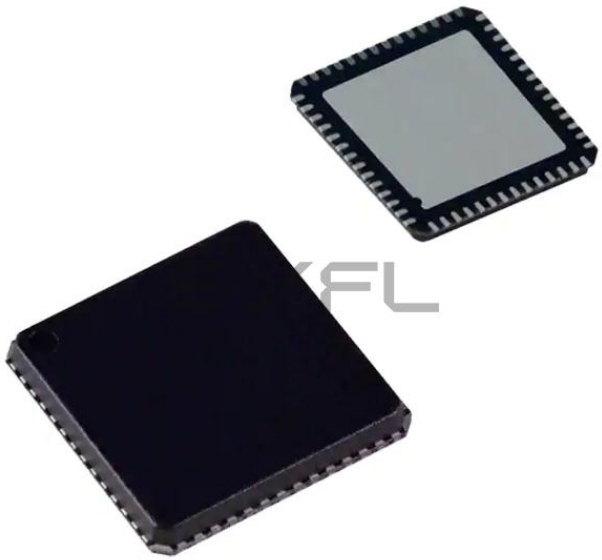


Welcome to E-XFL.COM

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	8052
Core Size	8-Bit
Speed	16MHz
Connectivity	EBI/EMI, I²C, SPI, UART/USART
Peripherals	PSM, Temp Sensor, WDT
Number of I/O	34
Program Memory Size	62KB (62K x 8)
Program Memory Type	FLASH
EEPROM Size	4K x 8
RAM Size	2.25K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 8x12b; D/A 2x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	56-VFQFN Exposed Pad, CSP
Supplier Device Package	56-LFCSP-VQ (8x8)
Purchase URL	https://www.e-xfl.com/product-detail/analog-devices/aduc831bcpz

Data Pointer (DPTR)
The Data Pointer is made up of three 8-bit registers, named DPP (page byte), DPH (high byte) and DPL (low byte). These are used to provide memory addresses for internal and external code access and external data access. It may be manipulated as a 16-bit register (DPTR = DPH, DPL), although INC DPTR instructions will automatically carry over to DPP, or as three independent 8-bit registers (DPP, DPH, DPL).

The ADuC831 supports dual data pointers. Refer to the Dual Data Pointer section.

Program Status Word (PSW)
The PSW SFR contains several bits reflecting the current status of the CPU as detailed in Table I.

SFR Address	D0H
Power-On Default Value	00H
Bit Addressable	Yes

Table I. PSW SFR Bit Designations

Bit	Name	Description		
7	CY	Carry Flag		
6	AC	Auxiliary Carry Flag		
5	F0	General-Purpose Flag		
4	RS1	Register Bank Select Bits		
3	RS0	RS1	RS0	Selected Bank
		0	0	0
		0	1	1
		1	0	2
		1	1	3
2	OV	Overflow Flag		
1	F1	General-Purpose Flag		
0	P	Parity Bit		

Power Control SFR (PCON)
The PCON SFR contains bits for power-saving options and general-purpose status flags as shown in Table II.

SFR Address	87H
Power-On Default Value	00H
Bit Addressable	No

Table II. PCON SFR Bit Designations

Bit	Name	Description
7	SMOD	Double UART Baud Rate
6	SERIPD	I^2C/SPI Power-Down Interrupt Enable
5	INT0PD	$\overline{INT0}$ Power-Down Interrupt Enable
4	ALEOFF	Disable ALE Output
3	GF1	General-Purpose Flag Bit
2	GF0	General-Purpose Flag Bit
1	PD	Power-Down Mode Enable
0	IDL	Idle Mode Enable

SPECIAL FUNCTION REGISTERS

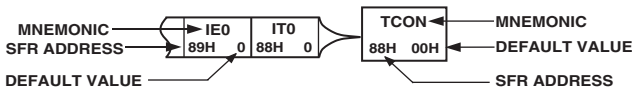
All registers except the program counter and the four general-purpose register banks, reside in the special function register (SFR) area. The SFR registers include control, configuration, and data registers that provide an interface between the CPU and other on-chip peripherals.

Figure 6 shows a full SFR memory map and SFR contents on Reset. Unoccupied SFR locations are shown dark-shaded in the figure below (NOT USED). Unoccupied locations in the SFR address space are not implemented, i.e., no register exists at this location. If an unoccupied location is read, an unspecified value is returned. SFR locations reserved for on-chip testing are shown lighter shaded below (RESERVED) and should not be accessed by user software. Sixteen of the SFR locations are also bit addressable and denoted by '1' in the figure below, i.e., the bit addressable SFRs are those whose address ends in 0H or 8H.

ISPI FFH 0	WCOL FEH 0	SPE FDH 0	SPIM FCH 0	CPOL FBH 0	CPHA FAH 1	SPR1 F9H 0	SPR0 F8H 0	BITS
F7H 0	F6H 0	F5H 0	F4H 0	F3H 0	F2H 0	F1H 0	F0H 0	BITS
MDO EFH 0	MDE EEH 0	MCO EDH 0	MDI ECH 0	I2CM EBH 0	I2CRS EAH 0	I2CTX E9H 0	I2CI E8H 0	BITS
E7H 0	E6H 0	E5H 0	E4H 0	E3H 0	E2H 0	E1H 0	E0H 0	BITS
ADCI DFH 0	DMA DEH 0	CCONV DDH 0	SCONV DCH 0	CS3 DBH 0	CS2 DAH 0	CS1 D9H 0	CS0 D8H 0	BITS
CY D7H 0	AC D6H 0	F0 D5H 0	RS1 D4H 0	RS0 D3H 0	OV D2H 0	FI D1H 0	P D0H 0	BITS
TF2 CFH 0	EXF2 CEH 0	RCLK CDH 0	TCLK CCH 0	EXEN2 CBH 0	TR2 CAH 0	CNT2 C9H 0	CAP2 C8H 0	BITS
PRE3 C7H 0	PRE2 C6H 0	PRE1 C5H 0	PRE0 C4H 1	WDIR C3H 0	WDS C2H 0	WDE C1H 0	WDWR C0H 0	BITS
PSI BFH 0	PADC BEH 0	PT2 BDH 0	PS BCH 0	PT1 BBH 0	PX1 BAH 0	PT0 B9H 0	PX0 B8H 0	BITS
RD B7H 1	WR B6H 1	T1 B5H 1	T0 B4H 1	INT1 B3H 1	INT0 B2H 1	TxD B1H 1	RxD B0H 1	BITS
EA AFH 0	EADC AEH 0	ET2 ADH 0	ES ACH 0	ET1 ABH 0	EX1 AAH 0	ET0 A9H 0	EX0 A8H 0	BITS
A7H 1	A6H 1	A5H 1	A4H 1	A3H 1	A2H 1	A1H 1	A0H 1	BITS
SM0 9FH 0	SM1 9EH 0	SM2 9DH 0	REN 9CH 0	TB8 9BH 0	RB8 9AH 0	TI 99H 0	RI 98H 0	BITS
97H 1	96H 1	95H 1	94H 1	93H 1	92H 1	T2EX 91H 1	T2 90H 1	BITS
TF1 8FH 0	TR1 8EH 0	TF0 8DH 0	TR0 8CH 0	IE1 8BH 0	IT1 8AH 0	IE0 89H 0	IT0 88H 0	BITS
87H 1	86H 1	85H 1	84H 1	83H 1	82H 1	81H 1	80H 1	BITS

SPICON[1] F8H 04H	DAC0L F9H 00H	DAC0H FAH 00H	DAC1L FBH 00H	DAC1H FCH 00H	DACCON FDH 04H	RESERVED	RESERVED
B[1] F0H 00H	ADCOFSL[3] F1H 00H	ADCOFSH[3] F2H 20H	ADCGAINL[3] F3H 00H	ADCGAINH[3] F4H 00H	ADCCON3 F5H 00H	RESERVED	SPIDAT F7H 00H
I2CCON[1] E8H 00H	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	ADCCON1 EFH 00H
ACC[1] E0H 00H	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
ADCCON2[1] D8H 00H	ADCDATAL D9H 00H	ADCDATAH DAH 00H	RESERVED	RESERVED	RESERVED	RESERVED	PSMCON DFH DEH
PSW[1] D0H 00H	RESERVED	DMAL D2H 00H	DMAH D3H 00H	DMAP D4H 00H	RESERVED	RESERVED	RESERVED
T2CON[1] C8H 00H	RESERVED	RCAP2L CAH 00H	RCAP2H CBH 00H	TL2 CCH 00H	TH2 CDH 00H	RESERVED	RESERVED
WDCON[1] C0H 10H	RESERVED	CHIPID C2H 3XH	RESERVED	RESERVED	RESERVED	EDARL C6H 00H	EDARH C7H 00H
IP[1] B8H 00H	ECON B9H 00H	RESERVED	RESERVED	EDATA1 BCH 00H	EDATA2 BDH 00H	EDATA3 BEH 00H	EDATA4 BFH 00H
P3[1] B0H FFH	PWM0L B1H 00H	PWM0H B2H 00H	PWM1L B3H 00H	PWM1H B4H 00H	NOT USED	NOT USED	SPH B7H 00H
IE[1] A8H 00H	IEIP2 A9H A0H	RESERVED	RESERVED	RESERVED	RESERVED	PWMCON AEH 00H	CFG831[4] AFH 10H
P2[1] A0H FFH	TIMECON A1H 00H	HTHSEC A2H 00H	SEC A3H 00H	MIN A4H 00H	HOUR A5H 00H	INTVAL A6H 00H	DPCON A7H 00H
SCON[1] 98H 00H	SBUF 99H 00H	I2CDAT 9AH 00H	I2CADD 9BH 55H	NOT USED	T3FD 9DH 00H	T3CON 9EH 00H	NOT USED
P1[1, 2] 90H FFH	NOT USED	NOT USED	NOT USED	NOT USED	NOT USED	NOT USED	NOT USED
TCON[1] 88H 00H	TMOD 89H 00H	TL0 8AH 00H	TL1 8BH 00H	TH0 8CH 00H	TH1 8DH 00H	RESERVED	RESERVED
P0[1] 80H FFH	SP 81H 07H	DPL 82H 00H	DPH 83H 00H	DPP 84H 00H	RESERVED	RESERVED	PCON 87H 00H

SFR MAP KEY:



NOTES:

[1]SFRs WHOSE ADDRESS ENDS IN 0H OR 8H ARE BIT ADDRESSABLE.
[2]THE PRIMARY FUNCTION OF PORT1 IS AS AN ANALOG INPUT PORT; THEREFORE, TO ENABLE THE DIGITAL SECONDARY FUNCTIONS ON THESE PORT PINS, WRITE A '0' TO THE CORRESPONDING PORT 1 SFR BIT.
[3]CALIBRATION COEFFICIENTS ARE PRECONFIGURED ON POWER-UP TO FACTORY CALIBRATED VALUES.
[4]VALUE DEPENDS ON EXTERNAL CRYSTAL.

