PMGR Dynamic Power Management

The RA6W1 is a highly integrated, ultra-low-power Wi-Fi MCU that enables users to develop a complete Wi-Fi solution on a single chip. The RA6W2 module comprises the RA6W1 (Wi-Fi) and the DA14531 (Bluetooth® Low Energy) MCU. This document describes the power management features supported by the RA6W1 and RA6W2 and explains how the PMGR module can be used for low-power applications.

Contents

Figures

Tables

1. Terms and Definitions

AP	Access Point
BSP	Board Support Package
COM	Communication Port
DPM	Dynamic Power Management
GPIO	General Purpose Input Output
MCU	Micro-controller Unit
PMGR	Power Manager
RTC	Real Time Clock
UART	Universal Asynchronous Receiver Transceiver
WAP	Wireless Application Protocol

2. Introduction

Power Manager (PMGR) is a new generic power management FSP module designed to simplify and standardize power management across RA6W1/RA6W2/RRQ61001/RRQ61051 applications. Power management in modern embedded systems is critical for optimizing battery life and reducing energy consumption. However, different chips and modems implement power-saving features in various ways, making it difficult for developers to manage power efficiently across different solutions.

To address this challenge, the PMGR provides an abstraction layer that enables developers to define power constraints at the application level rather than dealing with complex chip-specific power modes. This ensures that power optimization is handled efficiently and consistently across Renesas' portfolio.

There are five Sleep modes supported by the RA6W1/RA6W2/RRQ61001/RRQ61051—Sleep mode 1, Sleep mode 2, Sleep mode 3, Sleep mode 3 + DPM, and Sleep mode 4. This application note focuses on developing and utilizing power-saving techniques supported by the RA6W1/RA6W2/RRQ61001/RRQ61051 using the PMGR API.

3. Sleep Modes

Currently, the RA6W1/RA6W2/RRQ61001/RRQ61051 supports five sleep modes: Sleep mode 1, Sleep mode 2, Sleep mode 3, Sleep 3 + DPM LPM, and Sleep mode 4.

3.1 Sleep Mode 1

Sleep 1 mode is a deep sleep/reset state. In this state, power is applied (VBAT), but all internal circuitry is powered off. When reset is de-asserted, the device can start immediately without waiting for power stabilization.
This mode is purely hardware-controlled and cannot be managed through the PMGR API through software.
Use case: Stays in deep sleep and wakes only when triggered by an external event, making it ideal for low-activity devices such as panic buttons or motion sensors.

3.2 Sleep Mode 2

In this state, all power domains are off except for the RTC and wake-up pins, allowing a timer or an external event to trigger the device to wake up from sleep.
Use case: Ideal for devices that wake periodically or in response to sensor input, such as temperature loggers or smart meters.

3.3 Sleep Mode 3

In this state, all power domains are off except for the wake-up pins and retention memory (RTM). The RTM preserves system information. There is no RTC in this mode.
Use case: Suited for devices that need to retain data or settings during sleep, such as smart thermostats or fitness trackers.

3.4 Sleep Mode 3 + DPM LPM

In this state, all power domains are off, except for the wake-up pins and retention memory. The retention memory retains system information, including the Wi-Fi state, enabling a fast wake-up for simple network management tasks such as DTIM processing.
Additionally, this sleep mode allows the device (Station mode) to remain in a low-power state while maintaining its connection with the Access Point, ensuring efficient power management without losing network connectivity. If required, a full system wake-up can also be triggered.
Use case: Ideal for always-connected Wi-Fi devices, such as smart locks or alarms, enabling periodic wake-ups and unicast-triggered communication while efficiently managing power.

3.5 Sleep Mode 4

In this state, all power domains are turned off except for the Real-Time Clock (RTC), wake-up pins, RAM, and LDO. Upon wake-up, the Cortex-M33 CPU resumes execution from the same task context.

- RAM remains powered (all 704 kB)
- Wake-up events: device can wake on a unicast beacon packet with fast recovery
- Application continuity: no software or application reinitialization is required.

Use case: Ideal for Wi-Fi-connected devices that pause all activity and then wake instantly to the exact screen and state present before sleep.

4. PMGR

Figure 1 shows the PMGR internal block diagram. The PMGR module consists of multiple internal blocks that work together to manage low-power modes efficiently. Applications communicate with the PMGR through the rm_pmgr_w blocks. Here rm_wifi represents the Wi-Fi application, which also operates on top of the PMGR to support DPM LPM. When FreeRTOS has no active tasks to execute, the Idle Task invokes the PMGR, which then determines the appropriate sleep mode to trigger.

The rm_pmgr_w module interacts with pm_lld, a low-level power management driver responsible for switching between different low-power modes. DPM functions are handled by dpm_lld, which checks preconditions for DPM sleep, notifies callbacks when the system is entering into low-power mode, and initiates DPM sleep by calling sleep3_lld.

The sleep3_lld (PTIM) is a lightweight firmware running in the DPM LPM state, ensuring Wi-Fi connectivity by monitoring AP frames and sending keep-alive frames. The PMGR communicates with the RTC to manage wake-up timers and with GPIO to monitor interrupts for waking up from low-power modes. Network connectivity during DPM LPM is maintained by storing essential parameters such as socket states, ARP tables, and DHCP configurations in retention memory in association with LWIP. When sleep3_lld receives a packet, it forwards it to the MAC driver (MAC DPM DRV), which then delivers the packet to the upper layers when the application is ready to receive it.

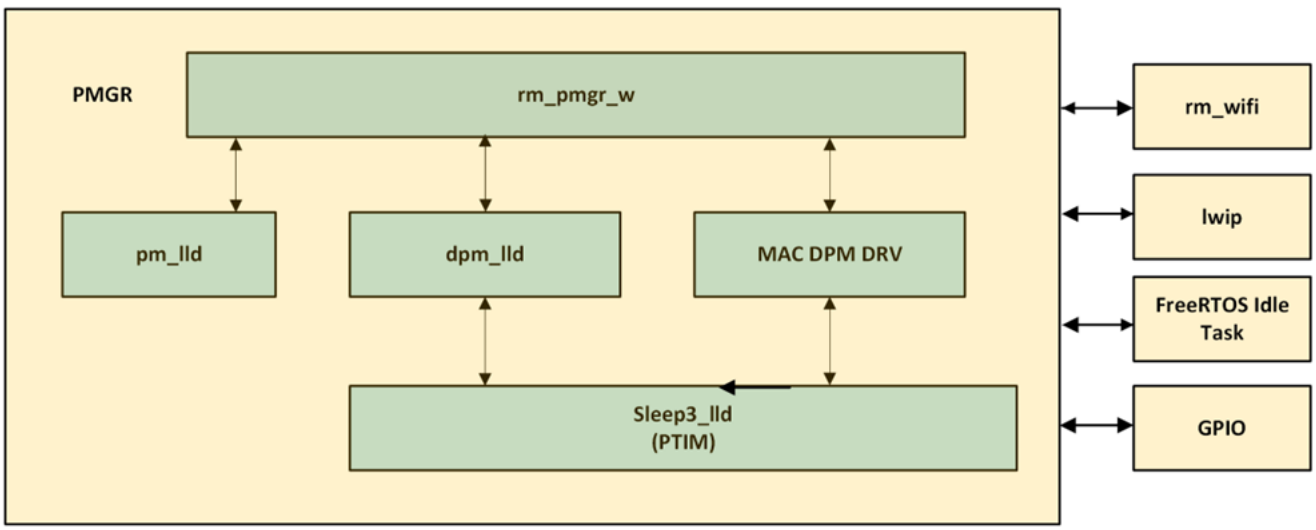

Figure 1. PMGR internal block diagram

PMGR Interaction with Application

Figure 2 shows how the PMGR interacts with applications.

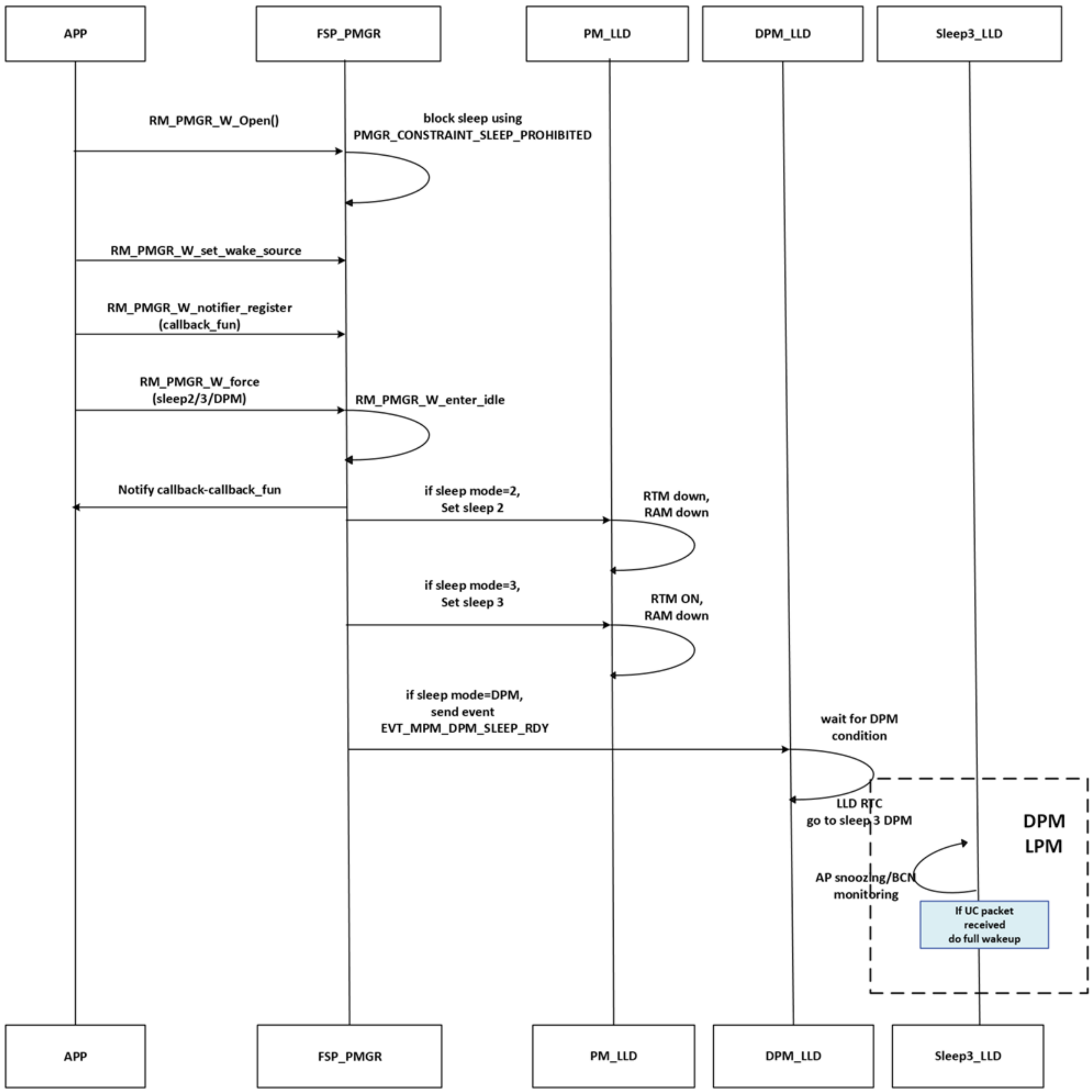

Figure 2. PMGR interaction with application

5. PMGR DPM

Dynamic Power Management (DPM) is a technology to achieve low power consumption while connecting to access point (AP) or peer for a long time. If device has no actions for sending data to peer devices or communicating with external devices, the device can keep a low power state while waiting to receive any data from peer.

5.1 DPM Modes

DPM provides two different sub-modes: RTOS Running mode and DPM Low Power Mode (LPM). RTOS Running mode allows a device to communicate over the network and with external devices, while DPM LPM enables a device to receive data only from the AP.

In DPM mode, the DPM service starts, monitors the state of applications, and manages the transition from RTOS Running mode to DPM LPM. Additionally, the DPM service controls timer functions and transfers received data to the relevant applications.

Figure 3 shows how DPM mode operates. DPM can be enabled only in Wireless Application Protocol (WAP) that is. in 802.11 Station mode and when the device is in a Wi-Fi-connected state. Therefore, provisioning is required to configure AP profiles before entering DPM LPM. Transitioning from RTOS running mode to DPM LPM may take some time due to various processing tasks, see Figure 7. For application monitoring, register the job name with the DPM service first. To register job names, use RM_PMGR_W_dpm_job_name_set(). Each time the device enters RTOS Running mode, the DPM service and applications are restarted, and applications must be re-registered with the DPM service.

The DPM service monitors sleep constraints. If an application needs to prevent sleep, it calls RM_PMGR_W_add_sleep_constraint() with the constraint PMGR_CONSTRAINT_POWER_RAM. When the application is ready for sleep, it calls RM_PMGR_W_remove_sleep_constraint() with the same constraint.

