Escuela Universitaria
Politécnica - La Almunia
Centro adscrito
Universidad Zaragoza

ESCUELA UNIVERSITARIA POLITÉCNICA
DE LA ALMUNIA DE DOÑA GODINA (ZARAGOZA)

Esquemas y planos

Diseño e implementación de un BMS para BVE (Battery Electric Vehicle)

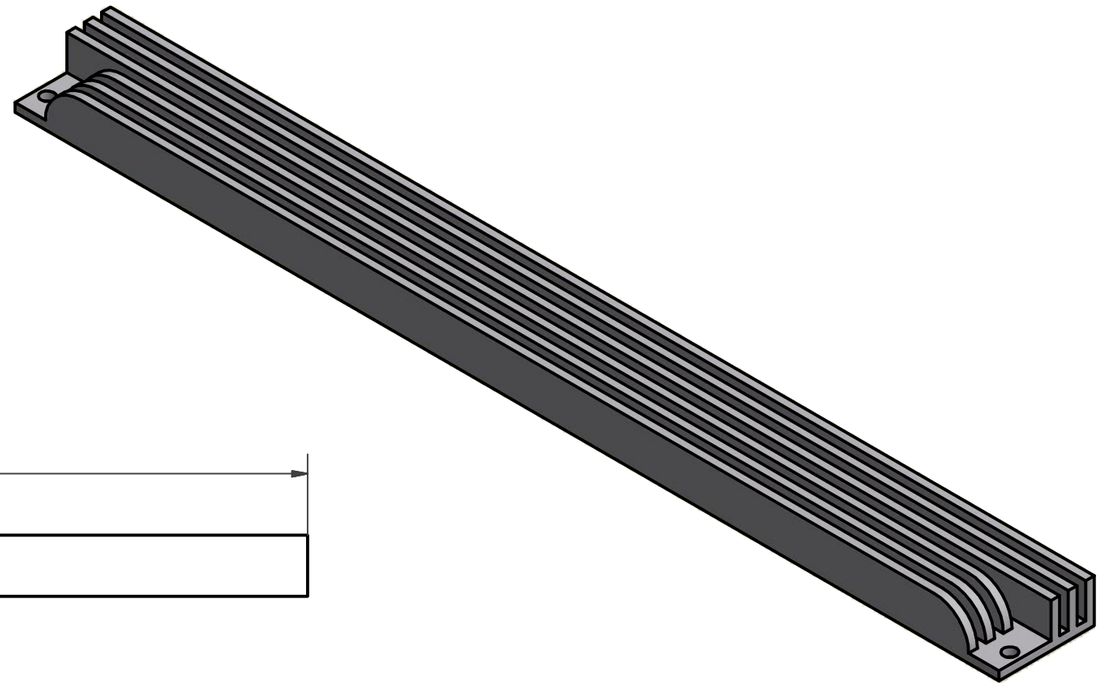
Design and implementation of a BMS for BVE (Battery Electric Vehicle)

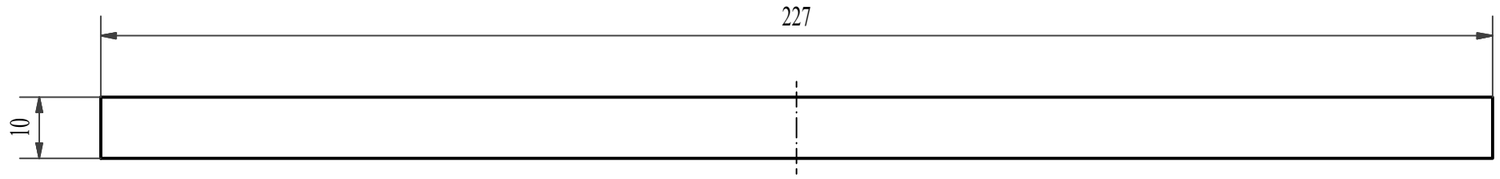
424.18.63

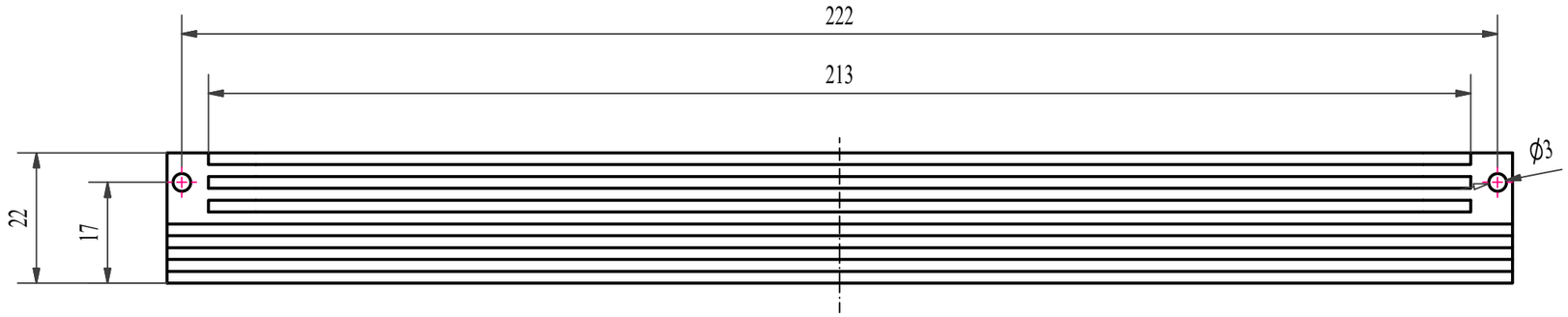
Autor: Alfonso Mareca Miralles

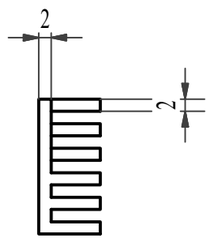
Director: Dr. David Asiain Ansorena

Fecha: 27 de noviembre del 2018









1	1	Disipador de balanceo	424.18.63.000	Aluminio extruido - 6063 300x22x15
MARCA	CTDAD	DENOMINACIÓN Y CARACTERISTICAS	Nº PLANO / ABRE. NORMA	MATERIAL/OBSERVACIONES

Observaciones Generales

Proyecto: Diseño de un BMS para BVE
Palabras clave:
Empresa: EUPLA
Estado del proyecto: Finalizado
Versión: V4

Observaciones de plano

Plano nº: 1 **de:** 1
Formato: A3
Coment: Las aletas tienen un espaciado de 2mm.
Todos los chaflanes a 8mm.

	Fecha	Nombre
Dibujado	24/11/2018	Alfonso Mareca
Comprobado	04/11/2009	
Idem.s.normas		UNE-EN

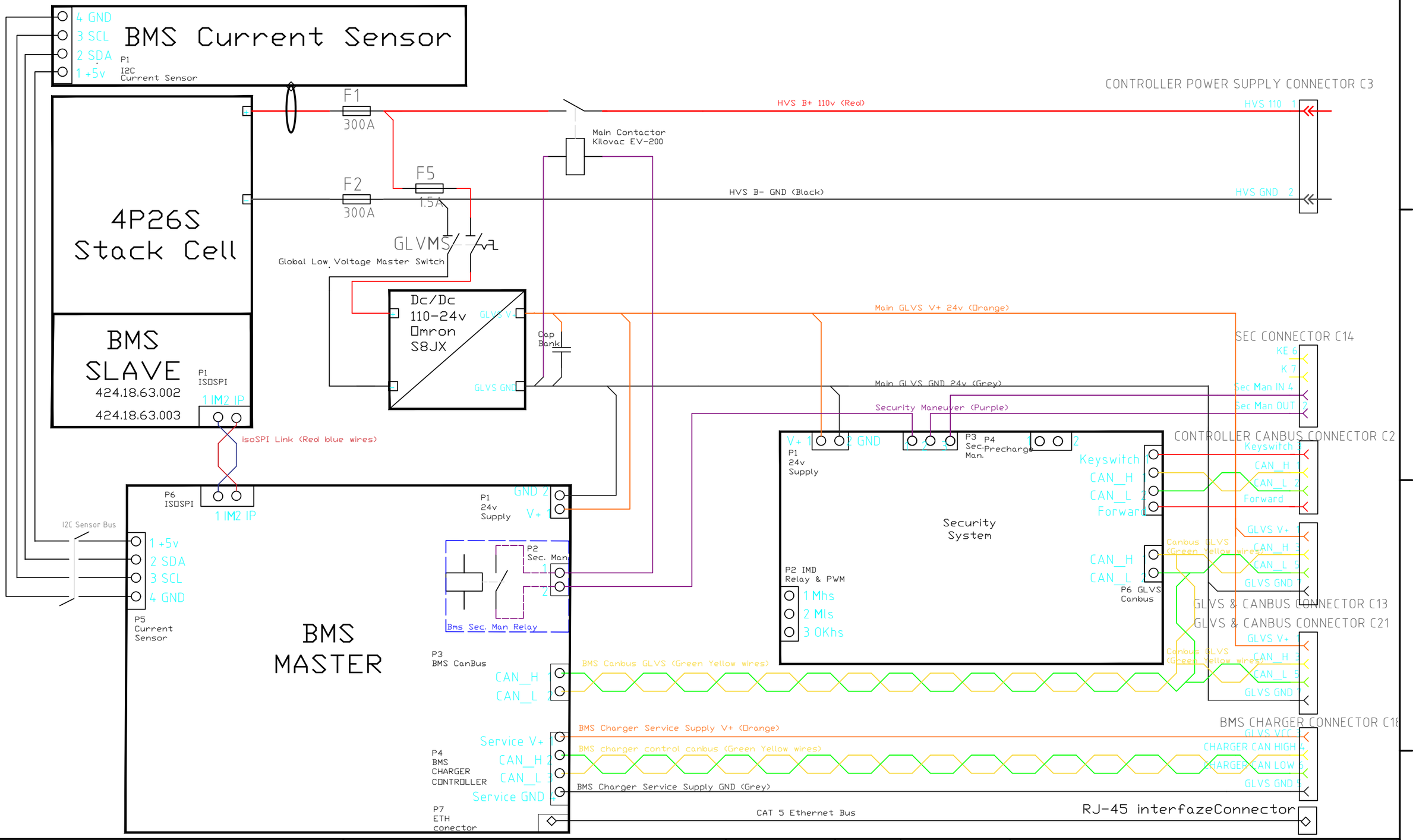
ESCALA 1:1

Diseño de un BMS para BVE

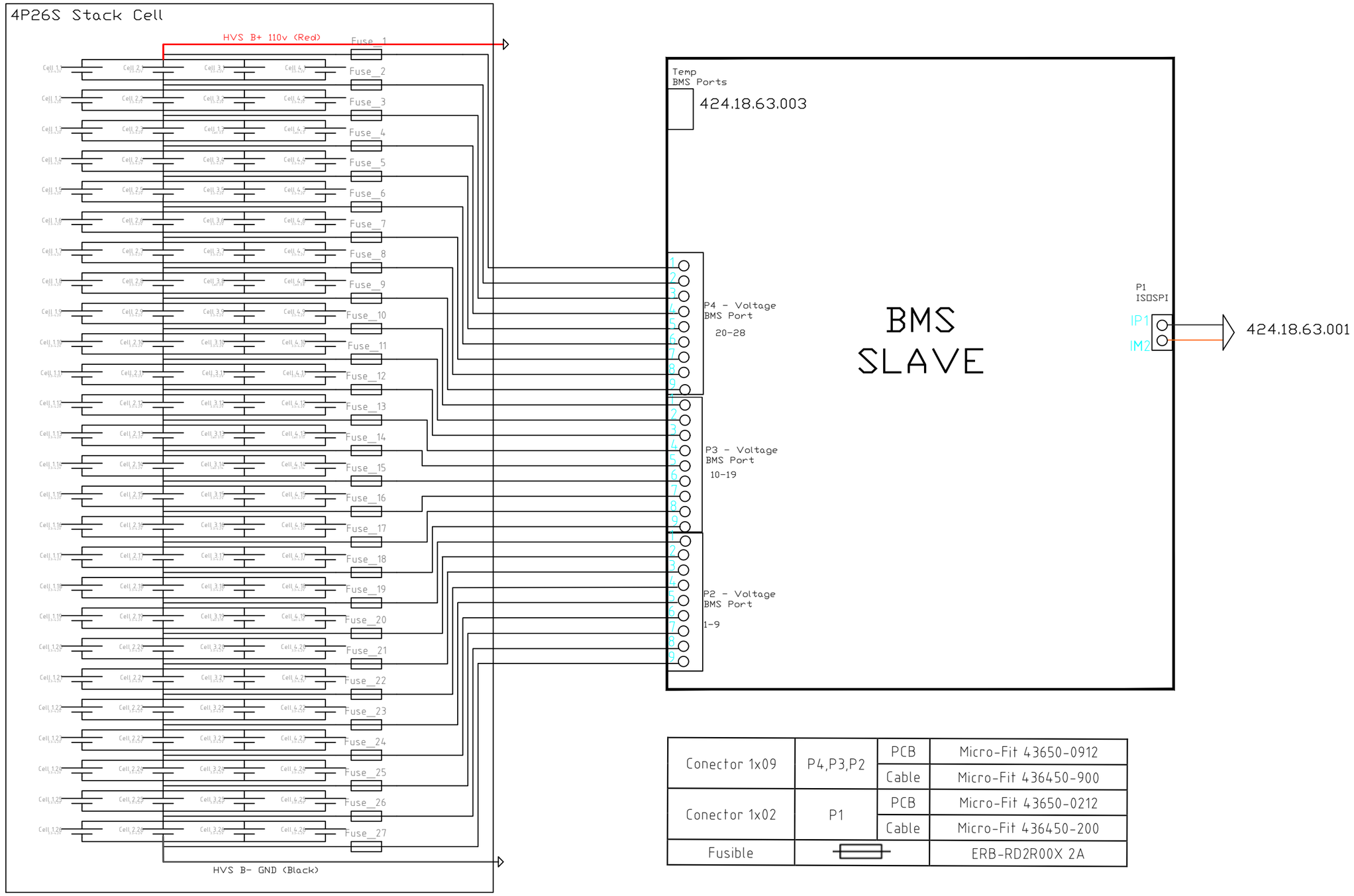
Diseño mecánico
Disipador de balanceo

Escuela Universitaria Politécnica - La Almunia
Centro adscrito
Universidad Zaragoza

Nº P.: 424.18.63
Nº O.: 424.18.63.000
Nom.Ar.: disipador.idw



Observaciones Generales	Observaciones de plano		Fecha	Nombre	
Proyecto: Diseño de un BMS para BVE	**Plano nº:** 1 **de:** 1	**Dibujado**	24/11/2018	Alfonso Mareca	Escuela Universitaria Politécnica - La Almunia
Palabras clave:	**Formato:** A3	**Comprobado**			Centro adscrito
Empresa: EUPLA	**Coment:** Calibre AWG18 para todo el cableado del Maestro serie AlphaWire Ecowire	**Idem.s.normas**			Universidad Zaragoza
Estado del proyecto: Finalizado		**ESCALA**	**Diseño de un BMS para BVE**		**Nº P.:** 424.18.63
Versión: V1			Esquemas eléctricos Conexionado del bms en la bateria		**Nº O.:** 424.18.63.001 **Nom.Ar.:** 424.18.63.001.dwg



Conector 1x09	P4,P3,P2	PCB	Micro-Fit 43650-0912
		Cable	Micro-Fit 436450-900
Conector 1x02	P1	PCB	Micro-Fit 43650-0212
		Cable	Micro-Fit 436450-200
Fusible			ERB-RD2R00X 2A

Observaciones Generales

Proyecto: Diseño de un BMS para BVE
Palabras clave:
Empresa: EUPLA
Estado del proyecto: Finalizado
Versión: V1

Observaciones de plano

Plano nº: 1 **de:** 1
Formato: A3
Coment: Calibre AWG18 para todo el cableado de lectura de tensión. Serie AlphaWire Ecowire

	Fecha	Nombre
Dibujado	24/11/2018	Alfonso Mareca
Comprobado		
Idem.s.normas		

ESCALA

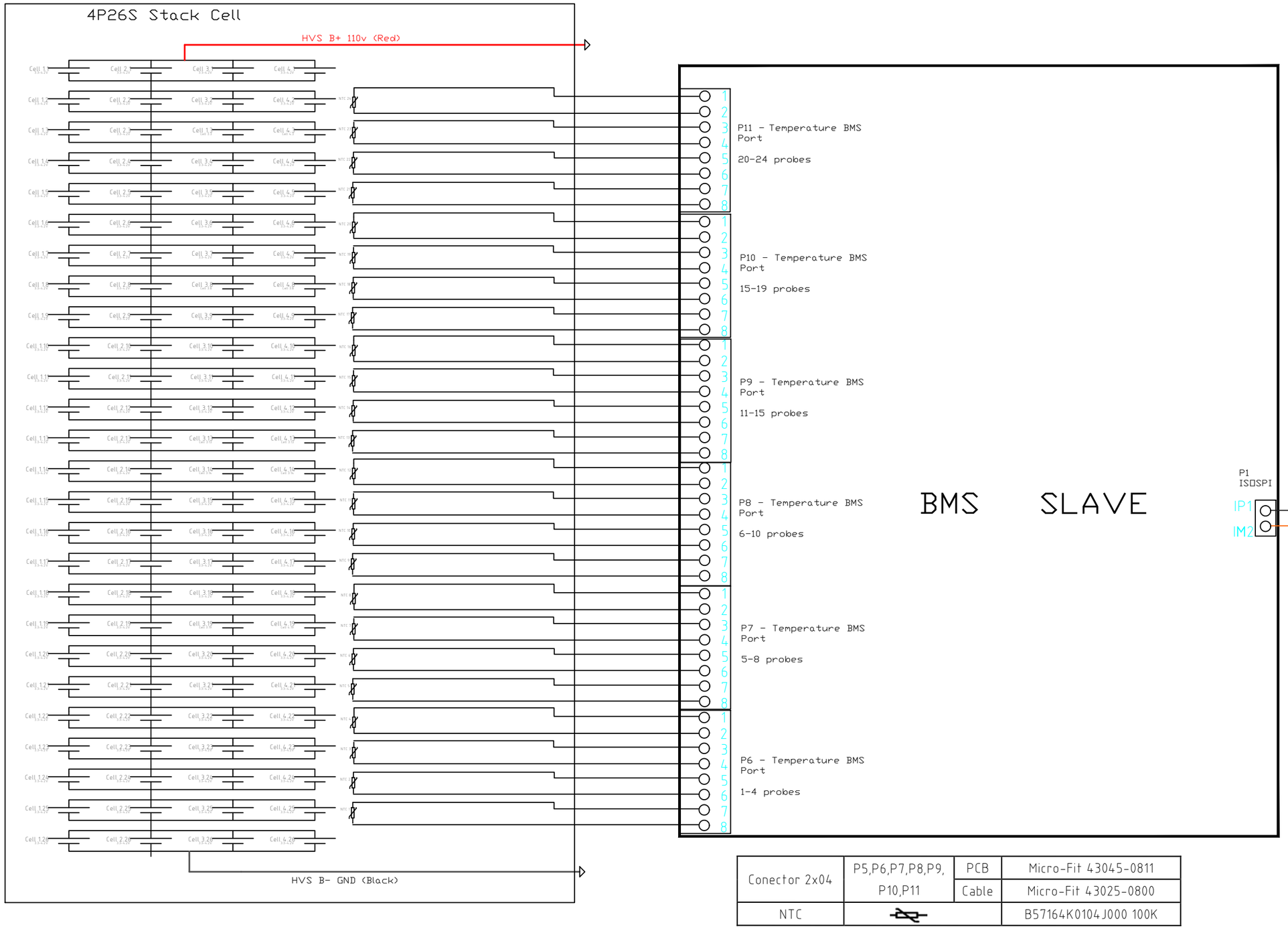
Escuela Universitaria Politécnica - La Almunia
Centro adscrito
Universidad Zaragoza

Diseño de un BMS para BVE

Esquemas eléctricos

Lectura en tension del esclavo

Nº P.: 424.18.63
Nº O.: 424.18.63.002
Nom.Ar.: 424.18.63.002.dwg



Conector 2x04	P5,P6,P7,P8,P9, P10,P11	PCB	Micro-Fit 43045-0811
		Cable	Micro-Fit 43025-0800
NTC			B57164K0104J000 100K

Observaciones Generales	Observaciones de plano		Fecha	Nombre
Proyecto: Diseño de un BMS para BVE	**Plano nº:** 1 **de:** 1	**Dibujado**	24/11/2018	Alfonso Mareca
Palabras clave:	**Formato:** A3	**Comprobado**		
Empresa: EUPLA	**Coment: Calibre AWG22 para todas las sondas de temperatura Serie AlphaWire Ecowire**	**Idem.s.normas**		
Estado del proyecto: Finalizado		**ESCALA**		
Versión: V1				

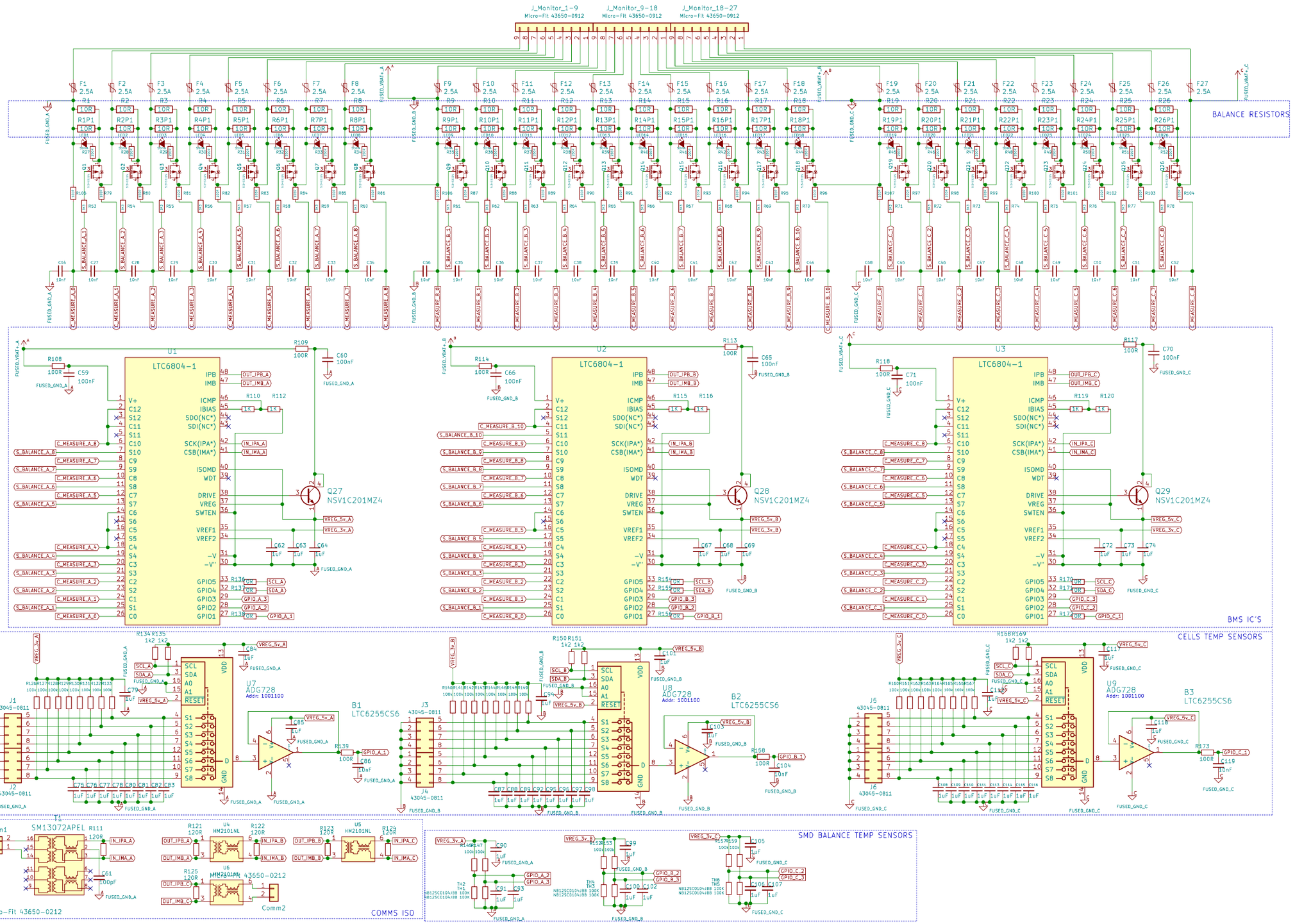
Escuela Universitaria Politécnica - La Almunia
Centro adscrito
Universidad Zaragoza

Diseño de un BMS para BVE

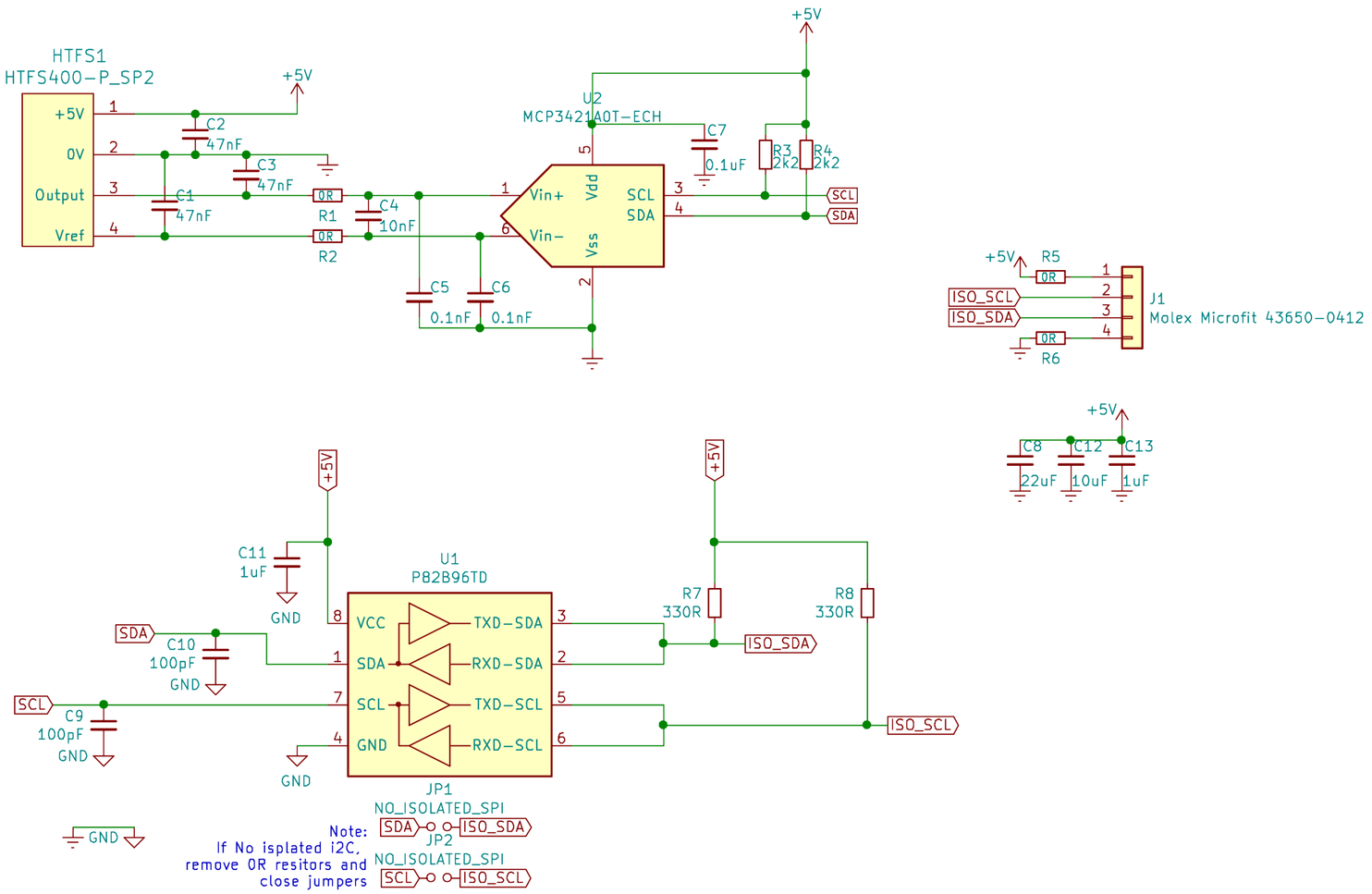
Esquemas eléctricos

Sondas de temperatura del esclavo

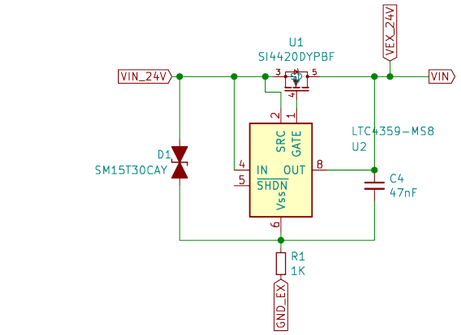
Nº P.: 424.18.63
Nº O.: 424.18.63.003
Nom.Ar.: 424.18.63.003.dwg

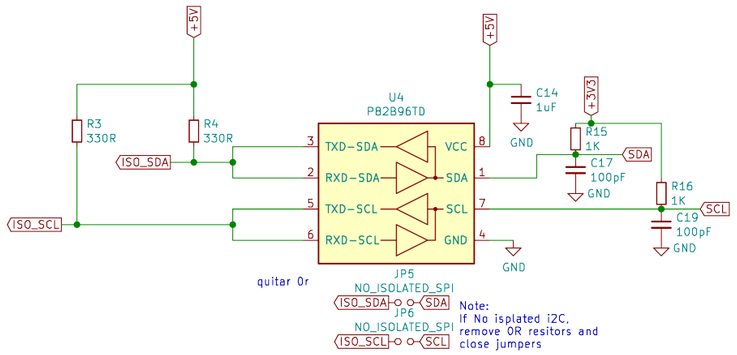


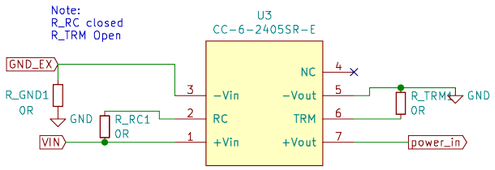
Observaciones Generales	Observaciones de plano		Fecha	Nombre	
Proyecto: Diseño de un BMS para BVE	**Plano nº:** 1 **de:** 1	Dibujado	24/11/2018	Alfonso Mareca	Escuela Universitaria Politécnica - La Almunia Centro adscrito Universidad Zaragoza
Palabras clave:	**Formato:** A3	Comprobado			
Empresa: EUPLA	**Coment:**	Idem.s.normas			
Estado del proyecto: Finalizado		**ESCALA**	**Diseño de un BMS para BVE** Esquemas eléctricos Esquema electrónico del esclavo		**Nº P.:** 424.18.63 **N O.:** 424.18.63.004 **Nom.Ar.:** slave.sch
Versi n: V5					

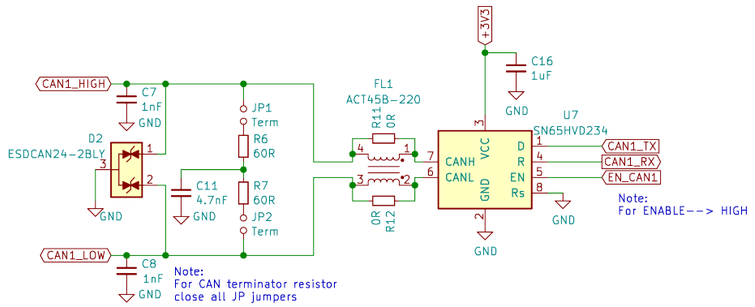


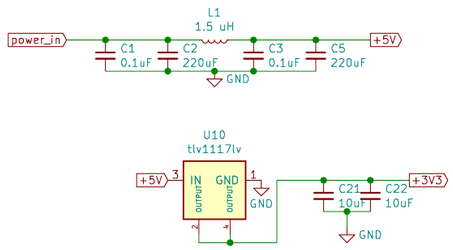
Observaciones Generales	Observaciones de plano		Fecha	Nombre		
Proyecto: Diseño de un BMS para BVE	**Plano nº:** 1 **de:** 1	**Dibujado**	24/11/2018	Alfonso Mareca		Escuela Universitaria Politécnica - La Almunia Centro adscrito Universidad Zaragoza
Palabras clave:	**Formato:** A3	**Comprobado**				
Empresa: EUPLA	**Coment:**	**Idem.s.normas**				
Estado del proyecto: Finalizado		**ESCALA**	**Diseño de un BMS para BVE** Esquemas eléctricos Esquema electrónico del sensor de corriente		**Nº P.:** 424.18.63 **N O.:** 424.18.63.005 **Nom.Ar.:** sensor.sch	
Versi n: V5						

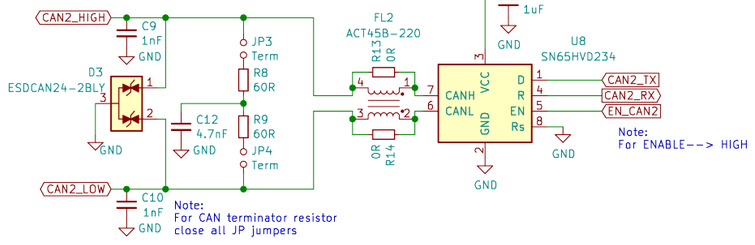


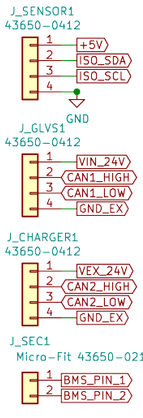


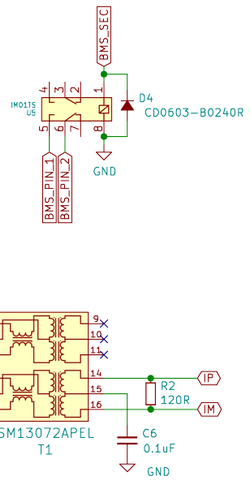


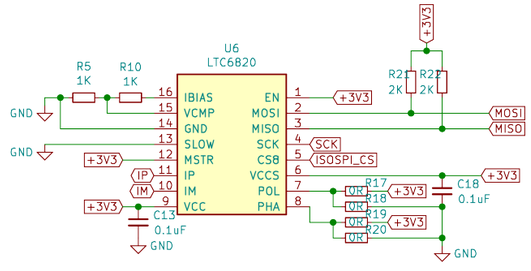






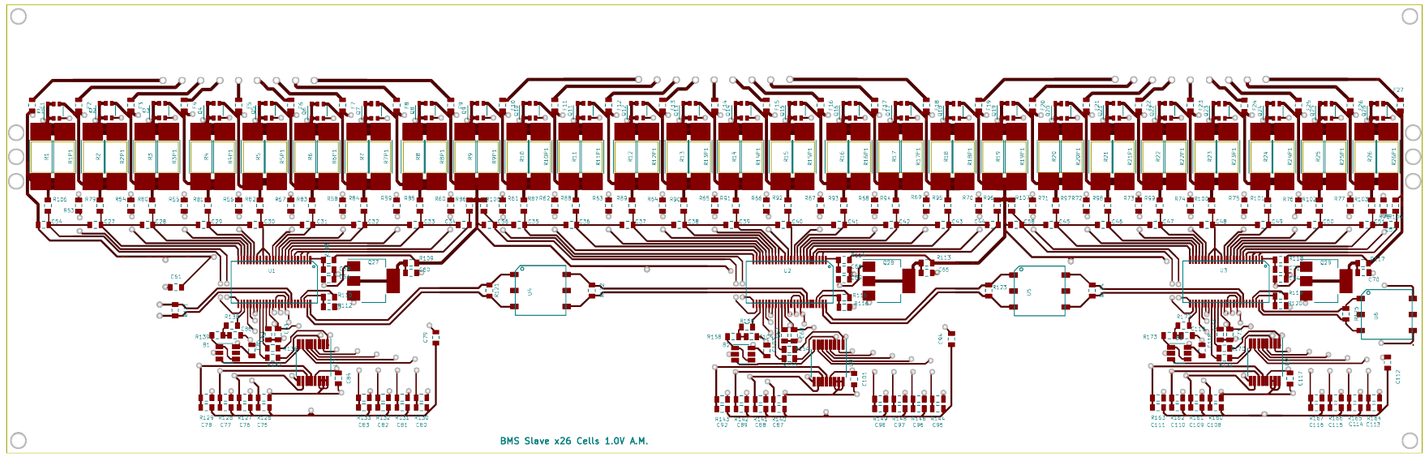




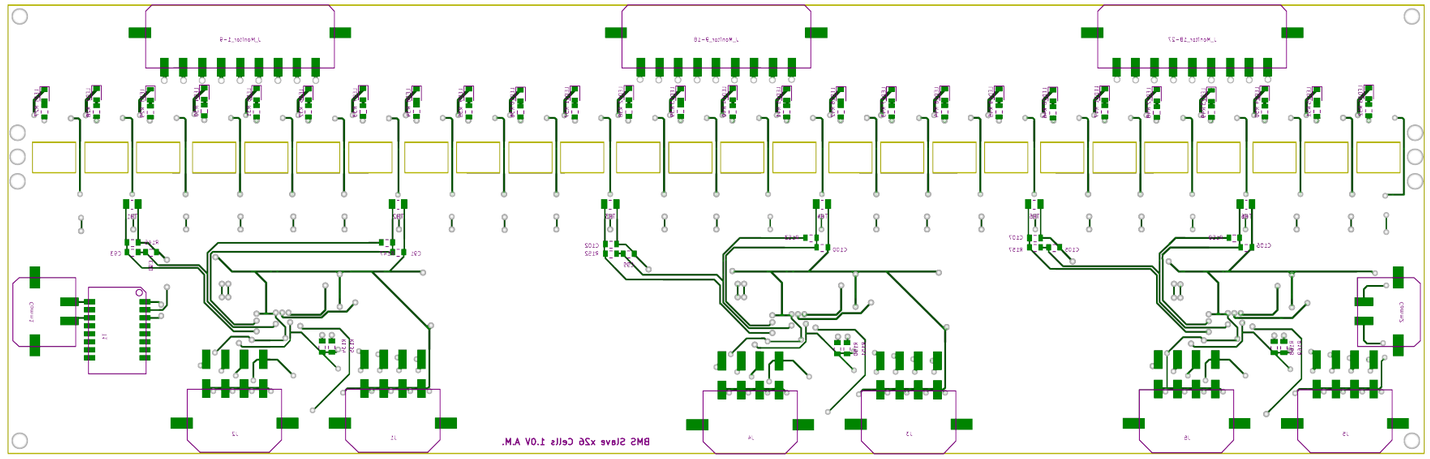


Observaciones Generales	Observaciones de plano		Fecha	Nombre		
Proyecto: Diseño de un BMS para BVE	**Plano nº:** 1 **de:** 1	Dibujado	24/11/2018	Alfonso Mareca	Escuela Universitaria Politécnica - La Almunia	
Palabras clave:	**Formato:** A3	Comprobado			Centro adscrito	
Empresa: EUPLA	**Coment:**	Idem.s.normas			Universidad Zaragoza	
Estado del proyecto: Finalizado		**ESCALA**	**Diseño de un BMS para BVE**		**Nº P.:** 424.18.63	
Versi n: V5			Esquemas eléctricos		**N O.:** 424.18.63.006	
			Esquema electrónico del maestro		**Nom.Ar.:** maestro.sch	

Vista superior

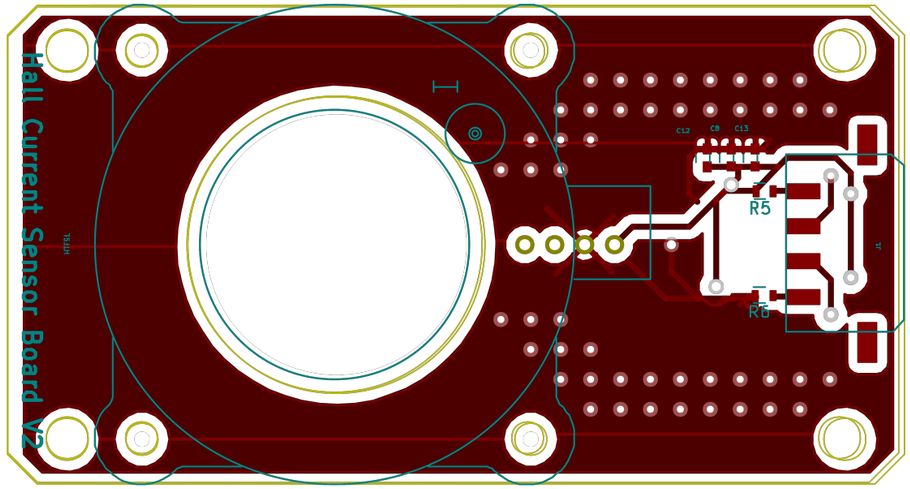


Vista inferior

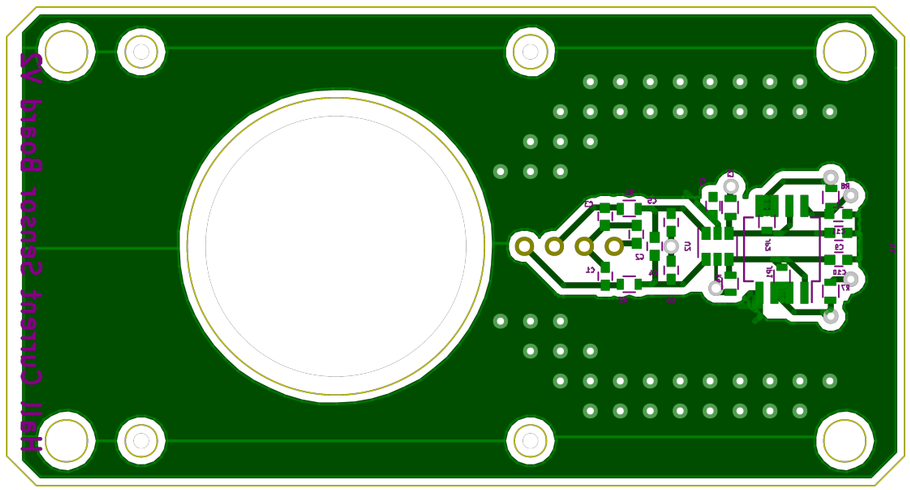


Observaciones Generales	Observaciones de plano		Fecha	Nombre		
Proyecto: Diseño de un BMS para BVE	**Plano nº:** 1 **de:** 1	**Dibujado**	24/11/2018	Alfonso Mareca		Escuela Universitaria Politécnica - La Almunia Centro adscrito Universidad Zaragoza
Palabras clave:	**Formato:** A3	**Comprobado**				
Empresa: EUPLA	**Coment:** Las lista de componentes se adjunta en los Anexos	**Idem.s.normas**				
Estado del proyecto: Finalizado		**ESCALA** 1:1	**Diseño de un BMS para BVE** Planos de PCB Plano de la PCB del esclavo		**Nº P.:** 424.18.63 **N O.:** 424.18.63.007 **Nom.Ar.:** slave.pcb	
Versi n: V5						

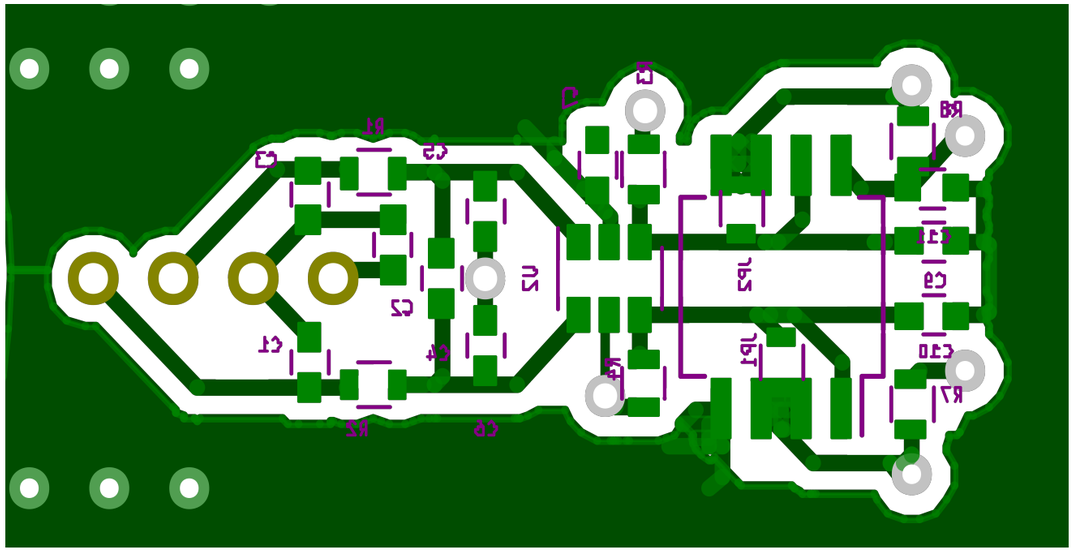
Vista superior



Vista inferior

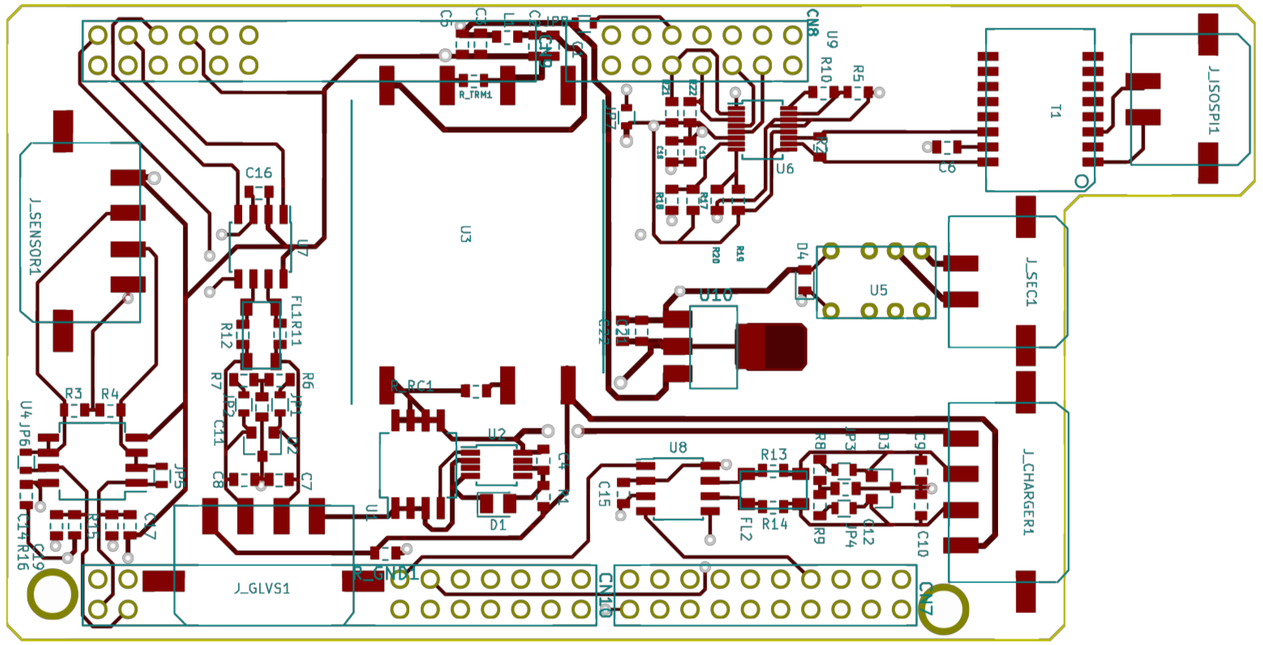


Detalle de la vista inferior 6:1

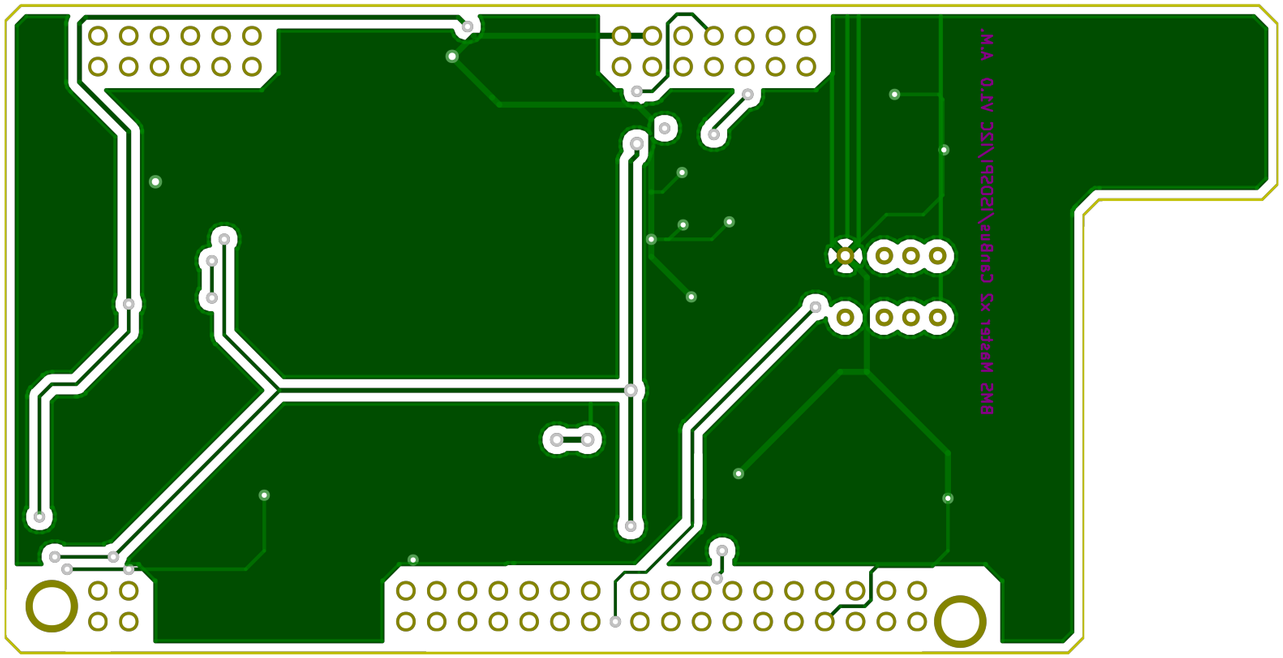


Observaciones Generales	Observaciones de plano		Fecha	Nombre		
Proyecto: Diseño de un BMS para BVE	**Plano nº:** 1 **de:** 1	Dibujado	24/11/2018	Alfonso Mareca	Escuela Universitaria Politécnica - La Almunia	
Palabras clave:	**Formato:** A3	Comprobado			Centro adscrito Universidad Zaragoza	
Empresa: EUPLA	**Coment:** Las lista de componentes se adjunta en los Anexos	Idem.s.normas				
Estado del proyecto: Finalizado		**ESCALA** 2:1	**Diseño de un BMS para BVE** Planos de PCB Plano de la PCB del **sensor de corriente**		**Nº P.:** 424.18.63 **N O.:** 424.18.63.008 **Nom.Ar.:** sensor.pcb	
Versi n: V5						

Vista superior



Vista inferior



Observaciones Generales	Observaciones de plano		Fecha	Nombre	Escuela Universitaria Politécnica - La Almunia
Proyecto: Diseño de un BMS para BVE	**Plano nº:** 1 **de:** 1	**Dibujado**	24/11/2018	Alfonso Mareca	Centro adscrito Universidad Zaragoza
Palabras clave:	**Formato:** A3	**Comprobado**			
Empresa: EUPLA	**Coment:** Las lista de componentes se adjunta en los Anexos	**Idem.s.normas**			
Estado del proyecto: Finalizado		**ESCALA** 2:1	**Diseño de un BMS para BVE** Planos de PCB Plano de la PCB del **maestro**		**Nº P.:** 424.18.63 **N O.:** 424.18.63.009 **Nom.Ar.:** maestro.pcb
Versi n: V5					

Escuela Universitaria
Politécnica - La Almunia
Centro adscrito
Universidad Zaragoza

ESCUELA UNIVERSITARIA POLITÉCNICA
DE LA ALMUNIA DE DOÑA GODINA (ZARAGOZA)

ANEXOS

Diseño e implementación de un BMS para BVE (Battery Electric Vehicle)

Design and implementation of a BMS for BVE (Battery Electric Vehicle)

424.18.63

Autor: Alfonso Mareca Miralles

Director: Dr. David Asiain Ansorena

Fecha: 27 de noviembre del 2018

INDICE DE CONTENIDO

ANEXO 1. (CÓDIGO FUENTE)

La programación del proyecto se ha realizado en la IDE Atollic TrueSTUDIO for STM32. Mediante el inicializador STM32CubeMX se genera una estructura de archivos base para el funcionamiento básico de las interfaces inicializadas.

A continuación se detalla el árbol del proyecto indicando que son librerías y archivos de desarrollo propio y cuales han sido incluidas por STM32CubeMX.

Core			
	Inc		Cabeceras del código por defecto, se incluyen las de:
		can.h	spi.h
		gpio.h	stm32f4xx_hal_conf.h
		i2c.h	stm32f4xx_it.h
		main.h	usart.h
	Src		Código por defecto:
		can.c	spi.c
		gpio.c	stm32f4xx_hal_msp.c
		i2c.c	stm32f4xx_it.c
		main.c	system_stm32f4xx.c
			usart.c
	BMS_Libraries		Librería principal del BMS:
		BMS_defs.h	BMS_defs.c
		LTC_Core.h	LTC_Core.c
	BMS_LTC_Libraries		Librería de funciones del LTC6804:
		LTC68041.h	LTC68041.c
		LTC_Temp.h	LTC_Temp.c
		LTC_balancing.h	LTC_balancing.c
	BMS_Sensor		Librería del sensor HALL:
		BMS_C_Sensor.h	BMS_C_Sensor.c
	Network_Modules		Librería de red:
		BMS_ETH_Comm.h	BMS_ETH_Comm.c
		BMS_ETH_Core.h	BMS_ETH_Core.c
		Broadcast_ssdp.h	Broadcast_ssdp.c
		DHCP_Client.h	DHCP_Client.c
	BMS_CanComms_Libraries		Librería de can:
		CAN_msg.h	CAN_msg.c
Debug			Carpetas generadas por defecto que incluyen:
startup			La librería HAL, LL y LWIP.
Drivers			El archivo de configuración de memoria.
html			El archivo de proyecto de STM32CubeMX.
LWIP			Archivos automatizdos de debug.
STM32F429ZI_FLASH.ld			Archivos generados por Doxygen.
BMS_SLAVE.Doxyfile			
BMS_SLAVE.ioc			

Funciones relacionadas con el LTC6804, lectura escritura, temperaturas y balanceo.

Funciones de comunicación con el MCP3421.

Todas las funciones añadidas a LWIP para la comunicación con la interfaz.

Funciones de mensajes utilizados en CANbus.

El punto de entrada (main), con el bucle principal.
Interfaces utilizadas (Can, I2C, spi, usart).
Vectorización de interrupciones.

El punto de entrada (main).
Interfaces utilizadas (Can, I2C, spi, usart).
Vectorización de interrupciones.

Incluye la programación de las estructuras principales, así como de las funciones de seguridad.

A continuación de expondrá el código fuente para los archivos con código de desarrollo propio. Los directorios scr y inc se agrupan como Punto de entrada e interfaces.

1.1. PUNTO DE ENTRADA E INTERFACES (INC Y SCR)

A continuación se expone el código fuente de los directorios inc y scr que incluyen:

Inc	
can.h	spi.h
gpio.h	stm32f4xx_hal_conf.h
i2c.h	stm32f4xx_it.h
main.h	usart.h
Scr	
can.c	spi.c
gpio.c	stm32f4xx_hal_msp.c
i2c.c	stm32f4xx_it.c
main.c	system_stm32f4xx.c
	usart.c

Gpio->Configuración de las gpio.
Spi-> Configuración de la interfaz SPI.
Can-> Configuración de la interfaz CAN.
USART-> Configuración de las UART.
Stm32f4xx_it-> Configuración de las interrupciones.

Se ha remarcado el código de desarrollo propio que se expondrá a continuación, el resto es generado automáticamente por STM32CubeMX con las opciones especificadas en el desarrollo.

1.1.1. Main.c

```
/**********************************************************************
 *         2018, Alfonso Mareca Miralles, Eupla Racing Team           *
 **********************************************************************/

/**
 * @file     Main.c
 * @author   Alfonso Mareca Miralles <alfonso@euplart.com>
 * @author           Basado en el codigo ST .
 * @date     30 Mar 2018
 * @brief    Main y bucle principal
 *
 */
#include "main.h"
#include "stm32f4xx_hal.h"
#include "can.h"
#include "crc.h"
#include "i2c.h"
#include "lwip.h"
#include "spi.h"
#include "usart.h"
#include "usb_otg.h"
#include "gpio.h"
#include "ethernetif.h"
#include <string.h>
#include "../BMS_CanComms_Libraries/CAN_msg.h"
#include "../Network_Modules/DHCP_Client.h"
#include "../Network_Modules/Broadcast_ssdp.h"
#include "../Network_Modules/BMS_ETH_Core.h"
#include "../Network_Modules/BMS_ETH_Comm.h"
#include "../BMS_LTC_Libraries/LTC68041.h"
#include "../BMS_LTC_Libraries/LTC_balancing.h"
#include "../BMS_LTC_Libraries/LTC_Temp.h"
#include "../BMS_Libraries/delay.h"
#include "../BMS_Libraries/BMS_defs.h"
#include "../BMS_Sensor/BMS_C_Sensor.h"
```

```
#include "../BMS_CanComms_Libraries/CAN_msg.h"
#include "../BMS_Libraries/LTC_Core.h"

#define MAX_STACK_SIZE 0x2000
extern int __io_putchar(int ch) __attribute__((weak));
extern int __io_getchar(void) __attribute__((weak));
uint32_t TxMailbox;
void SystemClock_Config(void);

/**
 * @brief  Main, inicializacion de sistemas.
 *
 */
int main(void) {

	HAL_Init();
	SystemClock_Config();
	HAL_GPIO_WritePin(CAN1_ENABLE_GPIO_Port, CAN1_ENABLE_Pin, GPIO_PIN_SET);
	HAL_GPIO_WritePin(CAN2_ENABLE_GPIO_Port, CAN2_ENABLE_Pin, GPIO_PIN_SET);
	MX_GPIO_Init();
	MX_USART3_UART_Init();
	MX_USB_OTG_FS_PCD_Init();
	MX_I2C2_Init();
	MX_CAN1_Init();
	MX_CAN2_Init();
	CAN2_Start_Config();   //Filtro y start CAN2
	CAN1_Start_Config();   //Filtro y start CAN1
	HAL_CAN_Start(&hcan1);
	MX_CRC_Init();
	MX_LWIP_Init();
	eth_mode = wire_stop;
	SysTick_Init();
	C_Sensor_Init(0);
	bms_status = LTC_slave_init(&ltc_reg, &ltc_conf, &bms_status, &bms_mode);
	Sec_Man_Relay_init();
	char *send_buffer = malloc(sizeof(char) * 250);
	bms_read = read;
	//Bucle principal
	while (1) {
		//Nucleo de lectura y seguridad
		BMS_Core();
		//comunicaciones tcp-ip
		MX_LWIP_Process();
		eth_mode = eth_link(&eth_mode);
		eth_mode = eth_init_deinit(&eth_mode);
		eth_mode = eth_GUI_Comms(send_buffer, &eth_mode, &ltc_reg, &ltc_conf,
		&current_s_conf, &bms_status, &bms_mode);
		HAL_GPIO_TogglePin(LD3_GPIO_Port, LD3_Pin);
		//comunicacion can
		BMS_Can_Comms();
	}
}
/**
 * @brief  Relojes, creado por defecto.
 * @retval None
 */
void SystemClock_Config(void) {

	RCC_OscInitTypeDef RCC_OscInitStruct;
	RCC_ClkInitTypeDef RCC_ClkInitStruct;
	__HAL_RCC_PWR_CLK_ENABLE();
	__HAL_PWR_VOLTAGESCALING_CONFIG(PWR_REGULATOR_VOLTAGE_SCALE1);
	RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSE;
	RCC_OscInitStruct.HSEState = RCC_HSE_BYPASS;
	RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
```

```
	RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSE;
	RCC_OscInitStruct.PLL.PLLM = 4;
	RCC_OscInitStruct.PLL.PLLN = 168;
	RCC_OscInitStruct.PLL.PLLP = RCC_PLLP_DIV2;
	RCC_OscInitStruct.PLL.PLLQ = 7;
	if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK) {
		_Error_Handler(__FILE__, __LINE__);
	}
	RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK | RCC_CLOCKTYPE_SYSCLK |
	RCC_CLOCKTYPE_PCLK1 | RCC_CLOCKTYPE_PCLK2;
	RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_PLLCLK;
	RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
	RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV4;
	RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV2;
	if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_5) != HAL_OK) {
		_Error_Handler(__FILE__, __LINE__);
	}
	HAL_SYSTICK_Config(HAL_RCC_GetHCLKFreq() / 1000);
	HAL_SYSTICK_CLKSourceConfig(SYSTICK_CLKSOURCE_HCLK);
	HAL_NVIC_SetPriority(SysTick_IRQn, 0, 0);
}

/**
 * @brief  Implementacion de printf por uart.
 * @retval None
 */
int __io_putchar(int ch) {
	HAL_UART_Transmit(&huart3, (uint8_t *) &ch, 1, 0xFFFF);
	return ch;
}
int _read(int file, char *ptr, int len) {
	int DataIdx;
	for (DataIdx = 0; DataIdx < len; DataIdx++) {
		*ptr++ = __io_getchar();
	}
	return len;
}

int _write(int file, char *ptr, int len) {
	int DataIdx;
	for (DataIdx = 0; DataIdx < len; DataIdx++) {
		__io_putchar(*ptr++);
	}
	return len;
}
/**
 * @brief  Errores por defecto.
 * @retval None
 */
void _Error_Handler(char *file, int line) {
	while (1) {
	}
}
```

1.1.2. Can.c

```
/**************************************************************************
 *         2018, Alfonso Mareca Miralles, Eupla Racing Team               *
 **************************************************************************/

/**
 * @file     Can.c
 * @author   Alfonso Mareca Miralles <alfonso@euplart.com>
 * @author           Basado en el codigo de ST.
 * @date     30 Mar 2018
 * @brief    Inicializacion de interfaces can
 *
 */
#include "can.h"
#include "gpio.h"

CAN_FilterTypeDef  Filter_hcan1;
CAN_FilterTypeDef  Filter_hcan2;
CAN_HandleTypeDef hcan1;
CAN_HandleTypeDef hcan2;

/**
 * @brief  Configuracion de interfaz CAN1, generado con STM32CUbeMX
 *
 */
void MX_CAN1_Init(void)
{
  hcan1.Instance = CAN1;
  hcan1.Init.Prescaler = 6;
  hcan1.Init.Mode = CAN_MODE_NORMAL;
  hcan1.Init.SyncJumpWidth = CAN_SJW_1TQ;
  hcan1.Init.TimeSeg1 = CAN_BS1_11TQ;
  hcan1.Init.TimeSeg2 = CAN_BS2_2TQ;
  hcan1.Init.TimeTriggeredMode = DISABLE;
  hcan1.Init.AutoBusOff = DISABLE;
  hcan1.Init.AutoWakeUp = DISABLE;
  hcan1.Init.AutoRetransmission = DISABLE;
  hcan1.Init.ReceiveFifoLocked = DISABLE;
  hcan1.Init.TransmitFifoPriority = DISABLE;
  if (HAL_CAN_Init(&hcan1) != HAL_OK)
  {
    _Error_Handler(__FILE__, __LINE__);
  }

}
/**
 * @brief  Configuracion de interfaz CAN2, generado con STM32CUbeMX
 *
 */
void MX_CAN2_Init(void)
{

  hcan2.Instance = CAN2;
  hcan2.Init.Prescaler = 6;
  hcan2.Init.Mode = CAN_MODE_NORMAL;
  hcan2.Init.SyncJumpWidth = CAN_SJW_1TQ;
  hcan2.Init.TimeSeg1 = CAN_BS1_11TQ;
  hcan2.Init.TimeSeg2 = CAN_BS2_2TQ;
  hcan2.Init.TimeTriggeredMode = DISABLE;
  hcan2.Init.AutoBusOff = DISABLE;
  hcan2.Init.AutoWakeUp = DISABLE;
  hcan2.Init.AutoRetransmission = DISABLE;
```

```
  hcan2.Init.ReceiveFifoLocked = DISABLE;
  hcan2.Init.TransmitFifoPriority = DISABLE;
  if (HAL_CAN_Init(&hcan2) != HAL_OK)
  {
    _Error_Handler(__FILE__, __LINE__);
  }
}

static uint32_t HAL_RCC_CAN1_CLK_ENABLED=0;
/**
 * @brief  Inicializacion de  CAN, generado con STM32CUbeMX
 *
 */
void HAL_CAN_MspInit(CAN_HandleTypeDef* canHandle)
{
  GPIO_InitTypeDef GPIO_InitStruct;
  if(canHandle->Instance==CAN1)
  {
    HAL_RCC_CAN1_CLK_ENABLED++;
    if(HAL_RCC_CAN1_CLK_ENABLED==1){
      __HAL_RCC_CAN1_CLK_ENABLE();
    }
    GPIO_InitStruct.Pin = GPIO_PIN_0|GPIO_PIN_1;
    GPIO_InitStruct.Mode = GPIO_MODE_AF_PP;
    GPIO_InitStruct.Pull = GPIO_NOPULL;
    GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_VERY_HIGH;
    GPIO_InitStruct.Alternate = GPIO_AF9_CAN1;
    HAL_GPIO_Init(GPIOD, &GPIO_InitStruct);
  }
  else if(canHandle->Instance==CAN2)
  {
    __HAL_RCC_CAN2_CLK_ENABLE();
    HAL_RCC_CAN1_CLK_ENABLED++;
    if(HAL_RCC_CAN1_CLK_ENABLED==1){
      __HAL_RCC_CAN1_CLK_ENABLE();
    }
    GPIO_InitStruct.Pin = GPIO_PIN_12|GPIO_PIN_6;
    GPIO_InitStruct.Mode = GPIO_MODE_AF_PP;
    GPIO_InitStruct.Pull = GPIO_NOPULL;
    GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_VERY_HIGH;
    GPIO_InitStruct.Alternate = GPIO_AF9_CAN2;
    HAL_GPIO_Init(GPIOB, &GPIO_InitStruct);
    HAL_NVIC_SetPriority(CAN2_RX0_IRQn, 0, 0);
    HAL_NVIC_EnableIRQ(CAN2_RX0_IRQn);
    HAL_NVIC_SetPriority(CAN2_RX1_IRQn, 0, 0);
    HAL_NVIC_EnableIRQ(CAN2_RX1_IRQn);
  }
}
/**
 * @brief  De-inicializacion de  CAN, generado con STM32CUbeMX
 *
 */
void HAL_CAN_MspDeInit(CAN_HandleTypeDef* canHandle)
{

  if(canHandle->Instance==CAN1)
  {
    HAL_RCC_CAN1_CLK_ENABLED--;
    if(HAL_RCC_CAN1_CLK_ENABLED==0){
      __HAL_RCC_CAN1_CLK_DISABLE();
    }
    HAL_GPIO_DeInit(GPIOD, GPIO_PIN_0|GPIO_PIN_1);
  }
  else if(canHandle->Instance==CAN2)
  {
```

```
    __HAL_RCC_CAN2_CLK_DISABLE();
    HAL_RCC_CAN1_CLK_ENABLED--;
    if(HAL_RCC_CAN1_CLK_ENABLED==0){
      __HAL_RCC_CAN1_CLK_DISABLE();
    }

    /**CAN2 GPIO Configuration
    PB12     ------> CAN2_RX
    PB6     ------> CAN2_TX
    */
    HAL_GPIO_DeInit(GPIOB, GPIO_PIN_12|GPIO_PIN_6);
    HAL_NVIC_DisableIRQ(CAN2_RX0_IRQn);
    HAL_NVIC_DisableIRQ(CAN2_RX1_IRQn);
  }
}

/**
 * @brief  Configuracion de CAN1
 *
 */
void CAN1_Start_Config(void){

        Filter_hcan1.FilterBank = 0;
        Filter_hcan1.FilterMode = CAN_FILTERMODE_IDMASK;
        Filter_hcan1.FilterScale = CAN_FILTERSCALE_32BIT;
        Filter_hcan1.FilterIdHigh = 0x0000;
        Filter_hcan1.FilterIdLow = 0x0000;
        Filter_hcan1.FilterMaskIdHigh = 0x0000;
        Filter_hcan1.FilterMaskIdLow = 0x0000;
        Filter_hcan1.FilterFIFOAssignment = CAN_FILTER_FIFO0;
        Filter_hcan1.FilterActivation = ENABLE;
        Filter_hcan1.SlaveStartFilterBank = 0;
        if(HAL_CAN_ConfigFilter(&hcan1, &Filter_hcan1) != HAL_OK)
        {
          Error_Handler();
        }
        if (HAL_CAN_Start(&hcan1) != HAL_OK)
        {
                Error_Handler();
        } else {
                printf("Inicializado CAN1(Y configurado filtro)\n\n");
        }
        if(HAL_CAN_ConfigFilter(&hcan1, &Filter_hcan1) != HAL_OK)
        {
          Error_Handler();
        }
        printf("CAN1 configurado filtro)\n\n");
}
/**
 * @brief  Configuración de CAN2
 *
 */
void CAN2_Start_Config(void){
        if (HAL_CAN_Start(&hcan2) != HAL_OK)
        {
          /* Start Error */
                printf("Error de inicializacion de CAN2\n\n");
          Error_Handler();
        } else {
                        printf("Inicializado CAN2(Y configurado filtro)\n\n");
        }
}
```

