AzureWave Technologies, Inc.

# AW-HM581

## 802.11ah Module EVK

## User Guide

**Rev. 0.1**

**(For Standard)**

# Revision History

Document NO:

| Version | Revision Date | DCN NO. | Description | Initials | Approved |
|---|---|---|---|---|---|
| 0.1 | 2022/09/28 | | Initial Version | Daniel Lee | NC Chen |

# Table of Contents

# 1 Overview

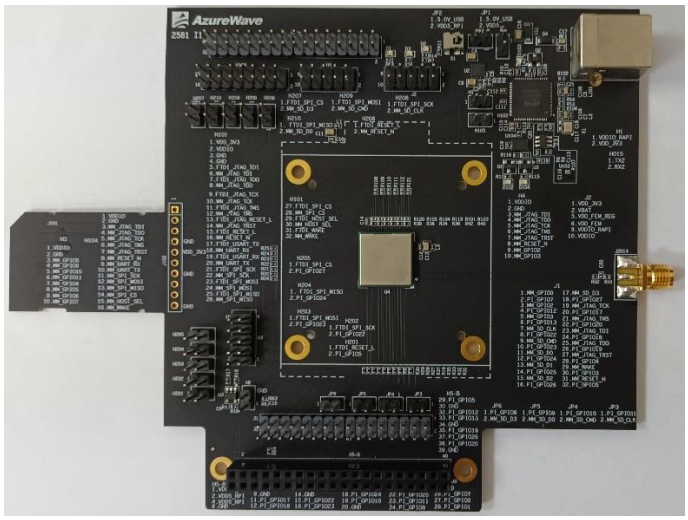
## 1.1 Device supported

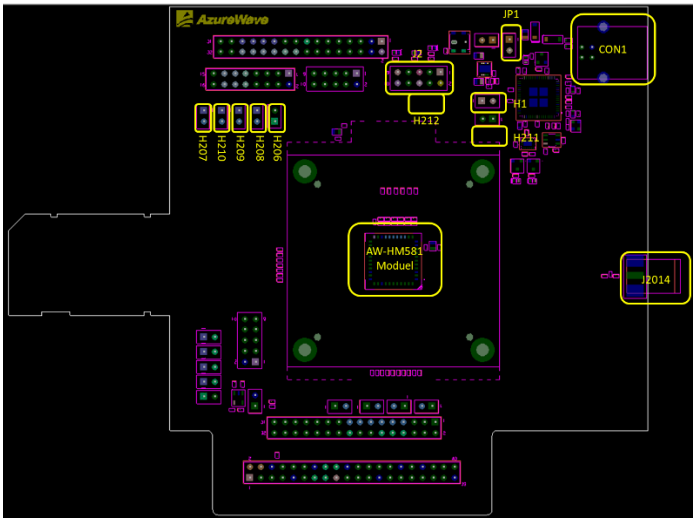
This document supports the AW-HM581 (13 x 13 mm LGA Module). The AW-HM581 EVB can be operated in MFG mode.

# 2. Basic Setup and Requirements for MFG Mode

This section provides the detailed information about the setting for AW-HM581 EVB. shows the overview of the AW-HM581 EVB physical photo and PCB placement (TOP). The description of jumpers' functions and settings on EVB is as follows:

**Azurewave AW-HM581 EVB physical photo and PCB placement**





## 2.1 Power Supply

a. The 5.0V power supply can be provided by USB connector (CON101) and short JP1
b. The 3.3V power supply for AW-HM581 VDD_FEM is converted from the 5.0V power supply through the LDO on the EVB by short pin5 and pin6 of J2.
c. The 3.3V power supply for AW-HM581 VBAT is converted from the 5V power supply through the LDO on the EVB by short pin1 and pin2 of J2.
d. The 3.3V power supply for AW-HM581 VDDIO is converted from the 5.0V power supply through the LDO on the EVB by short H1 and pin9 and pin10 of J2.

