Sistema de control de temperatura óptico en el embarque

Grado en Ingeniería Radioelectrónica

Escuela de Ingenierías Marina, Náutica y Radioelectrónica

EIMANAR

Paula Gallego López

Tutor: Ana Isabel Vázquez Mejías

29/06/2021

AGRADECIMIENTOS

Mis más sinceros agradecimientos a todas las personas que me han ayudado a lo largo de estos 4 años.

En primer lugar, a mi familia, tanto padres, abuelos y hermano por apoyarme en la decisión de estudiar el Grado de Ingeniería Radioelectrónica.

En segundo lugar, a mi tutora Ana Isabel, por su ayuda en la idea, planificación, información y organización de este Trabajo Fin de Grado.

También, agradecer a los profesores de la Universidad de Cádiz por hacerme sentir cómoda en las clases y por animarme a seguir adelante con los estudios. Después de este período universitario sin duda he crecido como persona y me gustaría agradecerlo.

A todos, muchísimas gracias

“Sistema de control de temperatura óptico en el embarque”

RESUMEN

En el presente proyecto final de grado, se ha desarrollado un sistema de control de temperatura óptico para ser utilizado en el embarque y en otros lugares donde halla espacios cerrados. Está formado por dos partes: la primera, una cámara térmica que realiza un cribado de seguridad de forma rápida y segundo, un termómetro para confirmar. Las dos partes tienen un sistema visual para detectar si la temperatura es correcta o no. Este sistema pretende evitar que se extienda cualquier virus que tenga como síntoma la elevación de la temperatura de una persona.

“Optical temperature control system on boarding”

ABSTRACT

In this final degree project, an optical temperature control system has been developed to be used in boarding and other places where there are closed spaces. It is made up of two parts: the first, a thermal camera that makes a security screening quickly and second, a thermometer to finish. Both parts have a visual system to detect if the temperature is correct or not. This system wants to prevent any virus from spreading that is symptomatic of a person’s elevated temperature

ÍNDICE

ÍNDICE DE ILUSTRACIONES

ÍNDICE DE VIDEOS

ÍNDICE DE TABLAS

1 INTRODUCCIÓN

En la situación actual en la que nos encontramos, se deben de tomar muchas medidas de seguridad y prevención con respecto a la pandemia que vivimos. Por este motivo, una de las medidas es la toma de temperatura del personal que accede al barco. Ya que la fiebre es uno de los síntomas más comunes del Covid y podemos intentar evitar la propagación de este virus y otros similares, si los trabajadores no presentan una temperatura fuera de lo normal (37º-38ºC).

El uso de los termómetros, cámaras y escáneres térmicos han aumentado en este último año, gracias a los cuales podemos conocer la temperatura de las personas sin contacto. Por este motivo, el diseño de un dispositivo óptico que pueda medir la temperatura de los tripulantes y viajeros de un buque a la hora de embarcar es importante. Ya que se evita el contacto a la hora de tomarla y se puede detectar posibles contagios.

Las medidas que actualmente se están tomando con relación a la pandemia mundial que vivimos, deberían seguir para evitar posibles situaciones no deseadas. En especial, mantenerlas en los embarques ya que, los barcos y aeronaves son espacios cerrados en los que se encuentran trabajadores y pasajeros. Si se detectase incidencias en relación con una temperatura elevada, reduciría la probabilidad de que nos encontrarnos con situaciones en las que los tripulantes y pasajeros sufran un contagio de alguna enfermedad.

En la actualidad existen diferentes tipos de termómetros y cámaras térmicas.

Algunos de los termómetros que hay en el mercado son:

- De vidrio o líquidos
- Digitales
- Sin contacto o pirómetros
- Con lámina bimetálica
- De gas
- De resistencia

Ilustración 1.1:Tipos de termómetros

En el caso de las cámaras térmicas son:

- Cámaras térmicas portátiles
- Cámaras térmicas fijas
- Columna de medición
- Por reconocimiento facial

Ilustración 1.2:Cámaras fijas

Ilustración 1.3:Cámara portátil

Ilustración 1.4:Columna de medición

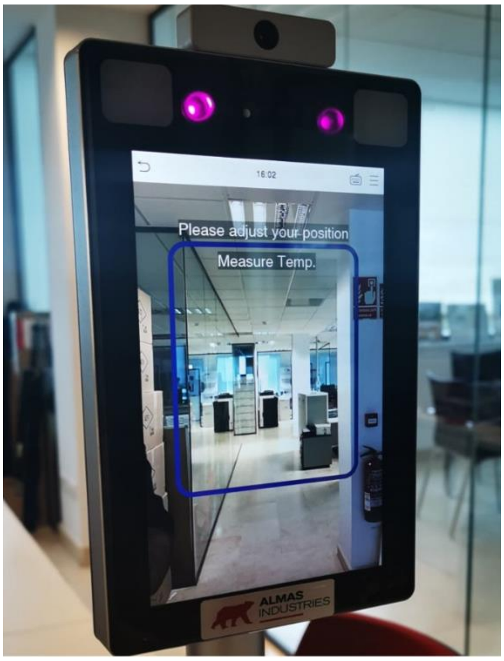


Ilustración 1.5: Por reconocimiento facial

1.1 OBJETIVOS Y REQUERIMIENTOS TÉCNICOS

Objetivo: desarrollar un sistema capaz de medir la temperatura y evitar posibles contagios o propagación de cualquier peligro potencial de contagios por los diferentes virus que existen. Detectando las personas con fiebre, es posible intentar reducir los contagios.

El sistema deberá ser capaz de efectuar medidas de las personas cuando se acerquen al termómetro o pasen por la cámara térmica y superen una temperatura se active una alerta visual o auditiva para conocer el posible caso de peligro.

La medición debe de tener cierta precisión para no efectuar medidas de temperaturas erróneas. Se debe de elegir sensores que no cuenten con mucho rango de error.

1.2 METODOLOGÍA

Las fases del proyecto son:

1. Investigación sobre lo que puede ofrecer el mercado en relación a los diferentes termómetros y cámaras térmicas que es posible comprar actualmente
2. Conocer los sensores que han facilitado para el diseño de este proyecto, en este caso los sensores seleccionados son: AMG8833 y MLX90614
3. Planificación del sistema que se quiere conseguir, se planteará el objetivo de lo que se quiere diseñar
4. Diseño del sistema, se crean los programas de las dos partes que forman el proyecto y se realizará el montaje
5. Pruebas del funcionamiento, una vez que se creen los programas y se realice el montaje se comprobará su funcionamiento
6. Si no se obtiene el objetivo en algunas de las partes, se retoma el diseño de los programas o el montaje para solucionar el problema y se volverá a comprobar
7. Conclusiones, cuando se obtiene el funcionamiento deseado se realizará el estudio de los resultados obtenidos y se presentarán las conclusiones del proyecto

1.3 ORGANIZACIÓN DE LA MEMORIA

En el primer capítulo se expone la motivación y el objetivo de la realización del proyecto para poder evitar la propagación de cualquier peligro potencial. También, se comentan algunos termómetros y cámaras térmicas que existen actualmente.

En el segundo, se expone las características técnicas de los elementos seleccionados que se van a utilizar en el proyecto: sensor GY-906, pantalla LCD de 2x16, Arduino mega 2560 R3, módulo de comunicación I2C, AMG8833 y Raspberry Pi 3 model B.

En el tercer capítulo se entra en la parte técnica del desarrollo del producto. Se muestra el diseño de las conexiones realizadas del hardware y cómo se ha diseñado el software utilizado en el proyecto.

Se comentará un poco los lenguajes de programación usados en este proyecto en el cuarto capítulo.

El quinto capítulo corresponde a pruebas de funcionamiento en las que se incluyen los ensayos realizados, datos y vídeos de los mismos.

En el sexto capitulo, se encuentran las posibles mejoras que se le podrían implementar a este proyecto.

En el capítulo siete, aparecerá el presupuesto necesario para la realización del proyecto.

Por último, las conclusiones obtenidas al terminar el trabajo y diferentes anexos donde se puede encontrar los códigos de los programas y los data sheet de los sensores.

2 MATERIAL

2.1 GY-906

El módulo detecta la temperatura por medio de radiación infrarroja, el cual puede medir la temperatura de un objeto a distancia. En él se encuentra el sensor MLX90614 y es fabricado por la empresa Melexis. [1]

La comunicación se realiza por SMBus (subconjunto de bus I2C), por lo que la lectura es sencilla y se puede usar más de un sensor de forma simultánea. Al SMBus se le conectan los chips de monitorización del sistema, estos sirven para medir temperaturas de objetos, velocidad de ventiladores, voltajes, etc.[1]

Según la ley Stefan-Boltzmann, todo objeto que este por encima del cero absoluto emite radiación y su espectro es proporcional a su temperatura. El sensor MLX90614 recoge esta radiación y la salida es una señal eléctrica proporcional a la temperatura de los objetos en su campo de visión. Viene calibrado de fábrica con una precisión de 0,5ºC. El sensor, al recoger esta radiación puede obtener dos tipos de lecturas[1]:

- Rango de temperatura ambiente: -40ºC a 170ºC
- Rango de temperatura de un objeto: -70 ºC a 380 ºC

Ilustración 2.1: Sensor GY-906

Para realizar las conexiones de este módulo, necesitamos 4 conexiones: VIN, GND, SCL y SDA. La comunicación de datos I2C es en serie y sincrónica. Una de las señales marca el tiempo, pulsos de reloj (SCL- System Clock) y la otra intercambia datos entre los dispositivos (SCA- System Data). [1][2]

2.2 PANTALLA LCD 2X16

La pantalla LCD (Liquid Crystal Dysplay), es una pantalla de cristal líquido en el que se visualiza contenido. La pantalla es de 16x2, es decir, 2 filas de 16 caracteres cada una.[3]

Esta tiene diferentes tipos de pines: alimentación, control y bus de datos bidireccionales. También, los pines de ánodo de led backlight y cátodo de led backlight.[3]

Pines de Alimentación:

- Vss: Gnd
- Vdd: 5 voltios
- Vee: pin de contraste, se conecta un potenciómetro de 10 K conectado.

Pines de control:

- RS: es el pin de selección de registro control de datos (0) o registro de datos (1). Cuando es 0 hay un registro de control/instrucción y cuando es 1 es un registro de datos o un carácter.
- RW: es el pin de escritura (0) o de lectura (1). Permite escribir un dato en la pantalla o leer un dato desde la pantalla.
- E: es el pin enable o de habilitación. Si es 0 corresponde a que la LCD no esta activada para recibir datos, si es 1 se encuentra activa para leer o escribir.

Pines de bus de datos: corresponde a los pines D0 a D7. Se puede realizar una comunicación utilizando 8 bits (D0 a D7) o 4 bits más significativos (D4 a D7).[3]

Existe dos tipos de memoria, DDRAM (Data Display Ram) y CGROM. La primera, es donde se almacena la información que será representada y la segunda, es donde se almacena una tabla con 192 caracteres que se pueden visualizar en la pantalla. En la parte CGRAM (Character Generator Rom) se puede almacenar caracteres propios[3]

Ilustración 2.2: Pantalla LCD

La pantalla LCD tiene dos formas de conectarla con Arduino.

- La primera, consiste en conectar por medio de cables individuales los diferentes pines de la pantalla al Arduino, también se debe de usar un potenciómetro de 10 K, para ajustar el contraste, conectado a V0.

A continuación, se mostrarán unas imágenes de cómo se realizaría las conexiones:

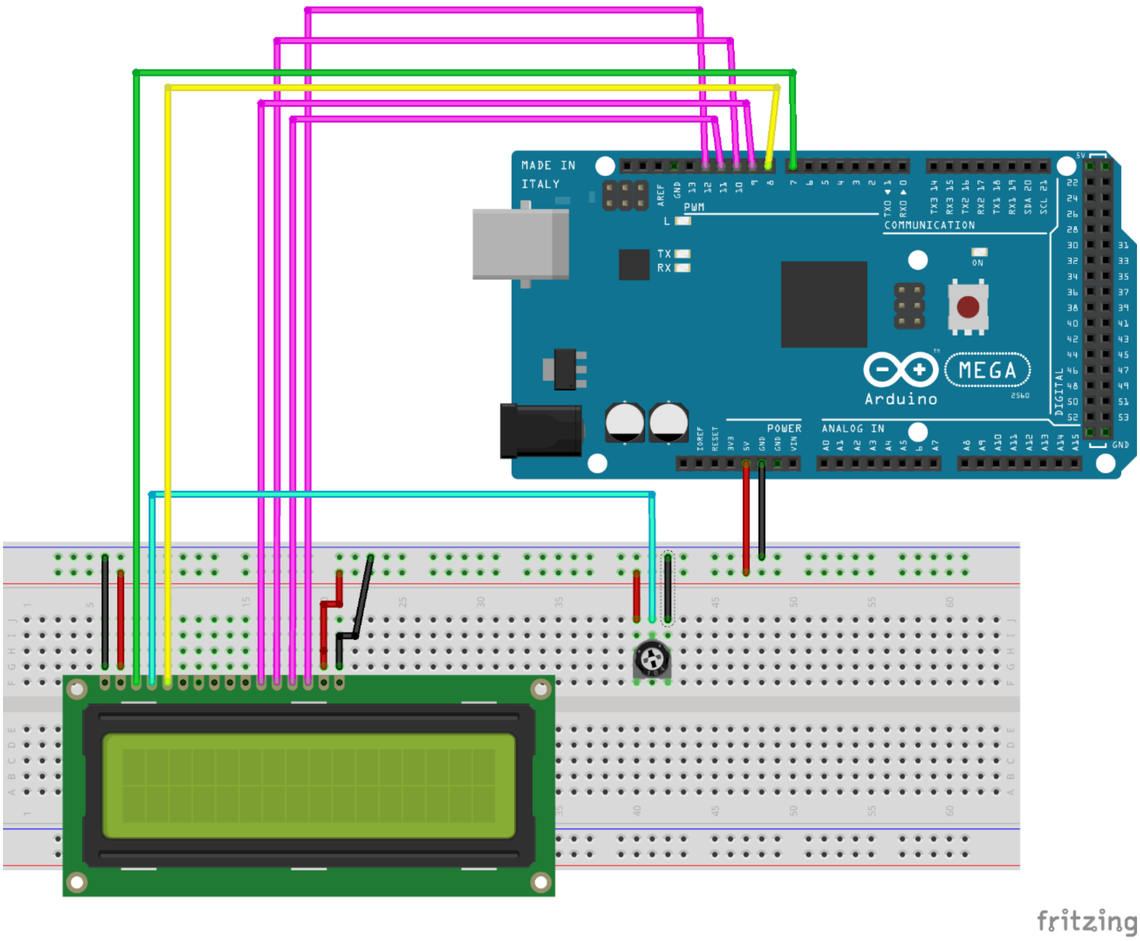


Ilustración 2.3: Conexión LCD con Arduino (opción 1)

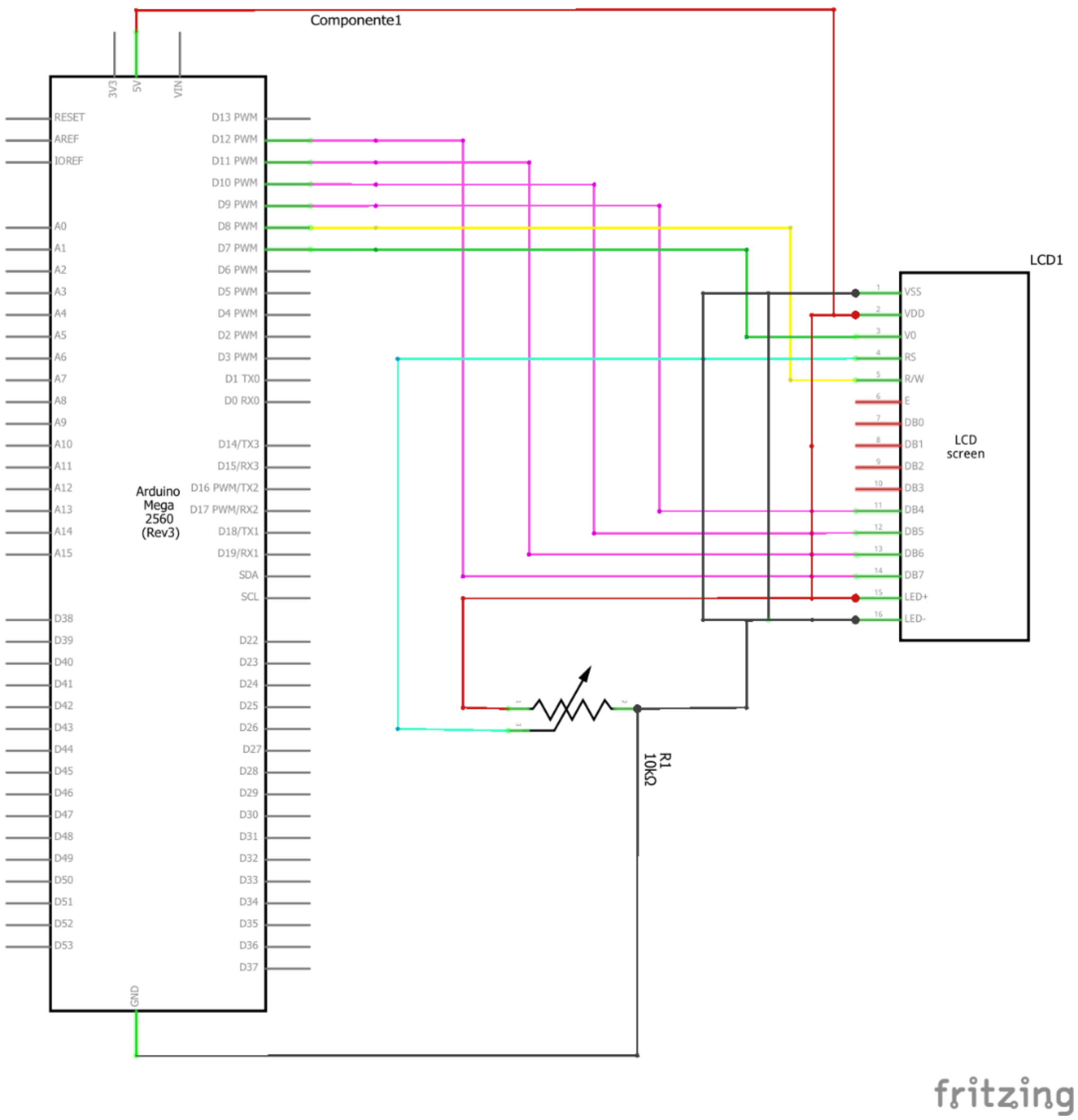


Ilustración 2.4: Esquema electrónico (opción 1)

- La segunda, utilizada en este caso sería usando el módulo I2C. Este simplifica el caso anterior al usar solo dos pines de comunicación (SCL y SCA). Este módulo es compatible con los LCD 1602 y LCD 2004. El Módulo está basado en el controlador I2C PCF8574. Esta, ofrece una transmisión rápida y transforma la transmisión de datos en serie de la pantalla LCD en una señal I2C. Solo necesita 4 conexiones (2 de comunicación y 2 de alimentación) en lugar de 16.[4]

A continuación, se mostrarán unas imágenes de cómo se realizaría las conexiones con el módulo I2C y también, una imagen real de este módulo:

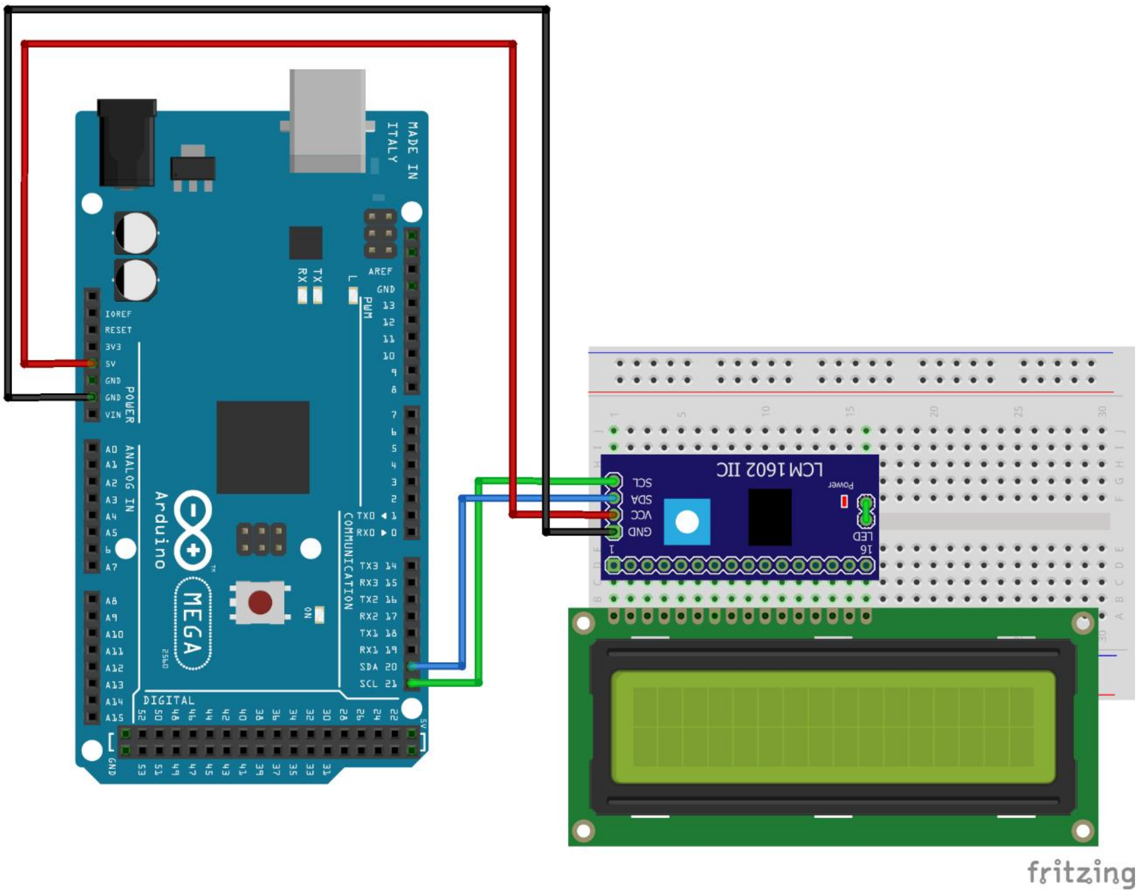


Ilustración 2.5:Conexión LCD con Arduino (opción 2)

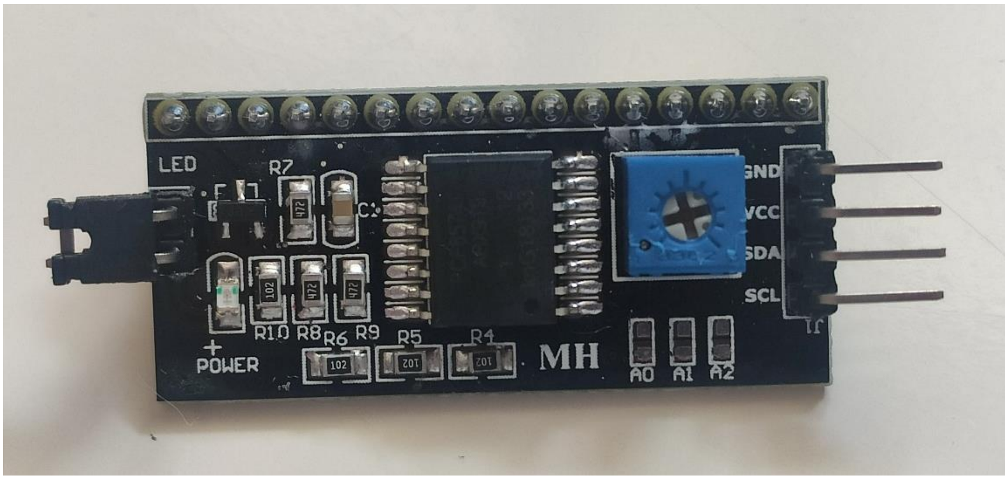


Ilustración 2.6:Módulo I2C

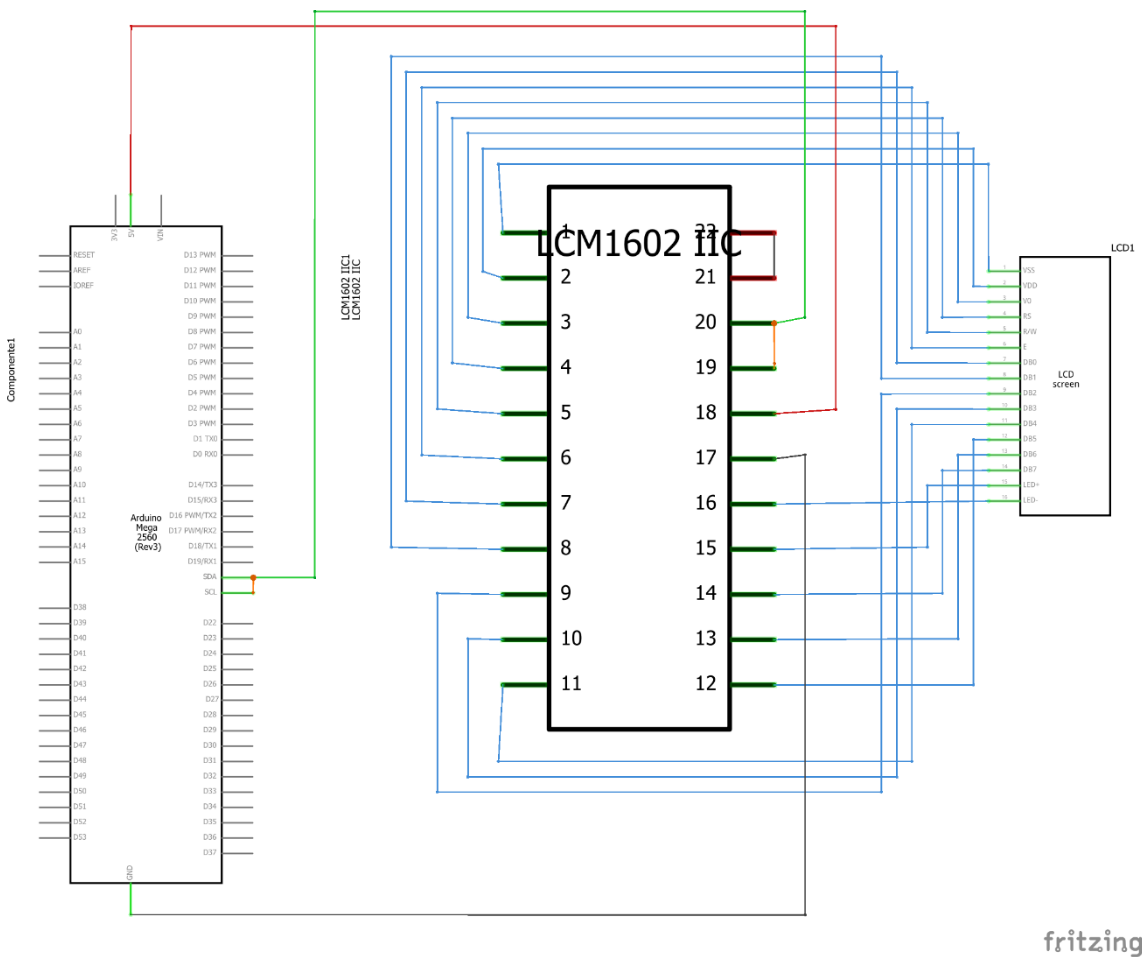


Ilustración 2.7:Esquema electrónico (opción 2)

2.3 ARDUINO MEGA

Arduino es una plataforma de desarrollo basada en una placa electrónica de hardware libre que incorpora un microcontrolador programable, en este caso Arduino Mega 2560 R3 utiliza el modelo Atmega2560 y una serie de pines de entradas y salidas (analógicas y digitales). Estos pines permiten establecer conexiones entre el microcontrolador y los diferentes sensores y actuadores de una manera muy sencilla. Se conecta al ordenador por un cable USB, lo que permite cargar el programa diseñado. El entorno de programación se puede conseguir de forma gratuita en su página Web, el cual nos permite escribir, depurar, editar y grabar los programas o sketches. También, se puede conectar a una fuente de alimentación externa.[5][6] [7]

Como se comentó anteriormente, Arduino tiene software y hardware libre. En el caso del software esto quiere decir que proporciona una serie de libertades como[7]:

- Libertad de usarse para cualquier propósito y en cualquier sistema informático.
- Acceso libre a la fuente del código para estudiar su funcionamiento y adaptarlo a las necesidades de cada usuario.
- Libertad de distribuir copias
- Tener la libertad de mejorarlo y distribuir las copias de estas.