1.2. LIBRERÍA PRINCIPAL (BMS_LIBRARIES).

Se expone el código fuente de la librería BMS_Libraries:

BMS_Libraries		
	BMS_defs.h	BMS_defs.c
	LTC_Core.h	LTC_Core.c

Librería principal del BMS:
BMS_Defs-> Definiciones del sistema
LTC_Core-> Funciones principales de cálculo y seguridad

1.2.1. BMS_defs.h

```
/*****************************************************************************
*        2018, Alfonso Mareca Miralles, Eupla Racing Team                    *
*****************************************************************************/

/**
 * @file     BMS_defs.h
 * @author   Alfonso Mareca Miralles <alfonso@euplart.com>
 * @date     30 Sep 2018
 *
 * @brief    Datos de configuracion y registros del BMS slave
 */
#ifndef LTC_DEFS_H_
#define LTC_DEFS_H_

#include "main.h"
#include <stdio.h>
#include <stdlib.h>
/**
 * Numero de chips del BMS
 */
#define LTC_IC_NUM  3
#define FILTER_NUM  2
#define UDP_SERVER_PORT    7   /* define the UDP local connection port */
#define UDP_CLIENT_PORT    7   /* define the UDP remote connection port */
#define TEST_DATA_COMMS

/**
 * Modos de Error del BMS
 */
typedef enum {bms_ok,bms_error} bms_error_t;
typedef struct {
        bms_error_t ok;
        bms_error_t error_uv;
        bms_error_t error_ov;
        bms_error_t error_oc;
        bms_error_t error_otM;
        bms_error_t error_otBal;
        bms_error_t ev_stop;
} status_t;
/**
 * Modos de Operacion del BMS
 */
typedef enum {normal_op,charge_op,balance_op,log_op} op_modes_t;
op_modes_t bms_mode;

typedef enum {read,noread} read_t;
read_t bms_read;
```

```
typedef enum {sent,nosent} cancomms_status_t;
cancomms_status_t can_comms;

/**
 * Definicion del tipo de dato que contiene las configuraciones no estaticas del BMS
 */
typedef struct {
        uint8_t ic_num;         //!< Numero de chips en el BMS
        uint16_t cell_ov;       //!< Valor de Sobrevoltaje (V*10000)
        uint16_t cell_uv;       //!< (V*10000)
        uint16_t cell_ot;       //!<  (T*100)
        uint16_t bal_ot;        //!< Maxima temperatura de balanceo(T*100)
        uint16_t cell_margin;   //!< Margen de Voltaje de balanceo (V*10000)
        uint16_t t_fail_uv;     //!< Margen hasta fallo de infra voltaje  (ms)
        uint16_t t_fail_ov;     //!< Margen hasta fallo de infra voltaje  (ms)
} ltc_conf_t;
/**
 * Definicion del tipo de dato que contiene los datos temporales, y datos recogidos y/o enviados al esclavo
 */
typedef struct {
        uint16_t v_cell[LTC_IC_NUM][12];        //!< Vector de Valores de voltaje(V*10000)
        uint16_t v_modules[26];                 //!< Vector de Valores de voltaje del BMS(V*10000)
        uint16_t v_aux[LTC_IC_NUM][6];          //!< Vector de valores de las gpio del BMS(V*10000)
        uint8_t  tx_cfg[LTC_IC_NUM][6];         //!< Registro de datos de envio
        uint8_t  rx_cfg[LTC_IC_NUM][8];         //!< Regidtro de datos de recepcion
        uint16_t v_temp_bal[6];
        uint16_t temp_bal[6];                   //!< Vector de Valores de temperatura del disipador de
balanceo(T*100)
        uint16_t v_temp_cell[24];               //!< Vector de valores de temperatura de los modulos
(T*100)
        uint16_t temp_modules[24];              //!< Vector de valores de temperatura de los modulos
(T*100)
} ltc_reg_t;

typedef struct {
        uint16_t soc;           //!< soc value (value*100)
        uint16_t soh;           //!< soh value (value*100)
        uint16_t total_v;       //!< total tension value (value*100)
        int16_t max_peak_c;     //!< Max peak current value (value*100)
        uint16_t max_temp_module;       //!< Max temp module value(value*100)
        uint16_t max_temp_bal;  //!< Max temp bal value(value*100)
        uint16_t max_disbalance;//!< Max disbalance value(value*1000)
        uint16_t min_v_cell;
        uint16_t max_v_cell;
        uint8_t  cell_balance[26];
        uint16_t temp_average;
} bms_general_data_t;

/**
 * Definicion del tipo de dato que contiene las configuraciones no estaticas del BMS
 */
typedef struct {
        int16_t bat_oc;         //!< Maxima corriente pico(A*100)
        int16_t bat_uc;
        uint8_t gain;
} current_conf_t;
/**
 * Definicion del tipo de dato que contiene los datos temporales, y datos recogidos y/o enviados al sensor
 */
typedef struct {
        int16_t amps;           //!< Corriente pico del sensor *0.1
} current_reg_t;
/**
 * Definicion del tipo de dato que contiene los datos necesarios para calibrar las sondas de temperatura
```

```
 */
typedef struct {
        uint16_t v_high;        //!< Valor de tension de la toma 2º(superior)
        uint16_t v_low;         //!< Valor de tension de la toma 1º(inferior)
        uint16_t t_high;        //!< Valor de temperatura de ref de la toma 2º(superior)
        uint16_t t_low;         //!< Valor de temperatura de ref de la toma 1º(inferior)
} ltc_temp_call_t;

typedef struct {
        uint16_t v_high;        //!< Valor de tension de la toma 2º(superior)
        uint16_t v_low;         //!< Valor de tension de la toma 1º(inferior)
        uint16_t A_high;        //!< Valor de corriente de ref de la toma 2º(superior)
        uint16_t A_low;         //!< Valor de corriente de ref de la toma 1º(inferior)
} bms_current_call_t;


//Externalizacion de variables al entorno global:
extern op_modes_t bms_mode;
extern status_t  bms_status;

extern ltc_reg_t ltc_reg;
extern current_reg_t current_s_reg;

extern ltc_conf_t ltc_conf;
extern current_conf_t current_s_conf;
extern bms_general_data_t bms_general_data;

extern uint8_t ltc_active_cells[9][12];
extern uint8_t  ltc_active_t_bal[6];
extern ltc_temp_call_t ltc_mux_cal;
extern ltc_temp_call_t ltc_bal_cal;

extern bms_current_call_t bms_current_calibration;
extern read_t bms_read;
extern cancomms_status_t can_comms;
extern uint8_t  BALANCE_MODE;

// Pagina 32 tabla 16 control de comunicaciones
#define LTC_ICOM_START          0x60        //!< LTC REG -- SPI
#define LTC_ICOM_STOP           0x10        //!< LTC REG -- SPI
#define LTC_ICOM_BLANK          0x00        //!< LTC REG -- SPI
#define LTC_ICOM_NO_TRANSMIT    0x70        //!< LTC REG -- SPI
#define LTC_FCOM_MASTER_ACK     0x00        //!< LTC REG -- SPI
#define LTC_FCOM_MASTER_NACK    0x08        //!< LTC REG -- SPI
#define LTC_FCOM_MASTER_NACK_STOP  0x09     //!< LTC REG -- SPI
#define LTC_CMD_WRCFG   0x001               //!< LTC REG -- Escritura en registro de configuracion
#define LTC_CMD_WRCOMM  0x721               //!< LTC REG -- Escritura en registro de SPI comm
#define LTC_CMD_RDCOMM  0x722               //!< LTC REG -- Lectura en registro de SPI comm
#define LTC_CMD_STCOMM  0x723               //!< LTC REG -- Envio de SPI comm
#define LTC_CMD_WRCFG   0x001
#define LTC_CMD_RDCFG   0x002
#define LTC_CMD_RDCVA   0x004
#define LTC_CMD_RDCVB   0x006
#define LTC_CMD_RDCVC   0x008
#define LTC_CMD_RDCVD   0x00A
#define LTC_CMD_RDAUXA  0x00C
#define LTC_CMD_RDAUXB  0x00E
#define LTC_CMD_RDSTATA 0x010
#define LTC_CMD_RDSTATB 0x012
#define LTC_CMD_ADCV    0x260
#define LTC_CMD_ADOW    0x228
#define LTC_CMD_CVST    0x207
#define LTC_CMD_ADAX    0x460
#define LTC_CMD_AXST    0x407
#define LTC_CMD_ADSTAT  0x468
```

```
#define LTC_CMD_STATST   0x40F
#define LTC_CMD_ADCVAX   0x46F
#define LTC_CMD_CLRCELL  0x711
#define LTC_CMD_CLRAUX   0x712
#define LTC_CMD_CLRSTAT  0x713
#define LTC_CMD_PLADC    0x714
#define LTC_CMD_DIAGN    0x715
#define LTC_CMD_WRCOMM   0x721
#define LTC_CMD_RDCOMM   0x722
#define LTC_CMD_STCOMM   0x723

void Serial_Print_Cells(ltc_reg_t *ltc_reg,ltc_conf_t *ltc_conf,uint8_t  ltc_active_cells[9][12]);
void Serial_Print_Status(status_t *bms_status,ltc_conf_t *ltc_conf,status_t *status);
void Serial_Print_Gpio(ltc_reg_t *ltc_reg,ltc_conf_t *ltc_conf);

#endif /* LTC_DEFS_H_ */
```

1.2.2. BMS_defs.c

```
/*****************************************************************************
 *        2018, Alfonso Mareca Miralles, Eupla Racing Team              *
 *****************************************************************************/

/**
 * @file     BMS_defs.c
 * @author   Alfonso Mareca Miralles <alfonso@euplart.com>
 * @date     30 Sep 2018
 *
 * @brief    Datos de configuracion y registros del BMS slave Inicializacion de algunas variables
 */
#include "../BMS_Libraries/BMS_defs.h"
#include "../BMS_Libraries/delay.h"

#include "stm32f4xx_hal.h"
#include "main.h"
#include <stdio.h>
#include <stdlib.h>

//            v_high v_low  v_range t_high t_low t_range   --> voltage 0.0001  Degree --> 0.01
status_t bms_status;

ltc_temp_call_t ltc_mux_cal = { 12750,14246,3230,1950};
ltc_temp_call_t ltc_bal_cal = { 11943,15540,4350,2580};  //{ 13325,15623,3650,2300};
bms_current_call_t bms_current_calibration = { 13325,15623,3650,2300};
ltc_reg_t ltc_reg = {0};
ltc_conf_t ltc_conf = {LTC_IC_NUM,  // Numero de chip 0..9
           42100,   // Cell over voltage value  *0.001
           30000,  // Cell under voltage value *0.001
           9000,     // Cell over temperature value       *0.01
           9000,     // Balancing over temperature value  *0.01
           30,                  // Margen de Balanceo       *0.001
           1000,   //!< Margen hasta fallo de infra voltaje  (ms)
           1000,    //!< Margen hasta fallo de infra voltaje  (ms)
                  };

bms_general_data_t bms_general_data = {
                     10000,      //!< soc value        *0.01
                     10000,      //!< soh value       *0.01
                     0,                  //!< total tension value  *0.01
                     0,                  //!< Max peak current value          *0.01
                     0,                  //!< Max temp module value(value*10)   *0.01
                     0,                  //!< Max temp bal value(value*10)      *0.01
                     0,                  //!< Max disbalance value(value*1000)  *0.001
```

```
                    0,
                    0
};
current_reg_t current_s_reg;
current_conf_t current_s_conf = {5000,-5000,1};
uint8_t  ltc_active_cells[9][12] = {
                                                    {1,1,1,1,0,0,1,1,1,1,0,0},
                                                        {1,1,1,1,1,0,1,1,1,1,1,0},
                                                    {1,1,1,1,0,0,1,1,1,1,0,0},

                                                    {1,1,1,1,0,0,1,1,1,1,0,0},
                                                    {1,1,1,1,1,0,1,1,1,1,1,0},
                                                    {1,1,1,1,0,0,1,1,1,1,0,0},

                                                    {1,1,1,1,0,0,1,1,1,1,0,0},
                                                    {1,1,1,1,1,0,1,1,1,1,1,0},
                                                    {1,1,1,1,0,0,1,1,1,1,0,0}};

uint8_t  ltc_active_t_bal[6] = {1,0,1,0,1,0};
uint8_t BALANCE_MODE = 0;
#endif
```

1.2.3. BMS_Core.h

```
/*****************************************************************************
 *        2018, Alfonso Mareca Miralles, Eupla Racing Team                   *
 *****************************************************************************/

/**
 * @file     BMS_Core.h
 * @author   Alfonso Mareca Miralles <alfonso@euplart.com>
 * @date     30 Sep 2018
 *
 * @brief    Header de la libreria que Contiene las funciones principales y de seguridad
 */
#ifndef LTC_CORE_H_
#define LTC_CORE_H_
#include "../BMS_Libraries/BMS_defs.h"
#include "../BMS_LTC_Libraries/LTC68041.h"
status_t LTC_slave_init(ltc_reg_t *ltc_reg,ltc_conf_t *ltc_conf,status_t *status,op_modes_t *bms_mode);
void       LTC_Module_V(ltc_reg_t *ltc_reg,ltc_conf_t *ltc_conf, uint8_t ltc_active_cells[9][12]);
void LTC_Module_V_one(ltc_reg_t *ltc_reg,ltc_conf_t *ltc_conf,status_t *status, uint8_t
ltc_active_cells[9][12]);
void Sec_Votage_Modules(ltc_reg_t *ltc_reg,ltc_conf_t *ltc_conf,status_t *status);
void Sec_Temp_Modules(ltc_reg_t *ltc_reg,ltc_conf_t *ltc_conf,status_t *status);
void Sec_Temp_Bal(ltc_reg_t *ltc_reg,ltc_conf_t *ltc_conf,status_t *status);
status_t Sec_C_Sensor(current_reg_t *current_s_reg,current_conf_t *current_s_conf,status_t *status);
uint16_t kick_mean(uint16_t temp_mod[],uint8_t filt_num);

void BMS_Core(void);
void Sec_Man_Relay(status_t *bms_status,ltc_conf_t *ltc_conf,status_t *status);
void Sec_Man_Relay_init(void);

void BMS_General_Data_Calc(ltc_reg_t *ltc_reg,current_reg_t *current_s_reg,ltc_conf_t
*ltc_conf,bms_general_data_t *bms_general_data);

#endif /* BMS_CORE_H_ */
```

1.2.4. BMS_Core.c

```
/*****************************************************************************
 *        2018, Alfonso Mareca Miralles, Eupla Racing Team                   *
 *****************************************************************************/

/**
 * @file     BMS_Core.c
 * @author   Alfonso Mareca Miralles <alfonso@euplart.com>
 * @date     30 Sep 2018
 *
 * @brief    Contiene las funciones principales y de seguridad
 */
#include "LTC_Core.h"
#include "main.h"
#include "spi.h"
#include "../BMS_LTC_Libraries/LTC68041.h"
#include "../BMS_Libraries/BMS_defs.h"
#include "../BMS_Libraries/delay.h"
#include "can.h"
#include "../BMS_LTC_Libraries/LTC68041.h"
#include "../BMS_LTC_Libraries/LTC_balancing.h"
#include "../BMS_LTC_Libraries/LTC_Temp.h"
#include "../BMS_Sensor/BMS_C_Sensor.h"
CAN_TxHeaderTypeDef TxHeader_can1;
CAN_TxHeaderTypeDef TxHeader_can2;
CAN_RxHeaderTypeDef RxHeader_can2;
uint8_t TxData_can1[0];
uint8_t TxData_can2[0];
uint8_t RxData_can2[8];
uint32_t TxMailbox;

/**
 * @brief  Ejecucion principal de lectura y seguridad
 *
 */
void BMS_Core(void){
   LTC_Module_V_one(&ltc_reg, &ltc_conf, &bms_status, ltc_active_cells);
         bms_read = LTC_All_Mux_T(&ltc_reg, &ltc_conf, &bms_status, &ltc_mux_cal);
   LTC_All_Bal_T(&ltc_reg, &ltc_conf, &ltc_bal_cal);
   C_Sensor_Read(0, &current_s_reg);
   if (BALANCE_MODE == 1) {
          LTC_Balancing_Update(&ltc_reg, &ltc_conf, &bms_mode, ltc_active_cells, &bms_status);
   }
   BMS_General_Data_Calc(&ltc_reg, &current_s_reg, &ltc_conf, &bms_general_data);
         Sec_Man_Relay(&bms_status, &ltc_conf, &bms_status);
}

/**
 * @brief  Calculo de datos para la monitorizacion
 *
 */
void BMS_General_Data_Calc(ltc_reg_t *ltc_reg, current_reg_t *current_s_reg, ltc_conf_t *ltc_conf,
bms_general_data_t *bms_general_data) {
         bms_general_data->soh = 1000;
         bms_general_data->total_v = 0;
         for (int i = 0; i < 26; i++) {
                  bms_general_data->total_v = ((bms_general_data->total_v) + (ltc_reg->v_modules[i] /
100));
         }

         if (current_s_reg->amps > bms_general_data->max_peak_c) {
                  bms_general_data->max_peak_c = current_s_reg->amps;
```

```
		}

		for (int i = 0; i < 24; i++) {
			if (ltc_reg->temp_modules[i] > bms_general_data->max_temp_module) {
				bms_general_data->max_temp_module = ltc_reg->temp_modules[i];
			}
		}

		for (int i = 0; i < 6; i++) {
			if (ltc_reg->temp_bal[i] > bms_general_data->max_temp_bal) {
				bms_general_data->max_temp_bal = ltc_reg->temp_bal[i];
			}
		}

		uint16_t lowest_v = 42000;
		for(uint8_t cell=0;cell<26;cell++){
			if(ltc_reg->v_modules[cell]<lowest_v){
				lowest_v = ltc_reg->v_modules[cell];
			}

		}
		bms_general_data->min_v_cell=lowest_v;
		uint16_t highest_v = 0;
		for(uint8_t cell=0;cell<26;cell++){
				if (ltc_reg->v_modules[cell]> highest_v) {
					highest_v = ltc_reg->v_modules[cell];

			}
		}
		bms_general_data->max_v_cell=highest_v;
		uint16_t hi_lo = highest_v - lowest_v;
			bms_general_data->max_disbalance = hi_lo;
			//printf("Max disbalance: %u", hi_lo);
		uint16_t temp_temp=0;
		uint8_t counter =0;
		for(int i=0;i<24;i++){
			temp_temp=temp_temp+(ltc_reg->temp_modules[i]/100);
			if(temp_temp>10){
				counter++;
			}
		}
		bms_general_data->temp_average=(temp_temp/counter);

}
/**
 * @brief  Inicializacion del esclavo
 *
 */
status_t LTC_slave_init(ltc_reg_t *ltc_reg, ltc_conf_t *ltc_conf, status_t *status, op_modes_t *bms_mode) {
		MX_SPI3_Init();
		LTC_Set_ADC(MD_NORMAL, DCP_DISABLED, CELL_CH_ALL, AUX_CH_ALL);
		LTC_Send_ADAX();
		LTC_Send_ADCV();
		for (int i = 0; i < ltc_conf->ic_num; i++) {
			ltc_reg->tx_cfg[i][0] = 0xFE;
			ltc_reg->tx_cfg[i][1] = 0x00;
			ltc_reg->tx_cfg[i][2] = 0x00;
			ltc_reg->tx_cfg[i][3] = 0x00;
			ltc_reg->tx_cfg[i][4] = 0x00;
			ltc_reg->tx_cfg[i][5] = 0x00;
		}
		*bms_mode = normal_op;
		status->error_oc = bms_ok;
		status->error_otBal = bms_ok;
		status->error_otM = bms_ok;
```

```
		status->error_ov = bms_ok;
		status->error_uv = bms_ok;
		status->ev_stop = bms_ok;
		status->ok = bms_ok;
		for (int i = 0; i < 24; i++) {
			ltc_reg->temp_modules[i] = 0;
		}
		for (int i = 0; i < 6; i++) {
			ltc_reg->temp_bal[i] = 0;
		}
		return *status;
}
/**
 * @brief  Calculo de tensiones y ejecucion de seguridad
 *
 */
void LTC_Module_V_one(ltc_reg_t *ltc_reg, ltc_conf_t *ltc_conf, status_t *status, uint8_t
ltc_active_cells[9][12]) {
		uint8_t counter = 0;
		static uint32_t t_preb = 0;
		if (t_preb == 0) {
			t_preb = HAL_GetTick();
		}
		if (HAL_GetTick() - t_preb > 500) {  //estaba a 500
			t_preb = 0;
			//LTC_WakeUp_Sleep();
			LTC_WR_CMD(ltc_conf->ic_num, ltc_reg->tx_cfg, LTC_CMD_WRCFG);
			LTC_Send_ADCV();
			LTC_Send_ADCV();
			HAL_Delay(1);
			LTC_Send_ADCV();
			LTC_RDcv(0, ltc_conf->ic_num, ltc_reg->v_cell);
			for (int ltc = 0; ltc < ltc_conf->ic_num; ltc++) {
				for (int cell = 0; cell < 12; cell++) {
					if (ltc_active_cells[ltc][cell] != 0) {
						ltc_reg->v_modules[counter] = ltc_reg->v_cell[ltc][cell];
						/*if(ltc_reg->v_modules[counter]<=ltc_conf->cell_uv){
						 status->error_uv=bms_error; }
						 else status->error_uv=bms_ok;*/
						printf("C voltaje %u\n",ltc_reg->v_modules[counter]);
						counter++;
					}
				}
			}
			Sec_Votage_Modules(ltc_reg, ltc_conf, status);
		}
}
/**
 * @brief  Seguridad de tension
 *
 */
void Sec_Votage_Modules(ltc_reg_t *ltc_reg, ltc_conf_t *ltc_conf, status_t *status) {
		uint8_t counter = 0;
		for (int ltc = 0; ltc < ltc_conf->ic_num; ltc++) {
			for (int cell = 0; cell < 12; cell++) {
				if (ltc_active_cells[ltc][cell] != 0) {
					if ((ltc_reg->v_modules[counter] <= ltc_conf->cell_uv)) {
						status->error_uv = bms_error;
					} else
						status->error_uv = bms_ok;
					if ((ltc_reg->v_modules[counter] >= ltc_conf->cell_ov)) {
						status->error_ov = bms_error;
					} else
						status->error_ov = bms_ok;
					counter++;
```

```
                        }
                }
        }
}
/**
 * @brief  Seguridad de temperatura
 *
 */
void Sec_Temp_Modules(ltc_reg_t *ltc_reg, ltc_conf_t *ltc_conf, status_t *status) {
        for (uint8_t i = 0; i <= 24; i++) {
                if (ltc_reg->temp_modules[i] >= ltc_conf->cell_ot) {
                        status->error_otM = bms_error;

                }
        }
}
/**
 * @brief  Seguridad de temperatura de balanceo
 *
 */
void Sec_Temp_Bal(ltc_reg_t *ltc_reg, ltc_conf_t *ltc_conf, status_t *status) {
        uint8_t counter = 0;
        for (uint8_t i = 0; i <= 2; i++) {
                if ((ltc_reg->temp_bal[counter] >= ltc_conf->bal_ot) && ltc_active_t_bal[counter]) {
                        status->error_otBal = bms_error;
                }
        }

}
/**
 * @brief  Seguridad de corriente
 *
 */
status_t Sec_C_Sensor(current_reg_t *current_s_reg, current_conf_t *current_s_conf, status_t *status) {
        if (((uint16_t) current_s_reg->amps) >= current_s_conf->bat_oc) {
                status->error_oc = bms_error;
                return *status;

        }
        if ((current_s_reg->amps) <= current_s_conf->bat_uc) {
                status->error_oc = bms_error;
                return *status;

        }
        status->error_oc = bms_ok;
        return *status;
}
/**
 * @brief  Estudio de tiempos y seguridad general
 *
 */
void Sec_Man_Relay(status_t *bms_status, ltc_conf_t *ltc_conf, status_t *status) {
        static uint32_t t_temp_ov = 0;
        static uint32_t t_temp_uv = 0;
        if ((status->error_uv == bms_ok) && (status->error_ov == bms_ok)) {
                //&&(status->error_oc==bms_ok)&&(status->error_otM==bms_ok)&&(status-
>error_otBal==bms_ok)){
                status->ok = bms_ok;
        } else {
                status->ok = bms_error;
        }

        if (status->ok == bms_ok) {
                t_temp_uv = 0;
                t_temp_uv = 0;
```

```
		}
		if ((status->ok != bms_ok) && (t_temp_uv == 0)) {
			t_temp_uv = HAL_GetTick();
		}
		if ((status->ok != bms_ok) && (t_temp_ov == 0)) {
			t_temp_ov = HAL_GetTick();
		}
		if ((t_temp_ov != 0) || (t_temp_uv != 0)) {
			if (HAL_GetTick() - t_temp_ov >= ltc_conf->t_fail_ov) {
				t_temp_ov = 0;
				HAL_GPIO_WritePin(SEC_MAN_ENABLE_GPIO_Port,
SEC_MAN_ENABLE_Pin, GPIO_PIN_RESET);
				status->ev_stop = bms_error;
				return;
			}
			if (HAL_GetTick() - t_temp_uv >= ltc_conf->t_fail_uv) {
				t_temp_uv = 0;
				HAL_GPIO_WritePin(SEC_MAN_ENABLE_GPIO_Port,
SEC_MAN_ENABLE_Pin, GPIO_PIN_RESET);
				status->ev_stop = bms_error;
				return;
			}

		}
		if (status->error_oc != bms_ok) {
			HAL_GPIO_WritePin(SEC_MAN_ENABLE_GPIO_Port, SEC_MAN_ENABLE_Pin,
GPIO_PIN_RESET);
			status->ev_stop = bms_error;
			return;
		}
		if (status->error_otM != bms_ok) {
			HAL_GPIO_WritePin(SEC_MAN_ENABLE_GPIO_Port, SEC_MAN_ENABLE_Pin,
GPIO_PIN_RESET);
			status->ev_stop = bms_error;
			return;
		}
		if (status->error_otBal != bms_ok) {
			HAL_GPIO_WritePin(SEC_MAN_ENABLE_GPIO_Port, SEC_MAN_ENABLE_Pin,
GPIO_PIN_RESET);
			status->ev_stop = bms_error;
			return;
		}

		return;
}

void Sec_Man_Relay_init(void) {
		HAL_GPIO_WritePin(SEC_MAN_ENABLE_GPIO_Port, SEC_MAN_ENABLE_Pin,
GPIO_PIN_SET);
}
```

1.3. LIBRERÍA DEL LTC6804 (BMS_LTC_LIBRARIES)

Se expone el código fuente de la librería BMS_LTC_Libraries:

BMS_LTC_Libraries		Librería de funciones del LTC6804:
LTC68041.h	LTC68041.c	LTC6801-> Funciones especificas de comunicación
LTC_Temp.h	LTC_Temp.c	LTC_Temp-> Funciones de temperatura
LTC_balancing.h	LTC_balancing.c	LTC_Balancing-> Funciones de balanceo

Se ha remarcado el código de desarrollo propio que se expondrá a continuación, el resto es generado automáticamente por STM32CubeMX con las opciones especificadas en el desarrollo.

1.3.1. LTC6801.h

```
/*****************************************************************************
 *        2018, Alfonso Mareca Miralles, Eupla Racing Team                   *
 *****************************************************************************/

/**
 * @file     LTC68041.h
 * @author   Alfonso Mareca Miralles <alfonso@euplart.com>
 * @author           Header, Basado en el codigo de Analog Devices para el LTC6804 <evt-contact@mit.edu> .
 * @date     30 Sep 2018
 *
 */

#ifndef LTC68041_H
#define LTC68041_H


#ifndef LTC6804_CS
#define LTC6804_CS spi_cs
#endif

#include "main.h"
#include <stdio.h>
#include <stdlib.h>
/**
 * @brief  Generacion de tabla crc
 *
 */
void generate_crc15_table()
{
  int remainder;
  for(int i = 0; i<256;i++)
  {
    remainder =  i<< 7;
    for (int bit = 8; bit > 0; --bit)
       {

         if ((remainder & 0x4000) > 0)//equivalent to remainder & 2^14 simply check for MSB
          {
             remainder = ((remainder << 1)) ;
             remainder = (remainder ^ 0x4599);
         }
         else
          {
            remainder = ((remainder << 1));
```

```
        }
    }

  crc15Table[i] = remainder&0xFFFF;

 }
}


static const unsigned int crc15Table[256];
/**
 * @brief  Modos e filtrado
 */
#define MD_FAST 1
#define MD_NORMAL 2
#define MD_FILTERED 3
/**
 * @brief  Modos de lectura de tension
 */
#define CELL_CH_ALL 0
#define CELL_CH_1and7 1
#define CELL_CH_2and8 2
#define CELL_CH_3and9 3
#define CELL_CH_4and10 4
#define CELL_CH_5and11 5
#define CELL_CH_6and12 6
/**
 * @brief  Modos de lectura de GPIO
 */
#define AUX_CH_ALL 0
#define AUX_CH_GPIO1 1
#define AUX_CH_GPIO2 2
#define AUX_CH_GPIO3 3
#define AUX_CH_GPIO4 4
#define AUX_CH_GPIO5 5
#define AUX_CH_VREF2 6



void LTC6804_initialize();
void LTC_Set_ADC(uint8_t MD, uint8_t DCP, uint8_t CH, uint8_t CHG);
void LTC_Send_ADCV();
void LTC_Send_ADAX();
uint8_t LTC_RDcv(uint8_t reg, uint8_t total_ic, uint16_t cell_codes[][12]);
void LTC_RDcv_REG(uint8_t reg, uint8_t nIC, uint8_t *data);
int8_t LTC_RDaux(uint8_t reg, uint8_t nIC, uint16_t aux_codes[][6]);
void LTC_RDaux_REG(uint8_t reg, uint8_t nIC,uint8_t *data);
void LTC6804_clrcell();
void LTC6804_clraux();
void LTC_WR_CMD(uint8_t nIC,uint8_t config[][6],uint16_t cmd_code);
int8_t LTC_RD_CFG(uint8_t nIC, uint8_t r_config[][8]);
void LTC_isoSPI_Wake_Idle();
void LTC_WakeUp_Sleep();
uint16_t LTC_PEC15_CALC(uint8_t len, uint8_t *data);
void LTC_isoSPI_Write_Array( uint8_t length, uint8_t *data);
void LTC_isoSPI_Write_Read(uint8_t *TxData, uint8_t TXlen, uint8_t *rx_data, uint8_t RXlen);
void LTC_REFON(uint8_t total_ic);

#endif
```

1.3.2. LTC6801.c

```
/*****************************************************************************
 *        2018, Alfonso Mareca Miralles, Eupla Racing Team                   *
 *****************************************************************************/

/**
 * @file     LTC68041.c
 * @author   Alfonso Mareca Miralles <alfonso@euplart.com>
 * @author           Basado en el codigo de Analog Devices para el LTC6804 <evt-contact@mit.edu> .
 * @date     30 Sep 2018
 *
 */
#include "../BMS_LTC_Libraries/LTC68041.h"
#include <stdint.h>
#include "spi.h"
#include "../BMS_Libraries/delay.h"
#include <stdlib.h>
#include "string.h"
#include "../BMS_Libraries/BMS_defs.h"
#define SPI_CS_1 HAL_GPIO_WritePin(SPI3_ENABLE_GPIO_Port,SPI3_ENABLE_Pin,GPIO_PIN_SET)
#define SPI_CS_0 HAL_GPIO_WritePin(SPI3_ENABLE_GPIO_Port,SPI3_ENABLE_Pin,GPIO_PIN_RESET)
//Micro Delay
#define delayUS_ASM(us) do {\
asm volatile (     "MOV R0,%[loops]\n\t"\
                                "1: \n\t"\
                                "SUB R0, #1\n\t"\
                                "CMP R0, #0\n\t"\
                                "BNE 1b \n\t" : : [loops] "r" (16*us) : "memory"\
                            );\
} while(0)

uint8_t ADCV[2]; //!< Cell Voltage conversion command.
uint8_t ADAX[2]; //!< GPIO conversion command.
/**
 * @brief  Configuracion de los ADC
 *
 */
void LTC_Set_ADC(uint8_t MD, uint8_t DCP,uint8_t CH,uint8_t CHG)
{
  uint8_t md_bits;
  md_bits = (MD & 0x02) >> 1;
  ADCV[0] = md_bits + 0x02;
  md_bits = (MD & 0x01) << 7;
  ADCV[1] =  md_bits + 0x60 + (DCP<<4) + CH;
  md_bits = (MD & 0x02) >> 1;
  ADAX[0] = md_bits + 0x04;
  md_bits = (MD & 0x01) << 7;
  ADAX[1] = md_bits + 0x60 + CHG ;

}

/**
 * @brief  Envio de la configuracion del ADCV
 *
 */
void LTC_Send_ADCV()
{

  uint8_t cmd[4];
  uint16_t cmd_pec;
  ADCV[0]=0b011;
  ADCV[1]=0b11110000;
```

```
  cmd[0] = ADCV[0];
  cmd[1] = ADCV[1];
  cmd_pec = LTC_PEC15_CALC(2, ADCV);
  cmd[2] = (uint8_t)(cmd_pec >> 8);
  cmd[3] = (uint8_t)(cmd_pec);
  LTC_isoSPI_Wake_Idle ();
  LTC_isoSPI_Write_Array(4,cmd);
}


/**
 * @brief Envio de la configuracion ADAX
 *
 */
void LTC_Send_ADAX()
{
  uint8_t cmd[4];
  uint16_t cmd_pec;
  cmd[0] = ADAX[0];
  cmd[1] = ADAX[1];
  cmd_pec = LTC_PEC15_CALC(2, ADAX);
  cmd[2] = (uint8_t)(cmd_pec >> 8);
  cmd[3] = (uint8_t)(cmd_pec);
  LTC_isoSPI_Wake_Idle ();
  LTC_isoSPI_Write_Array(4,cmd);
}

/**
 * @brief  Lectura de tensiones
 *
 */
uint8_t LTC_RDcv(uint8_t reg,uint8_t total_ic,uint16_t ltc_cell_codes[][12])
{

  const uint8_t NUM_RX_BYT = 8;
  const uint8_t BYT_IN_REG = 6;
  const uint8_t CELL_IN_REG = 3;

  uint8_t *cell_data;
  uint8_t pec_error = 0;
  uint16_t parsed_cell;
  uint16_t received_pec;
  uint16_t data_pec;
  uint8_t data_counter=0;
  cell_data = (uint8_t *) malloc((NUM_RX_BYT*total_ic)*sizeof(uint8_t));
  if (reg == 0)
  {
    for (uint8_t cell_reg = 1; cell_reg<5; cell_reg++)
    {
      data_counter = 0;
      LTC_RDcv_REG(cell_reg, total_ic,cell_data );
      for (uint8_t current_ic = 0 ; current_ic < total_ic; current_ic++)
      {
        for (uint8_t current_cell = 0; current_cell<CELL_IN_REG; current_cell++)
        {
          parsed_cell = cell_data[data_counter] + (cell_data[data_counter + 1] << 8);
          ltc_cell_codes[current_ic][current_cell  + ((cell_reg - 1) * CELL_IN_REG)] = parsed_cell;
          data_counter = data_counter + 2;
        }
        received_pec = (cell_data[data_counter] << 8) + cell_data[data_counter+1];
        data_pec = LTC_PEC15_CALC(BYT_IN_REG, &cell_data[current_ic * NUM_RX_BYT]);
        if (received_pec != data_pec)
        {
          pec_error = -1;
        }
```

```
      data_counter=data_counter+2;
    }
   }
  }
  else
  {
   LTC_RDcv_REG(reg, total_ic,cell_data);
   for (uint8_t current_ic = 0 ; current_ic < total_ic; current_ic++)
   {
    for (uint8_t current_cell = 0; current_cell < CELL_IN_REG; current_cell++)
    {
     parsed_cell = cell_data[data_counter] + (cell_data[data_counter+1]<<8);
     ltc_cell_codes[current_ic][current_cell + ((reg - 1) * CELL_IN_REG)] = 0x0000FFFF & parsed_cell;
     data_counter= data_counter + 2;
    }
    received_pec = (cell_data[data_counter] << 8 )+ cell_data[data_counter + 1];
    data_pec = LTC_PEC15_CALC(BYT_IN_REG, &cell_data[current_ic * NUM_RX_BYT]);
    if (received_pec != data_pec)
    {
     pec_error = -1;
    }
    data_counter= data_counter + 2;
   }
  }
  free(cell_data);
  return(pec_error);
}


/**
 * @brief  Lectura de un registro de tensiones
 *
 */

void LTC_RDcv_REG(uint8_t reg,uint8_t total_ic,uint8_t *data )
{
  const uint8_t REG_LEN = 8;
  uint8_t cmd[4];
  uint16_t cmd_pec;

  //1
  if (reg == 1)     //1: RDCVA
  {
    cmd[1] = LTC_CMD_RDCVA&&0x0F;
    cmd[0] = LTC_CMD_RDCVA>>8;
  }
  else if (reg == 2) //2: RDCVB
  {
    cmd[1] = LTC_CMD_RDCVB&&0x0F;
    cmd[0] = LTC_CMD_RDCVB>>8;
  }
  else if (reg == 3) //3: RDCVC
  {
    cmd[1] = LTC_CMD_RDCVC&&0x0F;
    cmd[0] = LTC_CMD_RDCVC>>8;
  }
  else if (reg == 4) //4: RDCVD
  {
    cmd[1] = LTC_CMD_RDCVC&&0x0F;
    cmd[0] = LTC_CMD_RDCVC>>8;
  }
  cmd_pec = LTC_PEC15_CALC(2, cmd);
  cmd[2] = (uint8_t)(cmd_pec >> 8);
  cmd[3] = (uint8_t)(cmd_pec);
  LTC_isoSPI_Wake_Idle ();
```

```
  LTC_isoSPI_Write_Read(cmd,4,data,(REG_LEN*total_ic));

}
/**
 * @brief  Lectura de registros auxiliares GPIO
 *
 */
int8_t LTC_RDaux(uint8_t reg,uint8_t total_ic, uint16_t aux_codes[][6] )
{


  const uint8_t NUM_RX_BYT = 8;
  const uint8_t BYT_IN_REG = 6;
  const uint8_t GPIO_IN_REG = 3;
  uint8_t *data;
  uint8_t data_counter = 0;
  int8_t pec_error = 0;
  uint16_t parsed_aux;
  uint16_t received_pec;
  uint16_t data_pec;
  data = (uint8_t *) malloc((NUM_RX_BYT*total_ic)*sizeof(uint8_t));
  if (reg == 0)
  {
    for (uint8_t gpio_reg = 1; gpio_reg<3; gpio_reg++)
    {
      data_counter = 0;
      LTC_RDaux_REG(gpio_reg, total_ic,data);
      for (uint8_t current_ic = 0 ; current_ic < total_ic; current_ic++)
      {
        for (uint8_t current_gpio = 0; current_gpio< GPIO_IN_REG; current_gpio++)
        {
          parsed_aux = data[data_counter] + (data[data_counter+1]<<8);
          aux_codes[current_ic][current_gpio +((gpio_reg-1)*GPIO_IN_REG)] = parsed_aux;
          data_counter=data_counter+2;
        }

        received_pec = (data[data_counter]<<8)+ data[data_counter+1];
        data_pec = LTC_PEC15_CALC(BYT_IN_REG, &data[current_ic*NUM_RX_BYT]);
        if (received_pec != data_pec)
        {
          pec_error = -1;
        }
        data_counter=data_counter+2;
      }
    }
  }
  else
  {
          LTC_RDaux_REG(reg, total_ic, data);
    for (int current_ic = 0 ; current_ic < total_ic; current_ic++)
    {
      for (int current_gpio = 0; current_gpio<GPIO_IN_REG; current_gpio++)
      {
        parsed_aux = (data[data_counter] + (data[data_counter+1]<<8));
        aux_codes[current_ic][current_gpio +((reg-1)*GPIO_IN_REG)] = parsed_aux;
        data_counter=data_counter+2;
      }
      received_pec = (data[data_counter]<<8) + data[data_counter+1];
      data_pec = LTC_PEC15_CALC(BYT_IN_REG, &data[current_ic*NUM_RX_BYT]);
      if (received_pec != data_pec)
      {
        pec_error = -1;
      }
      data_counter=data_counter+2;
    }
```

```
  }
  free(data);
  return (pec_error);
}
/**
 * @brief  Lectura de un registro auxiliar GPIO
 *
 */
void LTC_RDaux_REG(uint8_t reg,uint8_t total_ic, uint8_t *data )
{
  const uint8_t REG_LEN = 8;
  uint8_t cmd[4];
  uint16_t cmd_pec;

  //1
  if (reg == 1)
  {
    cmd[1] = (uint8_t)LTC_CMD_RDAUXA;
    cmd[0] = (uint8_t)LTC_CMD_RDAUXA>>8;
  }
  else if (reg == 2)
  {
    cmd[1] = (uint8_t)LTC_CMD_RDAUXB;
    cmd[0] = (uint8_t)LTC_CMD_RDAUXB>>8;
  }
  else
  {
            cmd[1] = (uint8_t)LTC_CMD_RDAUXA;
            cmd[0] = (uint8_t)LTC_CMD_RDAUXA>>8;
  }
  cmd_pec = LTC_PEC15_CALC(2, cmd);
  cmd[2] = (uint8_t)(cmd_pec >> 8);
  cmd[3] = (uint8_t)(cmd_pec);
  LTC_isoSPI_Wake_Idle ();
  LTC_isoSPI_Write_Read(cmd,4,data,(REG_LEN*total_ic));

}
/**
 * @brief  Limpieza de registros de tension
 *
 */
void LTC6804_clrcell()
{
  uint8_t cmd[4];
  uint16_t cmd_pec;
  cmd[0] = 0x07;
  cmd[1] = 0x11;
  cmd_pec = LTC_PEC15_CALC(2, cmd);
  cmd[2] = (uint8_t)(cmd_pec >> 8);
  cmd[3] = (uint8_t)(cmd_pec );
  LTC_isoSPI_Wake_Idle ();
  LTC_isoSPI_Write_Read(cmd,4,0,0);
}
/**
 * @brief  Limpieza de registros auxiliares GPIO
 *
 */
void LTC6804_clraux()
{
  uint8_t cmd[4];
  uint16_t cmd_pec;
  cmd[0] = 0x07;
  cmd[1] = 0x12;
  cmd_pec = LTC_PEC15_CALC(2, cmd);
  cmd[2] = (uint8_t)(cmd_pec >> 8);
```

```
  cmd[3] = (uint8_t)(cmd_pec);
  LTC_isoSPI_Wake_Idle ();
  LTC_isoSPI_Write_Read(cmd,4,0,0);
}
/**
 * @brief  Configuracion de un registro especifico
 *
 */
void LTC_WR_CMD(uint8_t total_ic, uint8_t config[][6],uint16_t cmd_code)
{
  const uint8_t BYTES_IN_REG = 6;
  const uint8_t CMD_LEN = 4+(8*total_ic);
  uint8_t *cmd;
  uint16_t cfg_pec, cmd_pec;
  uint8_t cmd_index;
  cmd = (uint8_t *)malloc(CMD_LEN*sizeof(uint8_t));
  cmd[0] =(uint8_t)(cmd_code >> 8);
  cmd[1] =(uint8_t)cmd_code;
  cmd_pec = LTC_PEC15_CALC(2, cmd);
  cmd[2] = (uint8_t)(cmd_pec >> 8);
  cmd[3] = (uint8_t)(cmd_pec);
  cmd_index = 4;
  for (uint8_t current_ic = total_ic; current_ic > 0; current_ic--)
  {
    for (uint8_t current_byte = 0; current_byte < BYTES_IN_REG; current_byte++)
    {
      cmd[cmd_index] = config[current_ic-1][current_byte];
      cmd_index = cmd_index + 1;
    }
    cfg_pec = (uint16_t)LTC_PEC15_CALC(BYTES_IN_REG, &config[current_ic-1][0]);
    cmd[cmd_index] = (uint8_t)(cfg_pec >> 8);
    cmd[cmd_index + 1] = (uint8_t)cfg_pec;
    cmd_index = cmd_index + 2;
  }
  LTC_isoSPI_Wake_Idle ();
  LTC_isoSPI_Write_Array(CMD_LEN, cmd);
  free(cmd);
}
/**
 * @brief  Lectura del registro de configuracion
 *
 */
int8_t LTC_RD_CFG(uint8_t total_ic, uint8_t r_config[][8])
{
  const uint8_t BYTES_IN_REG = 8;

  uint8_t cmd[4];
  uint8_t *rx_data;
  int8_t pec_error = 0;
  uint16_t data_pec, cmd_pec;
  uint16_t received_pec;
  rx_data = (uint8_t *) malloc((8*total_ic)*sizeof(uint8_t));
  cmd[0] = LTC_CMD_RDCFG>>8;
  cmd[1] = LTC_CMD_RDCFG&&0x0F;
  cmd_pec = LTC_PEC15_CALC(2, cmd);
  cmd[2] = (uint8_t)(cmd_pec >> 8);
  cmd[3] = (uint8_t)(cmd_pec);
  LTC_isoSPI_Wake_Idle ();
  LTC_isoSPI_Write_Read(cmd, 4, rx_data, (BYTES_IN_REG*total_ic));
  for (uint8_t current_ic = 0; current_ic < total_ic; current_ic++)
  {
    for (uint8_t current_byte = 0; current_byte < BYTES_IN_REG; current_byte++)
    {
      r_config[current_ic][current_byte] = rx_data[current_byte + (current_ic*BYTES_IN_REG)];
    }
```

```
    received_pec = (r_config[current_ic][6]<<8) + r_config[current_ic][7];
    data_pec = LTC_PEC15_CALC(6, &r_config[current_ic][0]);
    if (received_pec != data_pec)
    {
      pec_error = -1;
    }
  }
  free(rx_data);
  return(pec_error);
}

/**
 * @brief  Wake del LTC6804
 *
 */
void LTC_isoSPI_Wake_Idle()
{
        SPI_CS_0;
        HAL_SPI_Transmit(&hspi3,(uint8_t *)0x00,1,5000);
        delayUS_ASM(300);    // 100 microsec*300microsec

        SPI_CS_1;

}
void LTC_REFON(uint8_t total_ic) {
        uint8_t tx_cfg[total_ic][6];
        for (uint8_t ltc = 0; ltc < total_ic; ltc++) {
                        tx_cfg[ltc][0] |= 0xFE;
                        tx_cfg[ltc][1] |= 0x00;
                        tx_cfg[ltc][2] |= 0x00;
                        tx_cfg[ltc][3] |= 0x00;
                        tx_cfg[ltc][4] |= 0x00;
                        tx_cfg[ltc][5] |= 0x00;
                }
    LTC_WR_CMD(total_ic,tx_cfg,LTC_CMD_WRCFG);
}
/**
 * @brief  Sleep del LTC6804
 *
 */
void LTC_WakeUp_Sleep()
{
                SPI_CS_0;
                HAL_SPI_Transmit(&hspi3,(uint8_t *)0x00,1,5000);
                delayUS_ASM(300);    // 100 microsec*300microsec

                SPI_CS_1;

}
/**
 * @brief  Calculo del PEC
 *
 */
uint16_t LTC_PEC15_CALC(uint8_t len,uint8_t *data)
{
  uint16_t remainder,addr;

  remainder = 16;
  for (uint8_t i = 0; i<len; i++)
  {
    addr = ((remainder>>7)^data[i])&0xff;
    remainder = (remainder<<8)^crc15Table[addr];
  }
  return(remainder*2);
}
```

```
/**
 * @brief  Escritura de un vector al daisy-chain
 *
 */
void LTC_isoSPI_Write_Array(uint8_t len,uint8_t data[])
{
          SPI_CS_0;
          delayUS_ASM(7);
          HAL_SPI_Transmit(&hspi3,(uint8_t *)data,len,5000);
   SPI_CS_1;
}
/**
 * @brief  Escritura y lectura de un vector al daisy-chain
 *
 */
void LTC_isoSPI_Write_Read(uint8_t tx_Data[], uint8_t tx_len,uint8_t *rx_data,uint8_t rx_len)
{
  SPI_CS_0;
  delayUS_ASM(7);
  HAL_SPI_Transmit(&hspi3,(uint8_t *)tx_Data,tx_len,5000);
  HAL_SPI_Receive(&hspi3,(uint8_t *)rx_data,rx_len,5000);
  SPI_CS_1;
}
```

1.3.3. LTC_balancing.h

```
/*****************************************************************************
 *        2018, Alfonso Mareca Miralles, Eupla Racing Team              *
 *****************************************************************************/

/**
 * @file     LTC_balancing.h
 * @author   Alfonso Mareca Miralles <alfonso@euplart.com>
 * @author           Basado en el codigo de MIT EVT <evt-contact@mit.edu> .
 * @date     30 Sep 2018
 *
 * @brief    Header del archivo donde se incluyen las funciones de control de balanceo
 */


#ifndef LTC_balancing_H
#define LTC_balancing_H

#include "main.h"
#include <stdio.h>
#include <stdlib.h>

#include "../BMS_Libraries/BMS_defs.h"



void LTC_Balancing_Update(ltc_reg_t *ltc_reg,ltc_conf_t *ltc_conf,op_modes_t* bms_mode, uint8_t
ltc_active_cells[9][12],status_t *bms_status);
void LTC_Balancing_Stop(ltc_reg_t *ltc_reg,ltc_conf_t *ltc_conf);
void LTC_Balancing_Set(uint8_t cell, uint8_t state, uint8_t ltc,uint8_t tx_cfg[][6]);

#endif
```

1.3.4. LTC_balancing.c

```
/*******************************************************************************
 *        2018, Alfonso Mareca Miralles, Eupla Racing Team                     *
 *******************************************************************************/

/**
 * @file     LTC_balancing.c
 * @author   Alfonso Mareca Miralles <alfonso@euplart.com>
 * @author           Basado en el codigo de MIT EVT <evt-contact@mit.edu> .
 * @date     30 Sep 2018
 *
 * @brief    Archivo donde se incluyen las diferentes funciones de control de balanceo
 */

#include "../BMS_LTC_Libraries/LTC_balancing.h"
#include <stdio.h>
#include <stdlib.h>
#include "../BMS_Libraries/delay.h"
#include "string.h"
#include "../BMS_LTC_Libraries/LTC68041.h"

/**
 * @fn LTC_Balancing_Update (ltc_reg_t *ltc_reg, ltc_conf_t *ltc_conf, uint8_t ltc_active_cells[9][12])
 * @brief Realiza la lectura y estudio de las celdas a balancear de forma temporizada,
 * para realizar el estudio de los valores de tensión el balanceo se detiene durante un corto periodo de tiempo,
 * para luego ser activado durante otro periodo de tiempo.
 * @param ltc_reg            estructura que contiene los datos de tensión obtenidos asi como otras variables
 * @param ltc_conf           estructura que contiene los valores de configuracion del bms
 * @param ltc_active_cells variable que define que puntos de lectura sun celas y cuales son nulas
 */

void LTC_Balancing_Update(ltc_reg_t *ltc_reg,ltc_conf_t *ltc_conf,op_modes_t* bms_mode, uint8_t
ltc_active_cells[9][12],status_t *bms_status) {
        static uint32_t t_preb = 0;
        uint8_t counter = 0;
        if (*bms_mode == charge_op) {
                if (t_preb == 0) {
                        LTC_Balancing_Stop(ltc_reg, ltc_conf);
                        t_preb = HAL_GetTick();
                }
                if ((HAL_GetTick() - t_preb > 250) && (HAL_GetTick() - t_preb < 300)) {
                        uint16_t lowest_v = ltc_conf->cell_ov;
                        for (uint8_t ltc = 0; ltc < ltc_conf->ic_num; ltc++) {
                                for (uint8_t cell = 0; cell < 12; cell++) {
                                        if (ltc_active_cells[ltc][cell] != 0 && ltc_reg->v_cell[ltc][cell] <
lowest_v && (ltc_reg->v_cell[ltc][cell] > ltc_conf->cell_uv)) {
                                                lowest_v = ltc_reg->v_cell[ltc][cell];

                                        }
                                }

                        }
#ifdef DEBUG_LTC
                        printf("\nLa celda mas baja es: %1.4f  \n", lowest_v * 0.0001);
#endif
                        for (uint8_t ltc = 0; ltc < ltc_conf->ic_num; ltc++) {
                                for (uint8_t cell = 0; cell < 12; cell++) {
                                        if ((ltc_active_cells[ltc][cell] != 0) && (ltc_reg->v_cell[ltc][cell] >
(lowest_v + ltc_conf->cell_margin))) {
                                                LTC_Balancing_Set(cell, 1, ltc, ltc_reg->tx_cfg);
                                                bms_general_data.cell_balance[counter] = 1;
```

```
                                        } else {
                                                LTC_Balancing_Set(cell, 0, ltc, ltc_reg->tx_cfg);
                                                bms_general_data.cell_balance[counter] = 0;
                                        }
                                        counter++;
                                }
                        }
                };
                if (HAL_GetTick() - t_preb > 700) {
                        LTC_WR_CMD(ltc_conf->ic_num, ltc_reg->tx_cfg, LTC_CMD_WRCFG);
#ifdef DEBUG_LTC
                        printf("\nEnvio Balanceo\n");
#endif
                }
                if (HAL_GetTick() - t_preb > 1500) {  //estable a 1500  900 sirve
                        t_preb = 0;
                }
        }
}


/**
 * @fn LTC_Balancing_Stop(ltc_reg_t *ltc_reg,ltc_conf_t *ltc_conf)
 * @brief Realiza el paro forzado del balanceo
 * @param ltc_reg           estructura que contiene los datos de tensión obtenidos asi como otras variables
 * @param ltc_conf          estructura que contiene los valores de configuracion del bms
 */
void LTC_Balancing_Stop(ltc_reg_t *ltc_reg,ltc_conf_t *ltc_conf) {
        for (uint8_t ltc = 0; ltc < ltc_conf->ic_num; ltc++) {
                for (uint8_t cell = 0; cell < 12; cell++) {
                        LTC_Balancing_Set(cell, 0, ltc, ltc_reg->tx_cfg);
                }
        }
   LTC_WR_CMD(ltc_conf->ic_num,ltc_reg->tx_cfg,LTC_CMD_WRCFG);

}

/**
 * @fn LTC_Balancing_Set(uint8_t cell, uint8_t state, uint8_t ltc,uint8_t tx_cfg[][6])
 * @brief Realiza la ordenacion de bits a enviar al ltc6804 para el balanceo
 * @param cell     Celda a balancear
 * @param state   estado para balanceo 1->activo 0->deshabilitado
 * @param ltc      Chip del cual se  desea balancear
 * @param tx_cfg Dato sobre el que guardar la configuración
 */
void LTC_Balancing_Set(uint8_t cell, uint8_t state, uint8_t ltc,uint8_t tx_cfg[][6]) {
        //  cell -= ltc * 12;

        if (cell >= 0 && cell < 8) {
                if (state)
                        tx_cfg[ltc][4] |= 0b1 << cell;
                else
                        tx_cfg[ltc][4] &= ~(0b1 << cell);
        } else if (cell > 7 && cell < 12) {
                if (state)
                        tx_cfg[ltc][5] |= 0b1 << (cell - 8);
                else
                        tx_cfg[ltc][5] &= ~(0b1 << (cell - 8));
        }
}
```

1.3.5. LTC_Temp.h

```
/*****************************************************************************
 *        2018, Alfonso Mareca Miralles, Eupla Racing Team                   *
 *****************************************************************************/

/**
 * @file     LTC_Temp.h
 * @author   Alfonso Mareca Miralles <alfonso@euplart.com>
 * @date     30 Sep 2018
 *
 * @brief    Header del archivo  donde se incluyen las diferentes funciones de la lectura de Temperatura
 */


#ifndef BMS_LTC_LIBRARIES_LTC_TEMP_H_
#define BMS_LTC_LIBRARIES_LTC_TEMP_H_
#include "main.h"
#include <stdio.h>
#include <stdlib.h>
#include "../BMS_Libraries/BMS_defs.h"

void LTC_Mux(uint8_t total_ic, uint8_t mux, uint8_t channel);
void LTC_All_Bal_T(ltc_reg_t *ltc_reg,ltc_conf_t *ltc_conf,ltc_temp_call_t *ltc_bal_cal);
read_t LTC_All_Mux_T(ltc_reg_t *ltc_reg, ltc_conf_t *conf,status_t *status, ltc_temp_call_t *ltc_mux_cal);
void LTC_All_Mux_V(ltc_reg_t *ltc_reg,ltc_conf_t *ltc_conf, uint16_t data[24]);
#endif /* BMS_LTC_LIBRARIES_LTC_TEMP_H_ */
```

1.3.6. LTC_Temp.c

```
/*****************************************************************************
 *        2018, Alfonso Mareca Miralles, Eupla Racing Team                   *
 *****************************************************************************/

/**
 * @file     LTC_Temp.c
 * @author   Alfonso Mareca Miralles <alfonso@euplart.com>
 * @date     30 Sep 2018
 *
 * @brief    Archivo donde se incluyen las diferentes funciones de la lectura de Temperatura
 */
#include "../BMS_LTC_Libraries/LTC_Temp.h"
#include <stdio.h>
#include <stdlib.h>
#include "../BMS_Libraries/delay.h"
#include "string.h"
#include "../BMS_LTC_Libraries/LTC68041.h"
#include "../BMS_Libraries/LTC_Core.h"
#define DEBUG_LTC

/**
 * @fn LTC_All_Bal_T(ltc_reg_t *ltc_reg,ltc_conf_t *ltc_conf,ltc_temp_call_t *ltc_bal_cal)
 * @brief Realiza la lectura de las sondas de temperatura de balanceo
 * @param ltc_reg estructura donde se almacenaran las temperaturas logueadas
 * @param ltc_conf          estructura que contiene los valores de configuracion del bms
 * @param ltc_bal_cal estructura que contiene los valores de calibracion para las ntc de balanceo
 */
void LTC_All_Bal_T(ltc_reg_t *ltc_reg,ltc_conf_t *ltc_conf,ltc_temp_call_t *ltc_bal_cal) {
        static uint32_t t_preb = 0;
        uint8_t counter = 0;
        int16_t t_rangue = ltc_bal_cal->t_high - ltc_bal_cal->t_low;
        int16_t v_rangue = ltc_bal_cal->v_high - ltc_bal_cal->v_low;
```

```
		if (t_preb == 0) {
			t_preb = HAL_GetTick();
		};
		if (HAL_GetTick() - t_preb > 500) {
			for (uint8_t ltc = 0; ltc< ltc_conf->ic_num;ltc++){
				for (uint8_t i = 0; i<2;i++){
					t_preb = 0;
					ltc_reg->v_temp_bal[counter] = ltc_reg->v_aux[ltc][i+1];
					ltc_reg->temp_bal[counter] = ((((ltc_reg->v_aux[ltc][i+1] - ltc_bal_cal->v_low)* t_rangue) / v_rangue) + ltc_bal_cal->t_low);
					if (ltc_reg->temp_bal[counter]*0.01 > 500){
						ltc_reg->temp_bal[counter]= 0;
					}
					counter++;
				}
			}
		}
}
/**
 * @fn LTC_All_Mux_T(ltc_reg_t *ltc_reg,ltc_conf_t *ltc_conf,ltc_temp_call_t *ltc_mux_cal)
 * @brief Realiza la lectura de todas las sondas de temperatura multiplexadas con su correspondiente calibracion a Grados centigrados
 * @param ltc_reg estructura donde se almacenaran las temperaturas logueadas
 * @param ltc_conf		estructura que contiene los valores de configuracion del bms
 * @param ltc_mux_cal datos de calibracion a dos puntos
 */
read_t LTC_All_Mux_T(ltc_reg_t *ltc_reg, ltc_conf_t *conf,status_t *status, ltc_temp_call_t *ltc_mux_cal) {
		static uint8_t chann = 0;
		int16_t t_rangue = ltc_mux_cal->t_high - ltc_mux_cal->t_low;
		int16_t v_rangue = ltc_mux_cal->v_high - ltc_mux_cal->v_low;
		//LTC_WakeUp_Sleep();
		static uint32_t t_preb = 0;
		if (t_preb == 0) {
			t_preb = HAL_GetTick();
			LTC_WR_CMD(conf->ic_num, ltc_reg->tx_cfg, LTC_CMD_WRCFG);
			LTC_Mux(conf->ic_num, 0, chann);
		}
		if (HAL_GetTick() - t_preb >300) {
			t_preb = 0;
			LTC_WR_CMD(conf->ic_num, ltc_reg->tx_cfg, LTC_CMD_WRCFG);
			LTC_Send_ADAX();
			LTC_Send_ADAX();
			LTC_RDaux(0, conf->ic_num, ltc_reg->v_aux);

			if (chann == 0) {
				for (int i = 0; i < ltc_conf.ic_num; i++) {
					ltc_reg->v_temp_cell[7 + 8 * i] = ltc_reg->v_aux[i][0];
				}
			} else {
				for (int i = 0; i < ltc_conf.ic_num; i++) {
					ltc_reg->v_temp_cell[(chann - 1) + 8 * i] = ltc_reg->v_aux[i][0];
				}
				for (int i = 0; i < 24; i++) {
					uint16_t old = ltc_reg->temp_modules[i];
					ltc_reg->temp_modules[i] = ((((ltc_reg->v_temp_cell[i] - ltc_mux_cal->v_low) * t_rangue) / v_rangue) + ltc_mux_cal->t_low);
					if ((ltc_reg->temp_modules[i] * 0.01) > 110) {
						ltc_reg->temp_modules[i] = old;
					}
				}
			}
			chann++;
			Sec_Temp_Modules(ltc_reg,conf,status);
			Sec_Temp_Bal(ltc_reg,conf,status);
		}
```

```
		if (chann == 8) {

			chann = 0;
			return read;
		}
		return noread;
}
/**
 * @fn LTC_All_Mux_V(ltc_conf_t *ltc_conf, uint16_t data[24])
 * @brief Realiza la lectura de todas las sondas de temperatura multiplexadas en bruto en voltajes
 * @param ltc_conf		estructura que contiene los valores de configuracion del bms
 * @param data variable de salida de todos los voltajes
 */

void LTC_All_Mux_V(ltc_reg_t *ltc_reg,ltc_conf_t *ltc_conf, uint16_t data[24]) {
		static uint32_t t_preb = 0;
		static uint8_t chann = 0;
		if (t_preb == 0) {
			LTC_WakeUp_Sleep();
			LTC_Set_ADC(MD_NORMAL, DCP_DISABLED, CELL_CH_ALL, AUX_CH_ALL);
			LTC_Send_ADAX();
			//LTC_REFON(total_ic);
			t_preb = HAL_GetTick();
			LTC_Mux(ltc_conf->ic_num, 0, 1);
		};

		if (HAL_GetTick() - t_preb > 10) {
			//LTC_Send_ADAX();
			if (HAL_GetTick() - t_preb > 50) {

				t_preb = 0;
				LTC_RDaux(0, ltc_conf->ic_num, ltc_reg->v_aux);
				for (uint8_t i = 0; i < ltc_conf->ic_num; i++) {
					data[chann] = ltc_reg->v_aux[i][0];
				}
				chann++;
			}
		}
		if (chann > 8) {
			chann = 0;
		};
}
/**
 * @fn LTC_Mux(uint8_t total_ic, uint8_t mux, uint8_t channel)
 * @brief Configura los mux  incluye el envio
 * @param total_ic		Numero de chips del BMS
 * @param mux			Mux a seleccionar  depende de la adress seleccionada  0 1 2 3
 * @param channel		Canal a fijar  0 al 7
 */

void LTC_Mux(uint8_t total_ic, uint8_t mux, uint8_t channel) {
		uint8_t config[total_ic][6];
		uint16_t i = 0;

		uint8_t address = 0x4C | (mux % 3);		//direccion base del multiplexor
		uint8_t data = 1 << (channel % 8);
		if (channel == 0xFF) {					// Si no hay canal seleccionado, alta
impedancia
			data = 0x00;
		}
		for (i = 0; i < total_ic; i++) {
			config[i][0] = LTC_ICOM_START | (address >> 3);				// 0x6 :
LTC6804: ICOM START from Master
			config[i][1] = LTC_FCOM_MASTER_NACK | (address << 5);
			config[i][2] = LTC_ICOM_BLANK | (data >> 4);
```

```
                config[i][3] = LTC_FCOM_MASTER_NACK_STOP | (data << 4);
                config[i][4] = LTC_ICOM_NO_TRANSMIT;
// 0x1 : ICOM-STOP
                config[i][5] = 0x00;
          // 0x0 : dummy (Dn)

                                                // 9: MASTER NACK + STOP (FCOM)
        }
        LTC_WR_CMD(total_ic, config, LTC_CMD_WRCOMM);
        for (i = 0; i < total_ic; i++) {
                for (uint8_t t = 0; t < 6; t++) {
                        config[i][t] = 0xFF;
                }
        }
        LTC_WR_CMD(total_ic, config, LTC_CMD_STCOMM);

}
```

1.4. LIBRERÍA DEL SENSOR (BMS_SENSOR)

Se expone el código fuente de la librería BMS_Sensor:

BMS_Sensor		Librería del sensor HALL:
BMS_C_Sensor.h	BMS_C_Sensor.c	Funciones de comunicación con el MCP3421.