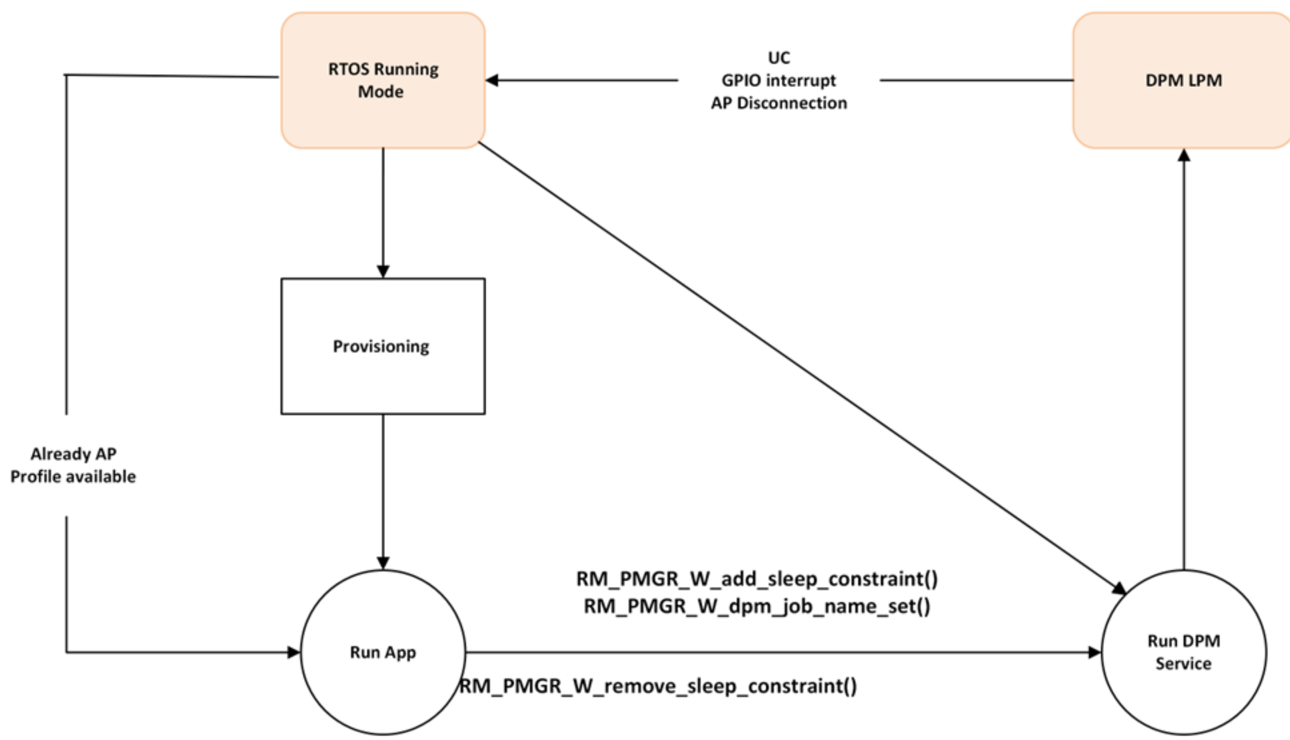

Figure 3. DPM modes

5.2 Power Saving Modes in DPM LPM

DPM mode uses 802.11 standard features to achieve low power consumption while keeping connection with the AP. The 802.11 PS features are legacy PS and TWT. Legacy PS mechanism is based on DTIM beacon interval. It listens to DTIM beacon based on DPM configuration and schedule other tasks (PS-Poll transmission) according to Beacon TIM element information. TWT mechanism is based on TWT setup to schedule AP-STA frames exchange.

The software supports both legacy PS (DTIM) and TWT features and can switch between both modes dynamically. Figure 4 shows power states in the DPM mode using Legacy PS. The device can send data to peer devices over the network and communicate with external devices like peripherals or a host device only in RTOS running mode.

When the device enters DPM LPM, a firmware for Delivery Traffic Indication Message (DTIM) runs on retention RAM (or RTM) with DTIM interval periodically. The PTIM is a tiny firmware image only for checking data from AP such as UC, BC/MC, or BCN. The device stays in Sleep mode 3 when PTIM is not active in DPM LPM.

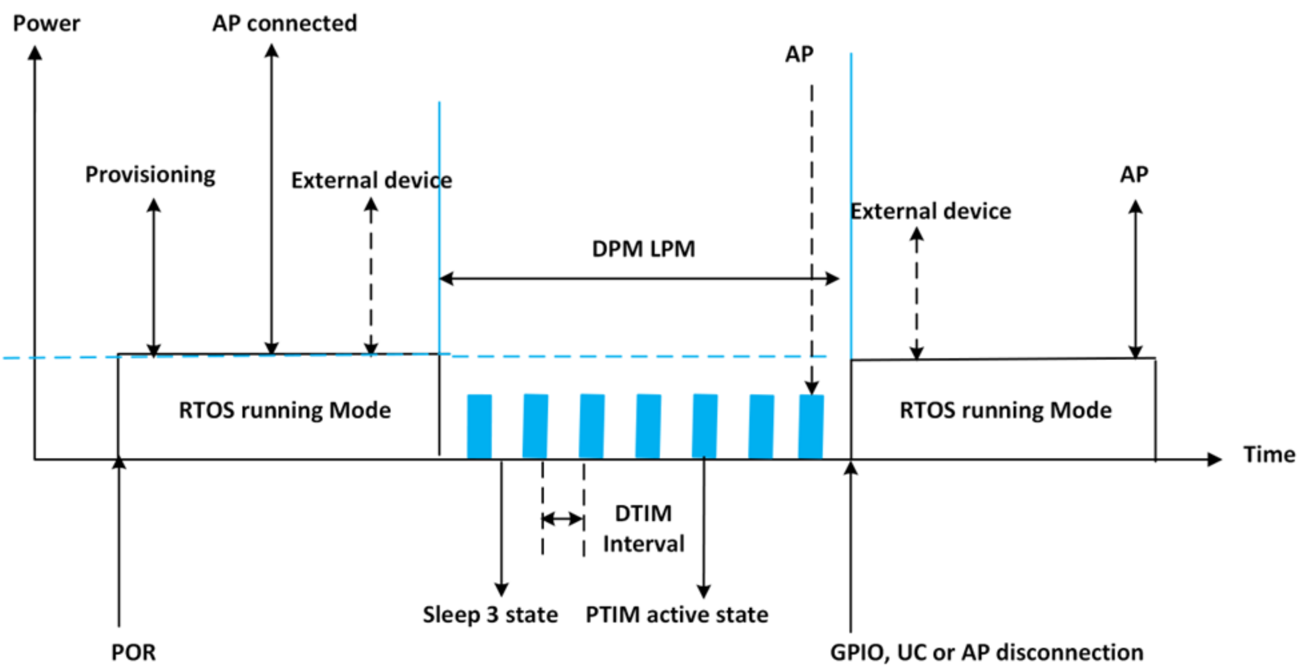

Figure 4. Power states in DPM mode with legacy DTIM

Target Wake Time (TWT) is a new power saving feature introduced in the IEEE 802.11ax standard. TWT enables STAs and AP to schedule specific times for the STA to wake up and communicate. Generally, TWT agreement is defined by:

- TWT Service period (TWT-SP) – the STA wake duration.
- TWT wake interval – the period between two consecutive TWT SP, see Figure 5.

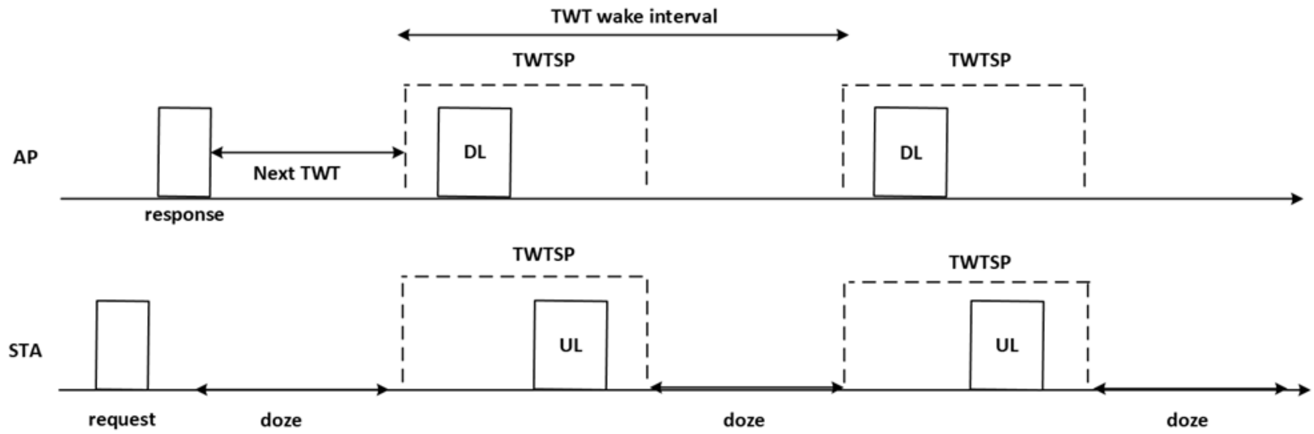

Figure 5. TWT basic flow

5.3 DPM Service and Application

When data from AP or peer device are received, a registered DPM timer is expired or GPIO interrupts occur in DPM LPM, the device wakes up from DPM LPM and DPM service runs in RTOS running mode. Figure 6 shows how the DPM service manages the received data and callback of DPM timer, and monitors applications in RTOS Running mode.

IIf the network session is connected with peer devices and there is received data from peers, the application needs to be registered with the port number within 200 ms because DPM service checks the port number where the data is received from peers and then transfers the data to the application with the same port number. Also, the application must notify that it is in ready state for receiving the data to the DPM service within 7000 ms using RM_PMGR_W_dpm_rcv_ready_set () after waking up from DPM LPM. Otherwise, DPM service drops the data because network stack cannot keep the data for a long time.

DPM timer is registered by the application and can be expired in RTOS Running mode or DPM LPM. When the timer is expired, the DPM service checks whether the application is registered and ready to get the callback. Therefore, the application must be registered at every wake-up from DPM LPM and notify the DPM service that it is ready using RM_PMGR_W_dpm_wakeup_done().

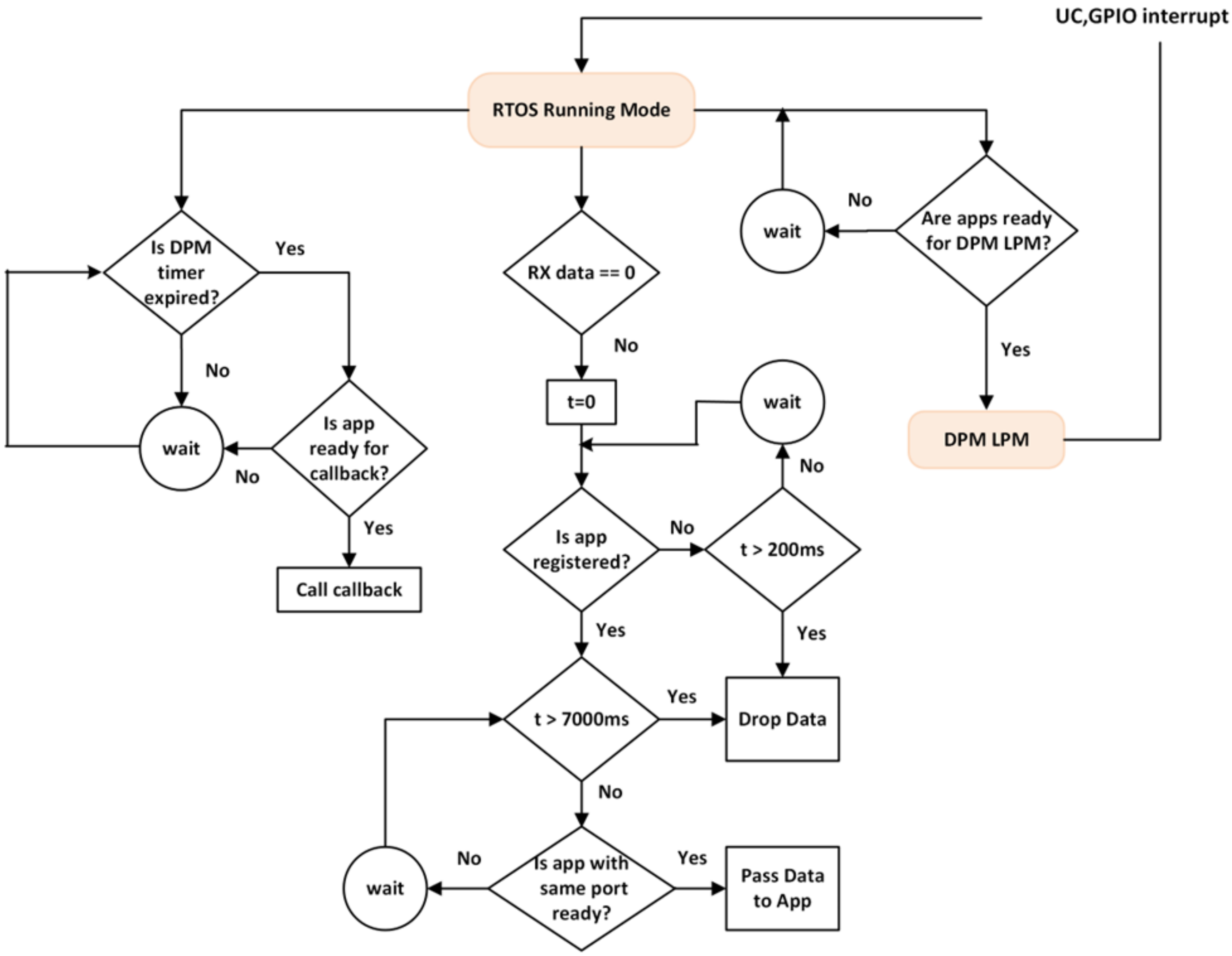

Figure 6. DPM service in RTOS running mode

Figure 7 shows how to register applications and notify the state of applications to the DPM service.

