PIC16(L)F1938/1939 Silicon Errata and Data Sheet Clarifications

PIC16(L)F1938/1939

www.microchip.com Product Pages: PIC16F1938, PIC16F1939

Introduction

The PIC16(L)F1938/1939 devices that you have received conform functionally to the current device data sheet (DS40001574**D**), except for the anomalies described in this document.

The silicon issues discussed in the following pages are for silicon revisions with the Device and Revision IDs listed in the table below.

The errata described in this document will be addressed in future revisions of the PIC16(L)F1938/1939 silicon.

Note: This document summarizes all silicon errata issues from all revisions of silicon, previous as well as current.

Table 1. Silicon Device Identification

Part Number	DEVICE ID[13:0]			
	DEV[8:0]	REV[4:0]		
		A1	A2	A3
PIC16F1938	10 0011 101	0 0001	0 0010	0 0011
PIC16LF1938	10 0100 101	0 0001	0 0010	0 0011
PIC16F1939	10 0011 110	0 0001	0 0010	0 0011
PIC16LF1939	10 0100 110	0 0001	0 0010	0 0011

Important: Refer to the **Device/Revision ID** section in the device data sheet for more detailed information on Device Identification and Revision IDs for your specific device.

Silicon Issue Summary

Table 2. Silicon Issue Summary

Module	Feature	Item No.	Issue Summary	Affected Revisions		
				A1	A2	A3
Analog-to-Digital Converter (ADC)	ADC Conversion	1.1.1	ADC Conversion May Not Complete	X		
Enhanced Capture Compare PWM (ECCP)	Enhanced PWM	1.2.1	PWM 0% Duty Cycle Direction Change	X		
	Enhanced PWM	1.2.2	PWM 0% Duty Cycle Port Steering	X		
Timer1	Timer1 Gate Toggle Mode	1.3.1	T1 Gate Flip-Flop Does Not Clear	X		
In-Circuit Serial Programming™ (ICSP™)	Low-Voltage Programming	1.4.1	Bulk Erase Not Available with LVP	X		
Oscillator (OSC)	Clock Switching	1.5.1	Clock Switching Can Cause a Single Corrupted Instruction	X	X	
	Oscillator Start-Up Timer (OSTS) Bit	1.5.2	OSTS Bit Remains Set	X	X	
	Oscillator Start-Up Timer (OSTS) Bit	1.5.3	OSTS Bit Remains Clear	X	X	X
Enhanced Universal Synchronous Asynchronous Receiver Transmitter (EUSART)	Auto-Baud Detect	1.6.1	Auto-Baud Detect May Store Incorrect Count Value in the SPBRG Registers	X	X	
Brown-Out Reset (BOR)	Wake-Up from Sleep	1.7.1	Device Resets on Wake-Up from Sleep (PIC16**LF**1938/1939 devices only)	X	X	
Master Synchronous Serial Port (MSSP)	SPI Master Mode	1.8.1	The Buffer Full (BF) Bit or MSSP Interrupt Flag (SSPIF) Bit Becomes Set Half of a SCK Cycle Early	X	X	X

Note: Only those issues indicated in the last column apply to the current silicon revision.

1. Silicon Errata Issues

NOTICE This document summarizes all silicon errata issues from all revisions of silicon, previous and current. Only the issues indicated by the bold font in the following tables apply to the current silicon revision.

1.1 Module: Analog-to-Digital Converter (ADC)

1.1.1 ADC Conversion May Not Complete

An ADC conversion may not complete under these conditions:

1. When F_{OSC} is greater than 8 MHz and it is the clock source used for the ADC converter.
2. The ADC is operating from its dedicated ADCRC oscillator and the device is not in Sleep mode (at any F_{OSC} frequency). When this occurs, the ADC Interrupt Flag (ADIF) does not get set, the GO/$\overline{\text{DONE}}$ bit does not get cleared, and the conversion result does not get loaded into the ADRESH and ADRESL result registers.

Work around

Method 1:

Select the system clock, F_{OSC}, as the ADC clock source and reduce the F_{OSC} frequency to 8 MHz or less when performing ADC conversions.

