Developers Guide

Ayla OEM Installation Guide for Black Box

Version: 4.0
Date Released: April 4, 2017
Document Number: AY006DBB3-4

Ayla Networks

Copyright Statement

Trademarks Statement

Referenced Documents

Ayla Networks does not supply all documents that are referenced in this document with the equipment. Ayla Networks reserves the right to decide which documents are supplied with products and services.

Contact Information

Ayla Networks TECHNICAL SUPPORT and SALES

Contact Technical Support: https://support.aylanetworks.com
or via email at support@aylanetworks.com

Contact Sales: https://www.aylanetworks.com/company/contact-us

Ayla Networks REGIONAL OFFICES

HEADQUARTERS

Chicago
10 N. Martingale Road, Suite 400
Schaumburg, IL 601073

Silicon Valley
4250 Burton Drive, Suite 100
Santa Clara, CA 95054
Phone: +1 408 830 9844
Fax: +1 408 716 2621

Boston
275 Grove Street, Suite 2-400
Newton, MA 02466

Table of Contents

Table of Figures

1 Introduction

This document provides an overview of the OEM Package and what is included in the OEM Package, the steps for installing the OEM Package, and a Quick Start version of the OEM installation. Included is how to run the verification programs for Ayla Modules.

1.1 Audience

This document is written for Project Managers and engineers. The OEM Package content is helpful for Project Managers. The entire document is useful to engineers.

1.2 Related Documentation

- OEM Roles and Privileges (AY006URO)
- Ayla Module Firmware Update Manual (AY006MUU0)
- Ayla Module and MCU Interface Specification (AY006MSP0)

1.3 Customer Support

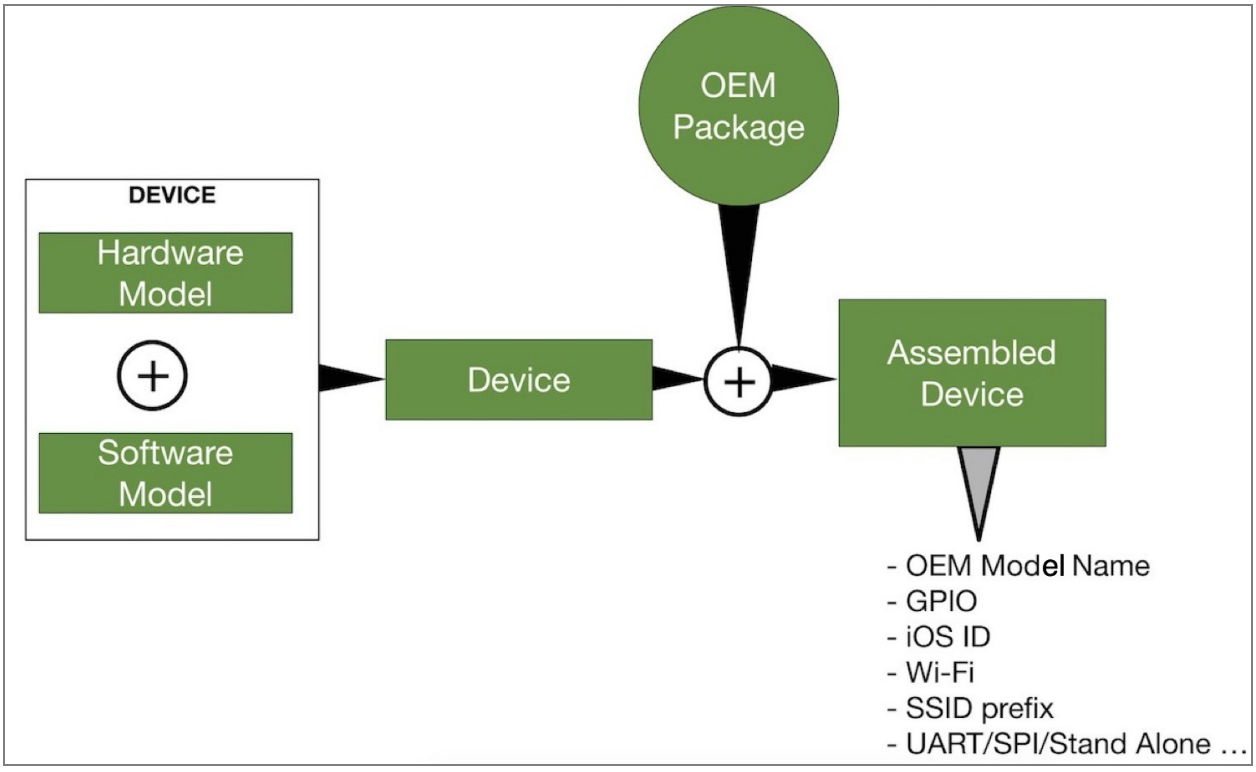
Support is available at http://support.aylanetworks.com. The support site provides access to the knowledge base. You can also create a support ticket for additional help.

2 OEM package Overview

The OEM Package is deployed to the Ayla Module to configure the Module for the chosen product implementation.

The OEM Package sets and verifies the OEM ID, Module ID and Device ID so a secure connection can be established with the Ayla Cloud Service. The OEM Package also provides updates to the Ayla modules. Figure 1 shows the major OEM Package pieces.

