

WebMite RP2350B + Pimoroni RM2

Wi-Fi Warm-Restart Reliability Patch - RM2 V1

Release

Detailed problem analysis, rejected approaches, final safe patch, build verification, and field validation

Status: RM2 V1 is field-tested successfully. Wi-Fi reconnects after CPU RESTART without a manual hardware reset. The release contains no SoftReset() modification.

Firmware base	WebMite MMBasic RP2350B Edition V6.04.00RC1
Source tag	PicoMite-V6.04.00RC1
Source commit	083337152f09e84a0e538afb18986b530fe5804f
Board target	pimoroni_pico_plus2_w_rp2350 / external Pimoroni RM2 (CYW43439)
Final patch location	net/WiFi.c only
Final UF2	WebMiteRP2350V6.04.00RC1_RM2_V1.uf2
SHA-256	E93490EF4AA9D35B0D2E2D1ACB0980806862286D22F3A18AED2DD60C172A21E7
Final result	CPU RESTART -> Wi-Fi reconnects automatically

Release naming note: the final validated code was developed through internal patch revision V7, then released under the user-selected firmware filename WebMiteRP2350V6.04.00RC1_RM2_V1.uf2. The filename V1 is the production/release name documented in this report.

1. Executive summary

Problem. On the WeAct RP2350B board with a Pimoroni RM2 CYW43439 module, Wi-Fi worked after power-on and after pressing the physical RST/RUN button, but failed after WebMite CPU RESTART, WATCHDOG restart, and option-triggered software restarts. The router was not the cause.

Key measurement. After a warm restart, `cyw43_arch_init_with_country()` returned 0, so the CYW43 driver initialized successfully. The first actual Wi-Fi join then returned -8 (`PICO_ERROR_CONNECT_FAILED / LINK_FAIL`). A second join attempt, after allowing the polling backend to settle, succeeded and obtained the normal IP address.

Final solution. Keep the original WebMite `SoftReset()` implementation completely unchanged. Patch only `net/WiFi.c` so that the join is retried up to three times, but only when the specific result is `PICO_ERROR_CONNECT_FAILED`. Between retries, poll the CYW43 async backend for about one second. Other errors, including authentication failures, are not retried unnecessarily.

Safety correction. An earlier experimental patch called `cyw43_arch_deinit()` and manipulated RM2 pins inside `SoftReset()`. That design was rejected because `SoftReset()` can be called from the 1 ms timer interrupt for watchdog and hang-detection events. Shutting down the network stack from that interrupt context is unsafe, and it also delays starting the hardware watchdog. RM2 V1 contains none of that code.

Final design rule: `SoftReset()` stays upstream-original. CYW43 hardware reset remains the SDK driver's responsibility. Reliability is implemented at the Wi-Fi connection layer, where the failure actually occurs.

2. Hardware and firmware configuration

MCU board	WeAct RP2350B / Pico 3 style board
Wi-Fi module	Pimoroni RM2 breakout, CYW43439
RM2 WL_ON	GP23
RM2 DAT	GP24
RM2 CS	GP25
RM2 CLK	GP29
Power	3.3 V and GND
WebMite CPU speed used during tests	250 MHz
Wi-Fi mode	Station mode; Fritz!Box network; Telnet console enabled

Board-specific GP23 issue. The WeAct board originally connected its KEY user-button circuit to GP23 through a 0-ohm resistor. Removing that resistor isolated GP23 from the KEY circuit and fixed

cold-start Wi-Fi behavior when powered from a power bank. After that hardware correction, cold boot and physical RST were reliable, while CPU RESTART still failed. This separated the board-level GP23 problem from the later warm-restart connection problem.

3. Original failure pattern

Reset / start method	Wi-Fi result before RM2 V1	Observation
Cold power-up	OK	Reliable after the GP23 KEY-circuit resistor was removed
Physical RST/RUN button	OK	Wi-Fi always returned
CPU RESTART	FAIL	Wi-Fi did not return without manual RST
WATCHDOG HW	FAIL	Same warm-restart symptom
OPTION-triggered restart	FAIL	For example after CPU speed option changes

