**Table 32.4. Packet Handler Registers**

| Add | R/W | Function/Description | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | POR Def |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 30 | R/W | Data Access Control | enpacrx | lsbfrst | crcdonly | skip2ph | enpactx | encrc | crc[1] | crc[0] | 8Dh |
| 31 | R | EzMAC status | 0 | rxcrc1 | pksrch | pkrx | pkvalid | crcerror | pktx | pksent | — |
| 32 | R/W | Header Control 1 | bcen[3:0] | | | | hdch[3:0] | | | | 0Ch |
| 33 | R/W | Header Control 2 | skipsyn | hdlen[2] | hdlen[1] | hdlen[0] | fixpklen | synclen[1] | synclen[0] | prealen[8] | 22h |
| 34 | R/W | Preamble Length | prealen[7] | prealen[6] | prealen[5] | prealen[4] | prealen[3] | prealen[2] | prealen[1] | prealen[0] | 08h |
| 35 | R/W | Preamble Detection Control | preath[4] | preath[3] | preath[2] | preath[1] | preath[0] | rssi_off[2] | rssi_off[1] | rssi_off[0] | 2Ah |
| 36 | R/W | Sync Word 3 | sync[31] | sync[30] | sync[29] | sync[28] | sync[27] | sync[26] | sync[25] | sync[24] | 2Dh |
| 37 | R/W | Sync Word 2 | sync[23] | sync[22] | sync[21] | sync[20] | sync[19] | sync[18] | sync[17] | sync[16] | D4h |
| 38 | R/W | Sync Word 1 | sync[15] | sync[14] | sync[13] | sync[12] | sync[11] | sync[10] | sync[9] | sync[8] | 00h |
| 39 | R/W | Sync Word 0 | sync[7] | sync[6] | sync[5] | sync[4] | sync[3] | sync[2] | sync[1] | sync[0] | 00h |
| 3A | R/W | Transmit Header 3 | txhd[31] | txhd[30] | txhd[29] | txhd[28] | txhd[27] | txhd[26] | txhd[25] | txhd[24] | 00h |
| 3B | R/W | Transmit Header 2 | txhd[23] | txhd[22] | txhd[21] | txhd[20] | txhd[19] | txhd[18] | txhd[17] | txhd[16] | 00h |
| 3C | R/W | Transmit Header 1 | txhd[15] | txhd[14] | txhd[13] | txhd[12] | txhd[11] | txhd[10] | txhd[9] | txhd[8] | 00h |
| 3D | R/W | Transmit Header 0 | txhd[7] | txhd[6] | txhd[5] | txhd[4] | txhd[3] | txhd[2] | txhd[1] | txhd[0] | 00h |
| 3E | R/W | Transmit Packet Length | pklen[7] | pklen[6] | pklen[5] | pklen[4] | pklen[3] | pklen[2] | pklen[1] | pklen[0] | 00h |
| 3F | R/W | Check Header 3 | chhd[31] | chhd[30] | chhd[29] | chhd[28] | chhd[27] | chhd[26] | chhd[25] | chhd[24] | 00h |
| 40 | R/W | Check Header 2 | chhd[23] | chhd[22] | chhd[21] | chhd[20] | chhd[19] | chhd[18] | chhd[17] | chhd[16] | 00h |
| 41 | R/W | Check Header 1 | chhd[15] | chhd[14] | chhd[13] | chhd[12] | chhd[11] | chhd[10] | chhd[9] | chhd[8] | 00h |
| 42 | R/W | Check Header 0 | chhd[7] | chhd[6] | chhd[5] | chhd[4] | chhd[3] | chhd[2] | chhd[1] | chhd[0] | 00h |
| 43 | R/W | Header Enable 3 | hden[31] | hden[30] | hden[29] | hden[28] | hden[27] | hden[26] | hden[25] | hden[24] | FFh |
| 44 | R/W | Header Enable 2 | hden[23] | hden[22] | hden[21] | hden[20] | hden[19] | hden[18] | hden[17] | hden[16] | FFh |
| 45 | R/W | Header Enable 1 | hden[15] | hden[14] | hden[13] | hden[12] | hden[11] | hden[10] | hden[9] | hden[8] | FFh |
| 46 | R/W | Header Enable 0 | hden[7] | hden[6] | hden[5] | hden[4] | hden[3] | hden[2] | hden[1] | hden[0] | FFh |
| 47 | R | Received Header 3 | rxhd[31] | rxhd[30] | rxhd[29] | rxhd[28] | rxhd[27] | rxhd[26] | rxhd[25] | rxhd[24] | — |
| 48 | R | Received Header 2 | rxhd[23] | rxhd[22] | rxhd[21] | rxhd[20] | rxhd[19] | rxhd[18] | rxhd[17] | rxhd[16] | — |
| 49 | R | Received Header 1 | rxhd[15] | rxhd[14] | rxhd[13] | rxhd[12] | rxhd[11] | rxhd[10] | rxhd[9] | rxhd[8] | — |
| 4A | R | Received Header 0 | rxhd[7] | rxhd[6] | rxhd[5] | rxhd[4] | rxhd[3] | rxhd[2] | rxhd[1] | rxhd[0] | — |
| 4B | R | Received Packet Length | rxplen[7] | rxplen[6] | rxplen[5] | rxplen[4] | rxplen[3] | rxplen[2] | rxplen[1] | rxplen[0] | — |

### 32.6.5. Data Whitening, Manchester Encoding, and CRC

Data whitening can be used to avoid extended sequences of 0s or 1s in the transmitted data stream to achieve a more uniform spectrum. When enabled, the payload data bits are XORed with a pseudo-random sequence output from the built-in PN9 generator. The generator is initialized at the beginning of the payload. The receiver recovers the original data by repeating this operation. Manchester encoding can be used to ensure a dc-free transmission and good synchronization properties. When Manchester encoding is used, the effective datarate is unchanged but the actual datarate (preamble length, etc.) is doubled due to the nature of the encoding. The effective datarate when using Manchester encoding is limited to 128 kbps. The implementation of Manchester encoding is shown in Figure 32.18. Data whitening and Manchester encoding can be selected with "Register 70h. Modulation Mode Control 1". The CRC is configured via "Register 30h. Data Access Control". Figure 32.17 demonstrates the portions of the packet which have Manchester encoding, data whitening, and CRC applied. CRC can be applied to only the data portion of the packet or to the data, packet length and header fields. Figure 32.18 provides an example of how the Manchester encoding is done and also the use of the Manchester invert (enmaniv) function.

Not Recommended for New Designs

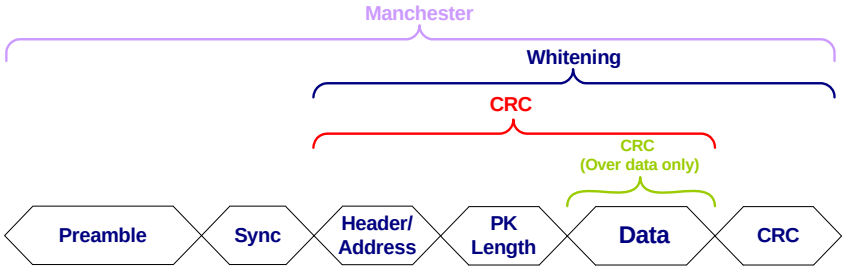



**Figure 32.17. Operation of Data Whitening, Manchester Encoding, and CRC**

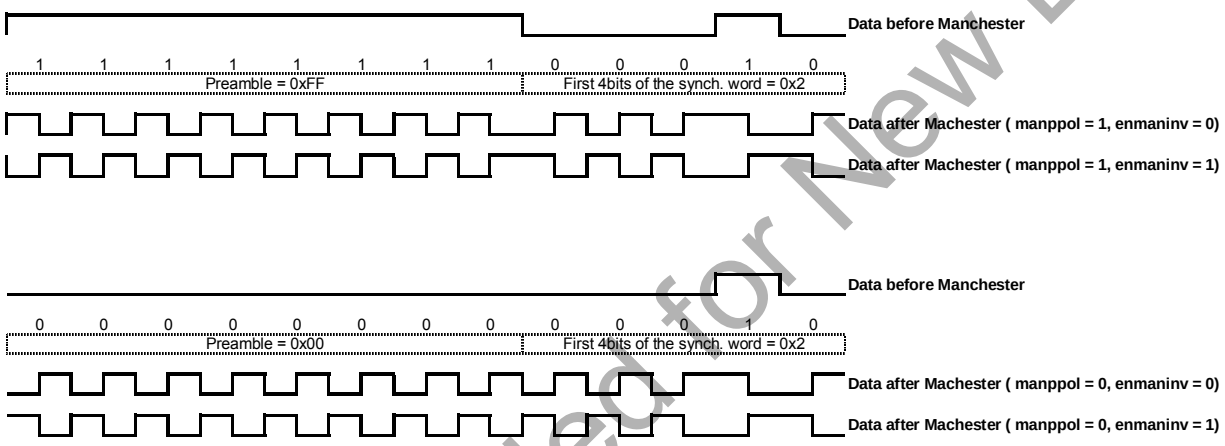



**Figure 32.18. Manchester Coding Example**

### 32.6.6. Preamble Detector

The EZRadioPRO transceiver has integrated automatic preamble detection. The preamble length is configurable from 1–255 bytes using the prealen[7:0] field in "Register 33h. Header Control 2" and "Register 34h. Preamble Length", as described in “32.6.2. Packet Configuration” . The preamble detection threshold, preath[4:0] as set in "Register 35h. Preamble Detection Control 1", is in units of 4 bits. The preamble detector searches for a preamble pattern with a length of preath[4:0].

If a false preamble detect occurs, the receiver will continuing searching for the preamble when no sync word is detected. Once preamble is detected (false or real) then the part will then start searching for sync. If no sync occurs then a timeout will occur and the device will initiate search for preamble again. The timeout period is defined as the sync word length plus four bits and will start after a non-preamble pattern is recognized after a valid preamble detection. The preamble detector output may be programmed onto one of the GPIO or read in the interrupt status registers.

### 32.6.7. Preamble Length

The preamble detection threshold determines the number of valid preamble bits the radio must receive to qualify a valid preamble. The preamble threshold should be adjusted depending on the nature of the application. The required preamble length threshold will depend on when receive mode is entered in relation to the start of the transmitted packet and the length of the transmit preamble. With a shorter than recommended preamble detection threshold the probability of false detection is directly related to how long the receiver operates on noise before the transmit preamble is received. False detection on noise may cause the actual packet to be missed. The preamble detection threshold is programmed in register 35h. For most applications with a preamble length longer than 32 bits the default value of 20 is recommended for the pre-

Not Recommended for New Designs

amble detection threshold. A shorter Preamble Detection Threshold may be chosen if occasional false detections may be tolerated. When antenna diversity is enabled a 20-bit preamble detection threshold is recommended. When the receiver is synchronously enabled just before the start of the packet, a shorter preamble detection threshold may be used. Table 32.5 demonstrates the recommended preamble detection threshold and preamble length for various modes.

It is possible to use the transceiver in a raw mode without the requirement for a 010101... preamble. Contact customer support for further details.

**Table 32.5. Minimum Receiver Settling Time**

| Mode | Approximate Receiver Settling Time | Recommended Preamble Length with 8-Bit Detection Threshold | Recommended Preamble Length with 20-Bit Detection Threshold |
|---|---|---|---|
| (G)FSK AFC Disabled | 1 byte | 20 bits | 32 bits |
| (G)FSK AFC Enabled | 2 byte | 28 bits | 40 bits |
| (G)FSK AFC Disabled +Antenna Diversity Enabled | 1 byte | — | 64 bits |
| (G)FSK AFC Enabled +Antenna Diversity Enabled | 2 byte | — | 8 byte |
| OOK | 2 byte | 3 byte | 4 byte |
| OOK + Antenna Diversity Enabled | 8 byte | — | 8 byte |

**Note:** The recommended preamble length and preamble detection threshold listed above are to achieve 0% PER. They may be shortened when occasional packet errors are tolerable.

### 32.6.8. Invalid Preamble Detector

When scanning channels in a frequency hopping system it is desirable to determine if a channel is valid in the minimum amount of time. The preamble detector can output an invalid preamble detect signal. which can be used to identify the channel as invalid. After a configurable time set in Register 60h[7:4], an invalid preamble detect signal is asserted indicating an invalid channel. The period for evaluating the signal for invalid preamble is defined as (inv_pre_th[3:0] x 4) x Bit Rate Period. The preamble detect and invalid preamble detect signals are available in "Register 03h. Interrupt/Status 1" and “Register 04h. Interrupt/Status 2.”

### 32.6.9. Synchronization Word Configuration

The synchronization word length for both TX and RX can be configured in Reg 33h, synclen[1:0]. The expected or transmitted sync word can be configured from 1 to 4 bytes as defined below:

- synclen[1:0] = 00—Expected/Transmitted Synchronization Word (sync word) 3.
- synclen[1:0] = 01—Expected/Transmitted Synchronization Word 3 first, followed by sync word 2.
- synclen[1:0] = 10—Expected/Transmitted Synchronization Word 3 first, followed by sync word 2, followed by sync word 1.
- synclen[1:0] = 1—Send/Expect Synchronization Word 3 first, followed by sync word 2, followed by sync word 1, followed by sync word 0.

The sync is transmitted or expected in the following sequence: sync 3→sync 2→sync 1→sync 0. The sync word values can be programmed in Registers 36h–39h. After preamble detection the part will search for sync for a fixed period of time. If a sync is not recognized in this period then a timeout will occur and the search for preamble will be re-initiated. The timeout period after preamble detections is defined as the value programmed into the sync word length plus four additional bits.

Not Recommended for New Designs

### 32.6.10. Receive Header Check

The header check is designed to support 1–4 bytes and broadcast headers. The header length needs to be set in register 33h, hdlen[2:0]. The headers to be checked need to be set in register 32h, hdch[3:0]. For instance, there can be four bytes of header in the packet structure but only one byte of the header is set to be checked (i.e., header 3). For the headers that are set to be checked, the expected value of the header should be programmed in chhd[31:0] in Registers 3F–42. The individual bits within the selected bytes to be checked can be enabled or disabled with the header enables, hden[31:0] in Registers 43–46. For example, if you want to check all bits in header 3 then hden[31:24] should be set to FF but if only the last 4 bits are desired to be checked then it should be set to 00001111 (0F). Broadcast headers can also be programmed by setting bcen[3:0] in Register 32h. For broadcast header check the value may be either “FFh” or the value stored in the Check Header register. A logic equivalent of the header check for Header 3 is shown in Figure 32.19. A similar logic check will be done for Header 2, Header 1, and Header 0 if enabled.

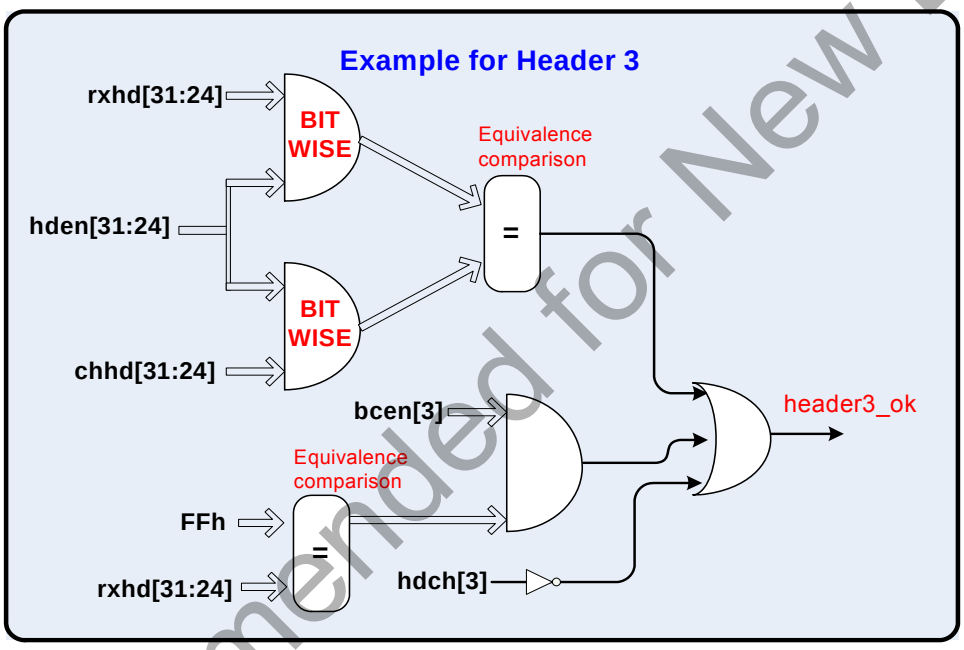



**Figure 32.19. Header**

### 32.6.11. TX Retransmission and Auto TX

The transceiver is capable of automatically retransmitting the last packet loaded in the TX FIFO. Automatic retransmission is set by entering the TX state with the txon bit without reloading the TX FIFO. This feature is useful for beacon transmission or when retransmission is required due to the absence of a valid acknowledgement. Only packets that fit completely in the TX FIFO can be automatically retransmitted.

An automatic transmission function is available, allowing the radio to automatically start or stop a transmission depending on the amount of data in the TX FIFO.

When autotx is set in “Register 08. Operating & Function Control 2", the transceiver will automatically enter the TX state when the TX FIFO almost full threshold is exceeded. Packets will be transmitted according to the configured packet length. To stop transmitting, clear the packet sent or TX FIFO almost empty interrupts must be cleared by reading register.

Not Recommended for New Designs

## 32.7. RX Modem Configuration

A Microsoft Excel parameter calculator or Wireless Development Suite (WDS) calculator is provided to determine the proper settings for the modem. The calculator can be found on www.silabs.com or on the CD provided with the demo kits. An application note is available to describe how to use the calculator and to provide advanced descriptions of the modem settings and calculations.

### 32.7.1. Modem Settings for FSK and GFSK

The modem performs channel selection and demodulation in the digital domain. The channel filter bandwidth is configurable from 2.6 to 620 kHz. The receiver data-rate, modulation index, and bandwidth are set via registers 1C–25h. The modulation index is equal to 2 times the peak deviation divided by the data rate (Rb).

When Manchester coding is disabled, the required channel filter bandwidth is calculated as BW = 2Fd + Rb where Fd is the frequency deviation and Rb is the data rate.

## 32.8. Auxiliary Functions

The EZRadioPRO has some auxiliary functions that duplicate the directly accessible MCU peripherals: ADC, temperature sensor, and 32 kHz oscillator. These auxiliary functions are retained primarily for compatibility with the Si4430/1/2. The directly accessed MCU peripherals typically provide lower system current consumption and better analog performance. However some of these EZRadioPRO auxiliary functions offer features not directly duplicated in the MCU directly accessed peripherals, such as the Low Duty Cycle Mode operation.

### 32.8.1. Smart Reset

The EZRadioPRO transceiver contains an enhanced integrated SMART RESET or POR circuit. The POR circuit contains both a classic level threshold reset as well as a slope detector POR. This reset circuit was designed to produce a reliable reset signal under any circumstances. Reset will be initiated if any of the following conditions occur:

- Initial power on, $V_{DD}$ starts from gnd: reset is active till $V_{DD}$ reaches $V_{RR}$ (see table);
- When $V_{DD}$ decreases below $V_{LD}$ for any reason: reset is active till $V_{DD}$ reaches $V_{RR}$;
- A software reset via “Register 08h. Operating Mode and Function Control 2”: reset is active for time $T_{SWRST}$
- On the rising edge of a $V_{DD}$ glitch when the supply voltage exceeds the following time functioned limit:

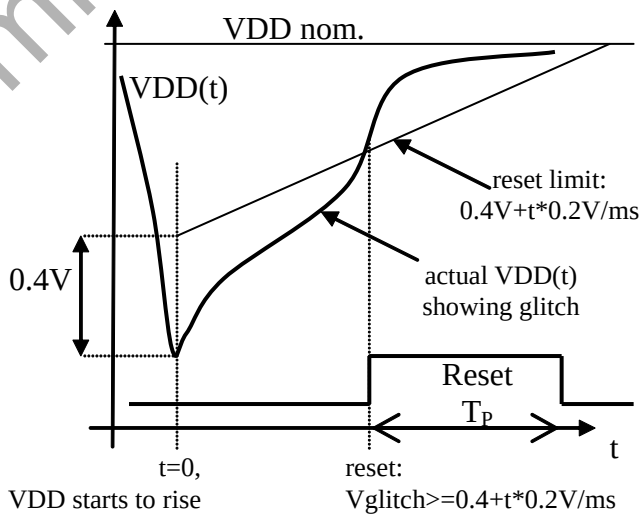



**Figure 32.20. POR Glitch Parameters**

Not Recommended for New Designs

**Table 32.6. POR Parameters**

| Parameter | Symbol | Comment | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| Release Reset Voltage | VRR | | 0.85 | 1.3 | 1.75 | V |
| Power-On $V_{DD}$ Slope | SVDD | tested $V_{DD}$ slope region | 0.03 | — | 300 | V/ms |
| Low $V_{DD}$ Limit | VLD | VLD<VRR is guaranteed | 0.7 | 1 | 1.3 | V |
| Software Reset Pulse | TSWRST | | 50 | — | 470 | us |
| Threshold Voltage | VTSD | | — | 0.4 | — | V |
| Reference Slope | k | | — | 0.2 | | V/ms |
| $V_{DD}$ Glitch Reset Pulse | TP | Also occurs after SDN, and initial power on | 5 | 16 | 25 | ms |

The reset will initialize all registers to their default values. The reset signal is also available for output and use by the microcontroller by using the default setting for GPIO_0. The inverted reset signal is available by default on GPIO_1.

### 32.8.2. Output Clock

The 30 MHz crystal oscillator frequency is divided down internally and may be output on GPIO2. This feature is useful to lower BOM cost by using only one crystal in the system. The output clock on GPIO2 may be routed to the XTAL2 input to provide a synchronized clock source between the MCU and the EZRadioPRO peripheral. The output clock frequency is selectable from one of 8 options, as shown below. Except for the 32.768 kHz option, all other frequencies are derived by dividing the crystal oscillator frequency. The 32.768 kHz clock signal is derived from an internal RC oscillator or an external 32 kHz crystal. The default setting for GPIO2 is to output the clock signal with a frequency of 1 MHz.

| Add | R/W | Function/ Description | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | POR Def. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0A | R/W | Output Clock | | | clkt[1] | clkt[0] | enlfc | mclk[2] | mclk[1] | mclk[0] | 06h |

| mclk[2:0] | Modulation Source |
|---|---|
| 000 | 30 MHz |
| 001 | 15 MHz |
| 010 | 10 MHz |
| 011 | 4 MHz |
| 100 | 3 MHz |
| 101 | 2 MHz |
| 110 | 1 MHz |
| 111 | 32.768 kHz |

Since the crystal oscillator is disabled in SLEEP mode in order to save current, the low-power 32.768 kHz clock can be automatically switched to become the output clock. This feature is called enable low frequency clock and is enabled by the enlfc bit in “Register 0Ah. Microcontroller Output Clock." When enlfc = 1 and the chip is in SLEEP mode then the 32.768 kHz clock will be provided regardless of the setting of mclk[2:0]. For example, if mclk[2:0] = 000, 30 MHz will be provided through the GPIO output pin in all IDLE,

Not Recommended for New Designs

TX, or RX states. When the chip enters SLEEP mode, the output clock will automatically switch to 32.768 kHz from the RC oscillator or 32.768 XTAL.

