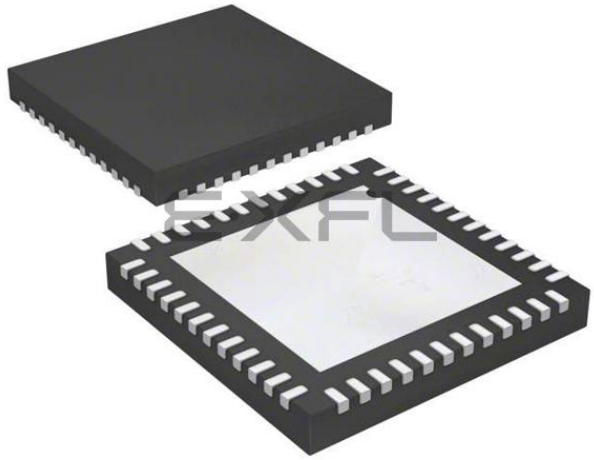

Welcome to E-XFL.COM

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	ARM7®
Core Size	16/32-Bit
Speed	20.48MHz
Connectivity	LINbus, SPI, UART/USART
Peripherals	PSM, Temp Sensor, WDT
Number of I/O	9
Program Memory Size	96KB (48K x 16)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	1.5K x 32
Voltage - Supply (Vcc/Vdd)	3.5V ~ 18V
Data Converters	A/D 2x16b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 115°C (TA)
Mounting Type	Surface Mount
Package / Case	48-VFQFN Exposed Pad, CSP
Supplier Device Package	48-LFCSP-VQ (7x7)
Purchase URL	https://www.e-xfl.com/product-detail/analog-devices/aduc7036dcpz

TIMING SPECIFICATIONS

SPI Timing Specifications

Table 2. SPI Master Mode Timing—Phase Mode = 1

Parameter	Description	Min	Typ	Max	Unit
t_{SL}	SCLK low pulse width[1]		(SPIDIV + 1) × t_{HCLK}		ns
t_{SH}	SCLK high pulse width[1]		(SPIDIV + 1) × t_{HCLK}		ns
t_{DAV}	Data output valid after SCLK edge[2]			(2 × t_{UCLK}) + (2 × t_{HCLK})	ns
t_{DSU}	Data input setup time before SCLK edge	0			ns
t_{DHD}	Data input hold time after SCLK edge[2]	3 × t_{UCLK}			ns
t_{DF}	Data output fall time		3.5		ns
t_{DR}	Data output rise time		3.5		ns
t_{SR}	SCLK rise time		3.5		ns
t_{SF}	SCLK fall time		3.5		ns

[1] t_{HCLK} depends on the clock divider (CD) bits in the POWCON MMR. $t_{HCLK} = t_{UCLK}/2^{CD}$.
[2] t_{UCLK} = 48.8 ns. It corresponds to the 20.48 MHz internal clock from the PLL before the clock divider.

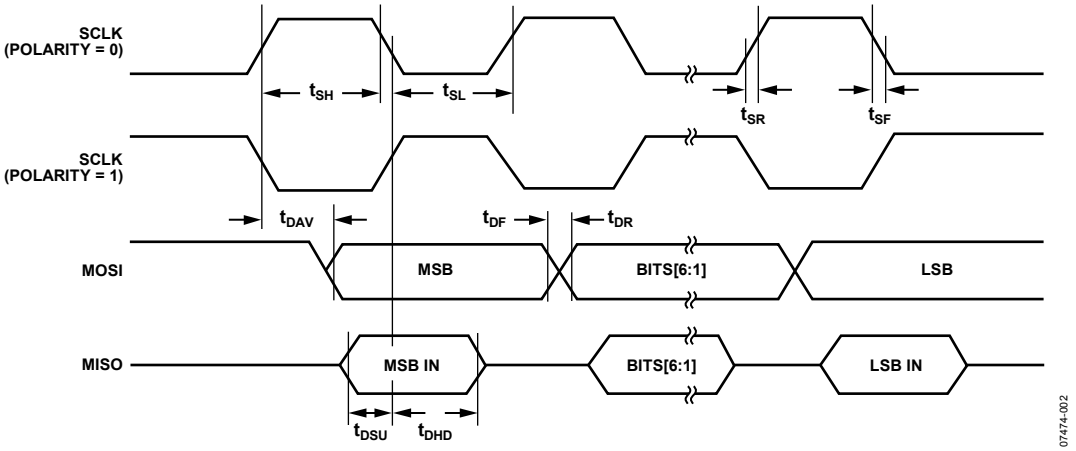

Figure 2. SPI Master Mode Timing—Phase Mode = 1

Table 5. SPI Slave Mode Timing (Phase Mode = 0)

Parameter	Description	Min	Typ	Max	Unit
$t_{\overline{SS}}$	$\overline{SS}$ to SCLK edge		0.5 t_{SL}		ns
t_{SL}	SCLK low pulse width[1]		(SPIDIV + 1) × t_{HCLK}		ns
t_{SH}	SCLK high pulse width[1]		(SPIDIV + 1) × t_{HCLK}		ns
t_{DAV}	Data output valid after SCLK edge[1, 2]			(3 × t_{UCLK}) + (2 × t_{HCLK})	ns
t_{DSU}	Data input setup time before SCLK edge	0			ns
t_{DHD}	Data input hold time after SCLK edge[1, 2]	4 × t_{UCLK}			ns
t_{DF}	Data output fall time		3.5		ns
t_{DR}	Data output rise time		3.5		ns
t_{SR}	SCLK rise time		3.5		ns
t_{SF}	SCLK fall time		3.5		ns
t_{DOCS}	Data output valid after $\overline{SS}$ edge[2]			(3 × t_{UCLK}) + (2 × t_{HCLK})	ns
t_{SFS}	$\overline{SS}$ high after SCLK edge		0.5 t_{SL}		ns

[1] t_{HCLK} depends on the clock divider (CD) bits in the POWCON MMR. $t_{HCLK} = t_{UCLK}/2^{CD}$.
[2] t_{UCLK} = 48.8 ns. It corresponds to the 20.48 MHz internal clock from the PLL before the clock divider.

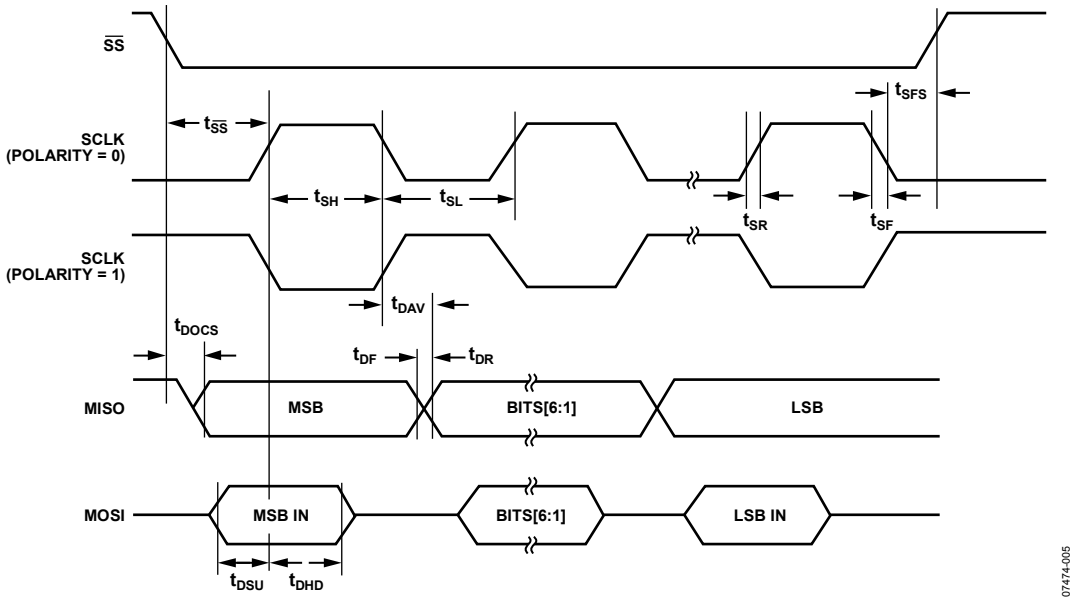

Figure 5. SPI Slave Mode Timing—Phase Mode = 0

The minimum latency for FIQ or IRQ interrupts is five cycles. This consists of the shortest time the request can take through the synchronizer plus the time to enter the exception mode.

