车用集成精密电池传感器

ADuC7039

特性

高精度ADC
双通道、同步采样、16位Σ-Δ型ADC
可编程ADC吞吐量：10 Hz至1 kHz
片内5 ppm/°C基准电压源
电流通道
　全差分、缓冲输入
　可编程增益
　ADC输入范围：−200 mV至+300 mV
　数字比较器，内置电流累加器功能
电压通道
　缓冲、片内衰减器，适用于12 V电池输入
温度通道
　外部和片内温度传感器方案
微控制器
　16位/32位RISC架构ARM7TDMI-S内核
　20.48 MHz PLL
　片内精密振荡器
JTAG端口支持代码下载和调试
存储器
　64 kB Flash/EE存储器选项，4 kB SRAM
　Flash/EE耐久性：10,000个周期，数据保持时间：20年
　通过JTAG和LIN在线下载
片内外设
　SAEJ2602/LIN 2.1兼容从机
　SPI
　GPIO端口
　1个通用定时器
　唤醒和看门狗定时器
　片内上电复位
电源
　12 V电池电源直接供电
　功耗：7.5 mA (10 MHz)
　低功耗监控模式
封装和温度范围
　32引脚6 mm × 6 mm LFCSP封装
　额定工作温度范围：−40°C至+115°C
通过汽车应用认证

应用

汽车系统电池检测/管理

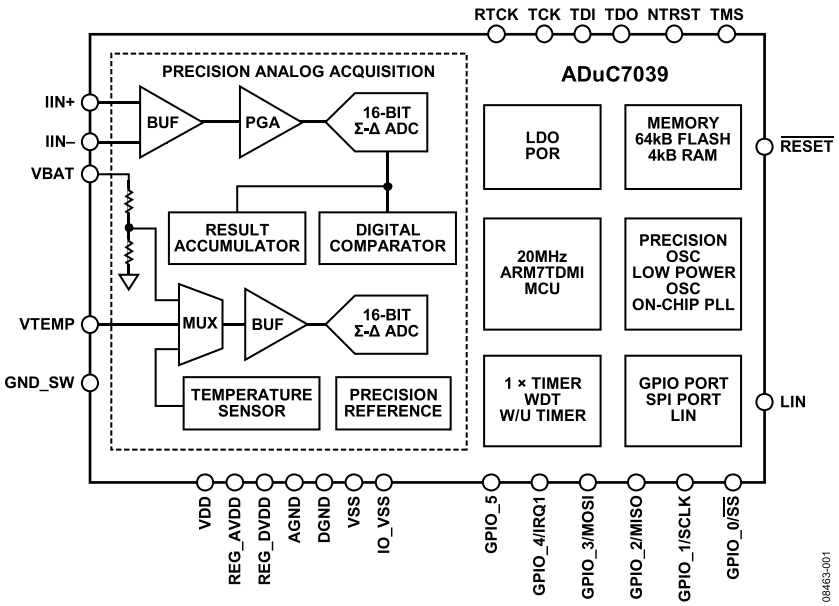


图1.

Rev. D Document Feedback

One Technology Way, P.O. Box 9106, Norwood, MA 02062-9106, U.S.A.
Tel: 781.329.4700

Technical Support www.analog.com

目录

修订历史

2013年3月—修订版C至修订版D

“固定10 MHz频率”改为“默认10 MHz频率”(通篇) 1
更改“特性”部分和图1 1
更改表1 5
更改表4 12
更改“系统内核校验和”部分 24
更改图9 25
更改“ADCxOF默认值” 41
更改图20 49
更改“序列示例”部分 50
重新组织“LIN振荡器校准”部分的布局 56
更改图22和表44 59
更改表59 80
更改“高压状态寄存器功能描述” 81
更改图31 83
更改“器件标识”部分，增加表64 88
更新外形尺寸 91
更改“订购指南” 91
添加“汽车应用级产品”部分 91

2012年1月—修订版B至修订版C

更改“LVF电平测试条件/注释” 5
引脚25从GPIO_4改为GPIO_4/IRQ1 10
更改“Flash/EE存储器安全性”部分 22
更改图9 25
更改图22和表44 59
更改表49中的GPIO_4端口信号与功能 69
更改图33 90
更新外形尺寸 91

2010年8月—修订版A至修订版B

更改“快速温度转换模式”部分 46
更改表62中的位4描述(LINCON[6]参考) 86
更改“器件标识”部分 87

2010年6月—修订版0至修订版A

“特性”部分增加SAEJ2602 1
更改表2 9
更改表34中的位2和位1描述 39
更改“系统时钟”部分 49
增加“异步时钟域的定时器同步”部分、图23和图24 60
增加“定时器编程”部分 61
更改“推荐原理图”文本 89

2010年3月—修订版0：初始版

技术规格

电气规格

除非另有说明，V_{DD} = 3.5 V至18 V，V_{REF} = 1.2 V内部基准电压，f_{CORE} = 20.48 MHz(采用片内精密振荡器驱动)，且所有规格T_A = −40°C至+115°C。

表1.

参数	测试条件/注释	最小值	典型值	最大值	单位
ADC技术规格					
转换速率[1]	ADC正常工作模式	10		1000	Hz
	ADC低功耗模式，斩波开启	10		650	Hz
电流通道					
无失码[1]	对所有ADC更新速率和ADC模式有效	16			位
积分非线性(INL)[1, 2]			±10	±60	FSR的ppm
失调误差[1, 2, 3]	斩波关闭，外部短路，用户系统校准后，1 LSB = (36.6/增益) μV	−10	±3	+10	LSB
失调误差[1, 3]	斩波开启，外部短路，低功耗模式，MCU关断	+250	±50	−300	nV
失调误差[1, 3]	斩波开启，外部短路，用户系统校准后	0		3.0	μV
	测试条件：CD = 0，VDD = 18 V				
失调误差[1, 3]	斩波开启，外部短路，用户系统校准后		±0.5		μV
	测试条件：CD = 0，VDD = 4 V				
失调误差漂移[1, 2, 4]	斩波关闭，增益为8至64，正常模式		±0.03		LSB/°C
失调误差漂移[1, 4]	斩波关闭，对于ADC增益512有效		±30		nV/°C
失调误差漂移[1, 4]	斩波开启		±5		nV/°C
总增益误差[1, 3, 5, 6]	增益为4时工厂校准，正常模式	−0.5	±0.1	+0.5	%
	低功耗模式	−1	±0.2	+1	%
增益漂移[1, 7]			±3		ppm/°C
PGA增益不匹配误差			±0.1		%
输出噪声[1]	10 Hz更新速率，增益 = 512，斩波使能		100	150	nV rms
	1 kHz更新速率，增益 = 512，ADCFLT = 0x0007		0.6	0.9	μV rms
	1 kHz更新速率，增益 = 32，ADCFLT = 0x0007		0.8	1.2	μV rms
	1 kHz更新速率，增益 = 8，ADCFLT = 0x8101		2.1	4.1	μV rms
	1 kHz更新速率，增益 = 8，ADCFLT = 0x0007		1.6	2.4	μV rms
	1 kHz更新速率，增益 = 4，ADCFLT = 0x0007		2.6	3.9	μV rms
	ADC低功耗模式，250Hz更新速率，斩波使能，增益 = 512		0.6	0.9	μV rms
电压通道[8]					
无失码[1]	在所有ADC更新速率下有效	16			位
积分非线性(INL)[1]			±10	±60	FSR的ppm
失调误差[1, 3]	斩波关闭，1 LSB = 439.5 μV，两点校准后	−10	±1	+10	LSB
失调误差[1, 3]	斩波开启，两点校准后	−1	±0.3	+1	LSB
失调误差漂移[4]	斩波关闭		±0.03		LSB/°C
总增益误差[1, 3, 5, 6]	包括电阻不匹配	−0.25	±0.06	+0.25	%
总增益误差[1, 3, 5, 6]	温度范围 = −25°C至+65°C	−0.15	±0.03	+0.15	%
增益漂移[1, 7]	包括电阻不匹配漂移		±3		ppm/°C
输出噪声[1, 9]	10 Hz更新速率，斩波开启		60	90	μV rms
	1 kHz更新速率，ADCFLT = 0x0007		180	270	μV rms
温度通道					
无失码[1]	在所有ADC更新速率下有效	16			位
积分非线性(INL)[1]			±10	±60	FSR的ppm
失调误差[3, 10]	斩波关闭，1 LSB = 19.84 μV（单极性模式下）	−10	±3	+10	LSB
失调误差[1, 3]	斩波开启	−5	+1	+5	LSB
失调误差漂移[1]	斩波关闭		0.03		LSB/°C
总增益误差[1, 3, 10]	VREF = (REG_AVDD, GND_SW)/2	−0.25	±0.06	+0.25	%
增益漂移[1]			3		ppm/°C
输出噪声[1]	1 kHz更新速率		7.5	11.25	μV rms

参数	测试条件/注释	最小值	典型值	最大值	单位
ADC模拟输入规格					
电流通道					
绝对输入电压范围[1]	同时适用于IIN+和IIN−	−200		+300	mV
输入电压范围[11, 12]					
	增益 = 4		±300		mV
	增益 = 8		±150		mV
	增益 = 32		±37.5		mV
	增益 = 512		±2.3		mV
输入漏电流[1]		−3		+3	nA
输入失调电流[1, 13]			0.5	1.5	nA
电压通道					
绝对输入电压范围[1]	电压ADC规格在此范围内有效	4		18	V
输入电压范围[1]			0 to 28.8		V
VBAT输入电流	VBAT = 18 V	3	5.5	8	µA
温度通道	V_{REF} = (REG_AVDD, GND_SW)/2				
绝对输入电压范围[14]		100		1300	mV
输入电压范围[1]			0至V_{REF}		V
VTEMP输入电流[1]			2.5	100	nA
基准电压源					
内部VREF			1.2		V
上电时间[1]			0.5		ms
初始精度[1]	在T_A = 25°C时测定	−0.15		+0.15	%
温度系数[1, 15]		−20	±5	+20	ppm/°C
长期稳定性[16]			100		ppm/1000小时
ADC诊断					
VREF/136精度[1]	任意增益设置，完成选定增益的自校准后	8.4		9.4	mV
电压衰减器电流源精度[1]	当电流源开启时差分电压在衰减器上递增	2.3		3.6	V
电阻衰减器[1]					
分频比			24		
电阻不匹配漂移	包括在电压通道总增益误差中		±3		ppm/°C
ADC接地开关					
接地电阻		15	20	30	kΩ
温度传感器[17]	未校准				
精度	MCU处于关断或待机模式，T_A= −40°C至+85°C	−10	±3	+10	°C
	MCU处于关断或待机模式，T_A= −20°C至+60°C	−8	±2	+8	°C
上电复位(POR)[1]					
POR触发电平	参考VDD引脚的电压	2.85	3.0	3.15	V
POR迟滞			300		mV
POR复位超时			20		ms
低压标志(LVF)					
LVF电平	参考REG_DVDD引脚的电压	1.9	2.1	2.3	V
看门狗定时器(WDT)					
超时周期[1]	32 kHz时钟，256预分频	0.008		524	秒
超时步幅			7.8		ms
Flash/EE存储器[1]					
耐久性[18]		10,000			周期
数据保持时间[19]		20			年
数字输入[1]	除NTRST外的所有数字输入				
输入漏电流	输入高电平 = REG_DVDD		±1	±10	µA
输入上拉电流[1]	输入低电平 = 0 V	10	20	80	µA
输入电容[1]			10		pF
输入漏电流	仅限NTRST：输入低电平 = 0 V		±1	±10	µA
输入下拉电流	仅限NTRST：输入高电平= REG_DVDD	30	55	100	µA
逻辑输入[1]	所有逻辑输入				
输入低电压(VINL)				0.4	V
输入高电压(VINH)		2.0			V

参数	测试条件/注释	最小值	典型值	最大值	单位
片内振荡器					
低功耗振荡器			128		kHz
精度	标称电源电压和室温下进行用户校准后；包括1000小时寿命测试的漂移数据	−3		+3	%
精密振荡器			128		kHz
精度	运行时校准后	−1		+1	%
MCU时钟速率[1]	默认设置		10.24		MHz
MCU启动时间[1]					
上电时	包括内核上电执行时间		25		ms
复位事件后	包括内核上电执行时间		5		ms
从MCU关断起	振荡器运行		2		ms
内部PLL锁定时间			1		ms
LIN通用I/O					
波特率		1000		20,000	位/秒
V_{DD}	LIN接口工作电压范围	7		18	V
输入电容[1]			5.5		pF
LIN比较器响应时间[1]	使用22 Ω电阻		38	90	µs
LIN直流参数					
$I_{LIN\ DOM\ MAX}$	LIN总线处于主动状态时驱动器电流限值，VBAT = VBAT(最大值)	40		200	mA
$I_{LIN\_PAS\_REC}$[1]	驱动器关闭，7.0 V < V_{BUS} < 18 V, VDD = V_{LIN} − 0.7 V			20	µA
$I_{LIN\_PAS\_DOM}$[1]	输入漏电流，V_{LIN} = 0 V	−1			mA
$I_{LIN\_NO\_GND}$[1, 20]	控制单元接地断开，GND = V_{DD}，0 V < V_{LIN} < 18 V，VBAT = 12 V	−1		+1	mA
$I_{BUS\_NO\_BAT}$[1]	VBAT断开，V_{DD} = GND，0 V < V_{BUS} < 18 V			100	µA
V_{LIN_DOM}[1]	LIN接收器主动状态，V_{DD} > 7.0 V			0.4 V_{DD}	V
V_{LIN_REC}[1]	LIN接收器被动状态，V_{DD} > 7.0 V	0.6 V_{DD}			V
V_{LIN_CNT}[1]	LIN接收器中心电压，V_{DD} > 7.0 V	0.475 V_{DD}	0.5 V_{DD}	0.525 V_{DD}	V
V_{HYS}[1]	LIN接收器迟滞电压			0.175 V_{DD}	V
$V_{LIN\_DOM\_DRV\_LOSUP}$[1]	LIN主动输出电压，V_{DD} = 7.0 V				
R_L 500 Ω				1.2	V
R_L 1000 Ω		0.6			V
$V_{LIN\_DOM\_DRV\_HISUP}$[1]	LIN主动输出电压，V_{DD} = 18 V				
R_L 500 Ω				2	V
R_L 1000 Ω		0.8			V
$V_{LIN\_RECESSIVE}$[1]	LIN被动输出电压	0.8 V_{DD}			V
VBAT偏移[20]		0		0.115 V_{DD}	V
GND偏移[20]		0		0.115 V_{DD}	V
R_{SLAVE}	从机端接电阻	20	30	47	kΩ
$V_{SERIAL\ DIODE}$[20]	串行二极管处的压降，D_{Ser_Int}	0.4	0.7	1	V
LIN交流参数[1]	总线负载条件(CBUS||RBUS)：1 nF||1 kΩ; 6.8 nF||660 Ω; 10 nF||500 Ω				
D1	占空比1 $TH_{REC(MAX)}$ = 0.744 × VBAT $TH_{DOM(MAX)}$ = 0.581 × VBAT V_{SUP} = 7.0 V … 18 V; t_{BIT} = 50 µs D1 = $t_{BUS\_REC(MIN)}$/(2 × t_{BIT})	0.396			
D2	占空比2 $TH_{REC(MIN)}$ = 0.284 × VBAT $TH_{DOM(MIN)}$ = 0.422 × VBAT V_{SUP} = 7.0 V … 18 V; t_{BIT} = 50 µs D2 = $t_{BUS\_REC(MAX)}$/(2 × t_{BIT})			0.581	
D3[1, 20]	$TH_{REC(MAX)}$ = 0.778 × VBAT $TH_{DOM(MAX)}$ = 0.616 × VBAT V_{DD} = 7.0 V ... 18 V t_{BIT} = 96 µs D3 = $t_{BUS\_REC(MIN)}$/(2 × t_{BIT})	0.417			
D4[1, 20]	$TH_{REC(min)}$ = 0.389 × VBAT $TH_{DOM(min)}$ = 0.251 × VBAT			0.590	

参数	测试条件/注释	最小值	典型值	最大值	单位
	V_{DD} = 7.0 V … 18 V t_{BIT} = 96 µs D4 = $t_{BUS\_REC(MAX)}/(2 \times t_{BIT})$				
t_{RX_PDR}[1, 20]	接收器传播延迟			6	µs
t_{RX_SYM}[1, 20]	接收器传播延迟上升沿相对于下降沿的对称性(t_{RX_SYM} = $t_{RF\_PDR} - t_{RX\_PDF}$)	−2		+2	µs
封装热规格					
热阻(θ_{JA})[21]	32引脚CSP，堆叠式芯片		32		°C/W
电源要求[1]					
电源电压					
VDD(电池电源)		3.5		18	V
REG_DVDD, REG_AVDD[22]		2.45	2.6	2.75	V
功耗					
I_{DD}(MCU正常模式)[23]	ADC关闭(20.48 MHz)		10	20	mA
	ADC关闭(10.24 MHz)		7.5	16	mA
I_{DD}(MCU关断)[1]	ADC低功耗模式，在T_A = −10°C至+40°C的环境温度范围内测定，ADC连续转换		750	1000	µA
I_{DD}(MCU关断)[1]	精密振荡器关闭，ADC关闭 平均电流，在唤醒和看门狗定时器从低功耗振荡器获取时钟信号时测定(−40°C至+85°C)		95	300	µA
	平均电流，唤醒和看门狗定时器从低功耗振荡器获取时钟信号时测定，−10°C至+40°C环境温度范围		95	175	µA
I_{DD}(电流ADC)			1.7		mA
I_{DD}(电压/温度ADC)			0.5		mA
I_{DD}(精密振荡器)			400		µA

[1] 未经生产测试，但在产品发布时由设计和/或特性数据保证。
[2] 对于电流ADC增益设置PGA <= 64有效。
[3] 这些数值包括温度漂移。
[4] 失调误差漂移包括在失调误差中。这一典型规格反映温度漂移引起的失调误差。该典型值为温度漂移特性数据分布的平均值。
[5] 包括内部基准电压温度漂移。
[6] 在给定温度和增益下对电流通道进行用户系统校准可以消除这一误差。
[7] 增益漂移包括在总增益误差中。这一典型规格反映温度漂移引起的增益误差。该典型值为温度漂移特性数据分布的平均值。
[8] 电压通道规格包括电阻衰减器输入级。
[9] RMS噪声折合到电压衰减器输入端。例如，在f_{ADC} = 1 kHz时，ADC输入端的典型rms噪声为7.5 µV，并由衰减器比例调整(除以24)以产生这些等效输入噪声值。
[10] 初始自校准后有效。
[11] 通过修改增益校准寄存器的工厂设定值或利用系统校准，可将ADC输入范围扩展10%，也可缩小ADC输入范围(LSB大小)。
[12] 在ADC低功耗模式下，输入范围固定在±2.34375 mV。
[13] 对于低于10mV的差分输入有效。
[14] 为使T-ADC精确工作，VTEMP和GND_SW的电压绝对值至少必须为100mV。
[15] 利用盒子方法测定。
[16] 长期稳定性规格为非累积性。在后续1000小时周期内的漂移大幅低于第一个1000小时周期。
[17] 芯片温度。
[18] 耐久性是分别在−40°C、+25°C及+125°C时依据JEDEC 22标准方法A117认定为10,000个周期。在25°C时的典型耐久性为170,000个周期。
[19] 根据JEDEC 22标准方法A117，保持期限相当于85°C结温(T_J)时的寿命。保持期限会随着结温递减。
[20] 这些参数未经生产测试，但得到LIN兼容性测试支持。
[21] 热阻可用来计算环境到芯片温度的热梯度。
[22] 内部调节电源在REG_DVDD (I_{SOURCE} = 5 mA)和REG_AVDD (I_{SOURCE} =1 mA)处提供。
[23] 在Flash/EE存储器编程和擦除周期期间的典型额外电源电流消耗分别为7mA和5mA。

时序规格

LIN定时规格

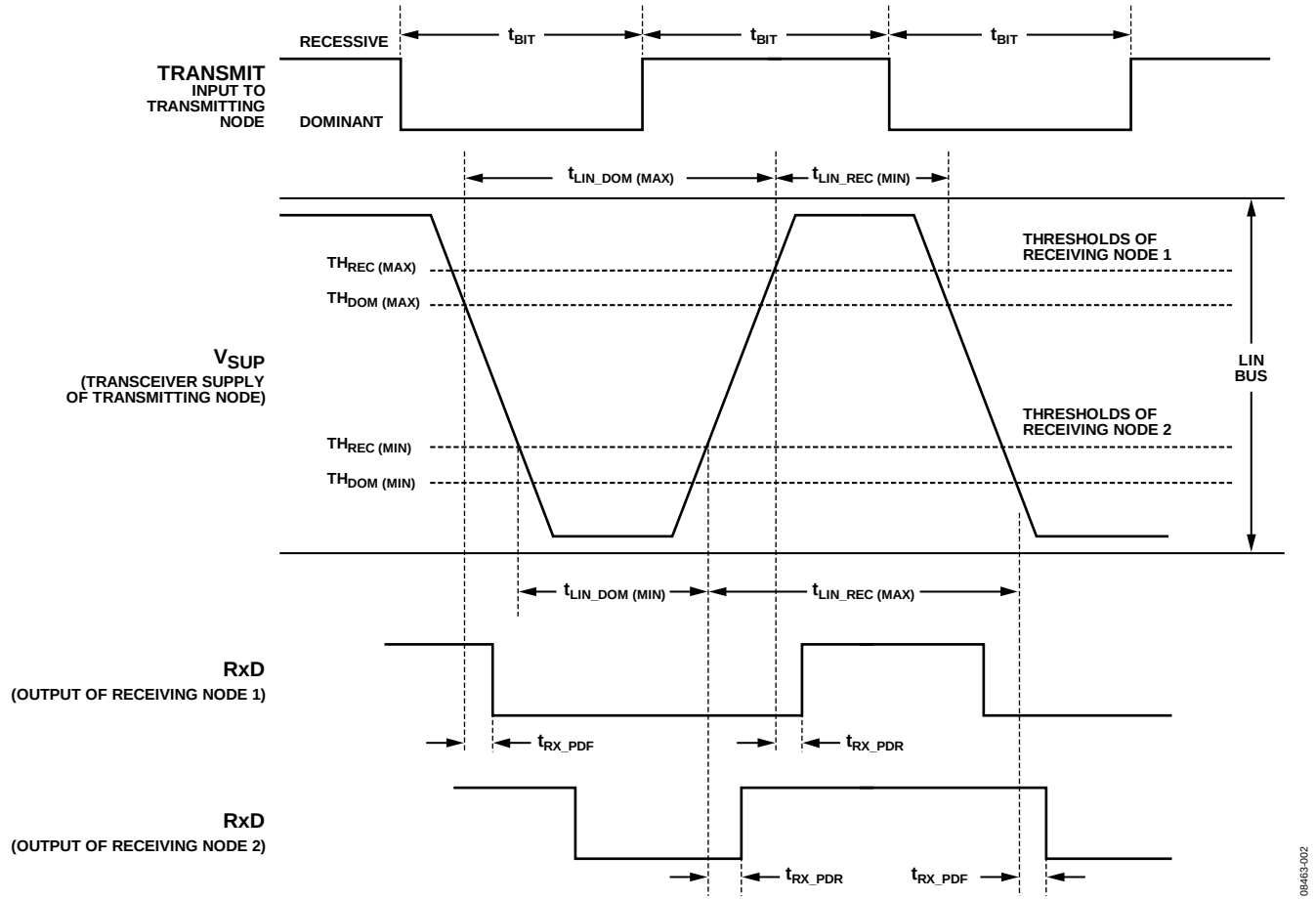


图2. LIN 2.1时序规格

绝对最大额定值

除非另有说明，T_A = −40°C至+115°C。

表2.

参数	额定值
AGND至DGND至VSS至IO_VSS	−0.3 V至+0.3 V
VBAT至AGND	−22 V至+40 V
VDD至VSS	−0.3 V至+40 V
LIN至IO_VSS	−16 V至+40 V
LIN短路电流[1]	200 mA
数字I/O电压至DGND	−0.3 V至REG_DVDD + 0.3 V
ADC输入至AGND	−0.3 V至REG_AVDD + 0.3 V
ESD (HBM)额定值	
HBM-ADI0082(基于ANSI/ESD STM5.1-2007)；LIN和VBAT除外的所有其它引脚	2.5 kV
LIN和VBAT	±6 kV
IEC61000-4-2，适用于LIN和VBAT	±7 kV
存储温度	150°C
结温	
瞬变	150°C
连续	130°C
引脚温度	
回流焊(15秒)	260°C

1 LIN引脚可以承受200 mA电流2秒。LIN工作期间，需要使能该器件的有源内部短路保护HVCFG[1] = 0，这是默认工作模式。如果发生短路事件，LIN引脚最多在90 µs后就会断开。

注意，超出上述绝对最大额定值可能会导致器件永久性损坏。这只是额定最值，并不能以这些条件或者在任何其它超出本技术规范操作章节中所示规格的条件下，推断器件能否正常工作。长期在绝对最大额定值条件下工作会影响器件的可靠性。

ESD警告

ESD(静电放电)敏感器件。

带电器件和电路板可能会在没有察觉的情况下放电。尽管本产品具有专利或专有保护电路，但在遇到高能量ESD时，器件可能会损坏。因此，应当采取适当的ESD防范措施，以避免器件性能下降或功能丧失。

引脚配置和功能描述

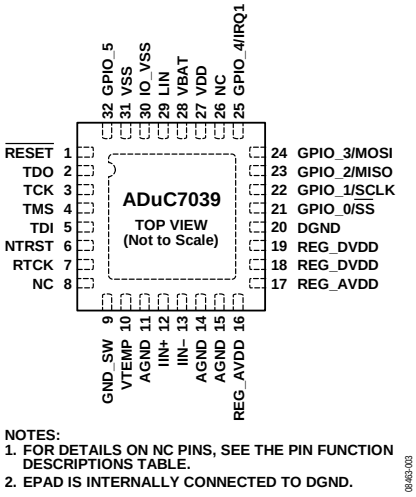


图3. 引脚配置

表3. 引脚功能描述

引脚编号	引脚名称	类型[1]	描述
1	$\overline{\text{RESET}}$	I	复位输入引脚。低电平有效。此引脚具有一个连接至REG_DVDD的内部弱上拉电阻。不用时可断开。如需更高安全性和鲁棒性，建议通过电阻将此引脚绑定至REG_DVDD。
2	TDO	O	JTAG测试数据输出。数据输出引脚，是器件标准6针JTAG调试端口之一。只能用作输出引脚。上电时禁用并通过内部弱上拉电阻拉高。不用时可断开。
3	TCK	I	JTAG测试时钟。时钟输入引脚，是器件标准6针JTAG调试端口之一。只能用作输入引脚，内置弱上拉电阻。不用时可断开。
4	TMS	I	JTAG测试模式选择。模式选择引脚，是器件标准6针JTAG调试端口之一。只能用作输入引脚，内置弱上拉电阻。不用时可断开。
5	TDI	I	JTAG测试数据输入。数据输入引脚，是器件标准6针JTAG调试端口之一。只能用作输入引脚，内置弱上拉电阻。不用时可断开。
6	NTRST	I	JTAG测试复位。复位输入引脚，是器件标准6针JTAG调试端口之一。只能用作输入引脚，内置弱下拉电阻。不用时可断开。同时受到片上内核监控以使能LIN引导加载模式。
7	RTCK	O	JTAG返回测试时钟。此输出引脚用于将JTAG时钟速度调整到ADuC7039可能的最高速率。
8	NC		不连接。此引脚内部连接，因此不外接。
9	GND_SW	I	切换至内部模拟地基准。外部温度通道和外部基准电压的负输入端。不用时，直接接到AGND系统地。
10	VTEMP	I	NTC/PTC温度测量的外部引脚。
11, 14, 15	AGND	S	片内精密模拟电路的接地基准。
12	IIN+	I	电流通道的正差分输入
13	IIN–	I	电流通道的负差分输入。
16, 17	REG_AVDD	S	片内调节器的标称2.6 V模拟输出。引脚16和引脚17必须一起连接到一个接地电容。
18, 19	REG_DVDD	S	片内调节器的标称2.6 V数字输出。引脚18和引脚19必须一起连接到接地电容。
20	DGND	S	片内数字电路的接地基准。

引脚编号	引脚名称	类型[1]	描述
21	GPIO_0/$\overline{SS}$	I/O	通用数字I/O 0或SPI接口。默认情况下，此引脚配置为输入端。内置弱上拉电阻，不用时可断开。 此多功能引脚可以配置为以下两种状态之一： 通用数字I/O 0 SPI接口，从机选择输入
22	GPIO_1/SCLK	I/O	通用数字I/O 1或SPI接口。默认情况下，此引脚配置为输入端。内置弱上拉电阻，不用时可断开。 此多功能引脚可以配置为以下两种状态之一： 通用数字I/O 1 SPI接口，串行时钟输入
23	GPIO_2/MISO	I/O	通用数字I/O 2或SPI接口。默认情况下，此引脚配置为输入端。内置弱上拉电阻，不用时可断开。 此多功能引脚可以配置为以下两种状态之一： 通用数字I/O 2 SPI接口，主机输入/从机输出引脚
24	GPIO_3/MOSI	I/O	通用数字I/O 3或SPI接口。默认情况下，此引脚配置为输入端。内置弱上拉电阻，不用时可断开。 此多功能引脚可以配置为以下两种状态之一： 通用数字I/O 3 SPI接口，主机输出/从机输入引脚
25	GPIO_4/IRQ1	I/O	通用I/O 4。外部中断请求1。默认情况下，此引脚配置为输入端。内置弱上拉电阻，不用时可断开。
26	NC		不连接。未内接并保留待用。因此请勿外接。
27	VDD	S	片内调节器的电池电源。
28	VBAT	I	电阻分压器的电池电压输入。
29	LIN	I/O	LIN串行接口输入/输出引脚。
30	IO_VSS	S	LIN引脚的接地基准。
31	VSS	S	接地基准。内部电压调节器的接地基准。
32	GPIO_5	I/O	通用I/O 5。默认情况下，此引脚配置为输入端。内置弱上拉电阻，不用时可断开。
EPAD	EPAD		裸露焊盘内部连接到DGND。