1.4.1. BMS_C_Sensor.h

```
/*****************************************************************************
 *        2018, Alfonso Mareca Miralles, Eupla Racing Team                   *
 *****************************************************************************/

/**
 * @file     BMS_C_Sensor.h
 * @author   Alfonso Mareca Miralles <alfonso@euplart.com>
 * @date     30 Sep 2018
 *
 * @brief    Archivo donde se incluyen las cabeceras de las funciones para el sensor de corriente
 */
#include "main.h"
#include "stm32f4xx_hal.h"
#include "../BMS_Libraries/BMS_defs.h"

#ifndef BMS_SENSOR_BMS_C_SENSOR_H_
#define BMS_SENSOR_BMS_C_SENSOR_H_
void C_Sensor_Init(uint8_t gain);
int32_t MCP_Read(void);
float MCP_calc(uint8_t gain);
void C_Sensor_Read(uint8_t gain,current_reg_t *current_s_reg);

#endif /* BMS_SENSOR_BMS_C_SENSOR_H_ */
```

1.4.2. BMS_C_Sensor.c

```
/*****************************************************************************
 *        2018, Alfonso Mareca Miralles, Eupla Racing Team                   *
 *****************************************************************************/

/**
 * @file     BMS_C_Sensor.c
 * @author   Alfonso Mareca Miralles <alfonso@euplart.com>
 * @date     30 Sep 2018
 *
 * @brief    Archivo donde se incluyen las las funciones para el sensor de corriente
 */
#include "BMS_C_Sensor.h"
#include "main.h"
#include "stm32f4xx_hal.h"
#include "i2c.h"
#include "../BMS_Libraries/BMS_defs.h"
#include "../BMS_Libraries/LTC_Core.h"
#define MCP_Adress_Read 0xD1
#define MCP_Adress_write 0xD0
/**
 * @brief Inicializacion del MCP
 */
void C_Sensor_Init(uint8_t gain) {
        uint8_t Check[4] = { 0 };
        uint8_t Configuration_byte = 0x1C | gain;
        uint8_t Configuration_byte2 = 0x9C | gain;
        HAL_I2C_Master_Transmit(&hi2c2, MCP_Adress_write, &Configuration_byte, 1, 0x10);
        HAL_I2C_Master_Receive(&hi2c2, MCP_Adress_Read, Check, 4, 0x10);
        if (Check[3] == Configuration_byte || Check[3] == Configuration_byte2) {
                printf("Sensor de corriente configurado\n");
        } else {
                printf("Sensor de corriente NO configurado\n");
        }

}
/**
 * @brief Lectura del MCP
 */
int32_t MCP_Read(void) {
        uint8_t pData[4] = { 0, 0, 0, 0 };
        uint8_t Temp, Temp2;
        uint32_t comp;
        HAL_I2C_Master_Receive(&hi2c2, MCP_Adress_Read, pData, 4, 0x10);

        Temp = pData[0];
        Temp = Temp & 0b00000011;
        Temp2 = Temp && 0b00000001;
        if ((Temp >> 1) == 0x01) {
                if (pData[1] == 0xFF) {
                        return 0x00;
                } else {
                        comp = ~(((Temp2 << 16) | (pData[1] << 8) | (pData[2]))) & 0x1FFFF;
                        return -1 * (int32_t) comp;
                }

        }
        if ((Temp >> 1) == 0x00) {
                return (int32_t) ((pData[0] << 16) | (pData[1] << 8) | (pData[2]));
        }
        return 0x00;
}
```

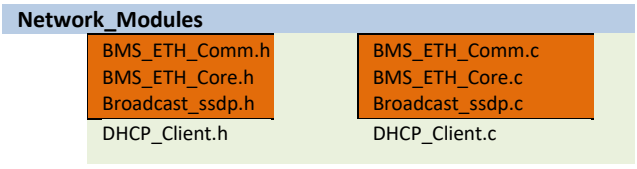
```
/**
 * @brief Calculo del MCP
 */
float MCP_calc(uint8_t gain) {
	int32_t dato = MCP_Read();
	return ((dato * 2.048) / (131071 + 1)) / (gain + 1);
}
/**
 * @brief Calculo del sensor de corriente
 */
void C_Sensor_Read(uint8_t gain, current_reg_t *current_s_reg) {

	float value;
	int16_t value2;
	static uint32_t t_sensor = 0;
	if (t_sensor == 0) {
		t_sensor = HAL_GetTick();
	}
	if (HAL_GetTick() - t_sensor > 10) {
		t_sensor = 0;
		value = MCP_calc(gain);
		value = value * 400 / 1.25;
		if ((value < 5) && (value > -5)) {
			value = 0x00;
		}
		value2 = (int16_t) (value);
		current_s_reg->amps = value2;
		if ((value > 5) && (value < -5)) {

		}
	}
}
```

1.5. LIBRERÍA ETHERNET (NETWORK_MODULES)

Se expone el código fuente de la librería Network_Modules:



Librería de red:
ETH_Core-> Principales funciones de comunicación
ETH_Comms-> Mensajeria eth
Broadcast-> Funcionalidad de conexión a la interfaz
BHCP_Client->Cliente DHCP

Se ha remarcado el código de desarrollo propio que se expondrá a continuación, el resto es generado automáticamente por STM32CubeMX con las opciones especificadas en el desarrollo.

1.5.1. BMS_ETH_Core.h

```
/*******************************************************************************
 *        2018, Alfonso Mareca Miralles, Eupla Racing Team                     *
 *******************************************************************************/

/**
 * @file     BMS_ETH_Core.h
 * @author   Alfonso Mareca Miralles <alfonso@euplart.com>
 * @author           Header de las funciones Ethernet
 * @date     30 Sep 2018
```

```
 *
 */

#include "main.h"
#include "lwip.h"
#include "../BMS_Libraries/BMS_defs.h"
#ifndef BMS_ETH_CORE_H_
#define BMS_ETH_CORE_H_
#define TEST_DATA_COMMS
//#define DEBUG_COMMS
#define DEBUG_LTC


typedef enum {wire_on,wire_stop,wire_off,wire_dhcp,wire_search,gui_init,gui} eth_modes_t;
typedef enum  {celsius, volt} temp_send_mode_t;
typedef struct {
        ip4_addr_t ip;
        uint16_t port;

} bms_gui_t;
bms_gui_t bms_gui_comms;
eth_modes_t eth_link(eth_modes_t *eth_mode);
eth_modes_t eth_init_deinit(eth_modes_t *eth_mode);
extern eth_modes_t eth_mode;
extern temp_send_mode_t eth_temp_mode;


void Comms_callback(void *arg, struct udp_pcb *upcb, struct pbuf *p, const ip_addr_t *addr, u16_t port);
void Comms_init(struct netif *netif);
void Comms_Handler(struct netif *netif,char *payload);

ltc_reg_t bms_example;

#endif /* BMS_ETH_CORE_H_ */
```

1.5.2. BMS_ETH_Core.c

```
/*******************************************************************************
 *        2018, Alfonso Mareca Miralles, Eupla Racing Team              *
 *******************************************************************************/

/**
 * @file     BMS_ETH_Core.c
 * @author   Alfonso Mareca Miralles <alfonso@euplart.com>
 * @author           Funcionesunciones Ethernet
 * @date     30 Sep 2018
 *
 */

#include "BMS_ETH_Core.h"
#include "main.h"
#include "lwip.h"
#include "DHCP_Client.h"
#include "Broadcast_ssdp.h"
#include "stm32f4xx_hal.h"
#include <string.h>
#include "../BMS_Libraries/delay.h"

extern struct netif gnetif;

static struct udp_pcb *comms_pcb;
eth_modes_t eth_mode;
temp_send_mode_t eth_temp_mode;
```

```
/**
 * @brief  Funcion principal de Ethernet
 *
 */
eth_modes_t eth_init_deinit(eth_modes_t *eth_mode){
	if (*eth_mode == wire_on) {

	printf("Red Ethernet conectada\n");

				netif_set_link_up(&gnetif);
				netif_set_default(&gnetif);
				netif_set_up(&gnetif);
				DHCP_state = DHCP_START;
				return wire_dhcp;
			}
			if ((*eth_mode == wire_dhcp)||(*eth_mode == wire_search)) {
			DHCP_enum_t aux = DHCP_Client_Handler(&gnetif);
			if ((aux == DHCP_ADDRESS_ASSIGNED)) {
				Broadcast_ssdp_Handler(&gnetif, eth_mode);

			}
	}
	if (*eth_mode == wire_stop) {

	printf("Red Ethernet desconectada\n");

			dhcp_stop(&gnetif);
			netif_set_down(&gnetif);
			netif_set_link_down(&gnetif);
			*eth_mode = wire_off;
	}
	return *eth_mode;
}
/**
 * @brief Estudio del Link
 *
 */
eth_modes_t eth_link(eth_modes_t *eth_mode){


			uint32_t reg;
			HAL_ETH_ReadPHYRegister(&heth, PHY_BSR, &reg);
			if(reg==0x782D){
				if (*eth_mode==wire_off){
				return wire_on;
				} else {
				return *eth_mode;
				}

				}
			if(reg!=0x782D){
				return wire_stop;
				}


	return *eth_mode;
};
/**
 * @brief  Handler principal de ETH
 *
 */
void Comms_Handler(struct netif *netif,char *payload)
{
#ifdef DEBUG_COMMS
	//printf("Envio Datos\n");
```

```
#endif

                struct pbuf * p_send = pbuf_alloc(PBUF_TRANSPORT, strlen(payload), PBUF_REF);
                char * pa = payload;
                p_send->payload = pa;
                udp_connect(comms_pcb, &bms_gui_comms.ip, bms_gui_comms.port);
                udp_sendto(comms_pcb, p_send , &bms_gui_comms.ip, bms_gui_comms.port);
                pbuf_free(p_send);
                udp_disconnect(comms_pcb);
}
/**
 * @brief  Inicializacion de Eth
 *
 */
void Comms_init(struct netif *netif)
{
        err_t err_comms;
        comms_pcb = udp_new();
        if (comms_pcb)
        {
                err_comms = udp_bind(comms_pcb,IP4_ADDR_ANY,bms_gui_comms.port);
                if(err_comms == ERR_OK)
                {
                        udp_recv(comms_pcb, Comms_callback, NULL);
                }
                else
                {
                        udp_remove(comms_pcb);
                }
        }
}
/**
 * @brief  Callback de recepcion
 *
 */
void Comms_callback(void *arg, struct udp_pcb *upcb, struct pbuf *p, const ip_addr_t *addr, u16_t port)
{
        if (strlen(p->payload) > 5) {
        char * pch;
        pch = strstr(p->payload,"$BMS_GUI");
        if(strncmp ( pch, "$BMS_GUI:TEMP_MODE,", 19)==0){
                pch =pch+19;
                if (strncmp ( pch, "0", 1)==0){
                        eth_temp_mode =celsius;
                        }
                if (strncmp ( pch, "1", 1)==0){
                        eth_temp_mode =volt;
                        }
        }
        }
        pbuf_free(p);
}
```

1.5.3. BMS_ETH_Comm.h

```
/*****************************************************************************
 *         2018, Alfonso Mareca Miralles, Eupla Racing Team                  *
 *****************************************************************************/

/**
 * @file     BMS_ETH_Comm.h
 * @author   Alfonso Mareca Miralles <alfonso@euplart.com>
 * @author           Header de los mensajes Eth
```

```
 * @date    30 Sep 2018
 *
 */
#ifndef BMS_ETH_COMM_H_
#define BMS_ETH_COMM_H_
#include "main.h"
#include "lwip.h"
#include "BMS_ETH_Core.h"
#define TEST_DATA_COMMS
#define DEBUG_COMMS
#define DEBUG_LTC
void vol_cells_comm(ltc_reg_t *bms_data,char * p_Send);
void temp_cells_comm(ltc_reg_t *bms_data,temp_send_mode_t *mode, char * p_Send);
void temp_bal_comm(ltc_reg_t *bms_data,temp_send_mode_t *mode, char * p_Send);
void general_data_comm(bms_general_data_t *bms_general_data,temp_send_mode_t *mode, char *
p_Send);
void bms_status_comm(status_t  *bms_status, char * p_Send);
void bms_op_mode_comm(op_modes_t *bms_mode, char * p_Send);
void send_conf_comm(ltc_conf_t *bms_conf,current_conf_t *current_s_conf, char * p_Send);

eth_modes_t eth_GUI_Comms(char *send_buffer,eth_modes_t *eth_mode,ltc_reg_t *bms_data,ltc_conf_t
*conf,current_conf_t *current_s_conf,status_t *bms_status,op_modes_t* bms_mode);

#endif /* BMS_ETH_CORE_H_ */
```

1.5.4. BMS_ETH_Comm.c

```
/*******************************************************************************
 *         2018, Alfonso Mareca Miralles, Eupla Racing Team                    *
 *******************************************************************************/

/**
 * @file    BMS_ETH_Comm.h
 * @author  Alfonso Mareca Miralles <alfonso@euplart.com>
 * @author          Mensajes ETH
 * @date    30 Sep 2018
 *
 */
#include "BMS_ETH_Comm.h"
#include "BMS_ETH_Core.h"
#include "main.h"
#include "../BMS_Libraries/delay.h"
#include "../BMS_Libraries/BMS_defs.h"
#include "lwip.h"
extern struct netif gnetif;

#include <string.h>
#include "lwip.h"
#include "DHCP_Client.h"
#include <string.h>

eth_modes_t eth_GUI_Comms(char *send_buffer, eth_modes_t *eth_mode, ltc_reg_t *bms_data, ltc_conf_t
*conf,current_conf_t *current_s_conf,status_t *bms_status, op_modes_t* bms_mode) {
        static uint8_t count = 0;

#ifdef TEST_DATA_COMMS
        //TEST_FAKE_DATA(&ltc_reg, &bms_general_data);
#endif
        if ((*eth_mode == gui_init) || (*eth_mode == gui)) {

                //printf("COMMS:Modo envio de datos\n");
                if (*eth_mode == gui_init) {
                        Comms_init(&gnetif);
```

```
                        *eth_mode = gui;
#ifdef DEBUG_COMMS
                        printf("init de la conexion\n\n");
#endif
                }
                //vol_cells_comm(send_buffer);
                strcpy(send_buffer, "");
                if (count == 70) {
                        count = 0;
                }
                if (count == 0) {
                        vol_cells_comm(&ltc_reg, send_buffer);
                }
                if (count == 10) {
                        temp_cells_comm(&ltc_reg, &eth_temp_mode, send_buffer);
                }
                if (count == 20) {
                        temp_bal_comm(&ltc_reg, &eth_temp_mode, send_buffer);
                }
                if (count == 30) {
                        general_data_comm(&bms_general_data, &eth_temp_mode, send_buffer);
                }
        /*      if (count == 40) {
                        bms_status_comm(bms_status,  send_buffer);
                }
                if (count == 50) {
                        bms_op_mode_comm(bms_mode, send_buffer);
                                }
                if (count == 60) {
                        send_conf_comm(conf,current_s_conf,send_buffer);
                                }*/
                Comms_Handler(&gnetif, send_buffer);
                count++;

        }

        return *eth_mode;
}

void vol_cells_comm(ltc_reg_t *bms_data, char * p_Send) {
        strcpy(p_Send, "$BMS_Core:V_Cells");
        for (int i = 0; i < 26; i++) {
                char temp[5];
                sprintf((char *) temp, "%u", bms_data->v_modules[i]);
                strcat(p_Send, ",");
                strcat(p_Send, temp);
        }
}

void temp_cells_comm(ltc_reg_t *bms_data, temp_send_mode_t *mode, char * p_Send) {
        strcpy(p_Send, "$BMS_Core:T_Cells");
        for (int i = 0; i < 24; i++) {
                char temp[5];
                if (*mode == volt) {
                        sprintf((char *) temp, "%u", bms_data->v_temp_cell[i]);
                        strcat(p_Send, ",");
                        strcat(p_Send, temp);
                } else {
                        sprintf((char *) temp, "%u", bms_data->temp_modules[i]);
                        strcat(p_Send, ",");
                        strcat(p_Send, temp);
                }

        }
}
```

```
void temp_bal_comm(ltc_reg_t *bms_data, temp_send_mode_t *mode, char * p_Send) {
	strcpy(p_Send, "$BMS_Core:T_Bal");
	for (int i = 0; i < 6; i++) {
		char temp[5];
		if (*mode == volt) {
			sprintf((char *) temp, "%u", bms_data->v_temp_bal[i]);
			strcat(p_Send, ",");
			strcat(p_Send, temp);
		} else {
			sprintf((char *) temp, "%u", bms_data->temp_bal[i]);
			strcat(p_Send, ",");
			strcat(p_Send, temp);
		}

	}
}

void general_data_comm(bms_general_data_t *bms_general_data, temp_send_mode_t *mode, char *
p_Send) {
	char temp[5];
	strcpy(p_Send, "$BMS_Core:G_Data");
	sprintf((char *) temp, "%u", bms_general_data->soc);
	strcat(p_Send, ",");
	strcat(p_Send, temp);
	sprintf((char *) temp, "%u", bms_general_data->soh);
	strcat(p_Send, ",");
	strcat(p_Send, temp);
	sprintf((char *) temp, "%u", bms_general_data->total_v);
	strcat(p_Send, ",");
	strcat(p_Send, temp);
	sprintf((char *) temp, "%u", bms_general_data->max_peak_c);
	strcat(p_Send, ",");
	strcat(p_Send, temp);
	sprintf((char *) temp, "%u", bms_general_data->max_temp_module);
	strcat(p_Send, ",");
	strcat(p_Send, temp);
	sprintf((char *) temp, "%u", bms_general_data->max_temp_bal);
	strcat(p_Send, ",");
	strcat(p_Send, temp);
	sprintf((char *) temp, "%u", bms_general_data->max_disbalance);
	strcat(p_Send, ",");
	strcat(p_Send, temp);

	for (int i = 0; i < 26; i++) {
		char temp[5];
		sprintf((char *) temp, "%u", bms_general_data->cell_balance[i]);
		strcat(p_Send, ",");
		strcat(p_Send, temp);
	}
}

void bms_status_comm(status_t *bms_status, char * p_Send) {
	char temp[1];
	strcpy(p_Send, "$BMS_Core:Flags");
	if (bms_status->error_uv == bms_ok) {
		sprintf((char *) temp, "%u", 0);
	} else {
		sprintf((char *) temp, "%u", 1);
	}
	strcat(p_Send, ",");
	strcat(p_Send, temp);

	if (bms_status->error_ov == bms_ok) {
		sprintf((char *) temp, "%u", 0);
```

```
		} else {
			sprintf((char *) temp, "%u", 1);
		}
		strcat(p_Send, ",");
		strcat(p_Send, temp);

		if (bms_status->error_oc == bms_ok) {
			sprintf((char *) temp, "%u", 0);
		} else {
			sprintf((char *) temp, "%u", 1);
		}
		strcat(p_Send, ",");
		strcat(p_Send, temp);

		if (bms_status->error_otM == bms_ok) {
			sprintf((char *) temp, "%u", 0);
		} else {
			sprintf((char *) temp, "%u", 1);
		}
		strcat(p_Send, ",");
		strcat(p_Send, temp);

		if (bms_status->error_otBal == bms_ok) {
			sprintf((char *) temp, "%u", 0);
		} else {
			sprintf((char *) temp, "%u", 1);
		}
		strcat(p_Send, ",");
		strcat(p_Send, temp);

		if (bms_status->ev_stop == bms_ok) {
			sprintf((char *) temp, "%u", 0);
		} else {
			sprintf((char *) temp, "%u", 1);
		}
		strcat(p_Send, ",");
		strcat(p_Send, temp);


}

void bms_op_mode_comm(op_modes_t *bms_mode, char * p_Send) {
		strcpy(p_Send, "$BMS_Core:Mode");
		char temp[1];
		if (*bms_mode == normal_op) {
				sprintf((char *) temp, "%u", 0);
			} else {
				sprintf((char *) temp, "%u", 1);
			}
			strcat(p_Send, ",");
			strcat(p_Send, temp);

			if (*bms_mode ==  charge_op) {
				sprintf((char *) temp, "%u", 0);
			} else {
				sprintf((char *) temp, "%u", 1);
			}
			strcat(p_Send, ",");
			strcat(p_Send, temp);

			if (*bms_mode ==  balance_op) {
				sprintf((char *) temp, "%u", 0);
			} else {
				sprintf((char *) temp, "%u", 1);
			}
```

```
                strcat(p_Send, ",");
                strcat(p_Send, temp);
                if (*bms_mode ==  log_op) {
                        sprintf((char *) temp, "%u", 0);
                } else {
                        sprintf((char *) temp, "%u", 1);
                }
                strcat(p_Send, ",");
                strcat(p_Send, temp);
}

void send_conf_comm(ltc_conf_t *bms_conf,current_conf_t *current_s_conf, char * p_Send) {
        strcpy(p_Send, "$BMS_Core:Config");
        char temp[5];
        sprintf((char *) temp, "%u", bms_conf->ic_num);
        strcat(p_Send, ",");
        strcat(p_Send, temp);
        sprintf((char *) temp, "%u", bms_conf->cell_ov);
        strcat(p_Send, ",");
        strcat(p_Send, temp);
        sprintf((char *) temp, "%u", bms_conf->cell_uv);
        strcat(p_Send, ",");
        strcat(p_Send, temp);
        sprintf((char *) temp, "%u", bms_conf->cell_ot);
        strcat(p_Send, ",");
        strcat(p_Send, temp);
        sprintf((char *) temp, "%u", bms_conf->bal_ot);
        strcat(p_Send, ",");
        strcat(p_Send, temp);
        sprintf((char *) temp, "%u", bms_conf->cell_margin);
        strcat(p_Send, ",");
        strcat(p_Send, temp);
        sprintf((char *) temp, "%u", bms_conf->t_fail_uv);
        strcat(p_Send, ",");
        strcat(p_Send, temp);
        sprintf((char *) temp, "%u", bms_conf->t_fail_ov);
        strcat(p_Send, ",");
        strcat(p_Send, temp);
        sprintf((char *) temp, "%u", current_s_conf->bat_oc);
        strcat(p_Send, ",");
        strcat(p_Send, temp);
        sprintf((char *) temp, "%u", current_s_conf->bat_uc);
        strcat(p_Send, ",");
        strcat(p_Send, temp);

}
```

1.5.5. Broadcast_ssdp.h

```
/*****************************************************************************
 *         2018, Alfonso Mareca Miralles, Eupla Racing Team                  *
 *****************************************************************************/

/**
 * @file     Bradcast_ssdp.h
 * @author   Alfonso Mareca Miralles <alfonso@euplart.com>
 * @author           Header del cliente SSDP de la interfaz
 * @date     30 Sep 2018
 *
 */
#include "main.h"
#include "lwip.h"
```

(Código fuente)

```
#include "../Network_Modules/BMS_ETH_Core.h"

#define UDP_SSDP_PORT    1900

char ssdpBuff[1024];


void Broadcast_ssdp_init(struct netif *netif);
void Broadcast_ssdp_callback(void *arg, struct udp_pcb *upcb, struct pbuf *p, const ip_addr_t *addr,
u16_t port);
void Broadcast_ssdp_Handler(struct netif *netif,eth_modes_t *eth_mode);
```

1.5.6. Broadcast_ssdp.c

```
/*******************************************************************************
 *         2018, Alfonso Mareca Miralles, Eupla Racing Team                    *
 *******************************************************************************/

/**
 * @file     Bradcast_ssdp.c
 * @author   Alfonso Mareca Miralles <alfonso@euplart.com>
 * @author           Cliente SSDP de la interfaz
 * @date     30 Sep 2018
 *
 */
#include "Broadcast_ssdp.h"
#include "lwip.h"
#include <string.h>
#include "BMS_ETH_Core.h"
#include "../BMS_Libraries/delay.h"

#define BMS_GUI_SERVICE_NAME "BMS_GUI"
static struct udp_pcb *ssdp_pcb;
extern struct netif gnetif;
/**
 * @brief Handler de la mensajeria ssdp
 */
void Broadcast_ssdp_Handler(struct netif *netif, eth_modes_t *eth_mode) {
        static uint32_t Temp_timer = 0;

        if (Temp_timer == 0) {
                Temp_timer = HAL_GetTick();
        }
        if (*eth_mode == wire_dhcp) {
#ifdef DEBUG_COMMS
                printf("-SSDP SIMPLE: Enviando Broadcast\n");
#endif
                Broadcast_ssdp_init(netif);
                *eth_mode = wire_search;

        }
        if ((HAL_GetTick() - Temp_timer > 1000) && (*eth_mode == wire_search)) {
#ifdef DEBUG_COMMS
                printf("-Enviado Broadcast\n");
#endif
                Temp_timer = 0;
                char dato_env[90];
                sprintf((char *) dato_env, "M-SEARCH *
HTTP/1.1\r\nHOST:%s:1900\r\nST:upnp:BMS_Core\r\nMAN:\"ssdp:discover\"\r\nMX:2\r\n",
ip4addr_ntoa((const ip4_addr_t *) &netif->ip_addr));
                struct pbuf * p_send = pbuf_alloc(PBUF_TRANSPORT, strlen(dato_env), PBUF_REF);
                p_send->payload = dato_env;
                udp_connect(ssdp_pcb, IP_ADDR_BROADCAST, UDP_SSDP_PORT);
```

```
                udp_sendto(ssdp_pcb, p_send, IP_ADDR_BROADCAST, UDP_SSDP_PORT);
                pbuf_free(p_send);
                udp_disconnect(ssdp_pcb);
        }

}
/**
 * @brief Inicializacion de la mensajeria ssdp
 */
void Broadcast_ssdp_init(struct netif *netif) {
        err_t err_ssdp;
        ssdp_pcb = udp_new();
        if (ssdp_pcb) {
#ifdef DEBUG_COMMS
                printf("-SSDP SIMPLE: Bind de red y callback\n");
#endif

                err_ssdp = udp_bind(ssdp_pcb, IP_ADDR_ANY, UDP_SSDP_PORT);
                if (err_ssdp == ERR_OK) {
                        udp_recv(ssdp_pcb, Broadcast_ssdp_callback, NULL);
                } else {
                        udp_remove(ssdp_pcb);
                }
        }
}
/**
 * @brief Callback de la mensajeria ssdp
 */
void Broadcast_ssdp_callback(void *arg, struct udp_pcb *upcb, struct pbuf *p, const ip_addr_t *addr,
u16_t port) {
        /* Cadena a Buscar:
         *      M-SEARCH *
HTTP/1.1\r\nHOST:123.456.789:1900\r\nST:upnp:BMS_GUI\r\nMAN:\"ssdp:discover\"\r\nMX:2\r\n
         */
        static uint8_t counter;
        if (strlen(p->payload) > 48) {
                char ip4_gui_buff[20] = "";
                char ip4_port_buff[5] = "ffff";
                char BMS_service_buff[8] = "ffffff";
                char * pch;
                pch = strstr(p->payload, "HOST");
                pch = strstr(pch, ":");
                pch++;
                for (int i = 0; i < 20; i++) {
                        if (*(pch + i) != ':') {
                                strncat(ip4_gui_buff, (pch + i), 1);
                        } else {
                                i = 20;
                        };
                }
                pch = strstr(pch, ":");
                pch++;
                strncpy(ip4_port_buff, pch, 4);
                pch = strstr(p->payload, "upnp:");
                strncpy(BMS_service_buff, pch + 5, 7);
                //printf("%s\n",(char *)pch);
                if (strncmp(BMS_service_buff, BMS_GUI_SERVICE_NAME, 7) == 0) {
                        if (counter == 0) {
                                bms_gui_comms.port = atoi(ip4_port_buff);
                                bms_gui_comms.ip = *addr;

                        }
                        counter++;
                }
                if (counter >= 2) {
```

```
                        pbuf_free(p);
                        udp_recv(ssdp_pcb, NULL, NULL);
                        udp_disconnect(ssdp_pcb);
                        udp_remove(ssdp_pcb);
                        eth_mode = gui_init;

                        counter = 0;
                }
        } else {
                pbuf_free(p);
                counter = 0;
        }

}
```

1.6. LIBRERÍA CAN (BMS_CANCOMMS)

Se expone el código fuente de la librería BMS_CanComms:

BMS_CanComms_Libraries		Librería de can:
CAN_msg.h	CAN_msg.c	Funciones de mensajes utilizados en CANbus.

1.6.1. CAN_msg.h

```
/*****************************************************************************
 *        2018, Alfonso Mareca Miralles, Eupla Racing Team                   *
 *****************************************************************************/

/**
* @file     CAN_msg.h
 * @author   Alfonso Mareca Miralles <alfonso@euplart.com>
 * @date     30 Sep 2018
 *
 * @brief    Header de los Mesajes usados en la red CAN
 */


#ifndef CAN_MSG_H_
#define C_MSG_H_

#include "main.h"
#include <stdio.h>
#include <stdlib.h>
#include "can.h"
#include "../BMS_Libraries/BMS_defs.h"

void BMS_Can_Comms(void);
cancomms_status_t CAN_Send_msg12(ltc_reg_t *ltc_reg, current_reg_t *current_s_reg,
bms_general_data_t *bms_general_data);
cancomms_status_t CAN_Send_msg11(ltc_reg_t *ltc_reg, current_reg_t *current_s_reg,
bms_general_data_t *bms_general_data);
cancomms_status_t CAN_Send_msg10(ltc_reg_t *ltc_reg, current_reg_t *current_s_reg,
bms_general_data_t *bms_general_data);
cancomms_status_t CAN_Send_msg9(ltc_reg_t *ltc_reg, current_reg_t *current_s_reg, bms_general_data_t
*bms_general_data);
cancomms_status_t CAN_Send_msg8(ltc_reg_t *ltc_reg, current_reg_t *current_s_reg, bms_general_data_t
*bms_general_data);
cancomms_status_t CAN_Send_msg7(ltc_reg_t *ltc_reg, current_reg_t *current_s_reg, bms_general_data_t
*bms_general_data);
```

```
cancomms_status_t CAN_Send_msg6(ltc_reg_t *ltc_reg, current_reg_t *current_s_reg, bms_general_data_t
*bms_general_data);
cancomms_status_t CAN_Send_msg5(ltc_reg_t *ltc_reg, current_reg_t *current_s_reg, bms_general_data_t
*bms_general_data);
cancomms_status_t CAN_Send_msg4(ltc_reg_t *ltc_reg, current_reg_t *current_s_reg, bms_general_data_t
*bms_general_data,status_t *status);
cancomms_status_t CAN_Send_msg3(ltc_reg_t *ltc_reg, current_reg_t *current_s_reg, bms_general_data_t
*bms_general_data,status_t *status);
cancomms_status_t CAN_Send_msg2(ltc_reg_t *ltc_reg, current_reg_t *current_s_reg, bms_general_data_t
*bms_general_data);
cancomms_status_t CAN_Send_msg1(ltc_reg_t *ltc_reg, current_reg_t *current_s_reg, bms_general_data_t
*bms_general_data);
cancomms_status_t CAN_Send_Alive(ltc_reg_t *ltc_reg, current_reg_t *current_s_reg, bms_general_data_t
*bms_general_data);

#endif /* CAN_DEFS_H_ */
```

1.6.2. CAN_msg.c

```
/*******************************************************************************
 *        2018, Alfonso Mareca Miralles, Eupla Racing Team              *
 *******************************************************************************/

/**
 * @file     CAN_msg.c
 * @author   Alfonso Mareca Miralles <alfonso@euplart.com>
 * @date     30 Sep 2018
 *
 * @brief    Mesajes usados en la red CAN
 */
#include "CAN_msg.h"
#include "can.h"
#include "../BMS_Libraries/BMS_defs.h"
uint32_t            TxMailbox;
/**
 * @brief Temporizacion del envio de mensajes
 */
void BMS_Can_Comms(void){
static     uint32_t timer_can = 0;
static     uint8_t can_counter = 0;
           if (timer_can == 0) {
                                      timer_can = HAL_GetTick();
                                  }
                                  if ((HAL_GetTick() - timer_can > 50) && (can_counter < 10)) {
                                      CAN_Send_Alive(&ltc_reg, &current_s_reg, &bms_general_data);
                                      CAN_Send_msg1(&ltc_reg, &current_s_reg, &bms_general_data);
                                      CAN_Send_msg2(&ltc_reg, &current_s_reg, &bms_general_data);
                                      //          printf("1 envio\n");
                                      timer_can = 0;
                                      can_counter++;
                                  }
                                  if ((HAL_GetTick() - timer_can > 50) && (can_counter < 20)) {
                                      CAN_Send_msg3(&ltc_reg, &current_s_reg, &bms_general_data,
&bms_status);
                                      CAN_Send_msg4(&ltc_reg, &current_s_reg, &bms_general_data,
&bms_status);
                                      //          printf("2 envio\n");
                                      can_counter++;
                                      timer_can = 0;
                                  }
                                  if ((HAL_GetTick() - timer_can > 50) && (can_counter < 30)) {
                                      CAN_Send_msg5(&ltc_reg, &current_s_reg, &bms_general_data);
                                      CAN_Send_msg6(&ltc_reg, &current_s_reg, &bms_general_data);
```

```
                        //          printf("3 envio\n");
                        can_counter++;
                        timer_can = 0;
                }
                if ((HAL_GetTick() - timer_can > 50) && (can_counter < 40)) {
                        CAN_Send_msg7(&ltc_reg, &current_s_reg, &bms_general_data);
                        CAN_Send_msg8(&ltc_reg, &current_s_reg, &bms_general_data);
                        //          printf("4 envio\n");
                        can_counter++;
                        timer_can = 0;
                }
                if ((HAL_GetTick() - timer_can > 50) && (can_counter < 50)) {
                        CAN_Send_msg9(&ltc_reg, &current_s_reg, &bms_general_data);
                        CAN_Send_msg10(&ltc_reg, &current_s_reg, &bms_general_data);
                        //          printf("5 envio\n");
                        can_counter++;
                        timer_can = 0;
                }
                if ((HAL_GetTick() - timer_can > 50) && (can_counter < 60)) {
                        CAN_Send_msg11(&ltc_reg, &current_s_reg, &bms_general_data);
                        CAN_Send_msg12(&ltc_reg, &current_s_reg, &bms_general_data);
                        CAN_Send_Alive(&ltc_reg, &current_s_reg, &bms_general_data);
                        can_counter++;
                        timer_can = 0;
                        //          printf("6 envio\n");
                }
                if (can_counter >= 59) {
                        can_counter = 0;
                }
}
/**
 * @brief Mensajes al cargador
 */
HAL_StatusTypeDef CAN_Charger_Stop(void) {
        static CAN_TxHeaderTypeDef TxHeader_can1;
        static uint8_t TxData_can1[1];
        TxHeader_can1.StdId = 0x11;
        TxHeader_can1.RTR = CAN_RTR_DATA;
        TxHeader_can1.IDE = CAN_ID_STD;
        TxHeader_can1.DLC = 2;
        TxHeader_can1.TransmitGlobalTime = DISABLE;
        TxData_can1[0] = 0xFF;
        if (HAL_CAN_AddTxMessage(&hcan2, &TxHeader_can1, TxData_can1, &TxMailbox) != HAL_OK)
{
        }
        return HAL_OK;
}
/**
 * @brief Mensajes al cargador
 */
HAL_StatusTypeDef CAN_Charger_Start(void) {
        static CAN_TxHeaderTypeDef TxHeader_can2;
        static uint8_t TxData_can2[1];
        TxHeader_can2.StdId = 0x11;
        TxHeader_can2.RTR = CAN_RTR_DATA;
        TxHeader_can2.IDE = CAN_ID_STD;
        TxHeader_can2.DLC = 2;
        TxHeader_can2.TransmitGlobalTime = DISABLE;
        TxData_can2[0] = 0xAA;
        if (HAL_CAN_AddTxMessage(&hcan2, &TxHeader_can2, TxData_can2, &TxMailbox) != HAL_OK)
{
        }
        while (HAL_CAN_GetTxMailboxesFreeLevel(&hcan2) != 3) {
        }
        return HAL_OK;
```

```
}
/**
 * @brief Mensajes al vehiculo 1
 */
cancomms_status_t CAN_Send_msg1(ltc_reg_t *ltc_reg, current_reg_t *current_s_reg, bms_general_data_t
*bms_general_data) {
	/*##-4- Start the Transmission process ###################################*/
	CAN_TxHeaderTypeDef   TxHeader_can1;
	uint8_t TxData_can1[6];
		TxHeader_can1.StdId = 0x1F1;
		TxHeader_can1.RTR = CAN_RTR_DATA;
		TxHeader_can1.IDE = CAN_ID_STD;
		TxHeader_can1.DLC = 6;
		TxHeader_can1.TransmitGlobalTime = DISABLE;
		TxData_can1[5] =(uint8_t)((bms_general_data->soh)&0x00FF);
		TxData_can1[4] =(uint8_t)((bms_general_data->soh>>8)&0x00FF);
		uint16_t vicen = 23456;
		TxData_can1[3] =(uint8_t) ((vicen)&0x00FF);
		TxData_can1[2] =(uint8_t) ((vicen >> 8)&0x00FF);
		TxData_can1[1] =(uint8_t) ((bms_general_data->soc)&0x00FF);
		TxData_can1[0] =(uint8_t) ((bms_general_data->soc >> 8)&0x00FF);

			if(HAL_CAN_AddTxMessage(&hcan1, &TxHeader_can1, TxData_can1,
&TxMailbox) != HAL_OK)
			{
				return sent;
			}
		return nosent;
}
/**
 * @brief Mensajes al vehiculo 2
 */
cancomms_status_t CAN_Send_msg2(ltc_reg_t *ltc_reg, current_reg_t *current_s_reg, bms_general_data_t
*bms_general_data) {
	/*##-4- Start the Transmission process ###################################*/
	CAN_TxHeaderTypeDef   TxHeader_can1;
	uint8_t TxData_can1[6];
	TxHeader_can1.StdId = 0x1F2;
	TxHeader_can1.RTR = CAN_RTR_DATA;
	TxHeader_can1.IDE = CAN_ID_STD;
	TxHeader_can1.DLC = 6;
	TxHeader_can1.TransmitGlobalTime = DISABLE;
	TxData_can1[5] =(uint8_t)((bms_general_data->max_peak_c)&0x00FF);
	TxData_can1[4] =(uint8_t)((bms_general_data->max_peak_c>>8)&0x00FF);
	TxData_can1[3] =(uint8_t)((bms_general_data->total_v)&0x00FF);
	TxData_can1[2] =(uint8_t)((bms_general_data->total_v>>8)&0x00FF);
	TxData_can1[1] =(uint8_t)(current_s_reg->amps&0x00FF);
	TxData_can1[0] =(uint8_t)((current_s_reg->amps >> 8)&0x00FF);
	if(HAL_CAN_AddTxMessage(&hcan1, &TxHeader_can1, TxData_can1, &TxMailbox) != HAL_OK)
	{
		return sent;
	}
	return nosent;

}
/**
 * @brief Mensajes al vehiculo 3
 */
cancomms_status_t CAN_Send_msg3(ltc_reg_t *ltc_reg, current_reg_t *current_s_reg, bms_general_data_t
*bms_general_data,status_t *status) {
	/*##-4- Start the Transmission process ###################################*/
	CAN_TxHeaderTypeDef   TxHeader_can1;
	uint8_t TxData_can1[1];
	TxHeader_can1.StdId = 0x1F3;
	TxHeader_can1.RTR = CAN_RTR_DATA;
```

```
		TxHeader_can1.IDE = CAN_ID_STD;
		TxHeader_can1.DLC = 1;
		TxHeader_can1.TransmitGlobalTime = DISABLE;
		TxData_can1[0]=0x00;
		if ((status->error_ov)||(status->error_uv)){
			TxData_can1[0]=TxData_can1[0] | 0b11110000;
		}
		if ((status->error_otBal)||(status->error_otM)){
			TxData_can1[0]=TxData_can1[0] | 0b00001100;
		}
		if ((status->error_otBal)||(status->error_otM)){
			TxData_can1[0]=TxData_can1[0] | 0b00000011;
		}
		if(HAL_CAN_AddTxMessage(&hcan1, &TxHeader_can1, TxData_can1, &TxMailbox) != HAL_OK)
		{
			return sent;
		}
		return nosent;

}
/**
 * @brief Mensajes al vehiculo 4
 */
cancomms_status_t CAN_Send_msg4(ltc_reg_t *ltc_reg, current_reg_t *current_s_reg, bms_general_data_t
*bms_general_data,status_t *status) {
		/*##-4- Start the Transmission process ###########################################*/
		CAN_TxHeaderTypeDef   TxHeader_can1;
		uint8_t TxData_can1[8];
		TxHeader_can1.StdId = 0x1F4;
		TxHeader_can1.RTR = CAN_RTR_DATA;
		TxHeader_can1.IDE = CAN_ID_STD;
		TxHeader_can1.DLC = 8;
		TxHeader_can1.TransmitGlobalTime = DISABLE;
		TxData_can1[7] =(uint8_t)(bms_general_data->max_temp_bal &0x00FF);
		TxData_can1[6] =(uint8_t)((bms_general_data->max_temp_bal >> 8)&0x00FF);
		TxData_can1[5] =(uint8_t)(bms_general_data->max_disbalance &0x00FF);
		TxData_can1[4] =(uint8_t)((bms_general_data->max_disbalance >> 8)&0x00FF);
		TxData_can1[3] =(uint8_t)(bms_general_data->max_v_cell&0x00FF);
		TxData_can1[2] =(uint8_t)((bms_general_data->max_v_cell >> 8)&0x00FF);
		TxData_can1[1] =(uint8_t)(bms_general_data->min_v_cell&0x00FF);
		TxData_can1[0] =(uint8_t)((bms_general_data->min_v_cell >> 8)&0x00FF);
		if(HAL_CAN_AddTxMessage(&hcan1, &TxHeader_can1, TxData_can1, &TxMailbox) != HAL_OK)
		{
			return sent;
		}
		return nosent;
}
/**
 * @brief Mensajes al vehiculo 5
 */
cancomms_status_t CAN_Send_msg5(ltc_reg_t *ltc_reg, current_reg_t *current_s_reg, bms_general_data_t
*bms_general_data) {
		/*##-4- Start the Transmission process ###########################################*/
		CAN_TxHeaderTypeDef   TxHeader_can1;
		uint8_t TxData_can1[8];
		TxHeader_can1.StdId = 0x135;
		TxHeader_can1.RTR = CAN_RTR_DATA;
		TxHeader_can1.IDE = CAN_ID_STD;
		TxHeader_can1.DLC = 8;
		TxHeader_can1.TransmitGlobalTime = DISABLE;
		TxData_can1[7] =(uint8_t)((ltc_reg->v_modules[3])&0x00FF);
		TxData_can1[6] =(uint8_t)((ltc_reg->v_modules[3]>>8)&0x00FF);
		TxData_can1[5] =(uint8_t)((ltc_reg->v_modules[2])&0x00FF);
		TxData_can1[4] =(uint8_t)((ltc_reg->v_modules[2]>>8)&0x00FF);
		TxData_can1[3] =(uint8_t)(ltc_reg->v_modules[1]&0x00FF);
```

```
		TxData_can1[2] =(uint8_t)((ltc_reg->v_modules[1] >> 8)&0x00FF);
		TxData_can1[1] =(uint8_t)(ltc_reg->v_modules[0]&0x00FF);
		TxData_can1[0] =(uint8_t)((ltc_reg->v_modules[0] >> 8)&0x00FF);
		if(HAL_CAN_AddTxMessage(&hcan1, &TxHeader_can1, TxData_can1, &TxMailbox) != HAL_OK)
		{
			return sent;
		}
		return nosent;
}
/**
 * @brief Mensajes al vehiculo 6
 */
cancomms_status_t CAN_Send_msg6(ltc_reg_t *ltc_reg, current_reg_t *current_s_reg, bms_general_data_t
*bms_general_data) {
		/*##-4- Start the Transmission process ###########################################*/
		CAN_TxHeaderTypeDef   TxHeader_can1;
		uint8_t TxData_can1[8];
		TxHeader_can1.StdId = 0x136;
		TxHeader_can1.RTR = CAN_RTR_DATA;
		TxHeader_can1.IDE = CAN_ID_STD;
		TxHeader_can1.DLC = 8;
		TxHeader_can1.TransmitGlobalTime = DISABLE;
		TxData_can1[7] =(uint8_t)((ltc_reg->v_modules[7])&0x00FF);
		TxData_can1[6] =(uint8_t)((ltc_reg->v_modules[7]>>8)&0x00FF);
		TxData_can1[5] =(uint8_t)((ltc_reg->v_modules[6])&0x00FF);
		TxData_can1[4] =(uint8_t)((ltc_reg->v_modules[6]>>8)&0x00FF);
		TxData_can1[3] =(uint8_t)(ltc_reg->v_modules[5]&0x00FF);
		TxData_can1[2] =(uint8_t)((ltc_reg->v_modules[5] >> 8)&0x00FF);
		TxData_can1[1] =(uint8_t)(ltc_reg->v_modules[4]&0x00FF);
		TxData_can1[0] =(uint8_t)((ltc_reg->v_modules[4] >> 8)&0x00FF);
		if(HAL_CAN_AddTxMessage(&hcan1, &TxHeader_can1, TxData_can1, &TxMailbox) != HAL_OK)
		{
			return sent;
		}
		return nosent;
}
/**
 * @brief Mensajes al vehiculo 7
 */
cancomms_status_t CAN_Send_msg7(ltc_reg_t *ltc_reg, current_reg_t *current_s_reg, bms_general_data_t
*bms_general_data) {
		/*##-4- Start the Transmission process ###########################################*/
		CAN_TxHeaderTypeDef   TxHeader_can1;
		uint8_t TxData_can1[8];
		TxHeader_can1.StdId = 0x137;
		TxHeader_can1.RTR = CAN_RTR_DATA;
		TxHeader_can1.IDE = CAN_ID_STD;
		TxHeader_can1.DLC = 8;
		TxHeader_can1.TransmitGlobalTime = DISABLE;
		TxData_can1[7] =(uint8_t)((ltc_reg->v_modules[11])&0x00FF);
		TxData_can1[6] =(uint8_t)((ltc_reg->v_modules[11]>>8)&0x00FF);
		TxData_can1[5] =(uint8_t)((ltc_reg->v_modules[10])&0x00FF);
		TxData_can1[4] =(uint8_t)((ltc_reg->v_modules[10]>>8)&0x00FF);
		TxData_can1[3] =(uint8_t)(ltc_reg->v_modules[9]&0x00FF);
		TxData_can1[2] =(uint8_t)((ltc_reg->v_modules[9] >> 8)&0x00FF);
		TxData_can1[1] =(uint8_t)(ltc_reg->v_modules[8]&0x00FF);
		TxData_can1[0] =(uint8_t)((ltc_reg->v_modules[8] >> 8)&0x00FF);
		if(HAL_CAN_AddTxMessage(&hcan1, &TxHeader_can1, TxData_can1, &TxMailbox) != HAL_OK)
		{
			return sent;
		}
		return nosent;
}
/**
 * @brief Mensajes al vehiculo 8
```

```
 */
cancomms_status_t CAN_Send_msg8(ltc_reg_t *ltc_reg, current_reg_t *current_s_reg, bms_general_data_t
*bms_general_data) {
	/*##-4- Start the Transmission process ###################################*/
	CAN_TxHeaderTypeDef   TxHeader_can1;
	uint8_t TxData_can1[8];
	TxHeader_can1.StdId = 0x138;
	TxHeader_can1.RTR = CAN_RTR_DATA;
	TxHeader_can1.IDE = CAN_ID_STD;
	TxHeader_can1.DLC = 8;
	TxHeader_can1.TransmitGlobalTime = DISABLE;
	TxData_can1[7] =(uint8_t)((ltc_reg->v_modules[15])&0x00FF);
	TxData_can1[6] =(uint8_t)((ltc_reg->v_modules[15]>>8)&0x00FF);
	TxData_can1[5] =(uint8_t)((ltc_reg->v_modules[14])&0x00FF);
	TxData_can1[4] =(uint8_t)((ltc_reg->v_modules[14]>>8)&0x00FF);
	TxData_can1[3] =(uint8_t)(ltc_reg->v_modules[13]&0x00FF);
	TxData_can1[2] =(uint8_t)((ltc_reg->v_modules[13] >> 8)&0x00FF);
	TxData_can1[1] =(uint8_t)(ltc_reg->v_modules[12]&0x00FF);
	TxData_can1[0] =(uint8_t)((ltc_reg->v_modules[12] >> 8)&0x00FF);
	if(HAL_CAN_AddTxMessage(&hcan1, &TxHeader_can1, TxData_can1, &TxMailbox) != HAL_OK)
	{
		return sent;
	}
	return nosent;
}
/**
 * @brief Mensajes al vehiculo 9
 */
cancomms_status_t CAN_Send_msg9(ltc_reg_t *ltc_reg, current_reg_t *current_s_reg, bms_general_data_t
*bms_general_data) {
	/*##-4- Start the Transmission process ###################################*/
	CAN_TxHeaderTypeDef   TxHeader_can1;
	uint8_t TxData_can1[8];
	TxHeader_can1.StdId = 0x139;
	TxHeader_can1.RTR = CAN_RTR_DATA;
	TxHeader_can1.IDE = CAN_ID_STD;
	TxHeader_can1.DLC = 8;
	TxHeader_can1.TransmitGlobalTime = DISABLE;
	TxData_can1[7] =(uint8_t)((ltc_reg->v_modules[19])&0x00FF);
	TxData_can1[6] =(uint8_t)((ltc_reg->v_modules[19]>>8)&0x00FF);
	TxData_can1[5] =(uint8_t)((ltc_reg->v_modules[18])&0x00FF);
	TxData_can1[4] =(uint8_t)((ltc_reg->v_modules[18]>>8)&0x00FF);
	TxData_can1[3] =(uint8_t)(ltc_reg->v_modules[17]&0x00FF);
	TxData_can1[2] =(uint8_t)((ltc_reg->v_modules[17] >> 8)&0x00FF);
	TxData_can1[1] =(uint8_t)(ltc_reg->v_modules[16]&0x00FF);
	TxData_can1[0] =(uint8_t)((ltc_reg->v_modules[16] >> 8)&0x00FF);
	if(HAL_CAN_AddTxMessage(&hcan1, &TxHeader_can1, TxData_can1, &TxMailbox) != HAL_OK)
	{
		return sent;
	}
	return nosent;
}
/**
 * @brief Mensajes al vehiculo 10
 */
cancomms_status_t CAN_Send_msg10(ltc_reg_t *ltc_reg, current_reg_t *current_s_reg,
bms_general_data_t *bms_general_data) {
	/*##-4- Start the Transmission process ###################################*/
	CAN_TxHeaderTypeDef   TxHeader_can1;
	uint8_t TxData_can1[8];
	TxHeader_can1.StdId = 0x13A;
	TxHeader_can1.RTR = CAN_RTR_DATA;
	TxHeader_can1.IDE = CAN_ID_STD;
	TxHeader_can1.DLC = 8;
	TxHeader_can1.TransmitGlobalTime = DISABLE;
```

```
		TxData_can1[7] =(uint8_t)((ltc_reg->v_modules[23])&0x00FF);
		TxData_can1[6] =(uint8_t)((ltc_reg->v_modules[23]>>8)&0x00FF);
		TxData_can1[5] =(uint8_t)((ltc_reg->v_modules[22])&0x00FF);
		TxData_can1[4] =(uint8_t)((ltc_reg->v_modules[22]>>8)&0x00FF);
		TxData_can1[3] =(uint8_t)(ltc_reg->v_modules[21]&0x00FF);
		TxData_can1[2] =(uint8_t)((ltc_reg->v_modules[21] >> 8)&0x00FF);
		TxData_can1[1] =(uint8_t)(ltc_reg->v_modules[20]&0x00FF);
		TxData_can1[0] =(uint8_t)((ltc_reg->v_modules[20] >> 8)&0x00FF);
		if(HAL_CAN_AddTxMessage(&hcan1, &TxHeader_can1, TxData_can1, &TxMailbox) != HAL_OK)
		{
			return sent;
		}
		return nosent;
}
/**
 * @brief Mensajes al vehiculo 11
 */
cancomms_status_t CAN_Send_msg11(ltc_reg_t *ltc_reg, current_reg_t *current_s_reg,
bms_general_data_t *bms_general_data) {
		/*##-4- Start the Transmission process ###########################################*/
		CAN_TxHeaderTypeDef   TxHeader_can1;
		uint8_t TxData_can1[4];
		TxHeader_can1.StdId = 0x13B;
		TxHeader_can1.RTR = CAN_RTR_DATA;
		TxHeader_can1.IDE = CAN_ID_STD;
		TxHeader_can1.DLC = 8;
		TxHeader_can1.TransmitGlobalTime = DISABLE;
		TxData_can1[3] =(uint8_t)(ltc_reg->v_modules[25]&0x00FF);
		TxData_can1[2] =(uint8_t)((ltc_reg->v_modules[25] >> 8)&0x00FF);
		TxData_can1[1] =(uint8_t)(ltc_reg->v_modules[24]&0x00FF);
		TxData_can1[0] =(uint8_t)((ltc_reg->v_modules[24] >> 8)&0x00FF);
		if(HAL_CAN_AddTxMessage(&hcan1, &TxHeader_can1, TxData_can1, &TxMailbox) != HAL_OK)
		{
			return sent;
		}
		return nosent;
}
/**
 * @brief Mensajes al vehiculo 12
 */
cancomms_status_t CAN_Send_msg12(ltc_reg_t *ltc_reg, current_reg_t *current_s_reg,
bms_general_data_t *bms_general_data) {
		/*##-4- Start the Transmission process ###########################################*/
		CAN_TxHeaderTypeDef   TxHeader_can1;
		uint8_t TxData_can1[6];
		TxHeader_can1.StdId = 0x1FC;
		TxHeader_can1.RTR = CAN_RTR_DATA;
		TxHeader_can1.IDE = CAN_ID_STD;
		TxHeader_can1.DLC = 6;
		TxHeader_can1.TransmitGlobalTime = DISABLE;
		TxData_can1[5] =0x00;
		TxData_can1[4] =0x00;
		TxData_can1[3] =(uint8_t)(bms_general_data->max_temp_module &0x00FF);
		TxData_can1[2] =(uint8_t)((bms_general_data->max_temp_module >> 8)&0x00FF);
		TxData_can1[1] =(uint8_t)(bms_general_data->temp_average&0x00FF);
		TxData_can1[0] =(uint8_t)((bms_general_data->temp_average >> 8))&0x00FF;
		if(HAL_CAN_AddTxMessage(&hcan1, &TxHeader_can1, TxData_can1, &TxMailbox) != HAL_OK)
		{
			return sent;
		}
		return nosent;
}
/**
 * @brief Mensajes al vehiculo 13
 */
```

(Código fuente)

```
cancomms_status_t CAN_Send_Alive(ltc_reg_t *ltc_reg, current_reg_t *current_s_reg, bms_general_data_t
*bms_general_data) {
	/*##-4- Start the Transmission process ###########################################*/
	CAN_TxHeaderTypeDef   TxHeader_can1;
	uint8_t TxData_can1[1];
	TxHeader_can1.StdId = 0x1FF;
	TxHeader_can1.RTR = CAN_RTR_DATA;
	TxHeader_can1.IDE = CAN_ID_STD;
	TxHeader_can1.DLC = 6;
	TxHeader_can1.TransmitGlobalTime = DISABLE;
	TxData_can1[0] =0xFF;
	if(HAL_CAN_AddTxMessage(&hcan1, &TxHeader_can1, TxData_can1, &TxMailbox) != HAL_OK)
	{
		return sent;
	}
	return nosent;

}
```

ANEXO 2. (LISTADOS BOM DE LAS PLACAS)

A continucacion se detallan los componentes usados para cada placa

2.1. ESCLAVO

Componente	Referencia en placa	Valor/modelo	Huella	Cantidad en la pcb
LTC6255CS6	B1 B2 B3	LTC6255CS6	TSOT-23-6	3
Condensador	C59 C60 C65 C66 C70 C71	100nF	0603	6
Condensador	C61	100pF	0603	1
Condensador	C27 C28 C29 C30 C31 C32 C33 C34 C35 C36 C37 C38 C39 C40 C41 C42 C43 C44 C45 C46 C47 C48 C49 C50 C51 C52 C54 C56 C58 C86 C104 C119	10nF	0603	32
Condensador	C62 C63 C64 C67 C68 C69 C72 C73 C74 C75 C76 C77 C78 C79 C80 C81 C82 C83 C84 C85 C87 C88 C89 C90 C91 C92 C93 C94 C95 C96 C97 C98 C99 C100 C101 C102 C103 C105 C106 C107 C108 C109 C110 C111 C112 C113 C114 C115 C116 C117 C118	1uF	0603	51
Conector 1x02	Comm1 Comm2	Micro-Fit 43650-0212	43650-0212	2
Fusible	F1 F2 F3 F4 F5 F6 F7 F8 F9 F10 F11 F12 F13 F14 F15 F16 F17 F18 F19 F20 F21 F22 F23 F24 F25 F26 F27	2.5A	0603	27
Conector 2x04	J1 J2 J3 J4 J5 J6	43045-0811	43045-0811	6
Conector 1x09	J_Monitor_1-9 J_Monitor_9-18 J_Monitor_18-27	Micro-Fit 43650-0912	43650-0912	3
Led	LED1 LED2 LED3 LED4 LED5 LED6 LED7 LED8 LED9 LED10 LED11 LED12 LED13 LED14 LED15 LED16 LED17 LED18 LED19 LED20 LED21 LED22 LED23 LED24 LED25 LED26	50mA	0603	26
Transistor	Q27 Q28 Q29	NSV1C201MZ4	SOT-223	3
Mosfet	Q1 Q2 Q3 Q4 Q5 Q6 Q7 Q8 Q9 Q10 Q11 Q12 Q13 Q14 Q15 Q16 Q17 Q18 Q19 Q20 Q21 Q22 Q23 Q24 Q25 Q26	Si3493DDV	TSOP-6	26
Resistencia	R136 R137 R138 R154 R155 R156 R170 R171 R172	0R	0603	9
Resistencia	R79 R80 R81 R82 R83 R84 R85 R86 R87 R88 R89 R90 R91 R92 R93 R94 R95 R96 R97 R98 R99 R100 R101 R102 R103 R104 R105 R106 R107 R108 R109 R113 R114 R117 R118 R139 R158 R173	100R	0603	38
Resistencia	R126 R127 R128 R129 R130 R131 R132 R133 R140 R141 R142 R143 R144 R145 R146 R147 R148 R149 R152 R153 R157 R159 R160 R161 R162 R163 R164 R165 R166 R167	100k	0603	30
Resistencia	R1 R1P1 R2 R2P1 R3 R3P1 R4 R4P1 R5 R5P1 R6 R6P1 R7 R7P1 R8 R8P1 R9 R9P1 R10 R10P1 R11 R11P1 R12 R12P1 R13 R13P1 R14 R14P1 R15 R15P1 R16 R16P1 R17 R17P1 R18 R18P1 R19 R19P1 R20 R20P1 R21 R21P1 R22 R22P1 R23 R23P1 R24 R24P1 R25 R25P1 R26 R26P1	10R	2512	52
Resistencia	R111 R121 R122 R123 R124 R125	120R	0603	6
Resistencia	R110 R112 R115 R116 R119 R120	1K	0603	6
Resistencia	R134 R135 R150 R151 R168 R169	1k2	0603	6
Resistencia	R27 R28 R29 R30 R31 R32 R33 R34 R35 R36 R37 R38 R39 R40 R41 R42 R43 R44 R45 R46 R47 R48 R49 R50 R51 R52	200R	0603	26
Resistencia	R53 R54 R55 R56 R57 R58 R59 R60 R61 R62 R63 R64 R65 R66 R67 R68 R69 R70 R71 R72 R73 R74 R75 R76 R77 R78	3K3	0603	26
SM13072APEL	T1	SM13072APEL	SM13072	1
Resistencia	TH1 TH2 TH3 TH4 TH5 TH6	100K	0805	6
ADG728	U7 U8 U9	ADG728	TSSOP-16	3
HM2101NL	U4 U5 U6	HM2101NL	HM2101NL	3
LTC6804-1	U1 U2 U3	LTC6804-1	LTC6804-1	3

2.2. SENSOR DE CORRIENTE

Componente	Referencia en la placa	Valor/modelo	Huella	Cantidad en la pcb
Condensador	C5 C6 C9 C10	0.1nF	0603	4
Condensador	C7	0.1uF	0603	1
Condensador	C4	10nF	0603	1
Condensador	C12	10uF	0603	1
Condensador	C11 C13	1uF	0603	2
Condensador	C8	22uF	0603	1
Condensador	C1 C2 C3	47nF	0603	3
Sensor Hall	HTFS1	HTFS400-P_SP2	HTFS	1
Conector 1x4	J1	Molex Microfit 43650-0412	43650-0412	1
Jumper	JP1 JP2	NO_ISOLATED_SPI	0603	2
Resistencia	R1 R2 R5 R6	0R	0603	4
Resistencia	R3 R4	2k2	0603	2
Resistencia	R7 R8	330R	0603	2
ADC	U2	MCP3421A0T-ECH	SOT-23-6	1
I2C Buffer	U1	P82B96TD	SO-8	1

2.1. MAESTRO

Componente	References	Valor/modelo	Huella	Cantidad en la pcb
Condensador	C1 C3 C6 C13 C18	0.1uF	0603	5
Condensador	C17 C19	100pF	0603	2
Condensador	C21 C22	10uF	0603	2
Condensador	C7 C8 C9 C10	1nF	0603	4
Condensador	C14 C15 C16	1uF	0603	3
Condensador	C2 C5	220uF	0603	2
Condensador	C11 C12	4.7nF	0603	2
Condensador	C4	47nF	0603	1
Diodo	D4	CD0603-B0240R	0603	1
Transceiver CAN	D2 D3	ESDCAN24-2BLY	SOT-23	2
TVS	D1	SM15T30CAY	0805	1
Filtro	FL1 FL2	ACT45B-220	ACT45B	2
Conector 1x04	Micro-Fit 43650-0412	43650-0412	43650-0412	3
Conector 1x02	Micro-Fit 43650-0212	Micro-Fit 43650-0212	43650-0212	2
Jumper	JP7	0R	0603	1
Jumper	JP8	0R	0603	1
Jumper	JP5 JP6	0R	0603	2
Jumper	JP1 JP2 JP3 JP4	0R	0603	4
Bobina	L1	1.5 uH	0603	1
Resistencia	R11 R12 R13 R14 R17 R18 R19 R20 R_GND1 R_RC1 R_TRM1	0R	0603	11
Resistencia	R2	120R	0603	1
Resistencia	R1 R5 R10 R15 R16	1K	0603	5
Resistencia	R21 R22	2K	0603	2
Resistencia	R3 R4	330R	0603	2
Resistencia	R6 R7 R8 R9	60R	0603	4
SM13072APEL	T1	SM13072APEL	SM13072APEL	1
DC/DC	U3	CC-6-2405SR-E	CC-6-2405SR-E	1
Rele	U5	IM01TS	IM01TS	1

Diodo ideal	U2	LTC4359-MS8	MSOP-8	1
Transceptor isoSPI	U6	LTC6820	MSOP-16	1
Placa de desarrollo	U9	NUCLEO-144 F429ZI	Nucleo-144	1
Buffer i2C	U4	P82B96TD	SO-8	1
Mosfet	U1	SI4420DYPBF	SO-8	1
Transceptor Can	U7 U8	SN65HVD234	SOIC-8	2
Regulador Lineal	U10	tlv1117lv	SOT223	1

ANEXO 3. (DOCUMENTACIÓN TÉCNICA)

A continuación se agregará la docmentacion tecnica de los principales componentes usados.