Figure 1 OEM Package Overview



The OEM Package must be verified as a final check of the configuration data. The `verify_config.txt` file validates configuration results.

2.1 Firmware Updates

Ayla uses the Agile software development method, which includes rapid iterative development and frequent releases. Each time the OEM Package connects with the module, the firmware is updated to the most recent version.

Ayla strongly recommends that customers update to latest minor firmware version of the major firmware release used. This takes advantage of Ayla's continuous, iterative improvement of product reliability, new features, and error reduction.

2.2 OEM Configuration

When OEM configuration information is applied to the module, a script transforms the default blank production module into a unique, customized customer product application module. Product application-specific information includes:

- GPIO usage:
  - signal input signals
  - Inputs
  - Outputs
  - IO being Pulled up
  - IO being Pulled down
  - Status signal output signals
- Control Stand-alone or Host MCU mode
- In Host MCU mode:
  - SPI
  - UART
  - Baud rate
  - Number of parity bits
  - Number of stop bits
  - Number of data bits
  - Even, odd or no parity

Additional custom configuration can be enabled with the OEM Package. For example: to provide a better end-user experience, a Wi-Fi profile SSID and password can be programmed directly into the module for a custom gateway.

2.3 OEM ID

The OEM ID connects the module to the customer organization. A manufacturer can access the appropriate group of templates located in the Ayla Cloud.

2.4 Model ID

The OEM Model ID provides additional template filtering. This allows more granularity of control and data collection at the product level.

2.5 OEM Key

The OEM Key validates the OEM ID and Model ID during communication with Ayla Cloud Services. The key is an additional layer of security for each communication attempt.

2.6 OEM Package Run Procedure

This is a high-level overview of the steps for running the OEM Package.

- Install the OEM package onto a Windows machine (virtual is fine).
- Updates require TCL as well as packages for TCL. (This installation is only required for the initial setup.)

Follow this procedure:

1. Connect a serial adapter between the Windows machine and the module.

   The method can vary depending on your device design.

   - Ayla recommends **.10-inch** spaced header (allows a FTDI Chip TTL-232R-3V3 serial cable to be attached).
   - RX and TX signals are required.
   - No flow control is enabled.

2. Apply the firmware update.

   The `update.bat` command runs the OEM module firmware update. After the update, the script performs the configuration.

3. Apply the configuration (Module must be in Set-up mode)

   The `update.bat` command starts the second stage of the OEM Package install. This applies configuration settings in the `configure.txt` file.

3 Installation

The setup assumes the following:

- A PC running Windows/XP with Service Pack 3.
- The PC runs in Windows 32-bit mode.
- A serial port connected to the Ayla module to be configured or verified.

  If the serial port is based on a Future Devices USB chip, refer to the document *Module Firmware Update Manual* (AY006MUU0).

3.1 User Setup

4. On the PC, create a Windows user with administrator rights.
5. Log in as the administrator.

3.2 TCL Installation

6. Download and run TCL for Windows 32-bit from the web site:

   http://www.activestate.com/activetcl/downloads

7. Choose **Active TCL 8.5.10** or **8.5.x for Windows**.

   IMPORTANT: TCL 8.6 is not compatible.

8. Set the default install directory (e.g., **C:\Tcl**).

3.3 Install TCL Packages

9. Open a Windows command window.
10. To add the required TCL packages, enter:

```
C:\> teacup install Expect
C:\> teacup install xml
```

3.4 Installation of OEM Package

The package is delivered in a file with a name such as *oem_pkg-<version>.zip*.

<version> is the OEM-package version number. This is not the module firmware version number. For this example, '1.12' is the <version>.

11. Copy the file to "C:\ayla" (or a similar directory).
12. Use Windows Explorer to locate and select the zip file.
13. Right-click and select **Extract all...**.
14. Click **Next** (accept the default destination directory).

    This puts everything in a **oem_pkg-1.12** directory.
15. In Command window, change to the **oem_pkg-1.12** directory:

    ```
    C:\> cd C:\ayla\
    ```

    If there a directory "oem_pkg" exists (from a previous installation), remove it.

    ```
    C:\ayla> rmdir oem_pkg
    ```
16. Rename the new oem_pkg:

    ```
    C:\ayla> ren oem_pkg-1.12 oem_pkg
    ```

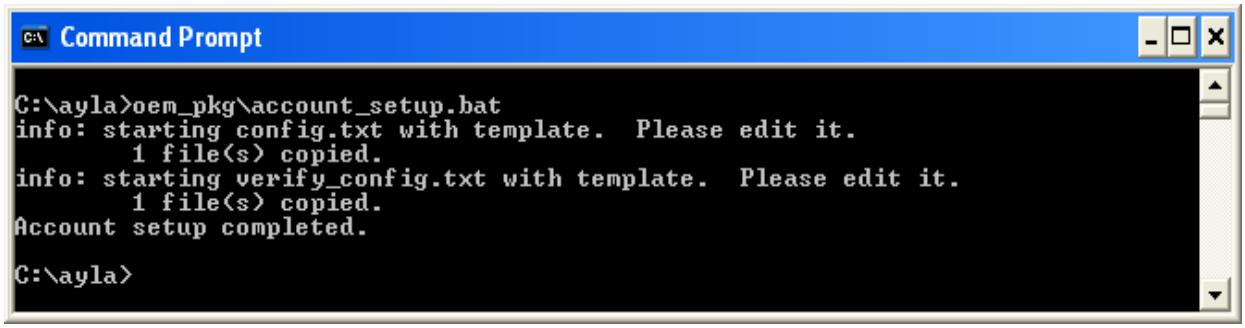
3.5 Account Setup

To setup required directories and tools (e.g., devcon), enter:

```
C:\ayla> oem_pkg\account_setup.bat
```

See Figure 2 for script output.