[1] I = 输入，O = 输出，S = 电源。

术语

转换速率

转换速率是指在ADC稳定后从ADC获得输出结果的速度。

此器件采用了Σ-Δ转换技术，这意味着当以相对较高的采样率对ADC前端信号进行过采样时，后接数字滤波器将抽取输出数据，以1 Hz至1 kHz的输出速率输出一个16位有效数据转换结果。

注意，当软件在同一ADC上切换不同输入时，必须先清空数字滤波器，然后对新结果求平均值。这个操作可能需要花费多个转换周期，具体的周期数取决于ADC的配置和滤波器的类型。

积分非线性(INL)

转换结果编码偏离通过其传递函数端点的直线的最大偏差。传递函数端点是指，在零点位置比第一个编码的跃变点低1/2 LSB的点，以及在满刻度位置比最后一个编码的跃变点高1/2 LSB的点(111 . . . 110至111 . . . 111)。该误差表示为满量程的百分比。

无失码

无失码是衡量ADC微分非线性的指标。误差用位来表示，2 N(N为无失码位数)指在整个ADC输入范围内保证发生的代码数目(ADC输出结果)。

失调误差

ADC第一个码转换输入电压和理想码跃迁的偏差即是失调误差。

失调误差漂移

失调误差漂移是指绝对失调误差的温度变化值。该误差表示为LSB/°C或nV/°C。

增益误差

增益误差是衡量ADC量程误差的指标。它衡量的是传递函数上任两点的测量量程和理想值的偏差。

输出噪声

输出噪声是指ADC输入直流信号时输出代码分布的标准偏差(即1 × Σ)，表示为μV或nV rms。如下式所定义，输出噪声或均方根噪声可用来求出ADC的有效分辨率：

$$\textit{有效分辨率} = \log_2(\text{满量程/均方根噪声})，\text{单位为“位”}$$

峰峰值噪声指ADC输入直流信号时，在输出代码分布6.6 × Σ范围内的代码偏差。因此，峰峰值噪声是均方根噪声的6.6倍。

如下式所定义，峰峰值噪声可用来求出在6.6 × Σ限值内无码闪烁的ADC(无噪声代码)分辨率：

$$\textit{无噪声代码分辨率} = \log_2(\text{满量程/峰峰值噪声})，\text{单位为“位”}$$

表4. 数据手册中用到的首字母缩写词

缩写	定义
ADC	模数转换器
ARM	高级精简指令集机器
JTAG	联合测试行动小组
LIN	局域互连网络
LSB	最低有效字节/位
MCU	微控制器单元
MMR	存储器映射寄存器
MSB	最高有效字节/位
OTP	一次性可编程
PID	保护标识符
POR	上电复位
rms	均方根

工作原理

ADuC7039是一款适用于在12V汽车电子应用中进行电池监控的完整系统解决方案，集成了所有在各种工作条件下对12 V电池参数(如电池电流、电压和温度)进行精确智能监控、处理和诊断等必需的功能。

可直接从12 V电池供电，从而最大程度减少了外部系统组件。片内低压差调节器向两个集成16位Σ-Δ ADC供电。这些ADC精确测量电池电流、电压及温度以采集汽车电池的运行和充电状态参数。

片内还集成了一个基于Flash/EE存储器的ARM7™微控制器(MCU)，用来预处理获得的电池变量并管理ADuC7039通过片内集成的局域互连网络(LIN)接口与主电子控制单元(ECU)的通信。

MCU可以配置为正常工作模式或灵活的省电工作模式。

在正常工作模式下，MCU通过锁相环(PLL)从片内振荡器间接接收时钟信号，最大时钟速率为10.24 MHz。在省电模式下，MCU可以完全关断，只有通过唤醒定时器、上电复位(POR)或LIN通信事件才能唤醒。

ADC可配置为正常(全功率)工作模式，在各种采样转换事件后中断MCU。

片内出厂固件支持通过LIN或JTAG串行接口端口进行在线Flash/EE编程，也支持通过JTAG接口进行非介入仿真。这些功能都集成在支持ADuC7039的低成本QuickStart™开发系统中。

ADuC7039直接从12V电池供电，工作温度范围为−40°C至+115°C。在115°C至125°C温度范围内仍能工作，但性能会降低。

ARM7TDMI-S内核概览

ARM7内核是一款由ARM®公司开发的32位精简指令集计算机(RISC)。ARM7TDMI-S采用冯诺依曼架构，指令和数据使用单32位总线。它支持8位、16位和32位数据长度，指令字长度可为16位或32位，具体的位数取决于内核的工作模式。

ARM7TDMI-S采用ARM7内核，具有四种附加特性，如表5所列。

表5. ARM7TDMI-S

特性	描述
T	支持Thumb®指令集(16位)
D	支持调试
M	增强乘法器
I	包含EmbeddedICE™模块以支持嵌入式系统调试

Thumb模式(T)

一条ARM指令的长度为32位。ARM7TDMI-S处理器支持压缩至16位的第二指令集，即Thumb指令集。Thumb指令集具有16位存储器的更快代码执行速度和更大的代码密度，使ARM7TDMI-S内核特别适合嵌入式应用。

然而，Thumb模式有三个缺点：

- 与ARM相比，执行同一个任务时，Thumb代码通常需要更多的指令。因此，ARM代码特别适合于在大多数应用中最大程度提升时间关键代码的性能。
- Thumb指令集不含一些异常处理指令，所以可能需要ARM代码来进行异常处理。
- 发生中断时，内核矢量指向存储器内的中断地址并执行该地址单元内的代码。第一条命令须包含在ARM代码内。

乘法器(M)

ARM7TDMI-S指令集包括一个增强乘法器，具有四条额外指令用来进行32位与32位相乘或32位与32位乘加(MAC)，得到64位结果。

EmbeddedICE (I)

EmbeddedICE模块为ARM7TDMI-S提供集成片内调试功能。它包括断点和观察点寄存器，可用来实现非介入用户代码调试。这些寄存器可以通过JTAG测试端口来控制。当遇到一个断点或观察点时，处理器会暂停运行，进入调试状态。一旦进入调试状态，就可以查询处理器寄存器，也可以查询Flash/EE、SRAM和存储器映射寄存器。

ARM7异常

ARM7支持5种类型的异常，每种类型分别对应一种优先处理模式。这5种异常类型为：

- 正常中断或IRQ。这种异常用于内部和外部事件的通用中断处理。
- 快速中断或FIQ。这种异常用于数据传输或低延迟通信通道处理。FIQ的优先级高于IRQ。
- 存储器中止(预取和数据)
- 尝试执行未定义指令
- 软件中断(SWI)指令，用于调用操作系统。

一般情况下，程序员将中断定义为IRQ，但要获得优先级更高的中断，可将中断定义为FIQ类型。

异常优先级及其矢量地址见表6。

表6. 异常优先级和矢量地址

优先级	异常	地址
1	硬件复位	0x00
2	存储器中止(数据)	0x10
3	FIQ	0x1C
4	IRQ	0x18
5	存储器中止(预取)	0x0C
6	软件中断[1]	0x08
6	未定义指令[1]	0x04

[1] 软件中断和未定义指令异常的优先级相同且互斥。

ARM寄存器

ARM7TDMI-S共有16个标准寄存器。R0至R12用于数据操作，R13是堆栈指针，R14是链接寄存器，而R15是程序计数器，指示当前正在执行的指令。链接寄存器存放用户转移地址(使用转移和链接命令情况)或发生异常时的命令。

堆栈指针用于保存堆栈的当前位置。通常，ARM7TDMI-S堆栈始于可用RAM空间顶部，并根据需要递减。每个异常分别对应一个堆栈。用户可根据目标应用的需要来配置每个堆栈的大小。在ADuC7039上，堆栈从0x00040FFC开始递减。利用C语言等高级语言进行编程时，必须确保不产生堆栈溢出。这取决于所用编译器的性能。

异常发生后，异常模式专用的寄存器将取代某些标准寄存器。所有的异常模式均有各自的替换寄存器组，用于堆栈指针(R13)和链接寄存器(R14)，如图4所示。快速中断模式(FIQ)有更多寄存器(R8到R12)，用于快速中断处理。由于非关键寄存器的数目不断增加，在进行中断处理时，无需保存或恢复这些寄存器，从而缩短了中断处理过程的响应时间。

有关程序员模型和ARM7TDMI-S内核架构的详情，请参见直接由ARM公司提供的ARM7DMI-S技术手册和ARM架构手册。

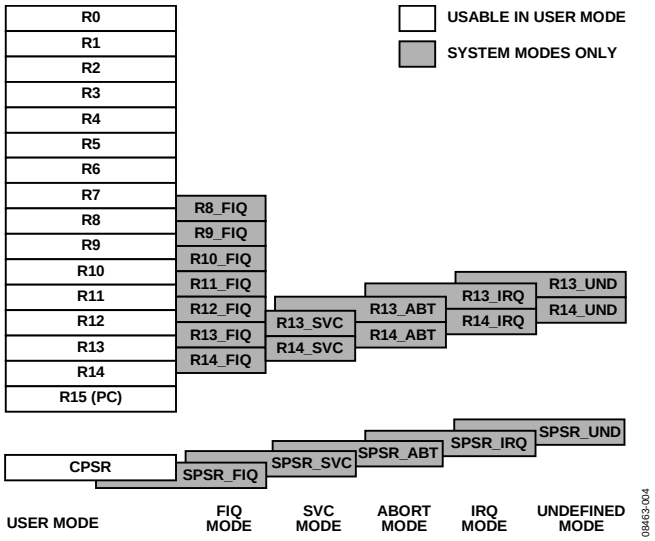


图 4. 寄存器结构图

中断延迟时间

快速中断请求(FIQ)的最大延迟时间包括请求通过同步器、最长指令(LDM)完成载入所有寄存器(包括PC)以及完成数据中止和FIQ跳转的最长可能时间总和。延迟结束时，ARM7TDMI-S执行0x1C(FIQ中断矢量地址)处的指令。如果系统采用一个连续的10.24 MHz处理器时钟，则最长总时间为50个处理器周期，即略少于5 μs。IRQ最大延迟时间的计算方法类似，但必须考虑到FIQ优先级更高，可能任意延长进入IRQ处理例行程序的时间。如果不使用LDM命令，这个时间可以缩短到42个周期；一些编译器可以选择不使用此命令进行编译。另一个选择是在Thumb模式下运行器件，这样可以将时间缩短至22个周期。

FIQ或IRQ中断的最短延迟时间为5个周期，包括请求通过同步器的最短时间和进入异常模式所需的时间。

注意，发生异常时，ARM7TDMI-S最初(第一个指令)在ARM(32位)模式下运行。如有必要，在执行中断服务例行程序时，用户可立即从ARM模式切换到Thumb模式。

存储器结构

ARM7是一个采用冯•诺依曼架构的MCU内核，它将存储器看作一个2^{32}个字节的线性阵列。如图5所示，ADuC7039将此线性阵列映射成四个不同的用户区域：可重映射的存储区域、SRAM区域、Flash/EE区域及存储器映射寄存器(MMR)区域。

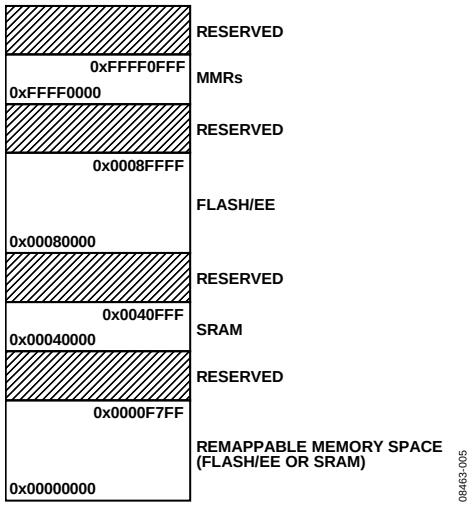


图5. ADuC7039存储器映射，64 kB闪存选项

- 该存储空间的前64 kB用作可重新映射片内Flash/EE或SRAM的区域。
- ADuC7039在存储器映射图顶部具有第二块4 kB区域以定址MMR，用来配置并监控所有片内外设。
- ADuC7039的SRAM大小为4 kB。
- ADuC7039的片内Flash/EE存储器大小为64 kB，其中62 kB供用户使用，剩余2 kB保留供片上内核使用。

访问(读写)存储器映射中未定义区域，将导致数据中止异常。

存储格式

ADuC7039存储被配置成从小到大顺序格式：LSB位于最低字节地址，MSB位于最高字节地址。

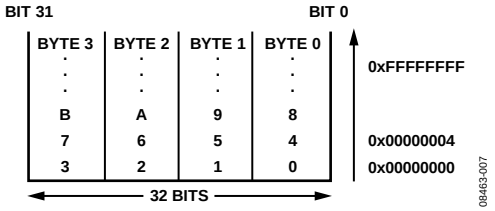


图 6. 从小到大顺序格式

SRAM

ADuC7039的SRAM容量为4 kB(1024 × 32位，即1024个字)，存储起始地址为0x40000。

RAM空间既可用作数据存储器，也可用作易失性程序空间。

由于SRAM阵列被配置成32位宽的存储器阵列，ARM代码可以直接从SRAM以完全时钟速度执行。SRAM支持8、16及32位段读写。

重映射

ARM异常矢量位于存储器阵列底部，其存储地址范围为0x00000000至0x00000020。

默认情况下，复位后Flash/EE存储器会被逻辑映射到地址0x00000000。如果要将SRAM逻辑重映射到地址0x00000000，需要在0xFFFF0220处设置SYSMAP MMR的位0。要使Flash/EE返回到0x00000000，SYSMAP的位0需清0。

有时需要将RAM重映射到0x00000000，以便在擦除Flash/EE存储器一页的同时，执行SRAM中的代码。

重映射操作

发生复位时，ADuC7039会自动执行工厂内置的程序代码。这就是所谓的隐藏内核，用户代码不能访问。如果ADuC7039处于正常模式，它首先执行内核的上电配置例行程序，然后跳转到复位矢量地址0x00000000，执行用户的复位异常例行程序。由于复位时Flash/EE镜像到存储器阵列底部，复位例行程序必须始终写入Flash/EE内。

重映射命令必须从绝对Flash/EE地址处开始执行，而不能从被镜像的存储器重映射段开始执行，因为该段可能被SRAM所取代。如果在从镜像位置执行代码时执行重映射操作，可能会发生预取/数据中止，或者用户可能观察到异常程序操作。

任何一种复位都会将Flash/EE存储器逻辑重映射到存储器阵列底部。

SYSMAP寄存器

名称： SYSMAP

地址： 0xFFFF0220

默认值： 由内核更新

访问类型： 读/写

功能： 该8位寄存器允许用户代码将RAM或Flash/EE存储空间重映射到ARM存储器空间底部(起始地址为0x00000000)。

表7. SYSMAP MMR位分配

位	描述
7至1	保留。这些位保留，应由用户代码写入0。
0	重映射位。 通过将该位置1，用户可将SRAM重映射至0x00000000。 复位后该位自动清0，以重映射Flash/EE到0x00000000。

复位

一共有四种类型的复位：外部复位、上电复位、看门狗复位及软件复位。RSTSTA寄存器指示上一次复位的来源，也可由用户代码写入以启动软件复位事件。此寄存器的清0可通过在0xFFFF0234处写入RSTCLR MMR来完成。RSTCLR的位分配镜像RSTSTA的位。在复位异常服务程序执行时，可以使用这两种寄存器来识别复位源。所有四种复位事件的含义如表9所示。

RSTSTA寄存器

名称： RSTSTA

地址： 0xFFFF0230

默认值： N/A

访问类型： 读/写

功能： 此8位寄存器指示上一次复位事件的来源，也可由用户代码写入以启动软件复位。

RSTCLR寄存器

名称： RSTCLR

地址： 0xFFFF0234

访问类型：只写

功能： 此8位只写寄存器将对应RSTSTA位清0。

表8. RSTSTA/RSTCLR MMR位分配

位	描述
7至4	未用。全部不用，始终读取0。
3	外部复位。 发生外部复位时，该位由硬件置1。 通过设定RSTCLR中的相应位，可将该位清0。
2	软件复位。 该位由用户代码置1时，可产生软件复位。 通过设定RSTCLR中的相应位，可将该位清0。[1]
1	看门狗超时。 发生看门狗超时事件时，该位由硬件置1。 通过设定RSTCLR中的相应位，可将该位清0。
0	上电复位。 发生上电复位时，该位由硬件置1。 通过设定RSTCLR中的相应位，可将该位清0。

[1] 如果RSTSTA的软件复位位已置1，任何不将该位清0的RSTCLR写入操作都会产生软件复位。

表9. 器件复位含义

	影响							
复位	将外部引脚复位到默认状态	执行内核	复位所有外部MMR(不包括RSTSTA)	复位所有高压间接寄存器	复位外设	复位看门狗定时器	RAM有效[1]	RSTSTA(复位事件后的状态)
POR	是	是	是	是	是	是	是/否[2]	RSTSTA[0] = 1
看门狗	是	是	是	是	是	否	是	RSTSTA[1] = 1
软件	是	是	是	是	是	否	是	RSTSTA[2] = 1
外部引脚	是	是	是	是	是	否	是	RSTSTA[3] = 1

[1] 在LIN下载后复位的情况下RAM无效。

[2] LVF使能后，对RAM的影响取决于HVSTA[2]的内容。当利用HVCFG[4]使能LVF时，如果LVF状态位HVSTA[2]为1，则POR复位机制不会破坏RAM。详情见"低压标志(LVF)"部分。

FLASH/EE存储器

ADuC7039片内集成Flash/EE存储器技术，向用户提供非易失、在线可编程存储空间。

类似于EEPROM，Flash存储器可在字节水平上进行系统内编程，但在编程前必须先擦除，擦除单位为页模块。因此，Flash存储器常常被称为Flash/EE存储器，这一名称更确切。

总之，由于具有非易失性、在线编程、高密度及低成本等特点，Flash/EE是更理想的存储设备。通过集成于ADuC7039内的Flash/EE存储器技术，用户可以在线更新程序代码空间而不必在远程操作节点处替换一次性可编程(OTP)设备。

Flash/EE存储器的物理地址为0x80000。硬件复位时，它被逻辑映射到0x00000000。所有Flash/EE存储器位置的工厂预设默认内容均为0xFFFF。Flash/EE可采用8、16及32位段读取，并采用16位段写入。额定耐久性为10,000个周期。此额定值基于循环操作(即擦除和编程)每个字节的次数。通过软件实现冗余方案，可以确保所耐久性超过10,000个周期。

用户还可在代码执行时将数据变量写入Flash/EE存储器，例如用以存储电池诊断参数数据。

整个Flash/EE都可以用作用户的代码和非易失性数据存储器。数据和代码之间没有区别，因为ARM代码及数据共用同一空间。Flash/EE存储器的实际宽度为16位，这意味着在ARM模式(32位指令)下每次取指令必须两次访问Flash/EE存储器。ARM7TDMI-S以10.24 MHz默认时钟频率工作，但Flash/EE存储器控制器以20.48 MHz时钟频率工作，这意味着Flash/EE存储器控制器可在一个内核时钟周期内透明读取第二个16位半字(32位ARM操作代码的一部分)。

Flash/EE的页面大小为512个字节。一般情况下，Flash/EE控制器擦除一页需要20 ms，写入一个16位字需要50 µs。

一个16位位置在前后两次擦除中间最多只能写入两次，即字节操作而非位操作。如果一个位置连续写入两次以上，Flash/EE页内容就有可能被破坏。

Flash/EE存储器可通过LIN接口或集成JTAG端口在串行下载模式下进行在线编程。

串行下载(在线编程)

AduC7039支持通过LIN引脚下载代码。相关协议参见“应用笔记AN-946：通过LIN—协议6进行Flash/EE存储器编程”。

JTAG访问

ADuC7039具有一个片内JTAG调试端口，支持进行代码下载和调试。

Flash/EE MMR接口

ADuC7039通过片内存储器控制器来管理Flash/EE存储器的访问和控制。控制器将Flash/EE存储器当作一个64 kB的存储模块来管理。

注意，如果从Flash/EE存储器执行，MCU内核会暂停，直到命令完成。用户软件必须确保Flash/EE控制器在PLL关断前完成任何擦除或写入周期。如果PLL在擦除或写入周期结束前关断，可能会破坏Flash/EE页面。用户代码、LIN和JTAG编程都利用了Flash/EE控制接口，它包括下列MMR：

- FEESTA：只读寄存器，反映Flash/EE控制接口的状态。
- FEEMOD：设置Flash/EE控制接口的工作模式。
- FEECON：8位命令寄存器。命令含义如表10所述。
- FEEDAT：16位数据寄存器。
- FEEADR：16位地址寄存器。
- FEESIG：由于签名命令启动，保持24位代码签名。
- FEEHID：MMR保护。Flash/EE存储器代码空间的读写保护控制。如果先前是通过FEEPRO寄存器配置，FEEHID可能要求软件密钥来使能访问。
- FEEPRO：FEEHID寄存器的缓冲器，存储FEEHID值，以便它在随后的复位和上电事件时自动下载到FEEHID寄存器。

下面详细说明各Flash/EE控制MMR的位分配。

FEECON寄存器

名称： FEECON

地址： 0xFFFF0E08

默认值： 0x07

访问类型： 读/写

功能： 此8位寄存器由用户代码写入，以控制Flash/EE存储器控制器的工作模式。

表10. FEECON中的命令代码

代码	命令	描述
0x00[1]	保留	保留；此命令不能由用户代码写入。
0x01[1]	单次读取	将FEEADR索引的16位数据载入FEEDAT。
0x02[1]	单次写入	将FEEDAT中的数据写入FEEADR索引的地址。此操作耗时50 µs。
0x03[1]	擦除写入	擦除由FEEADR索引的存储页，并且把FEEDAT中的数据写入FEEADR所指的存储区域。此操作耗时20 ms。
0x04[1]	单次验证	将FEEADR索引的地址中的数据与FEEDAT中的数据进行比较，比较的结果由FEESTA的第1位显示。
0x05[1]	单次擦除	擦除由FEEADR索引的页面。
0x06[1]	批量擦除	擦除62 kB的用户空间。2 kB内核被保护。此操作耗时2.48 s。为防止意外执行，执行此指令要求一个命令序列；详见"批量擦除命令执行序列"部分。
0x07	空闲	默认命令。
0x08	保留	保留；此命令不能由用户代码写入。
0x09	保留	保留；此命令不能由用户代码写入。
0x0A	保留	保留；此命令不能由用户代码写入。
0x0B	签名	处理器运行该命令后，会产生一个基于LFSR(线性反馈移位寄存器)的24位签名，该签名被写入FEESIG。参见"Flash/EE存储器签名"部分。
0x0C	保护	此命令仅可运行一次。FEEPRO的值会被保存，只能利用批量擦除(0x06)或软件保护密钥加以清除。
0x0D	保留	保留；此命令不能由用户代码写入。
0x0E	保留	保留；此命令不能由用户代码写入。
0x0F	Ping	无操作；产生中断。

[1] 当上述任一命令执行完毕，FEECON就会立即读取0x07。

批量擦除命令执行序列

考虑到批量擦除命令的后果严重，启动此操作前必须执行特定代码序列。

1. 确保FEESTA已清0。
2. FEEMOD的位3置1。
3. 将0xFFC3写入FEEADR。
4. 将0x3CFF写入FEEDAT。
5. 运行FEECON中的批量擦除命令(0x06)。

命令序列示例

执行批量擦除的命令序列见以下示例：

```
int a = FEESTA;           // Ensure FEESTA is cleared
FEEMOD = 0x08
FEEADR = 0xFFC3
FEEDAT = 0x3CFF
FEECON = 0x06;            //Mass erase command
while (FEESTA & 0x04){}   //Wait for command to finish
```

FEESTA寄存器

名称： FEESTA

地址： 0xFFFF0E00

Default Value: 0xXXX0

访问类型： 只读

功能： 此16位只读寄存器可由用户代码读取，反映Flash/EE存储器控制器的当前状态。

表11. FEESTA MMR位分配

位	描述
15至4	保留。
3	Flash/EE中断状态位。 中断发生后，即当一条命令执行完毕且FEEMOD寄存器中的Flash/EE中断使能位被置1时，该位自动置1。 当用户代码读取FEESTA寄存器时，该位自动清0。
2	Flash/EE控制器繁忙。 Flash/EE控制器繁忙时，该位自动置1。 控制器空闲时，该位自动清0。
1	命令失败。 当命令写入FEECON不成功时，该位自动置1。 当用户代码读取FEESTA寄存器时，该位自动清0。
0	命令成功。 命令成功执行时，MCU自动将该位置1。 当用户代码读取FEESTA寄存器时，该位自动清0。

FEEMOD寄存器

名称： FEEMOD

地址： 0xFFFF0E04

默认值： 0x0000

访问类型： 读/写

功能： 此寄存器由用户代码写入，以配置Flash/EE存储器控制器的工作模式。

表12. FEEMOD MMR位分配

位	描述
15至7	未用。全部保留供日后使用并应由用户代码写入0。
6至5	Flash/EE安全锁定位。这些位必须写入[6:5] = 1、0，以完成Flash/EE安全保护序列。
4	Flash/EE控制器命令完成中断使能。 该位由用户代码置1时，使能Flash/EE控制器在Flash/EE命令完成时产生中断。 该位由用户代码清0时，禁止在Flash/EE命令完成时产生Flash/EE中断。
3	Flash/EE擦除/写入使能。 该位由用户代码置1时，通过FEECON使能Flash/EE擦除和写入访问。 该位由用户代码清0时，通过FEECON禁用Flash/EE擦除和写入访问。
2	保留。
1	Flash/EE控制器中止使能。 该位由用户代码置1时，使能Flash/EE控制器中止功能。 该位由用户代码清0时，禁用Flash/EE控制器中止功能。
0	保留。

FEEADR寄存器

名称： FEEADR

地址： 0xFFFF0E10

默认值： 由内核更新

访问类型： 读/写

功能： 该16位寄存器决定通过FEECON执行任何Flash/EE命令的操作地址。

FEEDAT寄存器

名称： FEEDAT

地址： 0xFFFF0E0C

默认值： 0x0000

访问类型： 读/写

功能： 此16位寄存器存放Flash/EE存储器的读写数据。

Flash/EE存储器签名

Flash/EE存储器中可供用户使用的空间(62 kB或其中一部分)可利用FEESIG寄存器和签名命令进行签名。

此功能自动读取FEEADR和FEEDAT MMR指定的存储器部分中的代码：

- FEEADR包含一个位于要签名部分的前半页的地址。
- FEEDAT包含一个位于要签名部分的最后一页之上的前半页的地址。如本例中的图7所示，页0和页1被签名。

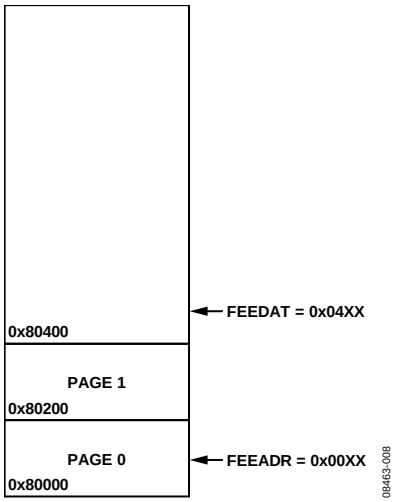


图7. 签名命令索引

- 如果FEEADR和FEEDAT的8个MSB相同，即这些MMR指向同一页面，则不会给任何页签名。
- 签名不包括最后两个16位位置，这些位置保留用于用户编程的签名。
- 可以给半页签名，方法是在FEEADR和FEEDAT中指定半页地址。例如，要给页0的后半页和页1的前半页签名，则应设置FEEADR = 0x0100且FEEDAT = 0x0300。

上电时，在转入用户代码之前，片上内核也会利用此功能来检查页0的有效性。给器件编程时，应将页0的签名存储在地址0x801FC。详情参见“ADuC7039内核”部分。

Flash/EE存储器签名寄存器

名称： FEESIG

地址： 0xFFFF0E18

默认值： 由内核更新

访问类型： 只读

功能： 该MMR包含Flash/EE存储器的24位签名。

用户代码签名示例

```
Int a = FEESTA;           // Ensure FEESTA is cleared
FEEADR = 0x0000;          // Start page address
FEEDAT = 0x0600;          // Stop (page + 1) address
FEECON = 0x0B;            // Signs Page 0 to Page 2 excluding
                          // Address 0x805FC
while (FEESTA & 0x04){}   // Wait for command to finish
```

