

Erriez Oregon THN128 433MHz temperature sensor library for Arduino  
1.0.0

Generated by Doxygen 1.8.13



# Contents

<b>1</b>	<b><a href="#">Oregon THN128 433MHz temperature sensor transmit/receive library for Arduino</a></b>	<b>1</b>
<b>2</b>	<b><a href="#">Class Index</a></b>	<b>5</b>
2.1	<a href="#">Class List . . . . .</a>	5
<b>3</b>	<b><a href="#">File Index</a></b>	<b>7</b>
3.1	<a href="#">File List . . . . .</a>	7
<b>4</b>	<b><a href="#">Class Documentation</a></b>	<b>9</b>
4.1	<a href="#">OregonTHN128Data_t Struct Reference . . . . .</a>	9
4.1.1	<a href="#">Detailed Description . . . . .</a>	9
4.1.2	<a href="#">Member Data Documentation . . . . .</a>	9
4.1.2.1	<a href="#">channel . . . . .</a>	9
4.1.2.2	<a href="#">lowBattery . . . . .</a>	10
4.1.2.3	<a href="#">rawData . . . . .</a>	10
4.1.2.4	<a href="#">rollingAddress . . . . .</a>	10
4.1.2.5	<a href="#">temperature . . . . .</a>	10

<b>5</b>	<b>File Documentation</b>	<b>11</b>
5.1	src/ErriezOregonTHN128.c File Reference	11
5.1.1	Detailed Description	12
5.1.2	Macro Definition Documentation	12
5.1.2.1	GET_TEMP	12
5.1.2.2	SET_TEMP	12
5.1.3	Function Documentation	12
5.1.3.1	OregonTHN128_CheckCRC()	12
5.1.3.2	OregonTHN128_DataToRaw()	13
5.1.3.3	OregonTHN128_RawToData()	13
5.1.3.4	OregonTHN128_TempToString()	14
5.2	src/ErriezOregonTHN128Transmit.c File Reference	14
5.2.1	Detailed Description	15
5.2.2	Macro Definition Documentation	15
5.2.2.1	RF_TX_PIN_DISABLE	15
5.2.2.2	RF_TX_PIN_INIT	15
5.2.3	Function Documentation	16
5.2.3.1	OregonTHN128_Transmit()	16
5.2.3.2	OregonTHN128_TxBegin()	17
5.2.3.3	OregonTHN128_TxEnd()	17
5.2.3.4	OregonTHN128_TxRawData()	17
5.3	src/ErriezOregonTHN128Transmit.h File Reference	18
5.3.1	Detailed Description	19
5.3.2	Function Documentation	19
5.3.2.1	OregonTHN128_Transmit()	19
5.3.2.2	OregonTHN128_TxBegin()	19
5.3.2.3	OregonTHN128_TxRawData()	19
	<b>Index</b>	<b>21</b>

# Chapter 1

## Oregon THN128 433MHz temperature sensor transmit/receive library for Arduino

This is a transmit/receive library Arduino library with the Oregon THN128 433MHz wireless protocol.

Tested on an Arduino UNO.

### Protocol

### Data

### Example low power receive

```
{c++}
#include <LowPower.h>
#include <ErriezOregonTHN128Receive.h>

// RF pin 2 (INT0) or pin 3 (INT1) defines
#define RF_RX_PIN      2

void printReceivedData(OregonTHN128Data_t *data)
{
    bool negativeTemperature = false;
    static uint32_t rxCount = 0;
    int16_t tempAbs;
    char msg[80];

    // Convert to absolute temperature
    tempAbs = data->temperature;
    if (tempAbs < 0) {
        negativeTemperature = true;
        tempAbs *= -1;
    }
    snprintf(msg, sizeof(msg), "RX %lu: Rol: %d, Channel %d, Temp: %s%d.%d, Low batt: %d (0x%08lx)",
        rxCount++,
        data->rollingAddress, data->channel,
        (negativeTemperature ? "-" : ""), (tempAbs / 10), (tempAbs % 10), data->lowBattery,
        data->rawData);
    Serial.println(msg);
}

void setup()
{
    // Initialize serial port
    Serial.begin(115200);
    Serial.println(F("Oregon THN128 433MHz temperature receive"));

    // Turn LED on
    pinMode(LED_BUILTIN, OUTPUT);
}
```

```

    digitalWrite(LED_BUILTIN, HIGH);

    // Initialize receiver
    OregonTHN128_RxBegin(RF_RX_PIN);
}

void loop()
{
    OregonTHN128Data_t data;

    // Check temperature received
    if (OregonTHN128_Available()) {
        digitalWrite(LED_BUILTIN, LOW);

        // Read temperature
        OregonTHN128_Read(&data);

        // Print received data
        printReceivedData(&data);

        // Wait ~30 seconds before receiving next temperature
        Serial.flush();
        LowPower.powerDown(SLEEP_8S, ADC_OFF, BOD_OFF);
        LowPower.powerDown(SLEEP_8S, ADC_OFF, BOD_OFF);
        LowPower.powerDown(SLEEP_8S, ADC_OFF, BOD_OFF);
        LowPower.powerDown(SLEEP_2S, ADC_OFF, BOD_OFF);

        digitalWrite(LED_BUILTIN, HIGH);

        // Enable receive
        OregonTHN128_RxEnable();
    }
}