Note that the ARM7TDMI initially (first instruction) runs in ARM (32-bit) mode when an exception occurs. The user can immediately switch from ARM mode to Thumb mode if required, for example, when executing interrupt service routines.

MEMORY ORGANIZATION

The ARM7 MCU core, which has a von Neumann-based architecture, sees memory as a linear array of 2^{32} byte locations. As shown in Figure 13, the ADuC7036 maps this into four distinct user areas: a memory area that can be remapped, an SRAM area, a Flash/EE area, and a memory mapped register (MMR) area.

- The first 94 kB of this memory space is used as an area into which the on-chip Flash/EE or SRAM can be remapped.
- The ADuC7036 features a second 4 kB area at the top of the memory map used to locate the MMRs, through which all on-chip peripherals are configured and monitored.
- The ADuC7036 features an SRAM size of 6 kB.
- The ADuC7036 features 96 kB of on-chip Flash/EE memory, 94 kB of which are available to the user and 2 kB of which are reserved for the on-chip kernel.

Any access, either a read or a write, to an area not defined in the memory map results in a data abort exception.

Memory Format

The ADuC7036 memory organization is configured in little endian format: the least significant byte is located in the lowest byte address and the most significant byte in the highest byte address.

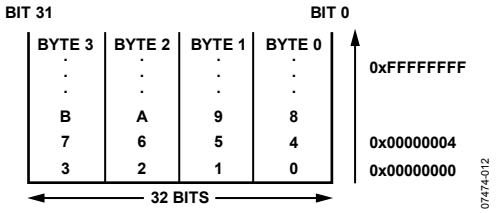

Figure 12. Little Endian Format

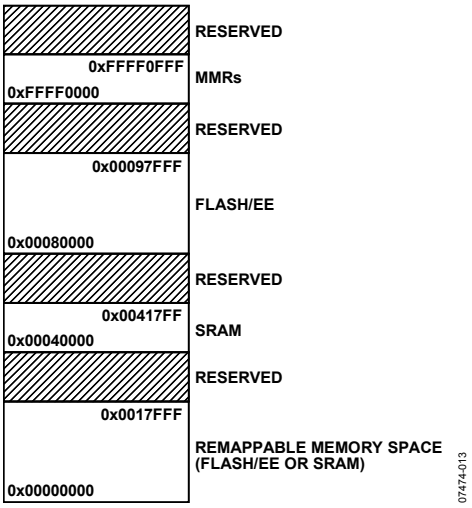

Figure 13. Memory Map

SRAM

The ADuC7036 features 6 kB of SRAM, organized as 1536 × 32 bits, that is, 1536 words located at 0x00040000.

The RAM space can be used as data memory and also as a volatile program space.

ARM code can run directly from SRAM at full clock speed because the SRAM array is configured as a 32-bit-wide memory array. SRAM is readable/writeable in 8-, 16-, and 32-bit segments.

Remap

The ARM exception vectors are situated at the bottom of the memory array, from Address 0x00000000 to Address 0x00000020.

By default, after a reset, the Flash/EE memory is logically mapped to Address 0x00000000.

It is possible to logically remap the SRAM to Address 0x00000000. This is accomplished by setting Bit 0 of the SYSMAP0 MMR located at 0xFFFF0220. To revert Flash/EE to 0x00000000, Bit 0 of SYSMAP0 is cleared.

It may be desirable to remap RAM to 0x00000000 to optimize the interrupt latency of the ADuC7036 because code can run in full 32-bit ARM mode and at maximum core speed. It should be noted that when an exception occurs, the core defaults to ARM mode.

FLASH/EE MEMORY

The ADuC7036 incorporates Flash/EE memory technology on chip to provide the user with nonvolatile, in-circuit reprogrammable memory space.

Like EEPROM, flash memory can be programmed in-system at a byte level, although it must first be erased, with the erasure performed in page blocks. Therefore, flash memory is often and more correctly referred to as Flash/EE memory.

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 within the ADuC7036, Flash/EE memory technology allows the user to update program code space in-circuit without the need to replace one-time programmable (OTP) devices at remote operating nodes.

The Flash/EE memory is located at Address 0x80000. Upon a hard reset, the Flash/EE memory maps to Address 0x00000000. The factory-set default contents of all Flash/EE memory locations is 0xFF. Flash/EE can be read in 8-, 16-, and 32-bit segments and written in 16-bit segments. The Flash/EE is rated for 10,000 endurance cycles. This rating is based on the number of times that each byte is cycled, that is, erased and programmed. Implementing a redundancy scheme in the software ensures that none of the flash locations reach 10,000 endurance cycles.

The user can also write data variables to the Flash/EE memory during run-time code execution, for example, for storing diagnostic battery parameter data.

The entire Flash/EE is available to the user as code and nonvolatile data memory. There is no distinction between data and program space during ARM code processing. The real width of the Flash/EE memory is 16 bits, meaning that in ARM mode (32-bit instruction), two accesses to the Flash/EE are necessary for each instruction fetch. When operating at speeds of less than 20.48 MHz, the Flash/EE memory controller can transparently fetch the second 16-bit halfword (part of the 32-bit ARM operation code) within a single core clock period. Therefore, for speeds less than 20.48 MHz (that is, CD > 0), it is recommended to use ARM mode. For 20.48 MHz operation (that is, CD = 0), it is recommended to operate in Thumb mode.

The page size of this Flash/EE memory is 512 bytes. Typically, it takes the Flash/EE controller 20 ms to erase a page, regardless of CD. Writing a 16-bit word at CD = 0, 1, 2, or 3 requires 50 µs; at CD = 4 or 5, 70 µs; at CD = 6, 80 µs; and at CD = 7, 105 µs.

It is possible to write to a single 16-bit location only twice between erasures; that is, it is possible to walk bytes, not bits. If a location is written to more than twice, the contents of the Flash/EE page may become corrupt.

PROGRAMMING FLASH/EE MEMORY IN-CIRCUIT

The Flash/EE memory can be programmed in-circuit, using a serial download mode via the LIN interface or the integrated JTAG port.

Serial Downloading (In-Circuit Programming)

The ADuC7036 facilitates code download via the LIN/BSD pin.

JTAG Access

The ADuC7036 features an on-chip JTAG debug port to facilitate code downloading and debugging.

ADuC7036 Flash/EE Memory

The total 96 kB of Flash/EE is organized as 47,000 × 16 bits. Of this total, 94 kB is designated as user space, and 2 kB is reserved for boot loader/kernel space.

FLASH/EE CONTROL INTERFACE

The access to and control of the Flash/EE memory on the ADuC7036 are managed by an on-chip memory controller. The controller manages the Flash/EE memory as two separate blocks (Block 0 and Block 1).

Block 0 consists of the 32 kB of Flash/EE memory that is mapped from Address 0x00090000 to Address 0x00097FFF, including the 2 kB kernel space that is reserved at the top of this block.

Block 1 consists of the 64 kB of Flash/EE memory that is mapped from Address 0x00080000 to Address 0x0008FFFF.

It should be noted that the MCU core can continue to execute code from one memory block while an active erase or program cycle is being carried out on the other block. If a command operates on the same block as the code currently executing, the core is halted until the command is complete. This also applies to code execution.

User code, LIN, and JTAG programming use the Flash/EE control interface, consisting of the following MMRs:

- FEExSTA (x = 0 or 1): Read only register. Reflects the status of the Flash/EE control interface.
- FEExMOD (x = 0 or 1): Sets the operating mode of the Flash/EE control interface.
- FEExCON (x = 0 or 1): 8-bit command register. The commands are interpreted as described in Table 13.
- FEExDAT (x = 0 or 1): 16-bit data register.
- FEExADR (x = 0 or 1): 16-bit address register.
- FEExSIG (x = 0 or 1): Holds the 24-bit code signature as a result of the signature command being initiated.
- FEExHID (x = 0 or 1): Protection MMR. Controls read and write protection of the Flash/EE memory code space. If previously configured via the FEExPRO register, FEExHID may require a software key to enable access.
- FEExPRO (x= 0 or 1): A buffer of the FEExHID register. Stores the FEExHID value and is automatically downloaded to the FEExHID registers on subsequent reset and power-on events.