Method 2:

Select the dedicated ADCRC oscillator as the ADC conversion clock source and perform all conversions with the device in `SLEEP`

Method 3:

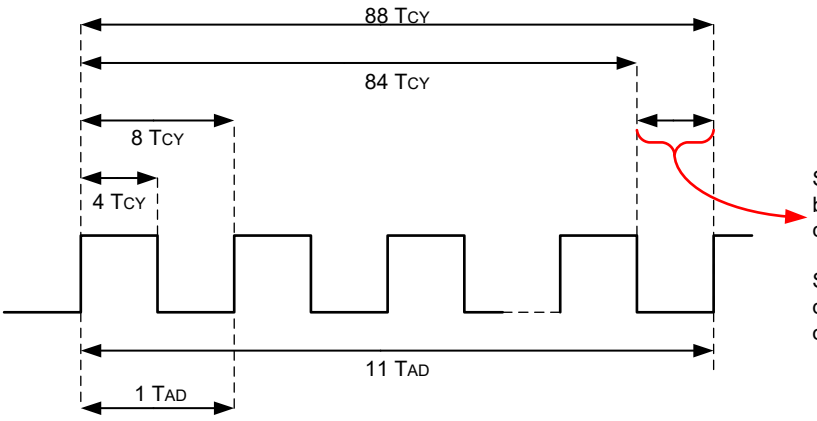
This method is provided if the application cannot use Sleep mode and requires continuous operation at frequencies above 8 MHz. This method requires early termination of an ADC conversion. Provide a fixed time delay in software to stop the ADC conversion manually, after all 10 bits are converted, but before the conversion would complete automatically. The conversion is stopped by clearing the GO/$\overline{\text{DONE}}$ bit in software, and must be cleared during the last ½ T_{AD} cycle, before the conversion would have completed automatically.

In Figure 1-1, 88 instruction cycles will be required to complete the full conversion. Each T_{AD} cycle consists of 8 T_{CY} periods. A fixed delay is provided to stop the A/D conversion after 86 instruction cycles and terminate the conversion at the correct time.

Note: The exact delay time will depend on the T_{AD} divisor (ADCS) selection. The T_{CY} counts shown in Figure 1-1 apply to this example only. Refer to Table 1-1 for examples of the required delay counts for other configurations.

Figure 1-1. Instruction Cycle Delay Calculation Example

FOSC = 32 MHz
TCY = 4/32 MHz = 125 ns
TAD = 1 µs, ADCS = FOSC/32



See the '**ADC Clock Period (TAD) vs Device Operating Frequency**' table in the ADC section of the data sheet.

Table 1-1. Instruction Cycle Delay Counts by T_{AD} Selection

T_{AD}	Instruction Cycle Delay Counts by T_{AD} Selection
$F_{OSC}/64$	172
$F_{OSC}/32$	86
$F_{OSC}/16$	43

Example 1-1. Code Example of Instruction Cycle Delay

```
BSF     ADCON0, ADGO     ; Start ADC Conversion
                         ; Provide 86 instruction cycle delay here
BCF     ADCON0, ADGO     ; Terminate the conversion manually
MOVF    ADRESH, W        ; Read the conversion result
```

Affected Silicon Revisions

A1	A2	**A3**					
X							

1.2 Module: Enhanced Capture Compare PWM (ECCP)

1.2.1 PWM 0% Duty Cycle Direction Change

When the PWM is configured for Full-Bridge mode and the duty cycle is set to 0%, writing the PxM[1:0] bits to change the direction has no effect on PxA and PxC outputs.

Work around

Increase the duty cycle to a value greater than 0% before changing directions.

Affected Silicon Revisions

A1	A2	**A3**					
X							

1.2.2 PWM 0% Duty Cycle Port Steering

In PWM mode, when the duty cycle is set to 0% and the STRxSYNC bit is set, writing the STRxA, STRxB, STRxC, and the STRxD bits to enable/disable steering to port pins has no effect on the outputs.

