QUICK REFERENCE · TK42

LM75 Temperature Sensor

Degrees in two bytes, on two wires, at address 0x48.

VCC goes to your board's logic supply

The pull-ups on SDA and SCL end at VCC, so the bus rests at whatever VCC is. **5V on an Uno, 3V3 on an ESP32, ESP32-S3 or Pico.** 5V on VCC beside a 3.3 V board puts 5 V on two of its pins.

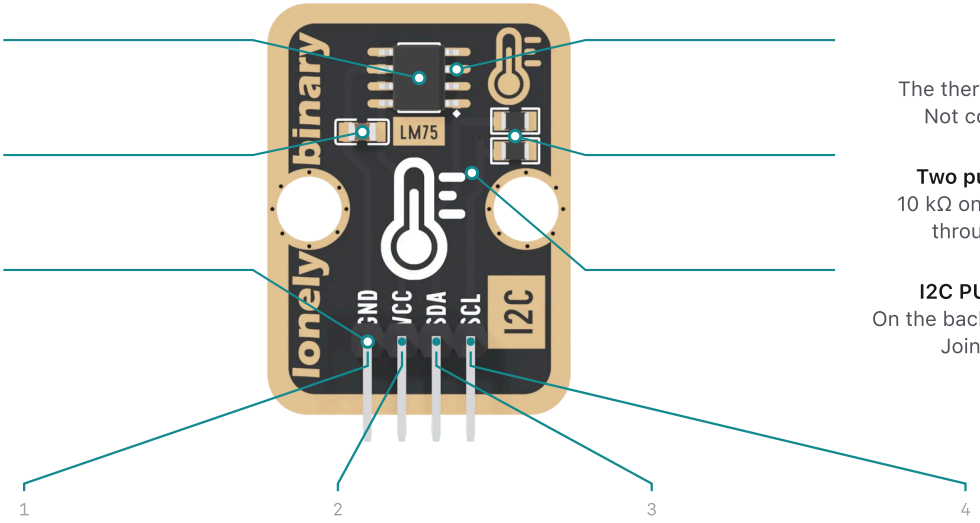
Pins, and which to wire

Parts up, header at the bottom. All four pins are wired. There is no light: the chip answering at 0x48 is the sign it works.

LM75B at 0x48
A0 to A2 on GND: fixed. One TK42 per bus.

Supply capacitor
100 nF, VCC to GND, beside the chip.

GND is the square pad
Then VCC, SDA, SCL. The back prints no names.



OS, pin 3
The thermostat output. Not connected here.

Two pull-ups to VCC
10 kΩ on SDA and SCL, through the jumper.

I2C PULL UP jumper
On the back, behind here. Joined as shipped.

Front	GND	VCC	SDA	SCL
I2C	GND	VCC	SDA	SCL

Read the two bytes signed

Register 0 holds 256 times the temperature as a signed 16-bit number. Put the bytes in an int16_t and divide by 256. The common alternative, an int shifted right by five, prints 252 °C for −4 °C on an ESP32 or a Pico.

Three numbers to read it by

Step

0.125 °C a count

11 bits on NXP's sheet; UMW's claims 12, 0.0625 °C. Dividing by 256 gives either.

Accuracy

±2 °C at most

From -25 to 100 °C on NXP's sheet; ±3 °C to the ends of the range.

Update rate

100 ms apart

It converts on its own clock. Reading faster returns the last value again.

The step is sixteen times finer than the error: a change of 0.125 °C is real, and the number itself may be 2 °C from the air's. It measures its own chip, so keep it away from warm parts.

Numbers from NXP's LM75B datasheet (LM75B Rev. 6.1, 6 February 2015), the looser of the two makers' sheets. The fitted part is UMW's LM75BD; UMW's sheet (UMW LM75B, July 2025) claims 12 bits, ±1 °C and a reading every 80 ms. Not measured on a bench.

Print the temperature, twice a second

Arduino IDE · Board: your board · ESP32-S3: USB CDC On Boot: Enabled

```
#include <Wire.h>
const int LM75_ADDR = 0x48;  // fixed on the TK42

bool readCelsius(float &c) {
  Wire.beginTransmission(LM75_ADDR);
  Wire.write(0x00);                    // register 0
  if (Wire.endTransmission() != 0) return false;
  if (Wire.requestFrom(LM75_ADDR, 2) != 2) return false;
  uint8_t hi = Wire.read();            // whole degrees
  uint8_t lo = Wire.read();            // the fraction
  int16_t raw = (int16_t)((hi << 8) | lo);
  c = raw / 256.0;                     // right below zero too
  return true;
}

void setup() {
  Serial.begin(115200);
  delay(1000);
  Wire.begin();                        // default I2C pins
  float c;
  readCelsius(c);                      // first one thrown away
}

void loop() {
  float c;
  if (readCelsius(c)) { Serial.print(c, 3); Serial.println(" C"); }
  else Serial.println("No LM75 at 0x48: check SDA and SCL.");
  delay(500);
}
```

```
23.625 C
23.625 C
23.750 C
```

Serial Monitor at 115200, a line every half second. A finger on the chip makes the number climb within a second or two.

Wiring

	FRONT	UNO	ESP32	S3	PICO
1	**GND**	GND	GND	GND	GND
2	**VCC**	5V	3V3	3V3	3V3(OUT)
3	**SDA**	A4	GPIO 21	GPIO 8	GP4
4	**SCL**	A5	GPIO 22	GPIO 9	GP5

Hole 1 is the square pad. VCC is the board's logic supply, never 5V beside a 3.3 V board. The SDA and SCL pins are each board's defaults, so Wire.begin() needs no arguments.

In MicroPython

On an ESP32, ESP32-S3 or Pico: `i2c = I2C(0, sda=Pin(21), scl=Pin(22))` (S3: 8 and 9; Pico: 4 and 5), then `b = i2c.readfrom_mem(0x48, 0, 2)` and `struct.unpack(">h", b)[0] / 256`. The `>h` reads the two bytes big-endian and signed, as the chip sends them.

When it does not work

Nothing answers at 0x48

Count from the square pad: GND, VCC, SDA, SCL. SDA and SCL swapped is the usual fault.

0.000 on the first line

Read before the first conversion ended. Throw the first reading away.

The same value, again and again

Reading faster than it converts, or a steady room. A finger on the chip tells which.

Below zero it reads about 250 °C

The bytes went into a 32-bit int. Read them into an int16_t and divide by 256.

A degree or two high

The chip is warmer than the air. Move it away from the microcontroller and regulators.

Two TK42s on one bus

Both answer at 0x48, and the address cannot move. Use a second I2C bus.

Specifications

Chip	UMW LM75BD (LM75B), SOIC-8	**Address**	0x48, fixed
Range	-55 to 125 °C	**Accuracy**	±2 °C, -25 to 100 °C (max)
Step	0.125 °C, 11 bits	**Supply**	2.8 to 5.5 V: your logic supply
Pull-ups	10 kΩ to VCC, cut jumper	**Clock**	I2C up to 400 kHz