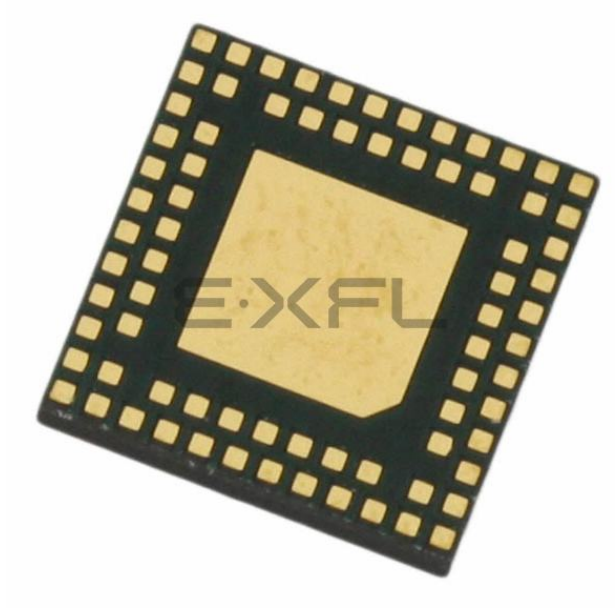
Silicon Labs - C8051F962-B-GMR Datasheet



Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	8051
Core Size	8-Bit
Speed	25MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	DMA, POR, PWM, WDT
Number of I/O	57
Program Memory Size	128KB (128K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	8.25K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.8V
Data Converters	A/D 16x10b/12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	76-VFQFN Dual Rows, Exposed Pad
Supplier Device Package	76-DQFN (6x6)
Purchase URL	https://www.e-xfl.com/product-detail/silicon-labs/c8051f962-b-gmr

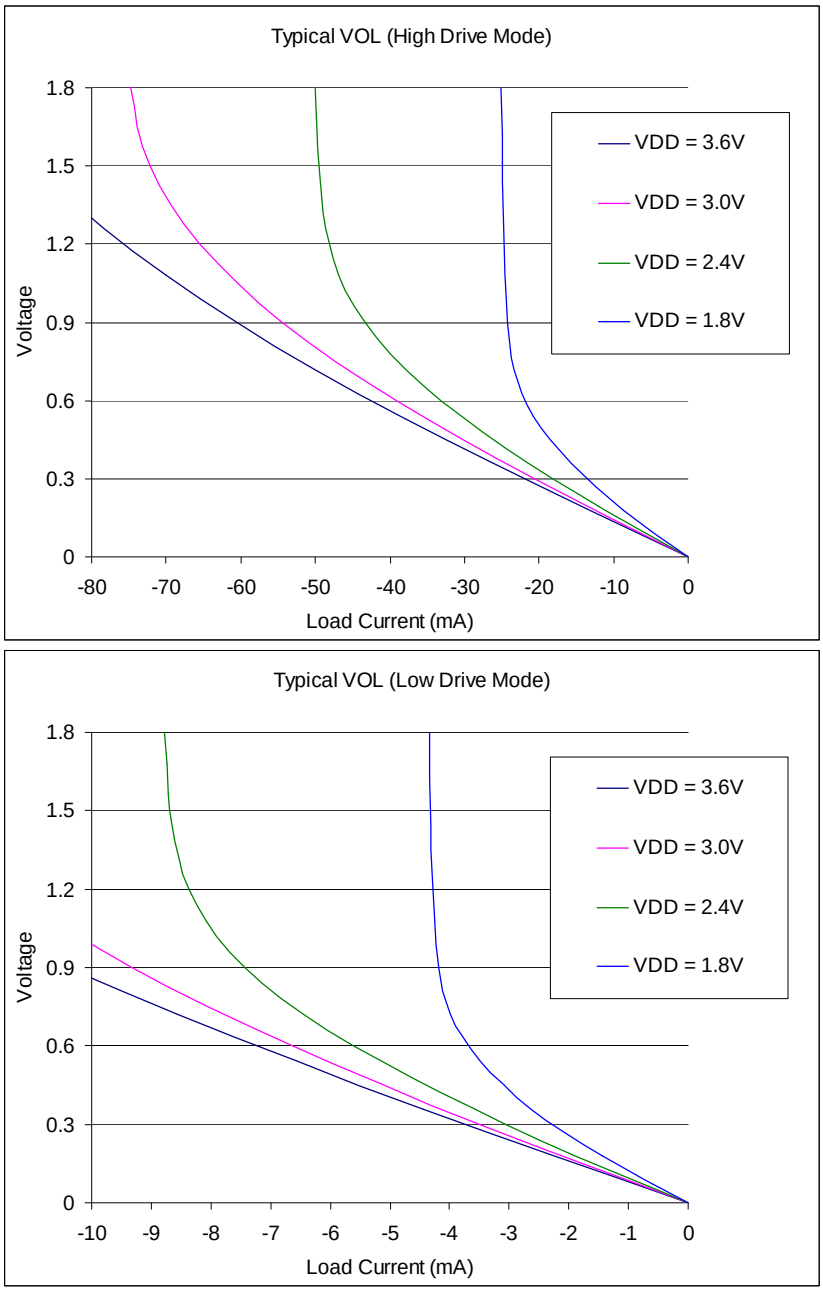


Figure 4.3. Typical VOL Curves, 1.8–3.6 V

Table 4.11. SmaRTClock Characteristics

V_{BAT} = 1.8 to 3.8 V; T_A = –40 to +85 °C unless otherwise specified; Using factory-calibrated settings.

