MANUAL DE USUARIO DE LA LIBRERÍA RTC_LIB

Pedro José Vera Gallego

UNIVERSIDAD DE VALLADOLID

1- Resumen de la funcionalidad de la librería

La librería RTC_lib, ha sido diseñada para el manejo de los dispositivos RTC de la marca Dallas, en concreto los dispositivos que utilizan el bus I2C, DS1307-DS3231.

Con esta librería es posible:

- Establecer la fecha y hora deseada.
- Leer la fecha y hora actual del RTC.
- Variar la fecha o la hora (un dato en concreto y en una unidad).
- Establecer distintos tipos de alarma.

2- Lista de constantes definidas

- ON → 1
- OFF → 0
- PM → 1
- AM → 0
- FORMATO_LARGO → 1
- FORMATO_CORTO → 2
- LUNES → 1
- MARTES → 2
- MIERCOLES → 3
- JUEVES → 4
- VIERNES → 5
- SABADO → 6
- DOMINGO → 7

3- Lista de funciones

- RTC
- leerHora ()
- Modo12h (AM_PM)
- escribeHora (hora, min, seg)
- escribeFecha (dia, mes, anno)
- escribeDiaSem (diaSem)
- variaHora (varia)
- variaMin (varia)
- variaAnno (varia)
- variaMes (varia)
- variaDia (varia)
- variaDiaSem (varia)
- verHora (formato)
- verFecha (formato, orden)
- verDia (formato)
- verMes (formato)
- verTemperatura ()
- alarmasON (tipo)
- escribeAlarma (dia, hora, min, seg)
- limpiaAlarma ()

4- Detalles de cada función

- RTC

  Con esta sentencia se define la clase. Es necesario declararla al inicio del programa para el uso del resto de funciones.

  p. ej: RTC rtc;

- leerHora

  Función que de tipo Reloj, estructura que contiene los valores de día, mes, año, dia de la semana, hora, hora formato 12h, minutos, segundos y cambio horario.

  No recibe ningún argumento y devuelve una estructura tipo Reloj.

  p. *ej:* Reloj x = leerHora ();

- Modo12h (AM_PM)

  Función para elegir el tipo de modo horario a utilizar, de 12 horas o de 24 horas. No devuelve ningún valor.

  Se recibe como argumento AM_PM, que es una variable booleana, la cual será 1 si se quiere el modo 12 horas y 0 si se quiere el modo 24 horas.

  Se pueden emplear las constantes ON y OFF.

  p. ej: Modo12h (OFF);

- escribeHora (hora, min, seg)

  Con esta función se inicializan, o simplemente se modifican, los registros de hora, minutos y segundos del RTC. No devuelve ningún valor.

  Los argumentos recibidos son la hora en primer lugar, después minutos y finalmente segundos.

  Es posible no escribir alguno de los valores, si el inferior no se ha escrito, siendo por defecto 0. Si un valor no se escribe y el siguiente valor si ha escrito, se producirá error de compilación.

  p. ej: escribeHora (10, 25); //Se establecerían las 10:25:00

- escribeFecha (dia, mes, año)

  Con esta función se inicializan, o simplemente se modifican, los registros de días, meses y años del RTC. No devuelve ningún valor.

  Los argumentos recibidos son el día en primer lugar, después el mes y finalmente el año.

  Es posible no escribir alguno de los valores, si el siguiente no se ha escrito, siendo por defecto 0. Si un valor no se escribe y el siguiente valor si ha escrito, se producirá error de compilación.

  p. ej: escribeFecha (6, 7, 2017); //Se establecería 6/07/2017

- escribeDiaSem (diaSem)

  Con esta función se modifica el registro de los días de la semana del RTC. No devuelve ningún valor.

  Se recibe como argumento el día de la semana.

  Se pueden emplear las constantes de los días de la semana.

  p. ej: escribeDiaSem (LUNES);

- variaHora (varia)

  Con esta función se modifica el valor del registro de la hora en una unidad.

  Se recibe como argumento una variable booleana, que será 1 si se quiere aumentar el valor o 0 si se quiere decrementar.

  Se pueden emplear las constantes ON y OFF.

  p. ej: variaHora (ON);

- variaMin (varia)

  Con esta función se modifica el valor del registro de los minutos en una unidad.

  Se recibe como argumento una variable booleana, que será 1 si se quiere aumentar el valor o 0 si se quiere decrementar.

  Se pueden emplear las constantes ON y OFF.

  p. ej: variaMin (ON);

- variaAnno (varia)

  Con esta función se modifica el valor del registro de los años en una unidad.

  Se recibe como argumento una variable booleana, que será 1 si se quiere aumentar el valor o 0 si se quiere decrementar.

  Se pueden emplear las constantes ON y OFF.

  p. ej: variaAnno (ON);

- variaMes (varia)

  Con esta función se modifica el valor del registro de mes en una unidad.

  Se recibe como argumento una variable booleana, que será 1 si se quiere aumentar el valor o 0 si se quiere decrementar.

  Se pueden emplear las constantes ON y OFF.

  p. ej: variaMes (ON);

- variaDia (varia)

  Con esta función se modifica el valor del registro de los días en una unidad.

  Se recibe como argumento una variable booleana, que será 1 si se quiere aumentar el valor o 0 si se quiere decrementar.

  Se pueden emplear las constantes ON y OFF.

  p. ej: variaDia (ON);

- variaDiaSem (varia)

  Con esta función se modifica el valor del registro de los días de la semana en una unidad.

  Se recibe como argumento una variable booleana, que será 1 si se quiere aumentar el valor o 0 si se quiere decrementar.