Another available feature for the output clock is the clock tail, clkt[1:0] in “Register 0Ah. Microcontroller Output Clock." If the low frequency clock feature is not enabled (enlfc = 0), then the output is disabled in SLEEP mode. Setting the clkt[1:0] field will provide additional cycles of the output clock before it shuts off.

| clkt[1:0] | Modulation Source |
|---|---|
| 00 | 0 cycles |
| 01 | 128 cycles |
| 10 | 256 cycles |
| 11 | 512 cycles |

If an interrupt is triggered, the output clock will remain enabled regardless of the selected mode. As soon as the interrupt is read the state machine will then move to the selected mode. The minimum current consumption will not be achieved until the interrupt is read. For instance, if the EZRadioPRO peripheral is commanded to SLEEP mode but an interrupt has occurred the 30 MHz XTAL will not be disabled until the interrupt has been cleared.

### 32.8.3. General Purpose ADC

The EZRadioPRO peripheral includes an 8-bit SAR ADC independent of ADC0. It may be used for general purpose analog sampling, as well as for digitizing the EZRadioPRO temperature sensor reading. In most cases, the ADC0 subsystem directly accessible from the MCU will be preferred over the ADC embedded inside the EZRadioPRO peripheral. Registers 0Fh "ADC Configuration", 10h "Sensor Offset" and 4Fh "Amplifier Offset" can be used to configure the ADC operation. Details of these registers are in “AN440: EZRadioPRO Detailed Register Descriptions.”

Every time an ADC conversion is desired, bit 7 "adcstart/adcdone" in Register 0Fh “ADC Configuration” must be set to 1. The conversion time for the ADC is 350 µs. After the ADC conversion is done and the adcdone signal is showing 1, then the ADC value may be read out of “Register 11h: ADC Value." When the ADC is doing its conversion, the adcstart/adcdone bit will read 0. When the ADC has finished its conversion, the bit will be set to 1. A new ADC conversion can be initiated by writing a 1 to the adcstart/adcdone bit.

The architecture of the ADC is shown in Figure 32.21. The signal and reference inputs of the ADC are selected by adcsel[2:0] and adcref[1:0] in register 0Fh “ADC Configuration”, respectively. The default setting is to read out the temperature sensor using the bandgap voltage (VBG) as reference. With the VBG reference the input range of the ADC is from 0–1.02 V with an LSB resolution of 4 mV (1.02/255). Changing the ADC reference will change the LSB resolution accordingly.

A differential multiplexer and amplifier are provided for interfacing external bridge sensors. The gain of the amplifier is selectable by adcgain[1:0] in Register 0Fh. The majority of sensor bridges have supply voltage (VDD) dependent gain and offset. The reference voltage of the ADC can be changed to either $V_{DD}/2$ or $V_{DD}/3$. A programmable $V_{DD}$ dependent offset voltage can be added using soffs[3:0] in register 10h.

Not Recommended for New Designs

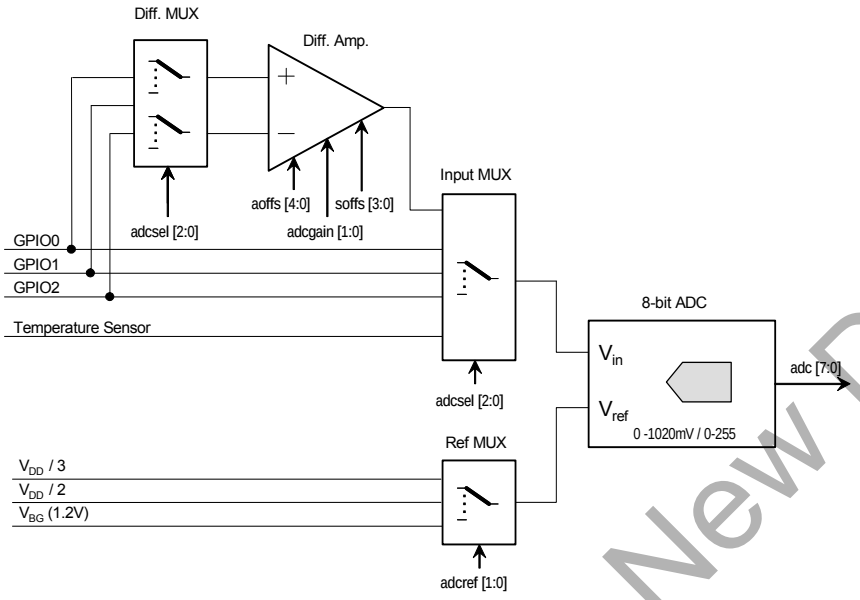



**Figure 32.21. General Purpose ADC Architecture**

| Add | R/W | Function/ Description | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | POR Def. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0F | R/W | ADC Configuration | adcstart/adcdone | adcsel[2] | adcsel[1] | adcsel[0] | adcref[1] | adcref[0] | adcgain[1] | adcgain[0] | 00h |
| 10 | R/W | Sensor Offset | | | | | soffs[3] | soffs[2] | soffs[1] | soffs[0] | 00h |
| 11 | R | ADC Value | adc[7] | adc[6] | adc[5] | adc[4] | adc[3] | adc[2] | adc[1] | adc[0] | — |

### 32.8.4. Temperature Sensor

The EZRadioPRO peripheral includes an integrated on-chip analog temperature sensor independent of the temperature sensor associated with ADC0. The temperature sensor will be automatically enabled when the temperature sensor is selected as the input of the EZRadioPRO ADC or when the analog temp voltage is selected on the analog test bus. The temperature sensor value may be digitized using the EZRadioPRO general-purpose ADC and read out through "Register 10h. ADC Sensor Amplifier Offset." The range of the temperature sensor is configurable. Table 32.7 lists the settings for the different temperature ranges and performance.

To use the Temp Sensor:

1. Set the input for ADC to the temperature sensor, "Register 0Fh. ADC Configuration"—adcsel[2:0] = 000
2. Set the reference for ADC, "Register 0Fh. ADC Configuration"—adcref[1:0] = 00
3. Set the temperature range for ADC, "Register 12h. Temperature Sensor Calibration"—tsrange[1:0]
4. Set entsoffs = 1, "Register 12h. Temperature Sensor Calibration"
5. Trigger ADC reading, "Register 0Fh. ADC Configuration"—adcstart = 1
6. Read temperature value—Read contents of "Register 11h. ADC Value"

Not Recommended for New Designs

| Add | R/W | Function/ Description | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | POR Def. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 12 | R/W | Temperature Sensor Control | tsrange[1] | tsrange[0] | entsoffs | entstrim | tstrim[3] | tstrim[2] | vbgtrim[1] | vbgtrim[0] | 20h |
| 13 | R/W | Temperature Value Offset | tvoffs[7] | tvoffs[6] | tvoffs[5] | tvoffs[4] | tvoffs[3] | tvoffs[2] | tvoffs[1] | tvoffs[0] | 00h |

**Table 32.7. Temperature Sensor Range**

| entoff | tsrange[1] | tsrange[0] | Temp. range | Unit | Slope | ADC8 LSB |
|---|---|---|---|---|---|---|
| 1 | 0 | 0 | –64 … 64 | °C | 8 mV/°C | 0.5 °C |
| 1 | 0 | 1 | –64 … 192 | °C | 4 mV/°C | 1 °C |
| 1 | 1 | 0 | 0 … 128 | °C | 8 mV/°C | 0.5 °C |
| 1 | 1 | 1 | –40 … 216 | °F | 4 mV/°F | 1 °F |
| 0* | 1 | 0 | 0 … 341 | °K | 3 mV/°K | 1.333 °K |

**Note:** Absolute temperature mode, no temperature shift. This mode is only for test purposes. POR value of EN_TOFF is 1.

The slope of the temperature sensor is very linear and monotonic. For absolute accuracy better than 10 °C calibration is necessary. The temperature sensor may be calibrated by setting entsoffs = 1 in “Register 12h. Temperature Sensor Control” and setting the offset with the tvoffs[7:0] bits in “Register 13h. Temperature Value Offset.” This method adds a positive offset digitally to the ADC value that is read in “Register 11h. ADC Value.” The other method of calibration is to use the tstrim which compensates the analog circuit. This is done by setting entstrim = 1 and using the tstrim[2:0] bits to offset the temperature in “Register 12h. Temperature Sensor Control.” With this method of calibration, a negative offset may be achieved. With both methods of calibration better than ±3 °C absolute accuracy may be achieved.

The different ranges for the temperature sensor and ADC8 are demonstrated in Figure 32.22. The value of the ADC8 may be translated to a temperature reading by ADC8Value x ADC8 LSB + Lowest Temperature in Temp Range. For instance for a tsrange = 00, Temp = ADC8Value x 0.5 – 64.

Not Recommended for New Designs

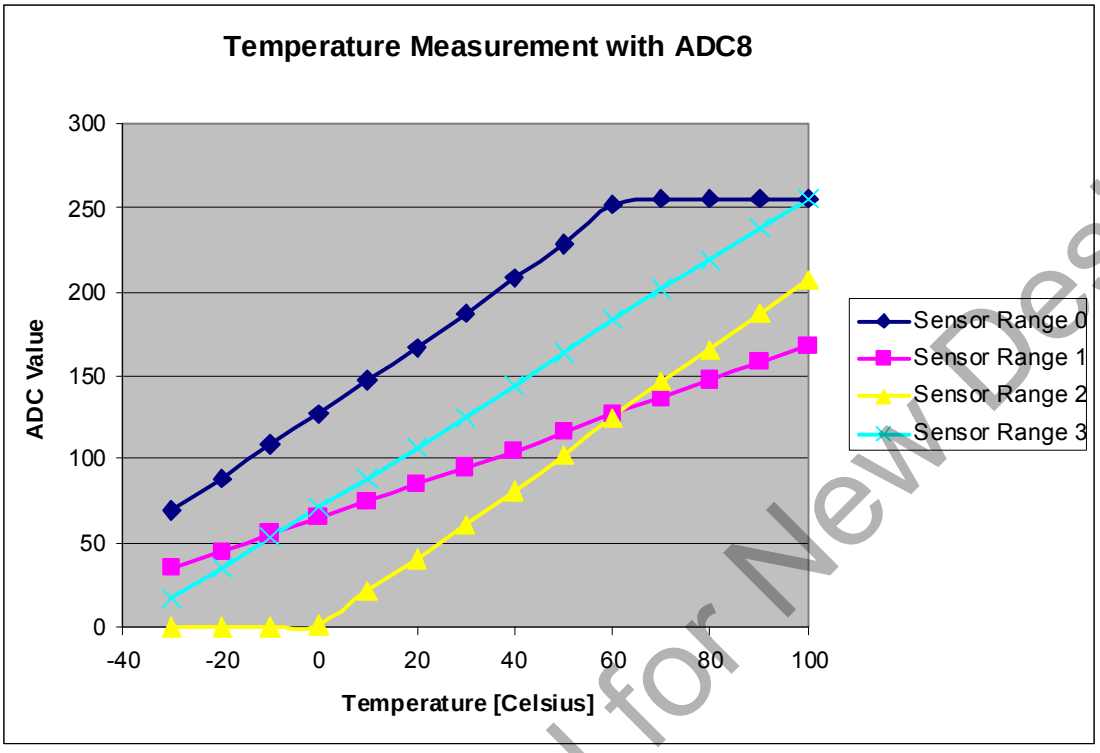



**Figure 32.22. Temperature Ranges using ADC8**

### 32.8.5. Low Battery Detector

The Low Battery Detector (LBD) feature of the EZRadioPRO peripheral is not supported in the Si102x/3x. Use ADC0 instead. Refer to “5. SAR ADC with 16-bit Auto-Averaging Accumulator and Autonomous Low Power Burst Mode” on page 77 for details.

### 32.8.6. Wake-Up Timer and 32 kHz Clock Source

The EZRadioPRO peripheral contains an integrated wake-up timer independent of the SmaRTClock which can be used to periodically wake the chip from SLEEP mode using the interrupt pin. The wake-up timer runs from the internal 32.768 kHz RC Oscillator. The wake-up timer can be configured to run when in SLEEP mode. If enwt = 1 in "Register 07h. Operating Mode and Function Control 1" when entering SLEEP mode, the wake-up timer will count for a time specified defined in Registers 14–16h, "Wake Up Timer Period". At the expiration of this period an interrupt will be generated on the nIRQ pin if this interrupt is enabled. The software will then need to verify the interrupt by reading the Registers 03h–04h, "Interrupt Status 1 & 2". The wake-up timer value may be read at any time by the wtv[15:0] read only registers 17h–18h.

The formula for calculating the Wake-Up Period is the following:

$$WUT = \frac{32 \times M \times 2^{R}}{32.768} ms$$

| WUT Register | Description |
|---|---|
| wtr[4:0] | R Value in Formula |
| wtm[15:0] | M Value in Formula |

Use of the D variable in the formula is only necessary if finer resolution is required than can be achieved by using the R value.

Not Recommended for New Designs

| Add | R/W | Function/Description | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | POR Def. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 14 | R/W | Wake-Up Timer Period 1 | | | | wtr[4] | wtr[3] | wtr[2] | wtr[1] | wtr[0] | 03h |
| 15 | R/W | Wake-Up Timer Period 2 | wtm[15] | wtm[14] | wtm[13] | wtm[12] | wtm[11] | wtm[10] | wtm[9] | wtm[8] | 00h |
| 16 | R/W | Wake-Up Timer Period 3 | wtm[7] | wtm[6] | wtm[5] | wtm[4] | wtm[3] | wtm[2] | wtm[1] | wtm[0] | 00h |
| 17 | R | Wake-Up Timer Value 1 | wtv[15] | wtv[14] | wtv[13] | wtv[12] | wtv[11] | wtv[10] | wtv[9] | wtv[8] | — |
| 18 | R | Wake-Up Timer Value 2 | wtv[7] | wtv[6] | wtv[5] | wtv[4] | wtv[3] | wtv[2] | wtv[1] | wtv[0] | — |

There are two different methods for utilizing the wake-up timer (WUT) depending on if the WUT interrupt is enabled in “Register 06h. Interrupt Enable 2.” If the WUT interrupt is enabled then nIRQ pin will go low when the timer expires. The chip will also change state so that the 30 MHz XTAL is enabled so that the microcontroller clock output is available for the microcontroller to use to process the interrupt. The other method of use is to not enable the WUT interrupt and use the WUT GPIO setting. In this mode of operation the chip will not change state until commanded by the microcontroller. The different modes of operating the WUT and the current consumption impacts are demonstrated in Figure 32.23.

A 32 kHz XTAL may also be used for better timing accuracy. By setting the x32 ksel bit in Register 07h "Operating & Function Control 1", GPIO0 is automatically reconfigured so that an external 32 kHz XTAL may be connected to this pin. In this mode, the GPIO0 is extremely sensitive to parasitic capacitance, so only the XTAL should be connected to this pin with the XTAL physically located as close to the pin as possible. Once the x32 ksel bit is set, all internal functions such as WUT, microcontroller clock, and LDC mode will use the 32 kHz XTAL and not the 32 kHz RC oscillator.

The 32 kHz XTAL accuracy is comprised of both the XTAL parameters and the internal circuit. The XTAL accuracy can be defined as the XTAL initial error + XTAL aging + XTAL temperature drift + detuning from the internal oscillator circuit. The error caused by the internal circuit is typically less than 10 ppm.

Not Recommended for New Designs

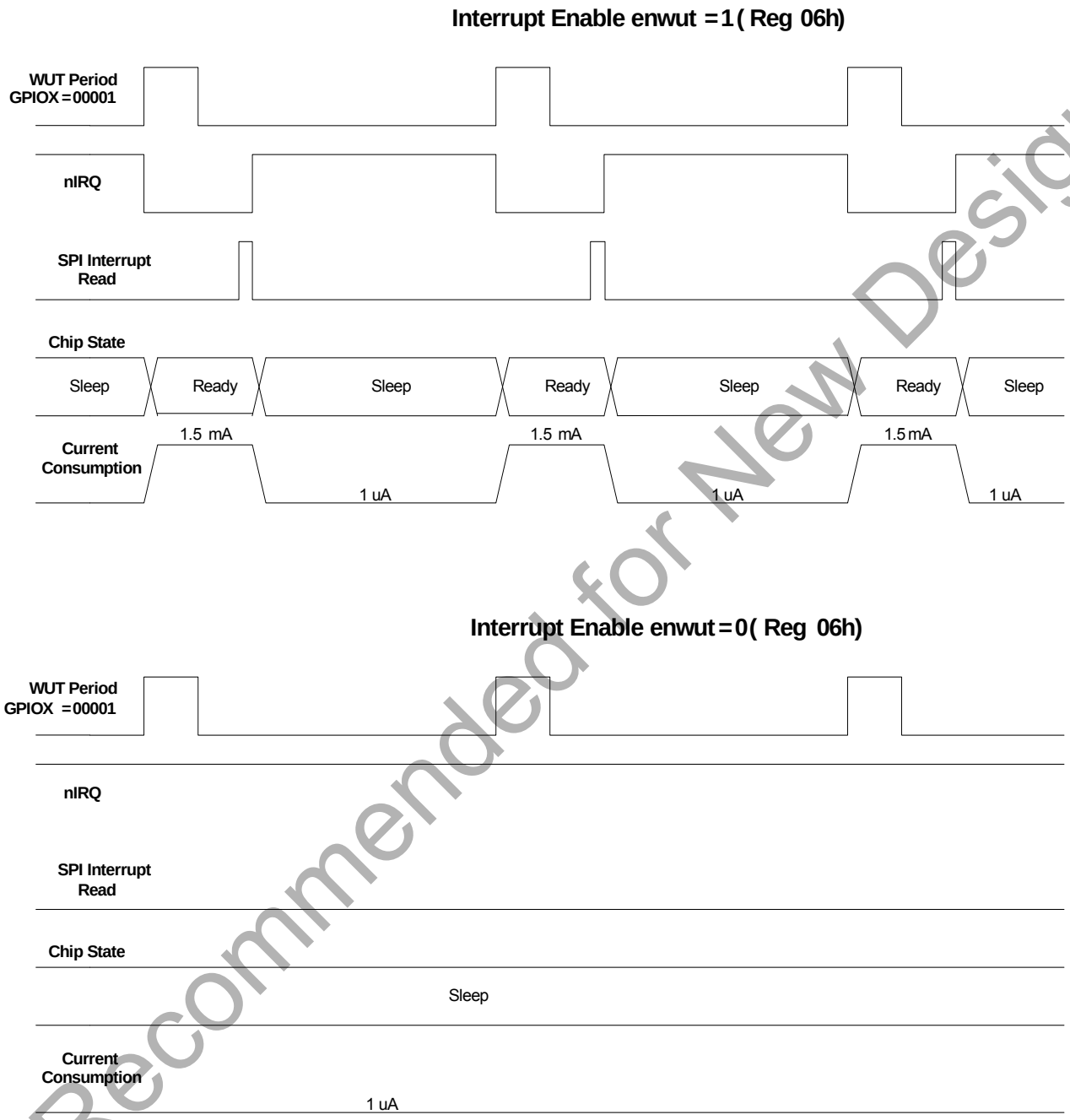



**Figure 32.23. WUT Interrupt and WUT Operation**

### 32.8.7. Low Duty Cycle Mode

The Low Duty Cycle Mode is available to automatically wake-up the receiver to check if a valid signal is available. The basic operation of the low duty cycle mode is demonstrated in the figure below. If a valid preamble or sync word is not detected the chip will return to sleep mode until the beginning of a new WUT period. If a valid preamble and sync are detected the receiver on period will be extended for the low duty cycle mode duration (TLDC) to receive all of the packet. The WUT period must be set in conjunction with the low duty cycle mode duration. The R value (“Register 14h. Wake-up Timer Period 1”) is shared

Not Recommended for New Designs

between the WUT and the TLDC. The ldc[7:0] bits are located in “Register 19h. Low Duty Cycle Mode Duration.” The time of the TLDC is determined by the formula below:

$$TLDC = ldc[7:0] \times \frac{4 \times 2^{R}}{32.768}\ ms$$

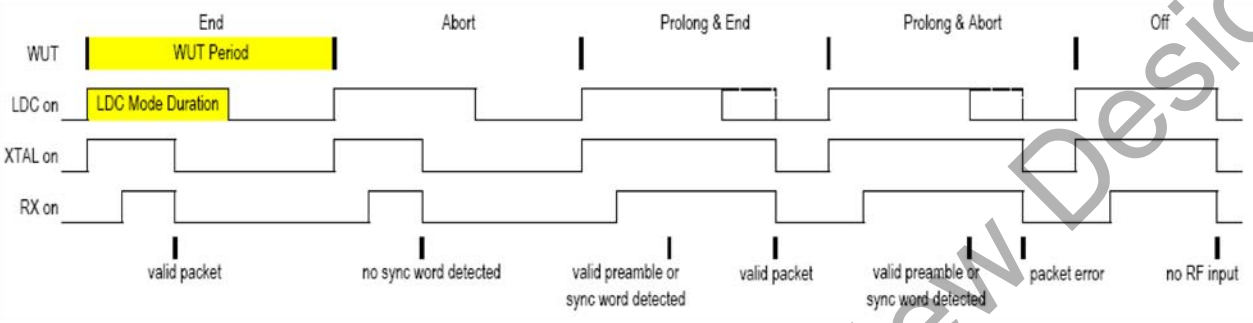



**Figure 32.24. Low Duty Cycle Mode**

### 32.8.8. GPIO Configuration

Three general purpose IOs (GPIOs) are available. Numerous functions such as specific interrupts, TRSW control, etc. can be routed to the GPIO pins as shown in the tables below. When in Shutdown mode all the GPIO pads are pulled low.

**Note:** The ADC should not be selected as an input to the GPIO in standby or sleep modes and will cause excess current consumption.

| Add | R/W | Function/ Description | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | POR Def. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0B | R/W | GPIO0 Configuration | gpio0drv[1] | gpio0drv[0] | pup0 | gpio0[4] | gpio0[3] | gpio0[2] | gpio0[1] | gpio0[0] | 00h |
| 0C | R/W | GPIO1 Configuration | gpio1drv[1] | gpio1drv[0] | pup1 | gpio1[4] | gpio1[3] | gpio1[2] | gpio1[1] | gpio1[0] | 00h |
| 0D | R/W | GPIO2 Configuration | gpio2drv[1] | gpio2drv[0] | pup2 | gpio2[4] | gpio2[3] | gpio2[2] | gpio2[1] | gpio2[0] | 00h |
| 0E | R/W | I/O Port Configuration | | extitst[2] | extitst[1] | extitst[0] | itsdo | dio2 | dio1 | dio0 | 00h |

The GPIO settings for GPIO1 and GPIO2 are the same as for GPIO0 with the exception of the 00000 default setting. The default settings for each GPIO are listed below:

| GPIO | 00000—Default Setting |
|---|---|
| GPIO0 | POR |
| GPIO1 | POR Inverted |
| GPIO2 | Output Clock |

For a complete list of the available GPIOs see “AN440: EZRadioPRO Detailed Register Descriptions”.

The GPIO drive strength may be adjusted with the gpioXdrv[1:0] bits. Setting a higher value will increase the drive strength and current capability of the GPIO by changing the driver size. Special care should be

Not Recommended for New Designs

taken in setting the drive strength and loading on GPIO2 when the microcontroller clock is used. Excess loading or inadequate drive may contribute to increased spurious emissions.

Pin 6, ANT may be used as an alternate to control a TR switch. Pin 6 is a hardwired version of GPIO setting 11000, Antenna 2 Switch used for antenna diversity. It can be manually controlled by the antdiv[2:0] bits in register 08h if antenna diversity is not used. See AN440, register 08h for more details.

### 32.8.9. Antenna Diversity

To mitigate the problem of frequency-selective fading due to multi-path propagation, some transceiver systems use a scheme known as antenna diversity. In this scheme, two antennas are used. Each time the transceiver enters RX mode the receive signal strength from each antenna is evaluated. This evaluation process takes place during the preamble portion of the packet. The antenna with the strongest received signal is then used for the remainder of that RX packet. The same antenna will also be used for the next corresponding TX packet.

