Morse Micro



MM-IoT-SDK Software Release Notes

MCU Software Package

Copyright © Morse Micro

Table of Contents

1 Overview 6
2 Software Change History 7
2.1 Release 2.14.3 7
2.1.1 New Features 7
2.1.2 Other Improvements 7
2.1.3 Bug Fixes 8
2.1.4 Known Issues 8
2.1.4.1 Makefile Infrastructure 8
2.1.4.2 RTOS 9
2.1.4.3 Example Applications 9
2.1.4.4 AP Mode (Alpha Release) 9
2.1.4.5 DPP (Beta Release) 9
2.1.4.6 Stability 10
2.1.5 Migration from 2.13.1 10
2.1.5.1 Relay Mode API changes 10
2.1.5.2 Elliptic Curve Cryptography 10
2.1.5.3 MMHAL includes 10
2.1.6 Components 10
2.2 Release 2.13.1 11
2.2.1 New Features 11
2.2.2 Other Improvements 12
2.2.3 Bug Fixes 12
2.2.4 Known Issues 13
2.2.4.1 Makefile Infrastructure 13
2.2.4.2 RTOS 13
2.2.4.3 Example Applications 13
2.2.4.4 AP Mode (Alpha Release) 13
2.2.4.5 DPP (Beta Release) 14
2.2.4.6 Stability 14
2.2.5 Migration from 2.12.3 14
2.2.5.1 BCF Files 14
2.2.6 Components 14
2.3 Release 2.12.3 15
2.3.1 New Features 15
2.3.2 Other Improvements 15
2.3.3 Bug Fixes 16
2.3.4 Known Issues 16

2.3.4.1 Makefile Infrastructure 16
2.3.4.2 RTOS 16
2.3.4.3 Example Applications 17
2.3.4.4 AP Mode 17
2.3.4.5 DPP 17
2.3.4.6 Stability 17
2.3.5 Migration from 2.11.2 18
2.3.5.1 MMWLAN API 18
2.3.5.2 Flash HAL API 18
2.3.5.3 UART HAL API 18
2.3.5.4 MMAGIC API 18
2.3.6 Components 18
2.4 Release 2.11.2 19
2.4.1 New Features 19
2.4.2 Other Improvements 20
2.4.3 Bug Fixes 22
2.4.4 Known Issues 22
2.4.4.1 Makefile Infrastructure 22
2.4.4.2 RTOS 22
2.4.4.3 Example Applications 22
2.4.4.4 AP Mode 23
2.4.4.5 DPP 23
2.4.4.6 Stability 23
2.4.5 Migration from 2.10.4 23
2.4.5.1 Toolchain updates 23
2.4.5.2 MMWLAN API 24
2.4.5.3 MMWLAN HAL API 24
2.4.5.4 MMAGIC API 24
2.4.6 Components 24
2.5 Release 2.10.4 25
2.5.1 New Features 25
2.5.2 Other Improvements 25
2.5.3 Bug Fixes 28
2.5.4 Known Issues 29
2.5.4.1 Makefile Infrastructure 29
2.5.4.2 RTOS 29
2.5.4.3 Example Applications 30
2.5.4.4 Standby/Offload Features 30
2.5.4.5 AP 30

2.5.4.6 DPP 30
2.5.4.7 Stability 31
2.5.5 Migration from 2.9.7 31
2.5.5.1 Combined MM6108 and MM8108 libraries 31
2.5.5.2 MMWLAN API 31
2.5.5.3 MMHAL API 32
2.5.5.4 Cryptography 32
2.5.5.5 Makefile and infrastructure 32
2.5.5.6 Emmet 33
2.5.5.7 Regulatory Database 33
2.5.5.8 Notable new API 33
2.5.6 Deprecation Notice 33
2.5.6.1 FreeRTOS-plus-TCP network stack 33
2.5.6.2 Platform IO 33
2.5.6.3 Bootloader example application 33
2.5.6.4 Offload functionality 34
2.5.7 Components 34
2.6 Release 2.9.7 34
2.6.1 New Features 34
2.6.2 Other Improvements 35
2.6.3 Bug Fixes 36
2.6.4 Known Issues 37
2.6.4.1 Makefile Infrastructure 37
2.6.4.2 Stability 37
2.6.4.3 Throughput 38
2.6.4.4 Example Applications 38
2.6.4.5 Standby/Offload Features 38
2.6.4.6 Soft AP 38
2.6.5 Migration from 2.8.2 39
2.6.5.1 mbedTLS configuration 39
2.6.5.2 MMOSAL API 39
2.6.5.3 MMWLAN API 39
2.6.5.4 MMHAL API 40
2.6.5.5 MMAGIC 40
2.6.5.6 MQTT agent 40
2.6.5.7 Makefile infrastructure 41
2.6.5.8 Morse Firmware/BCFs 41
2.6.6 Components 41
3 System Details 41

3.1 System Memory Requirements 41
3.2 HaLow Throughput Performance 42
4 Revision History 42

1 Overview

The Morse Micro IoT SDK includes the host-side components needed to use an MCU with an MM6108 or MM8108 transceiver. This includes drivers, the Upper MAC (UMAC) layer, the network stack, and the RTOS.

The MM IoT SDK release package includes the following:

- **morselib:** a library containing the Morse Micro driver, UMAC, and WPA supplicant.
- **Source code:** RTOS (FreeRTOS), network stack (LwIP, mbedTLS, coreMQTT, etc.), platform BSPs, and shim layers.
- **Example applications:** Various example applications to aid in understanding how to use the SDK and accelerate development efforts.
- **Additional libraries developed by Morse Micro:** mmconfig for key-value persistent storage and mmagic for CLI interface generation.
- **API documentation:** for Makefile-based development.
- **Getting Started Guide:** for Makefile-based development.
- **Setup scripts:** development machine setup scripts and tools.

The SDK supports Makefile-based development on Linux with experimental support on macOS. Ubuntu 22.04 x86_64 is the officially supported Linux distribution.

2 Software Change History

2.1 Release 2.14.3

This section outlines significant changes from release 2.13.1.

2.1.1 New Features

- Updated Morse firmware to version 2.1.
- Updated the relay enable API to bring up both the relay subsystem and the interface. This simplifies bringing up a relay device.
- Added relay recovery timeout mechanism when node loses path to root.
- Added support for AP duty cycle limits, enabling initial support for EU/GB/IN country codes in AP mode.
- Open-sourced core Emmet test engine code.
    - This brings feature parity between compiling from source and pre-compiled morselib binary files.
    - Pre-compiled moreslib binary files are now marked as deprecated.
    - Compiling morselib from source is now the default build configuration.

2.1.2 Other Improvements

- Updated relay depth change callback to relay status callback. Also unified depth and state getter into a single status getter.
- Updated relay depth override API to reflect test-only usage.
- Support added for broadcast deauth on AP disable.
- Optimized AP mode bring-up duration.
- Updated AP mode to reject excess connections earlier in the connection process.
- Improved error handling in AP mode.
- Improved handling of unexpected packet sequence numbers and reduced packet loss.
- Improved robustness of MM chip boot procedure.
- Improved failure logging in assert handler.
- Enabled MBedTLS Assembly instructions for optimized bignum operations.
- Enabled MBedTLS NIST elliptic curve point optimizations (MBEDTLS_ECP_NIST_OPTIM) for platforms without hardware acceleration. This reduces association times significantly (~2.8 seconds on u575) at a cost of an additional ~2.7 KB FLASH.
- Removed unconditional compilation of optional NIST curves P-384 and P-521. Compilation can be re-enabled with build variables. Saves ~11 KB FLASH on platforms that perform cryptographic SAE operations via software.
- Explicitly set supplicant SAE groups to mandatory group 19 only, when no default is specified. This saves >1.3s of time to first connection.