Note that user software must ensure that the Flash/EE controller completes any erase or write cycle before the PLL is powered down. If the PLL is powered down before an erase or write cycle is completed, the Flash/EE page or byte may be corrupted.

Command Sequence for Executing a Mass Erase

Given the significance of the mass erase command, the following specific code sequence must be executed to initiate this operation:

Set Bit 3 in FEExMOD.

Write 0xFFC3 in FEExADR.

Write 0x3CFF in FEExDAT.

Run the mass erase command (Code 0x06) in FEExCON.

This sequence is illustrated by the following example:

```
Int a = FEExSTA;   // Ensure FEExSTA is
cleared
FEExMOD = 0x08
FEExADR = 0xFFC3
FEExDAT = 0x3CFF
FEExCON = 0x06;    // Mass erase command
while (FEExSTA & 0x04){} //Wait for command
to finish
```

It should be noted that to run the mass erase command via FEE0CON, the write protection on the lower 64 kB must be disabled. That is, FEE1HID/FEE1PRO are set to 0xFFFFFFFF. This setting can be accomplished by first removing the protection or by erasing the lower 64 kB.

FEE0STA and FEE1STA Registers

Name: FEE0STA and FEE1STA

Address: 0xFFFF0E00 and 0xFFFF0E80

Default Value: 0x20

Access: Read only

Function: These 8-bit, read only registers can be read by user code, and they reflect the current status of the Flash/EE memory controllers.

Table 14. FEE0STA and FEE1STA MMR Bit Designations

Bit	Description[1]
7 to 4	Not used. These bits are not used and always read as 0.
3	Flash/EE interrupt status bit. Set automatically when an interrupt occurs, that is, when a command is complete and the Flash/EE interrupt enable bit in the FEExMOD register is set. Cleared automatically when the FEExSTA register is read by user code.
2	Flash/EE controller busy. Set automatically when the Flash/EE controller is busy. Cleared automatically when the controller is not busy.
1	Command fail. Set automatically when a command written to FEExCON completes unsuccessfully. Cleared automatically when the FEExSTA register is read by user code.
0	Command successful. Set automatically by the MCU when a command is completed successfully. Cleared automatically when the FEE0STA register is read by user code.

[1] The x represents 0 or 1, designating Flash/EE Block 0 or Flash/EE Block 1.

FEE0ADR and FEE1ADR Registers

Name: FEE0ADR and FEE1ADR

Address: 0xFFFF0E10 and 0xFFFF0E90

Default Value: 0x0000 (FEE1ADR). For FEE0ADR, see the System Identification FEE0ADR section.

Access: Read/write access

Function: These 16-bit registers dictate the address acted upon when a Flash/EE command is executed via FEExCON.

MEMORY MAPPED REGISTERS

The memory mapped register (MMR) space is mapped into the top 4 kB of the MCU memory space and accessed by indirect addressing, loading, and storage commands through the ARM7 banked registers. An outline of the memory mapped register bank for the ADuC7036 is shown in Figure 17.

The MMR space provides an interface between the CPU and all on-chip peripherals. All registers except the ARM7 core registers (described in the ARM Registers section) reside in the MMR area.

As shown in Table 19 to Table 30 in the Complete MMR Listing section, the MMR data widths vary from one byte (eight bits) to four bytes (32 bits). The ARM7 core can access any of the MMRs (single byte or multiple byte width registers) with a 32-bit read or write access.

The resultant read, for example, is aligned per little endian format, as described in the ARM Registers section. However, errors result if the ARM7 core tries to access 4-byte (32-bit) MMRs with a 16-bit access. In the case of a 16-bit write access to a 32-bit MMR, the 16 most significant bits (the upper 16 bits) are written as 0s. The case of a 16-bit read access to a 32-bit MMR, only 16 of the MMR bits can be read.

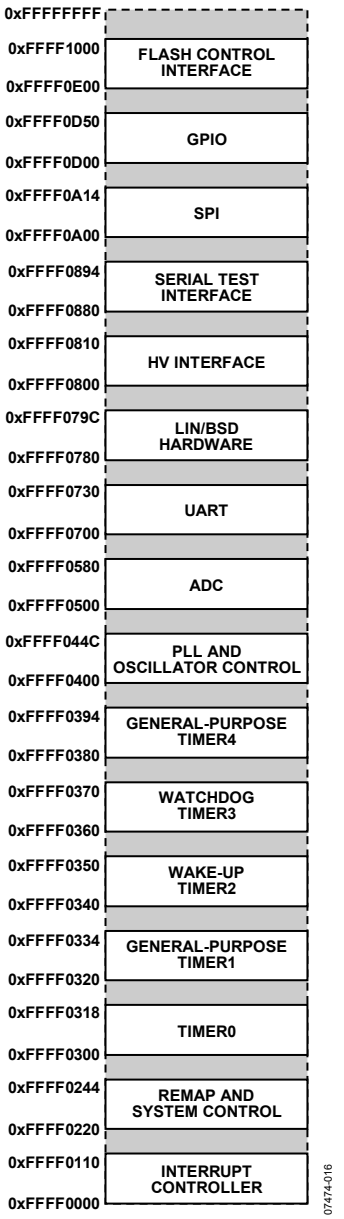

Figure 17. Top-Level MMR Map

COMPLETE MMR LISTING

In Table 19 to Table 30, addresses are listed in hexadecimal code. Access types include R for read, W for write, and RW for read and write.

Table 19. IRQ Address Base = 0xFFFF0000

Address	Name	Byte	Access Type	Default Value	Description
0x0000	IRQSTA	4	R	0x00000000	Active IRQ source. See the Interrupt System section and Table 50.
0x0004	IRQSIG[1]	4	R	N/A	Current state of all IRQ sources (enabled and disabled). See the Interrupt System section and Table 50.
0x0008	IRQEN	4	RW	0x00000000	Enabled IRQ sources. See the Interrupt System section and Table 50.
0x000C	IRQCLR	4	W	N/A	MMR to disable IRQ sources. See the Interrupt System section and Table 50.
0x0010	SWICFG	4	W	N/A	Software interrupt configuration MMR. See the Programmed Interrupts section and Table 51.
0x0100	FIQSTA	4	R	0x00000000	Active IRQ source. See the Interrupt System section and Table 50.
0x0104	FIQSIG[1]	4	R	N/A	Current state of all IRQ sources (enabled and disabled). See the Interrupt System section and Table 50.
0x0108	FIQEN	4	RW	0x00000000	Enabled IRQ sources. See the Interrupt System section and Table 50.
0x010C	FIQCLR	4	W	N/A	MMR to disable IRQ sources. See the Interrupt System section and Table 50.

[1] Depends on the level on the external interrupt pins (GPIO_0, GPIO_5, GPIO_7, and GPIO_8).

Table 20. System Control Address Base = 0xFFFF0200

Address	Name	Byte	Access Type	Default Value	Description
0x0220	SYSMAP0	1	RW	N/A	Remap control register. See the Remap Operation section and Table 10.
0x0230	RSTSTA	1	RW	Varies; depends on type of reset	Reset status MMR. See the Reset section and Table 11 and Table 12.
0x0234	RSTCLR	1	W	N/A	RSTSTA clear MMR. See the Reset section and Table 11 and Table 12.
0x0238	SYSSER0[1]	4	RW	N/A	System Serial Number 0. See the Part Identification section and Table 98 for details.
0x023C	SYSSER1[1]	4	RW	N/A	System Serial Number 1. See the Part Identification section and Table 99 for details.
0x0560	SYSALI[1]	4	R	N/A	System assembly lot ID. See the Part Identification section for details.
0x0240	SYSCHK[1]	4	RW	N/A	Kernel checksum. See the System Kernel Checksum section.

[1] Updated by kernel.