4. Investigation and diagnostic sequence

- 1. Hardware isolation:** The 0-ohm resistor connecting the WeAct KEY circuit to GP23 was removed. This solved power-on behavior but did not solve CPU RESTART.
- 2. V1 experiment:** Forcing WL_ON low after restart was tested. It did not restore Wi-Fi.
- 3. V2/V3 experiment:** CYW43 deinitialization plus manual GP23/24/25/29 handling around SoftReset() was tested. Wi-Fi still did not reconnect reliably and, more importantly, this approach was later identified as unsafe in interrupt context.
- 4. Driver-init instrumentation:** cyw43_arch_init_with_country() was instrumented. After CPU RESTART it returned 0, proving that the driver initialization itself succeeded.
- 5. Connection instrumentation:** WebConnect() was instrumented to expose the real result of cyw43_arch_wifi_connect_timeout_ms(). The first warm-restart join returned -8, not a 30-second timeout.
- 6. Retry experiment V5:** When the -8 failure was followed by a one-second poll/settle interval and another join call, attempt 2 returned 0 and Wi-Fi connected successfully.
- 7. Final cleanup (development revision V7):** All SoftReset(), RM2 pin-reset, and debug instrumentation changes were removed. Only the connection retry logic in net/WiFi.c remained. After field validation, this code was released under the filename WebMiteRP2350V6.04.00RC1_RM2_V1.uf2.

5. Diagnostic evidence

The decisive diagnostic behavior was:

```
Normal/cold start:
```

```

cyw43_init = 0
first Wi-Fi join rc = 0
Wi-Fi = connected

After CPU RESTART before the retry patch:
cyw43_init = 0
first Wi-Fi join rc = -8
link status = FAIL

Retry test after CPU RESTART:
attempt 1 -> rc = -8
wait/poll about 1 second
attempt 2 -> rc = 0
Connected 192.168.178.143

```

Interpretation. The CYW43439 was not failing to initialize. The failure occurred one layer later, during the first station join after a warm RP2350 reset. Therefore the correct intervention point is WebConnect(), not SoftReset().

6. Why the SoftReset() patch was rejected

The experimental SoftReset modification performed operations similar to the following before starting the hardware watchdog:

```

if (startupcomplete)
{
    cyw43_arch_deinit();
    startupcomplete = 0;
}

gpio_init(23);
gpio_put(23, 0);
gpio_set_dir(23, GPIO_OUT);

gpio_init(24);
gpio_set_dir(24, GPIO_IN);
gpio_init(25);
gpio_set_dir(25, GPIO_IN);
gpio_init(29);
gpio_set_dir(29, GPIO_IN);

uSec(300000);
watchdog_enable(1, 1);

```

Why this is unsafe:

- SoftReset() is used by WebMite not only for user-initiated CPU RESTART but also from the 1 ms timer callback for watchdog timeout and hang-detection paths.
- cyw43_arch_deinit() and the polling async/network stack are not safe operations to perform from that interrupt context, especially if the interrupt arrived while foreground code was already inside networking code.
- The hardware watchdog was started only after the network shutdown and 300 ms delay. If the deinitialization blocked or crashed, the recovery watchdog might never be armed.
- The CYW43 driver already performs its own WL_REG_ON reset sequence during initialization (low, delay, high, stabilization delay). Duplicating this in SoftReset() was unnecessary.
- Experimentally, the V3 SoftReset sequence did not solve the actual join failure anyway.

RM2 V1 safety decision: PicoMite.c and misc/External.c are restored exactly to the official RC1 Git HEAD. SoftReset() is not patched.

7. Original SoftReset() retained in RM2 V1

```
void SoftReset(int code)
{
    _excep_code = code;
#ifdef USBKEYBOARD
    USBenabled = false;
    uSec(50000); // wait for outstanding requests to complete
#endif
    watchdog_enable(1, 1);
    while (1) {}
}
```

This function is left exactly as provided by the upstream PicoMite-V6.04.00RC1 source. No CYW43 calls, no RM2 GPIO changes, and no additional delay are present.