Figure 2 Account Setup



4 Module Configuration

The script `ayla_config.tcl` configures the module.

This configuration script uses the config.txt file in the directory that contains `oem_pkg`. The `account_setup` script initializes this file (if it doesn't exist) This script must be edited to fit your requirements.

Edit the `config.txt` file to include the module settings.

NOTE To preserve the ASCII format of the file, use a simple text editor (e.g., Notepad). Avoid WordPad and more complicated editors that might change the file to wide characters or to an internal format.

4.1 Configuration File

The file contents look something like this. Editable values are in **bold**.

```
# 1.7 example ayla_config configuration
#
# Configure your serial port if it is not automatically detected
#
#com_port com4
#
# Specify the expected module firmware version
#
mod_version 2.4.7
#
# Give name of ODM - Original Device Manufacturer for end user
#
odm your-company-name-here
#
# Module CLI commands to be executed during startup
#
commands
#
# Configure the AP profile as desired
# The default is to be Ayla-<mac-address>
#
wifi profile ap
```

```
wifi ssid-mac Ayla-
wifi security none
wifi profile enable
#
# Specify the IOS app string to be given during Wi-Fi setup
#
wifi setup_ios_app iMDA
wifi commit
#
# Replace the next three items with your OEM_ID, OEM_model, and OEM key
#
oem your-oem-id
oem model your-oem-model
oem key your-oem-key
commands end
```

Helpful Tips

- # (pound-sign) – these are commented lines and will be ignored when the script is run.
- Initial lines set the name and value for a configuration item.
- Between `commands` and `commands end`, the key-value pairs enter module settings. Do NOT include `save` or `setup_mode` commands - the script provides these.
- `# com_port` – (commented out) this can set the serial port (optional). The script might run faster if the serial port is specified, but not usually needed. A test of the configuration without this line, indicates which port the script uses. If, script does not select the correct port, change with the Windows `mode` command.
- `mod_version` - identifies firmware version expected to run on the module. The configuration fails if this is not an exact match to the module version.

See 5 Module Command Line Interface for a list of useful CLI commands.

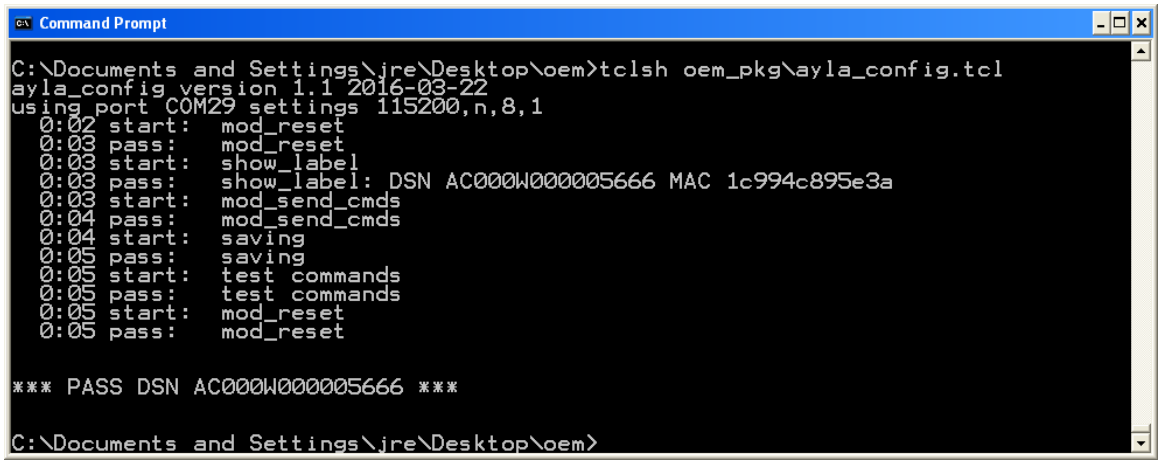
4.2 Run ayla_config script

To run the configuration, type:

```
C:\ayla> tclsh oem_pkg\ayla_config.tcl
```

The output will be similar to Figure 3.

Figure 3 Run ayla_config



4.3 Options for ayla_config

When ayla_config completes, it normally disables setup_mode on the module and saves the configuration. This step cannot easily be un-done.

To test the configuration without disabling setup_mode and without saving at the end, use `--dryrun`

```
C:\ayla> tclsh oem_pkg\ayla_config.tcl --dryrun
```

If more detailed output is needed, use `--log_user` (shows serial port interaction with the module).

```
C:\ayla> tclsh oem_pkg\ayla_config.tcl --log_user
```

5 Module Command Line Interface (CLI)

The Ayla module accepts commands over the serial port from ayla_config. These commands are usually supplied using the configuration file as described in 4.1 Configuration File.

