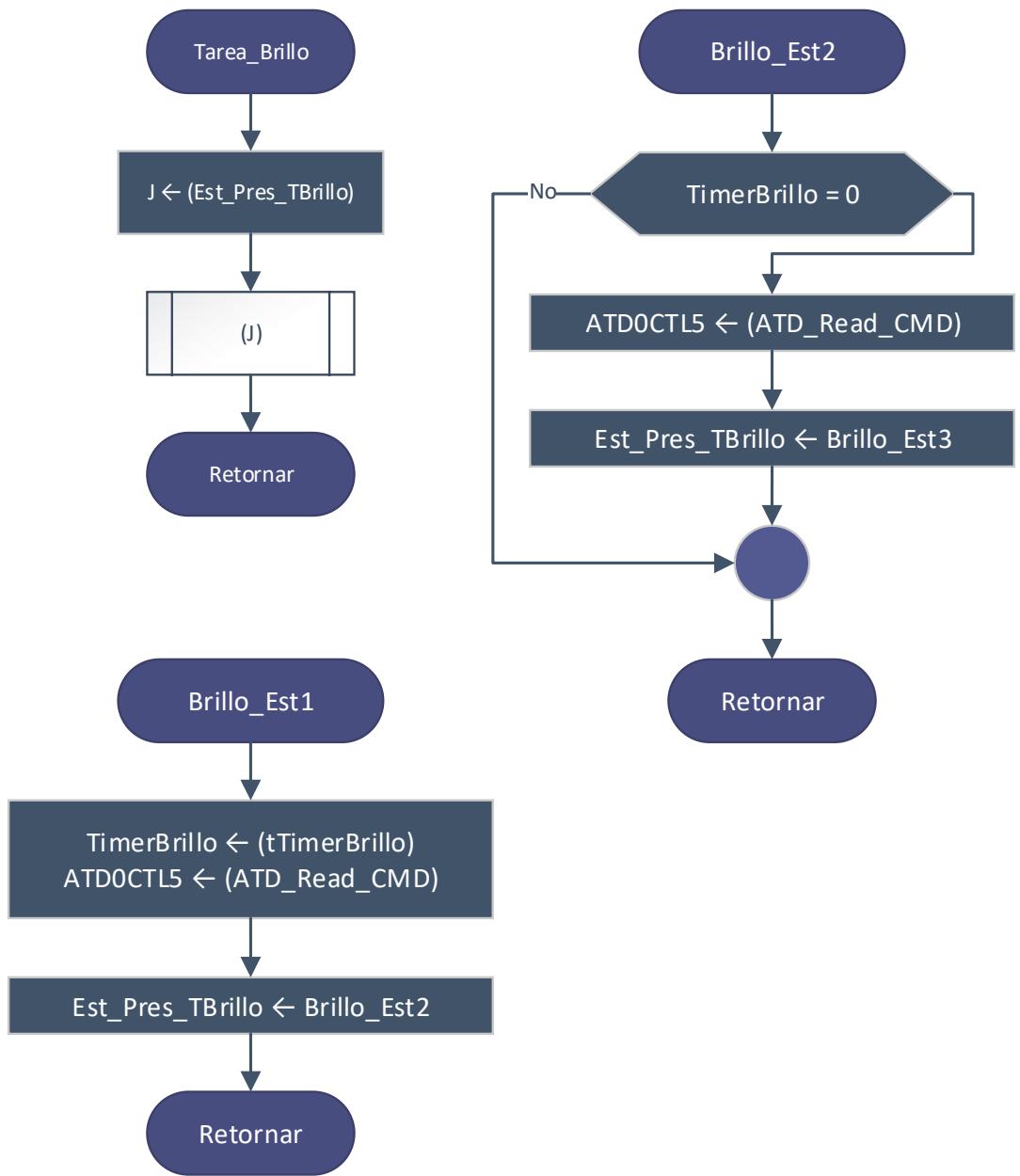


Figura 13: Diagrama de flujo de la tarea brillo, parte 1.

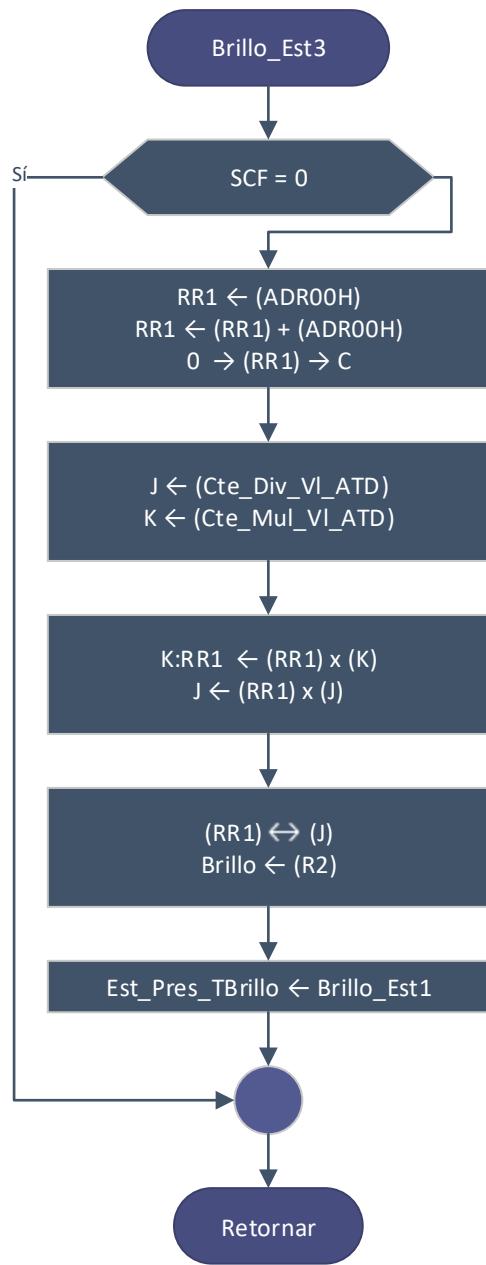


Figura 14: Diagrama de flujo de la tarea brillo, parte 2.

2.8. Tarea Teclado

La tarea realiza lecturas del teclado matricial con rechazo de rebotes y sin *roll-over*.

Las estructuras de datos se encuentran en la tabla 7.

Cuadro 7: Valores y estructuras de datos utilizados en la tarea teclado.

Nombre	Tipo	Magnitud o tamaño	Descripción
PortTCL	Valor	POR TA	Registro de datos del puerto A
Teclas_MAX	Valor	2	Cantidad máxima de teclas a leer del teclado
Teclas	Valor	\$1110	Etiqueta de ubicación de la tabla de Teclas
Tecla_BORRAR	Valor	\$0B	Para detectar la tecla borrar.
Tecla_ENTER	Valor	\$0E	Para detectar la tecla enter.
Ninguna_Tecla	Valor	\$FF	Caso que no se presione ninguna tecla.
Num_Array	Valor	\$1010	Arreglo de destino de los valores leídos.
MAX_TCL	Variáble	Byte	Cantidad máxima de teclas a leer
Tecla	Variáble	Byte	Valor de tecla leído
Tecla_IN	Variáble	Byte	Valor temporal de tecla, usado para detectar los rebotes y evitar teclas presionadas.
Cont_TCL	Variáble	Byte	Offset para recorrer el arreglo de números leídos y determinar cuántos se tienen.
PATRON	Variáble	Byte	Patrón a escribir en el puerto A del teclado
Est_Pres_TCL	Variáble	Word	Variáble de estado de la máquina de estados TCL

Los diagramas de flujo se presentan en las figuras 15, 16, 17 y 18:

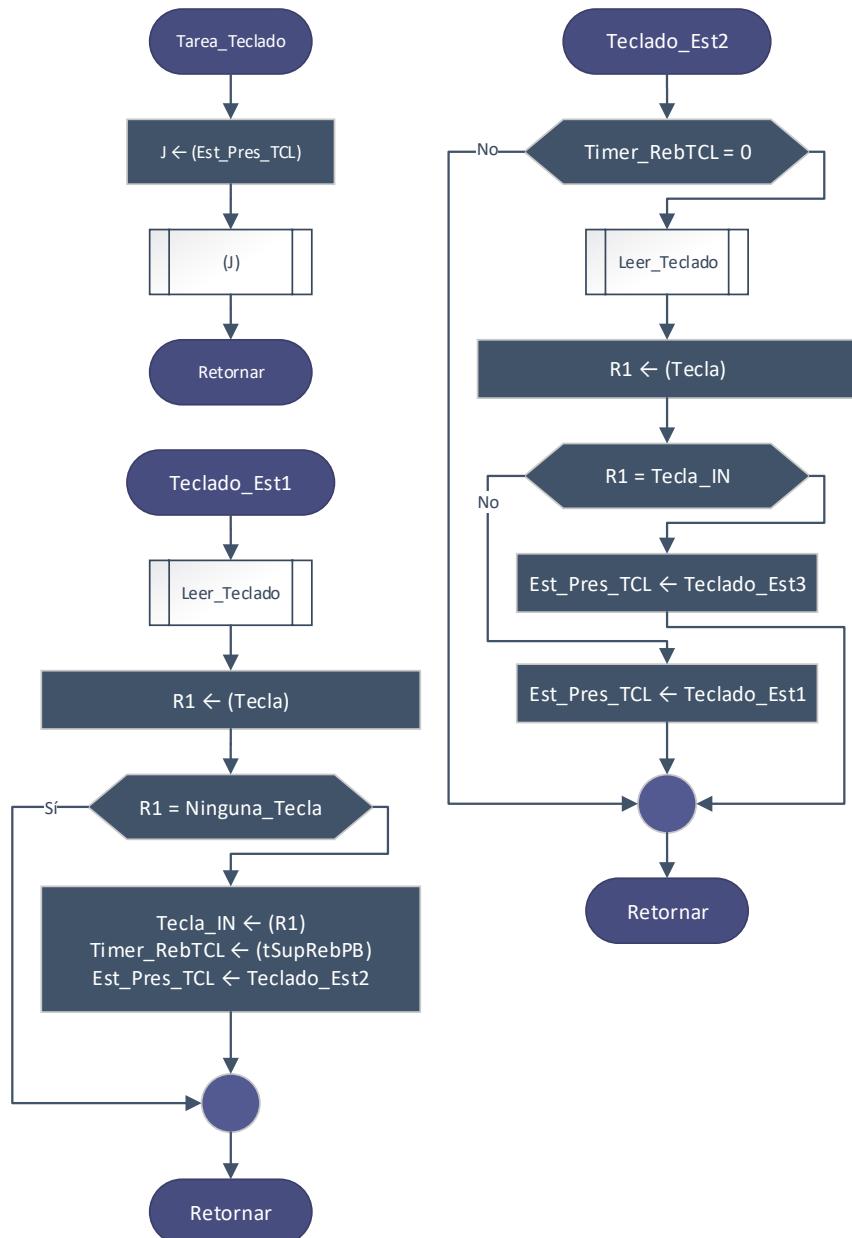


Figura 15: Diagrama de flujo de la tarea teclado, parte 1.

