EXS82 connectivity demonstration (MQTT)

MicroEJ

Mar 09, 2021

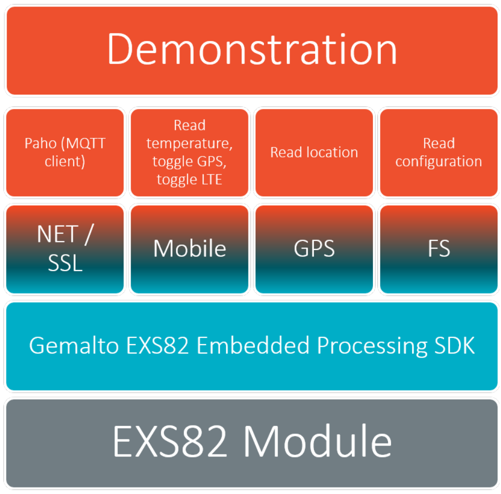
CONTENTS

CHAPTER

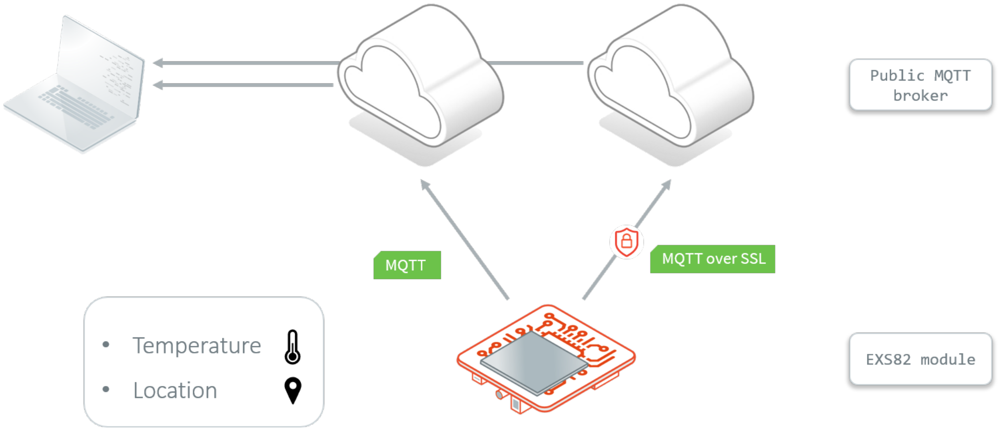
ONE

OVERVIEW

This package demonstrates the MicroEJ support of the Gemalto EXS82 Embedded Processing SDK.



It contains a MicroEJ (Java) application that is derived from the MQTT THALES showcase : https://developer.gemalto.com/showcase/concept-board-connection-azure-iot-cloud-mqtt-protocol.



CHAPTER

TWO

REQUIREMENTS

To follow the steps described in this documentation, you will need:

- A Gemalto Cinterion LGADevKit with:
    - a Gemalto EXS82 module
    - a wideband antenna
    - a SIM card
    - cables
- The Gemalto EXS82 Embedded Processing SDK Version 00.024 with:
    - the THALES firmware for EXS82 version 00.024
    - the tools to interact with the module
- A Python 2.7 installation with the serial module
- A web browser

CHAPTER
THREE

USAGE

This demo is divided in two parts:

- *MQTT demo on the virtual device*: run the MicroEJ application on the virtual device and get messages from the Internet.
- *Build the application binaries*: rebuild the MicroEJ application binaries from sources.

The first phase only requires a PC with a Java Runtime Environment installed. The second requires the MICROEJ SDK and aims to rebuild the binaries used by the two previous steps from sources.

MQTT demo setup helps to setup a nice looking MQTT web client to see messages from the (virtual) device. *Configuration* explains how to configure the application.

3.1 MQTT demo setup

To see messages sent by the (virtual) device, one will need to setup an MQTT client.

The following is a nice looking MQTT web client:

1. Open your web browser and access http://www.hivemq.com/demos/websocket-client/
2. In the "Connection" section, set the configuration (See *Configuration*). The default configuration for this demo is:
   - host : `broker.hivemq.com`
   - port : `8000`
   - leave other fields unchanged
3. Click on "Connect" to open the connection.
4. Skip the "Publish" section.
5. In the "Subscriptions" section, subscribe to the MQTT topic used by the demo (`MQTTDemo` by default):
   - Click on "Add New Topic Subscription"
   - Set topic to `MQTTDemo`
   - Click on "Subscribe"
6. The messages sent by the demo will be shown in the messages box.