Work around

Increase the duty cycle to a value greater than 0% before enabling/disabling steering to port pins.

Affected Silicon Revisions

A1	A2	**A3**					
X							

1.3 Module: Timer1

1.3.1 T1 Gate Flip-Flop Does Not Clear

When Timer1 Gate Toggle mode is enabled, clearing the TMR1ON bit does not clear the output value of the flip-flop and hold it clear.

When Timer1 Gate Toggle mode is enabled, it is possible to measure the full-cycle length of a Timer1 gate signal. To perform this function, the Timer1 gate source is routed through a flip-flop that changes state on every incrementing edge of the gate signal. Timer1 Gate Toggle mode is enabled by setting the T1GTM bit of the T1GCON register. When working properly, clearing either the T1GTM bit or the TMR1ON bit would also clear the output value of this flip-flop, and hold it clear. This is done in order to control which edge is being measured.

Work around

Clear the T1GTM bit in the T1GCON register to clear, and hold, clear the output value of the flip-flop.

Affected Silicon Revisions

A1	A2	**A3**					
X							

1.4 Module: In-Circuit Serial Programming™ (ICSP™)

1.4.1 Bulk Erase Not Available with LVP

A bulk erase of the Program Flash Memory or Data Memory cannot be executed in Low-Voltage Programming mode.

Work around

Method 1:

If ICSP Low-Voltage Programming mode is required, use row erases to erase the program memory, as described in the **"Program/Verify Mode"** section of the Programming Specification. Data Memory must be over-written with the desired values.

Method 2:

Use ICSP High-Voltage Programming mode if a bulk erase is required.

Note: Only the bulk erase feature will erase Program or Data Memory if code or data protection is enabled. Method 2 must be used if code or data protection is enabled.

Affected Silicon Revisions

A1	A2	**A3**					
X							

1.5 Module: Oscillator (OSC)

1.5.1 Clock Switching Can Cause a Single Corrupted Instruction

When switching clock sources between INTOSC clock source and an external clock source, one corrupted instruction may be executed after the switch occurs.

This issue does not affect Two-Speed Start-up or the Fail-Safe Clock Monitor operation.

Work around

When switching from an external oscillator clock source, first switch to 16 MHz HFINTOSC. Once running at 16 MHz HFINTOSC, configure IRCF to run at the desired internal oscillator frequency.

When switching from an internal oscillator (INTOSC) to an external oscillator clock source, first switch to HFINTOSC High-Power mode (8 MHz or 16 MHz). Once running from HFINTOSC, switch to the external oscillator clock source.

Affected Silicon Revisions

A1	A2	**A3**					
X	X						

1.5.2 OSTS Bit Remains Set

During the Two-Speed Start-up sequence, the Oscillator Start-up Timer (OST) is enabled to count 1024 clock cycles. After the count is reached, the OSTS bit is set, the system clock is held low until the next falling edge of the external crystal (LP, XT, or HS mode), before switching to the external clock source.

When an external oscillator is configured as the primary clock and Fail-Safe Clock mode is enabled (FCMEN = 1), any of the following conditions will result in the OST failing to restart:

- $\overline{\text{MCLR}}$ Reset
- Wake from Sleep
- Clock change from INTOSC to Primary Clock

This anomaly will manifest itself as a clock failure condition for external oscillators which take longer than the clock failure time-out period to start.

Work around

None.

Affected Silicon Revisions

A1	A2	**A3**					
X	X						

1.5.3 OSTS Bit Remains Clear

When using an external crystal in HS mode and the 4xPLL is enabled, once the Two-Speed Start-up sequence has expired, the OSTS bit remains clear incorrectly indicating that the microcontroller is operating from the internal oscillator. This issue only occurs when the 4xPLL is enabled; when the 4xPLL is disabled, the OSTS bit operates as expected.

Work around

Use the PLL ready flag (PLLR bit of the OSCSTAT register) to determine that the microcontroller is operating from the external crystal. The PLLR bit will only be set when the clock source used by the

PLL is ready; therefore, the PLLR bit will indicate when both the external crystal and the PLL are ready.

