

NTPTime Library

Library Version: 2.0
Library Reference: <https://github.com/rrobinet/NTPTime>
Document Version: 2.0
Save Date: 10 December 2017

Introduction

The NTP Time library allows gathering time from appropriate timeserver pool.

It automatically adjusts the time according to the daylight saving time and the days according to the leap years.

It is aimed to work transparently with a variety of processing devices accessing the Internet such as:

- Arduino Ethernet and Arduino + Ethernet shield
 - ESP32 (typically WEMOS LOLIN32)
 - ESP8266 (Typically WEMOS mini D1 R2)
 - It is a modified version of the one of one of Andreas Spiess V1.0 2016-6-28; see:
 - <https://github.com/SensorsIot/NTPtimeESP/blob/master/NTPtimeESP.h>
- and
- <http://www.sensorsiot.org>
 - https://www.youtube.com/channel/UCu7_D0o48KbfhpEohoP7YSQ

Description

Board Type

To access a NTP server, the processor board should be equipped with an Ethernet or WiFi interface.

For Arduino, two configuration are tested, using an Arduino UNO / ATMEGA 2560 + an Ethernet shield or an Arduino Ethernet

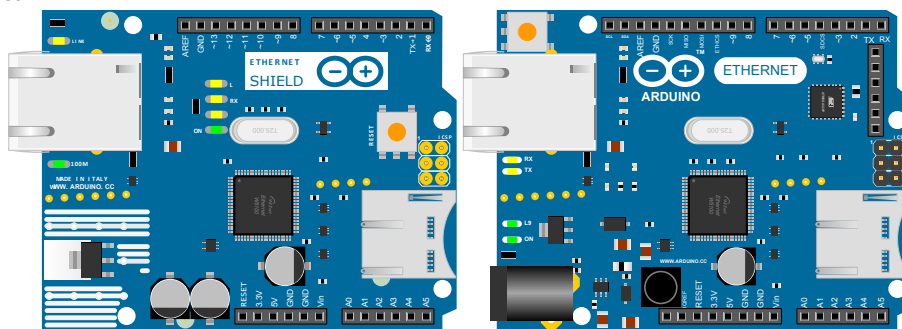


Figure 1: Arduino Ethernet Shield and Arduino Ethernet

For Expressif devices which include a Wifi interface, two modules are currently tested:

EDP2866 WeMos mini D1 or WeMos mini pro

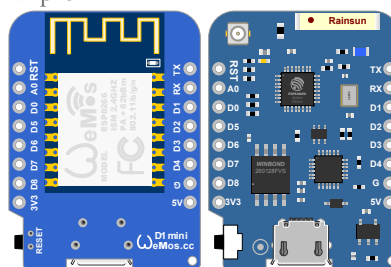


Figure 2: WeMos D1 mini and Pro

And ESP32 WeMos Lolin32

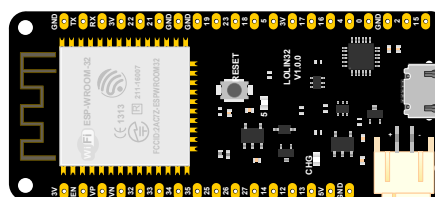


Figure 3: WeMos Lolin32

NTPTime Library

The Library is made as transparent as possible; this means that it already includes the Ethernet, a WiFi UDP library that doesn't have to be include in the sketch.

It also provides board type recognition based on the board description such as

ARDUINO_LOLIN32 for WeMos Lolin32

ARDUINO_ESP8266_WEMOS_D1MINI and ARDUINO_ESP8266_ESP01 for ESP8266 devices

And (ARDUINO_AVR_ETHERNET for Arduino ones.

Through this recognition, appropriate Ethernet libraries are activated

NTPTime variables

Variables are:

- NTP parse packet time-out
Default: RECV_TIMEOUT_DEFAULT = 1 second
Maximum: MAX_RECV_TIMEOUT = 60 seconds
- NTP get time time-out
Default: SEND_INTRVL_DEFAULT = 10 second
- NTP Packet Size
NTP_PACKET_SIZE = 48;

NTPTime Data structure

One structure is defined as:

```
struct strDateTime
{
    byte hour;
    byte minute;
    byte second;
    int year;
    byte month;
    byte day;
    byte dayofWeek;
    String amPm = " ";
    boolean valid;
};
```

Such a structure shall be given a local instance name to be able to store the NTP data.

Note: The dayofWeek is a value from 1 to 7 with 1 equal to Sunday.

Example

```
strDateTime <DataInstanceName>
```

NTPtime Class

Syntax

NTPtime::NTPTime

NTPtime::strDateTime getNTPtime

NTPtime:: setSendInterval

NTPtime:: setRecvTimeout

NTPtime::printDateTime

NTPTime

Description

Defines a NTPtime instance associated to a NTP valid server pool.

