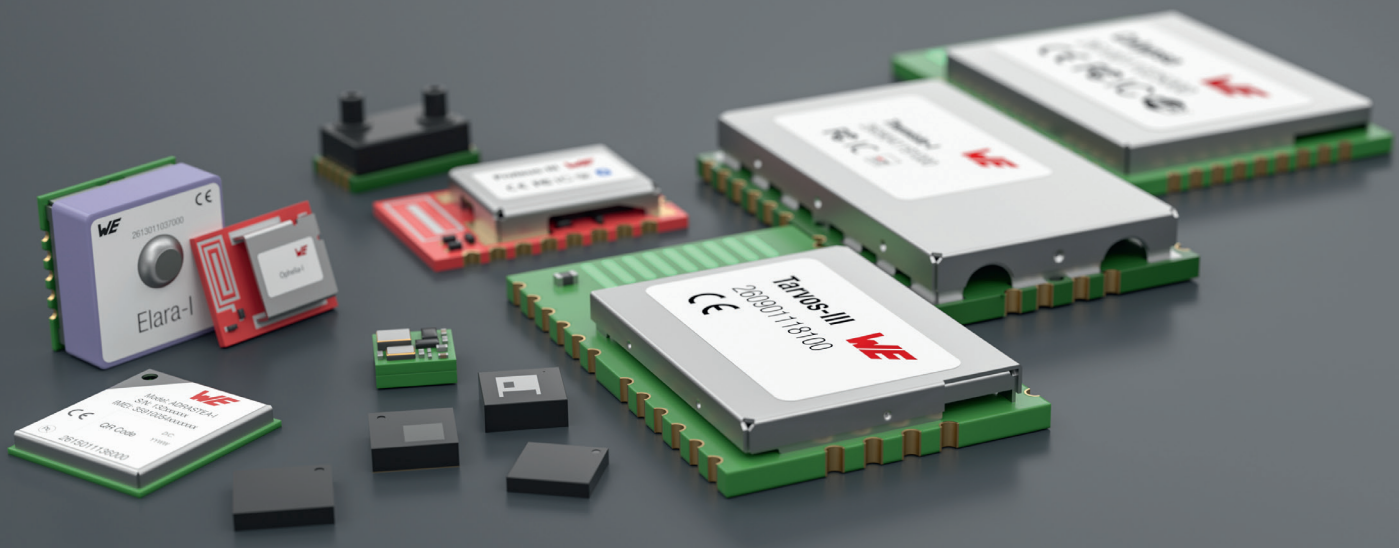

ANR028

Calypso Transparent Mode UART-to-WiFi Bridge

Version 1.1

July 19, 2023

WÜRTH ELEKTRONIK MORE THAN YOU EXPECT

Revision history

Manual version	Notes	Date
1.0	• Initial version	January 2022
1.1	• Updated `Important notes`, meta data and document style	July 2023

Contents

1 Introduction

The Calypso WLAN module developed by Würth Elektronik eiSos is intended to be used as a radio sub-system in order to provide WLAN (IEEE 802.11) communication capabilities to systems. The UART acts as the primary interface between the module and a host micro-controller. The module can be fully configured and controlled using a set of AT-commands sent as messages via UART. Once configured, the module independently manages WLAN connectivity allowing the host controller to utilize its resources for its application tasks.

Firmware version 2.0.0 of the radio module now supports the so-called "Transparent Mode" mode of operation, which when chosen instead of the "AT commands mode", allows applications to utilize the module as a UART-to-WiFi bridge.
In transparent mode, the Calypso automatically connects to a pre-configured Wi-Fi access point and opens a socket for communication with a preconfigured remote endpoint (TCP server, TCP client or UDP endpoint). Afterwards, the Calypso acts as a transparent bridge between the UART and the created socket. This means that all data sent to the Calypso via UART is forwarded to the socket and all data received on the socket is output on the UART.

This application note gives a short overview of this new feature.

2 Functional description

When booting the Calypso radio module, the voltage level of the pins *APP_MODE_0* and *APP_MODE_1* is checked to detect the mode of operation. In case both pins have a high level, the Calypso starts in transparent mode.
In transparent mode, the Calypso first tries to connect to a WiFi access point. In case this fails due to the absence of the configured access point or wrong access point credentials, it retries until a connection is established.
On success, the pin *STATUS_IND_0* turns high and the Calypso tries to open a socket. In case of failure, it retries until a socket has been opened.
As soon as a socket has been opened and the connection has been established successfully, the pin *STATUS_IND_1* turns high and the UART of the Calypso is switched on. All UART data is forwarded to the socket and vice versa. In this state the pin */RTS* of the Calypso indicates when the radio module is ready to receive data via UART. If the UART flow control is disabled, the pin stays active (LOW) as long as a socket is available. In case the UART flow control is enabled, the */RTS* pin stays active (LOW) as long as a socket is available and the UART is ready to receive more data.