Figure 6. Special Function Register Locations and Reset Values

Configuring the ADC

The ADuC831's successive approximation ADC is driven by a divided down version of the master clock. To ensure adequate ADC operation, this ADC clock must be between 400 kHz and 6 MHz, and optimum performance is obtained with ADC clock between 400 kHz and 4.5 MHz. Frequencies within this range can easily be achieved with master clock frequencies from 400 kHz to well above 16 MHz with the four ADC clock divide ratios to choose from. For example, with a 12 MHz master clock, set the ADC clock divide ratio to 4 (i.e., ADCCLK = MCLK/4 = 3 MHz) by setting the appropriate bits in ADCCON1 (ADCCON1.5 = 1, ADCCON1.4 = 0).

The total ADC conversion time is 15 ADC clocks, plus 1 ADC clock for synchronization, plus the selected acquisition time (1, 2, 3, or 4 ADC clocks). For the example above, with three clocks acquisition time, total conversion time is 19 ADC clocks (or 6.3 μs for a 3 MHz ADC clock).

In continuous conversion mode, a new conversion begins each time the previous one finishes. The sample rate is then simply the inverse of the total conversion time described above. In the example above, the continuous conversion mode sample rate would be 157.8 kHz.

If using the temperature sensor as the ADC input, the ADC should be configured to use an ADCCLK of MCLK/16 and four acquisition clocks.

Increasing the conversion time on the temperature monitor channel improves the accuracy of the reading. To further improve the accuracy, an external reference with low temperature drift should also be used.

ADC DMA Mode

The on-chip ADC has been designed to run at a maximum conversion speed of 4 μs (247 kHz sampling rate). When converting at this rate, the ADuC831 MicroConverter has 4 μs to read the ADC result and store the result in memory for further postprocessing, otherwise the next ADC sample could be lost. In an interrupt driven routine the MicroConverter would also have to jump to the ADC Interrupt Service routine, which will also increase the time required to store the ADC results. In applications where the ADuC831 cannot sustain the interrupt rate, an ADC DMA mode is provided.

To enable DMA mode, Bit 6 in ADCCON2 (DMA) must be set. This allows the ADC results to be written directly to a 16 MByte external static memory SRAM (mapped into data memory space) without any interaction from the ADuC831 core. This mode allows the ADuC831 to capture a contiguous sample stream at full ADC update rates (247 kHz).

A Typical DMA Mode Configuration Example

To set the ADuC831 into DMA mode a number of steps must be followed:

1. The ADC must be powered down. This is done by ensuring MD1 and MD0 are both set to 0 in ADCCON1.
2. The DMA address pointer must be set to the start address of where the ADC results are to be written. This is done by writing to the DMA mode address pointers DMAL, DMAH, and DMAP. DMAL must be written to first, followed by DMAH, and then by DMAP.
3. The external memory must be preconfigured. This consists of writing the required ADC channel IDs into the top four bits of every second memory location in the external SRAM starting at the first address specified by the DMA address pointer. As the ADC DMA mode operates independent from the ADuC831 core, it is necessary to provide it with a stop command. This is done by duplicating the last channel ID to be converted followed by "1111" into the next channel selection field. A typical preconfiguration of external memory is as follows.

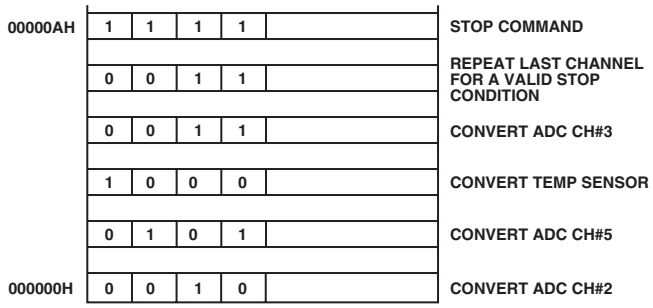


Figure 14. Typical DMA External Memory Preconfiguration

4. The DMA is initiated by writing to the ADC SFRs in the following sequence:
   a. ADCCON2 is written to enable the DMA mode, i.e., MOV ADCCON2, #40H; DMA mode enabled.
   b. ADCCON1 is written to configure the conversion time and power-up of the ADC. It can also enable Timer 2 driven conversions or external triggered conversions if required.
   c. ADC conversions are initiated. This is done by starting single conversions, starting Timer 2 running for Timer 2 conversions or by receiving an external trigger.

When the DMA conversions are completed, the ADC interrupt bit ADCI, is set by hardware and the external SRAM contains the new ADC conversion results as shown below. It should be noted that no result is written to the last two memory locations.

When the DMA mode logic is active, it takes the responsibility of storing the ADC results away from both the user and ADuC831 core logic. As it writes the results of the ADC conversions to external memory, it takes over the external memory interface from the core. Thus, any core instructions that access the external memory while DMA mode is enabled will not get access to it. The core will execute the instructions and they will take the same time to execute but they will not gain access to the external memory.

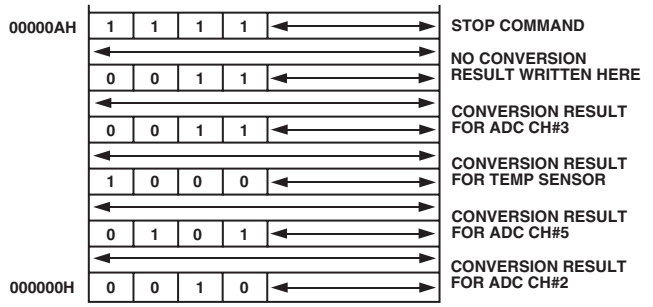


Figure 15. Typical External Memory Configuration Post ADC DMA Operation

INITIATING CALIBRATION IN CODE

When calibrating the ADC, using ADCCON1 the ADC should be set up into the configuration in which it will be used. The ADCCON3 register can then be used to set the device up and calibrate the ADC offset and gain.

```
MOV ADCCON1,#08CH     ;ADC on; ADCCLK set
                      ;to divide by 16, 4
                      ;acquisition clock
```

To calibrate device offset:

```
MOV ADCCON2,#0BH      ;select internal AGND
MOV ADCCON3,#25H      ;select offset calibration,
                      ;31 averages per bit,
                      ;offset calibration
```

To calibrate device gain:

```
MOV ADCCON2,#0CH      ;select internal VREF
MOV ADCCON3,#27H      ;select offset calibration,
                      ;31 averages per bit,
                      ;offset calibration
```

To calibrate system offset:
Connect system AGND to an ADC channel input (0).

```
MOV ADCCON2,#00H      ;select external AGND
MOV ADCCON3,#25H      ;select offset calibration,
                      ;31 averages per bit
```

To calibrate system gain:
Connect system V_{REF} to an ADC channel input (1).

```
MOV ADCCON2,#01H      ;select external VREF
MOV ADCCON3,#27H      ;select offset calibration,
                      ;31 averages per bit,
                      ;offset calibration
```

The calibration cycle time, T_{CAL}, is calculated by the following equation assuming a 16 MHz crystal:

$$T_{CAL} = 14 \times ADCCLK \times NUMAV \times (16 + T_{ACQ})$$

For an ADCCLK/F_{CORE} divide ratio of 16, a T_{ACQ} = 4 ADCCLK, NUMAV = 15, the calibration cycle time is:

$$T_{CAL} = 14 \times (1/1000000) \times 15 \times (16 + 4)$$
$$T_{CAL} = 4.2\ ms$$

In a calibration cycle the ADC busy flag (Bit 7), instead of framing an individual ADC conversion as in normal mode, will go high at the start of calibration and only return to zero at the end of the calibration cycle. It can therefore be monitored in code to indicate when the calibration cycle is completed. The following code can be used to monitor the BUSY signal during a calibration cycle:

```
WAIT:
   MOV A, ADCCON3    ;move ADCCON3 to A
   JB ACC.7, WAIT    ;If Bit 7 is set jump to
                      WAIT else continue
```

NONVOLATILE FLASH/EE MEMORY

Flash/EE Memory Overview

The ADuC831 incorporates Flash/EE memory technology on-chip to provide the user with nonvolatile, in-circuit reprogrammable code, and data memory space. Flash/EE memory is a relatively recent type of nonvolatile memory technology and is based on a single transistor cell architecture.

This technology is basically an outgrowth of EPROM technology and was developed through the late 1980s. Flash/EE memory takes the flexible in-circuit reprogrammable features of EEPROM and combines them with the space efficient/density features of EPROM (see Figure 17).

Because Flash/EE technology is based on a single transistor cell architecture, a Flash memory array, like EPROM, can be implemented to achieve the space efficiencies or memory densities required by a given design. Like EEPROM, Flash memory can be programmed in-system at a byte level, although it must first be erased; the erase being performed in page blocks. Thus, Flash memory is often and more correctly referred to as Flash/EE memory.

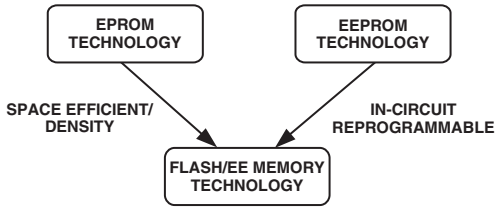


Figure 17. Flash/EE Memory Development

Overall, Flash/EE memory represents a step closer to the ideal memory device that includes nonvolatility, in-circuit programmability, high density and low cost. Incorporated in the ADuC831, Flash/EE memory technology allows the user to update program code space in-circuit, without the need to replace onetime programmable (OTP) devices at remote operating nodes.

Flash/EE Memory and the ADuC831

The ADuC831 provides two arrays of Flash/EE memory for user applications. 62 kBytes of Flash/EE program space are provided on-chip to facilitate code execution without any external discrete ROM device requirements. The program memory can be programmed in-circuit using the serial download mode provided, using conventional third party memory programmers, or via a user defined protocol that can configure it as data if required.

A 4 kByte Flash/EE data memory space is also provided on-chip. This may be used as a general-purpose nonvolatile scratchpad area. User access to this area is via a group of six SFRs. This space can be programmed at a byte level, although it must first be erased in 4-byte pages.

ADuC831 Flash/EE Memory Reliability

The Flash/EE program and data memory arrays on the ADuC831 are fully qualified for two key Flash/EE memory characteristics, namely Flash/EE Memory Cycling Endurance and Flash/EE Memory Data Retention.