Table 21. Timer Address Base = 0xFFFF0300

Address	Name	Byte	Access Type	Default Value	Description
0x0300	T0LD	2	RW	0x0000	Timer0 load register. See the Timer0—Lifetime Timer and Timer0 Load Register sections.
0x0304	T0VAL0	2	R	0x0000	Timer0 Value Register 0. See the Timer0—Lifetime Timer and Timer0 Value Registers sections.
0x0308	T0VAL1	4	R	0x00000000	Timer0 Value Register 1. See the Timer0—Lifetime Timer and Timer0 Value Registers sections.
0x030C	T0CON	4	RW	0x00000000	Timer0 control MMR. See the Timer0—Lifetime Timer and Timer0 Control Register sections.
0x0310	T0CLRI	1	W	N/A	Timer0 interrupt clear register. See the Timer0—Lifetime Timer and Timer0 Load Register sections.
0x0314	T0CAP	2	R	0x0000	Timer0 capture register. See the Timer0—Lifetime Timer and Timer0 Capture Register sections.
0x0320	T1LD	4	RW	0x00000000	Timer1 load register. See the Timer1 and Timer1 Load Register sections.
0x0324	T1VAL	4	R	0xFFFFFFFF	Timer1 value register. See the Timer1 and Timer1 Value Register sections.
0x0328	T1CON	4	RW	0x01000000	Timer1 control MMR. See the Timer1 and Timer1 Control Register sections.
0x032C	T1CLRI	1	W	N/A	Timer1 interrupt clear register. See the Timer1 and Timer1 Clear Register sections.
0x0330	T1CAP	4	R	0x00000000	Timer1 capture register. See the Timer1 and Timer1 Capture Register sections.

Table 24. UART Base Address = 0XFFFF0700

Address	Name	Byte	Access Type	Default Value	Description
0x0700	COMTX	1	W	N/A	UART transmit register. See the UART Tx Register section.
	COMRX	1	R	0x00	UART receive register. See the UART Rx Register section.
	COMDIV0	1	RW	0x00	UART Standard Baud Rate Generator Divisor Value 0. See the UART Divisor Latch Register 0 section.
0x0704	COMIEN0	1	RW	0x00	UART interrupt enable MMR 0. See the UART Interrupt Enable Register 0 section and Table 84.
	COMDIV1	1	RW	0x00	UART Standard Baud Rate Generator Divisor Value 1. See the UART Divisor Latch Register 1 section.
0x0708	COMIID0	1	R	0x01	UART Interrupt Identification 0. See the UART Interrupt Identification Register 0 section and Table 85.
0x070C	COMCON0	1	RW	0x00	UART Control Register 0. See the UART Control Register 0 section and Table 81.
0x0710	COMCON1	1	RW	0x00	UART Control Register 1. See the UART Control Register 1 section and Table 82.
0x0714	COMSTA0	1	R	0x60	UART Status Register 0. See the UART Status Register 0 section and Table 83.
0X072C	COMDIV2	2	RW	0x0000	UART fractional divider MMR. See the UART Fractional Divider Register section and Table 86.

Table 25. LIN Hardware Sync Base Address = 0XFFFF0780

Address	Name	Byte	Access Type	Default Value	Description
0x0780	LHSSTA	4	R	0x00000000	LHS status MMR. See the LIN Hardware Synchronization Status Register section and Table 92.
0x0784	LHSCON0	2	RW	0x0000	LHS Control MMR 0. See the LIN Hardware Synchronization Control Register 0 section and Table 93.
0x0788	LHSVAL0	2	R	0x0000	LHS Timer0 MMR. See the LIN Hardware Synchronization Timer0 Register section.
0x078C	LHSCON1	1	RW	0x32	LHS Control MMR 1. See the LIN Hardware Synchronization Control Register 1 section and Table 94.
0x0790	LHSVAL1	2	RW	0x0000	LHS Timer1 MMR. See the LIN Hardware Synchronization Break Timer1 Register section.
0x0794	LHSCAP	2	R	0x0000	LHS capture MMR. See the LIN Hardware Synchronization Capture Register section.
0x0798	LHSCMP	2	RW	0x0000	LHS compare MMR. See the LIN Hardware Synchronization Compare Register section.

Table 26. High Voltage Interface Base Address = 0xFFFF0800

Address	Name	Byte	Access Type	Default Value	Description
0x0804	HVCON	1	RW	N/A	High voltage interface control MMR. See the High Voltage Interface Control Register section and Table 71 and Table 72.
0x080C	HVDAT	2	RW	N/A	High voltage interface data MMR. See the High Voltage Data Register section and Table 73.

Table 30. Flash/EE Base Address = 0xFFFF0E00

Address	Name	Byte	Access Type	Default Value	Description
0x0E00	FEE0STA	1	R	0x20	Flash/EE status MMR.
0x0E04	FEE0MOD	1	RW	0x00	Flash/EE control MMR.
0x0E08	FEE0CON	1	RW	0x07	Flash/EE control MMR. See Table 13.
0x0E0C	FEE0DAT	2	RW	0x0000	Flash/EE data MMR.
0x0E10	FEE0ADR	2	RW	Nonzero	Flash/EE address MMR.
0x0E18	FEE0SIG	3	R	0xFFFFFF	Flash/EE LFSR MMR.
0x0E1C	FEE0PRO	4	RW	0x00000000	Flash/EE protection MMR. See the Flash/EE Memory Security section and Table 16.
0x0E20	FEE0HID	4	RW	0xFFFFFFFF	Flash/EE protection MMR. See the Flash/EE Memory Security section and Table 16.
0x0E80	FEE1STA	1	R	0x20	Flash/EE status MMR.
0x0E84	FEE1MOD	1	RW	0x00	Flash/EE control MMR.
0x0E88	FEE1CON	1	RW	0x07	Flash/EE control MMR. See Table 13.
0x0E8C	FEE1DAT	2	RW	0x0000	Flash/EE data MMR.
0x0E90	FEE1ADR	2	RW	0x0000	Flash/EE address MMR.
0x0E98	FEE1SIG	3	R	0xFFFFFF	Flash/EE LFSR MMR.
0x0E9C	FEE1PRO	4	RW	0x00000000	Flash/EE protection MMR. See the Flash/EE Memory Security section and Table 17.
0x0EA0	FEE1HID	4	RW	0xFFFFFFFF	Flash/EE protection MMR. See the Flash/EE Memory Security section and Table 17.

ADC Configuration Register

Name: ADCCFG

Address: 0xFFFF051C

Default Value: 0x00

Access: Read/write

Function: This 8-bit ADC configuration MMR controls extended functionality related to the on-chip ADCs.

Table 42. ADCCFG MMR Bit Designations

Bit	Description
7	Analog ground switch enable. Set to 1 by user software to connect the external GND_SW pin (Pin 15) to an internal analog ground reference point. This bit can be used to connect and disconnect external circuits and components to ground under program control and, thereby, minimize dc current consumption when the external circuit or component is not used. This bit is used in conjunction with ADCMDE[6] to select a 20 kΩ resistor to ground. Cleared by user code to disconnect the external GND_SW pin.
6, 5	Current channel (32-bit) accumulator enable. 00 = accumulator disabled and reset to 0. The accumulator must be disabled for a full ADC conversion (ADCSTA[0] set twice) before the accumulator can be reenabled to ensure that the accumulator is reset. 01 = accumulator active. Positive current values are added to the accumulator total; the accumulator can overflow if allowed to run for >65,535 conversions. Negative current values are subtracted from the accumulator total; the accumulator is clamped to a minimum value of 0. 10 = accumulator active. Positive current values are added to the accumulator total; the accumulator can overflow if allowed to run for >65,535 conversions. The absolute values of negative current are subtracted from the accumulator total; the accumulator in this mode continues to accumulate negatively, below 0. 11 = not defined.
4, 3	Current channel ADC comparator enable. 00 = comparator disabled. 01 = comparator active, interrupt asserted if absolute value of I-ADC conversion result is |I| ≥ ADC0TH. 10 = comparator count mode active, interrupt asserted if absolute value of an I-ADC conversion result is |I| ≥ ADC0TH for the number of ADC0TCL conversions. A conversion value of |I| < ADC0TH resets the threshold counter value (ADC0THV) to 0. 11 = comparator count mode active, interrupt asserted if absolute value of an I-ADC conversion result is |I| ≥ ADC0TH for the number of ADC0TCL conversions. A conversion value of |I| < ADC0TH decrements the threshold counter value (ADC0THV) toward 0.
2	Current channel ADC overrange enable. Set by user to enable a coarse comparator on the current channel ADC. If the current reading is grossly (>30% approximate) overrange for the active gain setting, then the overrange bit in the ADCSTA MMR is set. The current must be outside this range for greater than 125 µs for the flag to be set. This feature should not be used in ADC low power mode. Cleared by user code to disable the overrange feature.
1	Not used. This bit is reserved for future functionality and should be written as 0 by user code.
0	Current channel ADC, result counter enable. Set by user to enable the result count mode. In this mode, an I-ADC interrupt is generated only when ADC0RCV = ADC0RCL. This allows the I-ADC to continuously monitor current but only interrupt the MCU core after a defined number of conversions. The voltage/temperature ADC also continues to convert if enabled, but again, only the last conversion result is available (intermediate V-/T-ADC conversion results are not stored) when the ADC counter interrupt occurs.