Affected Silicon Revisions

A1	A2	**A3**					
X	X	**X**					

1.6 Module: Enhanced Universal Synchronous Asynchronous Receiver Transmitter (EUSART)

1.6.1 Auto-Baud Detect May Store Incorrect Count Value in the SPBRG Registers

When using automatic baud detection (ABDEN), on occasion, an incorrect count value can be stored at the end of the auto-baud detection in the SPBRGH:SPBRGL (SPBRG) registers. The SPBRG value may be off by several counts. This condition happens sporadically when the device clock frequency drifts to a frequency where the SPBRG value oscillates between two different values. The issue is present regardless of the baud rate Configuration bit settings.

Work around

When using auto-baud, it is good practice to always verify the obtained value of SPBRG, to ensure it remains within the application specifications. Two recommended methods are shown below.

For additional auto-baud information, see Technical Brief TB3069, "*Use of Auto-Baud for Reception of LIN Serial Communications Devices: Mid-Range and Enhanced Mid-Range*".

Example 1-2. Method 1 - EUSART Auto-Baud Detect Work Around

```
#define SPBRG_16BIT  *((*int) &SPBRG; // Define location for 16-bit SPBRG value
const int DEFAULT_BAUD = 0x0067;          // Default Auto-Baud value
const int TOL = 0x05;                     // Baud Rate % tolerance
const int MIN_BAUD = DEFAULT_BAUD - TOL;  // Minimum Auto-Baud limit
const int MAX_BAUD = DEFAULT_BAUD + TOL;  // Maximum Auto-Baud limit

ABDEN = 1;                                 // Start Auto-Baud
while(ABDEN);                              // Wait until Auto-Baud completes
if((SPBRG_16BIT > MAX_BAUD) || (SPBRG_16BIT < MIN_BAUD))
{                                         // Compare if value is within limits
    SPBRG_16BIT = DEFAULT_BAUD;           // If out of spec, use DEFAULT_BAUD
}                                         // If in spec, continue using the
                                          // Auto-Baud value in SPBRG
```

Note: In firmware, define default, minimum, and maximum auto-baud (SPBRG) values according to the application requirements. For example, if the application runs at 9600 baud at 16 MHz, then the default SPBRG value would be (assuming 16-bit Asynchronous mode) `0x67`. The minimum and maximum allowed values can be calculated based on the application. In this example, a ±5% tolerance is required, so tolerance is `0x67` * 5% = `0x05`.

Example 1-3. Method 2 - EUSART Auto-Baud Detect Work Around

```
#define SPBRG_16BIT  *((*int)&SPBRG; // Define location for 16-bit SPBRG value
const int DEFAULT_BAUD = 0x0067;          // Default Auto-Baud value
const int TOL = 0x05;                     // Baud rate % tolerance
const int MIN_BAUD = DEFAULT_BAUD - TOL; // Minimum Auto-Baud limit
const int MAX_BAUD = DEFAULT_BAUD + TOL; // Maximum Auto-Baud limit
int Average_Baud;                        // Define Average_Baud variable
int Integrator;                          // Define Integrator variable
Average_Baud = DEFAULT_BAUD;             // Set initial average baud rate
Integrator = DEFAULT_BAUD * 15;          // The running 16 count average

ABDEN = 1;                               // Start Auto-Baud
while(ABDEN);                            // Wait until Auto-Baud completes
```

```
Integrator+ = SPGRB_16BIT;
Average_Baud = Integrator/16;
if((SPBRG_16BIT > MAX_BAUD) || (SPBRG_16BIT < MIN_BAUD))
{                                       // Check if value is within limits
    SPBRG_16BIT = Average_Baud;         // If out of spec, use previous average
}
else                                    // If in spec, calculate the running
{                                       // average but continue using the
    Integrator+ = SPBRG_16BIT;          // Auto-Baud value in SPBRG
    Average_Baud = Integrator/16;
    Integrator- = Average_Baud;
}
```

Note: Similar to Method 1, define default, minimum, and maximum auto-baud (SPBRG) values. In firmware, compute a running average of SPBRG. If the new SPBRG value falls outside the minimum or maximum limits, then use the current running average. For example, if the application runs at 9600 baud at 16 MHz, then the default SPBRG value would be (assuming 16-bit Asynchronous mode) 0x67. The minimum and maximum allowed values can be calculated based on the application. In this example, a ±5% tolerance is required, so tolerance is `0x67` * 5% = `0x05`.