There is an NTP Server pool in each country and are globally available under the DNS format:

<DNS Top Domain Country Code>.pool.ntp.org

See <http://www.pool.ntp.org/en/use.html>

The preferred pool is of course the one which is the closest to the application (typically the one of the local country)

To configure a NTP server use the command

Syntax

NTPtime.NTPtime (String NTPtime);

Parameters

NTPTime Library

Instance NTPtime:

Name of the NTP instance used by the sketch

String NTPtime:

Valid DNS name of NTP server pool (e.g.; us.pool.ntp.org, ntp1.teambelgium.net,...)

Default: None

Example

```
NTPtime NTPbe (be.pool.ntp.org) // NTP server pool instance NTPbe is associated to the Belgium NTP server pool
```

strDateTime getNTPtime

Description

This function gets NTP Unix time from the NTP server, validates the response, adjust the time zone and convert it to a readable data returned into the local data structure defined by the sketch.

Syntax

```
strDateTime NTPtime::getNTPtime(float _timeZone, byte _DayLightSaving, boolean _amPm, unsigned long _unixTime)
```

Parameters

- *_timeZone:*
A floated value that corresponds to the city time zone defined as a GMT offset. Typically (GMT+1) or 1.0 for Central Europe
We note that for India, the time zone is to be defined with a decimal point typically GMT+5.30 or 5.5
- *DayLightSaving:*
Adjust the Daylight saving time the winter or summer season according to the European or US scheme
Note: For US, since 2007 DST begins on second Sunday of March and ends on first Sunday of November. Time change occurs at 2AM locally
Values:
0: means no time adjusts
1: means European time adjust
2: means US time adjust
Other: no time adjusts
- *amPm:*
Used to select 12 or 24 hours display
Values:
0: means 24 hours display
1: means 12 hours AM / PM display
Other: 24 hours display
- *unixTime:*
Indicate that a conversion from UNIX time to Local time is required
Values:
0: No conversion required, true NTP time get request
non 0: A valid UNIX time to convert to Local time

Returns

Data structure *valid* variable true if NTP data is validated, else, return false

Example

```
strDateTime dateTime; // Define a dateTime structure instance
.....
// Request time; Time Zone is Centra Europ (1.0) and with summertime adjustment active
dateTime = NTPbe.getNTPtime(1.0, 1);
if(dateTime.valid)
{
    Serial.println ();
    Serial.print ("Date: "), Serial.print (dateTime.day), Serial.print ("-"), Serial.print (dateTime.month),
    Serial.print ("-"), Serial.println (dateTime.year);
}
```

setSendInterval

Description

This Boolean function allows changing the NTP request frequency while looping on the function. Unit value is second with a minimum of 10 second (see SEND_INTRVL_DEFAULT)

Syntax

NTPTime Library

NTPTime.setSendInterval(unsigned long _sendInterval);

Parameters

- *_sendInterval:*
Expected loop delay between NTP queries.
Unit is second
Default and Minimum is 10 seconds

Returns

False is value less than 10 sec (set to default)

Example

```
NTPbe.setSendInterval (10); // Set NTP request interval time out to 10 seconds
```

setRecvTimeout

Description

This Boolean function used as watchdog while parsing NTP packets. Unit value is second with a minimum of 0 and a maximum of 60 (see MAX_RECV_TIMEOUT)

Syntax

NTPTime.setRecvTimeout(unsigned long _recvTimeout);

Parameters

- *_recvTimeout:*
Watchdog time-out while parsing the NTP packets.
Default is 1 second
Maximum is 60 seconds

Returns

False is value above 60 seconds

Example

```
NTPbe.setRecvTimeout (2); // Set NTP parsing time out to 2 seconds
```

printDateTime

Description

Use to print over the serial line the contents of the data structure after a valid NTP fetch.

Syntax

NTPTime.printDateTime(strDateTime _dateTime);

Parameters

- *_dateTime:*
The name of the defined data structure instance

Returns

Serial print of the data structure with the following format:

```
Sunday-24-December-2017 11:55.0  
<dayOfWeek>-<day>-<month>-<year> <hour>:<minute>.<second> [<AM><PM>]
```

Example

```
strDateTime dateTime; // Define a dateTime structure instance  
.....  
NTPbe.printDateTime(dateTime);
```

Usage Example

The use of the NTP library returns a formatted data structure with the:
Hour, minute, second, year, month, day, dayOfWeek.

```
/*  
This sketch is an example of using the NTPTime library to get local time from an NTP server  
The NTPtime library adapts automatically the day and the hour according to the leap year and  
daylight saving time  
It also configure the appropriate library according to the processor type like:  
ESP32 Devices such as the Wemos Lolin 32  
ESP8266 Wemos Mini D1 and D2 and generic ESP8266  
Arduino Ethernet
```

NTPTime Library

The NTPTime library includes the WiFi, Ethernet and UDP libraries that do not need to be included in the sketch

However the sketch should also be adapted to the appropriate Ethernet IP settings for the particular device

Typically ESP32 and ESP8266 (Wifi) or Arduino Ethernet (Ethernet)

Note that in both cases DHCP is used

Appropriate NTP Time server pool may have to be selected according the particular country, to improve the reply time

More information of the library function are described in the library code.