- Added support for loading a set of channels from mmconfig for optimized scanning. This can significantly speed up association time for deployments with well-defined expected primary frequencies.
- Moved AP/Relay config load function to the mm_app_loadconfig.c helper file.
- Moved MMHAL littlefs API from mmhal_flash.h to dedicated mmhal_fs.h header file.
- Removed unused mmhal_uart_config API.
- Added APIs to mmconfig to read/write a MAC address.
- Updated mm_app_common raw printf logs with MMLOG.
- Reduced the log level of the mmipal netif link status message.
- Removed Emmet API for autoconnect enable/disable.
- Added simplified log message format to example Makefiles for production builds. This makes log messages easier to read and saves ~14KB of FLASH by reducing the size of LWIP assert messages.
- Removed miniterm alias and updated usage instructions to use pyserial-miniterm.
- Updated the user guide link and the BCF directory path.
- Updated the documentation for supported regulatory domains to include only domains available in MMREGDB.
- Improved shell path management for macOS.
- Improved support for CubeMX code generation on macOS.

2.1.3 Bug Fixes

- Fixed recovering relay node reporting new depth before recovery completed.
- Fixed buffer overrun during FW decompression.
- Fixed memory leak on WLAN shutdown from AP mode.
- Fixed AP mode sending DTIM beacons at the wrong time.
- Fixed printf always returning 0.

2.1.4 Known Issues

2.1.4.1 Makefile Infrastructure

- Building fails when the MM-IoT-SDK package directory has a space in the path
    - The MM-IoT-SDK package must be unpacked into a directory without spaces in its name.

2.1.4.2 RTOS

- Timeouts with FreeRTOS time slicing disabled
    - FreeRTOS time slicing is enabled by default in the MM-IOT-SDK FreeRTOS configuration. If this is disabled, timeouts may be observed when sending commands to the MM chip.
- System time drifts on STM32-based reference platforms because of the low-accuracy clock. Higher accuracy can be achieved by switching to the High-Speed External (HSE) clock source at the cost of higher power consumption when active.

2.1.4.3 Example Applications

- Failures when lwIP is built with LWIP_IPV4=0
    - If IPv4 is disabled when LWIP is built (i.e., IPv6-only), the IP stack may not work as expected. The workaround is to always enable IPv4.

2.1.4.4 AP Mode (Alpha Release)

Note that AP Mode is in alpha and is not intended for production use.

- Traffic is not forwarded between STAs
    - AP mode does not forward frames between connected STAs. Therefore, connected STAs cannot communicate with each other unless bridging is implemented at the application layer.
    - This functionality can be achieved by enabling relay mode and setting the AP as the root node. No changes are required on the STA devices to facilitate this.
- Recovery from health failures is not supported in AP mode
    - If a health check fails while operating in AP mode, the system will restart.
    - Mitigation is to disable the health check by invoking:
      `mmwlan_set_health_check_interval(0, 0);`
- OWE security is not supported
    - Mitigation is to use Open or SAE security.

2.1.4.5 DPP (Beta Release)

Note that DPP is in beta and is not intended for production use.

- DPP must be stopped explicitly before shutdown
    - If DPP has been started (via `mmwlan_dpp_start()`), it must be stopped explicitly via `mmwlan_dpp_stop()` prior to invocation of `mmwlan_shutdown()`.

2.1.4.6 Stability

- Some link instability may occur at specific signal levels, resulting in an elevated packet error rate.
    - Minor changes to the distance from the AP, antenna orientation, or environment will correct it.

2.1.5 Migration from 2.13.1

2.1.5.1 Relay Mode API changes

There have been significant breaking changes made to the relay mode API. If an application has been implemented using relay mode, the interface bring-up sequence will require updating. Refer to the Relay Mode documentation as well as the updated relay_mode example application.

2.1.5.2 Elliptic Curve Cryptography

The optional NIST curves P-384 and P-521 are no longer enabled by default.

If the application requires these curves, either add these curves to the mbedtls_config.h file, or define the make environment variable `ENABLE_MBEDTLS_P384=y` and/or `ENABLE_MBEDTLS_P521=y` as required.

If a compilation error about `MBEDTLS_ECP_MAX_BITS` being redefined occurs, a stale definition is likely. One such place is if the target platform has a copy of ecp_alt.h which defines:
`#define MBEDTLS_ECP_MAX_BITS 521`
If this is the case, remove the define and defer to the correct definition from MbedTLS.

2.1.5.3 MMHAL includes

If a compilation error regarding `mmhal_get_littlefs_config appears`, add/update the include path from `mmhal_flash.h` to `mmhal_fs.h` to resolve this.

2.1.6 Components

Component	Version
MM6108 Morse Firmware	mm6108-2.1.1
MM6108 BCF API	12.1.0*
MM8108 Morse Firmware	mm8108-2.1.0
MM8108 BCF API	13.1.0^
Regulatory Database	2.8.0
FreeRTOS kernel	11.2.0
LWIP	2.2.0 (+modifications)
mbedTLS	3.6.5
STM32U5 BSP	1.8.0
STM32WB55 BSP	1.14.1

* Backward compatible with 8.0.0, but features may be limited.

^ API not backward-compatible; BCF must be updated.

2.2 Release 2.13.1

This section outlines significant changes from release 2.12.3.

2.2.1 New Features

- Updated Morse firmware and BCFs to version 2.0.
    - NOTE: MM8108 2.0 has a breaking BCF API change. New BCF files **must** be used. See section "Migration from 2.12.3 - BCF Files" below.
- Added topology depth awareness to the relay subsystem.
- Added support for uniform SSID in relay mode.
- Added support for fast recovery of relay tree on link failure.

2.2.2 Other Improvements

- Added additional validation to mmhal_read_mac_addr to ensure the station MAC address is valid.
- Added support for constraining the channel list used when scanning, enabling targeted scans over a specified set of channels.
- Added additional threading tests to Porting Assistant.
- Increased timer pool for LWIP on-demand timers.
- Added mmwlan API to configure the beacon loss count threshold.
- Added a helper function to dump Morse chip stats to MCU console log in hex format.
- Added ability to insert Vendor IEs into beacons and probe responses.
- Added additional AP mode parameters to mmconfig.
- Implemented buffering of management frames for power-saving STAs.
- Improved validation of AP enable arguments.
- Added transmit of 4-address Quality of Service (QoS) Null frame when relay is enabled.
- Added relay API to query relay system state.
- Added relay API to query the current relay depth.
- Added test API to set relay depth override.
- Added a requirement to enable relay mode before enabling the STA or AP interface.
- Added relay API to register a callback on depth change.
- Updated relay disable to return success when not active.
- Restricted STA scan to the AP's primary channel when in relay mode.
- Prevent STA mode from being enabled when the system is a relay root node.
- Added LED status indicators to relay mode example application.
- Updated relay example application for recent API changes.

2.2.3 Bug Fixes

- Fixed unreleased TCP/IP lock in LWIP netif status callback.
- Fixed inaccurate documentation for mmconfig wlan.duty_cycle_mode.
- Fixed missing newline in LWIP error log.
- Fixed the AP interface incorrectly registering as the active network interface during relay mode.
- Fixed AP responding to probe requests on non-primary channels.
- Fixed AP beacon incorrectly signaling buffered traffic.
- Fixed relay node dropping broadcast traffic it generated.
- Fixed relay node dropping traffic from downstream STA.

2.2.4 Known Issues

2.2.4.1 Makefile Infrastructure

- Building fails when the MM-IoT-SDK package directory has a space in the path
    - The MM-IoT-SDK package must be unpacked into a directory without spaces in its name.

2.2.4.2 RTOS

- Timeouts with FreeRTOS time slicing disabled
    - FreeRTOS time slicing is enabled by default in the MM-IOT-SDK FreeRTOS configuration. If this is disabled, timeouts may be observed when sending commands to the MM chip.
- System time drifts on STM32-based reference platforms because of the low-accuracy clock. Higher accuracy can be achieved by switching to the High-Speed External (HSE) clock source at the cost of higher power consumption when active.

2.2.4.3 Example Applications

- Failures when lwIP is built with LWIP_IPV4=0
    - If IPv4 is disabled when LWIP is built (i.e., IPv6-only), the IP stack may not work as expected. The workaround is to always enable IPv4.

2.2.4.4 AP Mode (Alpha Release)

Note that AP Mode is in alpha and is not intended for production use.

- Traffic is not forwarded between STAs

    - AP mode does not forward frames between connected STAs. Therefore, connected STAs cannot communicate with each other unless bridging is implemented at the application layer.
    - This functionality can be achieved by enabling relay mode and setting the AP as the root node. No changes are required on the STA devices to facilitate this.
- Recovery from health failures is not supported in AP mode
    - If a health check failure occurs while operating in AP mode, it will trigger a system restart.
    - Mitigation is to disable the health check by invoking:
      `mmwlan_set_health_check_interval(0, 0);`
- OWE security is not supported
    - Mitigation is to use Open or SAE security.
- EU, GB, and IN regulatory domains are not supported in AP mode.