Endurance quantifies the ability of the Flash/EE memory to be cycled through many program, read, and erase cycles. In real terms, a single endurance cycle is composed of four independent, sequential events. These events are defined as:

a. Initial page erase sequence
b. Read/verify sequence
c. Byte program sequence
d. Second read/verify sequence

→ A single Flash/EE Memory Endurance Cycle

In reliability qualification, every byte in both the program and data Flash/EE memory is cycled from 00H to FFH until a first fail is recorded, signifying the endurance limit of the on-chip Flash/EE memory.

As indicated in the specification pages of this data sheet, the ADuC831 Flash/EE Memory Endurance qualification has been carried out in accordance with JEDEC Specification A117 over the industrial temperature range of –40°C to +25°C and +85°C to +125°C. The results allow the specification of a minimum endurance figure over supply and temperature of 100,000 cycles, with an endurance figure of 700,000 cycles being typical of operation at 25°C.

Retention quantifies the ability of the Flash/EE memory to retain its programmed data over time. Again, the ADuC831 has been qualified in accordance with the formal JEDEC Retention Lifetime Specification (A117) at a specific junction temperature (T_J = 55°C). As part of this qualification procedure, the Flash/EE memory is cycled to its specified endurance limit described above before data retention is characterized. This means that the Flash/EE memory is guaranteed to retain its data for its full specified retention lifetime every time the Flash/EE memory is reprogrammed. It should also be noted that retention lifetime, based on an activation energy of 0.6 eV, will derate with T_J as shown in Figure 18.

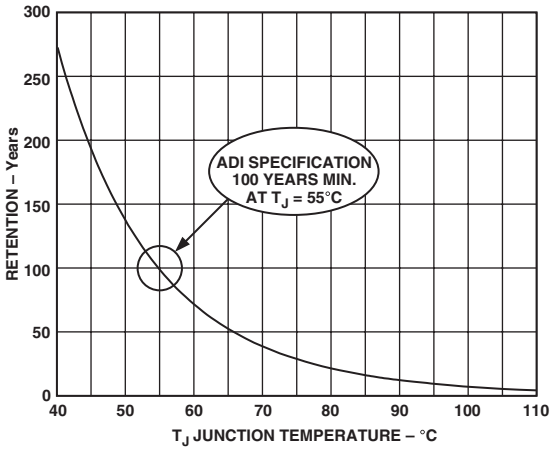


Figure 18. Flash/EE Memory Data Retention

Using the Flash/EE Program Memory

The 62 kByte Flash/EE program memory array is mapped into the lower 62 kBytes of the 64 kBytes program space addressable by the ADuC831, and is used to hold user code in typical applications.

The program memory Flash/EE memory arrays can be programmed in three ways:

(1) Serial Downloading (In-Circuit Programming)

The ADuC831 facilitates code download via the standard UART serial port. The ADuC831 will enter serial download mode after a reset or power cycle if the $\overline{\text{PSEN}}$ pin is pulled low through an external 1 kΩ resistor. Once in serial download mode, the user can download code to the full 62 kBytes of Flash/EE program memory while the device is in circuit in its target application hardware.

A PC serial download executable is provided as part of the ADuC831 QuickStart development system. The Serial Download protocol is detailed in a MicroConverter Applications Note uC004.

(2) Parallel Programming

The parallel programming mode is fully compatible with conventional third party Flash or EEPROM device programmers. In this mode Ports P0, P1, and P2 operate as the external data and address bus interface, ALE operates as the Write Enable strobe, and Port P3 is used as a general configuration port that configures the device for various program and erase operations during parallel programming. The high voltage (12 V) supply required for Flash programming is generated using on-chip charge pumps to supply the high voltage program lines.

The complete parallel programming specification is available on the MicroConverter home page at www.analog.com/microconverter.

(3) User Download Mode (ULOAD)

In Figure 19 we can see that it was possible to use the 62 kBytes of Flash/EE program memory available to the user as one single block of memory. In this mode all of the Flash/EE memory is read-only to user code.

However, the Flash/EE program memory can also be written to during runtime simply by entering ULOAD mode. In ULOAD mode the lower 56 kBytes of program memory can be erased and reprogrammed by user software as shown in Figure 19. ULOAD mode can be used to upgrade your code in the field via any user defined download protocol. Configuring the SPI port on the ADuC831 as a slave, it is possible to completely reprogram the 56 kBytes of Flash/EE program memory in only 5 seconds (see uC007).

Alternatively, ULOAD mode can be used to save data to the 56 kBytes of Flash/EE memory. This can be extremely useful in data logging applications where the ADuC831 can provide up to 60 kBytes of NV data memory on chip (4 kBytes of dedicated Flash/EE data memory also exist).

The upper 6 kBytes of the 62 kBytes of Flash/EE program memory is only programmable via serial download or parallel programming. This means that this space appears as read only to user code. Therefore, it cannot be accidentally erased or reprogrammed by erroneous code execution. This makes it very suitable to use the 6 kBytes as a bootloader. A Bootload Enable option exists in the serial downloader to "Always RUN from E000h after Reset." If using a bootloader, this option is recommended to ensure that the bootloader always executes the correct code after reset.

Programming the Flash/EE program memory via ULOAD mode is described in more detail in the description of ECON and also in technical note uC007.

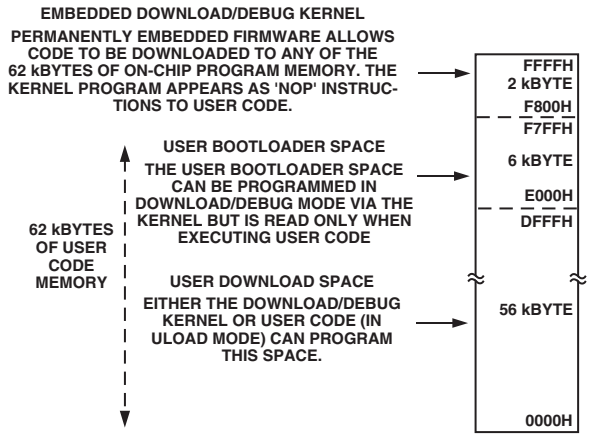


Figure 19. Flash/EE Program Memory Map in ULOAD Mode

Flash/EE Program Memory Security

The ADuC831 facilitates three modes of Flash/EE program memory security. These modes can be independently activated, restricting access to the internal code space. These security modes can be enabled as part of serial download protocol as described in technical note uC004 or via parallel programming. The security modes available on the ADuC831 are described as follows:

Lock Mode

This mode locks the code memory, disabling parallel programming of the program memory. However, reading the memory in parallel mode and reading the memory via a MOVC command from external memory is still allowed. This mode is deactivated by initiating a code-erase command in serial download or parallel programming modes.

Secure Mode

This mode locks code in memory, disabling parallel programming (program and verify/read commands) as well as disabling the execution of a 'MOVC' instruction from external memory, which is attempting to read the op codes from internal memory. Read/Write of internal data Flash/EE from external memory is also disabled. This mode is deactivated by initiating a code-erase command in serial download or parallel programming modes.

Serial Safe Mode

This mode disables serial download capability on the device. If Serial Safe mode is activated and an attempt is made to reset the part into serial download mode, i.e., RESET asserted and de-asserted with $\overline{\text{PSEN}}$ low, the part will interpret the serial download reset as a normal reset only. It will therefore not enter serial download mode but only execute a normal reset sequence. Serial Safe mode can only be disabled by initiating a code-erase command in parallel programming mode.

USING THE FLASH/EE DATA MEMORY

The 4 kBytes of Flash/EE data memory is configured as 1024 pages, each of four bytes. As with the other ADuC831 peripherals, the interface to this memory space is via a group of registers mapped in the SFR space. A group of four data registers (EDATA1–4) are used to hold the four bytes of data at each page. The page is addressed via the two registers EADRH and EADRL. Finally, ECON is an 8-bit control register that may be written with one of nine Flash/EE memory access commands to trigger various read, write, erase, and verify functions.

A block diagram of the SFR interface to the Flash/EE data memory array is shown in Figure 20.

ECON—Flash/EE Memory Control SFR

Programming of either the Flash/EE data memory or the Flash/EE program memory is done through the Flash/EE memory control SFR (ECON). This SFR allows the user to read, write, erase, or verify the 4 kBytes of Flash/EE data memory or the 56 kBytes of Flash/EE program memory.

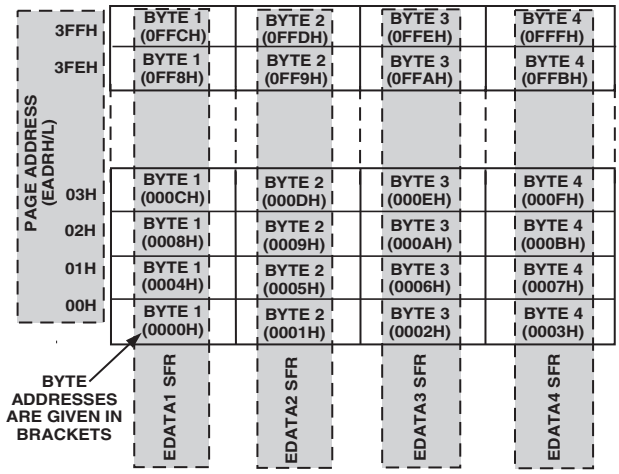


Figure 20. Flash/EE Data Memory Control and Configuration

Table VII. ECON—Flash/EE Memory Commands

ECON VALUE	COMMAND DESCRIPTION (NORMAL MODE) (Power-On Default)	COMMAND DESCRIPTION (ULOAD MODE)
01H READ	Results in 4 bytes in the Flash/EE data memory, addressed by the page address EADRH/L, being read into EDATA1–4.	Not Implemented. Use the MOVC instruction.
02H WRITE	Results in four bytes in EDATA1–4 being written to the Flash/EE data memory, at the page address given by EADRH/L (0 ≤ EADRH/L < 0400H. Note: The four bytes in the page being addressed must be pre-erased.	Results in bytes 0-255 of internal XRAM being written to the 256 bytes of Flash/EE program memory at the page address given by EADRH. (0 ≤ EADRH < E0H) Note: The 256 bytes in the page being addressed must be pre-erased.
03H	Reserved Command	Reserved Command
04H VERIFY	Verifies if the data in EDATA1–4 is contained in the page address given by EADRH/L. A subsequent read of the ECON SFR will result in a 0 being read if the verification is valid, or a nonzero value being read to indicate an invalid verification.	Not Implemented. Use the MOVC and MOVX instructions to verify the WRITE in software.
05H ERASE PAGE	Results in the Erase of the 4-byte page of Flash/EE data memory addressed by the page address EADRH/L.	Results in the 64-byte page of Flash/EE program memory, addressed by the byte address EADRH/L being erased. EADRL can equal any of 64 locations within the page. A new page starts whenever EADRL is equal to 00H, 40H, 80H, or C0H.
06H ERASE ALL	Results in the erase of entire 4 kBytes of Flash/EE data memory.	Results in the Erase of the entire 56 kBytes of ULOAD Flash/EE program memory.
81H READBYTE	Results in the byte in the Flash/EE data memory, addressed by the byte address EADRH/L, being read into EDATA1. (0 ≤ EADRH/L ≤ 0FFFH).	Not Implemented. Use the MOVC command.
82H WRITEBYTE	Results in the byte in EDATA1 being written into Flash/EE data memory, at the byte address EADRH/L.	Results in the byte in EDATA1 being written into Flash/EE program memory, at the byte address EADRH/L (0 ≤ EADRH/L ≤ DFFFH).
0FH EXULOAD	Leaves the ECON instructions to operate on the Flash/EE data memory.	Enters NORMAL mode directing subsequent ECON instructions to operate on the Flash/EE data memory.
F0H ULOAD	Enters ULOAD mode, directing subsequent ECON instructions to operate on the Flash/EE program memory.	Leaves the ECON instructions to operate on the Flash/EE program memory.