```

## Example low power transmit

```

{c++}
#include <LowPower.h>
#include <ErriezOregonTHN128Transmit.h>

// Pin defines
#define RF_TX_PIN          9

OregonTHN128Data_t data = {
    .rawData = 0,           // Raw data filled in by library
    .rollingAddress = 5,    // Rolling address 0..7
    .channel = 1,           // Channel 1, 2 or 3
    .temperature = 0,       // Temperature -99.9 .. 99.9 multiplied by 10
    .lowBattery = false,    // Low battery true or false
};

#ifdef __cplusplus
extern "C" {
#endif

// Function is called from library
void delay100ms()
{
    Serial.flush();
    digitalWrite(LED_BUILTIN, HIGH);
    LowPower.powerDown(SLEEP_60MS, ADC_OFF, BOD_OFF);
    digitalWrite(LED_BUILTIN, LOW);
    LowPower.powerDown(SLEEP_30MS, ADC_OFF, BOD_OFF);
}

#ifdef __cplusplus
}
#endif

void setup()
{
    // Initialize pins
    OregonTHN128_TxBegin(RF_TX_PIN);
}

void loop()
{
    // Set temperature
    data.temperature = 123; //12.3°C

    // Send temperature
    OregonTHN128_Transmit(&data);
}

```

---

```
// Wait some time
// Wait ~30 seconds before sending next temperature
Serial.flush();
LowPower.powerDown(SLEEP_8S, ADC_OFF, BOD_OFF);
LowPower.powerDown(SLEEP_8S, ADC_OFF, BOD_OFF);
LowPower.powerDown(SLEEP_8S, ADC_OFF, BOD_OFF);
LowPower.powerDown(SLEEP_4S, ADC_OFF, BOD_OFF);
}
```





## Chapter 2

# Class Index

### 2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<a href="#">OregonTHN128Data_t</a> . . . . .	9
----------------------------------------------	---



## Chapter 3

# File Index

### 3.1 File List

Here is a list of all documented files with brief descriptions:

src/ <a href="#">ErriezOregonTHN128.c</a>	
Oregon THN128 433MHz temperature transmit/receive library for Arduino . . . . .	<a href="#">11</a>
src/ <b>ErriezOregonTHN128.h</b> . . . . .	??
src/ <b>ErriezOregonTHN128Receive.cpp</b> . . . . .	??
src/ <b>ErriezOregonTHN128Receive.h</b> . . . . .	??
src/ <a href="#">ErriezOregonTHN128Transmit.c</a>	
Oregon THN128 433MHz temperature transmit library for Arduino . . . . .	<a href="#">14</a>
src/ <a href="#">ErriezOregonTHN128Transmit.h</a>	
Oregon THN128 433MHz temperature transmit library for Arduino . . . . .	<a href="#">18</a>



## Chapter 4

# Class Documentation

### 4.1 OregonTHN128Data\_t Struct Reference

#### Public Attributes

- uint32\_t [rawData](#)
- uint8\_t [rollingAddress](#)
- uint8\_t [channel](#)
- int16\_t [temperature](#)
- bool [lowBattery](#)

#### 4.1.1 Detailed Description

Definition at line 61 of file ErriezOregonTHN128.h.