3.2 MQTT demo on the virtual device

Requirements: a Java Runtime Environment

To run the demo on the virtual device, follow these steps:

1. Open the `virtual-device` folder
2. The configuration file is available in the `filesystem/` folder
3. Execute the launcher `launcher-<os>.[sh/bat]` :

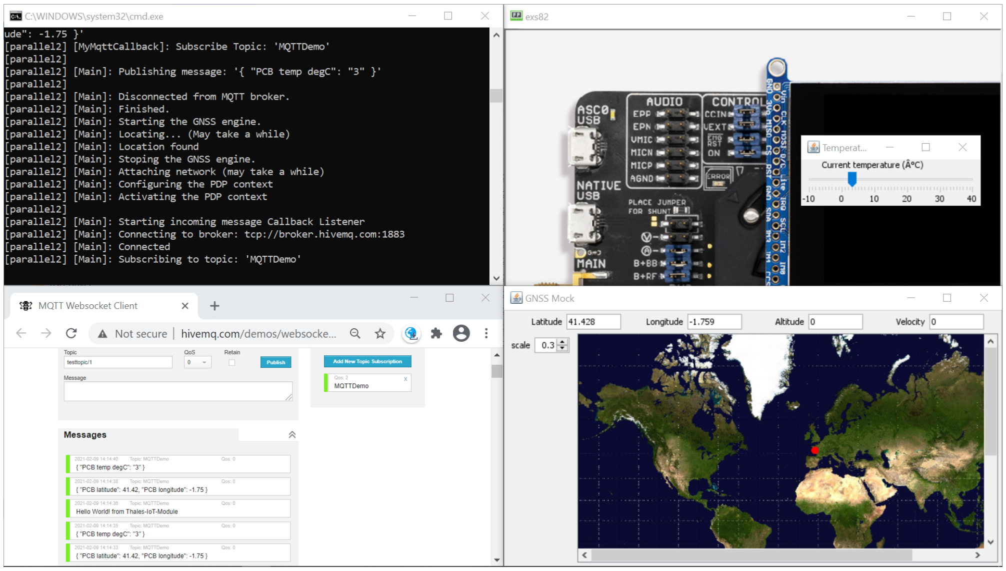
```
java version "1.8.0_221"
Java(TM) SE Runtime Environment (build 1.8.0_221-b11)
Java HotSpot(TM) 64-Bit Server VM (build 25.221-b11, mixed mode)
Buildfile: C:\Users\rlouedec\Downloads\demo-mqtt-0.1.0\demo-binaries\virtual-device\s3Standalone.
↪microejLaunch

standalone:init:
   [delete] Deleting directory C:\Users\rlouedec\Downloads\demo-mqtt-0.1.0\demo-binaries\virtual-
↪device\output
   [mkdir] Created dir: C:\Users\rlouedec\Downloads\demo-mqtt-0.1.0\demo-binaries\virtual-
↪device\output
   [mkdir] Created dir: C:\Users\rlouedec\Downloads\demo-mqtt-0.1.0\demo-binaries\virtual-
↪device\tmp\microejlaunch1144909490

standalone:properties:

standalone:run:

activity/microej:
   [echo] =============== [ Initialization Stage ] ===============

[...]

activity/s3:
[echo] =============== [ Launching on Simulator ] ===============

s3/exec/impl:

sleep.onWinXP:

mainMock:
[parallel2]
[parallel2] [Main]: PAHO Version = 'Version patched by SJM' if left unfactored.
[parallel2]
[parallel2] [Main]:  -------- Settings from demo.properties.list --------
[parallel2] [Main]:  broker.url:         tcp://broker.hivemq.com:1883
[parallel2] [Main]:  broker.certificate: null
[parallel2] [Main]:  broker.username:    null
[parallel2] [Main]:  broker.password:    null
[parallel2] [Main]:  sim.pin:            0000
[parallel2] [Main]:  sim.apn:            apn
[parallel2] [Main]:  sim.username:       username
[parallel2] [Main]:  sim.password:       password
[parallel2] [Main]:  mqtt.topic:         MQTTDemo
[parallel2] [Main]:  skip.gnss:          false
[parallel2] [Main]:  ----------------------- end ------------------------
```

(continues on next page)

(continued from previous page)

```
[parallel2] [Main]: Unlocking SIM
[parallel2] [Main]: Starting the GNSS engine.
[parallel2] [Main]: Locating... (May take a while)
[parallel2] [Main]: Location found
[parallel2] [Main]: Stoping the GNSS engine.
[parallel2] [Main]: Attaching network (may take a while)
[parallel2] [Main]: Configuring the PDP context
[parallel2] [Main]: Activating the PDP context
[parallel2]
[parallel2] [Main]: Starting incoming message Callback Listener
[parallel2] [Main]: Connecting to broker: tcp://broker.hivemq.com:1883
[parallel2] [Main]: Connected
[parallel2] [Main]: Subscribing to topic: 'MQTTDemo'
[parallel2] [Main]: Publish to topic: 'MQTTDemo'
[parallel2] [Main]: Publishing message: 'Hello World! from Thales-IoT-Module'
```

4. If the connection fails and you are using a VPN, disable it and retry.
5. Note that the screen stays black because this demo has no UI.
6. The demo will start pushing messages to the MQTT broker.
7. Play with the GNSS mock and the temperature monitor mock and observe the messages in your web browser.


