

Erriez Oregon THN128 433MHz temperature sensor library for Arduino
1.0.0

Generated by Doxygen 1.8.13

Contents

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

5	File Documentation	11
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/ErriezOregonTHN128Receive.c File Reference	14
5.2.1	Detailed Description	15
5.3	src/ErriezOregonTHN128Receive.h File Reference	15
5.3.1	Detailed Description	16
5.4	src/ErriezOregonTHN128Transmit.c File Reference	17
5.4.1	Detailed Description	18
5.4.2	Macro Definition Documentation	18
5.4.2.1	RF_TX_PIN_DISABLE	18
5.4.2.2	RF_TX_PIN_INIT	19
5.4.3	Function Documentation	19
5.4.3.1	OregonTHN128_Transmit()	19
5.4.3.2	OregonTHN128_TxBegin()	19
5.4.3.3	OregonTHN128_TxEnd()	20
5.4.3.4	OregonTHN128_TxRawData()	20
5.5	src/ErriezOregonTHN128Transmit.h File Reference	20
5.5.1	Detailed Description	21
5.5.2	Function Documentation	21
5.5.2.1	OregonTHN128_Transmit()	21
5.5.2.2	OregonTHN128_TxBegin()	22
5.5.2.3	OregonTHN128_TxRawData()	22

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:

OregonTHN128Data_t	
Data structure	9

Chapter 3

File Index

3.1 File List

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

src/ ErriezOregonTHN128.c	
Oregon THN128 433MHz temperature transmit/receive library for Arduino	11
src/ ErriezOregonTHN128.h	??
src/ ErriezOregonTHN128Receive.c	
Oregon THN128 433MHz temperature transmit/receive library for Arduino	14
src/ ErriezOregonTHN128Receive.h	
Oregon THN128 433MHz temperature receive library for Arduino	15
src/ ErriezOregonTHN128Transmit.c	
Oregon THN128 433MHz temperature transmit library for Arduino	17
src/ ErriezOregonTHN128Transmit.h	
Oregon THN128 433MHz temperature transmit library for Arduino	20

Chapter 4

Class Documentation

4.1 OregonTHN128Data_t Struct Reference

Data structure.

```
#include <ErriezOregonTHN128.h>
```

Public Attributes

- uint32_t [rawData](#)
- uint8_t [rollingAddress](#)
- uint8_t [channel](#)
- int16_t [temperature](#)
- bool [lowBattery](#)

4.1.1 Detailed Description

Data structure.

Definition at line 63 of file ErriezOregonTHN128.h.

4.1.2 Member Data Documentation

4.1.2.1 channel

```
uint8_t OregonTHN128Data_t::channel
```

Channel

Definition at line 66 of file ErriezOregonTHN128.h.

4.1.2.2 lowBattery

```
bool OregonTHN128Data_t::lowBattery
```

Low battery indication

Definition at line 68 of file ErriezOregonTHN128.h.

4.1.2.3 rawData

```
uint32_t OregonTHN128Data_t::rawData
```

Raw data

Definition at line 64 of file ErriezOregonTHN128.h.

4.1.2.4 rollingAddress

```
uint8_t OregonTHN128Data_t::rollingAddress
```

Rolling address

Definition at line 65 of file ErriezOregonTHN128.h.

4.1.2.5 temperature

```
int16_t OregonTHN128Data_t::temperature
```

Temperature

Definition at line 67 of file ErriezOregonTHN128.h.

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

- src/ErriezOregonTHN128.h

Chapter 5

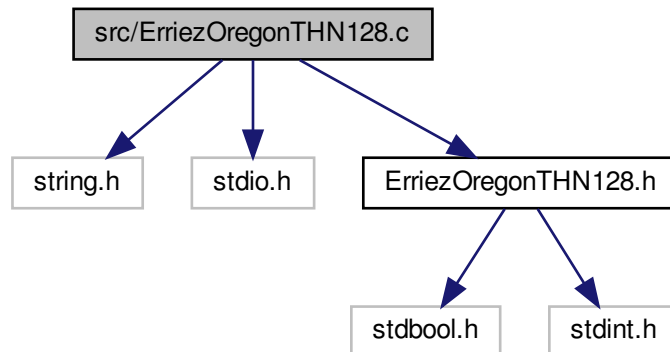
File Documentation

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

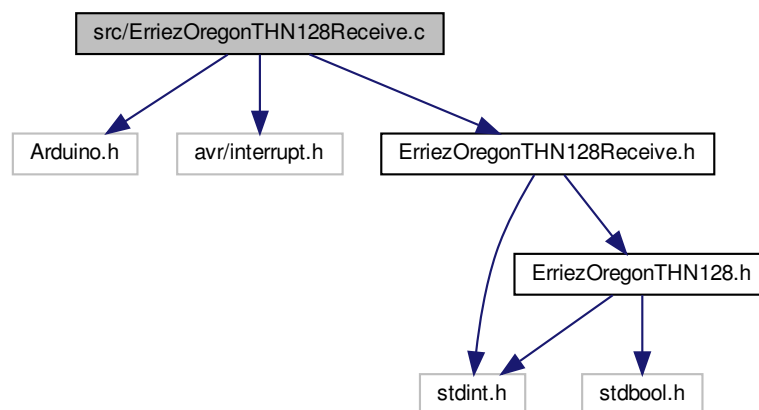
<i>temperatureStr</i>	Character buffer
<i>temperatureStrLen</i>	Size of character buffer
<i>temperature</i>	Input temperature

Definition at line 103 of file ErriezOregonTHN128.c.