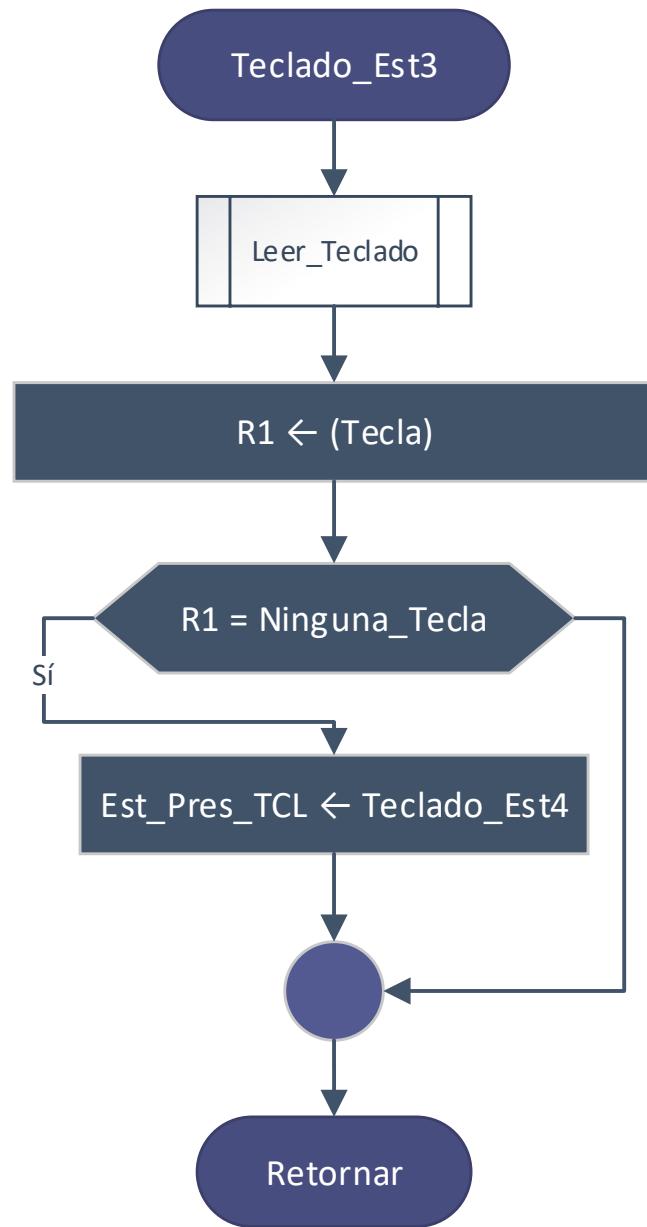


Figura 16: Diagrama de flujo de la tarea teclado, parte 2.

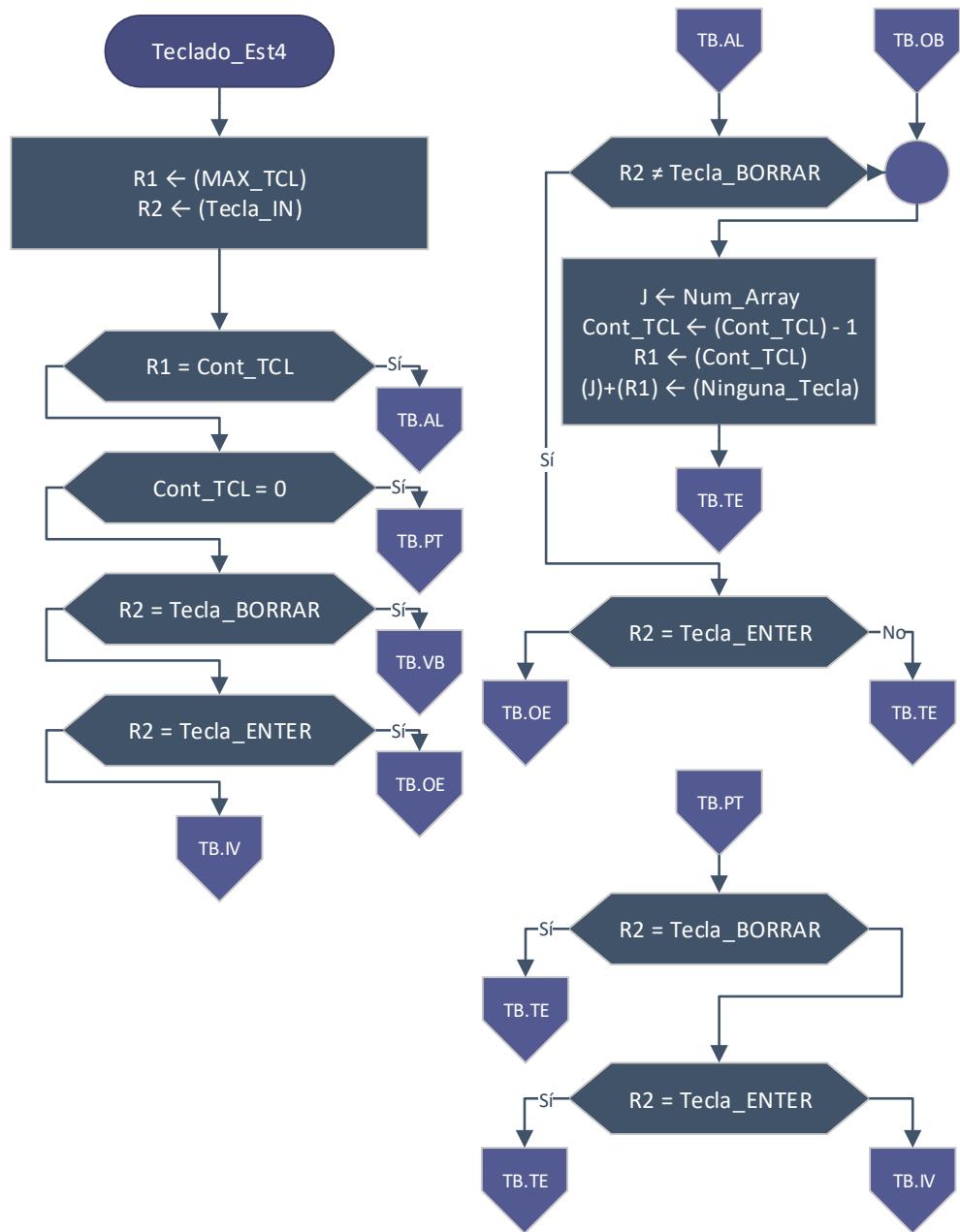


Figura 17: Diagrama de flujo de la tarea teclado, parte 3.

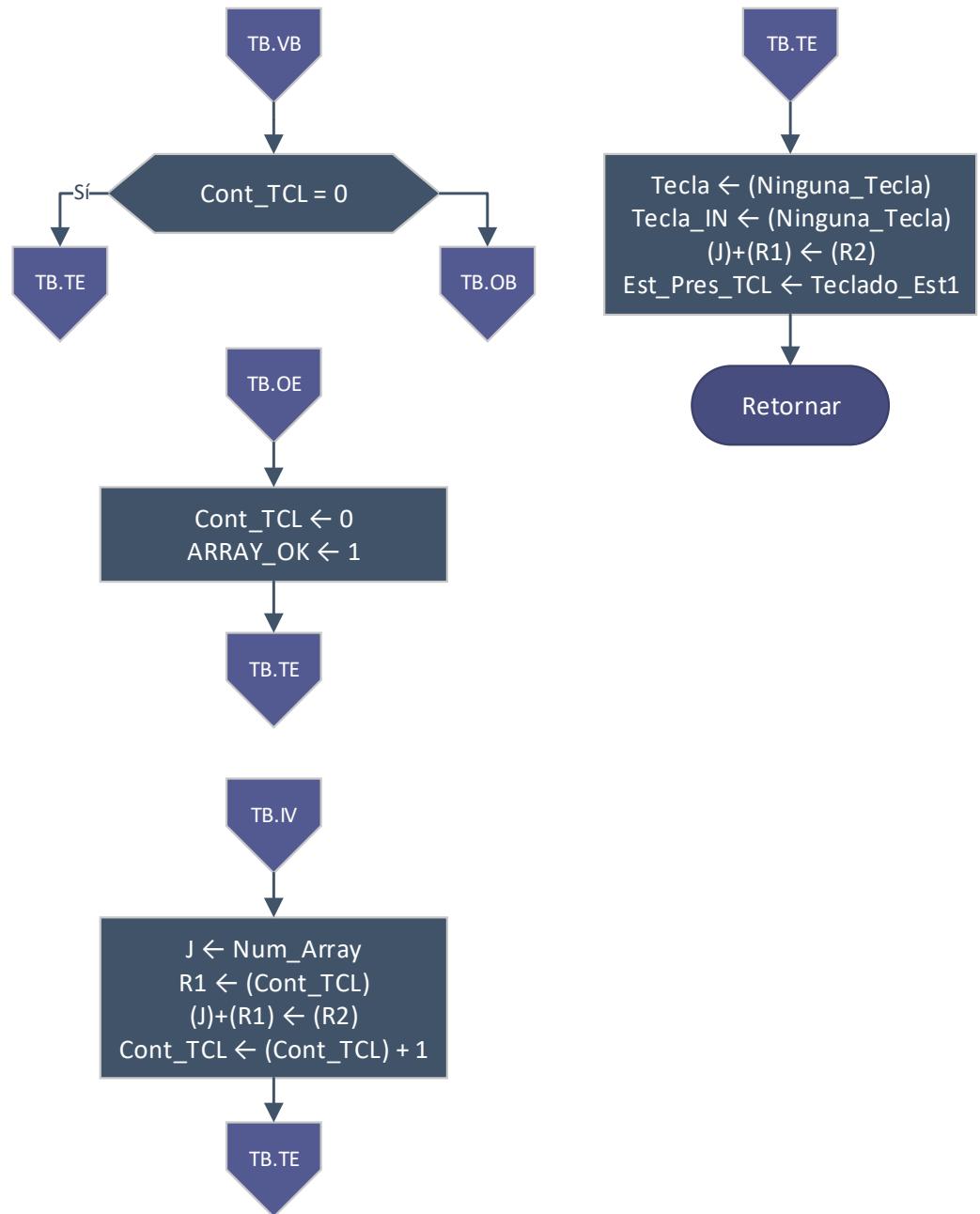


Figura 18: Diagrama de flujo de la tarea teclado, parte 4.

2.9. Tarea Led Testigo

Esta tarea rota los colores del led RGB a una razón de $500mS$ por color todo el tiempo mientras el sistema se encuentre funcionando.

En la tabla 8 se muestra el la estructura de datos utilizada por la tarea.

Cuadro 8: Estructura de datos utilizados en la tarea Led Testigo.

Nombre	Tipo	Magnitud o tamaño	Descripción
LED_Testigo	Variable	Byte	Valor actual del led testigo.
Timer_LED_Testigo	Variable	Byte	Timer de cambio de color para el led testigo.
tTimerLDTst	Valor	5	Tiempo de parpadeo de LED testigo centenas de milisegundos

En la figura 19 se muestra el diagrama de flujo de la tarea:

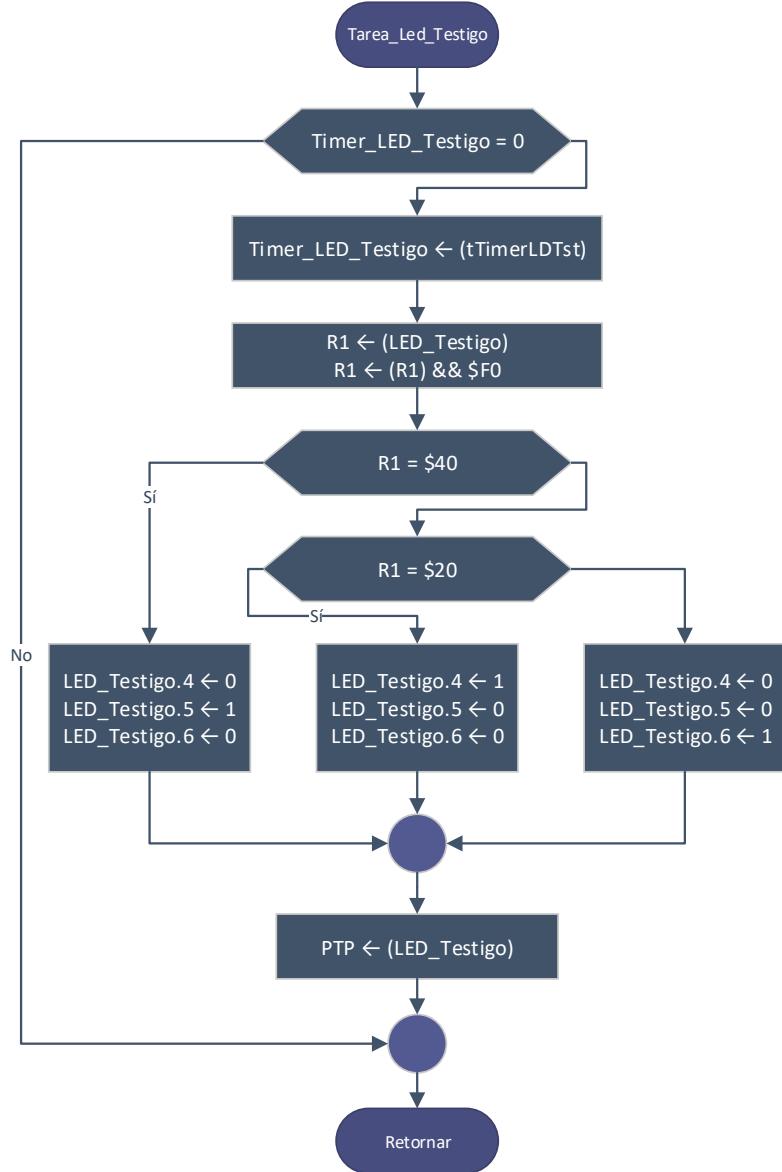


Figura 19: Diagrama de flujo de la tarea led testigo.