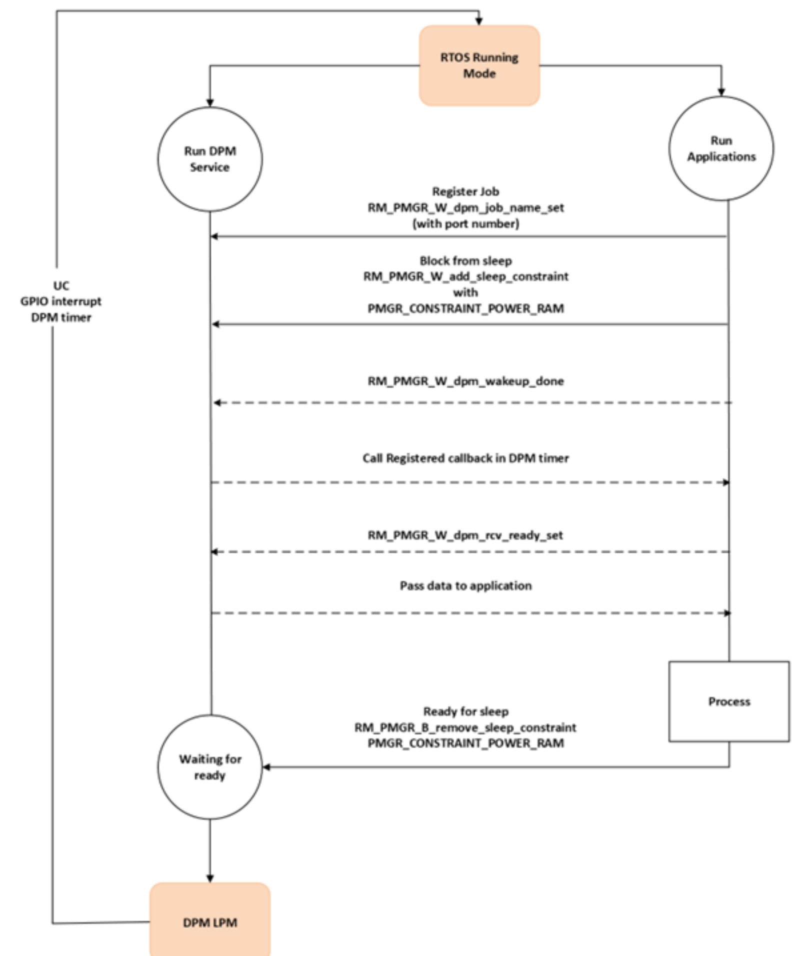

Figure 7. Application in DPM RTOS running mode

5.4 PMGR DPM and Bluetooth LE Dependency

The RA6W2 can enter DPM LPM mode only when Bluetooth LE is turned off, as Bluetooth LE is considered a power constraint. When Bluetooth LE is active, the system automatically adds the PMGR_CONSTRAINT_POWER_RAM constraint. This constraint is removed when Bluetooth LE is turned off.

The FSP API RM_BLE_ABS_Open applies a Bluetooth LE constraint, while RM_BLE_ABS_Close removes it. To allow the device to enter DPM LPM, you must either stop the Bluetooth LE service or set Bluetooth LE to Hibernate mode.

To stop the Bluetooth LE service:

1. Disconnect any active connections using R_BLE_GAP_Disconnect().
2. Stop advertising with R_BLE_GAP_StopAdv().
3. Call RM_BLE_ABS_Close() to disable Bluetooth LE and remove the constraint.

Alternatively, to place Bluetooth LE directly into Hibernate mode, use the AT command: AT+SLEEP=3.

To place Bluetooth LE into Hibernate mode from the application, set the parameter value for WIFI_PROFILE_DPM_BLE_HIBERNATE.

RM_MAP_PERSISTANT_W_Write_INT(RM_MAP_PERSISTANT_W_get_ctrl(), ENV_GROUP_WIFIPROFILE, WIFI_PROFILE_DPM_BLE_HIBERNATE, MODE_DISABLE).

6. Application Programming Interface

All PMGR APIs are documented in the RA6W1 Flexible Software Package. You can find them under **API Reference** > **Modules** > **Power** > **PMGR (rm_pmgr_w)**.

Figure 8. API reference

6.1 Add PMGR Module in FSP e² studio

To add The PMGR module to an FSP project in e² studio, go to **New Stack** > **Power** > **PMGR - Power Manager** (rm_ pmgr_w), see Figure 9.

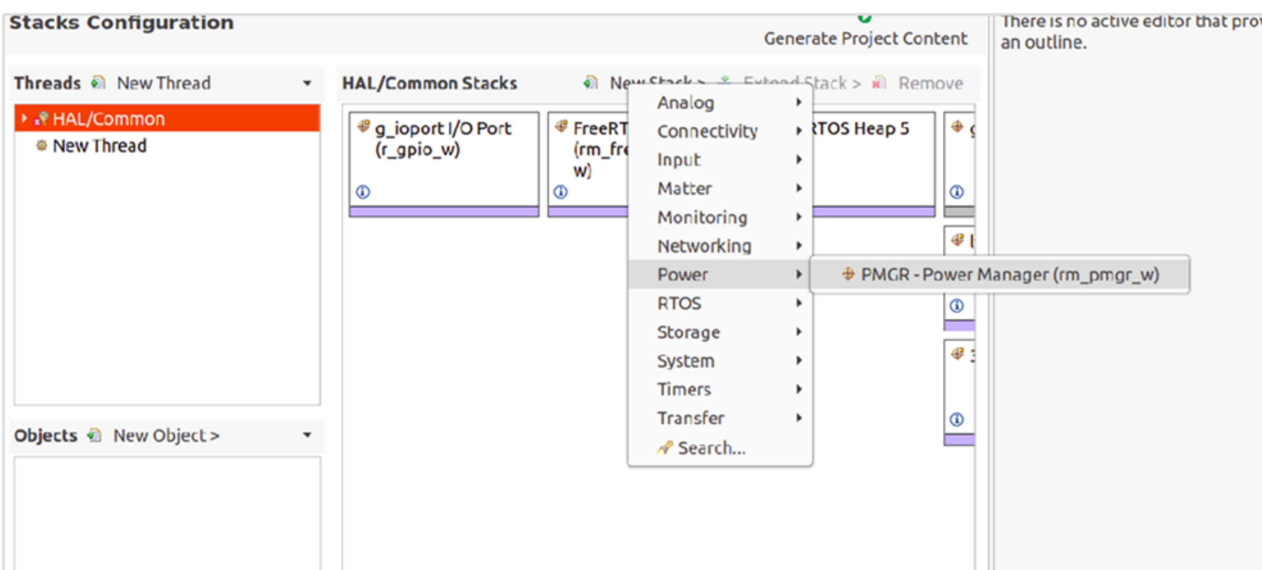

Figure 9. PMGR stack in FSP

6.2 PMGR Power Modes

The PMGR provides multiple power-saving modes to optimize energy consumption based on system requirements. These modes define how the RAM and retention memory are managed during low-power states.
Table 1 summarizes the available PMGR Power modes.

Table 1. Power modes

Power modes	RAM power	Retention memory	Behavior
PMGR_LLD_STATE2	OFF	OFF	Deep Sleep mode, no retention.
PMGR_LLD_STATE3	OFF	ON	RAM is powered down; retention memory is active.
PMGR_LLD_DPM	OFF	ON	Same as PMGR_LLD_STATE3, with AP snoozing is enabled.
PMGR_LLD_POWER_MODE_ SLEEP4	ON	ON	Same as PMGR_LLD_DPM with RAM powered on and M33 tick-less timer.
PMGR_LLD_CPU_CLOCK_ GATING	ON	N/A	The CPU clock halted, but the power is maintained.
PMGR_LLD_AUTO	Dynamic	Dynamic	The PMGR selects the optimal power mode automatically.

The PMGR uses RM_PMGR_W_force() API for forcefully setting power modes. This function overrides the decision made by PMGR, forcing usage of a given power state. Corresponding AT commands for setting Force mode is AT+PMGRFORCE, see *RA6W1 and RA6W2 AT Commands*.

6.3 PMGR Constraints

The PMGR constraints define restrictions on the system's low-power behavior. These constraints ensure that certain power states remain enabled or disabled based on application requirements. For example, an application might prohibit Sleep modes or require RAM to stay powered to prevent data loss.
The following constraints can be applied to control power states, see Table 2.

Table 2. Constraints

Constraints	Description
PMGR_CONSTRAINT_NONE	No constraints: all Sleep modes are allowed.
PMGR_CONSTRAINT_SLEEP_PROHIBITED	Prevent the system from entering any Sleep mode.
PMGR_CONSTRAINT_POWER_RAM	Ensure RAM remains powered on during low-power states.
PMGR_CONSTRAINT_POWER_RETENTION	Ensure retention memory stays powered.

Applications can apply or remove constraints dynamically using the PMGR API - RM_PMGR_W_add_sleep_ constraint() and RM_PMGR_W_remove_sleep_constraint(). Corresponding AT command for sleep constraint is AT+PMGRCONSTRAINT. See *RA6W1 and RA6W2 AT Commands* for more details.

6.4 Wake Sources

System middleware and customized applications are allowed to specify which components are used as wake source and as such should be kept powered on. Wake source is a bit map, the implementation is Boolean parameter per wake source (no matter how many times a wake source is set, single clear should set it to false).
Table 3 describes the available wake sources in the PMGR.

Table 3. Wake source

Wake source	Description
PMGR_WAKE_SOURCE_NON	No wake source defined.
PMGR_WAKE_SOURCE_GPT	GPT wake source.
PMGR_WAKE_SOURCE_GPIO	GPIO wake source.
PMGR_WAKE_SOURCE_ADC	ADC wake source.
PMGR_WAKE_SOURCE_WIFI	Wi-Fi/MAC wake source. Connectivity wake source.
PMGR_WAKE_SOURCE_BLE	Bluetooth LE wake source.

Applications can set or clear wake source using RM_PMGR_W_set_wake_source() and RM_PMGR_W_clr_wake_source(). Corresponding AT command is AT+PMGRWAKESRC.
To configure pin and port configuration for PMGR_WAKE_SOURCE_GPIO and PMGR_WAKE_SOURCE_ADC, Applications can set or clear using R_BSP_WakeupSourcePinSet and R_BSP()_WakeupSourcePinUnSet(). Corresponding AT command is AT+SETCONFIG .To get the port details use AT+GETCONFIG, see *RA6W1 and RA6W2 AT Commands*.

NOTE
set/clr wake source API in RA6W1/2 is available, but it only modifies wake_source bitmap. The side effect or expected operational behavior by setting/clearing the wake_source is not defined in RA6W1/2. The actual wake_source configuration should be done with each module's API independently. For example, use R_BSP_WakeupSourcePinSet to enable GPIO wake-up.

6.5 Notifier Register

The PMGR provides an interface for applications to register notifier callback functions using the RM_PMGR_W_notifier_register() API. This allows applications to execute specific actions when transitioning to low-power mode. Applications can define both the callback function and priority ensuring execution in a predefined order. The PMGR invokes these registered callbacks based on their assigned priority during the low-power transition.
Within the callback function, applications can perform tasks such as logging with delays or updating specific fields as needed.
By default, the maximum number of registered callbacks is defined by MAX_NOTIFIER_ARRAY (5), limiting the number of applications that can register for PMGR notifications. This value can be adjusted in the FSP PMGR properties to accommodate additional application requirements.
The PMGR notifier order can be configured as shown in Table 4.

Table 4. Notifier priority order

Notifier priority order	Description
PMGR_W_NOTIFIER_ORDER_SYS_LOW	Reserved for system low order.
PMGR_W_NOTIFIER_ORDER_CUSTOMISED_LOW	Reserved for customized application.
PMGR_W_NOTIFIER_ORDER_SYS_HIGH	Reserved for system high order.
PMGR_W_NOTIFIER_ORDER_HIEST	First executed.

NOTE
The notifier callback function is restricted to only adding debug logs or updating application fields. It cannot be used to prevent the system from entering Sleep mode.

6.6 User Data Area in Retention Memory

The PMGR provides a maximum of 10096 bytes of the user data area in RTM to store user application data. User data in RAM which is not stored in the RTM should not be retained during DPM LPM because all other areas of RAM are powered off during Sleep mode 3 of DPM LPM.

The application can use RM_PMGR_W_user_rtm_pool_alloc(), RM_PMGR_W_user_rtm_free(), and RM_PMGR_W_user_rtm_get() to allocate, free, and access user data from RTM, respectively. See APIs in *fsp_user_manual _v0.8.0* for more information.

NOTE
The RA6W1 has 64 kB of RTM memory, but it cannot be split into smaller parts. Because of this, the whole 64 kB remains powered ON in DPM LPM mode, even if only part of it is needed.

6.7 DPM Port Filtering

Port filtering allows the device to selectively wake up from DPM LPM by responding only to specific TCP/UDP port numbers or IP multicast addresses. This helps achieve low power consumption by preventing unnecessary wake-ups caused by unregistered network traffic.

The APIs listed in Table 5 enable filtering for a designated TCP/UDP port number or IP multicast address, ensuring that only relevant packets trigger a wake-up event while the device remains in low-power mode.

Table 5. Port filtering APIs

Port filtering API	Description
RM_WIFI_dpm_tcp_port_filter_set	Set TCP port number for DPM wake-up.
RM_WIFI_dpm_tcp_port_delete	Delete TCP port number.
RM_WIFI_dpm_udp_port_filter_set	Set UDP port number for DPM wake-up.
RM_WIFI_dpm_udp_port_filter_delete	Delete UDP port Port filtering allows the device to selectively wake up from DPM LPM by responding only to specific TCP/UDP port numbers or IP multicast addresses. This helps achieve low power consumption by preventing unnecessary wake-ups caused by unregistered network traffic.
RM_WIFI_dpm_ptim_mc_filter_set	Set IP multicast address to allow receiving packet in DPM sleep.
RM_WIFI_dpm_ptim_mcv6_filter_set	Set IPv6 multicast address to allow receiving packet in DPM sleep.

When waking up from LPM, the application restarts. It must call RM_PMGR_W_dpm_rcv_ready_set() to ensure that only the DPM service sends packets to the upper layer.