2.2.4.5 DPP (Beta Release)

Note that DPP is in beta and is not intended for production use.

- DPP must be stopped explicitly before shutdown
    - If DPP has been started (via `mmwlan_dpp_start()`), it must be stopped explicitly via `mmwlan_dpp_stop()` prior to invocation of `mmwlan_shutdown()`.

2.2.4.6 Stability

- Some link instability may occur at specific signal levels, resulting in an elevated packet error rate.
    - Minor changes to the distance from the AP, antenna orientation, or environment will correct it.

2.2.5 Migration from 2.12.3

2.2.5.1 BCF Files

MM8108 2.0.0 Firmware has a breaking API change in the BCF API. This requires updating the BCF file. The MM8108 chip will not perform any RF operations if an old BCF file is loaded.

NOTE: Even when there are no breaking changes, we recommend updating to the latest BCF file.

2.2.6 Components

Component	Version
MM6108 Morse Firmware	mm6108-2.0.1
MM6108 BCF API	12.1.0*
MM8108 Morse Firmware	mm8108-2.0.0
MM8108 BCF API	13.1.0^
Regulatory Database	2.8.0
FreeRTOS kernel	11.2.0
LWIP	2.2.0 (+modifications)
mbedTLS	3.6.5
STM32U5 BSP	1.8.0
STM32WB55 BSP	1.14.1

* Backward compatible with 8.0.0, but features may be limited.

^ API not backward-compatible; BCF must be updated.

2.3 Release 2.12.3

This section outlines significant changes from release 2.11.2.

2.3.1 New Features

- Device Provisioning Protocol (DPP), QR code mode functionality as STA (Beta)

2.3.2 Other Improvements

- Updated Morse firmware and BCFs to 1.17.9.
- Added validation to enforce that simultaneous AP and STA interfaces operate on the same channel.
- Added a benchmark test to Porting Assistant that verifies that the system executes instructions fast enough to meet the expected throughput.
- Removed 2MHz EU channels from the regulatory database.
- Added regulatory database documentation to the user guide.
- Added Makefile targets to simplify programming and reconfiguring devices.
- Added a check to example Makefiles that detects non-object files in `OBJS`.
- Added support for defragmentation of management frames. In practice, this affects only large management frames, such as DPP GAS frames.
- Decoupled DPP credentials from the PB_RESULT event. There is now a new MMWLAN_DPP_EVT_CONF_RECEIVED event that is raised when the configuration is available; this makes it portable for different DPP methods (e.g., DPP QR).

2.3.3 Bug Fixes

- Fixed a crash that occurred when mmwlan_sta_disable() was invoked while an association request retry timer was pending.
- Fixed a crash caused by calling an ISR-context FreeRTOS function from a non-ISR background task.
- Fixed system time drifting when exiting deep sleep.

2.3.4 Known Issues

2.3.4.1 Makefile Infrastructure

- Building fails when the MM-IoT-SDK package directory has a space in the path
    - The MM-IoT-SDK package must be unpacked into a directory without spaces in its name.

2.3.4.2 RTOS

- Timeouts with FreeRTOS time slicing disabled
    - FreeRTOS time slicing is enabled by default in the MM-IOT-SDK FreeRTOS configuration. If this is disabled, timeouts may be observed when sending commands to the MM chip.
- System time drifts on STM32-based reference platforms because of the low-accuracy clock. Higher accuracy can be achieved by switching to the High-Speed External (HSE) clock source at the cost of higher power consumption when active.

2.3.4.3 Example Applications

- Failures when lwIP is built with LWIP_IPV4=0
    - If IPv4 is disabled when LWIP is built (i.e., IPv6-only), the IP stack may not work as expected. The workaround is to always enable IPv4.

2.3.4.4 AP Mode

Note that AP Mode is alpha and not for production use.

- Traffic is not forwarded between STAs
    - AP mode does not forward frames between connected STAs. Therefore, connected STAs cannot communicate with each other unless bridging is implemented at the application layer.
    - This functionality can be achieved by enabling relay mode and setting the AP as the root node. No changes are required on the STA devices to facilitate this.
- Recovery from health failures is not supported in AP mode
    - If a health check failure occurs while operating in AP mode, it will trigger a system restart.
    - Mitigation is to disable the health check by invoking:
      ```
      mmwlan_set_health_check_interval(0, 0);
      ```
- OWE security is not supported
    - Mitigation is to use Open or SAE security.
- STAs may not be able to associate to the AP in EU and GB regulatory domains.

2.3.4.5 DPP

Note that DPP is beta and not for production use.

- DPP must be stopped explicitly before shutdown
  - If DPP has been started (via `mmwlan_dpp_start()`), it must be stopped explicitly via `mmwlan_dpp_stop()` prior to invocation of `mmwlan_shutdown()`.
- DPP is not currently supported in regulatory domains that contain multiple channels sharing the same center frequency but at different bandwidths, in particular JP and AU REVmf.

2.3.4.6 Stability

- Some link instability may be seen at some specific signal levels, resulting in an elevated packet error rate.
  - Minor changes to the distance from the AP, antenna orientation, or environment will correct it.

2.3.5 Migration from 2.11.2

2.3.5.1 MMWLAN API

Added QR Code Enrollee mode, selectable via the new `mode` field in `mmwlan_dpp_args`. The event enum was expanded to include granular auth/config phase events, and SSID/passphrase was moved from `args.pb_result` to `args.conf_received` (fired as `MMWLAN_DPP_EVT_CONF_RECEIVED` for both modes). Update callbacks accordingly and use `MMWLAN_DPP_ARGS_INIT` to initialize the expanded struct.

2.3.5.2 Flash HAL API

The Flash HAL API now uses `mmhal_flash_addr_t` for address in place of `uint32_t`. This type is defined as `uint32_t` by default unless overridden by `MMHAL_FLASH_ADDR_T`.

Add optional `mmhal_get_factory_mmconfig_partition()` to allow a factory default partition to be specified for mmconfig.

2.3.5.3 UART HAL API

Added new functions, `mmhal_uart_validate_config()` and `mmhal_uart_configure()`, for validating and setting UART configuration (baud rate and flow control).

2.3.5.4 MMAGIC API

DPP API has been updated.

2.3.6 Components

Component	Version
Morse firmware	1.17.9
Regulatory Database	2.8.0
FreeRTOS kernel	11.2.0
LWIP	2.2.0 (+modifications)
mbedTLS	3.6.5
STM32U5 BSP	1.8.0
STM32WB55 BSP	1.14.1
BCF API	12.1.0 (backward compatible to 8.0.0, but features may be limited)

2.4 Release 2.11.2

This section outlines significant changes from release 2.10.4.

2.4.1 New Features

- Simultaneous AP/STA support
- Subset of S1G Relay support, including Reachable Address Update [alpha]

2.4.2 Other Improvements

- Added MMWLAN API to register a callback to be invoked in case of unrecoverable errors.
- Added a workaround to prevent indefinite pause of transmission when traffic control resume events are missed during spread mode duty cycle operation.
- Added MMWLAN API to retrieve channel information for a VIF.
- Updated the FreeRTOS kernel to v11.2.
- Updated board support packages for STM32U5 and STM32H7 to the latest release version.
- Updated the mbedtls submodule version from 3.6.2 to 3.6.5. The main benefit is a large number of security fixes (11 issues with Security Advisories) and a few bug fixes. See the full changesets at https://github.com/Mbed-TLS/mbedtls/releases.
- Added missing `const` qualifiers in mmconfig.
- Error message with file hash: line number is now logged when a chip assert occurs.
- Added a mechanism to hold the transceiver in reset when the MCU is halted on assertion in debug builds.
- Added a function to assert/deassert the reset pin on the Wi-Fi HaLow transceiver in mmhal_wlan.
- mmutils.h now uses MMLOG_XX macros instead of calling printf() directly.
- Added additional information in the sta_status updates in the ap_mode example application.
- Fixed typos in `mmagic_controller.h` function docstrings.
- Increased the response timeout for `mmagic_mqtt_stop_agent` to improve reliability.
- Added connect_on_boot configuration option to MMAGIC.
- Added detection and disconnection of inactive STAs in AP mode.
- Implemented `mmwlan_ap_disable()` to fully disable AP mode operation, replacing the previous stub implementation.
- Implemented deauthentication of unrecognized stations in AP mode to ensure STAs with stale associations are properly notified when the AP has no record of their association.
- Added an optional mmpkt pool for management-class frames (used in AP mode).
- Extended API readiness labels and added documentation explaining usage.
- Added ap_ prefix to mmconfig WLAN parameters in ap_mode example application.
- Added mmwlan statistics for TX frames that timed out in the send path.
- Added MMWLAN_NOT_SUPPORTED status code.
- Reduced recovery time when a chip assertion is detected.