Y en el del hardware, significa que se pueden conseguir los ficheros esquemáticos de las placas de Arduino. Con estos se puede construir cada usuario su propia placa “a mano”. Aunque lo habitual es comprarla al fabricante. Al ser libre se puede estudiar su funcionamiento, modificarlo, mejorarlo y compartir los cambios.[7]

Características Arduino Mega 2560:

- 54 pines entradas y salidas digitales (14 de ellos salidas PWM)
- 16 entradas analógicas
- 4 puertos serie de hardware
- Oscilador de cristal de 16 Mhz
- Conexión USB
- Conectar de alimentación
- ICSP (programación en serie en circuito)
- Botón de reinicio
- 256 KB de memoria flash para almacenar código (8 KB reservados para el arranque)
- 8 KB de SRAM (memoria estática de acceso aleatorio)[8] [9]
- 4 KB de EEPROM (ROM programable y borrable eléctricamente)[10]

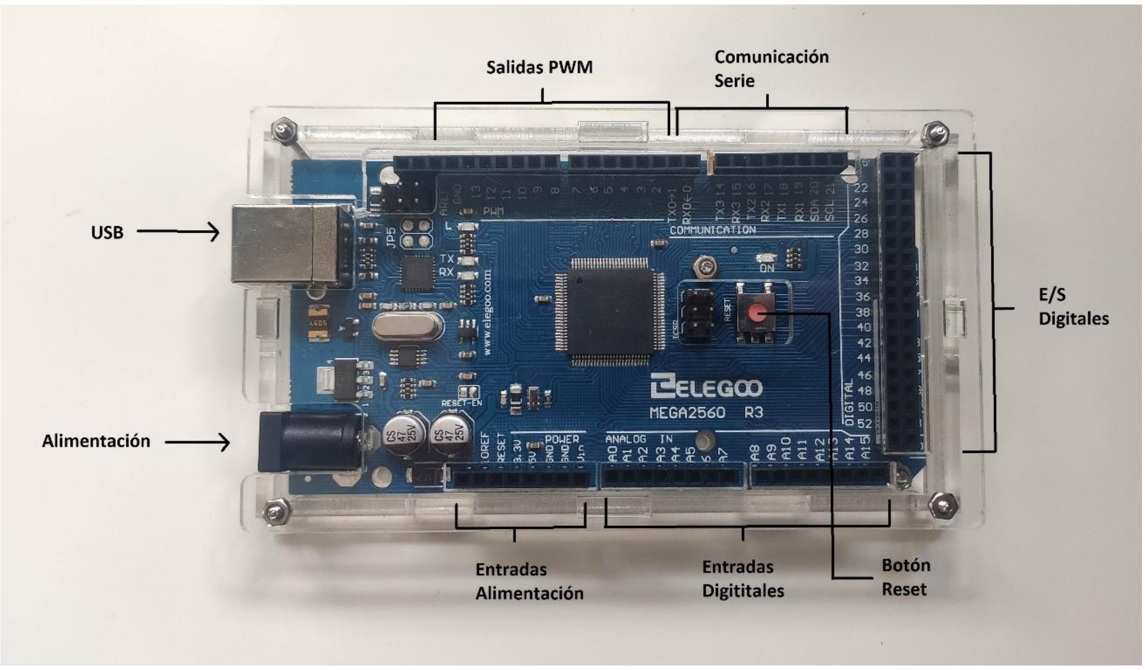


Ilustración 2.8: Arduino Mega 2560 R3

2.4 AMG8833

El sensor AMG8833 es una matriz de sensores térmicos IR de 8 x 8, es decir, nos proporciona un total de 64 medidas individuales. Nos permite medir temperaturas de 0ºC a 80ºC. La comunicación se realiza a través de I2C y envía los datos a un Arduino o Raspberry, para mostrar una matriz de 64 lecturas térmicas infrarrojas individuales.[11]

La precisión de lectura es de +- 2.5 ºC. detecta a una persona a una distancia de hasta 7 metros, su frecuencia máxima es de 10 Hz. Su ángulo de visión es de 60º, con una resolución de 0,25ºC y se alimenta con 3,3 V.[11]

Para realizar la conexión con Raspberry solo se necesita: alimetación (3,3 V), GND, SCL y SDA

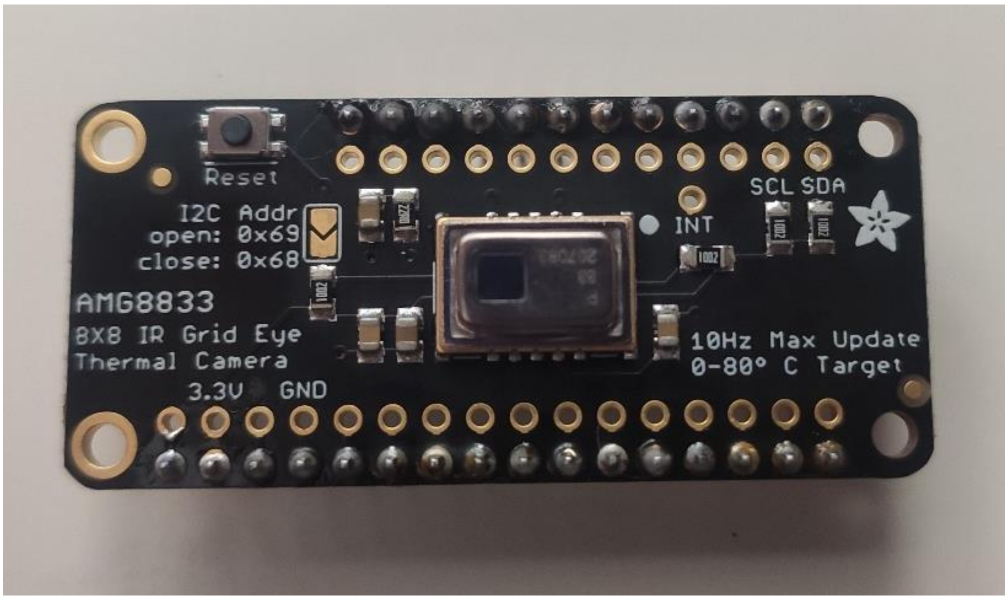


Ilustración 2.9: Sensor AMG8833

2.5 RASPBERRY

La Raspberry es una placa de ordenador pequeña y de bajo coste desarrollada por Raspberry Pi Foundation. Utiliza un software de código abierto, como Raspberry PI OS (anteriormente llamado Raspbian), Linux, Ubuntu…[12]

Para utilizarla solo hace falta el sistema operativo en la tarjeta micro SD, un teclado y ratón. Además de usarla como ordenador, se puede usar para realizar proyectos de domótica, videoconsolas retro, Smart TV, robótica…[12]

Gracias a la incorporación de una serie de pines que posee la placa se puede realizar la comunicación con un gran número de sensores, dando lugar a las aplicaciones comentadas anteriormente. [12]

En este caso la placa de Raspberry pi 3 model B, tiene las siguientes características[13]:

- CPU de cuatro núcleos a 1,2 GHz Broadcom BCM2837 de 64 bits
- 1 GB de RAM
- BCM43438 LAN inalámbrica y Bluetooth de baja energía (BLE) a bordo
- 100 Ethernet base
- GPIO extendido de 40 pines
- 4 puertos USB 2
- Salida estéreo de 4 polos y puerto de video compuesto
- HDMI de tamaño completo

- Puerto de cámara CSI para conectar una cámara Raspberry Pi
- Puerto de pantalla DSI para conectar una pantalla táctil Raspberry Pi
- Puerto micro SD para cargar su sistema operativo y almacenar datos
- Fuente de alimentación micro USB conmutada mejorada hasta 2.5A

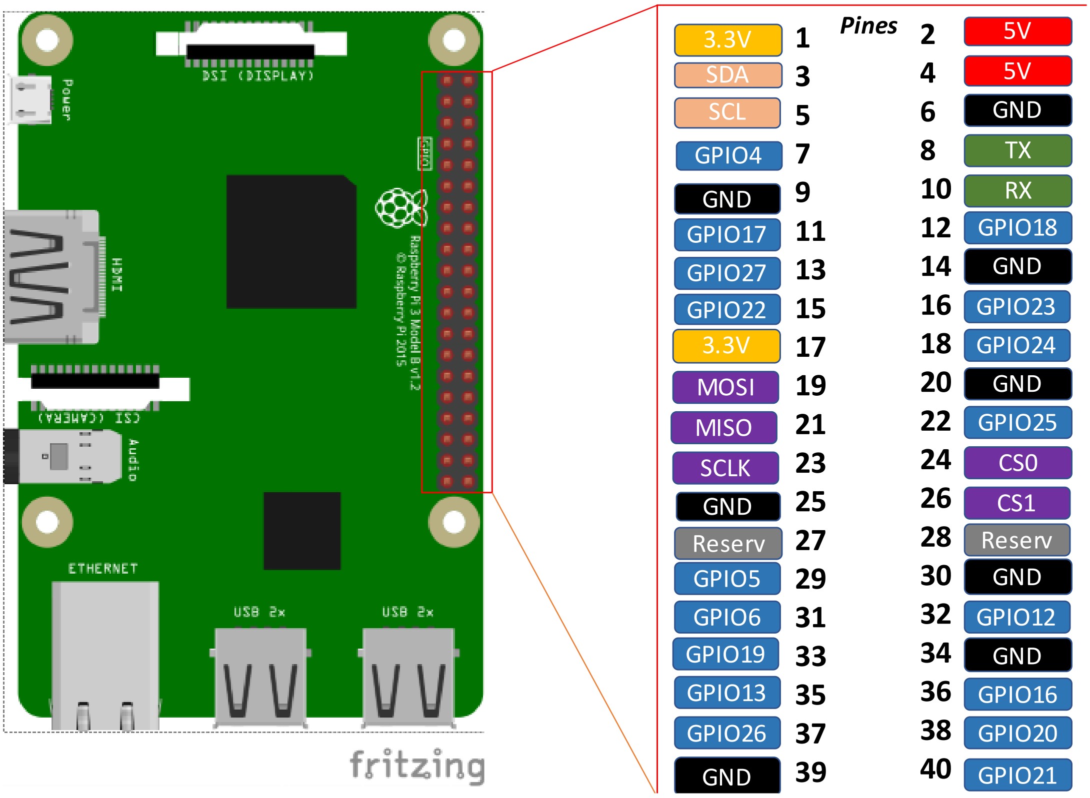


Ilustración 2.10: Pines Raspberry Pi 3 model B

3 DISEÑO E IMPLEMENTACIÓN

El proyecto está dividido en dos partes, la primera un termómetro sin contacto y la segunda una cámara térmica.

Cada una de las partes del proyecto tiene su función. En el caso de la cámara térmica puede realizar un cribado de seguridad en embarques de manera rápida y eficaz, con el fin de evitar contagios. El termómetro sería el dispositivo con el que de confirmar o desmentir un potencial peligro. Cuando la alarma suena en el caso de la cámara podría ser un posible foco y con el termómetro se confirma de manera individual si la temperatura es elevada o no.

En este apartado del proyecto se explicará cómo se ha realizado cada parte.

3.1 CÁMARA TÉRMICA

En este caso, los materiales usados son una Raspberry pi 3 model b, un módulo AMG8833 y un led a modo de alarma visual.

Para poder utilizar Raspberry se instaló en una tarjeta micro SD el sistema operativo Raspberry PI OS que se puede encontrar en su página oficial[14]. A continuación, se debe de activar la interfaz I2C para que se pueda establecer la comunicación entre el módulo AMG8833 y Raspberry, se puede realizar usando la línea de comando o la configuración de Raspberry PI.

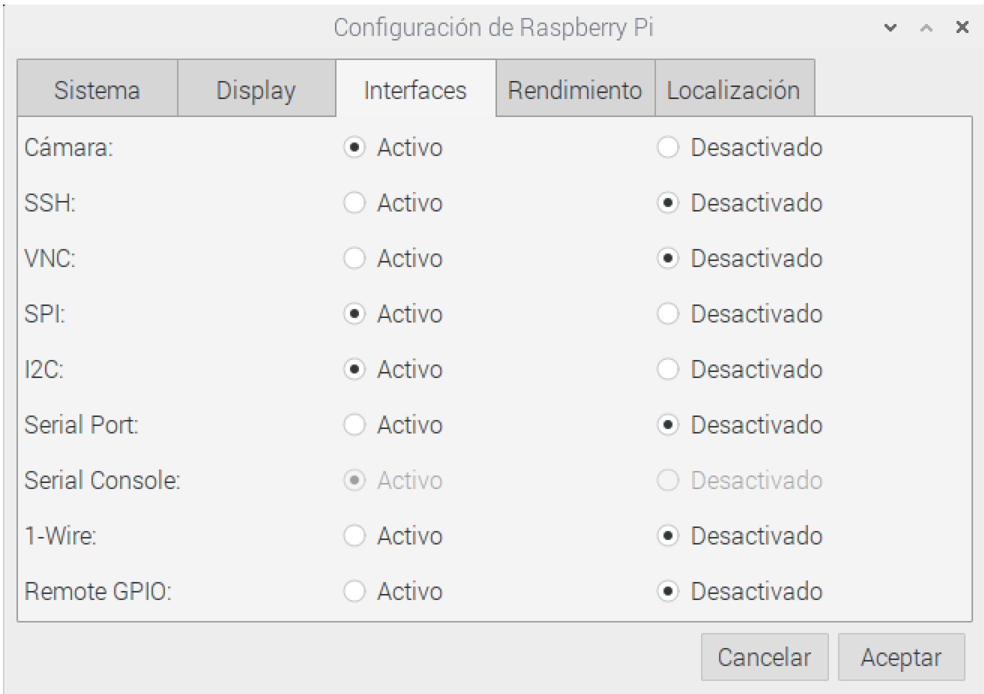


Ilustración 3.1: configuración Raspberry PI

Para activarlo por comando desde la terminal ejecutamos:

- sudo apt-get install -y Python-smbus
- sudo apt-get install -y i2c-tools

También, es recomendable previamente actualizar el software con sudo apt-get update

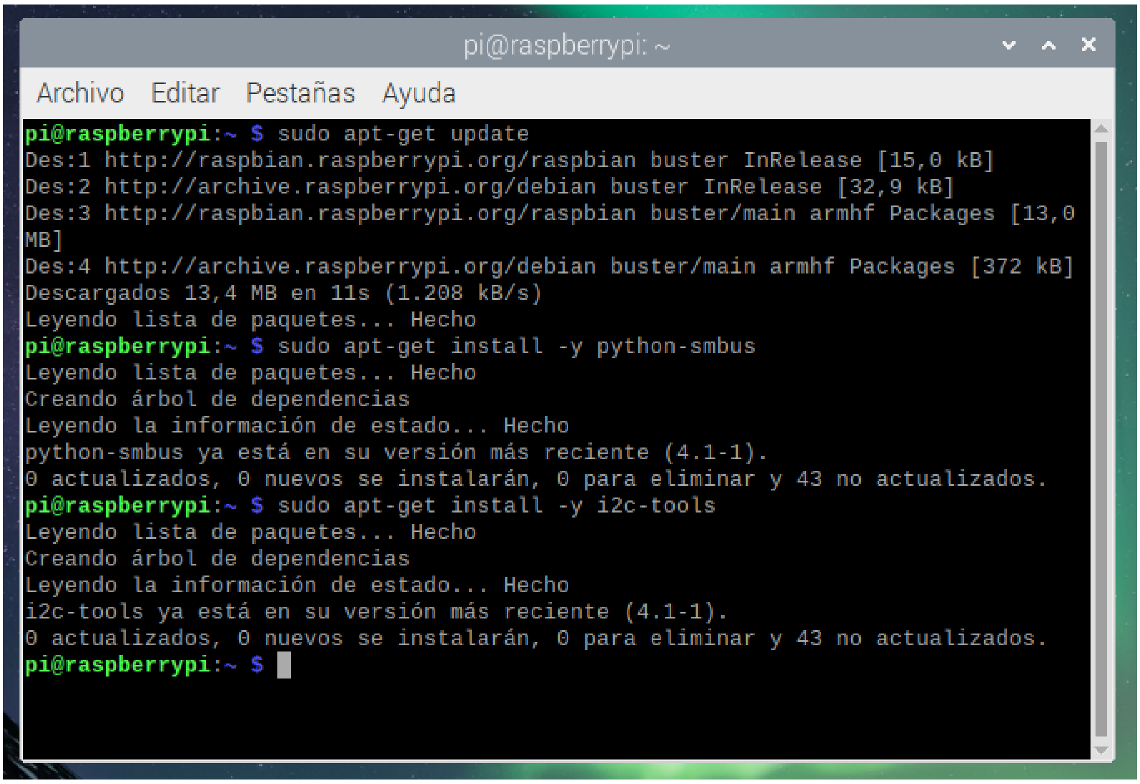


Ilustración 3.2:comandos para actualizar software en el terminal

Se conecta el sensor AMG8833 a la Raspberry. A continuación, se muestra imágenes de las conexiones.

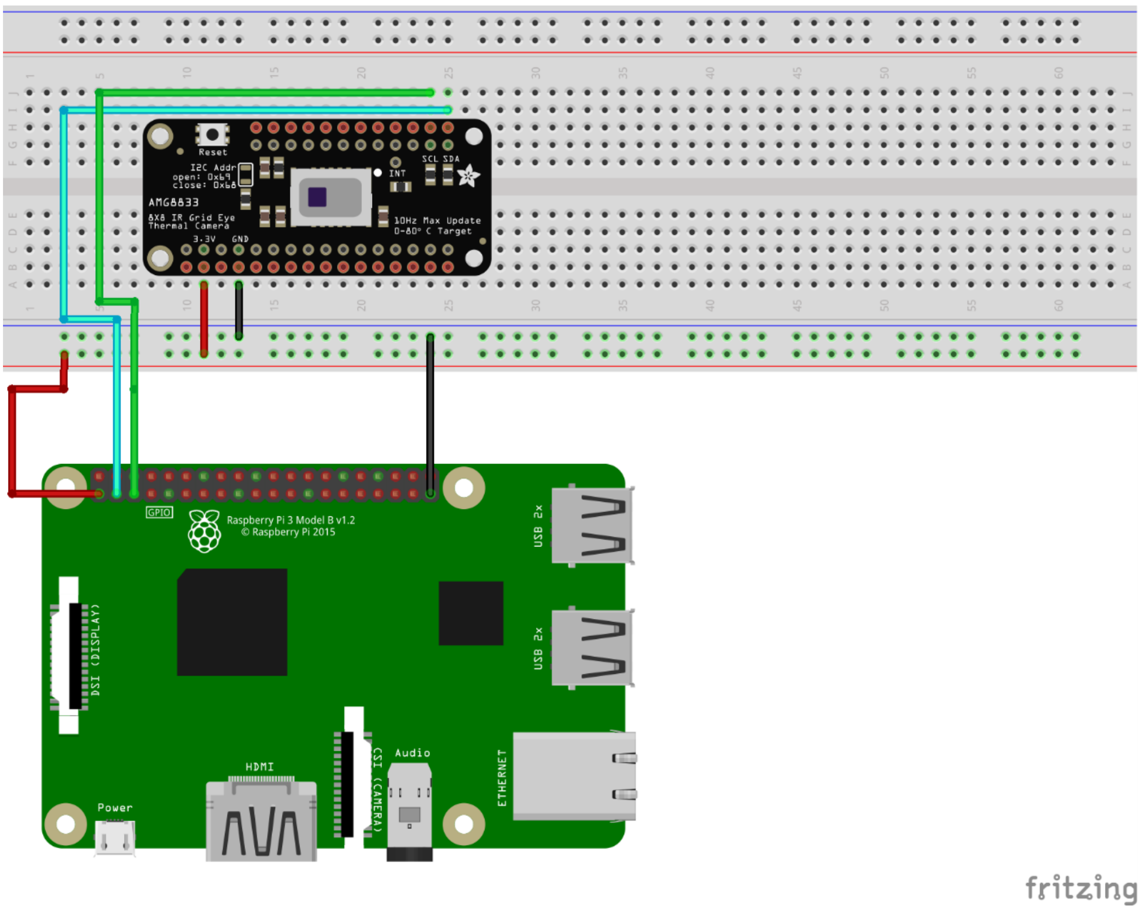


Ilustración 3.3:conexiones AMG8833

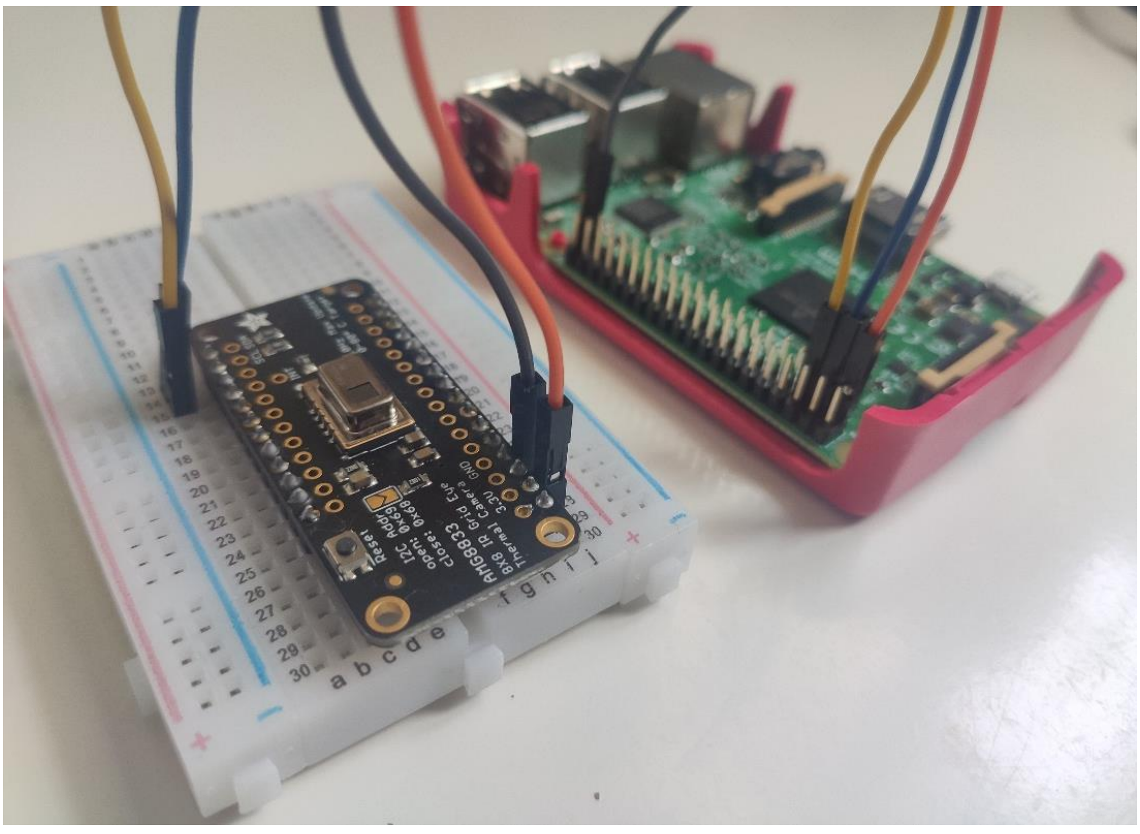


Ilustración 3.4:imagen conexión AMG8833

Para saber la dirección del dispositivo I2C, en la línea de comando se utiliza: sudo i2cdetect -y 1

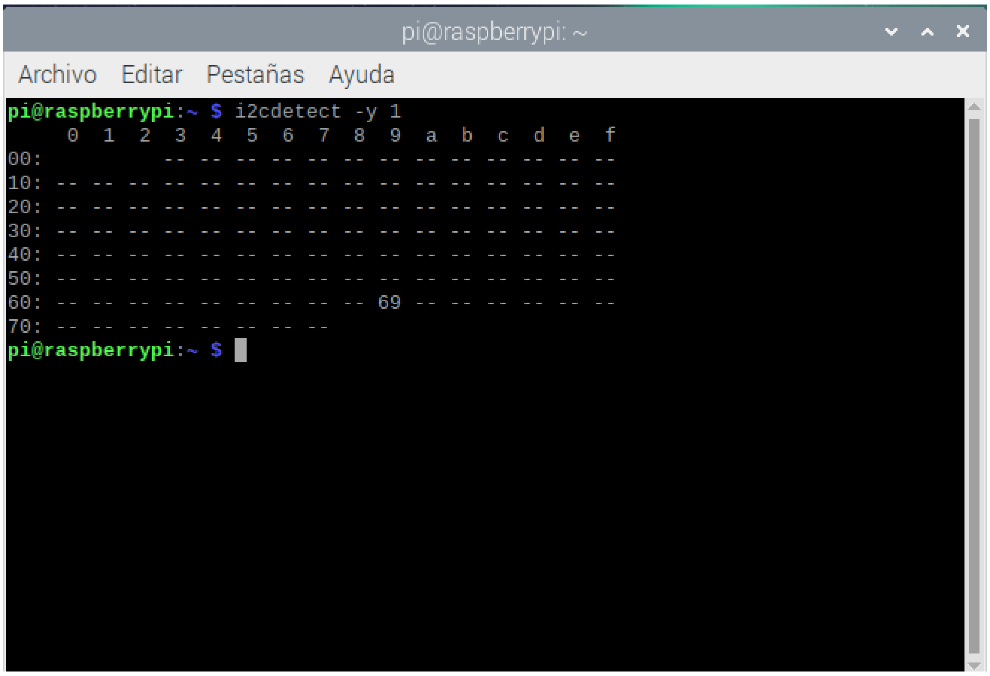


Ilustración 3.5:detección de dirección en la consola

Cuando se obtiene la dirección del dispositivo, se realiza una prueba de su funcionamiento con el código diseñado por Josh Hrisko que se estudiará posteriormente y se mostrará imágenes de las pruebas realizadas.[15]

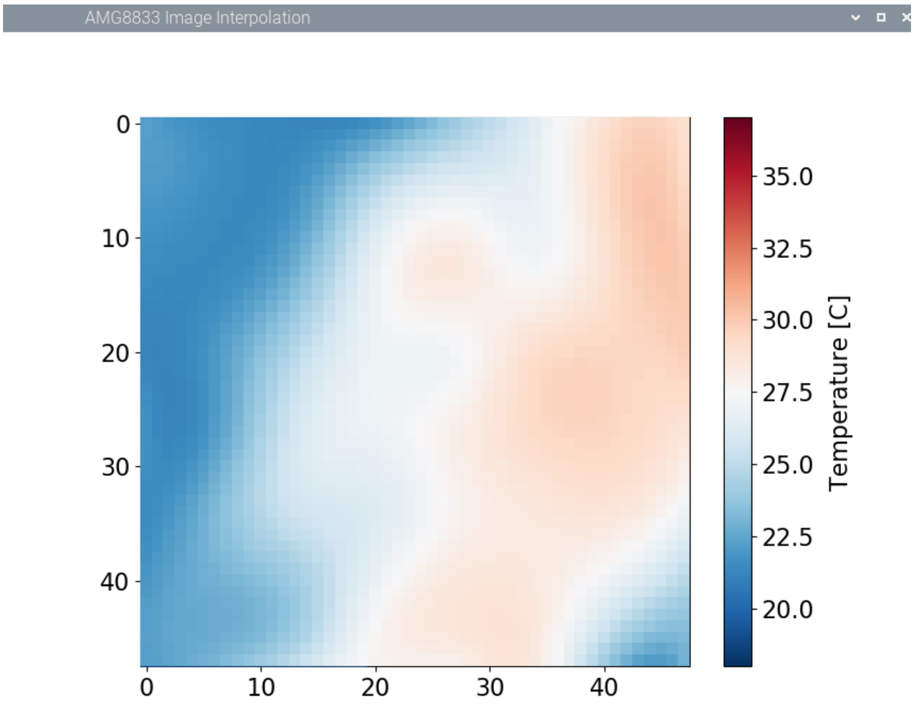


Ilustración 3.6:funcionamiento cámara térmica

Con esta prueba inicial básica, se ha comprobado el funcionamiento de la cámara térmica empleada. El siguiente paso, sería la modificación del proyecto base y conectar dos leds para realizar la alerta cuando se supere un valor que fijamos previamente. También, se podría utilizar una señal sonora como un zumbador o la combinación de ambos un led más un zumbador.

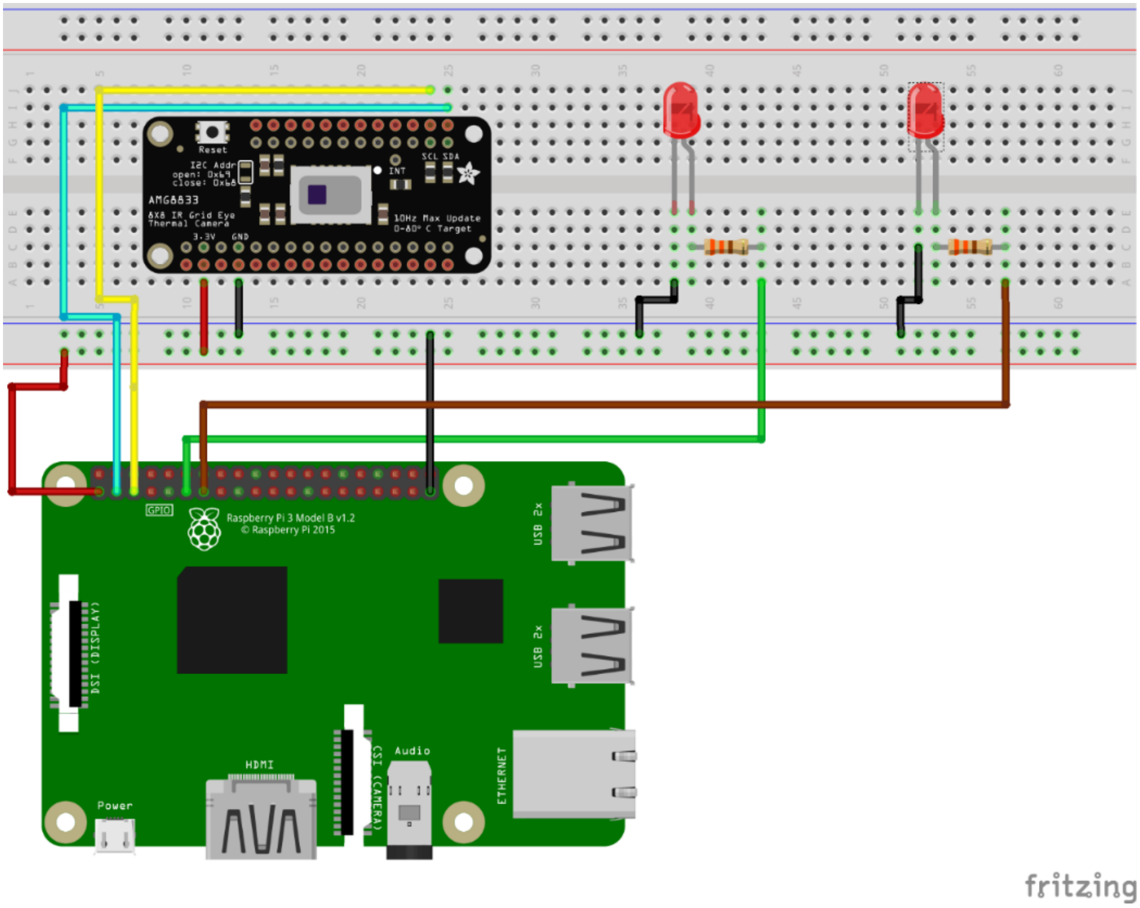


Ilustración 3.7:conexión AMG8833 y led

El sensor AMG8833 se conecta los pines SDA y SCL a los pines SDA y SCL de la raspberry. También, se alimenta con 3V y se conecta a tierra.