## 2.2 USB to SPI (H206 ~ H210)

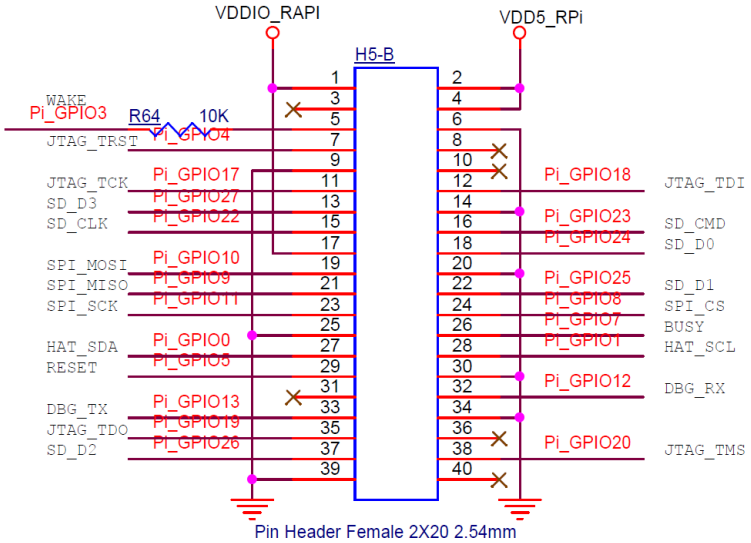
By shorting H206 ~ H210, the SPI signal of the AW-HM581 can be connected to the USB to SPI bridge IC.

## 2.3 Module Reset (S1)

The reset function can be controlled by S1.

## 2.4 GPIOs (H5-B)

The GPIO pins of AW-HM581 are connected to H5-B, the definition of each pin is as follows



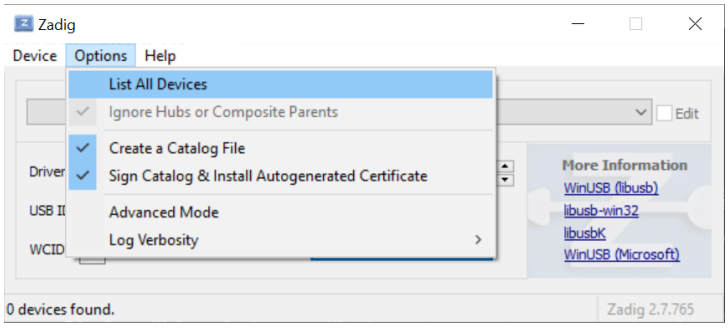
# 3. MFG Mode Tools and Code

This section gives instructions on how to use the provided tools and code to interface with AW-HM581 for the use of module testing.

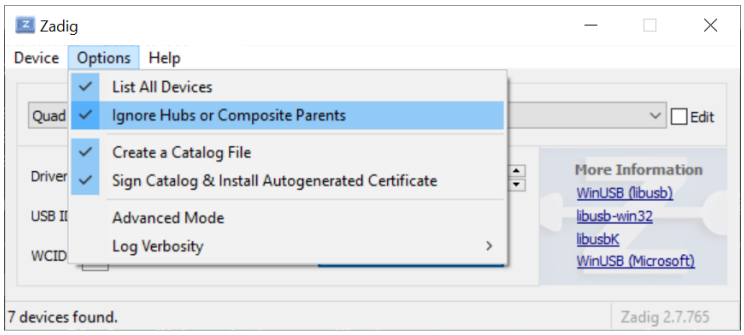
## 3.1 Software Setup (Windows)

Update FTDI Driver

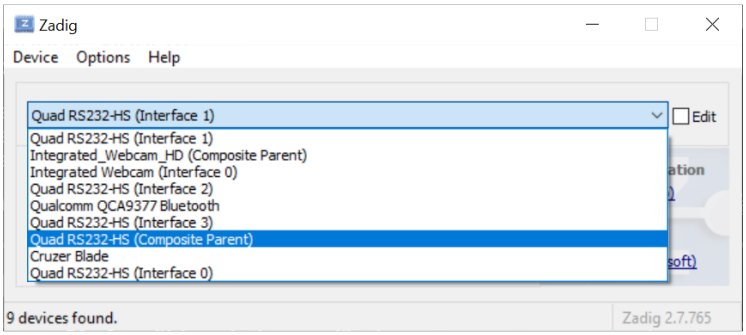
- Download and install the program, Zadig - https://zadig.akeo.ie/
- Launch Zadig
- Plug in the MM_DEBUG board to the computer
- Click on Options → List All Devices



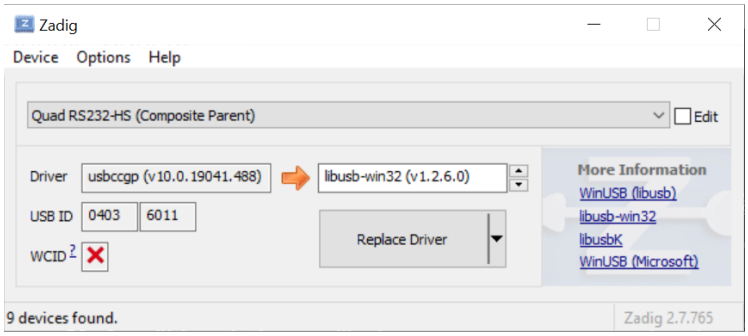

- Click on Options → Ignore Hubs or Composite Parents (this should now be deselected)