- Removed freertos-plus-tcp network stack support.
- Removed the bootloader that was previously provided with MM-IoT-SDK, as it was never productionized. In addition, the AWS OTA example that relied on it has been removed.
- Increased size of command pool in reference pktmem implementations to account for larger commands.
- Clarified behavior and parameters for mmwlan_ate_execute_command().
- The standby mode and offload features that had previously been deprecated have now been removed from the MM-IoT-SDK.
- Updated firmware binaries to v1.17.8.
- Refactored the mmiperf client and server implementations into separate source files to improve code maintainability.
- Added an API to stop iperf servers and in-progress iperf sessions.
- Removed load_mmwlan_settings() prototype from mmconfig.h. It is provided by mm_app_loadconfig.h.
- Updated the default NTP server to "time.aws.com" in the aws_iot example to increase robustness and stability.
- Updated scan example application to use the common version function from mm_app_common.
- Improved clarity of mmhal_flash documentation.
- Replaced ARM-specific inline Assembly code in FreeRTOSConfig.h with MMPORT_BREAKPOINT().
- Fixed a missing include in `mqtt_agent_task_powersave.c`.
- Relaxed handling of `CHANNEL_LIST_NOT_SET` errors during network interface initialization to allow startup to proceed without a fully configured channel list.
- Updated LWIP configuration to only set LWIP_PROVIDE_ERRNO if LWIP_ERRNO_STDINCLUDE is not defined.
- Renamed MMAGIC “tcp” module to “socket”.
- Added UDP client and server support to MMAGIC through the socket API, matching the existing TCP interface.
- Enum types in the MMAGIC API are now packed to enforce consistent serialization.
- Added MMAGIC DPP push button command support.
- Increased mmagic_controller_wlan_commit_all timeout.
- Added basic example application for relay mode.
- Added experimental support for MacOS (Apple silicon).
- Updated the default toolchain version to GCC14.
- LWIP assertion messages are now printed in LWIP_PLATFORM_ASSERT().
- Added morselib and HW version to the Porting Assistant example application output.
- Added mmpkt_list API function to dequeue multiple packets from a list at once.
- Added version information output to m2m reference applications.
- Made the receive timeout for the mbedTLS transport interface configurable.
- Added example application to demonstrate raw UDP and TCP client sockets.

- The ekh08-wb55 and ekh18-wb55 platforms now use the hardware crypto libraries by default, reducing association time.
- On-demand timers are added to the LWIP network stack to enable event-based timing, reducing power consumption. This option is toggled using ON_DEMAND_TIMERS_ENABLED and is enabled by default.
- Removed support for PlatformIO.
- Updated the README to link to the GitHub Pages HTML documentation.
- Added a new configuration variable to MMAGIC MQTT for the configuration of the keep alive interval.

2.4.3 Bug Fixes

- Fixed a bug where the CLI application would fail an assertion if a country code was not programmed to the config store.
- Fixed an incorrect return code in the porting assistant `test_wlan_io.c`, `process_sdio_spi_multi_byte_return()`.
- Fixed a bug that could cause system instability when the M2M agent datalink state was updated from an interrupt service routine context.
- Fixed a bug where MMAGIC M2M ping would fail and report incorrect results.
- Fixed AP enable API returning before the AP interface was started.
- Improved scan result filtering to increase robustness of AP selection.
- Fixed incorrect message in UART HAL.
- Fixed an issue where the LwIP network interface bypassed the standard `netif->input` dispatch path by calling `tcpip_input` directly.
- Improved stability of hardware reset functionality for SDIO platforms.
- Fixed incorrect range in implementation of `mmhal_random_u32(min, max)` for WB55 platforms.

2.4.4 Known Issues

2.4.4.1 Makefile Infrastructure

- Building fails when the MM-IoT-SDK package directory has a space in the path
    - The MM-IoT-SDK package must be unpacked into a directory without spaces in its name.

2.4.4.2 RTOS

- Timeouts with FreeRTOS time slicing disabled
    - FreeRTOS time slicing is enabled by default in the MM-IOT-SDK FreeRTOS configuration. If this is disabled, timeouts may be observed when sending commands to the MM chip.
- System time drifts when the host is in deep sleep

- System time may drift from wall clock time when deep sleep is enabled. This may result in iperf taking longer than expected to complete. A 1-2% drift has been observed on STM32U575 and 3-4% on STM32WB55, but this may vary with activity levels.

2.4.4.3 Example Applications

- Failures when lwIP is built with LWIP_IPV4=0
  - If IPv4 is disabled when LWIP is built (i.e., IPv6-only), the IP stack may not work as expected. The workaround is to always enable IPv4.

2.4.4.4 AP Mode

Note that AP Mode is alpha and not for production use.

- Traffic is not forwarded between STAs
  - AP mode does not forward frames between connected STAs. Therefore, connected STAs cannot communicate with each other unless bridging is implemented at the application layer.
  - This functionality can be achieved by enabling relay mode and setting the AP as the root node. No changes are required on the STA devices to facilitate this.
- Recovery from health failures is not supported in AP mode
  - If a health check failure occurs while operating in AP mode, it will trigger a system restart.
  - Mitigation is to disable the health check by invoking:
    ```
    mmwlan_set_health_check_interval(0, 0);
    ```
- OWE security is not supported
  - Mitigation is to use Open or SAE security.

2.4.4.5 DPP

Note that DPP is beta and not for production use.

- DPP must be stopped explicitly before shutdown
  - If DPP has been started (via `mmwlan_dpp_start()`), it must be stopped explicitly via `mmwlan_dpp_stop()` prior to invocation of `mmwlan_shutdown()`.
- DPP is not currently supported in regulatory domains that contain multiple channels sharing the same center frequency but at different bandwidths, in particular JP and AU REVmf.

2.4.4.6 Stability

- Some link instability may be seen at some specific signal levels, resulting in an elevated packet error rate.
    - Minor changes to the distance from the AP, antenna orientation, or environment will correct it.

2.4.5 Migration from 2.10.4

2.4.5.1 Toolchain updates

The GCC toolchain has been updated. Please rerun the setup script to ensure the correct version is installed at the expected location.

2.4.5.2 MMWLAN API

A new state, MMWLAN_AP_STA_AUTHENTICATED, has been added to the mmwlan_ap_sta_state enumeration.

The function mmwlan_ap_enable() now blocks by default until the AP interface is up. This can be overridden to the previous behavior by setting the async_start option in the AP arguments.

Deprecated Offload API has been removed and is no longer available for application use.

2.4.5.3 MMWLAN HAL API

New function mmhal_wlan_assert_reset() needs to be implemented by the WLAN HAL. Reference HALs have been updated to reflect this.

2.4.5.4 MMAGIC API

The `tcp` module of MMAGIC has been renamed to `socket`.

Deprecated Offload API has been removed.

2.4.6 Components

Component	Version
Morse firmware	1.17.8
Regulatory Database	2.7.0
FreeRTOS kernel	11.2.0
LWIP	2.2.0 (+modifications)
mbedTLS	3.6.5
STM32U5 BSP	1.8.0
STM32WB55 BSP	1.14.1
BCF API	12.1.0 (backward compatible to 8.0.0, but features may be limited)

2.5 Release 2.10.4

This section outlines significant changes from release 2.9.7.

2.5.1 New Features

- Combined MM6108 and MM8108 into a single release package
- Added support for the new AU 802.11-2024 and 802.11-REVmf channel maps.
- Added support for handling Extended Channel Switch Announcements (ECSAs) in STA mode.
- Added morselib source code.
- Added MM8108 support for ESP32.
- Added support for ESP32-P4.
- Added ap_mode example application for ESP32 platforms.