The offset coefficient is read from the ADC0OF calibration register and is a 16-bit, twos complement number. The range of this number, in terms of the signal chain, is effectively ±1. Therefore, 1 LSB of the ADC0OF register is not the same as 1 LSB of the ADC0DAT register.

A positive value of ADC0OF indicates that when offset is subtracted from the output of the filter, a negative value is added. The nominal value of this register is 0x0000, indicating zero offset is to be removed. The actual offset of the ADC can vary slightly from part to part and at different PGA gains. The offset within the ADC is minimized if the chopping mode is enabled (that is, ADCFLT[15] = 1).

The gain coefficient is read from the ADC0GN register and is a unitless scaling factor. The 16-bit value in this register is divided by 16,384 and then multiplied by the offset-corrected value. The nominal value of this register equals 0x5555, corresponding to a multiplication factor of 1.3333, and scales the nominal ±0.75 signal to produce a full-scale output signal of ±1. The resulting output signal is checked for overflow/underflow and converted to twos complement or unipolar mode before being output to the data register.

The actual gain and the required scaling coefficient for zero gain error vary slightly from part to part at different PGA settings in normal and low power modes. The value downloaded into ADC0GN during a power-on reset represents the scaling factor for a PGA gain of 1. If a different PGA setting is used, however, some gain error may be present. To correct this error, overwrite the calibration coefficients via user code or perform an ADC calibration.

The simplified ADC transfer function can be described as

$$ADC_{OUT} = \left[\frac{V_{IN} \times PGA}{V_{REF}} - ADCxOF\right] \times \frac{ADCxGN}{ADCxGN_{NOM}}$$

where the equation is valid for the voltage/temperature channel ADC.

For the current channel ADC,

$$ADC_{OUT} = \left[\frac{V_{IN} \times PGA}{V_{REF}} - K \times ADCxOF\right] \times \frac{ADCxGN}{ADCxGN_{NOM}}$$

where *K* is dependent on the PGA gain setting and ADC mode.

Normal Mode

In normal mode, K = 1 for PGA gains of 1, 4, 8, 16, 32, and 64; K = 2 for PGA gains of 2 and 128; K = 4 for a PGA gain of 256; and K = 8 for a PGA gain of 512.

Low Power Mode

In low power mode, K = 32 for a PGA gain of 128. In addition, if the REG_AVDD/2 reference is used, the K factor doubles.

Low Power Plus Mode

In low power plus mode, K = 8 for a PGA gain of 512. In addition, if the REG_AVDD/2 reference is used, the K factor doubles.

ADC DIAGNOSTICS

The ADuC7036 features a diagnostic capability and open-circuit detection on both ADCs.

Current ADC Diagnostics

The ADuC7036 features the capability to detect open-circuit conditions on the current channel inputs. This is accomplished using the two current sources on IIN+ and IIN−, which are controlled via ADC0CON[14:13].

Note that the IIN+ and IIN− current sources have a tolerance of ±30%. Therefore, a PGA gain ≥ 2 (ADC0CON[3:0] ≥ 0001) must be used when current sources are enabled.

Temperature ADC Diagnostics

The ADuC7036 features the capability to detect open-circuit conditions on the temperature channel inputs. This is accomplished using the two current sources on VTEMP and GND_SW, which are controlled via ADC1CON[14:13].

Voltage ADC Diagnostics

The ADuC7036 features the capability to detect open-circuit conditions on the voltage channel input. This is accomplished using the current source on the voltage attenuator, controlled by the high voltage register HVCFG1[7].

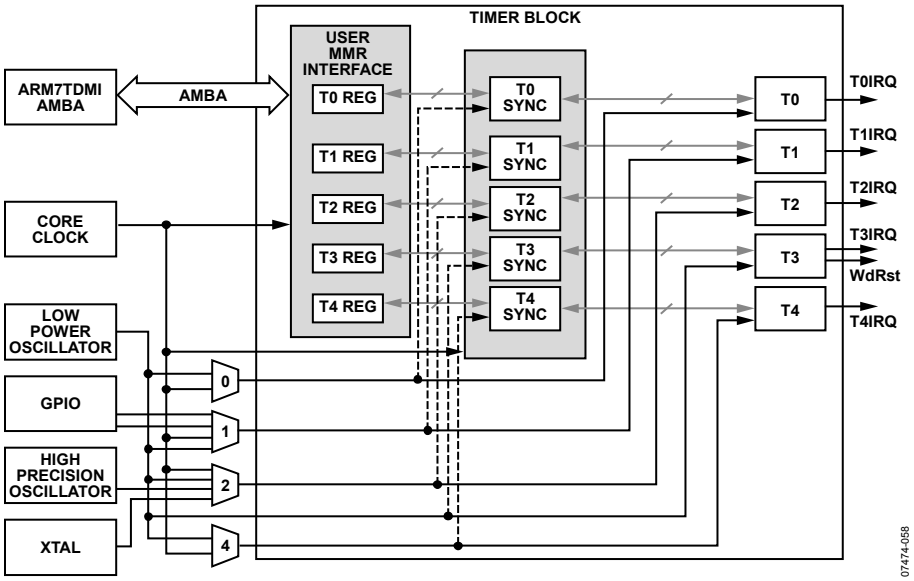

Figure 33. Timer Block Diagram

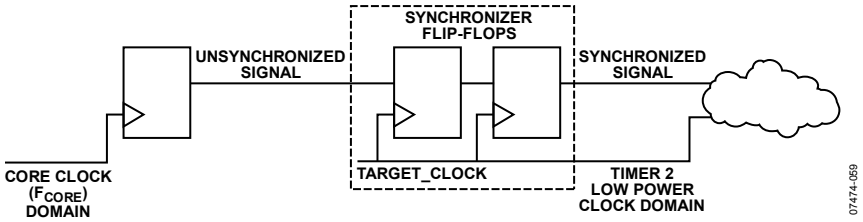

Figure 34. Synchronizer for Signals Crossing Clock Domains

As shown in Figure 33, the MMR logic and core timer logic reside in separate and asynchronous clock domains. Any data coming from the MMR core clock domain and being passed to the internal timer domain must be synchronized to the internal timer clock omain to ensure it is latched correctly into the core timer clock domain. This is achieved by using two flip-flops as shown in Figure 34 to not only synchronize but also to double buffer the data and thereby ensuring data integrity in the timer clock domain.

As a result of the synchronization block, while timer control data is latched almost immediately (with the fast, core clock) in the MMR clock domain, this data in turn will not reach the core timer logic for at least two periods of the selected internal timer domain clock.

PROGRAMMING THE TIMERS

Understanding synchronization across timer domains also requires that the user code carefully programs the timers when stopping or starting them. The recommended code controls the timer block when stopping and starting the timers and when using different clock domains. This can critical, especially if timers are enabled to generate an IRQ or FIQ exception; Timer2 is used as an example.