2.10. Tarea Leer PB0

Se encarga de detectar pulsos cortos y largos con rechazo de rebotes en el botón PH.0.

Los valores y estructuras de datos usados se muestran en [9](#):

Cuadro 9: Valores y estructuras de datos utilizados en la tarea Leer PB0.

Nombre	Tipo	Magnitud o tamaño	Descripción
PortPB	Valor	PTIH	Puerto donde se ubica el PB
tSupRebPB	Valor	10	Contador para supresión de rebotes, 10 mS
tShortP	Valor	25	Contador para detección de short press, 250 mS
tLongP	Valor	Word	Contador para detección de long press, 2 S
MaskPB0	Valor	\$01	Ubicación del SW5 en el puerto H.
Est_Pres_LeerPB0	Variable	Word	Estado de la máquina de estados para leer PB0.
Timer_RebPB0	Variable	Byte	Timer de rechazo de rebotes de PB0
Timer_SHP0	Variable	Byte	Timer de detección de short press para PB0
Timer_LP0	Variable	Byte	Timer de detección de long press para PB0
ShortP0	Valor	\$01	Valor para identificar un pulso corto en PH.0.
LongP0	Valor	\$02	Valor para identificar un pulso largo en PH.0.

En las figuras [20](#), [21](#) y [22](#) se muestran los diagramas de flujo de la tarea.

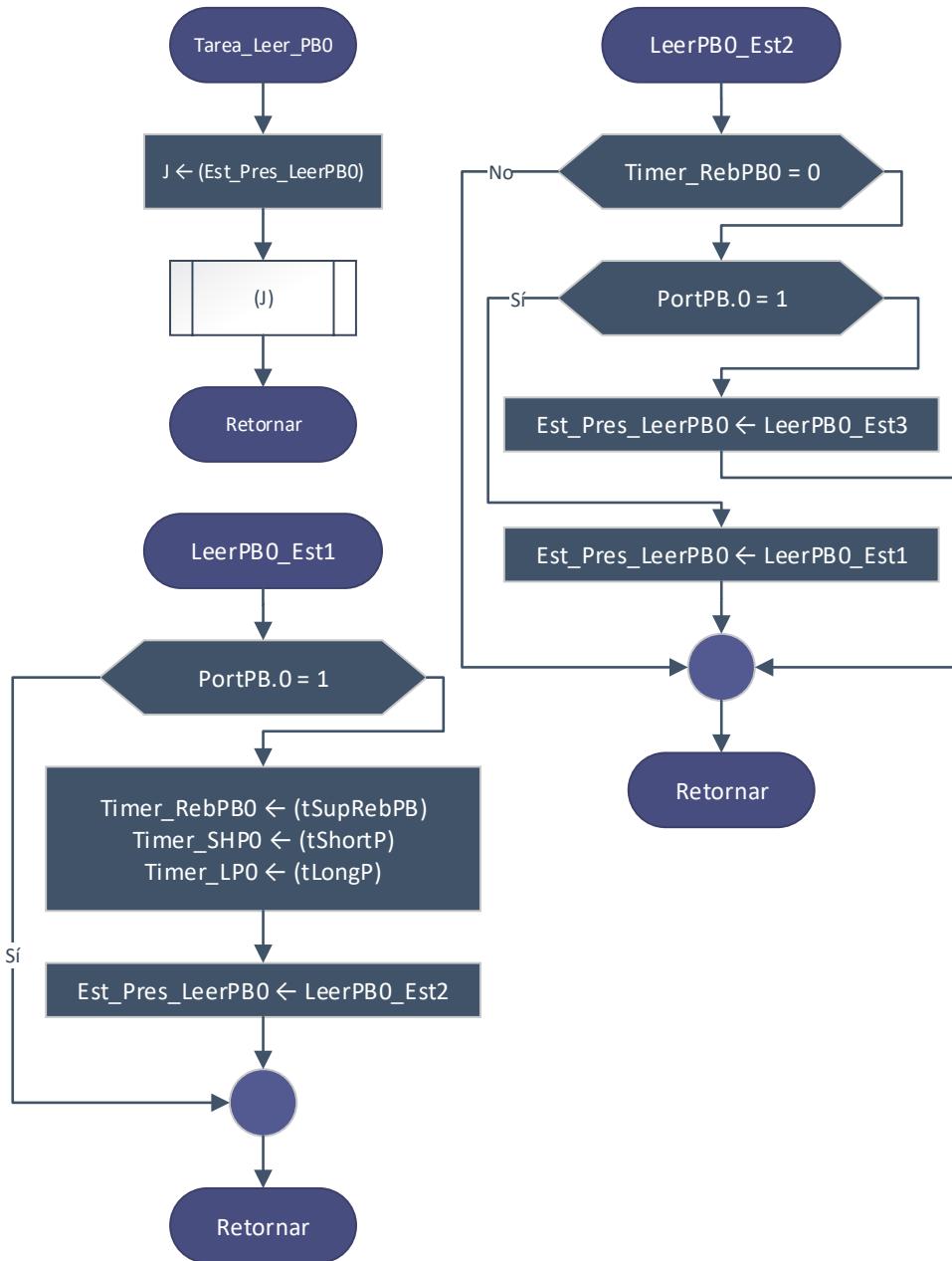


Figura 20: Diagrama de flujo de la tarea leer PB0, parte 1.

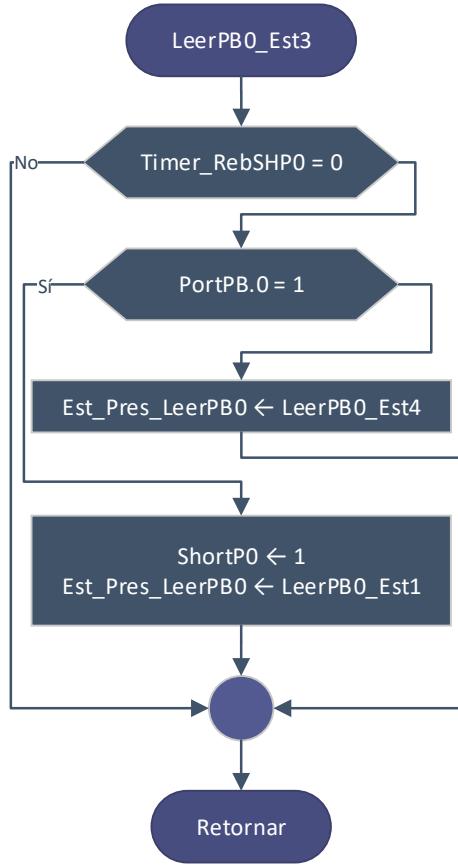


Figura 21: Diagrama de flujo de la tarea leer PB0, parte 2.

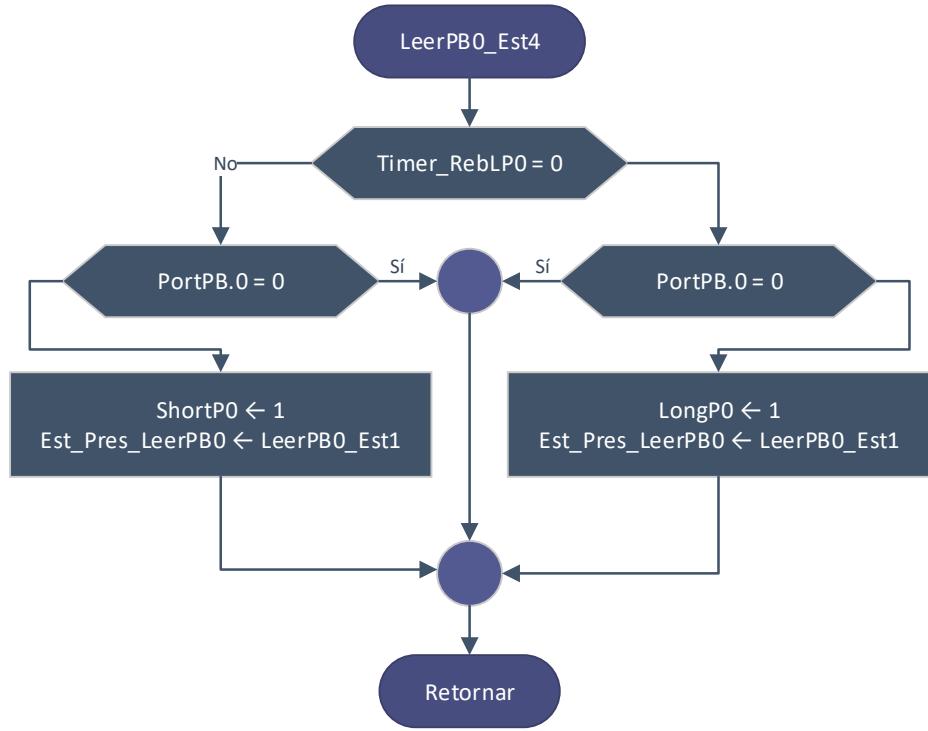


Figura 22: Diagrama de flujo de la tarea leer PB0, parte 3.

2.11. Tarea Leer PB1

Se encarga de detectar pulsos cortos y largos con rechazo de rebotes en el botón PH.3.

Los valores y estructuras de datos usados se muestran en [10](#):

Cuadro 10: Valores y estructuras de datos utilizados en la tarea Leer PB1.

Nombre	Tipo	Magnitud o tamaño	Descripción
PortPB	Valor	PTIH	Puerto donde se ubica el PB
tSupRebPB	Valor	10	Contador para supresión de rebotes, 10 mS
tShortP	Valor	25	Contador para detección de short press, 250 mS
tLongP	Valor	Word	Contador para detección de long press, 2 S
MaskPB1	Valor	\$08	Ubicación del SW2 en el puerto H.
Est_Pres_LeerPB1	Variable	Word	Estado de la máquina de estados para leer PB1.
Timer_RebPB1	Variable	Byte	Timer de rechazo de rebotes de PB1
Timer_SHP1	Variable	Byte	Timer de detección de short press para PB1
Timer_LP1	Variable	Byte	Timer de detección de long press para PB1
ShortP1	Valor	\$04	Valor para identificar un pulso corto en PH.1.
LongP1	Valor	\$08	Valor para identificar un pulso largo en PH.1.

En las figuras [23](#), [24](#) y [25](#) se muestran los diagramas de flujo de la tarea.