Parameter	Conditions	Min	Typ	Max	Units
Oscillator Frequency (LFO)		13.1	16.4	19.7	kHz

Table 4.12. ADC0 Electrical Characteristics

V_{BAT} = 1.8 to 3.8 V, VREF = 1.65 V (REFSL[1:0] = 11), –40 to +85 °C unless otherwise specified.

Parameter	Conditions	Min	Typ	Max	Units
DC Accuracy					
Resolution	12-bit mode 10-bit mode		12 10		bits
Integral Nonlinearity	12-bit mode[1] 10-bit mode	— —	±1 ±0.5	±3 ±1	LSB
Differential Nonlinearity (Guaranteed Monotonic)	12-bit mode[1] 10-bit mode	— —	±0.8 ±0.5	±2 ±1	LSB
Offset Error	12-bit mode 10-bit mode	— —	±<1 ±<1	±3 ±3	LSB
Full Scale Error	12-bit mode[2] 10-bit mode	— —	±1 ±1	±4 ±2.5	LSB
Dynamic performance (10 kHz sine-wave single-ended input, 1 dB below Full Scale, maximum sampling rate)					
Signal-to-Noise Plus Distortion[3]	12-bit mode 10-bit mode	62 54	65 58	— —	dB
Signal-to-Distortion[3]	12-bit mode 10-bit mode	— —	76 73	— —	dB
Spurious-Free Dynamic Range[3]	12-bit mode 10-bit mode	— —	82 75	— —	dB
Conversion Rate					
SAR Conversion Clock	Normal Power Mode Low Power Mode	— —	— —	8.33 4.4	MHz
Conversion Time in SAR Clocks	10-bit Mode 8-bit Mode	13 11	— —	— —	clocks
Track/Hold Acquisition Time	Initial Acquisition Subsequent Acquisitions (dc input, burst mode)	1.5 1.1	— —	— —	us
Throughput Rate	12-bit mode 10-bit mode	— —	— —	75 300	ksps

1. INL and DNL specifications for 12-bit mode do not include the first or last four ADC codes.
2. The maximum code in 12-bit mode is 0xFFFC. The Full Scale Error is referenced from the maximum code.
3. Performance in 8-bit mode is similar to 10-bit mode.

5.2.4. Settling Time Requirements

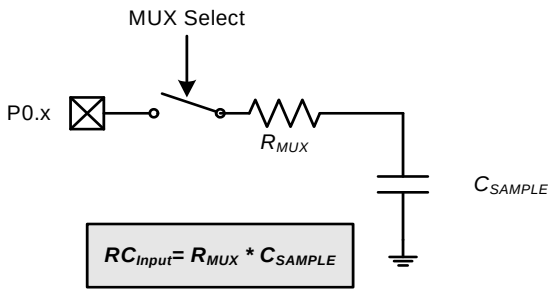
A minimum amount of tracking time is required before each conversion can be performed, to allow the sampling capacitor voltage to settle. This tracking time is determined by the AMUX0 resistance, the ADC0 sampling capacitance, any external source resistance, and the accuracy required for the conversion. Note that in low-power tracking mode, three SAR clocks are used for tracking at the start of every conversion. For many applications, these three SAR clocks will meet the minimum tracking time requirements, and higher values for the external source impedance will increase the required tracking time.

Figure 5.4 shows the equivalent ADC0 input circuit. The required ADC0 settling time for a given settling accuracy (SA) may be approximated by Equation . When measuring the Temperature Sensor output or V_{DD} with respect to GND, R_{TOTAL} reduces to R_{MUX}. See Table 4.12 for ADC0 minimum settling time requirements as well as the mux impedance and sampling capacitor values.

$$t = \ln\left(\frac{2^n}{SA}\right) \times R_{TOTAL} C_{SAMPLE}$$

ADC0 Settling Time Requirements

Where:
SA is the settling accuracy, given as a fraction of an LSB (for example, 0.25 to settle within 1/4 LSB)
t is the required settling time in seconds
R_{TOTAL} is the sum of the AMUX0 resistance and any external source resistance.
n is the ADC resolution in bits (10).



Note: The value of CSAMPLE depends on the PGA Gain. See Table 4.12 for details.

Figure 5.4. ADC0 Equivalent Input Circuits

5.2.5. Gain Setting

The ADC has gain settings of 1x and 0.5x. In 1x mode, the full scale reading of the ADC is determined directly by V_{REF}. In 0.5x mode, the full-scale reading of the ADC occurs when the input voltage is V_{REF} x 2. The 0.5x gain setting can be useful to obtain a higher input Voltage range when using a small V_{REF} voltage, or to measure input voltages that are between V_{REF} and V_{DD}. Gain settings for the ADC are controlled by the AMP0GN bit in register ADC0CF.

10. External Data Memory Interface and On-Chip XRAM

For C8051F96x devices, 8 kB of RAM are included on-chip and mapped into the external data memory space (XRAM). Additionally, an External Memory Interface (EMIF) is available on the C8051F960/2/4/6/8 devices, which can be used to access off-chip data memories and memory-mapped devices connected to the GPIO ports. The external memory space may be accessed using the external move instruction (MOVX) and the data pointer (DPTR), or using the MOVX indirect addressing mode using R0 or R1. If the MOVX instruction is used with an 8-bit address operand (such as @R1), then the high byte of the 16-bit address is provided by the External Memory Interface Control Register (EMI0CN, shown in SFR Definition 10.1).

Note: The MOVX instruction can also be used for writing to the flash memory. See Section "18. Flash Memory" on page 244 for details. The MOVX instruction accesses XRAM by default.

