Funcionamiento placa LoRa MakeAlc

La placa diseñada en el Maker para hacer proyectos enfocados a IoT está pensada para poder ser alimentada con un par de baterias AAA-LR03.
Para que la vida de las baterias sea lo más larga posible y no haya que estar cambiandolas continuamente el funcionamiento de la placa será el siguiente. La placa una vez conectada, tomará la medida correspondiente (o lo que tenga que hacer). Enviará los datos a través de Lora al servidor que corresponda y entrará en modo “sueño profundo” hasta que un evento externo la despierte. En ese momento se repite el ciclo.

Para tener un evento externo que nos despierte a intervalos regulares y que el consumo sea el mínimo posible se ha utilizado un temporizador TPL5010.
El tiempo entre eventos se fija con una resistencia. En función del valor de la misma los eventos se producirán en más o menos tiempo. En la placa actual el tiempo está fijado en un minuto. Si se desea un tiempo distinto, solo hay que reemplazar la resistencia actual por otra con el valor correspondiente.

El módulo TPL5010 cumple dos funciones. Una la de enviar una señal de forma periódica para despertar al micro y la otra vigilar que no se cuelgue. En caso de detectar un posible cuelgue se encargaria de reiniciar el micro.

La señal periódica se envia por el pin “Wake” del temporizador. Este pin estará conectado un pin del micro con una interrupción asociada para que se “despierte” al recibir la señal.
La forma de detectar si se ha colgado el micro es detectar si el micro a enviado una señal al pin “Done” del temporizador. Esta señal se tiene que recibir al menos 20 milisegundos antes de que el el temporizador envie una nueva señal por la patilla “Wake”.
En el repositori de GitHub del Maker dedicado a la placa Lora está el datasheet del temporizador. Entre toda la información hay una tabla con los valores de las resistencias a poner en función del tiempo que queramos establecer. Así como un cálculo para poder seleccionar el tiempo que queramos.

Dado que si el temporizador no recibe una señal en el pin “Done” reinicia el micro, tendremos que preparar lo programas para que al recibir la señal “Wake” envien la respuesta al pin “Done”.

En la cabecera del programa incluiremos el fichero con la definición de los pines de la placa
```
#include <pines.h>
```

En el Setup definiremos la patilla Done como salida para poder enviar la respuesta y asociaremos una interrupción a Wake para detectar cuando se activa.
```
pinmode(TPL5010_DONE, OUTPUT)
```

Para activar la detección de una interrupción se utiliza el comando “attachInterrupt(Interrupción a detectar, Función a ejecutar, Modo)
Si no sabemos que interrupción corresponde a el pin seleccionado para recibir la señal, utilizaremos el comando “digitalPinToInterrupt” al cual le pasamos el pin y nos devuelve el número de interrupción correspondiente. El comando quedará de la siguiente forma:
```
attachInterrupt(digitalPinToInterrupt(TPL5010_WAKE), Nombre Función, RISING);
```
(RISING se activa cuando pasamos de 0 a 1)

Teniendo estas dos líneas en el setup ya tenemos todo lo necesario para detctar la interrupción y enviar una señal al pin “Done” del temporizador.

Hay que tener en cuenta que como para que el temoprizador no reinicie el micro hay que enviar la señal a “Done” antes de la siguiente señal de Wake, si esperamos a recibir la primera el micro se resetaeará antes de recibirla, ya que el temporizador empieza a contar desde el mismo momento que alimentamos la placa.
Para evitar esto tendremos que enviar la primera señal antes de salir del setup. A partir de este momento solo cada vez que detectemos la interrupción.

Como esta placa se programa mediante el protocolo UPDI para subir el programa a la placa necesitamos un programador especial que funcione con este protocolo. Existe en el mercado uno llamado “Serial UPDI” que nos permite programar la placa mediante UPDI y monitorizar el funcionamiento mediante el monitor serial.

El cambio de protocolo entre UPDI y SERIAL se puede hacer de forma automática dejando el swicht del programador en modo auto (el programa que envie los datos a la placa tiene que “conocer” el funcionamiento de la misma) o seleccionando manualmente los modos SERIAL o UPDI.

El programa que utiliza PlatformIO para subir el programa a la placa es el avrdude. Este programa en su versión 7.0 soporta el programador “Serial UPDI” por lo que si no tenemos instalada la versión 7.0 o posterior tendremos que instalarla para de poder subir el programa a la placa.
El link de descarga de la versión 7.0 del programa en función del sistema operativo que tengamos la podemos consultar en: https://github.com/avrdudes/avrdude/blob/main/README.md

Instalación en Windows:
Podemos descargar del siguiente enlace un fichero ZIP para la instalación.
https://github.com/AsociacionMakerAlicante/PlacasMaker/raw/main/Install_avrdude.zip
En el ZIP de descarga encontramos tres archivos con el nombre de avrdude y distintas extensiones.
Solo tendremos que copiar los ficheros con extensión “exe” y “conf” a los directorios correspondientes.
El directorio por defecto para la plataforma de nuetra placa es:
%USERPROFILE%\.platformio\packages\tool-avrdude-megaavr
Como avrdude se utiliza en mas plataformas podemos encontrarnos varios directorios con nombres parecidos. La herramienta es la misma para todas las plataformas, por lo que podemos aprovechar apra actualizarlas toadas. En principio actualizaremos todos los directorio cuyo nombre empiece por “tool-avrdude”

Se incluye el fichero “Install_avrdude.ZIP” el fichero Install_avrdude.bat que se encarga de la instalación automática.