- Click on the drop down menu to find “Quad RS232-HS (Composite Parent)”



- Click the down arrow on the driver drop down menu, to select “libusb-win32 (v1.2.6.0)”




- Click “Replace Driver”
- You will then be prompted with a popup asking if you are sure. Click “Yes”.
- Eject and reinsert your AW-HM581 EVB

For more information, refer to the original pyftdi documentation -
https://eblot.github.io/pyftdi/installation.html#windows

## 3.2 Install Required Python Packages

- Open a terminal in the manufacturing_test repo
- Run the following command:

./install.bat

## 3.3 Instrument Setting - Litepoint

- **Changing Default IP Address**

  By default, the scripts will assume the litepoint has an IP address of 10.53.4.138. This can be configured in the litepoint_test/config.toml file.

- **Changing Default RF Port**

  By default, the scripts will assume the litepoint is using port RF1A as the connection to the litepoint. This can be configured in the litepoint_test/config.toml file.

- **Changing Default Port Attenuation**

  By default, the scripts will assume there is a 15dB attenuator in between the DUT and the litepoint, which

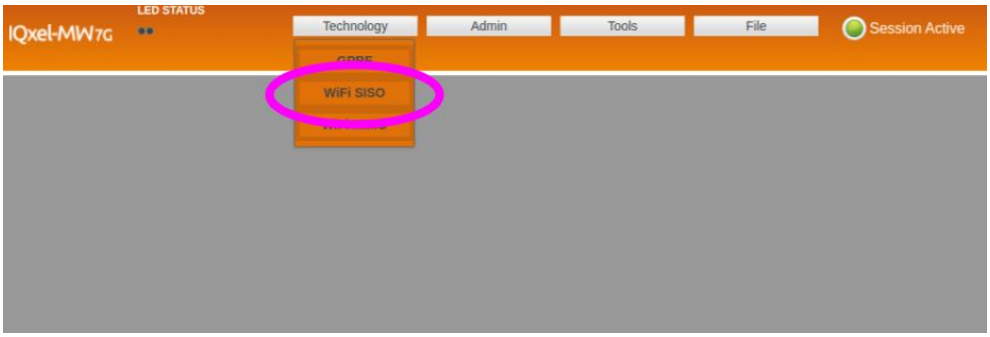
makes a total of 16.5dB of attenuation with the cable loss. This can be configured in the litepoint_test /config.toml file.

- **Adding Reference Waveform**

  For the Rx testing of the DUT, the litepoint will need to transmit packets that conform to the 802.11ah standard. To do this, a reference waveform will need to be loaded into litepoint.

  a. Connect to the IP address of the litepoint

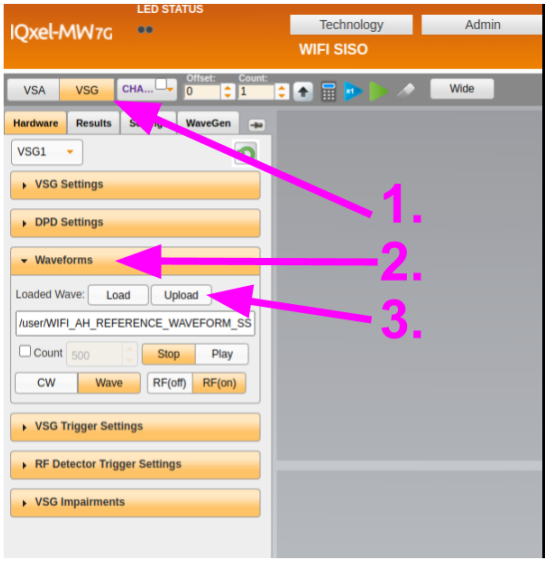
  b. Click Technology —> WiFi SISO:




  c. Click on "VSG"

  d. Open the "Waveforms" drop down menu.

  e. Click "Upload", and select the WIFI_AH_REFERENCE_WAVEFORM_SS1_BCC_Fs40M.iqvsg file found in tools/litepoint_lib from the manufacturing_test repo.



## 3.4 Example Scripts

From here, there are example scripts that are ready to be run.

### ftdi_load_firmware.py

#### Description

- Example usage of how to load the firmware binaries into the AW-HM581.

#### Expected Output

In terminal:

# python ftdi_load_firmware.py

Firmware load successful!

### ftdi_get_serial.py

#### Description

- Simple program to get the serial number from the DUT.
- Loads the firmware binaries into the AW-HM581.
- Reads the serial number programmed into the AW-HM581.
- Prints this serial number.