This chip fully supports antenna diversity with an integrated antenna diversity control algorithm. The required signals needed to control an external SPDT RF switch (such as PIN diode or GaAs switch) are available on the GPIOx pins. The operation of these GPIO signals is programmable to allow for different antenna diversity architectures and configurations. The antdiv[2:0] bits are found in register 08h “Operating & Function Control 2.” The GPIO pins are capable of sourcing up to 5 mA of current, so it may be used directly to forward-bias a PIN diode if desired.

The antenna diversity algorithm will automatically toggle back and forth between the antennas until the packet starts to arrive. The recommended preamble length for optimal antenna selection is 8 bytes. A special antenna diversity algorithm (antdiv[2:0] = 110 or 111) is included that allows for shorter preamble lengths for beacon mode in TDMA-like systems where the arrival of the packet is synchronous to the receiver enable. The recommended preamble length to obtain optimal antenna selection for synchronous mode is 4 bytes.

| Add | R/W | Function/Description | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | POR Def. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 08 | R/W | Operating & Function Control 2 | antdiv[2] | antdiv[1] | antdiv[0] | rxmpk | autotx | enldm | ffclrrx | ffclrtx | 00h |

**Table 32.8. Antenna Diversity Control**

| antdiv[2:0] | RX/TX State | | Non RX/TX State | |
|---|---|---|---|---|
| | GPIO Ant1 | GPIO Ant2 | GPIO Ant1 | GPIO Ant2 |
| 000 | 0 | 1 | 0 | 0 |
| 001 | 1 | 0 | 0 | 0 |
| 010 | 0 | 1 | 1 | 1 |
| 011 | 1 | 0 | 1 | 1 |
| 100 | Antenna Diversity Algorithm | | 0 | 0 |
| 101 | Antenna Diversity Algorithm | | 1 | 1 |
| 110 | Antenna Diversity Algorithm in Beacon Mode | | 0 | 0 |
| 111 | Antenna Diversity Algorithm in Beacon Mode | | 1 | 1 |

### 32.8.10. RSSI and Clear Channel Assessment

Received signal strength indicator (RSSI) is an estimate of the signal strength in the channel to which the receiver is tuned. The RSSI value can be read from an 8-bit register with 0.5 dB resolution per bit. Figure 32.25 demonstrates the relationship between input power level and RSSI value. The absolute value of the RSSI will change slightly depending on the modem settings. The RSSI may be read at anytime, but

Not Recommended for New Designs

an incorrect error may rarely occur. The RSSI value may be incorrect if read during the update period. The update period is approximately 10 ns every 4 Tb. For 10 kbps, this would result in a 1 in 40,000 probability that the RSSI may be read incorrectly. This probability is extremely low, but to avoid this, one of the following options is recommended: majority polling, reading the RSSI value within 1 Tb of the RSSI interrupt, or using the RSSI threshold described in the next paragraph for Clear Channel Assessment (CCA).

| Add | R/W | Function/Description | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | POR Def. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 26 | R | Received Signal Strength Indicator | rssi[7] | rssi[6] | rssi[5] | rssi[4] | rssi[3] | rssi[2] | rssi[1] | rssi[0] | — |
| 27 | R/W | RSSI Threshold for Clear Channel Indicator | rssith[7] | rssith[6] | rssith[5] | rssith[4] | rssith[3] | rssith[2] | rssith[1] | rssith[0] | 00h |

For CCA, threshold is programmed into rssith[7:0] in "Register 27h. RSSI Threshold for Clear Channel Indicator." After the RSSI is evaluated in the preamble, a decision is made if the signal strength on this channel is above or below the threshold. If the signal strength is above the programmed threshold then the RSSI status bit, irssi, in "Register 04h. Interrupt/Status 2" will be set to 1. The RSSI status can also be routed to a GPIO line by configuring the GPIO configuration register to GPIOx[3:0] = 1110.

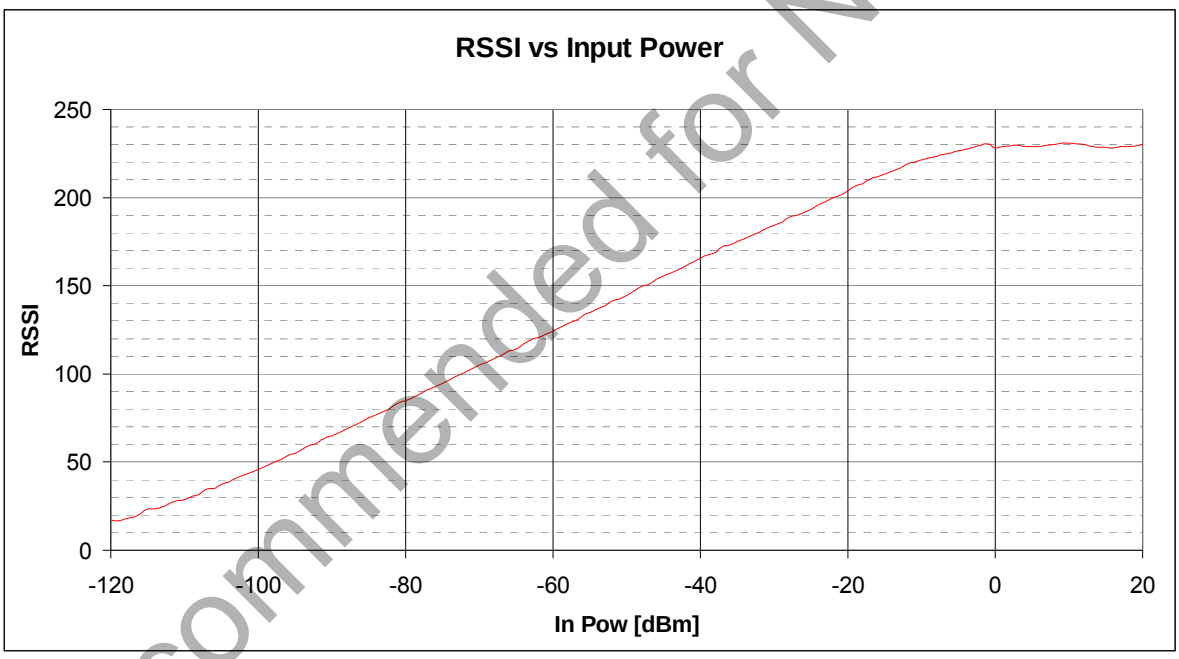



**Figure 32.25. RSSI Value vs. Input Power**

## 32.9. Reference Design

Reference designs are available at www.silabs.com for many common applications which include recommended schematics, BOM, and layout. TX matching component values for the different frequency bands can be found in the application notes “AN435: Si4032/4432 PA Matching” and “AN436: Si4030/4031/4430/4431 PA Matching.” RX matching component values for different frequency bands can be found in “AN427: EZRadioPRO Si433x and Si443x RX LNA Matching.”

Not Recommended for New Designs

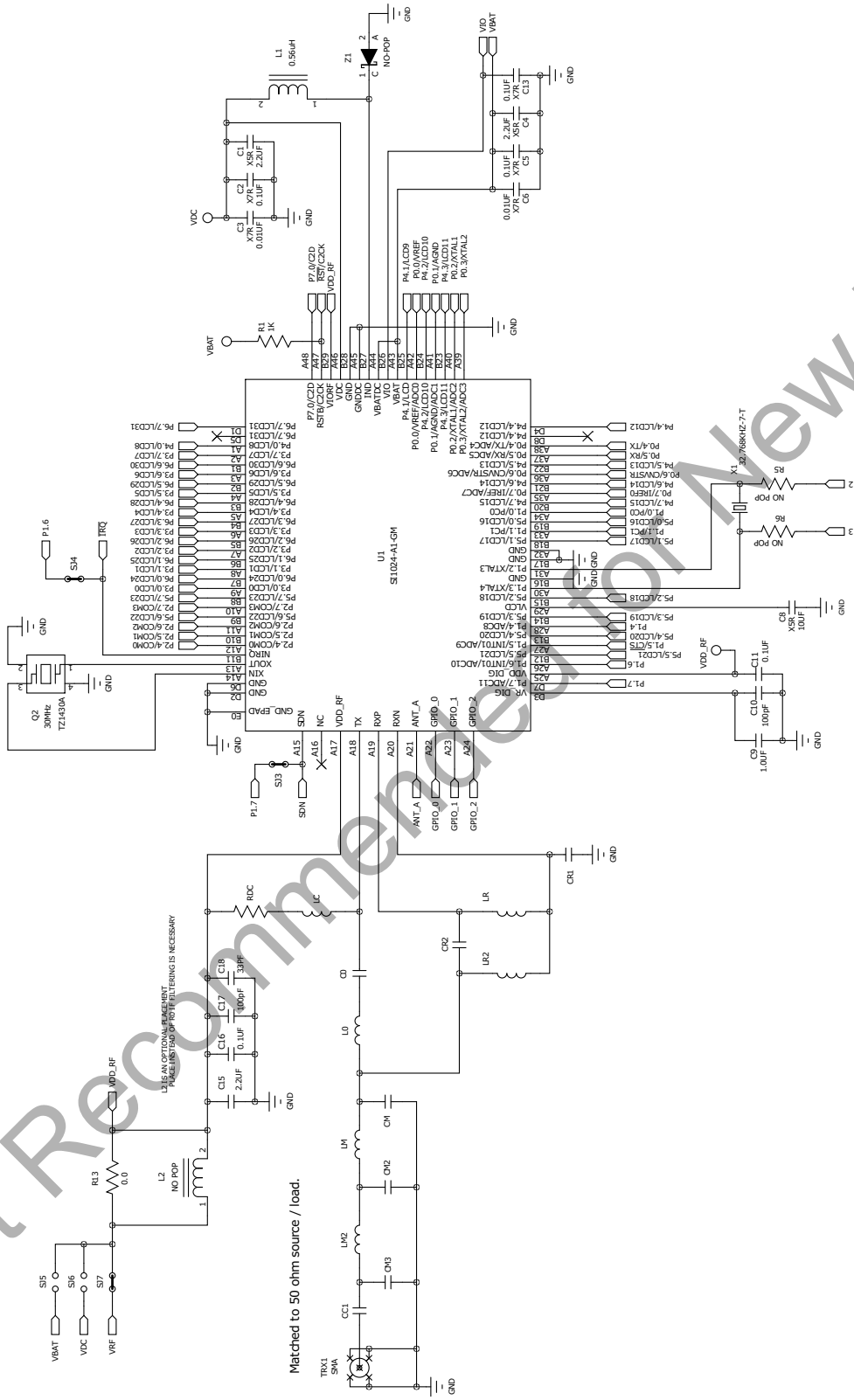



Figure 32.26. Si1024 Split RF TX/RX Direct-Tie Reference Design—Schematic

Not Recommended for New Designs

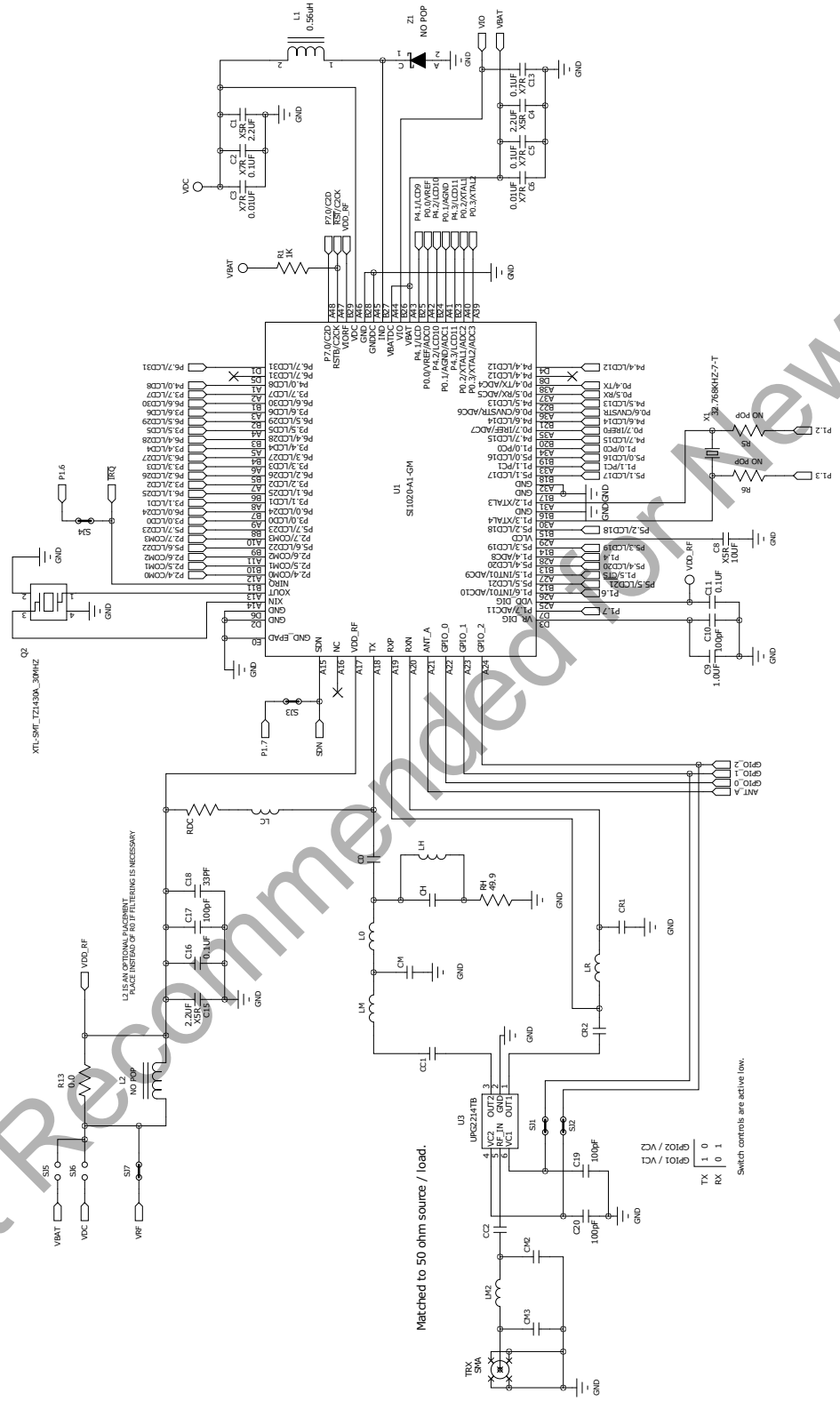



Figure 32.27. Si1020 Switch Matching Reference Design—Schematic

Not Recommended for New Designs

## 32.10. Application Notes and Reference Designs

A comprehensive set of application notes and reference designs are available to assist with the development of a radio system. A partial list of applications notes is given below.

For the complete list of application notes, latest reference designs and demos visit the Silicon Labs website.

- AN361: Wireless MBUS Implementation using EZRadioPRO Devices
- AN379: Antenna Diversity with EZRadioPRO
- AN414: EZRadioPRO Layout Design Guide
- AN415: EZRadioPRO Programming Guide
- AN417: Si4x3x Family Crystal Oscillators
- AN419: ARIB STD-T67 Narrow-Band 426/429 MHz Measured on the Si4431-A0
- AN427: EZRadioPRO Si433x and Si443x RX LNA Matching
- AN429: Using the DC-DC Converter on the F9xx Series MCU for Single Battery Operation with the EZRadioPRO RF Devices
- AN432: RX BER Measurement on EZRadioPRO with a Looped PN Sequence
- AN435: Si4032/4432 PA Matching
- AN436: Si4030/4031/4430/4431 PA Matching
- AN437: 915 MHz Measurement Results and FCC Compliance
- AN439: EZRadioPRO Quick Start Guide
- AN440: Si4430/31/32 Register Descriptions
- AN445: Si4431 RF Performance and ETSI Compliance Test Results
- AN451: Wireless M-BUS Software Implementation
- AN459: 950 MHz Measurement Results and ARIB Compliance
- AN460: 470 MHz Measurement Results for China
- AN463: Support for Non-Standard Packet Structures and RAW Mode
- AN466: Si4030/31/32 Register Descriptions
- AN467: Si4330 Register Descriptions
- AN514: Using the EZLink Reference Design to Create a Two-Channel PWM Motor Control Circuit
- AN539: EZMacPRO Overview

## 32.11. Customer Support

Technical support for the complete family of Silicon Labs wireless products is available by accessing the wireless section of the Silicon Labs' website at www.silabs.com/wireless. For MCU support, please visit www.silabs.com/mcu.

For answers to common questions please visit the wireless and mcu knowledge base at www.silabs.com/support/knowledgebase.

Not Recommended for New Designs

## 32.12. Register Table and Descriptions

### Table 32.9. EZRadioPRO Internal Register Descriptions

| Add | R/W | Function/Desc | Data | | | | | | | | POR Default |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | |
| 00 | R | Device Type | 0 | 0 | 0 | dt[4] | dt[3] | dt[2] | dt[1] | dt[0] | 00111 |
| 01 | R | Device Version | 0 | 0 | 0 | vc[4] | vc[3] | vc[2] | vc[1] | vc[0] | 07h |
| 02 | R | Device Status | ffovfl | ffunfl | rxffem | headerr | reserved | reserved | cps[1] | cps[0] | — |
| 03 | R | Interrupt Status 1 | ifferr | itxffafull | itxffaem | irxffafull | iext | ipksent | ipkvalid | icrcerror | — |
| 04 | R | Interrupt Status 2 | iswdet | ipreaval | ipreainval | irssi | iwut | ilbd | ichiprdy | ipor | — |
| 05 | R/W | Interrupt Enable 1 | enfferr | entxffafull | entxffaem | enrxffafull | enext | enpksent | enpkvalid | encrcerror | 00h |
| 06 | R/W | Interrupt Enable 2 | enswdet | enpreaval | enpreainval | enrssi | enwut | enlbd | enchiprdy | enpor | 03h |
| 07 | R/W | Operating & Function Control 1 | swres | enlbd | enwt | x32ksel | txon | rxon | pllon | xton | 01h |
| 08 | R/W | Operating & Function Control 2 | antdiv[2] | antdiv[1] | antdiv[0] | rxmpk | autotx | enldm | ffclrrx | ffclrtx | 00h |
| 09 | R/W | Crystal Oscillator Load Capacitance | xtalshft | xlc[6] | xlc[5] | xlc[4] | xlc[3] | xlc[2] | xlc[1] | xlc[0] | 7Fh |
| 0A | R/W | Microcontroller Output Clock | Reserved | Reserved | clkt[1] | clkt[0] | enlfc | mclk[2] | mclk[1] | mclk[0] | 06h |
| 0B | R/W | GPIO0 Configuration | gpio0drv[1] | gpio0drv[0] | pup0 | gpio0[4] | gpio0[3] | gpio0[2] | gpio0[1] | gpio0[0] | 00h |
| 0C | R/W | GPIO1 Configuration | gpio1drv[1] | gpio1drv[0] | pup1 | gpio1[4] | gpio1[3] | gpio1[2] | gpio1[1] | gpio1[0] | 00h |
| 0D | R/W | GPIO2 Configuration | gpio2drv[1] | gpio2drv[0] | pup2 | gpio2[4] | gpio2[3] | gpio2[2] | gpio2[1] | gpio2[0] | 00h |
| 0E | R/W | I/O Port Configuration | Reserved | extitst[2] | extitst[1] | extitst[0] | itsdo | dio2 | dio1 | dio0 | 00h |
| 0F | R/W | ADC Configuration | adcstart/*adc-done* | adcsel[2] | adcsel[1] | adcsel[0] | adcref[1] | adcref[0] | adcgain[1] | adcgain[0] | 00h |
| 10 | R/W | ADC Sensor Amplifier Offset | Reserved | Reserved | Reserved | Reserved | adcoffs[3] | adcoffs[2] | adcoffs[1] | adcoffs[0] | 00h |
| 11 | R | ADC Value | adc[7] | adc[6] | adc[5] | adc[4] | adc[3] | adc[2] | adc[1] | adc[0] | — |
| 12 | R/W | Temperature Sensor Control | tsrange[1] | tsrange[0] | entsoffs | entstrim | tstrim[3] | tstrim[2] | tstrim[1] | tstrim[0] | 20h |
| 13 | R/W | Temperature Value Offset | tvoffs[7] | tvoffs[6] | tvoffs[5] | tvoffs[4] | tvoffs[3] | tvoffs[2] | tvoffs[1] | tvoffs[0] | 00h |
| 14 | R/W | Wake-Up Timer Period 1 | Reserved | Reserved | Reserved | wtr[4] | wtr[3] | wtr[2] | wtr[1] | wtr[0] | 03h |
| 15 | R/W | Wake-Up Timer Period 2 | wtm[15] | wtm[14] | wtm[13] | wtm[12] | wtm[11] | wtm[10] | wtm[9] | wtm[8] | 00h |
| 16 | R/W | Wake-Up Timer Period 3 | wtm[7] | wtm[6] | wtm[5] | wtm[4] | wtm[3] | wtm[2] | wtm[1] | wtm[0] | 01h |
| 17 | R | Wake-Up Timer Value 1 | wtv[15] | wtv[14] | wtv[13] | wtv[12] | wtv[11] | wtv[10] | wtv[9] | wtv[8] | — |
| 18 | R | Wake-Up Timer Value 2 | wtv[7] | wtv[6] | wtv[5] | wtv[4] | wtv[3] | wtv[2] | wtv[1] | wtv[0] | — |
| 19 | R/W | Low-Duty Cycle Mode Duration | ldc[7] | ldc[6] | ldc[5] | ldc[4] | ldc[3] | ldc[2] | ldc[1] | ldc[0] | 00h |
| 1A | R/W | Low Battery Detector Threshold | Reserved | Reserved | Reserved | lbdt[4] | lbdt[3] | lbdt[2] | lbdt[1] | lbdt[0] | 14h |
| 1B | R | Battery Voltage Level | 0 | 0 | 0 | vbat[4] | vbat[3] | vbat[2] | vbat[1] | vbat[0] | — |
| 1C | R/W | IF Filter Bandwidth | dwn3_bypass | ndec[2] | ndec[1] | ndec[0] | filset[3] | filset[2] | filset[1] | filset[0] | 01h |
| 1D | R/W | AFC Loop Gearshift Override | afcbd | enafc | afcgearh[2] | afcgearh[1] | afcgearh[0] | 1p5 bypass | matap | ph0size | 40h |
| 1E | R/W | AFC Timing Control | swait_timer[1] | swait_timer[0] | shwait[2] | shwait[1] | shwait[0] | anwait[2] | anwait[1] | anwait[0] | 0Ah |
| 1F | R/W | Clock Recovery Gearshift Override | Reserved | Reserved | crfast[2] | crfast[1] | crfast[0] | crslow[2] | crslow[1] | crslow[0] | 03h |
| 20 | R/W | Clock Recovery Oversampling Ratio | rxosr[7] | rxosr[6] | rxosr[5] | rxosr[4] | rxosr[3] | rxosr[2] | rxosr[1] | rxosr[0] | 64h |
| 21 | R/W | Clock Recovery Offset 2 | rxosr[10] | rxosr[9] | rxosr[8] | stallctrl | ncoff[19] | ncoff[18] | ncoff[17] | ncoff[16] | 01h |
| 22 | R/W | Clock Recovery Offset 1 | ncoff[15] | ncoff[14] | ncoff[13] | ncoff[12] | ncoff[11] | ncoff[10] | ncoff[9] | ncoff[8] | 47h |
| 23 | R/W | Clock Recovery Offset 0 | ncoff[7] | ncoff[6] | ncoff[5] | ncoff[4] | ncoff[3] | ncoff[2] | ncoff[1] | ncoff[0] | AEh |
| 24 | R/W | Clock Recovery Timing Loop Gain 1 | Reserved | Reserved | Reserved | rxncocomp | crgain2x | crgain[10] | crgain[9] | crgain[8] | 02h |
| 25 | R/W | Clock Recovery Timing Loop Gain 0 | crgain[7] | crgain[6] | crgain[5] | crgain[4] | crgain[3] | crgain[2] | crgain[1] | crgain[0] | 8Fh |
| 26 | R | Received Signal Strength Indicator | rssi[7] | rssi[6] | rssi[5] | rssi[4] | rssi[3] | rssi[2] | rssi[1] | rssi[0] | — |
| 27 | R/W | RSSI Threshold for Clear Channel Indicator | rssith[7] | rssith[6] | rssith[5] | rssith[4] | rssith[3] | rssith[2] | rssith[1] | rssith[0] | 1Eh |
| 28 | R | Antenna Diversity Register 1 | adrssi1[7] | adrssia[6] | adrssia[5] | adrssia[4] | adrssia[3] | adrssia[2] | adrssia[1] | adrssia[0] | — |
| 29 | R | Antenna Diversity Register 2 | adrssib[7] | adrssib[6] | adrssib[5] | adrssib[4] | adrssib[3] | adrssib[2] | adrssib[1] | adrssib[0] | — |
| 2A | R/W | AFC Limiter | Afclim[7] | Afclim[6] | Afclim[5] | Afclim[4] | Afclim[3] | Afclim[2] | Afclim[1] | Afclim[0] | 00h |
| 2B | R | AFC Correction Read | afc_corr[9] | afc_corr[8] | afc_corr[7] | afc_corr[6] | afc_corr[5] | afc_corr[4] | afc_corr[3] | afc_corr[2] | 00h |
| 2C | R/W | OOK Counter Value 1 | afc_corr[9] | afc_corr[9] | ookfrzen | peakdeten | madeten | ookcnt[10] | ookcnt[9] | ookcnt[8] | 18h |
| 2D | R/W | OOK Counter Value 2 | ookcnt[7] | ookcnt[6] | ookcnt[5] | ookcnt[4] | ookcnt[3] | ookcnt[2] | ookcnt[1] | ookcnt[0] | BCh |
| 2E | R/W | Slicer Peak Hold | Reserved | attack[2] | attack[1] | attack[0] | decay[3] | decay[2] | decay[1] | decay[0] | 26h |
| 2F | Reserved | | | | | | | | | | |