10.1. Accessing XRAM

The XRAM memory space is accessed using the MOVX instruction. The MOVX instruction has two forms, both of which use an indirect addressing method. The first method uses the Data Pointer, DPTR, a 16-bit register which contains the effective address of the XRAM location to be read from or written to. The second method uses R0 or R1 in combination with the EMI0CN register to generate the effective XRAM address. Examples of both of these methods are given below.

10.1.1. 16-Bit MOVX Example

The 16-bit form of the MOVX instruction accesses the memory location pointed to by the contents of the DPTR register. The following series of instructions reads the value of the byte at address 0x1234 into the accumulator A:

```
MOV    DPTR, #1234h        ; load DPTR with 16-bit address to read (0x1234)
MOVX   A, @DPTR            ; load contents of 0x1234 into accumulator A
```

The above example uses the 16-bit immediate MOV instruction to set the contents of DPTR. Alternately, the DPTR can be accessed through the SFR registers DPH, which contains the upper 8-bits of DPTR, and DPL, which contains the lower 8-bits of DPTR.

10.1.2. 8-Bit MOVX Example

The 8-bit form of the MOVX instruction uses the contents of the EMI0CN SFR to determine the upper 8-bits of the effective address to be accessed and the contents of R0 or R1 to determine the lower 8-bits of the effective address to be accessed. The following series of instructions read the contents of the byte at address 0x1234 into the accumulator A.

```
MOV    EMI0CN, #12h        ; load high byte of address into EMI0CN
MOV    R0, #34h            ; load low byte of address into R0 (or R1)
MOVX   a, @R0              ; load contents of 0x1234 into accumulator A
```

Table 12.2. Example 32-bit CRC Outputs

Input	Output
0x63	0xF9462090
0xAA, 0xBB, 0xCC	0x41B207B3
0x00, 0x00, 0xAA, 0xBB, 0xCC	0x78D129BC

12.3. Preparing for a CRC Calculation

To prepare CRC0 for a CRC calculation, software should select the desired polynomial and set the initial value of the result. Two polynomials are available: 0x1021 (16-bit) and 0x04C11DB7 (32-bit). The CRC0 result may be initialized to one of two values: 0x00000000 or 0xFFFFFFFF. The following steps can be used to initialize CRC0.

1. Select a polynomial (Set CRC0SEL to 0 for 32-bit or 1 for 16-bit).
2. Select the initial result value (Set CRC0VAL to 0 for 0x00000000 or 1 for 0xFFFFFFFF).
3. Set the result to its initial value (Write 1 to CRC0INIT).

12.4. Performing a CRC Calculation

Once CRC0 is initialized, the input data stream is sequentially written to CRC0IN, one byte at a time. The CRC0 result is automatically updated after each byte is written. The CRC engine may also be configured to automatically perform a CRC on one or more flash sectors. The following steps can be used to automatically perform a CRC on flash memory.

1. Prepare CRC0 for a CRC calculation as shown above.
2. If necessary, set the IFBANK bits in the PSBANK for the desired code bank.
3. Write the index of the starting page to CRC0AUTO.
4. Set the AUTOEN bit in CRC0AUTO.
5. Write the number of flash sectors to perform in the CRC calculation to CRC0CNT.
   Note: Each flash sector is 1024 bytes.
6. Write any value to CRC0CN (or OR its contents with 0x00) to initiate the CRC calculation. The CPU will not execute code any additional code until the CRC operation completes.
7. Clear the AUTOEN bit in CRC0AUTO.
8. Read the CRC result using the procedure below.

Setting the IFBANK bits in the PSBANK SFR is only necessary when accessing the upper banks on 128 kB code bank devices (‘F960/1/2/3). Multiple CRCs are required to cover the entire 128 kB Flash array. When writing to the PSBANK SFR, the code initiating the auto CRC of flash must be executing from the common area.

12.5. Accessing the CRC0 Result

The internal CRC0 result is 32-bits (CRC0SEL = 0b) or 16-bits (CRC0SEL = 1b). The CRC0PNT bits select the byte that is targeted by read and write operations on CRC0DAT and increment after each read or write. The calculation result will remain in the internal CR0 result register until it is set, overwritten, or additional data is written to CRC0IN.

14.1.3. Configuration sfrs

The AES Module has two configuration sfrs. The AES0BCFG sfr is used to configure the AES core. Bits 0 and 1 are used to select the Key size. The AES core supports 128-bit, 192-bit and 256-bit encryption. Bit 2 selects encrypt or decrypt. The AES enable bit (bit 3) is used to enable the AES module and start and new encryption operation. The AES DONE bit (bit 5) is the AES interrupt flag that signals a block of data has been completely encrypted or decrypted and is ready to be read from the AES0YOUT sfr. Note that the AES DONE interrupt is not normally used when the AES module is used with the DMA. Instead the DMA interrupt is used to signal that the encrypted or decrypted data has been transferred completely to memory. The DMA done interrupt is normally only used with direct sfr access.

The AES0DCFG sfr is used to select the data path for the AES module. Bits 0 through 2 are used to select the input and output multiplexer configuration. The AES data path should be configured prior to initiating a new encryption or decryption operation.

14.1.4. Input Multiplexer

The input multiplexer is used to select either the contents of the AES0BIN sfr or the contents of the AES0BIN sfr exclusive ORed with the contents of the AES0XIN sfr. The exclusive OR input data path provides support for CBC encryption.

14.1.5. Output Multiplexer

The output multiplexer selects the data source for the AES0YOUT sfr. The three possible sources are the AES Core data output, the AES Core Key output, and the AES core data output exclusive ORed with the AES0XIN sfr.

The AES core data output is used for simple encryption and decryption.