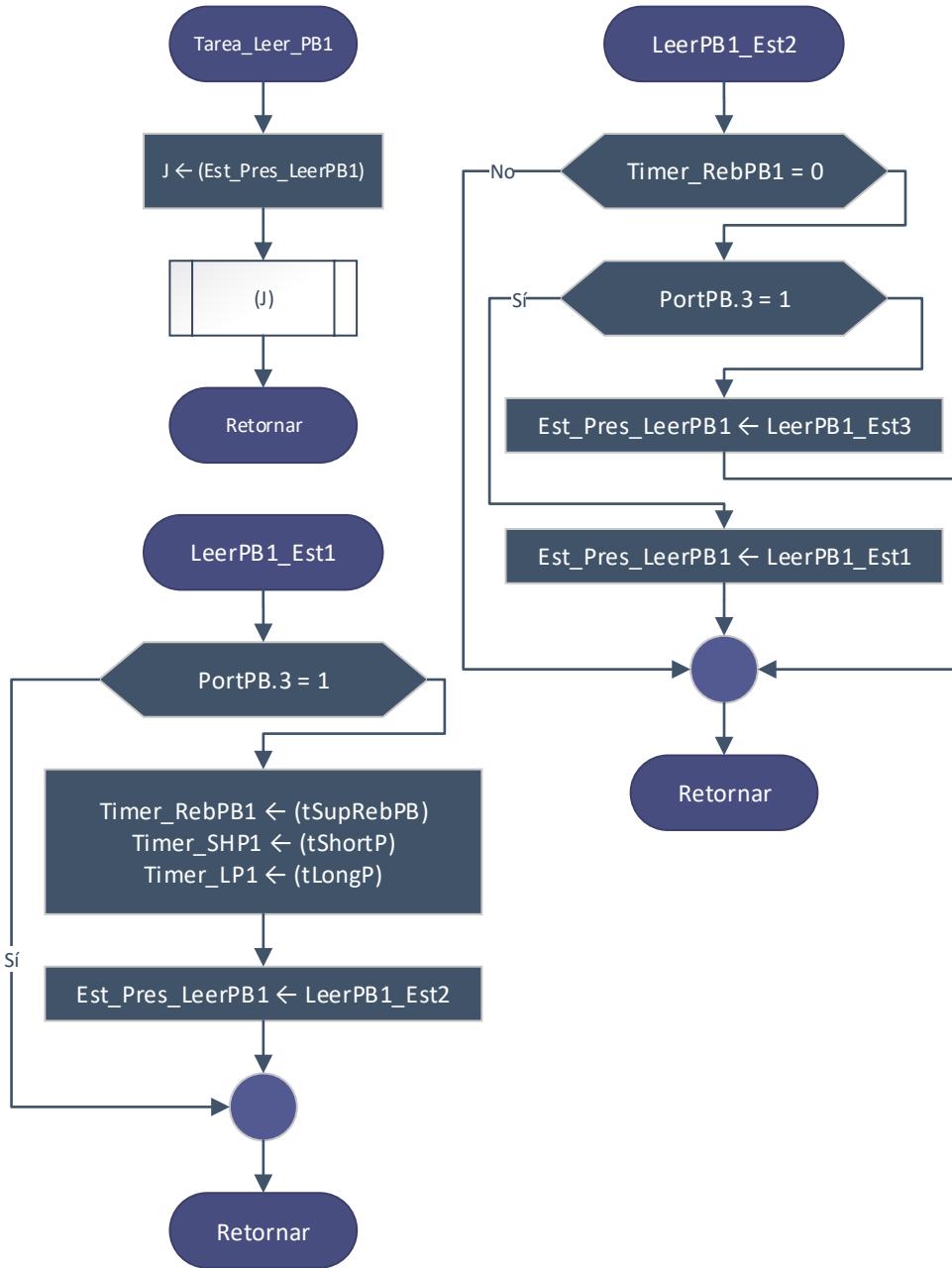


Figura 23: Diagrama de flujo de la tarea leer PB1, parte 1.

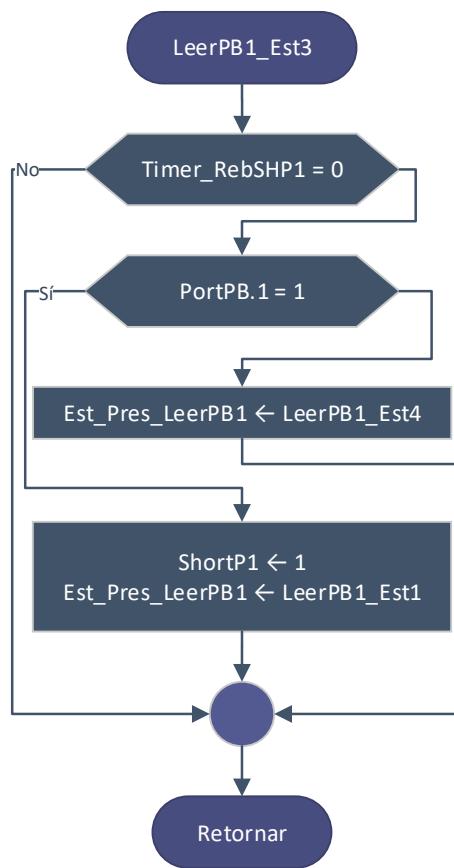


Figura 24: Diagrama de flujo de la tarea leer PB1, parte 2.

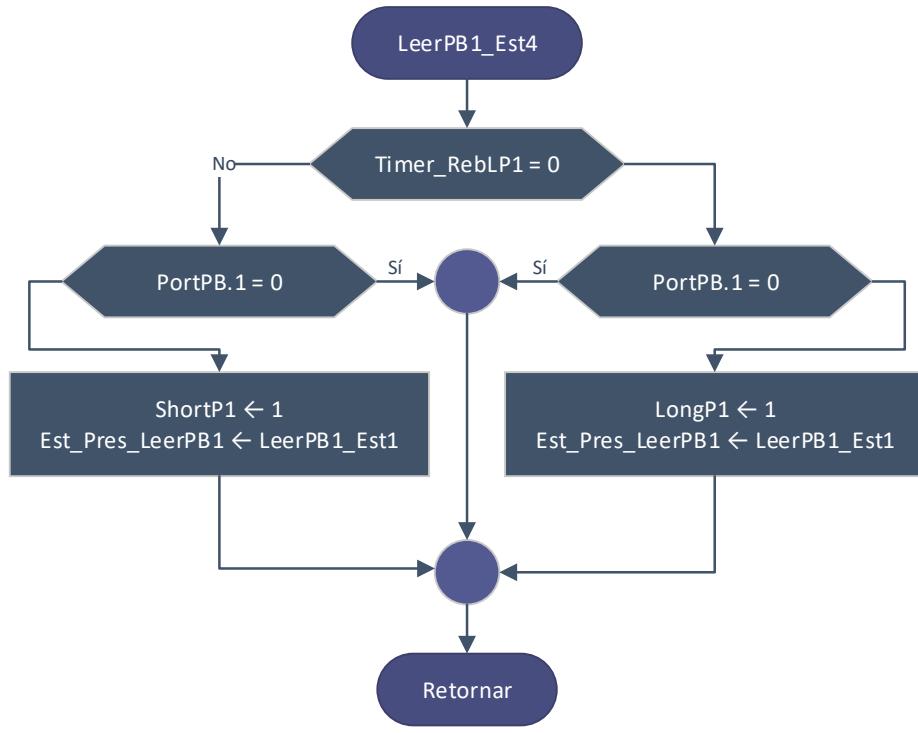


Figura 25: Diagrama de flujo de la tarea leer PB1, parte 3.

2.12. Tarea PantallaMUX

Esta tarea se encarga de multiplexar el encendido y la escritura de los datos en los led y los displays de 7 segmentos. Las estructuras de datos y valores se encuentran en [11](#):

Cuadro 11: Valores y estructuras de datos utilizados en la tarea Pantalla-MUX.

Nombre	Tipo	Magnitud o tamaño	Descripción
tTimerDigito	Valor	Word	Tiempo de encendido de cada display = 2ms
MaxCountTicks	Valor	100	Tiempo en ticks entre dos periodos de display
CounterTicks	Variable	Word	Contador de ticks de la máquina de tiempos.
Segment	Valor	\$1100	Tabla de equivalencias para códigos BCD-7seg
OFF	Valor	11	Offset de la tabla segment para apagar Dsp
GUIONES	Valor	10	Offset de la tabla segment para mostrar "
EstPres_PantallaMUX	Variable	Word	Estado actual de la máquina de pantalla multiplexada.
Dsp1	Variable	Byte	Valor del dígito 1 codificado para 8 segmentos
Dsp2	Variable	Byte	Valor del dígito 2 codificado para 8 segmentos
Dsp3	Variable	Byte	Valor del dígito 3 codificado para 8 segmentos
Dsp4	Variable	Byte	Valor del dígito 4 codificado para 8 segmentos
LEDS	Variable	Byte	Variable que almacena el estado de leds en PortB
Cont_Dig	Variable	Byte	Contador de dígitos colocados en el display
Brillo	Variable	Byte	Cantidad de ticks que pasa el display apagado

Mientras que en las figuras [26](#) y [27](#) se encuentran los diagramas de flujo:

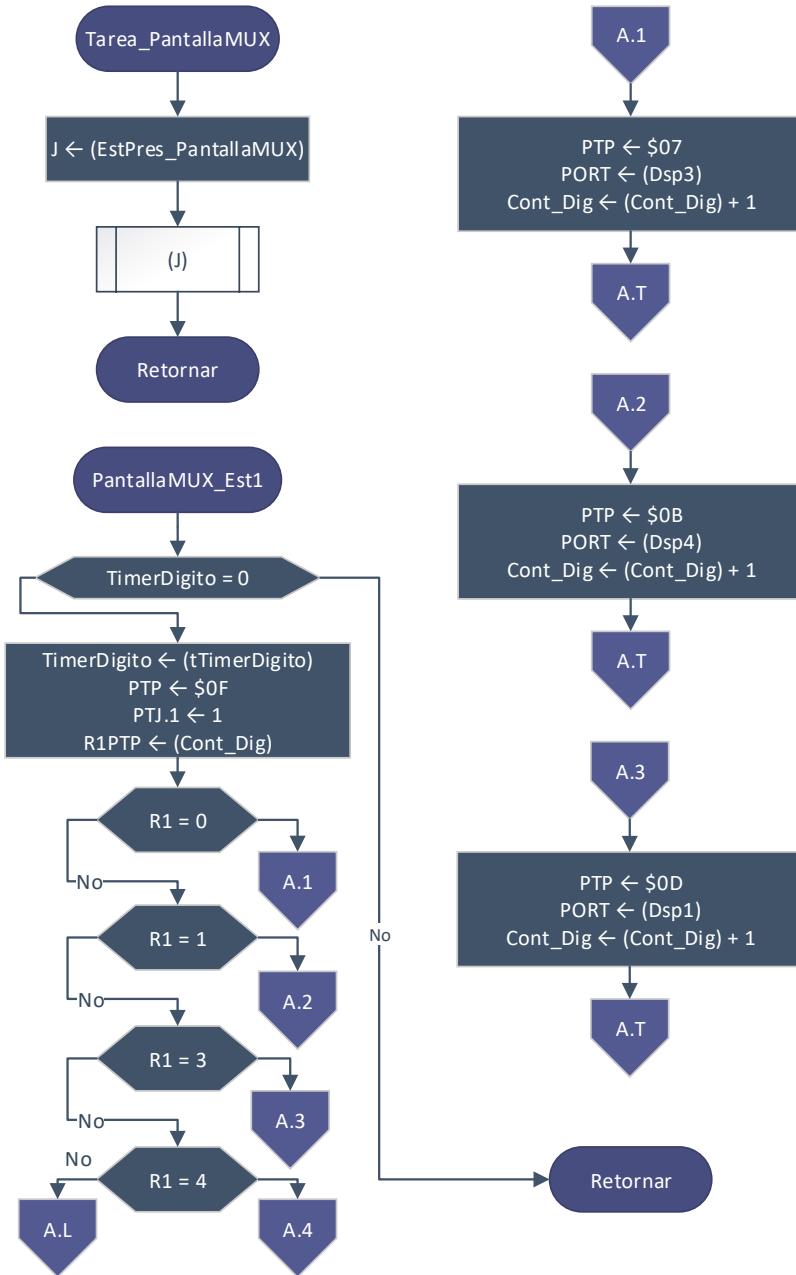


Figura 26: Diagrama de flujo de la tarea PantallaMUX, parte 1.