8. RM2 V1 patch in net/WiFi.c

The original single join call:

```
int connect_result = cyw43_arch_wifi_connect_timeout_ms(
    (char *)Option.SSID,
    (char *)(*Option.PASSWORD ? Option.PASSWORD : NULL),
    (*Option.PASSWORD ? CYW43_AUTH_WPA2_MIXED_PSK : CYW43_AUTH_OPEN),
    30000);
```

is replaced by the following retry wrapper:

```
int connect_result = PICO_ERROR_CONNECT_FAILED;

// A warm RP2350 reset can leave the first CYW43 join attempt in LINK_FAIL.
// Retry that specific failure up to two more times before giving up.
for (int rm2_attempt = 1; rm2_attempt <= 3; rm2_attempt++)
{
    connect_result = cyw43_arch_wifi_connect_timeout_ms(
        (char *)Option.SSID,
        (char *)(*Option.PASSWORD ? Option.PASSWORD : NULL),
        (*Option.PASSWORD ? CYW43_AUTH_WPA2_MIXED_PSK : CYW43_AUTH_OPEN),
        30000);

    if (connect_result == 0)
        break;

    if (connect_result != PICO_ERROR_CONNECT_FAILED || rm2_attempt == 3)
        break;

    // Give the poll backend one second to settle before the next join.
    for (int rm2_wait = 0; rm2_wait < 100; rm2_wait++)
    {
        cyw43_arch_poll();
        uSec(10000);
    }
}
```

9. Detailed patch behavior

Initial result value: connect_result is initialized to PICO_ERROR_CONNECT_FAILED only so it has a defined value before the loop. Each loop iteration immediately replaces it with the real join result.

Maximum of three joins: rm2_attempt runs from 1 through 3. This matches observed behavior on similar WebMite systems where two or three attempts can occasionally be required.

Success path: If the join returns 0, the loop exits immediately. The original WebConnect() success logic then continues unchanged: WIFIconnected is set, LastWifiErr becomes 0, and TCP/TFTP/UDP services are opened as configured.

Selective retry: A retry happens only when connect_result equals PICO_ERROR_CONNECT_FAILED. Authentication errors and other error classes are not blindly retried.

One-second settle interval: Between failed -8 attempts, cyw43_arch_poll() is called 100 times with 10 ms between calls. This lets the polling-based CYW43/lwIP context process pending work instead of simply sleeping.

Final failure behavior: If all three attempts return -8, or another non-retriable error occurs, control falls through to the original WebMite failure handling. The patch does not bypass or replace the existing error logic.

No reset recursion: The patch does not call CPU RESTART, SoftReset(), watchdog functions, or RM2 GPIO reset functions. It retries the operation that actually failed.

10. Relationship to the previous BASIC workaround

A previously used Pico 2 W / WebMite 6.02 BASIC workaround was:

```
Do While MM.Info(IP Address) = "0.0.0.0"  
  If Timer > 5000 Then CPU RESTART  
Loop
```

That approach can recover from an unreliable initial Wi-Fi connection, but it restarts the entire MCU and user program. RM2 V1 moves the recovery to the lower connection layer: the join itself is retried before WebMite reports failure. This avoids unnecessary full-system restarts and keeps SoftReset() focused on its recovery role.

11. Robust source-cleanliness verification

Before building the RM2 V1 release, a dedicated audit script verified all of the following:

- Git tag is PicoMite-V6.04.00RC1 and HEAD is 083337152f09e84a0e538afb18986b530fe5804f.
- git diff --check reports no patch/whitespace errors.
- The only tracked source change is net/WiFi.c.
- PicoMite.c is identical to Git HEAD.
- misc/External.c is identical to Git HEAD.
- The entire SoftReset() function body is identical to the upstream original.
- No old RM2 V3 reset comments, cyw43_arch_deinit() calls in critical files, RM2 GPIO reset code, 300 ms delay, or diagnostic markers remain.
- The final retry patch is present and complete.
- Line endings match repository expectations: PicoMite.c LF, External.c CRLF, WiFi.c LF.

Audit result: PASS - no remnants of the unsafe SoftReset version were found in the critical source files. Only the final Wi-Fi retry patch remained as a tracked modification.

12. Build environment and successful compilation

Pico SDK	2.3.0
ARM GCC	13.3.1 / Arm GNU Toolchain 13.3.Rel1

Build command	buildpicomite.bat WEBRP2350
Build target	WEBRP2350
Result	100% Built target PicoMite

```
FLASH: used 0x16B6CC / limit 0x16C000
       margin: +2356 bytes (+2.3 KB) [PASS]
RAM:   margin: +9504 bytes (+9.3 KB) [PASS]
HEAP:  alignment [PASS]
LFS:   alignment [PASS]
Build elapsed: 00:12:46.57
```

The generated UF2 was copied to the release filename WebMiteRP2350V6.04.00RC1_RM2_V1.uf2 and the two files were verified with SHA-256. Both produced the same hash:

```
E93490EF4AA9D35B0D2E2D1ACB0980806862286D22F3A18AED2DD60C172A21E7
```

13. Final field validation

The RM2 V1 firmware was flashed and tested on the target hardware. The reported final result was:

Result: Wi-Fi OK after CPU RESTART. No manual hardware RST was required.