*/

```
#include <NTPtime.h>                                // Include the NTPTime library

String NTPServer = "be.pool.ntp.org";
NTPtime NTPbe(NTPServer);                          // Choose server pool as required (here Belgium)
strDateTime dateTime;                               // Define a dateTime instance
float GMT;                                          // GMT ofset use decimal value; eg. 1.0
byte dayLight;                                    // Day Light saving time option (0: none - 1 European; 2-USA)
boolean amPm;                                     // 24 /12 option (1: 12 housr)
long int unixTime;                                // UNIX time to convert

void setup()
{
    Serial.begin(115200);
#ifdef ARDUINO_AVR_ETHERNET                        // Configure the Ethernet / IP in case of Arduino Ethernet
    static byte myMac[] = {0x0A, 0x0B, 0x0C, 0x0D, 0x0E, 0x0F}; // Server MAC address (Locally Administrated)
    Serial.println("Connecting to DHCP Server");
    while (Ethernet.begin(myMac) == 0)              // Wait until IP address is configured
    {
        Serial.print(".");
        delay (500);
    }
    Serial.println ("Connected to IP Network");
#else
    ESP8266
    const char *ssid      = "HomeT";                // Set you WiFi SSID for ESP32 or ESP8266
    const char *password  = "creative";              // Set you WiFi password for ESP32 or ESP8266
    Serial.println("Connecting to Wi-Fi");
    WiFi.mode(WIFI_STA);
    WiFi.begin (ssid, password);
    while (WiFi.status() != WL_CONNECTED)
    {
        Serial.print(".");
        delay(500);
    }
    Serial.println("WiFi connected");
#endif
    NTPbe.setSendInterval (10);                      // Set NTP request interval time out to 10 seconds
    NTPbe.setRecvTimeout (2);                        // Set NTP request receive time out to 2 seconds
}

void loop() {
    // first parameter: Time zone in floating point (for India); second parameter: 1 for European summer time; 2 for US daylight saving time (contributed by viewwer, not tested by me), third parameter is AM/PM option
    // fourth parameter: is UNIX time to convert (0 means actual NTP time)
    GMT = 1.0;                                       // Central Europe
    dayLight = 1;                                   // Europe
    amPm = 0;                                       // 24 hours format
    unixTime = 1514112900;                          // if != 0; Unix time to convert to local time
    Serial.print ("UNIX time: "); Serial.print (unixTime) ; Serial.print (" - GMT = "),
    Serial.print (GMT); Serial.print (" - DayLightSaving Time is: "), Serial.print (dayLight);
    Serial.print (" - 12 hours Display is set to: "), Serial.println (amPm?"TRUE":"FALSE");
    Serial.print ("Local Time: ");
    NTPbe.printDateTime(NTPbe.getNTPtime(GMT, dayLight, amPm, unixTime)); // Example using the printDateTime function

    delay (20000);
    GMT = 1.0;                                       // Central Europe
    dayLight = 0;                                   // None
    amPm = 1;                                       // 12 hours format
    unixTime = 0;                                   // if != 0; Unix time to convert to local time
}
```

NTPTime Library

```
    dateTime = NTPbe.getNTPtime(GMT, dayLight, amPm, unixTime);           // Call NTP get or
time conversion
    // check dateTime.valid before using the returned time
    // Use "setSendInterval" or "setRecvTimeout" if require
    if(dateTime.valid)
    {
        Serial.print("Actual time from NTP Server: "); Serial.println (NTPServer);
        Serial.print ("GMT = "), Serial.print (GMT); Serial.print (" - DayLightSaving Time is: "),
Serial.print (dayLight);
        Serial.print (" - 12 hours Display is set to: "), Serial.println (amPm?"TRUE":"FALSE");
Serial.print ("NTP Time: ");
        NTPbe.printDateTime(dateTime);                                // Example using the
printDateTime function
    }
}
```

Returns the following data:

For ESP devices

Connecting to DHCP Server

Connected to IP Network

UNIX time: 1514112900 - GMT = 1.00 - DayLightSaving Time is: 1 - 12 hours Display is set to: FALSE
Local Time: Sunday-24-December-2017 11:55.0

UNIX time: 1514112900 - GMT = 1.00 - DayLightSaving Time is: 1 - 12 hours Display is set to: FALSE
Local Time: Sunday-24-December-2017 11:55.0

Actual time from NTP Server: be.pool.ntp.org
GMT = 1.00 - DayLightSaving Time is: 0 - 12 hours Display is set to: TRUE
NTP Time: Tuesday-26-December-2017 3:19.27 PM

UNIX time: 1514112900 - GMT = 1.00 - DayLightSaving Time is: 1 - 12 hours Display is set to: FALSE
Local Time: Sunday-24-December-2017 11:55.0
2017-12-8-6 17H 45M 25S

- ooOoo-