These commands are generally for internal use only during setup of the module. Other usage is not documented or supported. Only the options useful for OEM setup of the module are documented here.

5.1 Wi-Fi Setup

Wi-Fi settings include settings for the device's AP mode and antenna selection (if applicable).

There are 11 profiles that contain network names (SSIDs), security types, and keys. The module uses Profiles 0 to 9 as networks to connect. Profile 10 is the 'ap' (Access Point) that contains the network name and security info the module uses for its own AP mode.

Display Wi-Fi settings

This command shows the current Wi-Fi state and settings

```
show wifi
```

Enable/disable Wi-Fi

The commands below enable or disable Wi-Fi. The change is effective with a `commit` command.

```
wifi enable 1
wifi enable 0
```

To enable or disable immediately, use:

```
wifi enable
wifi disable
```

Current Profile

These are related to the Wi-Fi profile. Wi-Fi commands have the concept of "current profile". Many commands can affect the current profile.

After startup, the current profile is Profile 0. The command `wifi profile ap` selects the AP profile. For example, `wifi profile 2` selects Profile 2.

```
wifi profile 2
wifi profile ap
```

Network name

This sets the SSID / Network name for the current profile.

```
wifi ssid my-network
```

If there is a word space in the name, add quotes around the name.
E.g., `wifi ssid "my network"`

AP Network Name with MAC address

This sets the AP profile network name to `oem-xxx` (where "xxx" is the Wi-Fi interface MAC address.

```
wifi profile ap
wifi ssid-mac oem-
```

This makes each module unique so that a serial number or MAC address is not required in the command. Wi-Fi must be enabled first and can only be used for AP profile.

Security Type

This sets the profile security type.

```
wifi security [none | WEP | WAP | WPA2_Personal]
```

For station profiles, this sets the lowest security used by the profile. For the AP profile, it is the only security level used. The AP profile security defaults to `none` (open).

Security Key

This sets the security key (Wi-Fi passphrase) for the profile.

```
wifi key "your-secret-key"
```

Enable/disable Profile

This enables or disables the profile.

```
wifi profile enable
wifi profile disable
```

Disabled profiles are remembered but not used unless specifically selected with a `join` command from the mobile Application or MCU. By default, all profiles (except AP profile) are disabled. IMPORTANT: AP profile cannot be disabled.

5.2 Wi-Fi Setup using Airkiss

As an alternative to Wi-Fi configuration with specified SSID and password in a profile, these commands enable setup from any mobile phone app that supports Airkiss protocol (e.g., WeChat).

Enable/disable Airkiss configuration

This command enables Airkiss configuration.

```
wifi setup_mode airkiss
```

After command is issued, subsequent module reboots listen for messages from an Airkiss-enabled sender application. It configures the received SSID and password. After 2-3 minutes, if no sender is detected, the module reverts to AP mode.

Airkiss configuration key

This sets the AES-128 key to encrypt SSID and password.

```
wifi setup_mode airkiss key genkey|<key>
```

This must be exactly 32 hexadecimal digits (e.g., 16-byte value). Alternatively, the value genkey can be used to randomly generate the key. The key value is exported in the OEM log file (generated by the script).

The key must be set for every Airkiss-configured device, even if the protocol is enabled by push-button or a host MCU.

5.3 GPIO and Status signal configuration

The GPIO facility configures module I/O pins. The feature has default Ready, Link, and Interrupt signals to be assigned to non-default pins. In some applications, the module can be used without a host MCU (GPIO mode).

GPIO mode allows a module to associate properties with module I/O pins. This can be useful in simple applications that don't require a separate MCU to perform the device functions. A module in GPIO mode does not support the SPI interface to an MCU.

The set of GPIO pins available varies between module part numbers.

GPIO Settings

The available GPIO pins list uses the `gpio` command without arguments.

```
gpio
gpio <pin>
gpio <property>
```

When a single argument matches a pin name or a property name assigned to a pin, settings for that single pin are shown.

Enable/disable GPIO mode

This enables or disables GPIO mode.

```
gpio enable
gpio disable
```

(Not necessary if only setting up Ready, Link, Interrupt or other status outputs.)

GPIO mode can only be changed in setup or mfg mode. GPIO settings may be displayed in user mode for diagnostic purposes. GPIO settings are not be completely applied until `gpio commit` (or configuration saved and module reset).

Output pins settings

This sets up a pin to be an output signal from the module.

```
gpio <pin> out [prop <name>][od][inv][persist][def <val>]
gpio pa1 out prop control1 inv persist def 1
```

It may be controlled by a module status variable or a property setting. The default output value is configured. If the keyword `persist` is used, the value is saved whenever it is changed, and restored when the module restarts.

Output signals can be push-pull or open-drain. The keyword `od` selects open-drain. If specified, “out” is implied.

IMPORTANT: open-drain outputs are not supported on some module types due to hardware limitations. For those modules, if `od` is specified, the command is not accepted.

By default, I/O pins are “active high,” 0=signal is low, 1=signal is high.

The `inv` keyword “inverts” the signal.

Input pins settings

This sets up a pin to be an input signal to the module.

```
gpio <pin> in [prop <name>] [inv] [toggle|set|rst <out-pin>] [pu|pd]
gpio pa2 in prop button1 inv toggle pa1 pu
```

Input pins can be floating, pulled up to a high level, or pulled down to ground by a weak internal resistor. For resisters, keyword `pu` = pull-up and `pd` = pull-down.