#### 4.1.2 Member Data Documentation

##### 4.1.2.1 channel

```
uint8_t OregonTHN128Data_t::channel
```

Channel

Definition at line 64 of file ErriezOregonTHN128.h.

#### 4.1.2.2 lowBattery

```
bool OregonTHN128Data_t::lowBattery
```

Low battery indication

Definition at line 66 of file ErriezOregonTHN128.h.

#### 4.1.2.3 rawData

```
uint32_t OregonTHN128Data_t::rawData
```

Raw data

Definition at line 62 of file ErriezOregonTHN128.h.

#### 4.1.2.4 rollingAddress

```
uint8_t OregonTHN128Data_t::rollingAddress
```

Rolling address

Definition at line 63 of file ErriezOregonTHN128.h.

#### 4.1.2.5 temperature

```
int16_t OregonTHN128Data_t::temperature
```

Temperature

Definition at line 65 of file ErriezOregonTHN128.h.

The documentation for this struct was generated from the following file:

- src/ErriezOregonTHN128.h

## Chapter 5

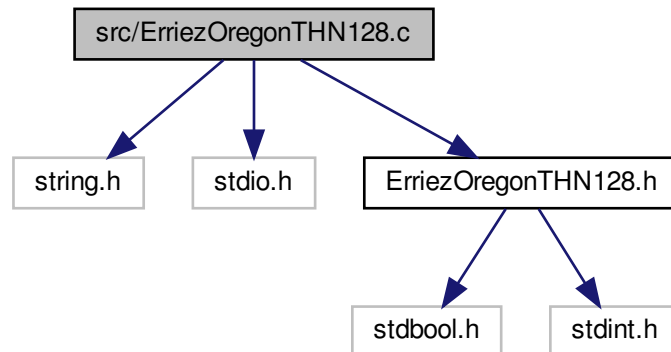
# File Documentation

### 5.1 src/ErriezOregonTHN128.c File Reference

Oregon THN128 433MHz temperature transmit/receive library for Arduino.

```
#include <string.h>
#include <stdio.h>
#include "ErriezOregonTHN128.h"
```

Include dependency graph for ErriezOregonTHN128.c:



#### Macros

- `#define SET_ROL_ADDR(x) (((x) & 0x07) << 0)`
- `#define GET_ROL_ADDR(x) (((x) & 0x07) << 0)`
- `#define SET_CHANNEL(x) (((x) - 1) & 0x03) << 6)`
- `#define GET_CHANNEL(x) (((x) >> 6) & 0x03) + 1)`
- `#define SET_TEMP(x)`
- `#define GET_TEMP(x)`
- `#define SIGN_BIT (1UL << 21)`
- `#define LOW_BAT_BIT (1UL << 23)`
- `#define SET_CRC(x) ((uint32_t)(x) << 24)`
- `#define GET_CRC(x) ((x) >> 24)`

## Functions

- bool [OregonTHN128\\_CheckCRC](#) (uint32\_t rawData)  
*Verify checksum.*
- void [OregonTHN128\\_TempToString](#) (char \*temperatureStr, uint8\_t temperatureStrLen, int16\_t temperature)  
*Convert temperature to string.*
- uint32\_t [OregonTHN128\\_DataToRaw](#) ([OregonTHN128Data\\_t](#) \*data)  
*Convert data structure to 32-bit raw data.*
- bool [OregonTHN128\\_RawToData](#) (uint32\_t rawData, [OregonTHN128Data\\_t](#) \*data)  
*Convert 32-bit raw data to [OregonTHN128Data\\_t](#) structure.*

### 5.1.1 Detailed Description

Oregon THN128 433MHz temperature transmit/receive library for Arduino.

Source: <https://github.com/Erriez/ErriezOregonTHN128> Documentation: <https://erriez.github.io/ErriezOregonTHN128>

### 5.1.2 Macro Definition Documentation

#### 5.1.2.1 GET\_TEMP

```
#define GET_TEMP(  
    x )
```

**Value:**

```
(((((x) >> 16) & 0x0f) * 100) + \  
    (((x) >> 12) & 0x0f) * 10) + \  
    ((x) >> 8) & 0x0f))
```

Definition at line 47 of file [ErriezOregonTHN128.c](#).