2.5.2 Other Improvements

- Added morselib version header (`mmversion.h`)
- Improved consistency of code style in the example application source code.
- Updated the list of BCFs that are now included in the MM-IoT-SDK release package.
- Updated ESP32 binary blob search behavior to be a recursive search.
- Moved BCFs into a morsefirmware/<chip>/bcfs/ directory structure to clarify which BCF is for which chip type.
- Added duty cycle limits to EU and GB regulatory domains (2.8% for STA, 10% for AP).
- Updated ap_mode example to use Operating Class 71 (US, 8 MHz channel spacing) by default.
- Moved application-specific regulatory database implementations into a dedicated Morse Micro regulatory database component.
- Set the default AU channel list to IEEE 802.11-REVmf. This will be used if the country code "AU" is specified.
- Added an assert on health check failure in AP mode, as health check recovery is not supported.
- Added the ability to configure the maximum allowed connections when enabling AP mode.
- Added MAC address helper functions to mmutils.h.
- Added a warning to the documentation and an error message to mmwlan_ap_enable to make it explicitly clear that OWE is not currently supported by AP mode.
- Disabled FreeRTOS-plus-TCP network stack for mm-ekh18-wb55 out of the box.
- Added support for MM8108 to the mm-iot-sdk release package and deprecated the mm-iot-sdk-mm8108 release package.
- Reduced RAM usage by 32 kB for MM8108 platforms.
- Added support for TLS 1.3 to MMAGIC.

- Added a new event to the MMAGIC TCP module that allows a Controller to receive notifications when data is ready to be read for a given TCP socket. This may be enabled on a per-socket basis. Only supported for the LWIP network stack.
- Added configurable logging API to mmlog.h.
- Added support for datapath_driver_rx_alloc_failures and datapath_driver_rx_read_failures UMAC stats on MM8108.
- Reordered contents of mmwlan.h to improve structure and clarity.
- Added support for libmorse_nosupplicant.a on an ESP32 by mangling libmorse and hostap symbols post-compilation and linking of the two components. This adds a new Kconfig option, BUILD_SUPPLICANT_FROM_SOURCE, that lets the user select the libmorse variant.
- Added hostap source files to the esp32 package and hostap as a valid ESP-IDF component. For brevity, the build configuration is statically defined to the set required to support AP mode when this component is used.
- Added a library mangler script to enable users to compile Morse Micro's hostap from source, useful for systems that already include hostap for another radio, e.g., ESP32.
- Added packet memory reservation for commands when using mmpktmem_heap. This improves command reliability on MM8108.
- Renamed AP mode API from softap to ap, and example application from softap to ap_mode.
- Improved handling of packet memory exhaustion on MM8108.
- HW Version string is now printed at boot for most example applications.
- Added noise measurements to scan results to provide more in-depth information on scanned channels. Provided WPA Supplicant with a measured noise value to improve the accuracy of SNR estimation.
- Enabled hardware acceleration of ECP cryptography operations for EKH05 platforms by default.
- Enabled building of Supplicant cryptographic code from source by default in example applications. The BUILD_SUPPLICANT_CRYPTO_FROM_SOURCE make variable is now set to “y” by default in the example Makefile, but can be set to an empty string to return to the previous behavior. This change may result in longer connection times on platforms where hardware acceleration for ECC cryptographic operations is not supported (particularly those with slower processors).
- Made IPAL implementations more permissive to config get/set operations when a given IP version is not enabled at compile time by the network stack configuration.
- The MMWLAN API specifies that it must only be invoked when MMWLAN is inactive (i.e., STA mode is not enabled) and will now return `MMWLAN_UNAVAILABLE` if you call it when the STA is enabled.
- Increased the maximum number of stations that can connect in AP mode from 4 to 20, and added make variables to simplify enabling AP mode and allocating sufficient memory for the expected station count.
- Python version 3.10 has been specified in the Pipfile as a requirement.
- Split up mmhal.h API into multiple files (mmhal_xxx.h) to better represent where each function is used.

- Revised Getting Started guide.
- Extended MMWLAN API to allow specification of the interface for transmitted and received packets, link status callback, and when retrieving the MAC address.
- Added MMWLAN_NOT_INITIALIZED error code to signal when the MMWLAN API is invoked before the subsystem is initialized.
- Removed Emmet dependency on MMHAL API. Applications that use set_button or set_led Emmet commands must now implement the corresponding emmet_hal functions.
- Added BCF for MM6108-MF16858 module.
- Added API to set the time after network activity before the STA will notify the AP that it will go to sleep using a QoS Null frame, and when the host will allow the Morse chip to sleep.
- Increased the default TCP client socket timeout to 10 seconds for both IP stacks.
- Added MMWLAN API to enable/disable non-TIM mode.
- Improved detection of connection loss. Stations will now disconnect on a beacon loss event after sending 2 QoS null packets to check whether the AP is reachable. Previously, 20 unacknowledged packets would also cause disconnects, and the station frequently sent QoS nulls to verify the AP's presence.
- Added support for power-saving stations in AP mode.
- Add releasing group-addressed traffic on the DTIM beacon in AP mode.
- Added support for handling Extended Channel Switch Announcements (ECSAs) in STA mode.
- Added MMWLAN API for notification and retrieval of STA status in AP mode.
- Added support for DTIM periods greater than 1 in AP mode.
- Moved offload_arp_response and offload_arp_refresh_s from mmconfig WLAN namespace to IP namespace.
- Replaced malloc/free usage in crypto_mbedtls_mm.c with mmosal_malloc/mmosal_free.
- Added support for dynamic wake. This means that instead of waiting a fixed duration (10 ms) after the WAKE pin is raised, the Morse chip will now indicate readiness. This may result in lower power consumption and potentially less delay during transmission.
- Reformatted source files in morselib/include files to ensure consistency. This will result in whitespace differences when comparing the old and new header files.
- Added API to configure the Control Response preamble bandwidth.
- FW_MBIN is now specified in the platform-specific Makefile.
- Improved robustness of the 4-way handshake in STA mode when connecting to an AP.
- Improved roaming behavior when background scan is enabled in STA mode.
- Added on-demand timer support to lwIP. This uses the RTOS timer to schedule events, allowing the host to sleep longer while the network stack is idle, reducing power consumption. Disabled by default. Set ON_DEMAND_TIMERS_ENABLED=1 to enable.

2.5.3 Bug Fixes

- Added missing stdint.h include to mmagic_controller.h.
- Fixed a bug where supplicant could get stuck in scanning state forever, unable to start a new scan with the error `Reject scan trigger since one is already pending`.
- Fixed an assertion that could occur on reception of a fragmented 4 address frame.
- Fixed a bug that prevented 2 and 4 MHz channels in the JP regulatory domain from being supported in AP mode.
- Fixed CFLAGS for ESP32C6 platform.
- Increased size of FLASH segment and decreased size of FILESYSTEM segment (currently unused) in m2m_agent application for mm-ekh08-u575 platform.
- Fixed a bug where stations would continuously send QOS_NULL frames every 100ms.
- Added missing implementation of mmwlan_register_tx_flow_control_cb().
- mm-ekh18-wb55 platform option is now present for the `program_configstore.py` script.
- Fixed a discrepancy between LWIP and MMOSAL timeout expectations in the LWIP port implementation.
- Fixed intermittent connection failure when using public key acceleration on STM32U585 platforms.
- Updated documentation of mmwlan_get_bssid() to match behavior.
- Updated the 4 address mode filter to avoid incorrectly dropping certain non-data frames.
- Fixed rate control drops from duty cycle exhaustion.
- Fixed a bug where the timeout argument to the wlan-connect CLI command was ignored, and corrected the argument parsing order for the same command.
- Fixed Random Number Generation (RNG) failure on concurrent access for several STM32-based platforms.
- Fixed a bug that caused received Action frames to be dropped when security mode was Open (i.e., no security). This would result in lower-than-expected throughput, since a Block Ack session could not be negotiated. This does not affect OWE or SAE operation.
- Fixed a bug where group-addressed traffic would not be encrypted when transmitted in AP mode.
- Removed semicolon from configBREAK macro in FreeRTOSConfig.h.
- Fixed a bug that could result in to-chip yaps delimiter CRC fail error messages on MM8108.
- Fixed a crash on mmwlan shutdown with DHCP offload enabled.
- Fixed Command error on boot when DHCP offload is enabled.
- Fixed a bug in the MQTT agent task that could result in a deadlock when resubscribing to MQTT subscriptions after a disconnect.
- Fixed MQTTAgent task stack overflow on TLS 1.3.
- Fixed the enum values in MMAGIC CLI documentation.
- Fixed a bug where MMAGIC Agent would assert if an out-of-range stream_id was given when closing a socket.