No all module types have available pull-ups and pull-downs. If used on these modules, an error message is output when the `gpio` configuration is committed.

By default, I/O pins are “active high”. 0=signal low, 1=signal high. The `inv` keyword “inverts” the signal.

Input pins are de-bounced by ignoring transitions that occur less than 15 milliseconds from the previous transition.

An input can be configured to affect an output pin with keywords: `toggle`, `set`, or `rst`.

If a property is assigned to an input pin, a Boolean datapoint of that property is sent to the service and/or mobile Application whenever the debounced input changes.

Disable pin

This disables the I/O pin, so that it is not used by the GPIO facility.

```
gpio <pin> disable
```

Ready, Link, or Interrupt Output setting

The module output signals are READY_N, LINK_N, and INTR_N. These indicate status to the MCU. READY_N and LINK_N are typically also connected to user-visible LEDs.

By default, these signals are assigned to pins. No configuration is needed. However, the signals can be assigned to alternative output pins.

To do this, set up the output pins, then use this command syntax:

```
gpio [ready|link|intr] <pin>
gpio ready pa1
```

The READY_N signal is assigned to Drive Output pa1 (which must be previously configured to accept the signal).

Status LED outputs setting

In GPIO mode, additional status LED outputs can be set up. For example, a yellow/green bicolor LED for Wi-Fi status and cloud-service connection status. Set these with this command syntax:

```
gpio [wifi|service] <pin>
```

Factory reset input setting

This configures an input pin to be used for resetting the configuration to the factory configuration.

```
gpio fact_rst <pin>
```

The factory configuration is the last saved setup mode - or when setup mode is disabled. The pin must be first set up as an input as shown in Setup Pin Settings (above).

When factory reset is a pushbutton input, user may restore the factory configuration by holding the factory reset pushbutton down while resetting or powering-on the module. (The button must be continuously held for about 5 seconds.)

During that time, Wi-Fi and Service LED outputs blink alternatively. If the button is held after 5 seconds, the module loads its factory configuration and does a reset. The previous startup configuration is discarded.

Since this pin is sampled only immediately after reset, an input that usually performs some other function can be used.

Registration button input setting

This configures an input pin assertion to send a message to the Ayla service and register the device to a user.

```
gpio reg <pin>
```

IMPORTANT: This should not be a button used frequently for another function. However, the button could also be used for factory reset.

Wi-Fi setup button input setting

This GPIO token configures a pushbutton for the alternative Wi-Fi setup mode (other than AP mode).

```
gpio wifi_setup_opt <pin>
```

Currently, only WeChat's Airkiss protocol is supported. This option is available from **bc-1.12** onwards.

Commit GPIO changes

Changes to module are applied to I/O pins.

```
gpio commit
```

To be fully effective, follow with `save` and `reset`.

5.4 Schedules

Schedules are supported in the module when GPIO mode is enabled. A small number (currently 14) schedule names is supported. These are numbered 0 thru 13.

Display Schedule Names

This displays the current table of schedule names.

```
sched
```

Configure Schedule Name

This configures the schedule with index 2 to have name "schedule3".

```
sched 2 name schedule3
```

The name must be a valid property name and must not be the same as any other property name.

See 7 Power Management.

5.5 Client Poll Interval

This command sets the number of seconds between requests to the Ayla Device Service.

```
client poll_interval <time>
```

The service is polled only when the Ayla Notification Service cannot be reached. If value = 0, no polling occurs.

6 Module Verification

Use the `ayla_verify.tcl` script to verify the correct product setup. It tests the connection to the Ayla service and confirms the correct template is associated with the device.

6.1 Configuration file

The `ayla_verify.tcl` program uses the config file `verify_config.txt`, in the oem.pkg directory. The `account_setup` script initializes this file, if it doesn't exist.

The file must be edited for your circumstances. Edit in Notepad (or similar text editor).

Example of ayla_verify settings

```
# 1.7 example ayla_verify configuration
#
# Configure your serial port if it is not automatically detected
#
#com_port com4
mod_version 2.4.7
#
# Give your OEM ID and OEM model here.
#
oem your-oem-id
oem_model your-oem-model
#
# Give name of ODM - Original Device Manufacturer
#
odm your-company-name-here
```

```
#
# Give info of test station access point to use for tests
#
test_wifi_ssid  your-wifi-ssid
test_wifi_key   your-wifi-key
test_wifi_sec   your-wifi-security
#
# Specify test_service as 1 if the device should connect to ADS service
# Specify test_service_props 1 if the script should connect to ADS to test
# the property list.
# Specify test_oem_auth as 1 to verify OEM authenticaton on bc-2.4.7 or later.
#
test_service 0
test_service_props 0
test_oem_auth 0
#
# If test_service and test_service_props are both 1, the following info
# is needed to verify setup.
#
# your-email and your-password are the IDs that you use to connect to the
# Ayla dashboard.  This test requires an account with the OEM-admin role.
# application-ID and application-secret are further info
# needed to inquire about the device although it isn't registered yet.
#
test_app_email  your-email
test_app_passwd your-password
test_app_id application-ID
test_app_secret application-secret
#
# If test_service and test_service_props are both 1, and you want to verify the
# property list on the service, the following comma-separated list of property
# names is used to verify that the correct template got associated.
#
test_props
Blue_button,Blue_LED,Green_LED,cmd,decimal_in,decimal_out,log,input,output,stream_down,st
ream_down_len,stream_down_match_len,stream_up,stream_up_len,version
#
```

```
# A module thruput test is done by default.
# To disable it, set test_thruput to 0.
# To set the minimum expected thruput, set test_thruput_min.
# Units are bytes/sec
#
test_thruput 1
test_thruput_min 200000
```