ADuC831 Configuration SFR (CFG831)
The CFG831 SFR contains the necessary bits to configure the internal XRAM, EPROM controller, PWM output selection and frequency, DAC buffer, and the extended SP. By default it configures the user into 8051 mode, i.e., extended SP is disabled, internal XRAM is disabled.

CFG831	**ADuC831 Config SFR**
SFR Address	AFH
Power-On Default Value	10*H
Bit Addressable	No

Table VIII. CFG831 SFR Bit Designations

Bit	Name	Description
7	EXSP	Extended SP Enable. When set to "1" by the user, the stack will rollover from SPH/SP = 00FFH to 0100H. When set to "0" by the user, the stack will roll over from SP = FFH to SP = 00H.
6	PWPO	PWM Pin Out Selection. Set to "1" by the user = PWM output pins selected as P3.4 and P3.3. Set to "0" by the user = PWM output pins selected as P2.6 and P2.7.
5	DBUF	DAC Output Buffer. Set to "1" by the user = DAC Output Buffer Bypassed. Set to "0" by the user = DAC Output Buffer Enabled.
4	EPM2	Flash/EE Controller and PWM Clock Frequency Configuration Bits.
3	EPM1	Frequency should be configured such that Fosc/Divide Factor = 32 kHz ± 50%.
2	EPM0	EPM2 EPM1 EPM0 Divide Factor 0 0 0 32 0 0 1 64 0 1 0 128 0 1 1 256 1 0 0 512 1 0 1 1024
1	RSVD	Reserved. This bit should always contain 0.
0	XRAMEN	XRAM Enable Bit. When set to "1" the internal XRAM will be mapped into the lower 2 kBytes of the external address space. When set to "0" the internal XRAM will not be accessible and the external data memory will be mapped into the lower 2 kBytes of external data memory.

*Note that the Flash/EE controller bits EPM2, EPM1, EPM0 are set to their correct values depending on the crystal frequency at power-up. The user should not modify these bits so all instructions to the CFG831 register should use the ORL, XRL, or ANL instructions. Value of 10H is for a 11.0592 MHz crystal.

USER INTERFACE TO OTHER ON-CHIP ADuC831 PERIPHERALS

The following section gives a brief overview of the various peripherals also available on-chip. A summary of the SFRs used to control and configure these peripherals is also given.

DAC

The ADuC831 incorporates two 12-bit, voltage output DACs on-chip. Each has a rail-to-rail voltage output buffer capable of driving 10 kΩ/100 pF. Each has two selectable ranges, 0 V to V_{REF} (the internal band gap 2.5 V reference) and 0 V to AV_{DD}. Each can operate in 12-bit or 8-bit mode. Both DACs share a control register, DACCON, and four data registers, DAC1H/L, DAC0H/L. It should be noted that in 12-bit asynchronous mode, the DAC voltage output will be updated as soon as the DACL data SFR has been written; therefore, the DAC data registers should be updated as DACH first, followed by DACL. Note: for correct DAC operation on the 0 to V_{REF} range, the ADC must be switched on. This results in the DAC using the correct reference value.

DACCON	**DAC Control Register**
SFR Address	FDH
Power-On Default Value	04H
Bit Addressable	No

Table IX. DACCON SFR Bit Designations

Bit	Name	Description
7	MODE	The DAC MODE bit sets the overriding operating mode for both DACs. Set to "1" = 8-Bit Mode (Write 8 Bits to DACxL SFR). Set to "0"= 12-Bit Mode.
6	RNG1	DAC1 Range Select Bit. Set to "1" = DAC1 Range 0–V_{DD}. Set to "0" = DAC1 Range 0–V_{REF}.
5	RNG0	DAC0 Range Select Bit. Set to "1" = DAC0 Range 0–$V_{DD.}$ Set to "0" = DAC0 Range 0–$V_{REF.}$
4	CLR1	DAC1 Clear Bit. Set to "0" = DAC1 Output Forced to 0 V. Set to "1" = DAC1 Output Normal.
3	CLR0	DAC0 Clear Bit. Set to "0" = DAC1 Output Forced to 0 V. Set to "1" = DAC1 Output Normal.
2	SYNC	DAC0/1 Update Synchronization Bit. When set to "1" the DAC outputs update as soon as DACxL SFRs are written. The user can simultaneously update both DACs by first updating the DACxL/H SFRs while SYNC is "0." Both DACs will then update simultaneously when the SYNC bit is set to "1."
1	$\overline{PD1}$	DAC1 Power-Down Bit. Set to "1" = Power-On DAC1. Set to "0" = Power-Off DAC1.
0	PD0	DAC0 Power-Down Bit. Set to "1" = Power-On DAC0. Set to "0" = Power-Off DAC0.

DACxH/L	**DAC Data Registers**			
Function	DAC Data Registers, written by user to update the DAC output.			
SFR Address	DAC0L (DAC0 Data Low Byte)	→ F9H; DAC1L (DAC1 Data Low Byte)	→ FBH	
	DAC0H (DAC0 Data High Byte)	→ FAH; DAC1H(DAC1 Data High Byte)	→ FCH	
Power-On Default Value	00H	→ All four Registers		
Bit Addressable	No	→ All four Registers		

The 12-bit DAC data should be written into DACxH/L right-justified such that DACxL contains the lower eight bits, and the lower nibble of DACxH contains the upper four bits.

Using the DAC

The on-chip DAC architecture consists of a resistor string DAC followed by an output buffer amplifier, the functional equivalent of which is illustrated in Figure 21. Details of the actual DAC architecture can be found in U.S. Patent Number 5969657 (www.uspto.gov). Features of this architecture include inherent guaranteed monotonicity and excellent differential linearity.

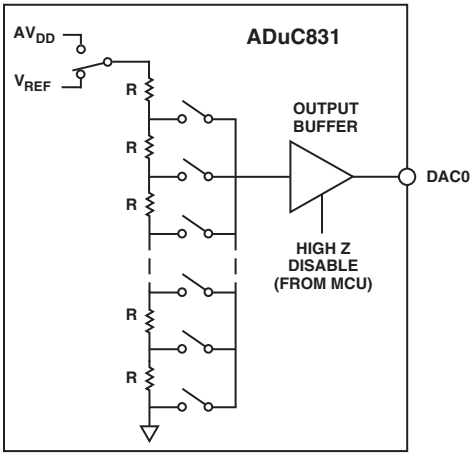


Figure 21. Resistor String DAC Functional Equivalent

As illustrated in Figure 21, the reference source for each DAC is user selectable in software. It can be either AV_{DD} or $V_{REF.}$ In 0-to-AV_{DD} mode, the DAC output transfer function spans from 0 V to the voltage at the AV_{DD} pin. In 0-to-V_{REF} mode, the DAC output transfer function spans from 0 V to the internal $V_{REF,}$ or if an external reference is applied, the voltage at the V_{REF} pin. The DAC output buffer amplifier features a true rail-to-rail output stage implementation. This means that, unloaded, each output is capable of swinging to within less than 100 mV of both AV_{DD} and ground. Moreover, the DAC's linearity specification (when driving a 10 kΩ resistive load to ground) is guaranteed through the full transfer function except codes 0 to 100, and, in 0-to-AV_{DD} mode only, codes 3945 to 4095. Linearity degradation near ground and V_{DD} is caused by saturation of the output amplifier, and a general representation of its effects (neglecting offset and gain error) is illustrated in Figure 22. The dotted line in Figure 22 indicates the ideal transfer function, and the solid line represents what the transfer function might look like with endpoint nonlinearities due to saturation of the output amplifier. Note that Figure 22 represents a transfer function in 0-to-V_{DD} mode only. In 0-to-V_{REF} mode (with V_{REF} < V_{DD}) the lower nonlinearity would be similar, but the upper portion of the transfer function would follow the "ideal" line right to the end (V_{REF} in this case, not V_{DD}), showing no signs of endpoint linearity errors.

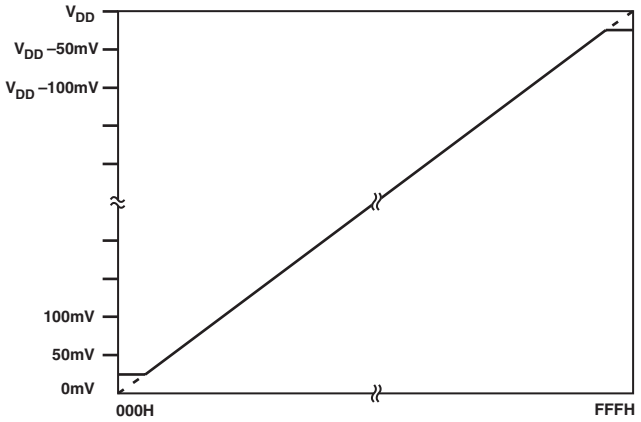


Figure 22. Endpoint Nonlinearities Due to Amplifier Saturation

The end point nonlinearities conceptually illustrated in Figure 22 get worse as a function of output loading. Most of the ADuC831's data sheet specifications assume a 10 kΩ resistive load to ground at the DAC output. As the output is forced to source or sink more current, the nonlinear regions at the top or bottom (respectively) of Figure 22 become larger. With larger current demands, this can significantly limit output voltage swing. Figure 23 and Figure 24 illustrate this behavior. It should be noted that the upper trace in each of these figures is only valid for an output range selection of 0-to-AV_{DD}. In 0-to-V_{REF} mode, DAC loading will not cause highside voltage drops as long as the reference voltage remains below the upper trace in the corresponding figure. For example, if AV_{DD} = 3 V and V_{REF} = 2.5 V, the highside voltage will not be affected by loads less than 5 mA. But somewhere around 7 mA the upper curve in Figure 24 drops below 2.5 V (V_{REF}) indicating that at these higher currents the output will not be capable of reaching V_{REF}.