3.1. LTC6084-1

LINEAR TECHNOLOGY

LTC6804-1/LTC6804-2

Multicell Battery Monitors

FEATURES

- **Measures Up to 12 Battery Cells in Series**
- **Stackable Architecture Supports 100s of Cells**
- **Built-In isoSPI™ Interface:**
  - **1Mbps Isolated Serial Communications**
  - **Uses a Single Twisted Pair, Up to 100 Meters**
  - **Low EMI Susceptibility and Emissions**
- **1.2mV Maximum Total Measurement Error**
- **290µs to Measure All Cells in a System**
- Synchronized Voltage and Current Measurement
- 16-Bit Delta-Sigma ADC with Frequency Programmable 3rd Order Noise Filter
- Engineered for ISO26262 Compliant Systems
- Passive Cell Balancing with Programmable Timer
- 5 General Purpose Digital I/O or Analog Inputs:
  - Temperature or other Sensor Inputs
  - Configurable as an I^2C or SPI Master
- 4µA Sleep Mode Supply Current
- 48-Lead SSOP Package

APPLICATIONS

- Electric and Hybrid Electric Vehicles
- Backup Battery Systems
- Grid Energy Storage
- High Power Portable Equipment

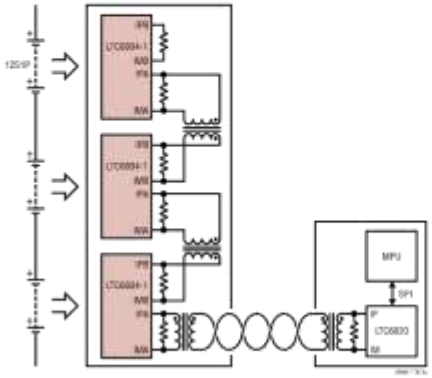
DESCRIPTION

The LTC®6804 is a 3rd generation multicell battery stack monitor that measures up to 12 series connected battery cells with a total measurement error of less than 1.2mV. The cell measurement range of 0V to 5V makes the LTC6804 suitable for most battery chemistries. All 12 cell voltages can be captured in 290µs, and lower data acquisition rates can be selected for high noise reduction.

Multiple LTC6804 devices can be connected in series, permitting simultaneous cell monitoring of long, high voltage battery strings. Each LTC6804 has an isoSPI interface for high speed, RF-immune, local area communications. Using the LTC6804-1, multiple devices are connected in a daisy-chain with one host processor connection for all devices. Using the LTC6804-2, multiple devices are connected in parallel to the host processor, with each device individually addressed.

Additional features include passive balancing for each cell, an onboard 5V regulator, and 5 general purpose I/O lines. In sleep mode, current consumption is reduced to 4µA. The LTC6804 can be powered directly from the battery, or from an isolated supply.

TYPICAL APPLICATION



Total Measurement Error vs Temperature of 5 Typical Units

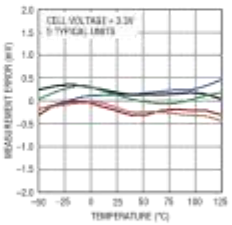


TABLE OF CONTENTS

ABSOLUTE MAXIMUM RATINGS (Note 1)

Total Supply Voltage V^+ to V^- 75V
Input Voltage (Relative to V^-)
- C0 ..... –0.3V to 0.3V
- C12 ..... –0.3V to 75V
- C(n) ..... –0.3V to MIN (8 • n, 75V)
- S(n) ..... –0.3V to MIN (8 • n, 75V)
- IPA, IMA, IPB, IMB ..... –0.3V to V_{REG} + 0.3V
- DRIVE Pin ..... –0.3V to 7V
- All Other Pins ..... –0.3V to 6V

Voltage Between Inputs
- V^+ to C12 ..... –5.5V
- C(n) to C(n – 1) ..... –0.3V to 8V
- S(n) to C(n – 1) ..... –0.3V to 8V
- C12 to C8 ..... –0.3V to 25V
- C8 to C4 ..... –0.3V to 25V
- C4 to C0 ..... –0.3V to 25V

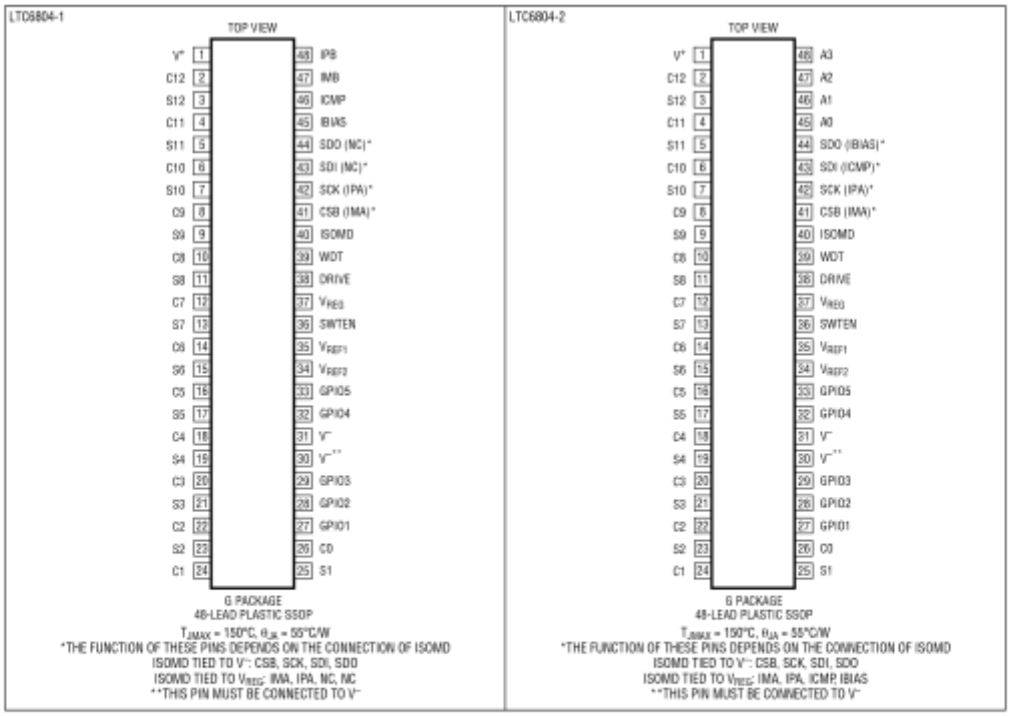
Current In/Out of Pins
- All Pins Except V_{REG}, IPA, IMA, IPB, IMB ..10mA
- IPA, IMA, IPB, IMB 30mA

Operating Temperature Range
- LTC6804I ..... –40°C to 85°C
- LTC6804H ..... –40°C to 125°C

Specified Temperature Range
- LTC6804I ..... –40°C to 85°C
- LTC6804H ..... –40°C to 125°C

Junction Temperature ..... 150°C
Storage Temperature ..... –65°C to 150°C
Lead Temperature (Soldering, 10sec) ..... 300°C

PIN CONFIGURATION



LTC6804-1/LTC6804-2

ORDER INFORMATION http://www.linear.com/product/LTC6804-1#orderinfo

TUBE	TAPE AND REEL	PART MARKING*	PACKAGE DESCRIPTION	SPECIFIED TEMPERATURE RANGE
LTC6804IG-1#PBF	LTC6804IG-1#TRPBF	LTC6804G-1	48-Lead Plastic SSOP	–40°C to 85°C
LTC6804HG-1#PBF	LTC6804HG-1#TRPBF	LTC6804G-1	48-Lead Plastic SSOP	–40°C to 125°C
LTC6804IG-2#PBF	LTC6804IG-2#TRPBF	LTC6804G-2	48-Lead Plastic SSOP	–40°C to 85°C
LTC6804HG-2#PBF	LTC6804HG-2#TRPBF	LTC6804G-2	48-Lead Plastic SSOP	–40°C to 125°C

Consult LTC Marketing for parts specified with wider operating temperature ranges. *The temperature grade is identified by a label on the shipping container. Parts ending with PBF are RoHS and WEEE compliant.

For more information on lead free part marking, go to: http://www.linear.com/leadfree/
For more information on tape and reel specifications, go to: http://www.linear.com/tapeandreel/. Some packages are available in 500 unit reels through designated sales channels with #TRMPBF suffix.

ELECTRICAL CHARACTERISTICS

The ● denotes the specifications which apply over the full operating temperature range, otherwise specifications are at T_A = 25°C. The test conditions are V^+ = 39.6V, V_{REG} = 5.0V unless otherwise noted.

SYMBOL	PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
ADC DC Specifications							
	Measurement Resolution		●		0.1		mV/bit
	ADC Offset Voltage	(Note 2)	●		0.1		mV
	ADC Gain Error	(Note 2)			0.01		%
			●		0.02		%
	Total Measurement Error (TME) in Normal Mode	C(n) to C(n – 1), GPIO(n) to V^- = 0			±0.2		mV
		C(n) to C(n – 1) = 2.0			±0.1	±0.8	mV
		C(n) to C(n – 1), GPIO(n) to V^- = 2.0	●			±1.4	mV
		C(n) to C(n – 1) = 3.3			±0.2	±1.2	mV
		C(n) to C(n – 1), GPIO(n) to V^- = 3.3	●			±2.2	mV
		C(n) to C(n – 1) = 4.2			±0.3	±1.6	mV
		C(n) to C(n – 1), GPIO(n) to V^- = 4.2	●			±2.8	mV
		C(n) to C(n – 1), GPIO(n) to V^- = 5.0			±1		mV
		Sum of Cells, V(CO) = V^-	●		±0.2	±0.75	%
		Internal Temperature, T = Maximum Specified Temperature			±5		°C
		V_{REG} Pin	●		±0.1	±0.25	%
		V_{REF2} Pin	●		±0.02	±0.1	%
		Digital Supply Voltage V_{REGD}	●		±0.1	±1	%

6804121c

ELECTRICAL CHARACTERISTICS

The ● denotes the specifications which apply over the full operating temperature range, otherwise specifications are at T_A = 25°C. The test conditions are V^+ = 39.6V, V_{REG} = 5.0V unless otherwise noted.

SYMBOL	PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
	Total Measurement Error (TME) in Filtered Mode	C(n) to C(n – 1), GPIO(n) to V^- = 0			±0.1		mV
		C(n) to C(n – 1) = 2.0			±0.1	±0.8	mV
		C(n) to C(n – 1), GPIO(n) to V^- = 2.0	●			±1.4	mV
		C(n) to C(n – 1) = 3.3			±0.2	±1.2	mV
		C(n) to C(n – 1), GPIO(n) to V^- = 3.3	●			±2.2	mV
		C(n) to C(n – 1) = 4.2			±0.3	±1.6	mV
		C(n) to C(n – 1), GPIO(n) to V^- = 4.2	●			±2.8	mV
		C(n) to C(n – 1), GPIO(n) to V^- = 5.0			±1		mV
		Sum of Cells, V(C0) = V^-	●		±0.2	±0.75	%
		Internal Temperature, T = Maximum Specified Temperature			±5		°C
		V_{REG} Pin	●		±0.1	±0.25	%
		V_{REF2} Pin	●		±0.02	±0.1	%
		Digital Supply Voltage V_{REGD}	●		±0.1	±1	%
	Total Measurement Error (TME) in Fast Mode	C(n) to C(n – 1), GPIO(n) to V^- = 0			±2		mV
		C(n) to C(n – 1), GPIO(n) to V^- = 2.0	●			±4	mV
		C(n) to C(n – 1), GPIO(n) to V^- = 3.3	●			±4.7	mV
		C(n) to C(n – 1), GPIO(n) to V^- = 4.2	●			±8.3	mV
		C(n) to C(n – 1), GPIO(n) to V^- = 5.0			±10		mV
		Sum of Cells, V(C0) = V^-	●		±0.3	±1	%
		Internal Temperature, T = Maximum Specified Temperature			±5		°C
		V_{REG} Pin	●		±0.3	±1	%
		V_{REF2} Pin	●		±0.1	±0.25	%
		Digital Supply Voltage V_{REGD}	●		±0.2	±2	%
	Input Range	C(n), n = 1 to 12	●	C(n – 1)		C(n – 1) + 5	V
		C0	●		0		
		GPIO(n), n = 1 to 5	●	0		5	V
I_L	Input Leakage Current When Inputs Are Not Being Measured	C(n), n = 0 to 12	●		10	±250	nA
		GPIO(n), n = 1 to 5	●		10	±250	nA
	Input Current When Inputs Are Being Measured	C(n), n = 0 to 12			±2		µA
		GPIO(n), n = 1 to 5			±2		µA
	Input Current During Open Wire Detection		●	70	100	130	µA

LTC6804-1/LTC6804-2

ELECTRICAL CHARACTERISTICS

The ● denotes the specifications which apply over the full operating temperature range, otherwise specifications are at T_A = 25°C. The test conditions are V^+ = 39.6V, V_{REG} = 5.0V unless otherwise noted.

SYMBOL	PARAMETER	CONDITIONS			MIN	TYP	MAX	UNITS
Voltage Reference Specifications								
V_{REF1}	1st Reference Voltage	V_{REF1} Pin, No Load		●	3.1	3.2	3.3	V
	1st Reference Voltage TC	V_{REF1} Pin, No Load				3		ppm/°C
	1st Reference Voltage Hysteresis	V_{REF1} Pin, No Load				20		ppm
	1st Reference Long Term Drift	V_{REF1} Pin, No Load				20		ppm/√kHr
V_{REF2}	2nd Reference Voltage	V_{REF2} Pin, No Load		●	2.990	3	3.010	V
		V_{REF2} Pin, 5k Load to V^-		●	2.988	3	3.012	V
	2nd Reference Voltage TC	V_{REF2} Pin, No Load				10		ppm/°C
	2nd Reference Voltage Hysteresis	V_{REF2} Pin, No Load				100		ppm
	2nd Reference Long Term Drift	V_{REF2} Pin, No Load				60		ppm/√kHr
General DC Specifications								
I_{VP}	V^+ Supply Current (See Figure 1: LTC6804 Operation State Diagram)	State: Core = SLEEP, isoSPI = IDLE	V_{REG} = 0V			3.8	6	µA
			V_{REG} = 0V	●		3.8	10	µA
			V_{REG} = 5V			1.6	3	µA
			V_{REG} = 5V	●		1.6	5	µA
		State: Core = STANDBY			18	32	50	µA
				●	10	32	60	µA
		State: Core = REFUP or MEASURE			0.4	0.55	0.7	mA
				●	0.375	0.55	0.725	mA
$I_{REG(CORE)}$	V_{REG} Supply Current (See Figure 1: LTC6804 Operation State diagram)	State: Core = SLEEP, isoSPI = IDLE	V_{REG} = 5V			2.2	4	µA
			V_{REG} = 5V	●		2.2	6	µA
		State: Core = STANDBY			10	35	60	µA
				●	6	35	65	µA
		State: Core = REFUP			0.2	0.45	0.7	mA
				●	0.15	0.45	0.75	mA
		State: Core = MEASURE			10.8	11.5	12.2	mA
				●	10.7	11.5	12.3	mA
$I_{REG(isoSPI)}$	Additional V_{REG} Supply Current if isoSPI in READY/ACTIVE States Note: ACTIVE State Current Assumes t_{CLK} = 1µs, (Note 3)	LTC6804-2: ISOMD = 1, $R_{B1} + R_{B2}$ = 2k	READY	●	3.9	4.8	5.8	mA
			ACTIVE	●	5.1	6.1	7.3	mA
		LTC6804-1: ISOMD = 0, $R_{B1} + R_{B2}$ = 2k	READY	●	3.7	4.6	5.6	mA
			ACTIVE	●	5.7	6.8	8.1	mA
		LTC6804-1: ISOMD = 1, $R_{B1} + R_{B2}$ = 2k	READY	●	6.5	7.8	9.5	mA
			ACTIVE	●	10.2	11.3	13.3	mA
		LTC6804-2: ISOMD = 1, $R_{B1} + R_{B2}$ = 20k	READY	●	1.3	2.1	3	mA
			ACTIVE	●	1.6	2.5	3.5	mA
		LTC6804-1: ISOMD = 0, $R_{B1} + R_{B2}$ = 20k	READY	●	1.1	1.9	2.8	mA
			ACTIVE	●	1.5	2.3	3.3	mA
		LTC6804-1: ISOMD = 1, $R_{B1} + R_{B2}$ = 20k	READY	●	2.1	3.3	4.9	mA
			ACTIVE	●	2.7	4.1	5.8	mA

ELECTRICAL CHARACTERISTICS

The ● denotes the specifications which apply over the full operating temperature range, otherwise specifications are at T_A = 25°C. The test conditions are V^+ = 39.6V, V_{REG} = 5.0V unless otherwise noted.

SYMBOL	PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
	V^+ Supply Voltage	TME Specifications Met (Note 6)	●	11	40	55	V
V_{REG}	V_{REG} Supply Voltage	TME Supply Rejection < 1mV/V	●	4.5	5	5.5	V
	DRIVE output voltage	Sourcing 1µA	 ●	5.4 5.2	5.6 5.6	5.8 6.0	V V
		Sourcing 500µA	●	5.1	5.6	6.1	V
V_{REGD}	Digital Supply Voltage		●	2.7	3.0	3.6	V
	Discharge Switch ON Resistance	V_{CELL} = 3.6V	●		10	25	Ω
	Thermal Shutdown Temperature				150		°C
$V_{OL(WDT)}$	Watchdog Timer Pin Low	WDT Pin Sinking 4mA	●			0.4	V
$V_{OL(GPIO)}$	General Purpose I/O Pin Low	GPIO Pin Sinking 4mA (Used as Digital Output)	●			0.4	V
ADC Timing Specifications							
t_{CYCLE} (Figure 3)	Measurement + Calibration Cycle Time When Starting from the REFUP State in Normal Mode	Measure 12 Cells	●	2120	2335	2480	µs
		Measure 2 Cells	●	365	405	430	µs
		Measure 12 Cells and 2 GPIO Inputs	●	2845	3133	3325	µs
	Measurement + Calibration Cycle Time When Starting from the REFUP State in Filtered Mode	Measure 12 Cells	●	183	201.3	213.5	ms
		Measure 2 Cells	●	30.54	33.6	35.64	ms
		Measure 12 Cells and 2 GPIO Inputs	●	244	268.4	284.7	ms
	Measurement + Calibration Cycle Time When Starting from the REFUP State in Fast Mode	Measure 12 Cells	●	1010	1113	1185	µs
		Measure 2 Cells	●	180	201	215	µs
		Measure 12 Cells and 2 GPIO Inputs	●	1420	1564	1660	µs
t_{SKEW1} (Figure 6)	Skew Time. The Time Difference between C12 and GPIO2 Measurements, Command = ADCVAX	Fast Mode	●	189	208	221	µs
		Normal Mode	●	493	543	576	µs
t_{SKEW2} (Figure 3)	Skew Time. The Time Difference between C12 and C0 Measurements, Command = ADCV	Fast Mode	●	211	233	248	µs
		Normal Mode	●	609	670	711	µs
t_{WAKE}	Regulator Start-Up Time	V_{REG} Generated from Drive Pin (Figure 28)	●		100	300	µs
t_{SLEEP}	Watchdog or Software Discharge Timer	SWTEN Pin = 0 or DCTO[3:0] = 0000	●	1.8	2	2.2	sec
		SWTEN Pin = 1 and DCTO[3:0] = 0000		0.5		120	min
t_{REFUP} (Figure 1, Figures 3 to 7)	Reference Wake-Up Time	State: Core = STANDBY	●	2.7	3.5	4.4	ms
		State: Core = REFUP	●			0	ms
f_S	ADC Clock Frequency		●	3.0	3.3	3.5	MHz
SPI Interface DC Specifications							
$V_{IH(SPI)}$	SPI Pin Digital Input Voltage High	Pins CSB, SCK, SDI	●	2.3			V
$V_{IL(SPI)}$	SPI Pin Digital Input Voltage Low	Pins CSB, SCK, SDI	●			0.8	V
$V_{IH(CFG)}$	Configuration Pin Digital Input Voltage High	Pins ISOMD, SWTEN, GPIO1 to GPIO5, A0 to A3	●	2.7			V
$V_{IL(CFG)}$	Configuration Pin Digital Input Voltage Low	Pins ISOMD, SWTEN, GPIO1 to GPIO5, A0 to A3	●			1.2	V

ELECTRICAL CHARACTERISTICS

The ● denotes the specifications which apply over the full operating temperature range, otherwise specifications are at T_A = 25°C. The test conditions are V⁺ = 39.6V, V_{REG} = 5.0V unless otherwise noted.

SYMBOL	PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
$I_{LEAK(DIG)}$	Digital Input Current	Pins CSB, SCK, SDI, ISOMD, SWTEN, A0 to A3	●			±1	µA
$V_{OL(SDO)}$	Digital Output Low	Pin SDO Sinking 1mA	●			0.3	V
isoSPI DC Specifications (See Figure 16)							
V_{BIAS}	Voltage on IBIAS Pin	READY/ACTIVE State IDLE State	● ●	1.9	2.0 0	2.1	V V
I_B	Isolated Interface Bias Current	R_{BIAS} = 2k to 20k	●	0.1		1.0	mA
A_{IB}	Isolated Interface Current Gain	$V_A \leq 1.6V$ I_B = 1mA I_B = 0.1mA	● ●	18 18	20 20	22 24.5	mA/mA mA/mA
V_A	Transmitter Pulse Amplitude	$V_A = \lvert V_{IP} - V_{IM} \rvert$	●			1.6	V
V_{ICMP}	Threshold-Setting Voltage on ICMP Pin	$V_{TCMP} = A_{TCMP} \cdot V_{ICMP}$	●	0.2		1.5	V
$I_{LEAK(ICMP)}$	Input Leakage Current on ICMP Pin	V_{ICMP} = 0V to V_{REG}	●			±1	µA
$I_{LEAK(IP/IM)}$	Leakage Current on IP and IM Pins	IDLE State, V_{IP} or V_{IM} = 0V to V_{REG}	●			±1	µA
A_{TCMP}	Receiver Comparator Threshold Voltage Gain	$V_{CM} = V_{REG}/2$ to V_{REG} – 0.2V, V_{ICMP} = 0.2V to 1.5V	●	0.4	0.5	0.6	V/V
V_{CM}	Receiver Common Mode Bias	IP/IM Not Driving			$(V_{REG} - V_{ICMP})/3 - 167mV)$		V
R_{IN}	Receiver Input Resistance	Single-Ended to IPA, IMA, IPB, IMB	●	27	35	43	kΩ
isoSPI Idle/Wakeup Specifications (See Figure 21)							
V_{WAKE}	Differential Wake-Up Voltage	t_{DWELL} = 240ns	●	200			mV
t_{DWELL}	Dwell Time at V_{WAKE} Before Wake Detection	V_{WAKE} = 200mV	●	240			ns
t_{READY}	Startup Time After Wake Detection		●			10	µs
t_{IDLE}	Idle Timeout Duration		●	4.3	5.5	6.7	ms
isoSPI Pulse Timing Specifications (See Figure 19)							
$t_{1/2PW(CS)}$	Chip-Select Half-Pulse Width		●	120	150	180	ns
$t_{INV(CS)}$	Chip-Select Pulse Inversion Delay		●			200	ns
$t_{1/2PW(D)}$	Data Half-Pulse Width		●	40	50	60	ns
$t_{INV(D)}$	Data Pulse Inversion Delay		●			70	ns
SPI Timing Requirements (See Figure 15 and Figure 20)							
t_{CLK}	SCK Period	(Note 4)	●	1			µs
t_1	SDI Setup Time before SCK Rising Edge		●	25			ns
t_2	SDI Hold Time after SCK Rising Edge		●	25			ns
t_3	SCK Low	$t_{CLK} = t_3 + t_4 \geq 1µs$	●	200			ns
t_4	SCK High	$t_{CLK} = t_3 + t_4 \geq 1µs$	●	200			ns
t_5	CSB Rising Edge to CSB Falling Edge		●	0.65			µs
t_6	SCK Rising Edge to CSB Rising Edge	(Note 4)	●	0.8			µs
t_7	CSB Falling Edge to SCK Rising Edge	(Note 4)	●	1			µs

6804121c

ELECTRICAL CHARACTERISTICS

The ● denotes the specifications which apply over the full operating temperature range, otherwise specifications are at T_A = 25°C. The test conditions are V^+ = 39.6V, V_{REG} = 5.0V unless otherwise noted.

SYMBOL	PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
isoSPI Timing Specifications (See Figure 19)							
t_8	SCK Falling Edge to SDO Valid	(Note 5)	●			60	ns
t_9	SCK Rising Edge to Short ±1 Transmit		●			50	ns
t_{10}	CSB Transition to Long ±1 Transmit		●			60	ns
t_{11}	CSB Rising Edge to SDO Rising	(Note 5)	●			200	ns
t_{RTN}	Data Return Delay		●		430	525	ns
$t_{DSY(CS)}$	Chip-Select Daisy-Chain Delay		●		150	200	ns
$t_{DSY(D)}$	Data Daisy-Chain Delay		●		300	360	ns
t_{LAG}	Data Daisy-Chain Lag (vs Chip-Select)		●	0	35	70	ns
$t_{S(GOV)}$	Data to Chip-Select Pulse Governor		●	0.8		1.05	µs

Note 1: Stresses beyond those listed under Absolute Maximum Ratings may cause permanent damage to the device. Exposure to any Absolute Maximum Rating condition for extended periods may affect device reliability and lifetime.

Note 2: The ADC specifications are guaranteed by the Total Measurement Error specification.

Note 3: The ACTIVE state current is calculated from DC measurements. The ACTIVE state current is the additional average supply current into V_{REG} when there is continuous 1MHz communications on the isoSPI ports with 50% data 1's and 50% data 0's. Slower clock rates reduce the supply current. See Applications Information section for additional details.

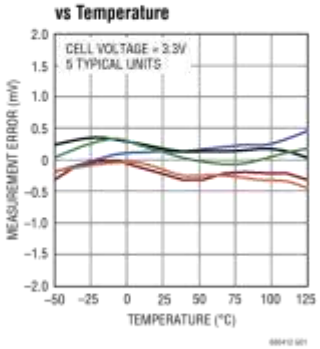
Note 4: These timing specifications are dependent on the delay through the cable, and include allowances for 50ns of delay each direction. 50ns corresponds to 10m of CAT-5 cable (which has a velocity of propagation of 66% the speed of light). Use of longer cables would require derating these specs by the amount of additional delay.

Note 5: These specifications do not include rise or fall time of SDO. While fall time (typically 5ns due to the internal pull-down transistor) is not a concern, rising-edge transition time t_{RISE} is dependent on the pull-up resistance and load capacitance on the SDO pin. The time constant must be chosen such that SDO meets the setup time requirements of the MCU.

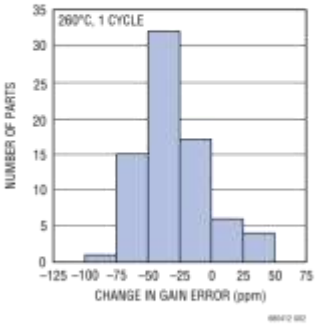
Note 6: V^+ needs to be greater than or equal to the highest C(n) voltage for accurate measurements. See the graph Top Cell Measurement Error vs V^+.

TYPICAL PERFORMANCE CHARACTERISTICS T_A = 25°C, unless otherwise noted.

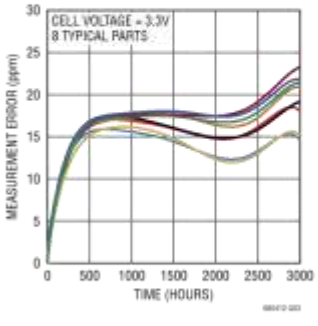
Measurement Error vs Temperature



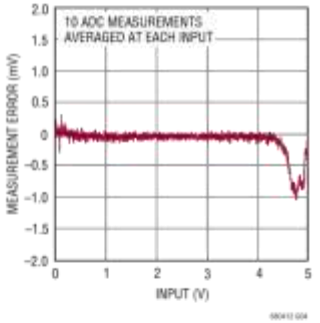
Measurement Error Due to IR Reflow



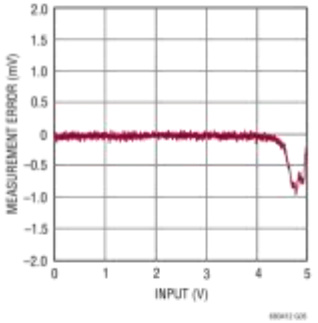
Measurement Error Long-Term Drift



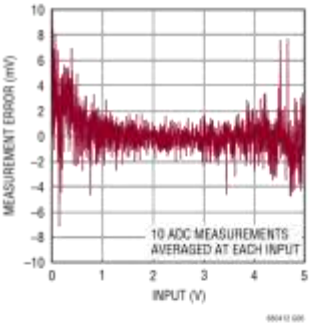
Measurement Error vs Input, Normal Mode



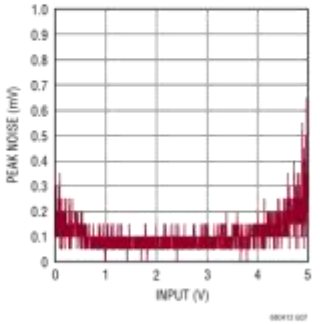
Measurement Error vs Input, Filtered Mode



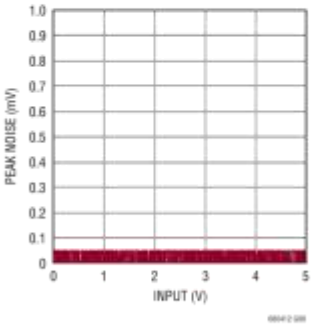
Measurement Error vs Input, Fast Mode



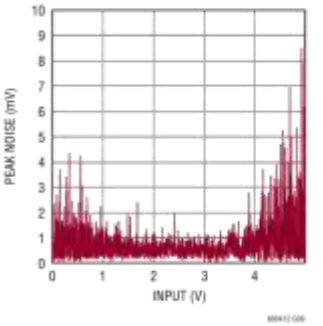
Measurement Noise vs Input, Normal Mode



Measurement Noise vs Input, Filtered Mode

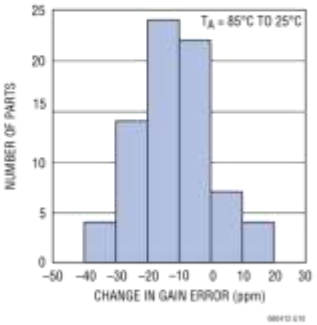


Measurement Noise vs Input, Fast Mode

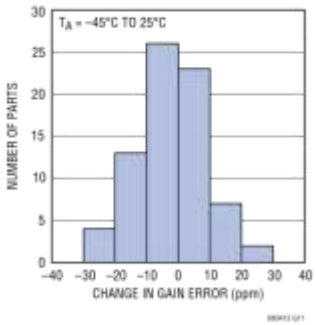


TYPICAL PERFORMANCE CHARACTERISTICS T_A = 25°C, unless otherwise noted.

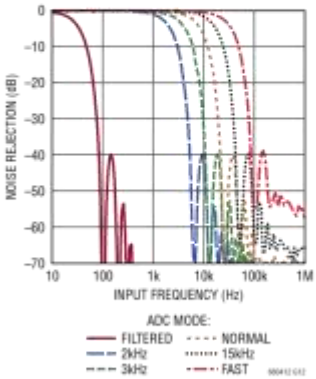
Measurement Gain Error Hysteresis, Hot



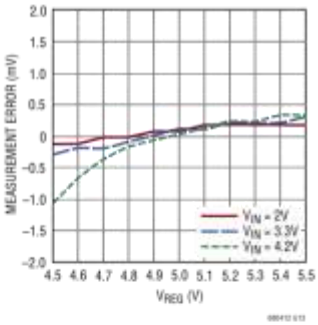
Measurement Gain Error Hysteresis, Cold



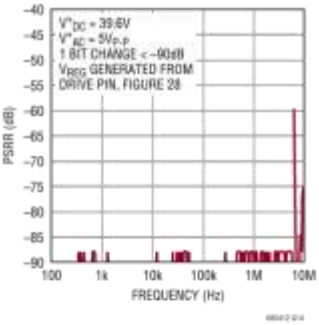
Noise Filter Response



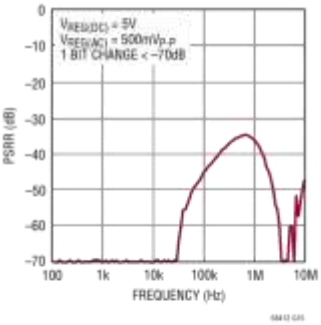
Measurement Error vs V_{REG}



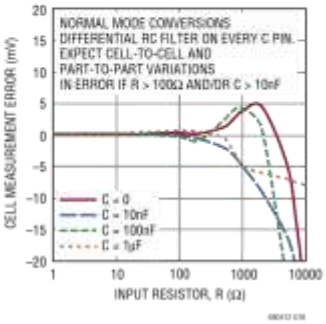
Measurement Error V⁺ PSRR vs Frequency



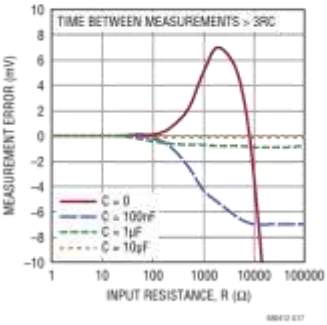
Measurement Error V_{REG} PSRR vs Frequency



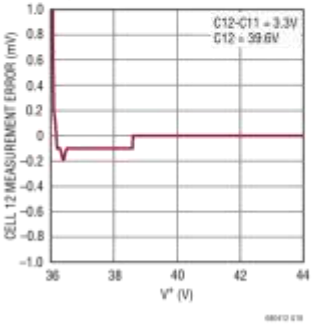
Cell Measurement Error vs Input RC Values



GPIO Measurement Error vs Input RC Values



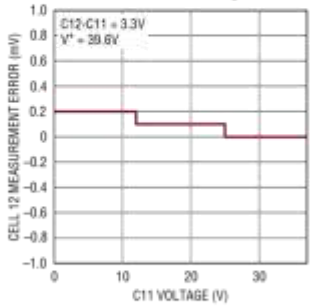
Top Cell Measurement Error vs V⁺



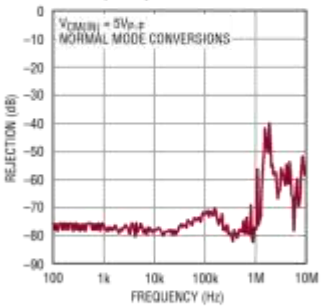
680412fc

TYPICAL PERFORMANCE CHARACTERISTICS T_A = 25°C, unless otherwise noted.

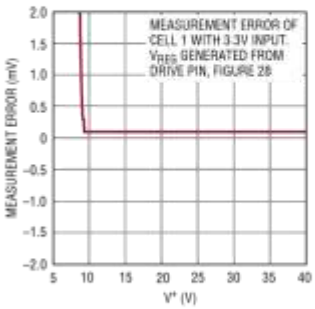
Cell Measurement Error vs Common Mode Voltage



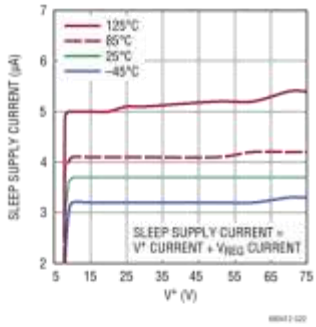
Cell Measurement CMRR vs Frequency



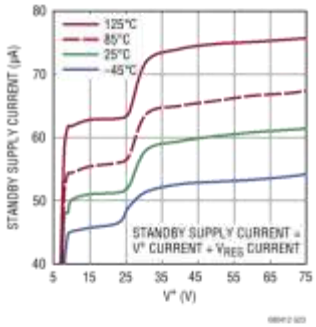
Measurement Error vs V+



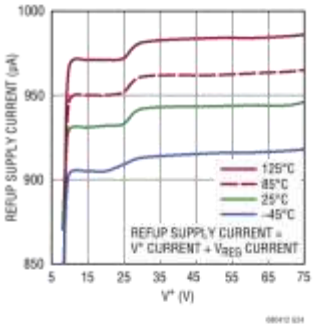
Sleep Supply Current vs V+



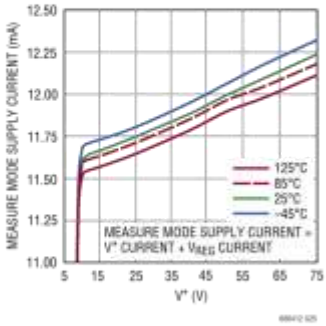
Standby Supply Current vs V+



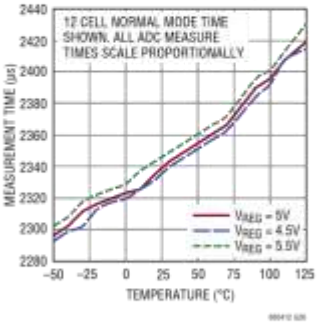
REFUP Supply Current vs V+



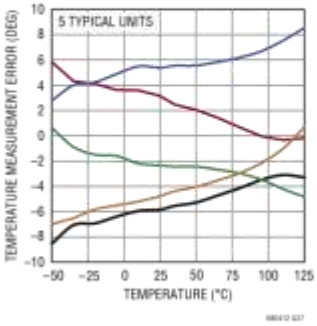
Measure Mode Supply Current vs V+



Measurement Time vs Temperature



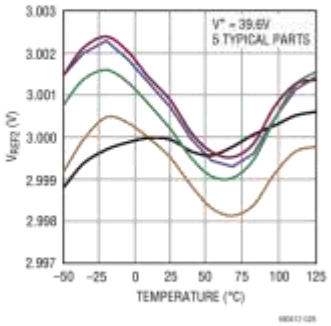
Internal Die Temperature Measurement Error vs Temperature



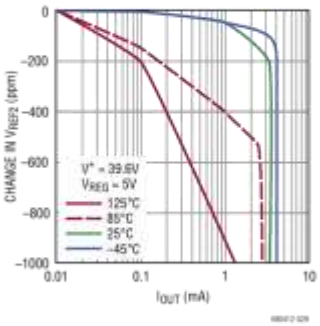
680412fc

TYPICAL PERFORMANCE CHARACTERISTICS T_A = 25°C, unless otherwise noted.

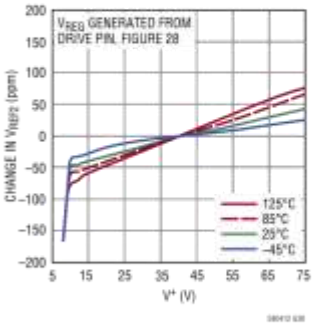
V_{REF2} vs Temperature



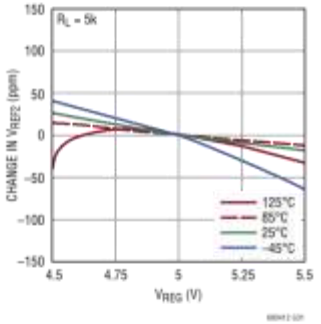
V_{REF2} Load Regulation



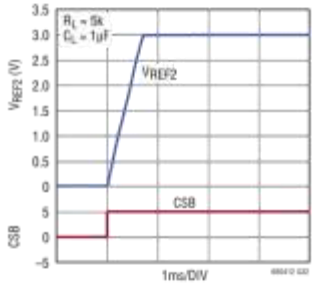
V_{REF2} V^+ Line Regulation



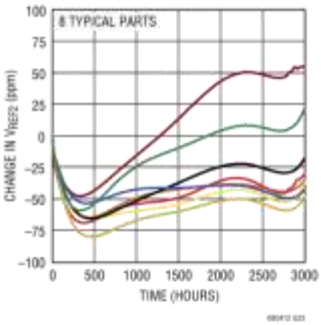
V_{REF2} V_{REG} Line Regulation



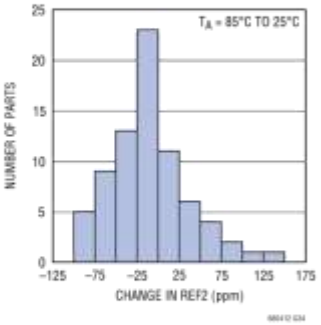
V_{REF2} Power-Up



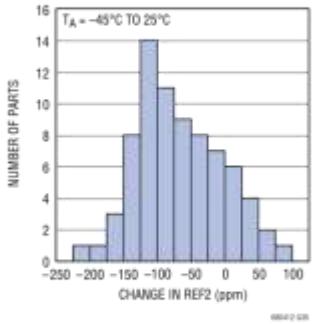
V_{REF2} Long-Term Drift



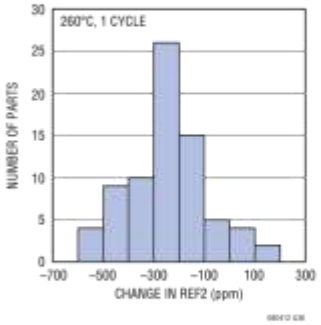
V_{REF2} Hysteresis, Hot



V_{REF2} Hysteresis, Cold



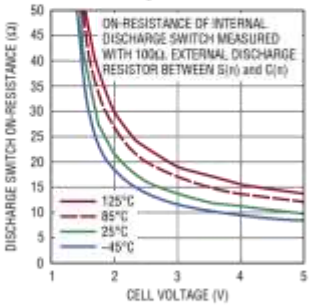
V_{REF2} Change Due to IR Reflow



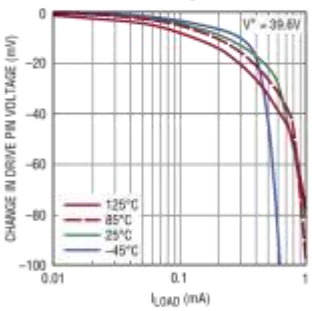
LTC6804-1/LTC6804-2

TYPICAL PERFORMANCE CHARACTERISTICS T_A = 25°C, unless otherwise noted.

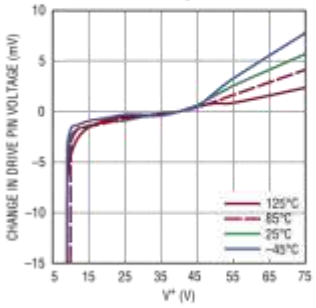
Discharge Switch On-Resistance vs Cell Voltage



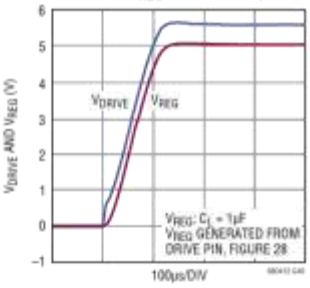
Drive Pin Load Regulation



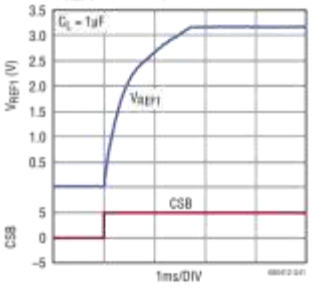
Drive Pin Line Regulation



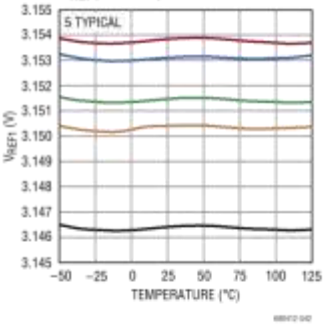
Drive and V_{REG} Pin Power-Up



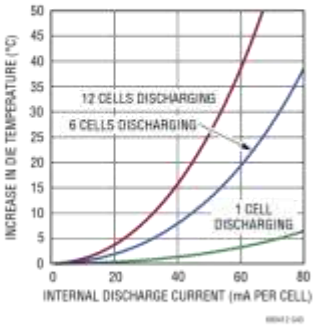
V_{REF1} Power-Up



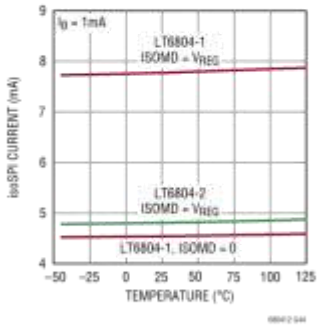
V_{REF1} vs Temperature



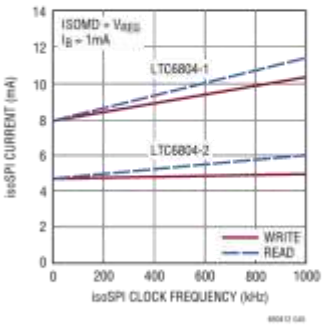
Internal Die Temperature Increase vs Discharge Current



isoSPI Current (READY) vs Temperature



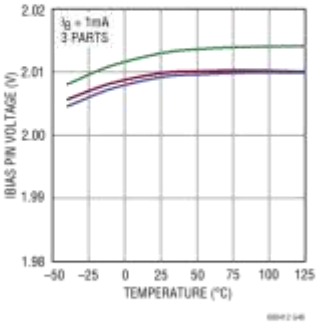
isoSPI Current (READY/ACTIVE) vs isoSPI Clock Frequency



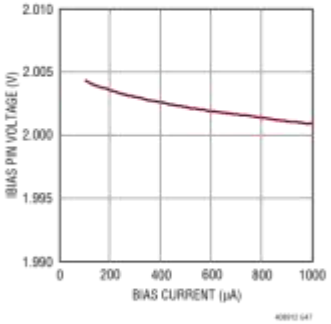
680412fc

TYPICAL PERFORMANCE CHARACTERISTICS T_A = 25°C, unless otherwise noted.

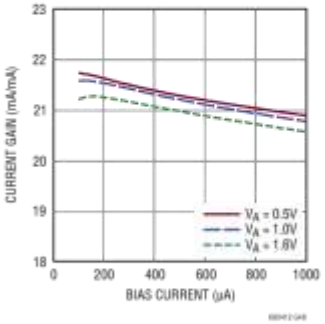
I_{BIAS} Voltage vs Temperature



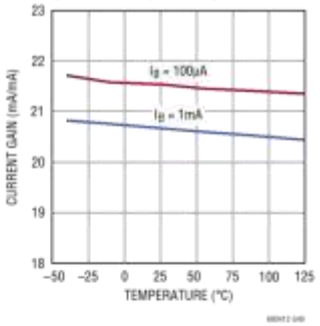
I_{BIAS} Voltage Load Regulation



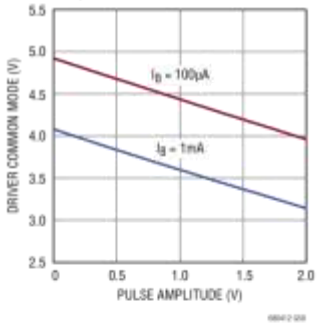
isoSPI Driver Current Gain (Port A/PortB) vs Bias Current



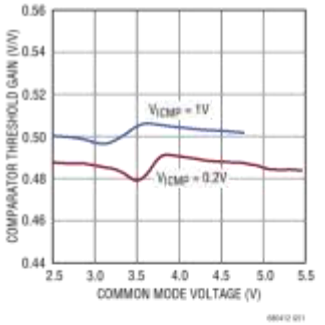
isoSPI Driver Current Gain (Port A/PortB) vs Temperature



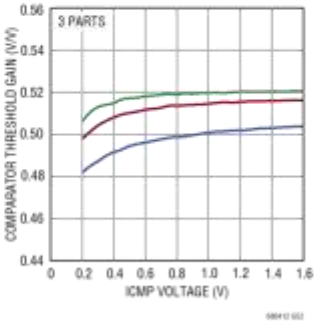
isoSPI Driver Common Mode Voltage (Port A/Port B) vs Pulse Amplitude



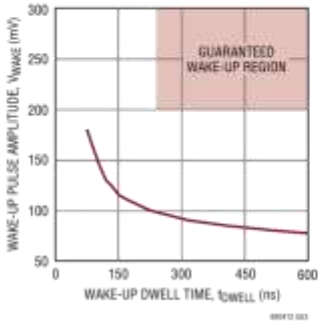
isoSPI Comparator Threshold Gain (Port A/Port B) vs Common Mode



isoSPI Comparator Threshold Gain (Port A/Port B) vs ICMP Voltage

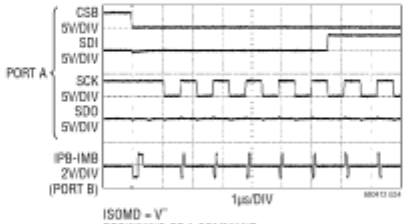


Typical Wake-Up Pulse Amplitude (Port A) vs Dwell Time



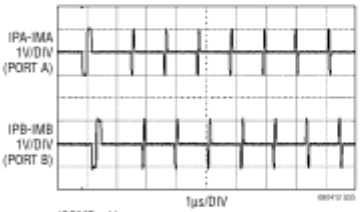
TYPICAL PERFORMANCE CHARACTERISTICS T_A = 25°C, unless otherwise noted.

Write Command to a Daisy-Chained Device (ISOMD = 0)



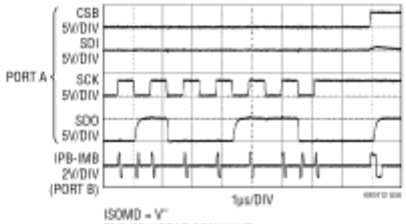
ISOMD = V⁻
BEGINNING OF A COMMAND

Write Command to a Daisy-Chained Device (ISOMD = 1)



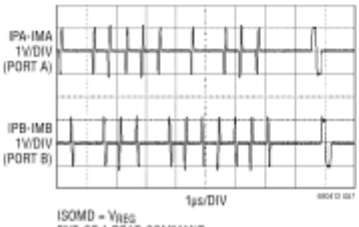
ISOMD = V_{REG}
BEGINNING OF A COMMAND

Data Read-Back from a Daisy-Chained Device (ISOMD = 0)



ISOMD = V⁻
END OF A READ COMMAND

Data Read-Back from a Daisy-Chained Device (ISOMD = 1)



ISOMD = V_{REG}
END OF A READ COMMAND

PIN FUNCTIONS

C0 to C12: Cell Inputs.

S1 to S12: Balance Inputs/Outputs. 12 N-MOSFETs are connected between S(n) and C(n–1) for discharging cells.

V⁺: Positive Supply Pin.

V⁻: Negative Supply Pins. The V⁻ pins must be shorted together, external to the IC.

V_{REF2}: Buffered 2nd reference voltage for driving multiple 10k thermistors. Bypass with an external 1µF capacitor.

V_{REF1}: ADC Reference Voltage. Bypass with an external 1µF capacitor. No DC loads allowed.

GPIO[1:5]: General Purpose I/O. Can be used as digital inputs or digital outputs, or as analog inputs with a measurement range from V⁻ to 5V. GPIO [3:5] can be used as an I^2C or SPI port.

SWTEN: Software Timer Enable. Connect this pin to V_{REG} to enable the software timer.

DRIVE: Connect the base of an NPN to this pin. Connect the collector to V⁺ and the emitter to V_{REG}.

V_{REG}: 5V Regulator Input. Bypass with an external 1µF capacitor.

ISOMD: Serial Interface Mode. Connecting ISOMD to V_{REG} configures Pins 41 to 44 of the LTC6804 for 2-wire isolated interface (isoSPI) mode. Connecting ISOMD to V⁻ configures the LTC6804 for 4-wire SPI mode.

WDT: Watchdog Timer Output Pin. This is an open drain NMOS digital output. It can be left unconnected or connected with a 1M resistor to V_{REG}. If the LTC6804 does not receive a wake-up signal (see Figure 21) within 2 seconds, the watchdog timer circuit will reset the LTC6804 and the WDT pin will go high impedance.

Serial Port Pins

	LTC6804-1 (DAISY-CHAINABLE)		LTC6804-2 (ADDRESSABLE)	
	ISOMD = V_{REG}	ISOMD = V⁻	ISOMD = V_{REG}	ISOMD = V⁻
PORT B (Pins 45 to 48)	IPB	IPB	A3	A3
	IMB	IMB	A2	A2
	ICMP	ICMP	A1	A1
	IBIAS	IBIAS	A0	A0
PORT A (Pins 41 to 44)	(NC)	SDO	IBIAS	SDO
	(NC)	SDI	ICMP	SDI
	IPA	SCK	IPA	SCK
	IMA	CSB	IMA	CSB

CSB, SCK, SDI, SDO: 4-Wire Serial Peripheral Interface (SPI). Active low chip select (CSB), serial clock (SCK), and serial data in (SDI) are digital inputs. Serial data out (SDO) is an open drain NMOS output pin. SDO requires a 5k pull-up resistor.

A0 to A3: Address Pins. These digital inputs are connected to V_{REG} or V⁻ to set the chip address for addressable serial commands.

IPA, IMA: Isolated 2-Wire Serial Interface Port A. IPA (plus) and IMA (minus) are a differential input/output pair.

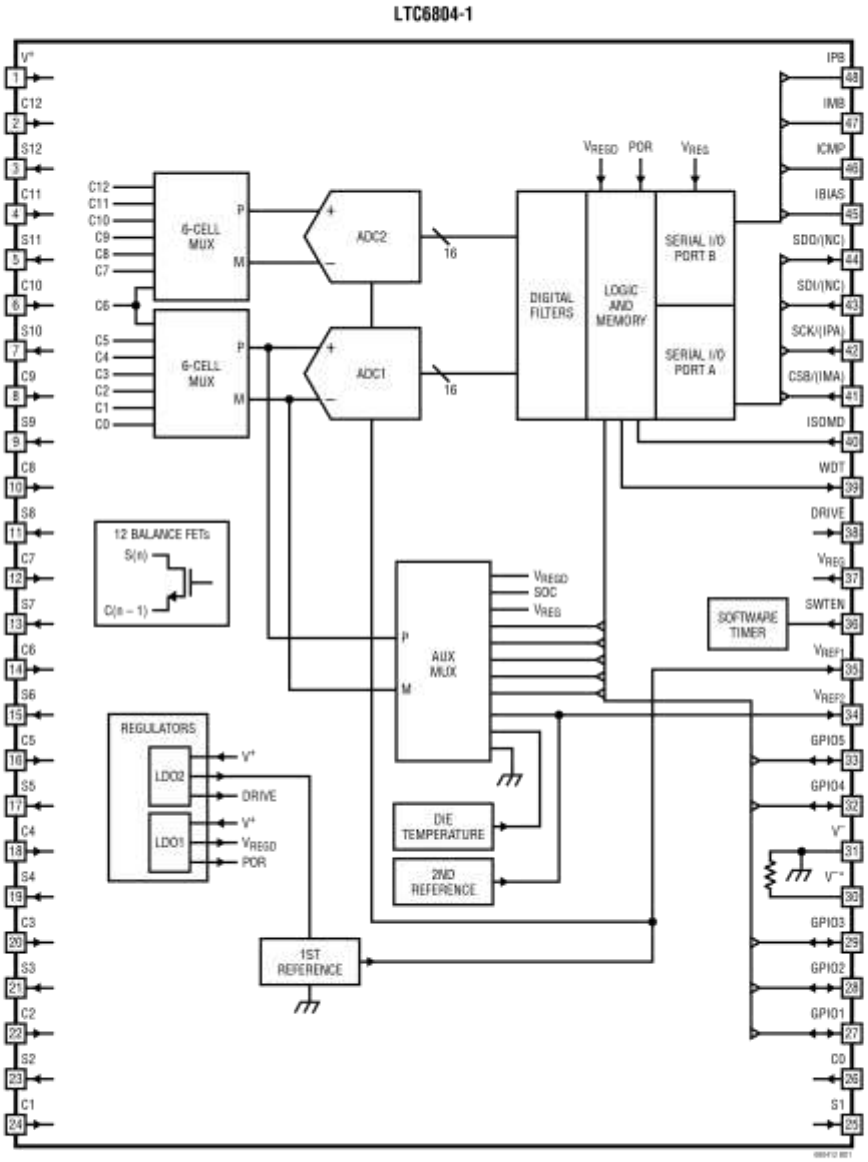
IPB, IMB: Isolated 2-Wire Serial Interface Port B. IPB (plus) and IMB (minus) are a differential input/output pair.

IBIAS: Isolated Interface Current Bias. Tie IBIAS to V⁻ through a resistor divider to set the interface output current level. When the isoSPI interface is enabled, the IBIAS pin voltage is 2V. The IPA/IMA or IPB/IMB output current drive is set to 20 times the current, I_B, sourced from the IBIAS pin.

ICMP: Isolated Interface Comparator Voltage Threshold Set. Tie this pin to the resistor divider between IBIAS and V⁻ to set the voltage threshold of the isoSPI receiver comparators. The comparator thresholds are set to 1/2 the voltage on the ICMP pin.

LTC6804-1/LTC6804-2

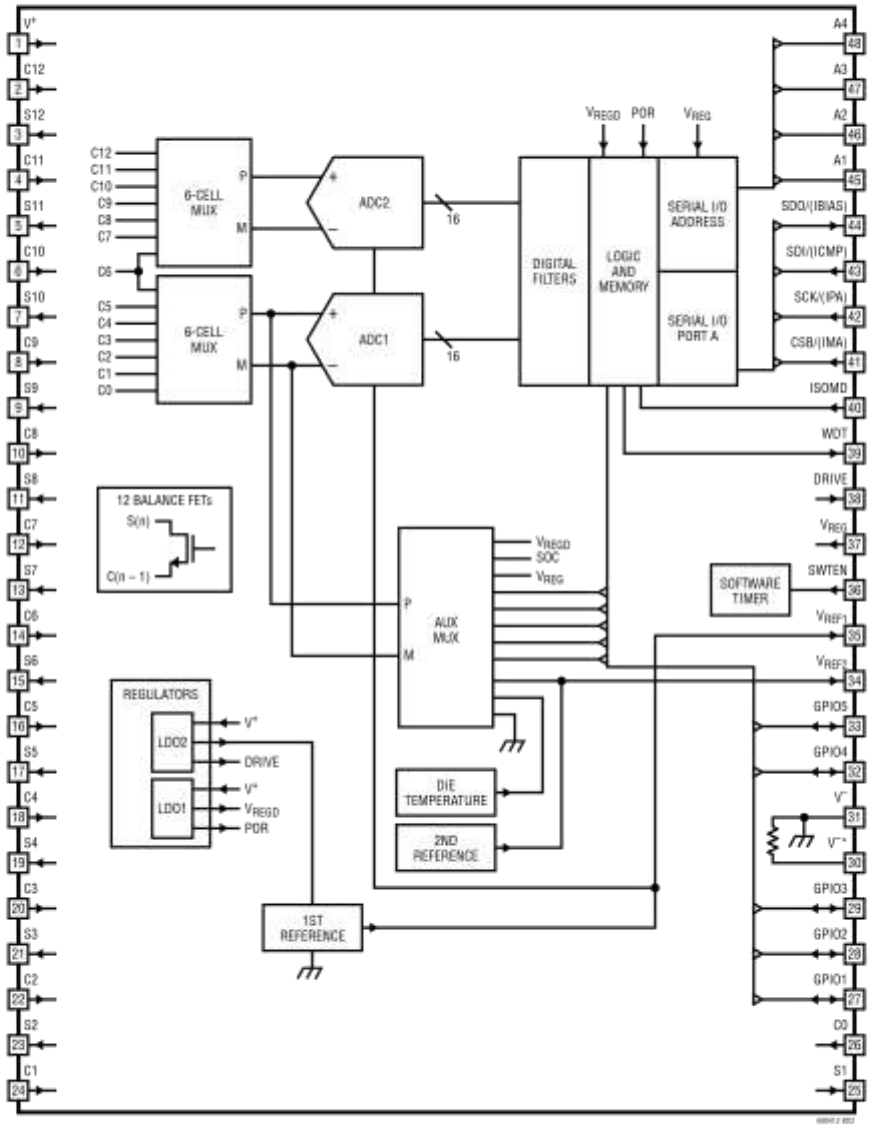
BLOCK DIAGRAM



6804121fc

BLOCK DIAGRAM

LTC6804-2



OPERATION

STATE DIAGRAM

The operation of the LTC6804 is divided into two separate sections: the core circuit and the isoSPI circuit. Both sections have an independent set of operating states, as well as a shutdown timeout.

LTC6804 CORE STATE DESCRIPTIONS

SLEEP State

The reference and ADCs are powered down. The watchdog timer (see Watchdog and Software Discharge Timer) has timed out. The software discharge timer is either disabled or timed out. The supply currents are reduced to minimum levels. The isoSPI ports will be in the IDLE state.

If a WAKEUP signal is received (see Waking Up the Serial Interface), the LTC6804 will enter the STANDBY state.

STANDBY State

The reference and the ADCs are off. The watchdog timer and/or the software discharge timer is running. The DRIVE pin powers the V_{REG} pin to 5V through an external transistor. (Alternatively, V_{REG} can be powered by an external supply).

When a valid ADC command is received or the REFON bit is set to 1 in the Configuration Register Group, the IC pauses for t_{REFUP} to allow for the reference to power up and then enters either the REFUP or MEASURE state. If there is no WAKEUP signal for a duration t_{SLEEP} (when both the watchdog and software discharge timer have expired) the LTC6804 returns to the SLEEP state. If the software discharge timer is disabled, only the watchdog timer is relevant.

REFUP State

To reach this state the REFON bit in the Configuration Register Group must be set to 1 (using the WRCFG command, see Table 36). The ADCs are off. The reference is powered up so that the LTC6804 can initiate ADC conversions more quickly than from the STANDBY state.

When a valid ADC command is received, the IC goes to the MEASURE state to begin the conversion. Otherwise, the LTC6804 will return to the STANDBY state when the REFON bit is set to 0, either manually (using WRCFG command) or automatically when the watchdog timer expires. (The LTC6804 will then move straight into the SLEEP state if both timers are expired).

MEASURE State

The LTC6804 performs ADC conversions in this state. The reference and ADCs are powered up.

After ADC conversions are complete the LTC6804 will transition to either the REFUP or STANDBY states, depending on the REFON bit. Additional ADC conversions can be initiated more quickly by setting REFON = 1 to take advantage of the REFUP state.

Note: Non-ADC commands do not cause a Core state transition. Only an ADC conversion or diagnostic commands will place the Core in the MEASURE state.

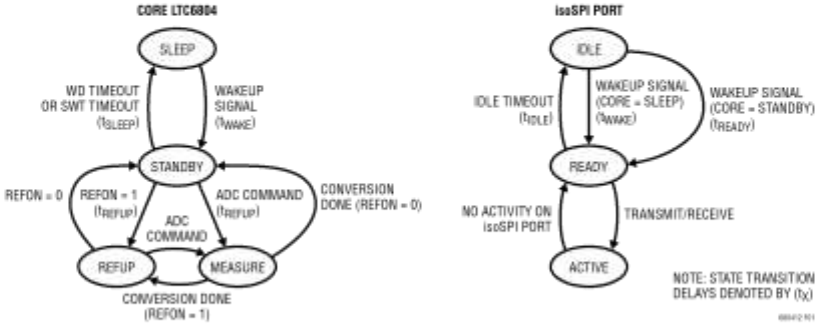


Figure 1. LTC6804 Operation State Diagram

OPERATION

isoSPI STATE DESCRIPTIONS

Note: The LTC6804-1 has two isoSPI ports (A and B), for daisy-chain communication. The LTC6804-2 has only one isoSPI port (A), for parallel-addressable communication.

IDLE State

The isoSPI ports are powered down.

When isoSPI port A receives a WAKEUP signal (see Waking Up the Serial Interface), the isoSPI enters the READY state. This transition happens quickly (within t_{READY}) if the Core is in the STANDBY state because the DRIVE and V_{REG} pins are already biased up. If the Core is in the SLEEP state when the isoSPI receives a WAKEUP signal, then it transitions to the READY state within t_{WAKE}.

READY State

The isoSPI port(s) are ready for communication. Port B is enabled only for LTC6804-1, and is not present on the LTC6804-2. The serial interface current in this state depends on if the part is LTC6804-1 or LTC6804-2, the status of the ISOMD pin, and $R_{BIAS} = R_{B1} + R_{B2}$ (the external resistors tied to the IBIAS pin).

If there is no activity (i.e., no WAKEUP signal) on port A for greater than t_{IDLE} = 5.5ms, the LTC6804 goes to the IDLE state. When the serial interface is transmitting or receiving data the LTC6804 goes to the ACTIVE state.

ACTIVE State

The LTC6804 is transmitting/receiving data using one or both of the isoSPI ports. The serial interface consumes maximum power in this state. The supply current increases with clock frequency as the density of isoSPI pulses increases.

POWER CONSUMPTION

The LTC6804 is powered via two pins: V^+ and V_{REG}. The V^+ input requires voltage greater than or equal to the top cell voltage, and it provides power to the high voltage elements of the core circuitry. The V_{REG} input requires 5V and provides power to the remaining core circuitry and the isoSPI circuitry. The V_{REG} input can be powered through an external transistor, driven by the regulated DRIVE output pin. Alternatively, V_{REG} can be powered by an external supply.

The power consumption varies according to the operational states. Table 1 and Table 2 provide equations to approximate the supply pin currents in each state. The V^+ pin current depends only on the Core state and not on the isoSPI state. However, the V_{REG} pin current depends on both the Core state and isoSPI state, and can therefore be divided into two components. The isoSPI interface draws current only from the V_{REG} pin.

$$I_{REG} = I_{REG(CORE)} + I_{REG(isoSPI)}$$

Table 1. Core Supply Current

STATE		I_{V+}	$I_{REG(CORE)}$
SLEEP	V_{REG} = 0V	3.8µA	0µA
	V_{REG} = 5V	1.6µA	2.2µA
STANDBY		32µA	35µA
REFUP		550µA	450µA
MEASURE		550µA	11.5mA

In the SLEEP state the V_{REG} pin will draw approximately 2.2µA if powered by a external supply. Otherwise, the V^+ pin will supply the necessary current.

ADC OPERATION

There are two ADCs inside the LTC6804. The two ADCs operate simultaneously when measuring twelve cells. Only one ADC is used to measure the general purpose inputs. The following discussion uses the term ADC to refer to one or both ADCs, depending on the operation being performed. The following discussion will refer to ADC1 and ADC2 when it is necessary to distinguish between the two circuits, in timing diagrams, for example.

ADC Modes

The ADCOPT bit (CFGR0[0]) in the configuration register group and the mode selection bits MD[1:0] in the conversion command together provide 6 modes of operation for the ADC which correspond to different over sampling ratios (OSR). The accuracy of these modes are summarized in Table 3. In each mode, the ADC first measures the inputs, and then performs a calibration of each channel. The names of the modes are based on the –3dB bandwidth of the ADC measurement.