Description

The first two lines are the same as those for ayla_config.

```
#com_port com4
mod_version 2.4.7
```

OEM

The next two items designate OEM ID and OEM model for the product. The configuration is checked to make sure these match - and needed to access the device service.

```
#
# Give your OEM ID and OEM model here.
#
oem your-oem-id
oem_model your-oem-model
```

The next line gives the ODM name, the name of the company building the device.

```
odm your-company-name-here
```

Testing

The next three lines give the Wi-Fi configuration that is used temporarily to connect the module to the device service.

```
test_wifi_ssid  your-AP-to-the-Internet-SSID
test_wifi_key   your-AP-key
test_wifi_sec   WPA2_Personal
```

The next three lines select test features:

```
test_service 0
test_service_props 0
test_oem_auth 0
```

Normally these are changed to 1 - except perhaps for `test_service_props`, which should be set to 0.

- `test_service`, if = 1, ayla_verify connects device to the service to test if the device is enabled on the configured service. If not specified in config file, default = 0.

- `test_service_props`, if = 1, ayla_verify connects to the service to ensure the correct associated template is attached. A list of selected properties for the device are used to confirm these are associated with the device (see below).

  Requires an account with an OEM-admin user role. If user should not have admin access, set value to 0. If not specified in config file, default = 1.

- `test_oem_auth`, if = 1, expects messages from the module to indicate whether OEM authentication and template association is successful. Messages are generated by modules (firmware versions 2.4.7 or later). If an earlier version, set to 0. If not specified in config file, default = 1e.

The next four lines give further information to login to the device service as an application on behalf of your email account:

```
test_app_email  your-email-for-ADS
test_app_passwd your-ADS-password
test_app_id     your-application-id
test_app_secret your-app-secret
```

Use these lines only if `test_service_props` = 1 and `test_service` = 1.

Next, this is a list of properties to be verified as on the device service for the device (one line with commas between property names). Only required if `test_service_props` = 1 and `test_service` = 1.

```
test_props prop1_name,prop2_name,version
```

The next two are optional controls for the throughput test on `test_ap_ssid` , between PC and module in AP mode. The PC must have connectivity to the module over that network.

For `test_thruput`, 1 = run throughput test.

For `test_thruput_min`, value = minimum throughput (bytes/second) for an HTTP GET operation.

```
test_thruput 1
test_thruput_min 200000
```