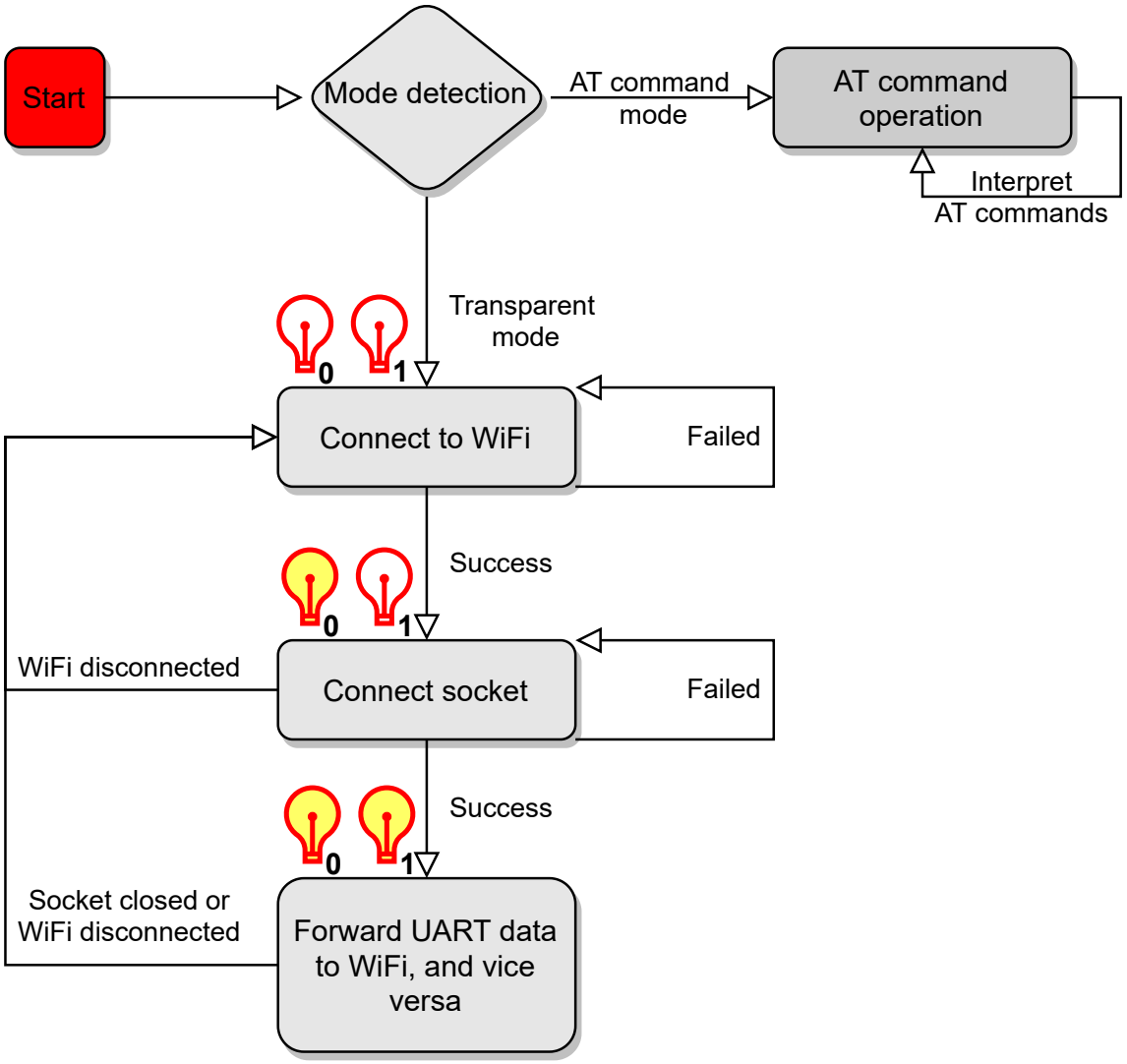

Figure 1: Flow chart - transparent mode

Before starting the module in transparent mode, various settings have to be defined in order to ensure successful WiFi connection and socket setup. For details regarding the required steps, please refer to the following chapters.

2.1 Power save mode

For battery powered systems with a requirement that the devices always remain online (i.e. connected to WiFi Network and socket but UART disabled), the Calypso offers a power save feature. This feature allows significant power saving while staying connected to the Wi-Fi network as well as sustaining an active connection. An average of less than 2 mA current consumption is achieved in station mode with an active socket connection. This power save feature is supported in the transparent mode, too. Steps to enable the power save feature are described in chapter 4.
As the module's UART RX is disabled in this state, the host is required to wake up the module

before sending data to it. Especially in power save mode, using UART flow control is recommended for any UART baudrate.
After a wake-up by the host (via *WAKE-UP* pin), it can send data and trigger transmission of one radio packet. The module will go back to power save automatically after the radio packet was sent. Another UART transfer from the host to the module must be preceded with a new wakeup by the host.
If data is received on the radio side it will be forwarded on the UART to the host. During that process, the host is not allowed to send data to the module.

The wakeup pin shall not be used during an active UART transfer from Calypso to host.

When in power save mode, the Calypso is ready to receive data on the UART 10 ms after a rising edge on the *WAKE-UP* pin.

The on-board HTTP server will not be available in the power save mode.

3 Wi-Fi connection setup

To successfully set up a connection to a WiFi access point, the user has to enter the network credentials such as SSID, password and security mode into the Calypso in advance. There are two ways of doing so:

1. Use the AT-commands interface to configure the Calypso via UART
2. Use the web interface to configure the module

3.1 Configure Wi-Fi via AT command interface

To enter the credentials of the access point via AT commands, the Calypso radio module must be started in "AT command mode" by applying a low level at the pins *APP_MODE_0* and *APP_MODE_1* during start-up.

If this has been done, the credentials can be added in two ways. Either to use the command `AT+wlanConnect` to connect to an access point, or to use the command `AT+wlanProfileAdd` to simply add the profile to the radio module (see user manual [1]).
Then the module automatically connects to one of the configured access points after device reset, in case the WLAN connection policy is set to `Auto` (or `Auto|Fast`).

```
AT+wlanPolicySet=connection,Auto|Fast,
```

Code 1: Example AT+wlanPolicySet

The policy is automatically set to `Auto|Fast` when the module is started in transparent mode.