Affected Silicon Revisions

A1	A2	**A3**					
X	X						

1.7 Module: Brown-Out Reset (BOR)

1.7.1 Device Resets on Wake-Up from Sleep (PIC16LF1938/1939 devices only)

This issue affects only the PIC16LF1938/1939 devices. These devices may undergo a BOR Reset when waking up from Sleep and BOR is re-enabled. A BOR Reset may also occur the moment the software BOR is enabled.

Under certain voltage and temperature conditions and when either SBODEN or BOR_NSLEEP is selected, the devices may occasionally reset when waking up from Sleep or BOR is enabled.

Work around

Method 1:

In applications where BOR use is not critical, turn off the BOR in the Configuration Word.

Method 2:

Set the FVREN bit of the FVRCON register. Maintain this bit 'ON' at all times.

Method 3:

When BOR module is needed only during run-time, use the software-enabled BOR by setting the SBODEN option in the Configuration Word. BOR should be turned off by software before Sleep, then follow the sequence below for turning BOR on after wake-up:

1. Wake-up event occurs.
2. Turn on the FVR (FVREN bit of the FVRCON register).
3. Wait until the FVRRDY bit is set.
4. Wait 15 µs after the FVRRDY bit is set.
5. Manually turn on the BOR.

Method 4:

Use the software-enabled BOR as described in Method 3, but use the following sequence:

1. Switch to the internal 32 kHz oscillator immediately before Sleep.

2. Upon wake-up, turn on the FVR (FVREN bit of the FVRCON register).
3. Manually turn on the BOR.
4. Switch the clock back to the preferred clock source.

Note: When using the software BOR follow the steps in Methods 3 or 4 above when enabling BOR for the first time during program execution.

Affected Silicon Revisions

A1	A2	**A3**					
X	X						

1.8 Module: Master Synchronous Serial Port (MSSP)

1.8.1 The Buffer Full (BF) Bit or MSSP Interrupt Flag (SSPIF) Bit Becomes Set Half of a SCK Cycle Early

When the MSSP is used in SPI Master mode and the CKE bit is clear (CKE = 0), the Buffer Full (BF) bit and the MSSP Interrupt Flag (SSPIF) bit becomes set half an SCK cycle early. If the user software immediately reacts to either of the bits being set, a write collision may occur as indicated by the WCOL bit being set.

Work around

To avoid a write collision, one of the following methods should be used:

Method 1:

Add a software delay of one SCK period after detecting the completed transfer (the BF bit or SSPIF bit becomes set) and prior to writing to the SSPBUF register. Verify the WCOL bit is clear after writing to SSPBUF. If the WCOL bit is set, clear the bit in software and rewrite the SSPBUF register.

Method 2:

As part of the MSSP initialization procedure, set the CKE bit (CKE = 1).

Affected Silicon Revisions

A1	A2	**A3**					
X	X	**X**					

2. Data Sheet Clarifications

The following typographic corrections and clarifications are to be noted for the latest version of the device data sheet (DS40001574**D**):

Note:
Corrections are shown in **bold**. Where possible, the original bold text formatting has been removed for clarity.