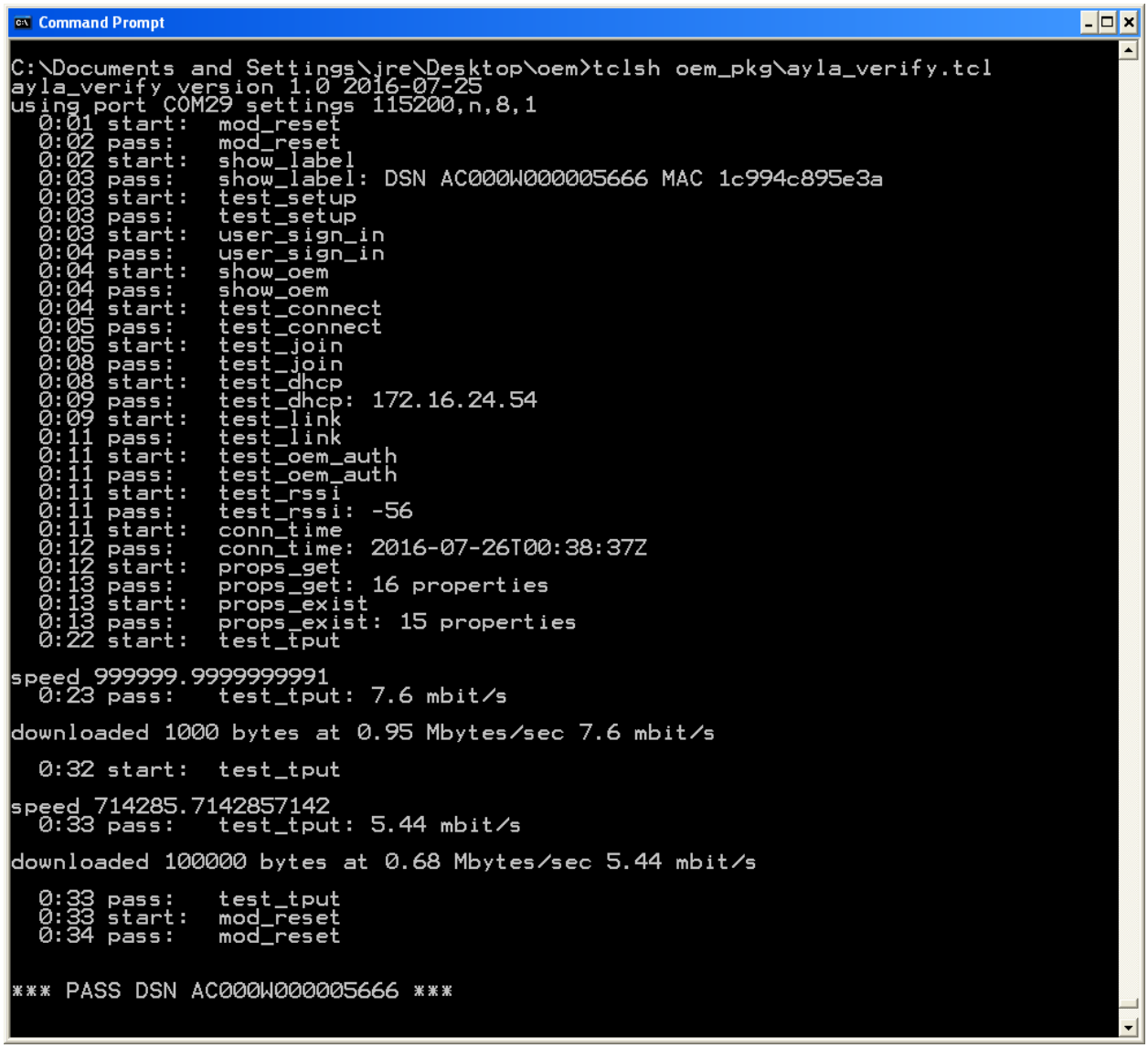
6.2 Run ayla_verify

To run the verification, connect the serial port to the device and enter the following command in your command tool window:

```
C:\ayla> tclsh oem_pkg\ayla_verify.tcl
```

The output will look something like Figure 4

Figure 4 Command line results for ayla_verfy



6.3 Options for ayla_verify

For more detailed, use the --log_user option. It returns serial port interaction with the module.

```
C:\ayla> tclsh oem_pkg\ayla_verify.tcl --log_user
```

7 Power Management

This section describes the power management modes supported by the Ayla- enabled Wi-Fi module.

NOTES
- Module power can be managed through the SPI or UART interface. Detailed information is available in the Ayla Module and MCU Interface Specification document.
- Available power management modes depend on the module type. Not all modules have the modes used in this document. For example, your module may not have a 'minimum' mode or 'standby' mode.

7.1 Power configuration settings

These are the power configuration settings for Ayla.

NOTE Average round-trip time and current consumption numbers in this document are approximate. Various factors affect trip time and current consumption. This includes, but is not limited to:
- amount of network traffic in the local network
- quality of the connection to the Ayla Device Service
- distance from AP (access point).

Power Mode

On the next restart, this is the starting power management mode.

`/power/mode`

Save this setting to apply it on the next restart.

Current Power Mode

This is the current power management mode.

`/power/current`

Use this to change the mode variable (effective immediately). Setting is not saved after restart.

Power mode variable values are:

- `max_perf` (hex value 0x32) - Use when power consumption is not an issue and the user needs the best possible performance from the network. In this mode, the CPU is running continuously. Wi-Fi is not in power savings mode. Average round-trip time of an ICMP echo packet from the local network is < 5 msecs. Average current consumption while connected to an AP is 70-100mA.
- `Default` (value 0x50) – the default mode. Module interfaces run consistently. The CPU may enter a light sleep state, and Wi-Fi in power savings mode. Power savings mode increases network latency and reduces power consumption. ICMP echo packet average round-trip time from a local network is 100-150

msecs. When idle, current consumption is 14-20 mA. The average use when posting properties to the Ayla Device Service is 40-60 mA.

- `minimum` (value 0x51) - This provides the maximum power saving while the module is connected to the cloud. The Wi-Fi is connected to the AP and periodically wakes up to check for incoming messages. The module wakes up less often in default mode. Average round-trip time of an ICMP echo packet from the local network is 300-350 msecs. While idle, current consumption is 10-15 mA. Current consumption while posting properties to Ayla Device Service the average is 40-60 mA.
- `standby` (value 0x52) – uses the minimum amount of power. The MCU actively manages when the module is turned on or off. When on, the module behaves in Minimum power management mode. When module is turned off, current consumption < 1 mA.

Minimum mode

Minimum Power management mode stops the CPU and module interfaces when the module has no work to do. The CPU wakes up when a module receives a frame from the Wi-Fi, if a module is targeted with a SPI slave select (NSS) or if it receives a UART frame from the MCU.

The MCU can talk to a module with SPI whether module is stopped or running. If module is in the stopped state it will wake up on an edge in NSS line. The MCU continues reading the SPI status byte, and the module responds with the value of 0xff until the module has woken up and is ready to process incoming requests. 0xff is an SPI status byte that must be ignored by the MCU.

In UART mode, the MCU can talk to the module whether it is stopped or running by just sending out an UART frame. On receiving a frame, if the module was stopped, it will wake up. However, to ensure the first frame is not lost (not ACKed), the MCU will have to resend the frame. On receiving the frame the second time, the module will be awake and will respond. Expected wait times for UART acknowledgements and number of retries may need to be calibrated for successful transmissions from the MCU (if the module is in this power mode).