Halting Timer2

When halting Timer2, it is recommended that the IRQEN bit for Timer2 be masked (using IRQCLR). This prevents unwanted IRQs from generating an interrupt in the MCU before the T2CON control bits have been latched in the Timer2 internal logic.

```
IRQCLR = WAKEUP_TIMER_BIT;     //Masking interrupts
T2CON=0x00;                    //Halting the timer
```

TIMER1—GENERAL-PURPOSE TIMER

Timer1 is a general-purpose, 32-bit up/down counter with a programmable prescaler. The prescaler source can be the low power 32.768 kHz oscillator, the core clock, or from one of two external GPIOs. This source can be scaled by a factor of 1, 16, 256, or 32,768. When the core is operating at 20.48 MHz and at CD = 0 with a prescaler of 1 (ignoring the external GPIOs), a minimum resolution of 48.83 ns results.

The counter can be formatted as a standard 32-bit value or as time expressed as hours:minutes:seconds:hundredths.

Timer1 has a capture register (T1CAP) that is triggered by the initial assertion of a selected IRQ source. When the capture register is triggered, the current timer value is copied to T1CAP, and the timer continues to run. This feature can be used to determine the assertion of an event with increased accuracy.

The Timer1 interface consists of five MMRS: T1LD, T1VAL, T1CAP, T1CLRI, and T1CON. T1LD, T1VAL, and T1CAP are 32-bit registers that hold 32-bit unsigned integers. T1VAL and T1CAP are read only. T1CLRI is an 8-bit register. Writing any value to this register clears the Timer1 interrupt. T1CON is a configuration MMR and is described in Table 54.

Timer1 features a postscaler that allows the user to count the number of Timer1 timeouts between 1 and 256. To activate the postscaler, the user sets Bit 23 and writes the desired number to count into Bits[24:31] of T1CON. When that number of timeouts is reached, Timer1 generates an interrupt if T1CON[18] is set.

Note that if the part is in a low power mode and Timer1 is clocked from the GPIO or low power oscillator source, then Timer1 continues to operate.

Timer1 reloads the value from T1LD when Timer1 overflows.

Timer1 Load Register

Name: T1LD

Address: 0xFFFF0320

Default Value: After a reset, this register contains the upper half of the assembly lot ID (0x00000000).

Access: Read/write

Function: This 32-bit register holds the 32-bit value that is loaded into the counter.

Timer1 Clear Register

Name: T1CLRI

Address: 0xFFFF032C

Access: Write only

Function: This 8-bit, write only MMR is written (with any value) by user code to clear the interrupt.

Timer1 Value Register

Name: T1VAL

Address: 0xFFFF0324

Default Value: 0xFFFFFFFF

Access: Read only

Function: This 32-bit register holds the current value of Timer1.

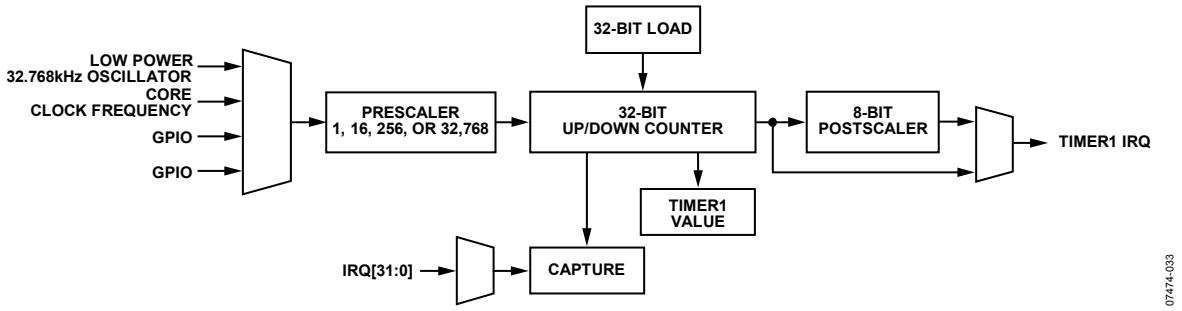

Figure 36. Timer1 Block Diagram

GPIO Port0 Data Register

Name: GP0DAT

Address: 0xFFFF0D20

Default Value: 0x000000XX

Access: Read/write

Function: This 32-bit MMR configures the direction of the GPIO pins assigned to Port0 (see Table 58). This register also sets the output value for GPIO pins configured as outputs and reads the status of GPIO pins configured as inputs.

Table 62. GP0DAT MMR Bit Designations

Bit	Description
31 to 29	Reserved. These bits are reserved and should be written as 0 by user code.
28	Port 0.4 direction select bit. Set to 1 by user code to configure the GPIO pin assigned to Port 0.4 as an output. Cleared to 0 by user code to configure the GPIO pin assigned to Port 0.4 as an input.
27	Port 0.3 direction select bit. Set to 1 by user code to configure the GPIO pin assigned to Port 0.3 as an output. Cleared to 0 by user code to configure the GPIO pin assigned to Port 0.3 as an input.
26	Port 0.2 direction select bit. Set to 1 by user code to configure the GPIO pin assigned to Port 0.2 as an output. Cleared to 0 by user code to configure the GPIO pin assigned to Port 0.2 as an input.
25	Port 0.1 direction select bit. Set to 1 by user code to configure the GPIO pin assigned to Port 0.1 as an output. Cleared to 0 by user code to configure the GPIO pin assigned to Port 0.1 as an input.
24	Port 0.0 direction select bit. Set to 1 by user code to configure the GPIO pin assigned to Port 0.0 as an output. Cleared to 0 by user code to configure the GPIO pin assigned to Port 0.0 as an input.
23 to 21	Reserved. These bits are reserved and should be written as 0 by user code.
20	Port 0.4 data output. The value written to this bit appears directly on the GPIO pin assigned to Port 0.4.
19	Port 0.3 data output. The value written to this bit appears directly on the GPIO pin assigned to Port 0.3.
18	Port 0.2 data output. The value written to this bit appears directly on the GPIO pin assigned to Port 0.2.
17	Port 0.1 data output. The value written to this bit appears directly on the GPIO pin assigned to Port 0.1.
16	Port 0.0 data output. The value written to this bit appears directly on the GPIO pin assigned to Port 0.0.
15 to 5	Reserved. These bits are reserved and should be written as 0 by user code.
4	Port 0.4 data input. This bit is a read only bit that reflects the current status of the GPIO pin assigned to Port 0.4. User code should write 0 to this bit.
3	Port 0.3 data input. This bit is a read only bit that reflects the current status of the GPIO pin assigned to Port 0.3. User code should write 0 to this bit.
2	Port 0.2 data input. This bit is a read only bit that reflects the current status of the GPIO pin assigned to Port 0.2. User code should write 0 to this bit.
1	Port 0.1 data input. This bit is a read only bit that reflects the current status of the GPIO pin assigned to Port 0.1. User code should write 0 to this bit.
0	Port 0.0 data input. This bit is a read only bit that reflects the current status of the GPIO pin assigned to Port 0.0. User code should write 0 to this bit.

GPIO Port1 Data Register

Name: GP1DAT

Address: 0xFFFF0D30

Default Value: 0x000000XX

Access: Read/write

Function: This 32-bit MMR configures the direction of the GPIO pins assigned to Port1 (see Table 58). This register also sets the output value for GPIO pins configured as outputs and reads the status of GPIO pins configured as inputs.

Table 63. GP1DAT MMR Bit Designations

Bit	Description
31 to 26	Reserved. These bits are reserved and should be written as 0 by user code.
25	Port 1.1 direction select bit. Set to 1 by user code to configure the GPIO pin assigned to Port 1.1 as an output. Cleared to 0 by user code to configure the GPIO pin assigned to Port 1.1 as an input.
24	Port 1.0 direction select bit. Set to 1 by user code to configure the GPIO pin assigned to Port 1.0 as an output. Cleared to 0 by user code to configure the GPIO pin assigned to Port 1.0 as an input.
23 to 18	Reserved. These bits are reserved and should be written as 0 by user code.
17	Port 1.1 data output. The value written to this bit appears directly on the GPIO pin assigned to Port 1.1.
16	Port 1.0 data output. The value written to this bit appears directly on the GPIO pin assigned to Port 1.0.
15 to 2	Reserved. These bits are reserved and should be written as 0 by user code.
1	Port 1.1 data input. This bit is a read only bit that reflects the current status of the GPIO pin assigned to Port 1.1. User code should write 0 to this bit.
0	Port 1.0 data input. This bit is a read only bit that reflects the current status of the GPIO pin assigned to Port 1.0. User code should write 0 to this bit.