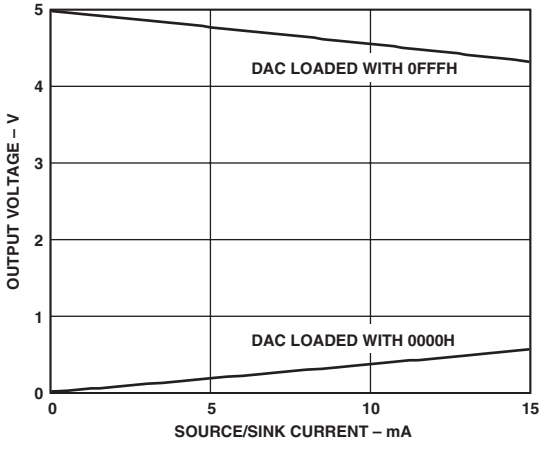


Figure 23. Source and Sink Current Capability with V_{REF} = V_{DD} = 5 V

PULSEWIDTH MODULATOR (PWM)

The PWM on the ADuC831 is highly flexible PWM offering programmable resolution and input clock, and can be configured for any one of six different modes of operation. Two of these modes allow the PWM to be configured as a Σ-Δ DAC with up to 16 bits of resolution. A block diagram of the PWM is shown in Figure 26.

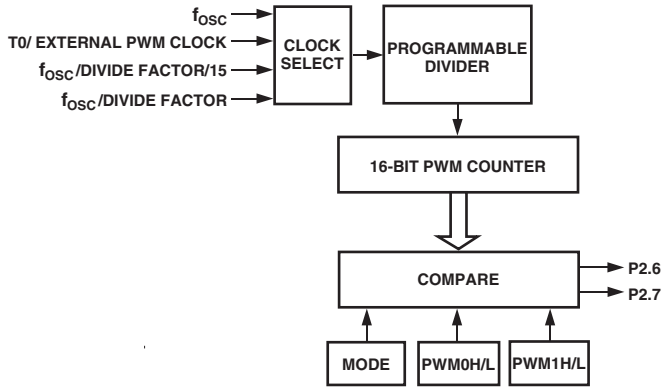


Figure 26. PWM Block Diagram

The PWM uses five SFRs: the control SFR (PWMCON), and four data SFRs (PWM0H, PWM0L, PWM1H, and PWM1L). PWMCON (as described below) controls the different modes of operation of the PWM as well as the PWM clock frequency. PWM0H/L and PWM1H/L are the data registers that determine the duty cycles of the PWM outputs. The output pins that the PWM uses are determined by the CFG831 register and they can be either P2.6 and P2.7 or P3.4 and P3.3. In this section of the data sheet, it is assumed that P2.6 and P2.7 are selected as the PWM outputs.

To use the PWM user software, first write to PWMCON to select the PWM mode of operation and the PWM input clock. Writing to PWMCON also resets the PWM counter. In any of the 16-bit modes of operation (modes 1, 3, 4, 6), user software should write to the PWM0L or PWM1L SFRs first. This value is written to a hidden SFR. Writing to the PWM0H or PWM1H SFRs updates both the PWMxH and the PWMxL SFRs but does not change the outputs until the end of the PWM cycle in progress. The values written to these 16-bit registers are then used in the next PWM cycle.

PWMCON	**PWM Control SFR**
SFR Address	AEH
Power-On Default Value	00H
Bit Addressable	No

Table X. PWMCON SFR Bit Designations

Bit	Name	Description
7	SNGL	Turns Off PWM output at P2.6 or P3.4 Leaving Port Pin Free for Digital I/O.
6	MD2	PWM Mode Bits
5	MD1	The MD2/1/0 bits choose the PWM mode as follows:
4	MD0	MD2 MD1 MD0 Mode 0 0 0 Mode 0: PWM Disabled 0 0 1 Mode 1: Single variable resolution PWM on P2.7 or P3.3 0 1 0 Mode 2: Twin 8-bit PWM 0 1 1 Mode 3: Twin 16-bit PWM 1 0 0 Mode 4: Dual NRZ 16-bit Σ-Δ DAC 1 0 1 Mode 5: Dual 8-bit PWM 1 1 0 Mode 6: Dual RZ 16-bit Σ-Δ DAC 1 1 1 Reserved for future use
3	CDIV1	PWM Clock Divider
2	CDIV0	Scale the clock source for the PWM counter as shown below: CDIV1 CDIV0 Description 0 0 PWM Counter = Selected Clock/1 0 1 PWM Counter = Selected Clock/4 1 0 PWM Counter = Selected Clock/16 1 1 PWM Counter = Selected Clock/64
1	CSEL1	PWM Clock Divider
0	CSEL0	Select the clock source for the PWM as shown below: CSEL1 CSEL0 Description 0 0 PWM Clock = $f_{OCS/DIVIDE\ FACTOR}$/15 (see CFG831 register) 0 1 PWM Clock = $f_{OCS/DIVIDE\ FACTOR}$ (see CFG831 register) 1 0 PWM Clock = External input at P3.4/T0 1 1 PWM Clock = f_{OSC}

MODE 4: Dual NRZ 16-Bit Σ-Δ DAC
Mode 4 provides a high speed PWM output similar to that of a Σ-Δ DAC. Typically, this mode will be used with the PWM clock equal to 16 MHz.

In this mode P2.6 and P2.7 are updated every PWM clock (62 ns in the case of 16 MHz). Over any 65536 cycles (16 bit PWM) PWM0 (P2.6) is high for PWM0H/L cycles and low for (65536 - PWM0H/L) cycles. Similarly PWM1 (P2.7) is high for PWM1H/L cycles and low for (65536 - PWM1H/L) cycles.

For example, if PWM1H was set to 4010H (slightly above one quarter of FS) then typically P2.7 will be low for three clocks and high for one clock (each clock is approximately 80 ns). Over every 65536 clocks the PWM will compensate for the fact that the output should be slightly above one quarter of full scale by having a high cycle followed by only two low cycles.

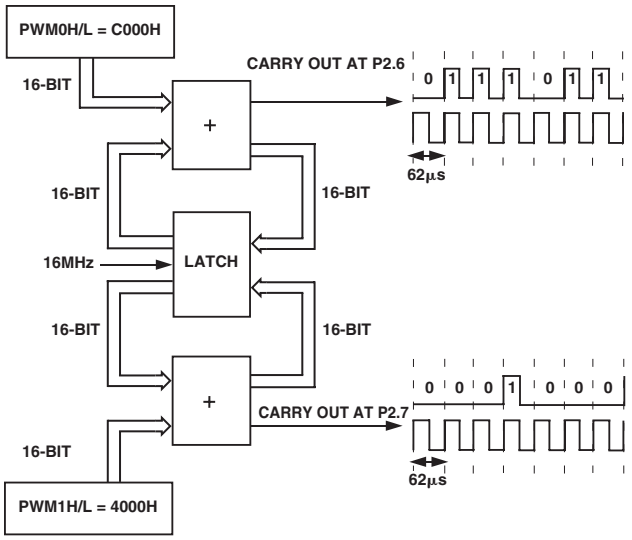


Figure 30. PWM Mode 4

For faster DAC outputs (at lower resolution) write 0s to the LSBs that are not required. If for example only 12-bit performance is required then write 0s to the 4LSBs. This means that a 12-bit accurate Σ-Δ DAC output can occur at 3.906 kHz. Similarly, writing 0s to the 8 LSBs gives an 8-bit accurate Σ-Δ DAC output at 62 kHz.

MODE 5: Dual 8-Bit PWM
In Mode 5, the duty cycle of the PWM outputs and the resolution of the PWM outputs are individually programmable. The maximum resolution of the PWM output is eight bits. The output resolution is set by the PWM1L and PWM1H SFRs for the P2.6 and P2.7 outputs, respectively. PWM0L and PWM0H sets the duty cycles of the PWM outputs at P2.6 and P2.7, respectively. Both PWMs have same clock source and clock divider.

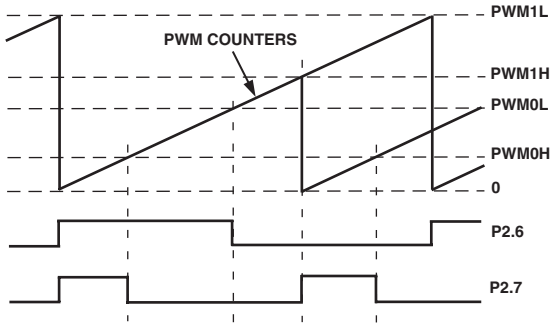


Figure 31. PWM Mode 5

MODE 6: Dual RZ 16-Bit Σ-Δ DAC
Mode 6 provides a high speed PWM output similar to that of a Σ-Δ DAC. Mode 6 operates very similarly to Mode 4. However, the key difference is that Mode 6 provides return to zero (RZ) Σ-Δ DAC output. Mode 4 provides non-return-to-zero Σ-Δ DAC outputs. The RZ mode ensures that any difference in the rise and fall times will not effect the Σ-Δ DAC INL. However, the RZ mode halves the dynamic range of the Σ-Δ DAC outputs from 0–AV_{DD} down to 0–AV_{DD}/2. For best results, this mode should be used with a PWM clock divider of four.

If PWM1H was set to 4010H (slightly above one quarter of FS), then typically P2.7 will be low for three full clocks (3 × 62 ns), high for half a clock (31 ns) and then low again for half a clock (31 ns) before repeating itself. Over every 65536 clocks the PWM will compensate for the fact that the output should be slightly above one quarter of full scale by leaving the output high for two half clocks in four every so often.