Los dos leds utilizados en este caso, se conecta la patilla negativa o cátodo a tierra y la positivo o ánodo a una resistencia de 330 Ω, a continuación, se conecta está a los pines 11 y 13 de la raspberry (GPIO 17 y 27)

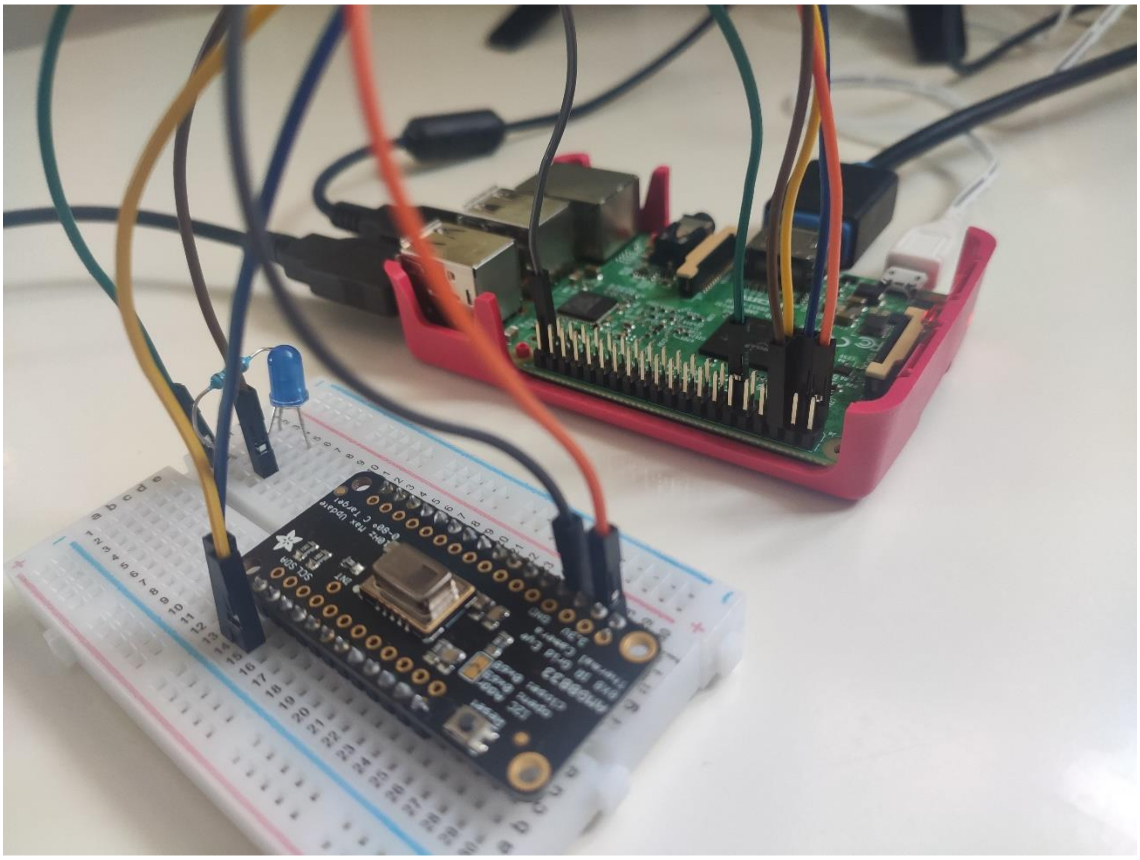


Ilustración 3.8:imagen de AMG8833 y Led

En GitHub se puede encontrar diferentes ejemplos para este sensor. Podemos obtener con alguno de ellos la visión de una cámara.

En este caso se usará de base el código escrito por Josh Hrisko, con el podemos ver en pantalla la imagen de la temperatura representada con un gradiente de colores que se encuentra al alcance del módulo AMG8833.[15]

Con esta base, se realizará una modificación de este código para poder añadir un led para alertar de posibles incidencias. La alerta que se emite será unos parpadeos del led, dependiendo de la temperatura, este tardará un tiempo en estar encendido y se encenderá un número de veces establecidas. Si la temperatura es de 37ºC, el led se encenderá 3 veces y tardará un segundo entre parpadeo. En el caso de que supere los 37 ºC se encenderá 5 veces, estará 2 segundos encendidos y tardará un segundo en volver a encenderse. El código final, se mostrará a continuación:

camara.py

```
# Librerías
import time,sys
sys.path.append('../')
# load AMG8833 module
import amg8833_i2c
import numpy as np
import matplotlib.pyplot as plt
from scipy import interpolate
from gpiozero import LED #importamos la clase led de la libreria gpiozero
from signal import pause #importamos la clase pause de la liberia signal

blue = LED(17)
yellow= LED (27)
#
#####################################
# Initialization of Sensor
#####################################
#
```

Ilustración 3.9: Código cámara térmica, parte 1

camara.py

```
#
t0 = time.time()
sensor = []
while (time.time()-t0)<1: # espere un segundo para que el sensor comience
    try:
        # AD0 = GND, addr = 0x68 | AD0 = 5V, addr = 0x69
        sensor = amg8833_i2c.AMG8833(addr=0x69) # empieza AMG8833
    except:
        sensor = amg8833_i2c.AMG8833(addr=0x68)
    finally:
        pass
time.sleep(0.1) # espera a que el sensor se asiente

# si no se encuentra ningun dispositivo, sale del script
if sensor==[]:
    print("No AMG8833 Found - Check Your Wiring")
    sys.exit(); # salir de la app si no se encuentra AMG8833
#
#####################################
```

Ilustración 3.10:Código cámara térmica, parte 2

camara.py

```
#####################################
# Propiedades de interpolación
#####################################
#
# resolución original
pix_res = (8,8) # resolución pixel
xx,yy = (np.linspace(0,pix_res[0],pix_res[0]),
                    np.linspace(0,pix_res[1],pix_res[1]))
zz = np.zeros(pix_res) # establecer matriz con ceros primero
# nueva resolución
pix_mult = 6 # multiplicar la interpolación
interp_res = (int(pix_mult*pix_res[0]),int(pix_mult*pix_res[1]))
grid_x,grid_y = (np.linspace(0,pix_res[0],interp_res[0]),
                            np.linspace(0,pix_res[1],interp_res[1]))
# interpolar función
def interp(z_var):
    # interpolación cúbica en la imagen
    # a una resolución de (pix mult*8 x pix mult*8)
```

Ilustración 3.11:Código cámara térmica, parte 3 camara.py

```python
# interpolar función
def interp(z_var):
    # interpolación cúbica en la imagen
    # a una resolución de (pix_mult*8 x pix_mult*8)
    f = interpolate.interp2d(xx,yy,z_var,kind='cubic')
    return f(grid_x,grid_y)
grid_z = interp(zz) # imagen interpolada
#
#####################################
# Figura de inicio y formato
#####################################
#
plt.rcParams.update({'font.size':16})
fig_dims = (10,9) # tamaño figura
fig,ax = plt.subplots(figsize=fig_dims) # comienzo figura
fig.canvas.set_window_title('AMG8833 Image Interpolation')
im1 = ax.imshow(grid_z,vmin=18,vmax=40,cmap=plt.cm.RdBu_r) # trazar imagen con límites de temperatura
cbar = fig.colorbar(im1,fraction=0.0475,pad=0.03) # barra de color
```

Ilustración 3.12:Código cámara térmica, parte 4

camara.py

```python
#####################################
# Figura de inicio y formato
#####################################
#
plt.rcParams.update({'font.size':16})
fig_dims = (10,9) # tamaño figura
fig,ax = plt.subplots(figsize=fig_dims) # comienzo figura
fig.canvas.set_window_title('AMG8833 Image Interpolation')
im1 = ax.imshow(grid_z,vmin=18,vmax=40,cmap=plt.cm.RdBu_r) # trazar imagen con límites de temperatura
cbar = fig.colorbar(im1,fraction=0.0475,pad=0.03) # barra de color
cbar.set_label('Temperature [C]',labelpad=10) # etiqueta de temperatura
fig.canvas.draw() # dibujar figura

ax_bgnd = fig.canvas.copy_from_bbox(ax.bbox) # fondo
fig.show() # mostrar figura
#
#####################################
# Trazar temperaturas de AMG88833 en tiempo real
```

Ilustración 3.13:Código cámara térmica, parte 5

camara.py

```python
#
#####################################
# Trazar temperaturas de AMG88833 en tiempo real
#####################################
#
pix_to_read = 64 # leer los 64 pixeles
while True:
    status,pixels = sensor.read_temp(pix_to_read) # leer pixeles con estado
    if status: # si hay un error en el pixel, vuelve a entrar en el bucle y vuelve  intentarlo
        continue

    T_thermistor = sensor.read_thermistor() # leer temperatura del termistor
    fig.canvas.restore_region(ax_bgnd) # restaurar fondo
    new_z = interp(np.reshape(pixels,pix_res)) # imagen interpolada
    im1.set_data(new_z) # actualizar el gráfico con nuevas temperaturas intepoladas
    ax.draw_artist(im1) # dibujar imagen de nuevo
    fig.canvas.blit(ax.bbox) # blitting
    fig.canvas.flush_events() # para el gráfico en tiempo real
```

Ilustración 3.14:Código cámara térmica, parte 6

camara.py

```python
    fig.canvas.blit(ax.bbox) # blitting
    fig.canvas.flush_events() # para el gráfico en tiempo real



    for i in range(0,64,1):

        if  37 <= pixels[i] <38: #cuando llega a 37 va a parpadear

            yellow.blink(1,1,3) # parpadeará 3 veces y tardará un segundo entre parpadeos

        if  pixels[i] >= 38: #cuando es igual o mayor a 38

            blue.blink(1,2,5) #pardaeará 5 veces y tardará 1 segundo entre apagado y encendido
                             #2 segundos entre encendido y apagado
```

Ilustración 3.15:Código cámara térmica, parte 7

3.2 TERMÓMETRO

Para realizar el termómetro, se ha usado una placa Arduino mega 2560 R3 junto con el sensor MLX90614, un led, una pantalla LCD y un módulo de comunicación I2C.

Para saber con certeza la dirección en la que esta el sensor MLX90614 se ha cambiado a 0x50, utilizando el siguiente código.[16]

```
direccionMLX §
#include <i2cmaster.h>

byte MLXAddr = 0x50<<1;   // Nueva dirección
//byte MLXAddr = 0;       // Dirección universal

void setup(){
 Serial.begin(9600);
 Serial.println("Iniciando...");

 i2c_init();                              //Inicializar i2c bus
 PORTC = (1 << PORTC4) | (1 << PORTC5);   //Pullups

 delay(5000);                    // Espera la conexión serial
 cambiarDirec(0x50, 0x00);       // Cambia la dirección al nuevo valor, en este caso cambia a 0x50
}

void loop(){
}

//Función con comentarios originales de http://forum.arduino.cc
word cambiarDirec(byte NuevaDirec1, byte NuevaDirec2) {

 Serial.println("> Cambiar dirección");

 i2c_start_wait(0 + I2C_WRITE);    //enviar condición de inicio y escirbir bit
 i2c_write(0x2E);                  //enviar comando para que el dsipositivo devuelva la dirección
 i2c_write(0x00);                  // enviar byte zero bajo para borrarlo
 i2c_write(0x00);                  //enviar byte zero alto para borrarlo
 if (i2c_write(0x6F) == 0) {
   i2c_stop();                     //Liberar bus y finalizar transacción
   Serial.println("  Dato borrado");
 }
 else {
   i2c_stop();                     //Liberar bus y finalizar transacción
   Serial.println("  Fallo borrando dato");
```

Ilustración 3.16: Código para establecer dirección sensor MLX90614, parte 1

```
direccionMLX §
  i2c_write(0x2E);                  //enviar comando para que el dsipositivo devuelva la dirección
  i2c_write(0x00);                  // enviar byte zero bajo para borrarlo
  i2c_write(0x00);                  //enviar byte zero alto para borrarlo
  if (i2c_write(0x6F) == 0) {
    i2c_stop();                     //Liberar bus y finalizar transacción
    Serial.println("  Dato borrado");
  }
  else {
    i2c_stop();                     //Liberar bus y finalizar transacción
    Serial.println("  Fallo borrando dato");
    return -1;
  }

  Serial.print("  Dato escrito: ");
  Serial.print(NuevaDirec1, HEX);
  Serial.print(", ");
  Serial.println(NuevaDirec2, HEX);

  for (int a = 0; a != 256; a++) {
    i2c_start_wait(0 + I2C_WRITE);  // enviar condición de inicio y escirbir bit
    i2c_write(0x2E);                // enviar comando para que el dsipositivo devuelva la dirección
    i2c_write(NuevaDirec1);            // enviar byte zero bajo para borrarlor
    i2c_write(NuevaDirec2);            // enviar byte zero alto para borrarlo
    if (i2c_write(a) == 0) {
      i2c_stop();                   //Liberar bus y finalizar transacción
      delay(100);                   // esperar 10 ms
      Serial.print("CRC correcto: 0x");
      Serial.println(a, HEX);
      return a;
    }
  }
  i2c_stop();                       // Liberar bus y finalizar transacción
  Serial.println("CRC Correcto no encontrado");
  return -1;
}
```

Ilustración 3.17: Código para establecer dirección sensor MLX90614, parte 2

Activando el monitor serie se puede comprobar que la dirección es la establecida:

```
COM7

Iniciando...
> Cambiar dirección
   Dato borrado
   Dato escrito: 50, 0
CRC correcto: 0x63
```

Ilustración 3.18:Dirección establecida del sensor MLX90614

Cuando se ha establecido la dirección, se comprueba que con el programa diseñado por Jaime Patarroyo muestra las temperaturas en el monitor serie[16]. Sería el que siguiente:

temperatura §

```
#include <i2cmaster.h>

int deviceAddress = 0x50<<1;     //dirección comunicación I2c

float celcius = 0;               // Variable que contiene la temperatura en Celcius.


void setup() {
  Serial.begin(9600);            // Inicia la comunicación serial a 9600bps.

  i2c_init();                                // Inicia el bus i2c.
  PORTC = (1 << PORTC4) | (1 << PORTC5);     // Habilita 'pullups'.
}

void loop() {
  celcius = temperatureCelcius(deviceAddress);  // Lee los datos del MLX90614
                                                // con la dirección dada,
                                                // los transforma en la
                                                // temperatura en Celcius y
                                                // la almacena en la variable
                                                // celcius.


  Serial.print("Celcius: ");           // Imprime ambas lecturas en el
  Serial.println(celcius);             // puerto serial.
  Serial.println();

  delay(1000);                         // Espera un segundo antes de imprimir de nuevo.
}

float temperatureCelcius(int address) { //Cambió de variable numerica tipo entera a decimal
  int dev = address;
  int data_low = 0;
  int data_high = 0;
  int pec = 0;
```

Ilustración 3.19:Código de Jaime Patarroyo, parte 1

temperatura §

```
  delay(1000);                          // Espera un segundo antes de imprimir de nuevo.
}

float temperatureCelcius(int address) { //Cambió de variable numerica tipo entera a decimal
  int dev = address;
  int data_low = 0;
  int data_high = 0;
  int pec = 0;

  // Escribe
  i2c_start_wait(dev+I2C_WRITE);
  i2c_write(0x07);

  // Lee
  i2c_rep_start(dev+I2C_READ);
  data_low = i2c_readAck();       // Lee 1 byte y envía ack.
  data_high = i2c_readAck();      // Lee 1 byte y envía ack
  pec = i2c_readNak();
  i2c_stop();

  // Esto convierte los bytes altos y bajos juntos y procesa la temperatura.
  double tempFactor = 0.02;       // 0.02 grados por LSB (medida de
                                  // resolución del MLX90614).
  double tempData = 0x0000;
  int frac;                       // Datos después del punto decimal.

  // Esto oculta el error del byte alto y lo mueve a la izquierda
  // 8 bits y agrega el byte bajo.
  tempData = (double)(((data_high & 0x007F) << 8) + data_low);
  tempData = (tempData * tempFactor)-0.01;
  float celcius = tempData - 273.15;

  // Retorna la temperatura en Celcius.
  return celcius;
}
```

Ilustración 3.20:Código de Jaime Patarroyo, parte 2

En la siguiente imagen, se muestra algunas de las temperaturas registradas por el sensor:

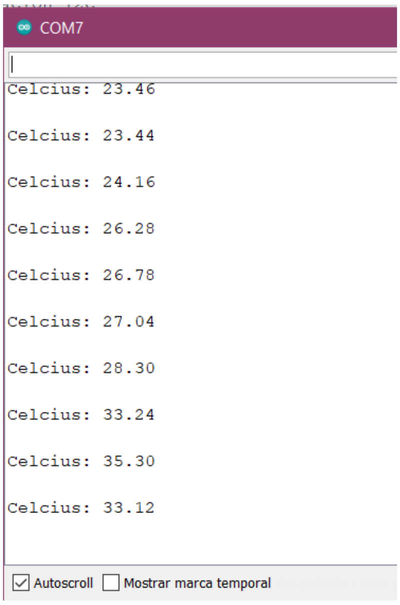


Ilustración 3.21: Registros de temperatura

En GitHub se puede encontrar las diferentes librerías usadas para realizar la programación. Las que se han usado han sido:

- LiquidCrystal_I2C.h, librería que permite controlar las pantallas por I2C [17]
- i2cmaster.h [18]
- Wire.h, librería que permite comunicarse con dispositivos I2C [19]

El programa está basando en el diseñado por Jaime Patarroyo. El cual puede medir la temperatura con el sensor y lo muestra en el monitor serie del programa de Arduino IDE.[16]

A partir de este programa original, se modifica para conseguir que la temperatura se muestre en una pantalla LCD y que cuando supere la temperatura de 37 ºC se encienda un led. Cuando este se activa, avisa de una incidencia.

A continuación, se muestra unas imágenes del código con las modificaciones realizadas para adaptarlo al proyecto planteado:

termometro

```
#include <LiquidCrystal_I2C.h>  // Libreria que permite controlar pantallas
#include <i2cmaster.h>          // Libreria para comunicación I2C
#include <Wire.h>               // Libreria que permite comunicarse con dispositivos I2C

LiquidCrystal_I2C lcd(0x27, 16, 2); // configuración de pines de la pantalla LCD

const int led=8;                //varible del Led

int deviceAddress = 0x50<<1;    //  0x50 es la dirección de comunicación de I2C

float celcius = 0;              // Variable que contiene la temperatura en Celcius.

void setup() {
  lcd.begin();
  Serial.begin(9600);           // Inicia la comunicación serial a 9600bps.
  pinMode(led,OUTPUT);

  i2c_init();                               // Inicia el bus i2c.
  PORTC = (1 << PORTC4) | (1 << PORTC5);    // Habilita 'pullups'.

}


float temperatureCelcius(int address) {
  int dev = address;
  int data_low = 0;
  int data_high = 0;
```

Ilustración 3.22: Código termómetro, parte 1

termometro

```
  // Escribe
  i2c_start_wait(dev+I2C_WRITE);
  i2c_write(0x07);

  // Lee
  i2c_rep_start(dev+I2C_READ);
  data_low = i2c_readAck();       // Lee 1 byte y envía ack.
  data_high = i2c_readAck();      // Lee 1 byte y envía ack
  pec = i2c_readNak();
  i2c_stop();

  // Esto convierte los bytes altos y bajos juntos y procesa la temperatura.
  double tempFactor = 0.02;       // 0.02 grados por LSB (medida de
                                  // resolución del MLX90614).
  double tempData = 0x0000;
  int frac;                       // Datos después del punto decimal.

  // Esto oculta el error del byte alto y lo mueve a la izquierda
  // 8 bits y agrega el byte bajo.
  tempData = (double)(((data_high & 0x007F) << 8) + data_low);
  tempData = (tempData * tempFactor)-0.01;
  float celcius = tempData - 273.15;

// Datos mostrados en pantalla
  lcd.backlight();
  lcd.clear();
  lcd.print("temperatura");
```

Ilustración 3.23:Código termómetro, parte 2 termometro

```
// Datos mostrados en pantalla
  lcd.backlight();
  lcd.clear();
  lcd.print("temperatura");
  lcd.setCursor ( 0, 1 );
  lcd.print(celcius);
  lcd.setCursor ( 1,3 );

// Condición para alerta usando led
  if(celcius<37)//Si la temperatura supera los 30°C se activa el led
  {digitalWrite(led, HIGH);}
  else {digitalWrite(led, LOW);}//Si no la supera no se activa


 // Retorna la temperatura en Celcius.
  return celcius;
}

void loop() {

  celcius = temperatureCelcius(deviceAddress);  // Lee los datos del MLX90614
                                                // con la dirección dada,
                                                // los transforma en la
                                                // temperatura en Celcius y
                                                // la almacena en la variable
                                                // celcius.
```

Ilustración 3.24:Código termómetro, parte 3

termometro

```
  if(celcius<37)//Si la temperatura supera los 30°C se activa el led
  {digitalWrite(led, HIGH);}
  else {digitalWrite(led, LOW);}//Si no la supera no se activa


 // Retorna la temperatura en Celcius.
  return celcius;
}

void loop() {

  celcius = temperatureCelcius(deviceAddress);  // Lee los datos del MLX90614
                                                // con la dirección dada,
                                                // los transforma en la
                                                // temperatura en Celcius y
                                                // la almacena en la variable
                                                // celcius.



  Serial.print("Celcius: ");             // Imprime ambas lecturas en el
  Serial.println(celcius);               // puerto serial.
  Serial.println();


  delay(1000);                           // Espera un segundo antes de imprimir de nuevo.
}
```

Ilustración 3.25:Código termómetro, parte 4

Las conexiones que se han realizado son las siguientes que se muestran en la imagen:

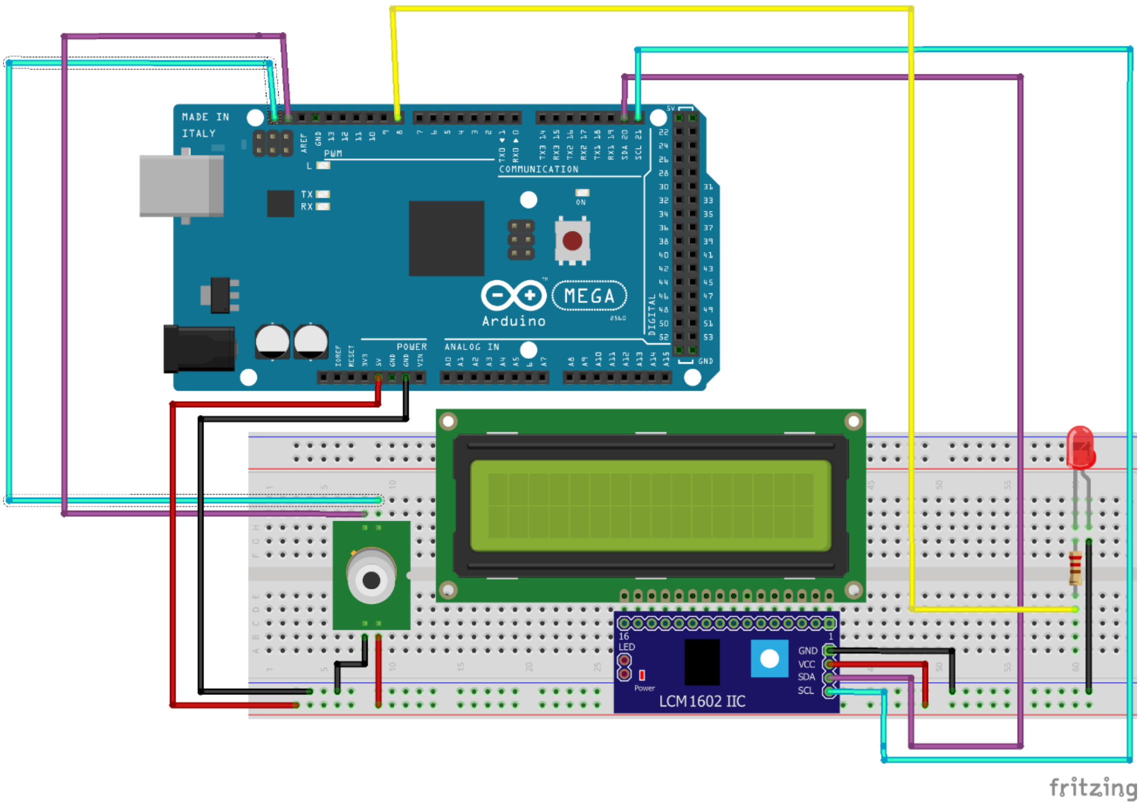


Ilustración 3.26: conexiones termómetro

La pantalla se conecta con ayuda del módulo de comunicaciones I2C a 5V, GND y los dos pines de comunicaciones SDA y SCL a las interrupciones 3 y 2 del Arduino que corresponderían a los pines 20 y 21 respectivamente.

El sensor térmico se conecta a la alimentación, GND y los pines de comunicación SDA y SCL a las entradas PWM SCA y SCL del Arduino

El led que se usa en esta parte del proyecto lleva conectada una resistencia de 220 Ω a la patilla positiva y a la entrada PWM número 4, la patilla negativa se conecta a tierra.

En la siguiente imagen se mostrará la medida tomada con este dispositivo.

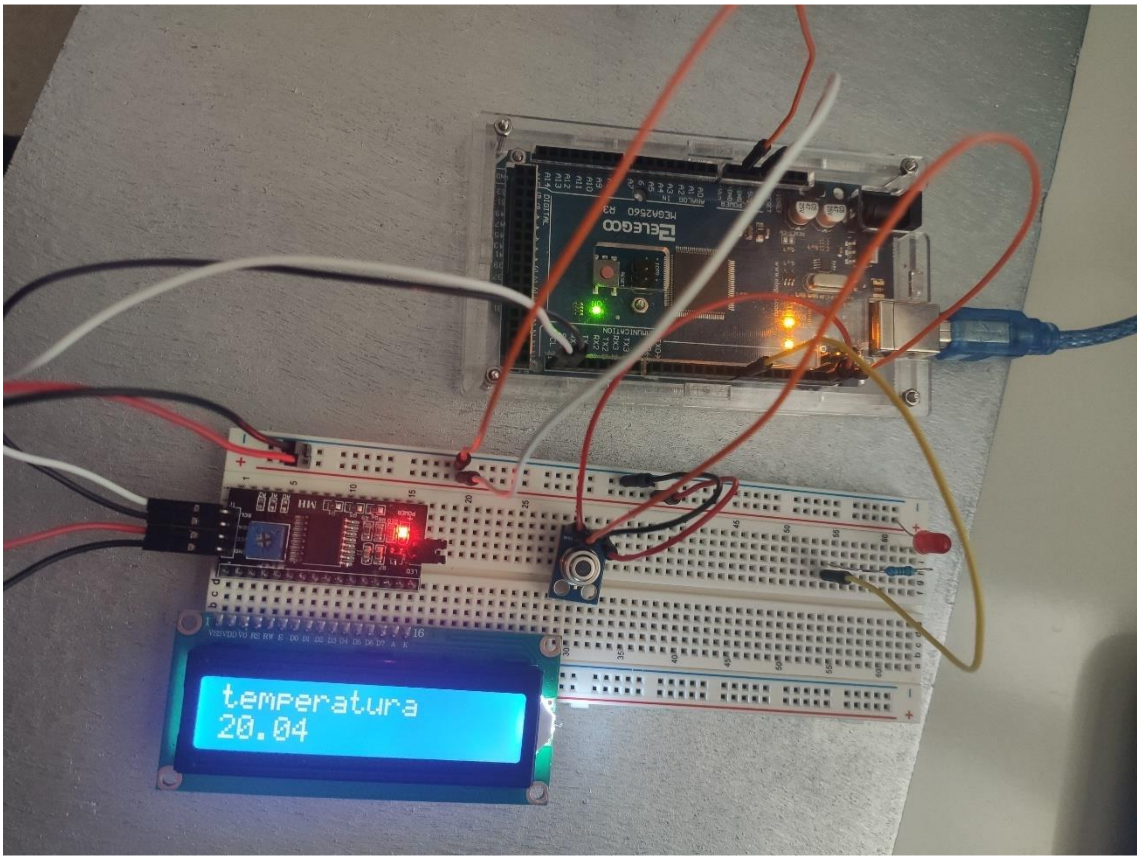


Ilustración 3.27:medida realizada con termómetro

4 LENGUAJES DE PROGRAMACIÓN

En este proyecto se han utilizado dos lenguajes de programación Python en el caso de Raspberry y C en Arduino.