3.1.1 AT+wlanConnect

```
AT+wlanConnect=[SSID],[BSSID],[SecurityType],[SecurityKey],[SecurityExtUser],[
    SecurityExtAnonUser],
[SecurityExtEapMethod]
```

Code 2: AT+wlanConnect

```
AT+wlanConnect=mySSID,,WPA_WPA2,myPassword,,,
```

Code 3: Example AT+wlanConnect

3.1.2 AT+wlanProfileAdd

The command `AT+wlanProfileAdd` allows you to store up to seven preferred WLAN profiles which you can use to specify multiple access points to be used in transparent mode. Each profile stores the access point credentials along with a profile priority, which determines the order of connection.

```
AT+wlanProfileAdd=[SSID],[BSSID],[SecurityType],[SecurityKey],[SecurityExtUser],
[SecurityExtAnonUser],[SecurityExtEapMethod],[priority]
```

Code 4: AT+wlanProfileAdd

The following example adds a profile with the highest priority (15).

```
AT+wlanProfileAdd=mySSID,,WPA_WPA2,myPassword,,,,15
```

Code 5: Example AT+wlanProfileAdd

3.2 Configure Wi-Fi via web interface

To enable easy configuration when integrated into an embedded system with limited HMI capabilities, the Calypso offers a provisioning mode. In this mode, the module acts as an AP and allows external devices with appropriate credentials to connect and access the on-board HTTP server. The user can conveniently browse the settings web page and configure the module using any web browser.

The web pages for provisioning require JavaScript.

3.2.1 Start in provisioning mode

There are two ways to set the Calypso to provisioning mode.

1. When starting the module in AT command mode the command

   ```
   AT+provisioningStart
   ```

   starts the provisioning.

2. Alternatively, apply a LOW signal to the *APP_MODE_0* pin, a HIGH signal to the *APP_MODE_1* pin and restart the module.

When the provisioning mode has been started successfully, the LED at *STATUS_IND_1* flashes with an interval of 1 s. The module has created an access point with a SSID "calypso_" followed by the MAC of the module (example "calypso_CAFFEE123456"). Now any Wi-Fi enabled device can connect to the access point using WPA2 security and the key "calypsowlan".

3.2.2 Add WLAN profile

On the device connected to the Calypso AP, open the website "calypso.net" in a browser.

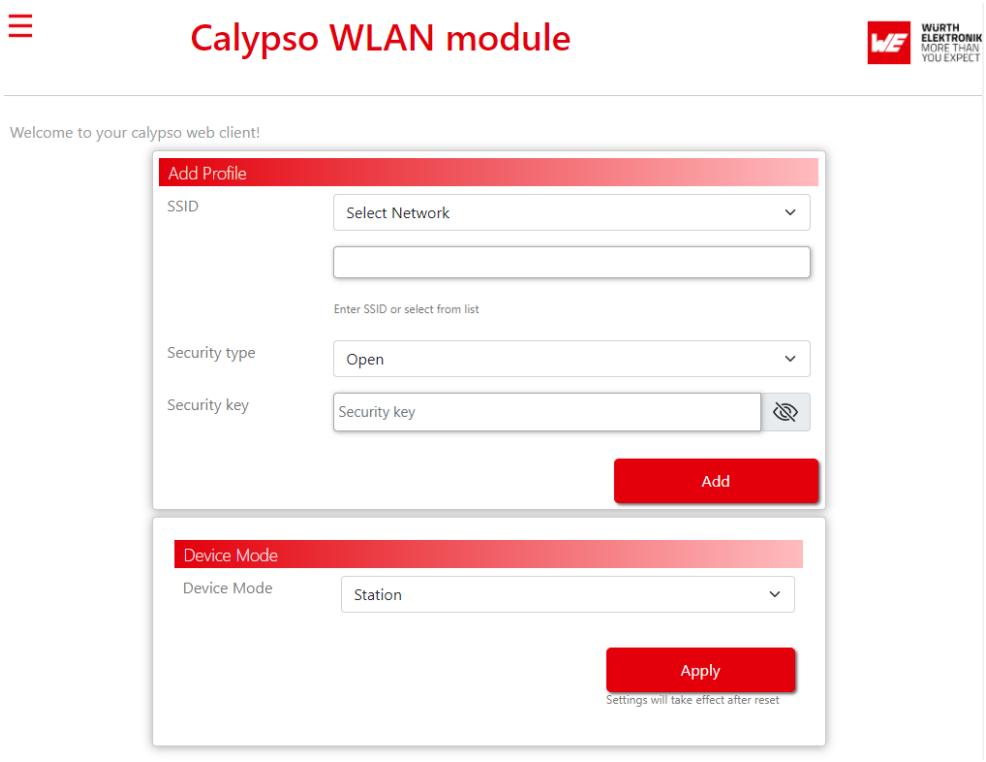

Figure 2: Provisioning main page

To save a WLAN profile in the module, select the SSID from the dropdown menu or enter the same manually in the text field. Check the correct security type, enter the key if necessary and click on the "Add" button. A pop-up appears confirming the addition of the profile.