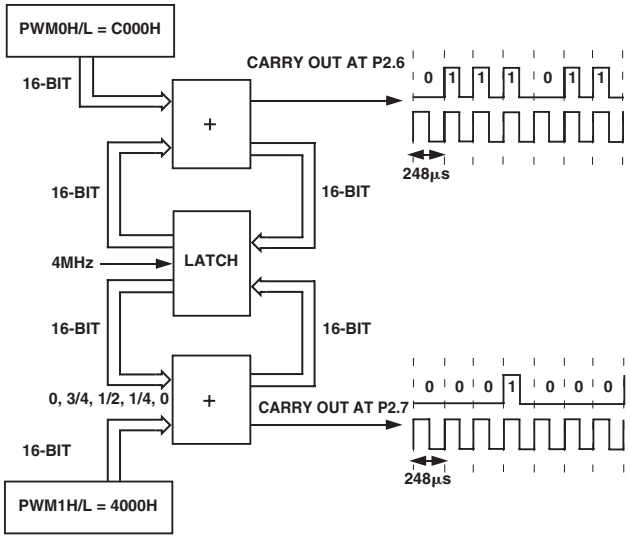


Figure 32. PWM Mode 6

8052 COMPATIBLE ON-CHIP PERIPHERALS

This section gives a brief overview of the various secondary peripheral circuits that are also available to the user on-chip. These remaining functions are mostly 8052 compatible (with a few additional features) and are controlled via standard 8052 SFR bit definitions.

Parallel I/O

The ADuC831 uses four input/output ports to exchange data with external devices. In addition to performing general-purpose I/O, some ports are capable of external memory operations while others are multiplexed with alternate functions for the peripheral features on the device. In general, when a peripheral is enabled, that pin may not be used as a general-purpose I/O pin.

Port 0

Port 0 is an 8-bit, open-drain, bidirectional I/O port that is directly controlled via the Port 0 SFR. Port 0 is also the multiplexed low-order address and data bus during accesses to external program or data memory.

Figure 36 shows a typical bit latch and I/O buffer for a Port 0 port pin. The bit latch (one bit in the port's SFR) is represented as a Type D flip-flop, which will clock in a value from the internal bus in response to a "write to latch" signal from the CPU. The Q output of the flip-flop is placed on the internal bus in response to a "read latch" signal from the CPU. The level of the port pin itself is placed on the internal bus in response to a "read pin" signal from the CPU. Some instructions that read a port activate the "read latch" signal, and others activate the "read pin" signal. See the following Read-Modify-Write Instructions section for more details.

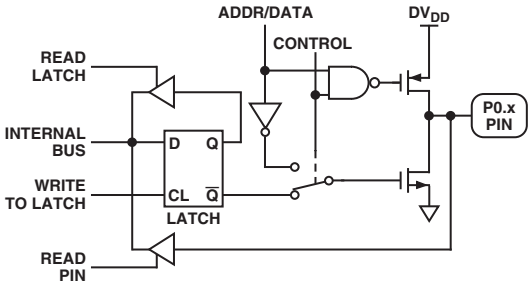


Figure 36. Port 0 Bit Latch and I/O Buffer

As shown in Figure 36, the output drivers of Port 0 pins are switchable to an internal ADDR and ADDR/DATA bus by an internal CONTROL signal for use in external memory accesses. During external memory accesses the P0 SFR gets 1s written to it (i.e., all of its bit latches become 1). When accessing external memory, the CONTROL signal in Figure 36 goes high, enabling push-pull operation of the output pin from the internal address or data bus (ADDR/DATA line). Therefore, no external pull-ups are required on Port 0 in order for it to access external memory.

In general-purpose I/O port mode, Port 0 pins that have 1s written to them via the Port 0 SFR will be configured as "open drain" and will therefore float. In this state, Port 0 pins can be used as high impedance inputs. This is represented in Figure 36 by the NAND gate whose output remains high as long as the CONTROL signal is low, thereby disabling the top FET. External pull-up resistors are therefore required when Port 0 pins are used as general-purpose outputs. Port 0 pins with 0s written to them will drive a logic low output voltage (V_{OL}) and will be capable of sinking 1.6 mA.

Port 1

Port 1 is also an 8-bit port directly controlled via the P1 SFR. Port 1 digital output capability is not supported on this device. Port 1 pins can be configured as digital inputs or analog inputs.

By (power-on) default these pins are configured as analog inputs, i.e., 1 written in the corresponding Port 1 register bit. To configure any of these pins as digital inputs, the user should write a 0 to these port bits to configure the corresponding pin as a high impedance digital input.

These pins also have various secondary functions described in Table XVII.

Table XVII. Port 1, Alternate Pin Functions

Pin	Alternate Function
P1.0	T2 (Timer/Counter 2 External Input)
P1.1	T2EX (Timer/Counter 2 Capture/Reload Trigger)
P1.5	$\overline{SS}$ (Slave Select for the SPI Interface)

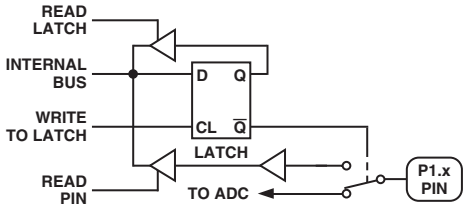


Figure 37. Port 1 Bit Latch and I/O Buffer

Port 2

Port 2 is a bidirectional port with internal pull-up resistors directly controlled via the P2 SFR. Port 2 also emits the high order address bytes during fetches from external program memory and middle and high order address bytes during accesses to the 24-bit external data memory space.

As shown in Figure 38, the output drivers of Ports 2 are switchable to an internal ADDR and ADDR/DATA bus by an internal CONTROL signal for use in external memory accesses (as for Port 0). In external memory addressing mode (CONTROL = 1) the port pins feature push-pull operation controlled by the internal address bus (ADDR line). However, unlike the P0 SFR during external memory accesses, the P2 SFR remains unchanged.

Timer 1 Generated Baud Rates

When Timer 1 is used as the baud rate generator, the baud rates in Modes 1 and 3 are determined by the Timer 1 overflow rate and the value of SMOD as follows:

$$Modes\ 1\ and\ 3\ Baud\ Rate = (2^{SMOD}/32) \times (Timer\ 1\ Overflow\ Rate)$$

The Timer 1 interrupt should be disabled in this application. The Timer itself can be configured for either timer or counter operation, and in any of its three running modes. In the most typical application, it is configured for timer operation in the Autoreload mode (high nibble of TMOD = 0010 binary). In that case, the baud rate is given by the formula:

$$Modes\ 1\ and\ 3\ Baud\ Rate = (2^{SMOD}/32) \times (Core\ Clock/(12 \times [256 - TH1]))$$

Table XXIV shows some commonly used baud rates and how they might be calculated from a core clock frequency of 11.0592 MHz and 12 MHz. Generally speaking, a 5% error is tolerable using asynchronous (start/stop) communications.

Table XXIV. Commonly-Used Baud Rates, Timer 1

Ideal Baud	Core CLK (MHz)	SMOD Value	TH1-Reload Value	Actual Baud	% Error
9600	12	1	–7 (F9H)	8929	7
19200	11.0592	1	–3 (FDH)	19200	0
9600	11.0592	0	–3 (FDH)	9600	0
2400	11.0592	0	–12 (F4H)	2400	0

Timer 2 Generated Baud Rates

Baud rates can also be generated using Timer 2. Using Timer 2 is similar to using Timer 1 in that the timer must overflow 16 times before a bit is transmitted/received. Because Timer 2 has a 16-bit Autoreload mode, a wider range of baud rates is possible using Timer 2.

$$Modes\ 1\ and\ 3\ Baud\ Rate = (1/16) \times (Timer\ 2\ Overflow\ Rate)$$

Therefore, when Timer 2 is used to generate baud rates, the timer increments every two clock cycles and not every core machine cycle as before. Thus, it increments six times faster than Timer 1, and therefore baud rates six times faster are possible. Because Timer 2 has 16-bit autoreload capability, very low baud rates are still possible.

Timer 2 is selected as the baud rate generator by setting the TCLK and/or RCLK in T2CON. The baud rates for transmit and receive can be simultaneously different. Setting RCLK and/or TCLK puts Timer 2 into its baud rate generator mode as shown in Figure 53.

In this case, the baud rate is given by the formula:

$$Modes\ 1\ and\ 3\ Baud\ Rate = (Core\ Clk)/(32 \times [65536 - (RCAP2H,\ RCAP2L)])$$

Table XXV shows some commonly used baud rates and how they might be calculated from a core clock frequency of 11.0592 MHz and 12 MHz.

Table XXV. Commonly Used Baud Rates, Timer 2

Ideal Baud	Core CLK (MHz)	RCAP2H Value	RCAP2L Value	Actual Baud	% Error
19200	12	–1 (FFH)	–20 (ECH)	19661	2.4
9600	12	–1 (FFH)	–41 (D7H)	9591	0.1
2400	12	–1 (FFH)	–164 (5CH)	2398	0.1
1200	12	–2 (FEH)	–72 (B8H)	1199	0.1
19200	11.0592	–1 (FFH)	–18 (EEH)	19200	0
9600	11.0592	–1 (FFH)	–36 (DCH)	9600	0
2400	11.0592	–1 (FFH)	–144 (70H)	2400	0
1200	11.0592	–2 (FFH)	–32 (E0H)	1200	0

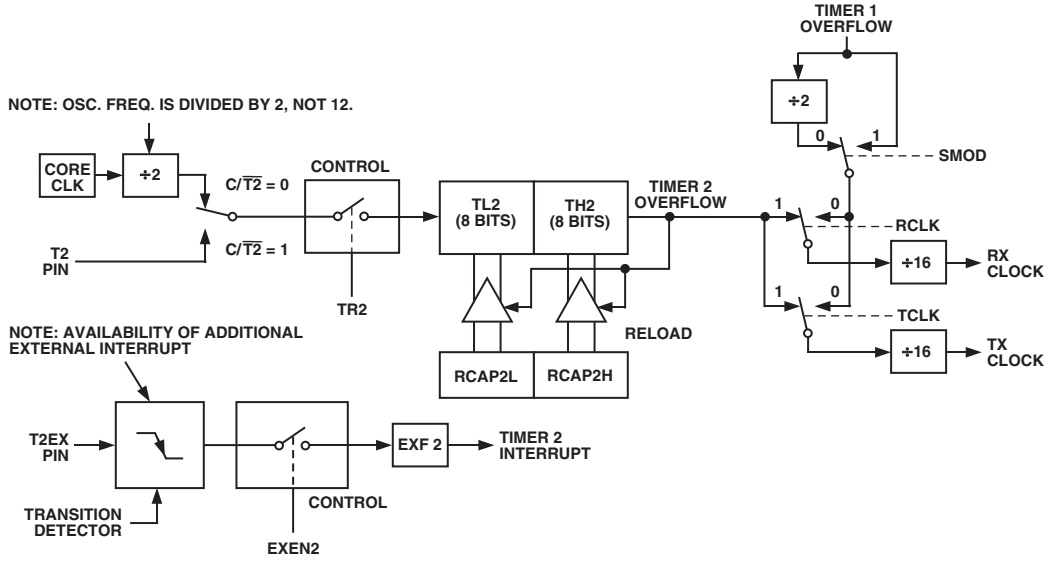


Figure 53. Timer 2, UART Baud Rates

Timer 3 Generated Baud Rates

The high integer dividers in a UART block mean that high speed baud rates are not always possible using some particular crystals. For example, using a 12 MHz crystal, a baud rate of 115200 is not possible. To address this problem, the ADuC831 has added a dedicated baud rate timer (Timer 3) specifically for generating highly accurate baud rates.