用户代码可以将FEESIG的内容与地址0x805FC的内容相比较。

多项式

ADI公司提供了一个软件例程来计算唯一的24位签名。

Flash/EE存储器安全性

用户可用的62 kB Flash/EE存储器可以利用FEEHID寄存器提供读写保护。

FEEHID的MSB(位31)保护整个Flash/EE不被JTAG读取。

FEEHID的位[30:0]保护页123至页0不被写入。每位保护4页，即2 kB。

FEEPRO寄存器镜像复制FEEHID MMR的位定义。FEEPRO MMR允许用户代码锁定Flash/EE存储器的保护或安全配置，这样在以后上电或复位事件时可自动载入保护配置。这种灵活性允许用户利用FEEHID MMR临时设置并测试保护设置，随后在将保护系统运送到现场时锁定所需保护配置(利用FEEPRO)。

Flash/EE存储器保护寄存器

名称： FEEHID和FEEPRO

地址： 0xFFFF0E20(用于FEEHID)和0xFFFF0E1C(用于FEEPRO)

默认值： 0xFFFFFFFF(用于FEEHID)和0x00000000(用于FEEPRO)

访问类型： 读/写

功能： 这些寄存器由用户代码写入以配置Flash/EE存储器保护。

表13. FEEHID和FEEPRO MMR位分配

位	描述
31	读保护位。 该位由用户代码清0时，可为62 kB Flash/EE模块代码提供读保护功能。 该位由用户代码置1时，允许通过JTAG对62 kB Flash/EE模块进行读取访问。
30至0	写保护位。 由用户代码置1时，这些位解除对62 kB Flash/EE代码存储器的页0至页123的保护。每位写保护4页，每页包括512字节。 由用户代码清0时，这些位为62 kB Flash/EE代码存储器的页0至页123提供写保护。每位写保护4页，每页包括512字节。

总之，有如下三种保护级别。

临时保护

临时保护可通过直接写入FEEHID MMR来设置和取消。此寄存器为易失性，因此保护只在器件上电时才能维持。此保护在周期供电后便不再重新载入。

加密永久性保护

加密永久性保护可通过FEEPRO设置以锁定保护配置。所需FEEPRO写入序列开头使用的软件密钥仅保存一次，必须在以后的任何FEEHID或FEEPRO MMR访问中使用。批量擦除可将软件保护密钥设回到0xFFFF，但同时也擦除了整个用户代码空间。

永久性保护

永久性保护可通过FEEPRO设置，设置方式类似于加密永久性保护，唯一不同在于所用软件密钥为0xDEADDEAD。FEEPRO写入序列保存后，只有批量擦除可将软件保护密钥设回到0xFFFFFFFF。这也会擦除整个用户代码空间。

写入软件保护密钥和设置永久性保护的序列

1. 写入和欲保护页相对应的FEEPRO。
2. 将新的(用户定义)32位软件保护密钥写入FEEADR的位[31:16]和FEEDAT的位[15:0]。
3. 将1、0写入FEEMOD的位[6:5]，并将FEEMOD的位3置1。
4. 执行FEECON中的写入密钥命令(0x0C)。

要取消或修改保护，可使用相同的序列，同时修改FEEPRO值。

上述写入密钥并设置永久性保护的序列示例如下，此示例为Flash/EE的页8至页11提供写保护。

序列示例

```
Int a = FEESTA;       // Ensure FEESTA is cleared
FEEPRO = 0xFFFFFFFB;  // Protect Page 8 to Page 11
FEEADR = 0x66BB;      // 32-bit key value (Bits[31:16])
FEEDAT = 0xAA55;      // 32-bit key value (Bits 15:0])
FEEMOD = 0x0048       // Lock security sequence
FEECON = 0x0C;        // Write key command
while (FEESTA & 0x04){}
                      // Wait for command to finish
```

Flash/EE存储器可靠性

此器件的Flash/EE存储器阵列完全满足两大Flash/EE存储器关键特性要求：周期耐久性和数据保持期限。

耐久性用于衡量Flash/EE存储器重复多个编程、读取及擦除周期的能力。一个耐久性周期包括4个独立、连续的事件，定义如下：

- 初始页面擦除序列
- 读取/验证序列
- 字节编程序列
- 第二个读取/验证序列

在进行可靠性验证时，Flash/EE存储器中三页(顶、中和底)的每半个字(16位宽)可循环存取10000次(从0x0000至0xFFFF)。如表1所示，此器件的Flash/EE存储器耐久性指标是根据JEDEC保存期限规格A117测试的。结果表明，随电源和温度变化，最小耐久性参数规格达10,000周期。

保持期限衡量Flash/EE存储器长时间保持编程数据的能力。该器件在特定结温(T_J = 85°C)下根据标准JEDEC保持期限规格A117进行测试，如表1所示。这表明每次对Flash/EE存储器进行重新编程时，都保证Flash/EE存储器在完全指定的保持期限内保持数据。此外应注意，保持期限(基于0.6 eV的激活能)随T_J递减，如图8所示。

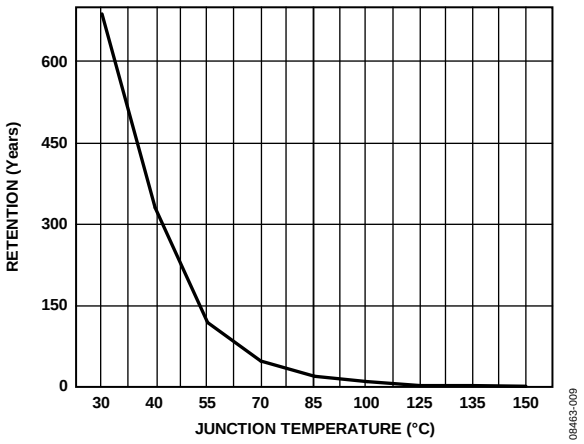


图8. Flash/EE存储器数据保持期限

ADuC7039内核

ADuC7039在顶部2 kB Flash/EE代码空间内常驻有一个片上内核。发生任何复位事件后，此内核计算自己的校验和，并将其与生产测试期间编程的校验和进行比较，以确保内核不含任何错误。如果有错误，SYSCHK寄存器恢复其默认值，且器件进入用户模式。在正常情况下，校验和写入SYSCHK MMR。

系统内核校验和

名称： SYSCHK

地址： 0xFFFF0244

默认值： 0x00000000(上电时由内核更新)

访问类型： 读/写

功能： 上电时，该32位寄存器保存内核校验和。

然后，内核将工厂校准数据从制造数据空间复制到不同片内外设中。内核校准的外设如下：

- 精密振荡器
- 低功耗振荡器
- REG_AVDD/REG_DVDD
- 基准电压源
- 电流ADC(失调和增益)
- 电压/温度ADC(失调和增益)

可被内核修改且不同于POR默认值的处理器寄存器和用户寄存器如下：

- R0至R15
- GP0CON
- SYSCHK
- FEEADR/FEEDAT/FEECON/FEESIG
- HVDAT/HVCON
- HVCFG
- T2LD

ADuC7039还有一个片内LIN下载器。

内核执行流程图如图9所示。如表66所述，当前内核版本可从R5获得。

复位后，一旦内核代码激活，便会禁用看门狗定时器。在内核执行期间，看门狗定时器有效，超时周期为500 ms。这确保内核发生错误时，ADuC7039会自动复位。进入LIN下载模式后，会定期刷新看门狗。

正常内核执行时间(不包括LIN下载)少于5 ms。LIN下载模式的进出只能通过复位完成。

正常内核执行期间会修改SRAM地址0至地址0x2B，LIN下载期间还会修改SRAM地址0xFF至地址0x110。

注意，即使NTRST = 0，也不会执行用户代码，除非地址0x801FC包含0x16400000或页0的CRC。如果地址0x801FC不包含这些信息，就会无法执行用户代码而转入LIN下载模式。 内核执行期间会禁止JTAG访问。

ADuC7039交货时，Flash用户空间被完全擦除，如果初次上电时NTRST = 0，则进入LIN下载模式。

当NTRST = 1时，则始终执行用户代码，并使能JTAG。

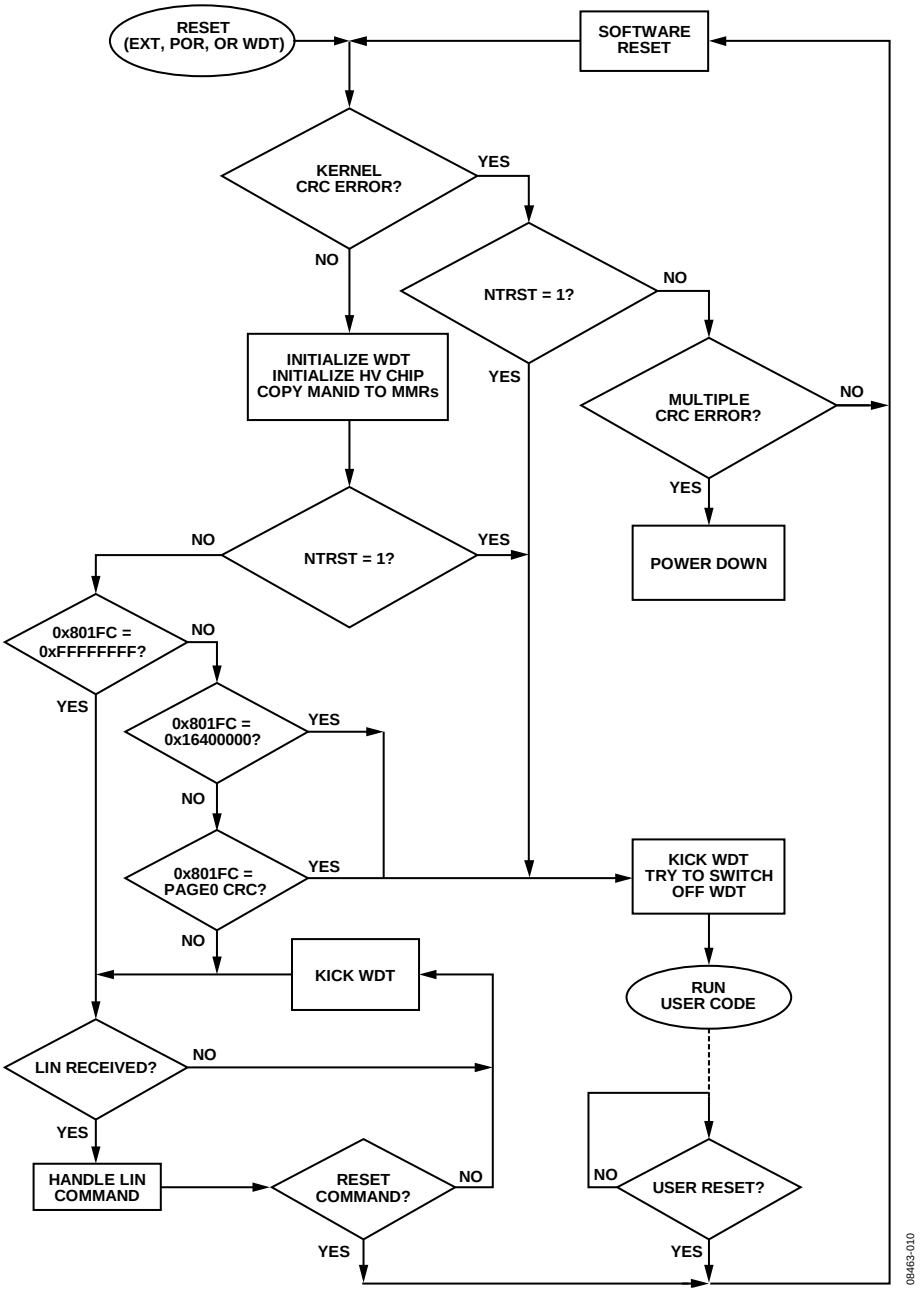


图9. ADuC7039内核流程图

存储器映射寄存器(MMR)

存储器映射寄存器(MMR)空间映射到顶部4 kB的MCU存储器空间，并且可以通过ARM7寄存器组的间接寻址、载入和存储命令来加以访问。ADuC7039寄存器映射图如图10所示。

MMR空间为CPU和所有片内外设提供接口。除ARM7内核寄存器(参见"ARM寄存器"部分)外，所有寄存器都常驻于MMR区域内。

如"完整MMR列表"部分中的MMR详细映射(表14至表23)所示，MMR数据宽度为1个字节(8位)到4个字节(32位)不等。ARM7内核可32位读写访问任何MMR(单字节或多字节宽度寄存器)。

例如，如上文所述，读取结果采用从小到大顺序格式排列。但如果ARM7内核试图以16位访问形式访问4字节(32位)MMR，就会导致错误。在16位写入32位MMR的情况下，16个最高有效位(高16位)全部被写入0。更明显的是，在16位读取32位MMR的情况下，只能读取MMR位中的16位。

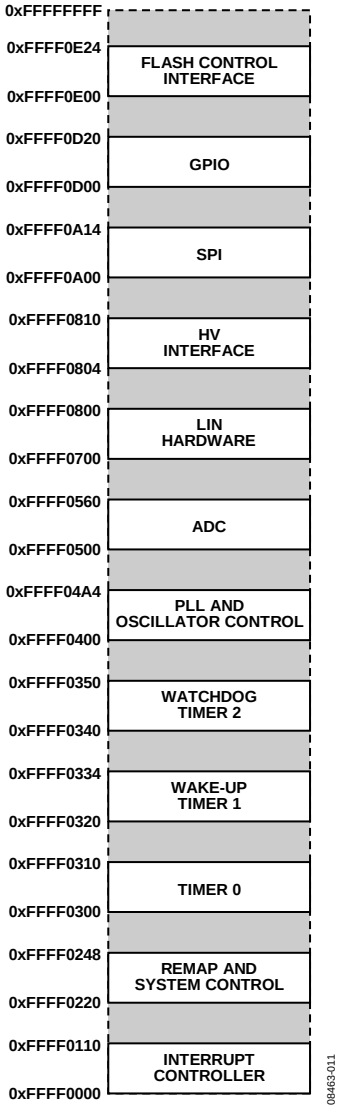


图10. 顶层MMR映射

完整MMR列表

在下列MMR列表中，地址用16进制码表示。访问类型：R表示读取，W表示写入，RW表示读写。

表14. IRQ基地址 = 0xFFFF0000

地址	名称	字节	访问类型	默认值	描述
0x0000	IRQSTA	4	R	0x00000000	活动IRQ源。
0x0004	IRQSIG	4	R	0x00003080	所有IRQ源的当前状态(使能和禁用)。
0x0008	IRQEN	4	RW	0x00000000	使能的IRQ源。
0x000C	IRQCLR	4	W	N/A	MMR禁用IRQ源。
0x0010	SWICFG	4	W	N/A	软件中断配置MMR。
0x0100	FIQSTA	4	R	0x00000000	活动IRQ源。
0x0104	FIQSIG	4	R	0x00003081	所有IRQ源的当前状态(使能和禁用)。
0x0108	FIQEN	4	RW	0x00000000	使能的IRQ源。
0x010C	FIQCLR	4	W	N/A	MMR禁用IRQ源。

表15. 系统控制基地址 = 0xFFFF0200

地址	名称	字节	访问类型	默认值	描述
0x0220	SYSMAP	1	RW	N/A	REMAP控制寄存器。
0x0230	RSTSTA	1	RW	N/A	复位状态MMR。
0x0234	RSTCLR	1	W	N/A	RSTSTA清零MMR。
0x0244	SYSCHK[1]	4	RW	N/A	内核校验和。参见"系统内核校验和"部分。

[1] 由内核更新。

表16. 定时器基地址 = 0xFFFF0300

地址	名称	字节	访问类型	默认值	描述
0x0300	T0LD	2	RW	0x0000	定时器0载入寄存器。
0x0304	T0VAL	2	R	0x0000	定时器0值寄存器0。
0x0308	T0CON	4	RW	0x0000	定时器0控制MMR。
0x030C	T0CLRI	1	W	N/A	定时器0中断清除寄存器。
0x0320	T1LD	4	RW	0x00000000	定时器1载入寄存器。
0x0324	T1VAL	4	R	0xFFFFFFFF	定时器1值寄存器。
0x0328	T1CON	4	RW	0x0000	定时器1控制MMR。
0x032C	T1CLRI	1	W	N/A	定时器1中断清除寄存器。
0x0340	T2LD[1]	4	RW	0x0050	定时器2载入寄存器。
0x0344	T2VAL	4	R	0x00000050	定时器2值寄存器。
0x0348	T2CON	2	RW	0x0000	定时器2控制MMR。
0x034C	T2CLRI	1	W	N/A	定时器2中断清除寄存器。

[1] 由内核更新。

表17.PLL基地址 = 0xFFFF0400

地址	名称	字节	访问类型	默认值	描述
0x0400	PLLSTA	4	R	N/A	PLL状态MMR。
0x0404	POWKEY0	4	W	N/A	POWCON预写密钥。
0x0408	POWCON	2	RW	0x079	电源控制寄存器。
0x040C	POWKEY1	4	W	N/A	POWCON写后密钥。
0x0410	PLLKEY0	4	W	N/A	PLLCON预写密钥。
0x0414	PLLCON	1	RW	0x00	PLL时钟源选择MMR。
0x0418	PLLKEY1	4	W	N/A	PLLCON写后密钥。
0x0440	OSCCON	1	RW	0x00	低功耗振荡器校准控制MMR。
0x0444	OSCSTA	1	R	0x00	低功耗振荡器校准状态MMR。
0x0448	0SCVAL0	2	R	0x0000	低功耗振荡器校准计数器0 MMR。
0x044C	OSCVAL1	2	R	0x0000	低功耗振荡器校准计数器1 MMR。
0x0480	LOCCON	1	RW	0x00	LIN振荡器校准控制寄存器。
0x0484	LOCUSR0	1	RW	N/A	低功耗振荡器用户调整寄存器。
0x0488	LOCUSR1	2	RW	N/A	精密振荡器用户调整寄存器。
0x048C	LOCMAX	3	RW	0x00000	LIN振荡器校准，最大波特率容差(LINBR + x)。
0x0490	LOCMIN	3	RW	0x00000	LIN振荡器校准，最小波特率容差(LINBR - x)。
0x0494	LOCSTA	1	R	0x01	LIN振荡器校准状态寄存器。
0x0498	LOCVAL0	1	R	N/A	低功耗振荡器当前调整值寄存器。
0x049C	LOCVAL1	2	R	N/A	精密振荡器当前调整值寄存器。
0x04A0	LOCKEY	2	W	N/A	LIN振荡器校准锁定寄存器。

表18. ADC基地址 = 0xFFFF0500

地址	名称	字节	访问类型	默认值	描述
0x0500	ADCSTA	2	R	0x0000	ADC状态MMR。
0x0504	ADCMSKI	1	RW	0x00	ADC中断源使能MMR。
0x0508	ADCMDE	1	RW	0x00	ADC模式寄存器。
0x050C	ADC0CON	2	RW	0x0002	电流ADC控制MMR。
0x0510	ADC1CON	2	RW	0x0000	V/T ADC控制MMR。
0x0518	ADCFLT	2	RW	0x0007	ADC滤波器控制MMR。
0x051C	ADCCFG	1	RW	0x00	ADC配置MMR。
0x0520	ADC0DAT	2	R	0x0000	电流ADC结果MMR。
0x0524	ADC1DAT	2	R	0x0000	V/T ADC结果MMR。
0x0530	ADC0OF[1]	2	RW	N/A	电流ADC失调MMR。
0x0534	ADC1OF[1]	2	RW	N/A	电压ADC失调MMR。
0x0538	ADC2OF[1]	2	RW	N/A	温度ADC失调MMR。
0x053C	ADC0GN[1]	2	RW	N/A	电流ADC增益MMR。
0x0540	ADC1GN[1]	2	RW	N/A	电压ADC增益MMR。
0x0544	ADC2GN[1]	2	RW	N/A	温度ADC增益MMR。
0x0548	ADC0RCL	2	RW	0x0001	电流ADC结果计数限值。
0x054C	ADC0RCV	2	R	0x0000	电流ADC结果计数值。
0x0550	ADC0TH	2	RW	0x0000	电流ADC结果阈值。
0x055C	ADC0ACC	4	R	0x00000000	电流ADC结果累加器。

[1] 由内核更新。

表19. LIN基地址 = 0xFFFF0700

地址	名称	字节	访问1类型	默认值	描述
0x0700	LINCON	2	RW	0x0000	LIN控制MMR。
0x0704	LINCS	1	RW	0xFF	LIN校验和MMR。
0x0708	LINBR	3	RW	0x00FA0	19位LIN波特率MMR。
0x070C	LINBK	3	RW	0x0157C	19位LIN断开MMR。
0x0710	LINSTA	2	R	0x0100	LIN状态MMR。
0x0714	LINDAT	1	RW	0x00	LIN数据MMR。
0x0718	LINLOW	3	RW	0x00000	强制拉低总线的LIN计数器。
0x071C	LINWU	3	RW	0x00013	LIN唤醒断开长度。

表20. 高压接口基地址 = 0xFFFF0804

地址	名称	字节	访问类型	默认值	描述
0x0804	HVCON	1	RW	N/A	高压接口控制MMR。
0x080C	HVDAT	1	RW	N/A	高压接口数据MMR。

表21. SPI基地址 = 0xFFFF0A00

地址	名称	字节	访问类型	默认值	描述
0x0A00	SPISTA	2	R	0x0000	SPI状态MMR。
0x0A04	SPIRX	1	R	0x00	SPI接收MMR。
0x0A08	SPITX	1	W	N/A	SPI发送MMR。
0x0A0C	SPIDIV	1	R/W	0x00	SPI波特率选择MMR。
0x0A10	SPICON	2	R/W	0x0000	SPI控制MMR。

表22. GPIO基地址 = 0xFFFF0D00

地址	名称	字节	访问类型	默认值	描述
0x0D00	GPCON	4	RW	0x00000000	GPIO端口控制MMR。
0x0D10	GPDAT[1]	4	RW	0x000000FF	GPIO端口数据控制MMR。
0x0D14	GPSET	4	W	N/A	GPIO端口数据设置MMR。
0x0D18	GPCLR	4	W	N/A	GPIO端口数据清零MMR。

[1] 取决于外部GPIO引脚上的电平。

表23. Flash/EE基地址 = 0xFFFF0E00

地址	名称	字节	访问类型	默认值	描述
0x0E00	FEESTA	2	R	0xXXX0	Flash/EE状态MMR
0x0E04	FEEMOD	2	RW	0x0000	Flash/EE控制MMR。
0x0E08	FEECON	1	RW	0x07	Flash/EE控制MMR。
0x0E0C	FEEDAT	2	RW	0x0000	Flash/EE数据MMR。
0x0E10	FEEADR[1]	2	RW	0xF009	Flash/EE地址MMR。
0x0E18	FEESIG	3	R	N/A	Flash/EE LFSR MMR。
0x0E1C	FEEPRO	4	RW	0x00000000	Flash/EE保护MMR。
0x0E20	FEEHID	4	RW	0xFFFFFFFF	Flash/EE保护MMR。

[1] 由内核更新。

16位Σ-Δ型模数转换器

ADuC7039内置两个独立的Σ-Δ型模数转换器(ADC)：电流通道ADC (I-ADC)和电压/温度通道ADC (V/T-ADC)。这些精确测量通道集成了衰减器、片内缓冲、可编程增益放大器、16位Σ-Δ调制器和数字滤波，以便在12V汽车电池系统中精确测量电流、电压和温度变量。

电流通道ADC (I-ADC)

I-ADC转换外部100 μΩ分流电阻检测的电池电流。片内可编程增益意味着I-ADC可处理高达±1500 A的电池电流。

如图11所示，I-ADC采用Σ-Δ转换技术来实现16位无失码性能。

Σ-Δ调制器将采样输入信号转换成占空比包含数字信息的数字脉冲序列，然后使用一个修改的Sinc3可编程、低通滤波器来抽取调制器输出数据流以提供有效的16位数据转换结果。在正常和低功耗模式下，输出速率可在10 Hz至1 kHz范围内编程。

I-ADC还内置了计数器、比较器和累加器逻辑，从而允许I-ADC结果在预定转换数目或I-ADC结果超过可编程阈值后产生中断。使能此功能后，32位累加器会自动对16位I-ADC结果求和。

在电流通道上获得第一个有效(完全稳定)结果的时间为斩波模式关闭时的三个ADC转换周期或斩波模式开启时的两个ADC转换周期。使能ADC连续中断选项后，即使ADC采样未建立，也能产生中断。

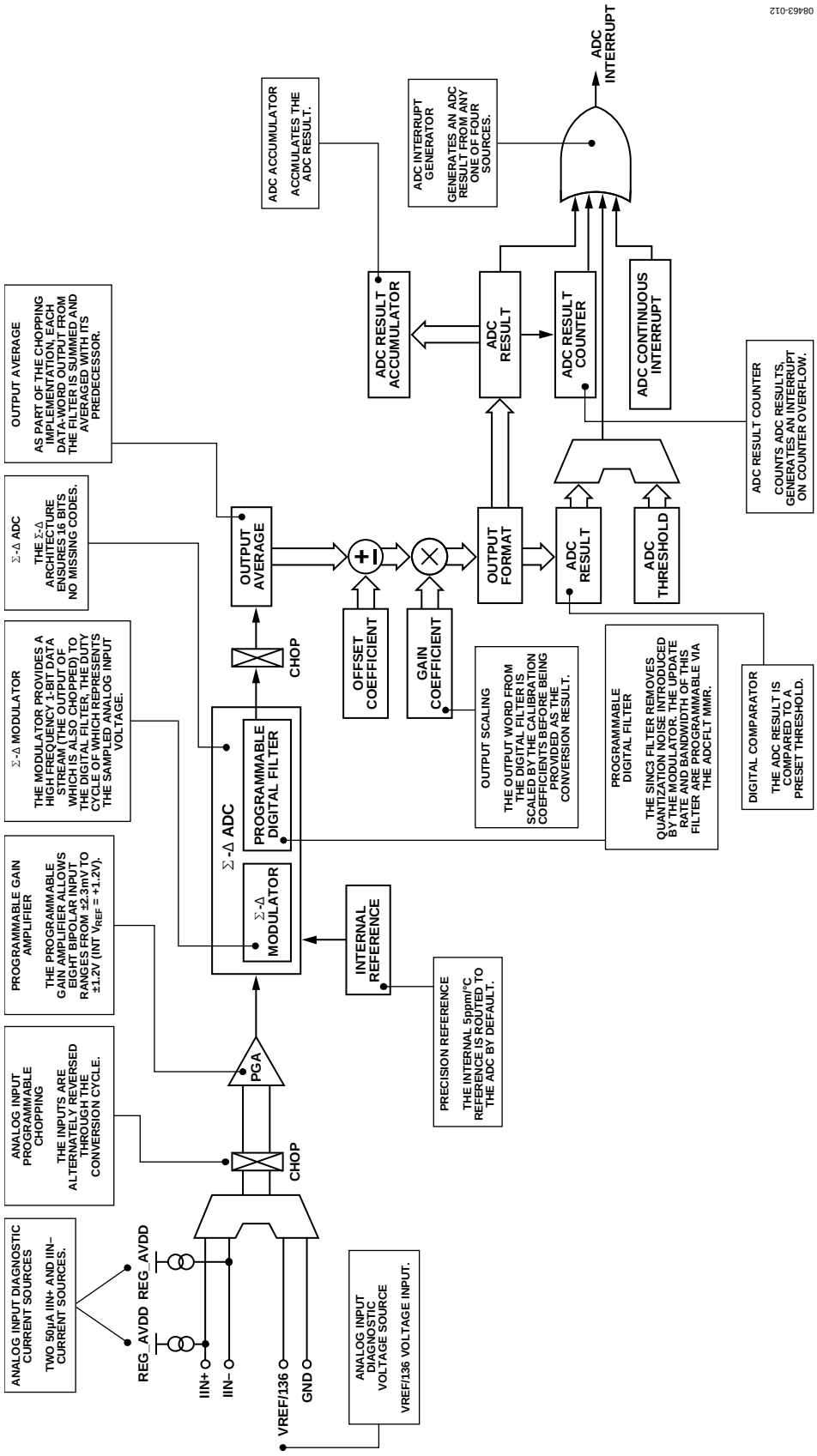


图11. 电流ADC顶层概览

电压/温度通道ADC (V/T-ADC)

电压/温度通道ADC (V/T-ADC)转换电压和温度等其它电池参数。此通道的输入可多路复用两个输入源中的一个：外部电压和片内温度传感器。

与上述电流通道ADC一样，V/T-ADC运用相同的Σ-Δ转换技术，包括一个修改的Sinc3低通滤波器以在10 Hz至1 kHz的可编程输出速率下提供有效的16位数据转换结果。由于已在电压通道内实现滤波，因而不需要外部RC滤波器网络。