2.5.4 Known Issues

2.5.4.1 Makefile Infrastructure

- Building fails when the MM IoT SDK package directory has a space in the path
    - The MM IoT SDK package must be unpacked into a directory without spaces in its name.

2.5.4.2 RTOS

- Timeouts with FreeRTOS time slicing disabled
    - FreeRTOS time slicing is enabled by default in the MM-IOT-SDK FreeRTOS configuration. If this is disabled, timeouts may be observed when sending commands to the MM chip.
- System time drifts when the host is in deep sleep
    - System time may drift from wall clock time when deep sleep is enabled. This may result, for example, in iperf taking longer than expected to complete. A 1-2% drift has been observed on STM32U575 and 3-4% on STM32WB55, but this may vary with activity levels.

2.5.4.3 Example Applications

- Failures when lwIP is built with LWIP_IPV4=0
    - If IPv4 is disabled when LWIP is built (i.e., IPv6-only), the IP stack may not work as expected. The workaround is to always enable IPv4.
- The CLI application requires the country code to be programmed into the config store
    - The CLI application will assert during initialization if a valid country code has not been programmed to the config store. Instructions for programming the config store are in the Getting Started Guide.
- Bootloader/OTAU support
    - The example bootloader and over-the-air-update (OTAU) support in the aws_iot example have not been tested to a level that makes them recommended for production use. The example bootloader may be deprecated in a future release.

2.5.4.4 Standby/Offload Features

- The Standby and Offload features of the MMWLAN API are deprecated and will be removed in a future release.

2.5.4.5 AP

Note that AP mode is beta and not for production use.

- Traffic is not forwarded between STAs
  - AP mode does not forward frames between connected STAs. Therefore, connected STAs cannot communicate with each other unless bridging is implemented at the application layer.
- Recovery from health failures is not supported in AP mode
  - If a health check failure occurs while operating in AP mode, it will trigger a system restart.
  - Mitigation is to disable the health check by invoking:
    `mmwlan_set_health_check_interval(0, 0);`
- OWE security is not supported
  - Mitigation is to use Open or SAE security.

2.5.4.6 DPP

Note that DPP is beta and not for production use.

- DPP must be stopped explicitly before shutdown
  - If DPP has been started (via `mmwlan_dpp_start()`), it must be stopped explicitly via `mmwlan_dpp_stop()` prior to invocation of `mmwlan_shutdown()`.
- DPP is not currently supported in regulatory domains that contain multiple channels sharing the same center frequency but at different bandwidths, in particular JP and AU REVmf.

2.5.4.7 Stability

- Some link instability may be seen at some specific signal levels, resulting in an elevated packet error rate.
  - Minor changes to the distance from the AP, antenna orientation, or environment will correct it.

2.5.5 Migration from 2.9.7

The following is a list of significant changes from the previous MM-IoT-SDK release. This is not an exhaustive list and does not include new feature additions or changes to example applications.

2.5.5.1 Combined MM6108 and MM8108 libraries

MM6108 and MM8108 devices are now both supported by a single library. The MMWLAN HAL implementation is required to provide the following function to differentiate chip type:

```
const struct mmhal_chip *mmhal_get_chip(void)
```

```
{
   /* This is a define that is set by the build system in the platform-xxx.mk file to select
the
    * Morse chip type. See the API documentation for mmhal_get_chip() for more information. */
   return &MMHAL_CHIP_TYPE;
}
```

Where MMHAL_CHIP_TYPE is defined to either `mmhal_mm6108` or `mmhal_mm6108`.

This function has been added to the platform HALs provided in the SDK.

2.5.5.2 MMWLAN API

AP mode API has been renamed. Formerly, this was referred to as *Soft AP mode*, with an `mmwlan_softap_` prefix in MMWLAN API. This is now referred to as *AP mode*, and the prefix has changed to `mmwlan_ap_`. Similarly, the *softap* example application has been renamed to *ap_mode*. Similarly, the name of the Makefile variable to enable AP mode has changed from `BUILD_SUPPLICANT_WITH_SOFTAP` to `BUILD_SUPPLICANT_WITH_AP`.

2.5.5.3 MMHAL API

The MMHAL header has been split up into multiple headers to better clarify usage.

mmhal_core.h	Required for core morselib functionality.
mmhal_wlan.h	Required for morselib MMWLAN functionality.
mmhal_os.h	Required for the MMOSAL provided with this SDK (mmosal_shim_freertos.c).
mmhal_app.h	Required for the example applications provided with this SDK.
mmhal_uart.h	Required for certain example applications that need UART I/O.
mmhal_flash.h	Required for MMCONFIG, littlefs HALs, and certain example applications.

The header mmhal.h is still included in the package, but now simply includes the above headers.

⚠ Note: `mmhal.h` is no longer included by `mmosal.h`. Files that previously relied on this implicit inclusion may need to be updated to explicitly include `mmhal.h`.

The mmhal.c source file provided in the example applications has been split into multiple files corresponding to `mmhal_xx.h` headers, and `wlan_hal.c` has been renamed to `mmhal_wlan.c` for consistency.

2.5.5.4 Cryptography

Example application Makefiles now enable `BUILD_SUPPLICANT_CRYPTO_FROM_SOURCE` by default. This can be overridden by setting `BUILD_SUPPLICANT_CRYPTO_FROM_SOURCE` to an empty string. The impact of this change is that morselib will use mbedTLS for cryptographic operations (including Elliptic Curve and hash operations). Platforms without hardware acceleration for these operations may observe an increase in connection time.

Hardware acceleration of some mbedTLS Elliptic Curve operations has been enabled for EKH05 reference platforms provided in this SDK.

2.5.5.5 Makefile and infrastructure

The firmware and BCF files have moved from the `framework/morsefirmware` directory into subdirectories appropriate to the chip (`framework/morsefirmware/mm6108`, `framework/morsefirmware/mm8108`).

2.5.5.6 Emmet

An additional hardware abstraction API has been added to emmet.h. These functions must be implemented by any application that uses Emmet.

2.5.5.7 Regulatory Database

The regulatory database has been moved from `mm_app_regdb.*` into a separate component called `mmregdb`. To receive regulatory database updates, existing applications should be updated to use this new component. See the provided example applications for reference.

2.5.5.8 Notable new API

A configurable logging API has been added to `framework/morselib/include/mmlog.h`.

A new version header, `framework/morselib/include/mmversion.h`, has been added.

2.5.6 Deprecation Notice

The following components of the MM-IoT-SDK have been marked as deprecated and will be removed in subsequent releases.

2.5.6.1 FreeRTOS-plus-TCP network stack

FreeRTOS-plus-TCP will be removed from future MM-IoT-SDK release packages. LWIP will continue to be the primary supported network stack.

2.5.6.2 Platform IO

Platform IO support will be removed from the core MM-IoT-SDK release package in the future. GNU Make will continue to be the primary supported build system supported by the core MM-IoT-SDK. Alternative build systems include:

- ARM CMSIS IDEs such as STM Cube IDE: https://github.com/MorseMicro/mm-iot-cmsis
- ESP32 IDF: https://github.com/MorseMicro/mm-iot-esp32
- Zephyr: https://github.com/MorseMicro/mm-iot-zephyr

2.5.6.3 Bootloader example application

The Bootloader example application provided with the MM-IoT-SDK will be removed in a future release. Alternatives include bootloaders and MCUBoot supplied by the MCU vendors.

2.5.6.4 Offload functionality

The existing support for the MM chips' offload functionality (including “Standby mode”) will be removed in a future release. Please see the API documentation for more specific deprecation notices.