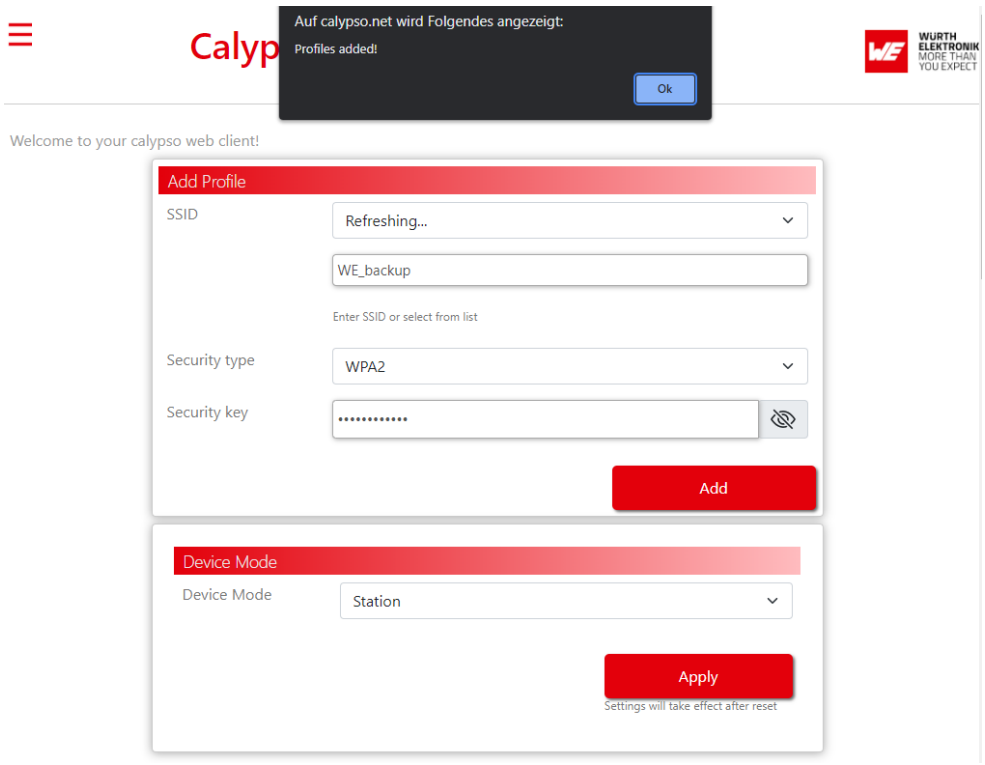

Figure 3: Provisioning main page

In provisioning mode, the module is set to start as an AP. In order to start the module in station mode, select "Station" in the "Device Mode" drop down menu and click on "Apply". Note that for the module to connect to one of the stored profiles after a reset or when the connection is lost, the WLAN connection policy has to be set to `Auto` (or `Auto|Fast`). The policy is automatically set to `Auto|Fast` when the module is started in transparent mode.

4 Socket setup

In case the Calypso radio module has successfully set up a connection to an access point in transparent mode, the pin *STATUS_IND_0* turns high. The next step is to open a socket.

The behavior of the module depends on the socket configuration:

- In case the module is configured to use a UDP socket, the socket will open and data transmission to the specified peer address can start immediately.
- In case the module is configured to use a TCP server socket, the Calypso radio module waits for a TCP client to connect to it, before data transmission can start.
- In case the module is configured to use a TCP client socket, the Calypso tries to connect to the TCP server at the specified peer address. As soon as the connection has been established, data transmission can start.

To enable the successful setup of a socket, various settings must be defined in advance. There are two ways of doing so:

1. Use the AT-commands interface to enter the socket settings via UART
2. Use the web interface to enter the socket settings

4.1 Configure the socket settings via AT commands

To enter the socket settings via AT commands, the Calypso radio module must be started in "AT command mode" by applying a low level at the pins *APP_MODE_0* and *APP_MODE_1* during start-up.
If this has been done, the socket type and information of the peer device can be specified using the `AT+set` command:

```
AT+set=transparent_mode,[option],[value1]
```

Code 6: AT+set

Available options:

- **power_save**: value1 determines whether the power save feature is activated or not (true or false)
- **remote_address**: value1 is the IP of the peer device
- **remote_port**: value1 is the port of the peer device
- **local_port**: value1 is the port of the Calypso (The local port must not be 80 or 8080.)
- **socket_type**:
  - value1 is **udp**: send/receive data to/from a UDP device at remote_address and remote_port
  - value1 is **tcp_server**: create a TCP server on local_port and send/receive data to/from the first peer device that connects

  - value1 is **tcp_client**: create a TCP connection to a TCP server at remote_address and remote_port and send/receive data to/from it

- **secure_method**:
  - value1 is **none**: no encryption and authentication used
  - value1 is **sslv3**: in case of a TCP socket, SSL v3 is used
  - value1 is **tlsv1**: in case of a TCP socket, TLS v1 is used
  - value1 is **tlsv1_1**: in case of a TCP socket, TLS v1.1 is used
  - value1 is **tlsv1_2**: in case of a TCP socket, TLS v1.2 is used
  - value1 is **sslv3_tlsv1_2**: in case of a TCP socket, the highest method between SSL v3 and TLS v1.2 is used

- **skip_date_verify**: value1 determines whether the date verification of the SSL certificate is skipped (true or false). This setting is only applicable if the secure method is not "none" and is useful if there is no SNTP server available in the network.

- **disable_cert_store**: value1 determines whether self-signed certificates should be accepted (true or false). This setting is only applicable if the secure method is not "none".

In case the parameter "secure_method" is not "none", additional parameters for the SNTP server and SSL certificate must be specified before a secure socket can be setup.
The SNTP server address can be specified via the command `AT+netappset`:

```
AT+netappset=sntp_client,server_address,0,time.google.com
```

Code 7: Example AT+netappset

Furthermore, the private key, SSL certificate and SSL root CA must be stored in the file system. To do so, the commands `AT+fileOpen`, `AT+fileWrite` and `AT+fileClose` must be used:

```
AT+fileOpen=[fileName],[options],[fileSize]
AT+fileWrite=[fileID],[offset],[format],[length],[data]
AT+fileClose=[fileID],[certificateFileName],[signature]
```

Code 8: Create a new file in file system

Depending on the file size, the use of multiple AT+fileWrite commands before the AT+fileClose command may be required to transfer the entire file's content to the file system. It is recommended to use chunks of not more than 1024 bytes per AT+fileWrite command.