The exclusive OR output data path provides support for CBC mode decryption and CTR mode encryption/decryption. The AES0XIN is the source for both input and output exclusive OR data. When the AES0XIN is used with the input exclusive OR data path, the AEXIN data is written in sequence with the AES0BIN data. When used with the output XRO data path, the AES0XIN data is written after the encryption or decryption operation is complete.

The Key output is used to generate an inverse key. To generate a decryption key from an encryption key, the AES core should be configured for an encryption operation. To generate an encryption key from a decryption key, the AES core should be configured for a decryption operation.

14.1.6. Internal State Machine

The AES Module has an internal state machine that manages the data flow. The internal state machine accommodates the two different usage scenarios. When using the DMA, the internal state machine will send peripheral requests to the DMA requesting the DMA to transfer data from xram to the AES module input sfrs. Upon the completion of one block of data, the AES module will send peripheral requests requesting data to be transferred from the AES0YOUT sfr to xram. These peripheral requests are managed by the internal state machine.

When not using the DMA, data must be written and read in a specific order. The DMA state machine will advance with each byte written or read.

The internal state machine may be reset by clearing the enable bit in the AESBGFG sfr. Clearing the enable bit before encryption or decryption operation will ensure that the state machine starts at the proper starting state.

When encrypting or decrypting multiple blocks it is not necessary to disable the AES module between blocks, as long as the proper sequence of events is obeyed.

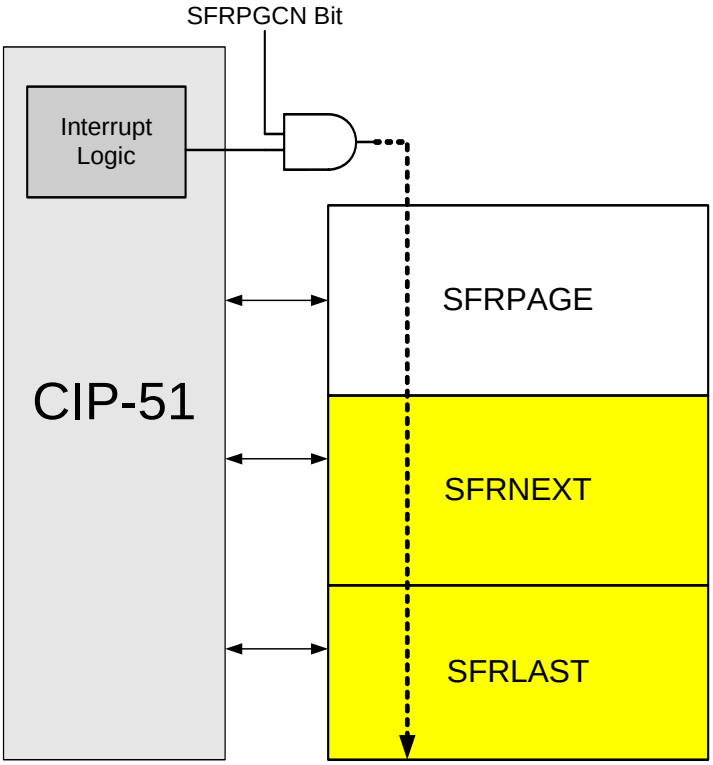


Figure 16.1. SFR Page Stack

Automatic hardware preserving and restoring of the SFR Page on interrupts may be enabled or disabled as desired using the SFR Automatic Page Control Enable Bit located in the SFR Page Control Register (SFR0CN). This function defaults to “enabled” upon reset. In this way, the autoswitching function will be enabled unless disabled in software.

A summary of the SFR locations (address and SFR page) are provided in Table 16.3 in the form of an SFR memory map. Each memory location in the map has an SFR page row, denoting the page in which that SFR resides. Certain SFRs are accessible from ALL SFR pages, and are denoted by the “(ALL PAGES)” designation. For example, the Port I/O registers P0, P1, P2, and P3 all have the “(ALL PAGES)” designation, indicating these SFRs are accessible from all SFR pages regardless of the SFRPAGE register value.

SFR Definition 17.3. EIE1: Extended Interrupt Enable 1

Bit	7	6	5	4	3	2	1	0
Name	ET3	ECP1	ECP0	EPCA0	EADC0	EWADC0	ERTC0A	ESMB0
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 = 0xE6

Bit	Name	Function
7	ET3	**Enable Timer 3 Interrupt.** This bit sets the masking of the Timer 3 interrupt. 0: Disable Timer 3 interrupts. 1: Enable interrupt requests generated by the TF3L or TF3H flags.
6	ECP1	**Enable Comparator1 (CP1) Interrupt.** This bit sets the masking of the CP1 interrupt. 0: Disable CP1 interrupts. 1: Enable interrupt requests generated by the CP1RIF or CP1FIF flags.
5	ECP0	**Enable Comparator0 (CP0) Interrupt.** This bit sets the masking of the CP0 interrupt. 0: Disable CP0 interrupts. 1: Enable interrupt requests generated by the CP0RIF or CP0FIF flags.
4	EPCA0	**Enable Programmable Counter Array (PCA0) Interrupt.** This bit sets the masking of the PCA0 interrupts. 0: Disable all PCA0 interrupts. 1: Enable interrupt requests generated by PCA0.
3	EADC0	**Enable ADC0 Conversion Complete Interrupt.** This bit sets the masking of the ADC0 Conversion Complete interrupt. 0: Disable ADC0 Conversion Complete interrupt. 1: Enable interrupt requests generated by the AD0INT flag.
2	EWADC0	**Enable Window Comparison ADC0 Interrupt.** This bit sets the masking of ADC0 Window Comparison interrupt. 0: Disable ADC0 Window Comparison interrupt. 1: Enable interrupt requests generated by ADC0 Window Compare flag (AD0WINT).
1	ERTC0A	**Enable SmaRTClock Alarm Interrupts.** This bit sets the masking of the SmaRTClock Alarm interrupt. 0: Disable SmaRTClock Alarm interrupts. 1: Enable interrupt requests generated by a SmaRTClock Alarm.
0	ESMB0	**Enable SMBus (SMB0) Interrupt.** This bit sets the masking of the SMB0 interrupt. 0: Disable all SMB0 interrupts. 1: Enable interrupt requests generated by SMB0.