Ilustración 4.1: lenguajes de programación utilizados en el proyecto, Python/C

Actualmente se puede encontrar una gran cantidad de programas en internet, los cuales se pueden utilizar o modificarlos para realizar tus proyectos propios. En este caso, se ha utilizado programas encontrados en GitHub, los cuales se han modificado con las nociones básicas obtenidas en el grado y con información de varias páginas de internet referenciadas en los diferentes apartados del texto.

Python es un lenguaje de programación cuya principal filosofía es que sea legible por cualquier persona con conocimientos básicos de programación. Además, posee una serie de características que lo hacen muy particular[20]:

- Es totalmente gratuito. Se trata de un lenguaje código abierto, por lo que no hay que pagar ninguna licencia para utilizarlo.
- Tiene una enorme comunidad. Su carácter gratuito hace que continuamente se estén desarrollando nuevas librerías y aplicaciones. Gracias a los foros, librerías y aplicaciones puedes resolver las dudas que puedan surgir.
- Es un lenguaje multiparadigma. Esto significa que combina propiedades de diferentes paradigmas de programación, lo que permite que sea muy flexible y fácil de aprender de manera independiente de los conocimientos del interesado.
- Python es apto para todas las plataformas. Podemos ejecutarlo en diferentes sistemas operativos como Windows, Linux o MacOS.[20]

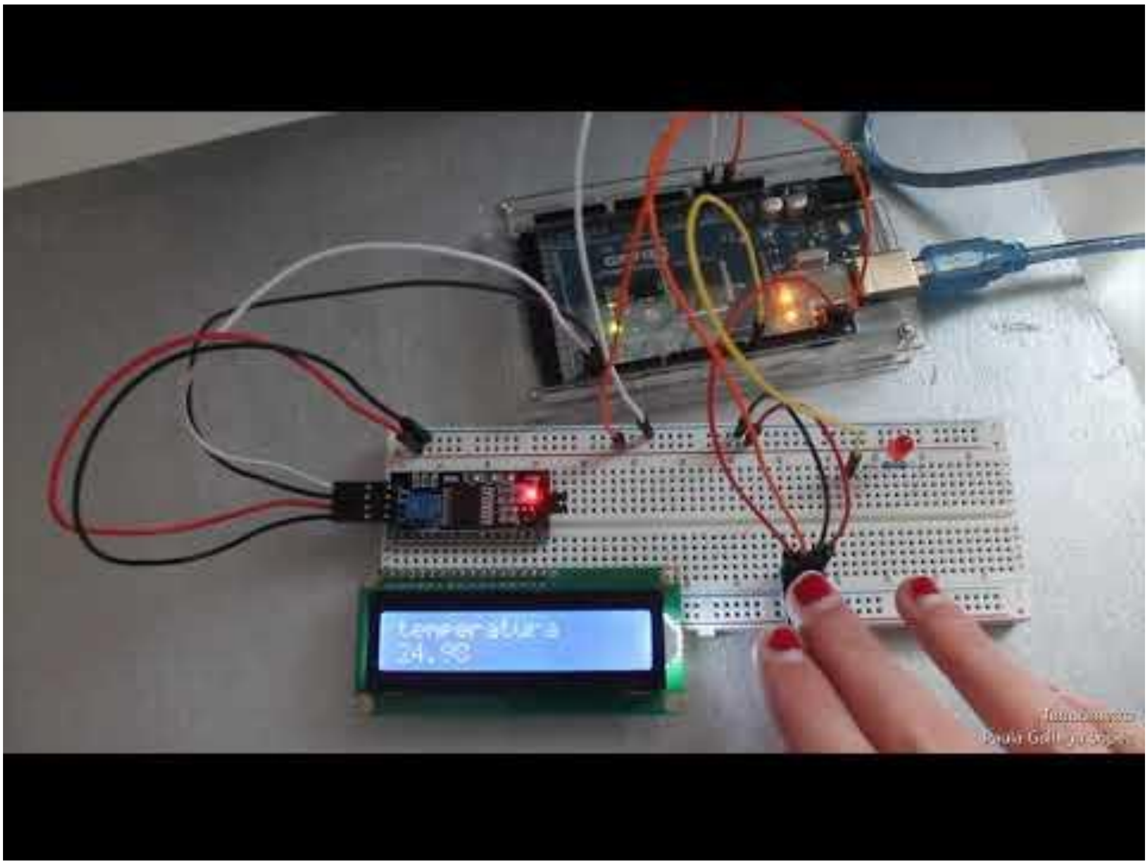
El lenguaje de programación C fue creado por Brian Kernighan y Dennis Ritchie a mediados de los años 70. C es el resultado de un proceso de desarrollo que comenzó con un lenguaje anterior, el BCPL, el cual influyó en el desarrollo por parte de Ken Thompson de un lenguaje llamado B, el cual es el antecedente directo del lenguaje C. Este proporciona una gran flexibilidad de programación y una muy baja comprobación de incorrecciones, de forma que el lenguaje deja bajo la responsabilidad del programador acciones que otros lenguajes realizan por si mismos.

Todo programa de C consta, básicamente, de un conjunto de funciones, y una función llamada main, la cual es la primera que se ejecuta al comenzar el programa, llamándose desde ella al resto de funciones que compongan nuestro programa.

El lenguaje C posee un número reducido de palabras reservadas que define el standard ANSI-C. Estas son algunas de ellas: auto, break, case, char, const, continue, default, do, double, else, float, for, if, int …. [21]

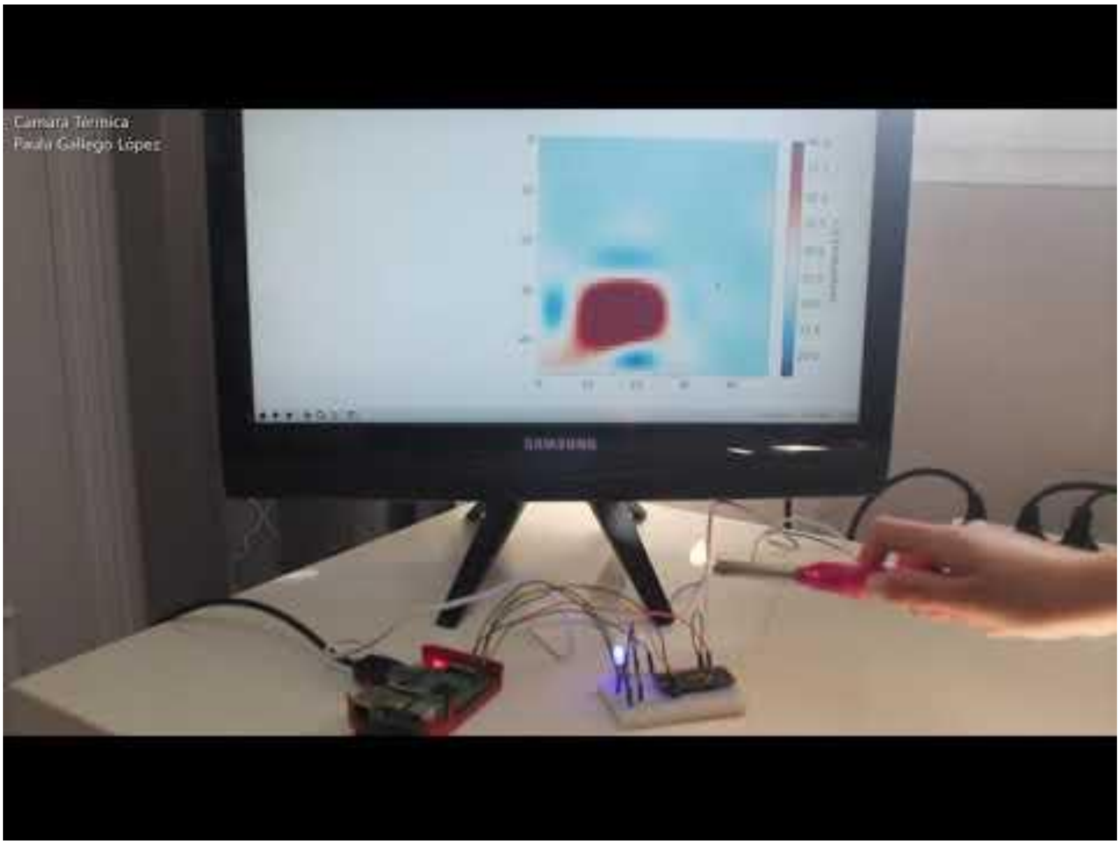
5 PRUEBAS

En este punto, se mostrarán algunos videos e imágenes del funcionamiento del proyecto. En primer lugar, se mostrará el termómetro y a continuación, la cámara térmica.



Vídeo 1: funcionamiento termómetro

En este video se puede observar que la temperatura que se muestra en pantalla varía según acerque el dedo. Al llegar a una temperatura de 28 ºC, que se ha establecido para realizar las pruebas, se puede ver como se enciende el led rojo que está en la placa protoboard.



Vídeo 2: funcionamiento cámara térmica

En este caso, se puede observar en la pantalla como la imagen se mueve a causa de lo que se encuentra en su rango de visión de sensor. Con ayuda de un mechero, se han encendido los dos leds que se ha usado para esta parte. Cuando la temperatura esta, en el rango de 37 ºC a 38 ºC (37ºC incluido), se enciende el led naranja 3 veces y cuando llega a 38 ºC se enciende el led azul parpadeando 5 veces.

6 MEJORAS

Este proyecto es posible mejorarlo para facilitar su utilización, a continuación, se comentarán algunas de las mejoras consideradas:

- Ambas partes del proyecto, tanto el termómetro como la cámara térmica no cuentan con una carcasa que proteja a los elementos electrónicos utilizados, así como para manejarlos con mayor facilidad. Con una impresora 3D se podría realizar estas carcasas y conseguir un proyecto visualmente más atractivo.
- En el apartado de la cámara térmica, se podría añadir una pantalla táctil para convertirlo en un equipo portátil. Una pantalla que se podría utilizar es la siguiente[22]:

Ilustración 6.1: pantalla táctil para Raspberry

- En la parte del termómetro, podríamos añadir un sensor de distancia y un altavoz. De modo que cuando la persona esté lo suficientemente cerca, avise de que está realizando la medida el sensor y dependiendo de la temperatura se encenderá la luz verde si está bien y la roja si no puede pasar.
- Mejorar la cámara térmica, seleccionando una cámara con una matriz mayor y con mayor definición. Esta opción sería más cara y se saldría del presupuesto del proyecto. Por ejemplo, el sensor MLX90640 de 32x24 pixeles que podemos encontrar tanto en Amazon y Aliexpress sobre 150 € [23], [24]. En tiendas de electrónica no se pueden encontrar ya que con la situación sanitaria que vivimos está agotado y el precio se ha elevado. Cuando se realizó la planificación del proyecto, los precios eran incluso más elevados y la disponibilidad menor.

- Para conseguir un sistema portátil, incorporar como una Power Bank como fuente de alimentación.
- Se podría cambiar a la última Raspberry pi más potente (Raspbery pi 4 model B). El precio de este modelo variará según la capacidad de RAM, ahora mismo los modelos existentes en el mercado son de 2, 4 y 8 GB. Los precios serían de 44,26 €; 61,5€ y 83,99€ respectivamente. [25]–[27]
- Otra opción, es establecer ciertas líneas de cribado y que los datos lleguen a una zona de vigilancia diferenciada y a distancia, donde se pueda monitorear el embarque. En esta se debería de incluir varios monitores según el número de líneas de cribado que se quiera establecer.

7 PRESUPUESTO

En este apartado, se realiza un desglose en forma de tabla del presupuesto del material empleado para realizar el proyecto, incluyendo cantidad, precio por unidad, proveedor y precio total. Los precios son orientativos ya que depende del proveedor del producto.

Los sensores escogidos para este proyecto cumplen con los requisitos planteados originalmente y, además, son de bajo coste comparándolos con otros dispositivos.

Material	Cantidad	Coste por unidad	Proveedor		Coste Total
GY-906	1	9,99 €	Solectro	[28]	9,99 €
LCD 2x16, I2C	1	6,49 €	Amazon	[29]	6,49 €
Arduino Mega 2560 R3	1	13,99 €	Arduino	[9]	35 €
AMG8833	1	39,90 €	Bricogeek	[30]	39,90 €
Raspberry PI 3 model B	1	37,44 €	Amazon	[31]	37,44 €
Led	1	5,50 €	Bricogeek	[32]	5,50 €
Cables Macho- Macho	1	1,90 €	Bricogeek	[33]	1,90 €
Cables Macho-Hembra	1	1,90 €	Bricogeek	[34]	1,90 €
Total					17,11 €

Tabla 1: Presupuesto del material

A continuación, aparecerá un presupuesto con las mejoras planeadas anteriormente:

Material	Cantidad	Coste por unidad	Proveedor		Coste Total
MLX90640	1	145,99€	Amazon	[24]	145,99 €
Raspberry Pi 4 model B, 4 GB RAM	1	61,5€	Kubii	[26]	61,5€
Carcasa	2				Por determinar
Buzzer	1	0,65€	Tiendatec	[35]	0,65€
Power bank	2	16,97€	PC Componentes	[36]	33,94€
HC-SR04	1	1,95€	Tiendatec	[37]	1,95€
LCD táctil 5’’	1	37,95€	Tiendatec	[38]	37,95€
Lenovo C24-25	1	115€	Amazon	[39]	115€
Total					396,98 €

Tabla 2: Presupuesto de mejoras

8 CONCLUSIONES

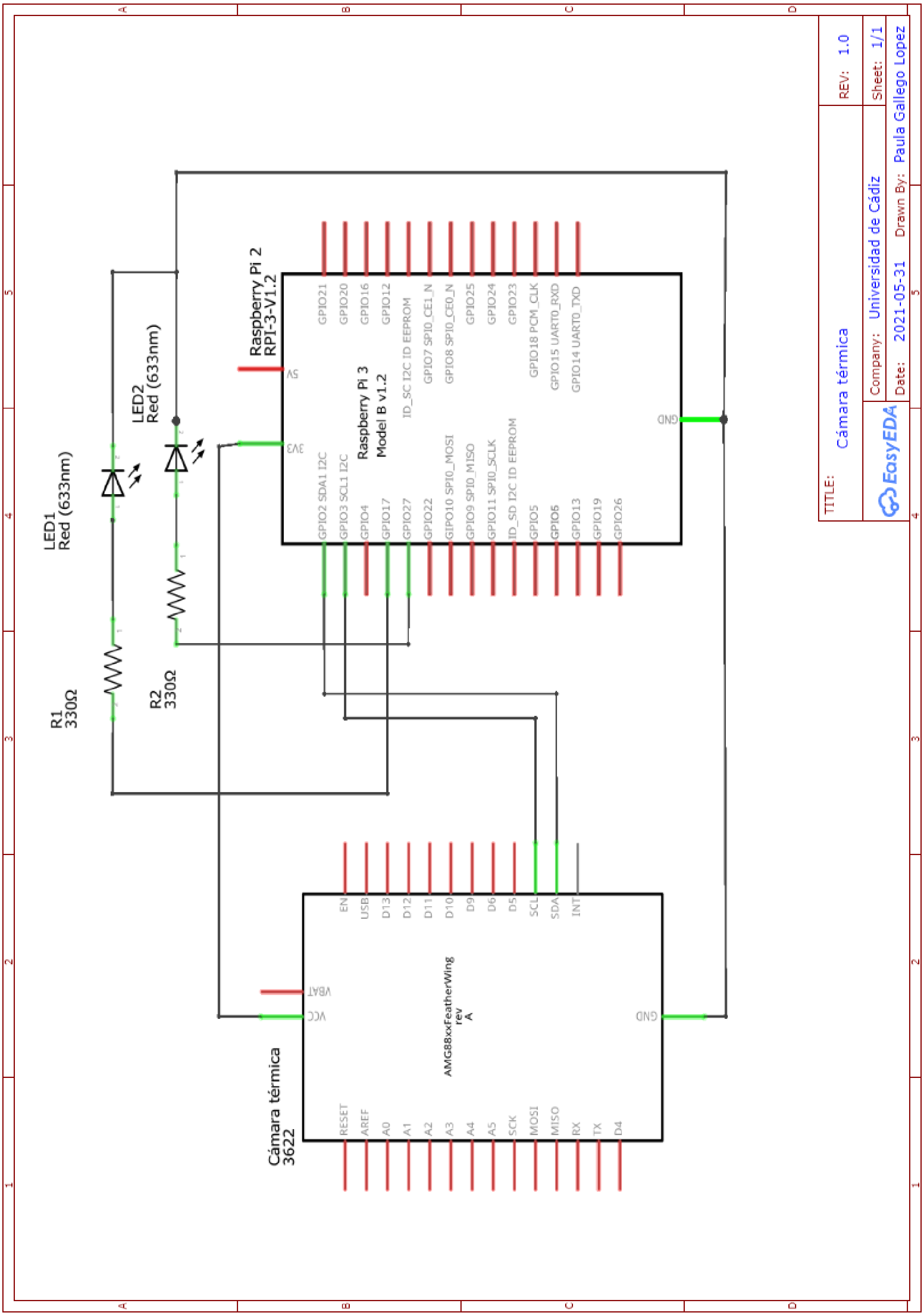
En este Trabajo Final de Grado se ha logrado el objetivo principal planteado, el desarrollo de un sistema de control de temperatura óptico formado en una cámara térmica y un termómetro sin contacto para realizar un cribado rápido (cámara térmica) y a continuación confirmar si las mediciones son correctas (termómetro).

Se ha obtenido como resultado una alternativa válida a las opciones que podemos obtener en el mercado, ya que gracias a la comunidad que existe de programadores se puede obtener fácilmente y modificar los códigos existentes para elaborar cada individuo su propio sistema.

También se ha logrado una opción más económica a las del mercado ya que en el caso de la cámara térmica el precio de venta esta entre 140 a 5000 € los modelos más avanzados y el termómetro de pared lo podemos encontrar desde 30 a 400 €. Nuestro sistema tendría un coste menos de 120 € el conjunto, lo que supondría un ahorro de dinero y la posibilidad de crearlo por tus propios medios y aprender en el proceso.

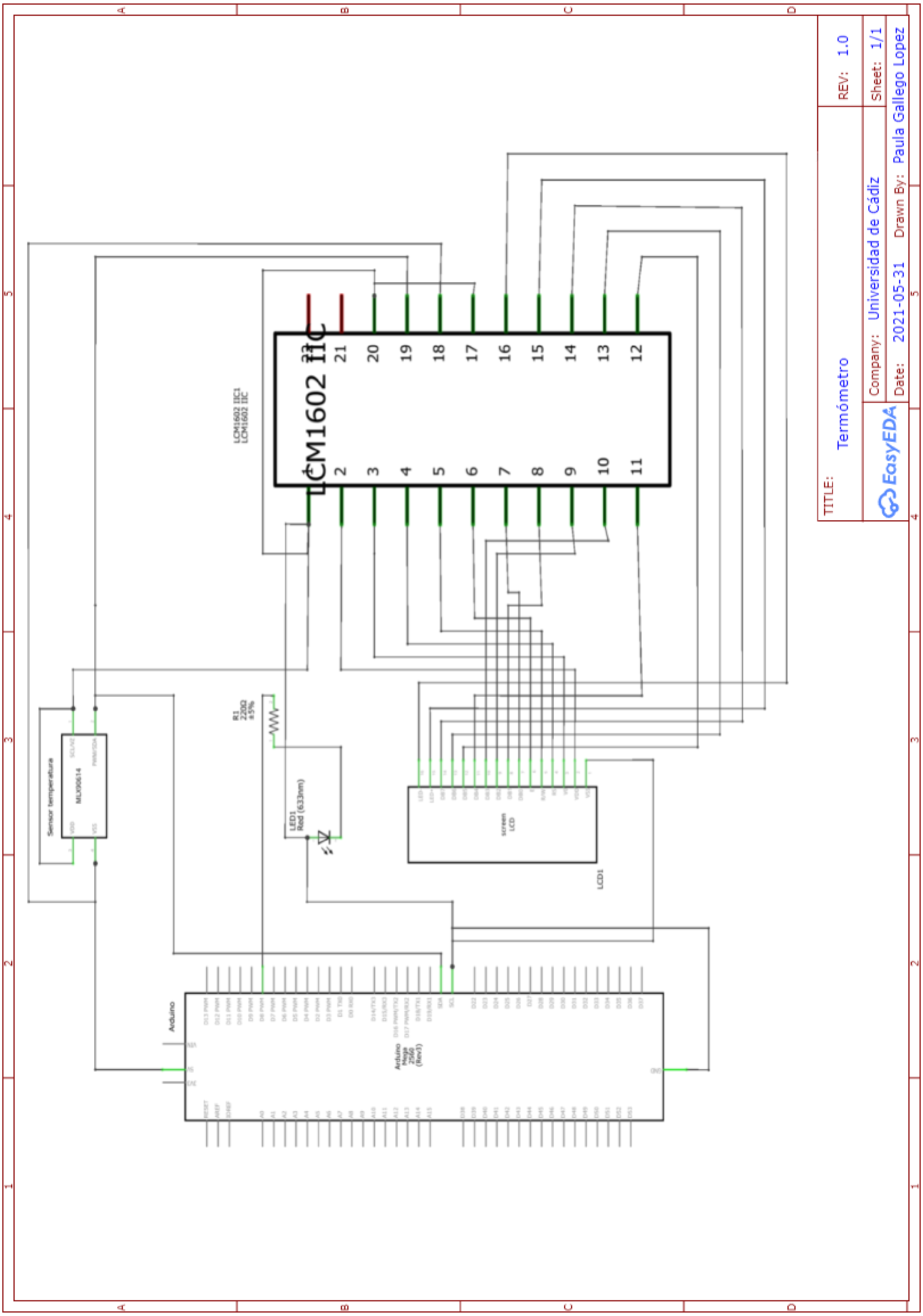
Una de las conclusiones derivada gracias a la realización del proyecto, han sido el conocimiento de una enorme comunidad en la que te puedes apoyar para realizar cualquier proyecto que te propongas con esta ayuda. Por lo tanto, el conocimiento de esta comunidad nos provee de una herramienta gracias a la cual, podemos afrontar nuevos retos y trabajar en el desarrollo colaborativo de proyectos de nuestro interés.

ANEXO 1



Raspberry Pi 2
RPI-3-V1.2
Raspberry Pi 3
Model B v1.2
LED1
Red (633nm)
LED2
Red (633nm)
R1
330Ω
R2
330Ω
Cámara térmica
3622
AMG88xxFeatherWing
rev
A
TITLE: Cámara térmica
REV: 1.0
Company: Universidad de Cádiz
Sheet: 1/1
Date: 2021-05-31
Drawn By: Paula Gallego Lopez
EasyEDA

ANEXO 2


TITLE: Termómetro
REV: 1.0
Company: Universidad de Cádiz
Sheet: 1/1
Date: 2021-05-31
Drawn By: Paula Gallego Lopez
EasyEDA
LCM1602 IIC
Sensor temperatura
MLX90614
R1
LED1
screen LCD
LCD1
Arduino

El anexo actual y el anterior han sido diseñados en por dos plataformas:

- Fritzing, la parte del esquema electrónico. Además, este ha sido utilizado para las ilustraciones de los dispositivos.[40]
- EasyEDA, la parte del cajetín. [41]

ANEXO 3

En este anexo, se muestra el código empleado para la cámara térmica con Python:

```
# Librerías
import time,sys
sys.path.append('../')
# load AMG8833 module
import amg8833_i2c
import numpy as np
import matplotlib.pyplot as plt
from scipy import interpolate
from gpiozero import LED #importamos la clase led de la libreria gpiozero
from signal import pause #importamos la clase pause de la liberia signal

blue = LED(17)
yellow= LED (27)

#####################################
# Initialization of Sensor
#####################################

t0 = time.time()
sensor = []
while (time.time()-t0)<1: # espere un segundo para que el sensor comience
    try:
        # AD0 = GND, addr = 0x68 | AD0 = 5V, addr = 0x69
        sensor = amg8833_i2c.AMG8833(addr=0x69) # empieza AMG8833
    except:
        sensor = amg8833_i2c.AMG8833(addr=0x68)
    finally:
        pass
time.sleep(0.1) # espera a que el sensor se asiente

# si no se encuentra ningun dispositivo, sale del script
if sensor==[]:
    print("No AMG8833 Found - Check Your Wiring")
    sys.exit(); # salir de la app si no se encuentra AMG8833

#####################################
# Propiedades de interpolación
#####################################

# resolución original
```

```
pix_res = (8,8) # resolución pixel
xx,yy = (np.linspace(0,pix_res[0],pix_res[0]),
                np.linspace(0,pix_res[1],pix_res[1]))
zz = np.zeros(pix_res) # establecer matriz con ceros primero
# nueva resolución
pix_mult = 6 # multiplicar la interpolación
interp_res = (int(pix_mult*pix_res[0]),int(pix_mult*pix_res[1]))
grid_x,grid_y = (np.linspace(0,pix_res[0],interp_res[0]),
                    np.linspace(0,pix_res[1],interp_res[1]))
# interpolar función
def interp(z_var):
   # interpolación cúbica en la imagen
   # a una resolución de (pix_mult*8 x pix_mult*8)
   f = interpolate.interp2d(xx,yy,z_var,kind='cubic')
   return f(grid_x,grid_y)
grid_z = interp(zz) # imagen interpolada

#####################################
# Figura de inicio y formato
#####################################

plt.rcParams.update({'font.size':16})
fig_dims = (10,9) # tamaño figura
fig,ax = plt.subplots(figsize=fig_dims) # comienzo figura
fig.canvas.set_window_title('AMG8833 Image Interpolation')
im1 = ax.imshow(grid_z,vmin=18,vmax=40,cmap=plt.cm.RdBu_r) # trazar imagen con límites
de                  temperatura
cbar = fig.colorbar(im1,fraction=0.0475,pad=0.03) # barra de color
cbar.set_label('Temperature [C]',labelpad=10) # etiqueta de temperatura
fig.canvas.draw() # dibujar figura

ax_bgnd = fig.canvas.copy_from_bbox(ax.bbox) # fondo
fig.show() # mostrar figura

#####################################
# Trazar temperaturas de AMG88833 en tiempo real
#####################################

pix_to_read = 64 # leer los 64 pixeles
while True:
   status,pixels = sensor.read_temp(pix_to_read) # leer pixeles con estado
   if status: # si hay un error en el píxel, vuelve a entrar en el bucle y vuelve a intentarlo
      continue

   T_thermistor = sensor.read_thermistor() # leer temperatura del termistor
```

```
fig.canvas.restore_region(ax_bgnd) # restaurar fondo
new_z = interp(np.reshape(pixels,pix_res)) # imagen interpolada
im1.set_data(new_z) # actualizar el gráfico con nuevas temperaturas interpoladas
ax.draw_artist(im1) # dibujar imagen de nuevo
fig.canvas.blit(ax.bbox) # blitting
fig.canvas.flush_events() # para el gráfico en tiempo real

for i in range(0,64,1):

  if  37 <= pixels[i] <38: #cuando llega a 37 va a parpadear
      yellow.blink(1,1,3) # parpadeará 3 veces y tardará un segundo entre parpadeos

  if  pixels[i] >= 38: #cuando es igual o mayor a 38
    blue.blink(1,2,5) #pardaeará 5 veces y tardará 1 segundo entre apagado y encendido
            #2 segundos entre encendido y apagado
```

ANEXO 4

A continuación, se muestra el código diseñado para el termómetro sin contacto:

```
#include <LiquidCrystal_I2C.h>  // Libreria que permite controlar pantallas por I2C
#include <i2cmaster.h>         // Libreria para sensor MLX90614
#include <Wire.h>              // Libreria que permite comunicarse con dispositivos I2C

LiquidCrystal_I2C lcd(0x27, 16, 2); // configuración de pines de la pantalla LCD
const int led=8;              //varible del Led
int deviceAddress = 0x50<<1;    //  0x50 es la dirección de comunicación de I2C
float celcius = 0;            // Variable que contiene la temperatura en Celcius.

void setup() {
 lcd.begin();
 Serial.begin(9600);          // Inicia la comunicación serial a 9600bps.
 pinMode(led,OUTPUT);

 i2c_init();                         // Inicia el bus i2c.
 PORTC = (1 << PORTC4) | (1 << PORTC5);    // Habilita ‘pullups’.
 }

float temperatureCelcius(int address) {   // Cambió de variable numerica tipo entera a decimal
 int dev = address;
 int data_low = 0;
 int data_high = 0;
 int pec = 0;

 // Escribe
 i2c_start_wait(dev+I2C_WRITE);
 i2c_write(0x07);

 // Lee
 i2c_rep_start(dev+I2C_READ);
 data_low = i2c_readAck();       // Lee 1 byte y envía ack.
 data_high = i2c_readAck();      // Lee 1 byte y envía ack
 pec = i2c_readNak();
 i2c_stop();

 // Esto convierte los bytes altos y bajos juntos y procesa la temperatura.
 double tempFactor = 0.02;       // 0.02 grados por LSB (medida de resolución del MLX90614).
 double tempData = 0x0000;
 int frac;                   // Datos después del punto decimal.

 // Esto oculta el error del byte alto y lo mueve a la izquierda
```

```
  // 8 bits y agrega el byte bajo.
  tempData = (double)(((data_high & 0x007F) << 8) + data_low);
  tempData = (tempData * tempFactor)-0.01;
  float celcius = tempData - 273.15;

// Datos mostrados en pantalla
  lcd.backlight();
  lcd.clear();
  lcd.print("temperatura");
  lcd.setCursor ( 0, 1 );
  lcd.print(celcius);
  lcd.setCursor ( 1,3 );

// Condición para alerta usando led
  if(celcius<38)   //Si la temperatura supera los 38ºC se activa el led
  {digitalWrite(led, HIGH);}
  else {digitalWrite(led, LOW);}   //Si no la supera no se activa

 // Retorna la temperatura en Celcius.
  return celcius;
}

void loop() {
celcius = temperatureCelcius(deviceAddress);  // Lee los datos del MLX90614 con la dirección
dada, los transforma en la temperatura en Celcius y la almacena en la variable celcius.

  Serial.print("Celcius: ");          // Imprime ambas lecturas en el
  Serial.println(celcius);            // puerto serial.
  Serial.println();
  delay(1000);                        // Espera un segundo antes de imprimir de nuevo.
}
```

ANEXO 6

Panasonic Infrared Array Sensor Grid-EYE (AMG88)

Infrared Array Sensor Grid-EYE

High Precision Infrared Array Sensor based on Advanced MEMS Technology

Features

- Temperature detection of two-dimensional area: 8 × 8 (64 pixels)
- Digital output (capability of temperature value output)
- Compact SMD package (adaptively to reflow mounting)
- RoHS compliant

Typical applications

- High function home appliances (microwaves and air-conditioners)
- Energy saving at office (air-conditioning/lighting control)
- Digital signage
- Automatic doors/elevators

Ordering information

AMG 8 8 _ _

- Vertical pixel 8: 8 pixels
- Horizontal pixel 8: 8 pixels
- Applied voltage 3 : 3.3 V.DC 5 : 5.0 V.DC
- Applied voltage 3 : High performance type High gain 4 : High performance type Low gain

Types

Tape and reel package : 1,000 pcs.