The files must be stored at the following predefined locations in the file system:

File	fileName
Private key	user/transparentmode_privatekey
SSL certificate	user/transparentmode_certificate
SSL root CA	user/transparentmode_rootca

Table 1: Pre-defined file names

4.1.1 Examples

```
/* enable power save feature */
AT+set=transparent_mode,power_save,true

/* set socket type */
AT+set=transparent_mode,socket_type,udp
AT+set=transparent_mode,secure_method,none

/* set remote address */
AT+set=transparent_mode,remote_address,192.178.168.22
AT+set=transparent_mode,remote_port,5001

/* set local port */
AT+set=transparent_mode,local_port,5001
```

Code 9: Example UDP socket configuration

```
/* enable power save feature */
AT+set=transparent_mode,power_save,true

/* set socket type */
AT+set=transparent_mode,socket_type,tcp_client
AT+set=transparent_mode,secure_method,none

/* set remote address */
AT+set=transparent_mode,remote_address,192.178.168.22
AT+set=transparent_mode,remote_port,5001
```

Code 10: Example TCP client socket configuration

```
/* enable power save feature */
AT+set=transparent_mode,power_save,true

/* set socket type */
AT+set=transparent_mode,socket_type,tcp_server
AT+set=transparent_mode,secure_method,none

/* set local port */
AT+set=transparent_mode,local_port,5001
```

Code 11: Example TCP server socket configuration

```
/* enable power save feature */
AT+set=transparent_mode,power_save,true

/* set socket type */
AT+set=transparent_mode,socket_type,tcp_client
AT+set=transparent_mode,secure_method,sslv3_tlsv1_2

/* set remote address */
AT+set=transparent_mode,remote_address,192.178.168.22
AT+set=transparent_mode,remote_port,5001

/* use SNTP server for certificate date verification */
AT+netappset=sntp_client,server_address,0,time.google.com
AT+set=transparent_mode,skip_date_verify,false
```

```
/* verifies public root CA */
AT+set=transparent_mode,disable_cert_store,false
```

Code 12: Example SSL client socket configuration in network with internet access

```
/* enable power save feature */
AT+set=transparent_mode,power_save,true

/* set socket type */
AT+set=transparent_mode,socket_type,tcp_client
AT+set=transparent_mode,secure_method,sslv3_tlsv1_2

/* set remote address */
AT+set=transparent_mode,remote_address,192.178.168.22
AT+set=transparent_mode,remote_port,5001

/* disable certificate date verification */
AT+set=transparent_mode,skip_date_verify,true

/* use self signed certificate */
AT+set=transparent_mode,disable_cert_store,true

/* load root CA in file system */
AT+fileOpen=user/transparentmode_rootca,CREATE|OVERWRITE,...
AT+fileWrite=...
AT+fileClose=...
```

Code 13: Example SSL client socket configuration in network without internet access

```
/* enable power save feature */
AT+set=transparent_mode,power_save,true

/* set socket type */
AT+set=transparent_mode,socket_type,tcp_server
AT+set=transparent_mode,secure_method,sslv3_tlsv1_2

/* set local port */
AT+set=transparent_mode,local_port,5001

/* disable date verification as SNTP is not available */
AT+set=transparent_mode,skip_date_verify,true

/* use self signed certificate */
AT+set=transparent_mode,disable_cert_store,true

/* load private key in file system */
AT+fileOpen=user/transparentmode_privatekey,CREATE|OVERWRITE,...
AT+fileWrite=...
AT+fileClose=...

/* load certificate in file system */
AT+fileOpen=user/transparentmode_certificate,CREATE|OVERWRITE,...
AT+fileWrite=...
AT+fileClose=...
```

Code 14: Example SSL server socket configuration with self-signed certificate

4.2 Configure the socket settings via web interface

In order to configure the socket settings of the module via web interface, the module needs to be in provisioning mode. See chapter 3.2.1 for instructions on how to start the module's provisioning mode. Once in this mode, open the website "http://calypso.net/usersettings.html" in a browser on the device connected to the Calypso AP.

This webpage uses RESTful APIs in order to perform GET and POST requests on resources on the Calypso. A detailed description of these API calls can be found in chapter "The HTTP server interface" of the Calypso user manual [1].

This page offers two tabs to get and set the usersettings of the radio module. Perform the following steps in order to configure the socket settings for transparent mode.

- Open the "SET" tab.
- Select "transparent_mode" from the category drop-down.
- Select the required setting from the "Setting" drop-down. For example, "socket_type".
- Select or enter the value for this setting. For example, "SOCKET_TYPE_TCP_SERVER".
- Click on the "POST" button to set the configuration.