Timer 3 can be used instead of Timer 1 or Timer 2 for generating very accurate high speed UART baud rates including 115200 and 230400. Timer 3 also allows a much wider range of baud rates to be obtained. In fact, every desired bit rate from 12 bit/s to 393216 bit/s can be generated to within an error of ±0.8%. Timer 3 also frees up the other three timers, allowing them to be used for different applications. A block diagram of Timer 3 is shown in Figure 54 below.

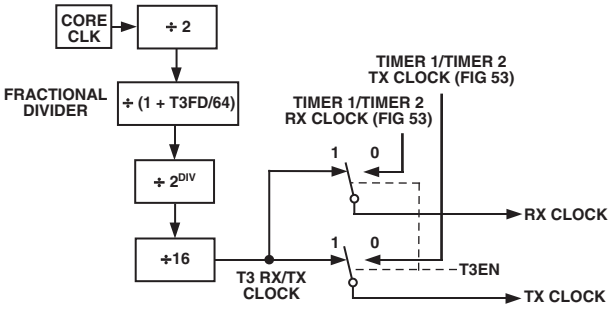


Figure 54. Timer 3, UART Baud Rates

Two SFRs (T3CON and T3FD) are used to control Timer 3. T3CON is the baud rate control SFR, allowing Timer 3 to be used to set up the UART baud rate, and setting up the binary divider (DIV).

Table XXVI. T3CON SFR Bit Designations

Bit	Name	Description
7	T3BAUDEN	T3UARTBAUD Enable Set to enable Timer 3 to generate the baud rate. When set, PCON.7, T2CON.4 and T2CON.5 are ignored. Cleared to let the baud rate be generated as per a standard 8052.
6		–
5		–
4		–
3		–
2	DIV2	Binary Divider Factor.
1	DIV1	DIV2 DIV1 DIV0 Bin Divider
0	DIV0	0 0 0 1 0 0 1 1 0 1 0 1 0 1 1 1 1 0 0 1 1 0 1 1 1 1 0 1 1 1 1 1

The appropriate value to write to the DIV2-1-0 bits can be calculated using the following formula where f_{CORE} is the crystal frequency:

Note: The DIV value must be rounded down.

$$DIV = \frac{\log\left(\frac{f_{CORE}}{32 \times Baud\ Rate}\right)}{\log(2)}$$

T3FD is the fractional divider ratio required to achieve the required baud rate. We can calculate the appropriate value for T3FD using the following formula.

Note: T3FD should be rounded to the nearest integer.

$$T3FD = \frac{2 \times f_{CORE}}{2^{DIV} \times Baud\ Rate}$$

Once the values for DIV and T3FD are calculated the actual baud rate can be calculated using the following formula.

$$Actual\ Baud\ Rate = \frac{2 \times f_{CORE}}{2^{DIV} \times (T3FD + 64)}$$

For example, to get a baud rate of 115200 while operating at 11.0592 MHz:

$$DIV = LOG\left(11059200 / \left(32 \times 115200\right)\right) / LOG2 = 1.58 = 1$$

$$T3FD = \left(2 \times 11059200\right) / \left(2^1 \times 115200\right) - 64 = 32 = 20H$$

Therefore, the actual baud rate is 115200 bit/s.

Table XXVII. Commonly Used Baud Rates Using Timer 3

Ideal Baud	Crystal	DIV	T3CON	T3FD	% Error
230400	11.0592	0	80H	20H	0.0
115200	11.0592	1	81H	20H	0.0
57600	11.0592	2	82H	20H	0.0
38400	11.0592	3	83H	08H	0.0
19200	11.0592	4	84H	08H	0.0
9600	11.0592	5	85H	08H	0.0
230400	12	0	80H	28H	0.16
115200	12	1	81H	28H	0.16
57600	12	2	82H	28H	0.16
38400	12	3	83H	0EH	0.16
19200	12	4	84H	0EH	0.16
9600	12	5	85H	0EH	0.16
230400	14	0	80H	3AH	0.39
115200	14	1	81H	3AH	0.39
57600	14	2	82H	3AH	0.39
38400	14	3	83H	1BH	0.16
19200	14	4	84H	1BH	0.16
9600	14	5	85H	1BH	0.16
230400	16	1	81H	05H	0.64
115200	16	2	82H	05H	0.64
57600	16	3	83H	05H	0.64
38400	16	3	83H	28H	0.16
19200	16	4	84H	28H	0.16
9600	16	5	85H	28H	0.16

INTERRUPT SYSTEM

The ADuC831 provides a total of nine interrupt sources with two priority levels. The control and configuration of the interrupt system is carried out through three Interrupt-related SFRs.

IE	Interrupt Enable Register
IP	Interrupt Priority Register
IEIP2	Secondary Interrupt Enable Register

IE **Interrupt Enable Register**

SFR Address	A8H
Power-On Default Value	00H
Bit Addressable	Yes

Table XXVIII. IE SFR Bit Designations

Bit	Name	Description
7	EA	Written by User to Enable "1" or Disable "0" All Interrupt Sources
6	EADC	Written by User to Enable "1" or Disable "0" ADC Interrupt
5	ET2	Written by User to Enable "1" or Disable "0" Timer 2 Interrupt
4	ES	Written by User to Enable "1" or Disable "0" UART Serial Port Interrupt
3	ET1	Written by User to Enable "1" or Disable "0" Timer 1 Interrupt
2	EX1	Written by User to Enable "1" or Disable "0" External Interrupt 1
1	ET0	Written by User to Enable "1" or Disable "0" Timer 0 Interrupt
0	EX0	Written by User to Enable "1" or Disable "0" External Interrupt 0

IP **Interrupt Priority Register**

SFR Address	B8H
Power-On Default Value	00H
Bit Addressable	Yes

Table XXIX. IP SFR Bit Designations

Bit	Name	Description
7	----	Reserved for Future Use
6	PADC	Written by User to Select ADC Interrupt Priority ("1" = High; "0" = Low)
5	PT2	Written by User to Select Timer 2 Interrupt Priority ("1" = High; "0" = Low)
4	PS	Written by User to Select UART Serial Port Interrupt Priority ("1" = High; "0" = Low)
3	PT1	Written by User to Select Timer 1 Interrupt Priority ("1" = High; "0" = Low)
2	PX1	Written by User to Select External Interrupt 1 Priority ("1" = High; "0" = Low)
1	PT0	Written by User to Select Timer 0 Interrupt Priority ("1" = High; "0" = Low)
0	PX0	Written by User to Select External Interrupt 0 Priority ("1" = High; "0" = Low)

IEIP2 **Secondary Interrupt Enable Register**

SFR Address	A9H
Power-On Default Value	A0H
Bit Addressable	No

Table XXX. IEIP2 SFR Bit Designations

Bit	Name	Description
7	----	Reserved for Future Use
6	PTI	Priority for Time Interval Interrupt
5	PPSM	Priority for Power Supply Monitor Interrupt
4	PSI	Priority for SPI/I^2C Interrupt
3	----	This Bit Must Contain Zero
2	ETI	Written by User to Enable "1" or Disable "0" Time Interval Counter Interrupt
1	EPSMI	Written by User to Enable "1" or Disable "0" Power Supply Monitor Interrupt
0	ESI	Written by User to Enable "1" or Disable "0" SPI/I^2C Serial Port Interrupt

Interrupt Priority

The Interrupt Enable registers are written by the user to enable individual interrupt sources, while the Interrupt Priority registers allow the user to select one of two priority levels for each interrupt. An interrupt of a high priority may interrupt the service routine of a low priority interrupt, and if two interrupts of different priority occur at the same time, the higher level interrupt will be serviced first. An interrupt cannot be interrupted by another interrupt of the same priority level. If two interrupts of the same priority level occur simultaneously, a polling sequence is observed as shown in Table XXXI.

Table XXXI. Priority within an Interrupt Level

Source	Priority	Description
PSMI	1 (Highest)	Power Supply Monitor Interrupt
WDS	2	Watchdog Timer Interrupt
IE0	2	External Interrupt 0
ADCI	3	ADC Interrupt
TF0	4	Timer/Counter 0 Interrupt
IE1	5	External Interrupt 1
TF1	6	Timer/Counter 1 Interrupt
I2CI + ISPI	7	SPI Interrupt
RI + TI	8	Serial Interrupt
TF2 + EXF2	9 (Lowest)	Timer/Counter 2 Interrupt
TII	11 (Lowest)	Time Interval Counter Interrupt

Interrupt Vectors

When an interrupt occurs, the program counter is pushed onto the stack and the corresponding interrupt vector address is loaded into the program counter. The Interrupt Vector Addresses are shown in Table XXXII.

Table XXXII. Interrupt Vector Addresses

Source	Vector Address
IE0	0003H
TF0	000BH
IE1	0013H
TF1	001BH
RI + TI	0023H
TF2 + EXF2	002BH
ADCI	0033H
I2CI + ISPI	003BH
PSMI	0043H
TII	0053H
WDS	005BH

ADuC831 HARDWARE DESIGN CONSIDERATIONS

This section outlines some of the key hardware design considerations that must be addressed when integrating the ADuC831 into any hardware system.

Clock Oscillator

The clock source for the ADuC831 can come either from an external source or from the internal clock oscillator. To use the internal clock oscillator, connect a parallel resonant crystal between XTAL1 and XTAL2, and connect a capacitor from each pin to ground as shown below.

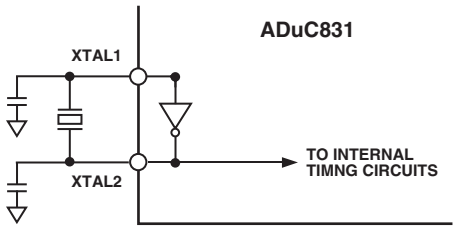


Figure 55. External Parallel Resonant Crystal Connections

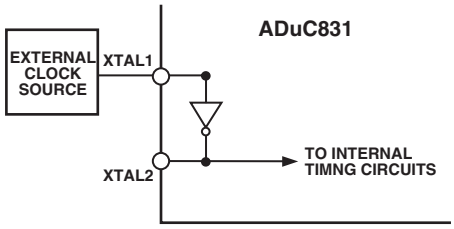


Figure 56. Connecting an External Clock Source

Whether using the internal oscillator or an external clock source, the ADuC831's specified operational clock speed range is 400 kHz to 16 MHz. The core itself is static, and will function all the way down to dc. But at clock speeds slower that 400 kHz the ADC will no longer function correctly. Therefore, to ensure specified operation, use a clock frequency of at least 400 kHz and no more than 16 MHz. Note: the Flash/EE memory may not program correctly at a clock frequency of less than 2 MHz.