  Se pueden emplear las constantes ON y OFF.

  p. ej: variaDiaSem (ON);

- verHora (formato)

  Función que convierte los valores de los registros de horas, minutos y segundos en una cadena de caracteres, para ser entendible por el usuario. Se devuelve un String.

  Como argumento se recibe un entero para seleccionar el formato deseado, que será 1 si se quiere un formato largo, en que se muestran los segundos o 0 si se quiere un formato corto, sin los segundos. Si no se añade, por defecto el formato es largo.

  Se pueden emplear las constantes FORMATO_LARGO y FORMATO_CORTO.

  p. ej: verHora (FORMATO_LARGO); //mostraría 00:00

- verFecha (formato, orden)

  Función que convierte los valores de los registros de día, mes y año en una cadena de caracteres, para ser entendible por el usuario. Se devuelve un String.

  Como argumento se recibe un entero para seleccionar el formato deseado, que será 1 si se quiere un formato largo, en que se muestran los segundos o 0 si se quiere un formato corto, sin los segundos.

  Se pueden emplear las constantes FORMATO_LARGO y FORMATO_CORTO.

  El otro parámetro que se pasa a la función es el orden, que selecciona el modo en que se quiere la cadena con los enteros:

  - 1 → Día/Mes/Año
  - 2 → Año/Mes/Día
  - 3 → Mes/Día/Año

  p. ej: verFecha (FORMATO_LARGO, 1); //mostraría 6/07/2017

- verDia (formato)

  Función que convierte el valor del registro día de la semana en una cadena de caracteres, para ser entendible por el usuario. Se devuelve un String.

  Como argumento se recibe un entero para seleccionar el formato deseado, que será 1 si se quiere un formato largo, en que se muestran los segundos o 0 si se quiere un formato corto, sin los segundos.

  Se pueden emplear las constantes FORMATO_LARGO y FORMATO_CORTO.

  p. ej: verDia (FORMATO_CORTO); //mostraría Lun

- verMes (formato)

  Función que convierte el valor del registro mes en una cadena de caracteres, para ser entendible por el usuario. Se devuelve un String.

  Como argumento se recibe un entero para seleccionar el formato deseado, que será 1 si se quiere un formato largo, en que se muestran los segundos o 0 si se quiere un formato corto, sin los segundos.

  Se pueden emplear las constantes FORMATO_LARGO y FORMATO_CORTO.

  p. ej: verMes (FORMATO_LARGO); //mostraría Julio

- verTemperatura ()

  Función que lee los valores almacenado en los registros de temperatura y devuelve un float con el valor de la temperatura en grados Celsius.

  p. ej: *float* temp = verTemperatura ();

- alarmasON (tipo)

  Con esta función se activan las alarmas y se selecciona el tipo de alarma que se quiere activar. No devuelve ningún valor.

  Como argumento se pasa un entero entre el 1 y el 8, que indica el tipo de alarma a usar. Las 4 primeras corresponden a la alarma 1 y las siguientes 4 a la alarma 2. Si no se añade será 8. Los tipos de alarma son los siguientes:

  - 1 → Alarma cada segundo.
  - 2 → Coincidencia de segundos con la alarma establecida.
  - 3 → Coincidencia de minutos y segundos.
  - 4 → Coincidencia de hora, minutos y segundos.
  - 5 → Alarma cada minuto.
  - 6 → Coincidencia de hora y minutos.
  - 7 → Coincidencia de día de la semana, hora y minutos.
  - 8 → Coincidencia de día del mes, hora y minutos.

  p. ej: alarmasON (4); //Alarma al coincidir hora/min/seg

- escibeAlarma (día, hora, min, seg)

  Con esta función se establece el instante en que se quiere forzar la alarma. No devuelve ningún valor.

  Esta función debe ser precedida por la función alarmasON.

  Como argumentos se reciben cuatro enteros, el día o día de la semana, dependiendo de la alarma establecida, la hora, los minutos y los segundos.

  En caso de añadir un campo que el tipo de alarma no necesita, no pasa nada, simplemente no se tendrá en cuenta.

  Es posible no incluir uno de los argumentos, si el siguiente no se ha añadido. Cuando no se añada un parámetro, será por defecto 0 para las horas, minutos y segundos, y 1 para el día.

  p. ej: escibeAlarma (10,15,10); //Alarma tipo 8, el 10 a las 15:10

- limpiaAlarma ()

  Función necesaria para limpiar los flags de las alarmas.

  Cuando se activa una alarma, cambia el valor del registro de estado. Esta función reestablece el valor inicial de dicho registro.

  Es necesario emplear esta función una vez se haya producido la interrupción deseada por la alarma establecida.

  No recibe ni devuelve ningún valor.

  p. ej: limpiaAlarma ();

5- Precauciones para el correcto funcionamiento

- Es importante inicializar la clase con la **RTC** para que el compilador no detecte errores.
- Antes de realizar funciones que requieran un control de tiempo, asegurarse de inicializar la hora al valor deseado mediante escibeHora y escibeFecha.
- Al almacenar valores de los registros, asegurarse que la lectura se asigna a una variable tipo **Reloj**.
- Emplear la función alarmasON con el tipo correcto de alarma que se desee establecer y a continuación establecerla con escibeAlarma.
- Asegurarse de usar limpiaAlarma tras producirse la interrupción.
- No emplear la función delay en el código fuente de nuestro proyecto, ya que las interrupciones no funcionarán correctamente.
- Asegurarse de que las variables que se vayan a modificar dentro de la interrupción a la que llama la alarma se declaran como volatile.