#### 5.1.2.2 SET\_TEMP

```
#define SET_TEMP(  
    x )
```

**Value:**

```
(((((uint32_t)(x) / 100) % 10) << 16) | \  
    (((uint32_t)(x) / 10) % 10) << 12) | \  
    ((x) % 10) << 8))
```

Definition at line 44 of file [ErriezOregonTHN128.c](#).

### 5.1.3 Function Documentation

#### 5.1.3.1 OregonTHN128\_CheckCRC()

```
bool OregonTHN128_CheckCRC (  
    uint32_t rawData )
```

Verify checksum.



**Parameters**

<i>rawData</i>	32-bit raw data input
----------------	-----------------------

**Returns**

true: Success, false: error

Definition at line 86 of file ErriezOregonTHN128.c.

**5.1.3.2 OregonTHN128\_DataToRaw()**

```
uint32_t OregonTHN128_DataToRaw (  
    OregonTHN128Data_t * data )
```

Convert data structure to 32-bit raw data.

**Parameters**

<i>data</i>	Input
-------------	-------

**Returns**

Output

Definition at line 126 of file ErriezOregonTHN128.c.

**5.1.3.3 OregonTHN128\_RawToData()**

```
bool OregonTHN128_RawToData (  
    uint32_t rawData,  
    OregonTHN128Data_t * data )
```

Convert 32-bit raw data to [OregonTHN128Data\\_t](#) structure.

**Parameters**

<i>rawData</i>	32-bit input
<i>data</i>	output

**Returns**

CRC true: Success, false: error

Definition at line 165 of file ErriezOregonTHN128.c.

### 5.1.3.4 OregonTHN128\_TempToString()

```
void OregonTHN128_TempToString (
    char * temperatureStr,
    uint8_t temperatureStrLen,
    int16_t temperature )
```

Convert temperature to string.

#### Parameters

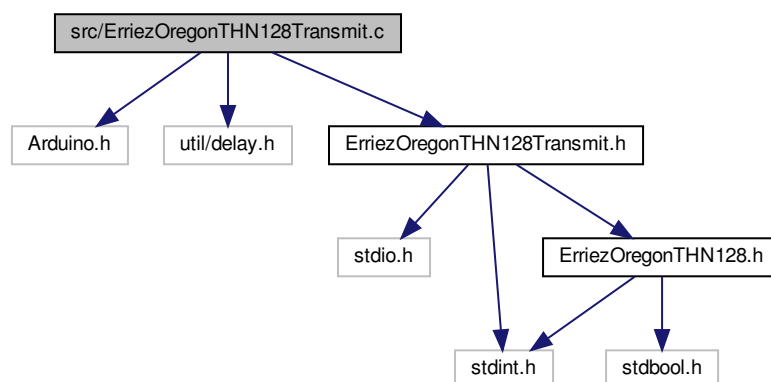
<i>temperatureStr</i>	Character buffer
<i>temperatureStrLen</i>	Size of character buffer
<i>temperature</i>	Input temperature

Definition at line 103 of file ErriezOregonTHN128.c.

## 5.2 src/ErriezOregonTHN128Transmit.c File Reference

Oregon THN128 433MHz temperature transmit library for Arduino.

```
#include <Arduino.h>
#include <util/delay.h>
#include "ErriezOregonTHN128Transmit.h"
Include dependency graph for ErriezOregonTHN128Transmit.c:
```



#### Macros

- `#define RF_TX_PIN_INIT(rfTxPin)`
- `#define RF_TX_PIN_DISABLE()`
- `#define RF_TX_PIN_HIGH() { *portOutputRegister(_rfTxPort) |= _rfTxBit; }`
- `#define RF_TX_PIN_LOW() { *portOutputRegister(_rfTxPort) &= ~_rfTxBit; }`

## Functions

- void `delay100ms` (void)  
*Transmit sync pulse.*
- void `OregonTHN128_TxBegin` (uint8\_t rfTxPin)  
*Transmit begin.*
- void `OregonTHN128_TxEnd` (void)  
*Disable transmit.*
- void `OregonTHN128_TxRawData` (uint32\_t rawData)  
*Transmit data.*
- void `OregonTHN128_Transmit` (OregonTHN128Data\_t \*data)  
*Transmit Transmit data.*

### 5.2.1 Detailed Description

Oregon THN128 433MHz temperature transmit library for Arduino.