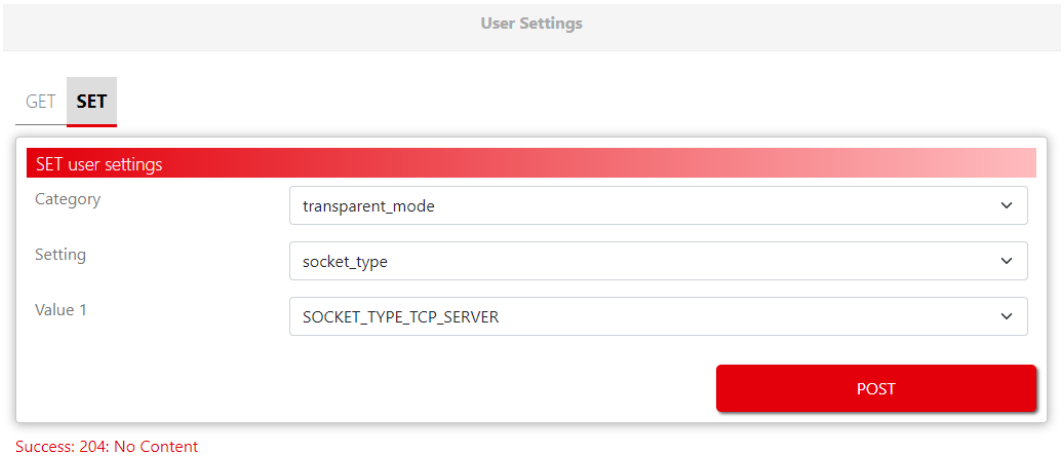

Figure 4: Configure socket setting

Follow the examples described in chapter 4.1.1 to completely configure the module's socket settings in order to work in transparent mode.

In the "GET" tab, select the category and the setting drop-downs to read the current values.

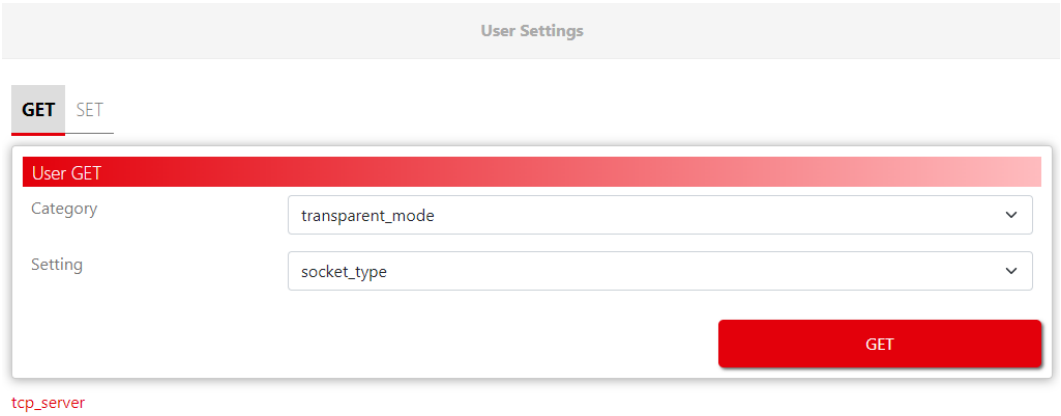

Figure 5: Read socket setting

The provisioning pages on the Calypso offer a possibility to upload certificates to the on-board file system. In order to upload a file, open the homepage "http://calypso.net/file.html" and:

- Click on the "File upload" option on the menu bar.
- Click on the "Choose file" button to open the file browser on the device.
- Browse and select the file to be uploaded.
- Click on "Upload file" to save the file on to the Calypso file system.
- All files are stored under the path "/user".

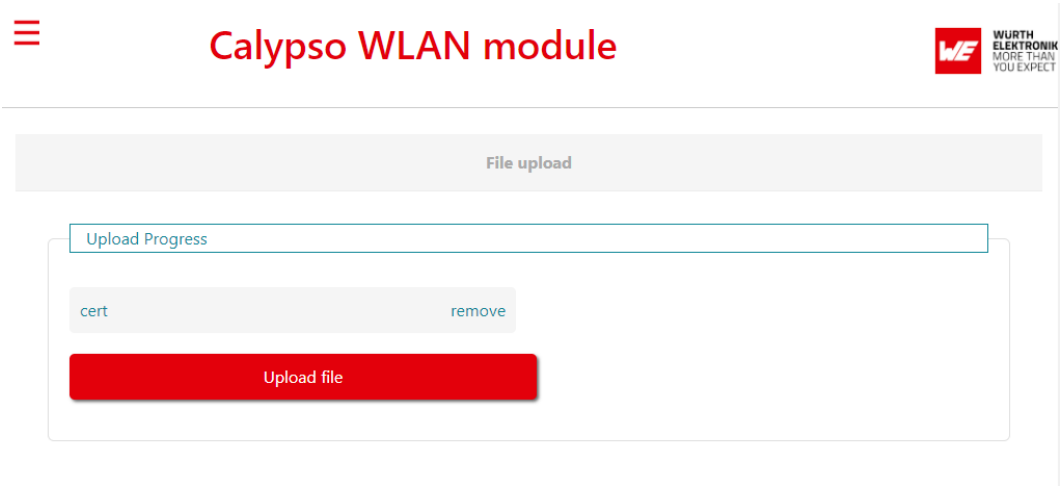

Figure 6: File upload

Follow the examples described in chapter 4.1.1 to completely configure the module's socket settings in order to work in transparent mode.

5 Data transmission

As soon as a socket has been opened successfully, the pin *STATUS_IND_1* turns high. Now data transmission can start. All data received on the socket will be sent via UART to the connected host controller. All data that is sent via UART to the Calypso radio module in transparent mode will be forwarded to the socket.

