Example Description

SPD1188 Peripheral Examples

Revision 1 – June 2020

1 示例工程说明

SPD1188 示例工程提供了覆盖每个外设主要特性的代码。每个示例代码可以直接用KEIL MDK 进行编译，然后将编译后的代码下载到目标芯片中即可运行。SPD1188 示例工程包含的示例代码如下:

模块名	示例	
	工程名	描述
ADC	ADC_Calibration	校准ADC 的增益及偏置数值。
	ADC_Continous_Mode	设置ADC 工作在连续采样模式（使用某一SOC结束信号触发另外一个SOC 开始工作）。
	ADC_Conversion_Priority	设置ADC SOC 优先级，使高优先级SOC 得以首先进行采样并转换。
	ADC_Differential_Trim_Result	设置ADC 为双端模式，并获取trim 之后的结果。
	ADC_Sequential_Mode	设置ADC 为工作在顺序采样模式（SPC1168 有3个ADC 采样器，分别为A/B/C，默认情况下采样器A 首先工作，然后是B，最后是C）。
	ADC_SHA_Open_Detect	检测A 采样器通路上被采样的ADC GPIO 是否存在虚焊等导致开路的情况。
	ADC_SHA_PPU	开启ADC 的PPU 功能，并使用PPU 检测ADC 采样结果是否超过设定的阈值范围。
	ADC_SHA_Short_Detect	检测A 采样器通路上被采样的ADC GPIO 是否存在短路情况。
	ADC_Simultaneous	设置ADC 模块的A/B 两个采样器同时进行采样。
	ADC_Single_End_Result	设置ADC 工作在单端模式（有一端被接到GND）。
	ADC_With_PGA	ADC 工作在双端模式，对经过PGA 放大后的两个ADC GPIO 进行采样。
AES	AES	设置AES 工作在CBC 模式，并对随机数据进行加密，随后解密，对比原始数据和解密后的数据，验证是否相同。
Comparator	Comparator	设置Comparator 分别以DAC2 和DAC3 作为高低阈值的参考电压，以检测Comparator 的输入是否超过此阈值。
CRC	CRC_Calculate	设置CRC 工作在16CCITT 模式，并计算一段字符串，将结果与网上计算过的结果对比，以检查CRC 是否工作正常。
DAC	DAC	设置DAC 输出某一数值的电压，并利用ADC 进行采样，判断ADC 的结果是否在合理范围区间，以此判断DAC 是否工作正常。
ECAP	ECAP_Continue_Absolute	设置ECAP 工作在连续采样的Absolute 模式，并连续对PWM 波形进行事件采样，以此连续计算PWM 的频率。
	ECAP_Countinue_Delta	设置ECAP 工作在连续采样的Delta 模式，并连续对PWM 波形进行事件采样，以此连续计算PWM 的频率。
	ECAP_Oneshot_Absolute	设置ECAP 工作在oneshot 的Absolute 模式，并对PWM 波形进行4 次事件采样，以此计算PWM的频率。
FLASH	Flash_EEPROM_Emulation	演示如何将flash 模拟成EEPROM，以增加flash的使用寿命。
	Flash_Frequency_Reduction	演示如何在系统运行过程中切换flash 的工作频率。
	Flash_M_Access_User	演示一个程序如何访问存储在flash 上另一个存储区域的函数，与此示例相配合的另外一个示例名称