Not Recommended for New Designs

## Table 32.9. EZRadioPRO Internal Register Descriptions (Continued)

| Add | R/W | Function/Desc | Data | | | | | | | | POR Default |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | |
| 30 | R/W | Data Access Control | enpacrx | lsbfrst | crcdonly | skip2ph | enpactx | encrc | crc[1] | crc[0] | 8Dh |
| 31 | R | EzMAC status | 0 | rxcrc1 | pksrch | pkrx | pkvalid | crcerror | pktx | pksent | — |
| 32 | R/W | Header Control 1 | bcen[3:0] | | | | hdch[3:0] | | | | 0Ch |
| 33 | R/W | Header Control 2 | skipsyn | hdlen[2] | hdlen[1] | hdlen[0] | fixpklen | synclen[1] | synclen[0] | prealen[8] | 22h |
| 34 | R/W | Preamble Length | prealen[7] | prealen[6] | prealen[5] | prealen[4] | prealen[3] | prealen[2] | prealen[1] | prealen[0] | 08h |
| 35 | R/W | Preamble Detection Control | preath[4] | preath[3] | preath[2] | preath[1] | preath[0] | rssi_off[2] | rssi_off[1] | rssi_off[0] | 2Ah |
| 36 | R/W | Sync Word 3 | sync[31] | sync[30] | sync[29] | sync[28] | sync[27] | sync[26] | sync[25] | sync[24] | 2Dh |
| 37 | R/W | Sync Word 2 | sync[23] | sync[22] | sync[21] | sync[20] | sync[19] | sync[18] | sync[17] | sync[16] | D4h |
| 38 | R/W | Sync Word 1 | sync[15] | sync[14] | sync[13] | sync[12] | sync[11] | sync[10] | sync[9] | sync[8] | 00h |
| 39 | R/W | Sync Word 0 | sync[7] | sync[6] | sync[5] | sync[4] | sync[3] | sync[2] | sync[1] | sync[0] | 00h |
| 3A | R/W | Transmit Header 3 | txhd[31] | txhd[30] | txhd[29] | txhd[28] | txhd[27] | txhd[26] | txhd[25] | txhd[24] | 00h |
| 3B | R/W | Transmit Header 2 | txhd[23] | txhd[22] | txhd[21] | txhd[20] | txhd[19] | txhd[18] | txhd[17] | txhd[16] | 00h |
| 3C | R/W | Transmit Header 1 | txhd[15] | txhd[14] | txhd[13] | txhd[12] | txhd[11] | txhd[10] | txhd[9] | txhd[8] | 00h |
| 3D | R/W | Transmit Header 0 | txhd[7] | txhd[6] | txhd[5] | txhd[4] | txhd[3] | txhd[2] | txhd[1] | txhd[0] | 00h |
| 3E | R/W | Transmit Packet Length | pklen[7] | pklen[6] | pklen[5] | pklen[4] | pklen[3] | pklen[2] | pklen[1] | pklen[0] | 00h |
| 3F | R/W | Check Header 3 | chhd[31] | chhd[30] | chhd[29] | chhd[28] | chhd[27] | chhd[26] | chhd[25] | chhd[24] | 00h |
| 40 | R/W | Check Header 2 | chhd[23] | chhd[22] | chhd[21] | chhd[20] | chhd[19] | chhd[18] | chhd[17] | chhd[16] | 00h |
| 41 | R/W | Check Header 1 | chhd[15] | chhd[14] | chhd[13] | chhd[12] | chhd[11] | chhd[10] | chhd[9] | chhd[8] | 00h |
| 42 | R/W | Check Header 0 | chhd[7] | chhd[6] | chhd[5] | chhd[4] | chhd[3] | chhd[2] | chhd[1] | chhd[0] | 00h |
| 43 | R/W | Header Enable 3 | hden[31] | hden[30] | hden[29] | hden[28] | hden[27] | hden[26] | hden[25] | hden[24] | FFh |
| 44 | R/W | Header Enable 2 | hden[23] | hden[22] | hden[21] | hden[20] | hden[19] | hden[18] | hden[17] | hden[16] | FFh |
| 45 | R/W | Header Enable 1 | hden[15] | hden[14] | hden[13] | hden[12] | hden[11] | hden[10] | hden[9] | hden[8] | FFh |
| 46 | R/W | Header Enable 0 | hden[7] | hden[6] | hden[5] | hden[4] | hden[3] | hden[2] | hden[1] | hden[0] | FFh |
| 47 | R | Received Header 3 | rxhd[31] | rxhd[30] | rxhd[29] | rxhd[28] | rxhd[27] | rxhd[26] | rxhd[25] | rxhd[24] | — |
| 48 | R | Received Header 2 | rxhd[23] | rxhd[22] | rxhd[21] | rxhd[20] | rxhd[19] | rxhd[18] | rxhd[17] | rxhd[16] | — |
| 49 | R | Received Header 1 | rxhd[15] | rxhd[14] | rxhd[13] | rxhd[12] | rxhd[11] | rxhd[10] | rxhd[9] | rxhd[8] | — |
| 4A | R | Received Header 0 | rxhd[7] | rxhd[6] | rxhd[5] | rxhd[4] | rxhd[3] | rxhd[2] | rxhd[1] | rxhd[0] | — |
| 4B | R | Received Packet Length | rxplen[7] | rxplen[6] | rxplen[5] | rxplen[4] | rxplen[3] | rxplen[2] | rxplen[1] | rxplen[0] | — |
| 4C-4E | Reserved | | | | | | | | | | |
| 4F | R/W | ADC8 Control | Reserved | Reserved | adc8[5] | adc8[4] | adc8[3] | adc8[2] | adc8[1] | adc8[0] | 10h |
| 50-5F | Reserved | | | | | | | | | | |
| 60 | R/W | Channel Filter Coefficient Address | Inv_pre_th[3] | Inv_pre_th[2] | Inv_pre_th[1] | Inv_pre_th[0] | chfiladd[3] | chfiladd[2] | chfiladd[1] | chfiladd[0] | 00h |
| 61 | Reserved | | | | | | | | | | |
| 62 | R/W | Crystal Oscillator/ Control Test | pwst[2] | pwst[1] | pwst[0] | clkhyst | enbias2x | enamp2x | bufovr | enbuf | 24h |
| 63-6C | Reserved | | | | | | | | | | |
| 6D | R/W | TX Power | Reserved | Reserved | Reserved | Reserved | lna_sw | txpow[2] | txpow[1] | txpow[0] | 18h |
| 6E | R/W | TX Data Rate 1 | txdr[15] | txdr[14] | txdr[13] | txdr[12] | txdr[11] | txdr[10] | txdr[9] | txdr[8] | 0Ah |
| 6F | R/W | TX Data Rate 0 | txdr[7] | txdr[6] | txdr[5] | txdr[4] | txdr[3] | txdr[2] | txdr[1] | txdr[0] | 3Dh |
| 70 | R/W | Modulation Mode Control 1 | Reserved | Reserved | txdtrtscale | enphpwdn | manppol | enmaninv | enmanch | enwhite | 0Ch |
| 71 | R/W | Modulation Mode Control 2 | trclk[1] | trclk[0] | dtmod[1] | dtmod[0] | eninv | fd[8] | modtyp[1] | modtyp[0] | 00h |
| 72 | R/W | Frequency Deviation | fd[7] | fd[6] | fd[5] | fd[4] | fd[3] | fd[2] | fd[1] | fd[0] | 20h |
| 73 | R/W | Frequency Offset 1 | fo[7] | fo[6] | fo[5] | fo[4] | fo[3] | fo[2] | fo[1] | fo[0] | 00h |
| 74 | R/W | Frequency Offset 2 | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | fo[9] | fo[8] | 00h |
| 75 | R/W | Frequency Band Select | Reserved | sbsel | hbsel | fb[4] | fb[3] | fb[2] | fb[1] | fb[0] | 75h |
| 76 | R/W | Nominal Carrier Frequency 1 | fc[15] | fc[14] | fc[13] | fc[12] | fc[11] | fc[10] | fc[9] | fc[8] | BBh |
| 77 | R/W | Nominal Carrier Frequency 0 | fc[7] | fc[6] | fc[5] | fc[4] | fc[3] | fc[2] | fc[1] | fc[0] | 80h |
| 78 | Reserved | | | | | | | | | | |
| 79 | R/W | Frequency Hopping Channel Select | fhch[7] | fhch[6] | fhch[5] | fhch[4] | fhch[3] | fhch[2] | fhch[1] | fhch[0] | 00h |
| 7A | R/W | Frequency Hopping Step Size | fhs[7] | fhs[6] | fhs[5] | fhs[4] | fhs[3] | fhs[2] | fhs[1] | fhs[0] | 00h |
| 7B | Reserved | | | | | | | | | | |
| 7C | R/W | TX FIFO Control 1 | Reserved | Reserved | txafthr[5] | txafthr[4] | txafthr[3] | txafthr[2] | txafthr[1] | txafthr[0] | 37h |
| 7D | R/W | TX FIFO Control 2 | Reserved | Reserved | txaethr[5] | txaethr[4] | txaethr[3] | txaethr[2] | txaethr[1] | txaethr[0] | 04h |
| 7E | R/W | RX FIFO Control | Reserved | Reserved | rxafthr[5] | rxafthr[4] | rxafthr[3] | rxafthr[2] | rxafthr[1] | rxafthr[0] | 37h |
| 7F | R/W | FIFO Access | fifod[7] | fifod[6] | fifod[5] | fifod[4] | fifod[3] | fifod[2] | fifod[1] | fifod[0] | — |

**Note:** Detailed register descriptions are available in “AN440: EZRadioPRO Detailed Register Descriptions.

Not Recommended for New Designs

## 32.13. Required Changes to Default Register Values

The following register writes should be performed during device initialization.

1. The value 0x40 should be written to Register 59h.
2. If the device will be operated in the 240–320 MHz or 480–640 MHz bands at a temperature above 60 °C, then Register 59h should be written to 0x43 and Register 5Ah should be written to 0x02.

Not Recommended for New Designs

## 33. Timers

Each MCU includes four counter/timers: two are 16-bit counter/timers compatible with those found in the standard 8051, and two are 16-bit auto-reload timer for use with the ADC, SMBus, or for general purpose use. These timers can be used to measure time intervals, count external events and generate periodic interrupt requests. Timer 0 and Timer 1 are nearly identical and have four primary modes of operation. Timer 2 and Timer 3 offer 16-bit and split 8-bit timer functionality with auto-reload. Additionally, Timer 2 and Timer 3 have a Capture Mode that can be used to measure the SmaRTClock, Comparator, or external clock period with respect to another oscillator. The ability to measure the Comparator period with respect to another oscillator is particularly useful when interfacing to capacitive sensors.

| Timer 0 and Timer 1 Modes: | Timer 2 Modes: | Timer 3 Modes: |
|---|---|---|
| 13-bit counter/timer | 16-bit timer with auto-reload | 16-bit timer with auto-reload |
| 16-bit counter/timer | | |
| 8-bit counter/timer with auto-reload | Two 8-bit timers with auto-reload | Two 8-bit timers with auto-reload |
| Two 8-bit counter/timers (Timer 0 only) | | |

Timers 0 and 1 may be clocked by one of five sources, determined by the Timer Mode Select bits (T1M–T0M) and the Clock Scale bits (SCA1–SCA0). The Clock Scale bits define a pre-scaled clock from which Timer 0 and/or Timer 1 may be clocked (See SFR Definition 33.1 for pre-scaled clock selection).

Timer 0/1 may then be configured to use this pre-scaled clock signal or the system clock. Timer 2 and Timer 3 may be clocked by the system clock, the system clock divided by 12. Timer 2 may additionally be clocked by the SmaRTClock divided by 8 or the Comparator0 output. Timer 3 may additionally be clocked by the external oscillator clock source divided by 8 or the Comparator1 output.

Timer 0 and Timer 1 may also be operated as counters. When functioning as a counter, a counter/timer register is incremented on each high-to-low transition at the selected input pin (T0 or T1). Events with a frequency of up to one-fourth the system clock frequency can be counted. The input signal need not be periodic, but it should be held at a given level for at least two full system clock cycles to ensure the level is properly sampled.

Not Recommended for New Designs

## SFR Definition 33.1. CKCON: Clock Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | T3MH | T3ML | T2MH | T2ML | T1M | T0M | SCA[1:0] | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x8E

| Bit | Name | Function |
|---|---|---|
| 7 | T3MH | **Timer 3 High Byte Clock Select.**<br>Selects the clock supplied to the Timer 3 high byte (split 8-bit timer mode only).<br>0: Timer 3 high byte uses the clock defined by the T3XCLK bit in TMR3CN.<br>1: Timer 3 high byte uses the system clock. |
| 6 | T3ML | **Timer 3 Low Byte Clock Select.**<br>Selects the clock supplied to Timer 3. Selects the clock supplied to the lower 8-bit timer in split 8-bit timer mode.<br>0: Timer 3 low byte uses the clock defined by the T3XCLK bit in TMR3CN.<br>1: Timer 3 low byte uses the system clock. |
| 5 | T2MH | **Timer 2 High Byte Clock Select.**<br>Selects the clock supplied to the Timer 2 high byte (split 8-bit timer mode only).<br>0: Timer 2 high byte uses the clock defined by the T2XCLK bit in TMR2CN.<br>1: Timer 2 high byte uses the system clock. |
| 4 | T2ML | **Timer 2 Low Byte Clock Select.**<br>Selects the clock supplied to Timer 2. If Timer 2 is configured in split 8-bit timer mode, this bit selects the clock supplied to the lower 8-bit timer.<br>0: Timer 2 low byte uses the clock defined by the T2XCLK bit in TMR2CN.<br>1: Timer 2 low byte uses the system clock. |
| 3 | T1M | **Timer 1 Clock Select.**<br>Selects the clock source supplied to Timer 1. Ignored when C/T1 is set to 1.<br>0: Timer 1 uses the clock defined by the prescale bits SCA[1:0].<br>1: Timer 1 uses the system clock. |
| 2 | T0M | **Timer 0 Clock Select.**<br>Selects the clock source supplied to Timer 0. Ignored when C/T0 is set to 1.<br>0: Counter/Timer 0 uses the clock defined by the prescale bits SCA[1:0].<br>1: Counter/Timer 0 uses the system clock. |
| 1:0 | SCA[1:0] | **Timer 0/1 Prescale Bits.**<br>These bits control the Timer 0/1 Clock Prescaler:<br>00: System clock divided by 12<br>01: System clock divided by 4<br>10: System clock divided by 48<br>11: External clock divided by 8 (synchronized with the system clock) |

Not Recommended for New Designs

## 33.1. Timer 0 and Timer 1

Each timer is implemented as a 16-bit register accessed as two separate bytes: a low byte (TL0 or TL1) and a high byte (TH0 or TH1). The Counter/Timer Control register (TCON) is used to enable Timer 0 and Timer 1 as well as indicate status. Timer 0 interrupts can be enabled by setting the ET0 bit in the IE register (Section “17.5. Interrupt Register Descriptions” on page 234); Timer 1 interrupts can be enabled by setting the ET1 bit in the IE register (Section “17.5. Interrupt Register Descriptions” on page 234). Both counter/timers operate in one of four primary modes selected by setting the Mode Select bits T1M1–T0M0 in the Counter/Timer Mode register (TMOD). Each timer can be configured independently. Each operating mode is described below.

### 33.1.1. Mode 0: 13-bit Counter/Timer

Timer 0 and Timer 1 operate as 13-bit counter/timers in Mode 0. The following describes the configuration and operation of Timer 0. However, both timers operate identically, and Timer 1 is configured in the same manner as described for Timer 0.

The TH0 register holds the eight MSBs of the 13-bit counter/timer. TL0 holds the five LSBs in bit positions TL0.4–TL0.0. The three upper bits of TL0 (TL0.7–TL0.5) are indeterminate and should be masked out or ignored when reading. As the 13-bit timer register increments and overflows from 0x1FFF (all ones) to 0x0000, the timer overflow flag TF0 (TCON.5) is set and an interrupt will occur if Timer 0 interrupts are enabled.

The C/T0 bit (TMOD.2) selects the counter/timer's clock source. When C/T0 is set to logic 1, high-to-low transitions at the selected Timer 0 input pin (T0) increment the timer register (Refer to Section “27.3. Priority Crossbar Decoder” on page 355 for information on selecting and configuring external I/O pins). Clearing C/T selects the clock defined by the T0M bit (CKCON.3). When T0M is set, Timer 0 is clocked by the system clock. When T0M is cleared, Timer 0 is clocked by the source selected by the Clock Scale bits in CKCON (see SFR Definition 33.1).

Setting the TR0 bit (TCON.4) enables the timer when either GATE0 (TMOD.3) is logic 0 or the input signal $\overline{\text{INT0}}$ is active as defined by bit IN0PL in register IT01CF (see SFR Definition 17.7). Setting GATE0 to 1 allows the timer to be controlled by the external input signal $\overline{\text{INT0}}$ (see Section “17.5. Interrupt Register Descriptions” on page 234), facilitating pulse width measurements

**Table 33.1. Timer 0 Running Modes**

| TR0 | GATE0 | $\overline{\text{INT0}}$ | Counter/Timer |
|---|---|---|---|
| 0 | X | X | Disabled |
| 1 | 0 | X | Enabled |
| 1 | 1 | 0 | Disabled |
| 1 | 1 | 1 | Enabled |
| **Note:** X = Don't Care | | | |

Setting TR0 does not force the timer to reset. The timer registers should be loaded with the desired initial value before the timer is enabled.

TL1 and TH1 form the 13-bit register for Timer 1 in the same manner as described above for TL0 and TH0. Timer 1 is configured and controlled using the relevant TCON and TMOD bits just as with Timer 0. The input signal $\overline{\text{INT1}}$ is used with Timer 1; the $\overline{\text{INT1}}$ polarity is defined by bit IN1PL in register IT01CF (see SFR Definition 17.7).

Not Recommended for New Designs

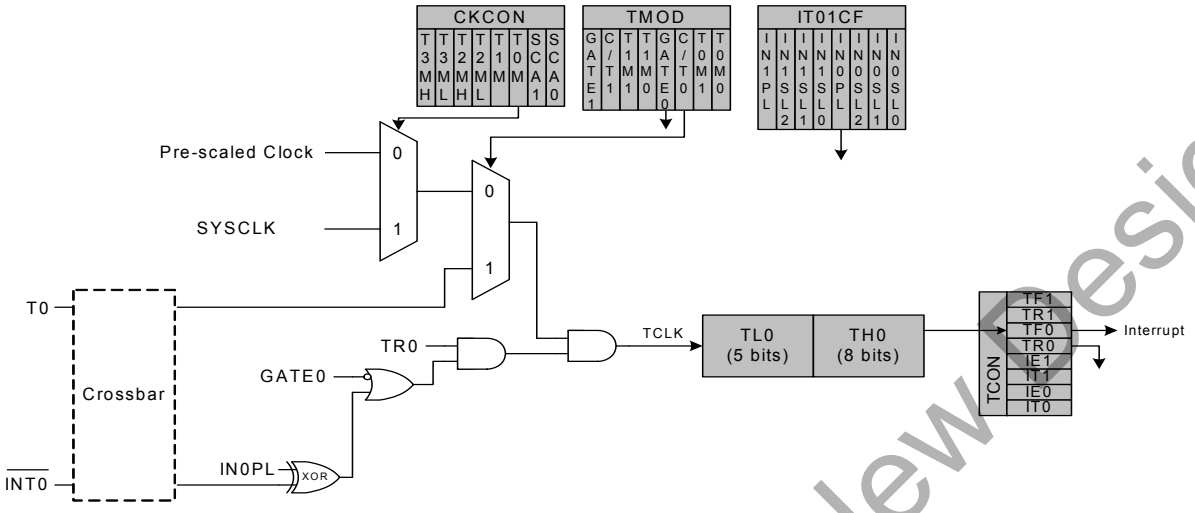



**Figure 33.1. T0 Mode 0 Block Diagram**

### 33.1.2. Mode 1: 16-bit Counter/Timer

Mode 1 operation is the same as Mode 0, except that the counter/timer registers use all 16 bits. The counter/timers are enabled and configured in Mode 1 in the same manner as for Mode 0.

### 33.1.3. Mode 2: 8-bit Counter/Timer with Auto-Reload

Mode 2 configures Timer 0 and Timer 1 to operate as 8-bit counter/timers with automatic reload of the start value. TL0 holds the count and TH0 holds the reload value. When the counter in TL0 overflows from all ones to 0x00, the timer overflow flag TF0 (TCON.5) is set and the counter in TL0 is reloaded from TH0. If Timer 0 interrupts are enabled, an interrupt will occur when the TF0 flag is set. The reload value in TH0 is not changed. TL0 must be initialized to the desired value before enabling the timer for the first count to be correct. When in Mode 2, Timer 1 operates identically to Timer 0.

Both counter/timers are enabled and configured in Mode 2 in the same manner as Mode 0. Setting the TR0 bit (TCON.4) enables the timer when either GATE0 (TMOD.3) is logic 0 or when the input signal INT0 is active as defined by bit IN0PL in register IT01CF (see Section “17.6. External Interrupts INT0 and INT1” on page 241 for details on the external input signals INT0 and INT1).