外部电池电压(VBAT)通过一个片内高压(24分压)电阻衰减器送到ADC输入端。与电流通道不同，该ADC通道的VBAT具有固定的0 V至28.8 V输入范围。

电池温度可通过片内温度传感器获得。

默认情况下，在电压/温度通道上切换输入通道后获得第一个有效(完全稳定)结果的时间为斩波模式关闭时的三个ADC转换周期。

ADC信号链顶层概览如图12所示。

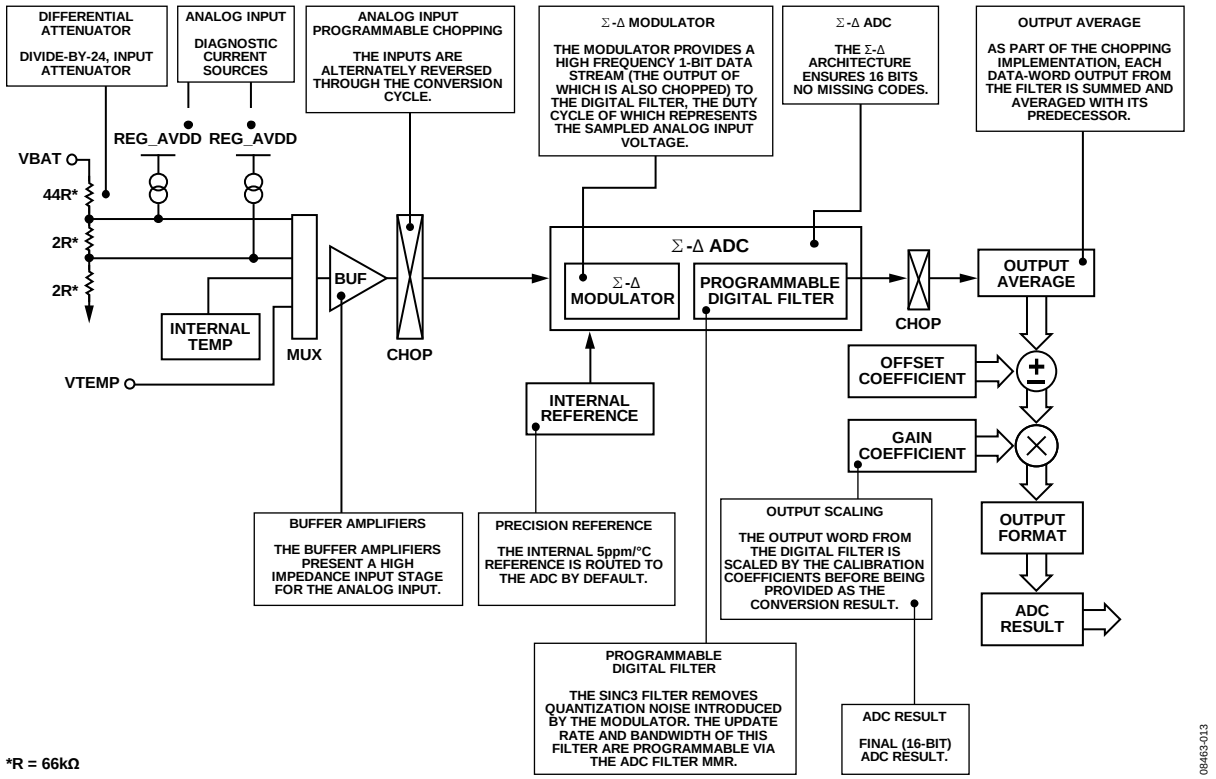


图12. 电压/温度ADC顶层概览

ADC接地开关

ADuC7039集成一个接地开关引脚GND_SW(引脚9)。通过这个开关，用户可以将外部设备与地动态断开，代之以20 kΩ电阻接地，从而减少NTC电路所需的外部元件数目，如图13所示。接地开关功能有利于降低特定应用电路板的功耗。

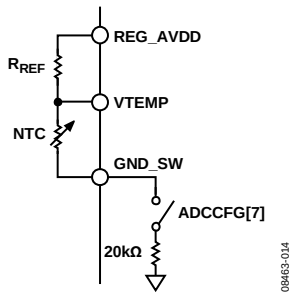


图13. 内部接地开关配置

ADC噪声性能表

表24、25及26列出了I-ADC和V/T-ADC在某些典型输出更新速率下的输出均方根噪声(单位为毫伏)。这些数据都是典型值，分别在0 V (I-ADC)、4V (V-ADC)和0.1 V (T-ADC)的差分输入电压下产生。输出均方根噪声是指ADC输入直流信号时输出代码分布的标准差(或1 Σ)，单位为μV rms。

表24. 电流通道ADC在正常功耗模式下的典型输出均方根噪声

		ADC输入范围		
ADCFLT	数据更新速率	±2.3 mV (512)	±37.5 mV (32)	±300 mV (4)[1]
0x961F	10 Hz	0.065 μV	0.087 μV	0.7 μV
0x007F	50 Hz	0.144 μV	0.170 μV	0.7 μV
0x0007	1 kHz	0.663 μV	0.780 μV	2.6 μV

[1] 最大绝对输入电压允许范围：对地−200 mV至+300 mV。

表25. 电压通道ADC的典型输出均方根噪声(折合到ADC电压衰减器输入端)

ADCFLT	数据更新速率	28.8 V ADC输入范围
0x961F	10 Hz	65 μV
0x007F	50 Hz	65 μV
0x0007	1 kHz	180 μV

表26. 温度通道ADC的典型输出均方根噪声

ADCFLT	数据更新速率	0 V至1.2 V ADC输入范围
0x961F	10 Hz	2.8 μV
0x007F	50 Hz	2.8 μV
0x0007	1 kHz	7.5 μV

ADC MMR接口

ADC通过许多MMR进行控制和配置，下面将详细说明这些寄存器。

ADCSTA MMR的所有高8位(位[15:8])只用作标志位，不产生中断。此MMR的低8位(位[7:0])进行逻辑或运算，以向MCU内核产生一个ADC中断。为了响应ADC中断，用户代码应查询ADCSTA MMR以确定中断源。如"ADC中断屏蔽寄存器"部分所述，每个ADC中断源都可以通过ADCMSKI MMR单独屏蔽。

所有ADC结果就绪位在ADC0DAT MMR读取后即被清0。如果电流通道ADC没有使能，那么所有ADC结果就绪位在读取ADC1DAT MMR后被清0。为了确保I-ADC和V/T-ADC转换数据能够同步，用户代码应先读取ADC1DAT MMR，然后读取ADC0DAT MMR。只有先将对应ADC结果就绪位清0，才能将新ADC转换结果写入到ADCxDAT MMR。这条规则的唯一例外是当ARM内核关断并且ADC子系统活动时。在此模式下，即使就绪位没有被清0，ADCxDAT寄存器仍始终保存最近一次的ADC转换结果。

ADC状态寄存器

名称： ADCSTA

地址： 0xFFFF0500

默认值： 0x0000

访问类型： 只读

功能： 该只读寄存器保持与ADC工作模式或电流状态相关的一般状态信息。

表27. ADCSTA MMR位分配

位	描述
15	ADC校准状态。 硬件自动将该位置1，表示一个ADC校准周期已完成。 写入下列任一寄存器后，该位自动清0：ADCMDE、ADCFLT或ADC0CON。
14	保留。
13	ADC电压/温度转换错误。 硬件自动将该位置1，表示电压转换超量程或欠量程。此时转换结果将箝位至负满量程(欠量程错误)或正满量程(超量程错误)。 向ADC1DAT寄存器写入一个有效的(量程范围内的)电压转换结果后，该位自动清0。
12	ADC电流转换错误。 硬件自动将该位置1，表示电流转换超量程或欠量程。此时转换结果将箝位至负满量程(欠量程错误)或正满量程(超量程错误)。 向ADC0DAT寄存器写入一个有效的(量程范围内的)电流转换结果后，该位自动清0。
11至6	未用。这些位保留供未来其它功能使用，不必由用户代码监控。
5	ADC连续中断位。 每次I-ADC转换后，该位自动置1。ADC的结果不一定有效。只有在ADCMDE MMR中使能时，该位才有效。 当用户代码读取ADC0DAT时，该位清0。
4	电流通道ADC比较器阈值。 该位仅在电流通道ADC比较器由ADCCFG寄存器使能的条件下有效。 当I-ADC转换结果的绝对值大于ADC0TH MMR中写入的值时，硬件将该位置1。 重新配置ADC时，硬件自动将该位清0。
3	保留。
2	温度转换结果就绪位。 当温度通道ADC使能时，一旦有效温度转换结果写入温度数据寄存器(ADC1DAT MMR)，硬件就会将该位置1。每次校准结束后也会置1。 当用户代码读取ADC1DAT或ADC0DAT时，该位清0。
1	电压转换结果就绪位。 当电压通道ADC使能时，一旦有效电压转换结果写入电压数据寄存器(ADC1DAT MMR)，硬件就会将该位置1。每次校准结束后也会置1。 当用户代码读取ADC1DAT或ADC0DAT时，该位清0。
0	电流转换结果就绪位。 当电流通道ADC使能时，一旦有效电流转换结果写入电流数据寄存器(ADC0DAT MMR)，硬件就会将该位置1。每次校准结束后也会置1。 当用户代码读取ADC0DAT时，该位清0。

ADC中断屏蔽寄存器

名称： ADCMSKI

地址： 0xFFFF0504

默认值： 0x00

访问类型： 读/写

功能： 该寄存器可单独使能各ADC中断源。此寄存器的位位置与ADCSTA MMR的低8位相同。若某位由用户代码置1，相应中断使能。默认情况下，所有位为0，表示禁用所有ADC中断源。注意，当ADCMSKI[5]置1时，ADCMSKI[2:0]不得置1。

ADC模式寄存器

名称： ADCMDE

地址： 0xFFFF0508

默认值： 0x00

访问类型： 读/写

功能： DC模式寄存器是一个8位寄存器，用于配置ADC子系统的工作模式。

表28. ADCMDE MMR位分配

位	描述
7至6	未用。全部保留供日后使用并应由用户代码写入0。
5	ADC使能连续中断。 通过将该位置1，可使能ADCSTA[5]。 该位默认置0。
4	保留。
3	ADC功耗模式配置。 0 = ADC正常模式。使能后，ADC以正常功耗工作，获得最佳电气性能。 1 = ADC低功耗模式。使能后，ADC以低功耗工作。在此模式下，增益固定为512。
2至0	ADC工作模式配置。 0、0、0 = ADC关断模式。所有ADC电路关断。 0、0、1 = ADC连续转换模式。在此模式下，任何使能的ADC都将连续转换。 0、1、0 = ADC单次转换模式。在此模式下，任何使能的ADC将执行一次转换。ADC在完成单次转换操作后进入空闲模式。一个单次转换需要花费两到三个ADC时钟周期，具体的周期数取决于是否启用了斩波模式。 0、1、1 = ADC空闲模式。在此模式下，ADC完全上电，但保持复位状态。 1、1、0 = ADC系统零电平校准。在此模式下，参照施加于ADC输入引脚的外部零电平电压，对使能的ADC通道进行零电平校准。校准是在用户编程的ADC设置下进行的；因此，与正常单次ADC转换一样，必须等待ADC滤波器完全建立后，校准结果才能完全建立。当ADCSTA[15]置1时，结果(失调系数)在ADCxDAT MMR中提供。用户软件应将转换结果复制到ADCxOF MMR。 1、1、1 = ADC系统满量程校准。在此模式下，参照施加于ADC输入引脚的外部满量程电压，对使能的ADC通道进行满量程校准。当ADCSTA[15]置1时，结果(增益系数)在ADCxDAT MMR中提供。用户软件应将转换结果复制到ADCxGN MMR。 其它 = 保留。

电流通道ADC控制寄存器

名称： ADC0CON

地址： 0xFFFF050C

默认值 0x0002

访问类型： 读/写

功能： 该电流通道ADC控制MMR是一个16位寄存器，用于配置I-ADC。

注意： 如果通过ADC0CON对电流ADC进行重新配置，则电压/温度ADC也会复位。

表29. ADC0CON MMR位分配

位	描述
15	电流通道ADC使能。 该位由用户代码置1时，使能I-ADC。 该位由用户代码清0时，关断I-ADC，并使ADCSTA MMR中的对应ADC就绪位复位至0。
14至13	IIN电流源使能。 00 = 电流源关闭。 01 = 在IIN+上使能50 μA电流源。 10 = 在IIN−上使能50 μA电流源。 11 = 同时在IIN-和IIN+上使能50 μA电流源。
12至10	未用。这些位保留供未来其它功能使用，应写入0。
9	电流通道ADC输出编码。 该位由用户代码置1时，I-ADC输出编码配置为单极性。 该位由用户代码清0时，I-ADC输出编码配置为二进制补码。
8	未用。该位保留供未来其它功能使用，应写入0。
7至6	电流通道ADC输入选择。 00 = IIN+、IIN−。 01 = IIN−、IIN− = 诊断性、内部短路配置。 10 = VREF/136，0 V，诊断性、测试电压，适用于增益设置<512。如果选择(REG_AVDD, AGND)/2基准电压，这种I-ADC输入选择将使用REG_AVDD，而不是VREF，导致将ADC0DAT缩小2倍。 11 = 未定义。
5	保留。
4	电流通道ADC基准电压选择。 0 = 选择内部1.2 V精密基准电压。 1 = 选择(REG_AVDD, AGND)/2。
3至0	电流通道ADC增益选择。注意：标称I-ADC满量程输入电压 =(VREF/增益)。 0001 = I-ADC增益2。此设置仅针对功能进行测试，因此它没有出现在电气规格中。 0010 = I-ADC增益4。 0011 = I-ADC增益8。 0100 = I-ADC增益16。 0101 = I-ADC增益32。 0110 = I-ADC增益64。 0111 = I-ADC增益128。 1000 = I-ADC增益256。 1001 = I-ADC增益512。 其它 = I-ADC增益未定义。

电压/温度通道ADC控制寄存器

名称： ADC1CON

地址： 0xFFFF0510

默认值： 0x0000

访问类型： 读/写

功能： 电压/温度通道ADC控制MMR是一个16位寄存器，用于配置V/T-ADC。如果重新配置这两个ADC，应先写入ADC1CON，再写入ADC0CON，确保这两个ADC同步启动。如果ADC0已经开启并正在转换，而ADC1已经关闭，则首先应开启ADC1，然后禁用再重新使能ADC0，使这两个ADC能够同步启动(这就是ADC启动时间)。

表30. ADC1CON MMR位分配

位	描述
15	电压/温度通道ADC使能。 该位由用户代码置1时，使能V/T-ADC。 该位清0时，关断V/T-ADC。
14至13	VTEMP电流源使能。 0、0 = 电流源关闭。 0、1 = 在VTEMP上使能50 μA电流源。 1、0 = 在GND_SW上使能50 μA电流源。 1、1 = 在VTEMP和GND_SW上使能50 μA电流源。
12至10	未用。这些位保留供未来其它功能使用，不能由用户代码修改。
9	电压/温度通道ADC输出编码。 该位由用户代码置1时，V/T-ADC输出编码配置为单极性。 该位由用户代码清0时，V/T-ADC输出编码配置为二进制补码。
8	未用。该位保留供未来使用，应由用户代码写入0。
7至6	电压/温度通道ADC输入选择。 0、0 = VBAT/24、AGND。选择VBAT衰减器。 0、1 = VTEMP、GND_SW。选择外部温度输入，转换结果写入ADC1DAT。 1、0 = 内部传感器。选择内部温度传感器输入，转换结果写入ADC1DAT。温度梯度为0.33 mV/°C；仅适用于内部温度传感器。 1、1 = 内部短路。短路输入。
5	未用。该位保留供未来使用，应由用户代码写入0。
4	电压/温度通道ADC基准电压选择。 0 = 选择内部1.2 V精密基准电压。 1 = 选择(REG_AVDD, GND_SW)/2。
3至0	未用。这些位保留供未来其它功能使用，不能由用户代码写入0。

ADC滤波器寄存器

名称： ADCFLT

地址： 0xFFFF0518

默认值： 0x0007

访问类型： 读/写

功能： ADC滤波器MMR是一个16位寄存器，用于控制片内ADC的速度和分辨率。

注意： 若ADCFLT被修改，电流和电压/温度ADC会复位。建议通过单次写操作写入此MMR的所有位。

表31. ADCFLT MMR位分配

位	描述
15	斩波使能。 该位由用户置1时，使能所有活动ADC的系统斩波。该位置1后，ADC失调误差和漂移很低，但如果AF = 0，ADC输出速率降为原来的三分之一(见此表中的Sinc3抽取系数位[6:0])。AF > 0时，ADC输出更新速率在斩波开启或关闭时均相同。斩波使能时，建立时间为两个输出周期。
14	移动平均。 该位由用户置1时，使能除二移动平均功能，以便降低ADC噪声。激活斩波功能后，该功能自动使能。斩波不激活时可选，且使能后(斩波不激活时)不会降低ADC输出速率，但建立时间会增加一个转换周期。 该位由用户清0时，禁用移动平均功能。
13至8	平均系数(AF)。 写入这些位的值用来实现可编程一阶Sinc3后置滤波器。平均系数可进一步降低ADC噪声，但同时，ADC的输出速率也会降低(见本表中对Sinc3抽取系数位[6:0]的说明)。
7	Sinc3修正。 该位由用户置1时，可以修正标准Sinc3频率响应，将滤波器阻带抑制提升约5 dB。这可以通过插入另一个陷波(NOTCH2)来实现。该陷波的频率公式如下： $f_{NOTCH2} = 1.333 \times f_{NOTCH}$ 其中，f_{NOTCH}表示响应中第一个陷波所在位置。
6至0	Sinc3抽取系数(SF)[1]。 写入这些位的值(SF)控制Sinc3滤波器的过采样(抽取系数)。以下公式用于计算Sinc3滤波器的输出速度： $f_{ADC} = (512{,}000/([SF + 1] \times 64))$ Hz 当斩波位(位15，斩波使能)的值为0且平均系数(AF)的值也为0时，上述公式可用。这一公式适用于所有小于等于125的SF值。 SF = 126时，强制f_{ADC}为60 Hz。 SF = 127时，强制f_{ADC}为50 Hz。 有关SF(不包括126和127)和AF值的f_{ADC}计算信息，请参见表32。

[1] 由于受数字滤波器内部数据路径的限制，用来产生所需ADC输出速率的Sinc3抽取系数(SF)和平均系数(AF)的组合也有限制。ADC最小更新速率以10 Hz为限。

表32. ADC转换速率和建立时间

斩波使能	平均系数	移动平均	f_{ADC}	$t_{SETTLING}$[1]
否	否	否	$\frac{512{,}000}{[SF+1]\times 64}$	$\frac{3}{f_{ADC}}$
否	否	是	$\frac{512{,}000}{[SF+1]\times 64}$	$\frac{4}{f_{ADC}}$
否	是	否	$\frac{512{,}000}{[SF+1]\times 64\times[3+AF]}$	$\frac{1}{f_{ADC}}$
否	是	是	$\frac{512{,}000}{[SF+1]\times 64\times[3+AF]}$	$\frac{2}{f_{ADC}}$
是	N/A	N/A	$\frac{512{,}000}{[SF+1]\times 64\times[3+AF]+3}$	$\frac{2}{f_{ADC}}$

[1] 在第一个ADC转换结果可用之前，每个ADC需要约60 µs的额外时间。

表33. SF和AF的允许组合

	AF范围		
SF	**0**	**1至7**	**8至63**
0至31	是	是	是
32至63	是	是	否
64至127	是	否	否

ADC配置寄存器

名称： ADCCFG

地址： 0xFFFF051C

默认值： 0x00

访问类型： 读/写

功能： 该8位ADC配置MMR用于控制与片内ADC相关的扩展功能。

表34. ADCCFG MMR位分配

位	描述
7	模拟接地开关使能。 该位由用户软件置1时，外部GND_SW引脚(引脚9)连接到内部20 kΩ接地电阻。 在程序控制下，该位可用于将外部电路和元件接地或断开，从而在不使用外部电路或元件时，将直流功耗降到最低。
6至5	未用。这些位保留供未来其它功能使用，不能由用户代码修改。
4	电流通道ADC累加器使能。 0 = 累加器禁用并复位至0。一次完整的ADC转换(ADCSTA[0]设置两次)完成后，必须禁用累加器，然后再使能累加器，以确保将其复位。 1 = 累加器使能。
3	电流通道ADC比较器使能。 0 = 比较器禁用。 1 = 比较器激活，如果I-ADC转换结果的绝对值|I| ≥ ADC0TH，则中断置位。
2	未用。该位保留供未来使用，应由用户代码写入0。
1	快速温度传感器模式。 该位由用户置1时，使能"快速温度转换模式"部分所述的功能。该位置1时，ADC1CON[7:6]暂时被忽略，以产生内部温度传感器的一个完全建立的结果。这种快速转换模式仅适用于内部温度传感器。 快速转换完成时，用户应将该位清0。
0	电流通道ADC结果计数器使能。 该位由用户置1时，使能结果计数模式。在此模式下，仅当ADC0RCV = ADC0RCL时产生I-ADC中断。这样不仅允许I-ADC持续监控电流，而且在达到预定转换次数后中断MCU内核。使能后，电压/温度ADC也会继续转换，但同样，发生ADC计数器中断时只保存最后一次转换结果(不存储中间V/T-ADC转换结果)。

电流通道ADC数据寄存器

名称： ADC0DAT

地址： 0xFFFF0520

默认值： 0x0000

访问类型： 只读

功能： 该ADC数据MMR保存I-ADC的16位转换结果。如果ADC0转换结果就绪位(ADCSTA[0])置1，则ADC不更新此MMR。MCU读取该MMR后，将清除所有置位的就绪标志(ADCSTA[2:0]和ADCSTA[5])。

电压/温度通道ADC数据寄存器

名称： ADC1DAT

地址： 0xFFFF0524

默认值： 0x0000

访问类型： 只读

功能： 该ADC数据MMR保存V/T-ADC的16位电压(或温度)转换结果。如果电压(或温度)转换结果就绪位(ADCSTA[1]或ADCSTA[2])置1，则ADC不更新此MMR。如果I-ADC未激活，MCU读取此MMR后即清除所有置位的就绪标志(ADCSTA[2:1]和ADCSTA[5])。

电流通道ADC失调校准寄存器

名称： ADC0OF

地址： 0xFFFF0530

默认值： 0x0000

访问类型： 读/写

功能： 该ADC失调MMR保存I-ADC的16位失调校准系数。上电时该寄存器配置为出厂默认值。然而，通过ADCMDE[2:0]执行校准后，用户代码可以用ADC0DAT包含的失调校准结果覆盖此默认值。

电流通道ADC增益校准寄存器

名称： ADC0GN

地址： 0xFFFF053C

默认值： 取决于器件，工厂编程

访问类型： 读/写

功能： 该增益MMR保存用于按比例调整I-ADC转换结果的16位增益校准系数。上电时该寄存器配置为出厂默认值。然而，通过ADCMDE[2:0]执行校准后，用户代码可以用ADC0DAT包含的增益校准结果覆盖此默认值。

电压通道ADC失调校准寄存器

名称： ADC1OF

地址： 0xFFFF0534

默认值： 0x0000

访问类型： 读/写

功能： 该ADC失调MMR保存电压通道的16位失调校准系数。上电时该寄存器配置为出厂默认值。然而，通过ADCMDE[2:0]执行校准后，用户代码可以用ADC1DAT包含的失调校准结果覆盖此默认值。

电压通道增益校准寄存器

名称： ADC1GN

地址： 0xFFFF0540

默认值： 取决于器件，工厂编程

访问类型： 读/写

功能： 该增益MMR保存用于按比例调整电压通道转换结果的16位增益校准系数。上电时该寄存器配置为出厂默认值。然而，通过ADCMDE[2:0]执行校准后，用户代码可以用ADC1DAT包含的增益校准结果覆盖此默认值。

温度通道ADC失调校准寄存器

名称： ADC2OF

地址： 0xFFFF0538

默认值： 0x0000

访问类型： 读/写

功能： 该ADC失调MMR保存温度通道的16位失调校准系数。上电时该寄存器配置为出厂默认值。然而，通过ADCMDE[2:0]执行校准后，用户代码可以用ADC1DAT包含的失调校准结果覆盖此默认值。

温度通道增益校准寄存器

名称： ADC2GN

地址： 0xFFFF0544

默认值： 取决于器件，工厂编程

访问类型： 读/写

功能： 该增益MMR保存用于按比例调整温度通道转换结果的16位增益校准系数。上电时该寄存器配置为出厂默认值。

电流通道ADC结果计数器限值寄存器

名称： ADC0RCL

地址： 0xFFFF0548

默认值： 0x0001

访问类型： 读/写

功能： 该16位MMR设置产生ADC中断前所必需的转换次数。默认情况下，该寄存器的值为0x01。要使能ADC计数器功能，必须将ADCCFG MMR中的ADC结果计数器使能位置1。

电流通道ADC阈值寄存器

名称： ADC0TH

地址： 0xFFFF0550

默认值： 0x0000

访问类型： 读/写

功能： 该16位MMR设置用来与I-ADC转换结果的绝对值进行比较的阈值。在单极性模式下，与ADC0TH[15:0]进行比较；而在二进制补码模式下，与ADC0TH[14:0]进行比较。

电流通道ADC结果计数寄存器

名称： ADC0RCV

地址： 0xFFFF054C

默认值： 0x0000

访问类型： 只读

功能： 该16位只读MMR保存当前的I-ADC转换结果数目。与ADC0RCL配合产生I-ADC中断屏蔽，从而产生较低的中断速率。ADC0RCV = ADC0RCL时，ADC0RCV值复位至0并重新开始计数。还可配合累加器(ADC0ACC)使用以计算电流均值。通过设置ADCCFG[0]，可以使能结果计数器。当重新配置I-ADC时，即当写入ADC0CON或ADCMDE时，此MMR复位至0。

电流通道ADC累加器寄存器

名称： ADC0ACC

地址： 0xFFFF055C

默认值： 0x00000000

访问类型： 只读

功能： 该32位MMR保存电流累加器值。应使用ADCSTA MMR的I-ADC就绪位来确定何时可以安全地读取MMR。在ADCCFG MMR内禁用累加器或重新配置电流通道ADC时复位至0。

ADC Sinc3数字滤波器响应

所有ADuC7039 ADC的整体频率响应主要取决于片内Sinc3数字滤波器的低通滤波器响应。Sinc3滤波器的作用是抽取ADC Σ-Δ调制器输出数据位流以产生有效的16位数据结果。两个ADC的数字滤波器响应完全相同，并通过16位ADC滤波器寄存器(ADCFLT)配置。这个寄存器决定ADC的总吞吐速率。编程的ADC吞吐速率决定ADC的噪声分辨率。对于电流通道ADC，吞吐速率和所选增益决定噪声分辨率。

整体频率响应和吞吐速率主要取决于Sinc3滤波器抽取系数(SF)位(ADCFLT[6:0])和平均系数(AF)位(ADCFLT[13:8])的配置。由于数字滤波器内部数据路径有限，可以用来产生所需输出速率的SF和AF组合也有限制，正常功耗模式下，ADC最小更新速率以10 Hz为限。ADC吞吐速率的计算方法详见ADCFLT位分配表，而AF和SF值的允许组合限制见表33。

默认情况下，ADCFLT = 0x0007时吞吐速率配置为1.0 kHz，同时禁用所有其它滤波器选项(斩波、移动平均、平均系数及Sinc3修正)。此时典型滤波器响应如图14所示。

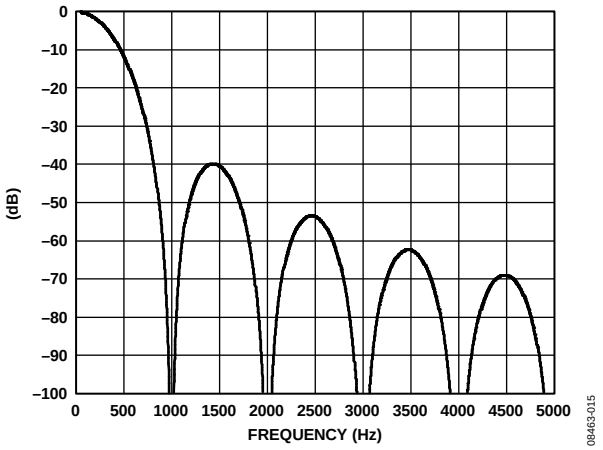


图14. f_{ADC} = 1.0 kHz (ADCFLT = 0x0007)时的典型数字滤波器响应

此外，ADCFLT寄存器还提供了一个附加Sinc3修正位(ADCFLT[7])。该位由用户代码置1时，可以修正标准Sinc3频率响应，进而将滤波器阻带抑制提升约5 dB。这可以通过插入另一个陷波(NOTCH2)来实现。该陷波的频率公式如下：

$$f_{NOTCH2} = 1.333 \times f_{NOTCH}$$

其中，f_{NOTCH}表示响应中第一个陷波所在位置。

该位有效时，ADC噪声会略微增加。图15显示Sinc3修正位有效时修正过的1 kHz滤波器响应。同标准1 kHz响应相比，与阻带抑制提升一样，新陷波清楚地出现在1.33 kHz处。