The trigger that causes the Calypso radio module to forward the data to the socket can be configured in the user settings of the module. There are two ways of setting the trigger:

1. Use the AT-commands interface to enter the UART trigger settings via UART
2. Use the web interface to enter the UART trigger settings

5.1 Configure the UART trigger settings via AT commands

To enter the UART trigger settings via AT commands, the Calypso radio module must be started in "AT command mode" by applying a low level at the pins *APP_MODE_0* and *APP_MODE_1* during start-up.
If this has been done, the UART trigger settings can be specified using the `AT+set` command:

```
AT+set=UART,[option],[value1]
```

Code 15: AT+set

Available options:

- **transparent_trigger**: value1 is a bitmask of the following options
  - **timer**: data is transmitted in transparent mode if a pause of at least transparent_timeout milliseconds is detected in the data stream
  - **1etx**: data is transmitted in transparent mode if the ETX (first byte of transparent_etx) character has been received by the Calypso
  - **2etx**: data is transmitted in transparent mode if the full ETX (both bytes of transparent_etx) has been received by the Calypso
  - **transmit_etx**: if this option is set and if 1etx or 2etx is used as trigger in transparent mode, the ETX character is not removed from the data stream and is forwarded to the socket
- **transparent_timeout**: value1 is timeout (6-1000) [ms]
- **transparent_etx**: value1 is 2 byte ETX in hexadecimal notation

```
AT+set=UART,transparent_trigger,2etx|timer
AT+set=UART,transparent_timeout,20
AT+set=UART,transparent_etx,0x0D0A
```

Code 16: Example Trigger on ETX 0x0D0A and after 20 ms

```
AT+set=UART,transparent_trigger,1etx|transmit_etx
AT+set=UART,transparent_etx,0x0D00
```

Code 17: Example Trigger on ETX 0x0D and forward ETX to socket

5.2 Configure the UART trigger settings via web interface

In order to configure the UART trigger settings via the web interface, follow the steps described in chapter 4.2 to access the usersettings page. Select the category "UART" and the setting "transparent_trigger", "transparent_timeout" or "transparent_etx" from the corresponding drop-downs to GET/SET the values.

This webpage uses RESTful APIs in order to perform GET and POST requests on resources on the Calypso. A detailed description of these API calls can be found in chapter "The HTTP server interface" of the Calypso user manual [1].

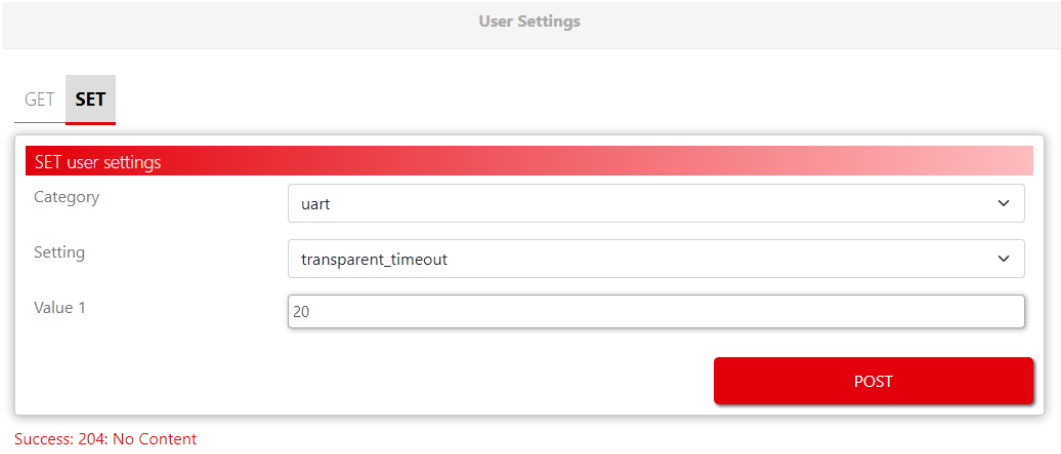

Figure 7: Configure UART trigger setting

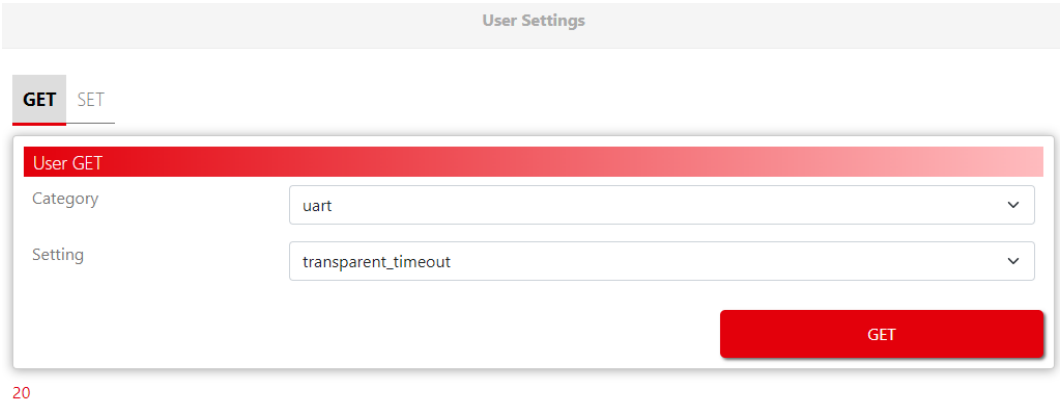

Figure 8: Read UART trigger setting

6 Throughput

In any case, to gain highest throughput from host to socket data transmission, it is important that the Calypso is able to forward the payload data from UART to the socket in the most efficient way. To do so, the Calypso internally moves data chunks of variable size from UART to the socket. The more data is received on the UART in a dedicated time frame, the smaller these chunks are. Thus, the best chunk size and therefore the highest throughput can be reached by either using a baud rate of about 921600 baud, or by adding a pause (about 10 ms, which is on average the latency for Calypso to forward a UART frame to WiFi) after every 1460 bytes in the data stream when using higher baud rates.