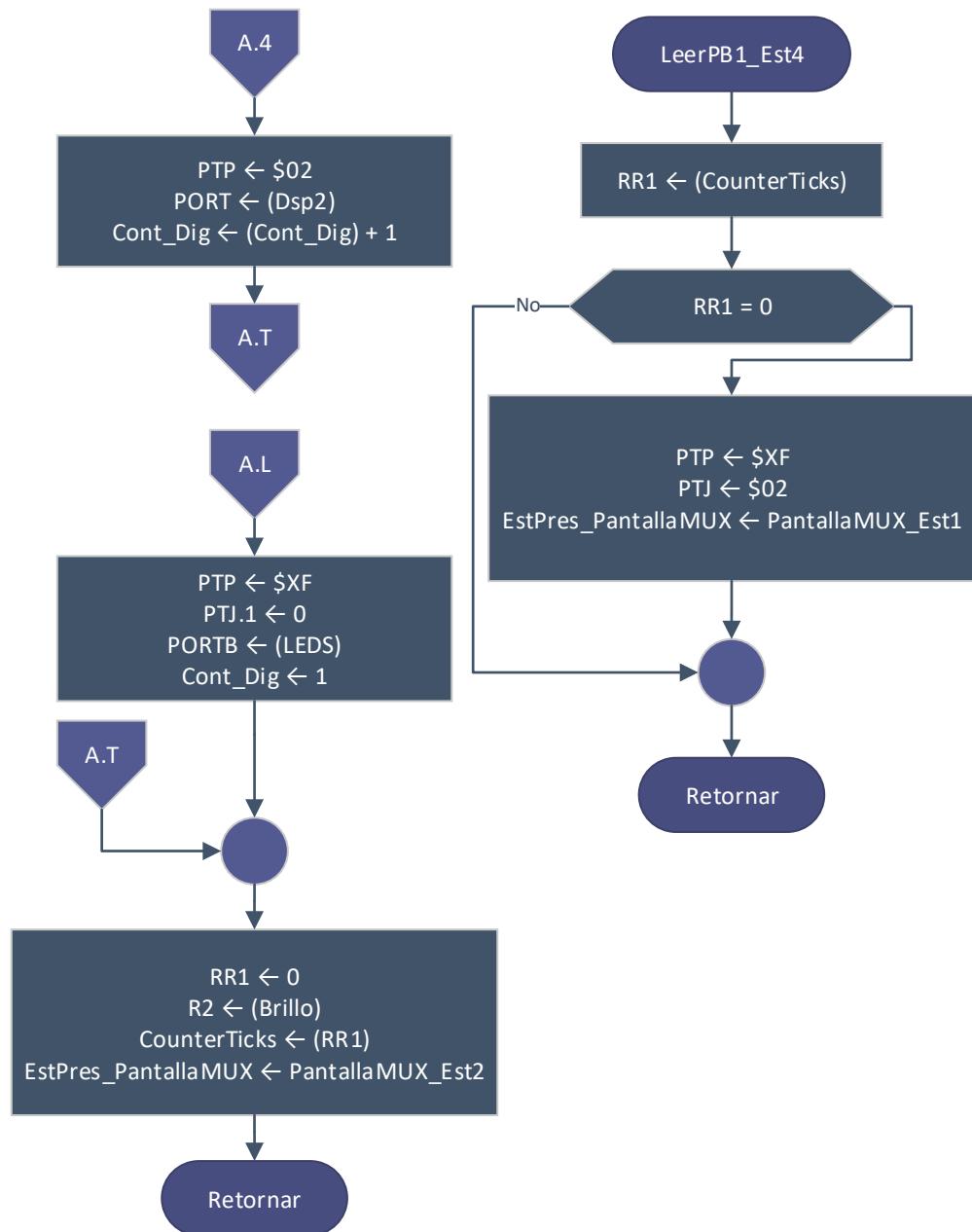


Figura 27: Diagrama de flujo de la tarea PantallaMUX, parte 2.

2.13. Tarea LCD

Esta tarea se encarga de enviar mensajes completos a la pantalla LCD. Los valores y estructuras de datos se encuentran en [12](#):

Cuadro 12: Valores y estructuras de datos utilizados en la tarea LCD.

Nombre	Tipo	Magnitud o tamaño	Descripción
EOB	Valor	\$FF	Carácter de finalización de bloque
ADD_L1	Valor	\$80	Dirección (en pantalla) de inicio de msg en línea 1
ADD_L2	Valor	\$C0	Dirección (en pantalla) de inicio de msg en línea 2
RS	Valor	\$01	Bandera para enviar un .
LCD_Ok	Valor	\$02	Determina si la LCD ya tiene los mensajes deseados en pantalla.
FinSendLCD	Valor	\$04	Se pone en alto cuando se terminó de enviar un mensaje.
Second_Line	Valor	\$08	Diferencia entre enviar la primera o segunda línea.
Msg_L1	Variable	Word	Mensaje de la línea 1 de la LCD
Msg_L2	Variable	Word	Mensaje de la línea 2 de la LCD
EstPres_TareaLCD	Variable	Word	Estado actual de la máquina TareaLCD

En la figura [28](#) se muestra el diagrama de flujo de la tarea.

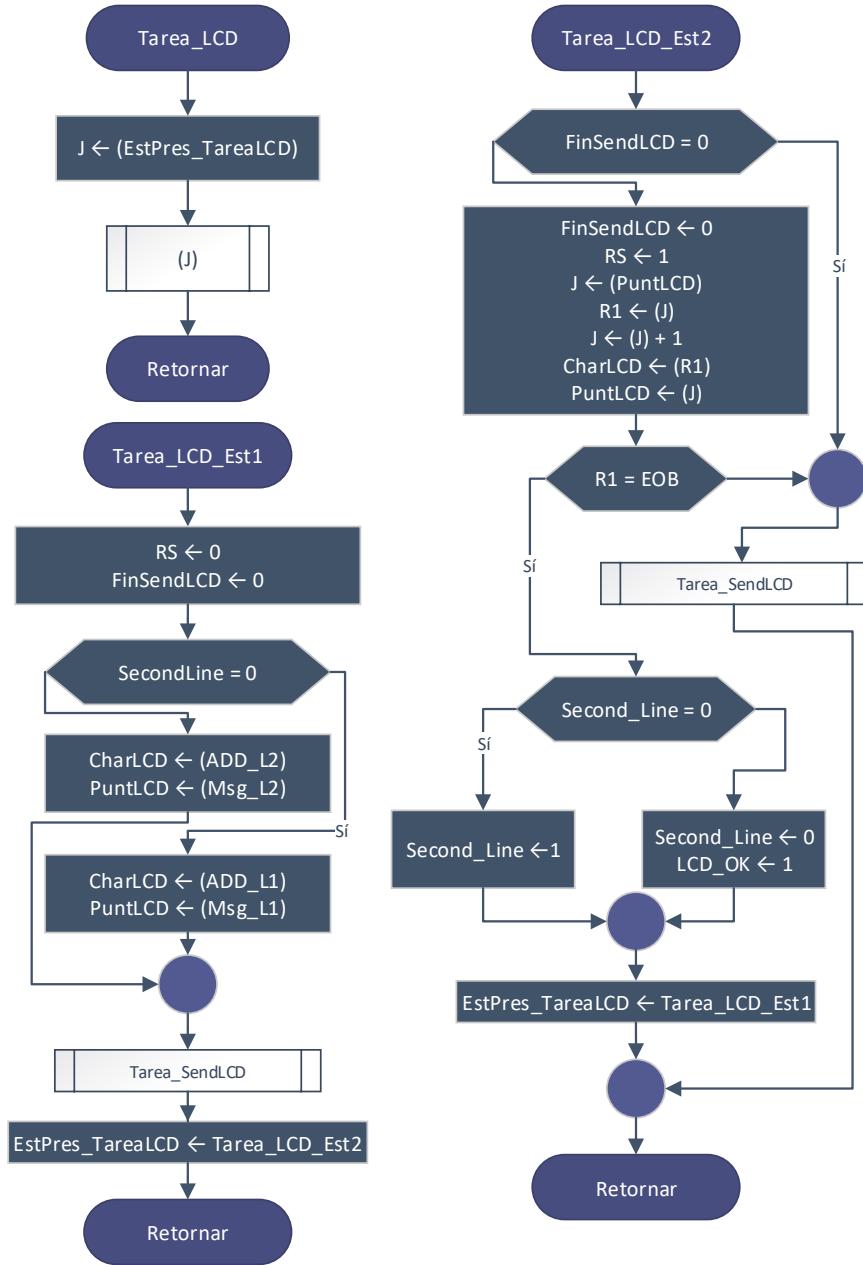


Figura 28: Diagrama de flujo de la tarea LCD.

2.14. Tarea Send LCD

Se encarga de la comunicación y el envío de la información a la pantalla LCD mediante sincronización estroboscópica.

Los valores y estructuras de datos se encuentran en [13](#):

Cuadro 13: Valores y estructuras de datos utilizados en la tarea Send_LCD.

Nombre	Tipo	Magnitud o tamaño	Descripción
tTimer2mS	Valor	Word	Valor a asignar en el timer: 2 x 1mS
tTimer260uS	Valor	13	Valor a asignar en el timer: 13 x 1 tick
tTimer40uS	Valor	Word	Valor a asignar en el timer: 2 x 1 tick
RS	Valor	\$01	Bandera para enviar un .
FinSendLCD	Valor	\$04	Se pone en alto cuando se terminó de enviar un mensaje.
CharLCD	Variable	Byte	Carácter a enviar a la pantalla
EstPres_SendLCD	Variable	Word	Estado actual de la máquina SendLCD

Mientras que los diagramas de flujo de la tarea se encuentran en las figuras [29](#) y [30](#):

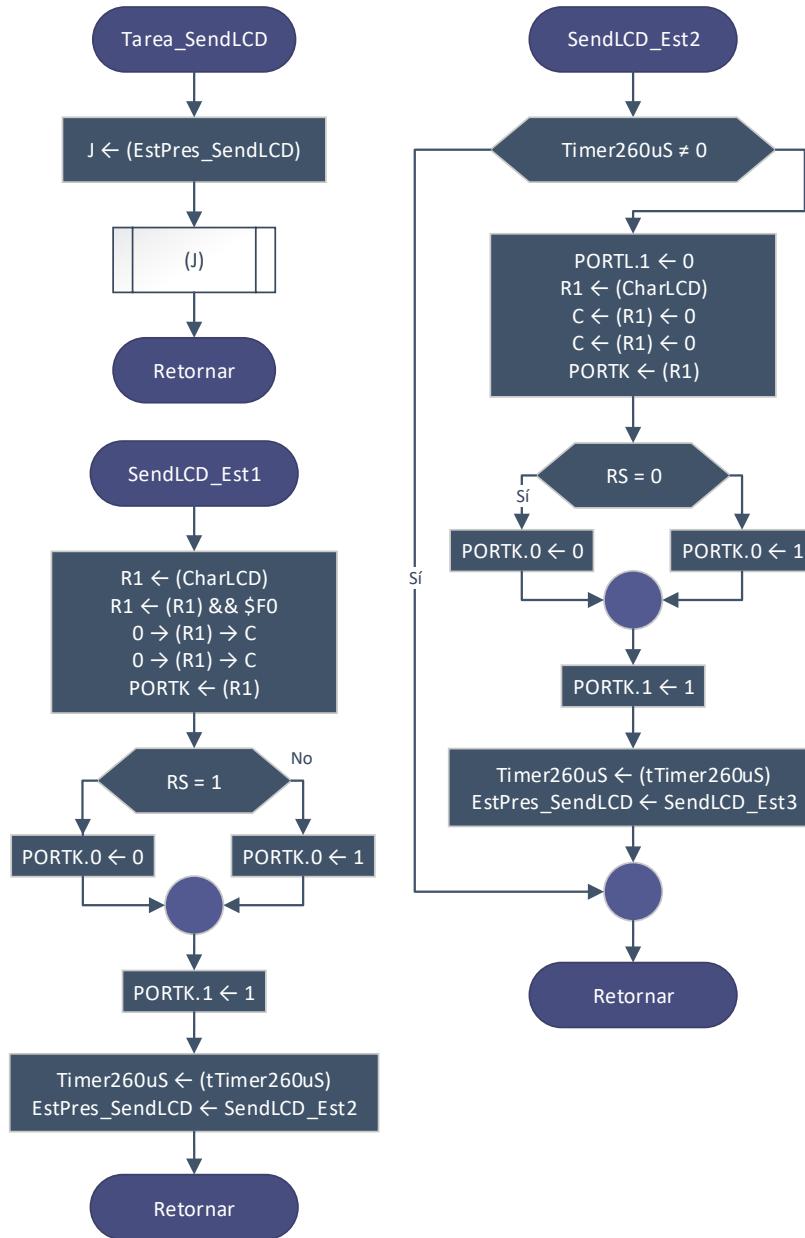


Figura 29: Diagrama de flujo de la tarea `Send_LCD`, parte 1.