Product name	Number of pixel	Operating voltage	Amplification factor	Part number
Infrared array sensor Grid-EYE High performance type	64 (Vertical 8 × Horizontal 8 Matrix)	3.3 V.DC	High performance type High gain	AMG8833
			High performance type Low gain	AMG8834
		5.0 V.DC	High performance type High gain	AMG8853
			High performance type Low gain	AMG8854

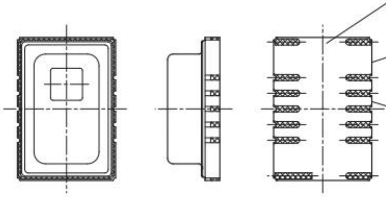
Rating

Item	Performance	
	High gain	Low gain
Applied voltage	3.3 V.DC±0.3 V.DC or 5.0 V.DC±0.5 V.DC	
Temperature range of measuring object	0 °C to 80 °C +32 °F to +176 °F	−20 °C to 100 °C −4 °F to +212 °F
Operating temperature range	0 °C to 80 °C +32 °F to +176 °F	−20 °C to 80 °C −4 °F to +176 °F
Storage temperature range	−20 °C to 80 °C −4 °F to +176 °F	−20 °C to 80 °C −4 °F to +176 °F

Panasonic Infrared Array Sensor Grid-EYE (AMG88)

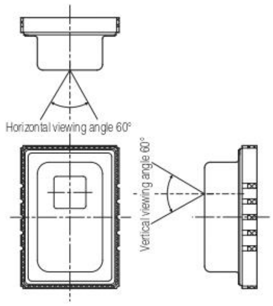
Pixel array and viewing field

(1) Pixel array
Pixel array from 1 to 64 is shown below.



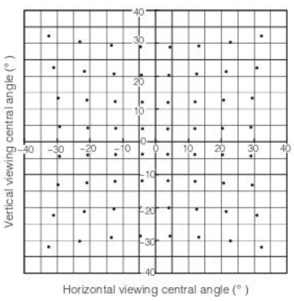
64	63	62	61	60	59	58	57
56	55	54	53	52	51	50	49
48	47	46	45	44	43	42	41
40	39	38	37	36	35	34	33
32	31	30	29	28	27	26	25
24	23	22	21	20	19	18	17
16	15	14	13	12	11	10	9
8	7	6	5	4	3	2	1

(2) Viewing field
Sensor viewing field (typical) is shown below.

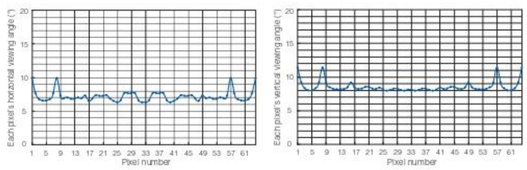


Optical properties

(1) Each pixel's viewing central angle
Sensor's optical center (the origin of graph below) gap: within ±5.6 ° (Typical) (Both horizontal and vertical directions)

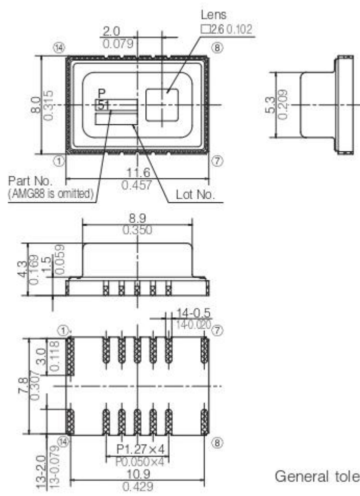


(2) Each pixel's viewing angle (half angle)
Central 4 pixels (Pixel No. 28, 29, 36, 37) viewing angle (half angle): horizontal direction 7.7 ° (Typical) vertical direction 8 ° (Typical)



Dimensions

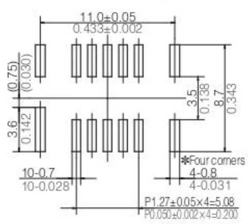
External dimensions



Number	Terminal Name	Number	Terminal Name
①	NC	⑧	NC
②	SDA	⑨	VDD
③	SCL	⑩	AVDD-PC
④	INT	⑪	NC
⑤	AD_SELECT	⑫	DVDD-PC
⑥	GND	⑬	VPP
⑦	NC	⑭	NC

Note : Leave terminal "NC (No.①,⑦,⑧,⑪ and ⑭)" unconnected.
Make electrical potential of terminals ⑨ and ⑬ the same.

Recommended PC board pad



General tolerance : ±0.2 ±0.08

unit : mm inch

Design and specifications are each subject to change without notice. Ask factory for the current technical specifications before purchase and/or use.
Should a safety concern arise regarding this product, please be sure to contact us immediately.

02 Apr. 2017

Panasonic Infrared Array Sensor Grid-EYE (AMG88)

Absolute maximum ratings

Item	Absolute maximum ratings	Terminal
Applied voltage	–0.3 V.DC to 6.5 V.DC	VDD
Input voltage	–0.3 V.DC to VDD +0.3 V.DC	SCL, SDA, AD_SELECT
Output sink current	–10 mA to 10 mA	INT, SDA
Static electricity (Human body model)	1 kV	All terminals
Static electricity (Machine model)	200 V	All terminals

Characteristics

Item	Performance	
	High performance type High gain	High performance type Low gain
Temperature accuracy	Typical ±2.5 °C ±4.5 °F	Typical ±3.0 °C ±5.4 °F
Human detection distance *1	7 m or less (reference value) 22.966 ft	
NETD *2	Typ. 0.05 °C 32.900 °F 1 Hz Typ. 0.16 °C 32.288 °F 10 Hz	
Viewing angle	Typical 60 °	
Optical axis gap	Within Typical ±5.6 °	
Current consumption	Typical 4.5 mA (normal mode) Typical 0.2 mA (sleep mode) Typical 0.8 mA (stand-by mode)	
Setup time	Typical 50 ms (Time to enable communication after setup) Typical 15 s (Time to stabilize output after setup)	

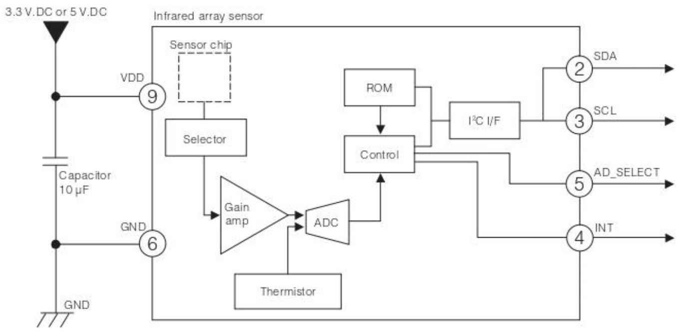
Note: *1 To have more than 4 °C 7.2 °F of temperature difference from background
Detection object size: 700 × 250 mm 27.559 × 9.843 inch (Assumable human body size)
*2 It is calculated from 4 pixels of centers.

Performance

Item	Performance
Number of pixel	64 (Vertical 8 × Horizontal 8 Matrix)
External interface	I²C (fast mode)
Frame rate	Typical 10 frames/sec or 1 frame/sec
Operating mode *1	Normal Sleep Stand-by (10 sec or 60 sec intermittence)
Output mode	Temperature output
Calculate mode	No moving average or Twice moving average
Temperature output resolution	0.25 °C 32.45 °F
Number of sensor address	2 (I²C slave address)
Thermistor output temperature range	–20 °C to 80 °C –4 °F to +176 °F
Thermistor output resolution	0.0625 °C 32.1125 °F

Note: *1 Normal Mode : normal operation mode; Sleep Mode: detection is off (output and data reading not possible); Standby Mode: 1 frame measuring intermittently every 10 or 60 sec.

Internal circuit



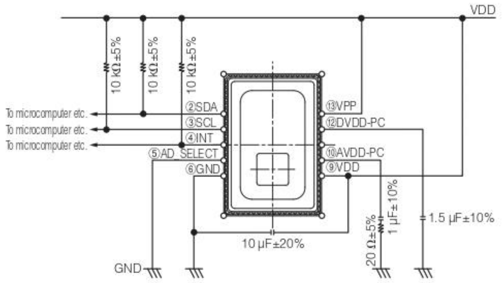
* INT terminal ④ normally has same voltage as VDD. When interrupting, same as GND (0V)

Design and specifications are each subject to change without notice. Ask factory for the current technical specifications before purchase and/or use.
Should a safety concern arise regarding this product, please be sure to contact us immediately.

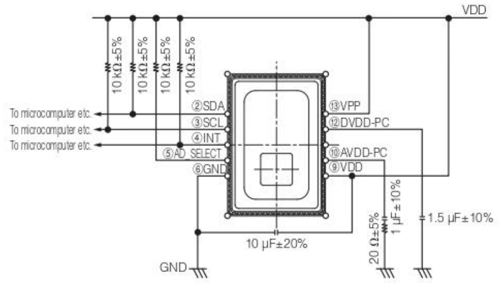
Panasonic Infrared Array Sensor Grid-EYE (AMG88)

External circuit

(1) In case of setting I²C slave address of the sensor 1101000
✻ Connect terminal ⑤ (AD_SELECT) to GND.



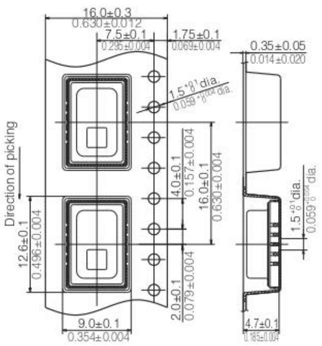
(2) In case of setting I²C slave address of the sensor 1101001
✻ Connect terminal ⑤ (AD_SELECT) to VDD.



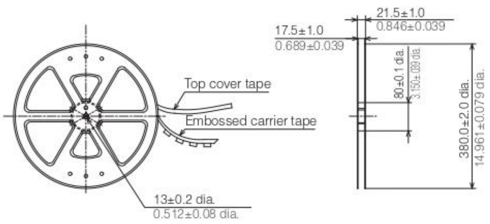
This circuit is an example to drive infrared array sensor "Grid-EYE", so that we will not take any responsibility of loss which is due to this circuit.

Packing format (Tape and reel)

Tape dimensions



Dimensions of tape reel



unit : mm inch

Notes

■ Precaution for fundamental structure of sensor

Infrared Array Sensor is a thermopile type infrared sensor which detects the amount of infrared rays. Below conditions generally degrade the temperature accuracy. Carefully check the performance and stability under actual use conditions, and perform temperature corrections when necessary.

- When heating elements exist near the mounting position of the sensor.
- When the sensor is exposed to cold or hot air.
- When the temperature of the sensor body rapidly changes.
- When substances (e.g., glasses, acrylics or steams), which hardly transmit a far infrared ray, exist between the sensor and the detected object.
- When substances (e.g., foreign substances or water), which hardly transmit a far infrared ray, adhere to the lense of the sensor.

■ Use environment

1) Temperature: See the specifications
2) Humidity: Between 15% and 85% R.H. (Avoid freezing and dew condensation)
3) Atmospheric pressure: Between 86 and 106 kPa
4) Vibrations and shocks may damage the sensor, and cause malfunction and performance deterioration. If loads and shocks are applied on the lense, the damaged sensor may cause malfunction and performance deterioration.
5) The product is not water/splash-proof. Perform water/dust-proofing and dew condensation/freezing countermeasures in accordance with use environment. When dew condensation occurs, responsiveness of heat source detection may delay for several seconds.
6) Avoid use and storage in the corrosive gas (organic solvent, sulfurous acid and hydrogen sulfide gases) to avoid malfunction and performance deterioration.
7) Use surge absorbers as applying the external surge voltage may damage the internal circuit.
8) Malfunction may occur near electric noises from static electricity, lightning, broadcast or amateur radio stations and mobile phones.
9) The sensor can continuously operate within the range of using ambient temperature (using ambient humidity). However, ensure that humidity is within the range described in the following page as humidity varies according to temperature. Avoid the continuous operation near the operational limit. The temperature range does not guarantee the durability.

Other precautions

These specifications are for individual components. Before use, carefully check the performance and quality under actual use conditions to enhance stability.

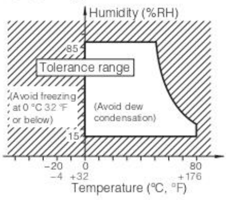
1) Once the individual sensor is dropped, do not use. Drop may cause functional disorders.
2) Writing to the unspecified register/with the unspecified bit may cause malfunction and performance deterioration. (please consult us)
3) Misconnection and use beyond the specified temperature range may damage the product.
4) Once below shocks are applied, do not use the product as applying highfrequency oscillation to the sensor body may damage the product.
   - Contact with metal objects
   - Contact with other sensors
5) Follow the instructions below as static electricity may damage the product.
   - For storage and transportation, avoid plastic containers which are easily electrified.
   - When storing and transporting the sensor, choose the environment where static electricity is hardly generated (e.g., humidity between 45 and 60 %) and protect the product by using electroconductive packaging materials.
   - Once unpacked, perform antistatic countermeasures.
     (1) Operators handling sensors must wear antistatic cloths and human body grounding devices.
     (2) Cover the surface of workbench by electro-conductive plates and ground measuring instruments and jigs.
     (3) Use the soldering iron which has a small leakage current or ground the soldering tip.
     (4) Ground the assembling equipment.
   - Use a stabilized power supply. A power superimposed noise may cause malfunction.

Range of using ambient temperature (using ambient humidity)

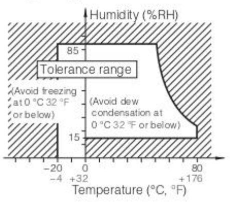
The sensor can continuously operate within the range of using ambient temperature (using ambient humidity). However, ensure that humidity is within the range below as humidity varies according to temperature. Avoid the continuous operation near the operational limit. Before use, check the stability under the usage environment as high humidity or high temperatures generally accelerates deterioration of the electronic component.

- The temperature range does not guarantee the durability

High gain type



Low gain type



Mounting

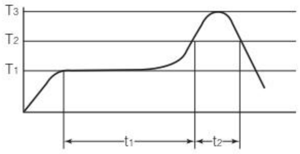
Use the land of the printed-circuit boardon which the sensor is securely fixed. The recommended printed-circuit board is FR4 (thickness 1.6 mm 0.063 inch). When mounting on the deprecated circuit board, carefully check the performance and quality under actual use conditions before use.

- A large noise on the power supply may cause malfunction. Place the recommended capacitor near the sensor (within 20 mm 0.787 inch of the wiring pattern length) between sensor input terminals (VDD-GND) to secure power superimposed noise resistance. Test with the actual machine and re-select the capacitor with optimal capacitance.
- Prevent the metal part of other electronic components from contacting with the sensor body as the upper face (where part numbers are imprinted) of the sensor is GND.

Soldering

When soldering, avoid the external thermal influence. Heat deformation may damage the sensor or deteriorate its performance. Use the non-corrosive rosin flux.

1) Manual soldering
   - Raise the temperature of the soldering tip between 350 and 400 °C 662 and 752 °F (30 and 60 W) and solder within 3 seconds.
   - The sensor output may vary if the load is applied on the terminal during soldering.
   - Keep the soldering tip clean.
2) Reflow soldering
   Below are recommended temperature profiles/conditions of reflow.
   - When printing cream solder, the screen printing method is recommended.
   - For the foot pattern, see the recommended diagram of the printed-circuit board.
   - Carefully align the terminal with the pattern as self-alignment may not be reliable.
   - The temperature of the profile is the value measured near the terminal on the printed-circuit board.
   - After reflowing, when performing reflow soldering on the back surface of the circuit board, use an adhesive to fix the board.



T_1 = 150 to 180 °C 302 °F to 356 °F
T_2 = 230 °C 446 °F
T_3 = Below 250 °C 482 °F
t_1 = 60 to 120 sec.
t_2 = Less than 30sec.

3) After soldering, do not apply stress on the soldered part when cutting or bending the circuit board.
4) Rework soldering
   - Complete rework at a time.
   - Use a flattened soldering tip when performing rework on the solder bridge. Do not add the flux.
   - Keep the soldering tip below the temperature described in the specifications.
5) Prevent human hands or metal pieces from contacting with the sensor terminal. Such contact may cause anomalous outlets as the terminal is exposed to the atmosphere.
6) After soldering, prevent chemical agents from adhering to the sensor when applying coating to avoid insulation deterioration of the circuit board.

Panasonic **Infrared Array Sensor Grid-EYE (AMG88)**

■ Wire connection

1) Correctly wire as in the connection diagram. Reverse connection may damage the product and degrade the performance.
2) Do not use idle terminals. Such use may damage the sensor.
3) For cable wiring, use shield wires with possibly short wiring lengths to prevent the influence of the noise.

■ Cleaning

Avoid ultrasonic cleaning as this may cause disconnection of the wire.

■ Storage and transportation

1) Excessive vibrations and shocks during transport may damage the product. Carefully handle the exterior box and the reel.
2) Extremely bad storage conditions may deteriorate solderability or characteristics, and defect the appearance. Recommended conditions of the storage place are below.
   - Temperature: 0 to 45 °C 32 to 113 °F
   - Humidity: Below 70 % R.H.
   - Atmosphere: Low-dust and free from noxious chemicals such as sulfurous acid gas
3) The package is moisture-proof due to its sensitivity to humidity. When storing the sensor, follow the instructions below.
   - Promptly use after opening. (within a week, below 30 °C 86 °F/60 % R.H.)
   - Once unpacked, preserving in a moisture-proof manner, such as keeping in a moisture-proof bag with silica gels, is recommended for long-term storage. (use within 3 months)

   ✻ During soldering, when adding thermal stress in a moisture absorbing state, moisture evaporates, swells and generates stress to the internal package. To avoid swellings and cracks in the surface of the package, follow the soldering conditions.

■ Special notes

We exert maximum efforts for quality control of the product, however :

1) To prevent occurrence of unexpected circumstances, please inform us of the specifications of your product, customers, use conditions and details of the attachment position.
2) Have sufficient margin values of driving/performance guarantee described in the specifications and apply safety measures with double circuits, if serious effects on human lives or property are predicted due to a quality failure of the product. Those countermeasures are also for the product liability.
3) A warranty period is one year after the delivery to your company. Quality assurance is limited to the items and the scopes described in the specifications.
   If a defect is found after the delivery, we will promptly provide a replacement or change/repair the defect part at the place of delivery in good faith. Exceptions are below.
   - Damages by a failure or a defect which arose after the delivery.
   - After the delivery, when storing and transporting, if conditions other than conditions in the specifications are applied to the product.
   - Damages by unforeseen phenomenon which cannot be predicted with the technologies available at the time of delivery.
   - Damages by natural and anthropogenic disasters, such as earthquake, flood, fire and war, which are beyond our reasonable control.

ANEXO 7

MLX90614 family

Single and Dual Zone
Infra Red Thermometer in TO-39

Features and Benefits

- Small size, low cost
- Easy to integrate
- Factory calibrated in wide temperature range:
  - -40 to 125 ˚C for sensor temperature and
  - -70 to 380 ˚C for object temperature.
- High accuracy of 0.5°C over wide temperature range (0..+50°C for both Ta and To)
- High (medical) accuracy calibration optional
- Measurement resolution of 0.02°C
- Single and dual zone versions
- SMBus compatible digital interface
- Customizable PWM output for continuous reading
- Available in 3V and 5V versions
- Simple adaptation for 8 to 16V applications
- Power saving mode
- Different package options for applications and measurements versatility
- Automotive grade

Applications Examples

- High precision non-contact temperature measurements;
- Thermal Comfort sensor for Mobile Air Conditioning control system;
- Temperature sensing element for residential, commercial and industrial building air conditioning;
- Windshield defogging;
- Automotive blind angle detection;
- Industrial temperature control of moving parts;
- Temperature control in printers and copiers;
- Home appliances with temperature control;
- Healthcare;
- Livestock monitoring;
- Movement detection;
- Multiple zone temperature control – up to 100 sensors can be read via common 2 wires
- Thermal relay/alert
- Body temperature measurement

Ordering Information

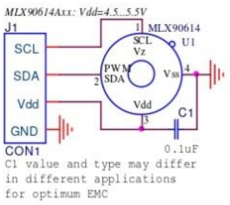
Part No. MLX90614 X X X
(1) (2) (3)

(1) Supply Voltage:
A - 5V power
(adaptable for 12V)
B - 3V power

(2) Number of thermopiles:
A – single zone
B – dual zone

(3) Package type:
A – Filter inside
B – Filter outside

1 Functional diagram



MLX90614 connection to SMBus

Figure 1 Typical application schematics

2 General Description

The MLX90614 is an Infra Red thermometer for non contact temperature measurements. Both the IR sensitive thermopile detector chip and the signal conditioning ASSP are integrated in the same TO-39 can.
Thanks to its low noise amplifier, 17-bit ADC and powerful DSP unit, a high accuracy and resolution of the thermometer is achieved.
The thermometer comes factory calibrated with a digital PWM and SMBus output.
As a standard, the 10-bit PWM is configured to continuously transmit the measured temperature in range of -20 to 120 ˚C, with an output resolution of 0.14 ˚C.
The POR default is SMBus interface

General description (continued)

The MLX90614 is built from 2 chips developed and manufactured by Melexis:
The Infra Red thermopile detector MLX81101.
The signal conditioning ASSP MLX90302, specially designed to process the output of IR sensor.
The device is available in an industry standard TO-39 package.

Thanks to the low noise amplifier, high resolution 17-bit ADC and powerful DSP unit of MLX90302 high accuracy and resolution of the thermometer is achieved. The calculated object and ambient temperatures are available in RAM of MLX90302 with resolution of 0.01 ˚C. They are accessible by 2 wire serial SMBus compatible protocol (0.02°C resolution) or via 10-bit PWM (Pulse Width Modulated) output of the device.

The MLX90614 is factory calibrated in wide temperature ranges: -40 to 125 ˚C for the ambient temperature and -70 to 382.2 ˚C for the object temperature. The 10-bit PWM is as a standard configured to transmit continuously the measured object temperature for an object temperature range of -20 to 120 ˚C with an output resolution of 0.14 ˚C. The PWM can be easily customized for virtually any range desired by customer by changing the content of 2 EEPROM cells. This has no effect on the factory calibration of the device.

The PWM pin can also be configured to act as a thermal relay (input is To), thus allowing for an easy and cost effective implementation in thermostats or temperature (freezing/boiling) alert applications. The temperature threshold is user programmable. In an SMBus system this feature can act as a processor interrupt that can trigger reading all slaves on the bus and to determine the precise condition.

As a standard, the MLX90614 is calibrated for an object emissivity of 1. It can be easily customized by the customer for any other emissivity in the range 0.1-1.0 without the need of recalibration with a black body.

The thermometer is available in 2 supply voltage options: 5V compatible or 3V (battery) compatible. The 5V can be easily adopted to operate from a higher supply voltage (8-16V, for example) by use of few external components (refer to “Applications information” section for details).

An optical filter (long-wave pass) that cuts off the visible and near infra-red radiant flux is integrated in the package to provide sunlight immunity.

Table of Contents

3 Glossary of Terms

PTAT	Proportional To Absolute Temperature sensor (package temperature)
PTC	Positive Temperature Coefficient sensor (package temperature)
POR	Power On Reset
HFO	High Frequency Oscillator (RC)
DSP	Digital Signal Processing
FIR	Finite Impulse Response. Digital filter
IIR	Infinite Impulse Response. Digital filter
IR	Infra-Red
PWM	Pulse With Modulation
DC	Duty Cycle (of the PWM) ; Direct Current (for settled conditions specifications)
FOV	Field Of View
SDA,SCL	Serial DAta, Serial CLock – SMBus compatible communication pins
Ta	Ambient Temperature measured from the chip – (the package temperature)
To	Object Temperature, 'seen' from IR sensor
ESD	Electro-Static Discharge
EMC	Electro-Magnetic Compatibility
TBD	To Be Defined

Note: sometimes the MLX90614xxx is referred to as "the module".

4 Maximum ratings

Parameter.	MLX90614AAA MLX90614ABA	MLX90614BAA MLX90614BBA	MLX90614AAB MLX90614ABB	MLX90614BAB MLX90614BBB
Supply Voltage, V_{DD} (over voltage)	7V	5V	7V	5V
Supply Voltage, V_{DD} (operating)	5.5 V	3.6V	5.5V	3.6V
Reverse Voltage	0.4 V			
Operating Temperature Range, T_A	-40 to +125°C		-40…+85°C	
Storage Temperature Range, T_S	-40…+125 °C		-40…+105°C	
ESD Sensitivity (AEC Q100 002)	2kV			
DC current into SCL/Vz (Vz mode)	2 mA			
DC sink current, SDA /PWM pin	25 mA			
DC source current, SDA/PWM pin	25 mA			
DC clamp current, SDA/PWM pin	25 mA			
DC clamp current, SCL pin	25 mA			

Table 1: Absolute maximum ratings for MLX90614

Exceeding the absolute maximum ratings may cause permanent damage. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

5 Pin definitions and descriptions

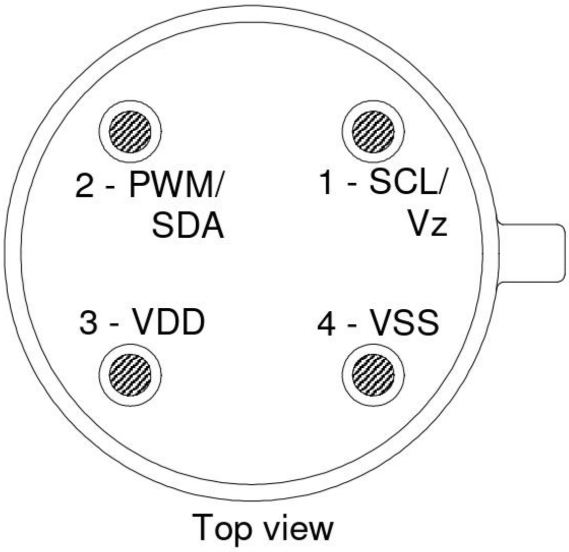


Figure 2: Pin description

Pin Name	Function
VSS	Ground. The metal can is also connected to this pin.
SCL / Vz	Serial clock input for 2 wire communications protocol. 5.7V zener is available at this pin for connection of external bipolar transistor to MLX90614A to supply the device from external 8 -16V source.
PWM / SDA	Digital input / output. In normal mode the measured object temperature is available at this pin Pulse Width Modulated.
VDD	External supply voltage.

Table 2: Pin description MLX90614

Note: for +12V (+8…+16V) powered operation refer to the Application information section. For EMC and isothermal conditions reasons it is highly recommended not to use any electrical connection to the metal can except by the Vss pin.
With the SCL/Vz and PWM/SDA pins operated in 2-wire interface mode, the input Schmidt trigger function is automatically enabled.