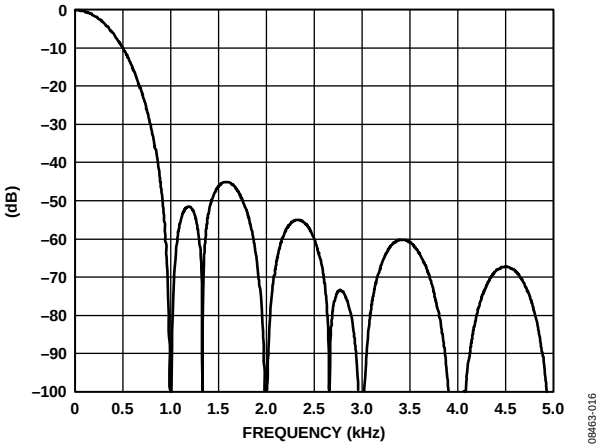


图15. f_{ADC} = 1.0 kHz (ADCFLT = 0x0087)时的修正后Sinc3数字滤波器响应

吞吐速率极低时，可使能ADCFLT寄存器内的斩波位，从而最大程度降低失调误差，更重要的是最大程度降低失调误差的温度漂移。

用户可使用两个主要变量(Sinc3抽取系数和平均系数)来选择最佳滤波器响应，但需要权衡滤波器带宽和ADC噪声。

例如，如果斩波位(ADCFLT[15])置1，SF值(ADCFLT[6:0])增加至0x1F(十进制31)且AF值(ADCFLT[13:8])选择为0x16(十进制22)，则产生10 Hz的ADC吞吐速率。此时的频率响应曲线如图16所示。

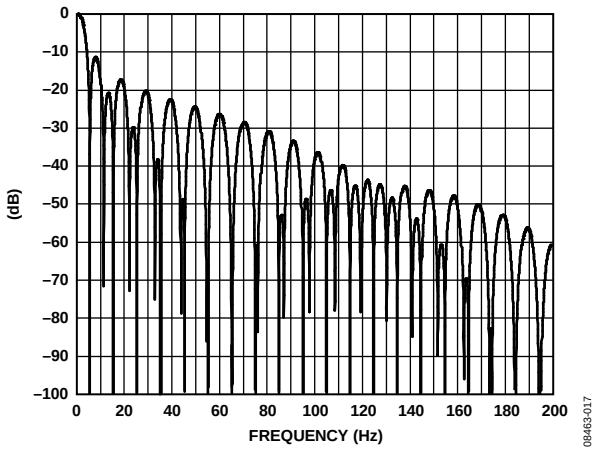


图16. f_{ADC} = 10 Hz, (ADCFLT = 0x961F)时的典型数字滤波器响应

在ADC低功耗模式下，ADC、Σ-Δ调制器时钟不再由512 kHz驱动，而直接采用片内低功耗128 kHz振荡器驱动。而在正常模式下，对于相同ADCFLT配置，所有滤波器值都应按比例缩小约4倍。

一般情况下，相同的ADC更新速率可通过在ADCFLT寄存器内设置不同的SF和AF值来获得。但实际上，不管ADCFLT的值如何，用户都要权衡频率响应和ADC噪声。要利用SF和AF组合获得最佳滤波器响应和ADC噪声，经验表明SF应选在16到40(十进制)或0x10到0x28之间，然后递增AF值直到获得所需ADC吞吐速率。表35列举了一些常见的ADCFLT配置。

ADC工作模式

通过相应地配置ADCMDE[3]，可以将ADC配置为低功耗或全功率(正常)工作模式。ARM7 MCU也可以配置为低功耗工作模式(POWCON[5:3])。内核功耗模式是独立控制的，和本部分所述ADC功耗模式无关。

ADC正常功耗模式

正常模式下，电流和电压/温度通道全部使能。ADC调制器时钟为512 kHz，使得ADC能够以10 Hz到1 kHz的速率提供正常转换结果(参见"ADC滤波器寄存器"部分)。两个通道都由MCU完全控制，可随时重新配置。所有通道的默认ADC更新速率为1.0 kHz。

需要强调的是，I-ADC和V/T-ADC通道可以在返回ADC完全关断模式前，启动定期性正常功耗模式的高精度单次转换周期。此配置由MCU通过ADCMDE MMR完全控制，更加灵活，从而实现电池电流、电压和温度设置的持续周期性监控，同时将平均直流功耗降到最低。

在ADC正常模式下，PLL不得关断。

ADC低功耗模式

在ADC低功耗模式下，I-ADC的功耗和精度都会降低。此时，ADC调制器时钟直接采用片内128 kHz低功耗振荡器驱动。在此模式下，ADC的增益固定为512。

所有ADC外设功能(结果计数器、数字比较器和累加器)均可在低功耗模式下使能。

在低功耗模式下，一般只有I-ADC以较低更新速率运行，持续监控电池电流。MCU处于关断模式，在I-ADC产生中断时唤醒。当I-ADC检测到当前转换结果超过预设阈值、设定点或设定次数时，就会产生中断。

ADC比较器和累加器

每个I-ADC结果也可以与ADCCFG[3]配置的预设阈值(ADC0TH)进行比较。如果ADC结果的绝对值(符号无关)大于预设比较器阈值，就会产生MCU中断。

最后，可配置32位累加器(ADC0ACC)功能(ADCCFG[5])，使I-ADC加上(或减去)多个采样结果。用户代码可直接读取累加值(ADC0ACC)，不需要任何进一步软件处理。

ADC连续中断模式

将ADCMDE[5]置1时，每个ADC转换周期结束时都会产生一个ADC中断，即使ADC滤波器尚未完全建立。要使能连续中断模式，ADCMSKI中的相应ADC中断位也必须置1。在此模式下，ADCSTA[2:0]用作有效标志，不应用来产生中断。

表35. 常见ADCFLT配置

ADC模式	SF	AF	其它配置	ADCFLT	f_{ADC}	t_{SETTLE}
正常	0x1F	0x16	斩波开启	0x961F	10 Hz	0.2 sec
正常	0x07	0x00	无	0x0007	1 kHz	3 ms
正常	0x07	0x00	Sinc3修正	0x0087	1 kHz	3 ms
低功耗	0x10	0x03	斩波开启	0x8310	20 Hz	100 ms
低功耗	0x10	0x09	斩波开启	0x8910	10 Hz	200 ms

ADC校准

如顶层框图(图11及12)详细所示，通过所有ADC通道的信号流描述如下：

1. 输入电压通过输入缓冲器(对于I-ADC，还要通过PGA)施加到Σ-Δ调制器。
2. 调制器输出施加到可编程数字抽取滤波器。
3. 如果使用斩波，则对滤波器输出结果进行平均。
4. 将偏移值(ADCxOF)从上述结果中减去。
5. 根据增益值(ADCxGN)按比例调整结果。
6. 最后，将结果格式化为二进制补码/单极性形式，并舍入到16位或箝位到正负满量程。

每个ADC通道(电流、电压和温度)都具有一个特定的失调和增益校正或校准系数与之相对应，这些系数存储在MMR的失调和增益寄存器(ADCxOF和ADCxGN)内。可以利用失调和增益寄存器来消除器件之外的系统失调和增益误差。

这些寄存器在上电时加载工厂预设校准值。这些工厂校准值因器件而异，反映了ADC内部电路的工艺差异。校准后，用户代码也可以覆盖这些寄存器。

在电流通道上启动系统校准时，ADC根据外部产生的零电平电压和满量程电压产生校准系数，这些电压在校准周期内施加于外部ADC输入。系数写入ADC通道的ADC0DAT MMR，而不是自动写入ADC0OF或ADC0GN MMR。用户代码必须将这些值复制到相应的寄存器。

失调校准的持续时间为ADC滤波器完全建立的时间，然后ADC返回空闲模式。校准周期启动后，任何进行中的ADC转换都会立即停止，校准以ADCFLT中设置的ADC更新速率自动进行，结束后ADC始终返回空闲模式。强烈建议以尽可能低的ADC更新速率来启动校准(ADCFLT中的SF值较高)，这样在校准期间可以尽可能降低ADC噪声的影响。

在电压通道上，必须执行两点校准，因为此输入的最小额定电压为4 V。

温度通道针对内部温度传感器经过工厂校准。

电压通道校准

要校准电压通道的失调和增益，必须采用两点校准方法。这种方法转换两个已知电压(例如8 V和16 V)，从而确定传递函数的斜率和偏移。增益系数可以除以计算得到的斜率，从而降低增益误差。

将计算得到的偏移的⅜(单极性模式)写入ADC1OF MMR，可以降低失调误差。

电流通道校准

使能斩波位(ADCFLT[15])后，ADC内部失调误差已降到最低，可能不需要进行失调校准。但如果斩波禁用，就需要进行初始失调校准，甚至可能需要反复校准，尤其是在发生较大温度变化之后。

视系统精度要求，可能需要在所有相关系统增益范围内进行增益校准，尤其是对于I-ADC(带内部PGA)。如果无法在所有增益范围内施加外部满量程电流，那么可以施加一个较低电流，然后按比例调整校准结果。例如，施加一个50%电流，然后将所产生的ADC0DAT值除以2并写回到ADC0GN。注意，由于ADC0GN只有16位，系统校准期间可以施加的输入信号是有下限的。输入范围(即系统零电平值和满量程值之间的差值)应大于标称满量程输入范围的40%(即>40%的VREF/增益)。

片内Flash/EE存储器可以存储多个校准系数。这些校准系数可根据系统配置由用户代码直接复制到相关校准寄存器内。

对于I-ADC，工厂设置或下线校准分两步：

1. 施加0 A电流。根据所需PGA设置等条件配置ADC，并写入ADCMDE[2:0]以进行系统零电平校准，从而将新失调校准值写入ADC0DAT内。用户代码必须将此值存储在ADC0OF或Flash/EE存储器中。
2. 针对所选PGA设置施加满量程电流。写入ADCMDE，进行系统满量程校准，从而将新增益校准值写入ADC0DAT内。用户软件必须将此值复制到ADC0GN MMR或Flash/EE存储器。

了解失调和增益校准寄存器

ADC信号流中的平均模块输出可视为小数且正负满量程输入范围约为±0.75。此范围小于±1.0，这是因为调制器有衰减，使输入信号具有一定的超量程能力。由于工艺容差，确切的衰减值因器件而略有不同。

失调系数读取自ADC0OF校准寄存器，此值是一个16位二进制补码。就信号链而言，此值的有效范围为±1.0。因此，ADC0OF寄存器的1 LSB和ADC0DAT寄存器的1 LSB不一样。

ADC0OF正值表示从滤波器输出中减去失调，负值则相加。此寄存器的标称值为0x0000，表明要消除零失调。实际失调可能因器件、PGA增益而异。启用斩波模式(即ADCFLT[15] = 1)可以将ADC内部失调降到最低。

增益系数是一个无单位比例因子，此寄存器的16位值除以16,384后再乘以失调校正值。此寄存器的标称值为0x5555，对应于乘法因子1.3333。这将把标称±0.75信号放大到±1.0的满量程输出信号，在进行上溢/下溢检查后转换成二进制补码或单极性模式，然后输出到数据寄存器。

实际增益和零增益误差所需的调整系数因器件、PGA设置而略有差异。上电复位时下载到ADC0GN的值对应PGA增益为4的比例因子。如果在不同的PGA设置下使用此值，可能会产生一定的增益误差。用户代码可以运行ADC校准并覆盖校准系数，以校正当前PGA设置的增益误差。

总之，简化后的ADC传递函数可描述为

$$ADC_{OUT} = \left[\frac{V_{IN} \times PGA}{V_{REF}} - ADCOF\right] \times \frac{ADCGN}{ADCGN_{NOM}}$$

此公式对于电压/温度通道ADC有效。

对于电流通道ADC，

$$ADC_{OUT} = \left[\frac{V_{IN} \times PGA}{V_{REF}} - K \times ADCOF\right] \times \frac{ADCGN}{ADCGN_{NOM}}$$

其中*K*取决于PGA增益设置和ADC模式。

PGA增益为4和32时，系数K为1。PGA增益为512时，系数K为8。

ADC配置

快速温度转换模式

电池温度可通过片内温度传感器获得。默认情况下，ADC输入从电压通道切换到温度通道或从温度通道切换到电压通道后，获得第一个有效(完全稳定)结果的时间为斩波模式关闭时的三个ADC转换周期，如图17所示。

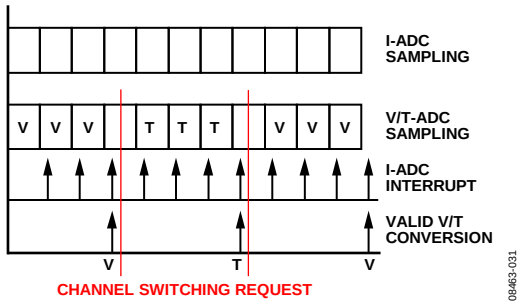


图17. 默认温度模式，斩波关闭

温度通道提供一种快速模式，用于尽可能缩短电压转换和温度转换之间的切换延迟时间，如图18和表36所示。

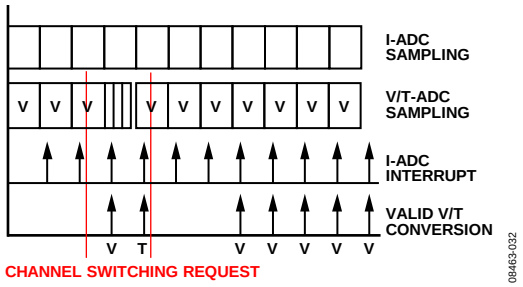


图18. 快速温度模式，斩波关闭(ADCFLT = 0x07)

请求进行快速温度转换时，会延迟一个ADC转换周期。温度测量结果可用之后、请求进行新温度转换之前，必须退出快速温度转换模式。

表36. 快速温度转换模式

中断	有效标志	用户代码
1	I和V	电压 = ADC1DAT
2	I和V	电压 = ADC1DAT
3	I和V	电压 = ADC1DAT 必须读取此数据，之后下一个温度通道标志才会有效。
4	I和T	温度 = ADC1DAT 快速温度请求位清0
5	I	
6	I	
7	I和V	电压 = ADC1DAT
8	I和V	电压 = ADC1DAT

快速温度选项不能用在ADC上电后的第一次转换上，至少必须在第一个ADC中断后才能设置，但不必等待一个有效的ADC结果出现。使用快速温度模式时，还有一个限制是SF >=1。此外，在此工作模式下，建议使用1 ms的转换速率，这是为了确保快速温度结果与当前通道结果同时出现。

写入ADCMDE、ADC0CON或ADCFLT以更改ADC配置时，快速温度位也必须清0，以确保器件正常工作。这种情况与ADC上电后的第一次转换相似。

I-ADC诊断

ADuC7039具有检测应用电路板开路状态的功能。这是通过ADC0CON[14:13]控制IIN+和IIN−上的两个电流源来实现的。

注意，这些电流源的容差为±30%。

电源支持电路

ADuC7039集成两个片内低压差(LDO)调节器，它们通过电池电压直接驱动来产生一个2.6 V内部电源。然后，该2.6 V电源用作ARM7 MCU和外设的电源，包括那些片内精密模拟电路。

数字LDO利用REG_DVDD上并联的两个外部电容工作，模拟LDO则利用REG_AVDD上的一个外部电容(0.47 μF)工作。

输出电容的ESR会影响LDO控制环路的稳定性。频率大于32 kHz时，建议使用ESR为5 Ω或以下的电容以确保调节器的稳定性。

此外还集成了上电复位(POR)及低压标志(LVF)功能，以确保MCU安全运行以及连续监控电池电源。POR电路的设计VDD(0 V至12 V)上电时间大于100 μs。因此，选择外部电源去耦元件时务必小心，确保VDD电源上电时间始终大于100 μs。VDD上的串联电阻和去耦电容组合应确保RC时间常数至少为100 μs，例如10 Ω和10 μF，如图33所示。

如图19所示，当VDD上的电源电压达到最小工作电压3 V时，POR信号将ARM内核保持在复位状态下20 ms。这样可以确保施加到ARM内核及相关外设的调节电源电压(REG_DVDD)大于最小工作电压，由此保证功能完整。RSTSTA MMR中的POR标志位置1，指示发生了POR复位事件。

电压低于POR电平时，可使能另一个低压标志(HVCFG [2])。它可用来指示SRAM的内容在复位事件后仍然有效。低压标志的示例如图19所示。使能后，可通过HVSTA[2]监控该位的状态。如果该位置1，则SRAM内容有效。如果该位清0，则SRAM内容可能遭到破坏。

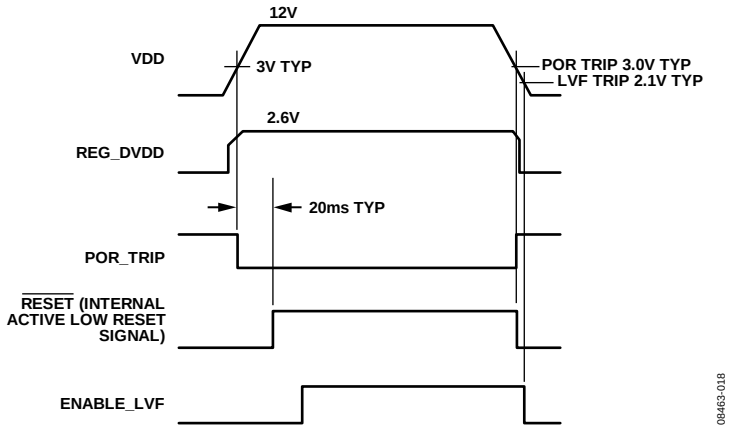


图19. 典型上电周期

系统时钟

ADuC7039集成一个灵活的时钟系统，其时钟源可以是下面两个集成的片内振荡器之一：精密振荡器和低功耗振荡器。

每个内部振荡器除以4即可产生32 kHz的时钟频率。两个内部振荡器之一提供32 kHz，然后PLL锁定在它的倍数(640)上，为系统提供稳定的20.48 MHz时钟。默认情况下，PLL由低功耗振荡器驱动。

内核默认以10.24 MHz的固定频率工作。

ADC由PLL输出驱动，PLL输出经分频后提供512 kHz的ADC时钟源。在低功耗模式下，ADC时钟源会从标准512 kHz切换到低功耗128 kHz振荡器。

注意，低功耗振荡器通过一个4分频电路来同时驱动看门狗和内核唤醒定时器。ADuC7039时钟系统的详细框图如图20所示。

两个集成的振荡器可以按照“振荡器校准”部分所述进行校准。

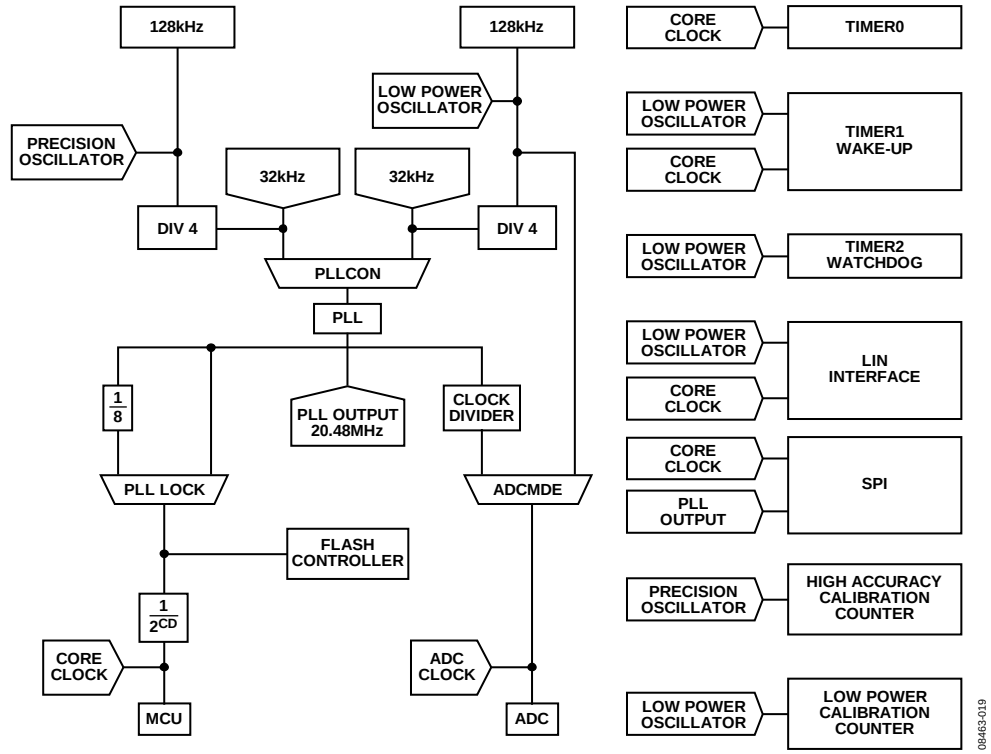


图20. ADuC7039系统时钟发生

工作模式和时钟模式由两个MMR控制(PLLCON和POWCON)，PLL状态(PLL锁定和PLL中断)由PLLSTA指示。

关断ADuC7039之前，建议将PLL时钟源切换到低功耗振荡器以缩短唤醒时间。低功耗振荡器始终有效。

当ADuC7039从关断模式中唤醒时，PLL一开始振荡，MCU内核就会执行代码。这发生在PLL锁定到20.48 MHz的频率之前。为了确保Flash/EE存储器控制器获得有效的时钟，当PLL正在锁定时，该控制器通过PLL输出8分频时钟源来驱动。PLL锁定后，PLL输出才从这个8分频切换到锁定的输出。

如果用户代码需要精确的PLL输出，用户代码必须在唤醒后、恢复正常代码执行前轮询锁定位(PLLSTA[1])。

PLL在唤醒后2 ms内锁定。

PLLCON为采用两个32位密钥保护的MMR：PLLKEY0(预写密钥)和PLLKEY1(写后密钥)。

PLLKEY0 = 0x000000AA

PLLKEY1 = 0x00000055

PLLCON为采用两个32位密钥保护的MMR：POWKEY0(预写密钥)和POWKEY1(写后密钥)。

POWKEY0 = 0x00000001

POWKEY1 = 0x000000F4

序列示例

写入两个MMR的示例如下：

写入POWCON MMR：

//Function Prototype

```
Void PowerDown (void)

PowerDown PROC
  LDR    r2, = 0x98765432       ; Load random number for multiplication
  LDR    r3, = 0x12345678
  LDR    r0, = 0xffff0400       ;Base address
  MOVS   r1,#0x1                ;POWKEY0 = 1
  STR    r1,[r0,#4]             ;Set POWKEY0
  MOVS   r1,#0x01               ;Set POWCON value to recommended value of 0x01 to ensure a
                                 10 MHz core clock
  STR    r1,[r0,#8]
  MOVS   r1,#0xf4
  STR    r1,[r0,#0xc]           ;Set POWKEY1
  UMLAL  r1,r3,r2,r0            ;longest possible assembly multiplication instruction
  BX     lr                     ;Flush ARM7 pipeline
ENDP
```

写入PLLCON MMR：

```
PLLKEY0      =     0xAA   //PLLCON key
PLLCON       =     0x1    //Switch to precision oscillator.
PLLKEY1      =     0x55   //PLLCON key
iA1*iA2                   //PSEUDOCODE-dummy cycle to prevent Flash/EE access during clock
change
```

PLLSTA寄存器

名称： PLLSTA

地址： 0xFFFF0400

默认值： 0xXX

访问类型： 只读

功能： 该8位寄存器允许用户代码监控PLL的锁定状态。

表37. PLLSTA MMR位分配

位	描述
7至2	保留。
1	PLL锁定状态位，只读。 PLL锁定并输出20.48 MHz时，该位自动置1。 PLL未锁定并输出f_{CORE} 8分频时钟源时，该位自动清0。
0	PLL中断。 PLL锁定状态位信号变为低电平时，该位置1。 用户代码写入1到该位可清0。

PLLCON预写密钥PLLKEY0

名称： PLLKEY0

地址： 0xFFFF0410

访问类型： 只写

密钥： 0x000000AA

功能： PLLCON是一个加密寄存器，PLLCON前后需写入32位密钥值。PLLKEY0为预写密钥。

PLLCON写后密钥PLLKEY1

名称： PLLKEY1

地址： 0xFFFF0418

访问类型： 只写

密钥： 0x00000055

功能： PLLCON是一个加密寄存器，PLLCON前后需写入32位密钥值。PLLKEY1为写后密钥。

PLLCON寄存器

名称： PLLCON

地址： 0xFFFF0414

默认值： 0x00

访问类型： 读/写

功能： 该8位寄存器允许用户代码从两种不同振荡器源中动态选择PLL时钟源。

表38. PLLCON MMR位分配

位	描述
7至1	保留。应由用户代码置0。
0	PLL时钟源。[1] 0 = 低功耗振荡器。 1 = 精密振荡器。

[1] 如果用户代码切换MCU时钟源，在将时钟切换写入PLLCON后应插入一个MCU空周期。

POWCON预写密钥POWKEY0

名称： POWKEY0

地址： 0xFFFF0404

访问类型： 只写

密钥： 0x00000001

功能： POWCON是一个加密寄存器，POWCON前后需写入32位密钥值。POWKEY0为预写密钥。

POWCON写后密钥POWKEY1

名称： POWKEY1

地址： 0xFFFF040C

访问类型： 只写

密钥： 0x000000F4

功能： POWCON是一个加密寄存器，POWCON前后需写入32位密钥值。POWKEY1为写后密钥。

POWCON寄存器

名称： POWCON

地址： 0xFFFF0408

默认值： 0x079

访问类型： 读/写

功能： 该12位寄存器允许用户代码动态进入各种低功耗模式。

表39. POWCON MMR位分配

位	描述
11至9	保留。这些位应该写入0。
8	精密振荡器使能。 该位由用户置1时，使能精密振荡器。 该位由用户清0时，关断精密振荡器。
7至6	保留。这些位应该写入0。
5	PLL关断。如果定时器外设由PLL输出时钟驱动，则关断。如果由有效时钟源驱动，则保持有效。 该位默认置1，并在唤醒事件时由硬件置1。 该位清0将关断PLL。内核或外设使能时不得关断PLL：位3、4及5必须同时清0。
4	外设关断。由此位关断的外设有：SRAM、Flash/EE存储器、GPIO接口以及SPI端口。 该位默认置1，并且/或者在唤醒事件时由硬件置1。即使该位置1，由低功耗振荡器驱动时，唤醒定时器(定时器2)仍可继续工作。 该位清0将关断外设。内核使能时，外设无法关断：位3和位4必须同时清0。即使该位清0，LIN仍能对唤醒事件作出响应。
3	内核关断。用户代码关断MCU时，在将关断命令写入POWCON后插入一个MCU空周期。 该位默认置1，并在唤醒事件时由硬件置1。 该位清0将关断ARM内核。
2至0	内核时钟分频器(CD)位。 000 = 20.48 MHz, 48.83 ns. 001 = 10.24 MHz，97.66 ns(上电默认设置) 010 = 5.12 MHz, 195.31 ns. 011 = 2.56 MHz, 390.63 ns. 100 = 1.28 MHz, 781.25 ns. 101 = 640 kHz, 1.56 µs. 110 = 320 kHz, 3.125 µs. 111 = 160 kHz, 6.25 µs.