OPERATION

Table 2. isoSPI Supply Current Equations

isoSPI STATE	DEVICE	ISOMD CONNECTION	$I_{REG(isoSPI)}$
IDLE	LTC6804-1/LTC6804-2	N/A	0mA
READY	LTC6804-1	V_{REG}	2.8mA + 5 • I_B Note: $I_B = V_{BIAS}/(R_{B1} + R_{B2})$
		V^-	1.6mA + 3 • I_B
	LTC6804-2	V_{REG}	1.8mA + 3 • I_B
		V^-	0mA
ACTIVE	LTC6804-1	V_{REG}	Write: $2.8mA+5\bullet I_B+(2\bullet I_B+0.4mA)\bullet\frac{1\mu s}{t_{CLK}}$ Read: $2.8mA+5\bullet I_B+(3\bullet I_B+0.5mA)\bullet\frac{1\mu s}{t_{CLK}}$
		V^-	$1.6mA+3\bullet I_B+(2\bullet I_B+0.2mA)\bullet\frac{1\mu s}{t_{CLK}}$
	LTC6804-2	V_{REG}	Write: $1.8mA+3\bullet I_B+(0.3mA)\bullet\frac{1\mu s}{t_{CLK}}$ Read: $1.8mA+3\bullet I_B+(I_B+0.3mA)\bullet\frac{1\mu s}{t_{CLK}}$
		V^-	0mA

Table 3. ADC Filter Bandwidth and Accuracy

MODE	–3dB FILTER BW	–40dB FILTER BW	TME SPEC AT 3.3V, 25°C	TME SPEC AT 3.3V,–40°C, 125°C
27kHz (Fast Mode)	27kHz	84kHz	±4.7mV	±4.7mV
14kHz	13.5kHz	42kHz	±4.7mV	±4.7mV
7kHz (Normal Mode)	6.8kHz	21kHz	±1.2mV	±2.2mV
3kHz	3.4kHz	10.5kHz	±1.2mV	±2.2mV
2kHz	1.7kHz	5.3kHz	±1.2mV	±2.2mV
26Hz (Filtered Mode)	26Hz	82Hz	±1.2mV	±2.2mV

Note: TME is the total measurement error.

Mode 7kHz (Normal):

In this mode, the ADC has high resolution and low TME (total measurement error). This is considered the normal operating mode because of the optimum combination of speed and accuracy.

Mode 27kHz (Fast):

In this mode, the ADC has maximum throughput but has some increase in TME (total measurement error). So this mode is also referred to as the fast mode. The increase in speed comes from a reduction in the oversampling ratio. This results in an increase in noise and average measurement error.

Mode 26Hz (Filtered):

In this mode, the ADC digital filter –3dB frequency is lowered to 26Hz by increasing the OSR. This mode is also referred to as the filtered mode due to its low –3dB frequency. The accuracy is similar to the 7kHz (Normal) mode with lower noise.

Modes 14kHz, 3kHz and 2kHz:

Modes 14kHz, 3kHz and 2kHz provide additional options to set the ADC digital filter –3dB frequency at 13.5kHz, 3.4kHz and 1.7kHz respectively. The accuracy of the 14kHz mode is similar to the 27kHz (fast) mode. The accuracy of 3kHz and 2kHz modes is similar to the 7kHz (normal) mode.

6804121c

OPERATION

The conversion times for these modes are provided in Table 5. If the core is in STANDBY state, an additional t_{REFUP} time is required to power up the reference before beginning the ADC conversions. The reference can remain powered up between ADC conversions if the REFON bit in Configuration Register Group is set to 1 so the core is in REFUP state after a delay t_{REFUP}. Then, the subsequent ADC commands will not have the t_{REFUP} delay before beginning ADC conversions.

ADC Range and Resolution

The C inputs and GPIO inputs have the same range and resolution. The ADC inside the LTC6804 has an approximate range from –0.82V to 5.73V. Negative readings are rounded to 0V. The format of the data is a 16-bit unsigned integer where the LSB represents 100µV. Therefore, a reading of 0x80E8 (33,000 decimal) indicates a measurement of 3.3V.

Delta-Sigma ADCs have quantization noise which depends on the input voltage, especially at low over sampling ratios (OSR), such as in FAST mode. In some of the ADC modes, the quantization noise increases as the input voltage approaches the upper and lower limits of the ADC range. For example, the total measurement noise versus input voltage in normal and filtered modes is shown in Figure 2.

The specified range of the ADC is 0V to 5V. In Table 4, the precision range of the ADC is arbitrarily defined as 0.5V to 4.5V. This is the range where the quantization noise is relatively constant even in the lower OSR modes (see Figure 2). Table 4 summarizes the total noise in this range for all six ADC operating modes. Also shown is the noise

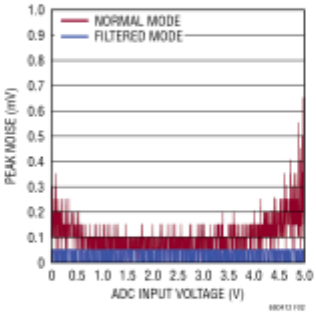


Figure 2. Measurement Noise vs Input Voltage

free resolution. For example, 14-bit noise free resolution in normal mode implies that the top 14 bits will be noise free with a DC input, but that the 15th and 16th least significant bits (LSB) will flicker.

ADC Range vs Voltage Reference Value:

Typical Delta-Sigma ADC's have a range which is exactly twice the value of the voltage reference, and the ADC measurement error is directly proportional to the error in the voltage reference. The LTC6804 ADC is not typical. The absolute value of V_{REF1} is trimmed up or down to compensate for gain errors in the ADC. Therefore, the ADC total measurement error (TME) specifications are superior to the V_{REF1} specifications. For example, the 25°C specification of the total measurement error when measuring 3.300V in 7kHz (normal) mode is ±1.2mV and the 25°C specification for V_{REF1} is 3.200V ±100mV.

Table 4. ADC Range and Resolution

MODE	FULL RANGE[1]	SPECIFIED RANGE	PRECISION RANGE[2]	LSB	FORMAT	MAX NOISE	NOISE FREE RESOLUTION[3]
27kHz (Fast)	–0.8192V to 5.7344V	0V to 5V	0.5V to 4.5V	100µV	Unsigned 16 Bits	±4mV$_{P-P}$	10 Bits
14kHz						±1mV$_{P-P}$	12 Bits
7kHz (Normal)						±250µV$_{P-P}$	14 Bits
3kHz						±150µV$_{P-P}$	14 Bits
2kHz						±100µV$_{P-P}$	15 Bits
26Hz (Filtered)						±50µV$_{P-P}$	16 Bits

1. Negative readings are rounded to 0V.
2. PRECISION RANGE is the range over which the noise is less than MAX NOISE.
3. NOISE FREE RESOLUTION is a measure of the noise level within the PRECISION RANGE.

OPERATION

Measuring Cell Voltages (ADCV Command)

The ADCV command initiates the measurement of the battery cell inputs, pins C0 through C12. This command has options to select the number of channels to measure and the ADC mode. See the section on Commands for the ADCV command format.

Figure 3 illustrates the timing of ADCV command which measures all twelve cells. After the receipt of the ADCV command to measure all 12 cells, ADC1 sequentially measures the bottom 6 cells. ADC2 sequentially measures the top 6 cells. After the cell measurements are complete, each channel is calibrated to remove any offset errors.

Table 5 shows the conversion times for the ADCV command measuring all 12 cells. The total conversion time is given by t_{6C} which indicates the end of the calibration step.

Figure 4 illustrates the timing of the ADCV command that measures only two cells.

Table 6 shows the conversion time for ADCV command measuring only 2 cells. t_{1C} indicates the total conversion time for this command.

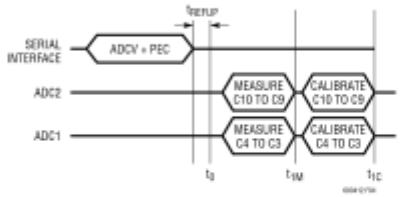


Figure 4. Timing for ADCV Command Measuring 2 Cells

Table 6. Conversion Times for ADCV Command Measuring Only 2 Cells in Different Modes

	CONVERSION TIMES (in µs)		
MODE	t_0	t_{1M}	t_{1C}
27kHz	0	57	201
14kHz	0	86	230
7kHz	0	144	405
3kHz	0	260	521
2kHz	0	493	754
26Hz	0	29,817	33,568

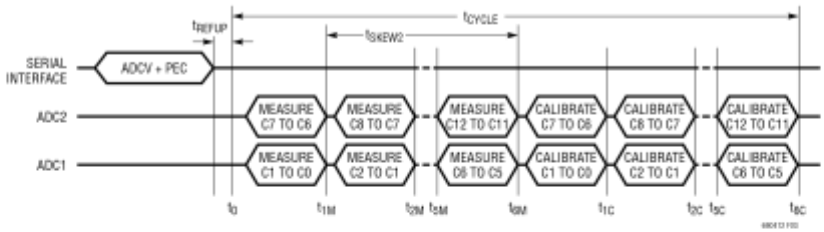


Figure 3. Timing for ADCV Command Measuring All 12 Cells

Table 5. Conversion Times for ADCV Command Measuring All 12 Cells in Different Modes

	CONVERSION TIMES (in µs)								
MODE	t_0	t_{1M}	t_{2M}	t_{5M}	t_{6M}	t_{1C}	t_{2C}	t_{5C}	t_{6C}
27kHz	0	57	103	243	290	432	568	975	1,113
14kHz	0	86	162	389	465	606	742	1,149	1,288
7kHz	0	144	278	680	814	1,072	1,324	2,080	2,335
3kHz	0	260	511	1,262	1,512	1,770	2,022	2,778	3,033
2kHz	0	493	976	2,425	2,908	3,166	3,418	4,175	4,430
26Hz	0	29,817	59,623	149,043	178,850	182,599	186,342	197,571	201,317

OPERATION

Under/Overvoltage Monitoring

Whenever the C inputs are measured, the results are compared to undervoltage and overvoltage thresholds stored in memory. If the reading of a cell is above the overvoltage limit, a bit in memory is set as a flag. Similarly, measurement results below the undervoltage limit cause a flag to be set. The overvoltage and undervoltage thresholds are stored in the configuration register group. The flags are stored in the status register group B.

Auxiliary (GPIO) Measurements (ADAX Command)

The ADAX command initiates the measurement of the GPIO inputs. This command has options to select which GPIO input to measure (GPIO1-5) and which ADC mode. The ADAX command also measures the 2nd reference. There are options in the ADAX command to measure each GPIO and the 2nd reference separately or to measure all 5 GPIOs and the 2nd reference in a single command. See the section on commands for the ADAX command format. All auxiliary measurements are relative to the V^- pin voltage. This command can be used to read external temperature by connecting the temperature sensors to the GPIOs. These sensors can be powered from the 2nd reference which is also measured by the ADAX command, resulting in precise ratiometric measurements.

Figure 5 illustrates the timing of the ADAX command measuring all GPIOs and the 2nd reference. Since all the 6 measurements are carried out on ADC1 alone, the conversion time for the ADAX command is similar to the ADCV command.

Measuring Cell Voltages and GPIOs (ADCVAX Command)

The ADCVAX command combines twelve cell measurements with two GPIO measurements (GPIO1 and GPIO2). This command simplifies the synchronization of battery cell voltage and current measurements when current sensors are connected to GPIO1 or GPIO2 inputs. Figure 6 illustrates the timing of the ADCVAX command. See the section on commands for the ADCVAX command format. The synchronization of the current and voltage measurements, t_{SKEW1}, in FAST MODE is within 208µs.

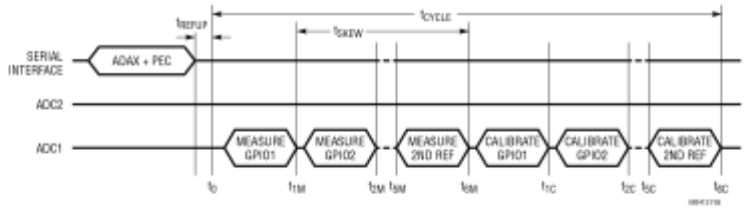


Figure 5. Timing for ADAX Command Measuring All GPIOs and 2nd Reference

Table 7. Conversion Times for ADAX Command Measuring All GPIOs and 2nd Reference in Different Modes

	CONVERSION TIMES (in µs)								
MODE	t_0	t_{1M}	t_{2M}	t_{5M}	t_{6M}	t_{1C}	t_{2C}	t_{5C}	t_{6C}
27kHz	0	57	103	243	290	432	568	975	1,113
14kHz	0	86	162	389	465	606	742	1,149	1,288
7kHz	0	144	278	680	814	1,072	1,324	2,080	2,335
3kHz	0	260	511	1,262	1,512	1,770	2,022	2,778	3,033
2kHz	0	493	976	2,425	2,908	3,166	3,418	4,175	4,430
26Hz	0	29,817	59,623	149,043	178,850	182,599	186,342	197,571	201,317

OPERATION

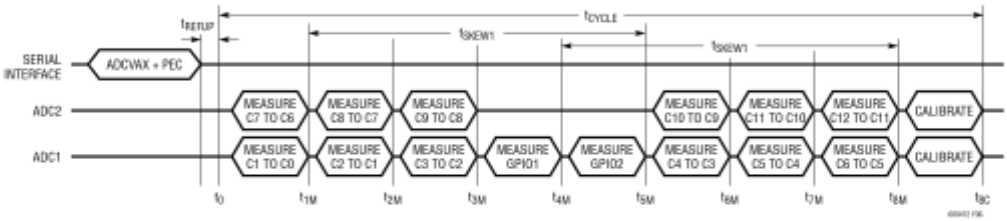


Figure 6. Timing of ADCVAX Command

Table 8. Conversion and Synchronization Times for ADCVAX Command in Different Modes

	CONVERSION TIMES (in µs)										SYNCHRONIZATION TIME (µs)
MODE	t_0	t_{1M}	t_{2M}	t_{3M}	t_{4M}	t_{5M}	t_{6M}	t_{7M}	t_{8M}	t_{8C}	t_{SKEW1}
27kHz	0	57	106	155	216	265	326	375	424	1,564	208
14kHz	0	86	161	237	320	396	479	555	630	1,736	310
7kHz	0	144	278	412	553	687	828	962	1,096	3,133	543
3kHz	0	260	511	761	1,018	1,269	1,526	1,777	2,027	4,064	1009
2kHz	0	493	976	1,459	1,949	2,432	2,923	3,406	3,888	5,925	1939
26Hz	0	29,817	59,623	89,430	119,244	149,051	178,864	208,671	238,478	268,442	119234

Table 8 shows the conversion and synchronization time for the ADCVAX command in different modes. The total conversion time for the command is given by t_{8C}.

DATA ACQUISITION SYSTEM DIAGNOSTICS

The battery monitoring data acquisition system is comprised of the multiplexers, ADCs, 1st reference, digital filters, and memory. To ensure long term reliable performance there are several diagnostic commands which can be used to verify the proper operation of these circuits.

Measuring Internal Device Parameters (ADSTAT Command)

The ADSTAT command is a diagnostic command that measures the following internal device parameters: sum of all cells (SOC), internal die temperature (ITMP), analog power supply (VA) and the digital power supply (VD). These parameters are described in the section below. All 6 ADC modes are available for these conversions. See the section on commands for the ADSTAT command format. Figure 7 illustrates the timing of the ADSTAT command measuring all 4 internal device parameters.

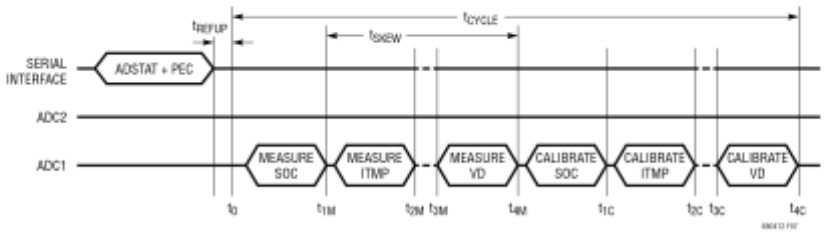


Figure 7. Timing for ADSTAT Command Measuring SOC, ITMP, VA, VD

OPERATION

Table 9 shows the conversion time of the ADSTAT command measuring all 4 internal parameters. t_{4C} indicates the total conversion time for the ADSTAT command.

Sum of Cells Measurement: The sum of all cells measurement is the voltage between C12 and C0 with a 20:1 attenuation. The 16-bit ADC value of sum of cells measurement (SOC) is stored in status register group A. Any potential difference between the C0 and V^- pins results in an error in the SOC measurement equal to this difference. From the SOC value, the sum of all cell voltage measurements is given by:

Sum of all Cells = SOC • 20 • 100µV

Internal Die Temperature: The ADSTAT command can measure the internal die temperature. The 16-bit ADC value of the die temperature measurement (ITMP) is stored in status register group A. From ITMP the actual die temperature is calculated using the expression:

Internal Die Temperature (°C) = (ITMP) • 100µV/(7.5mV)°C – 273°C

Power Supply Measurements: The ADSTAT command is also used to measure the analog power supply (V_{REG}) and digital power supply (V_{REGD}).

The 16-bit ADC value of the analog power supply measurement (VA) is stored in Status Register Group A. The 16-bit ADC value of the digital power supply measurement (VD) is stored in status register group B. From VA and VD, the power supply measurements are given by:

Analog power supply measurement (V_{REG}) = V_A • 100µV

Digital power supply measurement (V_{REGD}) = V_D • 100µV

The nominal range of V_{REG} is 4.5V to 5.5V. The nominal range of V_{REGD} is 2.7V to 3.6V.

Issuing an ADSTAT command with CHST = 100 runs an ADC measurement of just the digital supply (V_{REGD}). This is not recommended following an ADCV command. With large cell voltages, running the ADSTAT command with CST = 100 following an ADCV command with CH = 000 (all cells) can cause the LTC6804 to perform a power on reset. If using the ADSTAT command with CHST = 100, it is necessary to run an ADCV command with CH = 001 prior to running the ADSTAT command with CHST = 100. This charges the high voltage multiplexer to a low potential before the V_{REGD} measurement is executed. To save time, this sacrificial ADCV command run prior to running the V_{REGD} measurement can be executed in FAST mode (MD = 01).

Accuracy Check

Measuring an independent voltage reference is the best means to verify the accuracy of a data acquisition system. The LTC6804 contains a 2nd reference for this purpose. The ADAX command will initiate the measurement of the 2nd reference. The results are placed in auxiliary register group B. The range of the result depends on the ADC measurement accuracy and the accuracy of the 2nd reference, including thermal hysteresis and long term drift. Readings outside the range 2.985 to 3.015 indicate the system is out of its specified tolerance.

MUX Decoder Check

The diagnostic command DIAGN ensures the proper operation of each multiplexer channel. The command cycles through all channels and sets the MUXFAIL bit to 1 in status register group B if any channel decoder fails. The MUXFAIL bit is set to 0 if the channel decoder passes the

Table 9. Conversion Times for ADSTAT Command Measuring SOC, ITMP, VA, VD

	CONVERSION TIMES (in µs)								
MODE	t_0	t_{1M}	t_{2M}	t_{3M}	t_{4M}	t_{1C}	t_{2C}	t_{3C}	t_{4C}
27kHz	0	57	103	150	197	338	474	610	748
14kHz	0	86	162	237	313	455	591	726	865
7kHz	0	144	278	412	546	804	1,056	1,308	1,563
3kHz	0	260	511	761	1,011	1,269	1,522	1,774	2,028
2kHz	0	493	976	1,459	1,942	2,200	2,452	2,705	2,959
26Hz	0	29,817	59,623	89,430	119,237	122,986	126,729	130,472	134,218

680412fc

OPERATION

test. The MUXFAIL bit is also set to 1 on power-up (POR) or after a CLRSTAT command.

The DIAGN command takes about 400µs to complete if the core is in REFUP state and about 4.5ms to complete if the core is in STANDBY state. The polling methods described in the section Polling Methods can be used to determine the completion of the DIAGN command.

Digital Filter Check

The delta-sigma ADC is composed of a 1-bit pulse density modulator followed by a digital filter. A pulse density modulated bit stream has a higher percentage of 1s for higher analog input voltages. The digital filter converts this high frequency 1-bit stream into a single 16-bit word. This is why a delta-sigma ADC is often referred to as an oversampling converter.

The self test commands verify the operation of the digital filters and memory. Figure 8 illustrates the operation of the ADC during self test. The output of the 1-bit pulse density modulator is replaced by a 1-bit test signal. The test signal passes through the digital filter and is converted to a 16-bit value. The 1-bit test signal undergoes the same digital conversion as the regular 1-bit pulse from the modulator, so the conversion time for any self test command is exactly the same as the corresponding regular ADC conversion command. The 16-bit ADC value is stored in the same register groups as the regular ADC conversion command. The test signals are designed to place alternating one-zero patterns in the registers. Table 10 provides a list of the self test commands. If the digital filters and memory are working properly, then the registers will contain the values shown in Table 10. For more details see the section Commands.

ADC Clear Commands

LTC6804 has 3 clear commands – CLRCELL, CLRAUX and CLRSTAT. These commands clear the registers that store all ADC conversion results.

The CLRCELL command clears cell voltage register group A, B, C and D. All bytes in these registers are set to 0xFF by CLRCELL command.

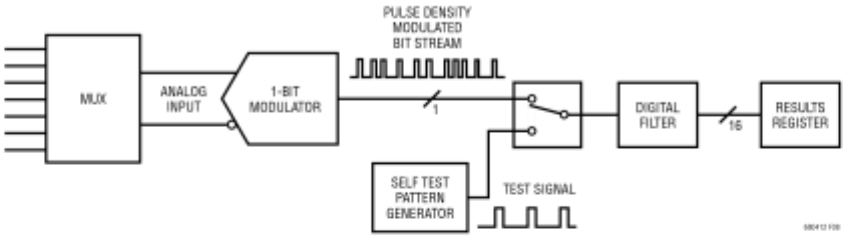


Figure 8. Operation of LTC6804 ADC Self Test

Table 10. Self Test Command Summary

COMMAND	SELF TEST OPTION	OUTPUT PATTERN IN DIFFERENT ADC MODES						RESULTS REGISTER GROUPS
		27kHz	14kHz	7kHz	3kHz	2kHz	26Hz	
CVST	ST[1:0]=01	0x9565	0x9553	0x9555	0x9555	0x9555	0x9555	C1V to C12V (CVA, CVB, CVC, CVD)
	ST[1:0]=10	0x6A9A	0x6AAC	0x6AAA	0x6AAA	0x6AAA	0x6AAA	
AXST	ST[1:0]=01	0x9565	0x9553	0x9555	0x9555	0x9555	0x9555	G1V to G5V, REF (AUXA, AUXB)
	ST[1:0]=10	0x6A9A	0x6AAC	0x6AAA	0x6AAA	0x6AAA	0x6AAA	
STATST	ST[1:0]=01	0x9565	0x9553	0x9555	0x9555	0x9555	0x9555	SOC, ITMP, VA, VD (STATA, STATB)
	ST[1:0]=10	0x6A9A	0x6AAC	0x6AAA	0x6AAA	0x6AAA	0x6AAA	

6804121c

OPERATION

The CLRAUX command clears auxiliary register group A and B. All bytes in these registers are set to 0xFF by CLRAUX command.

The CLRSTAT command clears status register group A and B except the REVCODE and RSVD bits in status register group B. A read back of REVCODE will return the revision code of the part. All OV flags, UV flags, MUXFAIL bit, and THSD bit in status register group B are set to 1 by CLRSTAT command. The THSD bit is set to 0 after RDSTATB command. The registers storing SOC, ITMP, VA and VD are all set to 0xFF by CLRSTAT command.

Open-Wire Check (ADOW Command)

The ADOW command is used to check for any open wires between the ADCs in the LTC6804 and the external cells. This command performs ADC conversions on the C pin inputs identically to the ADCV command, except two internal current sources sink or source current into the two C pins while they are being measured. The pull-up (PUP) bit of the ADOW command determines whether the current sources are sinking or sourcing 100µA.

The following simple algorithm can be used to check for an open wire on any of the 13 C pins (see Figure 9):

1) Run the 12-cell command ADOW with PUP = 1 at least twice. Read the cell voltages for cells 1 through 12 once at the end and store them in array $CELL_{PU(n)}$.
2) Run the 12-cell command ADOW with PUP = 0 at least twice. Read the cell voltages for cells 1 through 12 once at the end and store them in array $CELL_{PD(n)}$.
3) Take the difference between the pull-up and pull-down measurements made in above steps for cells 2-12: $CELL_{\Delta(n)} = CELL_{PU(n)} - CELL_{PD(n)}$.
4) For all values of n from 1 to 11: If $CELL_{\Delta(n+1)} < -400mV$, then C(n) is open. If the $CELL_{PU(1)} = 0.0000$, then C(0) is open. If the $CELL_{PD(12)} = 0.0000$, then C(12) is open.

The above algorithm detects open wires using normal mode conversions with as much as 10nF of capacitance remaining on the LTC6804 side of the open wire. However, if more external capacitance is on the open C pin, then the length of time that the open wire conversions are ran in steps 1 and 2 must be increased to give the 100µA current sources time to create a large enough difference for the algorithm to detect an open connection. This can be accomplished by running more than two ADOW commands in steps 1 and 2, or by using filtered mode conversions instead of normal mode conversions. Use Table 11 to determine how many conversions are necessary:

Table 11

	Number of ADOW Commands Required in Steps 1 and 2	
EXTERNAL C PIN CAPACITANCE	NORMAL MODE	FILTERED MODE
≤10nF	2	2
100nF	10	2
1µF	100	2
C	1+ROUNDUP(C/10nF)	2

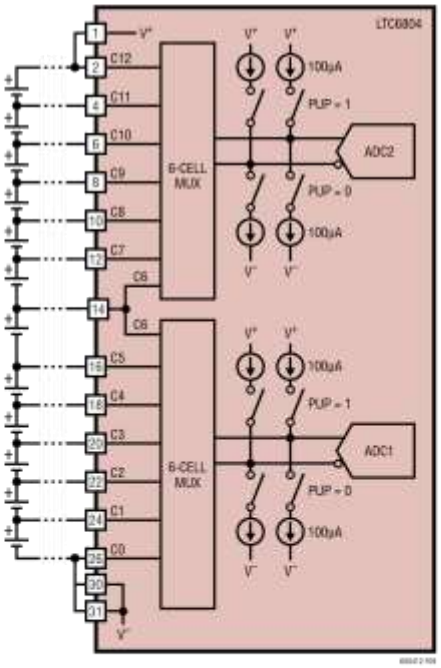


Figure 9. Open-Wire Detection Circuitry

OPERATION

Thermal Shutdown

To protect the LTC6804 from overheating, there is a thermal shutdown circuit included inside the IC. If the temperature detected on the die goes above approximately 150°C, the thermal shutdown circuit trips and resets the configuration register group to its default state. This turns off all discharge switches. When a thermal shutdown event has occurred, the THSD bit in status register group B will go high. This bit is cleared after a read operation has been performed on the status register group B (RDSTATB command). The CLRSTAT command sets the THSD bit high for diagnostic purposes, but does not reset the configuration register group.

Revision Code and Reserved Bits

The status register group B contains a 4-bit revision code and 2 reserved bits. If software detection of device revision is necessary, then contact the factory for details. Otherwise, the code can be ignored. In all cases, however, the values of all bits must be used when calculating the packet error code (PEC) on data reads.

WATCHDOG AND SOFTWARE DISCHARGE TIMER

When there is no wake-up signal (see Figure 21) for more than 2 seconds, the watchdog timer expires. This resets configuration register bytes CFGR0-CFGR3 in all cases. CFGR4 and CFGR5 are reset by the watchdog timer when the software timer is disabled. The WDT pin is pulled high by the external pull-up when the watchdog time elapses. The watchdog timer is always enabled and is reset by a qualified wake-up signal.

The software discharge timer is used to keep the discharge switches turned ON for programmable time duration. If the software timer is being used, the discharge switches are not turned OFF when the watchdog timer is activated.

To enable the software timer, SWTEN pin needs to be tied high to V_{REG} (Figure 10). The discharge switches can now be kept ON for the programmed time duration that is determined by the DCTO value written to the configuration register. Table 12 shows the various time settings and the corresponding DCTO value. Table 13 summarizes the status of the configuration register group after a watchdog timer or software timer event.

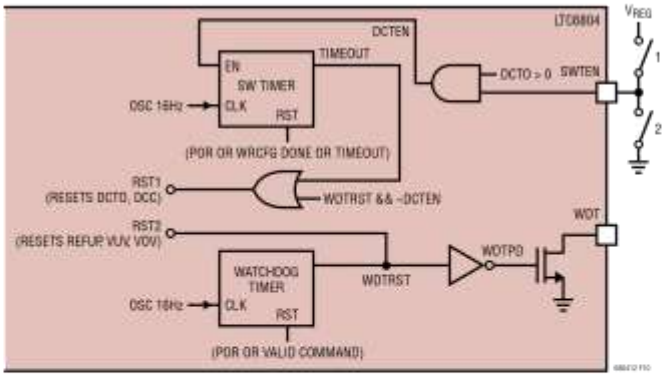


Figure 10. Watchdog and Software Discharge Timer

Table 12. DCTO Settings

DCTO	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Time Min	Disabled	0.5	1	2	3	4	5	10	15	20	30	40	60	75	90	120

OPERATION

Table 13

	WATCHDOG TIMER	SOFTWARE TIMER
SWTEN = 0, DCTO = XXXX	Resets CFGR0-5 When It Activates	Disabled
SWTEN = 1, DCTO = 0000	Resets CFGR0-5 When It Activates	Disabled
SWTEN = 1, DCTO ! = 0000	Resets CFGR0-3 When It Activates	Resets CFGR4-5 When It Fires

Unlike the watchdog timer, the software timer does not reset when there is a valid command. The software timer can only be reset after a valid WRCFG (write configuration register) command. There is a possibility that the software timer will expire in the middle of some commands.

If software timer activates in the middle of WRCFG command, the configuration register resets as per Table 14. However, at the end of the valid WRCFG command, the new data is copied to the configuration register. The new data is not lost when the software timer is activated.

If software timer activates in the middle of RDCFG command, the configuration register group resets as per Table 14. As a result, the read back data from bytes CRFG4 and CRFG5 could be corrupted.

I²C/SPI MASTER ON LTC6804 USING GPIOS

The I/O ports GPIO3, GPIO4 and GPIO5 on LTC6804-1 and LTC6804-2 can be used as an I²C or SPI master port to communicate to an I²C or SPI slave. In the case of an I²C master, GPIO4 and GPIO5 form the SDA and SCL ports of the I²C interface respectively. In the case of a SPI master, GPIO3, GPIO5 and GPIO4 become the chip select (CSBM), clock (SCKM) and data (SDIOM) ports of the SPI interface respectively. The SPI master on LTC6804 supports only SPI mode 3 (CHPA = 1, CPOL = 1).

Table 14

DCTO (READ VALUE)	TIME LEFT (MIN)
0	Disabled (or) Timer Has Timed Out
1	0 < Timer ≤ 0.5
2	0.5 < Timer ≤ 1
3	1 < Timer ≤ 2
4	2 < Timer ≤ 3
5	3 < Timer ≤ 4
6	4 < Timer ≤ 5
7	5 < Timer ≤ 10
8	10 < Timer ≤ 15
9	15 < Timer ≤ 20
A	20 < Timer ≤ 30
B	30 < Timer ≤ 40
C	40 < Timer ≤ 60
D	60 < Timer ≤ 75
E	75 < Timer ≤ 90
F	90 < Timer ≤ 120

The GPIOs are open drain outputs, so an external pull-up is required on these ports to operate as an I²C or SPI master. It is also important to write the GPIO bits to 1 in the CFG register group so these ports are not pulled low internally by the device.

COMM Register

LTC6804 has a 6-byte COMM register as shown in Table 15. This register stores all data and control bits required for I²C or SPI communication to a slave. The COMM register contains 3 bytes of data Dn[7:0] to be transmitted to or received from the slave device. ICOMn [3:0] specify control actions before transmitting/receiving the data byte. FCOMn [3:0] specify control actions after transmitting/receiving the data byte.

Table 15. COMM Register Memory Map

REGISTER	RD/WR	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
COMM0	RD/WR	ICOM0[3]	ICOM0[2]	ICOM0[1]	ICOM0[0]	D0[7]	D0[6]	D0[5]	D0[4]
COMM1	RD/WR	D0[3]	D0[2]	D0[1]	D0[0]	FCOM0[3]	FCOM0[2]	FCOM0[1]	FCOM0[0]
COMM2	RD/WR	ICOM1[3]	ICOM1[2]	ICOM1[1]	ICOM1[0]	D1[7]	D1[6]	D1[5]	D1[4]
COMM3	RD/WR	D1[3]	D1[2]	D1[1]	D1[0]	FCOM1[3]	FCOM1[2]	FCOM1[1]	FCOM1[0]
COMM4	RD/WR	ICOM2[3]	ICOM2[2]	ICOM2[1]	ICOM2[0]	D2[7]	D2[6]	D2[5]	D2[4]
COMM5	RD/WR	D2[3]	D2[2]	D2[1]	D2[0]	FCOM2[3]	FCOM2[2]	FCOM2[1]	FCOM2[0]

OPERATION

Table 16. Write Codes for ICOMn[3:0] and FCOMn[3:0] on I²C Master

CONTROL BITS	CODE	ACTION	DESCRIPTION
ICOMn[3:0]	0110	START	Generate a START Signal on I²C Port Followed By Data Transmission
	0001	STOP	Generate a STOP Signal on I²C port
	0000	BLANK	Proceed Directly to Data Transmission on I²C Port
	0111	No Transmit	Release SDA and SCL and Ignore the Rest of the Data
FCOMn[3:0]	0000	Master ACK	Master Generates an ACK Signal on Ninth Clock Cycle
	1000	Master NACK	Master Generates a NACK Signal on Ninth Clock Cycle
	1001	Master NACK + STOP	Master Generates a NACK Signal Followed by STOP Signal

Table 17. Write Codes for ICOMn[3:0] and FCOMn[3:0] on SPI Master

CONTROL BITS	CODE	ACTION	DESCRIPTION
ICOMn[3:0]	1000	CSBM low	Generates a CSBM Low Signal on SPI Port (GPIO3)
	1001	CSBM high	Generates a CSBM High Signal on SPI Port (GPIO3)
	1111	No Transmit	Releases the SPI Port and Ignores the Rest of the Data
FCOMn[3:0]	X000	CSBM low	Holds CSBM Low at the End of Byte Transmission
	1001	CSBM high	Transitions CSBM High at the End of Byte Transmission

If the bit ICOMn[3] in the COMM register is set to 1 the part becomes an I²C master and if the bit is set to 0 the part becomes a SPI master.

Table 16 describes the valid write codes for ICOMn[3:0] and FCOMn[3:0] and their behavior when using the part as an I²C master.

Table 17 describes the valid codes for ICOMn[3:0] and FCOMn[3:0] and their behavior when using the part as a SPI master.

Note that only the codes listed in Tables 16 and 17 are valid for ICOMn[3:0] and FCOMn[3:0]. Writing any other code that is not listed in Tables 16 and 17 to ICOMn[3:0] and FCOMn[3:0] may result in unexpected behavior on the I²C and SPI ports.

COMM Commands

Three commands help accomplish I²C or SPI communication to the slave device: WRCOMM, STCOMM, RDCOMM

WRCOMM Command: This command is used to write data to the COMM register. This command writes 6 bytes of data to the COMM register. The PEC needs to be written at the end of the data. If the PEC does not match, all data in the COMM register is cleared to 1's when CSB goes high. See the section Bus Protocols for more details on a write command format.

STCOMM Command: This command initiates I²C/SPI communication on the GPIO ports. The COMM register contains 3 bytes of data to be transmitted to the slave. During this command, the data bytes stored in the COMM register are transmitted to the slave I²C or SPI device and the data received from the I²C or SPI device is stored in the COMM register. This command uses GPIO4 (SDA) and GPIO5 (SCL) for I²C communication or GPIO3 (CSBM), GPIO4 (SDIOM) and GPIO5 (SCKM) for SPI communication.

The STCOMM command is to be followed by 24 clock cycles for each byte of data to be transmitted to the slave device while holding CSB low. For example, to transmit 3 bytes of data to the slave, send STCOMM command and its PEC followed by 72 clock cycles. Pull CSB high at the end of the 72 clock cycles of STCOMM command.

During I²C or SPI communication, the data received from the slave device is updated in the COMM register.

OPERATION

RDCOMM Command: The data received from the slave device can be read back from the COMM register using the RDCOMM command. The command reads back 6 bytes of data followed by the PEC. See the section Bus Protocols for more details on a read command format.

Table 18 describes the possible read back codes for ICOMn[3:0] and FCOMn[3:0] when using the part as an I^2C master. Dn[7:0] contains the data byte either transmitted by the I^2C master or received from the I^2C slave.

In case of the SPI master, the read back codes for ICOMn[3:0] and FCOMn[3:0] are always 0111 and 1111 respectively. Dn[7:0] contains the data byte either transmitted by the SPI master or received from the SPI slave.

Table 18. Read Codes for ICOMn[3:0] and FCOMn[3:0] on I^2C Master

CONTROL BITS	CODE	DESCRIPTION
ICOMn[3:0]	0110	Master Generated a START Signal
	0001	Master Generated a STOP Signal
	0000	Blank, SDA Was Held Low Between Bytes
	0111	Blank, SDA Was Held High Between Bytes
FCOMn[3:0]	0000	Master Generated an ACK Signal
	0111	Slave Generated an ACK Signal
	1111	Slave Generated a NACK Signal
	0001	Slave Generated an ACK Signal, Master Generated a STOP Signal
	1001	Slave Generated a NACK Signal, Master Generated a STOP Signal

Figure 11 illustrates the operation of LTC6804 as an I^2C or SPI master using the GPIOs.

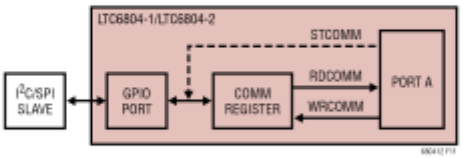


Figure 11. LTC6804 I^2C/SPI Master Using GPIOs

Any number of bytes can be transmitted to the slave in groups of 3 bytes using these commands. The GPIO ports will not get reset between different STCOMM commands. However, if the wait time between the commands is greater than 2 seconds, the watchdog will timeout and reset the ports to their default values.

To transmit several bytes of data using an I^2C master, a START signal is only required at the beginning of the entire data stream. A STOP signal is only required at the end of the data stream. All intermediate data groups can use a BLANK code before the data byte and an ACK/NACK signal as appropriate after the data byte. SDA and SCL will not get reset between different STCOMM commands.

To transmit several bytes of data using SPI master, a CSBM low signal is sent at the beginning of the 1st data byte. CSBM can be held low or taken high for intermediate data groups using the appropriate code on FCOMn[3:0]. A CSBM high signal is sent at the end of the last byte of data. CSBM, SDIOM and SCKM will not get reset between different STCOMM commands.

Figure 12 shows the 24 clock cycles following STCOMM command for an I^2C master in different cases. Note that if ICOMn[3:0] specified a STOP condition, after the STOP signal is sent, the SDA and SCL lines are held high and all data in the rest of the word is ignored. If ICOMn[3:0] is a NO TRANSMIT, both SDA and SCL lines are released, and rest of the data in the word is ignored. This is used when a particular device in the stack does not have to communicate to a slave.

Figure 13 shows the 24 clock cycles following STCOMM command for a SPI master. Similar to the I^2C master, if ICOMn[3:0] specified a CSBM HIGH or a NO TRANSMIT condition, the CSBM, SCKM and SDIOM lines of the SPI master are released and the rest of the data in the word is ignored.

OPERATION

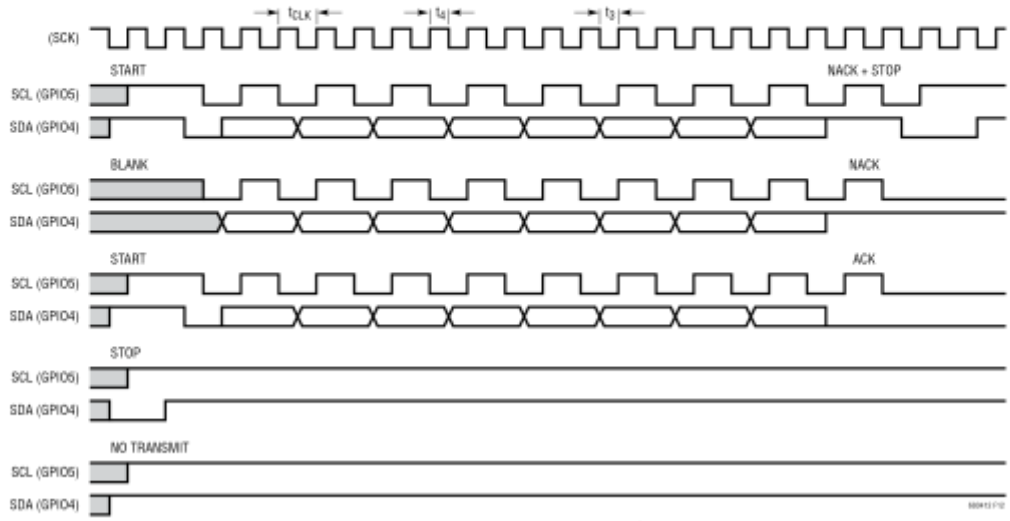


Figure 12. STCOMM Timing Diagram for an I²C Master

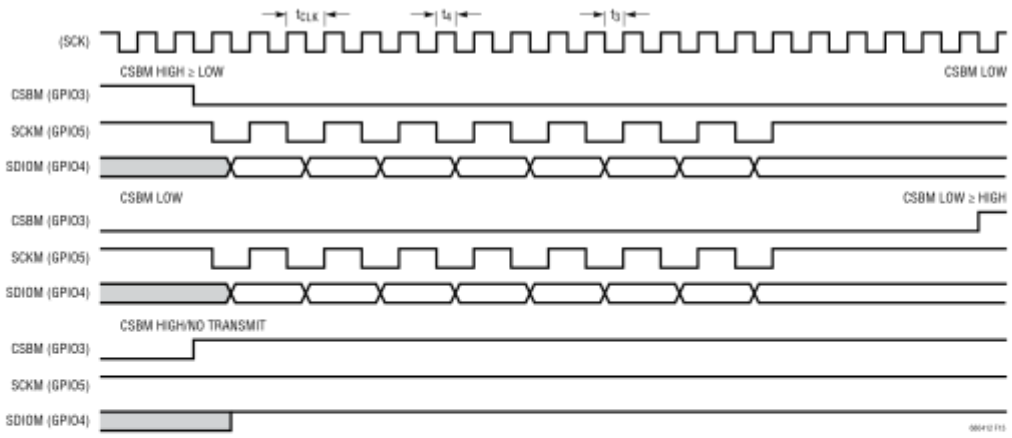


Figure 13. STCOMM Timing Diagram for a SPI Master

OPERATION

Timing Specifications of I²C and SPI master

The timing of the LTC6804 I²C or SPI master will be controlled by the timing of the communication at the LTC6804's primary SPI interface. Table 19 shows the I²C master timing relationship to the primary SPI clock. Table 20 shows the SPI master timing specifications.

Table 19. I²C Master Timing

I²C MASTER PARAMETER	TIMING RELATIONSHIP TO PRIMARY SPI INTERFACE	TIMING SPECIFICATIONS AT t_{CLK} = 1µs
SCL Clock Frequency	$1/(2 \cdot t_{CLK})$	Max 500kHz
t_{HD}; STA	t_3	Min 200ns
t_{LOW}	t_{CLK}	Min 1µs
t_{HIGH}	t_{CLK}	Min 1µs
t_{SU}; STA	$t_{CLK} + t_4$*	Min 1.03µs
t_{HD}; DAT	t_4*	Min 30ns
t_{SU}; DAT	t_3	Min 1µs
t_{SU}; STO	$t_{CLK} + t_4$*	Min 1.03µs
t_{BUF}	$3 \cdot t_{CLK}$	Min 3µs

*Note: When using isoSPI, t_4 is generated internally and is a minimum of 30ns. Also, $t_3 = t_{CLK} - t_4$. When using SPI, t_3 and t_4 are the low and high times of the SCK input, each with a specified minimum of 200ns.

SERIAL INTERFACE OVERVIEW

There are two types of serial ports on the LTC6804, a standard 4-wire serial peripheral interface (SPI) and a 2-wire isolated interface (isoSPI). Pins 41 through 44 are configurable as 2-wire or 4-wire serial port, based on the state of the ISOMD pin.

There are two versions of the LTC6804: the LTC6804-1 and the LTC6804-2. The LTC6804-1 is used in a daisy chain configuration, and the LTC6804-2 is used in an addressable bus configuration. The LTC6804-1 provides a second isoSPI interface using pins 45 through 48. The LTC6804-2 uses pins 45 through 48 to set the address of the device, by tying these pins to V^- or V_{REG}.

Table 20. SPI Master Timing

SPI MASTER PARAMETER	TIMING RELATIONSHIP TO PRIMARY SPI INTERFACE	TIMING SPECIFICATIONS AT t_{CLK} = 1µs
SDIOM Valid to SCKM Rising Setup	t_3	Min 200ns
SDIOM Valid from SCKM Rising Hold	$t_{CLK} + t_4$*	Min 1.03µs
SCKM Low	t_{CLK}	Min 1µs
SCKM High	t_{CLK}	Min 1µs
SCKM Period (SCKM_Low + SCKM_High)	$2 \cdot t_{CLK}$	Min 2µs
CSBM Pulse Width	$3 \cdot t_{CLK}$	Min 3µs
SCKM Rising to CSBM Rising	$5 \cdot t_{CLK} + t_4$*	Min 5.03µs
CSBM Falling to SCKM Falling	t_3	Min 200ns
CSBM Falling to SCKM Rising	$t_{CLK} + t_3$	Min 1.2µs
SCKM Falling to SDIOM Valid	Master requires < t_{CLK}	

*Note: When using isoSPI, t_4 is generated internally and is a minimum of 30ns. Also, $t_3 = t_{CLK} - t_4$. When using SPI, t_3 and t_4 are the low and high times of the SCK input, each with a specified minimum of 200ns.

OPERATION

4-WIRE SERIAL PERIPHERAL INTERFACE (SPI) PHYSICAL LAYER

External Connections

Connecting ISOMD to V^- configures serial Port A for 4-wire SPI. The SDO pin is an open drain output which requires a pull-up resistor tied to the appropriate supply voltage (Figure 14).

Timing

The 4-wire serial port is configured to operate in a SPI system using CPHA = 1 and CPOL = 1. Consequently, data on SDI must be stable during the rising edge of SCK. The timing is depicted in Figure 15. The maximum data rate is 1Mbps.

2-WIRE ISOLATED INTERFACE (isoSPI) PHYSICAL LAYER

The 2-wire interface provides a means to interconnect LTC6804 devices using simple twisted pair cabling. The interface is designed for low packet error rates when the cabling is subjected to high RF fields. Isolation is achieved through an external transformer.

Standard SPI signals are encoded into differential pulses. The strength of the transmission pulse and the threshold level of the receiver are set by two external resistors, R_{B1} and R_{B2}. The values of the resistors allow the user to trade off power dissipation for noise immunity.

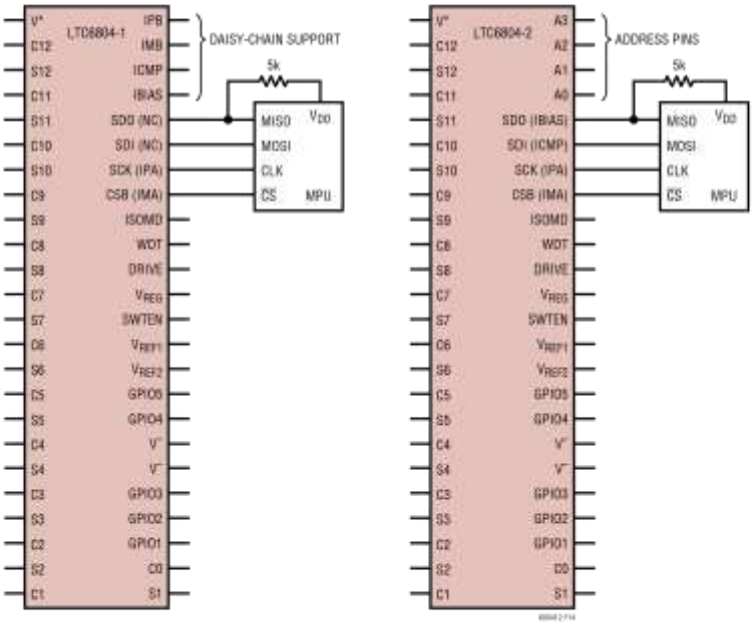


Figure 14. 4-Wire SPI Configuration

OPERATION

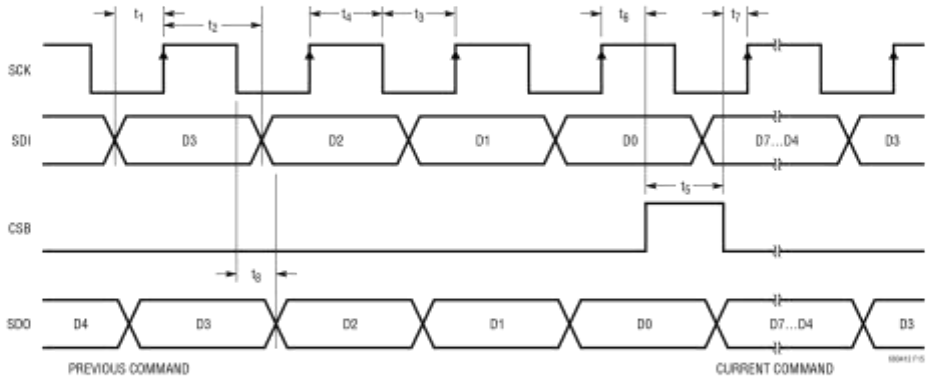


Figure 15. Timing Diagram of 4-Wire Serial Peripheral Interface

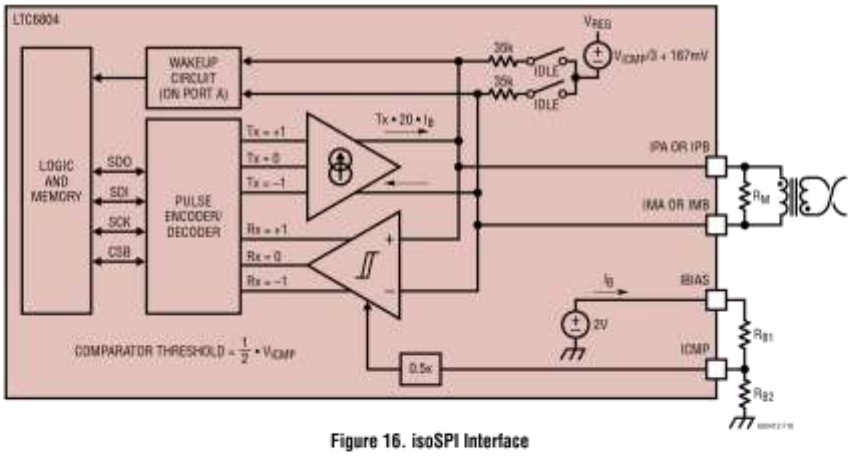


Figure 16. isoSPI Interface

Figure 16 illustrates how the isoSPI circuit operates. A 2V reference drives the IBIAS pin. External resistors R_{B1} and R_{B2} create the reference current I_B. This current sets the drive strength of the transmitter. R_{B1} and R_{B2} also form a voltage divider of the 2V reference at the ICMP pin. This sets the threshold voltage of the receiver circuit. Transmitted current pulses are converted into voltage by termination resistor R_M (in parallel with the characteristic impedance of the cable).

External Connections

The LTC6804-1 has 2 serial ports which are called Port B and Port A. Port B is always configured as a 2-wire interface (master). The final device in the daisy chain does not use this port, and it should be terminated into R_M. Port A is either a 2-wire or 4-wire interface (slave), depending on the connection of the ISOMD pin.

OPERATION

Figure 17a is an example of a robust interconnection of multiple identical PCBs, each containing one LTC6804-1. Note the termination in the final device in the daisy chain. The microprocessor is located on a separate PCB. To achieve 2-wire isolation between the microprocessor PCB and the 1st LTC6804-1 PCB, use the LTC6820 support IC. The LTC6820 is functionally equivalent to the diagram in Figure 16.

The LTC6804-2 has a single serial port (Port A) which can be 2-wire or 4-wire, depending on the state of the ISOMD pin. When configured for 2-wire communications, several devices can be connected in a multi-drop configuration, as shown in Figure 17b. The LTC6820 IC is used to interface the MPU (master) to the LTC6804-2's (slaves).

Using a Single LTC6804

When only one LTC6804 is needed, the LTC6804-2 is recommended. It does not have isoSPI Port B, so it requires fewer external components and consumes less power, especially when Port A is configured as a 4-wire interface.

However, the LTC6804-1 can be used as a single (non daisy-chained) device if the second isoSPI port (Port B) is properly biased and terminated, as shown in Figure 18c. ICMP should *not* be tied to GND, but can be tied directly to IBIAS. A bias resistance (2k to 20k) is required for IBIAS. Do *not* tie IBIAS directly to V_{REG} or V^-. Finally, IPB and IMB should be terminated into a 100Ω resistor (not tied to V_{REG} or V^-).

Selecting Bias Resistors

The adjustable signal amplitude allows the system to trade power consumption for communication robustness, and the adjustable comparator threshold allows the system to account for signal losses.

The isoSPI transmitter drive current and comparator voltage threshold are set by a resistor divider (R_{BIAS} = R_{B1} + R_{B2}) between the IBIAS and V^-. The divided voltage is connected to the ICMP pin which sets the comparator threshold to 1/2 of this voltage (V_{ICMP}). When either isoSPI interface is enabled (not IDLE) IBIAS is held at 2V, causing a current I_B to flow out of the IBIAS pin. The IP and IM pin drive currents are 20 • I_B.

As an example, if divider resistor R_{B1} is 2.8k and resistor R_{B2} is 1.21k (so that R_{BIAS} = 4k), then:

$$I_B = \frac{2V}{R_{B1}+R_{B2}} = 0.5mA$$

$$I_{DRV} = I_{IP} = I_{IM} = 20 \bullet I_B = 10mA$$

$$V_{ICMP} = 2V \bullet \frac{R_{B2}}{R_{B1}+R_{B2}} = I_B \bullet R_{B2} = 603mV$$

$$V_{TCMP} = 0.5 \bullet V_{ICMP} = 302mV$$

In this example, the pulse drive current I_{DRV} will be 10mA, and the receiver comparators will detect pulses with IP-IM amplitudes greater than ±302mV.

If the isolation barrier uses 1:1 transformers connected by a twisted pair and terminated with 120Ω resistors on each end, then the transmitted differential signal amplitude (±) will be:

$$V_A = I_{DRV} \bullet \frac{R_M}{2} = 0.6V$$

(This result ignores transformer and cable losses, which may reduce the amplitude).

isoSPI Pulse Detail

Two LTC6804 devices can communicate by transmitting and receiving differential pulses back and forth through an isolation barrier. The transmitter can output three voltage levels: $+V_A$, 0V, and $-V_A$. A positive output results from IP sourcing current and IM sinking current across load resistor R_M. A negative voltage is developed by IP sinking and IM sourcing. When both outputs are off, the load resistance forces the differential output to 0V.

To eliminate the DC signal component and enhance reliability, the isoSPI uses two different pulse lengths. This allows for four types of pulses to be transmitted, as shown in Table 21. A +1 pulse will be transmitted as a positive pulse followed by a negative pulse. A –1 pulse will be transmitted as a negative pulse followed by a positive pulse. The duration of each pulse is defined as $t_{1/2PW}$, since each is half of the required symmetric pair. (The total isoSPI pulse duration is 2 • $t_{1/2PW}$).

OPERATION

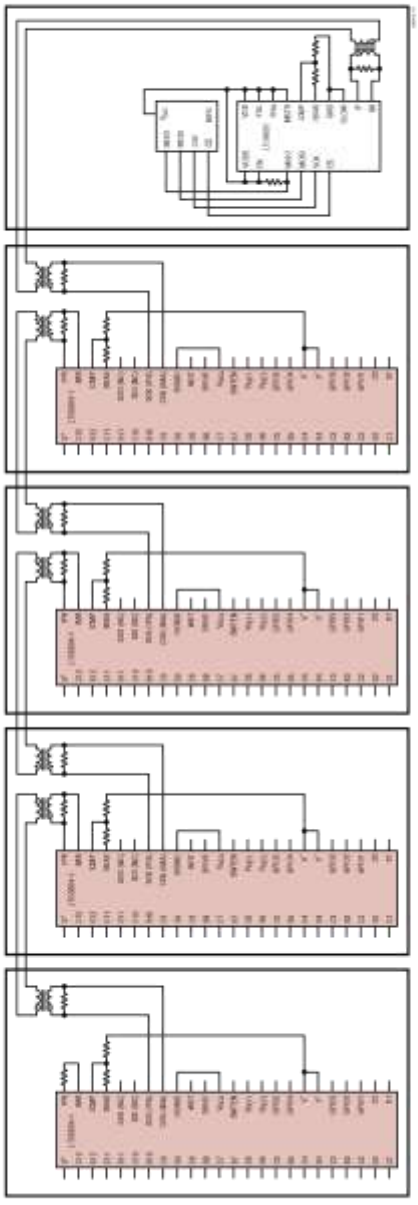


Figure 17a. Transformer-Isolated Daisy-Chain Configuration Using LTC6804-1

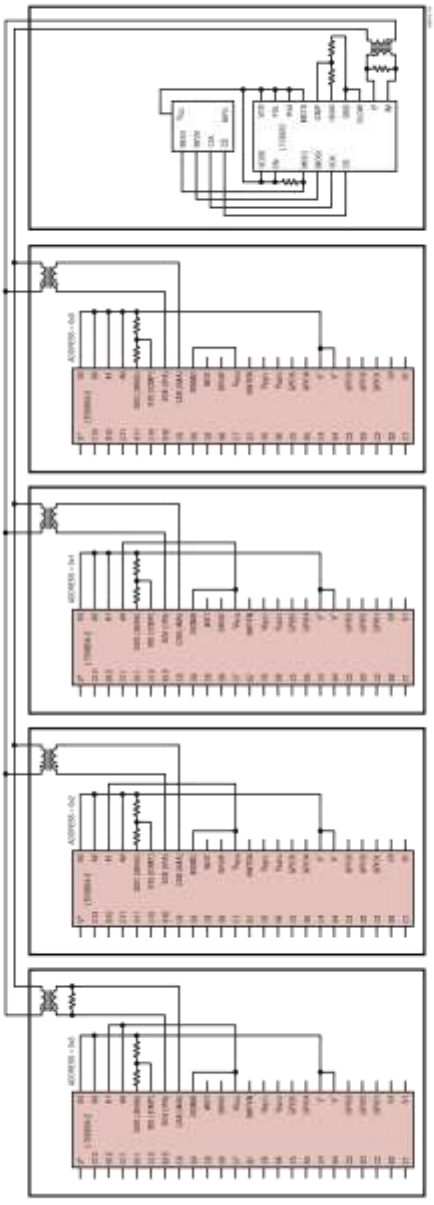


Figure 17b. Multi-Drop Configuration Using LTC6804-2

OPERATION

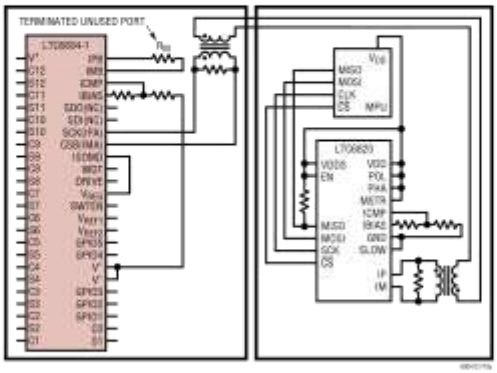


Figure 18a. Single-Device LTC6804-1 Using 2-Wire Port A

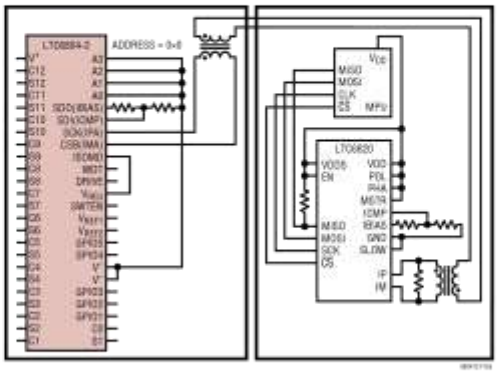


Figure 18b. Single-Device LTC6804-2 Using 2-Wire Port A

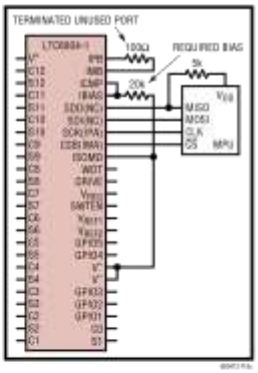


Figure 18c. Single-Device LTC6804-1 Using 4-Wire Port A

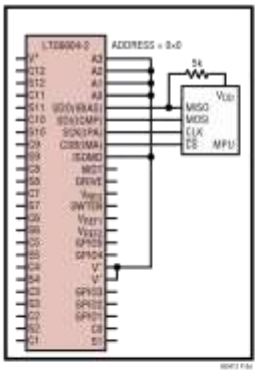


Figure 18d. Single-Device LTC6804-2 Using 4-Wire Port A

OPERATION

Table 21. isoSPI Pulse Types

PULSE TYPE	FIRST LEVEL ($t_{1/2PW}$)	SECOND LEVEL ($t_{1/2PW}$)	ENDING LEVEL
Long +1	$+V_A$ (150ns)	$-V_A$ (150ns)	0V
Long –1	$-V_A$ (150ns)	$+V_A$ (150ns)	0V
Short +1	$+V_A$ (50ns)	$-V_A$ (50ns)	0V
Short –1	$-V_A$ (50ns)	$+V_A$ (50ns)	0V

A host microcontroller does not have to generate isoSPI pulses to use this 2-wire interface. The first LTC6804 in the system can communicate to the microcontroller using the 4-wire SPI interface on its Port A, then daisy-chain to other LTC6804s using the 2-wire isoSPI interface on its Port B. Alternatively, an LTC6820 can be used to translate the SPI signals into isoSPI pulses.

LTC6804-1 Operation with Port A Configured for SPI

When the LTC6804-1 is operating with port A as an SPI (ISOMD = V⁻), the SPI detects one of four communication events: CSB falling, CSB rising, SCK rising with SDI = 0, and SCK rising with SDI = 1. Each event is converted into one of the four pulse types for transmission through the LTC6804-1 daisy chain. Long pulses are used to transmit CSB changes and short pulses are used to transmit data, as explained in Table 22.

Table 22. LTC6804-1 Port B (Master) isoSPI Port Function

COMMUNICATION EVENT (PORT A SPI)	TRANSMITTED PULSE (PORT B isoSPI)
CSB Rising	Long +1
CSB Falling	Long –1
SCK Rising Edge, SDI = 1	Short +1
SCK Rising Edge, SDI = 0	Short –1

On the other side of the isolation barrier (i.e. at the other end of the cable), the 2nd LTC6804 will have ISOMD = V_{REG}. Its Port A operates as a slave isoSPI interface. It receives each transmitted pulse and reconstructs the SPI signals internally, as shown in Table 23. In addition, during a READ command this port may transmit return data pulses.

Table 23. LTC6804-1 Port A (Slave) isoSPI Port Function

RECEIVED PULSE (PORT A isoSPI)	INTERNAL SPI PORT ACTION	RETURN PULSE
Long +1	Drive CSB High	None
Long –1	Drive CSB Low	
Short +1	1. Set SDI = 1 2. Pulse SCK	Short –1 Pulse if Reading a 0 bit
Short –1	1. Set SDI = 0 2. Pulse SCK	(No Return Pulse if Not in READ Mode or if Reading a 1 bit)

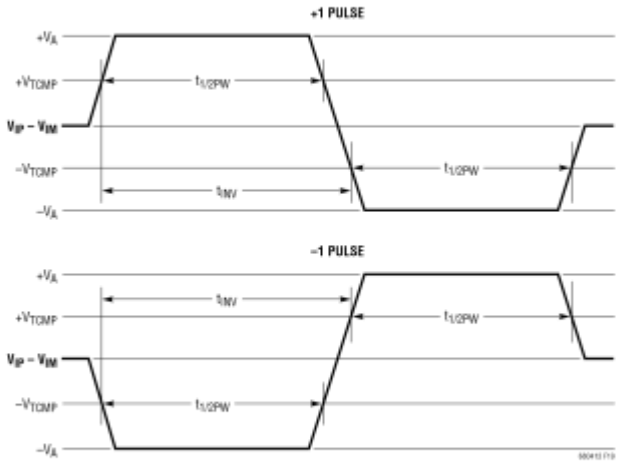


Figure 19. isoSPI Pulse Detail

OPERATION

The lower isoSPI port (Port A) never transmits long (CSB) pulses. Furthermore, a slave isoSPI port will only transmit short –1 pulses, never a +1 pulse. The master port recognizes a null response as a logic 1. This allows for multiple slave devices on a single cable without risk of collisions (Multidrop).

Figure 20 shows the isoSPI timing diagram for a READ command to daisy-chained LTC6804-1 parts. The ISOMD pin is tied to V^- on the bottom part so its Port A is configured as a SPI port (CSB, SCK, SDI and SDO). The isoSPI signals of three stacked devices are shown, labeled with the port (A or B) and part number. Note that ISO B1 and ISO A2 is actually the same signal, but shown on each end of the transmission cable that connects parts 1 and 2. Likewise, ISO B2 and ISO A3 is the same signal, but with the cable delay shown between parts 2 and 3.

Bits W_n-W_0 refers to the 16-bit command code and the 16-bit PEC of a READ command. At the end of bit W_0 the 3 parts decode the READ command and begin shifting out data which is valid on the next rising edge of clock SCK. Bits X_n-X_0 refer to the data shifted out by Part 1. Bits Y_n-Y_0 refer to the data shifted out by Part 2 and bits Z_n-Z_0 refer to the data shifted out by Part 3. All this data is read back from the SDO port on Part 1 in a daisy-chained fashion.

Waking Up the Serial Interface

The serial ports (SPI or isoSPI) will enter the low power IDLE state if there is no activity on Port A for a time of t_{IDLE}. The WAKEUP circuit monitors activity on pins 41 and 42.