6 Electrical Specifications

6.1 MLX90614Axx

All parameters are preliminary for T_A = 25 ˚C, $\mathbf{V_{DD}}$ **=5V** (unless otherwise specified)

Parameter	Symbol	Test Conditions	Min	Typ	Max	Units
Supplies						
External supply	V_{DD}		4.5	5	5.5	V
Supply current	I_{DD}	No load			1	mA
Supply current (programming)	I_{DDpr}	No load, erase/write EEPROM operations			1.5	mA
Zener voltage	Vz	Iz = 75…400 uA	5.6	5.75	5.8	V
Zener voltage	Vz(Ta)	Iz=70…400uA, full temperature range	5.15	5.75	6.24	V
Power On Reset						
POR level	V_{POR}	Power-up, power-down and brown-out	2.7	3.0	3.3	V
V_{DD} rise time	T_{POR}	Ensure POR signal			3	ms
Output valid (result in RAM)	Tvalid	After POR		0.15		s
Pulse width modulation[1]						
PWM resolution	PWMres	Data band		10		bit
PWM output period	$PWM_{T,def}$	Factory default, internal oscillator factory calibrated		1.024		ms
PWM period stability	$dPWM_T$	Internal oscillator factory calibrated, over the entire operation range and supply voltage	-4		+4	%
Output high Level	PWM_{HI}	I_{source} = 2 mA	V_{DD}-0.2			V
Output low Level	PWM_{LO}	I_{sink} = 2 mA			V_{SS}+0.2	V
Output drive current	$Idrive_{PWM}$	Vout,H = V_{DD} - 0.8V		20		mA
Output sink current	$Isink_{PWM}$	Vout,L = 0.8V		20		mA
Output settling time	Tset	100 pF capacitive load, full operating Ta range		500	TBD	ns
Output settling time	$Tset_{RC}$	220 Ohm in series with 47nF load on the wire, full Ta operating range	20		50	us

Parameter	Symbol	Test Conditions	Min	Typ	Max	Units
SMBus compatible 2-wire interface[2]						
Input high voltage	V_{IH}		1.8	2	2.2	V
Input high voltage	V_{IH}(Ta, V)	Over temperature and supply	1.6		2.4	V
Input low voltage	V_{IL}		0.7	1.0	1.3	V
Input low voltage	V_{IL}(Ta, V)	Over temperature and supply	0.5		1.5	V
Output low voltage	V_{OL}	SDA pin in open drain mode, over temperature and supply, Isink = 2mA			0.2	V
SCL leakage	I_{SCL},leak	V_{SCL}=4V, Ta=+85°C			30	uA
SDA leakage	I_{SDA},leak	V_{SDA}=4V, Ta=+85°C			0.3	uA
SCL capacitance	C_{SCL}				10	pF
SDA capacitance	C_{SDA}				10	pF
Slave address	SA	Factory default		5Ah		hex
SMBus Request	t_{REQ}	SCL low	1.024			ms
Timeout, low	$T_{imeout,L}$	SCL low			30	ms
Timeout, high	$T_{imeout,H}$	SCL high			50	us
Acknowledge setup time	Tsuac(MD)	8-th SCL falling edge, Master	0.5		1.5	us
Acknowledge hold time	Thdac(MD)	9-th SCL falling edge, Master	1.5		2.5	us
Acknowledge setup time	Tsuac(SD)	8-th SCL falling edge, Slave	2.5			us
Acknowledge hold time	Thdac(SD)	9-th SCL falling edge, Slave	1.5			us
EEPROM						
Data retention		Ta = +85°C	10			years
Erase/write cycles		Ta = +25°C	100,000			Times
Erase/write cycles		Ta = +125°C	10,000			Times
Erase cell time	Terase			5		ms
Write cell time	Twrite			5		ms

Notes: All the communication and refresh rate timings are given for the nominal calibrated HFO frequency and will vary with this frequency's variations.
1. All PWM timing specifications are given for single PWM output (factory default for MLX90614xAx). For the extended PWM output (factory default for the MLX90614xBx) each period has twice the timing specifications (refer to the PWM detailed description section). With large capacitive load lower PWM frequency is recommended. Thermal relay output (when configured) has the PWM DC specification and can be programmed as push-pull, or NMOS open drain. PWM is free-running, power-up factory default is SMBus, refer to 7.6, "Switching between PWM and SMBus communication" for details..
2. For SMBus compatible interface on 12V application refer to Application information section. SMBus compatible interface is described in details in the SMBus detailed description section. Maximum number of MLX90614xxx devices on one bus is 127, higher pullup currents are recommended for higher number of devices, faster bus data transfer rates, and increased reactive loading of the bus.
MLX90614xxx is always a slave device on the bus. MLX90614xxx can work in both low-power and high-power SMBus communication.
All voltage are with respect to the Vss (ground) unless otherwise noted.

Power saving mode is not available on the 5V version (MLX90614Axx).

6.2 MLX90614Bxx

All parameters are preliminary for T_A = 25 ˚C, $\mathbf{V_{DD}}$ **=3V** (unless otherwise specified)

Parameter	Symbol	Test Conditions	Min	Typ	Max	Units
Supplies						
External supply	V_{DD}		2.4	3	3.6	V
Supply current	I_{DD}	No load			1	mA
Supply current (programming)	I_{DDpr}	No load, erase/write EEPROM operations			1.5	mA
Power-down supply current	Isleep	no load	1	2.5	5	uA
Power-down supply current	Isleep	Full temperature range	1	2.5	6	uA
Power On Reset						
POR level	V_{POR}	Power-up, power-down and brown-out	1.6	1.85	2.1	V
V_{DD} rise time	T_{POR}	Ensure POR signal			1	ms
Output valid	Tvalid	After POR		0.15		s
Pulse width modulation[1]						
PWM resolution	PWMres	Data band		10		bit
PWM output period	$PWM_{T,def}$	Factory default, internal oscillator factory calibrated		1.024		ms
PWM period stability	$dPWM_T$	Internal oscillator factory calibrated, over the entire operation range and supply voltage	-4		+4	%
Output high Level	PWM_{HI}	I_{source} = 2 mA	V_{DD}-0.25			V
Output low Level	PWM_{LO}	I_{sink} = 2 mA			V_{SS}+0.25	V
Output drive current	$Idrive_{PWM}$	Vout,H = V_{DD} - 0.8V		15		mA
Output sink current	$Isink_{PWM}$	Vout,L = 0.8V		15		mA
Output settling time	Tset	100 pF capacitive load, full operating Ta range			150	ns
Output settling time	$Tset_{RC}$	220 Ohm in series with 47nF load on the wire, full Ta operating range		500	TBD	ns

Parameter	Symbol	Test Conditions	Min	Typ	Max	Units
SMBus compatible 2-wire interface[2]						
Input high voltage	V_{IH}		1.6	2	2.4	V
Input high voltage	V_{IH}(Ta,V)	Over temperature and supply	1.2	2	2.8	V
Input low voltage	V_{IL}		0.7	1.0	1.3	V
Input low voltage	V_{IL}(Ta,V)	Over temperature and supply	0.5	1.0	1.5	V
Output low voltage	V_{OL}	SDA pin in open drain mode, over temperature and supply, Isink = 2mA			0.25	V
SCL leakage	I_{SCL},leak	V_{SCL}=3V, Ta=+85℃			20	uA
SDA leakage	I_{SDA},leak	V_{SDA}=3V, Ta=+85℃			0.25	uA
SCL capacitance	C_{SCL}				10	pF
SDA capacitance	C_{SDA}				10	pF
Slave address	SA	Factory default		5Ah		hex
SMBus Request	t_{REQ}	SCL low	1.024			ms
Timeout,low	$T_{imeout,L}$	SCL low			30	ms
Timeout, high	$T_{imeout,H}$	SCL high			50	us
Acknowledge setup	Tsuac(MD	8-th SCL falling edge, Master	0.5		1.5	us
Acknowledge hold	Thdac(MD	9-th SCL falling edge, Master	1.5		2.5	us
Acknowledge setup	Tsuac(SD)	8-th SCL falling edge, Slave	2.5			us
Acknowledge hold	Thdac(SD	9-th SCL falling edge, Slave	1.5			us
EEPROM						
Data retention		Ta = +85℃	10			years
Erase/write cycles		Ta = +25℃	100,000			Times
Erase/write cycles		Ta = +125℃	10,000			Times
Erase cell time	Terase			5		ms
Write cell time	Twrite			5		ms

Note: refer to MLX90614Axx notes.

7 Detailed description

7.1 Block diagram

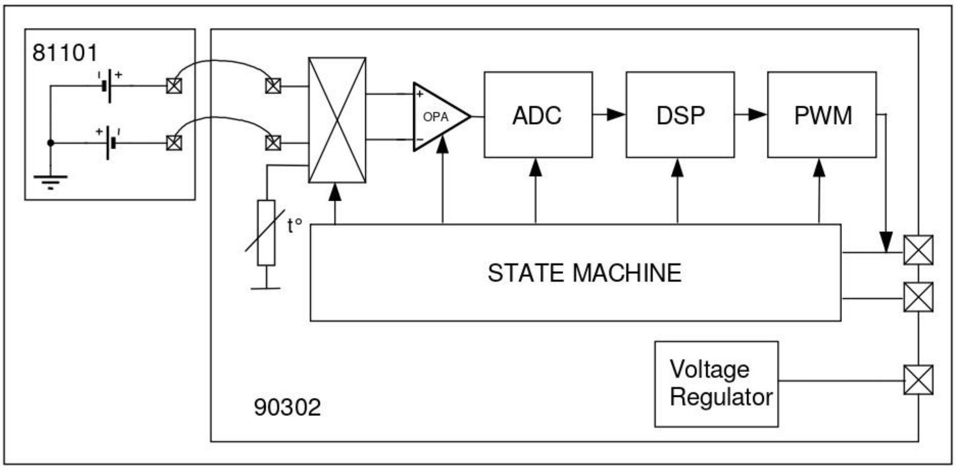


Figure 3: block diagram

7.2 Signal processing principle

The operation of the MLX90614 is controlled by an internal state machine, which controls the measurements and calculations of the object and ambient temperatures and does the post-processing of the temperatures to output them through the PWM output or the SMBus compatible interface.
The ASSP supports 2 IR sensors (second one not implemented in the MLX90614xAx).The output of the IR sensors is amplified by a low noise low offset chopper amplifier with programmable gain, converted by a Sigma Delta modulator to a single bit stream and fed to a powerful DSP for further processing. The signal is treated by programmable (by means of EEPROM contend) FIR and IIR low pass filters for further reduction of the band width of the input signal to achieve the desired noise performance and refresh rate. The output of the IIR filter is the measurement result and is available in the internal RAM. 3 different cells are available: One for the on-board temperature sensor (on chip PTAT or PTC) and 2 for the IR sensors.
Based on results of the above measurements, the corresponding ambient temperature Ta and object temperatures To are calculated. Both calculated temperatures have a resolution of 0.01 ˚C. The data for Ta and To can be read in two ways: Reading RAM cells dedicated for this purpose via the 2-wire interface (0.02°C resolution, fixed ranges), or through the PWM digital output (10 bit resolution, configurable range).
In the last step of the measurement cycle, the measured Ta and To are rescaled to the desired output resolution of the PWM) and the recalculated data is loaded in the registers of the PWM state machine, which creates a constant frequency with a duty cycle representing the measured data.

7.3 Block description

7.3.1 Amplifier

A low noise low offset amplifier with programmable gain is implemented for amplification of the IR sensor voltage. With a carefully designed input modulator and balanced input impedance, an offset as low as 0.5μV is achieved.

7.3.2 Supply regulator and POR

The module can operate from 2 different supplies:
VDD= 5V => MLX90614Axx
VDD=3.3V => MLX90614Bxx (battery or regulated supply)
Refer to "Applications information" section for information about adopting higher voltage supplies.

The Power On Reset (POR) is connected to Vdd supply. The on-chip POR circuit provides an active (high) level of the POR signal when the Vdd voltage rises above approximately 0.5V and holds the entire MLX90614xxx in reset until the Vdd is higher than the specified POR threshold V_{POR} (note that this level is different for MLX90614Axx and MLX90614Bxx). During the time POR is active, the POR signal is available as an open drain (active high) at the PWM/SDA pin. After the MLX90614xxx exits the POR condition, the function programmed in EEPROM takes precedence for that pin.

7.3.3 EEPROM

A limited number of addresses in the EEPROM memory can be changed by the customer. The whole EEPROM can be read via SMBus interface.

EEPROM (32X16)		
Name	Address	Write acces
To_{max}	000h	Yes
To_{min}	001h	Yes
PWMCTRL	002h	Yes
Ta range	003h	Yes
Ke	004h	Yes
Config Register1	005h	Yes
Melexis reserved	006h	No
...	...	...
Melexis reserved	00Dh	No
SMBus address	00Eh	Yes
Melexis reserved	00Fh	Yes
Melexis reserved	010h	No
...	...	...
Melexis reserved	018	No
Melexis reserved	019h	Yes
Melexis reserved	01Ah	No
Melexis reserved	01Bh	No
ID number	01Ch	No
ID number	01Dh	No
ID number	01Eh	No
ID number	01Fh	No

The addresses To_{max}, To_{min} and Ta range are for customer dependent object and ambient temperature ranges. For details see point 7.5.3 below in this document

The address **PWMCTRL** consists of control bits for configuring the PWM/SDA pin:

Bit 0	Select the type of PWM mode:	1 - Single PWM, factory default for MLX90614xAx	0 – Extended PWM, factory default for MLX90614xBx
Bit 1	Enable/disable the PWM:	1 - Enable PWM, disable SMBus	0 - Disable PWM (Enable SMBus), Factory default
Bit 2	Configuration of the pin PWM:	1 - Push-Pull,	0 – OpenDrain NMOS, factory default
Bit 3	Mode selection	1 - ThermoRelay,	0 - PWM, Factory default
Bits[8:4]	Extemded PWM definition	Number of repetitions divided by 2 of sensor 1 and 2 in Extended PWM mode. The number of repetitions can vary from 0 to 64 times.	
Bits[15:9]	PWM clock configuration	2MHz divided by number written in this place. (128 in case the number is 0.) A single PWM period consists of 2048 clocks and extended PWM of 4096 clocks for each period (2T in figure 6). The 2 MHz clock is valid for the nominal HFO frequency.	

The address **ConfigRegister1** consist of control bits for configuring the analog and digital parts:

<table>
<tr><td>Bits[2:0]</td><td>– Configure coefficients of IIR digital filter:</td><td colspan="2">
<table>
<tr><th>Bit 2</th><th>Bit 1</th><th>Bit 0</th><th>aa_1</th><th>b_1</th></tr>
<tr><td>0</td><td>x</td><td>x</td><td>0.5</td><td>0.5</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>0.571428571</td><td>0.428571428</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0.666(6)</td><td>0.333(3)</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>0.8</td><td>0.2</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td><td>0 (IIR bypassed)</td></tr>
</table>
</td></tr>
<tr><td>Bit 3</td><td>– Configure the type of ambient temperature sensor:</td><td>1 - PTC,</td><td>0 – PTAT.</td></tr>
<tr><td>Bits[5:4]</td><td>– Configure the type of data transmitted through PWM:</td><td colspan="2">
<table>
<tr><th>Bit 5</th><th>Bit 4</th><th>Data 1</th><th>Data 2</th></tr>
<tr><td>0</td><td>0</td><td>Ta</td><td>IR 1</td></tr>
<tr><td>0</td><td>1</td><td>Ta</td><td>IR 2</td></tr>
<tr><td>1</td><td>1</td><td>IR 1</td><td>IR 2</td></tr>
<tr><td>1</td><td>0</td><td>IR 2</td><td>Undefined*</td></tr>
</table>
</td></tr>
<tr><td>Bit 6</td><td>– Define the number IR sensors:</td><td>1 – 2 sensors,</td><td>0 -1 sensor.</td></tr>
<tr><td>Bit 7</td><td>– Define the sign Ks (Ks=dAlpha/dTobj) :</td><td colspan="2">Factory calibration, do not alter</td></tr>
<tr><td>Bits[10:8]</td><td>– Configure coefficient N of FIR digital filter:</td><td colspan="2">
<table>
<tr><th>Bit 10</th><th>Bit 9</th><th>Bit 8</th><th>N</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>8</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>16</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>32</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>64</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>128</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>256</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>512</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>1024</td></tr>
</table>
</td></tr>
<tr><td>Bits[13:11]</td><td>– Configure the gain of amplifier:</td><td colspan="2">
<table>
<tr><th>Bit 13</th><th>Bit 12</th><th>Bit 11</th><th>Gain</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1 (preamplifier bypassed)</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>3</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>6</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>12.5</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>25</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>50</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>100</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>100</td></tr>
</table>
</td></tr>
<tr><td>Bit 14</td><td colspan="3">Unused</td></tr>
<tr><td>Bit 15</td><td>– Define the sign of thermo-shock compensation:</td><td>1 - negative,</td><td>0 – positive.</td></tr>
</table>

Note: The following bits/registers should not be altered (except with special tools – contact Melexis for such tools availability) in order to keep the factory calibration relevant:
Ke [15..0] ; Config Register1 [13..11;7;3] ; addresses 00Fh and 019h.
** not recommended for extended PWM mode*

7.3.4 RAM

It is not possible to write into the RAM memory. It can only be read and only a limited number of RAM registers are of interest to the customer.

RAM (32x17)		
Name	Address	Read access
Melexis reserved	**000h**	**Yes**
…	**…**	**…**
Melexis reserved	**005h**	**Yes**
T_A	**006h**	**Yes**
T_{OBJ1}	**007h**	**Yes**
T_{OBJ2}	**008h**	**Yes**
Melexis reserved	**009h**	**Yes**
…	**…**	**…**
Melexis reserved	**01Fh**	**Yes**

7.4 SMBus compatible 2-wire protocol

The chip supports a 2 wires serial protocol, build with pins PWM/SDA and SCL.

- SCL – digital input, used as the clock for SMBus compatible communication. This pin has the auxiliary function for building an external voltage regulator. When the external voltage regulator is used, the 2-wire protocol is available only if the power supply regulator is overdriven.
- PWM/SDA – Digital input/output, used for both the PWM output of the measured object temperature(s) or the digital input/output for the SMBus. The pin can be programmed in EEPROM to operate as Push/Pull or open drain NMOS (open drain NMOS is factory default).

7.4.1 Functional description

The SMBus interface is a 2-wire protocol, allowing communication between the Master Device (MD) and one or more Slave Devices (SD). In the system only one master can be presented at any given time [1]. The MLX90614 can only be used as a slave device.

Generally, the MD initiates the start of data transfer by selecting a SD through the Slave Address (SA).

The MD has read access to the RAM and EEPROM and write access to 9 EEPROM cells (at addresses 0x20h, 0x21h, 0x22h, 0x23h, 0x24h, 0x25h*, 0x2Eh, 0x2Fh, 0x39h). If the access to the MLX90614 is a read operation it will respond with 16 data bits and 8 bit PEC only if its own slave address, programmed in internal EEPROM, is equal to the SA, sent by the master. The SA feature allows connecting up to 127 devices with only 2 wires, unless the system has some of the specific features described in paragraph 5.2 of reference [1]. In order to provide access to any device or to assign an address to a SD before it is connected to the bus system, the communication must start with zero SA followed by low RWB bit. When this command is sent from the MD, the MLX90614 will always respond and will ignore the internal chip code information.
Special care must be taken not to put two MLX90614 devices with the same SD addresses on the same bus as MLX90614 does not support ARP[1].
The MD can force the MLX90614 into low consumption mode “sleep mode” (3V version only).
Read flags like “EEBUSY” (1 – EEPROM is busy with executing the previous write/erase), “EE_DEAD” (1 – there is fatal EEPROM error and this chip is not functional**).
Note: This address is readable and writable. Bit 3 should not be altered as this will cancel the factory calibration.*
*Note**: EEPROM error signalling is implemented in automotive grade parts only.*

7.4.2 Differences with the standard SMBus specification (reference [1])

There are eleven command protocols for standard SMBus interface. The MLX90614 supports only two of them. Not supported commands are:

- Quick Command
- Byte commands - Sent Byte, Receive Byte, Write Byte and Read Byte
- Process Call
- Block commands – Block Write and Write-Block Read Process Call

Supported commands are:

- Read Word
- Write Word

7.4.3 Detailed description

The PWM/SDA pin of MLX90614 can operate also as PWM output, depending on the EEPROM settings. If PWM is enabled, after POR the PWM/SDA pin is directly configured as PWM output. The PWM mode can be avoided and the pin can be restored to its Data function by a special command. That is why hereafter both modes are treated separately.

7.4.3.1 Bus Protocol

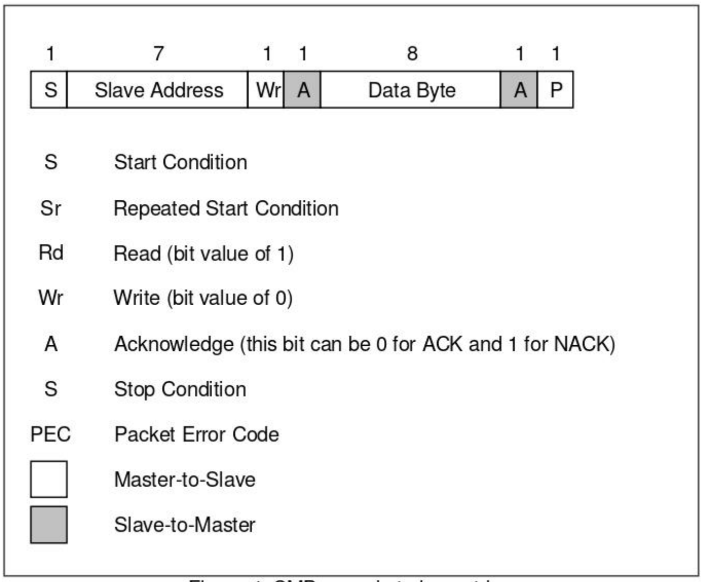


Figure 4: SMBus packet element key

After every 8 bits received by the SD an ACK/NACK takes place. When a MD initiates communication, it first sends the address of the slave and only the SD which recognizes the address will ACK the rest will remain silent. If the SD NACKs one of the bytes, the MD should stop the communication and repeat the message. A NACK could be received after the PEC. This means that there is error in the received message and the MD should try sending the message again. The PEC calculation includes all bits except the START, REPEATED START, STOP, ACK, and NACK bits. The PEC is a CRC-8 with polynomial X8+X2+X1+1. The Most Significant Bit of every byte is transferred first.

7.4.3.1.1 Read Word (depending on the command – RAM or EEPROM)

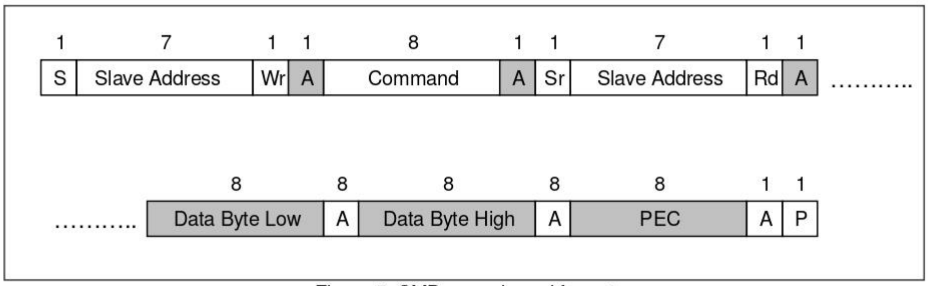


Figure 5: SMBus read word format

7.4.3.1.2 Write Word (depending on the command – RAM or EEPROM)

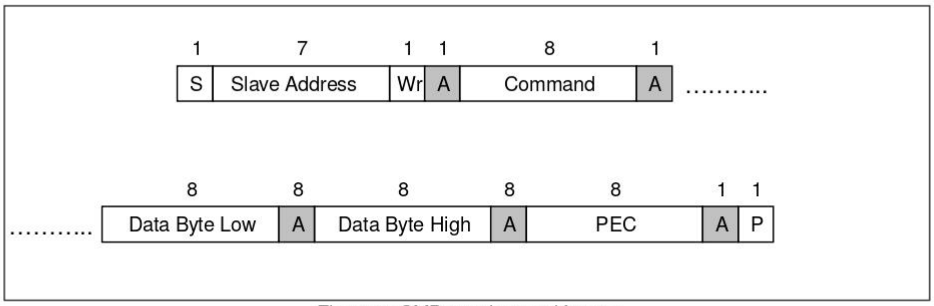


Figure 6: SMBus write word format

7.4.4 AC specification for SMBus

7.4.4.1 Timing

The MLX90614 meets all the timing specifications of the SMBus [1]. The maximum frequency of the MLX90614 SMBus is 100KHz and the minimum is 10KHz.
The specific timings in MLX90614's SMBus are:
SMBus Request (t_{REQ}) is the time that the SCL should be forced low in order to switch MLX90614 from PWM mode to SMBus mode;
Timeout L is the maximum allowed time for SCL to be low. After this time the MLX90614 will reset its communication block and will be ready for new communication;
Timeout H is the maximum time for which it is allowed for SCL to be high during communication. After this time MLX90614 will reset its communication block assuming that the bus is idle (according to the SMBus specification).
Tsuac(SD) is the time after the eighth falling edge of SCL that MLX90614 will force PWM/SDA low to acknowledge the last received byte.
Thdac(SD) is the time after the ninth falling edge of SCL that MLX90614 will release the PWM/SDA (so the MD can continue with the communication).
Tsuac(MD) is the time after the eighth falling edge of SCL that MLX90614 will release PWM/SDA (so that the MD can acknowledge the last received byte).
Thdac(MD) is the time after the ninth falling edge of SCL that MLX90614 will take control of the PWM/SDA (so it can continue with the next byte to transmit).

The indexes MD and SD for the latest timings are used – MD when the master device is making acknowledge; SD when the slave device is making acknowledge). For other timings see [1].

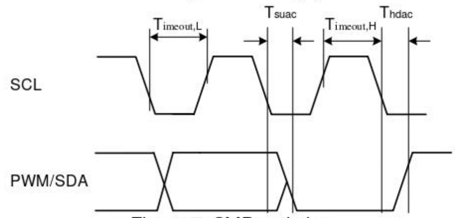


Figure 7: SMBus timing

7.4.5 Bit transfer

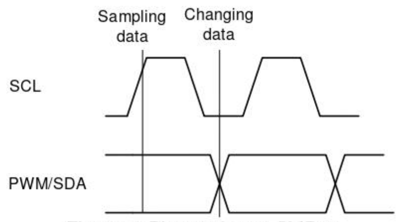


Figure 8: Bit transfer on SMBus

The data on PWM/SDA must be changed when SCL is low (min 300ns after the falling edge of SCL). The data is fetched by both MD and SDs on the rising edge of the SCL.

7.4.6 Commands

In application mode RAM and EEPROM can be read both with 32x16 sizes. If the RAM is read, the data are divided by two, due to a sign bit in RAM (for example, T_{OBJ1} - RAM address 0x07h will sweep between 0x27ADh to 0x7FFF as the object temperature rises from -70.01℃ to +382.19℃). The MSB read from RAM is an error flag (active high) for the linearized temperatures (T_{OBJ1}, T_{OBJ2} and T_a). The MSB for the raw data (e.g. IR sensor1 data) is a sign bit (sign and magnitude format).

Opcode	Command
000x xxxx*	RAM Access
001x xxxx*	EEPROM Access
1111_0000**	Read Flags
1111_1111	Enter SLEEP mode

Note: The xxxxx are the 5 LSBits of the memory map address to be read/written.*
*Note**: Behaves like read command. The MLX90614 returns PEC after 16 bits data of which only 4 are meaningful and if the MD wants it, it can stop the communication after the first byte. The difference between read and read flags is that the latter does not have a repeated start bit.*
Flags read are:
- Data[15] – EEBUSY – the previous write/erase EEPROM access is still in progress. High active.
- Data[14] – Unused
- Data[13] - EE_DEAD – EEPROM double error has occurred. High active.
- Data[12] – INIT – POR initialization routine is still ongoing. High active.
- Data[11] – not implemented..
- Data[10..0] – all zeros.

Flags read is a diagnostic feature. The MLX90614 can be used regardless of these flags.

7.4.7 Sleep Mode

MLX90614 can enter Sleep Mode via command "Enter SLEEP mode" sent via the SMBus interface. This mode is not available for the 5V supply version. To limit the current consumption to 2.5uA (typ), the SCL pin should be kept low during sleep. MLX90614 goes back into power-up default mode (via POR reset) by setting SCL pin high and then PWM/SDA pin low for at least t_{DDq}=13ms. **If EEPROM is configured for PWM (EN_PWM is high), the PWM interface will be selected after awakening and if PWM control [2], PPODB is 1 the MLX90614 will output a PWM pulse train with push-pull output.**

7.4.7.1 Enter Sleep Mode

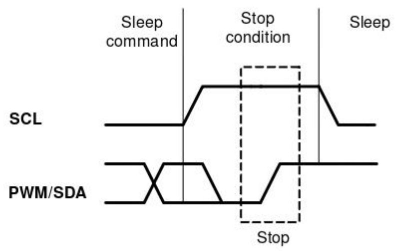


Figure 9: Enter sleep

7.4.7.2 Exit from Sleep Mode

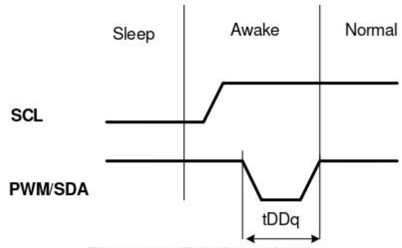


Figure 10: Exit Sleep Mode

7.5 PWM

The MLX90614 can be read via PWM or SMBus compatible interface. Selection of PWM output is done in EEPROM configuration (factory default is PWM). PWM output has two programmable formats, single and dual data transmission, providing single wire reading of two temperatures (dual zone object or object and ambient). The PWM period is derived from the on-chip oscillator and is programmable.