GPIO Port0 Set Register

Name: GP0SET

Address: 0xFFFF0D24

Access: Write only

Function: This 32-bit MMR allows user code to individually bit-address external GPIO pins to set them high only. User code can accomplish this using the GP0SET MMR without having to modify or maintain the status of the GPIO pins (as user code requires when using GP0DAT).

Table 65. GP0SET MMR Bit Designations

Bit	Description
31 to 21	Reserved. These bits are reserved and should be written as 0 by user code.
20	Port 0.4 set bit. Set to 1 by user code to set the external GPIO_4 pin high. Clearing this bit to 0 via user software has no effect on the external GPIO_4 pin.
19	Port 0.3 set bit. Set to 1 by user code to set the external GPIO_3 pin high. Clearing this bit to 0 via user software has no effect on the external GPIO_3 pin.
18	Port 0.2 set bit. Set to 1 by user code to set the external GPIO_2 pin high. Clearing this bit to 0 via user software has no effect on the external GPIO_2 pin.
17	Port 0.1 set bit. Set to 1 by user code to set the external GPIO_1 pin high. Clearing this bit to 0 via user software has no effect on the external GPIO_1 pin.
16	Port 0.0 set bit. Set to 1 by user code to set the external GPIO_0 pin high. Clearing this bit to 0 via user software has no effect on the external GPIO_0 pin.
15 to 0	Reserved. These bits are reserved and should be written as 0 by user code.

GPIO Port1 Set Register

Name: GP1SET

Address: 0xFFFF0D34

Access: Write only

Function: This 32-bit MMR allows user code to individually bit-address external GPIO pins to set them high only. User code can accomplish this using the GP1SET MMR without having to modify or maintain the status of the GPIO pins (as user code requires when using GP1DAT).

Table 66. GP1SET MMR Bit Designations

Bit	Description
31 to 18	Reserved. These bits are reserved and should be written as 0 by user code.
17	Port 1.1 set bit. Set to 1 by user code to set the external GPIO_6 pin high. Clearing this bit to 0 via user software has no effect on the external GPIO_6 pin.
16	Port 1.0 set bit. Set to 1 by user code to set the external GPIO_5 pin high. Clearing this bit to 0 via user software has no effect on the external GPIO_5 pin.
15 to 0	Reserved. These bits are reserved and should be written as 0 by user code.

High Voltage Configuration1 Register

Name: HVCFG1

Address: Indirectly addressed via the HVCON high voltage interface

Default Value: 0x00

Access: Read/write

Function: This 8-bit register controls the function of high voltage circuits on the ADuC7036. This register is not an MMR and does not appear in the MMR memory map. It is accessed via the HVCON register interface. Data to be written to this register is loaded through HVDAT, and data is read back from this register using HVDAT.

Table 75. HVCFG1 Bit Designations

Bit	Description
7	Voltage attenuator diagnostic enable bit. Set to 1 to turn on a 1.29 µA current source that adds 170 mV differential voltage to the voltage channel measurement. Cleared to 0 to disable the voltage attenuator diagnostic.
6	High voltage temperature monitor. The high voltage temperature monitor is an uncalibrated temperature monitor located on chip, close to the high voltage circuits. This monitor is completely separate to the on-chip, precision temperature sensor (controlled via ADC1CON[7:6]) and allows user code to monitor die temperature change close to the hottest part of the ADuC7036 die. The monitor generates a typical output voltage of 600 mV at 25°C and has a negative temperature coefficient of typically −2.1 mV/°C. Set to 1 to enable the on-chip, high voltage temperature monitor. When enabled, this voltage output temperature monitor is routed directly to the voltage channel ADC. Cleared to 0 to disable the on-chip, high voltage temperature monitor.
5	Voltage channel short enable bit. Set to 1 to enable an internal short (at the attenuator, before the ADC input buffers) on the voltage channel ADC and to allow noise to be measured as a self-diagnostic test. Cleared to 0 to disable an internal short on the voltage channel.
4	WU and STI readback enable bit. Set to 1 to enable input capability on the external WU and STI pins. In this mode, a rising or falling edge transition on the WU and STI pins generates a high voltage interrupt. When this bit is set, the state of the WU and STI pins can be monitored via the HVMON register (HVMON[7] and HVMON[5]). Cleared to 0 to disable input capability on the external WU and STI pins.
3	High voltage I/O driver enable bit. Set to 1 to reenable high voltage I/O pins (LIN/BSD, STI, and WU) that have been disabled as a result of a short-circuit current event (the event must last longer than 20 µs for the LIN/BSD and STI pins and longer than 400 µs for the WU pin). This bit must also be set to 1 to reenable the WU and STI pins if they were disabled by a thermal event. It should be noted that this bit must be set to clear any pending interrupt generated by the short-circuit event (even if the event has passed) as well as reenabling the high voltage I/O pins. Cleared to 0 automatically.
2	Enable/disable short-circuit protection (LIN/BSD and STI). Set to 1 to enable passive short-circuit protection on the LIN pin. In this mode, a short-circuit event on the LIN/BSD pin generates a high voltage interrupt, IRQ3 (if enabled in IRQEN[16]), and asserts the appropriate status bit in HVSTA but does not disable the short-circuiting pin. Cleared to 0 to enable active short-circuit protection on the LIN/BSD pin. In this mode, during a short-circuit event, the LIN/BSD pin generates a high voltage interrupt (IRQ3), asserts HVSTA[16], and automatically disables the short-circuiting pin. When disabled, the I/O pin can only be reenabled by writing to HVCFG1[3].
1	WU pin timeout (monoflop) counter enable/disable. Set to disable the WU I/O timeout counter. Cleared to enable a timeout counter that automatically deasserts the WU pin 1.3 sec after user code has asserted the WU pin via HVCFG0[4].
0	WU open-circuit diagnostic enable. Set to enable an internal WU I/O diagnostic pull-up resistor to the VDD pin, thus allowing detection of an open-circuit condition on the WU pin. Cleared to disable an internal WU I/O diagnostic pull-up resistor.

Example LIN Hardware Synchronization Routine

Using the following C-source code LIN initialization routine, LHSVAL1 begins to count on the first falling edge received on the LIN bus. If LHSVAL1 exceeds the value written to LHSVAL1, in this case 0x3F, a break compare interrupt is generated.

On the next falling edge, LHSVAL0 begins counting. LHSVAL0 monitors the number of falling edges and compares it to the value written to LHSCON1[7:4]. In this example, the number of edges to monitor is six falling edges of the LIN frame, or the five falling edges of the sync byte. When this number of falling edges is received, a stop condition interrupt is generated. It is at this point that the UART is configured to receive the protected identifier.