6.8 LPM Examples

Low-power mode is designed to reduce energy consumption during idle periods. It allows the device to shut down most functions while retaining essential operation. Low Power Mode achieves very low current usage, typically in the microamp range. This helps extend battery life in IoT devices by waking up only when needed for tasks or events.

6.8.1 Sleep Mode 2 Configuration

6.8.1.1 Using API

Figure 10 shows the Sleep mode 2 configuration process using PMGR APIs. After rebooting, the application initializes the PMGR module using RM_PMGR_W_Open(), which adds a sleep constraint for RAM. To allow Sleep mode 2, this constraint is removed using RM_PMGR_W_remove_sleep_constraint (PMGR_CONSTRAINT_POWER_RAM). The RM_PMGR_W_notifier_register() API, though optional, can be used to register a callback function for notification purposes before entering Sleep mode 2. The application then calls RM_PMGR_W_force () with a specified sleep duration (t_time) to transition into Sleep mode 2. Here the time should be provided in milliseconds. After the RTOS wakes up, the system restarts, and the application resets, requiring re-registration with the PMGR.

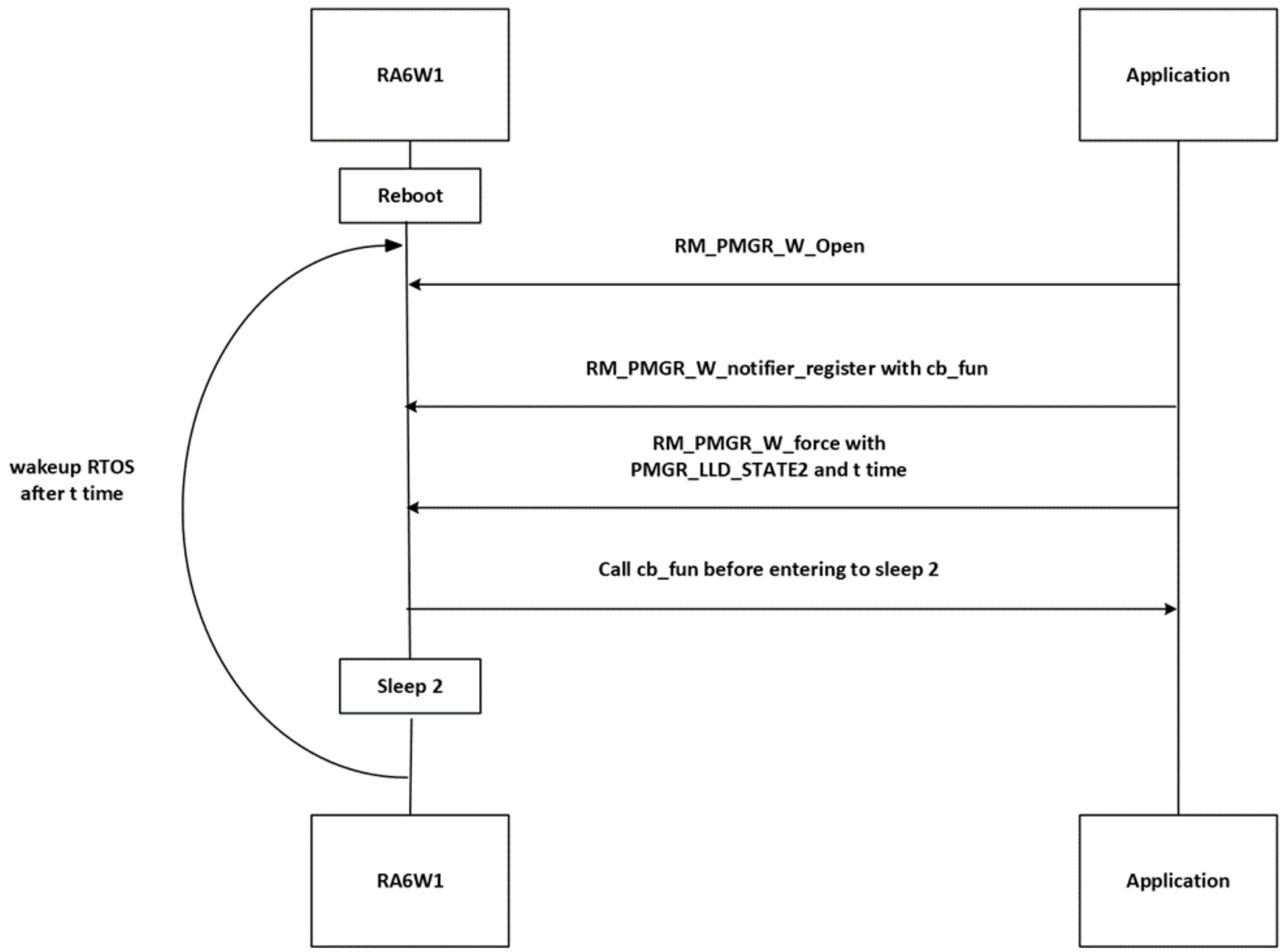

Figure 10. Sleep mode 2 configuration using API

6.8.1.2 Using AT Command

Figure 11 shows the Sleep mode 2 configuration using AT commands. After rebooting, the system initializes and sends the response +INIT:DONE,0. The MCU then issues AT+PMGRMCUWUDONE to notify that the wake-up process is complete. Next, the AT+PMGRCONSTRAINT=1,1 command is sent to remove the RAM power constraint.

Finally, AT+PMGRFORCE=2,10000 is used to force the device into Sleep mode 2 for 10 s. Time to be provided is in milliseconds. When the sleep duration expires, the system wakes up, and the process restarts.

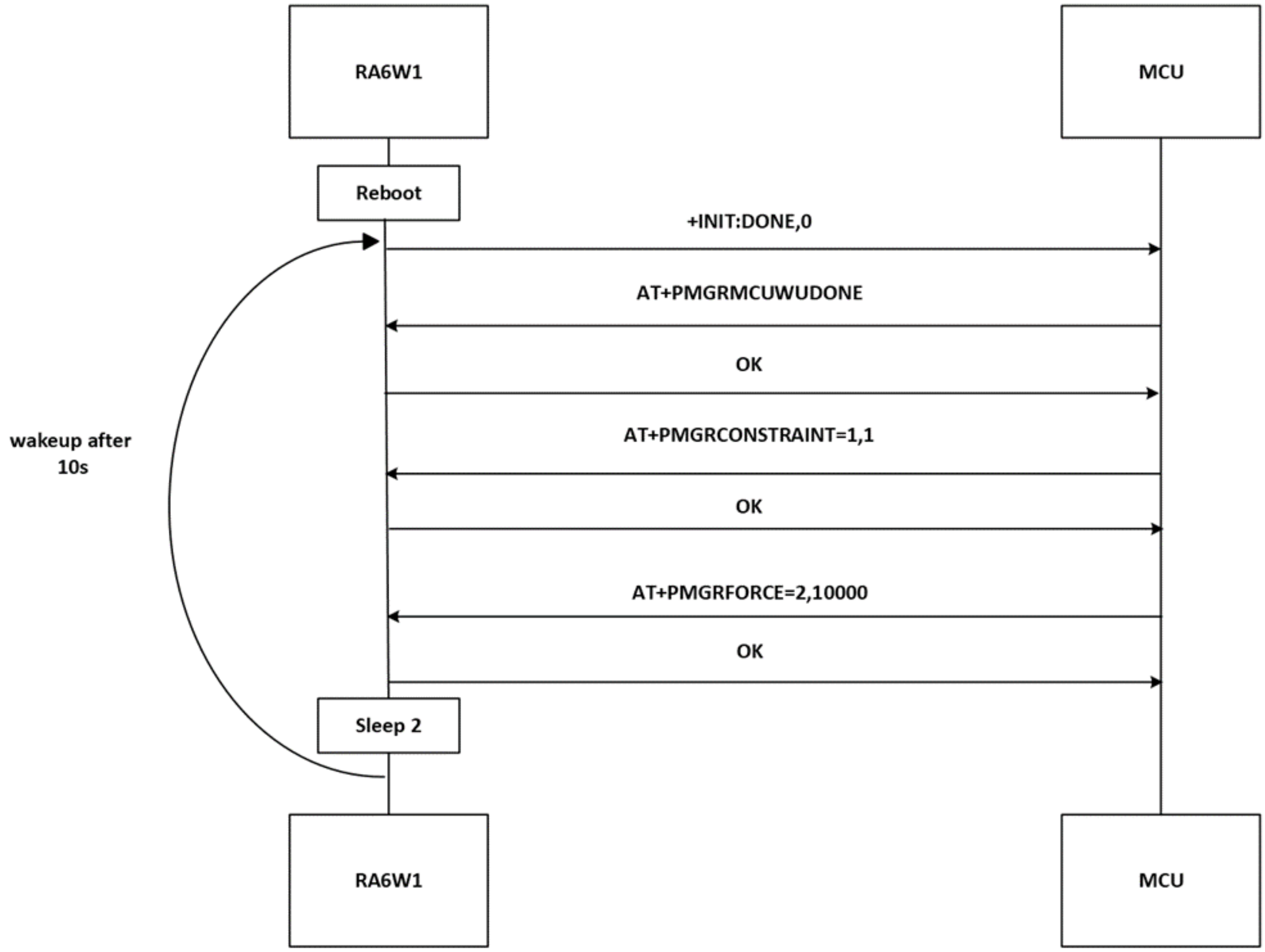

Figure 11. Sleep mode 2 configuration using AT command

6.8.2 Sleep Mode 3 Configuration

6.8.2.1 Using API

Figure 12 shows the Sleep mode 3 configuration using PMGR APIs. After rebooting, the application initializes the PMGR module with RM_PMGR_W_Open(). To retain necessary power during Sleep mode 3, RM_PMGR_W_add_sleep_constraint(PMGR_CONSTRAINT_POWER_RETENTION) is applied, while RM_PMGR_W_remove_sleep_constraint(PMGR_CONSTRAINT_POWER_RAM) removes RAM power constraints to allow Sleep mode 3 entry. Optionally, the application can register a callback function using RM_PMGR_W_notifier_register() for notification before entering sleep. The device then transitions into Sleep 3 mode using RM_PMGR_W_force(PMGR_LLD_STATE3, t_time), with the callback function executed beforehand. After the sleep duration expires, the RTOS wakes up, and the system restarts, requiring reinitialization of PMGR settings.

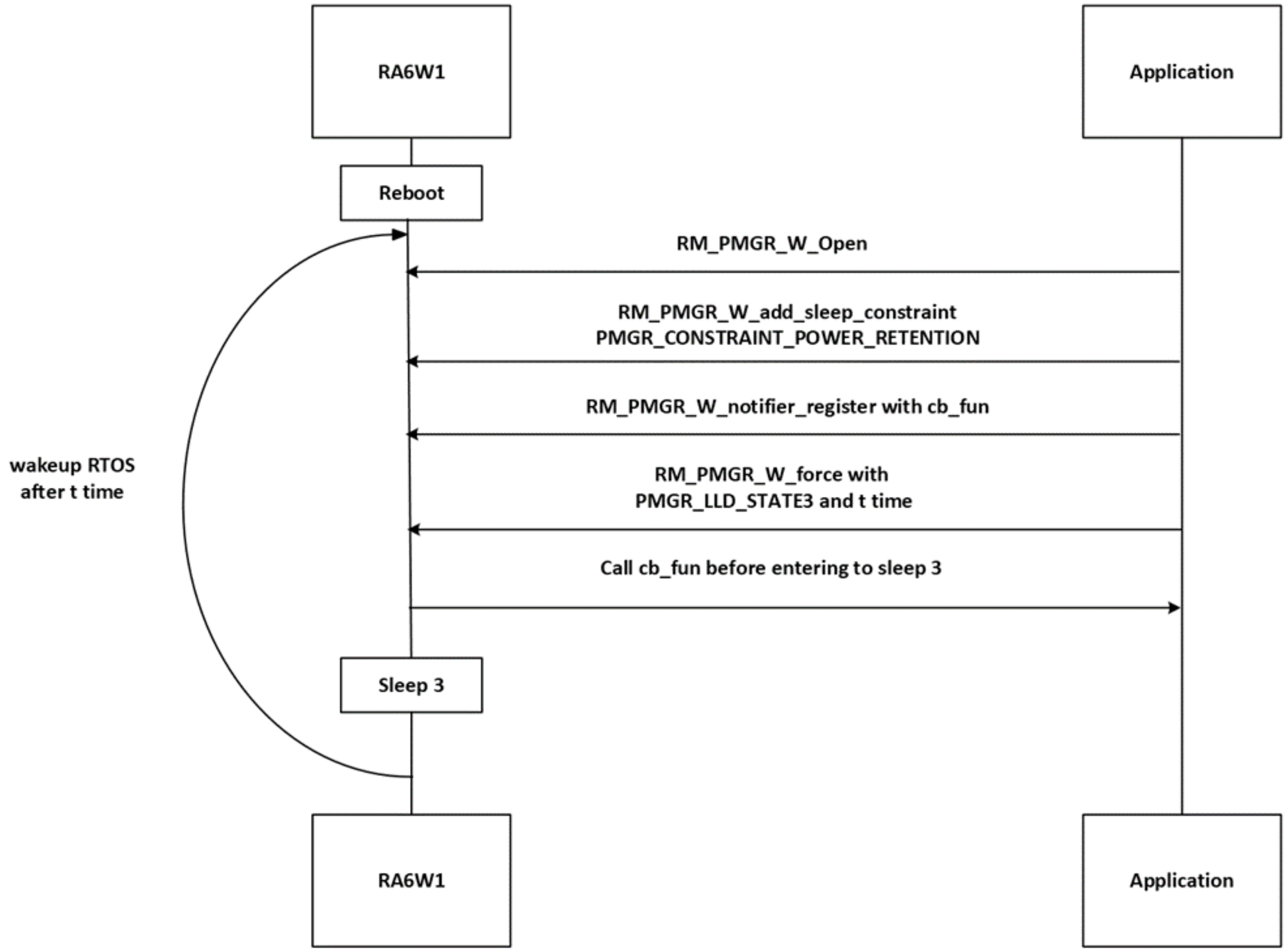

Figure 12. Sleep mode 3 configuration using API

6.8.2.2 Using AT Command

Figure 13 shows the Sleep mode 3 configuration using AT commands. After rebooting, the system sends the response +INIT:DONE,0 to indicate initialization completion. The MCU then issues AT+PMGRMCUWUDONE to notify that the wake-up process is complete, receiving an OK response. Next, AT+PMGRCONSTRAINT=1,1 is sent to remove the RAM power constraint, allowing entry into Sleep mode 3. The command AT+PMGRFORCE=3,10 forces the device into Sleep mode 3 for 10 s, confirmed with an OK response.