Important Notes:

- The Power-on Reset (POR) delay is not incurred after a supply monitor reset. See Section “4. Electrical Characteristics” on page 56 for complete electrical characteristics of the active mode supply monitors.
- Software should take care not to inadvertently disable the supply monitor as a reset source when writing to RSTSRC to enable other reset sources or to trigger a software reset. All writes to RSTSRC should explicitly set PORSF to 1 to keep the supply monitor enabled as a reset source.
- The supply monitor must be enabled before selecting it as a reset source**.** Selecting the supply monitor as a reset source before it has stabilized may generate a system reset. In systems where this reset would be undesirable, a delay should be introduced between enabling the supply monitor and selecting it as a reset source. See Section “4. Electrical Characteristics” on page 56 for minimum supply monitor turn-on time. **No delay should be introduced in systems where software contains routines that erase or write Flash memory.** The procedure for enabling the V_{DD} supply monitor and selecting it as a reset source is shown below:
  1. Enable the Supply Monitor (VDMEN bit in VDM0CN = 1).
  2. Wait for the Supply Monitor to stabilize (optional).
  3. Select the Supply Monitor as a reset source (PORSF bit in RSTSRC = 1).

24.2.2. Using the SmaRTClock Oscillator in Self-Oscillate Mode

When using Self-Oscillate Mode, the XTAL3 and XTAL4 pins are internally shorted together. The following steps show how to configure SmaRTClock for use in Self-Oscillate Mode:

1. Configure the XTAL3 and XTAL4 pins for analog I/O and disable the digital driver.
2. Set SmaRTClock to Self-Oscillate Mode (XMODE = 0).
3. Set the desired oscillation frequency:
   For oscillation at about 20 kHz, set BIASX2 = 0.
   For oscillation at about 40 kHz, set BIASX2 = 1.
4. The oscillator starts oscillating instantaneously.
5. Fine tune the oscillation frequency by adjusting the load capacitance (RTC0XCF).

24.2.3. Using the Low Frequency Oscillator (LFO)

The low frequency oscillator provides an ultra low power, on-chip clock source to the SmaRTClock. The typical frequency of oscillation is 16.4 kHz ±20%. No external components are required to use the LFO and the XTAL3 and XTAL4 pins may be used for general purpose I/O without any effect on the LFO.

The following steps show how to configure SmaRTClock for use with the LFO:

1. Enable and select the Low Frequency Oscillator (LFOEN = 1).
2. The LFO starts oscillating instantaneously.

When the LFO is enabled, the SmaRTClock oscillator increments bit 1 of the 32-bit timer (instead of bit 0). This effectively multiplies the LFO frequency by 2, making the RTC timebase behave as if a 32.768 kHz crystal is connected at the output.

24.2.4. Programmable Load Capacitance

The programmable load capacitance has 16 values to support crystal oscillators with a wide range of recommended load capacitance. If Automatic Load Capacitance Stepping is enabled, the crystal load capacitors start at the smallest setting to allow a fast startup time, then slowly increase the capacitance until the final programmed value is reached. The final programmed loading capacitor value is specified using the LOADCAP bits in the RTC0XCF register. The LOADCAP setting specifies the amount of on-chip load capacitance and does not include any stray PCB capacitance. Once the final programmed loading capacitor value is reached, the LOADRDY flag will be set by hardware to logic 1.

When using the SmaRTClock oscillator in Self-Oscillate mode, the programmable load capacitance can be used to fine tune the oscillation frequency. In most cases, increasing the load capacitor value will result in a decrease in oscillation frequency.Table 24.2 shows the crystal load capacitance for various settings of LOADCAP.

24.2.6. Missing SmaRTClock Detector

The missing SmaRTClock detector is a one-shot circuit enabled by setting MCLKEN (RTC0CN.6) to 1. When the SmaRTClock Missing Clock Detector is enabled, OSCFAIL (RTC0CN.5) is set by hardware if SmaRTClock oscillator remains high or low for more than 100 µs.

A SmaRTClock Missing Clock detector timeout can trigger an interrupt, wake the device from a low power mode, or reset the device. See Section “17. Interrupt Handler” on page 232, Section “19. Power Management” on page 257, and Section “22. Reset Sources” on page 278 for more information.

Note: The SmaRTClock Missing Clock Detector should be disabled when making changes to the oscillator settings in RTC0XCN.

24.2.7. SmaRTClock Oscillator Crystal Valid Detector

The SmaRTClock oscillator crystal valid detector is an oscillation amplitude detector circuit used during crystal startup to determine when oscillation has started and is nearly stable. The output of this detector can be read from the CLKVLD bit (RTX0XCN.4).

Notes:

1. The CLKVLD bit has a blanking interval of 2 ms. During the first 2 ms after turning on the crystal oscillator, the output of CLKVLD is not valid.
2. This SmaRTClock crystal valid detector (CLKVLD) is not intended for detecting an oscillator failure. The missing SmaRTClock detector (CLKFAIL) should be used for this purpose.
3. The CLKVLD bit output is driven low when BIASX2 is disabled.