This confirms that the RM2 V1 retry patch solves the observed warm-restart join failure while preserving the original interrupt-safe reset path.

14. Expected runtime behavior

```
CPU RESTART / watchdog / option-triggered reset
|
v
Original SoftReset() -> hardware watchdog reset
|
v
Normal WebMite boot
|
v
cyw43_arch_init_with_country()
|
v
WebConnect()
|
+--> Join attempt 1
|   |
|   +--> success -> normal network startup
|   |
|   +--> -8 -> poll/settle about 1 s
|
+--> Join attempt 2
|   |
|   +--> success -> normal network startup
|   |
|   +--> -8 -> poll/settle about 1 s
|
+--> Join attempt 3
|   |
|   +--> success -> normal network startup
|   |
|   +--> failure -> original WebMite failure handling
```

15. Scope, limitations, and maintenance notes

- This patch was developed for WebMite RP2350B V6.04.00RC1 with the external RM2/CYW43439 arrangement described in this report.
- The retry is intentionally limited to PICO_ERROR_CONNECT_FAILED. If future SDK versions change error semantics, revalidate the condition before carrying the patch forward.
- The patch should be re-applied manually or re-based when moving to a newer PicoMite/WebMite release; do not copy the entire old WiFi.c over a newer release.
- Keep SoftReset() unmodified unless an upstream, interrupt-safe design explicitly changes the architecture.
- If Wi-Fi later fails for a different reason (bad credentials, RF environment, DHCP configuration, hardware fault), this retry patch is not a substitute for diagnosing that separate cause.

Appendix A. Exact RM2 V1 source diff

```
diff --git a/net/WiFi.c b/net/WiFi.c
index fbe801f..311983d 100644
--- a/net/WiFi.c
+++ b/net/WiFi.c
@@ -257,7 +257,27 @@
     }
     cyw43_wifi_pm(&cyw43_state, CYW43_NO_POWERSAVE_MODE);
     MPrintfString(" connecting to WiFi...\r\n");
-    int connect_result = cyw43_arch_wifi_connect_timeout_ms((char *)Option.SSID, (char *)(*Option.PASSWORD ?
Option.PASSWORD : NULL), (*Option.PASSWORD ? CYW43_AUTH_WPA2_MIXED_PSK : CYW43_AUTH_OPEN), 30000);
+    int connect_result = PICO_ERROR_CONNECT_FAILED;
+
+    // A warm RP2350 reset can leave the first CYW43 join attempt in LINK_FAIL.
+    // Retry that specific failure up to two more times before giving up.
+    for (int rm2_attempt = 1; rm2_attempt <= 3; rm2_attempt++)
+    {
+        connect_result = cyw43_arch_wifi_connect_timeout_ms((char *)Option.SSID, (char *)(*Option.PASSWORD ?
Option.PASSWORD : NULL), (*Option.PASSWORD ? CYW43_AUTH_WPA2_MIXED_PSK : CYW43_AUTH_OPEN), 30000);
+
+        if (connect_result == 0)
+            break;
+
+        if (connect_result != PICO_ERROR_CONNECT_FAILED || rm2_attempt == 3)
+            break;
+
+        // Give the poll backend one second to settle before the next join.
+        for (int rm2_wait = 0; rm2_wait < 100; rm2_wait++)
+        {
+            cyw43_arch_poll();
+            uSec(10000);
+        }
+    }
+    if (connect_result)
+    {
+        MPrintfString("failed to connect.\r\n");
```

Appendix B. Final release identification

Release label	RM2 V1
UF2 filename	WebMiteRP2350V6.04.00RC1_RM2_V1.uf2
SHA-256	E93490EF4AA9D35B0D2E2D1ACB0980806862286D22F3A 18AED2DD60C172A21E7
Modified tracked file	net/WiFi.c
Unmodified critical files	PicoMite.c; misc/External.c; SoftReset()
Field result	Wi-Fi OK after CPU RESTART