The UART must be gated through LHSCON0[8] before the LIN bus returns high. If the LIN bus returns high when UART is not gated, UART communication errors may occur. This process is shown in detail in Figure 52. Example code to ensure the success of this process follows Figure 49.

```
void LIN_INIT(void )
{
       char HVstatus;
       GP2CON = 0x110000;  // Enable LHS on GPIO pins

       LHSCON0 = 0x1;      // Reset LHS interface

       do{
              HVDAT = 0x02; // Enable normal LIN Tx mode
              HVCON = 0x08; // Write to Config0
              do{
                     HVstatus = HVCON;
              }
              while(HVstatus & 0x1); // Wait until command is finished
       }
       while (!(HVstatus & 0x4)); // Transmit command is correct

       while((LHSSTA & 0x20) == 0 )
       {                   // Wait until the LHS hardware is reset
       }

       LHSCON1 = 0x062;    // Sets stop edge as the fifth falling edge
                           // and the start edge as the first falling
                           // edge in the sync byte
       LHSCON0 = 0x0114;   // Gates UART Rx line, ensuring no interference
                           // from the LIN into the UART
                           // Selects the stop condition as a falling edge
                           // Enables generation of an interrupt on the
                           // stop condition
                           // Enables the interface
       LHSVAL1 = 0x03F;    // Sets number of 131 kHz periods to generate a break interrupt
                           // 0x3F / 131 kHz ~ 480 µs, which is just over 9.5 Tbits
```

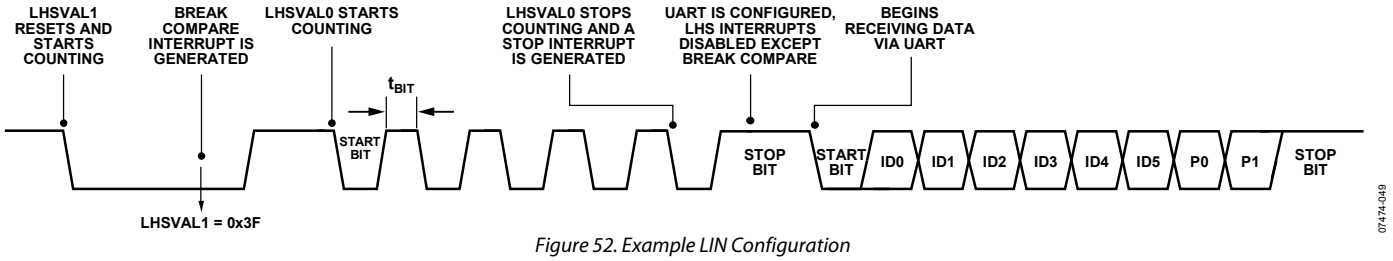

Figure 52. Example LIN Configuration

```
       while((GP2DAT & 0x10 ) == 0 )
       {}                  // Wait until LIN Bus returns high
       LHSCON0 = 0x4;      // Enable LHS to detect Break Condition Ungate RX Line
                           // Disable all Interrupts except Break Compare Interrupt
       IRQEN = 0x800;      // Enable UART Interrupt
                           // The UART is now configured and ready to be used for LIN
```

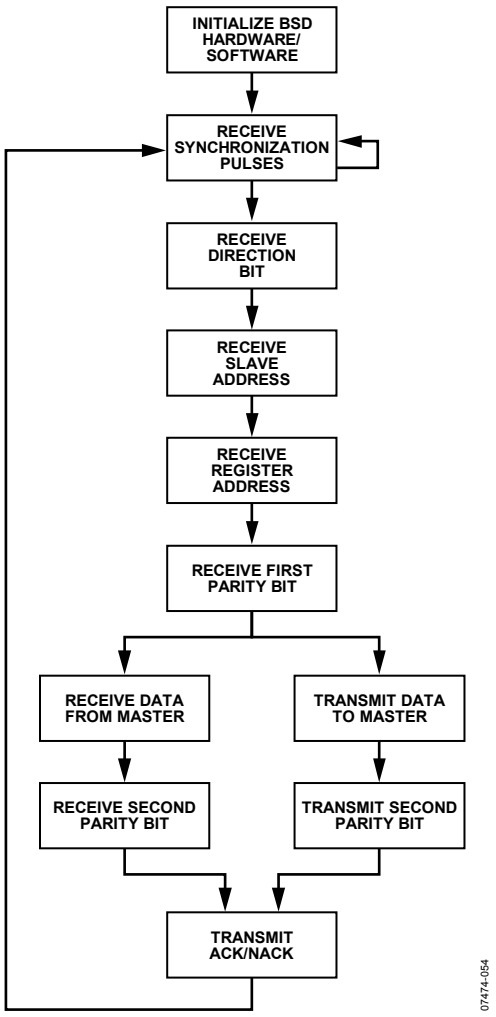

Figure 57. BSD Slave Node State Machine

BSD DATA RECEPTION

To receive data, the LIN/BSD peripheral must first be configured in BSD mode where LHSCON0[6] = 1. In this mode, LHSCON0[8] should be set to ensure that the LHS break timer (see the LIN Hardware Synchronization Break Timer1 Register section) generates an interrupt on the rising edge of the BSD bus.

The LHS break timer is cleared and starts counting on the falling edge of the BSD bus; the timer is subsequently stopped and generates an interrupt on the rising edge of the BSD bus. Given that the LHS break timer is clocked by the low power 131 kHz oscillator, the value in LHSVAL1 can be interpreted by user code to determine if the received data bit is a BSD sync pulse, 0, or 1.

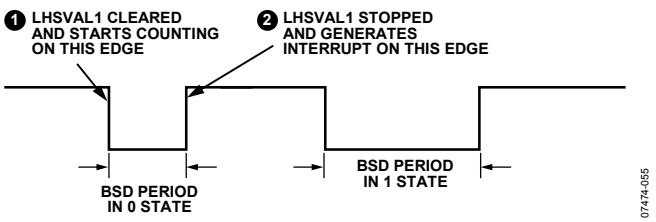

Figure 58. Master Transmit, Slave Read

BSD DATA TRANSMISSION

User code forces the GPIO_12 signal low for a specified time to transmit data in BSD mode. In addition, user code uses the sync timer (LHSVAL0), the LHS sync capture register (LHSCAP), and the LHS sync compare register (LHSCMP) to determine the length of time that the BSD bus should be held low for bit transmissions in the 0 or 1 state.

As described in the BSD Example Pulse Widths section, even when the slave is transmitting, the master always starts the bit transmission period by pulling the BSD bus low. If BSD mode is selected (LHSCON0[6] = 1), the LIN sync timer value is captured in LHSCAP on every falling edge of the BSD bus. The LIN sync timer runs continuously in BSD mode.

Then, user code can immediately force GPIO_12 low and read the captured timer value from LHSCAP. Next, the user can calculate how many clock periods (with a 5 MHz clock) should elapse before the GPIO_12 is driven high for a pulse width in the 0 or 1 state. The calcaulated number can be added to the LHSCAP value and written into the LHSCMP register. If LHSCON0[5] is set, the sync timer, which continues to count (being clocked by a 5 MHz clock), eventually equals the LHSCMP value and generates an LHS compare interrupt (LHSSTA[3]).

The response to this interrupt should be to force the GPIO_12 signal (and, therefore, the BSD bus) high. The software control of the GPIO_12 signal, along with the correct use of the LIN synchronization timers, ensures that valid pulse widths in the 0 and 1 states can be transmitted from the ADuC7036, as shown in Figure 59. Again, care must be taken if switching from BSD write mode to BSD read mode, as described in Table 93 (see the LHSCON0[8] bit.)

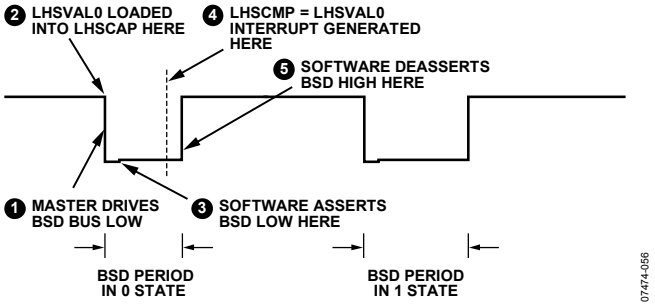

Figure 59. Master Read, Slave Transmit

WAKE-UP FROM BSD INTERFACE

The MCU core can be awakened from power-down via the BSD physical interface. Before entering power-down mode, user code should enable the start condition interrupt (LHSCON0[3]). When this interrupt is enabled, a high-to-low transition on the LIN/BSD pin generates an interrupt event and wakes up the MCU core.

System Identification FEE0ADR

Name: FEE0ADR

Address: 0xFFFF0E10

Default Value: Nonzero

Access: Read/write

Function: This 16-bit register dictates the address upon which any Flash/EE command executed via FEE0CON acts.

Note that this MMR is also used to identify the ADuC7036 family member and prerelease silicon revision.

Table 100. FEE0ADR System Identification MMR Bit Designations

Bit	Description
15 to 4	Reserved
3 to 0	ADuC703x family ID 0x2 = ADuC7032 0x3 = ADuC7033 0x4 = ADuC7034 0x6 = ADuC7036BCPZ and ADuC7036CCPZ 0x100 = ADuC7036DCPZ Others = reserved for future use