Not Recommended for New Designs

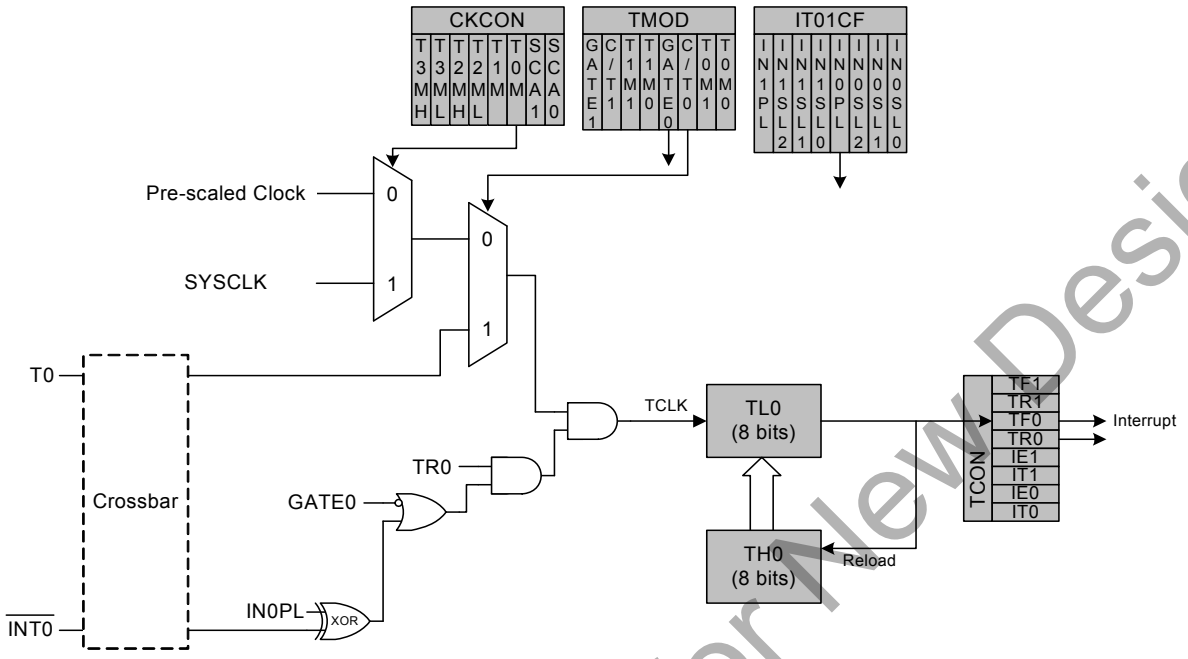



**Figure 33.2. T0 Mode 2 Block Diagram**

### 33.1.4. Mode 3: Two 8-bit Counter/Timers (Timer 0 Only)

In Mode 3, Timer 0 is configured as two separate 8-bit counter/timers held in TL0 and TH0. The counter/timer in TL0 is controlled using the Timer 0 control/status bits in TCON and TMOD: TR0, C/T0, GATE0 and TF0. TL0 can use either the system clock or an external input signal as its timebase. The TH0 register is restricted to a timer function sourced by the system clock or prescaled clock. TH0 is enabled using the Timer 1 run control bit TR1. TH0 sets the Timer 1 overflow flag TF1 on overflow and thus controls the Timer 1 interrupt.

Timer 1 is inactive in Mode 3. When Timer 0 is operating in Mode 3, Timer 1 can be operated in Modes 0, 1 or 2, but cannot be clocked by external signals nor set the TF1 flag and generate an interrupt. However, the Timer 1 overflow can be used to generate baud rates for the SMBus and/or UART, and/or initiate ADC conversions. While Timer 0 is operating in Mode 3, Timer 1 run control is handled through its mode settings. To run Timer 1 while Timer 0 is in Mode 3, set the Timer 1 Mode as 0, 1, or 2. To disable Timer 1, configure it for Mode 3.

Not Recommended for New Designs

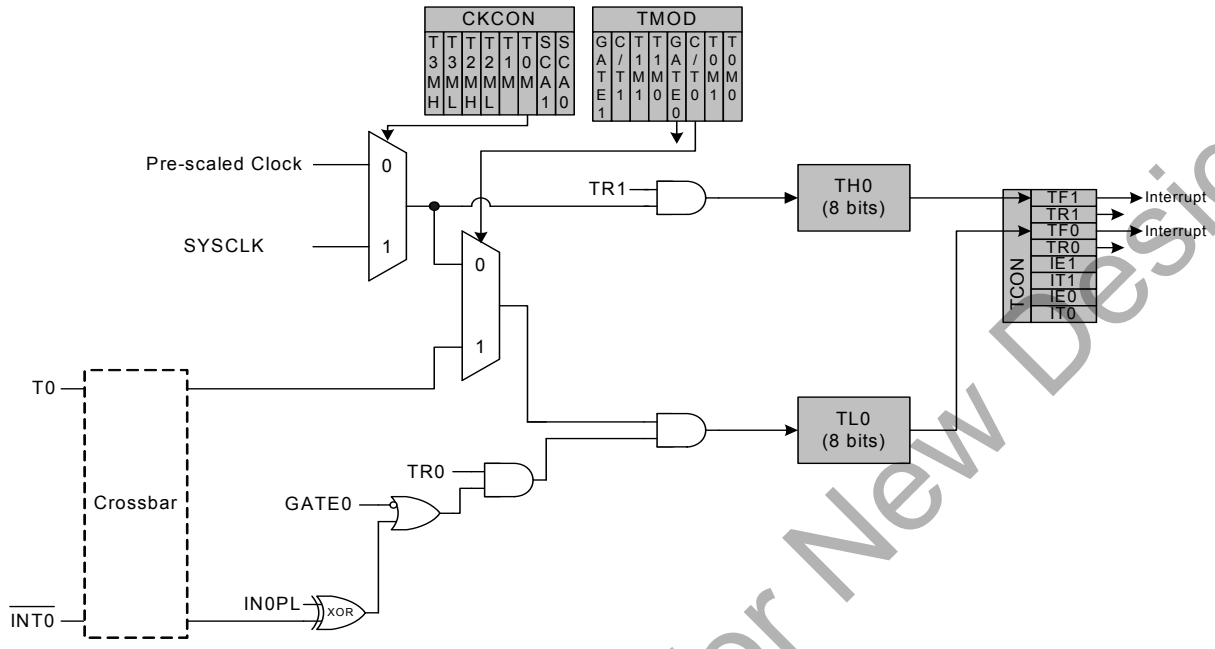



**Figure 33.3. T0 Mode 3 Block Diagram**

Not Recommended for New Designs

## SFR Definition 33.2. TCON: Timer Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TF1 | TR1 | TF0 | TR0 | IE1 | IT1 | IE0 | IT0 |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = All Pages; SFR Address = 0x88; Bit-Addressable

| Bit | Name | Function |
|---|---|---|
| 7 | TF1 | **Timer 1 Overflow Flag.**<br>Set to 1 by hardware when Timer 1 overflows. This flag can be cleared by software but is automatically cleared when the CPU vectors to the Timer 1 interrupt service routine. |
| 6 | TR1 | **Timer 1 Run Control.**<br>Timer 1 is enabled by setting this bit to 1. |
| 5 | TF0 | **Timer 0 Overflow Flag.**<br>Set to 1 by hardware when Timer 0 overflows. This flag can be cleared by software but is automatically cleared when the CPU vectors to the Timer 0 interrupt service routine. |
| 4 | TR0 | **Timer 0 Run Control.**<br>Timer 0 is enabled by setting this bit to 1. |
| 3 | IE1 | **External Interrupt 1.**<br>This flag is set by hardware when an edge/level of type defined by IT1 is detected. It can be cleared by software but is automatically cleared when the CPU vectors to the External Interrupt 1 service routine in edge-triggered mode. |
| 2 | IT1 | **Interrupt 1 Type Select.**<br>This bit selects whether the configured $\overline{\text{INT1}}$ interrupt will be edge or level sensitive. $\overline{\text{INT1}}$ is configured active low or high by the IN1PL bit in the IT01CF register (see SFR Definition 17.7).<br>0: $\overline{\text{INT1}}$ is level triggered.<br>1: $\overline{\text{INT1}}$ is edge triggered. |
| 1 | IE0 | **External Interrupt 0.**<br>This flag is set by hardware when an edge/level of type defined by IT1 is detected. It can be cleared by software but is automatically cleared when the CPU vectors to the External Interrupt 0 service routine in edge-triggered mode. |
| 0 | IT0 | **Interrupt 0 Type Select.**<br>This bit selects whether the configured $\overline{\text{INT0}}$ interrupt will be edge or level sensitive. $\overline{\text{INT0}}$ is configured active low or high by the IN0PL bit in register IT01CF (see SFR Definition 17.7).<br>0: $\overline{\text{INT0}}$ is level triggered.<br>1: $\overline{\text{INT0}}$ is edge triggered. |

Not Recommended for New Designs

## SFR Definition 33.3. TMOD: Timer Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | GATE1 | C/T1 | T1M[1:0] | | GATE0 | C/T0 | T0M[1:0] | |
| Type | R/W | R/W | R/W | | R/W | R/W | R/W | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x89

| Bit | Name | Function |
|---|---|---|
| 7 | GATE1 | **Timer 1 Gate Control.**<br>0: Timer 1 enabled when TR1 = 1 irrespective of $\overline{INT1}$ logic level.<br>1: Timer 1 enabled only when TR1 = 1 AND $\overline{INT1}$ is active as defined by bit IN1PL in register IT01CF (see SFR Definition 17.7). |
| 6 | C/T1 | **Counter/Timer 1 Select.**<br>0: Timer: Timer 1 incremented by clock defined by T1M bit in register CKCON.<br>1: Counter: Timer 1 incremented by high-to-low transitions on external pin (T1). |
| 5:4 | T1M[1:0] | **Timer 1 Mode Select.**<br>These bits select the Timer 1 operation mode.<br>00: Mode 0, 13-bit Counter/Timer<br>01: Mode 1, 16-bit Counter/Timer<br>10: Mode 2, 8-bit Counter/Timer with Auto-Reload<br>11: Mode 3, Timer 1 Inactive |
| 3 | GATE0 | **Timer 0 Gate Control.**<br>0: Timer 0 enabled when TR0 = 1 irrespective of $\overline{INT0}$ logic level.<br>1: Timer 0 enabled only when TR0 = 1 AND $\overline{INT0}$ is active as defined by bit IN0PL in register IT01CF (see SFR Definition 17.7). |
| 2 | C/T0 | **Counter/Timer 0 Select.**<br>0: Timer: Timer 0 incremented by clock defined by T0M bit in register CKCON.<br>1: Counter: Timer 0 incremented by high-to-low transitions on external pin (T0). |
| 1:0 | T0M[1:0] | **Timer 0 Mode Select.**<br>These bits select the Timer 0 operation mode.<br>00: Mode 0, 13-bit Counter/Timer<br>01: Mode 1, 16-bit Counter/Timer<br>10: Mode 2, 8-bit Counter/Timer with Auto-Reload<br>11: Mode 3, Two 8-bit Counter/Timers |

Not Recommended for New Designs

### 33.3.2. 8-Bit Timers with Auto-Reload

When T3SPLIT is set, Timer 3 operates as two 8-bit timers (TMR3H and TMR3L). Both 8-bit timers operate in auto-reload mode as shown in Figure 33.8. TMR3RLL holds the reload value for TMR3L; TMR3RLH holds the reload value for TMR3H. The TR3 bit in TMR3CN handles the run control for TMR3H. TMR3L is always running when configured for 8-bit Mode.

Each 8-bit timer may be configured to use SYSCLK, SYSCLK divided by 12, the external oscillator clock source divided by 8, or the SmaRTClock. The Timer 3 Clock Select bits (T3MH and T3ML in CKCON) select either SYSCLK or the clock defined by the Timer 3 External Clock Select bits (T3XCLK[1:0] in TMR3CN), as follows:

| T3MH | T3XCLK[1:0] | TMR3H Clock Source |
|---|---|---|
| 0 | 00 | SYSCLK / 12 |
| 0 | 01 | SmaRTClock |
| 0 | 10 | Reserved |
| 0 | 11 | External Clock / 8 |
| 1 | X | SYSCLK |

| T3ML | T3XCLK[1:0] | TMR3L Clock Source |
|---|---|---|
| 0 | 00 | SYSCLK / 12 |
| 0 | 01 | SmaRTClock |
| 0 | 10 | Reserved |
| 0 | 11 | External Clock / 8 |
| 1 | X | SYSCLK |

The TF3H bit is set when TMR3H overflows from 0xFF to 0x00; the TF3L bit is set when TMR3L overflows from 0xFF to 0x00. When Timer 3 interrupts are enabled, an interrupt is generated each time TMR3H overflows. If Timer 3 interrupts are enabled and TF3LEN (TMR3CN.5) is set, an interrupt is generated each time either TMR3L or TMR3H overflows. When TF3LEN is enabled, software must check the TF3H and TF3L flags to determine the source of the Timer 3 interrupt. The TF3H and TF3L interrupt flags are not cleared by hardware and must be manually cleared by software.

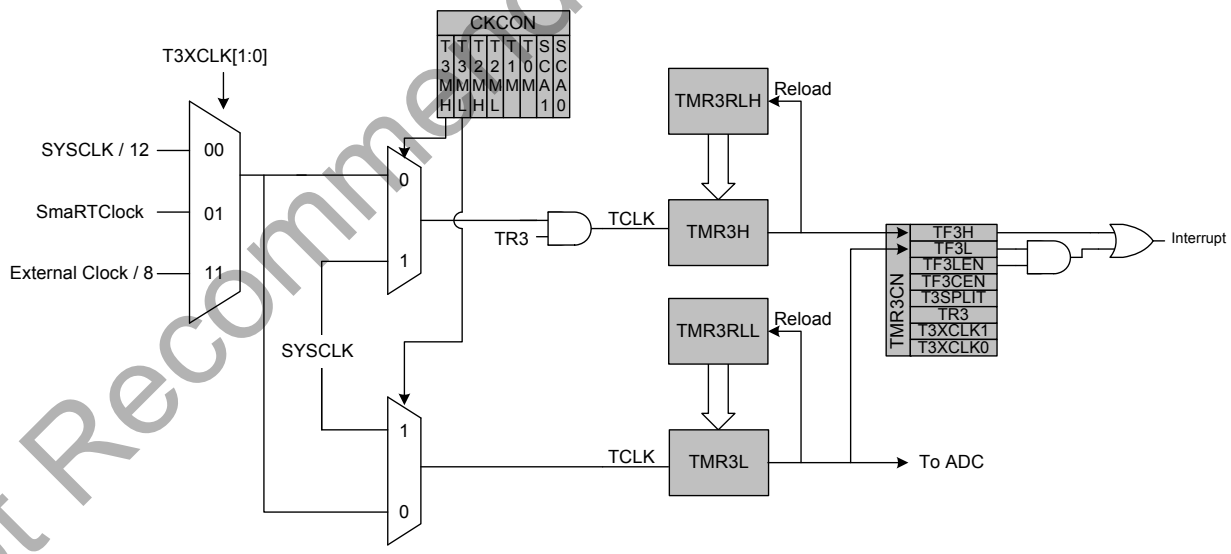



**Figure 33.8. Timer 3 8-Bit Mode Block Diagram**

### 33.3.3. SmaRTClock/External Oscillator Capture Mode

The Capture Mode in Timer 3 allows either SmaRTClock or the external oscillator period to be measured against the system clock or the system clock divided by 12. SmaRTClock and the external oscillator period can also be compared against each other.

Not Recommended for New Designs

Setting TF3CEN to 1 enables the SmaRTClock/External Oscillator Capture Mode for Timer 3. In this mode, T3SPLIT should be set to 0, as the full 16-bit timer is used.

When Capture Mode is enabled, a capture event will be generated either every SmaRTClock rising edge or every 8 external clock cycles, depending on the T3XCLK1 setting. When the capture event occurs, the contents of Timer 3 (TMR3H:TMR3L) are loaded into the Timer 3 reload registers (TMR3RLH:TMR3RLL) and the TF3H flag is set (triggering an interrupt if Timer 3 interrupts are enabled). By recording the difference between two successive timer capture values, the SmaRTClock or external clock period can be determined with respect to the Timer 3 clock. The Timer 3 clock should be much faster than the capture clock to achieve an accurate reading.

For example, if T3ML = 1b, T3XCLK1 = 0b, and TF3CEN = 1b, Timer 3 will clock every SYSCLK and capture every SmaRTClock rising edge. If SYSCLK is 24.5 MHz and the difference between two successive captures is 350 counts, then the SmaRTClock period is as follows:

350 x (1 / 24.5 MHz) = 14.2 µs.

This mode allows software to determine the exact frequency of the external oscillator in C and RC mode or the time between consecutive SmaRTClock rising edges, which is useful for determining the SmaRTClock frequency.

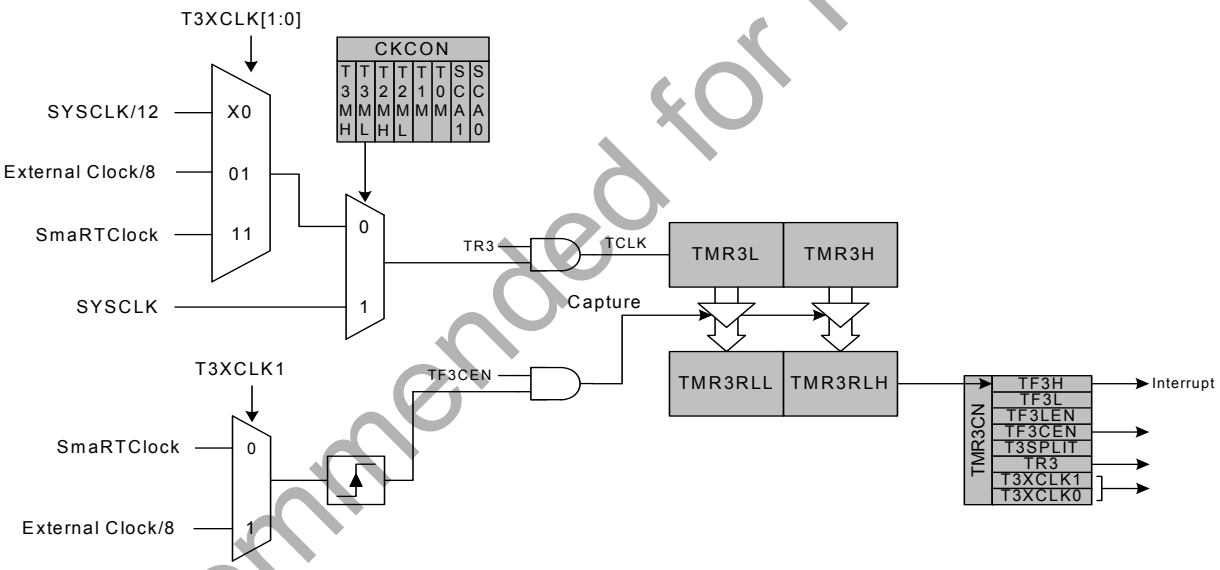



**Figure 33.9. Timer 3 Capture Mode Block Diagram**

Not Recommended for New Designs

## SFR Definition 33.13. TMR3CN: Timer 3 Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TF3H | TF3L | TF3LEN | TF3CEN | T3SPLIT | TR3 | T3XCLK[1:0] | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x91

| Bit | Name | Function |
|---|---|---|
| 7 | TF3H | **Timer 3 High Byte Overflow Flag.**<br>Set by hardware when the Timer 3 high byte overflows from 0xFF to 0x00. In 16 bit mode, this will occur when Timer 3 overflows from 0xFFFF to 0x0000. When the Timer 3 interrupt is enabled, setting this bit causes the CPU to vector to the Timer 3 interrupt service routine. This bit is not automatically cleared by hardware. |
| 6 | TF3L | **Timer 3 Low Byte Overflow Flag.**<br>Set by hardware when the Timer 3 low byte overflows from 0xFF to 0x00. TF3L will be set when the low byte overflows regardless of the Timer 3 mode. This bit is not automatically cleared by hardware. |
| 5 | TF3LEN | **Timer 3 Low Byte Interrupt Enable.**<br>When set to 1, this bit enables Timer 3 Low Byte interrupts. If Timer 3 interrupts are also enabled, an interrupt will be generated when the low byte of Timer 3 overflows. |
| 4 | TF3CEN | **Timer 3 SmaRTClock/External Oscillator Capture Enable.**<br>When set to 1, this bit enables Timer 3 Capture Mode. |
| 3 | T3SPLIT | **Timer 3 Split Mode Enable.**<br>When this bit is set, Timer 3 operates as two 8-bit timers with auto-reload.<br>0: Timer 3 operates in 16-bit auto-reload mode.<br>1: Timer 3 operates as two 8-bit auto-reload timers. |
| 2 | TR3 | **Timer 3 Run Control.**<br>Timer 3 is enabled by setting this bit to 1. In 8-bit mode, this bit enables/disables TMR3H only; TMR3L is always enabled in split mode. |
| 1:0 | T3XCLK[1:0] | **Timer 3 External Clock Select.**<br>This bit selects the “external” and “capture trigger” clock sources for Timer 3. If Timer 3 is in 8-bit mode, this bit selects the “external” clock source for both timer bytes. Timer 3 Clock Select bits (T3MH and T3ML in register CKCON) may still be used to select between the “external” clock and the system clock for either timer.<br>Note: External clock sources are synchronized with the system clock.<br>00: External Clock is SYSCLK /12. Capture trigger is SmaRTClock.<br>01: External Clock is External Oscillator/8. Capture trigger is SmaRTClock.<br>10: External Clock is SYSCLK/12. Capture trigger is External Oscillator/8.<br>11: External Clock is SmaRTClock. Capture trigger is External Oscillator/8. |

Not Recommended for New Designs

## SFR Definition 33.14. TMR3RLL: Timer 3 Reload Register Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR3RLL[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x92

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR3RLL[7:0] | **Timer 3 Reload Register Low Byte.**<br>TMR3RLL holds the low byte of the reload value for Timer 3. |

## SFR Definition 33.15. TMR3RLH: Timer 3 Reload Register High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR3RLH[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x93

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR3RLH[7:0] | **Timer 3 Reload Register High Byte.**<br>TMR3RLH holds the high byte of the reload value for Timer 3. |

Not Recommended for New Designs

## SFR Definition 33.16. TMR3L: Timer 3 Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR3L[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x94

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR3L[7:0] | **Timer 3 Low Byte.**<br>In 16-bit mode, the TMR3L register contains the low byte of the 16-bit Timer 3. In 8-bit mode, TMR3L contains the 8-bit low byte timer value. |

## SFR Definition 33.17. TMR3H Timer 3 High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR3H[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x95

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR3H[7:0] | **Timer 3 High Byte.**<br>In 16-bit mode, the TMR3H register contains the high byte of the 16-bit Timer 3. In 8-bit mode, TMR3H contains the 8-bit high byte timer value. |

Not Recommended for New Designs

# 34. Programmable Counter Array

The Programmable Counter Array (PCA0) provides enhanced timer functionality while requiring less CPU intervention than the standard 8051 counter/timers. The PCA consists of a dedicated 16-bit counter/timer and six 16-bit capture/compare modules. Each capture/compare module has its own associated I/O line (CEXn) which is routed through the Crossbar to Port I/O when enabled. The counter/timer is driven by a programmable timebase that can select between seven sources: system clock, system clock divided by four, system clock divided by twelve, the external oscillator clock source divided by 8, SmaRTClock divided by 8, Timer 0 overflows, or an external clock signal on the ECI input pin. Each capture/compare module may be configured to operate independently in one of six modes: Edge-Triggered Capture, Software Timer, High-Speed Output, Frequency Output, 8 to 11-Bit PWM, or 16-Bit PWM (each mode is described in Section “34.3. Capture/Compare Modules” on page 508). The external oscillator clock option is ideal for real-time clock (RTC) functionality, allowing the PCA to be clocked by a precision external oscillator while the internal oscillator drives the system clock. The PCA is configured and controlled through the system controller's Special Function Registers. The PCA block diagram is shown in Figure 34.1

**Important Note:** The PCA Module 5 may be used as a watchdog timer (WDT), and is enabled in this mode following a system reset. **Access to certain PCA registers is restricted while WDT mode is enabled**. See Section 34.4 for details.