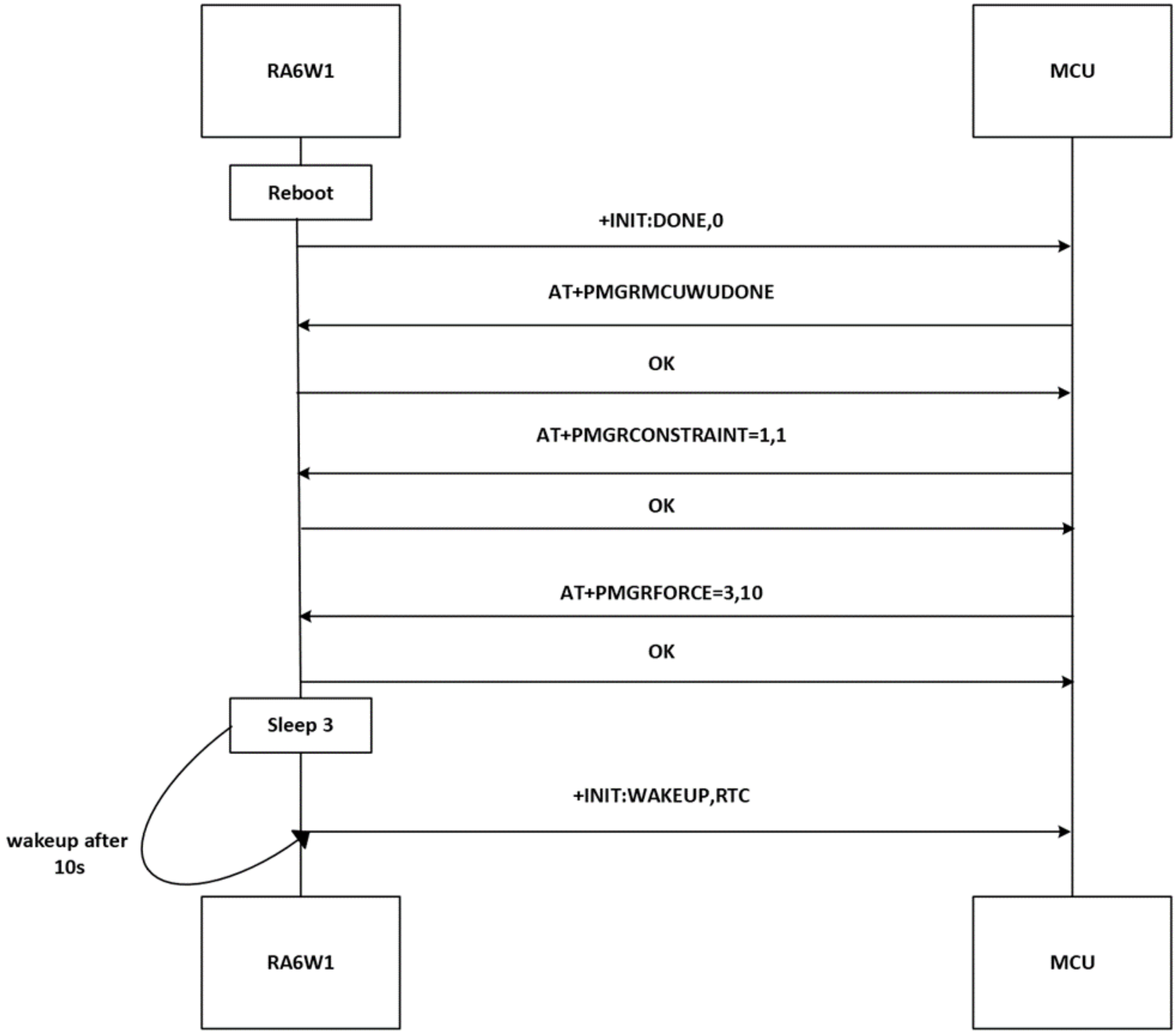

Figure 13. Sleep mode 3 configuration using AT command

6.8.3 Sleep 3 + DPM LPM Mode Configuration

6.8.3.1 Using API

The Sleep 3 DPM LPM mode is configurable only when an AP connection is available. After reboot, the system initializes by calling RM_PMGR_W_Open and sets the wake-up source using RM_PMGR_W_set_wake_source with PMGR_WAKE_SOURCE_WIFI. The retention constraint (PMGR_CONSTRAINT_POWER_RETENTION) is applied to ensure that the Wi-Fi connection and configuration are retained during sleep. When the AP connection is established and DHCP resolution is complete, an optional notifier is registered using RM_PMGR_W_notifier_register. Before entering Sleep 3 DPM LPM mode, the system either forces sleep using RM_PMGR_W_force with PMGR_LLD_DPM and a specified duration or removes the RAM power constraint using RM_PMGR_W_remove_sleep_constraint. A callback function (cb_fun) is executed before entering sleep. The system remains in low-power mode while maintaining the Wi-Fi connection, allowing it to wake up as needed, see Figure 14.

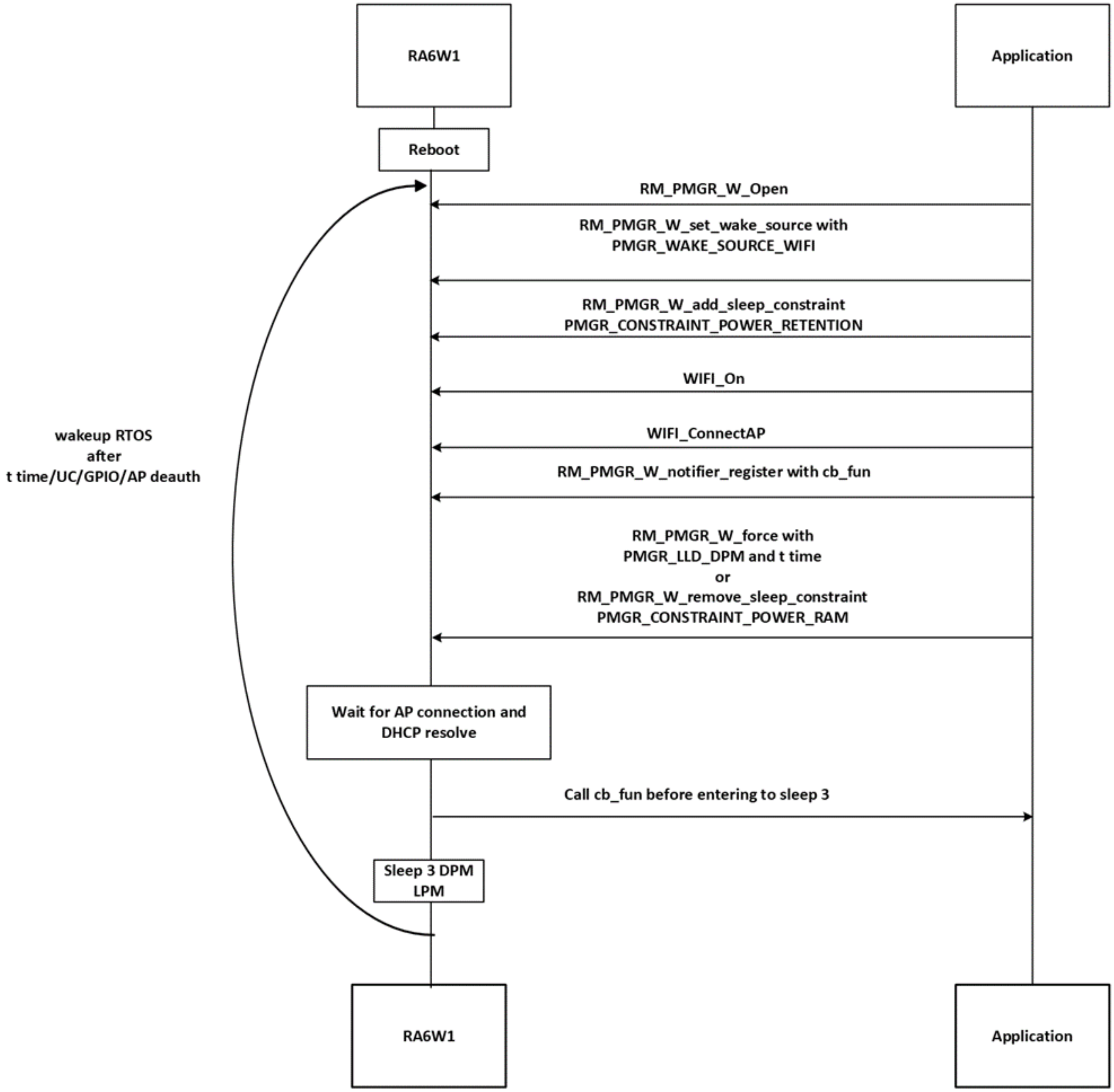

Figure 14. Sleep 3 DPM LPM mode configuration using API

6.8.3.2 Using AT Command

The Sleep 3 DPM LPM mode using AT commands is configurable only when an AP connection is available, ensuring that the Wi-Fi module maintains its connection while in low-power mode. Upon reboot, the RA6W1 module initializes and sends a +INIT:DONE,0 message to the MCU. The MCU then issues AT+PMGRMCUWUDONE to notify that the wake-up process is completed, receiving an OK response.
Following this, the MCU configures the module for AP provisioning using the necessary AT commands. After provisioning, the module autonomously attempts to connect to the AP and establish a connection. When DHCP is resolved, it notifies the MCU with the assigned SSID and IP address.
At this point, the device transitions into Sleep 3 DPM LPM mode, since DPM is enabled by default in the Wi-Fi module. While in LPM, the module maintains AP snoozing functionality. If a unicast (UC) packet is received, the device triggers an RTOS wake-up and notify the MCU of the wake-up reason using the +INIT:WAKEUP,UC message, see Figure 15.

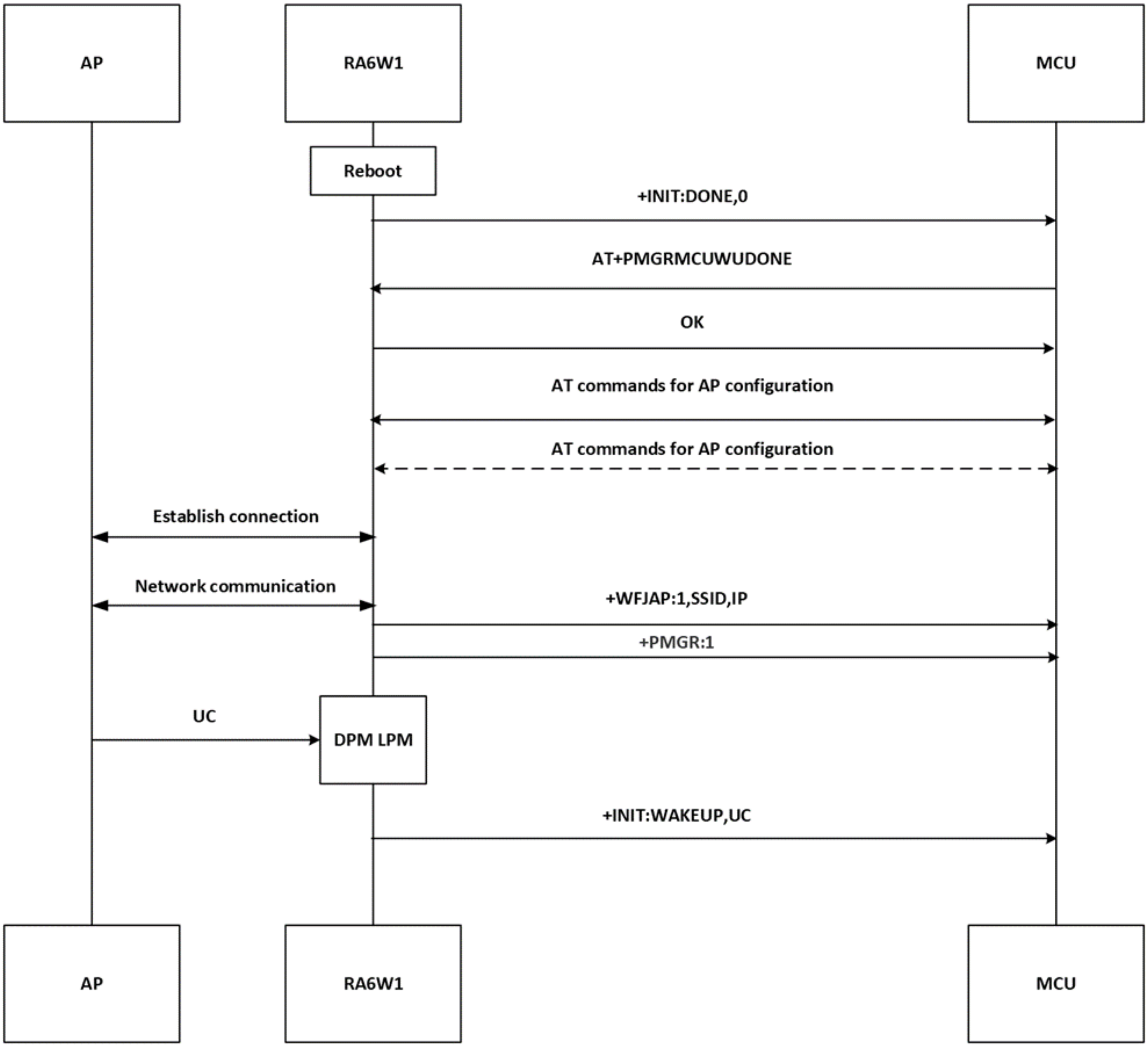

Figure 15. Sleep 3 DPM LPM default configuration using AT command

Apart from the default transition to Sleep 3 DPM LPM mode, the MCU can explicitly force the Wi-Fi module into this sleep mode using the AT+PMGRFORCE command. After rebooting, the Wi-Fi module initializes and sends +INIT:DONE,0. The MCU acknowledges wake-up by sending AT+PMGRMCUWUDONE. To prevent the module from entering sleep prematurely, the MCU applies a sleep constraint using AT+PMGRCONSTRAINT=1,1. This ensures that the module remains active until all required configurations are complete. The MCU provisions the device with an AP configuration, and the module connects to the AP. When DHCP is resolved, the module notifies the MCU with the SSID and assigned IP. The module sends +PMGR:1 indicating that it has entered DPM LPM mode, see Figure 16.