24.3. SmaRTClock Timer and Alarm Function

The SmaRTClock timer is a 32-bit counter that, when running (RTC0TR = 1), is incremented every SmaRTClock oscillator cycle. The timer has an alarm function that can be set to generate an interrupt, wake the device from a low power mode, or reset the device at a specific time. See Section “17. Interrupt Handler” on page 232, Section “19. Power Management” on page 257, and Section “22. Reset Sources” on page 278 for more information.

The SmaRTClock timer includes an Auto Reset feature, which automatically resets the timer to zero one SmaRTClock cycle after the alarm 0 signal is deasserted. When using Auto Reset, the Alarm match value should always be set to 2 counts less than the desired match value. When using the LFO in combination with Auto Reset, the right-justified Alarm match value should be set to 4 counts less than the desired match value. Auto Reset can be enabled by writing a 1 to ALRM (RTC0CN.2).

24.3.1. Setting and Reading the SmaRTClock Timer Value

The 32-bit SmaRTClock timer can be set or read using the six CAPTUREn internal registers. Note that the timer does not need to be stopped before reading or setting its value. The following steps can be used to set the timer value:

1. Write the desired 32-bit set value to the CAPTUREn registers.
2. Write 1 to RTC0SET. This will transfer the contents of the CAPTUREn registers to the SmaRTClock timer.
3. Operation is complete when RTC0SET is cleared to 0 by hardware.

The following steps can be used to read the current timer value:

1. Write 1 to RTC0CAP. This will transfer the contents of the timer to the CAPTUREn registers.
2. Poll RTC0CAP until it is cleared to 0 by hardware.
3. A snapshot of the timer value can be read from the CAPTUREn registers

Notes:

1. If the system clock is faster than 4x the SmaRTClock, then the HSMODE bit should be set to allow the set and capture operations to be concluded quickly (system clock used for transfers).
2. If the system clock is slower than 4x the SmaRTClock, then HSMODE should be set to zero, and RTC must be

Internal Register Definition 24.5. RTC0XCN: SmaRTClock Oscillator Control

Bit	7	6	5	4	3	2	1	0
Name	AGCEN	XMODE	BIASX2	CLKVLD	LFOEN			
Type	R/W	R/W	R/W	R	R/W	R	R	R
Reset	0	0	0	0	0	0	0	0

SmaRTClock Address = 0x05

Bit	Name	Function
7	AGCEN	**SmaRTClock Oscillator Automatic Gain Control (AGC) Enable.** 0: AGC disabled. 1: AGC enabled.
6	XMODE	**SmaRTClock Oscillator Mode.** Selects Crystal or Self Oscillate Mode. 0: Self-Oscillate Mode selected. 1: Crystal Mode selected.
5	BIASX2	**SmaRTClock Oscillator Bias Double Enable.** **Enables/disables the Bias Double feature.** 0: Bias Double disabled. 1: Bias Double enabled.
4	CLKVLD	**SmaRTClock Oscillator Crystal Valid Indicator.** Indicates if oscillation amplitude is sufficient for maintaining oscillation. This bit always reads 0 when BIASX2 is disabled. 0: Oscillation has not started or oscillation amplitude is too low to maintain oscillation. 1: Sufficient oscillation amplitude detected.
3	LFOEN	**Low Frequency Oscillator Enable and Select.** Overrides XMODE and selects the internal low frequency oscillator (LFO) as the SmaRTClock oscillator source. 0: XMODE determines SmaRTClock oscillator source. 1: LFO enabled and selected as SmaRTClock oscillator source.
2:0	Unused	Read = 000b; Write = Don't Care.

Internal Register Definition 24.6. RTC0XCF: SmaRTClock Oscillator Configuration

Bit	7	6	5	4	3	2	1	0
Name	AUTOSTP	LOADRDY			LOADCAP			
Type	R/W	R	R	R	R/W			
Reset	0	0	0	0	Varies	Varies	Varies	Varies

SmaRTClock Address = 0x06

Bit	Name	Function
7	AUTOSTP	**Automatic Load Capacitance Stepping Enable.** Enables/disables automatic load capacitance stepping. 0: Load capacitance stepping disabled. 1: Load capacitance stepping enabled.
6	LOADRDY	**Load Capacitance Ready Indicator.** Set by hardware when the load capacitance matches the programmed value. 0: Load capacitance is currently stepping. 1: Load capacitance has reached it programmed value.
5:4	Unused	Read = 00b; Write = Don't Care.
3:0	LOADCAP	**Load Capacitance Programmed Value.** Holds the user's desired value of the load capacitance. See Table 24.2 on page 302.

Internal Register Definition 24.8. CAPTUREn: SmaRTClock Timer Capture

Bit	7	6	5	4	3	2	1	0
Name	CAPTURE[31:0]							
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

SmaRTClock Addresses: CAPTURE0 = 0x00; CAPTURE1 = 0x01; CAPTURE2 =0x02; CAPTURE3: 0x03.

Bit	Name	Function
7:0	CAPTURE[31:0]	**SmaRTClock Timer Capture.** These 4 registers (CAPTURE3–CAPTURE0) are used to read or set the 32-bit SmaRTClock timer. Data is transferred to or from the SmaRTClock timer when the RTC0SET or RTC0CAP bits are set.
Note: The least significant bit of the timer capture value is CAPTURE0.0.		

Internal Register Definition 24.9. ALARM0Bn: SmaRTClock Alarm 0 Match Value

Bit	7	6	5	4	3	2	1	0
Name	ALARM0[31:0]							
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

SmaRTClock Address: ALARM0B0 = 0x08; ALARM0B1 = 0x09; ALARM0B2 = 0x0A; ALARM0B3 = 0x0B

Bit	Name	Function
7:0	ALARM0[31:0]	**SmaRTClock Alarm 0 Programmed Value.** These 4 registers (ALARM0B3–ALARM0B0) are used to set an alarm event for the SmaRTClock timer. The SmaRTClock alarm should be disabled (ALRM0EN=0) when updating these registers.
Note: The least significant bit of the alarm programmed value is ALARM0B0.0.		