Config Register[5:4]	PWM1 data	PWM2 data	*Tmin,1*	*Tmax,1*	*Tmin,2*	*Tmax,2*
00	Ta	T_{obj1}	Ta_{range},L	Ta_{range},H	To_{min}	To_{max}
01	Ta	T_{obj2}	Ta_{range},L	Ta_{range},H	To_{min}	To_{max}
11	T_{obj1}	T_{obj2}	To_{min}	To_{max}	To_{min}	To_{max}
10*	T_{obj2}	Undefined	To_{min}	To_{max}	*N.A.*	*N.A.*

Note: Serial data functions (2-wire / PWM) are multiplexed with a thermal relay function (described in the "Thermal relay" section).
** not recommended for extended PWM format operation*

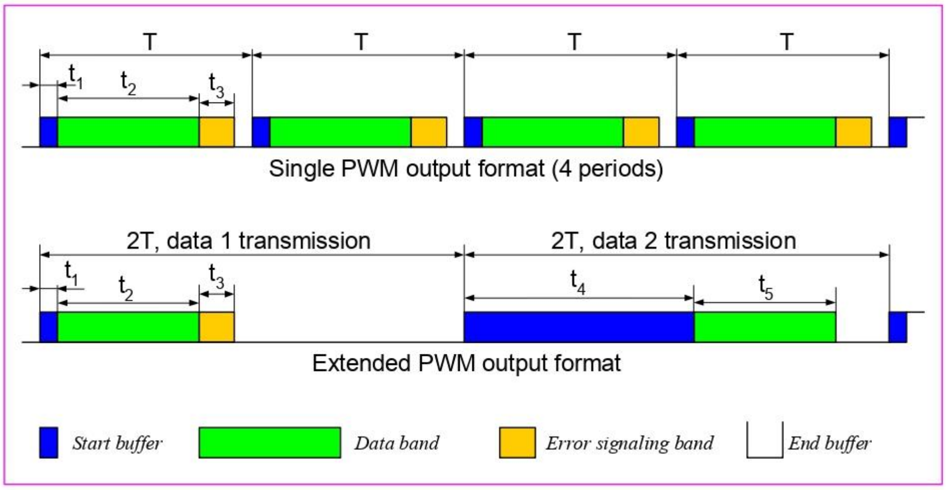


Figure 11: PWM timing

7.5.1 Single PWM format

In single PWM output mode the settings for PWM1 data only are used. The temperature reading can be calculated from the signal timing as:

$$Tout = \left[\frac{2t_2}{T} * (Tmax - Tmin) \right] + Tmin \,,$$

where Tmin and Tmax are the corresponding rescale coefficients in EEPROM for the selected temperature output (Ta, object temperature range is valid for both Tobj1 and Tobj2 as specified in the previous table) and T is the PWM period. Tout is T_{obj1}, T_{obj2} or T_a according to Config Register [5:4] settings.

The different time intervals t_1-t_3 have the following functions:
t_1: Start buffer. During this time the signal is always high. t_1 = 0.125*T (T is the PWM period, refer to fig. 11).
t_2: Valid Data Output Band, 0 to 1/2T. PWM output data resolution is 10 bit.
t_3: Error band – information for Fatal error in EEPROM (double error detected, not correctable). t_3 = 0.25 * T.
Therefore a PWM pulse train with a duty cycle of 0.875 will indicate a fatal error in EEPROM (for single PWM format).

Example:
T_{obj1} => Config Reg[5:4] = 11'b
To_{min} = 0°C => To_{min} [EEPROM] = 100 * (to_{min} + 273.15) = 6AB3h
To_{max} = +50°C => To_{max} [EEPROM] = 100 * (to_{max} + 273.15) = 7E3Bh
Captured PWM high duration is 0.495*T => t_2=(0.495 – 0.125)*T=0.370*T =>
measured object temperature = 2X0.370* (50°C -0°C)+0°C = +37.0°C.

7.5.2 Extended PWM format

The PWM format for extended PWM is shown in Figure 11. Note that with bits DUAL[5:1]>00h each period will be repeated 2N+1 times, where N is the decimal value of the number written in DUAL[5:1] (*DUAL[5:1] =PWM control & clock [8:4]*), like shown on Figure 12.

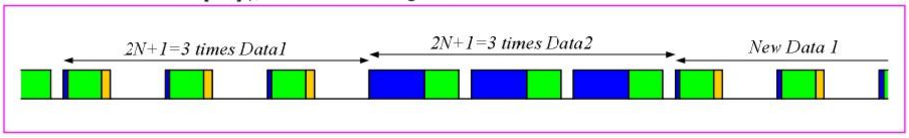


Figure 12: Extended PWM format with DUAL[5:1] = 01h (2 repetitions for each data)

The temperature transmitted in Data 1 field can be calculated using the following equation:

$$Tout_1 = \left[\frac{2t_2}{T} * (Tmax_1 - Tmin_1)\right] + Tmin_1$$

For Data 2 field the equation is:

$$Tout_2 = \left[\frac{2t_5}{T} * (Tmax_2 - Tmin_2)\right] + Tmin_2$$

Where $Tmin_1$,$Tmax_1$,$Tmin_2$ and $Tmax_2$ are given in Table 9, t_2=t_{high1}-t_1, and t_5=t_{high2}-t_4.
Time bands are: t_1=0.125*T, t_3=0.25*T and t_4=1.125*T. As shown in Figure 11, in extended PWM format the period is twice the period for the single PWM format. All equations provided herein are given for the single PWM period T. The EEPROM Error band signalling will be 43.75% duty cycle for Data1 and 93.75% for Data2.
Note:EEPROM error signalling is implemented in automotive grade parts only.

Example:
Configuration: Ta : T_{obj1} @ Data1 : Data2 => Config Reg[5:4] = 00b,
Ta_{min} = -5°C => Ta_{range}, L [EEPROM] = 100*(Ta_{min}+38.2)/64 = 33h,
Ta_{max} = +105°C => Ta_{range},H [EEPROM] = 100*(Ta_{max}+38.2)/64 = DFh,
Ta_{range} [EEPROM]=DF33h
To_{min} = 0°C => To_{min} [EEPROM] = 100 * (To_{min} + 273.15)/64 = 6AB3h
To_{max} = +50°C => To_{max} [EEPROM] = 100 * (To_{max} + 273.15)/64 = 7E3Bh
Captured high durations are 0.13068*(2T) and 0.7475*(2T), where 2T is each captured PWM period. Time band t_4 is provided for reliable determination between Data1 and Data2 data fields. Thus Data1 is represented by 0.13068*(2T) and Data2 – by 0.7475*(2T), and the temperatures can be calculated as follows:
t_2/T=(t_{high1}/T)-0.125=0.13636 => Ta=+25.0°C,
t_5/T=(t_{high2}/T)-1.125=0.370 => T_{obj1}=+37.0°C.

7.5.3 Customizing the temperature range for PWM output

The calculated ambient and object temperatures are stored in RAM with a resolution of 0.01 °C (16 bit). The PWM operates with a 10-bit word so the transmitted temperature is rescaled in order to fit in the desired range.
For this goal 2 cells in EEPROM are foreseen to store the desired range for To (To_{min} and To_{max}) and one for Ta (Ta_{range}: the 8MSB are foreseen for Ta_{max} and the 8LSB for Ta_{min}).
Thus the output range for To can be programmed with an accuracy of 0.01 °C, while the corresponding Ta range can be programmed with an accuracy of 2.56 °C.

The **object** data for PWM is rescaled according to the following equation:

$$T_{PWM_{obj}} = \frac{T_{RAM} - T_{MIN_{EEPROM}}}{K_{PWM\,obj}}, K_{PWM\,obj} = \frac{T_{MAX_{EEPROM}} - T_{MIN_{EEPROM}}}{1023}$$

The T_{RAM} is the linearised Tobj, 16-bit (0000…FFFFh, 0000 for -273.15°C and FFFFh for +382.2°C) and the result is a 10-bit word, in which 000h corresponds to To_{MIN}[°C], 3FFh corresponds to To_{MAX}[°C] and 1LSB corresponds to $\frac{To_{MAX} - To_{MIN}}{1023}$ [°C]

$$T_{MIN_{EEPORM}} = T_{MIN} * 100 \text{ LSB}$$

$$T_{MAX_{EEPORM}} = T_{MAX} * 100 \text{ LSB}$$

The **ambient** data for PWM is rescaled according to the following equation:

$$T_{PWM_{ambient}} = \frac{T_{RAM} - T_{MIN_{EEPROM}}}{K_{PWM_{ambient}}}, K_{PWM\,ambient} = \frac{T_{MAX_{EEPROM}} - T_{MIN_{EEPROM}}}{1023}$$

The result is a 10-bit word, where 000h corresponds to -38.2 °C (lowest Ta that can be read via PWM), 3FFh corresponds to 125 °C (highest Ta that can be read via PWM) and 1LSB corresponds to $\frac{T_{MAX} - T_{MIN}}{1023}$ [°C]

$$T_{MIN_{EEPORM}} = [T_{MIN} - (-38.2)] * \frac{100}{64} \text{ LSB}$$

$$T_{MAX_{EEPORM}} = [T_{MAX} - (-38.2)] * \frac{100}{64} \text{ LSB}$$

7.6 Switching Between PWM and SMBus communication

7.6.1 PWM is enabled

The diagram below illustrates the way of switching to SMBus if PWM is enabled (factory programmed POR default for MLX90614 is SMBus, PWM enabled). Note that the SCL pin needs to be kept high in order to use PWM.

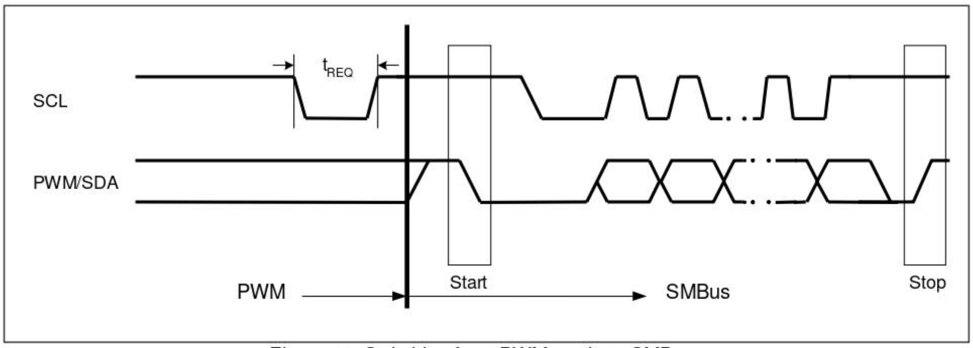


Figure 13: Switching from PWM mode to SMBus

7.6.2 Request condition

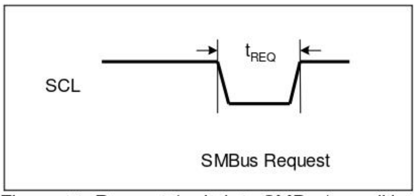


Figure 14: Request (switch to SMBus) condition

If PWM is enabled, the MLX90614's SMBus Request condition is needed to disable PWM and reconfigure PWM/SDA pin before starting SMBus communication. Once disabled PWM, it can be only enabled by switching Off-On of the supply or exit from Sleep Mode. The MLX90614's SMBus request condition requires forcing LOW the SCL pin for period longer than the request time (t_{REQ}). The SDA line value is ignored in this case.

7.6.3 PWM is disabled

If PWM is disabled by means of EEPROM the PWM/SDA pin is directly used for the SMBus purposes after POR. **Request condition should not be sent in this case**.

7.7 Computation of ambient and object temperatures

The IR sensor consists of serial connected thermo-couples with cold junctions placed at thick chip substrate and hot junctions, placed over thin membrane. The IR radiation absorbed from the membrane heats (or cools) it. The thermopile output signal is

$$V_{ir}(Ta, To) = A.(To^4 - Ta^4),$$

Where To is the object temperature absolute (Kelvin) temperature, Ta is the sensor die absolute (Kelvin) temperature, and A is the overall sensitivity.
An additional temperature sensor is needed for measuring the temperature of the chip temperature. After measurement of the output of both sensors, the corresponding ambient and object temperatures can be calculated. These calculations are done by the internal DSP, which produces digital outputs, linearly proportional to measured temperatures.

7.7.1 Ambient temperature Ta

The Sensor die temperature is measured with a PTC or a PTAT element. All the sensors' conditioning and data processing is handled on-chip and the linearized sensor die temperatureTa is made available in memory.
The resolution of the calculated Ta is 0.01 ˚C. The sensor is factory calibrated for the full automotive range (-40 to 125 ˚C). In RAM cell ,006h, 0000h corresponds to -40 ˚C and 4074h (16500d) corresponds to 125 ˚C. The conversions from RAM contend to real Ta is easy using the following relation:

$Ta[°K] = Tareg \times 0.01$ Note that via SMBus Ta is read divided by 2, or Ta,SMBus[°K]=Tareg x 0.02

7.7.2 Object temperature To

The result has a resolution of 0.01 ˚C and is available in RAM. To is derived from RAM as:

$To[°K] = Toreg \times 0.01$ Note that via SMBus To is read divided by 2, or To,SMBus[°K]=Toreg x 0.02

7.7.3 Calculation flow

The measurement, calculation and linearization are held by core, which executes a program form ROM. After POR the chip is initialized with calibration data from EEPROM. During this phase the number of IR sensor is selected and which temperature sensor will be used. Measurements, compensation and linearization routines run in a closed loop afterwards.

Processing ambient temperature includes:
- Offset measurement with fixed length FIR filter
- Additional filtering with fixe length IIR filter. The result is stored into RAM as T_{OS}
- Temperature sensor measurement using programmable length FIR *.
- Offset compensation
- Additional processing with programmable length IIR **. The result is stored into RAM as T_D.
- Calculation of the ambient temperature. The result is stored into RAM as T_A

Processing of the object temperature consists of three parts. The first one is common for both IR sensors, the third part can be skipped if only one IR sensor is used.
- IR offset:
  - Offset measurement with a fixed length FIR
  - Additional filtering with a fixed length IIR. The result is stored into RAM as IR_{OS}.

Gain measurement with fixed length FIR filter
Offset compensation
Additional gain filtering with fixed length IIR, storing the result into RAM as IR_G.
Gain compensation calculation, the result is stored into RAM as K_G

Object temperature:

IR1 sensor:

IR sensor measurement with programmable length FIR filter *.
Offset compensation
Gain compensation
Filtering with programmable length IIR filter**, storing the result into RAM as $IR1_D$.
Calculation of the object temperature. The result is available in RAM as T_{OBJ1}.

IR2 sensor:

IR sensor measurement with programmable length FIR filter *.
Offset compensation
Gain compensation
Filtering with programmable length IIR filter**, storing the result into RAM as $IR2_D$
Calculation of the object temperature. The result is available in RAM as T_{OBJ2}.

PWM calculation:

Recalculate the data for PWM with 10 bit resolution
Load data into PWM module

Note: The measurements with programmable filter length for FIR filter use the same EEPROM's sell for N.*
*Note**: The IIR filter with programmable filter length uses the same EEPROM's sell for L.*

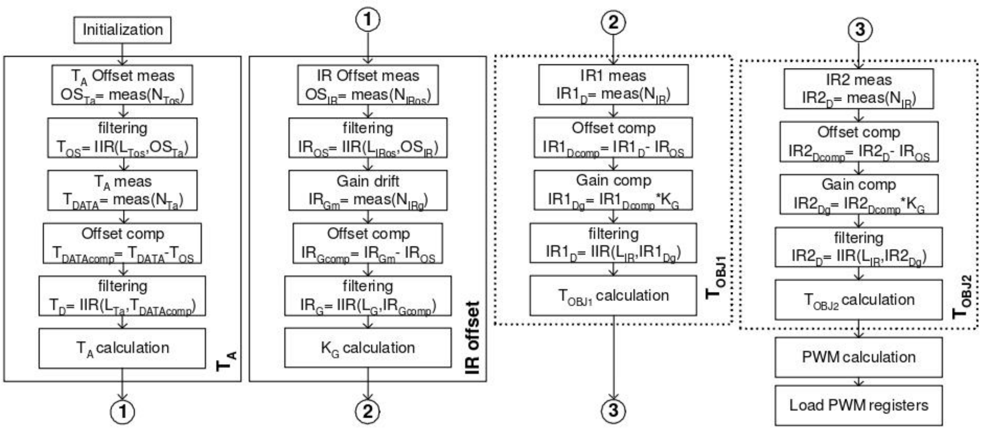


Figure 15: Software flow

7.8 Thermal relay

The MLX90614 can be configured to behave as a thermo relay with programmable threshold and hysteresis on the PWM/SDA pin. The input for the comparator unit of the relay is the object temperature from sensor 1
The output of the MLX90614 is NOT a relay driver but a logical output which should be connected to a relay driver if necessary.
In order to configure the MLX90614 to work as thermal relay two conditions must be met:

- Set bit TRPWMB high at address 002h in EEPROM
- Enable PWM output i.e. EN_PWM is set high

The PWM/SDA pin can be programmed as a push-pull or open drain NMOS (via bit PPODB in EEPROM PWMCTRL), which can trigger an external device. The temperature threshold data is determined by EEPORM at address 021h (Tomin) and the hysteresis at address 020h (To_{max}).
The logical state of the PWM/SDA pin is as follows:

PWM/SDA pin is high if $T_{obj}1 \geq threshold + hysteresis$

PWM/SDA pin is low if $T_{obj}1 \leq threshold - hysteresis$

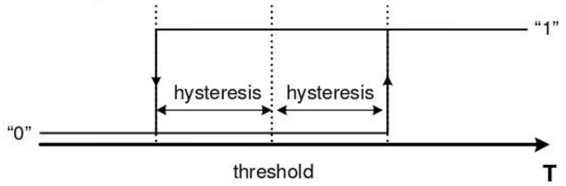


Figure 16: Thermal relay : "PWM" pin versus Tobj

The MLX90614 preserves its normal operation when configured as a thermal relay (PWM configuration and specification applies as a general rule also for the thermal relay) and therefore it can be read using the SMBus (entering the SMBus mode from both PWM and thermal relay configuration is the same).
For example, the MLX90614 can generate a wake-up alert for a system upon reaching a certain temperature and then be read as a thermometer. A reset condition (enter-and exit Sleep, for example) will be needed in order to return to the thermal relay configuration.

Example: threshold 5 ℃ => (5 + 273.15)*100 = 27815 = 6CA7h
hysteresis is 1℃ => 1 * 100 = 100 = 64h
PWM/SDA pin will be low at object temperature below 4 ℃
PWM/SDA pin will be high at object temperature higher that 6 ℃

8 Unique Features

The MLX90614 is a ready-to use low-cost non contact thermometer provided from Melexis with output data linearly dependent on the object temperature with high accuracy and extended resolution.
It supports versatile customization to a very wide range of temperatures, power supplies and refresh rates.
The user can program the internal object emissivity correction for objects with a low emissivity. An embedded error checking and correction mechanism provides high memory reliability.
The sensors is housed in an industry standard TO39 package for both single- and dual-zone IR thermometers. The thermometer is available in automotive grade and can use two different packages for wider applications' coverage.
The low power consumption and sleep mode make the thermometer ideally suited for handheld mobile applications.
The digital sensor interface can be either a power-up-and-measure PWM or an enhanced access SMBus compatible protocol. Systems with more than 100 devices can be built with only two signal lines. Dual zone non contact temperatures measurements available via a single line (extended PWM).
A built-in thermal relay function further extends the easy implementation of wide variety of freezing/boiling prevention and alert systems, as well as thermostats (no MCU is needed).

9 Performance Graphs

9.1 Temperature accuracy of the MLX90601AAA

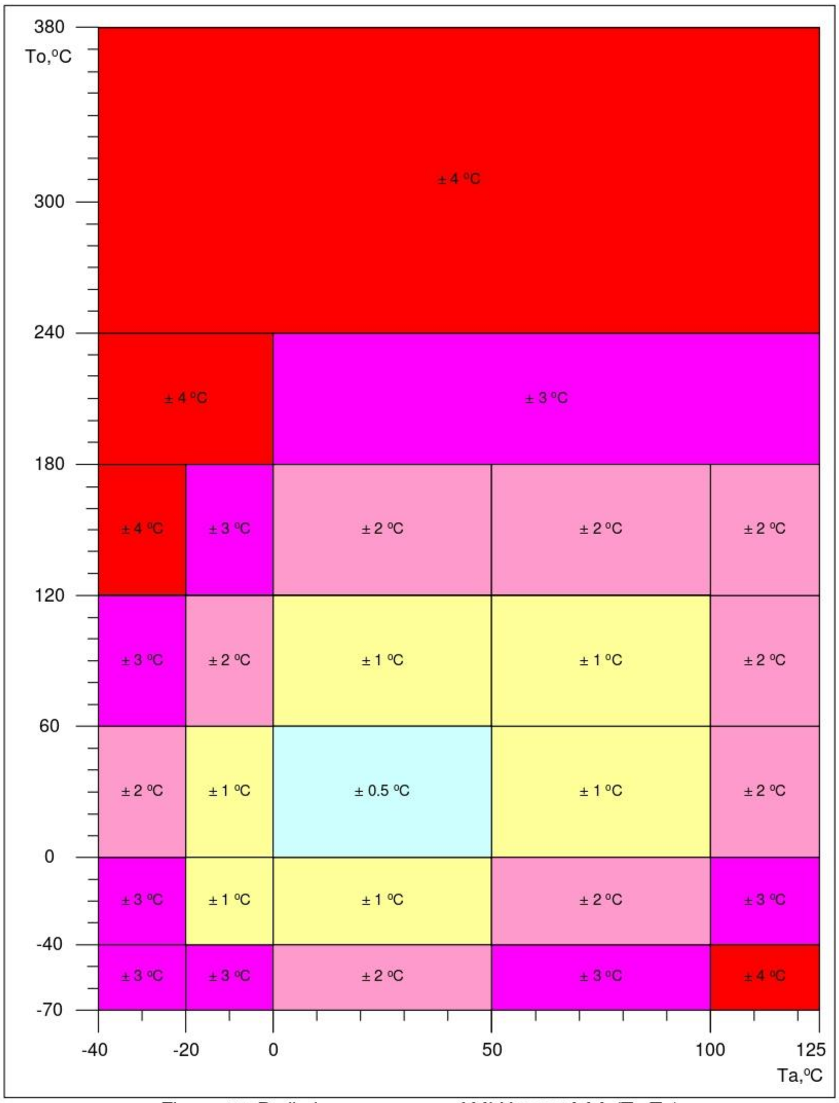


Figure 17: Preliminary accuracy of MLX90601AAA (Ta,To)

All accuracy specifications apply under settled isothermal conditions only.

A version of the MLX90614 with accuracy suited for medical applications is available upon request. The accuracy in the range Ta 10ºC - 40ºC and To 32ºC - 42ºC is shown in diagram below. The accuracy for the rest ranges is same as in previous diagram.

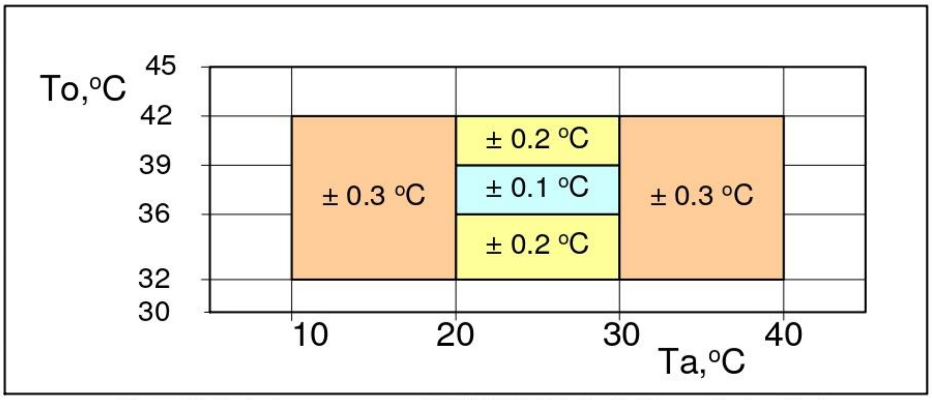


Figure 18: Preliminary accuracy of MLX90601BAA (Ta,To) for medical applications.

9.2 Field Of View (FOV)

Field of view is determined at 50% thermopile signal and with respect to the sensor main axis.

Parameter	MLX90614xAA	MLX90614xAB	MLX90614xBA	MLX90614xBB
Peak zone 1	±0°	±0	-25°	-30°
Width zone 1	72°	80°	70°	70°
Peak zone 2	Not applicable		-25°	+30°
Width zone 2			70°	70°

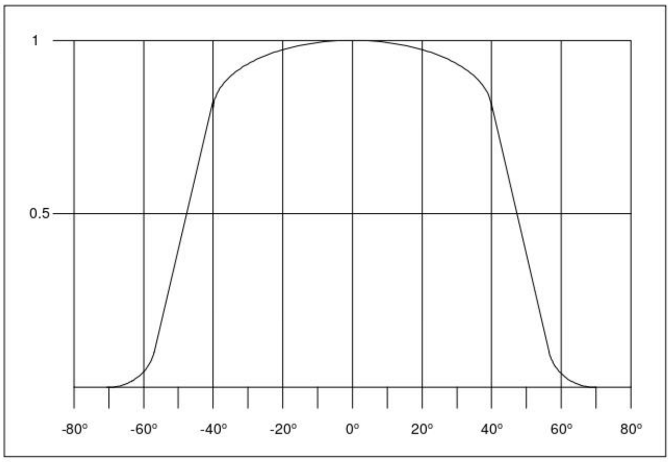


Figure 19: FOV of MLX90614xAA

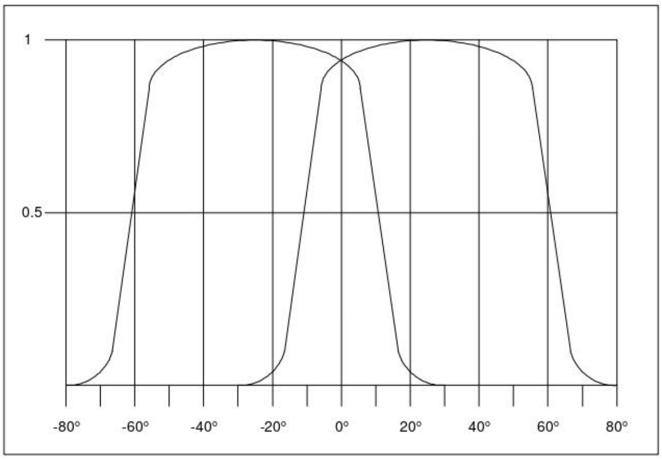


Figure 20: FOV of MLX90614xBA

Figure 21: identification of zone 1&2 relative to alignment tab.

10 Applications Information

10.1 Use of the MLX90614 thermometer in SMBus configuration

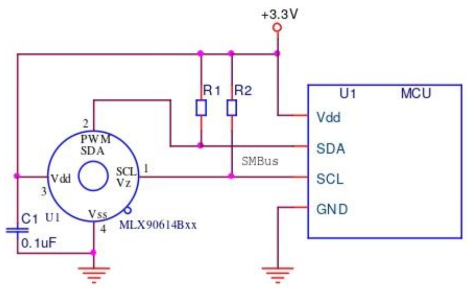


Figure 22: Connection of MLX90614 to SMBus, 3.3V power supply.

The MLX90614 has diode clamps SDA/SCL to Vdd so it is necessary to provide MLX90614 with power in order not to load the SMBus lines.

10.2 Use of multiple MLX90614s in SMBus configuration

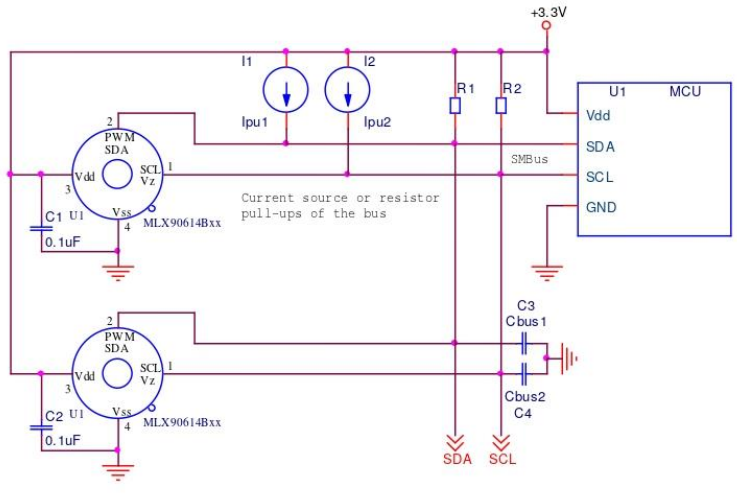


Figure 23: SMBus configuration of multiple sensors.

The MLX90614 supports a 7-bit slave address in EEPROM, thus allowing up to 127 devices to be read via two common wires. With the MLX90614BBx this results in 254 object temperatures measured remotely and an additional 127 ambient temperatures which are also available. Current source pull-ups may be preferred with higher capacitive loading on the bus (C3 and C4 represent the lines' parasitics), while simple resistive pull-ups provide the obvious low cost advantage.

