

PIC18(L)F2X/4XK22

16.2 Clock Accuracy with Asynchronous Operation

The factory calibrates the internal oscillator block output (HFINTOSC). However, the HFINTOSC frequency may drift as VDD or temperature changes, and this directly affects the asynchronous baud rate. Two methods may be used to adjust the baud rate clock, but both require a reference clock source of some kind.

The first (preferred) method uses the OSCTUNE register to adjust the HFINTOSC output. Adjusting the value in the OSCTUNE register allows for fine resolution changes to the system clock source. See Section 2.5 “Internal Clock Modes” for more information.

The other method adjusts the value in the Baud Rate Generator. This can be done automatically with the Auto-Baud Detect feature (see Section 16.3.1 “Auto-Baud Detect”). There may not be fine enough resolution when adjusting the Baud Rate Generator to compensate for a gradual change in the peripheral clock frequency.

REGISTER 16-1: TXSTAX: TRANSMIT STATUS AND CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-1	R/W-0
CSRC	TX9	TXEN ⁽¹⁾	SYNC	SENDB	BRGH	TRMT	TX9D
bit 7							bit 0

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as ‘0’

-n = Value at POR

‘1’ = Bit is set

‘0’ = Bit is cleared

x = Bit is unknown

bit 7	CSRC: Clock Source Select bit <u>Asynchronous mode:</u> Don't care <u>Synchronous mode:</u> 1 = Master mode (clock generated internally from BRG) 0 = Slave mode (clock from external source)
bit 6	TX9: 9-bit Transmit Enable bit 1 = Selects 9-bit transmission 0 = Selects 8-bit transmission
bit 5	TXEN: Transmit Enable bit ⁽¹⁾ 1 = Transmit enabled 0 = Transmit disabled
bit 4	SYNC: EUSART Mode Select bit 1 = Synchronous mode 0 = Asynchronous mode
bit 3	SENDB: Send Break Character bit <u>Asynchronous mode:</u> 1 = Send Sync Break on next transmission (cleared by hardware upon completion) 0 = Sync Break transmission completed <u>Synchronous mode:</u> Don't care
bit 2	BRGH: High Baud Rate Select bit <u>Asynchronous mode:</u> 1 = High speed 0 = Low speed <u>Synchronous mode:</u> Unused in this mode
bit 1	TRMT: Transmit Shift Register Status bit 1 = TSR empty 0 = TSR full
bit 0	TX9D: Ninth bit of Transmit Data Can be address/data bit or a parity bit.

Note 1: SREN/CREN overrides TXEN in Sync mode.

PIC18(L)F2X/4XK22

REGISTER 16-2: RCSTAX: RECEIVE STATUS AND CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0	R-0
SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
bit 7							bit 0

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

- bit 7 **SPEN:** Serial Port Enable bit
 1 = Serial port enabled (configures RXx/DTx and TXx/CKx pins as serial port pins)
 0 = Serial port disabled (held in Reset)
- bit 6 **RX9:** 9-bit Receive Enable bit
 1 = Selects 9-bit reception
 0 = Selects 8-bit reception
- bit 5 **SREN:** Single Receive Enable bit
Asynchronous mode:
 Don't care
Synchronous mode – Master:
 1 = Enables single receive
 0 = Disables single receive
 This bit is cleared after reception is complete.
Synchronous mode – Slave
 Don't care
- bit 4 **CREN:** Continuous Receive Enable bit
Asynchronous mode:
 1 = Enables receiver
 0 = Disables receiver
Synchronous mode:
 1 = Enables continuous receive until enable bit CREN is cleared (CREN overrides SREN)
 0 = Disables continuous receive
- bit 3 **ADDEN:** Address Detect Enable bit
Asynchronous mode 9-bit (RX9 = 1):
 1 = Enables address detection, enable interrupt and load the receive buffer when RSR<8> is set
 0 = Disables address detection, all bytes are received and ninth bit can be used as parity bit
Asynchronous mode 8-bit (RX9 = 0):
 Don't care
- bit 2 **FERR:** Framing Error bit
 1 = Framing error (can be updated by reading RCREGx register and receive next valid byte)
 0 = No framing error
- bit 1 **OERR:** Overrun Error bit
 1 = Overrun error (can be cleared by clearing bit CREN)
 0 = No overrun error
- bit 0 **RX9D:** Ninth bit of Received Data
 This can be address/data bit or a parity bit and must be calculated by user firmware.

PIC18(L)F2X/4XK22

REGISTER 16-3: BAUDCONX: BAUD RATE CONTROL REGISTER

R/W-0	R-1	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
ABDOVF	RCIDL	DTRXP	CKTXP	BRG16	—	WUE	ABDEN
bit 7							bit 0

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

bit 7	ABDOVF: Auto-Baud Detect Overflow bit <u>Asynchronous mode:</u> 1 = Auto-baud timer overflowed 0 = Auto-baud timer did not overflow <u>Synchronous mode:</u> Don't care
bit 6	RCIDL: Receive Idle Flag bit <u>Asynchronous mode:</u> 1 = Receiver is Idle 0 = Start bit has been detected and the receiver is active <u>Synchronous mode:</u> Don't care
bit 5	DTRXP: Data/Receive Polarity Select bit <u>Asynchronous mode:</u> 1 = Receive data (RXx) is inverted (active-low) 0 = Receive data (RXx) is not inverted (active-high) <u>Synchronous mode:</u> 1 = Data (DTx) is inverted (active-low) 0 = Data (DTx) is not inverted (active-high)
bit 4	CKTXP: Clock/Transmit Polarity Select bit <u>Asynchronous mode:</u> 1 = Idle state for transmit (TXx) is low 0 = Idle state for transmit (TXx) is high <u>Synchronous mode:</u> 1 = Data changes on the falling edge of the clock and is sampled on the rising edge of the clock 0 = Data changes on the rising edge of the clock and is sampled on the falling edge of the clock
bit 3	BRG16: 16-bit Baud Rate Generator bit 1 = 16-bit Baud Rate Generator is used (SPBRGHx:SPBRGx) 0 = 8-bit Baud Rate Generator is used (SPBRGx)
bit 2	Unimplemented: Read as '0'
bit 1	WUE: Wake-up Enable bit <u>Asynchronous mode:</u> 1 = Receiver is waiting for a falling edge. No character will be received but RCxIF will be set on the falling edge. WUE will automatically clear on the rising edge. 0 = Receiver is operating normally <u>Synchronous mode:</u> Don't care
bit 0	ABDEN: Auto-Baud Detect Enable bit <u>Asynchronous mode:</u> 1 = Auto-Baud Detect mode is enabled (clears when auto-baud is complete) 0 = Auto-Baud Detect mode is disabled <u>Synchronous mode:</u> Don't care