C:\WINDOWS\system32\cmd.exe
exs82
AUDIO
CONTROL
ASC0 USB
NATIVE USB
MAIN
Temperat...
Current temperature (Â°C)
-10
0
10
20
30
40
MQTT Websocket Client
Not secure | hivemq.com/demos/websocke...
Publish
Add New Topic Subscription
Messages
GNSS Mock

```
ude": -1.75 }'
[parallel2] [MyMqttCallback]: Subscribe Topic: 'MQTTDemo'
[parallel2]
[parallel2] [Main]: Publishing message: '{ "PCB temp degC": "3" }'
[parallel2]
[parallel2] [Main]: Disconnected from MQTT broker.
[parallel2] [Main]: Finished.
[parallel2] [Main]: Starting the GNSS engine.
[parallel2] [Main]: Locating... (May take a while)
[parallel2] [Main]: Location found
[parallel2] [Main]: Stoping the GNSS engine.
[parallel2] [Main]: Attaching network (may take a while)
[parallel2] [Main]: Configuring the PDP context
[parallel2] [Main]: Activating the PDP context
[parallel2]
[parallel2] [Main]: Starting incoming message Callback Listener
[parallel2] [Main]: Connecting to broker: tcp://broker.hivemq.com:1883
[parallel2] [Main]: Connected
[parallel2] [Main]: Subscribing to topic: 'MQTTDemo'
```

Topic: testtopic/1
QoS: 0
Retain: ☐

Qos: 2
MQTTDemo

2021-02-09 14:14:40 Topic: MQTTDemo Qos: 0
{ "PCB temp degC": "3" }

2021-02-09 14:14:38 Topic: MQTTDemo Qos: 0
{ "PCB latitude": 41.42, "PCB longitude": -1.75 }

2021-02-09 14:14:38 Topic: MQTTDemo Qos: 0
Hello World! from Thales-IoT-Module

2021-02-09 14:14:35 Topic: MQTTDemo Qos: 0
{ "PCB temp degC": "3" }

2021-02-09 14:14:33 Topic: MQTTDemo Qos: 0
{ "PCB latitude": 41.42, "PCB longitude": -1.75 }

Latitude: 41.428
Longitude: -1.759
Altitude: 0
Velocity: 0
scale: 0.3

3.3 Configuration

It is possible to configure the demo application using the `demo.properties.list` at the root of the filesystem.

The following properties are configurable:

- `broker.url` : the URL of the MQTT broker (`[tcp/ssl]://<domain>:<port>`)
- `broker.certificate` : the path of the root CA certificate of the MQTT broker (`/path/to/root-ca.cer`), only required for secured MQTT broker connections.
- `broker.username` : username used to access the broker (optional).

- `broker.password` : password used to access the broker (optional).
- `sim.pin` : PIN code used to unlock the SIM.
- `sim.apn` : APN of the network used for Internet communications.
- `sim.username` : username used to access the network.
- `sim.password` : password used to access the network.
- `mqtt.topic` : topic on which messages are published (defaults to `MQTTDemo`).
- `skip.gnss` : enable/disable the location detection (defaults to `false`).

The application must be restarted for the change to take effect.

3.4 Build the application binaries

3.4.1 Installation

1. Install the MicroEJ SDK distribution from https://repository.microej.com/packages/SDK/20.12/.
2. Launch the IDE and create a workspace on your filesystem.
3. Select the default Microej repository.
4. Import `offline-repository.zip` with "File > Import… > General > Projects from Folder or Archive > Archive…".
5. Configure the Microej Module Manager to use the repository with "Window > Preferences > MicroEJ > Module Manager > Workspace…" and select the `ivysettings.xml` at the root of the project imported.

3.4.2 Build the platform

1. Import all platform projects from `platform-sources.zip` with "File > Import… > General > Existing Projects into Workspace > Select archive file > Browse…".
2. Right click on the `exs82-threadx-configuration` project and click on `Build Module` .
3. Wait for the `exs82-threadx-platform` project to be populated.

3.4.3 Build the demonstration application

1. Import all platform projects from `demonstration-sources.zip` with "File > Import… > General > Existing Projects into Workspace > Select archive file > Browse…".

3.4.4 Run the demonstration on the virtual device

2. Run the `[Sim] demo-mqtt` launcher in "Run > Run configurations… > MicroEJ Application".
3. Refer to `MQTT demo on the virtual device` to play with the demo.

CHAPTER

FOUR

KNOWN ISSUES/LIMITATIONS

- The application doesn't currently support concurrent GNSS and network operations. Therefore, the network interface is shut down before locating the device.
- In case of failure, one may have to reboot the device as some resources may not be properly cleaned when stopping the application.