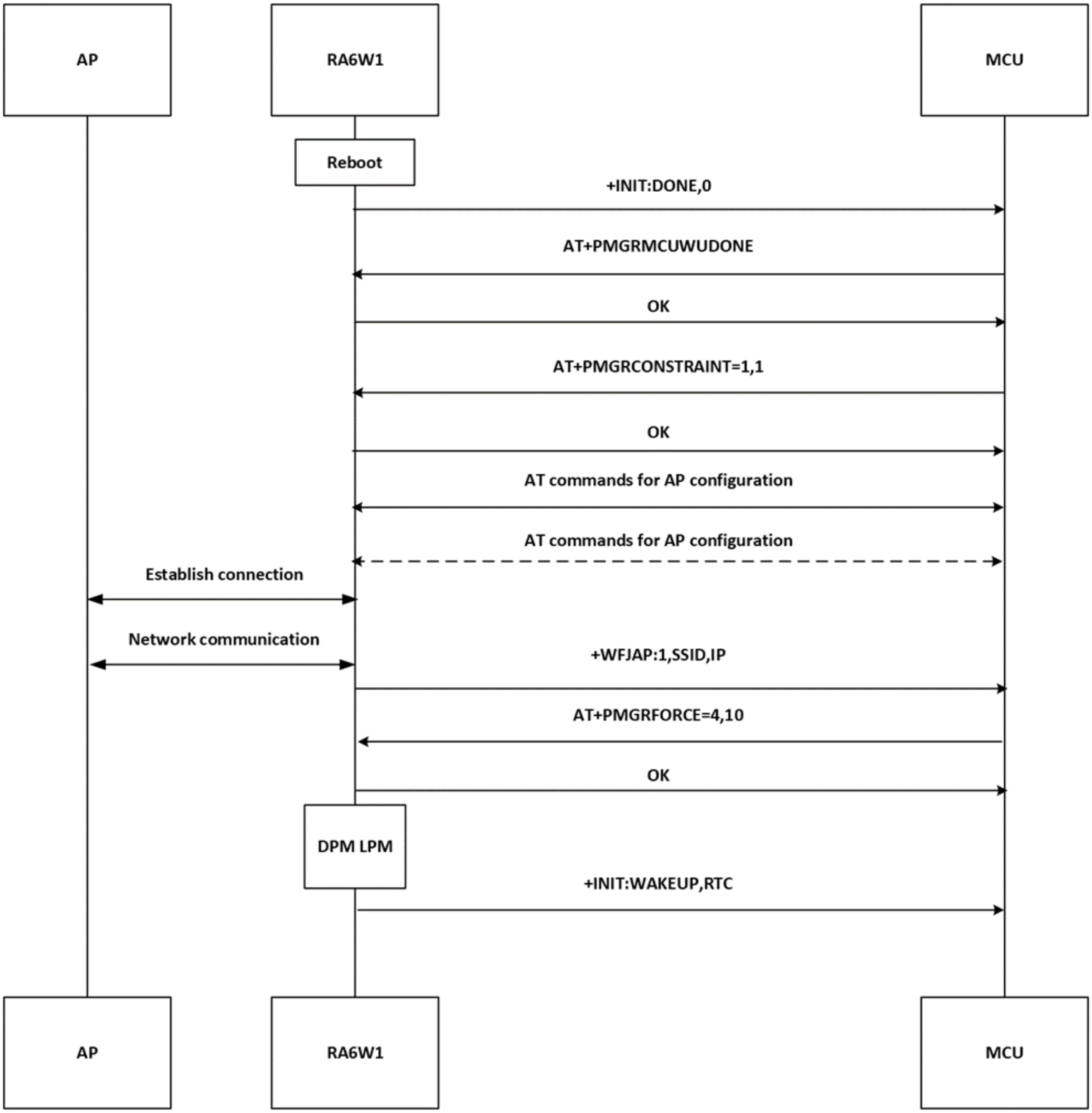

Figure 16. Sleep 3 DPM LPM mode configuration using Force AT command

6.8.4 Sleep Mode 4 Configuration

To configure Sleep mode 4:

1. Configure STA TIN with disabled DPM.
2. After connection use the following commands from UART terminal:
   - `wifi listen_interval 10 //` / this is equivalent to DTIM 10 in DPM
   - `wifi ps_mode_set 1 //` Enable/Disable Power-save mode.

NOTE
After the reboot you need to use the commands again (they are not stored in nvram).

6.9 Wake-up Reason from DPM LPM

Wake-up Reasons are defined by two parameters bsp_wakeup_source and dpm_wakeup_type. RM_PMGR_W_dpm_wakeup_src_get() used to get bsp_wakeup_source, which is the wake-up reason. RM_PMGR_W_dpm_wakeup_type_get() used to get dpm_wakeup_type, which is the wake-up type.

Table 6. BSP wake-up sources

BSP wake-up source	Values	Description
BSP_WAKEUP_RESET	0x00	Internal reset
BSP_WAKEUP_SOURCE_GPIO	0x01	Boot from GPIO wake-up signal
BSP_WAKEUP_SOURCE_WAKEUP_COUNTER	0x02	Boot from wake-up counter
BSP_WAKEUP_SOURCE_POR	0x04	Boot from power on reset
BSP_WAKEUP_SOURCE_WATCHDOG	0x08	Boot from RTC_watchdog (not cpu watchdog)
BSP_WAKEUP_SOURCE_SENSOR	0x10	Boot from sensor (ADC)
BSP_WAKEUP_SOURCE_PULSE	0x20	Boot from pulse sensor
BSP_WAKEUP_RESET_WITH_RETENTION	0x80	Boot from retention
BSP_WAKEUP_SOURCE_UNKNOWN	0xff	Boot from unknown reason

Table 7. DPM wake-up types

DPM wake-up type	Values	Description
DPM_UNKNOWN_WAKEUP	0	Unknown wake-up
DPM_RTCTIME_WAKEUP	1	RTC time wake-up
DPM_PACKET_WAKEUP	2	UC/BC/MC
DPM_USER_WAKEUP	3	In case of DPM1_USER_0 , DPM_USER_1 wake-up
DPM_NOACK_WAKEUP	4	In case of No ACK, No BCN
DPM_DEAUTH_WAKEUP	5	TIM received the Deauth packet
DPM_TIM_ERR_WAKEUP	6	TIM has something wrong and fault
DPM_TCP_KA_TIMEOUT_WAKEUP	8	TCP Keep alive timeout wake-up

The device can be awake by following main events in DPM LPM.

- GPIO from MCU, see Figure 17
- UC from AP, see Figure 18
- Disconnection from AP, see Figure 19