26.3. LCD Contrast Adjustment

The LCD Bias voltages which determine the LCD contrast are generated using the VBAT supply voltage or the on-chip charge pump. There are four contrast control modes to accomodate a wide variety of applications and supply voltages. The target contrast voltage is programmable in 60 mV steps from 1.9 to 3.72 V. The LCD contrast voltage is controlled by the LCD0CNTRST register and the contrast control mode is selected by setting the appropriate bits in the LCD0MSCN, LCD0MSCF, LCD0PWR, and LCD0VBMCN registers.

Note: An external 10 µF decoupling capacitor is required on the VLCD pin to create a charge reservoir at the output of the charge pump.

Table 26.1. Bit Configurations to select Contrast Control Modes

Mode	LCD0MSCN.2	LCD0MSCF.0	LCD0PWR.3	LCD0VBMCN.7
1	0	1	0	0
2	0	1	1	1
3	1*	0	1	1
4	1*	0	0	1
*** May be set to 0 to support increased load currents.**				

26.3.1. Contrast Control Mode 1 (Bypass Mode)

In Contrast Control Mode 1, the contrast control circuitry is disabled and the VLCD voltage follows the VBAT supply voltage, as shown in Figure 26.3. This mode is useful in systems where the VBAT voltage always remains constant and will provide the lowest LCD power consumption. Bypass Mode is selected using the following procedure:

1. Clear Bit 2 of the LCD0MSCN register to 0b (LCD0MSCN &= ~0x04)
2. Set Bit 0 of the LCD0MSCF register to 1b (LCD0MSCF |= 0x01)
3. Clear Bit 3 of the LCD0PWR register to 0b (LCD0PWR &= ~0x08)
4. Clear Bit 7 of the LCD0VBMCN register to 0b (LCD0VBMCN &= ~0x80)

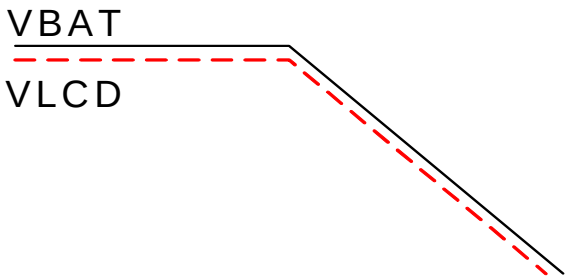


Figure 26.3. Contrast Control Mode 1

Table 28.1. SMBus Clock Source Selection

SMBCS1	SMBCS0	SMBus Clock Source
0	0	Timer 0 Overflow
0	1	Timer 1 Overflow
1	0	Timer 2 High Byte Overflow
1	1	Timer 2 Low Byte Overflow

The SMBCS1–0 bits select the SMBus clock source, which is used only when operating as a master or when the Free Timeout detection is enabled. When operating as a master, overflows from the selected source determine the absolute minimum SCL low and high times as defined in Equation 28.1. The selected clock source may be shared by other peripherals so long as the timer is left running at all times. For example, Timer 1 overflows may generate the SMBus and UART baud rates simultaneously. Timer configuration is covered in Section “32. Timers” on page 444.

$$T_{HighMin} = T_{LowMin} = \frac{1}{f_{ClockSourceOverflow}}$$

Equation 28.1. Minimum SCL High and Low Times

The selected clock source should be configured to establish the minimum SCL High and Low times as per Equation 28.1. When the interface is operating as a master (and SCL is not driven or extended by any other devices on the bus), the typical SMBus bit rate is approximated by Equation 28.1.

$$BitRate = \frac{f_{ClockSourceOverflow}}{3}$$

Equation 28.2. Typical SMBus Bit Rate

Figure 28.4 shows the typical SCL generation described by Equation 28.2. Notice that T_{HIGH} is typically twice as large as T_{LOW}. The actual SCL output may vary due to other devices on the bus (SCL may be extended low by slower slave devices, or driven low by contending master devices). The bit rate when operating as a master will never exceed the limits defined by Equation 28.2.

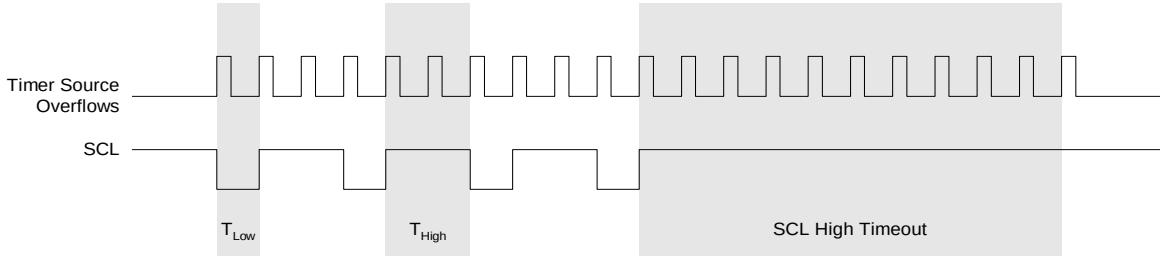


Figure 28.4. Typical SMBus SCL Generation

Setting the EXTHOLD bit extends the minimum setup and hold times for the SDA line. The minimum SDA setup time defines the absolute minimum time that SDA is stable before SCL transitions from low-to-high. The minimum SDA hold time defines the absolute minimum time that the current SDA value remains stable after SCL transitions from high-to-low. EXTHOLD should be set so that the minimum setup and hold times meet the SMBus Specification requirements of 250 ns and 300 ns, respectively. Table 28.2 shows the minimum setup and hold times for the two EXTHOLD settings. Setup and hold time extensions are typically necessary when SYSCLK is above 10 MHz.