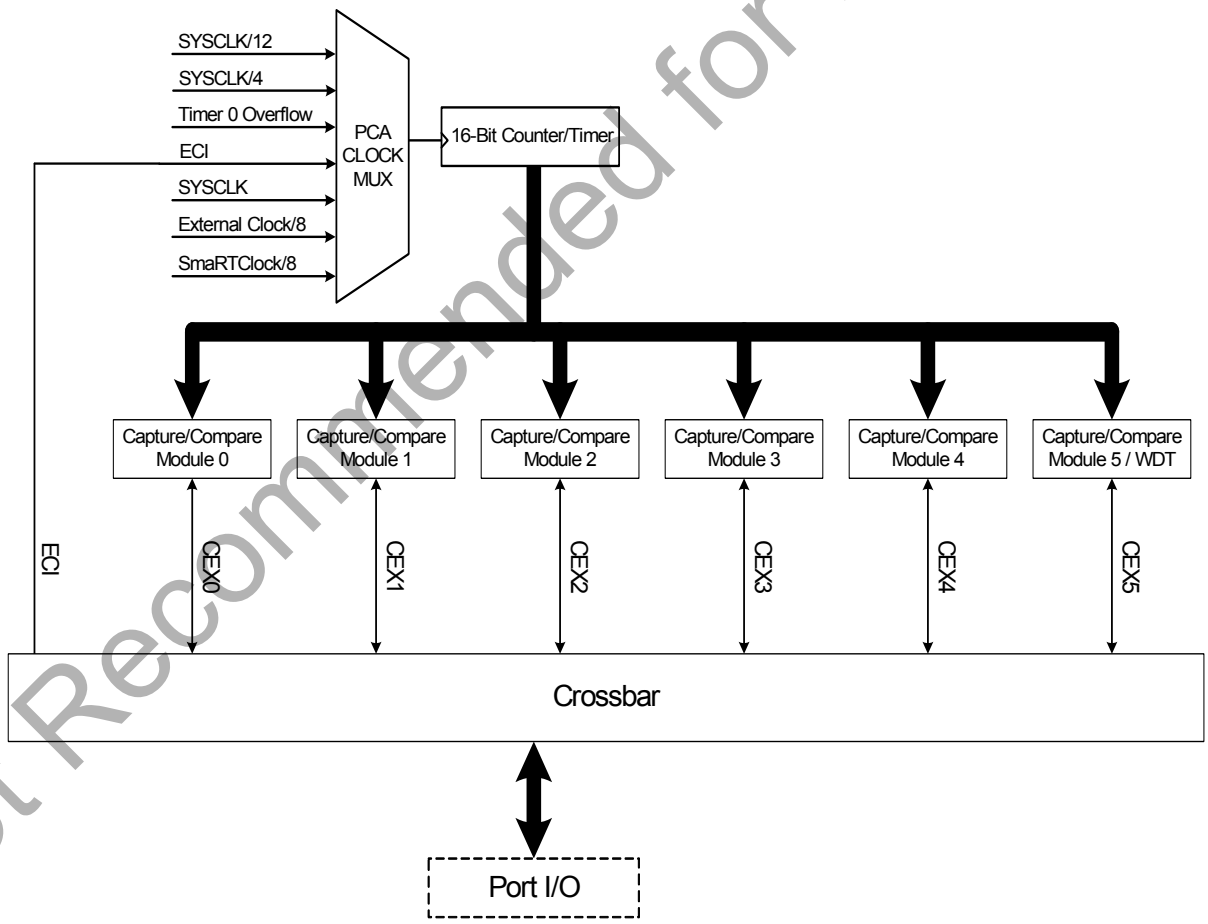



**Figure 34.1. PCA Block Diagram**

Not Recommended for New Designs

## 34.1. PCA Counter/Timer

The 16-bit PCA counter/timer consists of two 8-bit SFRs: PCA0L and PCA0H. PCA0H is the high byte (MSB) of the 16-bit counter/timer and PCA0L is the low byte (LSB). Reading PCA0L automatically latches the value of PCA0H into a “snapshot” register; the following PCA0H read accesses this “snapshot” register. **Reading the PCA0L Register first guarantees an accurate reading of the entire 16-bit PCA0 counter.** Reading PCA0H or PCA0L does not disturb the counter operation. The CPS2–CPS0 bits in the PCA0MD register select the timebase for the counter/timer as shown in Table 34.1.

When the counter/timer overflows from 0xFFFF to 0x0000, the Counter Overflow Flag (CF) in PCA0MD is set to logic 1 and an interrupt request is generated if CF interrupts are enabled. Setting the ECF bit in PCA0MD to logic 1 enables the CF flag to generate an interrupt request. The CF bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine, and must be cleared by software. Clearing the CIDL bit in the PCA0MD register allows the PCA to continue normal operation while the CPU is in Idle mode.

**Table 34.1. PCA Timebase Input Options**

| CPS2 | CPS1 | CPS0 | Timebase |
|---|---|---|---|
| 0 | 0 | 0 | System clock divided by 12 |
| 0 | 0 | 1 | System clock divided by 4 |
| 0 | 1 | 0 | Timer 0 overflow |
| 0 | 1 | 1 | High-to-low transitions on ECI (max rate = system clock divided by 4) |
| 1 | 0 | 0 | System clock |
| 1 | 0 | 1 | External oscillator source divided by 8[1] |
| 1 | 1 | 0 | SmaRTClock oscillator source divided by 8[2] |
| 1 | 1 | 1 | Reserved |

**Notes:**
1. External oscillator source divided by 8 is synchronized with the system clock.
2. SmaRTClock oscillator source divided by 8 is synchronized with the system clock.

Not Recommended for New Designs

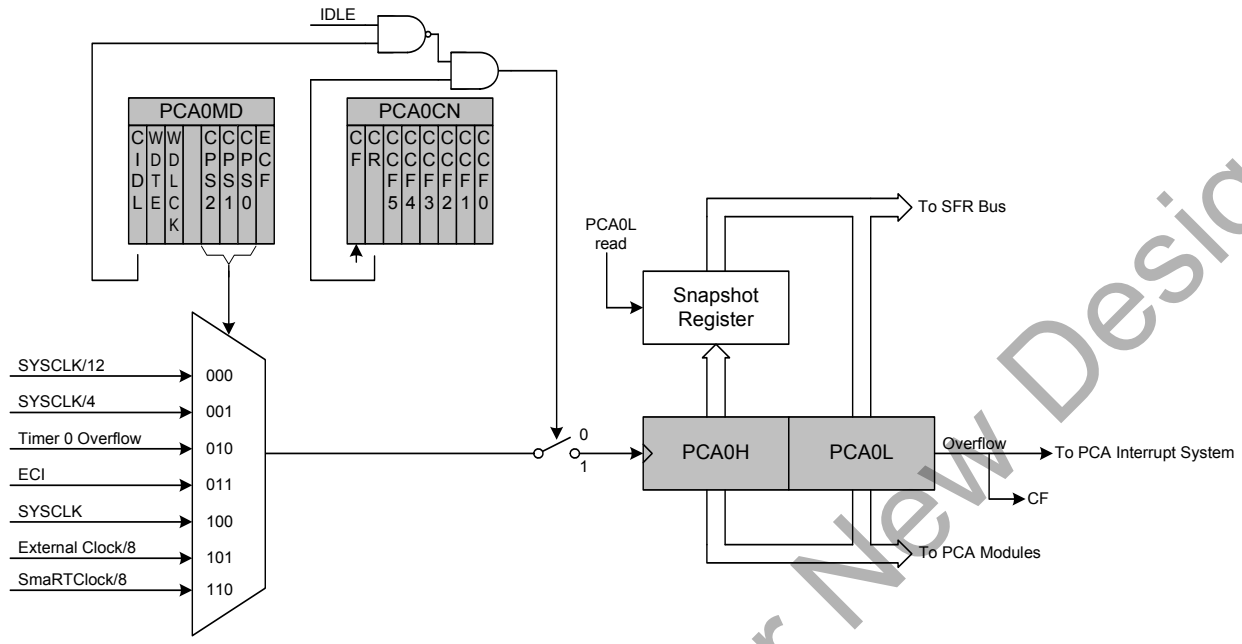



**Figure 34.2. PCA Counter/Timer Block Diagram**

## 34.2. PCA0 Interrupt Sources

Figure 34.3 shows a diagram of the PCA interrupt tree. There are eight independent event flags that can be used to generate a PCA0 interrupt. They are: the main PCA counter overflow flag (CF), which is set upon a 16-bit overflow of the PCA0 counter, an intermediate overflow flag (COVF), which can be set on an overflow from the 8th, 9th, 10th, or 11th bit of the PCA0 counter, and the individual flags for each PCA channel (CCF0, CCF1, CCF2, CCF3, CCF4, and CCF5), which are set according to the operation mode of that module. These event flags are always set when the trigger condition occurs. Each of these flags can be individually selected to generate a PCA0 interrupt, using the corresponding interrupt enable flag (ECF for CF, ECOV for COVF, and ECCFn for each CCFn). PCA0 interrupts must be globally enabled before any individual interrupt sources are recognized by the processor. PCA0 interrupts are globally enabled by setting the EA bit and the EPCA0 bit to logic 1.

Not Recommended for New Designs

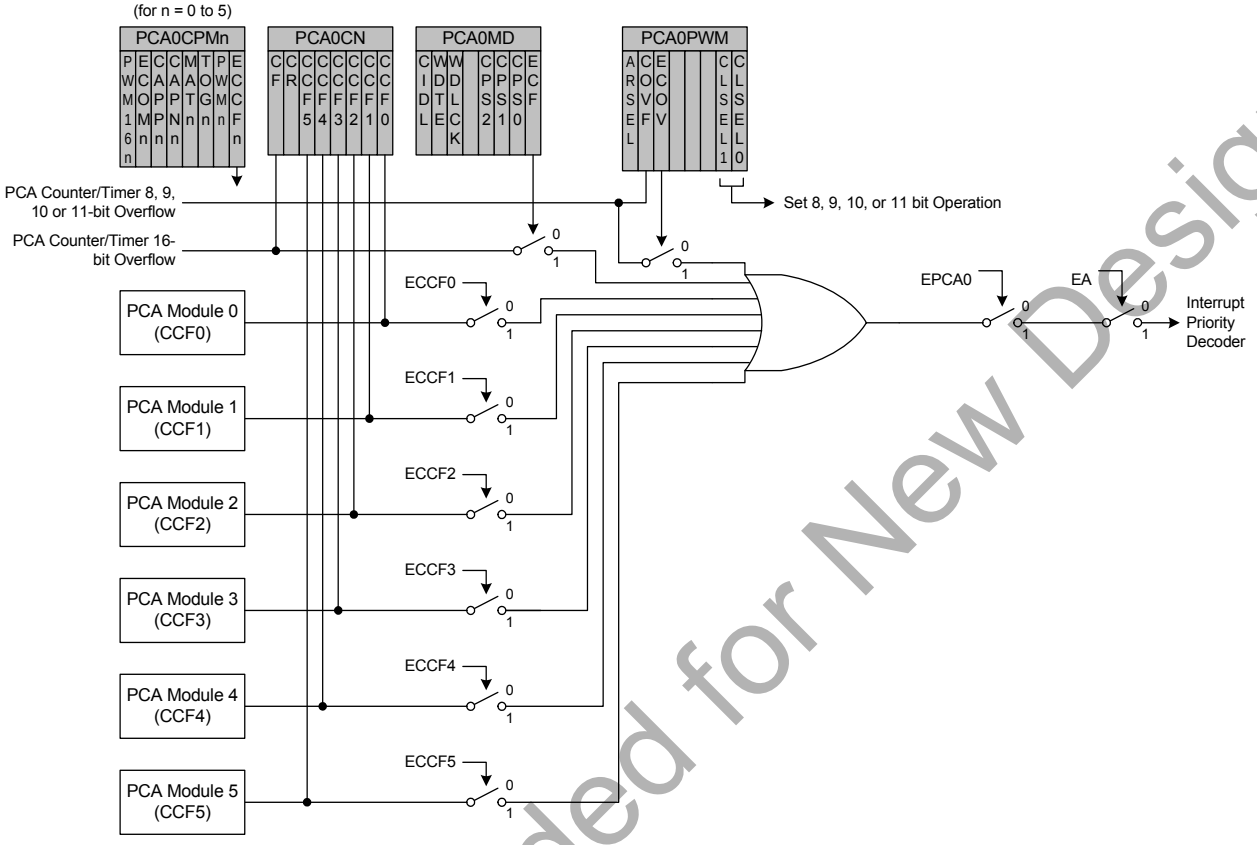



**Figure 34.3. PCA Interrupt Block Diagram**

## 34.3. Capture/Compare Modules

Each module can be configured to operate independently in one of six operation modes: edge-triggered capture, software timer, high speed output, frequency output, 8 to 11-bit pulse width modulator, or 16-bit pulse width modulator. Each module has Special Function Registers (SFRs) associated with it in the CIP-51 system controller. These registers are used to exchange data with a module and configure the module's mode of operation. Table 34.2 summarizes the bit settings in the PCA0CPMn and PCA0PWM registers used to select the PCA capture/compare module's operating mode. Note that all modules set to use 8, 9, 10, or 11-bit PWM mode must use the same cycle length (8-11 bits). Setting the ECCFn bit in a PCA0CPMn register enables the module's CCFn interrupt.

**Table 34.2. PCA0CPM and PCA0PWM Bit Settings for PCA Capture/Compare Modules**

| Operational Mode | PCA0CPMn | | | | | | | | PCA0PWM | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit Number | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 7 | 6 | 5 | 4–2 | 1–0 |
| Capture triggered by positive edge on CEXn | X | X | 1 | 0 | 0 | 0 | 0 | A | 0 | X | B | XXX | XX |
| Capture triggered by negative edge on CEXn | X | X | 0 | 1 | 0 | 0 | 0 | A | 0 | X | B | XXX | XX |
| Capture triggered by any transition on CEXn | X | X | 1 | 1 | 0 | 0 | 0 | A | 0 | X | B | XXX | XX |

Not Recommended for New Designs

**Table 34.2. PCA0CPM and PCA0PWM Bit Settings for PCA Capture/Compare Modules**

| Operational Mode | PCA0CPMn | | | | | | | | PCA0PWM | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Software Timer | X | C | 0 | 0 | 1 | 0 | 0 | A | 0 | X | B | XXX | XX |
| High Speed Output | X | C | 0 | 0 | 1 | 1 | 0 | A | 0 | X | B | XXX | XX |
| Frequency Output | X | C | 0 | 0 | 0 | 1 | 1 | A | 0 | X | B | XXX | XX |
| 8-Bit Pulse Width Modulator (Note 7) | 0 | C | 0 | 0 | E | 0 | 1 | A | 0 | X | B | XXX | 00 |
| 9-Bit Pulse Width Modulator (Note 7) | 0 | C | 0 | 0 | E | 0 | 1 | A | D | X | B | XXX | 01 |
| 10-Bit Pulse Width Modulator (Note 7) | 0 | C | 0 | 0 | E | 0 | 1 | A | D | X | B | XXX | 10 |
| 11-Bit Pulse Width Modulator (Note 7) | 0 | C | 0 | 0 | E | 0 | 1 | A | D | X | B | XXX | 11 |
| 16-Bit Pulse Width Modulator | 1 | C | 0 | 0 | E | 0 | 1 | A | 0 | X | B | XXX | XX |

**Notes:**
1. X = Don't Care (no functional difference for individual module if 1 or 0).
2. A = Enable interrupts for this module (PCA interrupt triggered on CCFn set to 1).
3. B = Enable 8th, 9th, 10th or 11th bit overflow interrupt (Depends on setting of CLSEL[1:0]).
4. C = When set to 0, the digital comparator is off. For high speed and frequency output modes, the associated pin will not toggle. In any of the PWM modes, this generates a 0% duty cycle (output = 0).
5. D = Selects whether the Capture/Compare register (0) or the Auto-Reload register (1) for the associated channel is accessed via addresses PCA0CPHn and PCA0CPLn.
6. E = When set, a match event will cause the CCFn flag for the associated channel to be set.
7. All modules set to 8, 9, 10 or 11-bit PWM mode use the same cycle length setting.

### 34.3.1. Edge-triggered Capture Mode

In this mode, a valid transition on the CEXn pin causes the PCA to capture the value of the PCA counter/timer and load it into the corresponding module's 16-bit capture/compare register (PCA0CPLn and PCA0CPHn). The CAPPn and CAPNn bits in the PCA0CPMn register are used to select the type of transition that triggers the capture: low-to-high transition (positive edge), high-to-low transition (negative edge), or either transition (positive or negative edge). When a capture occurs, the Capture/Compare Flag (CCFn) in PCA0CN is set to logic 1. An interrupt request is generated if the CCFn interrupt for that module is enabled. The CCFn bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine, and must be cleared by software. If both CAPPn and CAPNn bits are set to logic 1, then the state of the Port pin associated with CEXn can be read directly to determine whether a rising-edge or falling-edge caused the capture.

Not Recommended for New Designs

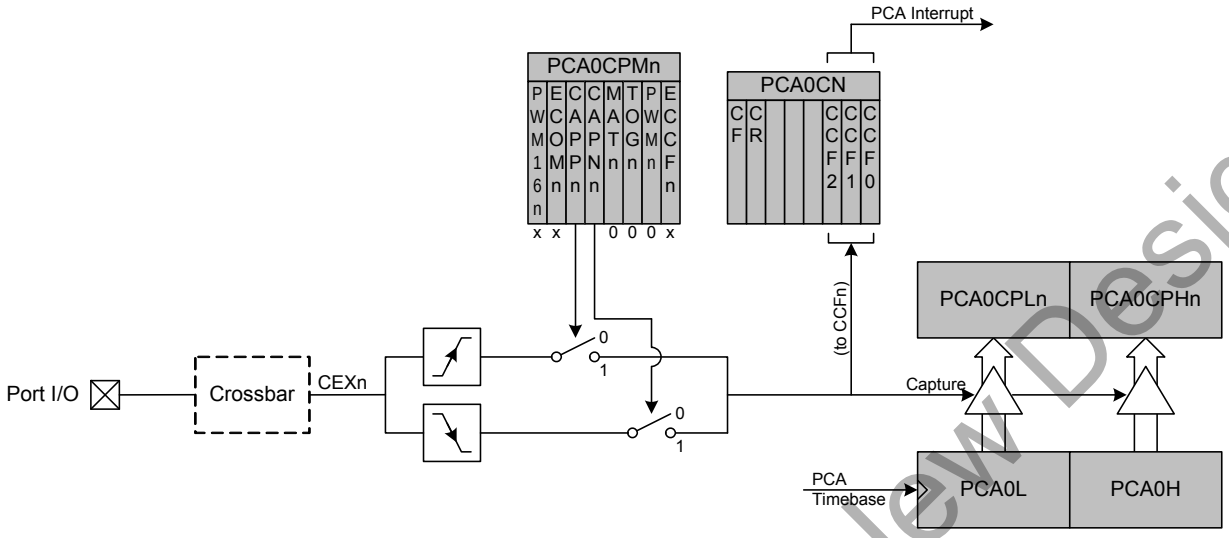



**Figure 34.4. PCA Capture Mode Diagram**

**Note:** The CEXn input signal must remain high or low for at least 2 system clock cycles to be recognized by the hardware.

### 34.3.2. Software Timer (Compare) Mode

In Software Timer mode, the PCA counter/timer value is compared to the module's 16-bit capture/compare register (PCA0CPHn and PCA0CPLn). When a match occurs, the Capture/Compare Flag (CCFn) in PCA0CN is set to logic 1. An interrupt request is generated if the CCFn interrupt for that module is enabled. The CCFn bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine, and must be cleared by software. Setting the ECOMn and MATn bits in the PCA0CPMn register enables Software Timer mode.

**Important Note About Capture/Compare Registers**: When writing a 16-bit value to the PCA0 Capture/Compare registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to 0; writing to PCA0CPHn sets ECOMn to 1.

Not Recommended for New Designs

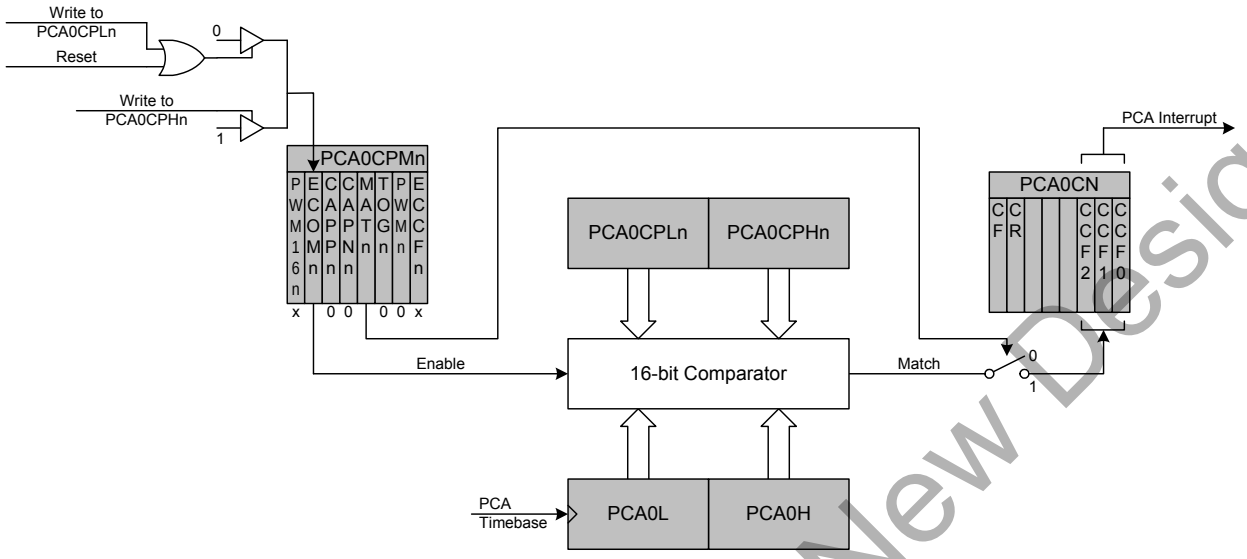



**Figure 34.5. PCA Software Timer Mode Diagram**

### 34.3.3. High-Speed Output Mode

In High-Speed Output mode, a module's associated CEXn pin is toggled each time a match occurs between the PCA Counter and the module's 16-bit capture/compare register (PCA0CPHn and PCA0CPLn). When a match occurs, the Capture/Compare Flag (CCFn) in PCA0CN is set to logic 1. An interrupt request is generated if the CCFn interrupt for that module is enabled. The CCFn bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine, and must be cleared by software. Setting the TOGn, MATn, and ECOMn bits in the PCA0CPMn register enables the High-Speed Output mode. If ECOMn is cleared, the associated pin will retain its state, and not toggle on the next match event.

**Important Note About Capture/Compare Registers**: When writing a 16-bit value to the PCA0 Capture/Compare registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to 0; writing to PCA0CPHn sets ECOMn to 1.