External Memory Interface

In addition to its internal program and data memories, the ADuC831 can access up to 64 kBytes of external program memory (ROM/PROM/etc.) and up to 16 MBytes of external data memory (SRAM).

To select from which code space (internal or external program memory) to begin executing instructions, tie the $\overline{EA}$ (external access) pin high or low, respectively. When $\overline{EA}$ is high (pulled up to V_{DD}), user program execution will start at address 0 of the internal 62 kBytes Flash/EE code space. When $\overline{EA}$ is low (tied to ground) user program execution will start at address 0 of the external code space.

A second very important function of the $\overline{EA}$ pin is described in the Single Pin Emulation Mode section.

External program memory (if used) must be connected to the ADuC831 as illustrated in Figure 57. Note that 16 I/O lines

TIMING SPECIFICATIONS[1, 2, 3] (AV_{DD} = DV_{DD} = 3.0 V or 5.0 V ± 10%. All specifications T_A = T_{MIN} to T_{MAX}, unless otherwise noted.)

Parameter		12 MHz			Variable Clock			Unit	Figure
		Min	Typ	Max	Min	Typ	Max		
CLOCK INPUT (External Clock Driven XTAL1)									
t_{CK}	XTAL1 Period		83.33		62.5		1000	ns	68
t_{CKL}	XTAL1 Width Low	20			20			ns	68
t_{CKH}	XTAL1 Width High	20			20			ns	68
t_{CKR}	XTAL1 Rise Time			20			20	ns	68
t_{CKF}	XTAL1 Fall Time			20			20	ns	68
t_{CYC}[4]	ADuC831 Machine Cycle Time		1			$12t_{CK}$		µs	

NOTES
[1]AC inputs during testing are driven at DV_{DD} – 0.5 V for a Logic 1 and 0.45 V for a Logic 0. Timing measurements are made at V_{IH} min for a Logic 1 and V_{IL} max for a Logic 0.
[2]For timing purposes, a port pin is no longer floating when a 100 mV change from load voltage occurs. A port pin begins to float when a 100 mV change from the loaded V_{OH}/V_{OL} level occurs.
[3]C_{LOAD} for Port0, ALE, $\overline{PSEN}$ outputs = 100 pF; C_{LOAD} for all other outputs = 80 pF unless otherwise noted.
[4]ADuC831 Machine Cycle Time is nominally defined as MCLKIN/12.

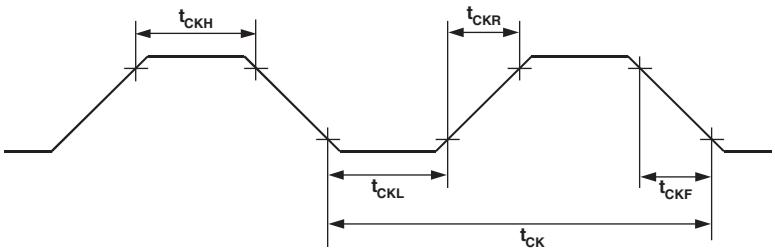


Figure 68. XTAL 1 Input

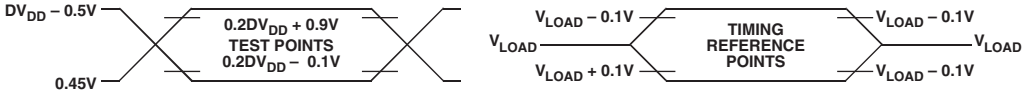


Figure 69. Timing Waveform Characteristics

Parameter		12 MHz Min	12 MHz Max	Variable Clock Min	Variable Clock Max	Unit	Figure
EXTERNAL PROGRAM MEMORY READ CYCLE							
t_{LHLL}	ALE Pulsewidth	127		$2t_{CK} - 40$		ns	70
t_{AVLL}	Address Valid to ALE Low	43		$t_{CK} - 40$		ns	70
t_{LLAX}	Address Hold after ALE Low	53		$t_{CK} - 30$		ns	70
t_{LLIV}	ALE Low to Valid Instruction In		234		$4t_{CK} - 100$	ns	70
t_{LLPL}	ALE Low to $\overline{PSEN}$ Low	53		$t_{CK} - 30$		ns	70
t_{PLPH}	$\overline{PSEN}$ Pulsewidth	205		$3t_{CK} - 45$		ns	70
t_{PLIV}	$\overline{PSEN}$ Low to Valid Instruction In		145		$3t_{CK} - 105$	ns	70
t_{PXIX}	Input Instruction Hold after $\overline{PSEN}$	0		0		ns	70
t_{PXIZ}	Input Instruction Float after $\overline{PSEN}$		59		$t_{CK} - 25$	ns	70
t_{AVIV}	Address to Valid Instruction In		312		$5t_{CK} - 105$	ns	70
t_{PLAZ}	$\overline{PSEN}$ Low to Address Float		25		25	ns	70
t_{PHAX}	Address Hold after $\overline{PSEN}$ High	0		0		ns	70

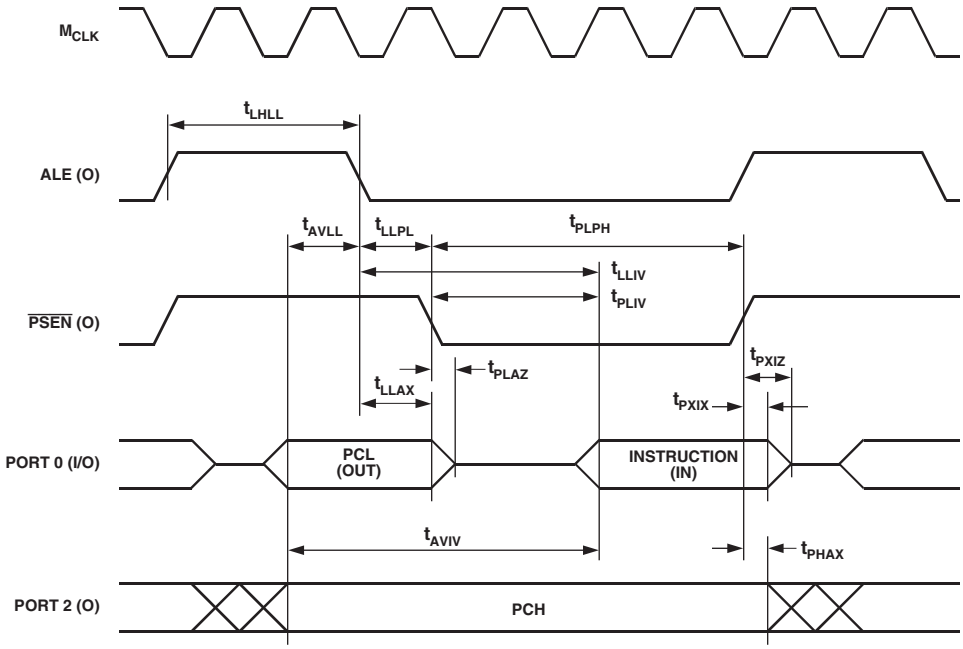


Figure 70. External Program Memory Read Cycle

Parameter		Min	Typ	Max	Unit	Figure
SPI SLAVE MODE TIMING (CPHA = 1)						
t_{SS}	$\overline{SS}$ to SCLOCK Edge	0			ns	77
t_{SL}	SCLOCK Low Pulsewidth		330		ns	77
t_{SH}	SCLOCK High Pulsewidth		330		ns	77
t_{DAV}	Data Output Valid after SCLOCK Edge			50	ns	77
t_{DSU}	Data Input Setup Time before SCLOCK Edge	100			ns	77
t_{DHD}	Data Input Hold Time after SCLOCK Edge	100			ns	77
t_{DF}	Data Output Fall Time		10	25	ns	77
t_{DR}	Data Output Rise Time		10	25	ns	77
t_{SR}	SCLOCK Rise Time		10	25	ns	77
t_{SF}	SCLOCK Fall Time		10	25	ns	77
t_{SFS}	$\overline{SS}$ High after SCLOCK Edge	0			ns	77

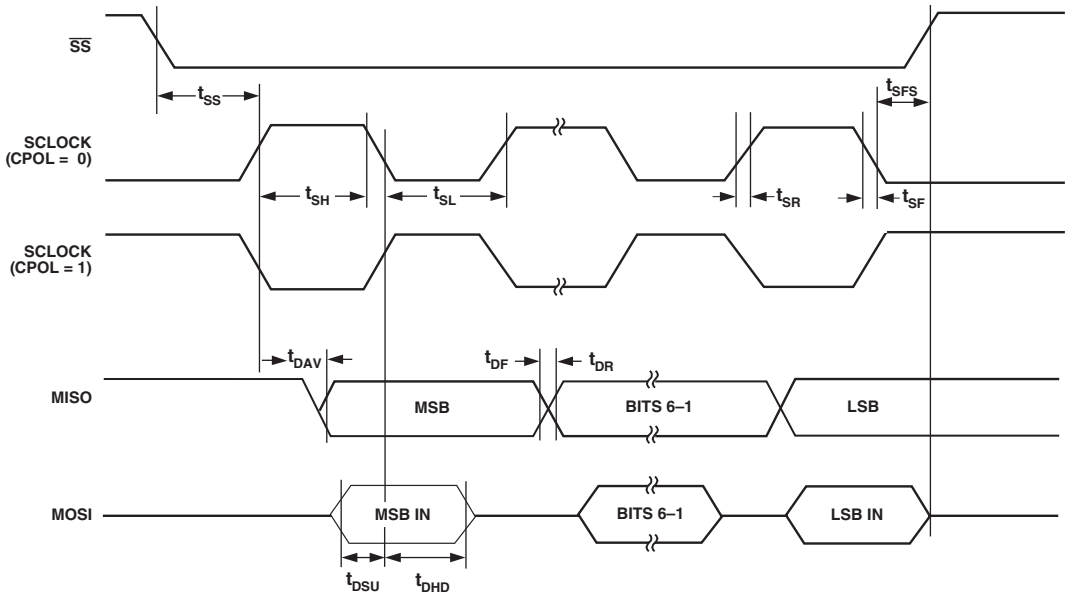


Figure 77. SPI Slave Mode Timing (CPHA = 1)