5.2 src/ErriezOregonTHN128Receive.c File Reference

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

```
#include <Arduino.h>
#include <avr/interrupt.h>
#include "ErriezOregonTHN128Receive.h"
Include dependency graph for ErriezOregonTHN128Receive.c:
```



Enumerations

- enum **RxState_t** {
StateSearchSync = 0, **StateMid0** = 1, **StateMid1** = 2, **StateEnd** = 3,
StateRxComplete = 4 }

Functions

- void **rfPinChange** (void)
- void **OregonTHN128_RxBegin** (uint8_t extIntPin)
- void **OregonTHN128_RxEnable** ()
- void **OregonTHN128_RxDisable** ()
- bool **OregonTHN128_Available** ()
- bool **OregonTHN128_Read** (OregonTHN128Data_t *data)

5.2.1 Detailed Description

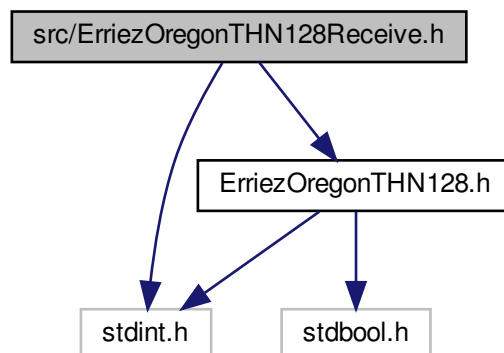
Oregon THN128 433MHz temperature transmit/receive library for Arduino.

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

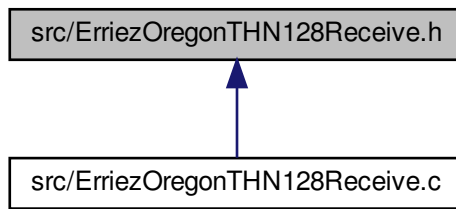
5.3 src/ErriezOregonTHN128Receive.h File Reference

Oregon THN128 433MHz temperature receive library for Arduino.

```
#include <stdint.h>
#include "ErriezOregonTHN128.h"
Include dependency graph for ErriezOregonTHN128Receive.h:
```



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



Functions

- void **OregonTHN128_RxBegin** (uint8_t extIntPin)
- void **OregonTHN128_RxEnable** ()
- void **OregonTHN128_RxDisable** ()
- bool **OregonTHN128_Available** (void)
- uint32_t **OregonTHN128_GetRawData** ()
- bool **OregonTHN128_Read** ([OregonTHN128Data_t](#) *data)

5.3.1 Detailed Description

Oregon THN128 433MHz temperature receive library for Arduino.

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

Protocol:

Transmit temperature twice every 30 seconds:

Bit: 0 7 0 7 0 7 0 7 +-----+-----+-----+-----+ +-----+-----+ |PREA|SYNC|Byte 0|Byte 1|Byte 2|Byte 3|
 |PREA|SYNC|Byte 0| ... +-----+-----+-----+-----+ /-----+-----+ |<----- 144ms -----
 —>|<- 100ms ->| 30 sec

Logic '0': Logic '1': +-----+ +-----+ | +-----+ +-----+ 1400 1500 1500 1400 (us)

PREA: Preamble 12x logic '1', 3000us low

SYNC: +-----+ | |

- +-----+ 5500us 5500us

Byte 0:

- Bit 0..3: Rolling address (Random value after power cycle)
- Bit 6..7: Channel: (0 = channel 1 .. 2 = channel 3)

Byte 1:

- Bit 0..3: TH3
- Bit 4..7: TH2

Byte 2:

- Bit 0..3: TH1
- Bit 5: Sign
- Bit 7: Low battery

Byte 3:

- Bit 0..7: CRC

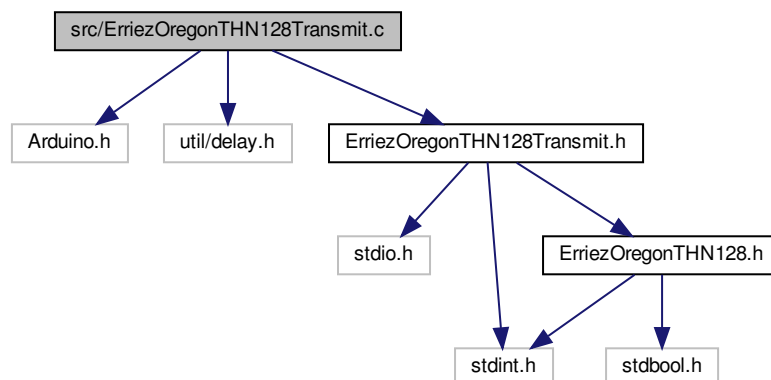
Example: Rolling address = 5, channel = 1, temperature = 27.8 'C, low battery = false TH1 = 2, TH2 = 7, TH3 = 8:
 Byte 0: 0x05 Byte 1: 0x78 Byte 2: 0x02 Byte 3: 0x7f

Bits in time: PRE=1 S B0=0x05 B1=0x78 B2=0x02 B3=0x7f 111111111111 S 10100000 00011110 01000000
 11111110

5.4 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.