If ISOMD = V^-, Port A is in SPI mode. Activity on the CSB or SCK pin will wake up the SPI interface. If ISOMD = V_{REG},

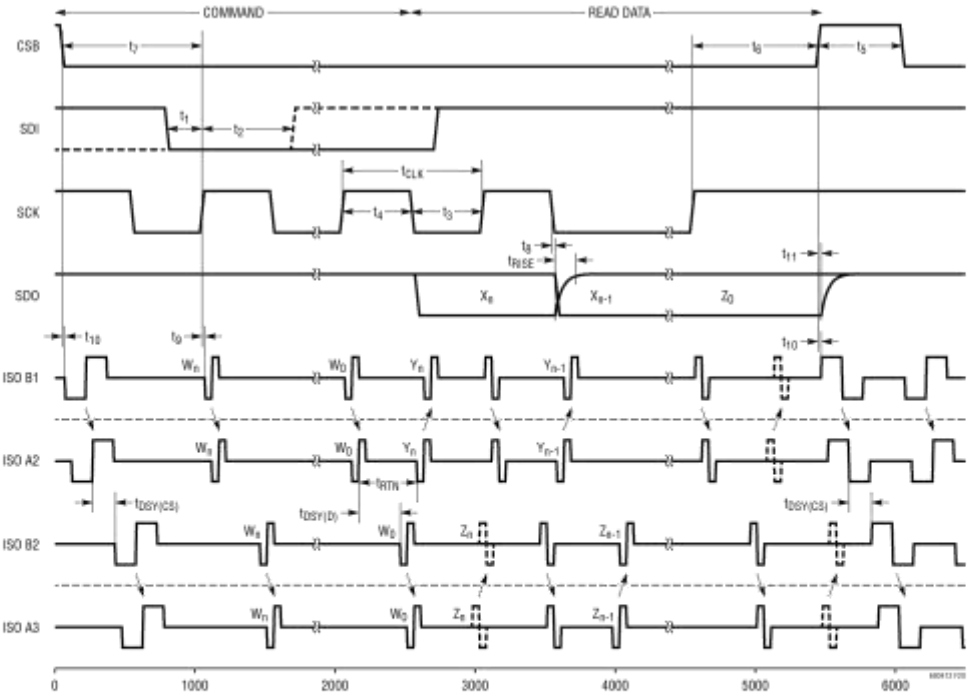


Figure 20. isoSPI Timing Diagram

6804121c

OPERATION

Port A is in isoSPI mode. Differential activity on IPA-IMB wakes up the isoSPI interface. The LTC6804 will be ready to communicate when the isoSPI state changes to READY within t_{WAKE} or t_{READY}, depending on the Core state (see Figure 1 and state descriptions for details.)

Figure 21 illustrates the timing and the functionally equivalent circuit. Common mode signals will not wake up the serial interface. The interface is designed to wake up after receiving a large signal single-ended pulse, or a low-amplitude symmetric pulse. The differential signal |SCK(IPA) – CSB(IMA)|, must be at least V_{WAKE} = 200mV for a minimum duration of t_{DWELL} = 240ns to qualify as a wake up signal that powers up the serial interface.

Waking a Daisy Chain — Method 1

The LTC6804-1 sends a Long +1 pulse on Port B after it is ready to communicate. In a daisy-chained configuration, this pulse wakes up the next device in the stack which will, in turn, wake up the next device. If there are 'N' devices in the stack, all the devices are powered up within the time N • t_{WAKE} or N • t_{READY}, depending on the Core State. For large stacks, the time N • t_{WAKE} may be equal to or larger than t_{IDLE}. In this case, after waiting longer than the time of N • t_{WAKE}, the host may send another dummy byte and wait for the time N • t_{READY}, in order to ensure that all devices are in the READY state.

Method 1 can be used when all devices on the daisy chain are in the IDLE state. This guarantees that they propagate the wake-up signal up the daisy chain. However, this method will fail to wake up all devices when a device in the middle of the chain is in the READY state instead of IDLE. When this happens, the device in READY state will not propagate the wake-up pulse, so the devices above it will remain IDLE. This situation can occur when attempting to wake up the daisy chain after only t_{IDLE} of idle time (some devices may be IDLE, some may not).

Waking a Daisy Chain — Method 2

A more robust wake-up method does not rely on the built-in wake-up pulse, but manually sends isoSPI traffic for enough time to wake the entire daisy chain. At minimum, a pair of long isoSPI pulses (–1 and +1) is needed for each device, separated by more than t_{READY} or t_{WAKE} (if the core state is STANDBY or SLEEP, respectively), but less than t_{IDLE}. This allows each device to wake up and propagate the next pulse to the following device. This method works even if some devices in the chain are not in the IDLE state. In practice, implementing method 2 requires toggling the CSB pin (of the LTC6820, or bottom LTC6804-1 with ISOMD = 0) to generate the long isoSPI pulses. Alternatively, dummy commands (such as RDCFG) can be executed to generate the long isoSPI pulses.

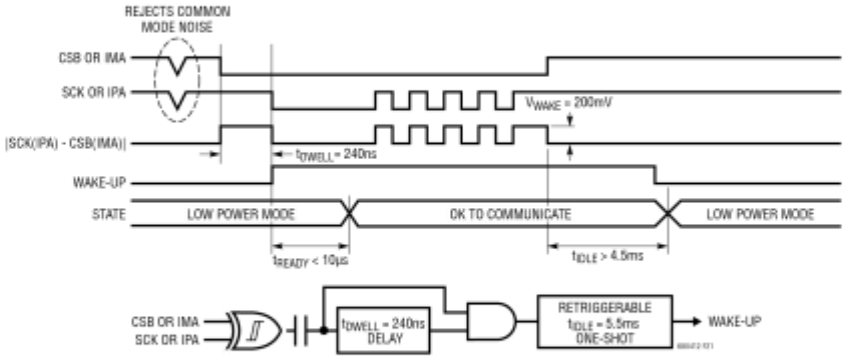


Figure 21. Wake-Up Detection and IDLE Timer

OPERATION

DATA LINK LAYER

All Data transfers on LTC6804 occur in byte groups. Every byte consists of 8 bits. Bytes are transferred with the most significant bit (MSB) first. CSB must remain low for the entire duration of a command sequence, including between a command byte and subsequent data. On a write command, data is latched in on the rising edge of CSB.

NETWORK LAYER

Packet Error Code

The packet error code (PEC) is a 15-bit cyclic redundancy check (CRC) value calculated for all of the bits in a register group in the order they are passed, using the initial PEC seed value of 000000000010000 and the following characteristic polynomial: x15 + x14 + x10 + x8 + x7 + x4 + x3 + 1. To calculate the 15-bit PEC value, a simple procedure can be established:

1. Initialize the PEC to 000000000010000 (PEC is a 15-bit register group)
2. For each bit DIN coming into the PEC register group, set

   IN0 = DIN XOR PEC [14]

   IN3 = IN0 XOR PEC [2]

   IN4 = IN0 XOR PEC [3]

   IN7 = IN0 XOR PEC [6]

   IN8 = IN0 XOR PEC [7]

   IN10 = IN0 XOR PEC [9]

   IN14 = IN0 XOR PEC [13]

3. Update the 15-bit PEC as follows

   PEC [14] = IN14,

   PEC [13] = PEC [12],

   PEC [12] = PEC [11],

   PEC [11] = PEC [10],

   PEC [10] = IN10,

   PEC [9] = PEC [8],

   PEC [8] = IN8,

   PEC [7] = IN7,

   PEC [6] = PEC [5],

   PEC [5] = PEC [4],

   PEC [4] = IN4,

   PEC [3] = IN3,

   PEC [2] = PEC [1],

   PEC [1] = PEC [0],

   PEC [0] = IN0

4. Go back to step 2 until all the data is shifted. The final PEC (16 bits) is the 15-bit value in the PEC register with a 0 bit appended to its LSB

Figure 22 illustrates the algorithm described above. An example to calculate the PEC for a 16-bit word (0x0001) is listed in Table 24. The PEC for 0x0001 is computed as 0x3D6E after stuffing a 0 bit at the LSB. For longer data streams, the PEC is valid at the end of the last bit of data sent to the PEC register.

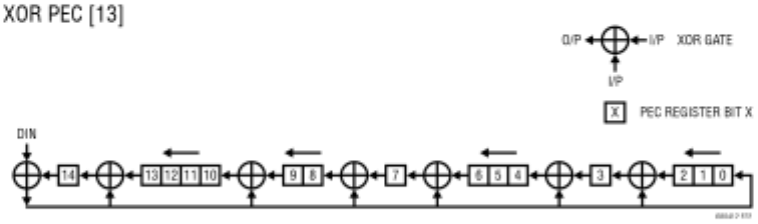


Figure 22. 15-Bit PEC Computation Circuit

OPERATION

LTC6804 calculates PEC for any command or data received and compares it with the PEC following the command or data. The command or data is regarded as valid only if the PEC matches. LTC6804 also attaches the calculated PEC at the end of the data it shifts out. Table 25 shows the format of PEC while writing to or reading from LTC6804.

While writing any command to LTC6804, the command bytes CMD0 and CMD1 (See Table 32 and Table 33) and the PEC bytes PEC0 and PEC1 are sent on Port A in the following order:

CMD0, CMD1, PEC0, PEC1

After a broadcast write command to daisy-chained LTC6804-1 devices, data is sent to each device followed by the PEC. For example, when writing the configuration register group to two daisy-chained devices (primary device P, stacked device S), the data will be sent to the primary device on Port A in the following order:

CFGR0(S), … , CFGR5(S), PEC0(S), PEC1(S), CFGR0(P), …, CFGR5(P), PEC0(P), PEC1(P)

After a read command for daisy-chained devices, each device shifts out its data and the PEC that it computed for its data on Port A followed by the data received on Port B. For example, when reading status register group B from two daisy-chained devices (primary device P, stacked device S), the primary device sends out data on port A in the following order:

STBR0(P), …, STBR5(P), PEC0(P), PEC1(P), STBR0(S), … , STBR5(S), PEC0(S), PEC1(S)

Table 24. PEC Calculation for 0x0001

PEC[14]	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	**0**
PEC[13]	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	**0**
PEC[12]	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	1	**1**
PEC[11]	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	1	1	**1**
PEC[10]	0	0	0	0	0	0	1	0	0	0	0	1	1	0	1	1	1	**1**
PEC[9]	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	1	**1**
PEC[8]	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0	**0**
PEC[7]	0	0	0	1	0	0	0	0	0	0	0	1	1	1	0	1	1	**1**
PEC[6]	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	**0**
PEC[5]	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	**1**
PEC[4]	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	**1**
PEC[3]	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	**0**
PEC[2]	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	**1**
PEC[1]	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	**1**
PEC[0]	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	**1**
IN14	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0		**0**
IN10	0	0	0	0	0	1	0	0	0	0	1	1	0	1	1	1		**PEC Word**
IN8	0	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0		
IN7	0	0	1	0	0	0	0	0	0	0	1	1	1	0	1	1		
IN4	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1		
IN3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0		
IN0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1		
DIN	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1		
Clock Cycle	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	

OPERATION

Table 25. Write/Read PEC Format

NAME	RD/WR	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
PEC0	RD/WR	PEC[14]	PEC[13]	PEC[12]	PEC[11]	PEC[10]	PEC[9]	PEC[8]	PEC[7]
PEC1	RD/WR	PEC[6]	PEC[5]	PEC[4]	PEC[3]	PEC[2]	PEC[1]	PEC[0]	0

Broadcast vs Address Commands

CONFIGURATION		TYPE OF COMMAND		
DEVICE	INTERFACE	READ	WRITE	POLL
LTC6804-2 (Address/Parallel)	SPI	Address-Only	Address or Broadcast	Address or Broadcast
	isoSPI			Address-Only†
LTC6804-1 (Daisy-Chain)	SPI or isoSPI	Broadcast-Only		N/A

†The LTC6804-2 will not return data pulses when using broadcast commands in isoSPI mode. Therefore, ADC commands will execute, but polling will not work.

Address Commands (LTC6804-2 Only)

An address command is one in which only the addressed device on the bus responds. Address commands are used only with LTC6804-2 parts. All commands are compatible with addressing. See Bus Protocols for Address command format.

Broadcast Commands (LTC6804-1 or LTC6804-2)

A broadcast command is one to which all devices on the bus will respond, regardless of device address. This command format can be used with LTC6804-1 and LTC6804-2 parts. See Bus Protocols for Broadcast command format. With broadcast commands all devices can be sent commands simultaneously.

In parallel (LTC6804-2) configurations, broadcast commands are useful for initiating ADC conversions or for sending write commands when all parts are being written with the same data. The polling function (automatic at the end of ADC commands, or manual using the PLADC command) can also be used with broadcast commands, but only with parallel SPI interfaces. Polling is not compatible with parallel isoSPI. Likewise, broadcast read commands should not be used in a parallel configuration (either SPI or isoSPI).

Daisy-chained (LTC6804-1) configurations support broadcast commands only, because they have no addressing. All devices in the chain receive the command bytes simultaneously. For example, to initiate ADC conversions in a stack of devices, a single ADCV command is sent, and all devices will start conversions at the same time. For read and write commands, a single command is sent, and then the stacked devices effectively turn into a cascaded shift register, in which data is shifted through each device to the next device in the stack. See the Serial Programming Examples section.

Polling Methods

The simplest method to determine ADC completion is for the controller to start an ADC conversion and wait for the specified conversion time to pass before reading the results. Polling is not supported with daisy-chain communication (LTC6804-1).

In parallel configurations that communicate in SPI mode (ISOMD pin tied low), there are two methods of polling. The first method is to hold CSB low after an ADC conversion command is sent. After entering a conversion command, the SDO line is driven low when the device is busy performing conversions (Figure 23). SDO is pulled high when the device completes conversions. However, the SDO will also go back high when CSB goes high even if the device has not completed the conversion. An addressed device drives the SDO line based on its status alone. A problem with this method is that the controller is not free to do other serial communication while waiting for ADC conversions to complete.

The next method overcomes this limitation. The controller can send an ADC start command, perform other tasks, and then send a poll ADC converter status (PLADC) command to determine the status of the ADC conversions (Figure 24). After entering the PLADC command, SDO will go low if the device is busy performing conversions. SDO is pulled high at the end of conversions. However, the SDO will also

6804121c

OPERATION

go high when CSB goes high even if the device has not completed the conversion. See Programming Examples on how to use the PLADC command with devices in parallel configuration.

In parallel configurations that communicate in isoSPI mode, the low side port transmits a data pulse only in response to a master isoSPI pulse received by it. So, after entering an address command in either method of polling described above, isoSPI data pulses are sent to the part to update the conversion status. These pulses can be sent using LTC6820 by simply clocking its SCK pin. In response to this pulse, the LTC6804-2 returns an isoSPI pulse if it is still busy performing conversions and does not return a pulse if it has completed conversions. If a CSB high isoSPI pulse is sent to the LTC6804-2, it exits the polling command. Note that broadcast poll commands are not compatible with parallel isoSPI.

Bus Protocols

Protocol Format: The protocol formats for both broadcast and address commands are depicted in Table 27 through Table 31. Table 26 is the key for reading the protocol diagrams.

Table 26. Protocol Key

CMD0	First Command Byte (See Tables 32 and 33)
CMD1	Second Command Byte (See Tables 32 and 33)
PEC0	First PEC Byte (See Table 25)
PEC1	Second PEC Byte (See Table 25)
n	Number of Bytes
...	Continuation of Protocol
	Master to Slave
	Slave to Master

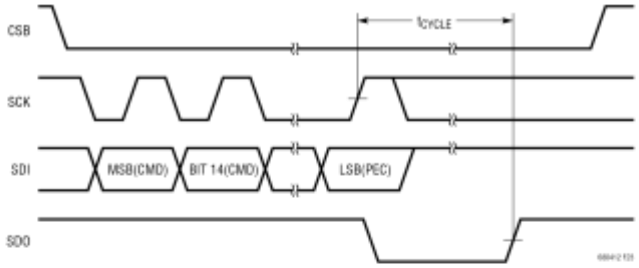


Figure 23. SDO Polling After an ADC Conversion Command

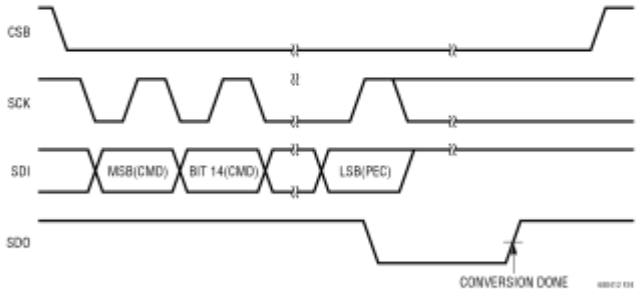


Figure 24. SDO Polling Using PLADC Command

OPERATION

Command Format: The formats for the broadcast and address commands are shown in Table 32 and Table 33 respectively. The 11-bit command code CC[10:0] is the same for a broadcast or an address command. A list of all the command codes is shown in Table 34. A broadcast command has a value 0 for CMD0[7] through CMD0[3]. An address command has a value 1 for CMD0[7] followed by the 4-bit address of the device (a3, a2, a1, a0) in bits CMD0[6:3]. An addressed device will respond to an address command only if the physical address of the device on pins A3 to A0 match the address specified in the address command. The PEC for broadcast and address commands must be computed on the entire 16-bit command (CMD0 and CMD1).

Commands

Table 34 lists all the commands and its options for both LTC6804-1 and LTC6804-2

Table 27. Broadcast/Address Poll Command

8	8	8	8	
CMD0	CMD1	PEC0	PEC1	Poll Data

Table 28. Broadcast Write Command (LTC6804-1)

8	8	8	8	8		8	8	8	8		8
CMD0	CMD1	PEC0	PEC1	Data Byte Low	...	Data Byte High	PEC0	PEC1	Shift Byte 1	...	Shift Byte *n*

Table 29.Broadcast/Address Write Command (LTC6804-2)

8	8	8	8	8		8	8	8
CMD0	CMD1	PEC0	PEC1	Data Byte Low	...	Data Byte High	PEC0	PEC1

Table 30. Broadcast Read Command (LTC6804-1)

8	8	8	8	8		8	8	8	8		8
CMD0	CMD1	PEC0	PEC1	Data Byte Low	...	Data Byte High	PEC0	PEC1	Shift Byte 1	...	Shift Byte *n*

Table 31. Address Read Command (LTC6804-2)

8	8	8	8	8		8	8	8
CMD0	CMD1	PEC0	PEC1	Data Byte Low	...	Data Byte High	PEC0	PEC1

Table 32. Broadcast Command Format

NAME	RD/WR	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
CMD0	WR	0	0	0	0	0	CC[10]	CC[9]	CC[8]
CMD1	WR	CC[7]	CC[6]	CC[5]	CC[4]	CC[3]	CC[2]	CC[1]	CC[0]

Table 33. Address Command Format

NAME	RD/WR	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
CMD0	WR	1	a3*	a2*	a1*	a0*	CC[10]	CC[9]	CC[8]
CMD1	WR	CC[7]	CC[6]	CC[5]	CC[4]	CC[3]	CC[2]	CC[1]	CC[0]

*ax is Address Bit x

OPERATION

Table 34. Command Codes

COMMAND DESCRIPTION	NAME	CC[10:0] - COMMAND CODE										
		10	9	8	7	6	5	4	3	2	1	0
Write Configuration Register Group	WRCFG	0	0	0	0	0	0	0	0	0	0	1
Read Configuration Register Group	RDCFG	0	0	0	0	0	0	0	0	0	1	0
Read Cell Voltage Register Group A	RDCVA	0	0	0	0	0	0	0	0	1	0	0
Read Cell Voltage Register Group B	RDCVB	0	0	0	0	0	0	0	0	1	1	0
Read Cell Voltage Register Group C	RDCVC	0	0	0	0	0	0	0	1	0	0	0
Read Cell Voltage Register Group D	RDCVD	0	0	0	0	0	0	0	1	0	1	0
Read Auxiliary Register Group A	RDAUXA	0	0	0	0	0	0	0	1	1	0	0
Read Auxiliary Register Group B	RDAUXB	0	0	0	0	0	0	0	1	1	1	0
Read Status Register Group A	RDSTATA	0	0	0	0	0	0	1	0	0	0	0
Read Status Register Group B	RDSTATB	0	0	0	0	0	0	1	0	0	1	0
Start Cell Voltage ADC Conversion and Poll Status	ADCV	0	1	MD[1]	MD[0]	1	1	DCP	0	CH[2]	CH[1]	CH[0]
Start Open Wire ADC Conversion and Poll Status	ADOW	0	1	MD[1]	MD[0]	PUP	1	DCP	1	CH[2]	CH[1]	CH[0]
Start Self-Test Cell Voltage Conversion and Poll Status	CVST	0	1	MD[1]	MD[0]	ST[1]	ST[0]	0	0	1	1	1
Start GPIOs ADC Conversion and Poll Status	ADAX	1	0	MD[1]	MD[0]	1	1	0	0	CHG [2]	CHG [1]	CHG [0]
Start Self-Test GPIOs Conversion and Poll Status	AXST	1	0	MD[1]	MD[0]	ST[1]	ST[0]	0	0	1	1	1
Start Status group ADC Conversion and Poll Status	ADSTAT	1	0	MD[1]	MD[0]	1	1	0	1	CHST [2]	CHST [1]	CHST [0]
Start Self-Test Status group Conversion and Poll Status	STATST	1	0	MD[1]	MD[0]	ST[1]	ST[0]	0	1	1	1	1
Start Combined Cell Voltage and GPIO1, GPIO2 Conversion and Poll Status	ADCVAX	1	0	MD[1]	MD[0]	1	1	DCP	1	1	1	1
Clear Cell Voltage Register Group	CLRCELL	1	1	1	0	0	0	1	0	0	0	1
Clear Auxiliary Register Group	CLRAUX	1	1	1	0	0	0	1	0	0	1	0
Clear Status Register Group	CLRSTAT	1	1	1	0	0	0	1	0	0	1	1
Poll ADC Conversion Status	PLADC	1	1	1	0	0	0	1	0	1	0	0
Diagnose MUX and Poll Status	DIAGN	1	1	1	0	0	0	1	0	1	0	1
Write COMM Register Group	WRCOMM	1	1	1	0	0	1	0	0	0	0	1
Read COMM Register Group	RDCOMM	1	1	1	0	0	1	0	0	0	1	0
Start I^2C/SPI Communication	STCOMM	1	1	1	0	0	1	0	0	0	1	1

OPERATION

Table 35. Command Bit Descriptions

NAME	DESCRIPTION	VALUES							
MD[1:0]	ADC Mode	MD	ADCOPT(CFGR0[0]) = 0			ADCOPT (CFGR0[0]) = 1			
		01	27kHz Mode (Fast)			14kHz Mode			
		10	7kHz Mode (Normal)			3kHz Mode			
		11	26Hz Mode (Filtered)			2kHz Mode			
DCP	Discharge Permitted	DCP							
		0	Discharge Not Permitted						
		1	Discharge Permitted						
CH[2:0]	Cell Selection for ADC Conversion			Total Conversion Time in the 6 ADC Modes					
		CH		27kHz	14kHz	7kHz	3kHz	2kHz	26Hz
		000	All Cells	1.1ms	1.3ms	2.3ms	3.0ms	4.4ms	201ms
		001	Cell 1 and Cell 7	201µs	230µs	405µs	501µs	754µs	34ms
		010	Cell 2 and Cell 8						
		011	Cell 3 and Cell 9						
		100	Cell 4 and Cell 10						
		101	Cell 5 and Cell 11						
		110	Cell 6 and Cell 12						
PUP	Pull-Up/Pull-Down Current for Open-Wire Conversions	PUP							
		0	Pull-Down Current						
		1	Pull-Up Current						
ST[1:0]	Self-Test Mode Selection			Self-Test Conversion Result					
		ST		27kHz	14kHz	7kHz	3kHz	2kHz	26Hz
		01	Self Test 1	0x9565	0x9553	0x9555	0x9555	0x9555	0x9555
		10	Self test 2	0x6A9A	0x6AAC	0x6AAA	0x6AAA	0x6AAA	0x6AAA
CHG[2:0]	GPIO Selection for ADC Conversion			Total Conversion Time in the 6 ADC Modes					
		CHG		27kHz	14kHz	7kHz	3kHz	2kHz	26Hz
		000	GPIO 1-5, 2nd Ref	1.1ms	1.3ms	2.3ms	3.0ms	4.4ms	201ms
		001	GPIO 1	201µs	230µs	405µs	501µs	754µs	34ms
		010	GPIO 2						
		011	GPIO 3						
		100	GPIO 4						
		101	GPIO 5						
		110	2nd Reference						
CHST[2:0]*	Status Group Selection			Total Conversion Time in the 6 ADC Modes					
		CHST		27kHz	14kHz	7kHz	3kHz	2kHz	26Hz
		000	SOC, ITMP, VA, VD	748µs	865µs	1.6ms	2.0ms	3.0ms	134ms
		001	SOC	201µs	230µs	405µs	501µs	754µs	34ms
		010	ITMP						
		011	VA						
		100	VD**						

*Note: Valid options for CHST in ADSTAT command are 0-4. If CHST is set to 5/6 in ADSTAT command, the LTC6804 treats it like ADAX command with CHG = 5/6.
**The use of the ADSTAT command with CHST = 100 is not recommended unless special care is taken. See the Data Acquisition System Diagnostics section for more details.

OPERATION

Table 36. Configuration Register Group

REGISTER	RD/WR	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
CFGR0	RD/WR	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	REFON	SWTRD	ADCOPT
CFGR1	RD/WR	VUV[7]	VUV[6]	VUV[5]	VUV[4]	VUV[3]	VUV[2]	VUV[1]	VUV[0]
CFGR2	RD/WR	VOV[3]	VOV[2]	VOV[1]	VOV[0]	VUV[11]	VUV[10]	VUV[9]	VUV[8]
CFGR3	RD/WR	VOV[11]	VOV[10]	VOV[9]	VOV[8]	VOV[7]	VOV[6]	VOV[5]	VOV[4]
CFGR4	RD/WR	DCC8	DCC7	DCC6	DCC5	DCC4	DCC3	DCC2	DCC1
CFGR5	RD/WR	DCTO[3]	DCTO[2]	DCTO[1]	DCTO[0]	DCC12	DCC11	DCC10	DCC9

Table 37. Cell Voltage Register Group A

REGISTER	RD/WR	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
CVAR0	RD	C1V[7]	C1V[6]	C1V[5]	C1V[4]	C1V[3]	C1V[2]	C1V[1]	C1V[0]
CVAR1	RD	C1V[15]	C1V[14]	C1V[13]	C1V[12]	C1V[11]	C1V[10]	C1V[9]	C1V[8]
CVAR2	RD	C2V[7]	C2V[6]	C2V[5]	C2V[4]	C2V[3]	C2V[2]	C2V[1]	C2V[0]
CVAR3	RD	C2V[15]	C2V[14]	C2V[13]	C2V[12]	C2V[11]	C2V[10]	C2V[9]	C2V[8]
CVAR4	RD	C3V[7]	C3V[6]	C3V[5]	C3V[4]	C3V[3]	C3V[2]	C3V[1]	C3V[0]
CVAR5	RD	C3V[15]	C3V[14]	C3V[13]	C3V[12]	C3V[11]	C3V[10]	C3V[9]	C3V[8]

Table 38. Cell Voltage Register Group B

REGISTER	RD/WR	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
CVBR0	RD	C4V[7]	C4V[6]	C4V[5]	C4V[4]	C4V[3]	C4V[2]	C4V[1]	C4V[0]
CVBR1	RD	C4V[15]	C4V[14]	C4V[13]	C4V[12]	C4V[11]	C4V[10]	C4V[9]	C4V[8]
CVBR2	RD	C5V[7]	C5V[6]	C5V[5]	C5V[4]	C5V[3]	C5V[2]	C5V[1]	C5V[0]
CVBR3	RD	C5V[15]	C5V[14]	C5V[13]	C5V[12]	C5V[11]	C5V[10]	C5V[9]	C5V[8]
CVBR4	RD	C6V[7]	C6V[6]	C6V[5]	C6V[4]	C6V[3]	C6V[2]	C6V[1]	C6V[0]
CVBR5	RD	C6V[15]	C6V[14]	C6V[13]	C6V[12]	C6V[11]	C6V[10]	C6V[9]	C6V[8]

Table 39. Cell Voltage Register Group C

REGISTER	RD/WR	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
CVCR0	RD	C7V[7]	C7V[6]	C7V[5]	C7V[4]	C7V[3]	C7V[2]	C7V[1]	C7V[0]
CVCR1	RD	C7V[15]	C7V[14]	C7V[13]	C7V[12]	C7V[11]	C7V[10]	C7V[9]	C7V[8]
CVCR2	RD	C8V[7]	C8V[6]	C8V[5]	C8V[4]	C8V[3]	C8V[2]	C8V[1]	C8V[0]
CVCR3	RD	C8V[15]	C8V[14]	C8V[13]	C8V[12]	C8V[11]	C8V[10]	C8V[9]	C8V[8]
CVCR4	RD	C9V[7]	C9V[6]	C9V[5]	C9V[4]	C9V[3]	C9V[2]	C9V[1]	C9V[0]
CVCR5	RD	C9V[15]	C9V[14]	C9V[13]	C9V[12]	C9V[11]	C9V[10]	C9V[9]	C9V[8]

Table 40. Cell Voltage Register Group D

REGISTER	RD/WR	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
CVDR0	RD	C10V[7]	C10V[6]	C10V[5]	C10V[4]	C10V[3]	C10V[2]	C10V[1]	C10V[0]
CVDR1	RD	C10V[15]	C10V[14]	C10V[13]	C10V[12]	C10V[11]	C10V[10]	C10V[9]	C10V[8]
CVDR2	RD	C11V[7]	C11V[6]	C11V[5]	C11V[4]	C11V[3]	C11V[2]	C11V[1]	C11V[0]
CVDR3	RD	C11V[15]	C11V[14]	C11V[13]	C11V[12]	C11V[11]	C11V[10]	C11V[9]	C11V[8]
CVDR4	RD	C12V[7]	C12V[6]	C12V[5]	C12V[4]	C12V[3]	C12V[2]	C12V[1]	C12V[0]
CVDR5	RD	C12V[15]	C12V[14]	C12V[13]	C12V[12]	C12V[11]	C12V[10]	C12V[9]	C12V[8]

OPERATION

Table 41. Auxiliary Register Group A

REGISTER	RD/WR	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
AVAR0	RD	G1V[7]	G1V[6]	G1V[5]	G1V[4]	G1V[3]	G1V[2]	G1V[1]	G1V[0]
AVAR1	RD	G1V[15]	G1V[14]	G1V[13]	G1V[12]	G1V[11]	G1V[10]	G1V[9]	G1V[8]
AVAR2	RD	G2V[7]	G2V[6]	G2V[5]	G2V[4]	G2V[3]	G2V[2]	G2V[1]	G2V[0]
AVAR3	RD	G2V[15]	G2V[14]	G2V[13]	G2V[12]	G2V[11]	G2V[10]	G2V[9]	G2V[8]
AVAR4	RD	G3V[7]	G3V[6]	G3V[5]	G3V[4]	G3V[3]	G3V[2]	G3V[1]	G3V[0]
AVAR5	RD	G3V[15]	G3V[14]	G3V[13]	G3V[12]	G3V[11]	G3V[10]	G3V[9]	G3V[8]

Table 42. Auxiliary Register Group B

REGISTER	RD/WR	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
AVBR0	RD	G4V[7]	G4V[6]	G4V[5]	G4V[4]	G4V[3]	G4V[2]	G4V[1]	G4V[0]
AVBR1	RD	G4V[15]	G4V[14]	G4V[13]	G4V[12]	G4V[11]	G4V[10]	G4V[9]	G4V[8]
AVBR2	RD	G5V[7]	G5V[6]	G5V[5]	G5V[4]	G5V[3]	G5V[2]	G5V[1]	G5V[0]
AVBR3	RD	G5V[15]	G5V[14]	G5V[13]	G5V[12]	G5V[11]	G5V[10]	G5V[9]	G5V[8]
AVBR4	RD	REF[7]	REF[6]	REF[5]	REF[4]	REF[3]	REF[2]	REF[1]	REF[0]
AVBR5	RD	REF[15]	REF[14]	REF[13]	REF[12]	REF[11]	REF[10]	REF[9]	REF[8]

Table 43. Status Register Group A

REGISTER	RD/WR	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
STAR0	RD	SOC[7]	SOC[6]	SOC[5]	SOC[4]	SOC[3]	SOC[2]	SOC[1]	SOC[0]
STAR1	RD	SOC[15]	SOC[14]	SOC[13]	SOC[12]	SOC[11]	SOC[10]	SOC[9]	SOC[8]
STAR2	RD	ITMP[7]	ITMP[6]	ITMP[5]	ITMP[4]	ITMP[3]	ITMP[2]	ITMP[1]	ITMP[0]
STAR3	RD	ITMP[15]	ITMP[14]	ITMP[13]	ITMP[12]	ITMP[11]	ITMP[10]	ITMP[9]	ITMP[8]
STAR4	RD	VA[7]	VA[6]	VA[5]	VA[4]	VA[3]	VA[2]	VA[1]	VA[0]
STAR5	RD	VA[15]	VA[14]	VA[13]	VA[12]	VA[11]	VA[10]	VA[9]	VA[8]

Table 44. Status Register Group B

REGISTER	RD/WR	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
STBR0	RD	VD[7]	VD[6]	VD[5]	VD[4]	VD[3]	VD[2]	VD[1]	VD[0]
STBR1	RD	VD[15]	VD[14]	VD[13]	VD[12]	VD[11]	VD[10]	VD[9]	VD[8]
STBR2	RD	C4OV	C4UV	C3OV	C3UV	C2OV	C2UV	C1OV	C1UV
STBR3	RD	C8OV	C8UV	C7OV	C7UV	C6OV	C6UV	C5OV	C5UV
STBR4	RD	C12OV	C12UV	C11OV	C11UV	C10OV	C10UV	C9OV	C9UV
STBR5	RD	REV[3]	REV[2]	REV[1]	REV[0]	RSVD	RSVD	MUXFAIL	THSD

Table 45. COMM Register Group

REGISTER	RD/WR	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
COMM0	RD/WR	ICOM0[3]	ICOM0[2]	ICOM0[1]	ICOM0[0]	D0[7]	D0[6]	D0[5]	D0[4]
COMM1	RD/WR	D0[3]	D0[2]	D0[1]	D0[0]	FCOM0[3]	FCOM0[2]	FCOM0[1]	FCOM0[0]
COMM2	RD/WR	ICOM1[3]	ICOM1[2]	ICOM1[1]	ICOM1[0]	D1[7]	D1[6]	D1[5]	D1[4]
COMM3	RD/WR	D1[3]	D1[2]	D1[1]	D1[0]	FCOM1[3]	FCOM1[2]	FCOM1[1]	FCOM1[0]
COMM4	RD/WR	ICOM2[3]	ICOM2[2]	ICOM2[1]	ICOM2[0]	D2[7]	D2[6]	D2[5]	D2[4]
COMM5	RD/WR	D2[3]	D2[2]	D2[1]	D2[0]	FCOM2[3]	FCOM2[2]	FCOM2[1]	FCOM2[0]

6804121c

OPERATION

Table 46. Memory Bit Descriptions

NAME	DESCRIPTION	VALUES
GPIOx	GPIOx Pin Control	Write: 0 -> GPIOx Pin Pull-Down ON; 1-> GPIOx Pin Pull-Down OFF (Default) Read: 0 -> GPIOx Pin at Logic 0; 1 -> GPIOx Pin at Logic 1
REFON	Reference Powered Up	1 -> Reference Remains Powered Up Until Watchdog Timeout 0 -> Reference Shuts Down after Conversions (Default)
SWTRD	SWTEN Pin Status (Read Only)	1 -> SWTEN Pin at Logic 1 0 -> SWTEN Pin at Logic 0
ADCOPT	ADC Mode Option Bit	ADCOPT: 0 -> Selects Modes 27kHz, 7kHz or 26Hz with MD[1:0] Bits in ADC Conversion Commands (Default). 1 -> Selects Modes 14kHz, 3kHz or 2kHz with MD[1:0] Bits in ADC Conversion Commands.
VUV	Undervoltage Comparison Voltage*	Comparison voltage = (VUV + 1) • 16 • 100µV Default: VUV = 0x000
VOV	Overvoltage Comparison Voltage*	Comparison voltage = VOV • 16 • 100µV Default: VOV = 0x000
DCC[x]	Discharge Cell x	x = 1 to 12 1 -> Turn ON Shorting Switch for Cell x 0 -> Turn OFF Shorting Switch for Cell x (Default)

DCTO	Discharge Time Out Value	DCTO (Write)	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
		Time (Min)	Disabled	0.5	1	2	3	4	5	10	15	20	30	40	60	75	90	120
		DCTO (Read)	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
		Time Left (Min)	Disabled or Timeout	0 to 0.5	0.5 to 1	1 to 2	2 to 3	3 to 4	4 to 5	5 to 10	10 to 15	15 to 20	20 to 30	30 to 40	40 to 60	60 to 75	75 to 90	90 to 120

CxV	Cell x Voltage*	x = 1 to 12	16-Bit ADC Measurement Value for Cell x Cell Voltage for Cell x = CxV • 100µV CxV Is Reset to 0xFFFF on Power-Up and After Clear Command
GxV	GPIO x Voltage*	x = 1 to 5	16-Bit ADC Measurement Value for GPIOx Voltage for GPIOx = GxV • 100µV GxV Is Reset to 0xFFFF on Power-Up and After Clear Command
REF	2nd Reference Voltage*		16-Bit ADC Measurement Value for 2nd Reference Voltage for 2nd Reference = REF • 100µV **Normal Range Is within 2.985V to 3.015V**
SOC	Sum of Cells Measurement*		16-Bit ADC Measurement Value of the Sum of All Cell Voltages Sum of All Cells Voltage = SOC • 100µV • 20
ITMP	Internal Die Temperature*		16-Bit ADC Measurement Value of Internal Die Temperature **Temperature Measurement (°C) = ITMP • 100µV/7.5mV/°C – 273°C**
VA	Analog Power Supply Voltage*		16-Bit ADC Measurement Value of Analog Power Supply Voltage Analog Power Supply Voltage = VA • 100µV **Normal Range Is within 4.5V to 5.5V**
VD	Digital Power Supply Voltage*		16-Bit ADC Measurement Value of Digital Power Supply Voltage Digital Power Supply Voltage = VA • 100µV **Normal Range Is within 2.7V to 3.6V**
CxOV	Cell x Overvoltage Flag	x = 1 to 12	Cell Voltage Compared to VOV Comparison Voltage 0 -> Cell x Not Flagged for Overvoltage Condition. 1 -> Cell x Flagged
CxUV	Cell x Undervoltage Flag	x = 1 to 12	Cell Voltage Compared to VUV Comparison Voltage 0 -> Cell x Not Flagged for Undervoltage Condition. 1 -> Cell x Flagged
REV	Revision Code	Device Revision Code. See Revision Code and Reserved Bits in Operation Section.	
RSVD	Reserved Bits	See Revision Code and Reserved Bits in Operation Section.	

OPERATION

Table 46. Memory Bit Descriptions

NAME	DESCRIPTION	VALUES						
MUXFAIL	Multiplexer Self-Test Result	Read: 0 -> Multiplexer Passed Self Test 1 -> Multiplexer Failed Self Test						
THSD	Thermal Shutdown Status	Read: 0 -> Thermal Shutdown Has Not Occurred 1 -> Thermal Shutdown Has Occurred THSD Bit Cleared to 0 on Read of Status Register Group B						
ICOMn	Initial Communication Control Bits	Write	I2C	0110	0001	0000	0111	
				START	STOP	BLANK	NO TRANSMIT	
			SPI	1000	1001	1111		
				CSB Low	CSB High	NO TRANSMIT		
		Read	I2C	0110	0001	0000	0111	
				START from Master	STOP from Master	SDA Low Between Bytes	SDA High Between Bytes	
			SPI	0111				
Dn	I²C/SPI Communication Data Byte	Data Transmitted (Received) to (From) I²C/SPI Slave Device						
FCOMn	Final Communication Control Bits	Write	I2C	0000	1000	1001		
				Master ACK	Master NACK	Master NACK + STOP		
			SPI	X000	1001			
				CSB Low	CSB High			
		Read	I2C	0000	0111	1111	0001	1001
				ACK from Master	ACK from Slave	NACK from Slave	ACK from Slave + STOP from Master	NACK from Slave + STOP from Master
			SPI	1111				

*Voltage equations use the decimal value of registers, 0 to 4095 for 12 bits and 0 to 65535 for 16 bits.

PROGRAMMING EXAMPLES

The following examples use a configuration of 3 stacked LTC6804-1 devices: S1, S2, S3. Port A on device S1 is configured in SPI mode (ISOMD pin low). Port A on devices S2 and S3 is configured in isoSPI mode (ISOMD pin high). Port B on S1 is connected to Port A on S2. Port B on S2 is connected to Port A on S3. The microcontroller communicates to the stack through Port A on S1.

Waking Up Serial Interface

1. Send a dummy byte. The activity on CSB and SCK will wake up the serial interface on device S1.
2. Wait for the amount of time 3 • t_{WAKE} in order to power up all devices S1, S2 and S3.

For large stacks where some devices may go to the IDLE state after waking, apply steps 3 and 4:

3. Send a second dummy byte.
4. Wait for the amount of time 3 • t_{READY}
5. Send commands

Write Configuration Registers

1. Pull CSB low
2. Send WRCFG command (0x00 0x01) and its PEC (0x3D 0x6E)
3. Send CFGR0 byte of device S3, then CFGR1(S3), … CFGR5(S3), PEC of CFGR0(S3) to CFGR5(S3)
4. Send CFGR0 byte of device S2, then CFGR1(S2), … CFGR5(S2), PEC of CFGR0(S2) to CFGR5(S2)
5. Send CFGR0 byte of device S1, then CFGR1(S1), … CFGR5(S1), PEC of CFGR0(S1) to CFGR5(S1)
6. Pull CSB high, data latched into all devices on rising edge of CSB

OPERATION

Calculation of serial interface time for sequence above:

Number of LTC6804-1s in daisy chain stack = *n*

Number of bytes in sequence (B):

Command: 2 (command byte) + 2 (command PEC) = 4

Data: 6 (Data bytes) + 2 (Data PEC) per LTC6804 = 8 bytes per device

B = 4 + 8 • *n*

Serial port frequency per bit = F

Time = (1/F) • B • 8 bits/byte = (1/F) • [4 + 8 • *n*] • 8

Time for 3 LTC6804 example above, with 1MHz serial port = (1/1e6) • (4 + 8 • 3) • 8 = 224µs

Note: This time will remain the same for all write and read commands.

Read Cell Voltage Register Group A

1. Pull CSB low
2. Send RDCVA command (0x00 0x04) and its PEC (0x07 0xC2)
3. Read CVAR0 byte of device S1, then CVAR1(S1), … CVAR5(S1), PEC of CVAR0(S1) to CVAR5(S1)
4. Read CVAR0 byte of device S2, then CVAR1(S2), … CVAR5(S2), PEC of CVAR0(S2) to CVAR0(S2)
5. Read CVAR0 byte of device S3, then CVAR1(S3), … CVAR5(S3), PEC of CVAR0(S3) to CVAR5(S3)
6. Pull CSB high

Start Cell Voltage ADC Conversion

(All cells, normal mode with discharge permitted) and poll status

1. Pull CSB low
2. Send ADCV command with MD[1:0] = 10 and DCP = 1 i.e. 0x03 0x70 and its PEC (0xAF 0x42)
3. Pull CSB high

Clear Cell Voltage Registers

1. Pull CSB low
2. Send CLRCELL command (0x07 0x11) and its PEC (0xC9 0xC0)
3. Pull CSB high

Poll ADC Status

(Parallel configuration and ISOMD = 0)

This example uses an addressed LTC6804-2 with address A [3:0] = 0011 and ISOMD = 0

1. Pull CSB low
2. Send PLADC command (0x9F 0x14) and its PEC (0x1C 0x48)
3. SDO output is pulled low if the LTC6804-2 is busy. The host needs to send clocks on SCK in order for the polling status to be updated from the addressed device.
4. SDO output is high when the LTC6804-2 has completed conversions
5. Pull CSB high to exit polling

Talk to an I²C Slave Connected to LTC6804

The LTC6804 supports I²C slave devices by connection to GPIO4(SDA) and GPIO5(SCL). One valuable use for this capability is to store production calibration constants or other information in a small serial EEPROM using a connection like shown in Figure 25.

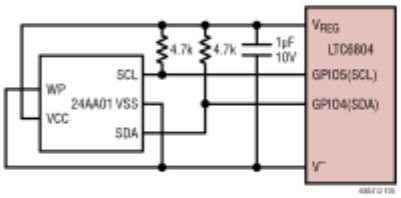


Figure 25. Connecting I²C EEPROM to LTC6804 GPIO Pins

OPERATION

This example uses a single LTC6804-1 to write a byte of data to an I²C EEPROM. The LTC6804 will send three bytes of data to the I²C slave device. The data sent will be B0 = 0xA0 (EEPROM address), B1 = 0x01 (write command), and B2 = 0xAA (data to be stored in EEPROM). The three bytes will be transmitted to the I²C slave device in the following format:

START – B0 – NACK – B1 – NACK – B2 – NACK – STOP

1. Write data to COMM register using WRCOMM command
   a. Pull CSB low
   b. Send WRCOMM command (0x07 0x21) and its PEC (0x24 0xB2)
   c. Send
   COMM0 = 0x6A, COMM1 = 0x08 ([START] [B0] [NACK]),
   COMM2 = 0x00, COMM3 = 0x18 ([BLANK] [B1] [NACK]),
   COMM4 = 0x0A, COMM5 = 0xA9 ([BLANK] [B2] [NACK+STOP])
   and PEC = 0x6D 0xFB for the above data
   d. Pull CSB high
2. Send the 3 bytes of data to I²C slave device using STCOMM command
   a. Pull CSB low
   b. Send STCOMM command (0x07 0x23) and its PEC (0xB9 0xE4)
   c. Send 72 clock cycles on SCK
   d. Pull CSB high
3. Data transmitted to slave during the STCOMM command is stored in the COMM register. Use the RDCOMM command to retrieve the data
   a. Pull CSB low
   b. Send RDCOMM command (0x07 0x22) and its PEC (0x32 0xD6)
   c. Read COMM0-COMM5 and the PEC for the 6 bytes of data.
   Assuming the slave acknowledged all 3 bytes of data, the read back data in this example would look like:
   COMM0 = 0x6A, COMM1 = 0x07, COMM2 = 0x70, COMM3 = 0x17, COMM4 = 0x7A, COMM5 = 0xA1, PEC = 0xD0 0xDE
   d. Pull CSB high

Note: If the slave returns data, this data will be placed in COMM0-COMM5.

Figure 26 shows the activity on GPIO5 (SCL) and GPIO4 (SDA) ports of the I²C master for 72 clock cycles during the STCOMM command in the above example.

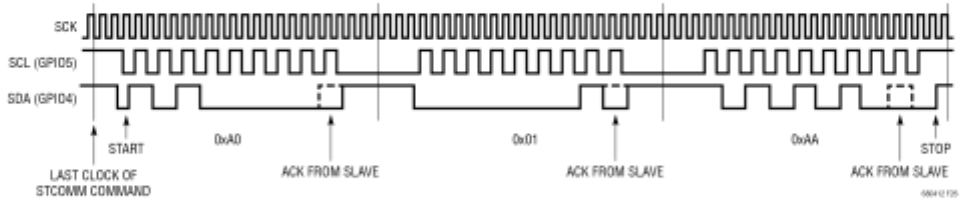


Figure 26. LTC6804 I²C Communication Example

OPERATION

Talk to a SPI Slave Connected to LTC6804

This example uses a single LTC6804-1 device which has a SPI device connected to it through GPIO3 (CSBM), GPIO4 (SDOM) and GPIO5 (SCKM). In this example, the LTC6804 device sends out 3 bytes of data B0 = 0x55, B1 = 0xAA and B2 = 0xCC to the SPI slave device in the following format: CSB low – B0 – B1 – B2 – CSB high

1. Write data to COMM register using WRCOMM command
   a. Pull CSBM low
   b. Send WRCOMM command (0x07 0x21) and its PEC (0x24 0xB2)
   c. Send
      COMM0 = 0x85, COMM1 = 0x50 ([CSBM low] [B0] [CSBM low]),
      COMM2 = 0x8A, COMM3 = 0xA0 ([CSBM low] [B1] [CSBM low]),
      COMM4 = 0x8C, COMM5 = 0xC9 ([CSBM low] [B2] [CSBM high])
      and PEC = 0x89 0xA4 for the above data.
   d. Pull CSB high
2. Send the 3 bytes of data to SPI slave device using STCOMM command
   a. Pull CSB low
   b. Send STCOMM command (0x07 0x23) and its PEC (0xB9 0xE4)
   c. Send 72 clock cycles on SCK
   d. Pull CSB high
3. Data transmitted to slave during the STCOMM command is stored in the COMM register. Use the RDCOMM command to retrieve the data.
   a. Pull CSB low
   b. Send RDCOMM command (0x07 0x22) and its PEC (0x32 0xD6)
   c. Read COMM0-COMM5 and the PEC for the 6 bytes of data. The read back data in this example would look like:
      COMM0 = 0x755F, COMM1 = 0x7AAF, COMM2 = 7CCF, PEC = 0xF2BA
   d. Pull CSB high

Note: If the slave returns data, this data will be placed in COMM0-COMM5.

Figure 27 shows the activity on GPIO3 (CSBM), GPIO5 (SCKM) and GPIO4 (SDOM) ports of SPI master for 72 clock cycles during the STCOMM command in the above example.

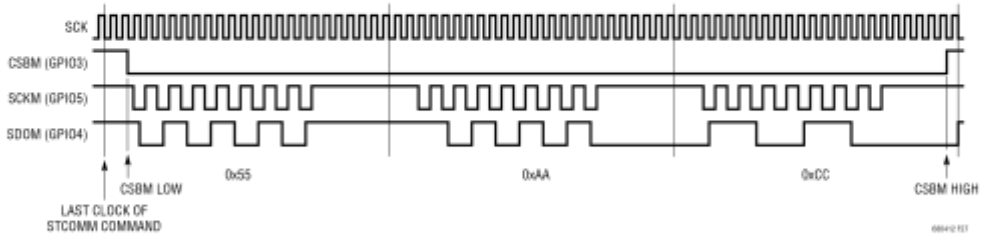


Figure 27. LTC6804 SPI Communication Example

APPLICATIONS INFORMATION

SIMPLE LINEAR REGULATOR

The LTC6804 draws most of its power from the V_{REG} input pin. 5V ±0.5V should be applied to V_{REG}. A regulated DC/DC converter can power V_{REG} directly, or the DRIVE pin may be used to form a discrete regulator with the addition of a few external components. When active, the DRIVE output pin provides a low current 5.6V output that can be buffered using a discrete NPN transistor, as shown in Figure 28. The collector power for the NPN can come from any potential of 6V or more above V^-, including the cells being monitored or an unregulated converter supply. A 100Ω/100nF RC decoupling network is recommended for the collector power connection to protect the NPN from transients. The emitter of the NPN should be bypassed with a 1µF capacitor. Larger capacitor values should be avoided because they increase the wake-up time of the LTC6804. Some attention to the thermal characteristic of the NPN is needed, as there can be significant heating with a high collector voltage.

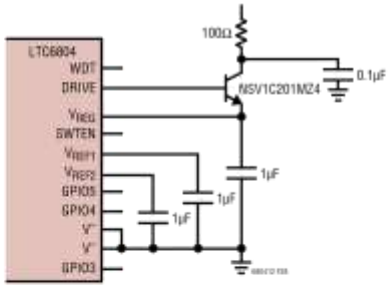


Figure 28. Simple V_{REG} Power Source Using NPN Pass Transistor

IMPROVED REGULATOR POWER EFFICIENCY

To minimize power consumption within the LTC6804, the current drawn on the V^+ pin has been designed to be very small (500µA). The voltage on the V^+ pin must be at least as high as the top cell to provide accurate measurement. The V^+ and V_{REG} pins can be unpowered to provide an exceptionally low battery drain shutdown mode. In many applications, the V^+ will be permanently connected to the top cell potential through a decoupling RC to protect against transients (100Ω/100nF is recommended).

For better running efficiency when powering from the cell stack, the V_{REG} may be powered from a buck converter rather than the NPN pass transistor. An ideal circuit for this is based on the LT3990 as shown in Figure 29. A 1k resistor should be used in series with the input to prevent inrush current when connecting to the stack and to reduce conducted EMI. The EN/UVLO pin should be connected to DRIVE so that the converter sleeps along with the LTC6804. The LTC6804 watchdog timer requires V_{REG} power to timeout. Therefore, if the EN/UVLO pin is not connected to DRIVE, care must be taken to allow the LTC6804 to timeout first before removing V_{REG} power; otherwise the LTC6804 will not enter sleep mode.

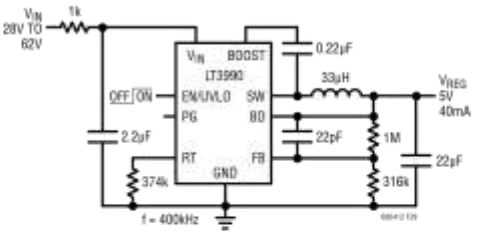


Figure 29. V_{REG} Powered from Cell Stack with High Efficiency

APPLICATIONS INFORMATION

FULLY ISOLATED POWER

A simple DC/DC flyback converter can provide isolated power for an LTC6804 from a remote 12V power source as shown in Figure 30. This circuit, along with the isoSPI transformer isolation, results in LTC6804 circuitry that is completely floating and uses almost no power from the batteries. Aside from reducing the amount of circuitry that operates at battery potential, such an arrangement prevents battery load imbalance. The LTC6804 watchdog timer requires V_{REG} power to timeout. Therefore, care must be taken to allow the LTC6804 to timeout first before removing V_{REG} power; otherwise the LTC6804 will not enter sleep mode. A diode should be added between the V+ and the top cell being monitored. This will prevent any current from conducting through internal parasitic paths inside the IC when the isolated power is removed.

READING EXTERNAL TEMPERATURE PROBES

Figure 31 shows the typical biasing circuit for a negative-temperature-coefficient (NTC) thermistor. The 10kΩ at 25°C is the most popular sensor value and the V_{REF2} output stage is designed to provide the current required to directly bias several of these probes. The biasing resistor is selected to correspond to the NTC value so the circuit will provide 1.5V at 25°C (V_{REF2} is 3V nominal). The overall circuit response is approximately –1%/°C in the range of typical cell temperatures, as shown in the chart of Figure 31 .

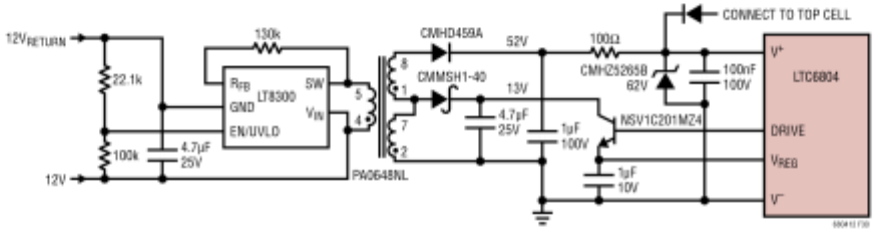


Figure 30. Powering LTC6804 from a Remote 12V Source

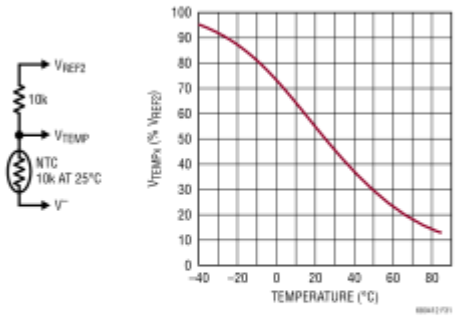


Figure 31. Typical Temperature Probe Circuit and Relative Output

APPLICATIONS INFORMATION

EXPANDING THE NUMBER OF AUXILIARY MEASUREMENTS

The LTC6804 provides five GPIO pins, each of which is capable of performing as an ADC input. In some applications there is need to measure more signals than this, so one means of supporting higher signal count is to add a MUX circuit such as shown in Figure 32. This circuit digitizes up to sixteen source signals using the GPIO1 ADC input and MUX control is provided by two other GPIO lines configured as an I^2C port. The buffer amplifier provides for fast settling of the selected signal to increase the usable conversion rate.

INTERNAL PROTECTION FEATURES

The LTC6804 incorporates various ESD safeguards to ensure a robust performance. An equivalent circuit showing the specific protection structures is shown in Figure 33. While pins 43 to 48 have different functionality for the -1 and -2 variants, the protection structure is the same. Zener-like suppressors are shown with their nominal clamp voltage, other diodes exhibit standard PN junction behavior.

FILTERING OF CELL AND GPIO INPUTS

The LTC6804 uses a delta-sigma ADC, which has delta-sigma modulator followed by a SINC3 finite impulse response (FIR) digital filter. This greatly reduces input filtering requirements. Furthermore, the programmable oversampling ratio allows the user to determine the best trade-off between measurement speed and filter cutoff frequency. Even with this high order lowpass filter, fast transient noise can still induce some residual noise in measurements, especially in the faster conversion modes. This can be minimized by adding an RC lowpass decoupling to each ADC input, which also helps reject potentially damaging high energy transients. Adding more than about 100Ω to the ADC inputs begins to introduce a systematic error in the measurement, which can be improved by raising the filter capacitance or mathematically compensating in software with a calibration procedure. For situations that demand the highest level of battery voltage ripple rejection, grounded capacitor filtering is recommended. This configuration has a series resistance and capacitors that decouple HF noise to V^-. In systems where noise is less

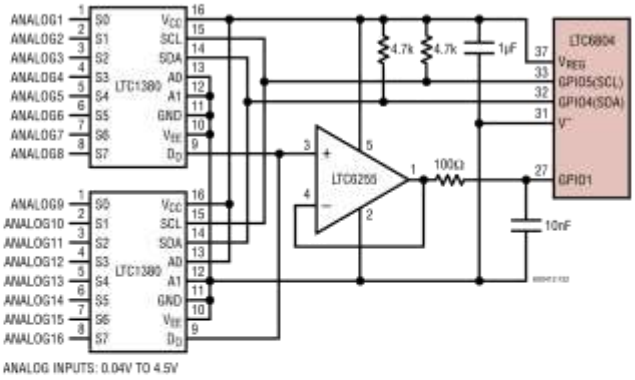
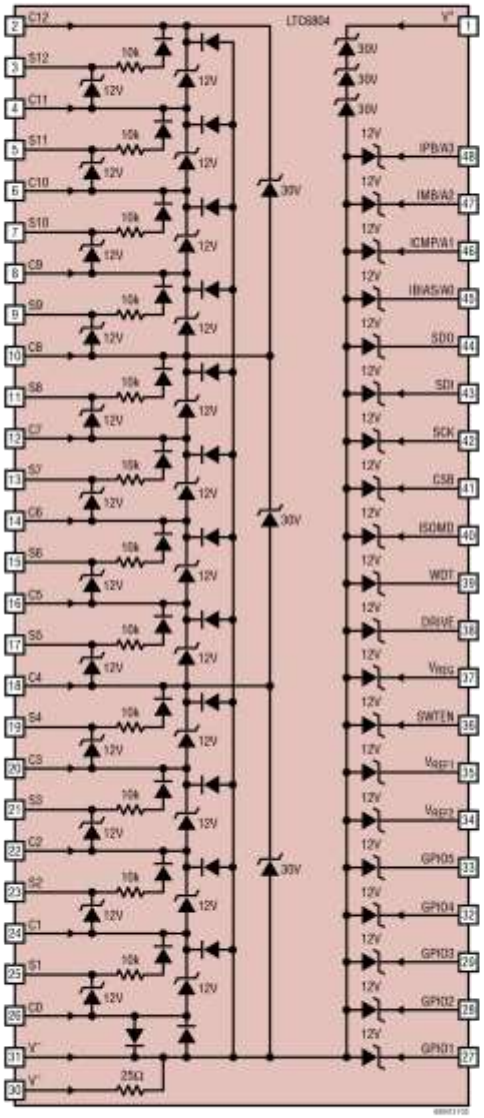


Figure 32. MUX Circuit Supports Sixteen Additional Analog Measurements

APPLICATIONS INFORMATION



NOTE: NOT SHOWN ARE PN DIODES TO ALL OTHER PINS FROM PIN 31

Figure 33. Internal ESD Protection Structure of LTC6804

periodic or higher oversample rates are in use, a differential capacitor filter structure is adequate. In this configuration there are series resistors to each input, but the capacitors connect between the adjacent C pins. However, the differential capacitor sections interact. As a result, the filter response is less consistent and results in less attenuation than predicted by the RC, by approximately a decade. Note that the capacitors only see one cell of applied voltage (thus smaller and lower cost) and tend to distribute transient energy uniformly across the IC (reducing stress events on the internal protection structure). Figure 34 shows the two methods schematically. Basic ADC accuracy varies with R, C as shown in the Typical Performance curves, but error is minimized if R = 100Ω and C = 10nF. The GPIO pins will always use a grounded capacitor configuration because the measurements are all with respect to V⁻.

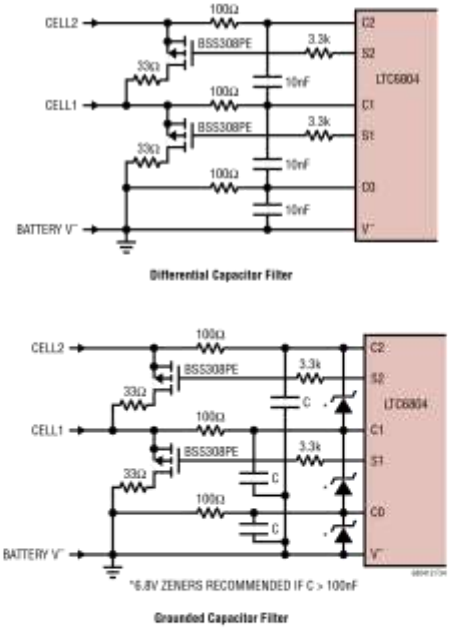


Figure 34. Input Filter Structure Configurations

APPLICATIONS INFORMATION

CELL BALANCING WITH INTERNAL MOSFETS

The S1 through S12 pins are used to balance battery cells. If one cell in a series becomes overcharged, an S output can be used to discharge the cell. Each S output has an internal N-channel MOSFET for discharging. The NMOS has a maximum on resistance of 20Ω. An external resistor should be connected in series with the NMOS to dissipate heat outside of the LTC6804 package as illustrated in Figure 35. It is still possible to use an RC to add additional filtering to cell voltage measurements but the filter R must remain small, typically around 10Ω to reduce the effect on the programmed balance current. When using the internal MOSFETs to discharge cells, the die temperature should be monitored. See Power Dissipation and Thermal Shutdown section.

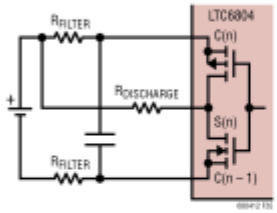


Figure 35. Internal Discharge Circuit

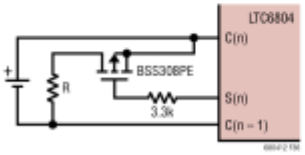


Figure 36. External Discharge Circuit

CELL BALANCING WITH EXTERNAL MOSFETS

The S outputs include an internal pull-up PMOS transistor. The S pins can act as digital outputs suitable for driving the gate of an external MOSFET. For applications requiring high battery discharge currents, connect a discrete PMOS switch device and suitable discharge resistor to the cell, and the gate terminal to the S output pin, as illustrated in Figure 36. Figure 34 shows external MOSFET circuits that include RC filtering.

DISCHARGE CONTROL DURING CELL MEASUREMENTS

If the discharge permited (DCP) command bit is high in a cell measurement command, then the S pin discharge states are not altered during the cell measurements. However, if the DCP bit is low, any discharge that is turned on will be turned off when the corresponding cell or adjacent cells are being measured. Table 47 illustrates this during an

Table 47. Discharge Control During an ADCV Command with DCP = 0

	CELL MEASUREMENT PERIODS						CELL CALIBRATION PERIODS					
	CELL1/7	CELL2/8	CELL3/9	CELL4/10	CELL5/11	CELL6/12	CELL1/7	CELL2/8	CELL3/9	CELL4/10	CELL5/11	CELL6/12
DISCHARGE PIN	t_0 to t_{1M}	t_{1M} to t_{2M}	t_{2M} to t_{3M}	t_{3M} to t_{4M}	t_{4M} to t_{5M}	t_{5M} to t_{6M}	t_{6M} to t_{1C}	t_{1C} to t_{2C}	t_{2C} to t_{3C}	t_{3C} to t_{4C}	t_{4C} to t_{5C}	t_{5C} to t_{6C}
S1	OFF	OFF	ON	ON	ON	OFF	OFF	OFF	ON	ON	ON	OFF
S2	OFF	OFF	OFF	ON	ON	ON	OFF	OFF	OFF	ON	ON	ON
S3	ON	OFF	OFF	OFF	ON	ON	ON	OFF	OFF	OFF	ON	ON
S4	ON	ON	OFF	OFF	OFF	ON	ON	ON	OFF	OFF	OFF	ON
S5	ON	ON	ON	OFF	OFF	OFF	ON	ON	ON	OFF	OFF	OFF
S6	OFF	ON	ON	ON	OFF	OFF	OFF	ON	ON	ON	OFF	OFF
S7	OFF	OFF	ON	ON	ON	OFF	OFF	OFF	ON	ON	ON	OFF
S8	OFF	OFF	OFF	ON	ON	ON	OFF	OFF	OFF	ON	ON	ON
S9	ON	OFF	OFF	OFF	ON	ON	ON	OFF	OFF	OFF	ON	ON
S10	ON	ON	OFF	OFF	OFF	ON	ON	ON	OFF	OFF	OFF	ON
S11	ON	ON	ON	OFF	OFF	OFF	ON	ON	ON	OFF	OFF	OFF
S12	OFF	ON	ON	ON	OFF	OFF	OFF	ON	ON	ON	OFF	OFF

6804121fc

APPLICATIONS INFORMATION

ADCV command with DCP = 0. In this table, OFF implies that a discharge is forced off during that period even if the corresponding DCC[x] bit is high in the configuration register. ON implies that if the discharge is turned on, it will stay on during that period. Refer to Figure 3 for the timing of the ADCV command.