2.1 V_{OL}/V_{OH} Note 4 Correction

Note 4 has been corrected as shown in the table below (in **bold**):

D080	V_{OL}	Output Low Voltage**(4)**					
		All I/O Pins	—	—	0.6	V	I_{OL} = 8.0 mA, V_{DD} = 5.0V I_{OL} = 6.0 mA, V_{DD} = 3.3V I_{OL} = 1.8 mA, V_{DD} = 1.8V
D090	V_{OH}	Output High Voltage**(4)**					
		All I/O Pins	V_{DD} - 0.7	—	—	V	I_{OH} = 3.5 mA, V_{DD} = 5.0V I_{OH} = 3.0 mA, V_{DD} = 3.3V I_{OH} = 1.0 mA, V_{DD} = 1.8

***** These parameters are characterized but not tested.

† Data in "Typ." column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Notes:

1. In RC oscillator configuration, the OSC1/CLKIN pin is a Schmitt Trigger input. It is not recommended to use an external clock in RC mode.
2. Negative current is defined as current sourced by the pin.
3. The leakage current on the $\overline{MCLR}$ pin is strongly dependent on the applied voltage level. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltages.
4. **Including the OSC2 pin with CLKOUT mode disabled.**

3. Appendix A: Revision History

Doc Rev.	Date	Comments
H	01/2025	Updated the document format to the current Microchip publication standard; module numbers under silicon errata have been renumbered accordingly. Added Data Sheet Clarification 2.1. Other minor corrections.
G	12/2015	Added Module 5.4; removed errata items for Timer0 Gate Source and HFINTOSC Ready/Stable bit that had been erroneously added to the document in a previous version.
F	12/2014	Added Module 8; other minor corrections.
E	01/2013	Added MPLAB X IDE; added silicon revision A3; added Module 7; other minor corrections.
D	02/2012	Updated Table 1; added Modules 5 and 6; removed Data Sheet Clarifications 1 and 2; other minor corrections.
C	12/2010	Updated errata to new format; added silicon revision A2.
B	07/2010	Revised Module 1.1; added Modules 3.2 and 4.1; other minor corrections.
A	05/2010	Initial release of this document.

Microchip Information

Trademarks

The “Microchip” name and logo, the “M” logo, and other names, logos, and brands are registered and unregistered trademarks of Microchip Technology Incorporated or its affiliates and/or subsidiaries in the United States and/or other countries (“Microchip Trademarks”). Information regarding Microchip Trademarks can be found at https://www.microchip.com/en-us/about/legal-information/microchip-trademarks.

ISBN: 978-1-6683-0150-0

Legal Notice

This publication and the information herein may be used only with Microchip products, including to design, test, and integrate Microchip products with your application. Use of this information in any other manner violates these terms. Information regarding device applications is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. Contact your local Microchip sales office for additional support or, obtain additional support at www.microchip.com/en-us/support/design-help/client-support-services.

THIS INFORMATION IS PROVIDED BY MICROCHIP “AS IS”. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE, OR WARRANTIES RELATED TO ITS CONDITION, QUALITY, OR PERFORMANCE.

IN NO EVENT WILL MICROCHIP BE LIABLE FOR ANY INDIRECT, SPECIAL, PUNITIVE, INCIDENTAL, OR CONSEQUENTIAL LOSS, DAMAGE, COST, OR EXPENSE OF ANY KIND WHATSOEVER RELATED TO THE INFORMATION OR ITS USE, HOWEVER CAUSED, EVEN IF MICROCHIP HAS BEEN ADVISED OF THE POSSIBILITY OR THE DAMAGES ARE FORESEEABLE. TO THE FULLEST EXTENT ALLOWED BY LAW, MICROCHIP’S TOTAL LIABILITY ON ALL CLAIMS IN ANY WAY RELATED TO THE INFORMATION OR ITS USE WILL NOT EXCEED THE AMOUNT OF FEES, IF ANY, THAT YOU HAVE PAID DIRECTLY TO MICROCHIP FOR THE INFORMATION.

Use of Microchip devices in life support and/or safety applications is entirely at the buyer’s risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip products:

- Microchip products meet the specifications contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is secure when used in the intended manner, within operating specifications, and under normal conditions.
- Microchip values and aggressively protects its intellectual property rights. Attempts to breach the code protection features of Microchip products are strictly prohibited and may violate the Digital Millennium Copyright Act.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of its code. Code protection does not mean that we are guaranteeing the product is “unbreakable”. Code protection is constantly evolving. Microchip is committed to continuously improving the code protection features of our products.