Source: <https://github.com/Erriez/ErriezOregonTHN128> Documentation: <https://erriez.github.io/ErriezOregonTHN128>

### 5.2.2 Macro Definition Documentation

#### 5.2.2.1 RF\_TX\_PIN\_DISABLE

```
#define RF_TX_PIN_DISABLE( )
```

**Value:**

```
{
    \
    if ((_rfTxPort >= 0) && (_rfTxBit >= 0)) {
        *portModeRegister(_rfTxPort) &= ~_rfTxBit;
    }
}
```

#### 5.2.2.2 RF\_TX\_PIN\_INIT

```
#define RF_TX_PIN_INIT(
    rfTxPin )
```

**Value:**

```
{
    \
    _rfTxPort = digitalPinToPort(rfTxPin);
    _rfTxBit = digitalPinToBitMask(rfTxPin);
    *portModeRegister(_rfTxPort) |= _rfTxBit;
}
```

## 5.2.3 Function Documentation

### 5.2.3.1 OregonTHN128\_Transmit()

```
void OregonTHN128_Transmit (
    OregonTHN128Data_t * data )
```

Transmit Transmit data.

## Parameters

<i>data</i>	Oregon THN128 input structure
-------------	-------------------------------

Definition at line 195 of file ErriezOregonTHN128Transmit.c.

### 5.2.3.2 OregonTHN128\_TxBegin()

```
void OregonTHN128_TxBegin (
    uint8_t rfTxPin )
```

Transmit begin.

## Parameters

<i>rfTxPin</i>	Arduino transmit pin
----------------	----------------------

Definition at line 138 of file ErriezOregonTHN128Transmit.c.

### 5.2.3.3 OregonTHN128\_TxEnd()

```
void OregonTHN128_TxEnd (
    void )
```

Disable transmit.

Set transmit pin to input

Definition at line 149 of file ErriezOregonTHN128Transmit.c.

### 5.2.3.4 OregonTHN128\_TxRawData()

```
void OregonTHN128_TxRawData (
    uint32_t rawData )
```

Transmit data.

## Parameters

<i>rawData</i>	32-bit raw data input
----------------	-----------------------

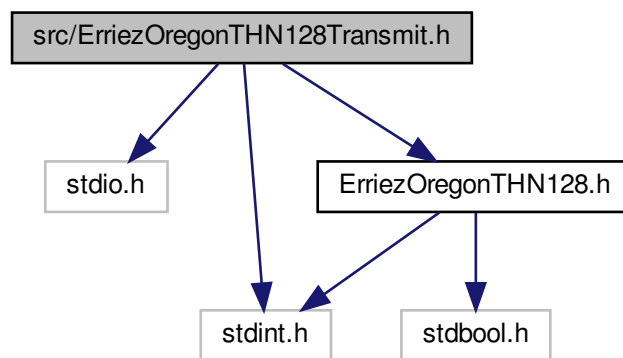
Definition at line 160 of file ErriezOregonTHN128Transmit.c.

### 5.3 src/ErriezOregonTHN128Transmit.h File Reference

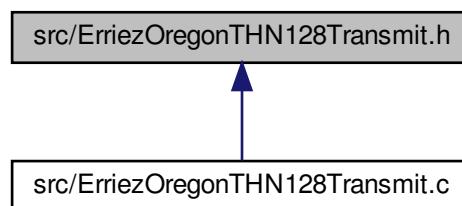
Oregon THN128 433MHz temperature transmit library for Arduino.

```
#include <stdio.h>
#include <stdint.h>
#include "ErriezOregonTHN128.h"
```

Include dependency graph for ErriezOregonTHN128Transmit.h:



This graph shows which files directly or indirectly include this file:



### Functions

- void [OregonTHN128\\_TxBegin](#) (uint8\_t rFTxPin)  
*Transmit begin.*
- void [OregonTHN128\\_TxRawData](#) (uint32\_t rawData)  
*Transmit data.*
- void [OregonTHN128\\_Transmit](#) ([OregonTHN128Data\\_t](#) \*data)  
*Transmit Transmit data.*

### 5.3.1 Detailed Description

Oregon THN128 433MHz temperature transmit library for Arduino.