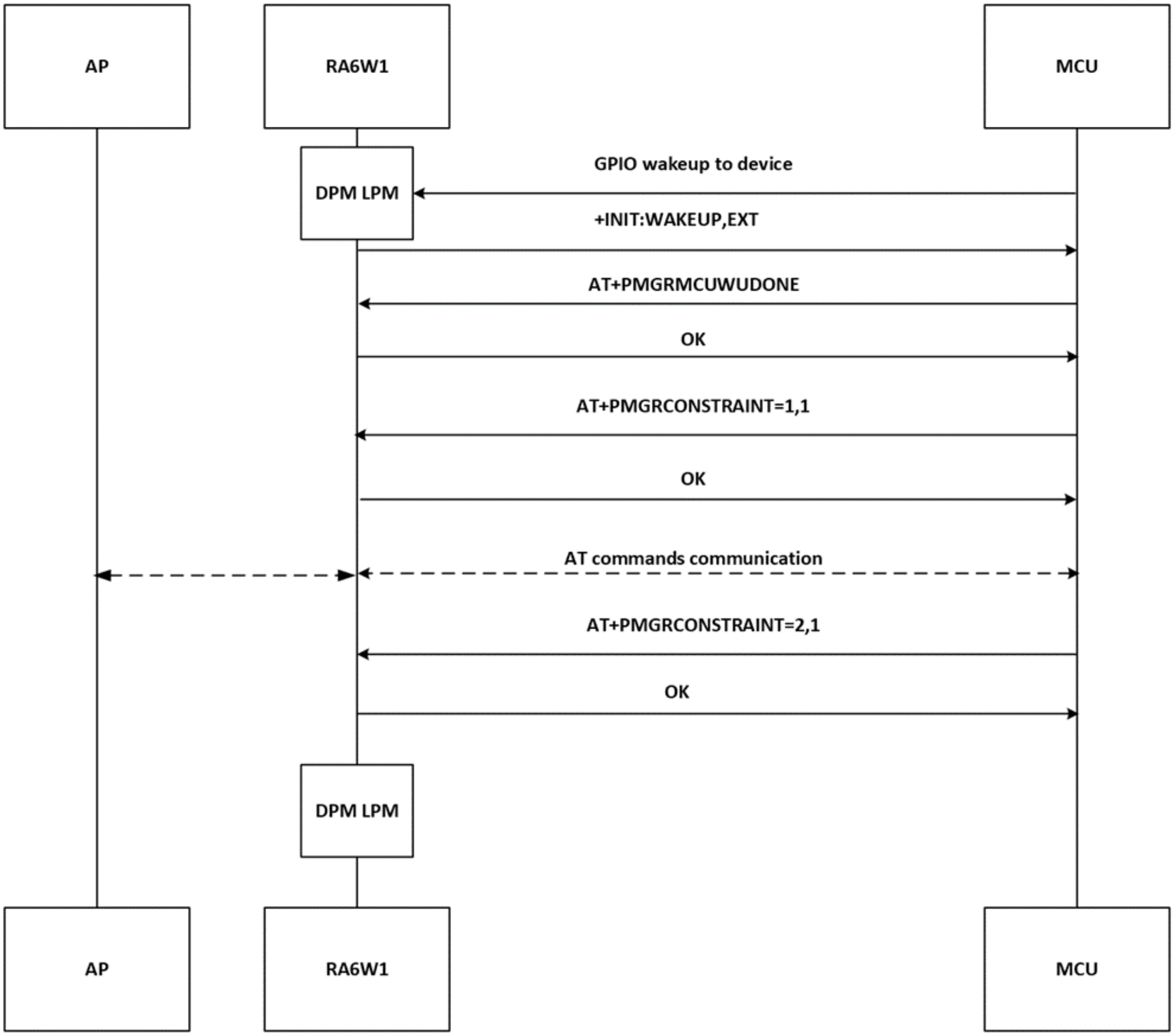

Figure 17. Flow of AT commands for waking up by GPIO of MCU

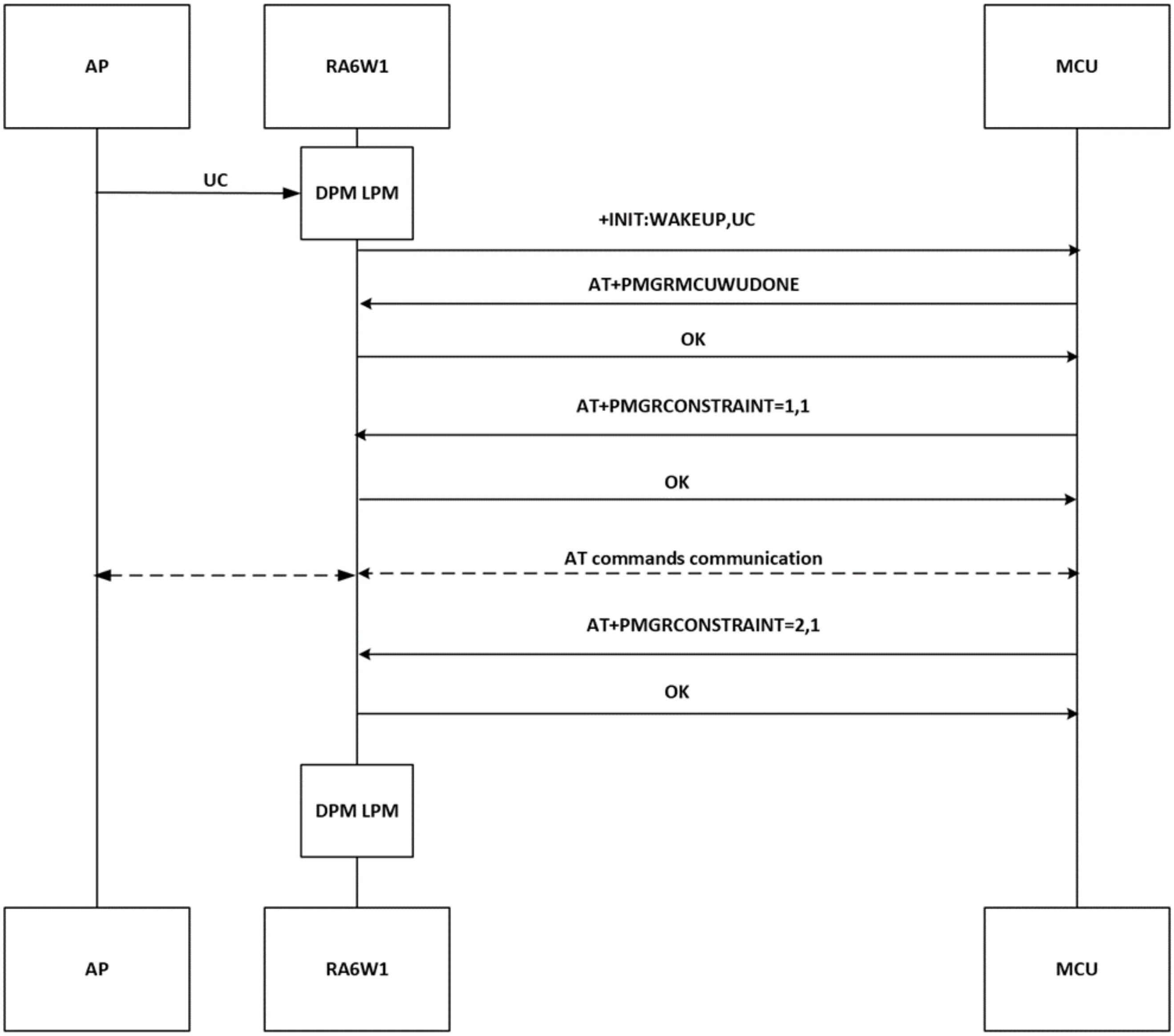

Figure 18. Flow of AT commands for waking up by UC from AP

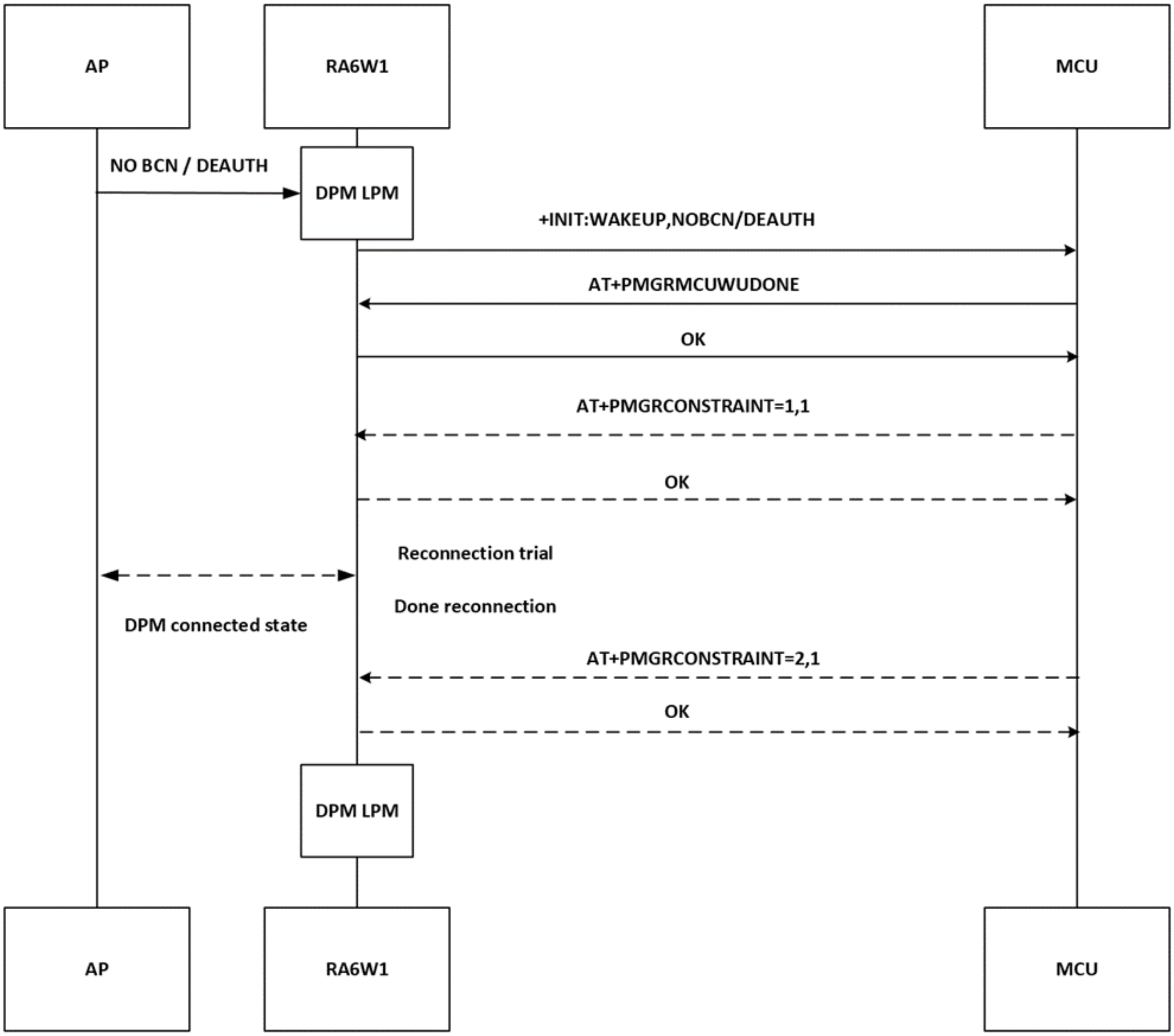

Figure 19. Flow of AT commands for waking up by disconnection from AP

6.10 DPM LPM Specific Configuration

- **DPM Keep Alive Time**
  To maintain AP connectivity in LPM mode, Keep Alive packets are sent periodically to the AP. By default, this interval is set to 30,000 ms. The command AT+PMGRDPMKA is available to configure this, with a valid range of 0 to 600,000 ms.
- **DTIM Count**
  This is the interval to check the AP's beacon. By default, this count is set to 10. The command AT+ PMGRDPMTIMWU is available to configure this, with a valid range of 1 to 6000.
- **TWT Configuration**
  TWT configuration is managed through API functions such as WIFI_TwtSetup() for setting up TWT and WIFI_ TwtTeardown() for disabling it. On the other hand, the RA6W1 provides an AT command-based approach, where TWT operations are controlled using AT+PMGRFORCE=5,<twt_action>, .

7. TCP Client Example with DPM

A TCP Client example with DPM demonstrates sending data to a server while minimizing power usage. The device connects to a TCP server, transmits data, and then enters Low Power Mode. During DPM LPM, power consumption is reduced significantly while retaining system state and RTC operation. The device wakes up periodically or on events to re-establish the connection and continue communication.