振荡器校准

ADuC7039具有两个振荡器和两种校准方案：

- 低功耗振荡器可以利用精密振荡器或LIN通信校准。时间值也可以由用户代码修改。
- 精密振荡器可以利用LIN通信校准。时间值也可以由用户代码修改。

每个振荡器都有专用校准MMR：

- LOCUSR0是低功耗振荡器用户调整寄存器。它是一个8位寄存器。提高LOCUSR0的值会降低低功耗振荡器的频率；降低该值则会提高其频率。根据标称频率128 kHz，典型调整范围在103 kHz到156 kHz之间。此MMR可以由用户代码直接写入，或者由硬件相对于LIN波特率自动更改。
- LOCUSR1是精密振荡器用户调整寄存器。它是一个10位MMR。提高LOCUSR1的值会降低精密振荡器的频率；降低该值则会提高其频率。根据标称频率128 kHz，典型调整范围在94 kHz到178 kHz之间。此MMR可以由用户代码直接写入，或者由硬件相对于LIN波特率自动更改。
- LOCVAL0是一个8位只读MMR，显示低功耗振荡器的当前调整值。
- LOCVAL1是一个10位只读MMR，显示精密振荡器的当前调整值。注意，从该寄存器可以读出11位，但只有10位用于校准。

低功耗振荡器初始校准

复位后，低功耗振荡器以128 kHz的频率运行，最大误差为中心频率128 kHz的−10%到+3%。客户生产线上的下线校准必须在25°C ± 5°C的温度范围内执行，使低功耗振荡器和精密振荡器的中心频率一致。校准后，低功耗振荡器的误差为中心频率的±3%。

该初始校准只需在下线时执行一次。进一步的校准可通过用户代码执行，以补偿低功耗振荡器的温度漂移。

低功耗振荡器校准序列

128 kHz低功耗振荡器可以利用128 kHz精密振荡器校准。实现这一功能需要用到两个专用校准计数器。

一个计数器为9位，由精密振荡器提供时钟。另一个计数器为10位，由低功耗振荡器提供时钟。时钟校准模式由下列MMR控制：

- OSCCON—校准控制位。
- OSCSTA—校准状态寄存器。
- OSCVAL0—9位计数器0。
- OSCVAL1—10位计数器1。

校准程序示例如图21所示。用户代码利用OSCCON配置并使能校准序列。当精密功耗振荡器校准计数器OSCVAL0达到0x1FF时，两个计数器禁用。

然后，用户代码回读低功耗振荡器校准计数器的值。有三种可能情形：

- OSCVAL0 = OSCVAL1。无需进一步操作。
- OSCVAL0 > OSCVAL1。低功耗振荡器缓慢运行中。必须降低LOCUSR0。
- OSCVAL0 < OSCVAL1。低功耗振荡器快速运行中。必须提高LOCUSR0。

更改LOCUSR0后，应再次执行校准程序，同时检查新频率。注意，LOCUSR0 MMR受密钥保护。写入LOCUSR0之前，应将值0x1324写入LOCKEY。

利用内部精密振荡器执行校准程序耗时约4 ms。

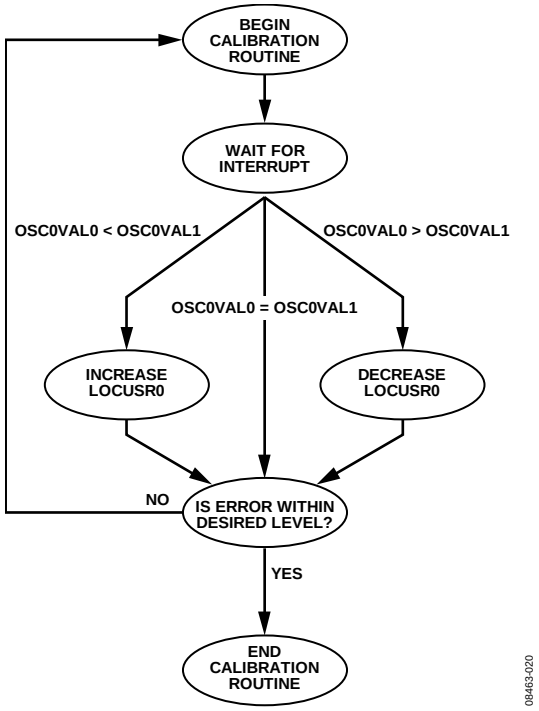


图21. OSCTRM校准程序

开始时钟校准程序前，用户必须将PLL时钟源切换到精密振荡器。否则，每次修改LOCUSR0时，PLL可能失锁，从而导致低功耗振荡器校准时间延长。

OSCCON寄存器

名称： OSCCON

地址： 0xFFFF0440

默认值： 0x00

访问类型： 读/写

功能： 该8位寄存器控制低功耗振荡器校准程序。

表40. OSCCON MMR位分配

位	描述
7 to 4	保留。应写入0。
3	校准复位。 该位由用户代码置1时，复位校准计数器并禁用校准逻辑。 校准复位后，该位由用户代码清0。
2	OSCVAL1复位。 该位由用户代码置1时，清除OSCVAL1。 清除OSCVAL1后，该位由用户代码清0。
1	OSCVAL0复位。 该位由用户代码置1时，清除OSCVAL0。 清除OSCVAL0后，该位由用户代码清0。
0	校准使能。 该位由用户代码置1时，开始校准。 该位由用户代码清0时，中止校准。

OSCSTA寄存器

名称： OSCSTA

地址： 0xFFFF0444

默认值： 0x00

访问类型： 只读

功能： 该8位寄存器指示低功耗振荡器校准程序的状态。

表41. OSCSTA MMR位分配

位	描述
7至3	保留。
2	振荡器校准结束。 校准周期完全结束时，该位由硬件置1。 读取OSCVAL1时，该位清0。
1	繁忙位。 校准正在进行时，该位由硬件置1。 校准结束时，该位由硬件清0。
0	校准完成中断。 位1解除置位时，该位由硬件置1。 读取OSCSTA MMR时，该位清0。

OSCVAL0寄存器

名称： OSCVAL0

地址： 0xFFFF0448

默认值： 0x00

访问类型： 只读

功能： 该9位计数器由128 kHz精密振荡器提供时钟。

OSCVAL1寄存器

名称： OSCVAL1

地址： 0xFFFF044C

默认值： 0x0000

访问类型： 只读

功能： 该10位计数器由128 kHz低功耗振荡器提供时钟。注意，可以读出11位，但只使用10位。

LIN振荡器校准

ADuC7039提供了第二种振荡器校准机制，这一新特性允许根据LIN数据包中的同步元素校准时钟。它基于一个固定且预定义的LIN波特率。新调整值从LIN通信获得。这种方法需要足够数量的LIN处理，温度每变化10度，大约需要1次。

如果LIN外设测得的波特率(位于LINBR MMR中)不在用户代码定义的限值(位于LOCMIN和LOCMAX MMR中)范围内，则所选振荡器调整位根据LIN振荡器校准控制寄存器选择的选项进行修改，幅度为LOCCON中定义的步进数。

两个只读调整寄存器指示当前针对每个振荡器使用的调整值。LIN振荡器校准模块修改这两个只读寄存器，但不修改用户寄存器LOCUSRx。校准完成后，可以禁用LIN振荡器校准，但禁用之前，须将LOCVALx的值复制到相应的LOCUSRx寄存器。下一部分给出了一个例子。状态寄存器指示所选振荡器的调整寄存器是否发生了改变。

有9个MMR可用：

LOCCON寄存器

名称： LOCCON

地址： 0xFFFF0480

默认值： 0x00

访问类型： 读/写(密钥保护)

功能： 通过LIN控制寄存器执行振荡器校准。

表42. LOCCON MMR位分配

位	描述
7至3	保留。读取0。
2至1	振荡器校准步进大小。 0x00 = 默认值，步进1。 0x01 = 步进2。 0x10 = 步进3。 0x11 = 步进4。
0	使能通过LIN执行振荡器校准。 该位由用户置1时，使能根据LIN波特率对选定的振荡器进行自动校准。 该位由用户清0时，禁用自动校准。

LOCUSR0寄存器

名称： LOCUSR0

地址： 0xFFFF0484

默认值： 由内核更新

访问类型： 读/写(密钥保护)

功能： 低功耗振荡器的用户调整寄存器。

LOCUSR1寄存器

名称： LOCUSR1

地址： 0xFFFF0488

默认值： Updated by kernel

访问类型： 读/写(密钥保护)

功能： 精密振荡器的用户调整寄存器。

LOCMAX寄存器

名称： LOCMAX

地址： 0xFFFF048C

默认值： 0x00000

访问类型： 读/写

功能： 对于预定的波特率，LINBR中预期的上限。

LOCMIN寄存器

名称： LOCMIN

地址： 0xFFFF0490

默认值： 0x00000

访问类型： 读/写

功能： 对于预定的波特率，LINBR中预期的下限。

LOCSTA寄存器

名称： LOCSTA

地址： 0xFFFF0494

默认值： 0x01

访问类型： 只读

功能： 校准状态寄存器。

表43. LOCSTA MMR位分配

位	描述
7至3	保留。读取0。
2	低功耗振荡器调整值已修改。 精密振荡器调整值发生改变时，该位由硬件置1。 读取LOCSTA MMR时，该位由硬件清0。
1	精密振荡器调整值已修改。 精密振荡器调整值发生改变时，该位由硬件置1。 读取LOCSTA MMR时，该位由硬件清0。
0	选定的振荡器。 选择低功耗振荡器进行校准时，该位由硬件置1。 选择精密振荡器进行校准时，该位由硬件清0。

LOCVAL0寄存器

名称： LOCVAL0

地址： 0xFFFF0498

默认值： 由内核更新

访问类型： 只读

功能： 低功耗振荡器当前调整值，只读。

LOCVAL1寄存器

名称： LOCVAL1

地址： 0xFFFF049C

默认值： 由内核更新

访问类型： 只读

功能： 精密振荡器当前调整值，只读。

LOCKEY寄存器

名称： LOCKEY

地址： 0xFFFF04A0

访问类型： 只写

功能： 必须写入才能解锁任何可写的校准寄存器。写入此MMR的值为0x1324。

开始LIN校准的典型序列如下：

```
LOCMIN = EXPECTED_LINBR_VALUE-0x20;          //Define tolerance
LOCMAX = EXPECTED_LINBR_VALUE+0x20;          //Define tolerance
LOCKEY = 0x1324;                             //Unlock key protection
LOCCON = 0x1;                                //Enable calibration with step size = 1
```

序列示例

利用LIN通信校准低功耗振荡器的示例如下：

```
LINCON = 0x800;                              //Enable LIN
expected_baudrate = 0x10AB;                  //Correspond to 19200 bps
LOCMIN = EXPECTED_LINBR_VALUE-0x20;          //Define tolerance
LOCMAX = EXPECTED_LINBR_VALUE+0x20;          //Define tolerance
LOCKEY = 0x1324;                             //Unlock key protection
LOCCON = 0x1;                                //Enable calibration with step size = 1
while ((LOCSTA & 0x05)!= 0x05){}             //Wait for the trim value to be modified
while ((LINBR<LOCMIN)||(LINBR>LOCMAX)){}     //Wait for the correct baud rate
temp_trim = LOCVAL0;                         //Store the trim value given
LOCKEY = 0x1324;                             //Unlock key protection
LOCUSR0 = temp_trim;                         //Write the trim value into the user MMR
LOCKEY = 0x1324;                             //Unlock key protection
LOCCON = 0;                                  //Turn off the LIN calibration block
```

中断系统

ADuC7039拥有由中断控制器控制的10个中断源。大多数中断都是由片内外设产生的，例如ADC和定时器。ARM7TDMI-S CPU内核只能识别以下两类中断：正常中断请求(IRQ)和快速中断请求(FIQ)。所有中断都可以被单独屏蔽。

中断系统的控制和配置由9个中断相关的寄存器管理：4个专用于IRQ，4个专用于FIQ，还有一个MMR用于选择编程中断源。每一个IRQ和FIQ寄存器中的控制位都代表相同的中断源，如表44所示。

内核开始执行中断服务程序(ISR)后，应立即保存IRQSTA/FIQSTA，以确保能够响应所有有效中断源。

ARM7TDMI-S内核中断产生程序如图22所示。

例如，设置定时器0每隔5 ms产生一次超时。第一次5 ms超时后，FIQSIG[2]或IRQSIG[2]置1，只能通过写入T0CLRI清0。如果在IRQEN或FIQEN内没有使能定时器0，则FIQSTA/IRQSTA[2]不会置1，也不会产生中断。如果在IRQEN或FIQEN内使能定时器0，则FIQSTA[2]或IRQSTA[2]置1或产生中断(FIQ或IRQ)。

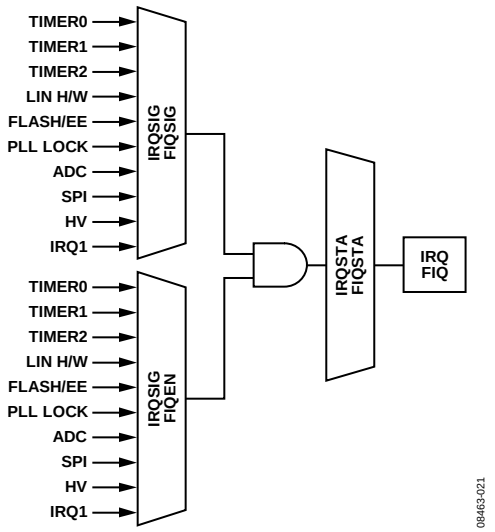


图22. 中断结构

注意，CPSR的IRQ和FIQ中断位定义仅通过ARM内核来控制中断识别，而非外设。例如，如果配置定时器2通过IRQEN产生一个IRQ，则CPSR内的IRQ中断位置1(禁用)并关断ADuC7039。中断发生后，外设唤醒，但ARM内核仍处于关断状态。这相当于POWCON = 0x71。发生这种情况时，ARM内核只能通过复位事件来上电。

表44. IRQ/FIQ MMR位分配

位	描述	注释
0	所有中断的逻辑或	仅限FIQ MMR
1	SWI	不用于IRQEN/CLR和FIQEN/CLR。
2	定时器0	
3	定时器1或唤醒定时器	
4	定时器2或看门狗定时器	
5	LIN	
6	Flash/EE中断	
7	PLL锁定	
8	ADC	
9	SPI	
10	高压	
11	低功耗振荡器校准完成	
12	保留	
13	IRQ1 (GPIO IRQ1)	GPIO4处的外部中断
14至31	保留	

IRQ

IRQ是进入处理器IRQ模式的一个异常信号。它用于内、外部事件的通用中断处理。

所有32位经逻辑"或"运算后，形成要发送给ARM7TDMI-S内核的IRQ信号。器件有如下4个32位寄存器专门用于IRQ。

IRQSIG

IRQSIG是一个只读寄存器，反映不同IRQ源的状态。如果一个外设产生了一个IRQ信号，IRQSIG中相应的位就会被置1；否则就清0。当特定外设的中断被清除时，IRQSIG位就会被清0。通过设置IRQEN MMR，可屏蔽所有IRQ中断源。

IRQEN

IRQEN提供当前使能屏蔽的值。将该寄存器的某一位置1时，可使能相应的中断请求，此时将产生IRQ异常。将某一位清0时，可禁用或屏蔽相应的中断请求，此时将无法产生IRQ异常。IRQEN寄存器无法用来禁用中断。

IRQCLR

IRQCLR是一个只写寄存器，可清除IRQEN寄存器的相应位，从而屏蔽相应的中断源。将该寄存器的某一位置1时，会清除IRQEN寄存器的相应位，但不影响其他位。寄存器IRQEN和IRQCLR配对使用，可以实现独立的使能屏蔽功能，而无需执行原子性读-改-写操作。

IRQSTA

IRQSTA是一个只读寄存器，提供当前使能的IRQ源的状态(IRQSIG和IRQEN对应位进行逻辑"与"操作)。置1时，这个源将向ARM7TDMI-S内核发出一个有效的IRQ中断请求。没有优先级编码器和中断矢量产生。该功能可以在软件中通过一个普通的中断处理程序实现。

快速中断请求(FIQ)

快速中断请求(FIQ)是进入处理器FIQ模式的一个异常信号。它用于数据传输或低延迟通信通道任务处理。FIQ接口与IRQ接口相同；此外，它还会提供二级中断(拥有最高优先级)。器件内有4个32位寄存器专门用于FIQ，包括：FIQSIG、FIQEN、FIQCLR和FIQSTA。

FIQSTA的位31至位1经逻辑"或"运算产生FIQ信号，传送到内核以及FIQ和IRQ寄存器的位0(FIQ源)。

逻辑上FIQEN和FIQCLR不允许一个中断源同时使能IRQ和FIQ屏蔽。FIQEN中的某一位被置1会导致IRQEN中的同一位被清0。同样，IRQEN中的某一位被置1会导致FIQEN中的同一位被清0。一个中断源可以被IRQEN屏蔽和FIQEN屏蔽禁用。

可编程中断

因为可编程中断是无法屏蔽的，所以它们由另一个寄存器SWICFG控制，通过这个寄存器可以同时写入IRQSTA和IRQSIG寄存器和/或FIQSTA和FIQSIG寄存器。

专用于设置软件中断的32位寄存器为SWICFG，见表45。该寄存器允许控制可编程源中断。

表45. SWICFG MMR位分配

位	描述
31至3	保留。
2	可编程中断FIQ。 通过将该位置1或清0，可将FIQSTA和FIQSIG寄存器的位1置1或清0。
1	可编程中断IRQ。 通过将该位置1或清0，可将IRQSTA和IRQSIG寄存器的位1置1或清0。
0	保留。

注意，任何中断信号的有效时间不得少于中断延迟时间，这样才能保证中断信号能够被中断控制器检测到，或者被用户在IRQSTA或FIQSTA寄存器中检测到。

定时器

ADuC7039具有3个通用定时器/计数器：

- 定时器0或通用定时器
- 定时器1或唤醒定时器
- 定时器2或看门狗定时器

通过写入数据到某一定时器的控制寄存器(TxCON)，可以启动相应的定时器。计数模式和速度取决于TxCON中选择的配置。

在正常模式下，每当计数器的值达到0(递减计数)或满量程(递增计数)时，都会产生一个IRQ中断。向某一定时器(TxCLRI)的清除寄存器内写入任一数据，可以清除IRQ中断。

这3个定时器可以在自由模式或周期模式下工作。

在自由模式下，利用TxLD寄存器中的值，计数器从最大值/最小值开始递减/递增至0/满量程，然后再次从最大值/最小值开始。这意味着，在自由模式下，当相关中断位置1时，TxVAL不会重新加载；当计数器下溢或上溢时，计数值翻转。

在周期模式下，计数器以加载寄存器(TxLD寄存器)中的值为起始值，递减/递增计数至0或满量程，然后再以该值为起始值，重新开始计数。这意味着，当相关中断位置1时，TxVAL会重新加载TxLD，计数再次从该值开始。

建议不要将0加载到TxLD寄存器中。通过访问计数器的值寄存器(TxVAL)，可以随时读出计数器的值。

异步时钟域的定时器同步

图23显示了用户定时器MMR与内核定时器模块之间的接口。用户代码可以直接访问所有定时器MMR，包括TxLD、TxVAL、TxCON和TxCLRI。数据必须从这些MMR传输到定时器子系统内的内核定时器(T0、T1和T2)。这些内核定时器由同步(SYNC)模块通过用户的MMR接口缓冲。SYNC模块的主要作用是提供一种方法来确保数据和所需的其它控制信号能够正确穿过异步时钟域。异步时钟域的一个例子是：MCU以10 MHz内核时钟运行，定时器1以32 KHz的低功耗振荡器频率运行。

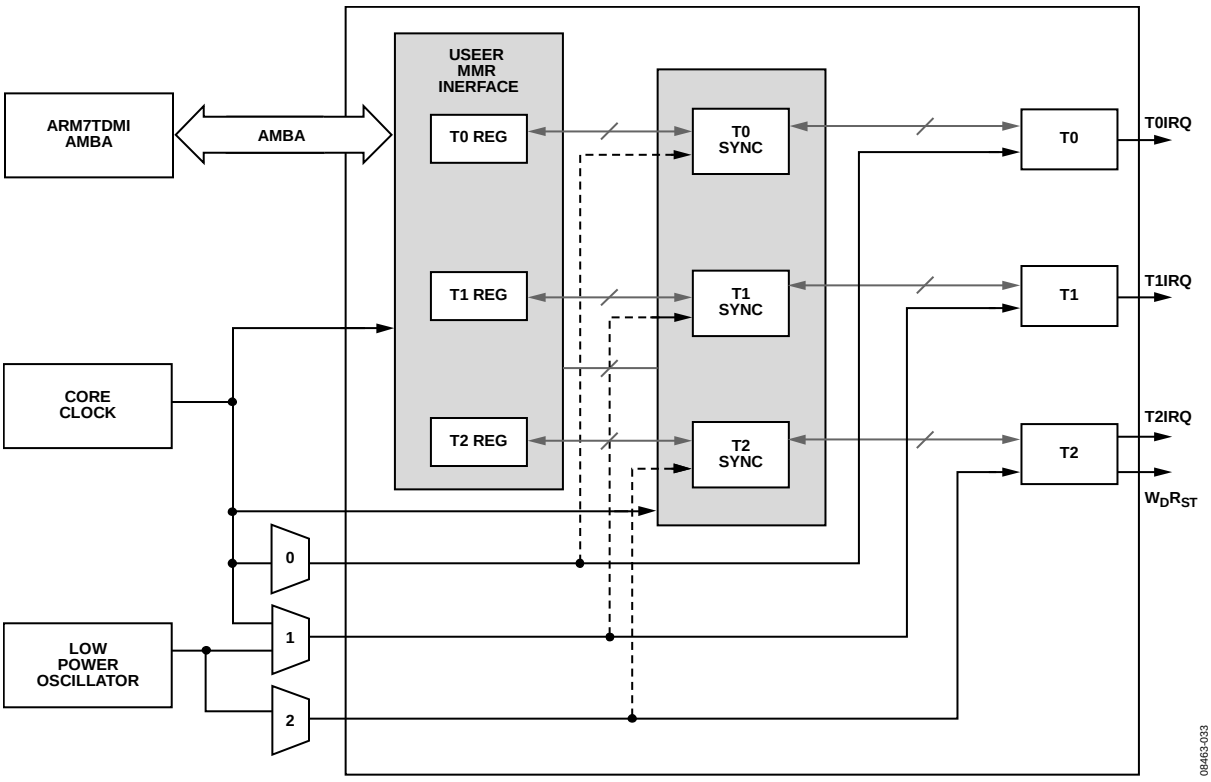


图23. 定时器功能框图

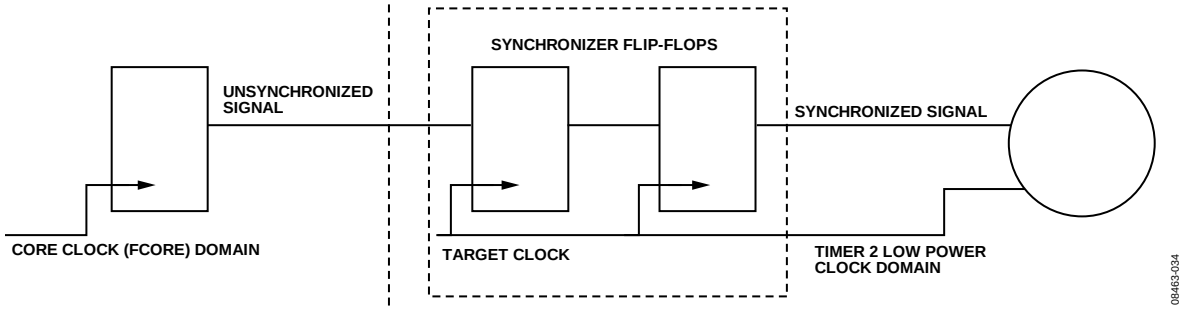


图24. 跨时钟域信号的同步器

从图23可以看出，MMR逻辑和内核定时器逻辑位于不同且异步的时钟域内。来自MMR内核时钟域并且传送到内部定时器域的任何数据必须与内部定时器时钟域同步，确保正确锁存到内核定时器时钟域。这是通过两个正反器实现的，如图24所示，数据不仅同步，而且经过双缓冲，从而确保数据在定时器时钟域中的完整性。

由于同步模块的存在，虽然MMR时钟域几乎是立即锁存(利用快速内核时钟)定时器控制数据，但此数据不会立即到达内核定时器逻辑，而是在所选内部定时器域时钟的至少2个周期之后。

定时器编程

不同定时器域的同步还要求用户代码对定时器进行仔细编程，决定何时停止或启动定时器。建议使用的代码控制定时器模块何时停止和启动定时器，以及何时使用不同的时钟域。如果使能定时器产生IRQ或FIQ异常，这种控制特别重要。下面以定时器1为例进行说明。

暂停定时器1

暂停定时器1时，建议屏蔽定时器1的IRQEN位(使用IRQCLR)，这可以防止不需要的IRQ在定时器1内部逻辑锁存T1CON控制位之前产生MCU中断。

```
IRQCLR = WAKEUP_TIMER_BIT;     // Masking interrupts
T1CON = 0x00;                  // halting the timer,
```

启动定时器1

启动定时器1时，建议首先将所需的TxLD值加载到定时器1。然后，根据需要设置T1CON位以启动定时器。这将使能定时器，但前提是定时器1时钟域已内部锁存T1CON位。因此，建议插入3个时钟周期(对于32 kHz定时器时钟源，则为100 μs)以上的延迟，使得T1LD值和T1CON值均可通过同步逻辑锁存并到达定时器1域。延迟之后，建议使用T1CLRI = 0x00清除所有(意外的)定时器1中断。最后，将IRQEN MMR中的相应位置1可以解除屏蔽定时器1系统中断。示例代码如下。

序列示例

假设定时器1按照上文所述中止。

```
T1LD = 0x1;                    // Reload timer
T1CON = 0x001F;                // Enable timer, Low power oscillator, 32768 prescaler, periodic
Delay(100us);                  // Include delay to ensure T1CON bits take effect
T1CLRI = 0 ;                   //* Clear Timer IRQ
IRQEN = WAKEUP_TIMER_BIT;      // Unmask Timer1
```

定时器0—通用定时器

定时器0是一个16位通用定时器，可递增计数或递减计数。定时器0采用内核时钟工作，预分频器为1或16,384。当预分频器为16,384时，产生1.6 ms的最低分辨率，定时器可以计数1分钟以上。

定时器0可以递增计数，也可以递减计数。16位值可通过写入T0LD以载入到计数器内。当前计数器值可从T0VAL读取。当定时器0上溢时，定时器0从T0LD寄存器中重新载入数值。

定时器0接口包括4个寄存器：

- T0LD是一个16位寄存器，用于保存载入计数器的16位值。
- T0VAL是一个16位寄存器，用于保存定时器0的当前16位值。
- T0CLRI是一个8位寄存器。向其中写入任意值，可以清除定时器0中断。
- T0CON是一个16位配置寄存器，如表46所示。

定时器0载入寄存器

名称： T0LD

地址： 0xFFFF0300

默认值： 0x0000

访问类型： 读/写

功能： T0LD为16位寄存器，用于保存载入计数器的16位值。

定时器0值寄存器

名称： T0VAL

地址： 0xFFFF0304

默认值： 0x0000

访问类型： 只读

功能： T0VAL为16位寄存器，用于保存定时器0的当前值。

定时器0控制寄存器

名称： T0CON

地址： 0xFFFF0308

默认值： 0x0000

访问类型： 读/写

功能： 该16位寄存器用于设置定时器0的工作模式。

定时器0清除寄存器

名称： T0CLRI

地址： 0xFFFF030C

访问类型： 只写

功能： 通过用户代码向该8位只写寄存器写入任意值，可以清除中断。

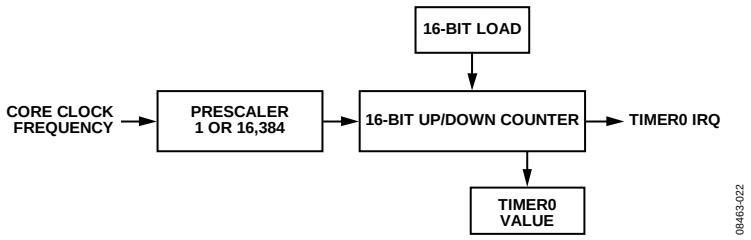


图25. 定时器0框图

表46. T0CON MMR位分配

位	描述
15至6	保留。
5	定时器0模式。 该位由用户代码置1时，选择周期模式。 该位由用户代码清0时，选择自由模式(默认)。
4	递增计数。 该位由用户代码置1时，定时器0递增计数。 该位由用户代码清0时，定时器0递减计数(默认)。
3	定时器0使能位。 该位由用户代码置1时，使能定时器0。 该位由用户代码清0时，禁用定时器0(默认)。
2	保留。
1至0	预分频器。 00 = 时钟源/1(默认)。 01 = 时钟源/1。 10 = 时钟源/16,384。 11 = 时钟源/16,384。

定时器1—唤醒定时器

定时器1是一个带有可编程预分频器的32位唤醒定时器，可递增计数或递减计数。所选时钟源(内核时钟或低功耗振荡器)的分频系数共有1、16、256或32,768四种。如果从低功耗振荡器获得时钟，当内核时钟被禁用时，唤醒定时器仍继续运行。

定时器1在溢出时从T1LD重新载入值。

定时器1接口有4个寄存器：

- T1LD和T1VAL均为32位寄存器，用于保存32位无符号整数。T1VAL为只读寄存器。
- T1CLRI是一个8位寄存器。向其中写入任意值，可以清除定时器1中断。
- T1CON是一个16位配置寄存器，如表47所示。

定时器1载入寄存器

名称： T1LD

地址： 0xFFFF0320

默认值： 0x00000000

访问类型： 读/写

功能： T1LD是一个32位寄存器，用于保存载入计数器的32位值。

定时器1值寄存器

名称： T1VAL

地址： 0xFFFF0324

默认值： 0xFFFFFFFF

访问类型： 只读

功能： T1VAL是一个32位寄存器，用于保存定时器1的当前值。

定时器1清除寄存器

名称： T1CLRI

地址： 0xFFFF032C

默认值： 只写

功能： 通过用户代码向该8位只写寄存器写入任意值，可以清除中断。

定时器1控制寄存器

名称： T1CON

地址： 0xFFFF0328

默认值： 0x0000

访问类型： 读/写

功能： T1CON是一个16位寄存器，用于设置定时器1的工作模式。

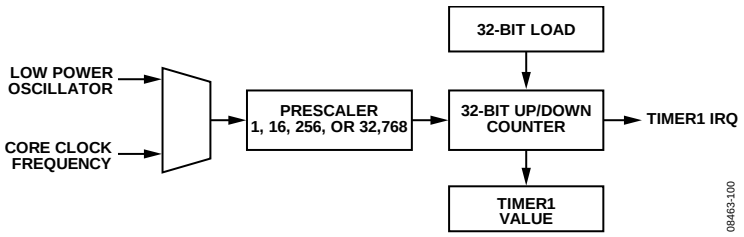


图26. 定时器1框图

表47. T1CON MMR位分配

位	描述
15至6	保留。这些位应该写入0。
5	定时器1模式。 该位由用户代码置1时，选择周期模式。 该位由用户代码清0时，选择自由模式(默认)。
4	递增计数。 该位由用户代码置1时，定时器1递增计数。 该位由用户代码清0时，定时器1递减计数(默认)。
3	定时器1使能位。 该位由用户代码置1时，使能定时器1。 该位由用户代码清0时，禁用定时器1(默认)。
2	时钟选择。 0 = 内核时钟(默认)。 1 = 低功耗(32.768 kHz)振荡器。
1至0	预分频器。 00 = 时钟源/1(默认)。 01 = 时钟源/16。 10 = 时钟源/256。 11 = 时钟源/32,768。