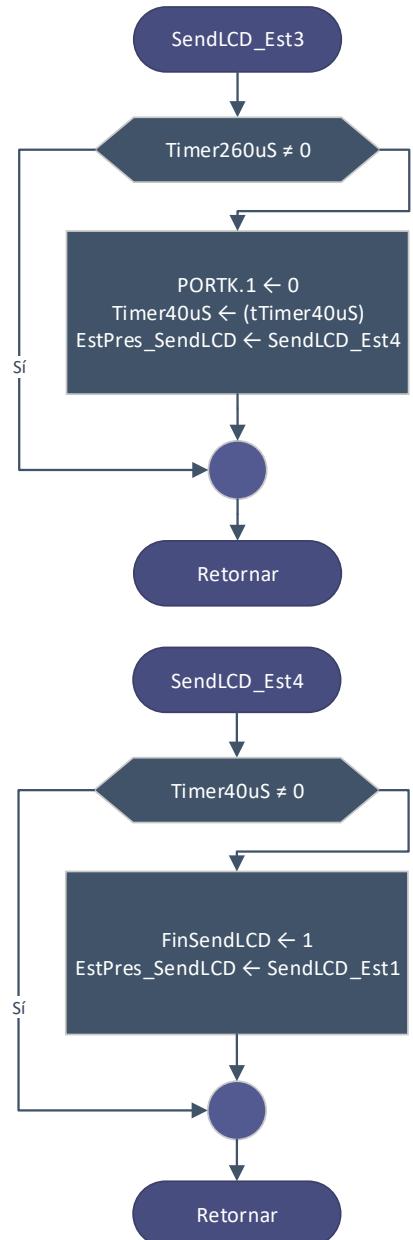


Figura 30: Diagrama de flujo de la tarea `Send_LCD`, parte 2.

2.15. Subrutina BCD BIN

Esta subrutina convierte el valor leído en el teclado matricial de BCD a binario para poder procesarlo después.

Parámetros de entrada: dirección de Num Array, se maneja como un valor.

Parámetros de salida: ValorLIM, se pasa por memoria.

Las estructuras de datos se encuentran en la tabla 14.

Cuadro 14: Valores y estructuras de datos utilizados en la subrutina BCD_BIN.

Nombre	Tipo	Magnitud o tamaño	Descripción
ValorLIM	Variable	Byte	Variable donde se almacena temporalmente la velocidad límite.
Num_Array	Valor	\$1010	Arreglo de destino de los valores leídos.

El diagrama de flujo de la subrutina se encuentra en [31](#):

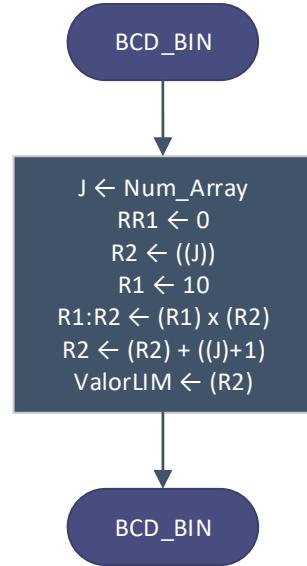


Figura 31: Diagrama de flujo de la subrutina BCD_BIN.

2.16. Subrutina Calcula

Realiza el cálculo de la cantidad de ticks que le tomó el vehículo cruzar por los sensores, obtiene así su velocidad e inicia los timers de la pantalla.

Parámetros de entrada: TimerVel, se pasa por memoria y corresponde por la resta del ticks transcurridos a ticks totales posibles.

Parámetros de salida: DeltaT, Vel_Calc, TimerPant y TimerFinPant, se pasan por memoria.

Ecuaciones utilizadas y análisis dimensional que justifica su uso:

$$\Delta T = tTimerVel - TimerVel \quad (3)$$

$$Vel_Calc = (40m/\Delta T) * (1/(1s/10dS)) * (1km/1000m) * (3600s/1h)$$

$$Vel_Calc = 1440/\Delta T \quad (4)$$

$$TimerPant = (200m/Vel_Calc)*(1/(1000m/km)*(1h/3600s))*(10dS/1s)$$

$$TimerPant = 7200/Vel_Calc \quad (5)$$

$$TimerFinPant = (300m/Vel_Calc)*(1/(1000m/km)*(1h/3600s))*(10dS/1s)$$

$$TimerFinPant = 10800/Vel_Calc \quad (6)$$

Se presentan los valores y estructuras de datos en la tabla 15:

Cuadro 15: Valores y estructuras de datos utilizados en la subrutina Calcula.

Nombre	Tipo	Magnitud o tamaño	Descripción
tTimerVel	Valor	100	Tiempo para esperar a que el vehículo pase por los sensores.
tTimerVel	Valor	100	Tiempo para esperar a que el vehículo pase por los sensores.
DeltaT	Variable	Byte	Diferencia de tiempo, en ticks, entre la activación de los sensores.
Vel_Calc	Variable	Byte	Valor calculado de velocidad
TimerPant	Variable timer	Byte	Temporizador para esperar a mientras se muestra la velocidad, base 100mS.
TimerFinPant	Variable timer	Byte	Timer para esperar a apagar la pantalla que ve el vehículo.
Cte_Dvndo_Vel	Valores	1440	Constante de dividendo para obtener la velocidad.
Cte_Dvndo_TP	Valores	7200	Constante de dividendo para obtener TimerPant
Cte_Dvndo_TFP	Valores	10800	Constante de dividendo para obtener TimerFinPant.

En la figura 32 se presentan el diagrama de flujo de la subrutina:

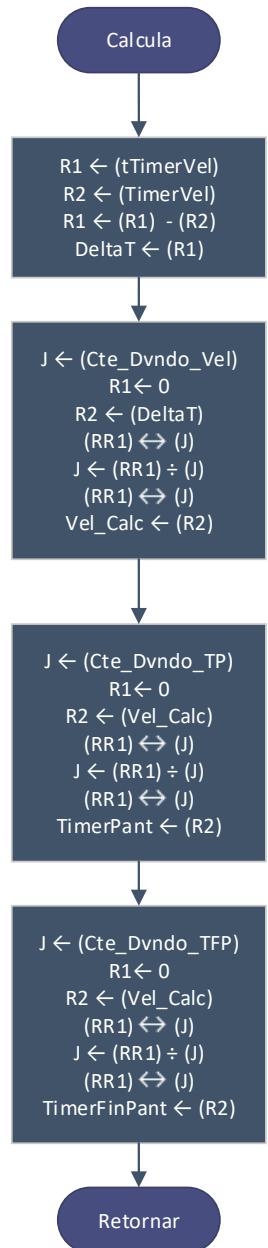


Figura 32: Diagrama de flujo de la subrutina Calcula.

2.17. Subrutina BIN BCD MUXP

Realiza la conversión de de binario a BCD para la tarea de multiplexación de los displays de 7 *segmentos*.

Parámetros de entrada: recibe el valor binario a través del acumulador *A*.

Parámetros de salida: salva el valor final en memoria en la variable *BCD*

La estructura de datos utilizada se encuentra en [16](#):

Cuadro 16: Valores y estructuras de datos utilizados en la subrutina BIN_BCD_MUXP.

Nombre	Tipo	Magnitud o tamaño	Descripción
BCD	Variable	Byte	Valor binario convertido a BCD

El diagrama de flujo de la subrutina se encuentra en [33](#)

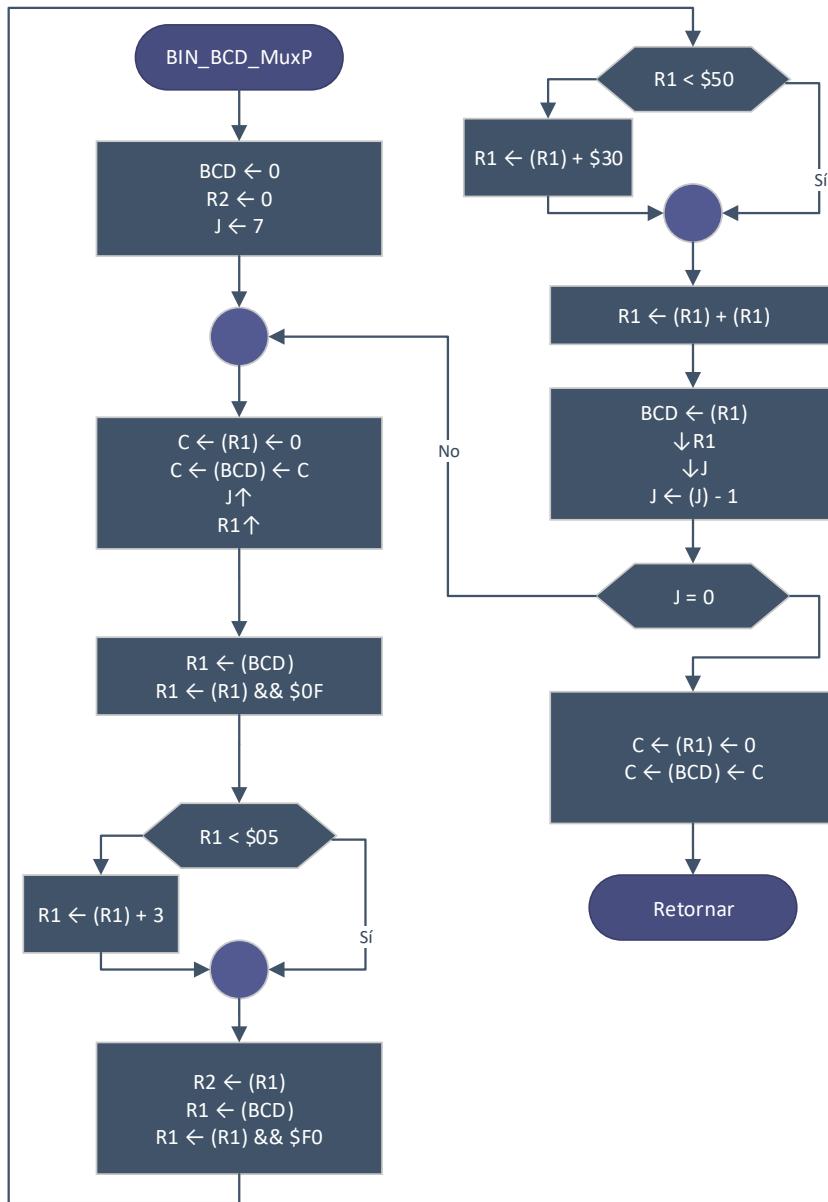


Figura 33: Diagrama de flujo de la subrutina `BIN_BCD_MUXP`.

2.18. Subrutina BCD 7SEG

Convierte valores en BCD a unos codificado para los displays de 7 segmentos.

Parámetros de entrada: por memoria BCD1 y BCD2.

Parámetros de salida: por memoria Dsp1, Dsp2, Dsp3 y Dsp4.