		为“Flash_User_FuncWrap”。
	Flash_Operation	演示flash 的读写擦操作。
	Flash_Sector_Protect	设置flash 的某一扇区（sector）为保护模式，对此扇区的写入及擦除操作将失败。
	Flash_User_FuncWrap	配合“Flash_M_Access_User”的示例代码，此示例演示如何将一个单独的函数接口存储在flash 中，并能够被其它程序访问。
	Flash_With_INT	此示例代码演示，如何新建一个工程，将所有的代码及数据全部放在RAM 中运行（包括终端向量表）。此工程示例，在写/擦除Flash 时，可能会有终端产生的应用情形中会用到。
GPIO	GPIO_Edge_Detect	检测GPIO 是否发生电平反转（采用边沿检测方式判断）。
	GPIO_Level_Test	检测GPIO 电平为高还是低。
GTimer	GTimer_INT	设置Timer 开始计时，结束后进入中断。
I2C	I2C_Master_Bulk_Polling_TxRx	设置I2C 作为Master 工作在Bulk 模式，并用polling 模式向Slave 发送数据，并用polling 模式接收Slave 返回的数据，对比发送和接收的数据是否一致；此示例为Master 端代码，与“I2C_Slave_Bulk_Polling_TxRx”配套。
	I2C_Master_INT_Rx	设置I2C 作为Master，采用polling 方式向Slave发送数据，并采用中断方式接收Slave 返回的数据。
	I2C_Master_Polling_TxRx	设置I2C 作为Master，并用polling 模式向Slave发送数据，并用polling 模式接收Slave 返回的数据，对比发送和接收的数据是否一致；此示例为Master 端代码，与“I2C_Slave_Polling_TxRx”配套。
	I2C_Slave_Bulk_Polling_TxRx	与“I2C_Master_Bulk_Polling_TxRx”配套，此示例代码为Slave 端代码。
	I2C_Slave_Polling_TxRx	与“I2C_Master_Polling_TxRx”配套，此示例代码为Slave 端代码。
PGA	PGA_Calibration	校准PGA 的增益及偏置数值。
High Voltage	High_Voltage_BOD_VBAT_VDDG	使能VBAT 和VDDG 的BOD 功能，当发生BOD 事件时，触发PWM Trip-Zone，关闭PWM 波形。
	High_Voltage_Buck3_FREQ_Dithering	使能Buck3 的Dithering 模式（抖频模式），由于示波器很难看出PFM 模式的抖动频率，所以示例代码中展示了在PWM 模式时抖频的方法，通过PWM 模式，可以在示波器中清晰的看出频率的抖动。
	High_Voltage_Buck3_FREQ_Setting	演示如何设置BUCK3 的频率。
	High_Voltage_BuckG_Cross_Protection	使能BuckG 的保护机制，使能之后，能隔绝Buck3 对BuckG 的影响。
	High_Voltage_BuckG_FREQ_Dithering	使能BuckG 的Dithering 模式（抖频模式），由于示波器很难看出PFM 模式的抖动频率，所以示例代码中展示了在PWM 模式时抖频的方法，通过PWM 模式，可以在示波器中清晰的看出频率的抖动。
	High_Voltage_BuckG_FREQ_Setting	演示如何设置BUCKG 的频率。
	High_Voltage_BuckG_Self_Shut_Down	将VDDG 欠压事件作为高压部分的Trip-Zone 事件，并使能BuckG 的自关闭功能，当VDDG 欠压事件发生时，触发Trip-Zone 事件，BuckG 自关断。
	High_Voltage_BuckG_UVTH	设置BuckG 欠压阈值，如果BuckG 的输入电压（也即VBAT 电压）低于设置数值时，将关闭BuckG。
	High_Voltage_Deep_Sleep	强制高压部分进入休眠模式，BuckG 和Buck3 将会关闭，通过拉低xRSTn 管脚可从休眠状态唤醒。
	High_Voltage_Motor_PGA	通过高压部分自带的PGA 监测感应电阻电压。
	High_Voltage_PRE_Driver	使能低压部分的PWM，并将波形送到高压