POWER DISSIPATION AND THERMAL SHUTDOWN

The internal MOSFETs connected to the pins S1 through S12 pins can be used to discharge battery cells. An external resistor should be used to limit the power dissipated by the MOSFETs. The maximum power dissipation in the MOSFETs is limited by the amount of heat that can be tolerated by the LTC6804. Excessive heat results in elevated die temperatures. Little or no degradation will be observed in the measurement accuracy for die temperatures up to 125°C. Damage may occur above 150°C, therefore the recommended maximum die temperature is 125°C. To protect the LTC6804 from damage due to overheating a thermal shutdown circuit is included. Overheating of the device can occur when dissipating significant power in the cell discharge switches. The thermal shutdown circuit is enabled whenever the device is not in sleep mode (see LTC6804 Core State Descriptions). If the temperature detected on the device goes above approximately 150°C the configuration registers will be reset to default states turning off all discharge switches. When a thermal shutdown has occurred, the THSD bit in the status register group B will go high. The bit is cleared after a read operation of the status register group B. The bit can also be set using the CLRSTAT command. Since thermal shutdown interrupts normal operation, the internal temperature monitor should be used to determine when the device temperature is approaching unacceptable levels.

METHOD TO VERIFY BALANCING CIRCUITRY

The functionality of the discharge circuitry is best verified by cell measurements. Figure 37 shows an example using the LTC6804 battery monitor IC. The resistor between the battery and the source of the discharge MOSFET causes cell voltage measurements to decrease. The amount of measurement change depends on the resistor values and the MOSFET on resistance.

The following algorithm could be used in conjunction with Figure 37:

1. Measure all cells with no discharging (all S outputs off) and read and store the results.
2. Turn on S1 and S7
3. Measure C1-C0, C7-C6
4. Turn off S1 and S7
5. Turn on S2 and S8
6. Measure C2-C1, C8-C7
7. Turn off S2 and S8

...

14. Turn on S6 and S12
15. Measure C6-C5, C12-C11
16. Turn off S6 and S12
17. Read the voltage register group to get the results of steps 2 thru 16.
18. Compare new readings with old readings. Each cell voltage reading should have decreased by a fixed percentage set by R_{B1} and R_{B2} (Figure 37). The exact amount of decrease depends on the resistor values and MOSFET characteristics.

Improved PEC Calculation

The PEC allows the user to have confidence that the serial data read from the LTC6804 is valid and has not been corrupted by any external noise source. This is a critical feature for reliable communication and the LTC6804 requires that a PEC be calculated for all data being read from and written to the LTC6804. For this reason it is important to have an efficient method for calculating the PEC. The code below demonstrates a simple implementation of a lookup table derived PEC calculation method. There are two functions, the first function init_PEC15_Table() should only be called once when the microcontroller starts and will initialize a PEC15 table array called pec15Table[]. This table will be used in all future PEC calculations. The pec15 table can also be hard coded into the microcontroller rather than running the init_PEC15_Table() function at startup. The pec15() function calculates the PEC and will return the correct 15 bit PEC for byte arrays of any given length.

APPLICATIONS INFORMATION

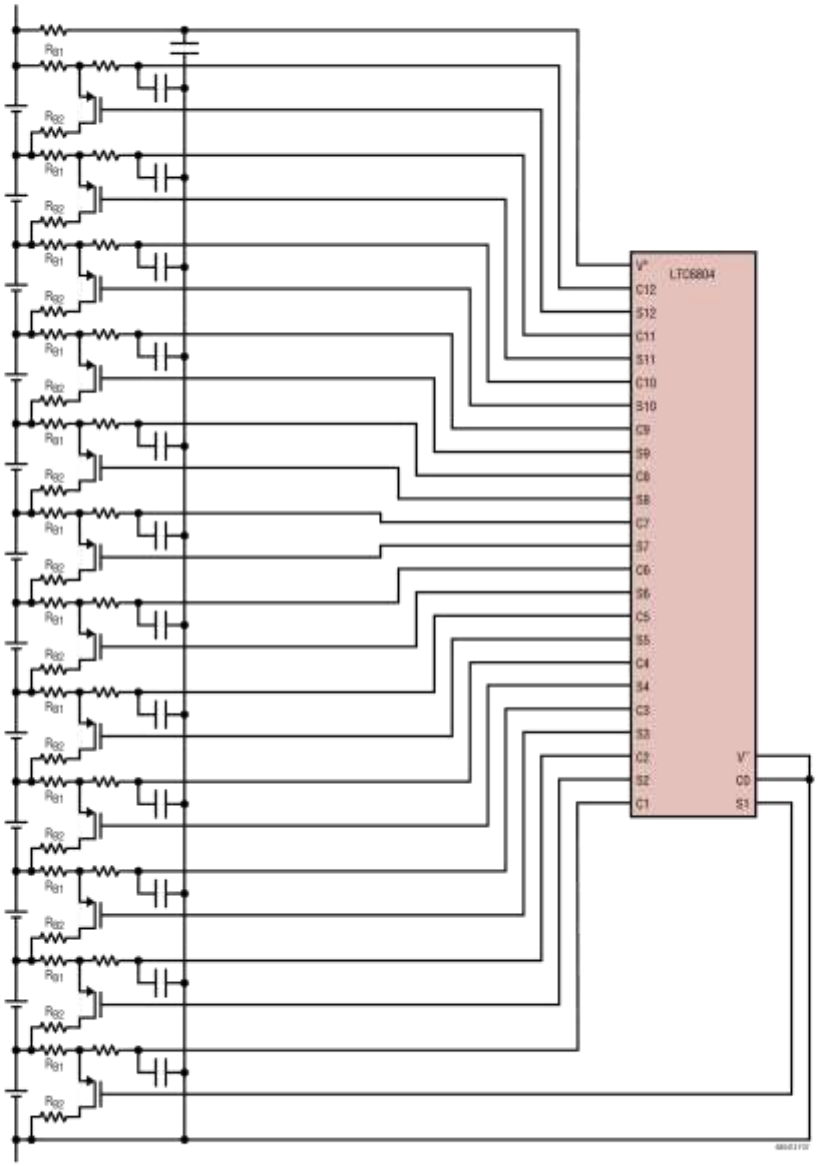


Figure 37. Balancing Self Test Circuit

APPLICATIONS INFORMATION

```
/***********************************
Copyright 2012 Linear Technology Corp. (LTC)
Permission to freely use, copy, modify, and distribute this software for any
purpose with or without fee is hereby granted, provided that the above
copyright notice and this permission notice appear in all copies:

THIS SOFTWARE IS PROVIDED "AS IS" AND LTC DISCLAIMS ALL WARRANTIES
INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO
EVENT SHALL LTC BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL
DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM ANY USE OF SAME, INCLUDING
ANY LOSS OF USE OR DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE
OR OTHER TORTUOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR
PERFORMANCE OF THIS SOFTWARE.
***********************************************************/

int16 pec15Table[256];
int16 CRC15_POLY = 0x4599;
void init_PEC15_Table()
{
    for (int i = 0; i < 256; i++)
    {
        remainder = i << 7;
        for (int bit = 8; bit > 0; --bit)
        {
            if (remainder & 0x4000)
            {
                remainder = ((remainder << 1));
                remainder = (remainder ^ CRC15poly)
            }
            else
            {
                remainder = ((remainder << 1));
            }
        }
        pec15Table[i] = remainder&0xFFFF;
    }
}

unsigned int16 pec15 (char *data , int len)
{
    int16 remainder,address;

    remainder = 16;//PEC seed
    for (int i = 0; i < len; i++)
    {
        address = ((remainder >> 7) ^ data[i]) & 0xff;//calculate PEC table address
        remainder = (remainder << 8 ) ^ pec15Table[address];
    }
    return (remainder*2);//The CRC15 has a 0 in the LSB so the final value must be multiplied by 2
}
```

APPLICATIONS INFORMATION

CURRENT MEASUREMENT WITH A HALL EFFECT SENSOR

The LTC6804 auxiliary ADC inputs (GPIO pins) may be used for any analog signal, including those from various active sensors that generate a compatible voltage. One such example that may be useful in a battery management setting is the capture of battery current. Hall-effect sensors are popular for measuring large battery currents since the technology provides a non-contact, low power dissipation solution. Figure 38 shows schematically a typical Hall sensor that produces two outputs that proportion to the V_{CC} provided. The sensor is powered from a 5V source and produces analog outputs that are connected to GPIO pins or inputs of the MUX application shown in Figure 32. The use of GPIO1 and GPIO2 as the ADC inputs has the possibility of being digitized within the same conversion sequence as the cell inputs (using the ADCVAX command), thus synchronizing cell voltage and cell current measurements.

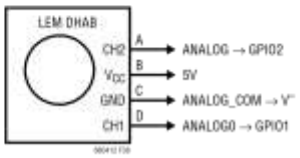


Figure 38. Interfacing a Typical Hall-Effect Battery Current Sensor to Auxiliary ADC Inputs

CURRENT MEASUREMENT WITH A SHUNT RESISTOR

It is possible to measure the battery current on the LTC6804 GPIO pins with a high performance current sense amplifier and a shunt. Figure 39 shows 2 LTC6102s being used to measure the discharge and charge currents on a 12-cell battery stack. To achieve a large dynamic range while maintaining a high level of accuracy the LTC6102 is required. The circuit shown is able to accurately measure ±200Amps to 0.1Amps. The offset of the LTC6102 will only contribute a 20mA error. To maintain a very low sleep current the V_{DRIVE} is used to disable the LTC6102 circuits so that they draw no current when the LTC6804 goes to sleep.

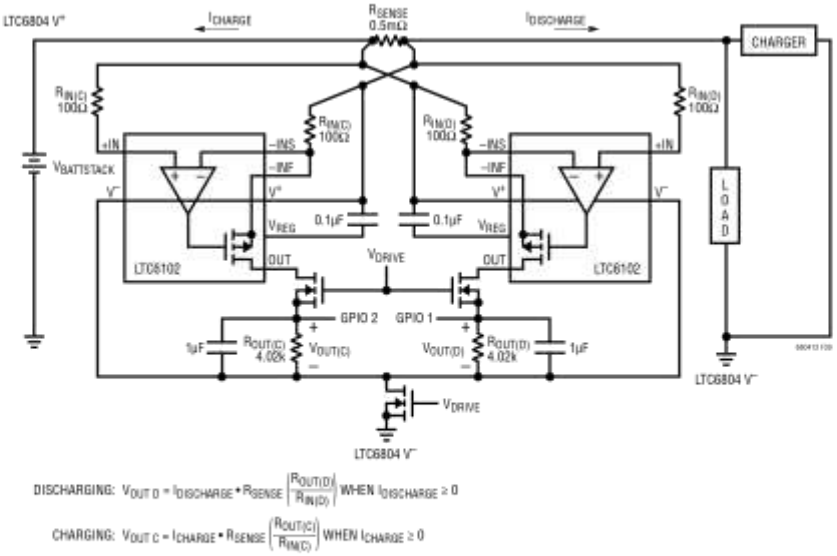


Figure 39. Monitoring Charge and Discharge Currents with a LTC6102

APPLICATIONS INFORMATION

USING THE LTC6804 WITH LESS THAN 12 CELLS

If the LTC6804 is powered by the battery stack, the minimum number of cells that can be monitored by the LTC6804 is governed by the supply voltage requirements of the LTC6804. V^+ must be at least 11V to properly bias the LTC6804. Figure 40 shows an example of the LTC6804 when used to monitor eight cells with best cell measurement synchronization. The 12 cells monitored by the LTC6804 are split into two groups of 6 cells and are measured using two internal multiplexers and two ADCs. To optimize measurement synchronization in applications with less than 12 Cells the unused C pins should be equally distributed between the top of the second mux (C12) and the top of the first mux (C6). If there are an odd number of cells being used, the top mux should have fewer cells connected. The unused cell channels should be tied to the other unused channels on the same mux and then connected to the battery stack through a 100Ω resistor. The unused inputs will result in a reading of 0V for those cells channels. It is also acceptable to connect in the conventional sequence with all unused cell inputs at the top.

isoSPI IBIAS and ICMP Setup

The LTC6804 allows the isoSPI links of each application to be optimized for power consumption or for noise immunity. The power and noise immunity of an isoSPI system is determined by the programmed I_B current, which controls the isoSPI signaling currents. Bias current I_B can range from 100µA to 1mA. Internal circuitry scales up this bias current to create the isoSPI signal currents equal to $20 \bullet I_B$. A low I_B reduces the isoSPI power consumption in the READY and ACTIVE states, while a high I_B increases the amplitude of the differential signal voltage V_A across the matching termination resistor, R_M. The I_B current is programmed by the sum of the R_{B1} and R_{B2} resistors connected between the 2V IBIAS pin and GND as shown in

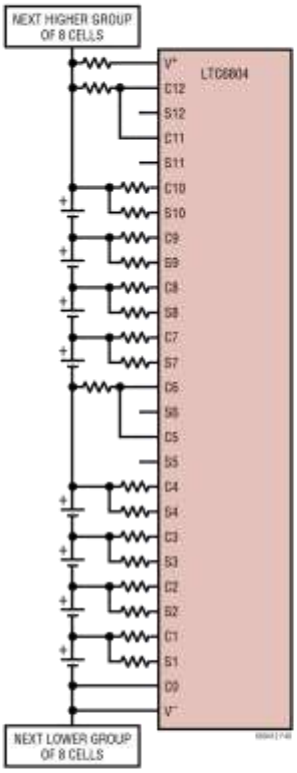


Figure 40. 8 Cell Connection Scheme

Figure 41. The receiver input threshold is set by the ICMP voltage that is programmed with the resistor divider created by the R_{B1} and R_{B2} resistors. The receiver threshold will be half of the voltage present on the ICMP pin.

APPLICATIONS INFORMATION

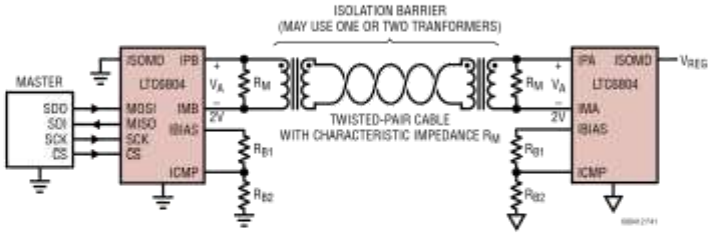


Figure 41. isoSPI Circuit

The following guidelines should be used when setting the bias current (100µA to 1mA) I_B and the receiver comparator threshold voltage V_{ICMP}/2:

R_M = Transmission Line Characteristic Impedance Z_0

Signal Amplitude $V_A = (20 \bullet I_B) \bullet (R_M/2)$

V_{TCMP} (Receiver Comparator Threshold) = $K \bullet V_A$

V_{ICMP} (voltage on ICMP pin) = $2 \bullet V_{TCMP}$

$R_{B2} = V_{ICMP}/I_B$

$R_{B1} = (2/I_B) - R_{B2}$

Select I_B and K (Signal Amplitude V_A to Receiver Comparator Threshold ratio) according to the application:

For lower power links: I_B = 0.5mA and K = 0.5

For full power links: I_B = 1mA and K = 0.5

For long links (>50m): I_B = 1mA and K = 0.25

For addressable multi-drop: I_B = 1mA and K = 0.4

For applications with little system noise, setting I_B to 0.5mA is a good compromise between power consumption and noise immunity. Using this I_B setting with a 1:1 transformer and R_M = 100Ω, R_{B1} should be set to 3.01k and R_{B2} set to 1k. With typical CAT5 twisted pair, these settings will allow for communication up to 50m. For applications in very noisy environments or that require cables longer than 50m it is recommended to increase I_B to 1mA. Higher drive current compensates for the increased insertion loss in the cable and provides high noise immunity. When using cables over 50m and a transformer with a 1:1 turns ratio and R_M = 100Ω, R_{B1} would be 1.5k and R_{B2} would be 499Ω.

The maximum clock rate of an isoSPI link is determined by the length of the isoSPI cable. For cables 10 meters or less, the maximum 1MHz SPI clock frequency is possible. As the length of the cable increases, the maximum possible SPI clock rate decreases. This dependence is a result of the increased propagation delays that can create possible timing violations. Figure 42 shows how the maximum data rate reduces as the cable length increases when using a CAT5 twisted pair.

Cable delay affects three timing specifications: t_{CLK}, t_6 and t_7. In the Electrical Characteristics table, each of these specifications is de-rated by 100ns to allow for 50ns of cable delay. For longer cables, the minimum timing parameters may be calculated as shown below:

t_{CLK}, t_6 and $t_7 > 0.9\mu s + 2 \bullet t_{CABLE}$(0.2m per ns)

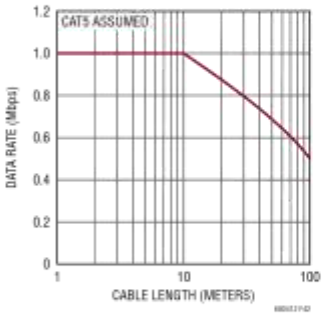


Figure 42. Data Rate vs Cable Length

APPLICATIONS INFORMATION

Implementing a Modular isoSPI Daisy Chain

The hardware design of a daisy-chain isoSPI bus is identical for each device in the network due to the daisy-chain point-to-point architecture. The simple design as shown in Figure 41 is functional, but inadequate for most designs. The termination resistor R_M should be split and bypassed with a capacitor as shown in Figure 43. This change provides both a differential and a common mode termination, and as such, increases the system noise immunity.

The use of cables between battery modules, particularly in automotive applications, can lead to increased noise susceptibility in the communication lines. For high levels of electromagnetic interference (EMC), additional filtering is recommended. The circuit example in Figure 43 shows the use of common mode chokes (CMC)to add common mode noise rejection from transients on the battery lines. The use of a center tapped transformer will also provide additional noise performance. A bypass capacitor connected to the center tap creates a low impedance for common mode noise (Figure 43b). Since transformers without a center tap can be less expensive, they may be preferred. In this case, the addition of a split termination resistor and a bypass capacitor (Figure 43a) can enhance the isoSPI performance. Large center tap capacitors greater than 10nF should be avoided as they may prevent the isoSPI common mode voltage from settling. Common mode chokes similar to those used in Ethernet or CANbus applications are recommended. Specific examples are provided in Table 49.

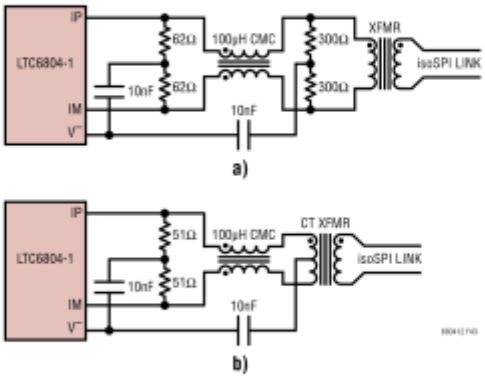


Figure 43. Daisy Chain Interface Components

An important daisy chain design consideration is the number of devices in the isoSPI network. The length of the chain determines the serial timing and affects data latency and throughput. The maximum number of devices in an isoSPI daisy chain is strictly dictated by the serial timing requirements. However, it is important to note that the serial read back time, and the increased current consumption, might dictate a practical limitation.

For a daisy chain, two timing considerations for proper operation dominate (see Figure 20):

1. t_6, the time between the last clock and the rising chip select, must be long enough.
2. t_5, the time from a rising chip select to the next falling chip select (between commands), must be long enough.

Both t_5 and t_6 must be lengthened as the number of LTC6804 devices in the daisy chain increases. The equations for these times are below:

$t_5 > (\#devices \bullet 70ns) + 900ns$

$t_6 > (\#devices \bullet 70ns) + 950ns$

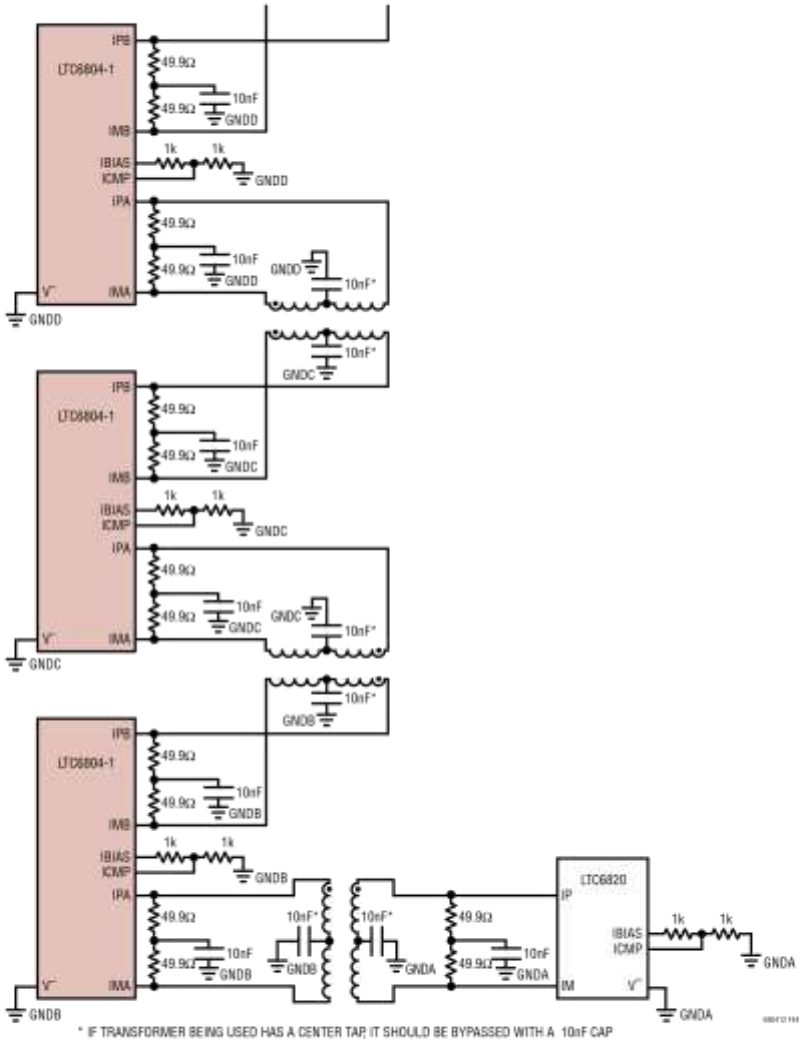


Figure 44. Daisy Chain Interface Components on Single Board

APPLICATIONS INFORMATION

Connecting Multiple LTC6804-1s on the Same PCB

When connecting multiple LTC6804-1 devices on the same PCB, only a single transformer is required between the LTC6804-1 isoSPI ports. The absence of the cable also reduces the noise levels on the communication lines and often only a split termination is required. Figure 44 shows an example application that has multiple LTC6804-1s on the same PCB, communicating to the bottom MCU through an LTC6820 isoSPI driver. If a transformer with a center tap is used, a capacitor can be added for better noise rejection. Additional noise filtering can be provided with discrete common mode chokes (not shown) placed to both sides of the single transformer.

On single board designs with low noise requirements, it is possible for a simplified capacitor-isolated coupling as shown in Figure 45 to replace the transformer. Dual Zener diodes are used at each IC to clamp the common mode voltage to stay within the receiver's input range. The optional common mode choke (CMC) provides noise rejection with symmetrically tapped termination. The 590Ω resistor creates a resistor divider with the termination resistors and attenuates common mode noise. The 590Ω value is chosen to provide the most noise attenuation while maintaining sufficient differential signal. The circuit is designed such that I_B and V_{ICMP} are the same as would be used for a transformer based system with cables over 50m.

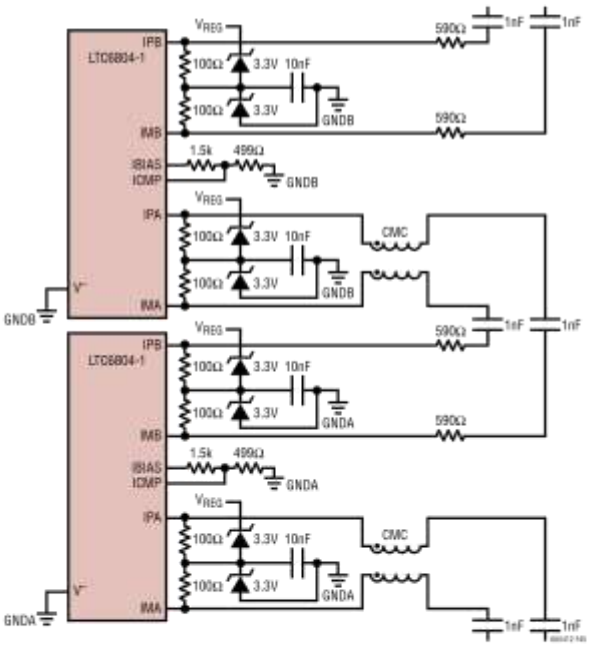


Figure 45. Capacitive Isolation Coupling for LTC6804-1s on the Same PCB

APPLICATIONS INFORMATION

Connecting an MCU to an LTC6804-1 with an isoSPI Data Link

The LTC6820 will convert standard 4-wire SPI into a 2-wire isoSPI link that can communicate directly with the LTC6804. An example is shown in Figure 46. The LTC6820 can be used in applications to provide isolation between the microcontroller and the stack of LTC6804s. The LTC6820 also enables system configurations that have the BMS controller at a remote location relative to the LTC6804 devices and the battery pack.

Configuring the LTC6804-2 in a Multi–Drop isoSPI Link

The addressing feature of the LTC6804-2 allows multiple devices to be connected to a single isoSPI master by distributing them along one twisted pair, essentially creating a large parallel SPI network. A basic multi-drop system is shown in Figure 47; the twisted pair is terminated only at the beginning (master) and the end of the cable. In between, the additional LTC6804-2s are connected to short stubs on the twisted pair. These stubs should be kept short, with as little capacitance as possible, to avoid degrading the termination along the isoSPI wiring.

When an LTC6804-2 is not addressed, it will not transmit data pulses. This scheme eliminates the possibility for collisions since only the addressed device returns data to the master. Generally, multi-drop systems are best confined to compact assemblies where they can avoid excessive isoSPI pulse-distortion and EMC pickup.

Basic Connection of the LTC6804-2 in a Multi-Drop Configuration

In a multi-drop isoSPI bus, placing the termination at the ends of the transmission line provides the best performance (with 100Ω typically). Each of the LTC6804 isoSPI ports should couple to the bus with a resistor network, as shown in Figure 48a. Here again, a center-tapped transformer offers the best performance and a common mode choke (CMC) increases the noise rejection further, as shown in Figure 48b. Figure 48b also shows the use of an RC snubber at the IC connections as a means to suppress resonances (the IC capacitance provides sufficient out-of-band rejection). When using a non-center-tapped transformer, a virtual CT can be generated by connecting a CMC as a voltage-splitter. Series resistors are recommended to decouple the LTC6804 and board parasitic capacitance from the transmission line. Reducing these parasitics on the transmission line will minimize reflections.

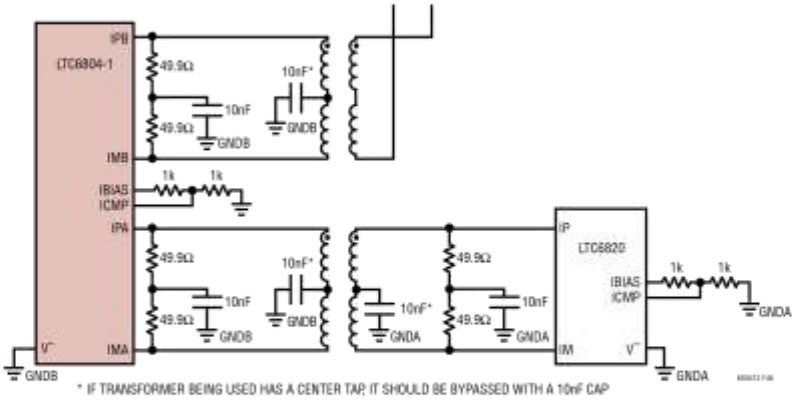


Figure 46. Interfacing an LTC6804-1 with a µC Using an LTC6820 for Isolated SPI Control

APPLICATIONS INFORMATION

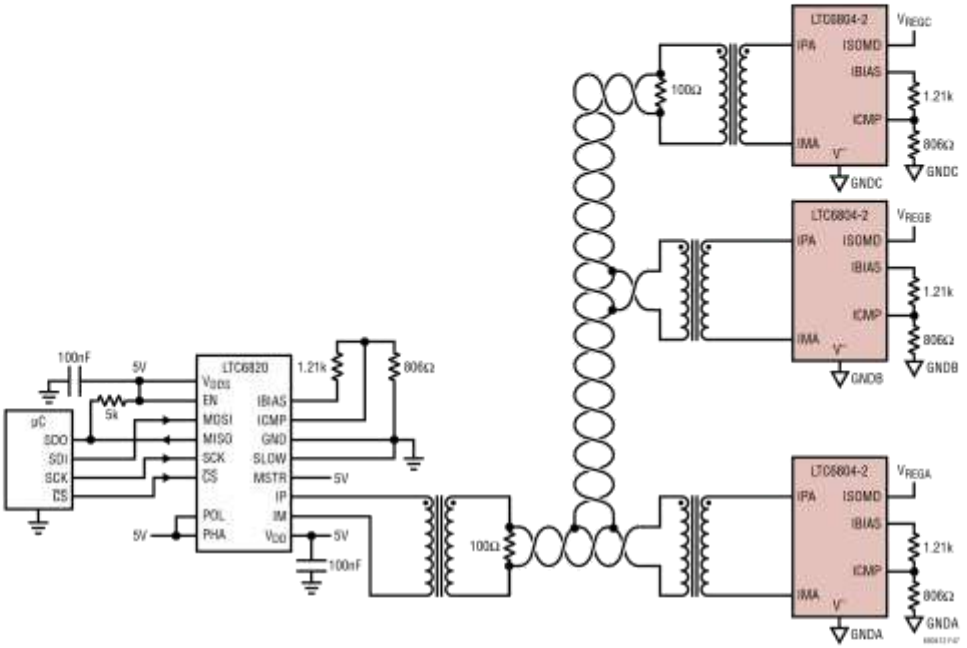


Figure 47. Connecting the LTC6804-2 in a Multi-Drop Configuration

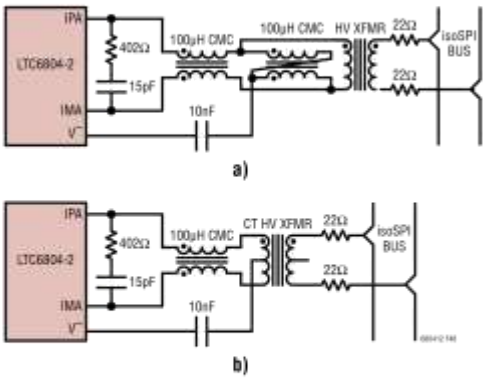


Figure 48. Preferred isoSPI Bus Couplings For Use With LTC6804-2

APPLICATIONS INFORMATION

Table 48. Recommended Transformers

MANUFACTURER	PART NUMBER	TEMPERATURE RANGE	$V_{WORKING}$	V_{HIPOT}/60s	CT	CMC	H	L	W (W/LEADS)	PINS	AEC-Q200
Dual Transformers											
Pulse	HX1188FNL	–40°C to 85°C	60V (est)	1.5kVrms	●	●	6.0mm	12.7mm	9.7mm	16SMT	–
Pulse	HX0068ANL	–40°C to 85°C	60V (est)	1.5kVrms	●	●	2.1mm	12.7mm	9.7mm	16SMT	–
Pulse	HM2100NL	–40°C to 105°C	1000V	4.3kVdc	–	●	3.4mm	14.7mm	14.9mm	10SMT	●
Pulse	HM2102NL	–40°C to 125°C	1000V	4.3kVdc	●	●	4.9mm	14.8mm	14.7mm	12SMT	●
Sumida	CLP178–C20114	–40°C to 125°C	1000V (est)	3.75kVrms	●	●	9mm	17.5mm	15.1mm	12SMT	–
Sumida	CLP0612–C20115		600Vrms	3.75kVrms	●	–	5.7mm	12.7mm	9.4mm	16SMT	–
Wurth Elektronik	7490140110	–40°C to 85°C	250Vrms	4kVrms	●	●	10.9mm	24.6mm	17.0mm	16SMT	–
Wurth Elektronik	7490140111	0°C to 70°C	1000V (est)	4.5kVrms	●	–	8.4mm	17.1mm	15.2mm	12SMT	–
Wurth Elektronik	749014018	0°C to 70°C	250Vrms	4kVrms	●	●	8.4mm	17.1mm	15.2mm	12SMT	–
Halo	TG110–AE050N5LF	–40°C to 85/125°C	60V (est)	1.5kVrms	●	●	6.4mm	12.7mm	9.5mm	16SMT	●
Single Transformers											
Pulse	PE–68386NL	–40°C to 130°C	60V (est)	1.5kVdc	–	–	2.5mm	6.7mm	8.6mm	6SMT	–
Pulse	HM2101NL	–40°C to 105°C	1000V	4.3kVdc	–	●	5.7mm	7.6mm	9.3mm	6SMT	●
Wurth Elektronik	750340848	–40°C to 105°C	250V	3kVrms	–	–	2.2mm	4.4mm	9.1mm	4SMT	–
Halo	TGR04–6506V6LF	–40°C to 125°C	300V	3kVrms	●	–	10mm	9.5mm	12.1mm	6SMT	–
Halo	TGR04–A6506NA6NL	–40°C to 125°C	300V	3kVrms	●	–	9.4mm	8.9mm	12.1mm	6SMT	●
TDK	ALT4532V–201–T001	–40°C to 105°C	60V (est)	–1kV	●	–	2.9mm	3.2mm	4.5mm	6SMT	●
Halo	TDR04–A550ALLF	–40°C to 105°C	1000V	5kVrms	●	–	6.4mm	8.9mm	16.6mm	6TH	●
Sumida	CEEH96BNP–LTC6804/11	–40°C to 125°C	600V	2.5kVrms	–	–	7mm	9.2mm	12.0mm	4SMT	–
Sumida	CEP99NP–LTC6804	–40°C to 125°C	600V	2.5kVrms	●	–	10mm	9.2mm	12.0mm	8SMT	–
Sumida	ESMIT–4180/A	–40°C to 105°C	250Vrms	3kVrms	–	–	3.5mm	5.2mm	9.1mm	4SMT	●
TDK	VGT10/9EE–204S2P4	–40°C to 125°C	250V (est)	2.8kVrms	●	–	10.6mm	10.4mm	12.7mm	8SMT	–

Transformer Selection Guide

As shown in Figure 41, a transformer or pair of transformers isolates the isoSPI signals between two isoSPI ports. The isoSPI signals have programmable pulse amplitudes up to $1.6V_{P-P}$ and pulse widths of 50ns and 150ns. To be able to transmit these pulses with the necessary fidelity the system requires that the transformers have primary inductances above 60µH and a 1:1 turns ratio. It is also necessary to use a transformer with less than 2.5µH of leakage inductance. In terms of pulse shape the primary inductance will mostly effect the pulse droop of the 50ns and 150ns pulses. If the primary inductance is too low, the pulse amplitude will begin to droop and decay over the pulse period. When the pulse droop is severe enough, the effective pulse width seen by the receiver will drop substantially, reducing noise margin. Some droop is acceptable as long as it is a relatively small percentage of the total pulse amplitude. The leakage inductance primarily affects the rise and fall times of the pulses. Slower rise and fall times will effectively reduce the pulse width. Pulse width is determined by the receiver as the time the signal is above the threshold set at the ICMP pin. Slow rise and fall times cut into the timing margins. Generally it is best to keep pulse edges as fast as possible. When evaluating transformers, it is also worth noting the parallel winding capacitance. While transformers have very good CMRR at low frequency, this rejection will degrade at higher frequencies, largely due to the winding to winding capacitance. When choosing a transformer, it is best to pick one with less parallel winding capacitance when possible.

When choosing a transformer, it is equally important to pick a part that has an adequate isolation rating for the application. The working voltage rating of a transformer is a key spec when selecting a part for an application.

Interconnecting daisy-chain links between LTC6804-1 devices see <60V stress in typical applications; ordinary pulse and LAN type transformers will suffice. Multi-drop connections and connections to the LTC6820, in general, may need much higher working voltage ratings for good long-term reliability. Usually, matching the working voltage to the voltage of the entire battery stack is conservative. Unfortunately, transformer vendors will often only specify one-second HV testing, and this is not equal to the long-term ("permanent") rating of the part. For example, according to most safety standards a 1.5kV rated transformer is expected to handle 230V continuously, and a 3kV device is capable of 1100V long-term, though manufacturers may not always certify to those levels (refer to actual vendor data for specifics). Usually, the higher voltage transformers are called "high-isolation" or "reinforced insulation" types by the suppliers. Table 48 shows a list of transformers that have been evaluated in isoSPI links.

In most applications a common mode choke is also necessary for noise rejection. Table 49 includes a list of suitable CMCs if the CMC is not already integrated into the transformer being used.

Table 49. Recommended Common Mode Chokes

MANUFACTURER	PART NUMBER
TDK	ACT45B-101-2P
Murata	DLW43SH101XK2

isoSPI Layout Guidelines

Layout of the isoSPI signal lines also plays a significant role in maximizing the noise immunity of a data link. The following layout guidelines are recommended:

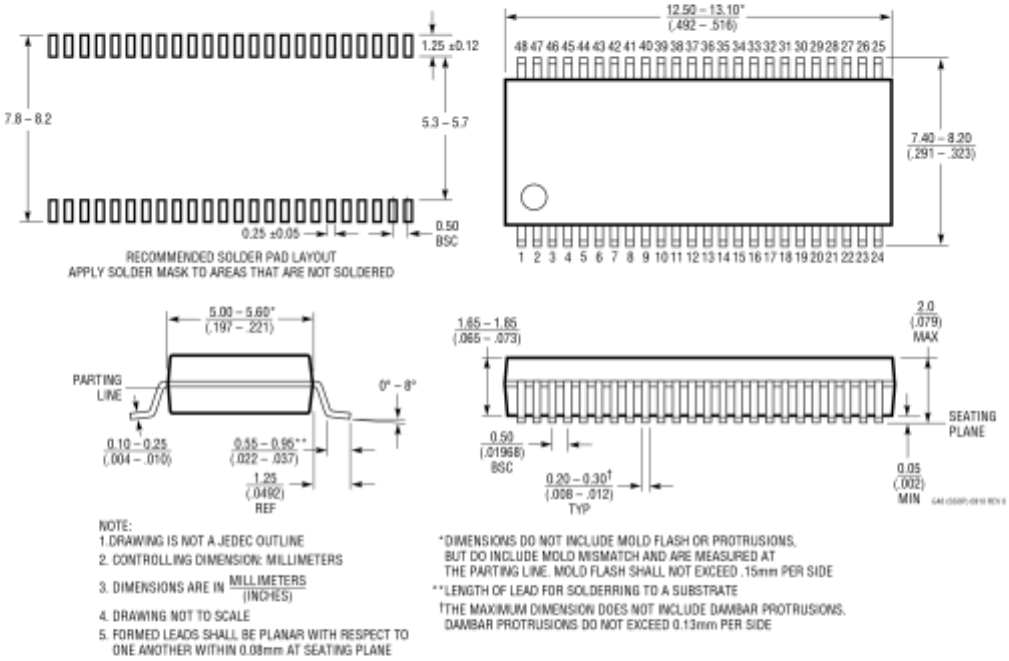
1. The transformer should be placed as close to the isoSPI cable connector as possible. The distance should be kept less than 2cm. The LTC6804 should be placed close to but at least 1cm to 2cm away from the transformer to help isolate the IC from magnetic field coupling.
2. A V^- ground plane should not extend under the transformer, the isoSPI connector or in between the transformer and the connector.
3. The isoSPI signal traces should be as direct as possible while isolated from adjacent circuitry by ground metal or space. No traces should cross the isoSPI signal lines, unless separated by a ground plane on an inner layer.

LTC6804-1/LTC6804-2

PACKAGE DESCRIPTION

Please refer to http://www.linear.com/product/LTC6804-1#packaging for the most recent package drawings.

G Package
48-Lead Plastic SSOP (5.3mm)
(Reference LTC DWG # 05-08-1887 Rev Ø)



NOTE:
1. DRAWING IS NOT A JEDEC OUTLINE
2. CONTROLLING DIMENSION: MILLIMETERS
3. DIMENSIONS ARE IN $\frac{\text{MILLIMETERS}}{(\text{INCHES})}$
4. DRAWING NOT TO SCALE
5. FORMED LEADS SHALL BE PLANAR WITH RESPECT TO ONE ANOTHER WITHIN 0.08mm AT SEATING PLANE

*DIMENSIONS DO NOT INCLUDE MOLD FLASH OR PROTRUSIONS, BUT DO INCLUDE MOLD MISMATCH AND ARE MEASURED AT THE PARTING LINE. MOLD FLASH SHALL NOT EXCEED .15mm PER SIDE
**LENGTH OF LEAD FOR SOLDERRING TO A SUBSTRATE
†THE MAXIMUM DIMENSION DOES NOT INCLUDE DAMBAR PROTRUSIONS. DAMBAR PROTRUSIONS DO NOT EXCEED 0.13mm PER SIDE

6804121c

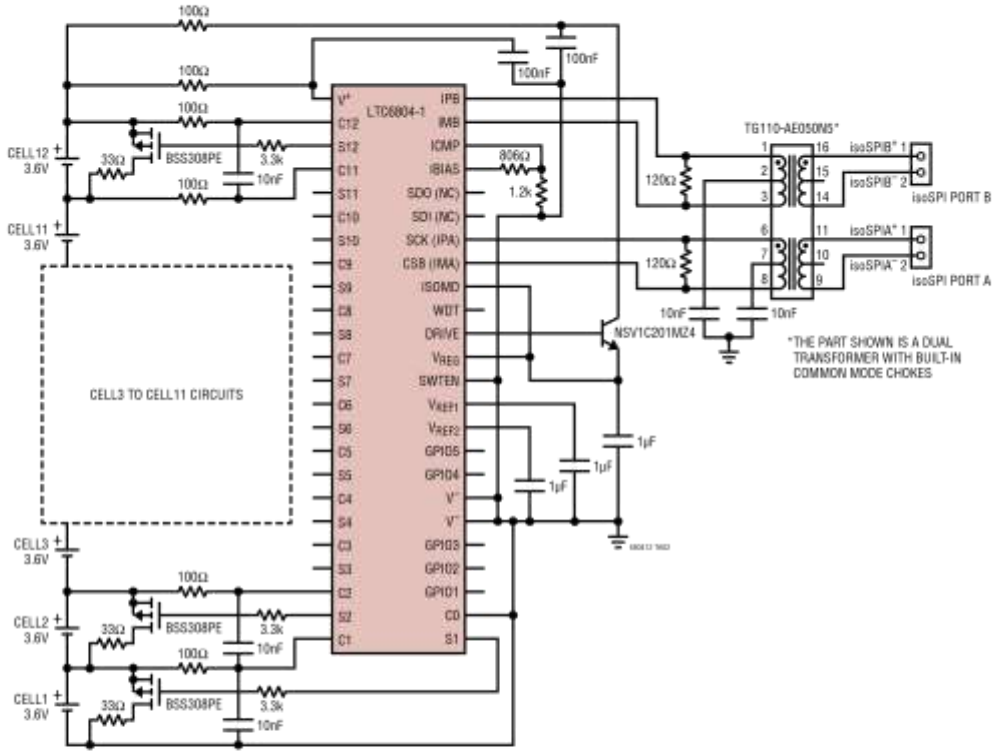
REVISION HISTORY

REV	DATE	DESCRIPTION	PAGE NUMBER
A	10/13	T_{JMAX} corrected from 125°C to 150°C	3
		WDT pin description updated	17, 30, 56, 57
		Information added to Recommended Transformers table	68
B	6/14	Correction to TME Test Conditions, V(CO) = V⁻	4, 5
		Description of T_{SLEEP} added to STANDBY State Discussion	20
		Correction to Temperature Range for TMS Spec, 125°C instead of 85°C	22
		Note regarding potential differences between CO and V⁻ added	27
		Correction to Measurement Range for Accuracy Check, 2.985V to 3.015V	27, 51
		Clarification of CLRSTAT command, which also clears RSVD bits	28
		Description of Reserved Bits Added	30, 51
		Clarification: Watchdog timer is reset by Qualified Wake-up Signal	30
		Clarification: SPI master supports only SPI mode 3	31
		Correction to data register, Dn[3:0] changed to Dn[7:0]	32
		Discussion of Address, Broadcast and Polling Commands edited for Clarity	43-46
C	10/16	Absolute maximum voltage between V⁺ to C12 Added	2
		Note added in table to define I_B	22
		Explanation added for issuing ADSTAT command with CHST = 100	27, 50
		Table 18 (read codes for I^2C master operation) added	33
		Explanation of setting SPI strength using R_{B1} and R_{B2}	36
		Explanation of the SPI terminating resistor, R_M	37
		Explanation of SPI termination and use of a single LTC6804	38
		Figure 18 added to single LTC6804 SPI termination	40
		Explanation of waking up the LTC6804 daisy chain	43
		Note added to Fully Isolated Power section to include a diode from V+ to top of cell	59
		Section added for isoSPI IBIAS and ICMP setup	67, 68
		Section added for modular isoSPI daisy chain	69
		Section added for multiple LTC6804s on the same PCB	71
		Section added for connecting an MCU to an LTC6804-1	72
		Section added for configuring an LTC6804-2 multidrop	72
		Section added for basic connection of an LTC6804-2 multidrop	72
		Figure 47, 48 added to show isoSPI connections	73
		Section added for Transformer Selection Guide	74
		Section added for isoSPI Layout Guidelines	75

LTC6804-1/LTC6804-2

TYPICAL APPLICATION

Basic 12-Cell Monitor with isoSPI Daisy Chain



RELATED PARTS

PART NUMBER	DESCRIPTION	COMMENTS
LTC6801	Independent Multicell Battery Stack Fault Monitor	Monitors Up to 12 Series-Connected Battery Cells for Undervoltage or Overvoltage. Companion to LTC6802, LTC6803 and LTC6804
LTC6802	Precision Multicell Battery Stack Monitor	1st Generation: Superseded by the LTC6804 and LTC6803 for New Designs
LTC6803	Precision Multicell Battery Stack Monitor	2nd Generation: Functionally Enhanced and Pin Compatible to the LTC6802
LTC6820	Isolated Bidirectional Communications Interface for SPI	Provides an Isolated Interface for SPI Communication Up to 100 Meters, Using a Twisted Pair. Companion to the LTC6804
LTC3300	High Efficiency Bidirectional Multicell Battery Balancer	Bidirectional Synchronous Flyback Balancing of Up to 6 Li-Ion or LiFePO4 Cells in Series. Up to 10A Balancing Current (Set by External Components). Bidirectional Architecture Minimizes Balancing Time and Power Dissipation. Up to 92% Charge Transfer Efficiency. 48-Lead Exposed Pad QFN and LQFP Packages

3.2. RESISTENCIAS DE BALANCEO (3521 TE)

SMD Power Resistors

Type 3521 Series

Key Features

- 2 Watts at 70°C
- Small Size to Power Ratio
- Supplied on Tape
- Available via Distribution
- Value Marked on Resistor
- 500 Volt Maximum Overload
- 250 Volt Working Voltage
- Low Profile
- Terminal Finish - Matte Sn over Ni
- MSL Level 2

TE Connectivity is pleased to introduce this low cost high power device, suitable for auto placement in volume, and for most applications, including high frequency operations, owing to the short lead structure. It is attractively priced and available on 7" reels of 4000 pieces.

Characteristics - Electrical

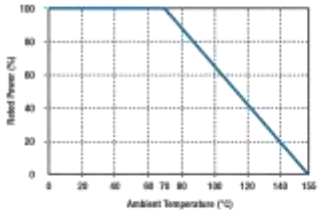
Power Rating:	2 Watts at 70°C**
Max. RCWV*:	250V
Max. Overload Voltage:	500V
Resistance Tolerance(%):	±1%
Resistance Range:	1R0 - 1M0
Temperature Coefficient:	<10R = ±200PPM 10R – 1M0 = ±100PPM >1M0 = ±200PPM
Operating Temperature:	-55°C – 155°C

* Rated continuous working voltage (RCWV) shall be determined from

$$RCWV = \sqrt{\text{Rated Power} \times \text{Resistance Value}}$$, or Maximum RCWV listed above, whichever is less

** Recommended Circuit Board Design - If this device is anticipated to run at full continuous power then action to improve the cooling should be taken. This can be a metal substrate, copper pad left under the chip, an opening in the PCB or enlarged silver conductor pads each end.

Power Derating Curve



For resistors operated in ambient temperatures above 70°C, power rating must be derated in accordance with this curve.

9-1773463-5 OS WR 03/2012

Dimensions are in millimeters and inches unless otherwise specified. Values in brackets are standard equivalents.

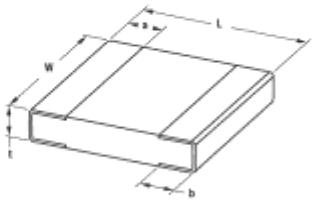
Dimensions are shown for reference purposes only. Specifications subject to change.

For e-mail, phone or live chat, go to: te.com/help

SMD Power Resistors

Type 3521 Series

Dimensions



L	W	a	b	t
6.30 ±0.20	3.20 ±0.20	0.60 ±0.20	0.50 ±0.20	0.55 ±0.10

Marking

4 digit marking system. First three digits are significant figures of resistance, fourth denotes number of zeros eg. 3302 = 33K – 33,000Ω.
For values below 10Ω the letter R is used as decimal point eg. 1R20 = 1R2 = 1.2Ω

Handling Recommendations

When flow soldering - the land width must be smaller than the Chip Resistor width to properly control the solder application. Generally, the land width can be Chip Resistor width (W) x 0.7 to 0.8. When reflow soldering - solder application amount can be adjusted. Thus the land width can be set to W x 1.0 to 1.3.

How to Order

3521	1K0	F	T
Common Part	Resistance Value	Tolerance	Pack Style
3521	1 ohm 1000 milli ohms 1R0 1K ohm 1000 ohms 1K0 1 Meg ohm 1000000 ohms 1M0	F - 1%	T - 4000 / reel

3.3. CONECTORES MOLEX MICRO-FIT 3.0

Micro-Fit 3.0 Connector System Product Family

molex

Applications

Consumer
- Washers and dryers
- Freezers
- Gaming machines
- Printers
- Refrigerators
- Scanners
- Security systems
- Vending machines
- White goods

Telecommunications/Networking
- Routers and switches
- Servers
- Storage

Medical
- Diagnostic equipment
- Patient monitors

Sustainable Energy
- Solar power

Automotive
- Harness manufacturers
- Non-sealed applications
- Inside devices

Refrigerator

Office Equipment

Servers

Specifications

REFERENCE INFORMATION
Packaging:
UL File No.: E29179
CSA File No.: LR19980
Mates With:
Micro-Fit 3.0 Single-Row Plug (Series 43640) and Header (Series 43650);
TPA Single-Row Receptacle (Series 171850)
Micro-Fit 3.0 Dual-Row Plug Housing (Series 43020) and Header (Series 43045 and 44914)
TPA Dual-Row Receptacle (Series 172952);
TPA-Dual Row Plug (Series 203632)
Terminal Used:
Micro-Fit 3.0 female crimp terminals (Series 43030 and 46235)
Designed In: Millimeters
RoHS: Yes
Halogen Free: Yes (Non-Glow-Wire Versions)
Glow Wire Compliant: Option Available, Contact Molex

MECHANICAL
Contact Insertion Force (max.): 14.7N per contact
Contact Retention to Housing (min.): 24.5N
Mating Force (max.): 8N per circuit
Unmating Force (min.): 2.4N per circuit
Durability (max.): typically 30 cycles, refer to product specifications

PHYSICAL
Housing: Nylon
Contact: Various (Phosphor Bronze, Copper Alloy – see Product Specifications for Particular Connector System)
Operating Temperature: -40 to +105°C

ELECTRICAL
Voltage (max.): 600V
Current (max.): 8.5A (Please see product specifications; determined by terminal used)
Contact Resistance (max.): 10 milliohms
Dielectric Withstanding Voltage: <5mA
Insulation Resistance (min.): 1,000 Megohms

Micro-Fit 3.0 Connector System Product Family

molex

Ordering Information

Series No.	Component	
43030	Female Terminal (Available in: 0.25µm Select Tin, 0.38µm and 0.76µm Select Gold)(18 - 30AWG)	
46725	Reduced Mating Force (RMF) Female Terminal, available with or without Lubricant (Available in: 0.25µm Select Tin, 0.38µm and 0.76µm Select Gold)(20 - 30AWG)	
203953	RMF Female Terminal (Available in: 0.25µm Select Tin)(18 AWG)	
43031	Male Terminal (Available in: 0.25µm Select Tin, 0.38µm and 0.76µm Select Gold)(18 - 30AWG)	
43031-6003	TPA Plug Male Terminal (Used with: Housing Series 2006710)(Available in: Pregplated Tin 1µm - 2.5µm, 0.38µm Min Selective Gold Over 2.5µm Min Selective Tin Over 1.27µm Overall Nickel, 0.76µm Min Selective Gold Over 2.5µm Min Selective Tin Over 1.27µm Overall Nickel, 2.5µm Min Matte Tin Over 1.27µm Overall Nickel)(20 – 30AWG)	
43025	Dual-Row Receptacle Housing (mates with 43020, 43045, 44914, 44242) (Available in: UL 94V-0, Glow Wire, Low-Halogen)(2 - 24 Circuits)	
43020	Dual-Row Plug (mates with 43025, 45132, 245132, 172952) (Available in: UL 94V-0, Glow Wire, Low-Halogen) (Panel/Non-Panel Mount)(2 - 24 Circuits)	
43045	Dual-Row Header (mates with 43025, 45132, 245132, 172952) (Available in: 0.25µm Select Tin, 0.38µm and 0.76µm Select Gold)(2 - 24 Circuits) (Right Angle and Vertical Orientations: Surface Mount, Through Hole - Kinked Pin and Through Hole Termination Interface Style)	
43640	Single-Row Receptacle (mates with 43640, 43650) (Available in: UL 94V-0, Glow Wire, Low-Halogen)(2 - 12 Circuits)	
43640	Single-Row Plug (mates with 43645, 171850) (Available in: UL 94V-0, Glow Wire, Low-Halogen)(2 - 12 Circuits) (Panel Mount and Non-Panel Mount)	
200075	Single Row TPA Plug (Mates with Micro-Fit 171850 and 43645) (UL94V-0)(Low Halogen)(Color – Black)(2-7 Circuits)	
43650	Single-Row Header (mates with 43645, 171850) (Available in: 0.25µm Select Tin, 0.38µm and 0.76µm Select Gold)(2 - 12 Circuits) (Right Angle and Vertical Orientations: Surface Mount, Through Hole - Kinked Pin and Through Hole Termination Interface Style)	
171850	TPA Single-Row Receptacle (mates with 43640, 43650) (Available In: UL 94V-0, Low-Halogen)(2 - 7 Circuits)	
172952	TPA Dual-Row Receptacle (2 TPA parts required for each receptacle) (mates with 43020, 43045, 44014) (Available In: UL 94V-0, 94V-2, Glow Wire, Low-Halogen)(4 - 22 Circuits) (Black and Natural Color)	
203632	TPA Dual-Row Plug (Mates with 172952 and 43025)	
172953	TPA Retainer (2 TPA parts required for each receptacle) (Available In: Low-Halogen)(4 - 22 Circuits) (Black and Natural Color)	
46132	Off-the-Shelf Micro-Fit TPA Discrete Wire Cable Assemblies (Available in: 4 - 8 Circuits)(1.0M, 150.00mm, 300.00mm Lengths)	

Micro-Fit 3.0 Connector System Product Family

molex

Ordering Information

Series No.	Component	
44133	BMI Dual Row, Panel Mount Receptacle Housing (mates with 44300, 44432, 44428, 45280) (Available in: UL 94V-0, Halogen-Free)(4 - 24 Circuits)	
44300	BMI Dual Row, Panel Mount Plug Housing (mates with 44133, 44769, 44764) (Available in: 4 - 24 Circuits)	
44428	BMI Dual Row, Surface Mount Compatible, Right Angle Headers (mates with 44769, 44764, 44133) (Available in: Tin, 0.38µm and 0.76µm Select Gold, Low-Halogen)(4 - 24 Circuits)	
44432	BMI Dual Row, Surface Mount Compatible, Vertical Header (mates with 44769, 44764, 44133) (Available in: Tin, 0.38µm and 0.76µm Select Gold, Low-Halogen)(4 - 24 Circuits)	
44764	BMI Dual Row, Right Angle PCB Receptacle Header with Press-fit Plastic Pegs (mates with 44300, 44428, 44432, 45280) (Available in: Tin, 0.38µm and 0.76µm Select Gold, Low-Halogen)(4 - 24 Circuits)	
44769	BMI Dual Row, Vertical Receptacle Header with Press-fit Plastic Pegs (mates with 44300, 44432, 45280, 44428) (Available in: Tin, 0.38µm and 0.76µm Select Gold, Low-Halogen)(4 - 24 Circuits)	
45280	BMI Vertical Compliant Pin Header, Dual Row (mates with 44769, 44764, 44133) (Available in: Tin, 0.38µm and 0.76µm Select Gold, UL 94V-0)(4 - 24 Circuits)	
44914	Compliant Pin Vertical Header, Dual Row (mates with 43025, 172662) (Available in: Tin, 0.38µm and 0.76µm Select Gold, Slow Wire)(4 - 24 Circuits)	
46622	BMI Single Row Header (mates with 46623, 46625) (Available in Tin, 0.38µm and 0.76µm Select Gold, UL 94V-0, Halogen-Free, Right Angle, Vertical)2 - 7 Circuits)	
46623	BMI Single Row Panel Mount Receptacle (mates with 46622) (UL 94V-0, Halogen-Free)(2 - 7 Circuits)	
46625	BMI Single Row Panel Mount Plug (mates with 46622) (UL 94V-0, Halogen-Free)(2 - 7 Circuits)	

Custom Product	Description
Contact Molex	Customized Micro-Fit Overmolded Cable Assemblies
	Customized Micro-Fit Discrete Wire Cable Assemblies

www.molex.com/link/microfit30.html

Order No. 987650-5984 Rev. 5 USA/0k/GF/2018.09

3.4. SONDAS DE TEMPERATURA B57164K0104

TDK

NTC thermistors for temperature measurement

Leaded NTC thermistors,
lead spacing 5 mm

Series/Type: **B57164K**
Date: January 2018

TDK

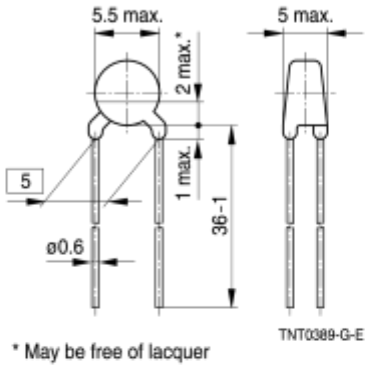
Temperature measurement and compensation	B57164K
Leaded NTC thermistors, lead spacing 5 mm	**K164**

Applications
- Temperature measurement and compensation

Features
- Wide resistance range
- Cost-effective
- Lacquer-coated thermistor disk
- Tinned copper leads
- Lead spacing 5 mm
- Marked with resistance and tolerance

Delivery mode
Bulk (standard), cardboard tape, reeled or in Ammo pack on request

Dimensional drawing



* May be free of lacquer

Dimensions in mm
Approx. weight 370 mg

General technical data

Climatic category	(IEC 60068-1)		55/125/21	
Max. power	(at 25 °C)	P_{25}	450	mW
Resistance tolerance		$\Delta R_R/R_R$	±5, ±10	%
Rated temperature		T_R	25	°C
Dissipation factor	(in air)	δ_{th}	approx. 7.5	mW/K
Thermal cooling time constant	(in air)	τ_c	approx. 20	s
Heat capacity		C_{th}	approx. 150	mJ/K

TDK

Temperature measurement and compensation	B57164K
Leaded NTC thermistors, lead spacing 5 mm	K164

Electrical specification and ordering codes

R_{25} Ω	No. of R/T characteristic	$B_{25/100}$ K	Ordering code
22	1203	2900 ±3%	B57164K0220+000
33	1203	2900 ±3%	B57164K0330+000
47	1302	3000 ±3%	B57164K0470+000
68	1303	3050 ±3%	B57164K0680+000
100	1305	3200 ±3%	B57164K0101+000
150	1305	3200 ±3%	B57164K0151+000
220	1305	3200 ±3%	B57164K0221+000
330	1306	3450 ±3%	B57164K0331+000
470	1306	3450 ±3%	B57164K0471+000
680	1307	3560 ±3%	B57164K0681+000
1 k	1011	3730 ±3%	B57164K0102+000
1.5 k	1013	3900 ±3%	B57164K0152+000
2.2 k	1013	3900 ±3%	B57164K0222+000
3.3 k	4001	3950 ±3%	B57164K0332+000
4.7 k	4001	3950 ±3%	B57164K0472+000
6.8 k	2903	4200 ±3%	B57164K0682+000
10 k	2904	4300 ±3%	B57164K0103+000
15 k	1014	4250 ±3%	B57164K0153+000
22 k	1012	4300 ±3%	B57164K0223+000
33 k	1012	4300 ±3%	B57164K0333+000
47 k	4003	4450 ±3%	B57164K0473+000
68 k	2005	4600 ±3%	B57164K0683+000
100 k	2005	4600 ±3%	B57164K0104+000
150 k	2005	4600 ±3%	B57164K0154+000
220 k	2007	4830 ±3%	B57164K0224+000
330 k	2006	5000 ±3%	B57164K0334+000
470 k	2006	5000 ±3%	B57164K0474+000

+ = Resistance tolerance
J = ±5%
K = ±10%

TDK

Temperature measurement and compensation	B57164K
Leaded NTC thermistors, lead spacing 5 mm	K164

Reliability data

Test	Standard	Test conditions	$\Delta R_{25}/R_{25}$ (typical)	Remarks
Storage in dry heat	IEC 60068-2-2	Storage at upper category temperature T: 125 °C t: 1000 h	< 3%	No visible damage
Storage in damp heat, steady state	IEC 60068-2-78	Temperature of air: 40 °C Relative humidity of air: 93% Duration: 21 days	< 3%	No visible damage
Rapid temperature cycling	IEC 60068-2-14	Lower test temperature: −55 °C Upper test temperature: 125 °C Number of cycles: 10	< 3%	No visible damage
Endurance		P_{max}: 450 mW t: 1000 h	< 3%	No visible damage
Long-term stability (empirical value)		Temperature: 70 °C t: 10000 h	< 5%	No visible damage

Note

- Contact of NTC thermistors with any liquids and solvents shall be prevented. It must be ensured that no water enters the NTC thermistors (e.g. through plug terminals).
- Avoid dewing and condensation unless thermistor is specified for these conditions.

TDK

Temperature measurement and compensation	B57164K
Leaded NTC thermistors, lead spacing 5 mm	K164

R/T characteristics

R/T No.	1011		1012		1013	
T (°C)	$B_{25/100}$ = 3730 K		$B_{25/100}$ = 4300 K		$B_{25/100}$ = 3900 K	
	R_T/R_{25}	α (%/K)	R_T/R_{25}	α (%/K)	R_T/R_{25}	α (%/K)
−55.0	70.014	6.9	87.237	6.8	77.285	7.0
−50.0	49.906	6.7	62.264	6.7	54.938	6.7
−45.0	36.015	6.4	44.854	6.5	39.507	6.5
−40.0	26.296	6.2	32.599	6.3	28.722	6.3
−35.0	19.411	6.0	23.893	6.1	21.099	6.1
−30.0	14.479	5.8	17.654	6.0	15.652	5.9
−25.0	10.903	5.6	13.098	5.8	11.715	5.7
−20.0	8.2923	5.4	9.8059	5.7	8.8541	5.6
−15.0	6.3591	5.2	7.4266	5.5	6.7433	5.4
−10.0	4.9204	5.1	5.6677	5.4	5.1815	5.2
−5.0	3.8279	4.9	4.3213	5.3	4.0099	5.1
0.0	3.0029	4.8	3.3208	5.1	3.1283	4.9
5.0	2.3773	4.6	2.5842	5.0	2.4569	4.8
10.0	1.8959	4.5	2.0238	4.9	1.9438	4.6
15.0	1.5207	4.3	1.5858	4.8	1.5475	4.5
20.0	1.228	4.2	1.2507	4.7	1.2403	4.4
25.0	1.0000	4.1	1.0000	4.5	1.0000	4.3
30.0	0.81779	3.9	0.7964	4.4	0.81104	4.1
35.0	0.67341	3.8	0.64053	4.3	0.66146	4.0
40.0	0.55747	3.7	0.51772	4.2	0.54254	3.9
45.0	0.46357	3.6	0.41958	4.1	0.44727	3.8
50.0	0.3874	3.6	0.34172	4.1	0.37067	3.7
55.0	0.32368	3.5	0.27877	4.0	0.30865	3.6
60.0	0.272	3.4	0.22861	3.9	0.25825	3.5
65.0	0.23041	3.3	0.18872	3.8	0.21707	3.4
70.0	0.19604	3.2	0.15645	3.7	0.18323	3.3
75.0	0.16735	3.1	0.13012	3.6	0.15535	3.3
80.0	0.14342	3.0	0.10863	3.6	0.13223	3.2
85.0	0.12347	3.0	0.091115	3.5	0.11302	3.1
90.0	0.10668	2.8	0.0767	3.4	0.096951	3.0
95.0	0.092734	2.8	0.064867	3.3	0.083487	3.0
100.0	0.080903	2.8	0.055047	3.3	0.072139	2.9
105.0	0.070616	2.7	0.046797	3.2	0.062559	2.8
110.0	0.061826	2.6	0.039904	3.1	0.054425	2.8
115.0	0.054282	2.6	0.034255	3.1	0.047508	2.7
120.0	0.047793	2.5	0.029498	3.0	0.041594	2.6
125.0	0.042249	2.4	0.025448	3.0	0.036532	2.6

Please read *Cautions and warnings* and *Important notes* at the end of this document.