Table 28.5. SMBus Status Decoding With Hardware ACK Generation Disabled (EHACK = 0)

Mode	Values Read				Current SMbus State	Typical Response Options	Values to Write			Next Status Vector Expected
	Status Vector	ACKRQ	ARBLOST	ACK			STA	STO	ACK	
Slave Transmitter	0100	0	0	0	A slave byte was transmitted; NACK received.	No action required (expecting STOP condition).	0	0	X	0001
		0	0	1	A slave byte was transmitted; ACK received.	Load SMB0DAT with next data byte to transmit.	0	0	X	0100
		0	1	X	A Slave byte was transmitted; error detected.	No action required (expecting Master to end transfer).	0	0	X	0001
	0101	0	X	X	An illegal STOP or bus error was detected while a Slave Transmission was in progress.	Clear STO.	0	0	X	-
Slave Receiver	0010	1	0	X	A slave address + R/W was received; ACK requested.	If Write, Acknowledge received address	0	0	1	0000
						If Read, Load SMB0DAT with data byte; ACK received address	0	0	1	0100
						NACK received address.	0	0	0	-
		1	1	X	Lost arbitration as master; slave address + R/W received; ACK requested.	If Write, Acknowledge received address	0	0	1	0000
						If Read, Load SMB0DAT with data byte; ACK received address	0	0	1	0100
						NACK received address.	0	0	0	-
						Reschedule failed transfer; NACK received address.	1	0	0	1110
	0001	0	0	X	A STOP was detected while addressed as a Slave Transmitter or Slave Receiver.	Clear STO.	0	0	X	-
		1	1	X	Lost arbitration while attempting a STOP.	No action required (transfer complete/aborted).	0	0	0	-
	0000	1	0	X	A slave byte was received; ACK requested.	Acknowledge received byte; Read SMB0DAT.	0	0	1	0000
						NACK received byte.	0	0	0	-
Bus Error Condition	0010	0	1	X	Lost arbitration while attempting a repeated START.	Abort failed transfer.	0	0	X	-
						Reschedule failed transfer.	1	0	X	1110
	0001	0	1	X	Lost arbitration due to a detected STOP.	Abort failed transfer.	0	0	X	-
						Reschedule failed transfer.	1	0	X	1110
	0000	1	1	X	Lost arbitration while transmitting a data byte as master.	Abort failed transfer.	0	0	0	-
						Reschedule failed transfer.	1	0	0	1110

32. 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 32.1 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.

33.1. 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 33.1.

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 33.1. 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.

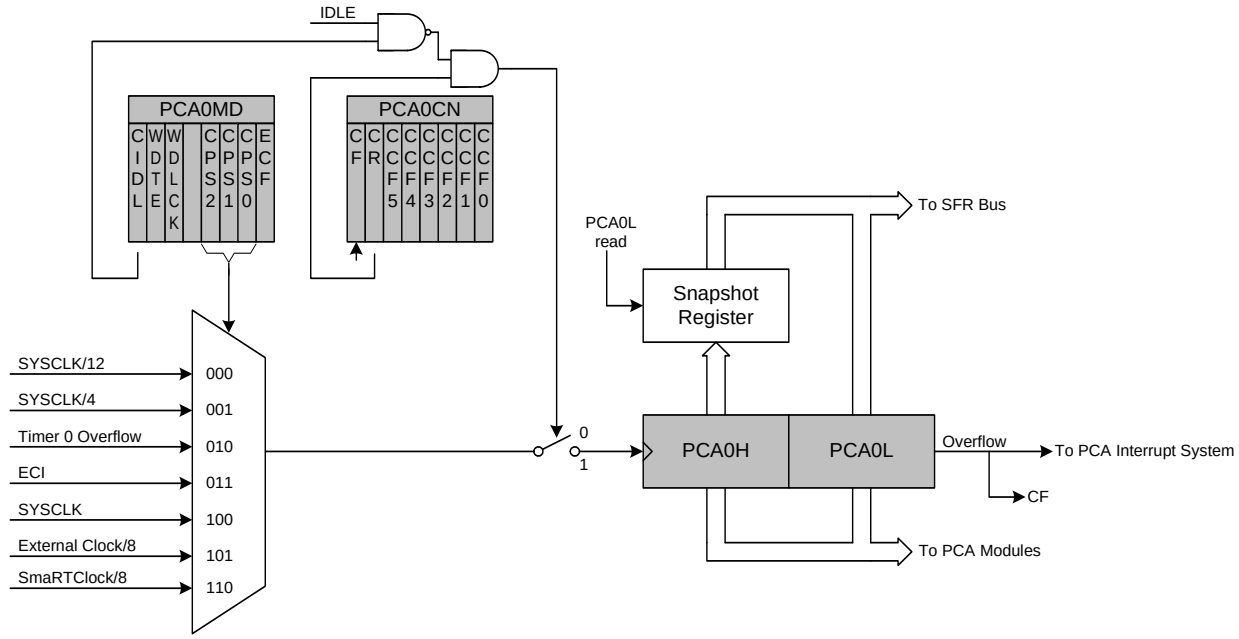


Figure 33.2. PCA Counter/Timer Block Diagram

33.2. PCA0 Interrupt Sources

Figure 33.3 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.