5.4.1 Detailed Description

Oregon THN128 433MHz temperature transmit library for Arduino.

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

5.4.2 Macro Definition Documentation

5.4.2.1 RF_TX_PIN_DISABLE

```
#define RF_TX_PIN_DISABLE( )
```

Value:

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

5.4.2.2 RF_TX_PIN_INIT

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

Value:

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

5.4.3 Function Documentation

5.4.3.1 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.

5.4.3.2 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.

5.4.3.3 OregonTHN128_TxEnd()

```
void OregonTHN128_TxEnd (
    void )
```

Disable transmit.

Set transmit pin to input

Definition at line 149 of file ErriezOregonTHN128Transmit.c.

5.4.3.4 OregonTHN128_TxRawData()

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

Transmit data.

Parameters

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

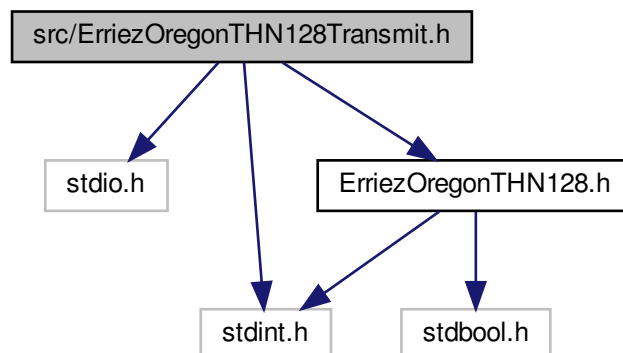
Definition at line 160 of file ErriezOregonTHN128Transmit.c.

5.5 src/ErriezOregonTHN128Transmit.h File Reference

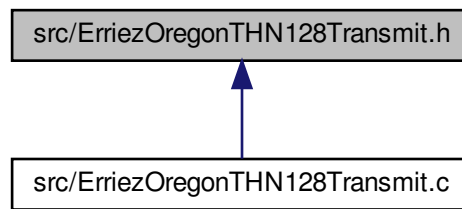
Oregon THN128 433MHz temperature transmit library for Arduino.

```
#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.

5.5.1 Detailed Description

Oregon THN128 433MHz temperature transmit library for Arduino.

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

5.5.2 Function Documentation

5.5.2.1 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.

5.5.2.2 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.

5.5.2.3 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, [19](#)
 OregonTHN128_TxBegin, [19](#)
 OregonTHN128_TxEnd, [19](#)
 OregonTHN128_TxRawData, [20](#)
 RF_TX_PIN_DISABLE, [18](#)
 RF_TX_PIN_INIT, [18](#)

ErriezOregonTHN128Transmit.h
 OregonTHN128_Transmit, [21](#)
 OregonTHN128_TxBegin, [22](#)
 OregonTHN128_TxRawData, [22](#)

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, [19](#)
 ErriezOregonTHN128Transmit.h, [21](#)

OregonTHN128_TxBegin
 ErriezOregonTHN128Transmit.c, [19](#)
 ErriezOregonTHN128Transmit.h, [22](#)

OregonTHN128_TxEnd
 ErriezOregonTHN128Transmit.c, [19](#)

OregonTHN128_TxRawData
 ErriezOregonTHN128Transmit.c, [20](#)
 ErriezOregonTHN128Transmit.h, [22](#)

OregonTHN128Data_t, [9](#)
 channel, [9](#)
 lowBattery, [9](#)
 rawData, [10](#)
 rollingAddress, [10](#)
 temperature, [10](#)

RF_TX_PIN_DISABLE
 ErriezOregonTHN128Transmit.c, [18](#)

RF_TX_PIN_INIT
 ErriezOregonTHN128Transmit.c, [18](#)

rawData
 OregonTHN128Data_t, [10](#)

rollingAddress
 OregonTHN128Data_t, [10](#)

SET_TEMP
 ErriezOregonTHN128.c, [12](#)

src/ErriezOregonTHN128.c, [11](#)

src/ErriezOregonTHN128Receive.c, [14](#)

src/ErriezOregonTHN128Receive.h, [15](#)

src/ErriezOregonTHN128Transmit.c, [17](#)

src/ErriezOregonTHN128Transmit.h, [20](#)

temperature
 OregonTHN128Data_t, [10](#)