Baud rate [Baud]	Throughput [kByte/s]	Test condition
921600	49	Flow control enabled, "\r\n" as trigger, pause of 10 ms after every 1460 Bytes
921600	69.5	Flow control enabled, "\r\n" as trigger, 1460 Bytes without pause
2000000	78	Flow control enabled, "\r\n" as trigger, pause of 10 ms after every 1460 Bytes
3000000	82	Flow control enabled, "\r\n" as trigger, pause of 10 ms after every 1460 Bytes

Table 2: Transparent mode - throughput examples

7 References

[1] Würth Elektronik. Calypso user manual. https://www.we-online.de/katalog/de/manual/2610011025000.

8 Important notes

The Application Note and its containing information ("Information") is based on Würth Elektronik eiSos GmbH & Co. KG and its subsidiaries and affiliates ("WE eiSos") knowledge and experience of typical requirements concerning these areas. It serves as general guidance and shall not be construed as a commitment for the suitability for customer applications by WE eiSos. While WE eiSos has used reasonable efforts to ensure the accuracy of the Information, WE eiSos does not guarantee that the Information is error-free, nor makes any other representation, warranty or guarantee that the Information is completely accurate or up-to-date. The Information is subject to change without notice. To the extent permitted by law, the Information shall not be reproduced or copied without WE eiSos' prior written permission. In any case, the Information, in full or in parts, may not be altered, falsified or distorted nor be used for any unauthorized purpose.

WE eiSos is not liable for application assistance of any kind. Customer may use WE eiSos' assistance and product recommendations for customer's applications and design. No oral or written Information given by WE eiSos or its distributors, agents or employees will operate to create any warranty or guarantee or vary any official documentation of the product e.g. data sheets and user manuals towards customer and customer shall not rely on any provided Information. THE INFORMATION IS PROVIDED "AS IS". CUSTOMER ACKNOWLEDGES THAT WE EISOS MAKES NO REPRESENTATIONS AND WARRANTIES OF ANY KIND RELATED TO, BUT NOT LIMITED TO THE NON-INFRINGEMENT OF THIRD PARTIES' INTELLECTUAL PROPERTY RIGHTS OR THE MERCHANTABILITY OR FITNESS FOR A PURPOSE OR USAGE. WE EISOS DOES NOT WARRANT OR REPRESENT THAT ANY LICENSE, EITHER EXPRESS OR IMPLIED, IS GRANTED UNDER ANY PATENT RIGHT, COPYRIGHT, MASK WORK RIGHT, OR OTHER INTELLECTUAL PROPERTY RIGHT RELATING TO ANY COMBINATION, MACHINE, OR PROCESS IN WHICH WE EISOS INFORMATION IS USED. INFORMATION PUBLISHED BY WE EISOS REGARDING THIRD-PARTY PRODUCTS OR SERVICES DOES NOT CONSTITUTE A LICENSE FROM WE eiSos TO USE SUCH PRODUCTS OR SERVICES OR A WARRANTY OR ENDORSEMENT THEREOF.

The responsibility for the applicability and use of WE eiSos' components in a particular customer design is always solely within the authority of the customer. Due to this fact it is up to the customer to evaluate and investigate, where appropriate, and decide whether the device with the specific characteristics described in the specification is valid and suitable for the respective customer application or not. The technical specifications are stated in the current data sheet and user manual of the component. Therefore the customers shall use the data sheets and user manuals and are cautioned to verify that they are current. The data sheets and user manuals can be downloaded at *www.we-online.com*. Customers shall strictly observe any product-specific notes, cautions and warnings. WE eiSos reserves the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time without notice.

WE eiSos will in no case be liable for customer's use, or the results of the use, of the components or any accompanying written materials. IT IS CUSTOMER'S RESPONSIBILITY TO VERIFY THE RESULTS OF THE USE OF THIS INFORMATION IN IT'S OWN PARTICULAR ENGINEERING AND PRODUCT ENVIRONMENT AND CUSTOMER ASSUMES THE ENTIRE RISK OF DOING SO OR FAILING TO DO SO. IN NO CASE WILL WE EISOS BE LIABLE FOR

CUSTOMER'S USE, OR THE RESULTS OF IT'S USE OF THE COMPONENTS OR ANY ACCOMPANYING WRITTEN MATERIAL IF CUSTOMER TRANSLATES, ALTERS, ARRANGES, TRANSFORMS, OR OTHERWISE MODIFIES THE INFORMATION IN ANY WAY, SHAPE OR FORM.

If customer determines that the components are valid and suitable for a particular design and wants to order the corresponding components, customer acknowledges to minimize the risk of loss and harm to individuals and bears the risk for failure leading to personal injury or death due to customers usage of the components. The components have been designed and developed for usage in general electronic equipment only. The components are not authorized for use in equipment where a higher safety standard and reliability standard is especially required or where a failure of the components is reasonably expected to cause severe personal injury or death, unless WE eiSos and customer have executed an agreement specifically governing such use. Moreover WE eiSos components are neither designed nor intended for use in areas such as military, aerospace, aviation, nuclear control, submarine, transportation, transportation signal, disaster prevention, medical, public information network etc. WE eiSos must be informed about the intent of such usage before the design-in stage. In addition, sufficient reliability evaluation checks for safety must be performed on every component which is used in electrical circuits that require high safety and reliability functions or performance. COSTUMER SHALL INDEMNIFY WE EISOS AGAINST ANY DAMAGES ARISING OUT OF THE USE OF THE COMPONENTS IN SUCH SAFETY-CRITICAL APPLICATIONS.

List of Figures

List of Tables

Contact
Würth Elektronik eiSos GmbH & Co. KG
Division Wireless Connectivity & Sensors

Max-Eyth-Straße 1
74638 Waldenburg
Germany

Tel.: +49 651 99355-0
Fax.: +49 651 99355-69
www.we-online.com/wireless-connectivity