When the module is coming out of stop mode, it responds a bit slower to the first incoming SPI request.

Standby mode

The MCU may control when mode is off or on. When the module is on, it behaves similarly to Minimum power mode. If idle, it may stop the CPU and any peripherals until there is incoming network or SPI traffic.

There are two ways to put the module into standby mode.

- The MCU can send a standby request to the module (`/power/standby` = 1.
- Turn itself off if a standby timer fires.

The timer is set when module starts. Adjust this with configuration variable `/power/standby_powered`

Before the module goes into standby, it sends a standby event to the MCU (AD_EVENT op-code with ATLV_EVENT_MASK = 0x01).

If the module was just active, it stays powered on even if the standby timer fires. The `awake_time` timer sets the time the module stays up.

With the module in standby, the MCU must reset the module to bring it back up. The MCU can use the standby event notification to remember that the module is in standby mode, and wake the module up when required by resetting it.

When the module is on, it drives the READY_N signal low when ready. When module is off, READY_N signal is floating. MCU should have a pull-up on this line.

IMPORTANT: If standby mode is used, the module must be programmed to boot up in standby mode (`setting /power/mode` set to Standby). If the module does not boot up in standby mode and `/power/current` set to Standby, the module may self-reset after 40s-60s when the hardware watchdog fires.

7.2 Wi-Fi access point mode

In AP mode, the module uses much more power. If Wi-Fi has not been configured to connect anywhere, the module is in AP mode (first time module is plugged in).

If power savings mode is set to Standby, Minimum or Default mode, module turns itself off after `unconf_powered` timer expires. Set this time with the configuration variable `/power/unconf_powered`. If Wi-Fi is configured during this time period, this timer is not applied. When power savings mode is Standby, this timer takes precedence over `/power/standby_powered`. The module can be turned on again by resetting it.

Current consumption while in AP mode is about 70mA when we're not transmitting, and about 256mA when transmitting a beacon. Average is ~72mA.

7.3 Minimize power consumption

Deciding which power management to use depends on the application. If the device is not battery-powered and lower latency performance is required, use `max_perf` or Default mode. If power savings is required and the module must remain connected to the cloud to receive notifications or commands, the Minimum mode is the best mode. This minimizes power consumption, while keeping the SPI/UART and cloud connection active (at the cost of lower latency).

If more power saving is needed, configure the module to Standby mode. When MCU has datapoints to send, it can wake up the module by resetting it. After the module is awake, it connects to the Ayla Device Service and accepts the first datapoint.

The MCU can continue to post additional datapoints, or return to Standby mode (set `/power/standby` to 1).

When the module is off, all lines connected to the MCU and module are floating. Input lines MISO, LINK_N, INTR_N, and READY_N should be pulled high. Alternatively, the MCU can detach other input pins, except for READY_N, when it sees READY_N start to float.

7.4 Change configurations

It is possible to change the various power management configurations using the command line interface (CLI). For debugging or evaluation use the `power` command.

Use the syntax in the sample shown below:

```
power <mode|current> <max_perf|default|min|standby>
power <awake_time|standby_powered|unconf_powered> <time_in_seconds>
```

This command controls how the power management behaves.

- “power current” - adjusts the currently active power savings level
- “power mode” - controls the power savings level that the module will be in when it boots up the next time

When the current power savings level is changed, related power management timers are reset.

- `power awake_time` - how long the module stays in Default power saving mode after activity. After the time has passed, the module goes back to the Minimum power saving mode. This timer is used when power saving mode is Default, Minimum or Standby.
- `power standby_powered` - how long the module stays active after booting up. After the timer fires, the module CPU goes to Standby low-power mode. This timer is only used when power savings mode is set to Standby.
- `power unconf_powered` - how long the module stays active after boot up when no valid Wi-Fi profile is present. This timer is only used when power savings mode is set to Minium or Standby, and there is no configured Wi-Fi profile.

Appendix A: Quick Start OEM Package Setup and Installation

This section provides directions for setting up and installing the Ayla Networks OEM package. If you have any questions, need more information or details on this process, please see the *Ayla OEM Installation Guide*.

1. In your destination folder, extract **oem_pkg-XXXX-X.X.zip** and rename **oem_pkg-XXXX-X.X** to **oem_pkg**.
2. In the same destination folder, extract **update-XXXXXXXX -X.X.XX.zip**.
3. The directory should look something like this:

   ```
   C:\ayla> dir
   <DIR> oem_pkg
   <DIR> update-AY001MUS-1.9.13
   ```

4. To setup an account, enter:

   ```
   C:\ayla> oem_pkg\account_setup
   ```

5. Edit the config.txt file as needed to set the values for your device.

   Use a simple text editor (e.g., Notepad) to preserve the ASCII format. Avoid WordPad (and other complicated editors). These could change the text to an unacceptable internal format.

6. To run update/config from a single command, type:

   ```
   C:\ayla> oem_pkg\ayla_update <path to update package>
   ```

   For example, if you followed the instructions above it would look something like this:

   ```
   C:\ayla> oem_pkg\ayla_update update-AY001MTC-2.4.7
   ```

This will update the module FW and then run the config script to configure the module. You should see a PASS or FAIL message when the script completes execution.