7.1 Using API

Figure 20. TCP client send function using API

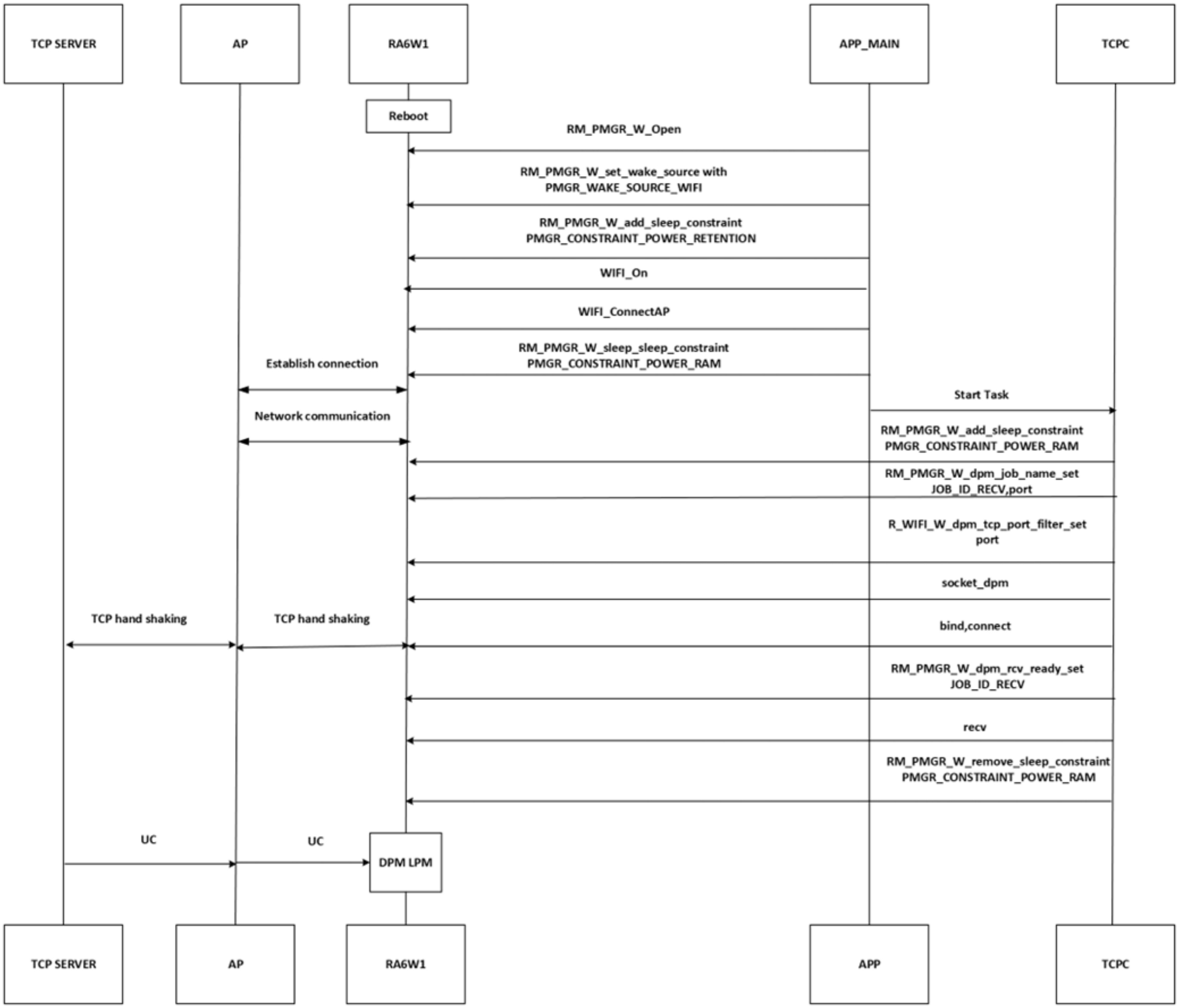

Figure 21. TCP client receive function using API

7.2 Using AT Command

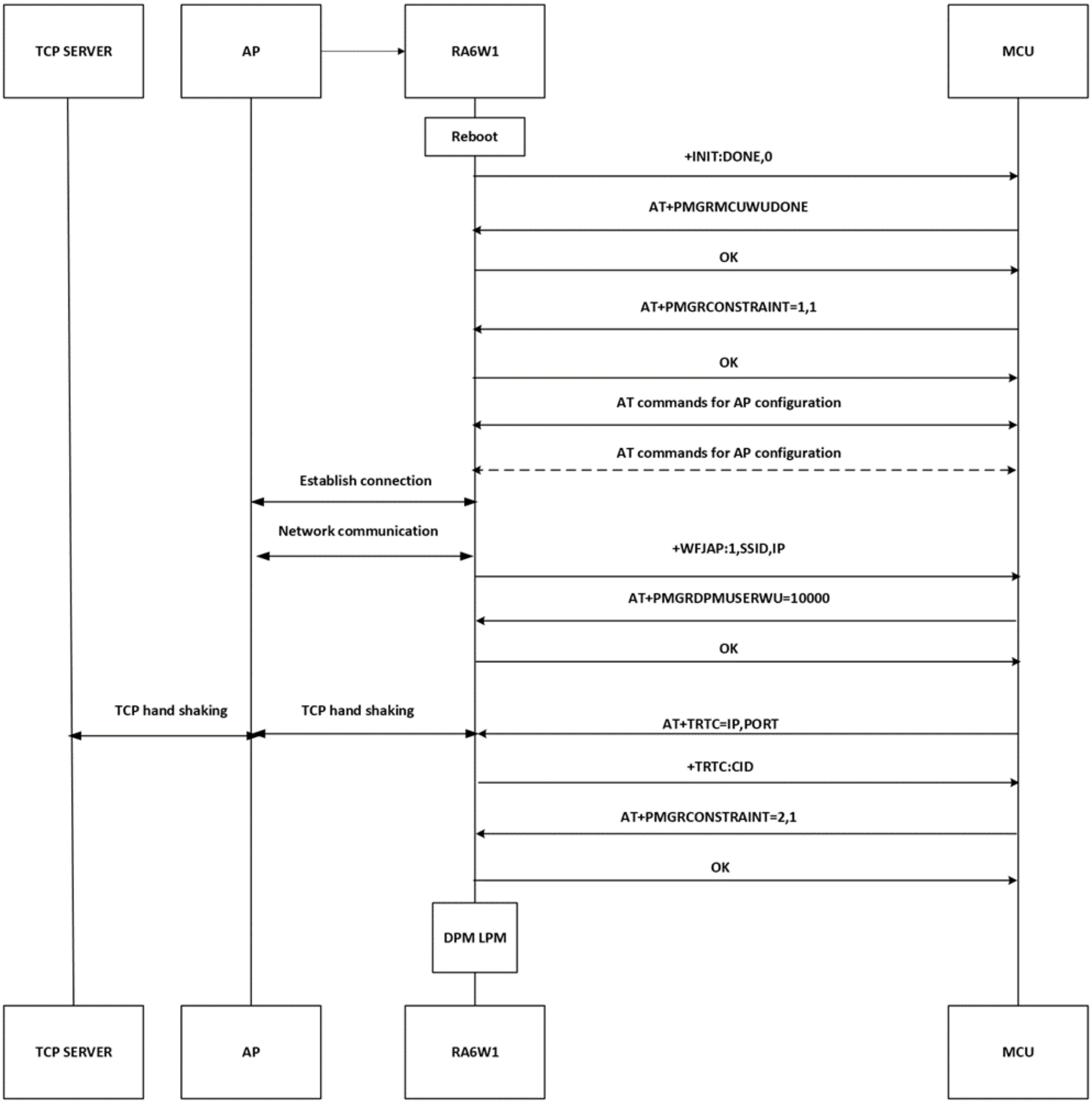

Figure 22. TCP client configuration on DPM

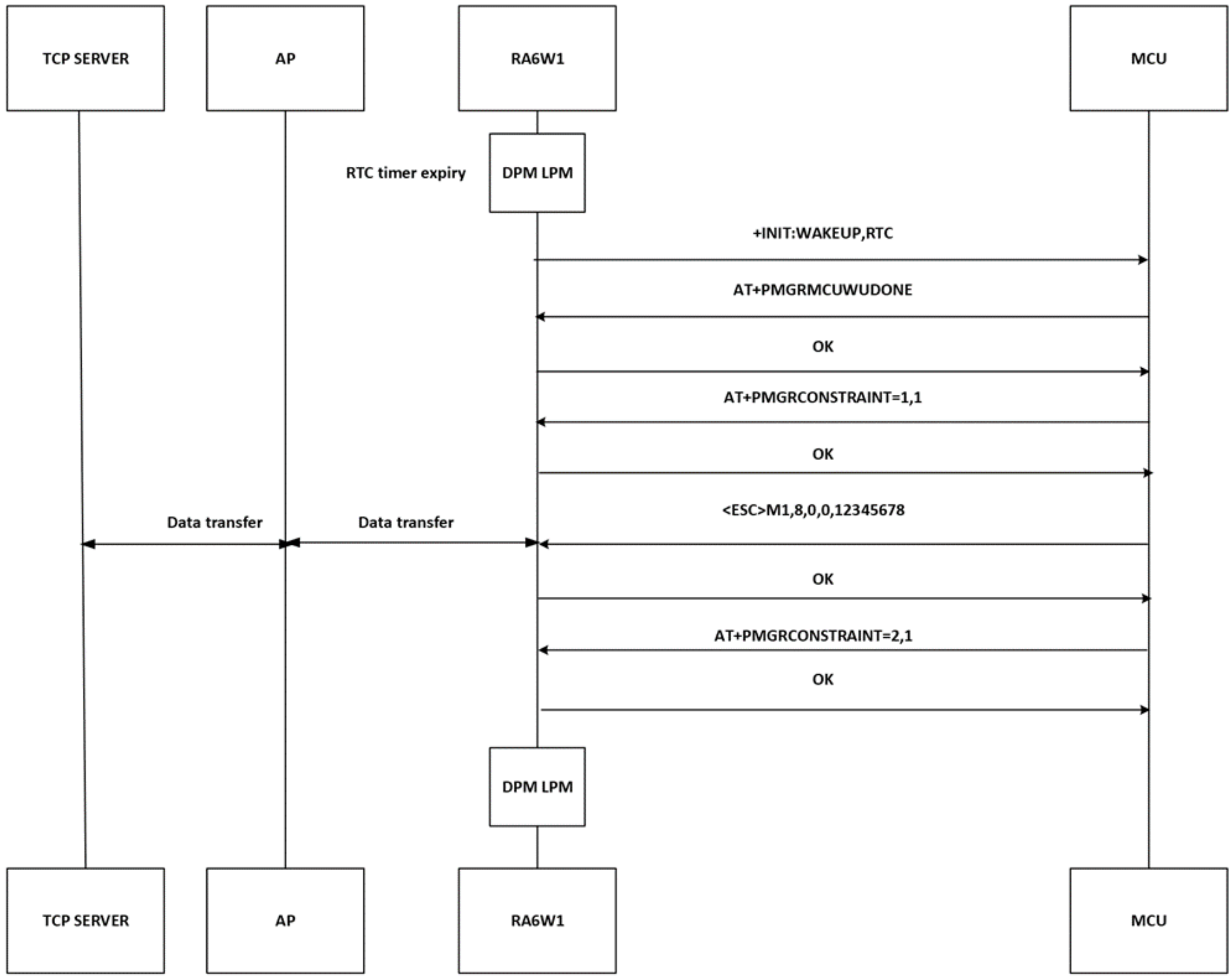

Figure 23. TCP client send flow using AT commands

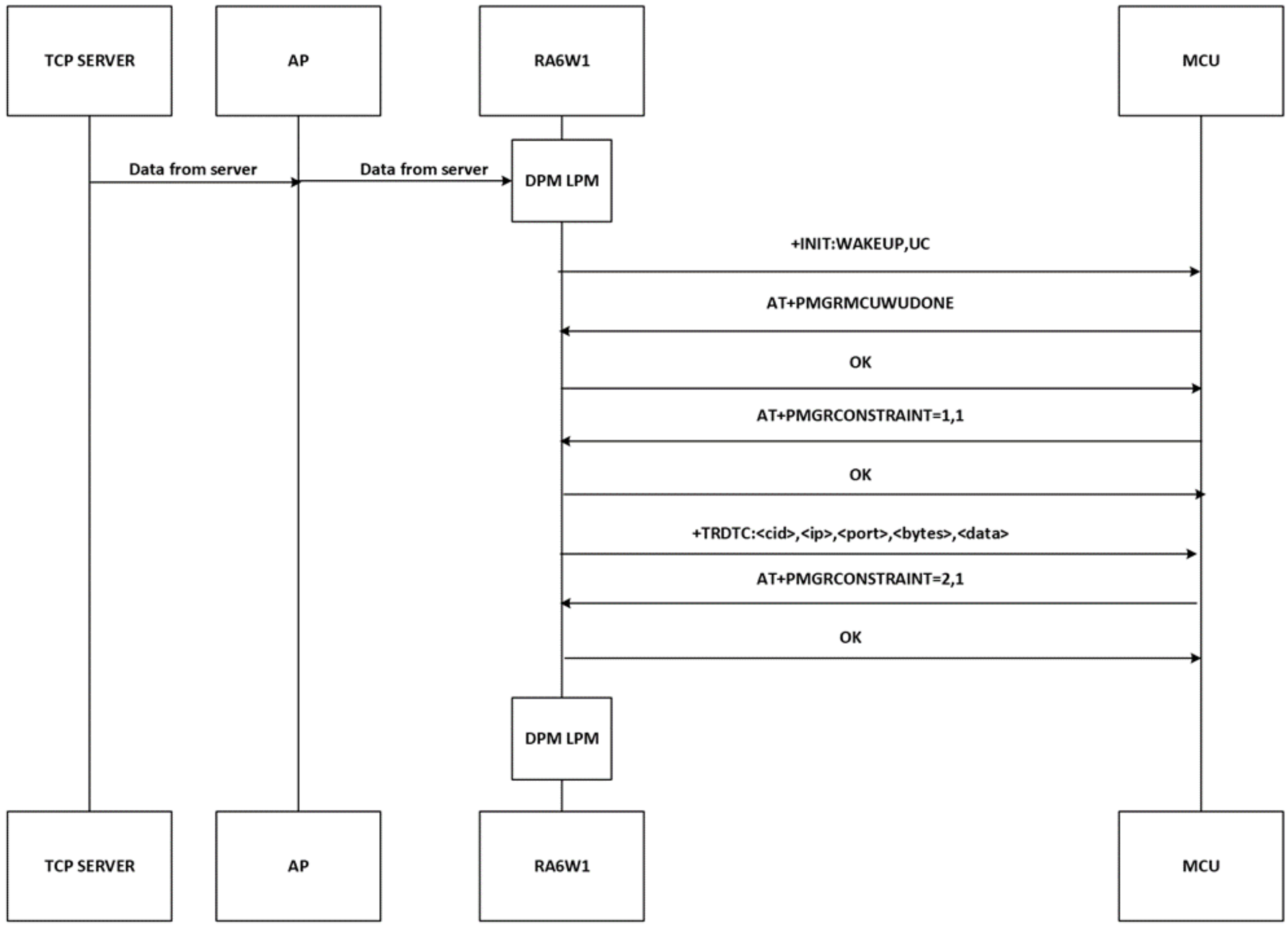

Figure 24. TCP client receive flow using AT commandsConclusions

Effective power management is critical for optimizing the performance, efficiency, and longevity of IoT devices. The PMGR plays a central role in regulating power consumption, enabling low-power states, and ensuring seamless transitions between different operating modes. By leveraging intelligent power management techniques, developers can achieve significant energy savings without compromising device functionality.

This application note has provided an in-depth overview of PMGR's capabilities, key features, and best practices for implementation. A well-designed power management strategy enhances battery life, reduces operational costs, and ensures reliable performance in diverse IoT applications. As IoT technology continues to evolve, advancements in power management solutions remain a cornerstone for achieving sustainable and efficient device operation.

Appendix A Comparison of AT Commands: DA16200 vs. RA6W1

The following comparison highlights the key differences and similarities between the AT command sets of DA16200 and RA6W1, helping developer's transition between the two modules efficiently.

Table 8. Comparison of AT Commands: DA16200 vs. RA6W1/RA6W2/RRQ61001/RRQ61051

Feature/Command	DA16200 behavior	RA6W1 behavior
DPM on/off	AT+DPM	Not supported. DPM enabled by default
DPM keep alive	AT+DPMKA	AT+PMGRDPMKA
Set DPM TIM wake-up count	AT+DPMTIMWU	AT+PMGRDPMTIMWU
Set DPM user wake-up time	AT+DPMUSERWU	AT+PMGRDPMUSERWU
Set the user application not to enter DPM sleep	AT+CLRDPMSLPEXT	AT+PMGRCONSTRAINT=1,1
Set the user application ready to enter DPM sleep	AT+SETDPMSLPEXT	AT+PMGRCONSTRAINT=2,1
Set Sleep mode 2	AT+SETSLEEP2EXT	AT+PMGRFORCE=2,<time>
Set Sleep mode 3	AT+SETSLEEP3EXT	AT+PMGRFORCE=3,<time>
ABNORMAL commands	Supported	Not supported
Wake source	Not supported	AT+PMGRWAKESRC
Constraints	Not supported	AT+PMGRCONSTRAINT
TWT	Not supported	AT+PMGRFORCE=5,<><>, …

Appendix B PMGR CLI Commands

Figure 25 shows the available PMGR CLI commands.

```
[/RRQ61000/pmgr] # pmgr
[/RRQ61000/pmgr] # help
pmgr commands:
        status           : PMGR show status
        add_constraint   : PMGR add constraint [constraint]
        remove_constraint : PMGR remove constraint [constraint]
        set_wake_source : PMGR set wake source [wake_source]
        clear_wake_source : PMGR clear wake source [wake_source]
        force            : PMGR force [state] [sleep_time_sec] where [sleep_time_sec] is optional
        dpm_sleep        : timer [id] [msec], ...
        dpm_timer        : timer [id] [sec], ...
        dpm_all_timer    : alltimer, alloc all dpm timer
        dpm_timer_list   : timer_list, print out timer list
        dpm_kill_timer   : kill timer [id]
        pm_sleep         : timer [id] [second], ...
        ip_cond          : IPv4/6 condition check for entering PMGR_LLD_DPM

[/RRQ61000/pmgr] #
```

Figure 25. PMGR CLI commands

Appendix C RTM CLI

Retention Memory information can be obtained using CLI rtm_info, sw, see Figure 26.

```
RTM_ARP_BASE            : 0x[/RRQ61000] #  28600960, size:0xf0
RTM_DNS_BASE            : 0x28600a50, size:0x98
RTM_RTC_TIMER_BASE      : 0x28600ae8, size:0x230
RTM_DPM_MONITOR_BASE    : 0x28600d18, size:0x7c
APP_ALLOC_SZ            : 0xa94

RTM_TCP_BASE            : 0x2860e400, size:0x660
  dpm_lwip_sock_names(0x2860e400, 640), dpm_tcp_sess_list(0x2860e680, 992:248*4)
  total_size: 1632(640+992)

RTM_USER_POOL_BASE      : 0x2860ea60, size:0x40

RTM_USER_DATA_BASE      : 0x2860eaa0, size:0x1560


<< DPM RTM MEM STATUS >>
mem: 0x2860eaa0 used:0 next:0x1558 prev:0x0 size:5456
Total mem    : 5464
node cnt     : 1
error_cnt    : 0
illegal cnt: 0
Used_mem     : 0
Free mem     : 5456

- Whole available to allocate ( No allocated ) ...
```

Figure 26. RTM information

8. Revision History

Revision	Date	Description
1.04	May 19, 2026	Changed the Device from RA6W1 to RA6W1/RA6W2/RRQ61001/RRQ61051.
1.03	Apr 17, 2026	Added Sleep mode examples.
1.02	Dec 3, 2025	Added Sleep mode 4 and updated RTC Timer related changes.
1.01	Aug 31, 2025	Updated Section 7.4.
1.00	May 30, 2025	First release.

CorporateHeadquarters

TOYOSU FORESIA, 3-2-24 Toyosu
Koto-ku, Tokyo 135-0061, Japan

www.renesas.com

Contact Information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit www.renesas.com/contact-us/