定时器2—看门狗定时器

定时器2共有两种工作模式：正常模式和看门狗模式。看门狗定时器用于强制处理器从非法软件状态恢复到正常工作状态。一旦看门狗定时器被使能，它需要周期性服务来阻止自身强制处理器执行复位操作。

当定时器2在正常模式下溢出时，或者在看门狗模式下写入T2CLRI时，定时器2将重新载入T2LD中的值。

正常模式

正常模式下的定时器2与16位工作模式下的定时器0相同(时钟源和预分频器除外)。时钟源采用低功耗振荡器，分频系数共有1、16和256三种。

看门狗模式

将T2CON[5]置1可进入看门狗模式。定时器2从T2LD寄存器中的超时值开始递减计数，直至计数值为0。当使用1/256的最大预分频和T2LD的满量程值时，最长超时时间为524秒。

Flash/EE存储器完成一次页擦除操作需要20 ms，为避免与此发生任何冲突，用户软件不得将超时周期设为小于30 ms。

当T2VAL到达0时，系统产生复位或中断，具体取决于T2CON[1]的值。要避免出现复位或中断事件，必须在T2VAL达到0之前将任意值写入T2CLRI。此操作可将T2LD的值重新载入计数器内，且重新开始一个新的超时周期。

一旦进入看门狗模式，T2LD和T2CON就会被写保护。

除非通过上电复位将看门狗定时器复位，否则，无法修改这两个寄存器内的数据。其它任何复位事件产生后，看门狗定时器仍会继续计数。看门狗定时器应在用户代码的最初行内配置以免陷入看门狗复位的无限循环内。

定时器2在JTAG调试访问期间自动暂停，并且只在JTAG放弃ARM7内核控制权后才会重新开始计数。默认情况下，定时器2可在关断期间继续计数。这可通过将T2CON[0]置1来禁用。建议采用默认值，即在关断期间仍允许看门狗定时器继续计数。

定时器2接口有4个寄存器：

- T2LD是一个16位寄存器，用于保存载入计数器的16位值。
- T2VAL是一个16位寄存器，用于保存定时器2的当前16位值。
- T2CLRI是一个8位寄存器。在正常模式下向这个寄存器写入任意值，将清除定时器2中断；在看门狗模式下向这个寄存器写入任意值，则重新开始一个超时周期。
- T2CON是一个16位配置寄存器，如表48所示。

定时器2载入寄存器

名称： T2LD

地址： 0xFFFF0340

默认值： 0x0050

访问类型： 读/写

功能： T2LD是一个16位寄存器，用于保存载入计数器的16位值。

定时器2清除寄存器

名称： T2CLRI

地址： 0xFFFF034C

访问类型： Write only

功能： 在正常模式下，通过用户代码向该8位只写寄存器写入任意值，可清除中断；在看门狗模式下则可复位超时。

定时器2值寄存器

名称： T2VAL

地址： 0xFFFF0344

默认值： 0x0050

访问类型： 只写

功能： T2VAL为16位寄存器，用于保存定时器2的当前值。

定时器2控制寄存器

名称： T2CON

地址： 0xFFFF0348

默认值： 0x0000

访问类型： 读/写

功能： 该16位寄存器用于设置定时器2的工作模式。

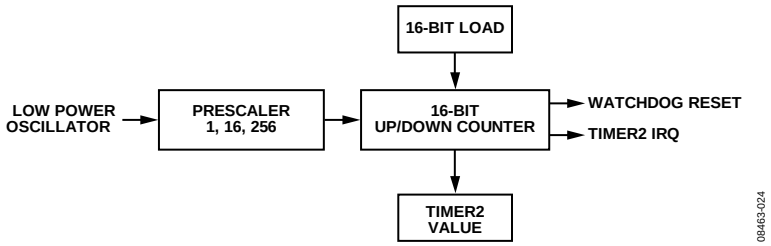


图27. 定时器2框图

表48. T2CON MMR位分配

位	描述
15至9	保留。这些位保留，应由用户代码写入0。
8	递增/递减计数使能。 该位由用户代码置1时，定时器2递增计数。 该位由用户代码清0时，定时器2递减计数。
7	定时器2使能。 该位由用户代码置1时，使能定时器2。 该位由用户代码清0时，禁用定时器2。
6	定时器2工作模式。 该位由用户代码置1时，选择周期模式。 该位由用户代码清0时，选择自由模式。
5	看门狗定时器模式使能。 该位由用户代码置1时，使能看门狗模式。 该位由用户代码清0时，禁用看门狗模式。
4	保留。这些位保留，应由用户代码写入0。
3至2	定时器2时钟预分频器。 00 = 时钟源/1（默认）。 01 = 时钟源/16。 10 = 时钟源/256。 11 = 保留。
1	看门狗定时器IRQ使能。 该位由用户代码置1时，可在看门狗计数值为0时产生IRQ中断而非复位。 该位由用户代码清0时，禁用IRQ选项。
0	PD_OFF. 该位由用户代码置1时，则利用POWCON MMR位4关断外设时停止定时器2。 该位由用户代码清0时，则利用POWCON MMR位4关断外设时使能定时器2。

通用输入/输出

ADuC7039共有6个通用双向输入/输出(GPIO)引脚。一般情况下，通过用户代码设置，GPIO引脚可以实现多种功能。默认情况下，GPIO引脚在GPIO模式下工作。所有GPIO引脚都具有一个内部上拉电阻，吸电流能力为0.8 mA，源电流能力为0.1 mA。典型GPIO结构如图28所示。

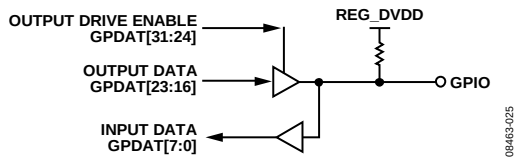


图28. ADuC7039 GPIO

这6个GPIO归属于一个6位宽端口。该端口的GPIO分配详见表49。

在正常操作期间，用户代码可利用这些通用寄存器来控制外部GPIO引脚的功能和状态。所有GPIO引脚在关断(POWCON)模式期间保持其外部电平(高或低)。

表49. 外部GPIO引脚到内部端口信号分配

GPIO引脚	端口信号	功能(由GPCON定义)
GPIO_0	P0.0	通用I/O。
	$\overline{SS}$	SPI从机选择I/O。
GPIO_1	P0.1	通用I/O。
	SCLK	SPI串行时钟I/O。
GPIO_2	P0.2	通用I/O。
	MISO	SPI主机输入、从机输出。
GPIO_3	P0.3	通用I/O。
	MOSI	SPI主机输出、从机输入。
GPIO_4	P0.4	通用I/O。
	LINRx	LIN一致性测试的LIN输入。
	IRQ1	外部中断请求1。
GPIO_5	P0.5	通用I/O。
	LINTx	LIN一致性测试的LIN输出。

端口引脚由以下4个MMR配置和控制：

- GPCON：控制寄存器
- GPDAT：配置和数据寄存器
- GPSET：数据设置
- GPCLR：数据清零

GPIO端口控制寄存器

名称： GPCON

地址： 0xFFFF0D00

默认值： 0x00000000

访问类型： 读/写

功能： 该32位MMR选择每个端口引脚的功能。

表50. GPCON MMR位分配

位	描述
31至21	保留。这些位保留，应由用户代码写入0。
20	GPIO_5功能选择位。 该位由用户代码清0时，GPIO_5引脚配置为通用I/O (GPIO)引脚。 该位由用户代码置1时，GPIO_5引脚配置为LIN一致性测试的LIN输出。
19至17	保留。这些位保留，应由用户代码写入0。
16	GPIO_4功能选择位。 该位由用户代码清0时，GPIO_4引脚配置为通用I/O (GPIO)引脚。 该位由用户代码置1时，GPIO_4引脚配置为LIN一致性测试的LIN输入。
15至13	保留。这些位保留，应由用户代码写入0。
12	GPIO_3功能选择位。 该位由用户代码清0时，GPIO_3引脚配置为通用I/O (GPIO)引脚。 该位由用户代码置1时，GPIO_3引脚配置为SPI端口的MOSI(主机输出从机输入)。
11至9	保留。这些位保留，应由用户代码写入0。
8	GPIO_2功能选择位。 该位由用户代码清0时，GPIO_2引脚配置为通用I/O (GPIO)引脚。 该位由用户代码置1时，GPIO_2引脚配置为SPI端口的MISO(主机输入从机输出)。
7至5	保留。这些位保留，应由用户代码写入0。
4	GPIO_1功能选择位。 该位由用户代码清0时，GPIO_1引脚配置为通用I/O (GPIO)引脚。 该位由用户代码置1时，GPIO_1引脚配置为SPI端口的SCLK(串行时钟I/O)。
3至1	保留。这些位保留，应由用户代码写入0。
0	GPIO_0功能选择位。 该位由用户代码清0时，GPIO_0引脚配置为通用I/O (GPIO)引脚。 该位由用户代码置1时，GPIO_0引脚配置为SPI端口的SS(从机选择输入)。

GPIO端口数据寄存器

名称： GPDAT

地址： 0xFFFF0D10

默认值： 0x000000FF

访问类型： 读/写

功能： 该32位寄存器用于配置GPIO引脚的方向，还设置用作输出端的GPIO引脚的输出值并读取用作输入端的GPIO引脚的状态。

表51. GPDAT MMR位分配

位	描述
31至30	保留。这些位保留，应由用户代码写入0。
29	端口0.5方向选择位。 该位由用户代码清0时，将分配给端口0.5的GPIO引脚配置为输入端。 该位由用户代码置1时，将分配给端口0.5的GPIO引脚配置为输出端。
28	端口0.4方向选择位。 该位由用户代码清0时，将分配给端口0.4的GPIO引脚配置为输入端。 该位由用户代码置1时，将分配给端口0.4的GPIO引脚配置为输出端。
27	端口0.3方向选择位。 该位由用户代码清0时，将分配给端口0.3的GPIO引脚配置为输入端。 该位由用户代码置1时，将分配给端口0.3的GPIO引脚配置为输出端。
26	端口0.2方向选择位。 该位由用户代码清0时，将分配给端口0.2的GPIO引脚配置为输入端。 该位由用户代码置1时，将分配给端口0.2的GPIO引脚配置为输出端。
25	端口0.1方向选择位。 该位由用户代码清0时，将分配给端口0.1的GPIO引脚配置为输入端。 该位由用户代码置1时，将分配给端口0.1的GPIO引脚配置为输出端。
24	端口0.0方向选择位。 该位由用户代码清0时，将分配给端口0.0的GPIO引脚配置为输入端。 该位由用户代码置1时，将分配给端口0.0的GPIO引脚配置为输出端。
23至22	保留。这些位保留，应由用户代码写入0。
21	端口0.5数据输出。写入到该位的值直接出现在分配给端口0.5的GPIO引脚上。
20	端口0.4数据输出。写入到该位的值直接出现在分配给端口0.4的GPIO引脚上。
19	端口0.3数据输出。写入到该位的值直接出现在分配给端口0.3的GPIO引脚上。
18	端口0.2数据输出。写入到该位的值直接出现在分配给端口0.2的GPIO引脚上。
17	端口0.1数据输出。写入到该位的值直接出现在分配给端口0.1的GPIO引脚上。
16	端口0.0数据输出。写入到该位的值直接出现在分配给端口0.0的GPIO引脚上。
15至6	保留。这些位保留，应由用户代码写入0。
5	端口0.5数据输入。只读，反映分配给端口0.5的GPIO引脚的当前状态。应由用户代码写入0。
4	端口0.4数据输入。只读，反映分配给端口0.4的GPIO引脚的当前状态。应由用户代码写入0。
3	端口0.3数据输入。只读，反映分配给端口0.3的GPIO引脚的当前状态。应由用户代码写入0。
2	端口0.2数据输入。只读，反映分配给端口0.2的GPIO引脚的当前状态。应由用户代码写入0。
1	端口0.1数据输入。只读，反映分配给端口0.1的GPIO引脚的当前状态。应由用户代码写入0。
0	端口0.0数据输入。只读，反映分配给端口0.0的GPIO引脚的当前状态。应由用户代码写入0。

GPIO端口设置寄存器

名称： GPSET

地址： 0xFFFF0D14

默认值： N/A

访问类型： 只写

功能： 该32位MMR允许用户代码对外部GPIO引脚分别进行位寻址以仅将其单独拉高。用户代码可利用GPSET MMR来实现此操作，而不必修改或维持任何其它GPIO引脚的状态（使用GPDAT时用户代码要求）。

表52. GPSET MMR位分配

位	描述
31至22	保留。这些位保留，应由用户代码写入0。
21	端口0.5设置位。 该位由用户代码置1时，将外部GPIO_5引脚拉高。 该位由用户软件清0时，不影响外部GPIO_5引脚。
20	端口0.4设置位。 该位由用户代码置1时，将外部GPIO_4引脚拉高。 该位由用户软件清0时，不影响外部GPIO_4引脚。
19	端口0.3设置位。 该位由用户代码置1时，将外部GPIO_3引脚拉高。 该位由用户软件清0时，不影响外部GPIO_3引脚。
18	端口0.2设置位。 该位由用户代码置1时，将外部GPIO_2引脚拉高。 该位由用户软件清0时，不影响外部GPIO_2引脚。
17	端口0.1设置位。 该位由用户代码置1时，将外部GPIO_1引脚拉高。 该位由用户软件清0时，不影响外部GPIO_1引脚。
16	端口0.0设置位。 该位由用户代码置1时，将外部GPIO_0引脚拉高。 该位由用户软件清0时，不影响外部GPIO_0引脚。
15至0	保留。这些位保留，应由用户代码写入0。

GPIO端口清零寄存器

名称： GPCLR

地址： 0xFFFF0D18

默认值： N/A

访问类型： 只写

功能： 该32位MMR允许用户代码对外部GPIO引脚分别进行位寻址以仅将其单独拉低。用户代码可利用GPCLR MMR来实现此操作，而不必修改或维持任何其它GPIO引脚的状态(使用GPDAT时用户代码要求)。

表53. GPCLR MMR位分配

位	描述
31至22	保留。这些位保留，应由用户代码写入0。
21	端口0.5清除位。 该位由用户代码置1时，将外部GPIO_5引脚拉低。 该位由用户软件清0时，不影响外部GPIO_5引脚。
20	端口0.4清除位。 该位由用户代码置1时，将外部GPIO_4引脚拉低。 该位由用户软件清0时，不影响外部GPIO_4引脚。
19	端口0.3清除位。 该位由用户代码置1时，将外部GPIO_3引脚拉低。 该位由用户软件清0时，不影响外部GPIO_3引脚。
18	端口0.2清除位。 该位由用户代码置1时，将外部GPIO_2引脚拉低。 该位由用户软件清0时，不影响外部GPIO_2引脚。
17	端口0.1清除位。 该位由用户代码置1时，将外部GPIO_1引脚拉低。 该位由用户软件清0时，不影响外部GPIO_1引脚。
16	端口0.0清除位。 该位由用户代码置1时，将外部GPIO_0引脚拉低。 该位由用户软件清0时，不影响外部GPIO_0引脚。
15至0	保留。这些位保留，应由用户代码写入0。

串行外设接口(SPI)

ADuC7039片内集成一个完整的硬件串行外设接口(SPI)。SPI是一种工业标准同步串行接口，允许同时双向传输8位数据(即全双工)，最大比特率可达5.12 Mb。

该SPI端口可配置为主机或从机操作，一般由4个引脚组成：MISO、MOSI、SCLK和$\overline{SS}$。

主机输入、从机输出(MISO)引脚

在主机模式下，MISO引脚被配置为输入线路；在从机模式下，配置为输出线路。主机上的MISO线路(数据输入)应与从机内的MISO线路(数据输出)相连。传送的数据是以字节(8位)为单位的串行数据，MSB优先。

主机输出、从机输入(MOSI)引脚

在主机模式下，MOSI引脚被配置为输出线路；在从机模式下，配置为输入线路。主机上的MOSI线路(数据输出)应与从机内的MOSI线路(数据输入)相连。传送的数据是以字节(8位)为单位的串行数据，MSB优先。

串行时钟I/O (SCLK)引脚

主机串行时钟(SCLK)用于同步MOSI SCLK周期中发送和接收的数据。所以，发送/接收一个字节需要8个SCLK周期。在主机模式下，SCLK引脚配置成输出端，而在从机模式下，配置成输入端。

在主机模式下，时钟的极性和相位由SPICON寄存器控制，SPIDIV寄存器的值决定了比特率。比特率的计算公式如下：

$$f_{SERIALCLOCK} = \frac{20.48\ \text{MHz}}{2 \times (1 + SPIDIV)}$$

主机模式支持的最大比特率为10.24 Mb。在从机模式下，可对SPICON寄存器进行设置，以配置预期输入时钟的相位和极性。从机可以从外部主机接收数据(速率可达5.12 Mb)。

在主机模式和从机模式下，数据都在SCLK信号的一个沿发送，并在另一个沿采样。所以，从机时钟的极性和相位必须与主机配置一致。

从机选择($\overline{SS}$)引脚

在SPI从机模式时，置位$\overline{SS}$引脚将启动数据传输，该引脚为一个低电平有效输入信号。然后，SPI端口开始发送和接收8位数据，直到$\overline{SS}$解除置位，传输结束。在从机模式下，$\overline{SS}$总是为输入。

在SPI主机模式下，$\overline{SS}$是低电平有效输出信号。传输开始后，它自动置位；传输完成后，它自动解除置位。

SPI MMR接口

下列MMR寄存器用来控制SPI接口：SPISTA、SPIRX、SPITX、SPIDIV和SPICON。

SPIRX寄存器

名称： SPIRX

地址： 0xFFFF0A04

默认值： 0x00

访问类型： 只读

功能： 该8位寄存器是SPI接收寄存器。

SPITX寄存器

名称： SPITX

地址： 0xFFFF0A08

默认值： N/A

访问类型： 只写

功能： 该8位寄存器是SPI发送寄存器。

SPIDIV寄存器

名称： SPIDIV

地址： 0xFFFF0A0C

默认值： 0x00

访问类型： 读/写

功能： 该6位寄存器是SPI波特率选择寄存器。

SPI状态寄存器

名称： SPISTA

地址： 0xFFFF0A00

默认值： 0x0000

访问类型： 只读

功能： 该16位寄存器用于存储主机、从机模式下SPI接口的状态。

表54. SPISTA MMR位分配

位	描述
15至12	保留位。
11	SPI接收FIFO存在过剩字节。 接收FIFO的字节数超过由SPICON寄存器中SPIRXMDE位规定的字节数后，该位置1。 FIFO中的字节数不超过由SPIRXMDE位规定的字节数时，该位清0。
10至8	SPI接收FIFO状态位。 [000] = 接收FIFO为空。 [001] = 该FIFO内有1个有效字节。 [010] = 该FIFO内有2个有效字节。 [011] = 该FIFO内有3个有效字节。 [100] = 该FIFO内有4个有效字节。
7	SPI接收FIFO溢出状态位。 当接收FIFO已满时，如果向该FIFO内加载新数据，该位置1。除非SPICON[12]置1，否则该位置1将产生一个中断。 读取SPISTA寄存器的内容后，该位清0。
6	SPI接收IRQ状态位。 产生接收中断时，该位置1。当SPICON[6]清0，并且接收到所需字节数后，该位置1。 读取SPISTA寄存器的内容后，该位清0。
5	SPI发送IRQ状态位。 产生发送中断时，该位置1。当SPICON[6]置1，并且已发送所需字节数后，该位置1。 读取SPISTA寄存器的内容后，该位清0。
4	SPI发送FIFO下溢。 当启动一次发送操作且发送FIFO内没有有效数据时，该位置1。除非SPICON[13]置1，否则该位置1将产生一个中断。 读取SPISTA寄存器的内容后，该位清0。
3至1	SPI发送FIFO状态位。 000 = 发送FIFO为空。 001 = 该FIFO内有1个有效字节。 010 = 该FIFO内有2个有效字节。 011 = 该FIFO内有3个有效字节。 100 = 该FIFO内有4个有效字节。
0	SPI中断状态位。 SPI中断发生时，该位置1。 读取SPISTA寄存器的内容后，该位清0。

SPI控制寄存器

名称： SPICON

地址： 0xFFFF0A10

默认值： 0x0000

访问类型： 读/写

功能： 该16位寄存器用于在主机和从机模式下配置SPI外设。

表55. SPICON MMR位分配

位	描述
15至14	SPI IRQ模式位。这些位用于配置在传输过程中何时发生发送/接收中断。 00 = 传输完1个字节后，产生发送中断。FIFO接收到1个或以上字节时，产生接收中断。 01 = 传输完2个字节后，产生发送中断。FIFO接收到1个或以上字节时，产生接收中断。 10 = 传输完3个字节后，产生发送中断。FIFO接收到3个或以上字节时，产生接收中断。 11 = 传输完4个字节后，产生发送中断。接收FIFO空间已满或有4个字节数据时，产生接收中断。
13	SPI发送FIFO清空使能位。 该位置1时，清空发送FIFO。该位无法自清0；需要一个单次清空操作时，应将该位置1。如果该位的值总保持为1，那么将发送0x00或上次发送的值，具体取决于SPICON[7]的设置。该位为1时，无法对发送FIFO进行写操作。 该位清0时，禁用发送FIFO清空。
12	SPI接收FIFO清除使能位。 该位置1时，清空接收FIFO。该位无法自清0；需要一个单次清空操作时，应将该位置1。该位置1后，所有向接收FIFO写数据的操作将被忽略，且系统不产生中断。如果置1并且SPICON[6] = 0，读取接收FIFO将启动一次数据传输。 该位清0时，禁用接收FIFO清空。
11	连续传输使能。 通过将该位置1，用户可使能连续传输。在主机模式下，数据传输连续进行，直到SPITX寄存器内无有效数据为止。SS置位，并在每一次8位串行传输期间保持置位，直到发送寄存器为空。 通过将该位清0，用户可禁用连续传输。每一次传输都是单独的8位串行传输。如果SPITX寄存器中存在有效数据，那么经过1个串行时钟周期的停转时间后，重新开始传输数据。
10	回送使能位。 通过将该位置1，用户可将MISO连接到MOSI并测试软件。 通过将该位清0，可返回正常模式。
9	从机MISO输出使能位。 将该位置1时，可禁用MISO引脚上的输出驱动器。该位置1后，MISO引脚变为开漏极。 将该位清0时，MISO正常工作。
8	SPIRX上溢覆盖使能。 用户将该位置1后，新接收到的串行数据将覆盖SPIRX寄存器中的有效数据。 用户将该位清0后，新接收到的串行数据会被丢弃。
7	发送FIFO为空时SPI发送0使能位。 若将该位置1，则在发送FIFO无有效数据时，SPI发送0x00。 若将该位清0，则在发送FIFO无有效数据时，SPI发送上次发送的值。
6	SPI传输和中断模式。 通过将该位置1，用户可在向SPITX寄存器写入数据时开始传输。只有当SPITX为空时产生中断。 通过将该位清0，用户可在从SPITX寄存器读取数据时开始传输。只有当SPIRX已满时产生中断。
5	LSB优先传输使能位。 用户将该位置1后，优先发送LSB。 用户将该位清0后，优先发送MSB。

位	描述
4	SPI线或模式使能位。 通过将该位置1，可使能开漏数据输出。数据输出引脚需要外部上拉电阻。 将该位清0时，启动正常输出模式。
3	串行时钟极性模式位。 用户将该位置1后，串行时钟高电平空闲。 用户将该位清0后，串行时钟低电平空闲。
2	串行时钟相位模式位。 用户将该位置1后，串行时钟脉冲出现在每一次串行位传输的起始位置。 用户将该位清0后，串行时钟脉冲出现在每一次串行位传输的末尾。
1	主机模式使能位。 通过将该位置1，用户可使能主机模式。 通过将该位清0，用户可使能从机模式。
0	SPI使能位。 通过将该位置1，用户可使能SPI。 通过将该位清0，用户可禁用SPI。

高压外设控制接口

ADuC7039集成了多个高压电路功能，它们通过一个包括两个MMR(HVCON和HVDAT)的寄存器接口来加以监控。HVCON寄存器充当命令字节解释器，允许微控制器间接地对两个高压状态/配置寄存器中的一个进行8位数据读写操作(值存放在HVDAT内)。这些高压寄存器不是MMR，但通常被称为间接寄存器，即只能通过HVCON和HVDAT MMR间接访问。

HVCON寄存器和间接高压寄存器之间的物理接口采用基于2.56 MHz串行时钟的两线(数据和时钟)式串行接口。因此，从MCU内核将命令写入HVCON到该命令或数据到达间接高压寄存器内需要一定的延时(最长10 µs)。从MCU内核将命令写入HVCON到间接寄存器数据被回读到HVDAT寄存器内也需要10 µs延时。MCU可轮询繁忙位(MCU读取时为HVCON位0)以确认读/写命令何时结束。

图29显示高压接口和相关电路的顶层架构。LIN物理接口通过此接口控制和监控。

高压接口包括两个MMR和两个间接寄存器：

- HVCON是一个8位寄存器，用作高压控制接口的命令字节解释器。写入此寄存器的字节被解释为对高压电路相关的两个间接寄存器的读或写命令。命令代码如表56所示。回读HVCON MMR可知操作是否成功。
- HVDAT是一个12位寄存器，用于保存高压接口寄存器的间接读写数据。
- HVCFG是一个8位寄存器，用于控制高压电路的功能，利用HVCON和HVDAT访问。
- HVSTA是一个8位只读寄存器，反映高压电路的状态。此寄存器不是MMR，因而不会出现在MMR存储器映射内。它通过HVCON寄存器接口访问，其数据通过HVDAT回读。高压中断控制器响应高压中断事件，将高压状态寄存器(HVSTA)的当前值同时自动载入HVDAT寄存器内。

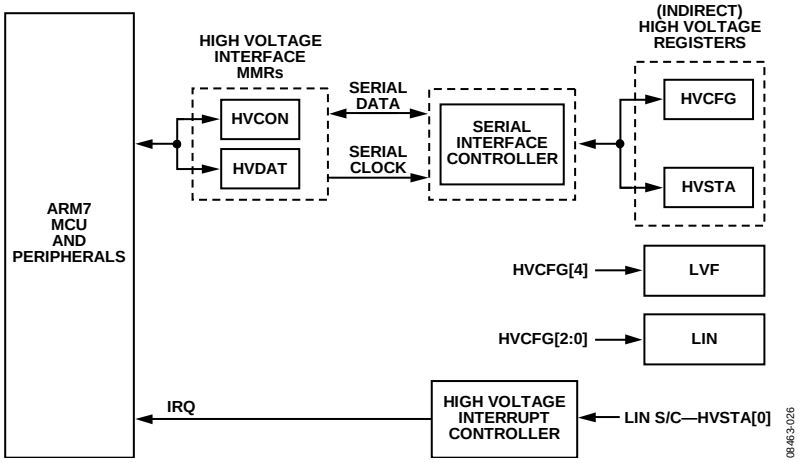


图29. 高压接口顶层框图

高压接口控制接口

名称： HVCON

地址： 0xFFFF0804

默认值： 由内核更新

访问类型： 读/写

读/写 该8位寄存器用作高压控制接口的命令字节解释器。写入此寄存器的字节被解释为对高压电路相关的两个间接寄存器的读或写命令。HVDAT寄存器用来存放间接寄存器的写入或回读数据。

表56. HVCON MMR写入位分配

位	描述
7至0	命令字节。含义为 0x00 = 将高压寄存器HVCFG回读到HVDAT。 0x02 = 将高压状态寄存器HVSTA回读到HVDAT。 0x08 = 将HVDAT内的值写入高压寄存器HVCFG。 其它 = 保留。

表57. HVCON MMR读取位分配

位	描述
7至3	保留。
2	发送到高压芯片命令状态。 1 = 命令成功。 0 = 命令失败。
1	高压芯片读取命令状态。 1 = 命令成功。 0 = 命令失败。
0	繁忙(只读)位。当用户代码读取此寄存器时，位0应视为高压接口的繁忙位。可用来决定读取请求是否已完成。 除非繁忙= 0，否则此表中所述高压(读/写)命令不能写入HVCON。 繁忙 = 1，高压接口繁忙且未完成前一个HVCON写入命令。位1、2无效。 繁忙 = 0，高压接口空闲且已完成HVCON写入命令。位1、2有效。

高压数据寄存器

名称： HVDAT

地址： 0xFFFF080C

默认值： 由内核更新

访问类型： 读/写

读/写 HVDAT是一个12位寄存器，用于保存高压接口寄存器HVDAT、HVSTA、HVCFG的间接读写数据。

表58. HVDAT MMR位分配

位	描述
11至8	高压数据HVDAT[7:0]相关的命令。这些位为只读且应写入0 0x00 = 将高压寄存器HVCFG回读到HVDAT。 0x02 = 将高压状态寄存器HVSTA回读到HVDAT。 0x08 = 将HVDAT内的值写入高压寄存器HVCFG。
7至0	欲读/写的高压数据

高压配置寄存器

名称： HVCFG

地址： 通过HVCON高压接口间接寻址

默认值： 0x00

访问类型： 读/写

功能： 该8位寄存器控制ADuC7039上的高压电路功能。此寄存器不是MMR，因而不会出现在MMR存储器映射内。通过HVCON寄存器接口访问。欲写入该寄存器的数据通过HVDAT MMR加载，从该寄存器回读数据也是通过HVDAT MMR。