Source: <https://github.com/Erriez/ErriezOregonTHN128> Documentation: <https://erriez.github.io/ErriezOregonTHN128>

### 5.3.2 Function Documentation

#### 5.3.2.1 OregonTHN128\_Transmit()

```
void OregonTHN128_Transmit (
    OregonTHN128Data_t * data )
```

Transmit Transmit data.

##### Parameters

<i>data</i>	Oregon THN128 input structure
-------------	-------------------------------

Definition at line 195 of file ErriezOregonTHN128Transmit.c.

#### 5.3.2.2 OregonTHN128\_TxBegin()

```
void OregonTHN128_TxBegin (
    uint8_t rFTxPin )
```

Transmit begin.

##### Parameters

<i>rFTxPin</i>	Arduino transmit pin
----------------	----------------------

Definition at line 138 of file ErriezOregonTHN128Transmit.c.

#### 5.3.2.3 OregonTHN128\_TxRawData()

```
void OregonTHN128_TxRawData (
    uint32_t rawData )
```

Transmit data.

**Parameters**

<i>rawData</i>	32-bit raw data input
----------------	-----------------------

Definition at line 160 of file ErriezOregonTHN128Transmit.c.



# Index

- channel
  - OregonTHN128Data\_t, [9](#)
- ErriezOregonTHN128.c
  - GET\_TEMP, [12](#)
  - OregonTHN128\_CheckCRC, [12](#)
  - OregonTHN128\_DataToRaw, [13](#)
  - OregonTHN128\_RawToData, [13](#)
  - OregonTHN128\_TempToString, [14](#)
  - SET\_TEMP, [12](#)
- ErriezOregonTHN128Transmit.c
  - OregonTHN128\_Transmit, [16](#)
  - OregonTHN128\_TxBegin, [17](#)
  - OregonTHN128\_TxEnd, [17](#)
  - OregonTHN128\_TxRawData, [17](#)
  - RF\_TX\_PIN\_DISABLE, [15](#)
  - RF\_TX\_PIN\_INIT, [15](#)
- ErriezOregonTHN128Transmit.h
  - OregonTHN128\_Transmit, [19](#)
  - OregonTHN128\_TxBegin, [19](#)
  - OregonTHN128\_TxRawData, [19](#)
- GET\_TEMP
  - ErriezOregonTHN128.c, [12](#)
- lowBattery
  - OregonTHN128Data\_t, [9](#)
- OregonTHN128\_CheckCRC
  - ErriezOregonTHN128.c, [12](#)
- OregonTHN128\_DataToRaw
  - ErriezOregonTHN128.c, [13](#)
- OregonTHN128\_RawToData
  - ErriezOregonTHN128.c, [13](#)
- OregonTHN128\_TempToString
  - ErriezOregonTHN128.c, [14](#)
- OregonTHN128\_Transmit
  - ErriezOregonTHN128Transmit.c, [16](#)
  - ErriezOregonTHN128Transmit.h, [19](#)
- OregonTHN128\_TxBegin
  - ErriezOregonTHN128Transmit.c, [17](#)
  - ErriezOregonTHN128Transmit.h, [19](#)
- OregonTHN128\_TxEnd
  - ErriezOregonTHN128Transmit.c, [17](#)
- OregonTHN128\_TxRawData
  - ErriezOregonTHN128Transmit.c, [17](#)
  - ErriezOregonTHN128Transmit.h, [19](#)
- OregonTHN128Data\_t, [9](#)
  - channel, [9](#)
  - lowBattery, [9](#)
  - rawData, [10](#)
  - rollingAddress, [10](#)
  - temperature, [10](#)
- RF\_TX\_PIN\_DISABLE
  - ErriezOregonTHN128Transmit.c, [15](#)
- RF\_TX\_PIN\_INIT
  - ErriezOregonTHN128Transmit.c, [15](#)
- rawData
  - OregonTHN128Data\_t, [10](#)
- rollingAddress
  - OregonTHN128Data\_t, [10](#)
- SET\_TEMP
  - ErriezOregonTHN128.c, [12](#)
- src/ErriezOregonTHN128.c, [11](#)
- src/ErriezOregonTHN128Transmit.c, [14](#)
- src/ErriezOregonTHN128Transmit.h, [18](#)
- temperature
  - OregonTHN128Data\_t, [10](#)