2.5.7 Components

Component	Version
Morse firmware	1.17.6
Regulatory Database	2.7.0
FreeRTOS kernel	10.5.1
LWIP	2.2.0 (+modifications)
FreeRTOS + TCP	4.3.1
mbedTLS	3.6.2
STM32U5 BSP	1.4.0
STM32WB55 BSP	1.14.1
BCF API	12.1.0 (backward compatible to 8.0.0, but features may be limited)

2.6 Release 2.9.7

This section outlines significant changes from release 2.8.2.

2.6.1 New Features

- Soft AP functionality (beta - not for production)
    - Support for operation in AP mode with a limited feature set:
        - Maximum 4 connected STAs
        - Automatic and Dynamic channel selection is not supported
        - Soft AP example application only supports IPv4 and static IP addresses
        - See also the Soft AP subsection under Known Issues below
- Device Provisioning Protocol (DPP), push button mode functionality as STA (Beta)
    - DPP push button allows a device to be provisioned onto a network by pressing a button on both the STA and on the AP
    - Only supported on EKH05 platforms, as Public Key Accelerator (PKA) HW is required to meet timing requirements
- Support for 4-address frames in STA mode
    - Allows a STA to act as a layer 2 bridge
- Support for duty cycle burst mode
    - In regulatory domains subject to duty cycle limitations, burst mode allows a device to consume its allowance in a burst rather than spread evenly over the duty cycle period

2.6.2 Other Improvements

- Improved documentation of mmconfig parameters
- Added support for home channel dwelling during background scan
- Added MMWLAN API to configure listen interval
- Added support for registering a callback on station events
- Added support for appending user provided IEs to association IEs
- Updated mbedTLS to latest release v3.6.2
- Added API to query preserved failure logs
- Improved reliability of the WLAN chip health check system
- Added support for MCS8/9 rate control override for MM8108.
- Remove unsupported software scan
- Reduced power consumption of unused sensors on EKH05
- Added logging of return address to Hard Fault handlers, which can aid in identifying the instruction that triggered the fault.
- Improved EAPOL frame transmission reliability
- Enforced TWT configuration to occur pre-association
- Removed package files for unsupported MCU STM32-F429
- Extracted mmhal_flash API from mmhal.c

- Updated EKH05 CubeMX code generation to v6.14.1
- Added additional morselib.a builds to mm-iot-sdk-mm8108 so that it supports the same architectures as mm-iot-sdk.
- Added support for debug-only assertions using the macro MMOSAL_DEV_ASSERT()
- Removed support for MMAGIC CLI on EKH08-WB55 due to insufficient FLASH space
- Disabled LWIP address collision detection by default to improve DHCP link bring-up time
- Enabled DHCP by default on example applications and assumed target IP address is gateway address, where applicable (e.g., ping, iperf examples).
- Extended mmconfig configuration options for the AWS IoT example application
- Removed LED validation from porting assistant
- Updated ESP32 porting assistant to match main version
- Added support for debug builds in SDK Makefiles
- Added support for configuring the scan interval when running ACE script wlan-sta-connect.py
- Provided MM Chip MAC address to MMHAL read MAC address API
- Added scan dwell time configuration to mmconfig
- Added support for transceiver snooze in between scan attempts
- Added support for snooze before association to reduce power consumption
- Added support for disabling debugging in low power modes when using OpenOCD to improve power consumption profiling
- Removed unneeded SWO initialisation from main.c. OpenOCD initialises the SWO when required.
- Added PKA hardware acceleration support for EKH05 in mbedTLS
- Deprecated MMOSAL notification API
- Added a MMAGIC LLC synchronization mechanism to enable a controller to reattach to a running M2M agent
- Added MMAGIC CLI command to query WLAN link status
- Added TLS support in MMAGIC TCP core
- Added support for NTP in MMAGIC m2m
- Added support for a configurable timeout on MMAGIC m2m command responses
- Restructured autogenerated MMAGIC source code files to remove .def files and replace with .c files. In doing this several existing .c files were also renamed for clarity. This change should be transparent to users of the autogenerated makefile fragments.
- Added support for statically allocating MMPKT memory for m2m_agent demo application
- Reduced maximum mmagic streams to a sensible limit and allocated additional RAM to m2m_agent application to allow allocating the maximum streams.
- Added support for connecting to AP with higher operating bandwidth
- Reduced default scan dwell duration
- Added GB country code to regulatory domain database
- Added support for EU 2MHz channels and 100% Duty cycle
- Enabled subband transmission by default
- Added support for customer provided BCFs

- Added initial support for ESP32C6 platform
- Added a Kconfig for specifying the Morse chip firmware binary when building for ESP32
- Improved SDK documentation viewability for ESP-IDF version on Github
- Improved display of scan results in MMAGIC CLI
- Added support for clearing MMAGIC CLI variables
- Disabled OTA update support in aws_iot example application by default
- Reduced static RAM usage by 32KB when using MM8108 devices
- Added ESP32C3 support to mm-iot-sdk-esp32

2.6.3 Bug Fixes

- Fixed `mmhal_get_time()` API missing explicit void argument
- Fixed reporting TWT Requester S1G capabilities as "Not Supported"
- Ensure MAC address persists over the app lifetime
- Fixed RAW priority 0 being treated as disabled
- Reduced power consumption on MM8108-EKH05 when HaLow chip is off
- Improved robustness of porting assistant BUSY pin testing
- Fixed MMAGIC CLI unable to set full IPV6 address
- Fixed memory leak on failed TX in MMAGIC Agent LLC.
- Fixed potential memory leak in m2m_agent application for SPI datalink buffer TX.
- Fixed MMAGIC CLI becoming unresponsive after executing one-shot system deep sleep command `sys-deep_sleep one_shot`
- Fixed m2m_agent SPI data link recovery from receive error.
- Fixed ignored MMAGIC wlan core configuration parameters
- Fixed MMAGIC CLI maximum wlan password length not matching MMWLAN maximum
- Improved MM8108 TCP throughput stability in environments experiencing packet loss

2.6.4 Known Issues

2.6.4.1 Makefile Infrastructure

- Building fails when MM IoT SDK package directory has a space in the path
    - The MM IoT SDK package must be unpacked into a directory that does not include a space in its name.

2.6.4.2 Stability

- Timeouts with FreeRTOS time slicing disabled
    - FreeRTOS time slicing is enabled by default in the MM-IOT-SDK FreeRTOS configuration. If this is disabled, timeouts may be observed when sending commands to the MM chip.
- System time drifts when host deep sleep is enabled

    - System time may drift from wall clock time when deep sleep is enabled. This may result, for example, in iperf taking longer than expected to complete. Drift of 1-2% has been observed on STM32U575 and 3-4% on STM32WB55, but this may vary with level of activity.
- DPP must be stopped explicitly before shutdown
    - If DPP has been started (via `mmwlan_dpp_start()`) it must be stopped explicitly via `mmwlan_dpp_stop()` prior to invocation of `mmwlan_shutdown()`.
- Assert after consecutive RNG failures
    - On STM32 platforms, in cases where multiple threads access the hardware random number generator (RNG), a priority inversion may occur resulting in an assertion in `mmhal_random_u32()`. This can be mitigated by protecting access to the STM32 RNG API with a mutex.

2.6.4.3 Throughput

- Low throughput with open security due to action frames being dropped
    - When connected with open security (no encryption), action frames are dropped on receive. The primary impact of this is that block ack negotiation fails, meaning A-MPDU aggregation is not enabled, and thus resulting in lower throughput. This does not affect SAE or OWE security modes.

2.6.4.4 Example Applications

- Failures when lwIP built with LWIP_IPV4=0
    - If IPv4 is disabled when LWIP is built (i.e., IPv6 only) then there may be issues with the IP stack not working as expected. The workaround is to always enable IPv4.
- CLI application requires country code to be programmed to config store
    - The CLI application will assert during initialization if a valid country code has not been programmed to the config store. Instructions for programming config store can be found in the Getting Started Guide.
- Bootloader/OTAU support
    - The example bootloader and over-the-air-update (OTAU) support in the aws_iot example has not been tested to a level where it is recommended for production use. The example bootloader may be deprecated in a future release.

2.6.4.5 Standby/Offload Features

- The Standby and Offload features of the MMWLAN API are in development and may not be fully functional.
    - Further improvements are planned in future releases that may result in minor API changes.