Not Recommended for New Designs

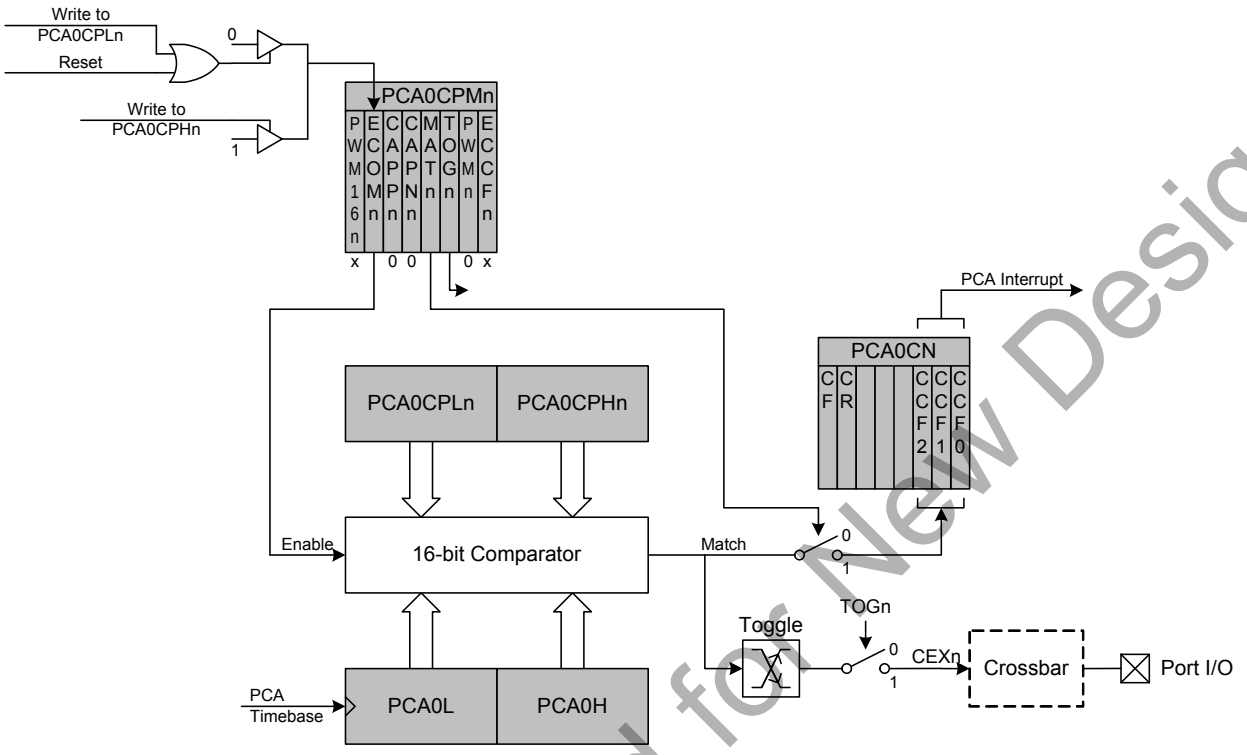



**Figure 34.6. PCA High-Speed Output Mode Diagram**

### 34.3.4. Frequency Output Mode

Frequency Output Mode produces a programmable-frequency square wave on the module's associated CEXn pin. The capture/compare module high byte holds the number of PCA clocks to count before the output is toggled. The frequency of the square wave is then defined by Equation 34.1.

$$F_{CEXn} = \frac{F_{PCA}}{2 \times PCA0CPHn}$$

**Note:** A value of 0x00 in the PCA0CPHn register is equal to 256 for this equation.

**Equation 34.1. Square Wave Frequency Output**

Where $F_{PCA}$ is the frequency of the clock selected by the CPS2–0 bits in the PCA mode register, PCA0MD. The lower byte of the capture/compare module is compared to the PCA counter low byte; on a match, CEXn is toggled and the offset held in the high byte is added to the matched value in PCA0CPLn. Frequency Output Mode is enabled by setting the ECOMn, TOGn, and PWMn bits in the PCA0CPMn register. The MATn bit should normally be set to 0 in this mode. If the MATn bit is set to 1, the CCFn flag for the channel will be set when the 16-bit PCA0 counter and the 16-bit capture/compare register for the channel are equal.

Not Recommended for New Designs

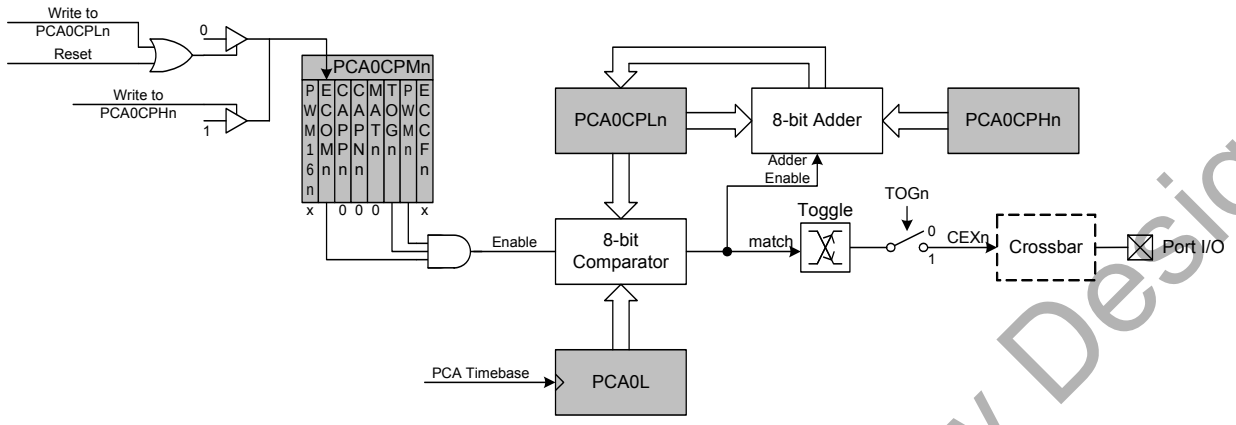



**Figure 34.7. PCA Frequency Output Mode**

### 34.3.5. 8-Bit, 9-Bit, 10-Bit and 11-Bit Pulse Width Modulator Modes

Each module can be used independently to generate a pulse width modulated (PWM) output on its associated CEXn pin. The frequency of the output is dependent on the timebase for the PCA counter/timer, and the setting of the PWM cycle length (8, 9, 10 or 11-bits). For backwards-compatibility with the 8-bit PWM mode available on other devices, the 8-bit PWM mode operates slightly different than 9, 10 and 11-bit PWM modes. **It is important to note that all channels configured for 8/9/10/11-bit PWM mode will use the same cycle length.** It is not possible to configure one channel for 8-bit PWM mode and another for 11-bit mode (for example). However, other PCA channels can be configured to Pin Capture, High-Speed Output, Software Timer, Frequency Output, or 16-bit PWM mode independently.

#### 34.3.5.1. 8-Bit Pulse Width Modulator Mode

The duty cycle of the PWM output signal in 8-bit PWM mode is varied using the module's PCA0CPLn capture/compare register. When the value in the low byte of the PCA counter/timer (PCA0L) is equal to the value in PCA0CPLn, the output on the CEXn pin will be set. When the count value in PCA0L overflows, the CEXn output will be reset (see Figure 34.8). Also, when the counter/timer low byte (PCA0L) overflows from 0xFF to 0x00, PCA0CPLn is reloaded automatically with the value stored in the module's capture/compare high byte (PCA0CPHn) without software intervention. Setting the ECOMn and PWMn bits in the PCA0CPMn register, and setting the CLSEL bits in register PCA0PWM to 00b enables 8-Bit Pulse Width Modulator mode. If the MATn bit is set to 1, the CCFn flag for the module will be set each time an 8-bit comparator match (rising edge) occurs. The COVF flag in PCA0PWM can be used to detect the overflow (falling edge), which will occur every 256 PCA clock cycles. The duty cycle for 8-Bit PWM Mode is given in Equation 34.2.

**Important Note About Capture/Compare Registers**: When writing a 16-bit value to the PCA0 Capture/Compare registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to 0; writing to PCA0CPHn sets ECOMn to 1.

$$\text{Duty Cycle} = \frac{(256 - PCA0CPHn)}{256}$$

**Equation 34.2. 8-Bit PWM Duty Cycle**

Using Equation 34.2, the largest duty cycle is 100% (PCA0CPHn = 0), and the smallest duty cycle is 0.39% (PCA0CPHn = 0xFF). A 0% duty cycle may be generated by clearing the ECOMn bit to 0.

Not Recommended for New Designs

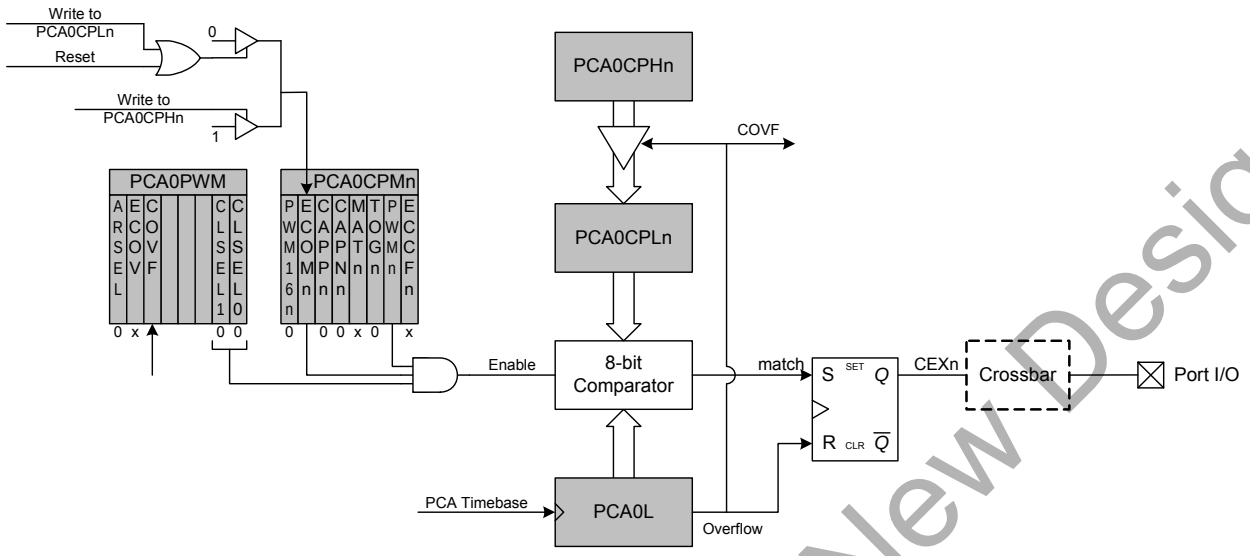



**Figure 34.8. PCA 8-Bit PWM Mode Diagram**

### 34.3.5.2. 9/10/11-bit Pulse Width Modulator Mode

The duty cycle of the PWM output signal in 9/10/11-bit PWM mode should be varied by writing to an "Auto-Reload" Register, which is dual-mapped into the PCA0CPHn and PCA0CPLn register locations. The data written to define the duty cycle should be right-justified in the registers. The auto-reload registers are accessed (read or written) when the bit ARSEL in PCA0PWM is set to 1. The capture/compare registers are accessed when ARSEL is set to 0.

When the least-significant N bits of the PCA0 counter match the value in the associated module's capture/compare register (PCA0CPn), the output on CEXn is asserted high. When the counter overflows from the Nth bit, CEXn is asserted low (see Figure 34.9). Upon an overflow from the Nth bit, the COVF flag is set, and the value stored in the module's auto-reload register is loaded into the capture/compare register. The value of N is determined by the CLSEL bits in register PCA0PWM.

The 9, 10 or 11-bit PWM mode is selected by setting the ECOMn and PWMn bits in the PCA0CPMn register, and setting the CLSEL bits in register PCA0PWM to the desired cycle length (other than 8-bits). If the MATn bit is set to 1, the CCFn flag for the module will be set each time a comparator match (rising edge) occurs. The COVF flag in PCA0PWM can be used to detect the overflow (falling edge), which will occur every 512 (9-bit), 1024 (10-bit) or 2048 (11-bit) PCA clock cycles. The duty cycle for 9/10/11-Bit PWM Mode is given in Equation 34.3, where N is the number of bits in the PWM cycle.

**Important Note About PCA0CPHn and PCA0CPLn Registers**: When writing a 16-bit value to the PCA0CPn registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to 0; writing to PCA0CPHn sets ECOMn to 1.

$$\text{Duty Cycle} = \frac{(2^N - PCA0CPn)}{2^N}$$

**Equation 34.3. 9, 10, and 11-Bit PWM Duty Cycle**

A 0% duty cycle may be generated by clearing the ECOMn bit to 0.

Not Recommended for New Designs

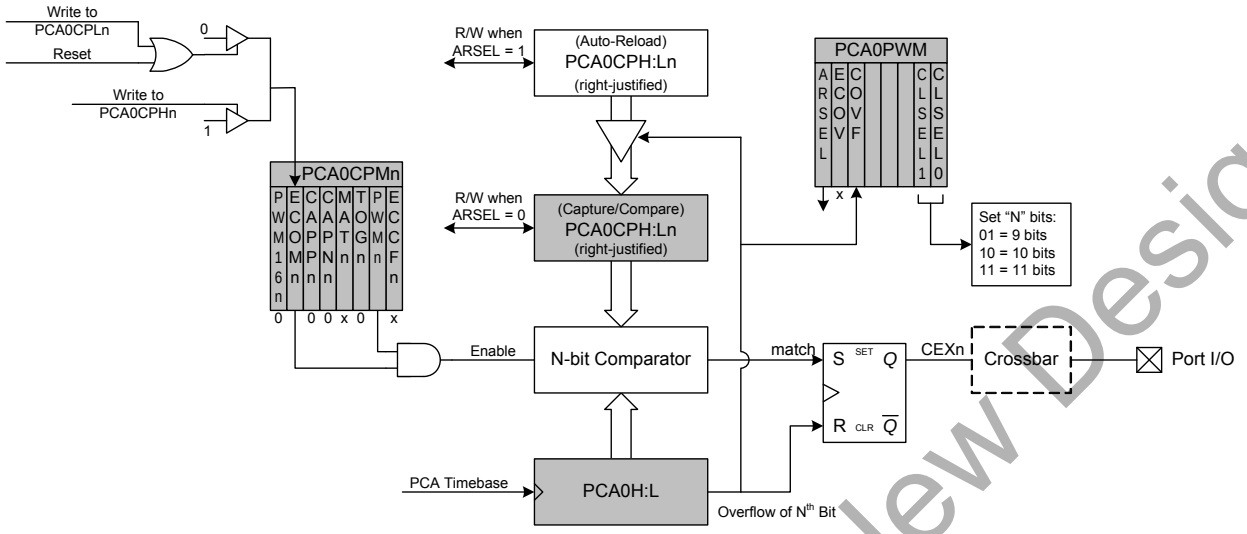



**Figure 34.9. PCA 9, 10 and 11-Bit PWM Mode Diagram**

### 34.3.6. 16-Bit Pulse Width Modulator Mode

A PCA module may also be operated in 16-Bit PWM mode. 16-bit PWM mode is independent of the other (8/9/10/11-bit) PWM modes. In this mode, the 16-bit capture/compare module defines the number of PCA clocks for the low time of the PWM signal. When the PCA counter matches the module contents, the output on CEXn is asserted high; when the 16-bit counter overflows, CEXn is asserted low. To output a varying duty cycle, new value writes should be synchronized with PCA CCFn match interrupts. 16-Bit PWM Mode is enabled by setting the ECOMn, PWMn, and PWM16n bits in the PCA0CPMn register. For a varying duty cycle, match interrupts should be enabled (ECCFn = 1 AND MATn = 1) to help synchronize the capture/compare register writes. If the MATn bit is set to 1, the CCFn flag for the module will be set each time a 16-bit comparator match (rising edge) occurs. The CF flag in PCA0CN can be used to detect the overflow (falling edge). The duty cycle for 16-Bit PWM Mode is given by Equation 34.4.

**Important Note About Capture/Compare Registers**: When writing a 16-bit value to the PCA0 Capture/Compare registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to 0; writing to PCA0CPHn sets ECOMn to 1.

$$\text{Duty Cycle} = \frac{(65536 - PCA0CPn)}{65536}$$

**Equation 34.4. 16-Bit PWM Duty Cycle**

Using Equation 34.4, the largest duty cycle is 100% (PCA0CPn = 0), and the smallest duty cycle is 0.0015% (PCA0CPn = 0xFFFF). A 0% duty cycle may be generated by clearing the ECOMn bit to 0.

Not Recommended for New Designs

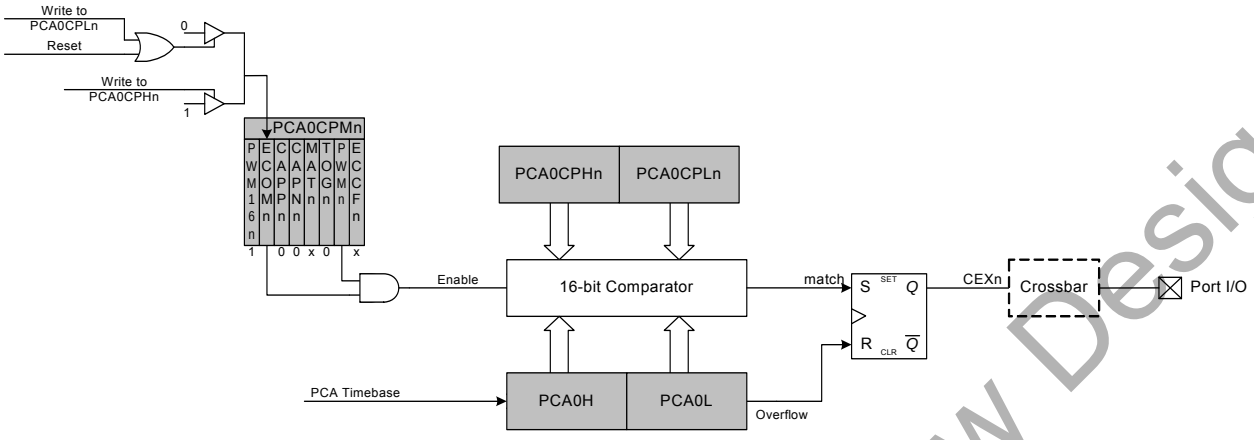



**Figure 34.10. PCA 16-Bit PWM Mode**

## 34.4. Watchdog Timer Mode

A programmable watchdog timer (WDT) function is available through the PCA Module 5. The WDT is used to generate a reset if the time between writes to the WDT update register (PCA0CPH2) exceed a specified limit. The WDT can be configured and enabled/disabled as needed by software.

With the WDTE bit set in the PCA0MD register, Module 5 operates as a watchdog timer (WDT). The Module 5 high byte is compared to the PCA counter high byte; the Module 5 low byte holds the offset to be used when WDT updates are performed. **The watchdog timer is enabled on reset. Writes to some PCA registers are restricted while the watchdog timer is enabled.** The WDT will generate a reset shortly after code begins execution. To avoid this reset, the WDT should be explicitly disabled (and optionally re-configured and re-enabled if it is used in the system).

### 34.4.1. Watchdog Timer Operation

While the WDT is enabled:

- PCA counter is forced on.
- Writes to PCA0L and PCA0H are not allowed.
- PCA clock source bits (CPS2–CPS0) are frozen.
- PCA Idle control bit (CIDL) is frozen.
- Module 5 is forced into software timer mode.
- Writes to the Module 5 mode register (PCA0CPM5) are disabled.

While the WDT is enabled, writes to the CR bit will not change the PCA counter state; the counter will run until the WDT is disabled. The PCA counter run control bit (CR) will read zero if the WDT is enabled but user software has not enabled the PCA counter. If a match occurs between PCA0CPH5 and PCA0H while the WDT is enabled, a reset will be generated. To prevent a WDT reset, the WDT may be updated with a write of any value to PCA0CPH5. Upon a PCA0CPH5 write, PCA0H plus the offset held in PCA0CPL5 is loaded into PCA0CPH5. (See Figure 34.11.)

Not Recommended for New Designs

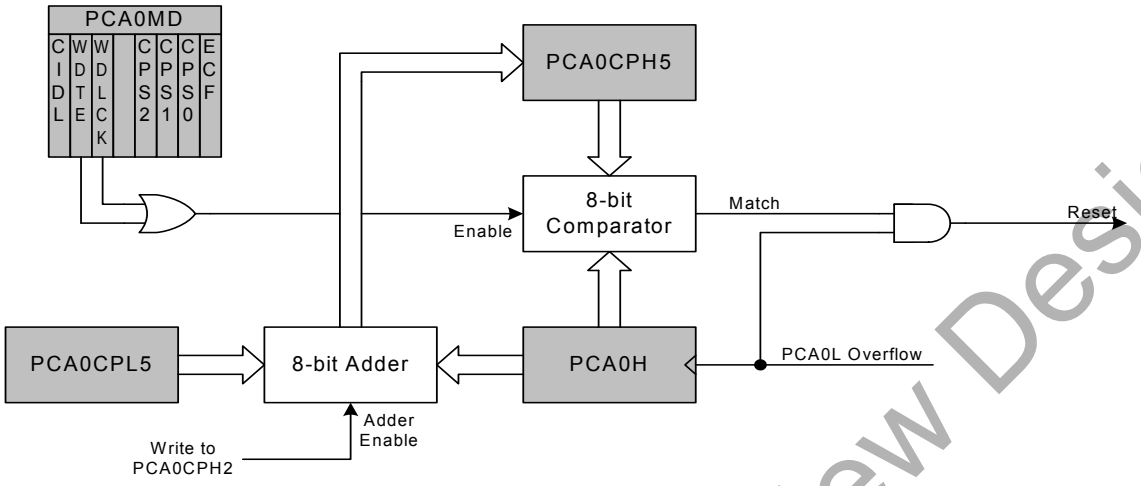



**Figure 34.11. PCA Module 5 with Watchdog Timer Enabled**

Note that the 8-bit offset held in PCA0CPH5 is compared to the upper byte of the 16-bit PCA counter. This offset value is the number of PCA0L overflows before a reset. Up to 256 PCA clocks may pass before the first PCA0L overflow occurs, depending on the value of the PCA0L when the update is performed. The total offset is then given (in PCA clocks) by Equation 34.5, where PCA0L is the value of the PCA0L register at the time of the update.

$$Offset = (256 \times PCA0CPL5) + (256 - PCA0L)$$

**Equation 34.5. Watchdog Timer Offset in PCA Clocks**

The WDT reset is generated when PCA0L overflows while there is a match between PCA0CPH5 and PCA0H. Software may force a WDT reset by writing a 1 to the CCF5 flag (PCA0CN.5) while the WDT is enabled.

### 34.4.2. Watchdog Timer Usage

To configure the WDT, perform the following tasks:

- Disable the WDT by writing a 0 to the WDTE bit.
- Select the desired PCA clock source (with the CPS2–CPS0 bits).
- Load PCA0CPL5 with the desired WDT update offset value.
- Configure the PCA Idle mode (set CIDL if the WDT should be suspended while the CPU is in Idle mode).
- Enable the WDT by setting the WDTE bit to 1.
- Reset the WDT timer by writing to PCA0CPH5.

The PCA clock source and idle mode select cannot be changed while the WDT is enabled. The watchdog timer is enabled by setting the WDTE or WDLCK bits in the PCA0MD register. When WDLCK is set, the WDT cannot be disabled until the next system reset. If WDLCK is not set, the WDT is disabled by clearing the WDTE bit.

The WDT is enabled following any reset. The PCA0 counter clock defaults to the system clock divided by 12, PCA0L defaults to 0x00, and PCA0CPL5 defaults to 0x00. Using Equation 34.5, this results in a WDT timeout interval of 256 PCA clock cycles, or 3072 system clock cycles. Table 34.3 lists some example timeout intervals for typical system clocks.