10.3 Thermal alert / thermostat

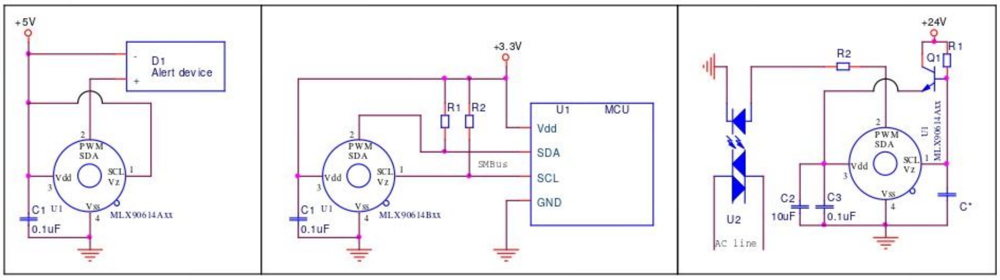


figure 24: Possible thermal relay configurations.

The MLX90614 can be configured in EEPROM to operate as a thermal relay. A non contact freezing or boiling prevention with 1 mA quiescent current can be built with two components only – the MLX90614 and a capacitor. The PWM/SDA pin can be programmed as a push-pull or open drain NMOS, which can trigger external device, such as a relay (refer to electrical specifications for load capability), buzzer, RF transmitter or a LED. This feature allows very simple thermostats to be built without the need of any MCU and zero design overhead required for firmware development. In conjunction with a MCU, this function can operate as a system alert that wakes up the MCU. Both object temperature and sensor die temperature can be also read in this configuration.

10.4 High voltage source operation

As a standard, the module MLX90614Axx works with a supply voltage of 5Volt. In addition, thanks to the integrated internal reference regulator available at pin SCL/Vz, this module can easily be powered from higher voltage source (like VDD=8…16V). Only a few external components as depicted in the diagram below are required to achieve this.

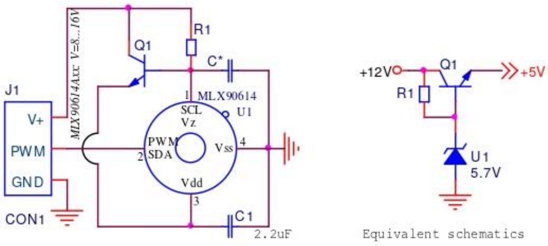


Figure 25: 12V regulator implementation

With the second (synthesized zener diode) function of the SCL/Vz pin used the 2-wire interface function is available only if the voltage regulator is overdriven (5V regulated power is forced to Vdd pin).
When the zener diode function of the SCL/Vz pin is used, the 2-wire SMBus function is only available if the voltage regulator is overdriven (5V regulated power is forced to the VDD pin).

11 Application Comments

Significant **contamination** at the optical input side (sensor filter) might cause unknown additional filtering/distortion of the optical signal and therefore result in unspecified errors.

IR sensors are inherently susceptible to errors caused by **thermal gradients**. There are physical reasons for that phenomena and, in spite of the careful design of the MLX90614xxx, it is recommended not to subject the MLX90614 to heat transfer and especially transient conditions.

Upon **power-up** the MLX90614 passes embedded checking and calibration routines. During these routines the output is not defined and it is recommended to wait for the specified POR time before reading the module. Very slow power-up may cause the embedded POR circuitry trigger on inappropriate levels, resulting in unspecified operation and is not recommended.

The MLX90614xxx is designed and calibrated to operate as a non contact thermometer in **settled conditions**. Using the module in very different way will result in unknown results.

Capacitive loading on a SMBus can degrade the communication. Some improvement is possible with use of current sources compared to resistors in pull-up circuitry. Further improvement is possible with specialized commercially available bus accelerators. With the MLX90614xxx additional improvement is possible with increasing the pull-up current (decreasing the pull-up resistor values). Input levels for SMBus compatible mode have higher overall tolerance than the SMBus specification, but the output low level is rather low even with the high-power SMBus specification for pull-up currents. Another option might be to go for a slower communication (clock speed), as the MLX90614xxx implements Schmidt triggers on it's inputs in SMBus compatible mode and is therefore not really sensitive to rise time of the bus (it is more likely the rise time to be an issue than the fall time, as far as the SMBus systems are open drain with pull-up).

For **ESD protection** there are clamp diodes between the Vss and Vdd and each of the other pins. This means that the MLX90614 might draw current from a bus in case the SCL and/or SDA is connected and the Vdd is lower than the bus pull-ups' voltage.

In **12V powered systems SMBus usage is constrained** because the SCL pin is used for the zener diode function. Therefore, higher than 5V applications are likely to use PWM output or external regulator. Nevertheless, in the 12V powered applications MLX90614 can be programmed (configured and customized) by forcing the Vdd to 5V externally and running the SMBus communication.

Sleep mode is available in MLX90614Bxx. This mode is entered and exited via the SMBus compatible 2-wire communication. On the other hand, the extended functionality of the SCL pin yields in increased leakage current through that pin. As a result, this pin needs to be forced low in power-down mode and the pull-up on the SCL line needs to be disabled in order to keep the overall power drain in power-down really small.

The **PWM pin is not designed for direct drive of inductive loads** (such as electro-magnetic relays). Some driver needs to be implemented for higher load, and auxiliary protection might be necessary even for light but inductive loading.

It is possible to use the MLX90614xxx in applications, powered directly from the AC line (trasformerless). In such cases it is very important not to forget that **the metal package of the sensor is not isolated** and therefore may occur to be connected to that line, too. Melexis can not be responsible for any application like this and highly recommends not to use the MLX90614xxx in that way.

Power dissipation within the package may affect performance in two ways: by heating the "ambient" sensitive element significantly beyond the actual ambient temperature, as well as by causing gradients over the package that will inherently cause thermal gradient over the cap. Loading the outputs also causes increased power dissipation. In case of using the MLX90614Axx internal zener voltage feature, the regulating external transistor should also not cause heating of the TO39 package.

High capacitive load on a PWM line will result in significant charging currents from the power supply, bypassing the capacitor and therefore causing EMC, noise, level degradation and power dissipation problems. A simple option is adding a series resistor between the PWM/SDA pin and the capacitive loaded line, in which case timing specifications have to be carefully reviewed. For example, with a PWM output that is set to 1.024 ms and the output format that is 11 bit, the time step is 0.5 μs and a settling time of 2 μs would introduce a 4 LSBs error.

Check www.melexis.com for most current application notes about MLX90614.

Standard information regarding manufacturability of Melexis products with different soldering processes
Our products are classified and qualified regarding soldering technology, solderability and moisture sensitivity level according to following test methods:

Reflow Soldering SMD's (Surface Mount Devices)

- IPC/JEDEC J-STD-020
  Moisture/Reflow Sensitivity Classification for Nonhermetic Solid State Surface Mount Devices
  (classification reflow profiles according to table 5-2)
- EIA/JEDEC JESD22-A113
  Preconditioning of Nonhermetic Surface Mount Devices Prior to Reliability Testing
  (reflow profiles according to table 2)

Wave Soldering SMD's (Surface Mount Devices) and THD's (Through Hole Devices)

- EN60749-20
  Resistance of plastic- encapsulated SMD's to combined effect of moisture and soldering heat
- EIA/JEDEC JESD22-B106 and EN60749-15
  Resistance to soldering temperature for through-hole mounted devices

Iron Soldering THD's (Through Hole Devices)

- EN60749-15
  Resistance to soldering temperature for through-hole mounted devices

Solderability SMD's (Surface Mount Devices) and THD's (Through Hole Devices)

- EIA/JEDEC JESD22-B102 and EN60749-21
  Solderability

For all soldering technologies deviating from above mentioned standard conditions (regarding peak temperature, temperature gradient, temperature profile etc) additional classification and qualification tests have to be agreed upon with Melexis.

The application of Wave Soldering for SMD's is allowed only after consulting Melexis regarding assurance of adhesive strength between device and board.

Melexis is contributing to global environmental conservation by promoting **lead free** solutions. For more information on qualifications of **RoHS** compliant products (RoHS = European directive on the Restriction Of the use of certain Hazardous Substances) please visit the quality page on our website:
http://www.melexis.com/quality.asp
The MLX90614 is RoHS compliant

12 ESD Precautions

Electronic semiconductor products are sensitive to Electro Static Discharge (ESD).
Always observe Electro Static Discharge control procedures whenever handling semiconductor products.

13 FAQ

When I measure aluminium and plastic parts settled at the same conditions I get significant errors on aluminium. Why?
Different materials have different **emissivity**. A typical value for aluminium (roughly polished) is 0.18 and for plastics values of 0.84…0.95 are typical. IR thermometers use the radiation flux between the sensitive element in the sensor and the object of interest, given by the equation

$$q = \varepsilon_1.\alpha_1.\left(T_1^{\,4}\right)\sigma.A_1.F_{a-b} - \varepsilon_2.\left(T_2^{\,4}\right)\sigma.A_2,$$

where
$\boldsymbol{\varepsilon_1}$ and $\boldsymbol{\varepsilon_2}$ are the emissivities of the two objects,
$\boldsymbol{\alpha_1}$ is the absorptivity of the sensor (in this case),
$\boldsymbol{\sigma}$ is the Stefan-Boltzmann constant,
$\mathbf{A_1}$ and $\mathbf{A_2}$ are the surface areas involved in the radiation heat transfer,
$\mathbf{F_{a\text{-}b}}$ is the shape factor,
$\mathbf{T_1}$ and $\mathbf{T_2}$ are known temperature of the sensor die (measured with specially integrated and calibrated element) and the object temperature that we need.
Note that these are all in Kelvin, heat exchange knows only physics.
When a body with low emissivity (such as aluminium) is involved in this heat transfer, the portion of the radiation incident to the sensor element that really comes from the object of interest decreases – and the reflected environmental IR emissions take place. (This is all for bodies with zero transparency in the IR band.)
The IR thermometer is calibrated to stay within specified accuracy – but it has no way to separate the incoming IR radiation into real object and reflected environmental part. Therefore, measuring objects with low emissivity is a very sophisticated issue and infra-red measurements of such materials is a specialised field.
What can be done to solve that problem? Look at paintings – for example, oil paints are likely to have emissivity of 0.85…0.95 – but keep in mind that the stability of the paint emissivity has inevitable impact on measurements.
It is also a good point to keep in mind that not everything that looks black is "black" also for IR. For example, even heavily oxidized aluminium has still emissivity as low as 0.30.
How high is enough? Not an easy question – but, in all cases the closer you need to get to the real object temperature the higher the needed emissivity will be, of course.
With the real life emissivity values the environmental IR comes into play via the reflectivity of the object (the sum of Emissivity, Reflectivity and Absorptivity gives 1.00 for any material). The larger the difference between environmental and object temperature is at given reflectivity (*with an opaque for IR material reflectivity equals 1.00 minus emissivity*) the bigger errors it produces.

After I put the MLX90614 in the dashboard I start getting errors larger than specified in spite that the module was working properly before that. Why?
Any object present in the FOV of the module provides IR signal. It is actually possible to introduce error in the measurements if the module is attached to the dashboard with an opening that enters the FOV. In that case portion of the dashboard opening will introduce IR signal in conjunction with constraining the effective FOV and thus compromising specified accuracy. Relevant opening that takes in account the FOV is a must for accurate measurements. Note that the basic FOV specification takes 50% of IR signal as threshold (in order to define the area, where the measurements are relevant), while the entire FOV at lower level is capable of introducing lateral IR signal under many conditions.

When a hot (cold) air stream hits my MLX90614 some error adds to the measured temperature I read. What is it?
IR sensors are inherently sensitive to difference in temperatures between the sensitive element and everything incident to that element. As a matter of fact, this element is not the sensor package, but the sensor die inside. Therefore, a thermal gradient over the sensor package will inevitably result in additional IR flux between the sensor package and the sensor die. This is real optical signal that can not be segregated from the target IR signal and will add errors to the measured temperature.
Thermal gradients with impact of that kind are likely to appear during transient conditions. The sensor used is developed with care about sensitivity to this kind of lateral phenomena, but their nature demands some care when choosing place to use the MLX90614 in order to make them negligible.

I measure human body temperature and I often get measurements that significantly differ from the +37°C I expect.
IR measurements are true surface temperature measurements. In many applications this means that the actual temperature measured by an IR thermometer will be temperature of the clothing and not the skin temperature. Emissivity (explained first in this section) is another issue with clothes that has to be considered. There is also the simple chance that the measured temperature is adequate – for example, in a cold winter human hand can appear at temperatures not too close to the well known +37°C.

I consider using MLX90614AAA to measure temperature within car compartment, but I am embarrassed about the Sun light that may hit the module. Is it a significant issue?
Special care is taken to cut off the visible light spectra as well as the NIR (near IR) before it reaches the sensitive sensor die. Even more, the glass (in most cases) is not transparent to the IR radiation used by the MLX90614. Glass has temperature and really high emissivity in most cases – it is “black” for IR of interest. Overall, Sun behind a window is most likely to introduce relatively small errors. Why is it not completely eliminated after all? Even visible light partially absorbed in the filter of the sensor has some heating potential – and there is no way that the sensor die will be “blind” for that heating right in front of it.

14 Package Information

The MLX90614 is packaged in an industry standard TO – 39 can.

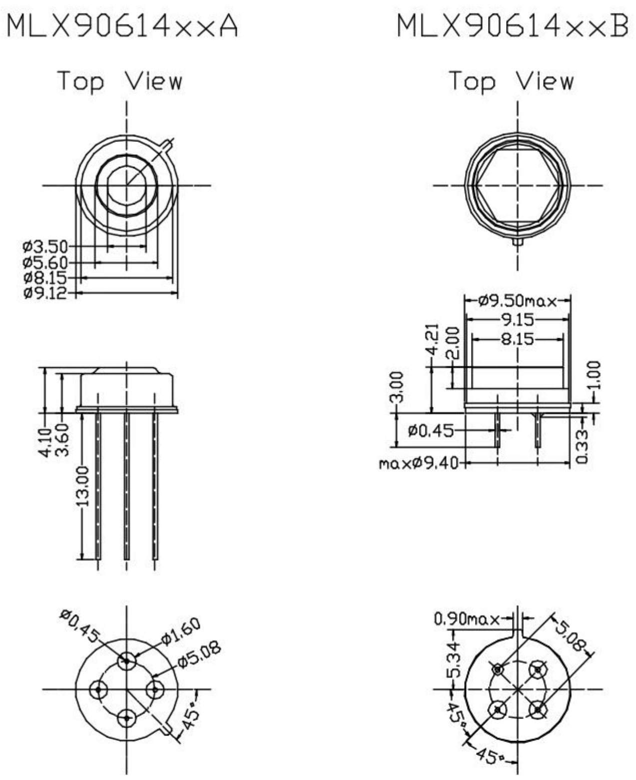


Figure 26: Packaging options

15 References

[1] **System Management Bus (SMBus) Specification** Version 2.0 August 3, 2000
SBS Implementers Forum Copyright . 1994, 1995, 1998, 2000
Duracell, Inc., Energizer Power Systems, Inc., Fujitsu, Ltd., Intel Corporation, Linear Technology Inc., Maxim Integrated Products, Mitsubishi Electric Semiconductor Company, PowerSmart, Inc., Toshiba Battery Co. Ltd., Unitrode Corporation, USAR Systems, Inc.

16 Disclaimer

Devices sold by Melexis are covered by the warranty and patent indemnification provisions appearing in its Term of Sale. Melexis makes no warranty, express, statutory, implied, or by description regarding the information set forth herein or regarding the freedom of the described devices from patent infringement. Melexis reserves the right to change specifications and prices at any time and without notice. Therefore, prior to designing this product into a system, it is necessary to check with Melexis for current information. This product is intended for use in normal commercial applications. Applications requiring extended temperature range, unusual environmental requirements, or high reliability applications, such as military, medical life-support or life-sustaining equipment are specifically not recommended without additional processing by Melexis for each application.
The information furnished by Melexis is believed to be correct and accurate. However, Melexis shall not be liable to recipient or any third party for any damages, including but not limited to personal injury, property damage, loss of profits, loss of use, interrupt of business or indirect, special incidental or consequential damages, of any kind, in connection with or arising out of the furnishing, performance or use of the technical data herein. No obligation or liability to recipient or any third party shall arise or flow out of Melexis' rendering of technical or other services.

For the latest version of this document, go to our website at
www.melexis.com

Or for additional information contact Melexis Direct:

Europe, Africa, Asia:
Phone: +32 1367 0495
E-mail: sales_europe@melexis.com

America:
Phone: +1 603 223 2362
E-mail: sales_usa@melexis.com

ISO/TS 16949 and ISO14001 Certified

BIBLIOGRAFÍA

[1] "Arduino y el termómetro infrarrojo a distancia MLX90614." https://www.luisllamas.es/arduino-y-el-termometro-infrarrojo-a-distancia-mlx90614/ (accessed Apr. 20, 2021).

[2] "Descripción y funcionamiento del Bus I2C | Robots Didácticos." http://robots-argentina.com.ar/didactica/descripcion-y-funcionamiento-del-bus-i2c/ (accessed Apr. 20, 2021).

[3] "TodoElectrodo: Lcd 16x2." http://todoelectrodo.blogspot.com/2013/02/lcd-16x2.html (accessed Apr. 20, 2021).

[4] "Módulo adaptador LCD a I2C." https://naylampmechatronics.com/lcd-alfanumerico/60-modulo-adaptador-lcd-a-i2c.html (accessed Apr. 22, 2021).

[5] "¿Qué es Arduino? | Arduino.cl - Compra tu Arduino en Línea." https://arduino.cl/que-es-arduino/ (accessed Apr. 20, 2021).

[6] "ELEGOO Mega 2560 R3 Tarjeta Placa Compatible con Arduino IDE con Microcontrolador Basada en el ATmega2560 ATmega16U2 con USB Cable Azul Versión Mega Kit: Amazon.es: Electrónica." https://www.amazon.es/ELEGOO-Microcontrolador-ATmega2560-ATmega16U2-Compatible/dp/B06Y3ZHPWC/ref=pd_lpo_147_t_0/258-7002493-6760934?_encoding=UTF8&pd_rd_i=B06Y3ZHPWC&pd_rd_r=8e0ab4f3-c2f3-4c5a-959d-5514f0ff1006&pd_rd_w=oR7mf&pd_rd_wg=eFTcj&pf_rd_p=4221015a-01c7-4a3d-a84d-985d938e9995&pf_rd_r=H5NQJA0JEQJKWN8DRW9Y&psc=1&refRID=H5NQJA0JEQJKWN8DRW9Y (accessed Apr. 20, 2021).

[7] Ó. Torrente Artero, "Arduino Curso Práctico de formación." Accessed: May 16, 2021. [Online]. Available: www.FreeLibros.me.

[8] "SRAM - Wikipedia, la enciclopedia libre." https://es.wikipedia.org/wiki/SRAM (accessed Apr. 21, 2021).

[9] "Arduino Mega 2560 Rev3 | Arduino Official Store." https://store.arduino.cc/arduino-mega-2560-rev3 (accessed May 16, 2021).

[10] "EEPROM - Wikipedia, la enciclopedia libre." https://es.wikipedia.org/wiki/EEPROM (accessed Apr. 21, 2021).

[11] "Adafruit AMG8833 IR Cámara Térmica Breakout - Melopero." https://www.melopero.com/es/shop/sensori/infrarossi/adafruit-amg8833-ir-thermal-camera-breakout/ (accessed Apr. 20, 2021).

[12] "Raspberry Pi: todo lo necesario para iniciarse como maker desde cero." https://www.xataka.com/makers/cero-maker-todo-necesario-para-empezar-raspberry-pi (accessed Apr. 20, 2021).

[13] "Buy a Raspberry Pi 3 Model B – Raspberry Pi." https://www.raspberrypi.org/products/raspberry-pi-3-model-b/ (accessed Apr. 20, 2021).

[14] "Operating system images – Raspberry Pi." https://www.raspberrypi.org/software/operating-systems/#raspberry-pi-os-32-bit (accessed May 22, 2021).

[15] "GitHub - makerportal/AMG8833_IR_cam: Python codes for development of a real-time thermal camera using a Raspberry Pi computer and AMG8833 infrared array." https://github.com/makerportal/AMG8833_IR_cam (accessed May 16, 2021).

[16] "Arduino - Leer multiples MLX90614 (termómetros infrarrojos) cambiando sus direcciones - Tecnología, Ofimática y Programación." https://ofimatica-programacion.blogspot.com/2017/05/arduino-mega-leer-dos-sensores-mlx90614.html (accessed Apr. 21, 2021).

[17] "GitHub - fdebrabander/Arduino-LiquidCrystal-I2C-library: Library for the LiquidCrystal LCD display connected to an Arduino board." https://github.com/fdebrabander/Arduino-LiquidCrystal-I2C-library (accessed May 16, 2021).

[18] "AVR/compile/src/i2c at master · RobotsWar/AVR · GitHub." https://github.com/RobotsWar/AVR/tree/master/compile/src/i2c (accessed May 16, 2021).

[19] "GitHub - SodaqMoja/Wire: A fork of the Arduino Wire library (avr only)." https://github.com/SodaqMoja/Wire (accessed May 16, 2021).

[20] "¿Para qué sirve Python? Razones para utilizarlo | ESIC." https://www.esic.edu/rethink/tecnologia/para-que-sirve-python (accessed May 23, 2021).

[21] B. W. Kernighan, B. W. Kernighan, and D. M. Ritchie, *El lenguaje de programación C*, 2ª. Prentice Hall Hispanoamericana, 1991.

[22] "PANTALLA 3.5 PULGADAS HDMI LCD IPS TACTIL RESISTIVO - KUBII." https://www.kubii.es/pantallas-pitft-raspberry-pi/2585-pantalla-35-pulgadas-hdmi-lcd-ips-tactil-resistivo-3272496297548.html (accessed May 16, 2021).

[23] "Cámara de imagen térmica MLX90640, matriz IR, 32x24 píxeles, campo de visión de 110 grados, interfaz I2C|Circuitos integrados| - AliExpress." https://es.aliexpress.com/item/1005001666859913.html?spm=a2g0o.search0302.0.0.738670ebSILDfy&algo_pvid=45f1f6f8-ae0f-4b6c-bcc1-f72495f8d473&algo_expid=45f1f6f8-ae0f-4b6c-bcc1-f72495f8d473-3&btsid=0b0a556416217606924491390ef04b&ws_ab_test=searchweb0_0,searchweb201602_,searchweb201603_ (accessed May 23, 2021).

[24] "Waveshare MLX90640 IR Array Thermal Imaging Camera 32×24 Pixels 110°Field of View Communicating Via I2C Interface Supports Raspberry Pi in Miscellaneous Industrial or Intelligent Control Applications: Amazon.es: Electrónica." https://www.amazon.es/Waveshare-Thermal-Camera-Communicating-Miscellaneous/dp/B07ZKJPZ7Z (accessed May 23, 2021).

[25] "Raspberry Pi 4 Modelo B - 2GB - KUBII." https://www.kubii.es/les-cartes-raspberry-pi/2771-nouveau-raspberry-pi-4-modele-b-2gb-0765756931175.html?src=raspberrypi (accessed Jun. 28, 2021).

[26] "Raspberry Pi 4 Modelo B - 4GB - KUBII." https://www.kubii.es/les-cartes-raspberry-pi/2772-nouveau-raspberry-pi-4-modele-b-4gb-kubii-0765756931182.html?src=raspberrypi (accessed Jun. 28, 2021).

[27] "Raspberry Pi 4 modelo B - 8GB." https://www.kubii.es/raspberry-pi-3-2-b/2955-raspberry-pi-4-modele-b-8gb-3272496301535.html?src=raspberrypi (accessed Jun. 28,

2021).

[28] “Módulo Sensor de Temperatura MLX90614ESF-BAA GY-906.” https://solectroshop.com/es/sensores-de-temperatura/5294-modulo-sensor-de-temperatura-mlx90614esf-gy-906-baa.html (accessed May 16, 2021).

[29] “AZDelivery Modulo Pantalla LCD Display Verde HD44780 1602 con interfaz I2C 16x2 caracteres negros compatible con Arduino con E-book incluido!: Amazon.es: Industria, empresas y ciencia.” https://www.amazon.es/AZDelivery-HD44780-pantalla-interfaz-caracteres/dp/B01N3B8JMN/ref=sr_1_2_sspa?__mk_es_ES=ÅMÅŽÕÑ&dchild=1&keywords=lcd+2x16+i2c&qid=1621157222&sr=8-2-spons&psc=1&smid=A1X7QLRQH87QA3&spLa=ZW5jcnlwdGVkUXVhbGlmaWVyPUExRDI2OFJUU1VBN083JmVuY3J5cHRlZElkPUEwNTQyNTIzM0FOR0FEUDdKNDcxTiZlbmNyeXB0ZWRBZElkPUEwMzk2MzUwMjJYNVVXOVhUNVg5TiZ3aWRnZXROYW1lPXNwX2F0ZiZhY3Rpb249Y2xpY2tSZWRpcmVjdCZkb05vdExvZ0NsaWNrPXRydWU= (accessed May 16, 2021).

[30] “Cámara térmica AMG8833 FeatherWing Arduino 3622 | BricoGeek.com.” https://tienda.bricogeek.com/adafruit-feather-wings/1035-camara-termica-amg8833-featherwing.html?search_query=amg8833&results=1 (accessed May 16, 2021).

[31] “RASPBERRY PI 3 MODEL B+ - Placa de base: Raspberry-Pi: Amazon.es: Informática.” https://www.amazon.es/Raspberry-Pi-Modelo-Placa-Color/dp/B07BFH96M3/ref=sr_1_4?__mk_es_ES=ÅMÅŽÕÑ&dchild=1&keywords=raspberry+pi+3+model+b&qid=1621157130&sr=8-4 (accessed May 16, 2021).

[32] “Pack de LED 300 unidades 3mm y 5mm, kit leds BricoGeek | BricoGeek.com.” https://tienda.bricogeek.com/diodos-led/1397-pack-de-led-300-unidades-35mm.html?search_query=led&results=210 (accessed May 16, 2021).

[33] “Cables DuPont Macho - Macho (20 cm / 40 unidades) BricoGeek | BricoGeek.com.” https://tienda.bricogeek.com/cables/1361-cables-dupont-macho-macho-20-cm-40-unidades.html?search_query=cables&results=168 (accessed May 16, 2021).

[34] “Cables DuPont Macho - Hembra (20 cm / 40 unidades) BricoGeek | BricoGeek.com.” https://tienda.bricogeek.com/cables/1362-cables-dupont-macho-hembra-20-cm-40-unidades.html?search_query=cables&results=168 (accessed May 16, 2021).

[35] “BUZZER (ZUMBADOR) PASIVO ELECTROMAGNETICO 5V - tiendatec.es.” https://www.tiendatec.es/electronica/componentes/buzzers/655-buzzer-zumbador-pasivo-electromagnetico-5v-8406551370005.html?utm_source=doofinder&utm_medium=results (accessed Jun. 14, 2021).

[36] “Xiaomi Mi PowerBank 3 10000 MAh 18W QuickCharge 30 Negra | PcComponentes.com.” https://www.pccomponentes.com/xiaomi-mi-powerbank-3-10000-mah-18w-quickcharge-30-negra (accessed Jun. 14, 2021).

[37] “MODULO HC-SR04 SENSOR DISTANCIA POR ULTRASONIDOS PARA ARDUINO Y RASPBERRY PI - tiendatec.es.” https://www.tiendatec.es/arduino/modulos/392-modulo-hc-sr04-sensor-distancia-por-ultrasonidos-para-arduino-y-raspberry-pi-8403921180011.html?utm_source=doofinder&utm_medium=results (accessed Jun. 14, 2021).

[38] “PANTALLA LCD TACTIL 5" 800x480 HDMI PARA RASPBERRY PI - tiendatec.es.” https://www.tiendatec.es/raspberry-pi/pantallas/525-pantalla-lcd-tactil-5-800x480-

hdmi-para-raspberry-pi-8405250330013.html (accessed Jun. 14, 2021).

[39] "Lenovo C24-25 - Monitor de 23.8" FullHD (1920x1080 pixeles, 16:9, 75Hz, 4ms, 1000:1, Puertos VGA + HDMI, 3 lados, sin bordes) Color Negro: Amazon.es: Informática." https://www.amazon.es/Lenovo-C24-25-Monitor-1920x1080-pixeles/dp/B08D8R4VLX/ref=sr_1_3?__mk_es_ES=ÅMÅŽÕÑ&crid=1I04SG7WANGUP&dchild=1&keywords=monitor+24+pulgadas&qid=1624872077&s=computers&sprefix=monitor%2Ccomputers%2C221&sr=1-3 (accessed Jun. 28, 2021).

[40] "Fritzing." https://fritzing.org/ (accessed Jun. 17, 2021).

[41] "EasyEDA – Simulador de circuitos y diseño de circuitos impresos online." https://easyeda.com/ (accessed Jun. 17, 2021).

[42] "AMG8833 pdf, AMG8833 description, AMG8833 datasheets, AMG8833 view ::: ALLDATASHEET :::" https://pdf1.alldatasheet.com/datasheet-pdf/view/1244670/PANASONIC/AMG8833.html (accessed Jun. 17, 2021).

[43] "MLX90614 Datasheet(PDF) - List of Unclassifed Manufacturers." https://www.alldatasheet.com/datasheet-pdf/pdf/218977/ETC2/MLX90614.html (accessed Jun. 17, 2021).