		Pre-driver，使其产生三项波形。
	High_Voltage_PRE_Driver_Self_Shut_Down	将VDDG 欠压事件作为高压部分的Trip-Zone 事件，并使能BuckG 的自关闭功能，当VDDG 欠压事件发生时，触发Trip-Zone 事件，Pre-driver 自关断。
	High_Voltage_TSensor_Over_Temperature_Measurement	通过ADC对高压部分的温度传感器进行电压数值采样，通过公式计算出目前高压部分温度。
	High_Voltage_VBAT_Div_30_Monitored_By_ADC	通过ADC测量VBAT/30的数值。
	High_Voltage_VDS_Monitoring	展示场管过流监测的用法。当三项中的任何一项发生过流时，将会触发PWM Trip-Zone，立即停止PWM波形的输出。
PWM	PWM_Complementary_Pair_Channel	设置某路PWM，并使其两个通道（A/B）产生一对互补的波形。
	PWM_Current_Protect_Trigger_TZ	采用PGA 对电流监测点的电压进行放大，并送入Comparator 进行过流或欠流检测，最后设置PWM接收Comparator 的信号作为触发TZ 的触发源，以此关闭发生过流或者欠流的PWM 波形。
	PWM_Force_SYNC	PWM0 发出同步信号同步使PWM1/2 的波形与PWM0 同步。
	PWM_Global_GPIO_SYNC	采用GPIO 同步PWM0/1/2 的波形。
	PWM_Global_Software_Force_SYNC	采用软件同步信号同步PWM0/1/2 的波形。
	PWM_Global_Timer_SYNC	采用计时器同步PWM0/1/2 的波形。
	PWM_GPIO_Trigger_Trip_Zone	设置GPIO 为触发Trip Zone 的触发源，示例中GPIO16 作为TZ0 触发源，并设置为CBC 模式，当为低电平是触发TZ 事件，PWM 关闭波形输出，直到清除TZ 标志位。GPIO17/18 作为TZ1/2触发源，并设置为one-shot 模式，当为低电平时，触发TZ 事件，PWM 关闭波形输出，但在下一个PWM 时钟又立马恢复波形输出。
	PWM_Single_Output_With_Down_Counting_Mode	PWM 以向下计数模式输出单通道波形。
	PWM_Single_Output_With_Up_Counting_Mode	PWM 以向上计数模式输出单通道波形。
	PWM_Single_Output_With_Up_Down_Counting_Mode	PWM 以上下计数模式输出单通道波形。
	PWM_TBCNT_0_SYNC	设置PWM0 在TBCNT=0 时发出同步信号，并发送给PWM1/2，使PWM1/2 波形与PWM0 同步。
	PWM_TBCNT_CMPD_SYNC	设置PWM0 在TBCNT=CMPD 时发出同步信号，并发送给PWM1/2，使PWM1/2 波形与PWM0同步。
	PWM_Trigger_ADC_Sample	设置PWM0 触发ADC A/B/C 三个采样器同时工作。
	PWM_TripPhase_Waveform	设置PWM 输出电机三相波形。
SSP	SSP_Master_B2B_TxRx_Polling	SSP 作为Master 以B2B 模式采用polling 方式向Slave 发送数据，并以polling 方式接收Slave 返回的数据，对比源数据与返回数据，检查数据是否一致；此示例为Master 端代码，与Slave 端的“SSP_Slave_TxRx_Polling”配套。
	SSP_Master_TxRx_INT	SSP 作为Master 采用中断方式向Slave 发送数据，并以中断方式接收Slave 返回的数据，对比源数据与返回数据，检查数据是否一致；此示例为Master 端代码，与Slave 端的“SSP_Slave_TxRx_INT”配套。
	SSP_Master_TxRx_Polling	SSP 作为Master 采用polling 方式向Slave 发送数据，并以polling 方式接收Slave 返回的数据，对比源数据与返回数据，检查数据是否一致；此示例为Master 端代码，与Slave 端的“SSP_Slave_B2B_TxRx_Polling”配套。

	SSP_Slave_B2B_TxRx_Polling	与“SSP_Master_B2B_TxRx_Polling”配套，此示例为Slave 端代码。
	SSP_Slave_TxRx_INT	与“SSP_Master_TxRx_INT”配套，此示例为Slave端代码。
	SSP_Slave_TxRx_Polling	与“SSP_Master_TxRx_Polling”配套，此示例代码为Slave 端代码。
T-Senser	TSensor	设置ECU 内部温度传感器，检测芯片温度。
UART	UART_RX_and_SentBack	此示例为UART 的数据接收端代码，采用polling方式接收发送端的数据，并将接收的数据发送回发送端。
	UART_RX_INT	UART 以中断方式接收发送端的数据。
	UART_TX_and_CheckRX	此示例为UART 的数据发送端代码，采用polling方式发送数据给接收端，并将接收接收端返回的数据进行数据对比。
WDT	WDT0_INT	WTD 计时500ms，结束后进入中断。
	WDT1_Feed_DOG	WTD 计时500ms，但在结束之前重新喂狗，使其不进入中断。
	WDT1_Rset_SYS	WTD 计时500ms，进入中断后不清中断，使系统复位。