2.6.4.6 Soft AP

- Power save is not supported on connected STAs
  - Soft AP does not support buffering of traffic for power saving STAs. Therefore, 802.11 power save must be disabled on STAs connecting to the AP.
- Traffic is not forwarded between STAs
  - Soft AP does not forward frames between connected STAs. Therefore it is not possible for connected STAs to communicate with each other.
- Soft AP does not correctly transmit group addressed frames
  - Group addressed frames are transmitted unencrypted, even when encryption is enabled, meaning that they are not able to be decoded by receiving STAs.
- Recovery from health failures is not supported in Soft AP mode
  - If a health check failure occurs while operating in Soft AP mode, it will not return to an operable state (for example, beacon transmission will not resume).

2.6.5 Migration from 2.8.2

The following is a list of significant changes from the previous release of the MM-IoT-SDK. This is not an exhaustive list and does not include new feature additions or changes to example applications.

2.6.5.1 mbedTLS configuration

For mbedTLS configuration files that use `mmhal_get_time()` for `MBEDTLS_PLATFORM_TIME_MACRO`, the definition must be updated to the following to avoid compilation errors:

```
#define MBEDTLS_PLATFORM_TIME_MACRO(x)          mmhal_get_time()
```

2.6.5.2 MMOSAL API

Newly added functions:

- `mmosal_printf()`: OS abstracted version of printf used by morselib
- `mmosal_extract_failure_info()`: Extract the oldest un-viewed failure log entry

The task notification API (`mmosal_task_wait_for_notification()`, `mmosal_task_notify()`, `mmosal_task_notify_from_isr()`) has been deprecated. This API is no longer used by morselib, and it may be removed from the MMOSAL API in the future.

The macro `MMOSAL_DEPRECATED_API_ENABLED` has been added to `mmosal.h`. If set to `0`, then the deprecated MMOSAL API will be excluded by the preprocessor. The default value is `1` to allow for backward compatibility, but it can be overridden to 0 for increased strictness.

2.6.5.3 MMWLAN API

Newly added API:

- `morse_chip_id_string` field added to `mmwlan_version` struct.
- `home_channel_dwell_time_ms` field added to `mmwlan_scan_config` struct.
- Duty cycle API (`mmwlan_set_duty_cycle_mode()` and `mmwlan_get_duty_cycle_stats()`) added.
- STA event callback API added (`sta_evt_cb` and `sta_evt_cb_arg` fields added to `mmwlan_sta_args` struct). *(Beta)*
- `extra_assoc_ies` and `extra_assoc_ies_len` fields added to `mmwlan_sta_args` struct.
- `use_4addr` field added `mmwlan_sta_args` struct.
- DPP API added (`mmwlan_dpp_start()` and `mmwlan_dpp_stop()`). *(Beta)*
- `dwell_on_home_ms` field added to `mmwlan_scan_args` struct.
- Soft AP API added (`mmwlan_softap_enable()` and `mmwlan_softap_disable()`). *(Beta)*

API changes:

- `MMWLAN_SCAN_DEFAULT_DWELL_TIME_MS` was reduced from 105 to 30 due to improvements in scan implementation.

2.6.5.4 MMHAL API

The behavior regarding the invocation of the function `mmhal_read_mac_addr()` has changed. This function will now only be invoked once after boot of the transceiver (e.g., via `mmwlan_boot()`) and the value will be cached, where previously it may have been invoked multiple times.

2.6.5.5 MMAGIC

Restructured auto-generated MMAGIC source code files to remove .def files and replace them with .c files. In doing this, several existing .c files were also renamed for clarity. This change should be transparent to users of the auto-generated makefile fragments.

Low-level MMAGIC Controller API added: `mmagic_controller_agent_sync()` and `mmagic_controller_request_agent_reset()`.

New API: *TLS*, *MQTT*, and *NTP*. Currently supported in M2M mode only.

Due to FLASH constraints, support for the MMAGIC CLI has been removed on the mm-ekh08-wb55 platform.

2.6.5.6 MQTT agent

MQTT agent has been moved out of the `aws-common` component and into its own `freertos-libs` component. This enables using the MQTT agent without introducing a dependency on AWS, such as in the MMAGIC MQTT module.

Existing Makefile-based applications using MQTT via AWS will need to include an additional Makefile fragment in their root Makefile::

```
include $(MMIOT_ROOT)/mk/mqtt-agent-task.mk
```

This change should be transparent to users of the auto-generated makefile fragments.

2.6.5.7 Makefile infrastructure

Setting of Makefile CFLAGS for optimization has been moved from misc.mk to Makefile for each example. Projects relying on this flag being set in misc.mk will need to be updated to set the flag in their Makefile.

2.6.5.8 Morse Firmware/BCFs

⚠ The firmware version has been updated from 1.15.3 to 1.16.4. Devices may need to have their BCF files updated to the latest version if they want the latest features on MM6108/MM8108.

2.6.6 Components

Component	Version
Morse firmware	1.16.4
Regulatory Database	2.4.1
FreeRTOS kernel	10.5.1
LWIP	2.2.0
FreeRTOS + TCP	4.3.1
mbedTLS	3.6.2
STM32U5 BSP	1.4.0
STM32WB55 BSP	1.14.1
BCF API	12.1.0 (backwards compatible to 8.0.0, but features may be limited)

3 System Details

3.1 System Memory Requirements

Supported architectures*:

- ARM Cortex-M4F
- ARM Cortex-M7F
- ARM Cortex-M33F

*Users can compile morselib from source to integrate with other architectures.

Memory	Recommended
RAM	256 KB+
Flash	2 MB+

Space required will depend on the specific application. This may not be sufficient to support, for example, cloud connectivity, where application-layer protocols such as TLS and MQTT are required.

3.2 HaLow Throughput Performance

Throughput is measured at 8 MHz, with RTS disabled (unless otherwise stated), SGI enabled, power-save enabled, using Lightweight IP (LWIP), and with iperf run for 60 seconds, taking the median of 5 iterations.

- MM-IoT-SDK 2.14 STA
- Linux AP with 2.1 drivers/firmware.

Test Type	Platform	UDP (Mbps)		TCP (Mbps)	
		STA -> AP	AP -> STA	STA -> AP	AP -> STA
Single STA	MM6108 EKH08-WB55	5.6	10	4.6	5.8
	MM6108 EKH05 SDIO	22	22	7.0	9.4
	MM6108 EKH05 SPI	17	18	6.4	8.2
	MM8108 EKH05 SDIO	24	30	8.5	10
	MM8108 EKH05 SPI	19	19	7.7	9.7
Multi STA (4 STA cumulative)	MM6108 EKH08-U575 SPI	20	26	10	14
	MM8108 EKH05 SPI	25	35	11	17

4 Revision History

Version Number	Release Date	Release Notes
1	01 Oct 2025	Initial Release
2	03 Mar 2026	Added 2.10.4
3	01 May 2026	Added 2.11.2
4	01 Jul 2026	Added 2.12.3
5	02 Sep 2026	Added 2.13.1
6	26 Oct 2026	Added 2.14.3

Morse Micro provides this information "as is" without warranties of any kind, express or implied. No guarantee is made as to the accuracy, completeness, or suitability of this information or Morse Micro's products for any specific purpose. Use of this information and products is at the user's sole risk. Morse Micro products are not designed or tested for use in mission-critical systems, and should not be used in such applications. Performance specifications are based on internal testing and are believed to be reliable; however, they are not guaranteed. It is the Buyer's responsibility to test and validate all product performance, compatibility, and compliance, both in isolation and within end applications. Morse Micro assumes no liability for the use or application of any product, circuit, or information described herein. No license or other rights—express or implied—are granted under Morse Micro's intellectual property. This document contains proprietary information of Morse Micro and is subject to change without notice. Wi-Fi®, Wi-Fi HaLow™, and the Wi-Fi logo are trademarks of Wi-Fi Alliance. ZigBee™ and Z-Wave™ are trademarks of their respective owners. All other trademarks are the property of their respective owners.



Morse Micro

Morse Micro Pty. Ltd. Corporate Headquarters

Level 8, 10-14 Waterloo Street, Surry Hills, NSW 2010, AUSTRALIA

Email: sales@morsemicro.com

Copyright © Morse Micro Pty. Ltd.