La tabla de estructuras de datos se encuentra en [17](#):

Cuadro 17: Valores y estructuras de datos utilizados en la subrutina BCD_7SEG.

Nombre	Tipo	Magnitud o tamaño	Descripción
Dsp1	Variable	Byte	Valor del dígito 1 codificado para 8 segmentos
Dsp2	Variable	Byte	Valor del dígito 2 codificado para 8 segmentos
Dsp3	Variable	Byte	Valor del dígito 3 codificado para 8 segmentos
Dsp4	Variable	Byte	Valor del dígito 4 codificado para 8 segmentos
BCD1	Variable	Byte	Valor binario 1 en BCD
BCD2	Variable	Byte	Valor binario 2 en BCD

El diagrama de flujo de la subrutina se encuentra en [34](#)

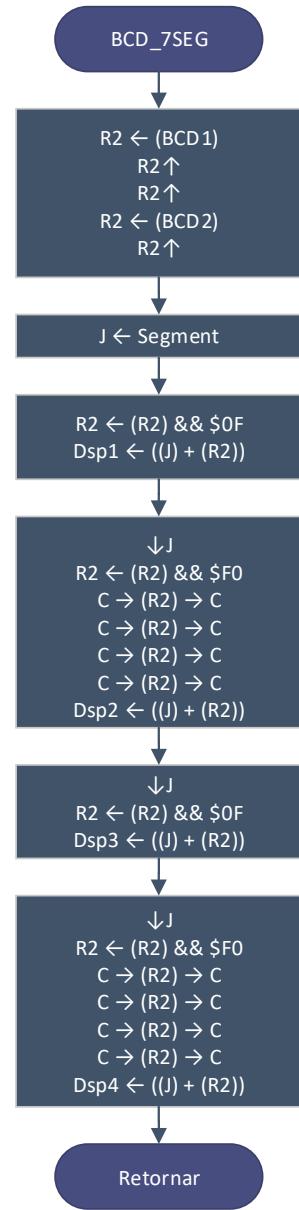


Figura 34: Diagrama de flujo de la subrutina BCD_7SEG.

2.19. Subrutina Borrar NumArray

Se encarga de borrar el array NumArray a valores $\$FF$, que son inválidos.

Parámetros de entrada: No recibe parámetros.

Parámetros de salida: No entrega parámetros si no que opera sobre el array.

La tabla de estructuras de datos se encuentra en [18](#):

Cuadro 18: Valores y estructuras de datos utilizados en la subrutina Borrar_NumArray.

Nombre	Tipo	Magnitud o tamaño	Descripción
Num_Array	Valor	\$1010	Arreglo de destino de los valores leídos.
ARRAY_OK	Variable	\$10	Etiqueta para detectar que Num_Array esté listo.
BNA_Bank_Val	Variable	\$FF	Valor en blanco para borrar NumArray

El diagrama de flujo de la subrutina se encuentra en [35](#)

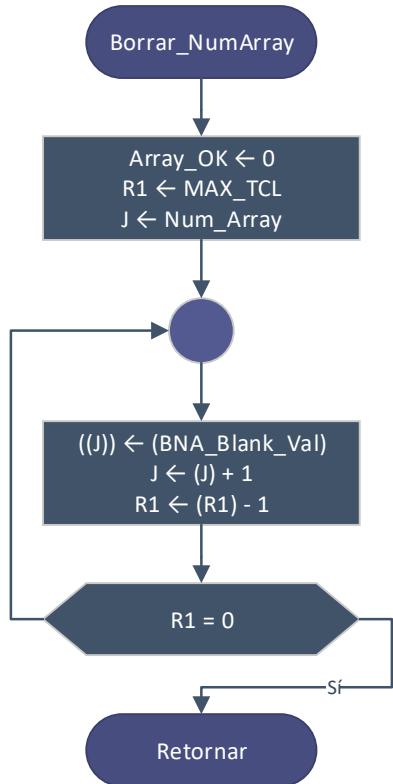


Figura 35: Diagrama de flujo de la subrutina `Borrar_NumArray`.

2.20. Subrutina Leer Teclado

Se encarga de la lectura multiplexada del teclado matricial. Tiene como referencia la tabla de equivalencia de teclas.

Parámetros de entrada: No recibe parámetros.

Parámetros de salida: Entrega el valor leído por memoria en la variable Tecla.

La tabla de estructuras de datos se encuentra en [19](#):

Cuadro 19: Valores y estructuras de datos utilizados en la subrutina Leer_Teclado.

Nombre	Tipo	Magnitud o tamaño	Descripción
Teclas	Valor	\$1110	Etiqueta de ubicación de la tabla de Teclas
Tecla	Variable	Byte	001 Valor de tecla leído

El diagrama de flujo de la subrutina se encuentra en [36](#)

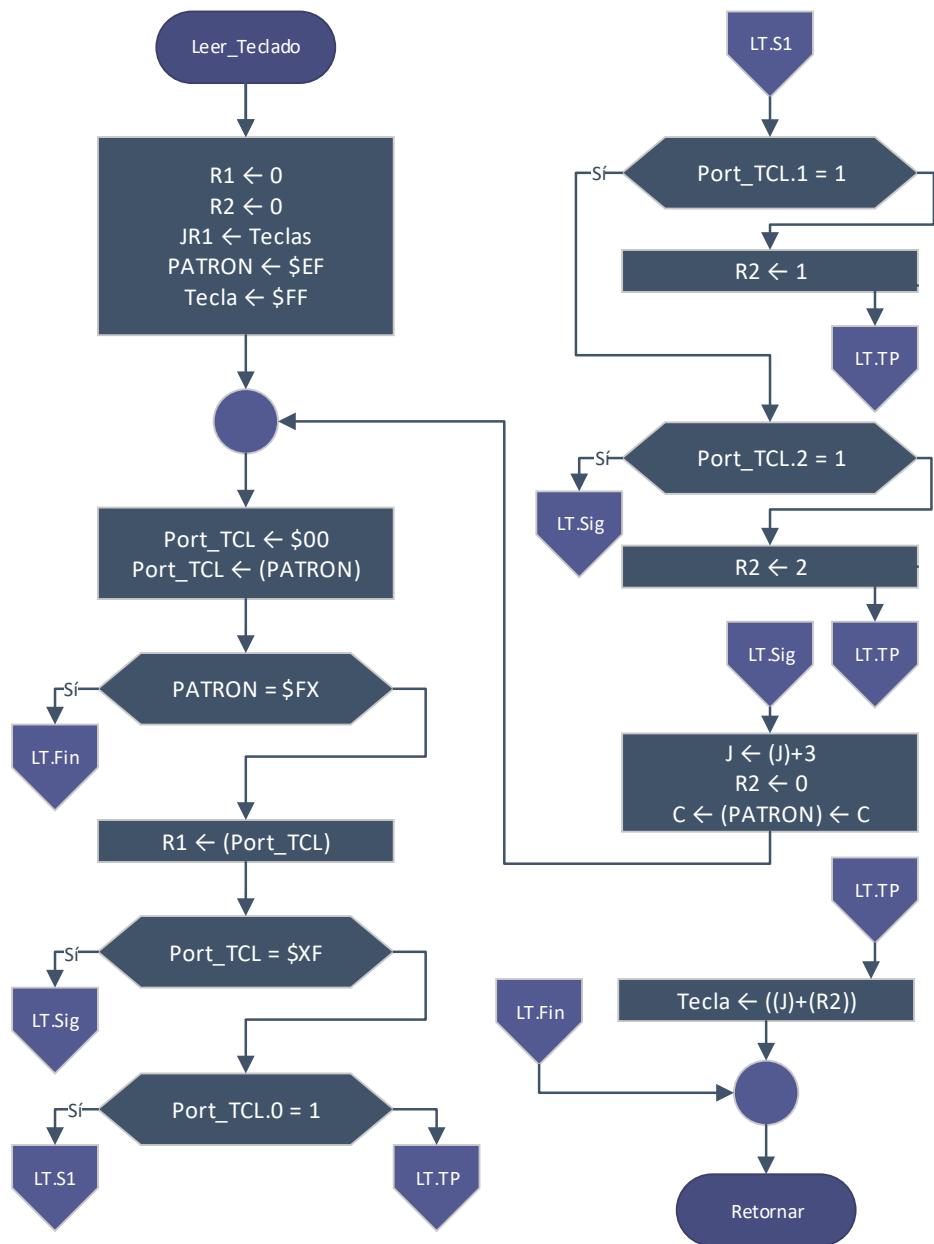


Figura 36: Diagrama de flujo de la subrutina Leer_Teclado.

2.21. Máquina de tiempos

Se utilizó una máquina que se encarga de toda la temporización del sistema operativo, se maneja por output compare del módulo contador según se vio en clase. Opera a una frecuencia de 50 kHz y tiene cuatro bases para los contadores: 1mS, 10mS, 100mS 1S. Se maneja por interrupción.

Parámetros de entrada: No recibe parámetros,

Parámetros de salida: Todos los timers del sistema por memoria.

La tabla de estructuras de datos se encuentra en [20](#):

Cuadro 20: Valores y estructuras de datos utilizados en la máquina de tiempos.

Nombre	Tipo	Magnitud o tamaño	Descripción
Timer1mS	Variable	Word	Timer base para 1mS.
Timer10mS	Variable	Word	Timer base para 10mS.
Timer100mS	Variable	Word	Timer base para 100mS.
Timer1S	Variable	Word	Timer base para 1S.
Contante_OC5_MT	EQU	30	Constante que se suma a TCNT para que la interrupción por output compare sea regular en el tiempo.

Los diagramas de flujo de la subrutina se encuentran en [37](#) y [38](#).

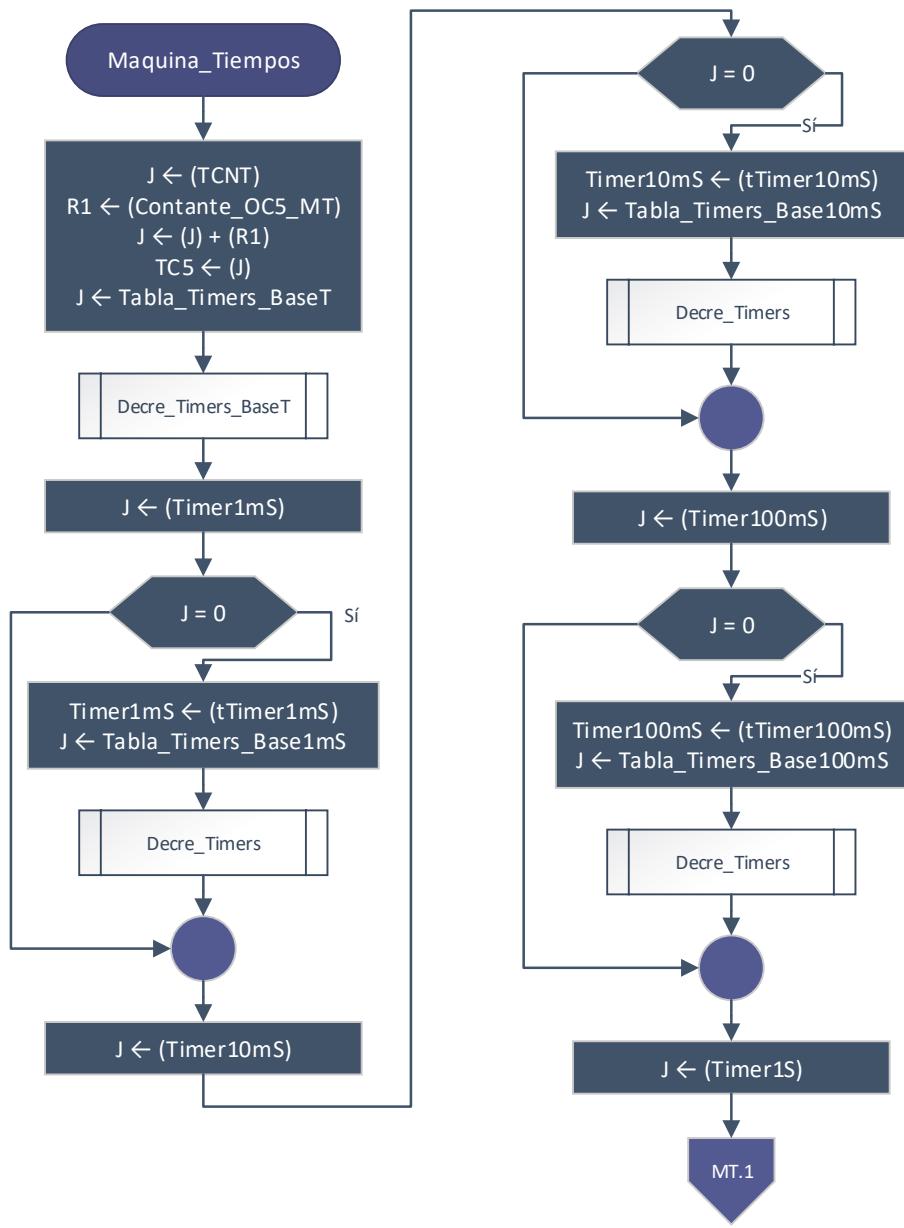


Figura 37: Diagrama de flujo de la máquina de tiempos, parte 1.