表59. HVCFG位分配

位	描述
7至5	保留。
4	低压标志(LVF)使能位。 该位清0时，禁用LVF功能。 该位置1时，使能LVF功能。上电后可通过HVSTA[2] 查询低压标志，以决定REG_DVDD电压先前是否下降到2.1 V以下。
3	电压衰减器诊断使能位。 该位置1时，接通电流源，将一个差分电压加到电压通道测量上。 该位清0时，禁用电压衰减器诊断功能。
2	LIN驱动器重新使能位。 该位置1时，重新使能因为短路电流事件而禁用的LIN驱动器。 该位自动清0。
1	使能/禁用LIN短路保护。 该位置1时，使能LIN引脚被动短路保护。此时LIN引脚上的短路事件会产生高压中断(前提是IRQEN[10]使能)，并置位HVSTA内的相应状态位，但不会禁用短路引脚。 该位清0时，使能LIN引脚主动短路保护。此时在短路事件期间，LIN引脚产生高压中断(IRQSTA[10])，置位HVSTA[0]，并自动禁用短路引脚。禁用后，I/O引脚只能通过写入HVCFG[2]来重新使能。 只有持续时间长于20 µs的LIN短路事件才能被检测到。
0	LIN工作模式。 0 = LIN禁用。 1 = LIN使能。

高压状态寄存器

名称： HVSTA

地址： 通过HVCON高压接口间接寻址

默认值： 0x00

访问类型： 只读，只应在高压中断时读取。

功能： 此8位只读寄存器反映低压标志和LIN短路中断状态。此寄存器不是MMR，因而不会出现在MMR存储器映射内。它通过HVCON寄存器接口访问，其数据通过HVDAT回读。高压中断控制器响应高压中断事件，将高压状态寄存器(HVSTA)的当前值同时自动载入HVDAT寄存器内。

表60. HVSTA位分配

位	描述
7至3	保留。这些位不能使用，保留供未来使用。
2	低压标志状态位。只在通过HVCFG[4]使能时有效。 上电时，如果REG_DVDD已下降至2.1 V以下，则该位为0。此时，RAM内容视为已被破坏。 上电时，如果REG_DVDD未下降至2.1 V以下，则该位为1。此时RAM内容视为有效。只能通过重新使能HVCFG0[4]内的低压标志来清0。
1	保留。该位不能使用，保留供未来使用。
0	LIN短路状态中断。 LIN正常工作时，该位为0；读取HVSTA寄存器后自动清0。 检测到LIN短路时，该位为1。此时LIN驱动器自动禁用。

低压标志(LVF)

ADuC7039有一个低压标志(LVF)，使能时，用户可以监控REG_DVDD(参见“低压标志(LVF)”部分)。通过HVCFG[4]使能后，可通过HVSTA[2]监控LVF的状态。如果REG_DVDD降至2.1 V以下，则HVSTA[2]清0且RAM内容已被破坏。使能低压标志后，只有REG_DVDD降至2.1 V以下或利用HVCFG[4]禁用LVF功能才能使其复位。

处理高压接口中断和高压通信

高压中断

高压电路还集成了一个中断控制器。如果通过IRQEN[10]使能，LIN短路事件将能置位高压中断信号并中断MCU内核。

尽管正常情况下MCU通过跳转到IRQ或FIQ矢量地址来响应此中断事件，但高压中断控制器仍同时自动地将高压状态寄存器(HVSTA)的当前值载入HVDAT寄存器内。此时HVCON[0]内的繁忙位被置1，指示传输正在进行中，并在10 µs后清0，指示可从HVDAT获取HVSTA内容。

该接口应由中断驱动。因此，中断处理程序可一直轮询HVCON内的繁忙位，直到其解除置位。在繁忙位清0后，必须检查HVCON[1]以确保正确读取数据。接下来，可读取HVDAT寄存器。此时，HVDAT保存着HVSTA寄存器的值。

读取HVSTA寄存器将清除中断，因此，不建议随时读取HVSTA。

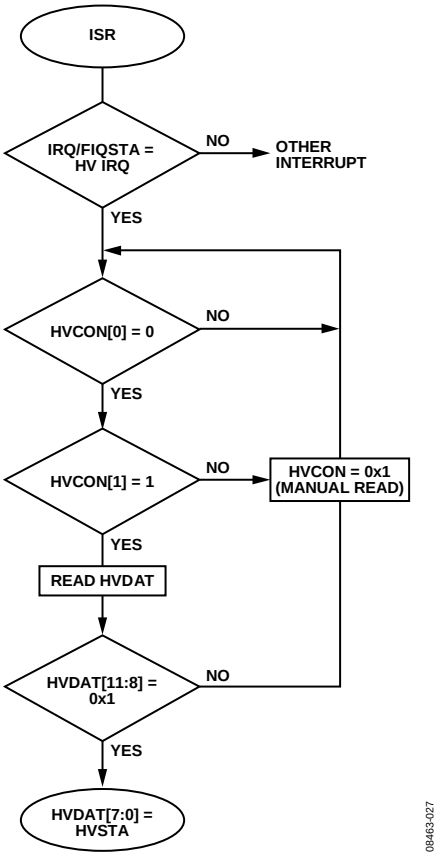


图30. 高压接口中断流程图

高压配置

下面是使能LIN的示例代码。

```
char HVstatus;
      do{                              // Enable LIN
            HVDAT = 0x01;              // Enable LIN physical layer mode
            HVCON = 0x08;              // Write to HVCFG
            do{
                  HVstatus = HVCON;
            }
            while(HVstatus & 0x1);     // Wait until command is finished
      }
      while (!(HVstatus & 0x4));       // Transmit command is correct
```

最好使用一个函数实现高压通信例程，然后在代码中调用此函数。

LIN(局域互连网络)接口

LIN物理接口

ADuC7039在ARM7 MCU内核和外部LIN总线之间具有一个高压物理接口。该LIN接口只能用作从机接口，支持1 kB至20 kB的工作速率并兼容LIN 2.1标准。低于1 kB的频率视为1 kB。片内集成从节点所需的上拉电阻，从而省去了外部电路。为获得最佳的EMC和故障保护性能，LIN引脚上建议使用一些外部元件(见图33)。

如图31所示，LIN协议利用IRQ/FIQ、专用LIN定时器和同样片内集成的高压收发器来仿真。当器件处于休眠模式下，LIN时钟从低功耗振荡器获得；正常模式下则从内核时钟获得。

LIN收发器通过2线式接口HVCFG[0]使能。LIN协议由LIN MMR部分所述的8个MMR控制。

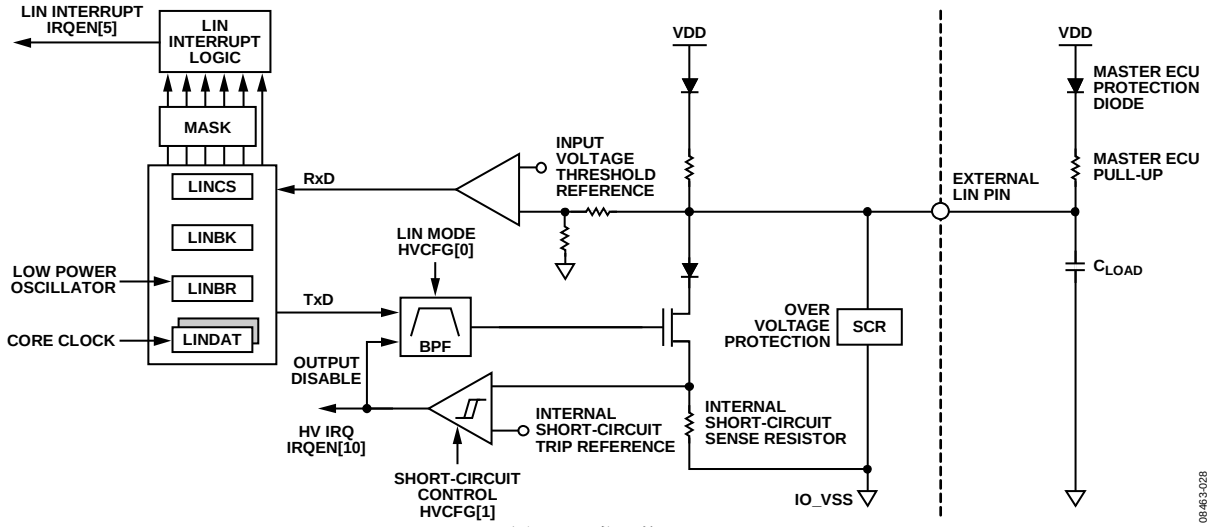


图31. LIN物理接口

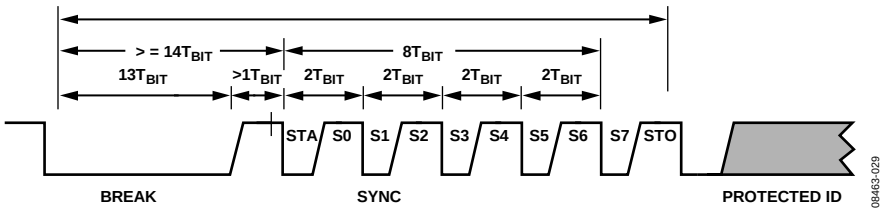


图32. LIN帧

LIN诊断

ADuC7039的LIN引脚上具有短路保护功能。如果在LIN引脚上检测到短路状况，HVSTA[0]就会置1。如果通过IRQEN[10]使能了中断，上述状况将产生一个高压中断。利用HVCFG1[2]重新使能LIN驱动器可将该位清0。可通过HVCFG[1]来禁用此功能。

LIN通信

LIN使用帧进行数据通信。帧包括报头、断开码、同步码、主机发送的PID、数据字节以及从机产生的校验和，如图32所示。在LIN通信中，PID决定从机的行为：接收、发送或忽略。

断开码

ADuC7039的LIN接口自动检测断开码，接收到有效的断开码后，就会在LINSTA寄存器中设置一个标志。断开码的最短长度为11个标称从机时钟周期(20 kB速率)。有效断开码的最长长度是可编程的，可以在LINBK MMR中配置。LIN接口在LIN通信期间的任何时候都能识别有效的断开码，如果现有通信期间出现断开码，就会置位冲突标志位(LINSTA的位4)。

同步码

LIN接口自动检测同步字节，并且为当前LIN帧的后续数据设置波特率。该操作对用户是透明的。

PID

LIN外设将PID视作一个数据字节。它在LINDAT MMR中提供。软件必须解码PID。接收到PID停止位后，产生一个中断以便读取LINDAT寄存器的内容，然后该内容将被下一个数据字节覆盖。

数据字节

同样，后续数据字节也会设置中断位，指示已接收到数据字节。

在发送模式下，一次最多可以发送8个数据字节，后跟一个校验和。

校验和

每接收或发送一个字节时都会自动计算校验和，结果的倒数值存储在LINCS MMR中。默认情况下，校验和计算采用LIN 2.1规范：诊断帧和保留帧使用传统校验和计算，其余帧则使用增强校验和计算。硬件自动识别PID并相应地计算校验和。用户代码无需写入LINCON寄存器以更改校验和计算。

若要在LIN 1.3模式下工作，用户代码LIN初始化例程必须将LINCON[7]置1，迫使硬件采用传统校验和计算。在接收模式下的LIN通信期间，不得更改此寄存器。

LIN MMR

LIN模块接口包括8个MMR：

- LINCON是一个16位控制寄存器。该寄存器如表61所示。数据接收期间不应访问该寄存器。
- LINDAT是一个8位用户可访问数据寄存器。此MMR为双缓冲型寄存器：一个阴影寄存器用于接收和发送数据，用户代码则读写LINDAT。LINSTA MMR指示数据的状态。
- LINSTA是一个16位状态寄存器。该寄存器如表62所示。
- LINBR是一个19位波特率寄存器。波特率由LIN外设自动设置，用户代码不应更改此寄存器。它表示8位传输期间内核时钟的周期数。
- LINBK是一个19位断开码定时器寄存器，控制断开码被LIN从机接口视为有效的最大长度。默认值为5500个内核时钟周期，代表11位数据以20 kHz的速率发送所需的时间。
- LINCS是一个8位校验和寄存器。它包含当前校验和计算的倒数结果。校验和在接收或发送每个字节时计算，具体取决于LINCON MMR的位4设置。发送模式下，最后要发送的数据被写入外设后，用户将该位置1。校验和自动发送。接收模式下，校验和在接收到每个字节时计算，无论此字节是数据字节还是帧校验和。例如，接收一个含4个数据字节的帧时，一旦接收到第四个数据字节，LINCS MMR就会包含预期的帧校验和。接收到正确的校验和之后，LINCS应包含0x00。
- LINLOW是一个19位计数器，以10 MHz的时钟速率工作，用于唤醒总线上的LIN节点。只要LINLOW MMR大于0，就会强制LIN总线变为低电平。例如，如果LINLOW = 0x234，则会以20 kB的速率产生一个11位断开码。
- LINWU是一个19位计数器，从低功耗振荡器获得时钟。它用在休眠模式下，以指定通过LIN唤醒器件的低电平信号的长度。例如，如果LINWU = 0x13，则会忽略任何短于150 μs的断开码，ADuC7039保持休眠状态。

LINCON寄存器

名称： LINCON

地址： 0xFFFF0700

默认值： 0x0000

访问类型： 读/写

功能： 该16位MMR控制LIN外设。

LINCS寄存器

名称： LINCS

地址： 0xFFFF0704

默认值： 0xFF

访问类型： 读/写

功能： 8位校验和寄存器。

LINBR寄存器

名称： LINBR

地址： 0xFFFF0708

默认值： 0x00FA0

访问类型： 读/写

功能： 19位波特率寄存器。

LINBK寄存器

名称： LINBK

地址： 0xFFFF070C

默认值： 0x0000157C

访问类型： 读/写

功能： 19位断开码定时器寄存器。

LINSTA寄存器

名称： LINSTA

地址： 0xFFFF0710

默认值： 0x0100

访问类型： 只读

功能： 16位状态寄存器。

LINDAT寄存器

名称： LINDAT

地址： 0xFFFF0714

默认值： 0x00

访问类型： 读/写

功能： 8位数据寄存器。

LINLOW寄存器

名称： LINLOW

地址： 0xFFFF0718

默认值： 0x00000

访问类型： 读/写

功能： 19位寄存器，产生一个LIN断开码以唤醒总线上的其它LIN外设。

LINWU寄存器

名称： LINWU

地址： 0xFFFF071C

默认值： 0x00013

访问类型： 读/写

功能： 19位寄存器，指定唤醒器件的断开码长度。

共有7个中断源，其中5个可以在LINCON MMR中予以屏蔽：

- LIN唤醒
- 接收到数据
- 发送就绪—可屏蔽
- 发送完成—可屏蔽
- 检测到冲突—可屏蔽
- 断开码—可屏蔽
- 帧错误范围内的最大负边沿数—可屏蔽

表61. LINCON MMR位分配

位	描述
15至13	保留。
12	LIN旁路位。 该位由用户代码置1时，仅控制LIN收发器，用于LIN一致性测试。 该位由用户代码清0时，在正常模式下工作。
11	LIN使能位。 该位由用户代码置1时，使能LIN接口。 该位由用户代码清0时，禁用LIN接口或复位接口。
10	UART使能位。 该位由用户代码置1时，允许不接收帧报头的传输，用于测试目的。 该位由用户代码清0时，在正常模式下工作。
9	同步码的定时，位0(单从机系统不需要)。 确保发送第二个断开码时，它能被识别为断开码，而不是作为同步码的一部分加以定时。如果起始码超过该位规定的时钟周期数，器件将认为它现在接收的是断开码，并继续计数低周期，以确定该断开码是否满足LINBK MMR中定义的最短时间要求。 由用户置1。同步码的第一位必须短于750个内核时钟周期(73 µs)。 通过将该位清0，用户可禁用此功能。
8	发送校验和。 该位由用户置1时，自动发送校验和。该位必须在发送完LINDAT中写入的最后一个数据字节以及设置好发送就绪位之后设置。 当校验和发送完毕时，该位由硬件自动清0。
7	校验和计算。 该位由用户置1时，自动计算传统校验和(不包括PID)。 该位由用户清0时，计算增强校验和(包括PID)。 通信期间，如果收到PID后修改该位的值，则复位校验和。
6	冲突检测和发送完成中断屏蔽。 通过将该位置1，用户可禁用冲突检测和发送完成中断。 通过将该位清0时，用户可使能冲突检测和发送完成中断。 使能中断时，无论发送完成位是何状态，所有检测到的冲突事件都会引起中断，且冲突状态位置位。使能屏蔽时，一旦发送完成，置位的冲突事件就会被屏蔽，不引起中断，且检测到冲突状态位也不会置1。即使用户已将该屏蔽位置1，但如果器件尚未置位发送完成状态位，则冲突仍会引起中断，并且检测到冲突状态位置1。
5	负边沿最大误差中断屏蔽。 通过将该位置1，用户可禁用负边沿最大误差中断。 通过将该位清0，用户可使能负边沿最大误差中断。如果负边沿计数器计数到一个帧含57个以上的边沿，就会产生中断。
4	冲突检测中断屏蔽。 通过将该位置1，用户可禁用冲突检测中断。 通过将该位清0，用户可使能冲突检测中断。
3	接收到断开码中断屏蔽。 通过将该位置1，用户可禁用断开码接收中断。 通过将该位清0，用户可使能断开码接收中断。
2	发送完成中断屏蔽。 通过将该位置1，用户可禁用发送完成中断。 通过将该位清0，用户可使能发送完成中断。
1	发送就绪中断屏蔽。 通过将该位置1，用户可禁用发送就绪中断。当LINCON[8]置1时，该位自动置1。 通过将该位清0，用户可使能发送就绪中断。
0	接收/发送模式。 该位由用户代码置1时，则在解码PID后发送数据字节。 接收到断开码时，该位自动清0。

表62. LINSTA MMR位分配

位	描述
15至11	保留。
10	LIN唤醒中断。 如果LIN唤醒ADuC7039，该位置1。唤醒功能(LINWU MMR)只能在POWCON[3] = 0时使用。 读取LINSTA MMR时，该位自动清0。
9	断开码时间最大值。 如果使能LIN接口后的第一个断开码在达到最大计数值之前结束，则该位为0。 如果使能LIN接口后的第一个断开码在达到最大计数值之后结束，则该位为1。 读取LINSTA MMR时，该位由硬件自动清0。 该位仅对通过LINCON[11]使能LIN外设后的第一个断开码有效。
8	PID奇偶校验错误。 如果LINDAT寄存器中的当前字节不能与LIN 2.1规范所述PID奇偶校验方案正确匹配，则该位由硬件自动置1。 如果LINDAT寄存器中的当前字节与PID奇偶校验方案正确匹配，则该位由硬件清0。 数据寄存器中何时真正有PID，须由用户判断。无论LINDAT MMR中是否有PID或数据字节，都会进行奇偶校验。
7	校验和匹配。 该位仅当LINSTA[0] = 1时有效。 如果LINCS中的值与LINDAT中的已接收数据不匹配，则该位自动置1。 读取LINSTA MMR时，或者LINDAT与LINCS匹配时(同时LINSTA[0] = 1)，该位清0。
6	帧错误。 如果没有帧错误，该位为0。读取LINSTA时，该位自动清0。 如果在接收数据字节期间发生帧错误，即未检测到有效停止，则该位置1。
5	负边沿最大误差中断(可屏蔽)。 如果负边沿数没有超过帧允许的数量，则该位为0。 如果负边沿数为57或更大，则该位置1。 读取LINSTA MMR时，该位由硬件自动清0。
4	LIN冲突检测中断(可屏蔽)。 如果器件由于总线冲突而停止传输，该位将由硬件自动置1。如果冲突检测和发送完成中断使能(LINCON[6] = 0)，并且发送完成中断位置1 (LINSTA[2] = 1)，则该位不会置1。 读取LINSTA MMR时，该位由硬件自动清0。
3	断开码接收中断(可屏蔽)。 如果未接收到有效断开码，该位为0。 如果在LIN总线上检测到11个标称位的低电平时间，则该位置1。读LINSTA MMR时，该位自动清0。
2	发送完成中断(可屏蔽)。 如果数据仍在LINDAT寄存器中，该位为0。 发送完所有数据后，该位置1。然后它将保持1值，直至LIN接收到一个断开码。接收模式下，它自动清0。
1	发送就绪中断(可屏蔽)。 如果先前写入LINDAT MMR中的数据仍然在LINDAT中，而尚未移送到发送寄存器，则该位为0。当发送就绪位为0时，写入数据到LINDAT MMR会覆盖先前要发送的字节。 如果先前写入LINDAT MMR中的数据现已处于发送寄存器中，则该位为1。
0	接收就绪中断。 如果LINDAT MMR中没有要读取的新数据，该位为0。 如果LINDAT MMR中有要读取的新数据，该位为1。读取LINDAT MMR时，该位清0。

器件标识

为便于追溯，器件上电时提供标识信息。上电时，ARM内部寄存器(R4至R6)提供制造批次ID、硅片掩模版本和内核版本等信息，如表65和表66所示。

为获得全面的追溯信息，需要记录R4、R5、R6和产品型号。R6中存储的组装批次ID是封装标识的一部分，如表63所示。

例如，对于2528206.1的组装批次ID，R6包含0x002693CE。上电时，产品型号包含在MMR FEEADR中。

表63. 标识实例 ADuC7039BCP6Z

行	LFCSP
行1	ADuC7039
行2	BCP6Z
行3	B60 #日期代码
行4	2528206.1

表64. 标识实例 ADuC7039WBCPZ

行	LFCSP
行1	ADuC7039
行2	WBCPZ
行3	B60 #日期代码
行4	2528206.1
行5	MALAYSIA

表65. 上电时的R4位分配

位	描述
31至27	晶圆号。从此位置读取的5位值提供晶圆制造批次ID(此器件出处)中的晶圆号(1至24)。配合R4[26:0]使用时，它提供个别晶圆可追溯性。
26至22	晶圆批次制造厂。从此位置读取的5位值反映与此晶圆批次相关的制造厂。配合R4[21:0]使用时，它提供晶圆批次可追溯性。
21至16	晶圆批次制造ID。从此位置读取的6位值是晶圆批次制造ID的一部分，配合R4[26:22]和R4[15:0]使用时，提供晶圆批次可追溯性。
15至0	晶圆批次制造ID。这16位LSB被解读为晶圆制造批次ID号。配合R5值(即制造批次ID)使用时，此编号是器件的唯一标识。

表66. 上电时的R5位分配

位	描述
31至28	硅片掩膜版本ID。此4位反映硅片掩膜ID号。具体来说，这半字节的16进制值应被解码成表示ASCII字符A至O的16进制数的低位半字节。例如： 位[19:16] = 0001 = 0x1，因此该值应解释为41，对应ASCII字符A，表示硅片掩膜版本A。 位[19:16] = 1011 = 0xB，因此该值应解释为4B，对应ASCII字符K，表示硅片掩膜版本K。 这个值的范围为1至15，解释为41至4F或ASCII字符A至O。
27至20	内核版本ID。此字节的16进制数应解释为一个ASCII字符，表示片内Flash/EE存储器的嵌入内核固件版本。例如，如果从此字节读取0x41，则解释为A，表示片上内核版本A。
19至16	保留。对于预发行样片，这些位表示器件内核的局部修订版本号。
15至0	器件ID。这16位LSB被解释为器件ID号。配合R4值(即制造批次ID)使用时，此编号是器件的唯一标识。

系统标识FEEADR

名称： FEEADR

地址： 0xFFFF0E10

默认值： 0xF009

访问类型： 读/写

功能： 该16位寄存器决定通过FEECON执行任何Flash/EE命令的操作地址。

注意： 该MMR还用以识别ADuC703x系列产品和预发布芯片版本。

表67. FEEADR系统标识MMR位分配

位	描述
15至4	保留
3至0	ADuC703x系列ID 0x0 = ADuC7030 0x2 = ADuC7032 0x3 = ADuC7033 0x4 = ADuC7034 0x6 = ADuC7036 0x9 = ADuC7039 其它 = 保留供将来使用

推荐原理图

本原理图包含为使ADuC7039正常工作而推荐使用的外部器件。

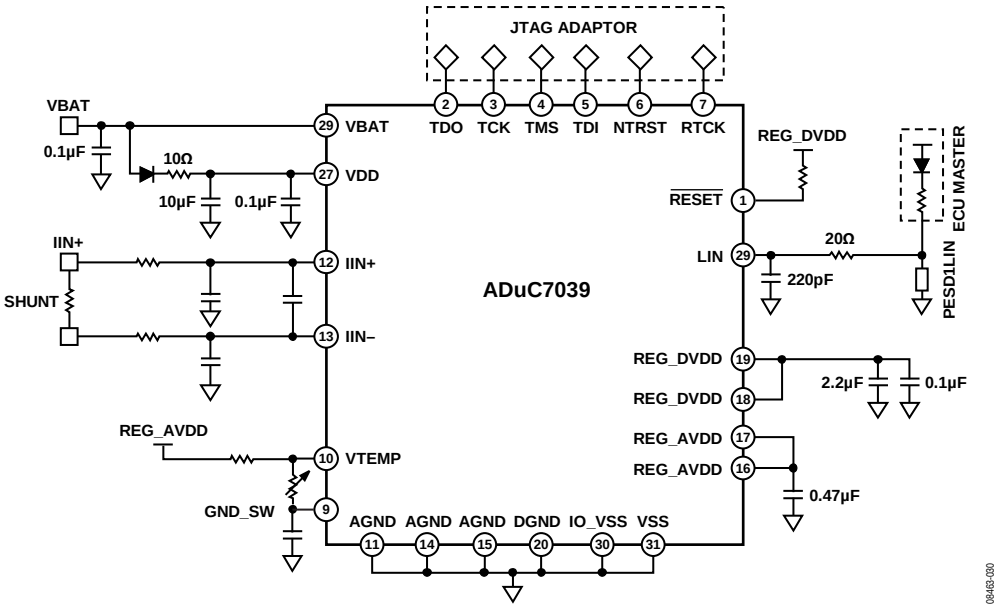
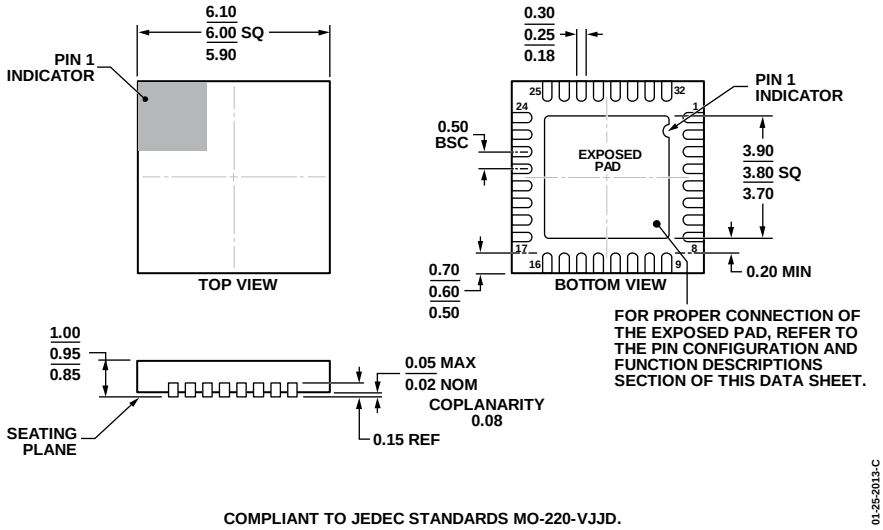


图33. 推荐原理图

外形尺寸



COMPLIANT TO JEDEC STANDARDS MO-220-VJJD.

图34. 32引脚引脚架构芯片级封装[LFCSP_VQ]
6 mm x 6 mm超薄四方体(CP-32-15)
图示尺寸单位：mm

订购指南

型号[1,2]	NOTE	温度范围	Flash/Ram	封装描述	封装选项
ADuC7039BCP6Z		−40°C至115°C	64K/4K	32引脚引脚架构芯片级封装[LFCSP_VQ]	CP-32-15
ADuC7039BCP6Z-RL		−40°C至115°C	64K/4K	32引脚引脚架构芯片级封装[LFCSP_VQ]	CP-32-15
ADuC7039WBCPZ	[3]	−40°C至115°C	64K/4K	32引脚引脚架构芯片级封装[LFCSP_VQ]	CP-32-15
ADuC7039WBCPZ-RL	[3]	−40°C至115°C	64K/4K	32引脚引脚架构芯片级封装[LFCSP_VQ]	CP-32-15
EVAL-ADUC7039QSPZ				评估板	

[1] Z = 符合RoHS标准的器件。

[2] 通过汽车应用认证。

[3] 推荐新设计使用。

汽车应用产品

ADuC7039W生产工艺受到严格控制，以满足汽车应用的质量和可靠性要求。请注意，车用型号的技术规格可能不同于商用型号；因此，设计人员应仔细阅读本数据手册的技术规格部分。只有显示为汽车应用级的产品才能用于汽车应用。欲了解特定产品的订购信息并获得这些型号的汽车可靠性报告，请联系当地ADI客户代表。

注释

D08463sc-0-3/13(D)