#### Expected Output

In terminal:

# python ftdi_get_serial.py

SERIAL = 0x60cb6c39

### ftdi_get_temp_sensor_val.py

#### Description

- Simple program to get the temperature reported from the DUT.
- Loads the firmware binaries into the AW-HM581.
- Sends a command to the AW-HM581 chip to retrieve the temperature reading.
- Prints the temperature in degrees.

#### Expected Output

In terminal:

# python ftdi_get_temp_sensor_val.py

Temperature = 25 degrees

## ftdi_start_rpg_tx.py

### Description

- Simple program that configures the chip to start transmitting randomly generated packets.
- Loads the firmware binaries into the AW-HM581.
- Sets the channel, mcs, bandwidth etc.
- Starts transmitting.
- Continues transmitting until the user presses a key.

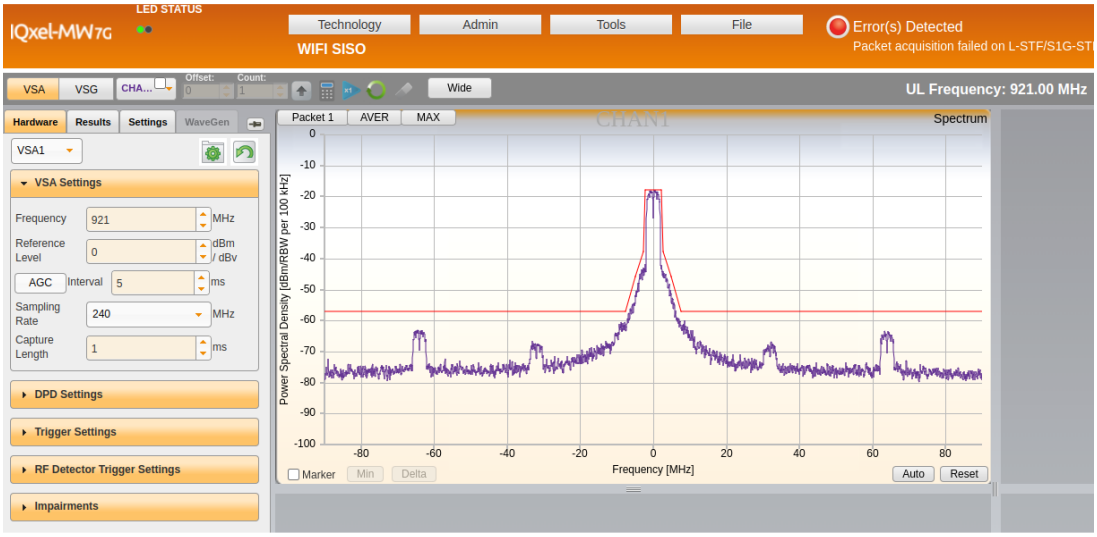
### Expected Output

In terminal:

# python ftdi_start_rpg_tx.py

Starting RPG TX. Press any key to stop.

On Litepoint:




## ftdi_start_rpg_rx.py

### Description

- Simple program that configures the chip to start receiving randomly generated packets.
- Loads the firmware binaries into the AW-HM581.
- Sets the channel, mcs, bandwidth etc.
- Starts receiving.
- Pressing the “Enter” button prints out the current RPG stats.
- Typing “clear” resets the RPG stats.

- Typing "exit" stops RPG, and exits the program.
- Note: The user will have to start transmitting packets from the litepoint that correspond to the parameters the DUT is configured to in order to see any received packets show up in the stats.

### Expected Output

```
# python ftdi_start_rpg_rx.py
Starting RPG RX. Options:
                'Enter Key' to get stats
                'clear' to clear the stats
                'exit' to stop RPG and end script
# Press enter key
(0, 0, 0)
# Press enter key
(28, 25, 0)
# Type "clear"
clear
(0, 0, 0)
# Press enter key
(1, 1, 0)
# Press enter key
(26, 26, 0)
# Type "exit"
exit
```

## ftdi_tx_litepoint_test.py

### Description

- Program that configures both the Litepoint, and the MM610x DUT, and tests the PHY TX performance of the DUT across a sweep of Bandwidths, MCS rates, power levels etc.
- The user will need to specify the Litepoint IP address, port being used, and attenuation on the port in the config.toml file found in the litepoint_test directory within the manufaturing_test repo.
- The test script does the following:
  - Gets a list of test conditions found in the test_items.toml file (found in the same directory as the

onfig.toml file).