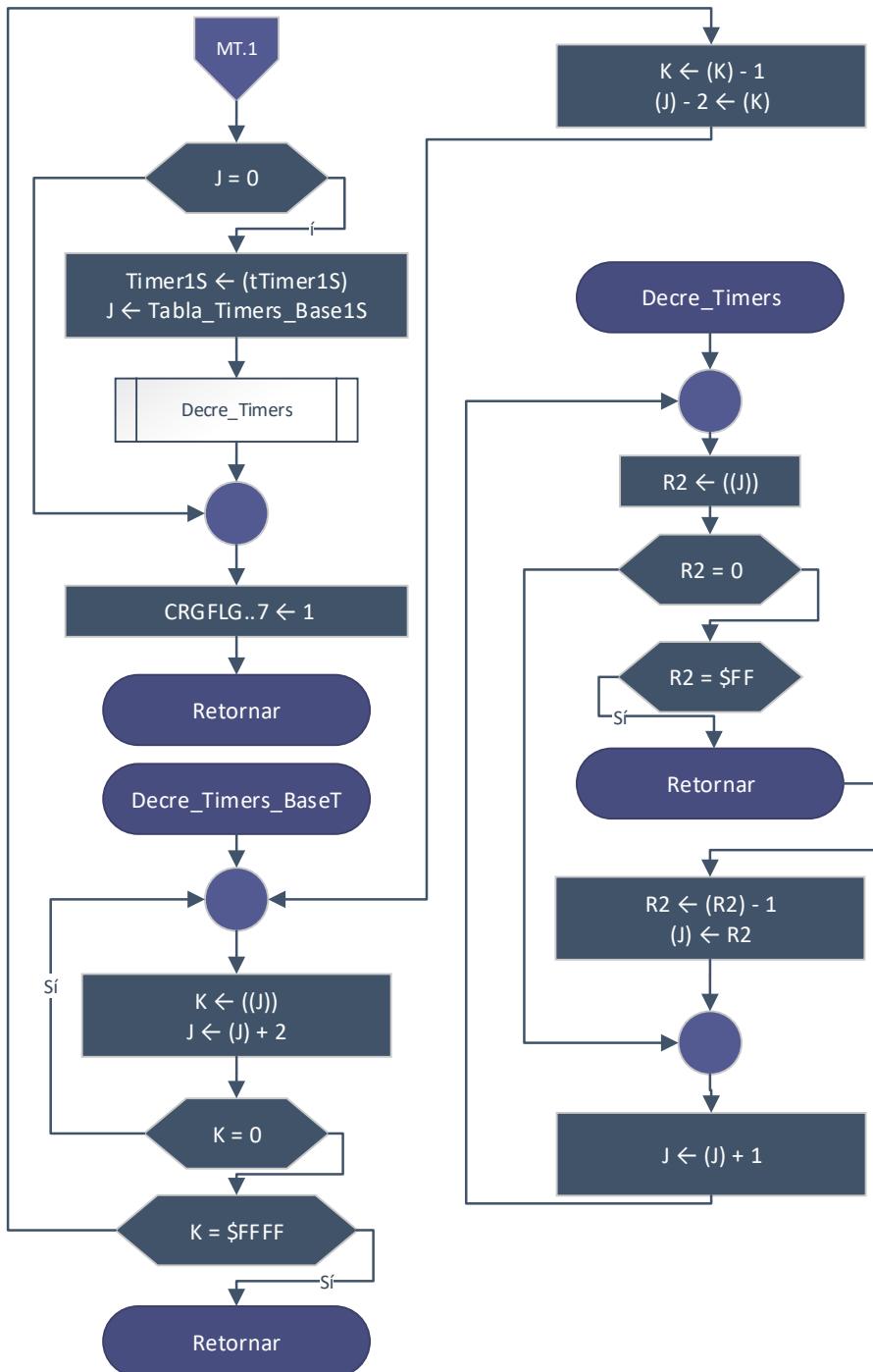


Figura 38: Diagrama de flujo de la máquina de tiempos, parte 2.

2.22. Rutina de inicialización de la pantalla LCD

Esta rutina se ejecuta sólo una vez en el ciclo de *power-on reset* y se encarga de configurar e inicializar la pantalla LCD de la *Dragon12+* conectada en el puerto K.

La tabla de estructuras de datos se encuentra en [21](#):

Cuadro 21: Valores y estructuras de datos utilizados en la inicialización de la pantalla LCD.

Nombre	Tipo	Magnitud o tamaño	Descripción
tTimer2mS	Valor	Word	Valor a asignar en el timer: 2 x 1mS
tTimer260uS	Valor	13	Valor a asignar en el timer: 13 x 1 tick
tTimer40uS	Valor	Word	Valor a asignar en el timer: 2 x 1 tick
RS	Valor	\$01	Bandera para enviar un .
FinSendLCD	Valor	\$04	Se pone en alto cuando se terminó de enviar un mensaje.
CharLCD	Variable	Byte	Carácter a enviar a la pantalla
EstPres_SendLCD	Variable	Word	Estado actual de la máquina SendLCD
Clear_LCD	Valor	\$01	Comando para limpiar la pantalla
IniDsp	Valor	\$102D	Posición de memoria del arreglo de comandos de inicialización de la LCD.

Los diagramas de flujo de la subrutina se encuentran en [39](#).

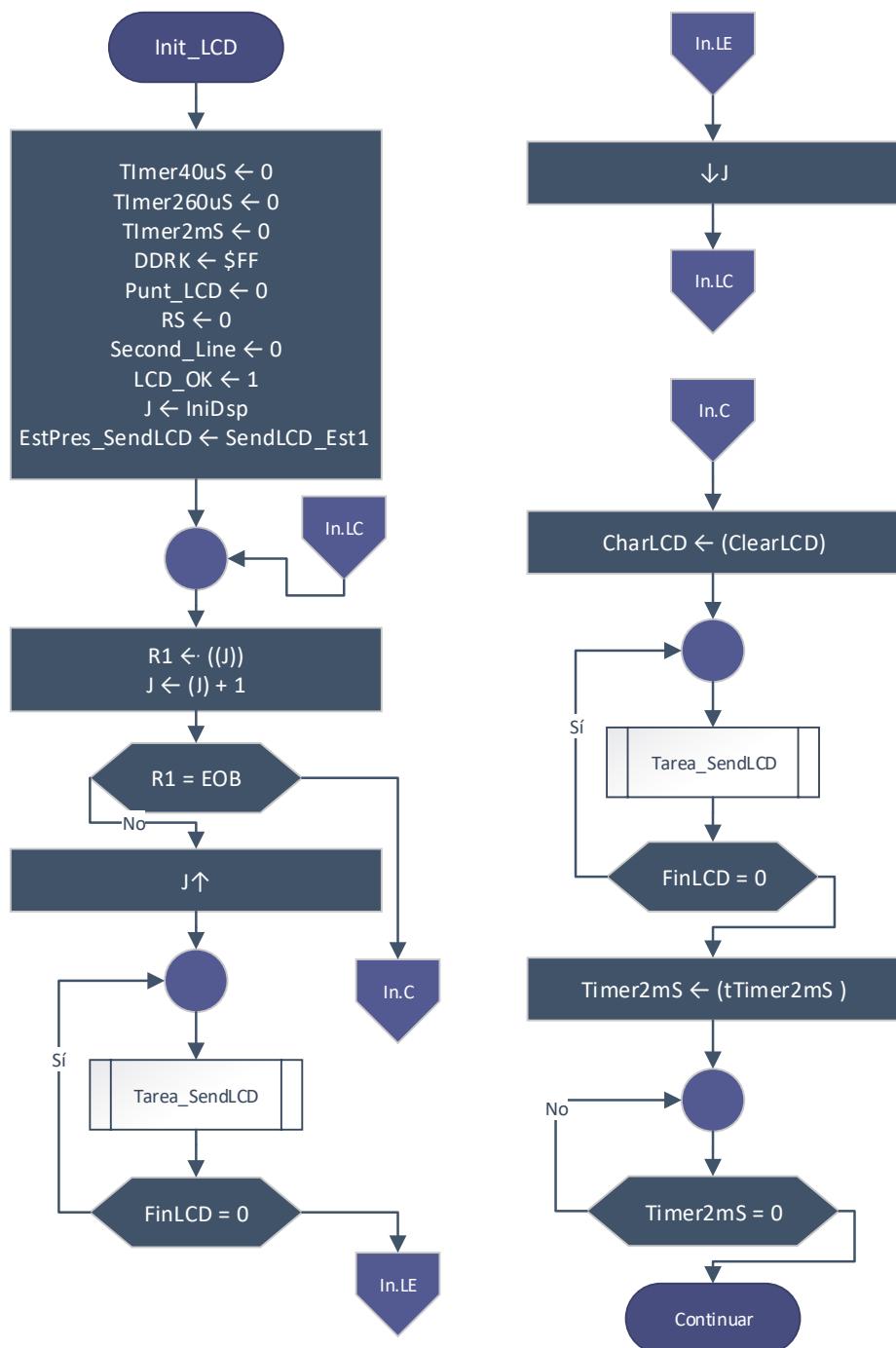


Figura 39: Diagrama de flujo de la inicialización de la pantalla LCD.

3. Conclusiones

Se logró resolver exitosamente el problema planteado a través del uso de la *Dragon12+*.

Se encuentra la tarjeta utilizada como un mecanismo eficaz de entrenamiento para la iniciación del estudiante en el campo de los sistemas incrustados.

Se probó funcional el procesador de punto fijo a la hora de interactuar con un mundo intrínsecamente analógico y laxo, lo que refuerza la necesidad del estudio de convertidores analógico digitales.

4. Comentarios personales

En general me gustó mucho el problema planteado, es algo que se presenta en la industria más allá de la academia, que puede llegar a quedarse corta a la hora de encarar la realidad nacional.

Me llevó más tiempo las subrutinas accesorias, 5 días, que las principales de los modos de operación 3 días, creo que eso es porque soy muy detallista a la hora de trabajar.

Me gustó mucho el curso en general y aprendí mucho, algo que no veo tan común en las materias de la escuela. Es bastante exigente pero lo veo como algo bueno porque fuerza al estudiante a adquirir una mejor comprensión del contexto, problemas y campo de conocimiento que se trate. Muchas gracias por todo.

5. Recomendaciones

En general todo bien, no tuve dificultades mayores a la hora de entender o resolver los problemas planteados. Estaría interesante acoplarle al sistema un radar real, pero eso daría para tres cursos más.

6. Bibliografía

Referencias

- [1] S. Barrett and D. Pack, *Embedded systems design and applications with the 6SHC12 and HCS12*. Pearson Education, Inc., 2005.