Not Recommended for New Designs

**Table 34.3. Watchdog Timer Timeout Intervals[1]**

| System Clock (Hz) | PCA0CPL5 | Timeout Interval (ms) |
|---|---|---|
| 24,500,000 | 255 | 32.1 |
| 24,500,000 | 128 | 16.2 |
| 24,500,000 | 32 | 4.1 |
| 3,062,500[2] | 255 | 257 |
| 3,062,500[2] | 128 | 129.5 |
| 3,062,500[2] | 32 | 33.1 |
| 32,000 | 255 | 24576 |
| 32,000 | 128 | 12384 |
| 32,000 | 32 | 3168 |

**Notes:**
1. Assumes SYSCLK/12 as the PCA clock source, and a PCA0L value of 0x00 at the update time.
2. Internal SYSCLK reset frequency = Internal Oscillator divided by 8.

Not Recommended for New Designs

## 34.5. Register Descriptions for PCA0

Following are detailed descriptions of the special function registers related to the operation of the PCA.

### SFR Definition 34.1. PCA0CN: PCA Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CF | CR | CCF5 | CCF4 | CCF3 | CCF2 | CCF1 | CCF0 |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xD8; Bit-Addressable

| Bit | Name | Function |
|---|---|---|
| 7 | CF | **PCA Counter/Timer Overflow Flag.**<br>Set by hardware when the PCA Counter/Timer overflows from 0xFFFF to 0x0000. When the Counter/Timer Overflow (CF) interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by software. |
| 6 | CR | **PCA Counter/Timer Run Control.**<br>This bit enables/disables the PCA Counter/Timer.<br>0: PCA Counter/Timer disabled.<br>1: PCA Counter/Timer enabled. |
| 5:0 | CCF[5:0] | **PCA Module n Capture/Compare Flag.**<br>These bits are set by hardware when a match or capture occurs in the associated PCA Module n. When the CCFn interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by software. |

Not Recommended for New Designs

## SFR Definition 34.2. PCA0MD: PCA Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CIDL | WDTE | WDLCK | | CPS2 | CPS1 | CPS0 | ECF |
| Type | R/W | R/W | R/W | R | R/W | R/W | R/W | R/W |
| Reset | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xD9

| Bit | Name | Function |
|---|---|---|
| 7 | CIDL | **PCA Counter/Timer Idle Control.**<br>Specifies PCA behavior when CPU is in Idle Mode.<br>0: PCA continues to function normally while the system controller is in Idle Mode.<br>1: PCA operation is suspended while the system controller is in Idle Mode. |
| 6 | WDTE | **Watchdog Timer Enable.**<br>If this bit is set, PCA Module 5 is used as the watchdog timer.<br>0: Watchdog timer disabled.<br>1: PCA Module 5 enabled as watchdog timer. |
| 5 | WDLCK | **Watchdog Timer Lock.**<br>This bit locks/unlocks the watchdog timer enable. When WDLCK is set, the watchdog timer may not be disabled until the next system reset.<br>0: Watchdog timer enable unlocked.<br>1: Watchdog timer enable locked. |
| 4 | Unused | Read = 0b, Write = don't care. |
| 3:1 | CPS[2:0] | **PCA Counter/Timer Pulse Select.**<br>These bits select the timebase source for the PCA counter<br>000: System clock divided by 12<br>001: System clock divided by 4<br>010: Timer 0 overflow<br>011: High-to-low transitions on ECI (max rate = system clock divided by 4)<br>100: System clock<br>101: External clock divided by 8 (synchronized with the system clock)<br>110: SmaRTClock divided by 8 (synchronized with the system clock)<br>111: Reserved |
| 0 | ECF | **PCA Counter/Timer Overflow Interrupt Enable.**<br>This bit sets the masking of the PCA Counter/Timer Overflow (CF) interrupt.<br>0: Disable the CF interrupt.<br>1: Enable a PCA Counter/Timer Overflow interrupt request when CF (PCA0CN.7) is set. |
| **Note:** When the WDTE bit is set to 1, the other bits in the PCA0MD register cannot be modified. To change the contents of the PCA0MD register, the watchdog timer must first be disabled. | | |

Not Recommended for New Designs

## SFR Definition 34.3. PCA0PWM: PCA PWM Configuration

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ARSEL | ECOV | COVF | | | | CLSEL[1:0] | |
| Type | R/W | R/W | R/W | R | R | R | R/W | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xDF

| Bit | Name | Function |
|---|---|---|
| 7 | ARSEL | **Auto-Reload Register Select.**<br>This bit selects whether to read and write the normal PCA capture/compare registers (PCA0CPn), or the Auto-Reload registers at the same SFR addresses. This function is used to define the reload value for 9, 10, and 11-bit PWM modes. In all other modes, the Auto-Reload registers have no function.<br>0: Read/Write Capture/Compare Registers at PCA0CPHn and PCA0CPLn.<br>1: Read/Write Auto-Reload Registers at PCA0CPHn and PCA0CPLn. |
| 6 | ECOV | **Cycle Overflow Interrupt Enable.**<br>This bit sets the masking of the Cycle Overflow Flag (COVF) interrupt.<br>0: COVF will not generate PCA interrupts.<br>1: A PCA interrupt will be generated when COVF is set. |
| 5 | COVF | **Cycle Overflow Flag.**<br>This bit indicates an overflow of the 8th, 9th, 10th, or 11th bit of the main PCA counter (PCA0). The specific bit used for this flag depends on the setting of the Cycle Length Select bits. The bit can be set by hardware or software, but must be cleared by software.<br>0: No overflow has occurred since the last time this bit was cleared.<br>1: An overflow has occurred since the last time this bit was cleared. |
| 4:2 | Unused | Read = 000b; Write = don't care. |
| 1:0 | CLSEL[1:0] | **Cycle Length Select.**<br>When 16-bit PWM mode is not selected, these bits select the length of the PWM cycle, between 8, 9, 10, or 11 bits. This affects all channels configured for PWM which are not using 16-bit PWM mode. These bits are ignored for individual channels configured to16-bit PWM mode.<br>00: 8 bits.<br>01: 9 bits.<br>10: 10 bits.<br>11: 11 bits. |

Not Recommended for New Designs

## SFR Definition 34.4. PCA0CPMn: PCA Capture/Compare Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PWM16n | ECOMn | CAPPn | CAPNn | MATn | TOGn | PWMn | ECCFn |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Address, Page: PCA0CPM0 = 0xDA, 0x0; PCA0CPM1 = 0xDB, 0x0; PCA0CPM2 = 0xDC, 0x0
PCA0CPM3 = 0xDD, 0x0; PCA0CPM4 = 0xDE, 0x0; PCA0CPM5 = 0xCE, 0x0

| Bit | Name | Function |
|---|---|---|
| 7 | PWM16n | **16-bit Pulse Width Modulation Enable.**<br>This bit enables 16-bit mode when Pulse Width Modulation mode is enabled.<br>0: 8 to 11-bit PWM selected.<br>1: 16-bit PWM selected. |
| 6 | ECOMn | **Comparator Function Enable.**<br>This bit enables the comparator function for PCA module n when set to 1. |
| 5 | CAPPn | **Capture Positive Function Enable.**<br>This bit enables the positive edge capture for PCA module n when set to 1. |
| 4 | CAPNn | **Capture Negative Function Enable.**<br>This bit enables the negative edge capture for PCA module n when set to 1. |
| 3 | MATn | **Match Function Enable.**<br>This bit enables the match function for PCA module n when set to 1. When enabled, matches of the PCA counter with a module's capture/compare register cause the CCFn bit in PCA0MD register to be set to logic 1. |
| 2 | TOGn | **Toggle Function Enable.**<br>This bit enables the toggle function for PCA module n when set to 1. When enabled, matches of the PCA counter with a module's capture/compare register cause the logic level on the CEXn pin to toggle. If the PWMn bit is also set to logic 1, the module operates in Frequency Output Mode. |
| 1 | PWMn | **Pulse Width Modulation Mode Enable.**<br>This bit enables the PWM function for PCA module n when set to 1. When enabled, a pulse width modulated signal is output on the CEXn pin. 8 to 11-bit PWM is used if PWM16n is cleared; 16-bit mode is used if PWM16n is set to logic 1. If the TOGn bit is also set, the module operates in Frequency Output Mode. |
| 0 | ECCFn | **Capture/Compare Flag Interrupt Enable.**<br>This bit sets the masking of the Capture/Compare Flag (CCFn) interrupt.<br>0: Disable CCFn interrupts.<br>1: Enable a Capture/Compare Flag interrupt request when CCFn is set. |
| **Note:** When the WDTE bit is set to 1, the PCA0CPM5 register cannot be modified, and module 5 acts as the watchdog timer. To change the contents of the PCA0CPM5 register or the function of module 5, the watchdog timer must be disabled. | | |

Not Recommended for New Designs

## SFR Definition 34.5. PCA0L: PCA Counter/Timer Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PCA0[7:0] | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xF9

| Bit | Name | Function |
|---|---|---|
| 7:0 | PCA0[7:0] | **PCA Counter/Timer Low Byte.**<br>The PCA0L register holds the low byte (LSB) of the 16-bit PCA Counter/Timer. |

**Note:** When the WDTE bit is set to 1, the PCA0L register cannot be modified by software. To change the contents of the PCA0L register, the watchdog Timer must first be disabled.

## SFR Definition 34.6. PCA0H: PCA Counter/Timer High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PCA0[15:8] | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xFA

| Bit | Name | Function |
|---|---|---|
| 7:0 | PCA0[15:8] | **PCA Counter/Timer High Byte.**<br>The PCA0H register holds the high byte (MSB) of the 16-bit PCA Counter/Timer. Reads of this register will read the contents of a “snapshot” register, whose contents are updated only when the contents of PCA0L are read (see Section 34.1). |

**Note:** When the WDTE bit is set to 1, the PCA0H register cannot be modified by software. To change the contents of the PCA0H register, the watchdog timer must first be disabled.

Not Recommended for New Designs

## SFR Definition 34.7. PCA0CPLn: PCA Capture Module Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PCA0CPn[7:0] | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Addresses: PCA0CPL0 = 0xFB, PCA0CPL1 = 0xE9, PCA0CPL2 = 0xEB,
PCA0CPL3 = 0xED, PCA0CPL4 = 0xFD, PCA0CPL5 = 0xD2

SFR Pages: PCA0CPL0 = 0x0, PCA0CPL1 = 0x0, PCA0CPL2 = 0x0,
PCA0CPL3 = 0x0, PCA0CPL4 = 0x0, PCA0CPL5 = 0x0

| Bit | Name | Function |
|---|---|---|
| 7:0 | PCA0CPn[7:0] | **PCA Capture Module Low Byte.**<br>The PCA0CPLn register holds the low byte (LSB) of the 16-bit capture module n. This register address also allows access to the low byte of the corresponding PCA channel's auto-reload value for 9, 10, or 11-bit PWM mode. The ARSEL bit in register PCA0PWM controls which register is accessed. |
| **Note:** A write to this register will clear the module's ECOMn bit to a 0. | | |

## SFR Definition 34.8. PCA0CPHn: PCA Capture Module High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PCA0CPn[15:8] | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Addresses: PCA0CPH0 = 0xFC, PCA0CPH1 = 0xEA, PCA0CPH2 = 0xEC,
PCA0CPH3 = 0xEE, PCA0CPH4 = 0xFE, PCA0CPH5 = 0xD3

SFR Pages: PCA0CPH0 = 0x0, PCA0CPH1 = 0x0, PCA0CPH2 = 0x0,
PCA0CPH3 = 0x0, PCA0CPH4 = 0x0, PCA0CPH5 = 0x0

| Bit | Name | Function |
|---|---|---|
| 7:0 | PCA0CPn[15:8] | **PCA Capture Module High Byte.**<br>The PCA0CPHn register holds the high byte (MSB) of the 16-bit capture module n. This register address also allows access to the high byte of the corresponding PCA channel's auto-reload value for 9, 10, or 11-bit PWM mode. The ARSEL bit in register PCA0PWM controls which register is accessed. |
| **Note:** A write to this register will set the module's ECOMn bit to a 1. | | |

Not Recommended for New Designs

# 35. C2 Interface

Si102x/3x devices include an on-chip Silicon Labs 2-Wire (C2) debug interface to allow Flash programming and in-system debugging with the production part installed in the end application. The C2 interface uses a clock signal (C2CK) and a bi-directional C2 data signal (C2D) to transfer information between the device and a host system. See the C2 Interface Specification for details on the C2 protocol.

## 35.1. C2 Interface Registers

The following describes the C2 registers necessary to perform Flash programming through the C2 interface. All C2 registers are accessed through the C2 interface as described in the C2 Interface Specification.

### C2 Register Definition 35.1. C2ADD: C2 Address

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | C2ADD[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Name | Function | |
|---|---|---|---|
| 7:0 | C2ADD[7:0] | **C2 Address.**<br>The C2ADD register is accessed via the C2 interface to select the target Data register for C2 Data Read and Data Write commands. | |
| | | **Address** | **Description** |
| | | 0x00 | Selects the Device ID register for Data Read instructions. |
| | | 0x01 | Selects the Revision ID register for Data Read instructions. |
| | | 0x02 | Selects the C2 Flash Programming Control register for Data Read/Write instructions. |
| | | 0xB4 | Selects the C2 Flash Programming Data register for Data Read/Write instructions. |

Not Recommended for New Designs

## C2 Register Definition 35.2. DEVICEID: C2 Device ID

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | DEVICEID[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |

C2 Address: 0x00

| Bit | Name | Function |
|---|---|---|
| 7:0 | DEVICEID[7:0] | **Device ID.**<br>This read-only register returns the 8-bit device ID: 0x2A (Si102x/3x). |

## C2 Register Definition 35.3. REVID: C2 Revision ID

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | REVID[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | Varies | Varies | Varies | Varies | Varies | Varies | Varies | Varies |

C2 Address: 0x01

| Bit | Name | Function |
|---|---|---|
| 7:0 | REVID[7:0] | **Revision ID.**<br>This register indicates the MCU revision.<br>0x01: Rev A.<br>0x02: Rev B. |

Not Recommended for New Designs

## C2 Register Definition 35.4. FPCTL: C2 Flash Programming Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | FPCTL[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

C2 Address: 0x02

| Bit | Name | Function |
|---|---|---|
| 7:0 | FPCTL[7:0] | **Flash Programming Control Register.**<br>This register is used to enable Flash programming via the C2 interface. To enable C2 Flash programming, the following codes must be written in order: 0x02, 0x01. Note that once C2 Flash programming is enabled, a system reset must be issued to resume normal operation. |

## C2 Register Definition 35.5. FPDAT: C2 Flash Programming Data

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | FPDAT[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

C2 Address: 0xB4

| Bit | Name | Function | |
|---|---|---|---|
| 7:0 | FPDAT[7:0] | **C2 Flash Programming Data Register.**<br>This register is used to pass Flash commands, addresses, and data during C2 Flash accesses. Valid commands are listed below. | |
| | | **Code** | **Command** |
| | | 0x06 | Flash Block Read |
| | | 0x07 | Flash Block Write |
| | | 0x08 | Flash Page Erase |
| | | 0x03 | Device Erase |

Not Recommended for New Designs

## 35.2. C2 Pin Sharing

The C2 protocol allows the C2 pins to be shared with user functions so that in-system debugging and Flash programming may be performed. This is possible because C2 communication is typically performed when the device is in the halt state, where all on-chip peripherals and user software are stalled. In this halted state, the C2 interface can safely “borrow” the C2CK ($\overline{\text{RST}}$) and C2D pins. In most applications, external resistors are required to isolate C2 interface traffic from the user application. A typical isolation configuration is shown in Figure 35.1.

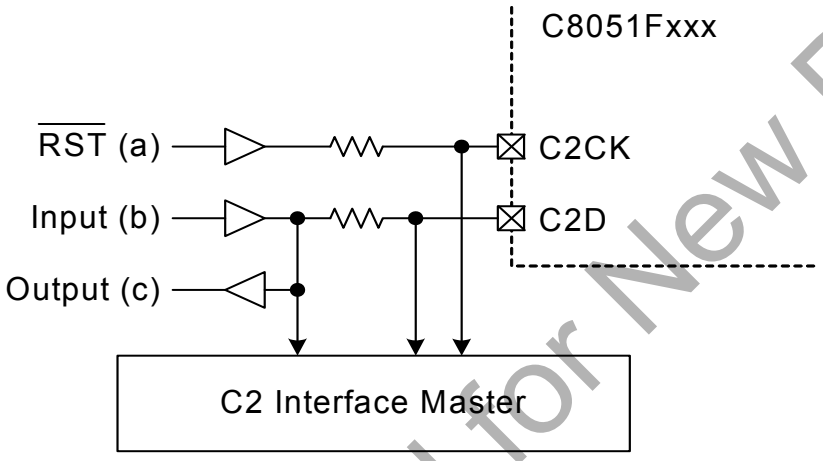



**Figure 35.1. Typical C2 Pin Sharing**

The configuration in Figure 35.1 assumes the following:

1. The user input (b) cannot change state while the target device is halted.
2. The $\overline{\text{RST}}$ pin on the target device is used as an input only.

Additional resistors may be necessary depending on the specific application.

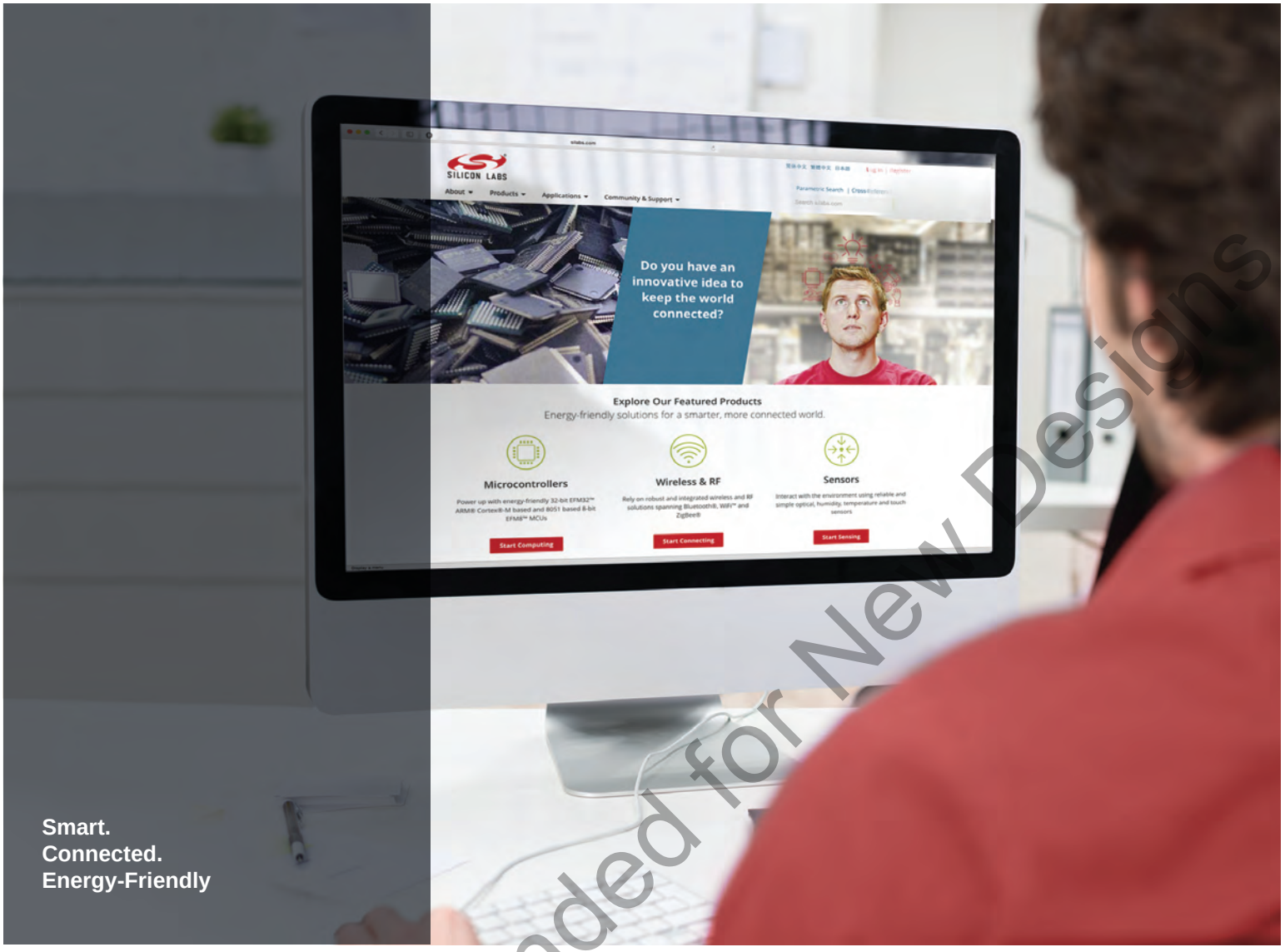
Not Recommended for New Designs

# Document Change List

## Revision 0.3 to Revision 1.0

- Updated part numbers to Revision B.
- Updated device package mechanical spec and PCB land pattern.
- Updated electrical specifications, including TBD values.
- Updated EZRadioPro Version Code.
- Removed support for EZRadioPro Low Battery Detect feature.
- Deleted SFR Page Stack Example in Special Function Registers chapter.
- Change description of SFRPGEN bit in SFRPGCN SFR definition.
- Added paragraph to Flash chapter to explain lock byte behavior on 128 kB devices.
- Corrected SFRPAGE in SPI1 SFR definitions.
- Fixed inconsistencies in VIORF pin definitions.
- Added note about IFBANK usage.
- Fixed inconsistencies in description of reset behavior.
- Added encryption/decryption times to SFR Definition.
- Fixed inconsistencies in various SFR Definitions.

Not Recommended for New Designs




Not Recommended for New Designs

**Products**
*www.silabs.com/products*

**Quality**
*www.silabs.com/quality*

**Support and Community**
*community.silabs.com*

**Disclaimer**
Silicon Laboratories intends to provide customers with the latest, accurate, and in-depth documentation of all peripherals and modules available for system and software implementers using or intending to use the Silicon Laboratories products. Characterization data, available modules and peripherals, memory sizes and memory addresses refer to each specific device, and "Typical" parameters provided can and do vary in different applications. Application examples described herein are for illustrative purposes only. Silicon Laboratories reserves the right to make changes without further notice and limitation to product information, specifications, and descriptions herein, and does not give warranties as to the accuracy or completeness of the included information. Silicon Laboratories shall have no liability for the consequences of use of the information supplied herein. This document does not imply or express copyright licenses granted hereunder to design or fabricate any integrated circuits. The products must not be used within any Life Support System without the specific written consent of Silicon Laboratories. A "Life Support System" is any product or system intended to support or sustain life and/or health, which, if it fails, can be reasonably expected to result in significant personal injury or death. Silicon Laboratories products are generally not intended for military applications. Silicon Laboratories products shall under no circumstances be used in weapons of mass destruction including (but not limited to) nuclear, biological or chemical weapons, or missiles capable of delivering such weapons.

**Trademark Information**
Silicon Laboratories Inc., Silicon Laboratories, Silicon Labs, SiLabs and the Silicon Labs logo, CMEMS®, EFM, EFM32, EFR, Energy Micro, Energy Micro logo and combinations thereof, "the world's most energy friendly microcontrollers", Ember®, EZLink®, EZMac®, EZRadio®, EZRadioPRO®, DSPLL®, ISOmodem ®, Precision32®, ProSLIC®, SiPHY®, USBXpress® and others are trademarks or registered trademarks of Silicon Laboratories Inc. ARM, CORTEX, Cortex-M3 and THUMB are trademarks or registered trademarks of ARM Holdings. Keil is a registered trademark of ARM Limited. All other products or brand names mentioned herein are trademarks of their respective holders.

SILICON LABS

**Silicon Laboratories Inc.**
**400 West Cesar Chavez**
**Austin, TX 78701**
**USA**

**http://www.silabs.com**