- Sets the litepoint up as a Vector Signal Analyser (VSA) with the corresponding parameters for a particular test condition.
- Starts transmitting randomly generated packets from the DUT with the corresponding parameters for a particular test condition.
- Evaluates whether that test was a pass by comparing to limits found in the litepoint_test/test_limits directory.
- Prints "Pass" or "Fail". If the test failed, it will also print the parameters which failed.
- Repeats this process for every test condition.
- Will store all the data into tx_data.csv at the end of the program.

### Expected Output

In terminal:

# python ftdi_tx_litepoint_test.py

Connecting to Litepoint...

Connecting to DUT...

Running Tx tests...

FAIL

Name = power_dbm:min | Limit = 20.5 | Value = 17.9882225

PASS

FAIL

Name = evm_db:max | Limit = -28 | Value = -27.036518

Name = power_dbm:min | Limit = 15.0 | Value = 12.9013672

PASS

PASS

PASS

## ftdi_rx_litepoint_test.py

### Description

- Program that configures both the Litepoint, and the AW-HM581, and tests the PHY RX performance of the DUT across a sweep of Bandwidths, MCS rates, power levels etc.
- The user will need to specify the Litepoint IP address, port being used, and attenuation on the port in the config.toml file found in the litepoint_test directory within the manufaturing_test repo.

- The test script does the following:
  - Gets a list of test conditions found in the test_items.toml file (found in the same directory as the config.toml file).
  - Sets the litepoint up as a Vector Signal Generator (VSG) with the corresponding parameters for a particular test condition.
  - Starts transmitting randomly generated packets from the litepoint with the corresponding parameters for a particular test condition.
  - Evaluates whether that test was a pass by comparing to limits found in the litepoint_test/test_limits directory.
  - Prints "Pass" or "Fail". If the test failed, it will also print the parameters which failed.
  - Repeats this process for every test condition.
  - Will store all the data into rx_data.csv at the end of the program.

### Expected Output

In terminal:

# python ftdi_rx_litepoint_test.py

Connecting to Litepoint...

Connecting to DUT...

Running Rx tests...

FAIL

Name = packet_error_rate:max | Limit = 10 | Value = 100.0

FAIL

Name = packet_error_rate:max | Limit = 10 | Value = 100.0

FAIL

Name = packet_error_rate:max | Limit = 10 | Value = 100.0

PASS

PASS

PASS

PASS

PASS

PASS

## ftdi_freq_vs_evm_sweep.py

## Description

- Program that configures both the Litepoint, and the AW-HM581 DUT, and tests the PHY TX performance of the DUT across a sweep of Bandwidths, MCS rates, power levels etc.
- The user will need to specify the Litepoint IP address, port being used, and attenuation on the port in the config.toml file found in the litepoint_test directory within the manufaturing_test repo.
- The test script does the following:
  - Gets a list of test conditions found in the test_items.toml file (found in the same directory as the config.toml file).
  - Sets the litepoint up as a Vector Signal Analyser (VSA) with the corresponding parameters for a particular test condition.
  - Starts transmitting randomly generated packets from the DUT with the corresponding parameters for a particular test condition.
  - Records the test conditions and test results.
  - Repeats this process for every test condition.
  - Plots the Frequency vs EVM for MCS 0 and MCS 7, over all bandwidths.
  - Stores all the data into data.csv at the end of the program.

## Expected Output

In terminal:

# python ftdi_freq_vs_evm_sweep.py

Connecting to Litepoint...

Connecting to DUT...

Running Tx tests...

Testing BW=1 MCS=0 FREQ=860MHz (1/112)

Testing BW=1 MCS=0 FREQ=870MHz (2/112)

Testing BW=1 MCS=0 FREQ=875MHz (3/112)

Testing BW=1 MCS=0 FREQ=880MHz (4/112)

Testing BW=1 MCS=0 FREQ=885MHz (5/112)

Testing BW=1 MCS=0 FREQ=890MHz (6/112)

Testing BW=1 MCS=0 FREQ=895MHz (7/112)

...

...

...

Testing BW=8 MCS=7 FREQ=910MHz (108/112)

Testing BW=8 MCS=7 FREQ=915MHz (109/112)

Testing BW=8 MCS=7 FREQ=920MHz (110/112)

Testing BW=8 MCS=7 FREQ=925MHz (111/112)

Testing BW=8 MCS=7 FREQ=930MHz (112/112)

## ftdi_get_mac_addr.py

### Description

- Simple program to get the MAC Address from the DUT.
- Loads the firmware binaries into the MM610x chip.
- Reads the MAC Address programmed into the AW-HM581.
- Prints this MAC Address.

### Expected Output

In terminal:

# python ftdi_get_mac_addr.py

MAC Address = 0C:BF:74:E1:0C:36