Page 5 of 31

TDK

Temperature measurement and compensation	B57164K
Leaded NTC thermistors, lead spacing 5 mm	K164

R/T characteristics

R/T No.	1014		1203		1302	
T (°C)	$B_{25/100}$ = 4250 K		$B_{25/100}$ = 2900 K		$B_{25/100}$ = 3000 K	
	R_T/R_{25}	α (%/K)	R_T/R_{25}	α (%/K)	R_T/R_{25}	α (%/K)
−55.0	83.935	6.8	30.252	5.6	33.701	5.9
−50.0	60.228	6.6	22.966	5.4	25.252	5.7
−45.0	43.593	6.4	17.612	5.2	19.149	5.4
−40.0	31.815	6.3	13.65	5.0	14.684	5.2
−35.0	23.404	6.1	10.671	4.8	11.38	5.0
−30.0	17.349	6.0	8.4216	4.7	8.9067	4.8
−25.0	12.946	5.8	6.7001	4.5	7.0357	4.6
−20.0	9.7439	5.7	5.3757	4.3	5.6065	4.5
−15.0	7.3737	5.5	4.3443	4.2	4.5044	4.3
−10.0	5.6247	5.4	3.5376	4.1	3.6471	4.2
−5.0	4.3063	5.3	2.8995	3.9	2.9746	4.0
0.0	3.3221	5.2	2.3929	3.8	2.4429	3.9
5.0	2.5779	5.0	1.9866	3.7	2.0194	3.8
10.0	2.0144	4.9	1.6596	3.5	1.6797	3.6
15.0	1.5848	4.8	1.3941	3.4	1.4053	3.5
20.0	1.2547	4.6	1.1777	3.3	1.1823	3.4
25.0	1.0000	4.6	1.0000	3.2	1.0000	3.3
30.0	0.79913	4.4	0.85337	3.1	0.85007	3.2
35.0	0.64287	4.3	0.7317	3.0	0.72608	3.1
40.0	0.51991	4.2	0.63032	2.9	0.623	3.0
45.0	0.42299	4.1	0.54534	2.9	0.53685	2.9
50.0	0.34573	4.1	0.47384	2.8	0.46453	2.9
55.0	0.28298	4.0	0.41336	2.7	0.40357	2.8
60.0	0.23277	3.8	0.36201	2.6	0.35193	2.7
65.0	0.19262	3.8	0.31822	2.5	0.30799	2.6
70.0	0.16005	3.7	0.28073	2.5	0.27047	2.6
75.0	0.13349	3.6	0.2485	2.4	0.23832	2.5
80.0	0.11175	3.5	0.22069	2.3	0.21067	2.4
85.0	0.093934	3.5	0.19663	2.3	0.18677	2.4
90.0	0.079231	3.4	0.17572	2.2	0.16607	2.3
95.0	0.067054	3.3	0.1575	2.2	0.14805	2.3
100.0	0.056932	3.2	0.14157	2.1	0.13233	2.2
105.0	0.048591	3.1	0.1276	2.1	0.11862	2.2
110.0	0.041605	3.1	0.11531	2.0	0.1066	2.1
115.0	0.035653	3.1	0.10447	2.0	0.096009	2.1
120.0	0.030636	3.0	0.094881	1.9	0.086667	2.0
125.0	0.026454	2.9	0.086371	1.9	0.078398	2.0

TDK

Temperature measurement and compensation	B57164K
Leaded NTC thermistors, lead spacing 5 mm	K164

R/T characteristics

R/T No.	1303		1305		1306	
T (°C)	$B_{25/100}$ = 3050 K		$B_{25/100}$ = 3200 K		$B_{25/100}$ = 3450 K	
	R_T/R_{25}	α (%/K)	R_T/R_{25}	α (%/K)	R_T/R_{25}	α (%/K)
−55.0	34.363	5.8	42.131	6.2	49.935	6.3
−50.0	25.827	5.6	31.129	5.9	36.64	6.1
−45.0	19.635	5.4	23.273	5.7	27.18	5.9
−40.0	15.089	5.2	17.592	5.5	20.37	5.7
−35.0	11.712	5.0	13.438	5.3	15.416	5.5
−30.0	9.1774	4.8	10.366	5.0	11.775	5.3
−25.0	7.2552	4.6	8.1005	4.9	9.0698	5.1
−20.0	5.7835	4.5	6.3856	4.8	7.0497	5.0
−15.0	4.6467	4.3	5.0364	4.7	5.5187	4.8
−10.0	3.7611	4.2	4.0067	4.4	4.3558	4.7
−5.0	3.0547	4.1	3.2217	4.3	3.4609	4.5
0.0	2.4986	4.0	2.6097	4.2	2.7705	4.4
5.0	2.0575	3.8	2.126	4.0	2.2313	4.3
10.0	1.7051	3.7	1.7438	3.9	1.8098	4.1
15.0	1.421	3.6	1.4415	3.8	1.4762	4.0
20.0	1.191	3.6	1.1987	3.7	1.2116	3.9
25.0	1.0000	3.3	1.0000	3.5	1.0000	3.8
30.0	0.85053	3.3	0.84185	3.4	0.82984	3.7
35.0	0.72386	3.2	0.7108	3.3	0.6922	3.6
40.0	0.61897	3.1	0.60317	3.2	0.58042	3.5
45.0	0.53134	3.0	0.51419	3.1	0.48899	3.4
50.0	0.45814	2.9	0.44037	3.1	0.41395	3.3
55.0	0.39637	2.9	0.37824	3.0	0.35197	3.2
60.0	0.34439	2.7	0.32636	2.9	0.3006	3.1
65.0	0.30081	2.7	0.28333	2.8	0.2578	3.0
70.0	0.26372	2.6	0.24697	2.7	0.22197	3.0
75.0	0.23212	2.5	0.21573	2.7	0.19189	2.9
80.0	0.20501	2.5	0.18908	2.6	0.16648	2.8
85.0	0.1815	2.4	0.16649	2.5	0.14498	2.7
90.0	0.16117	2.4	0.14709	2.5	0.12669	2.7
95.0	0.1433	2.3	0.13021	2.4	0.11109	2.6
100.0	0.12775	2.2	0.1156	2.3	0.097717	2.5
105.0	0.11458	2.1	0.10301	2.3	0.086235	2.5
110.0	0.10306	2.1	0.092038	2.2	0.076325	2.4
115.0	0.092752	2.1	0.082442	2.2	0.06776	2.4
120.0	0.083677	2.0	0.074035	2.1	0.06032	2.3
125.0	0.075739	2.0	0.066701	2.1	0.053852	2.2

TDK

Temperature measurement and compensation	B57164K
Leaded NTC thermistors, lead spacing 5 mm	K164

R/T characteristics

R/T No.	1307		2005		2006	
T (°C)	$B_{25/100}$ = 3560 K		$B_{25/100}$ = 4600 K		$B_{25/100}$ = 5000 K	
	R_T/R_{25}	α (%/K)	R_T/R_{25}	α (%/K)	R_T/R_{25}	α (%/K)
−55.0	51.115	5.7	120.22	7.0	200.55	8.7
−50.0	38.3	5.7	85.48	6.9	131.02	8.3
−45.0	28.847	5.6	61.004	6.8	87.171	8.0
−40.0	21.842	5.5	43.712	6.7	58.988	7.7
−35.0	16.627	5.4	31.459	6.6	40.545	7.4
−30.0	12.725	5.3	22.746	6.6	28.272	7.1
−25.0	9.7859	5.2	16.49	6.4	19.997	6.9
−20.0	7.5902	5.1	12.071	6.3	14.292	6.6
−15.0	5.8918	5.0	8.8455	6.1	10.35	6.4
−10.0	4.6124	4.8	6.5446	6.0	7.5614	6.4
−5.0	3.6247	4.7	4.8852	5.8	5.5343	6.2
0.0	2.8717	4.6	3.6781	5.6	4.086	6.0
5.0	2.2929	4.4	2.7944	5.4	3.0374	5.9
10.0	1.8442	4.3	2.1391	5.3	2.276	5.7
15.0	1.4941	4.2	1.6507	5.1	1.7188	5.6
20.0	1.2183	4.0	1.2823	5.1	1.3074	5.5
25.0	1.0000	3.9	1.0000	5.0	1.0000	5.3
30.0	0.82246	3.8	0.78393	4.8	0.76988	5.2
35.0	0.68231	3.7	0.61822	4.7	0.5954	5.1
40.0	0.56909	3.6	0.49053	4.6	0.46341	4.9
45.0	0.4767	3.5	0.39116	4.5	0.36327	4.8
50.0	0.40133	3.4	0.31371	4.3	0.28636	4.8
55.0	0.33894	3.3	0.25338	4.2	0.2262	4.7
60.0	0.28769	3.2	0.20565	4.2	0.17974	4.5
65.0	0.24573	3.1	0.16762	4.1	0.1438	4.4
70.0	0.21081	3.0	0.13726	4.0	0.1156	4.3
75.0	0.18147	3.0	0.11279	3.9	0.093296	4.3
80.0	0.15682	2.9	0.093053	3.8	0.075623	4.2
85.0	0.13601	2.8	0.077177	3.7	0.061619	4.1
90.0	0.11838	2.7	0.064263	3.6	0.050414	3.9
95.0	0.10342	2.7	0.053678	3.6	0.041532	3.8
100.0	0.090648	2.6	0.044996	3.5	0.034355	3.8
105.0	0.079672	2.5	0.037917	3.4	0.028525	3.7
110.0	0.070236	2.5	0.032063	3.4	0.023774	3.7
115.0	0.062118	2.4	0.027161	3.3	0.019852	3.6
120.0	0.055093	2.4	0.023079	3.2	0.016632	3.5
125.0	0.048901	2.3	0.01968	3.2	0.014016	3.4

TDK

Temperature measurement and compensation	B57164K
Leaded NTC thermistors, lead spacing 5 mm	K164

R/T characteristics

R/T No.	2007		2903		2904	
T (°C)	$B_{25/100}$ = 4830 K		$B_{25/100}$ = 4200 K		$B_{25/100}$ = 4300 K	
	R_T/R_{25}	α (%/K)	R_T/R_{25}	α (%/K)	R_T/R_{25}	α (%/K)
−55.0	185.87	8.4	120.03	7.7	121.46	7.4
−50.0	123.23	8.1	82.38	7.4	84.439	7.2
−45.0	82.888	7.8	57.248	7.2	59.243	7.1
−40.0	56.544	7.6	40.255	7.0	41.938	6.9
−35.0	39.061	7.3	28.627	6.7	29.947	6.7
−30.0	27.321	7.1	20.577	6.6	21.567	6.6
−25.0	19.326	6.8	14.876	6.4	15.641	6.3
−20.0	13.823	6.6	10.88	6.1	11.466	6.2
−15.0	10.001	6.4	8.0808	5.9	8.451	6.0
−10.0	7.3067	6.4	6.0612	5.8	6.2927	5.9
−5.0	5.3454	6.2	4.5649	5.6	4.7077	5.7
0.0	3.9484	5.9	3.4708	5.4	3.5563	5.5
5.0	2.9595	5.7	2.6625	5.2	2.7119	5.3
10.0	2.2358	5.6	2.0599	5.1	2.086	5.1
15.0	1.7001	5.4	1.6069	4.9	1.6204	5.0
20.0	1.3021	5.4	1.2631	4.8	1.2683	4.8
25.0	1.0000	5.2	1.0000	4.6	1.0000	4.7
30.0	0.7756	5.0	0.79593	4.5	0.7942	4.6
35.0	0.60507	4.9	0.63796	4.4	0.63268	4.5
40.0	0.47498	4.8	0.51467	4.2	0.5074	4.3
45.0	0.37533	4.7	0.41887	4.1	0.41026	4.2
50.0	0.29823	4.6	0.34272	4.0	0.33363	4.1
55.0	0.23763	4.5	0.28081	3.9	0.27243	4.0
60.0	0.19041	4.4	0.23141	3.8	0.2237	3.9
65.0	0.15356	4.3	0.19211	3.7	0.18459	3.8
70.0	0.12442	4.2	0.16027	3.6	0.15305	3.7
75.0	0.10131	4.1	0.13421	3.5	0.12755	3.6
80.0	0.08286	4.0	0.11288	3.4	0.10677	3.5
85.0	0.068004	3.9	0.095326	3.3	0.089928	3.4
90.0	0.056032	3.8	0.080828	3.2	0.076068	3.3
95.0	0.046379	3.8	0.068916	3.2	0.064524	3.3
100.0	0.038533	3.7	0.058989	3.1	0.054941	3.2
105.0	0.032169	3.6	0.050701	3.0	0.047003	3.1
110.0	0.026952	3.5	0.043735	3.0	0.040358	3.0
115.0	0.022658	3.4	0.037778	2.9	0.034743	3.0
120.0	0.019111	3.3	0.032736	2.8	0.030007	2.9
125.0	0.016201	3.3	0.028513	2.7	0.026006	2.8
130.0	–	–	–	–	0.022609	2.8
135.0	–	–	–	–	0.01972	2.7
140.0	–	–	–	–	0.017251	2.6
145.0	–	–	–	–	0.015139	2.6
150.0	–	–	–	–	0.013321	2.5
155.0	–	–	–	–	0.011754	2.5

Please read *Cautions and warnings* and *Important notes* at the end of this document.

Page 9 of 31

TDK

Temperature measurement and compensation	B57164K
Leaded NTC thermistors, lead spacing 5 mm	K164

R/T characteristics

R/T No.	4001		4003	
T (°C)	$B_{25/100}$ = 3950 K		$B_{25/100}$ = 4450 K	
	R_T/R_{25}	α (%/K)	R_T/R_{25}	α (%/K)
−55.0	88.052	7.3	103.81	6.8
−50.0	61.65	7.0	73.707	6.7
−45.0	43.727	6.8	52.723	6.6
−40.0	31.395	6.5	37.988	6.5
−35.0	22.802	6.3	27.565	6.4
−30.0	16.742	6.2	20.142	6.2
−25.0	12.367	6.0	14.801	6.1
−20.0	9.2353	5.6	10.976	5.9
−15.0	7.0079	5.4	8.1744	5.8
−10.0	5.3654	5.4	6.1407	5.7
−5.0	4.126	5.2	4.6331	5.5
0.0	3.2	5.0	3.5243	5.4
5.0	2.4986	4.9	2.6995	5.3
10.0	1.9662	4.7	2.0831	5.1
15.0	1.5596	4.6	1.6189	5.0
20.0	1.2457	4.5	1.2666	4.9
25.0	1.0000	4.4	1.0000	4.7
30.0	0.80355	4.2	0.78351	4.6
35.0	0.65346	4.1	0.62372	4.5
40.0	0.53456	4.0	0.49937	4.4
45.0	0.43966	3.9	0.40218	4.3
50.0	0.36357	3.8	0.32557	4.2
55.0	0.30183	3.7	0.26402	4.1
60.0	0.25189	3.6	0.21527	4.0
65.0	0.21136	3.5	0.17693	3.9
70.0	0.17819	3.4	0.14616	3.8
75.0	0.15089	3.3	0.12097	3.7
80.0	0.12833	3.2	0.10053	3.7
85.0	0.10948	3.1	0.083761	3.6
90.0	0.093748	3.0	0.070039	3.5
95.0	0.080764	2.9	0.058937	3.4
100.0	0.069842	2.9	0.049777	3.4
105.0	0.060455	2.9	0.042146	3.3
110.0	0.052498	2.8	0.035803	3.2
115.0	0.04574	2.7	0.030504	3.2
120.0	0.039972	2.7	0.026067	3.1
125.0	0.034984	2.6	0.022332	3.0
130.0	–	–	0.019186	3.0
135.0	–	–	0.016515	2.9
140.0	–	–	0.014253	2.9
145.0	–	–	0.012367	2.8
150.0	–	–	0.010758	2.8
155.0	–	–	0.0093933	2.7

TDK

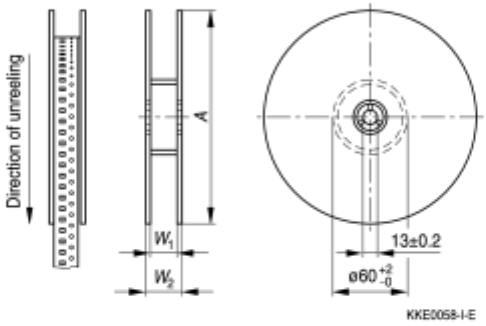
Temperature measurement and compensation	B57164K
Leaded NTC thermistors, lead spacing 5 mm	K164

Taping and packing

1 Taping of SMD NTC thermistors

Tape and reel packing according to IEC 60286-3.
Tape material: Cardboard or blister, tape width 8 ±0.30 mm

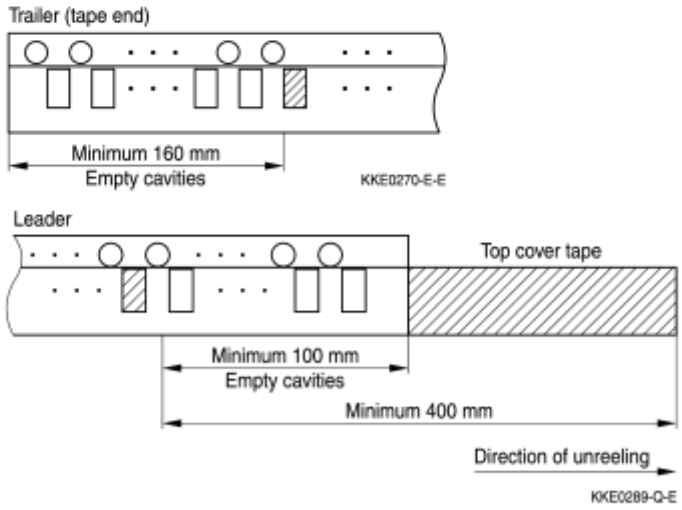
2 Reel packing



Dimensions in mm

	8-mm tape	
	180-mm reel	330-mm reel
A	180 +0/−3	330 +0/−2.0
W_1	8.4 +1.5/−0	8.4 +1.5/−0
W_2	14.4 max.	14.4 max.

Leader, trailer



TDK

Temperature measurement and compensation	B57164K
Leaded NTC thermistors, lead spacing 5 mm	K164

Packing units for discrete chip




Case size	Chip thickness	Cardboard tape	Blister tape	∅ 180-mm reel	∅ 330-mm reel
inch/mm	th	W	W	pcs.	pcs.
0402/1005	0.5 mm	8 mm	–	10000	50000
0603/1608	0.8 mm	8 mm	8 mm	4000	16000
0805/2012	0.8 mm	–	8 mm	2000/ 4000	16000
	1.2 mm	–	8 mm	3000	12000
1206/3216	0.8 mm	–	8 mm	2000	12000
	1.2 mm	–	8 mm	4000	12000

3 Packing codes

The last two digits of the complete ordering code state the packing mode:

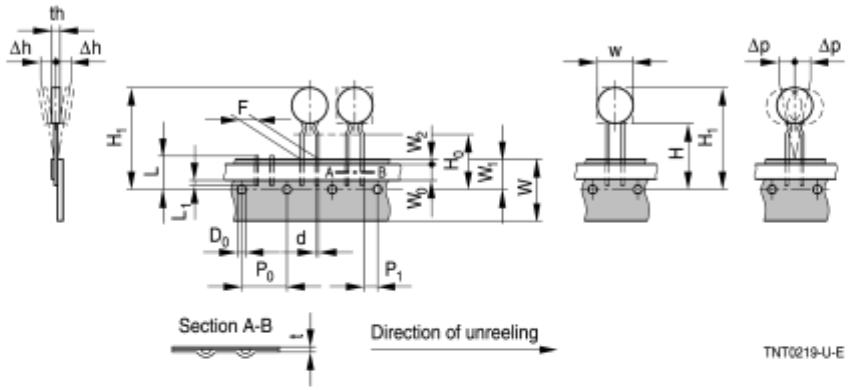
Last two digits			
60	SMD	Cardboard tape	180-mm reel packing
62	SMD	Blister tape	180-mm reel packing
70	SMD	Cardboard tape	330-mm reel packing
72	SMD	Blister tape	330-mm reel packing

TDK

4 Taping of radial leaded NTC thermistors

Dimensions and tolerances

Lead spacing F = 2.5 mm and 5.0 mm (taping to IEC 60286-2)



Dimensions (mm)

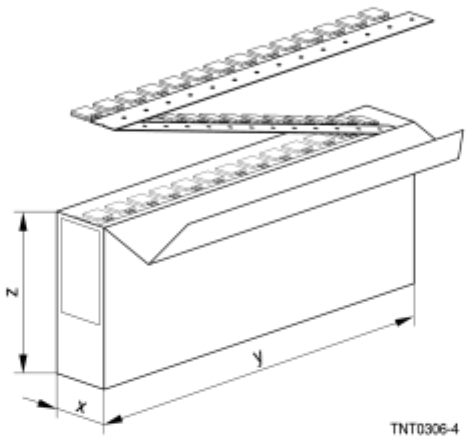
	Lead spacing 2.5 mm	Lead spacing 5 mm	Tolerance of lead spacing 2.5/5 mm	Remarks
w	11.0	11.5	max.	
th	5.0	6.0	max.	
d	0.5/0.6	0.5/0.6	±0.05	
P_0	12.7	12.7	±0.3	±1 mm / 20 sprocket holes
P_1	5.1	3.85	±0.7	
F	2.5	5.0	+0.6/−0.1	
Δh	0	0	±2.0	measured at top of component body
Δp	0	0	±1.3	
W	18.0	18.0	±0.5	
W_0	5.5	5.5	min.	peel-off force ≥5 N
W_1	9.0	9.0	+0.75/−0.5	
W_2	3.0	3.0	max.	
H	18.0	18.0	+2.0/−0	
H_0	16.0	16.0	±0.5	
H_1	32.2	32.2	max.	
D_0	4.0	4.0	±0.2	
t	0.9	0.9	max.	without wires
L	11.0	11.0	max.	
L_1	4.0	4.0	max.	

TDK

Temperature measurement and compensation	B57164K
Leaded NTC thermistors, lead spacing 5 mm	**K164**

Types of packing

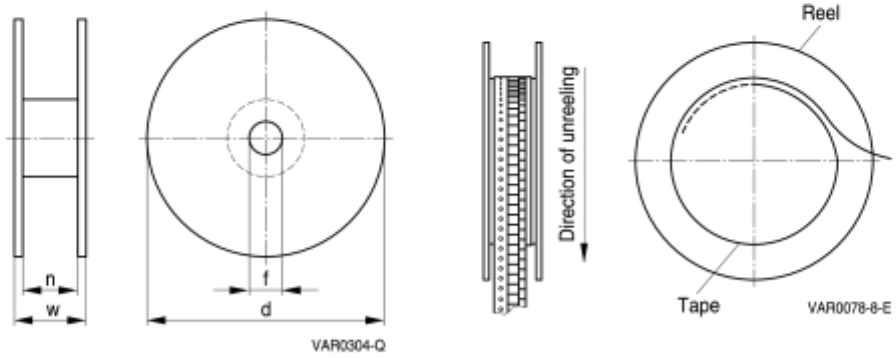
Ammo packing



Ammo type	x	y	z
I	80	240	210

Packing unit: 1000 - 2000 pcs./reel

Reel packing



Packing unit: 1000 - 2000 pcs./reel

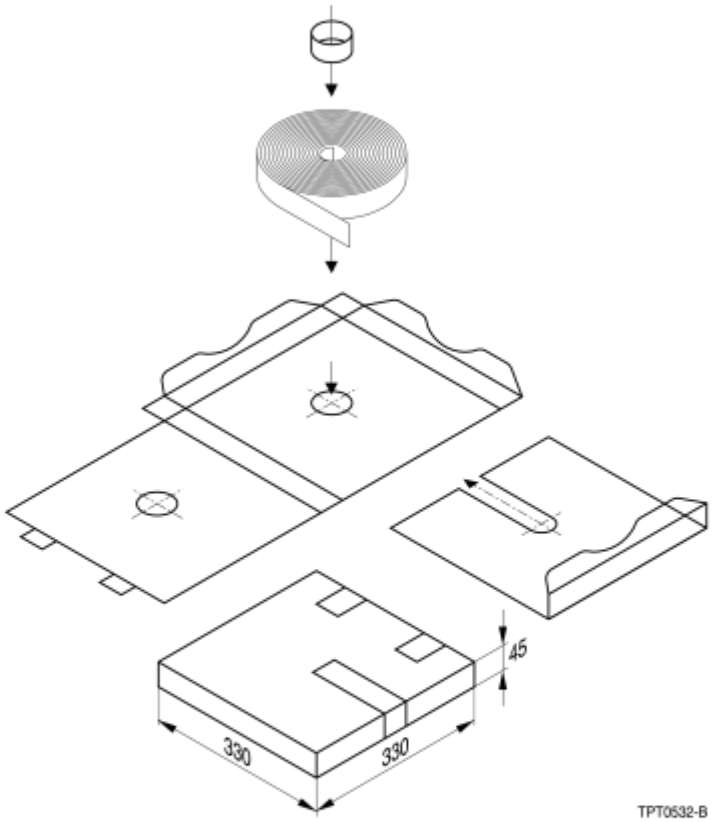
Reel dimensions (in mm)

Reel type	d	f	n	w
I	360 max.	31 ±1	approx. 45	54 max.

TDK

Temperature measurement and compensation	B57164K
Leaded NTC thermistors, lead spacing 5 mm	**K164**

Cassette packing



Packing unit: 1000 - 2000 pcs./cassette

Bulk packing
The components are packed in cardboard boxes, the size of which depends on the order quantity.

TDK

Temperature measurement and compensation	B57164K
Leaded NTC thermistors, lead spacing 5 mm	**K164**

5 Packing codes

The last two digits of the complete ordering code state the packing mode:

Last two digits			
00, 01, 02, 03,04, 05, 06, 07, 08	–	Bulk	–
40, 41	–	Bulk	–
45	–	Bulk	–
50	Radial leads, kinked	Cardboard tape	Cassette packing
51	Radial leads, kinked	Cardboard tape	360-mm reel packing
52	Radial leads, straight	Cardboard tape	Cassette packing
53	Radial leads, straight	Cardboard tape	360-mm reel packing
54	Radial leads, kinked	Cardboard tape	AMMO packing
55	Radial leads, straight	Cardboard tape	AMMO packing

(If no packing code is indicated, this corresponds to 40)

Example 1:	B57164K0102J000	Bulk
	B57164K0102J052	Cardboard tape, cassette packing
Example 2:	B57881S0103F002	Bulk
	B57881S0103F251	Cardboard tape, reel packing

TDK

Temperature measurement and compensation	B57164K
Leaded NTC thermistors, lead spacing 5 mm	K164

Mounting instructions

1 Soldering

1.1 Leaded NTC thermistors

Leaded thermistors comply with the solderability requirements specified by CECC.

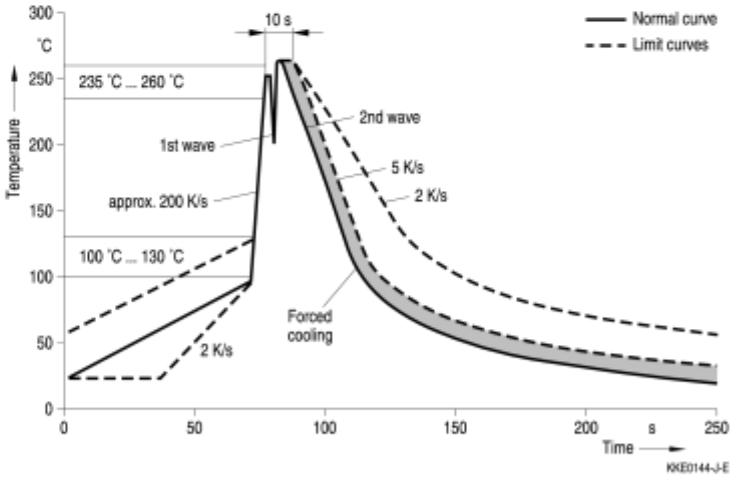
When soldering, care must be taken that the NTC thermistors are not damaged by excessive heat. The following maximum temperatures, maximum time spans and minimum distances have to be observed:

	Dip soldering	*Iron soldering*
Bath temperature	max. 260 °C	max. 360 °C
Soldering time	max. 4 s	max. 2 s
Distance from thermistor	min. 6 mm	min. 6 mm

Under more severe soldering conditions the resistance may change.

1.1.1 Wave soldering

Temperature characteristic at component terminal with dual wave soldering



1.2 Leadless NTC thermistors

In case of NTC thermistors without leads, soldering is restricted to devices which are provided with a solderable metallization. The temperature shock caused by the application of hot solder may produce fine cracks in the ceramic, resulting in changes in resistance.

To prevent leaching of the metallization, solder with silver additives or with a low tin content

TDK

should be used. In addition, soldering methods should be employed which permit short soldering times.

1.3 SMD NTC thermistors

SMD NTC thermistors can be provided with a nickel barrier termination or on special request with silver-palladium termination. The use of no-clean solder products is recommended. In any case mild, non-activated fluxes should be used. Flux residues after soldering should be minimized.

- SMD NTCs with AgPd termination are not approved for lead-free soldering.
- Nickel barrier termination

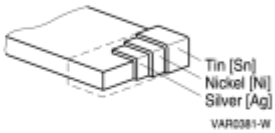


Figure 1
SMD NTC thermistors, structure of nickel barrier termination

The nickel barrier layer of the silver/nickel/tin termination (see figure 1) prevents leaching of the silver base metallization layer. This allows great flexibility in the selection of soldering parameters.

The tin prevents the nickel layer from oxidizing and thus ensures better wetting by the solder. The nickel barrier termination is tested for all commonly-used soldering methods according to IEC 60068-2-58. Insufficient preheating may cause ceramic cracks. Rapid cooling by dipping in solvent is not recommended.

The following test and process conditions apply for nickel barrier termination.

TDK

Temperature measurement and compensation	B57164K
Leaded NTC thermistors, lead spacing 5 mm	**K164**

1.3.1 Solderability (test to IEC 60068-2-58)

Preconditioning: Immersion into flux F-SW 32.
Evaluation criterion: Wetting of soldering areas ≥95%.

Solder	Bath temperature (°C)	Dwell time (s)
SnPb 60/40	215 ±3	3 ±0.3
SnAg (3.0 ... 4.0), Cu (0.5 ... 0.9)	245 ±3	3 ±0.3

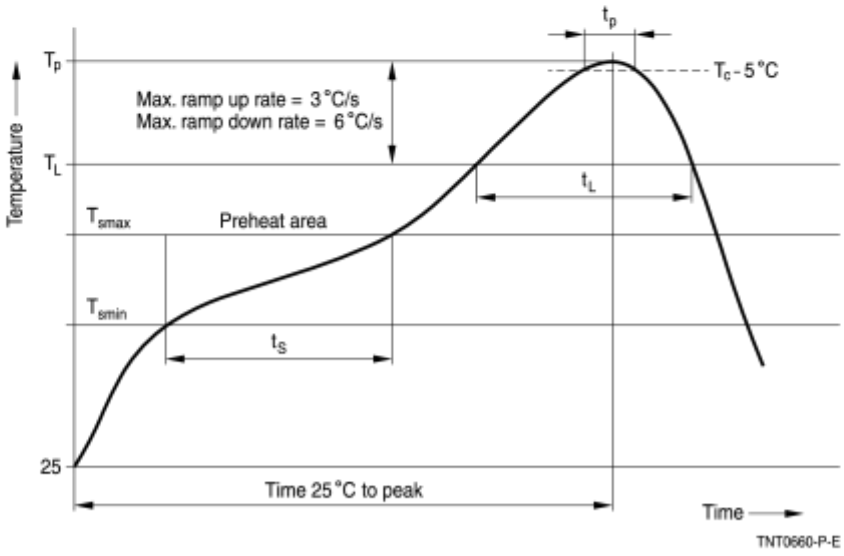
1.3.2 Resistance to soldering heat (test to IEC 60068-2-58)

Preconditioning: Immersion into flux F-SW 32.
Evaluation criterion: Leaching of side edges ≤1/3.

Solder	Bath temperature (°C)	Dwell time (s)
SnPb 60/40	260 ±5	10 ±1
SnAg (3.0 ... 4.0), Cu (0.5 ... 0.9)	260 ±5	10 ±1

1.3.3 Reflow soldering

Temperature ranges for reflow soldering acc. to IEC 60068-2-58 recommendations.



TDK

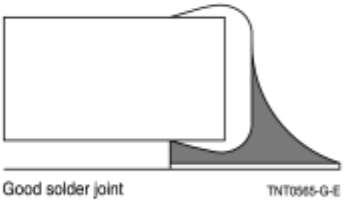
Temperature measurement and compensation	B57164K
Leaded NTC thermistors, lead spacing 5 mm	**K164**

Profile feature		Sn-Pb eutectic assembly	Pb-free assembly
Preheat and soak			
- Temperature min	T_{smin}	100 °C	150 °C
- Temperature max	T_{smax}	150 °C	200 °C
- Time	t_{smin} to t_{smax}	60 ... 120 s	60 ... 120 s
Average ramp-up rate	T_{smax} to T_p	3 °C/ s max.	3 °C/ s max.
Liquidous temperature	T_L	183 °C	217 °C
Time at liquidous	t_L	40 ... 150 s	40 ... 150 s
Peak package body temperature	T_p	215 °C ... 260 °C[1)]	235 °C ... 260 °C
Time above (T_P −5 °C)	t_p	10 ... 40 s	10 ... 40 s
Average ramp-down rate	T_p to T_{smax}	6 °C/ s max.	6 °C/ s max.
Time 25 °C to peak temperature		max. 8 minutes	max. 8 minutes

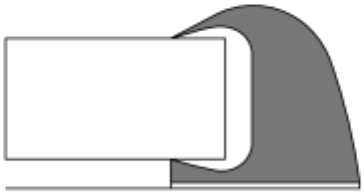
1) Depending on package thickness.

Notes: All temperatures refer to topside of the package, measured on the package body surface.
Number of reflow cycles: 3
Iron soldering should be avoided, hot air methods are recommended for repair purposes.

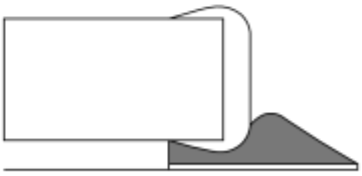
Solder joint profiles for silver/nickel/tin terminations



Good solder joint TNT0565-G-E



Too much solder
Pad geometry too large KKE0071-A-E

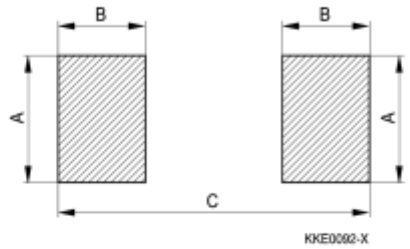


Poor wetting KKE0072-I-E

TDK

Temperature measurement and compensation	B57164K
Leaded NTC thermistors, lead spacing 5 mm	K164

1.3.4 Recommended geometry of solder pads



Recommended maximum dimensions (mm)

Case size inch/mm	A	B	C
0402/1005	0.6	0.6	1.7
0603/1608	1.0	1.0	3.0
0805/2012	1.3	1.2	3.4
1206/3216	1.8	1.2	4.5

2 Conductive adhesion

An alternative to soldering for silver-palladium terminated components is the gluing of thermistors with conductive adhesives. The benefit of this method is that it involves no thermal stress. The adhesives used must be chemically inert.

3 Clamp contacting

Pressure contacting by means of clamps is particularly suitable for applications involving frequent switching and high turn-on powers.

4 Robustness of terminations (leaded types)

The leads meet the requirements of IEC 60068-2-21. They may not be bent closer than 4 mm from the solder joint on the thermistor body or from the point at which they leave the feed-throughs. During bending, any mechanical stress at the outlet of the leads must be removed. The bending radius should be at least 0.75 mm.

TDK

Temperature measurement and compensation	B57164K
Leaded NTC thermistors, lead spacing 5 mm	**K164**

Tensile strength: Test Ua1:

Value of applied force for Ua1 test:

Diameter (d) of corresponding round leads	Force with tolerance of ±10%
∅ ≤ 0.25 mm	1.0 N
0.25 < ∅ ≤ 0.35 mm	2.5 N
0.35 < ∅ ≤ 0.50 mm	5.0 N
0.50 < ∅ ≤ 0.80 mm	10.0 N

Bending strength: Test Ub:

Two 90°-bends in opposite directions

Value of applied force for Ub test:

Diameter (d) of corresponding round leads	Force with tolerance of ±10%
∅ ≤ 0.25 mm	0.5 N
0.25 < ∅ ≤ 0.35 mm	1.25 N
0.35 < ∅ ≤ 0.50 mm	2.5 N
0.50 < ∅ ≤ 0.80 mm	5 N

Torsional strength: Test Uc: severity 2

The lead is bent by 90° at a distance of 6 to 6.5 mm from the thermistor body.

The bending radius of the leads should be approx. 0.75 mm. Two torsions of 180° each (severity 2).

When subjecting leads to mechanical stress, the following should be observed:

Tensile stress on leads

During mounting and operation tensile forces on the leads are to be avoided.

Bending of leads

Bending of the leads directly on the thermistor body is not permissible.

A lead may be bent at a minimum distance of twice the wire's diameter +4 mm from the solder joint on the thermistor body. During bending the wire must be mechanically relieved at its outlet. The bending radius should be at least 0.75 mm.

TDK

5 Sealing and potting

Sealing or potting processes can affect the reliability of the component.

When thermistors are sealed, potted or overmolded, there must be no mechanical stress caused by thermal expansion during the production process (curing / overmolding process) and during later operation. The upper category temperature of the thermistor must not be exceeded. Ensure that the materials used (sealing / potting compound and plastic material) are chemically neutral. As thermistors are temperature sensitive components it should be considered that molding can affect the thermal surrounding and may influence e.g. the response time.

Extensive testing is encouraged in order to determine whether overmolding or potting influences the functionality and/ or reliability of the component.

6 Cleaning

Cleaning processes can affect the reliability of the component.

If cleaning is necessary, mild cleaning agents are recommended. Cleaning agents based on water are not allowed. Washing processes may damage the product due to the possible static or cyclic mechanical loads (e.g. ultrasonic cleaning). They may cause cracks which might lead to reduced reliability and/ or lifetime.

7 Storage

In order to maintain their solderability, thermistors must be stored in a non-corrosive atmosphere. Humidity, temperature and container materials are critical factors.

Do not store SMDs where they are exposed to heat or direct sunlight. Otherwise, the packing material may be deformed or SMDs may stick together, causing problems during mounting. After opening the factory seals, such as polyvinyl-sealed packages, use the SMDs as soon as possible.

The components should be left in the original packing. Touching the metallization of unsoldered thermistors may change their soldering properties.

Storage temperature:	−25 °C up to 45 °C
Relative humidity (without condensation):	≤75% annual mean
	<95%, maximum 30 days per annum

Solder the thermistors listed in this data book after shipment from EPCOS within the time specified:

SMDs with AgPd termination:	6 months
SMDs with nickel barrier termination:	12 months
Leadless components:	12 months
Leaded components:	24 months

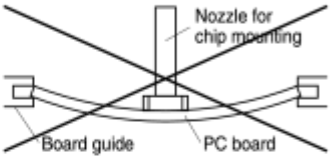
TDK

Temperature measurement and compensation	B57164K
Leaded NTC thermistors, lead spacing 5 mm	K164

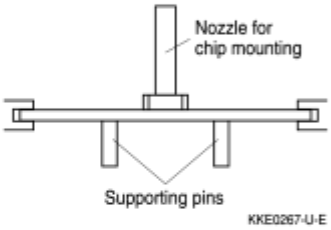
8 Placement and orientation of SMD NTC thermistors on PCB

a) Component placement

Incorrect

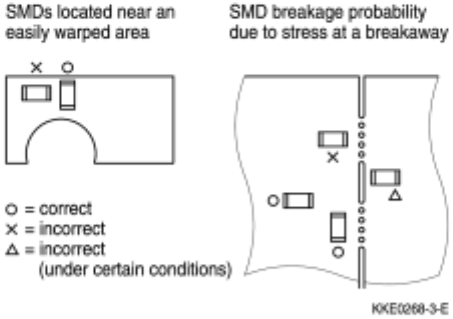


Correct



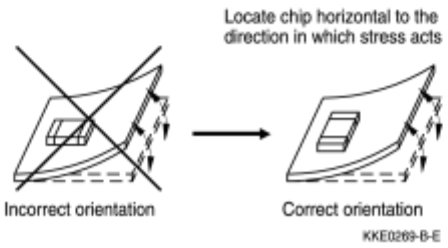
It is recommended that the PC board should be held by means of some adequate supporting pins such as shown left to prevent the SMDs from being damaged or cracked.

b) Cracks



When placing a component near an area which is apt to bend or a grid groove on the PC board, it is advisable to have both electrodes subjected to uniform stress, or to position the component's electrodes at right angles to the grid groove or bending line (see c) Component orientation).

c) Component orientation



Choose a mounting position that minimizes the stress imposed on the chip during flexing or bending of the board.

TDK

Temperature measurement and compensation	B57164K
Leaded NTC thermistors, lead spacing 5 mm	**K164**

Cautions and warnings

General

See "Important notes" on page 2.

Storage

- Store thermistors only in original packaging. Do not open the package prior to processing.
- Storage conditions in original packaging: storage temperature −25 °C ... +45 °C, relative humidity ≤75% annual mean, <95% maximum 30 days per annum, dew precipitation is inadmissible.
- Do not store thermistors where they are exposed to heat or direct sunlight. Otherwise, the packing material may be deformed or components may stick together, causing problems during mounting.
- Avoid contamination of thermistor surface during storage, handling and processing.
- Avoid storage of thermistors in harmful environments like corrosive gases (SO_x, Cl etc).
- Use the components as soon as possible after opening the original packaging.
- Solder thermistors within the time specified after shipment from EPCOS.
  For leaded components this is 24 months, for SMD components with nickel barrier termination 12 months, for leadless components this is 12 months, for SMD components with AgPd termination 6 months.

Handling

- NTC thermistors must not be dropped. Chip-offs or any other damage must not be caused during handling of NTCs.
- Do not touch components with bare hands. Gloves are recommended.
- Avoid contamination of thermistor surface during handling.
- Washing processes may damage the product due to the possible static or cyclic mechanical loads (e.g. ultrasonic cleaning). They may cause cracks to develop on the product and its parts, which might lead to reduced reliability or lifetime.

Bending / twisting leads

- A lead (wire) may be bent at a minimum distance of twice the wire's diameter plus 4 mm from the component head or housing. When bending ensure the wire is mechanically relieved at the component head or housing. The bending radius should be at least 0.75 mm.

Soldering

- Use resin-type flux or non-activated flux.
- Insufficient preheating may cause ceramic cracks.
- Rapid cooling by dipping in solvent is not recommended.
- Complete removal of flux is recommended.

TDK

Temperature measurement and compensation	B57164K
Leaded NTC thermistors, lead spacing 5 mm	**K164**

Mounting

- Ensure that no thermo-mechanical stress occurs due to production processes (curing or overmolding processes) when thermistors are sealed, potted or overmolded or during their subsequent operation. The maximum temperature of the thermistor must not be exceeded. Ensure that the materials used (sealing/potting compound and plastic material) are chemically neutral.
- Electrodes/contacts must not be scratched or damaged before/during/after the mounting process.
- Contacts and housing used for assembly with the thermistor must be clean before mounting.
- Ensure that adjacent materials are designed for operation at temperatures comparable to the surface temperature of the thermistor. Be sure that surrounding parts and materials can withstand the temperature.
- Avoid contamination of the thermistor surface during processing.
- The connections of sensors (e.g. cable end, wire end, plug terminal) may only be exposed to an environment with normal atmospheric conditions.
- Tensile forces on cables or leads must be avoided during mounting and operation.
- Bending or twisting of cables or leads directly on the thermistor body is not permissible.
- Avoid using chemical substances as mounting aids. It must be ensured that no water or other liquids enter the NTC thermistors (e.g. through plug terminals). In particular, water based substances (e.g. soap suds) must not be used as mounting aids for sensors.
- The use of no-clean solder products is recommended. In any case mild, non-activated fluxes should be used. Flux residues after soldering should be minimized.

Operation

- Use thermistors only within the specified operating temperature range.
- Use thermistors only within the specified power range.
- Environmental conditions must not harm the thermistors. Only use the thermistors under normal atmospheric conditions or within the specified conditions.
- Contact of NTC thermistors with any liquids and solvents shall be prevented. It must be ensured that no water enters the NTC thermistors (e.g. through plug terminals). For measurement purposes (checking the specified resistance vs. temperature), the component must not be immersed in water but in suitable liquids (e.g. perfluoropolyethers such as Galden).
- Avoid dewing and condensation unless thermistor is specified for these conditions.
- Bending or twisting of cables and/or wires is not permissible during operation of the sensor in the application.
- Be sure to provide an appropriate fail-safe function to prevent secondary product damage caused by malfunction.

This listing does not claim to be complete, but merely reflects the experience of EPCOS AG.

Display of ordering codes for EPCOS products

The ordering code for one and the same EPCOS product can be represented differently in data

TDK

Temperature measurement and compensation	B57164K
Leaded NTC thermistors, lead spacing 5 mm	**K164**

sheets, data books, other publications, on the EPCOS website, or in order-related documents such as shipping notes, order confirmations and product labels. **The varying representations of the ordering codes are due to different processes employed and do not affect the specifications of the respective products**. Detailed information can be found on the Internet under www.epcos.com/orderingcodes

TDK

Temperature measurement and compensation	B57164K
Leaded NTC thermistors, lead spacing 5 mm	K164

Symbols and terms

Symbol	English	German
A	Area	Fläche
AWG	American Wire Gauge	Amerikanische Norm für Drahtquerschnitte
B	B value	B-Wert
$B_{25/100}$	B value determined by resistance measurement at 25 °C and 100 °C	B-Wert, ermittelt durch Widerstands-messungen bei 25 °C und 100 °C
C_{th}	Heat capacitance	Wärmekapazität
I	Current	Strom
N	Number (integer)	Anzahl (ganzzahliger Wert)
P_{25}	Maximum power at 25 °C	Maximale Leistung bei 25 °C
P_{diss}	Power dissipation	Verlustleistung
P_{el}	Electrical power	Elektrische Leistung
P_{max}	Maximum power within stated temperature range	Maximale Leistung im angegebenenTemperaturbereich
$\Delta R_B/R_B$	Resistance tolerance caused by spread of B value	Widerstandstoleranz, die durch die Streuung des B-Wertes verursacht wird
R_{ins}	Insulation resistance	Isolationswiderstand
R_P	Parallel resistance	Parallelwiderstand
R_R	Rated resistance	Nennwiderstand
$\Delta R_R/R_R$	Resistance tolerance	Widerstandstoleranz
R_S	Series resistance	Serienwiderstand
R_T	Resistance at temperature T (e.g. R_{25} = resistance at 25 °C)	Widerstand bei Temperatur T (z.B. R_{25} = Widerstand bei 25 °C)
T	Temperature	Temperatur
ΔT	Temperature tolerance	Temperaturtoleranz
t	Time	Zeit
T_A	Ambient temperature	Umgebungstemperatur
T_{max}	Upper category temperature	Obere Grenztemperatur (Kategorietemperatur)
T_{min}	Lower category temperature	Untere Grenztemperatur (Kategorietemperatur)
T_{op}	Operating temperature	Betriebstemperatur
T_R	Rated temperature	Nenntemperatur
T_{surf}	Surface temperature	Oberflächentemperatur
V	Voltage	Spannung
V_{ins}	Insulation test voltage	Isolationsprüfspannung
V_{op}	Operating voltage	Betriebsspannung
V_{test}	Test voltage	Prüfspannung

TDK

Temperature measurement and compensation	B57164K
Leaded NTC thermistors, lead spacing 5 mm	**K164**

Symbol	English	German
α	Temperature coefficient	Temperaturkoeffizient
Δ	Tolerance, change	Toleranz, Änderung
δ_{th}	Dissipation factor	Wärmeleitwert
τ_c	Thermal cooling time constant	Thermische Abkühlzeitkonstante
τ_a	Thermal time constant	Thermische Zeitkonstante

Abbreviations / Notes

Symbol	English	German
SMD	Surface-mounted devices	Oberflächenmontierbares Bauelement
*	To be replaced by a number in ordering codes, type designations etc.	Platzhalter für Zahl im Bestellnummern-code oder für die Typenbezeichnung.
+	To be replaced by a letter.	Platzhalter für einen Buchstaben.
	All dimensions are given in mm.	Alle Maße sind in mm angegeben.
	The commas used in numerical values denote decimal points.	Verwendete Kommas in Zahlenwerten bezeichnen Dezimalpunkte.

Please read *Cautions and warnings* and *Important notes* at the end of this document.

Page 29 of 31

TDK

Important notes

The following applies to all products named in this publication:

1. Some parts of this publication contain **statements about the suitability of our products for certain areas of application**. These statements are based on our knowledge of typical requirements that are often placed on our products in the areas of application concerned. We nevertheless expressly point out **that such statements cannot be regarded as binding statements about the suitability of our products for a particular customer application**. As a rule, EPCOS is either unfamiliar with individual customer applications or less familiar with them than the customers themselves. For these reasons, it is always ultimately incumbent on the customer to check and decide whether an EPCOS product with the properties described in the product specification is suitable for use in a particular customer application.
2. We also point out that **in individual cases, a malfunction of electronic components or failure before the end of their usual service life cannot be completely ruled out in the current state of the art, even if they are operated as specified**. In customer applications requiring a very high level of operational safety and especially in customer applications in which the malfunction or failure of an electronic component could endanger human life or health (e.g. in accident prevention or lifesaving systems), it must therefore be ensured by means of suitable design of the customer application or other action taken by the customer (e.g. installation of protective circuitry or redundancy) that no injury or damage is sustained by third parties in the event of malfunction or failure of an electronic component.
3. **The warnings, cautions and product-specific notes must be observed.**
4. In order to satisfy certain technical requirements, **some of the products described in this publication may contain substances subject to restrictions in certain jurisdictions (e.g. because they are classed as hazardous)**. Useful information on this will be found in our Material Data Sheets on the Internet (www.epcos.com/material). Should you have any more detailed questions, please contact our sales offices.
5. We constantly strive to improve our products. Consequently, **the products described in this publication may change from time to time**. The same is true of the corresponding product specifications. Please check therefore to what extent product descriptions and specifications contained in this publication are still applicable before or when you place an order. We also **reserve the right to discontinue production and delivery of products**. Consequently, we cannot guarantee that all products named in this publication will always be available. The aforementioned does not apply in the case of individual agreements deviating from the foregoing for customer-specific products.
6. Unless otherwise agreed in individual contracts, **all orders are subject to the current version of the "General Terms of Delivery for Products and Services in the Electrical Industry" published by the German Electrical and Electronics Industry Association (ZVEI)**.

TDK

Important notes

7. The trade names EPCOS, CeraCharge, CeraDiode, CeraLink, CeraPad, CeraPlas, CSMP, CTVS, DeltaCap, DigiSiMic, ExoCore, FilterCap, FormFit, LeaXield, MiniBlue, MiniCell, MKD, MKK, MotorCap, PCC, PhaseCap, PhaseCube, PhaseMod, PhiCap, PowerHap, PQSine, PQvar, SIFERRIT, SIFI, SIKOREL, SilverCap, SIMDAD, SiMic, SIMID, SineFormer, SIOV, ThermoFuse, WindCap are **trademarks registered or pending** in Europe and in other countries. Further information will be found on the Internet at www.epcos.com/trademarks.

Page 31 of 31

3.5. SENSOR DE CORRIENTE HTFS-440/P

LEM

Current Transducer HTFS 200 .. 800-P/SP2

I_{PN} = 200 .. 800A

For the electronic measurement of currents: DC, AC, pulsed..., with galvanic separation between the primary circuit and the secondary circuit.

All data are given with R_L = 10 kΩ

Electrical data

Primary nominal rms current I_{PN} (A)	Primary current measuring range I_P (A)	Type	RoHS since Date code
200	±300	**HTFS 200-P/SP2**	45326
400	±600	**HTFS 400-P/SP2**	45060
800	±1200	**HTFS 800-P/SP2**	45060

V_{out}	Output voltage (Analog) @ I_P		V_{ref} ±(1.25 · I_P/I_{PN})	V
	I_P = 0		V_{ref} ±0.025	V
V_{ref}	Reference voltage	[1] - Output voltage	1/2 U_C ±0.025	V
		V_{ref} Output impedance	Typ. 200	Ω
		V_{ref} Load impedance	≥200	kΩ
R_L	Load resistance		≥2	kΩ
R_{out}	Output internal resistance		<5	Ω
C_L	Capacitive loading		4.7	nF
U_C	Supply voltage (±5 %)		5	V
I_C	Current consumption @ U_C = 5 V		19 (typ)	mA
			25 (max)	mA

Accuracy - Dynamic performance data

X	Accuracy [2] @ I_{PN}, T_A = 25 °C	≤±1	%
ε_L	Linearity error 0 .. 1.5 × I_{PN}	≤±0.5	%
TCV_{OE}	Temperature of coefficient of V_{OE} @ I_P = 0,	≤±0.1	mV/K
TCV_{ref}	Temperature of coefficient of V_{ref}	≤±190	ppm/K
TCG	Temperature of coefficient of V_{out}	≤±420	ppm/K
V_{OM}	Magnetic offset voltage @ I_P = 0 and specified R_M, after an overload of 3 × $I_{PN\,DC}$	<±0.5	%
V_{no}	Output voltage noise (DC .. 20 MHz)	<40	mVpp
t_{ra}	Reaction time to 10 % of I_{PN}	<2	µs
t_r	Step response time to 90 % of I_{PN}	<3.5	µs
di/dt	di/dt accurately followed	>100	A/µs
BW	Frequency bandwidth (-3 dB) [3]	DC .. 240	kHz

Notes: [1] It is possible to overdrive V_{ref} with an external reference voltage between 0.5 - 2.65 V
[2] Excluding offset and magnetic offset voltage
[3] Small signal only to avoid excessive heatings of the magnetic core.

Features

- Hall effect measuring principle
- Galvanic separation between primary and secondary circuit
- Low power consumption
- Single power supply +5 V
- Ratiometric offset
- Insulating plastic case recognized according to UL 94-V0
- T_A = -40 °C .. +105 °C.

Special feature

- PCB fixation with 4 pins ø 1 mm.

Advantages

- Small size and space saving
- Only one design for wide current ratings range
- High immunity to external interference
- V_{ref} IN/OUT.

Applications

- Forklift drives
- AC variable speed drives
- Static converters for DC motor drives
- Battery supplied applications
- Uninterruptible Power Supplies (UPS)
- Switched Mode Power Supplies (SMPS)
- Power supplies for welding applications.

Application domain

- Industrial.

74.96.44.002.0, 74.96.48.002.0, 74.96.56.002.0 Page 1/4
4November2014/version 8 LEM reserves the right to carry out modifications on its transducers, in order to improve them, without prior notice www.lem.com

LEM

Current Transducer HTFS 200 .. 800-P/SP2

General data

T_A	Ambient operating temperature	-40 .. +105	°C
T_S	Ambient storage temperature	-40 .. +105	°C
m	Mass	60	g
	Standards	EN 50178: 1997	

Isolation characteristics

U_d	Rms voltage for AC insulation test, 50 Hz, 1 min	2.5	kV
$\hat{U}_W$	Impulse withstand voltage 1.2/50 µs	4	kV
U_e	Partial discharge extinction rms voltage @ 10 pC	>1	kV
		Min	
d_{Cp}	Creepage distance	>4	mm
d_{Cl}	Clearance	>4	mm
CTI	Comparative Tracking Index (group IIIa)	>220	

Applications examples

	EN 50178	IEC 61010-1
d_{Cp}, d_{Cl}, $\hat{U}_W$	Rated insulation voltage	Nominal voltage
Basic insulation	300 V	300 V
Reinforced insulation	150 V	150 V

According to EN 50178 and IEC 61010-1 standards and following conditions:

- Over voltage category OV 3
- Pollution degree PD2
- Non-uniform field

Safety

This transducer must be used in limited-energy secondary circuits according to IEC 61010-1.

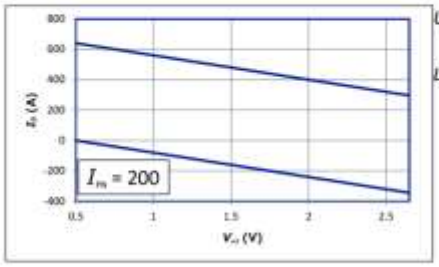
This transducer must be used in electric/electronic equipment with respect to applicable standards and safety requirements in accordance with the manufacturer's operating instructions.

Caution, risk of electrical shock

When operating the transducer, certain parts of the module can carry hazardous voltage (eg. primary busbar, power supply). Ignoring this warning can lead to injury and/or cause serious damage.
This transducer is a build-in device, whose conducting parts must be inaccessible after installation.A protective housing or additional shield could be used.
Main supply must be able to be disconnected.

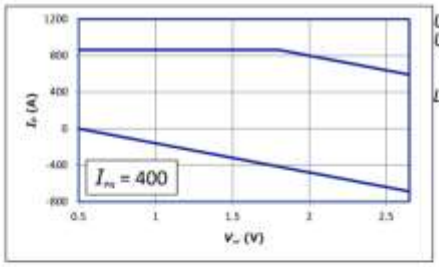
LEM

HTFS measuring range with external V_{ref}



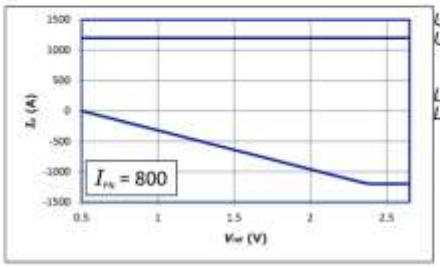
Upper limit: $I_P = -160 \times V_{ref} + 720$ (V_{ref} = 0.5 .. 2.65 V)

Lower limit: $I_P = -160 \times V_{ref} + 80$ (V_{ref} = 0.5 .. 2.65 V)



Upper limit: $I_P = 864$ (V_{ref} = 0.5 .. 1.8 V)
Upper limit: $I_P = -320 \times V_{ref} + 1440$ (V_{ref} = 1.8 .. 2.65 V)

Lower limit: $I_P = -320 \times V_{ref} + 160$ (V_{ref} = 0.5 .. 2.65 V)

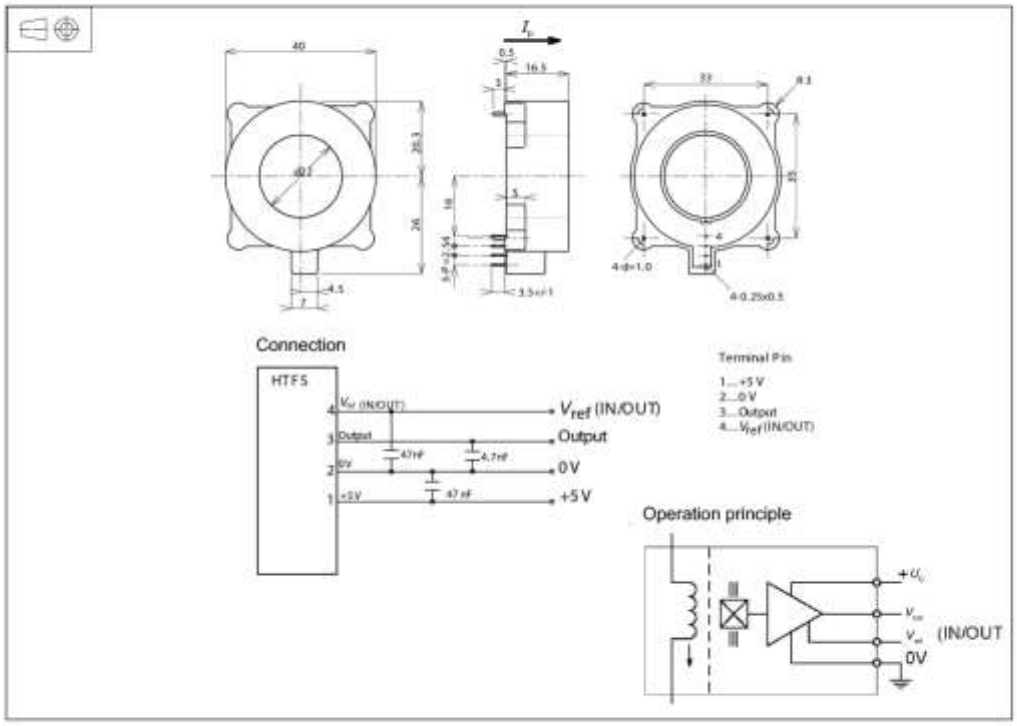


Upper limit: $I_P = 1200$ (V_{ref} = 0.5 .. 2.0 V)
Upper limit: $I_P = -640 \times V_{ref} + 2880$ (V_{ref} = 2.625 .. 2.65 V)

Lower limit: $I_P = -640 \times V_{ref} + 320$ (V_{ref} = 0.5 .. 2.4 V)
Lower limit: $I_P = -1200$ (V_{ref} = 2.4 .. 2.65 V)

LEM

Dimensions HTFS 200 .. 800-P/SP2 (in mm)



Mechanical characteristics

• General tolerance	±0.2 mm
• Fixation to PCB	4 pins × ø 1 mm
Recommended PCB hole	ø 1.2 mm
• Connection to secondary	4 pins 0.5 × 0.25 mm
Recommended PCB hole	ø 0.7 mm

Remarks

- V_{out} is positive when I_p flows in the direction of the arrow.
- **Temperature of the primary conductor should not exceed 120 °C.**

3.6. BUFFER DE CAPACITANCIA P82B96

P82B96

Dual bidirectional bus buffer

Rev. 08 — 10 November 2009 Product data sheet

1. General description

The P82B96 is a bipolar IC that creates a non-latching, bidirectional, logic interface between the normal I²C-bus and a range of other bus configurations. It can interface I²C-bus logic signals to similar buses having different voltage and current levels.

For example, it can interface to the 350 µA SMBus, to 3.3 V logic devices, and to 15 V levels and/or low-impedance lines to improve noise immunity on longer bus lengths.

It achieves this interface without any restrictions on the normal I²C-bus protocols or clock speed. The IC adds minimal loading to the I²C-bus node, and loadings of the new bus or remote I²C-bus nodes are not transmitted or transformed to the local node. Restrictions on the number of I²C-bus devices in a system, or the physical separation between them, are virtually eliminated. Transmitting SDA and SCL signals via balanced transmission lines (twisted pairs) or with galvanic isolation (opto-coupling) is simple because separate directional Tx and Rx signals are provided. The Tx and Rx signals may be directly connected, without causing latching, to provide an alternative bidirectional signal line with I²C-bus properties.

2. Features

- Bidirectional data transfer of I²C-bus signals
- Isolates capacitance allowing 400 pF on Sx/Sy side and 4000 pF on Tx/Ty side
- Tx/Ty outputs have 60 mA sink capability for driving low-impedance or high capacitive buses
- 400 kHz operation over at least 20 meters of wire (see *AN10148*)
- Supply voltage range of 2 V to 15 V with I²C-bus logic levels on Sx/Sy side independent of supply voltage
- Splits I²C-bus signal into pairs of forward/reverse Tx/Rx, Ty/Ry signals for interface with opto-electrical isolators and similar devices that need unidirectional input and output signal paths.
- Low power supply current
- ESD protection exceeds 3500 V HBM per JESD22-A114, 250 V DIP package, 400 V SO package MM per JESD22-A115, and 1000 V CDM per JESD22-C101
- Latch-up free (bipolar process with no latching structures)
- Packages offered: DIP8, SO8 and TSSOP8

3. Applications

- Interface between I²C-buses operating at different logic levels (for example, 5 V and 3 V or 15 V)
- Interface between I²C-bus and SMBus (350 µA) standard
- Simple conversion of I²C-bus SDA or SCL signals to multi-drop differential bus hardware, for example, via compatible PCA82C250
- Interfaces with opto-couplers to provide opto-isolation between I²C-bus nodes up to 400 kHz

4. Ordering information

Table 1. Ordering information

Type number	Package		
	Name	Description	Version
P82B96DP	TSSOP8	plastic thin shrink small outline package; 8 leads; body width 3 mm	SOT505-1
P82B96PN	DIP8	plastic dual in-line package; 8 leads (300 mil)	SOT97-1
P82B96TD	SO8	plastic small outline package; 8 leads; body width 3.9 mm	SOT96-1
P82B96TD/S900	SO8	plastic small outline package; 8 leads; body width 3.9 mm	SOT96-1

4.1 Ordering options

Table 2. Ordering options

Type number	Topside mark	Temperature range
P82B96DP	82B96	−40 °C to +85 °C
P82B96PN	P82B96PN	−40 °C to +85 °C
P82B96TD	P82B96T	−40 °C to +85 °C
P82B96TD/S900	P82B96T	−40 °C to +125 °C

5. Block diagram

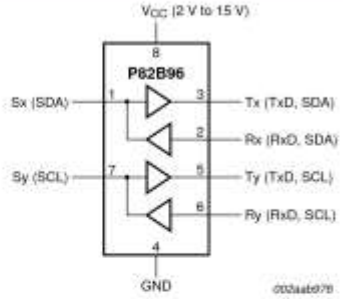


Fig 1. Block diagram of P82B96

6. Pinning information

6.1 Pinning

Fig 2. Pin configuration for DIP8

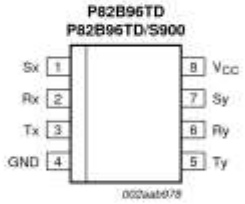


Fig 3. Pin configuration for SO8

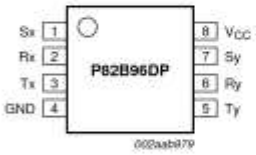


Fig 4. Pin configuration for TSSOP8

6.2 Pin description

Table 3. Pin description

Symbol	Pin	Description
Sx	1	I²C-bus (SDA or SCL)
Rx	2	receive signal
Tx	3	transmit signal
GND	4	negative supply
Ty	5	transmit signal
Ry	6	receive signal
Sy	7	I²C-bus (SDA or SCL)
V_{CC}	8	positive supply voltage

7. Functional description

Refer to Figure 1 "Block diagram of P82B96".

The P82B96 has two identical buffers allowing buffering of both of the I²C-bus (SDA and SCL) signals. Each buffer is made up of two logic signal paths, a forward path from the I²C-bus interface pin which drives the buffered bus, and a reverse signal path from the buffered bus input to drive the I²C-bus interface. Thus these paths are:

- sense the voltage state of the I²C-bus pin Sx (or Sy) and transmit this state to the pin Tx (Ty respectively), and
- sense the state of the pin Rx (Ry) and pull the I²C-bus pin LOW whenever Rx (Ry) is LOW.

The rest of this discussion will address only the 'x' side of the buffer; the 'y' side is identical.

The I²C-bus pin (Sx) is designed to interface with a normal I²C-bus.

The logic threshold voltage levels on the I²C-bus are independent of the IC supply V_{CC}. The maximum I²C-bus supply voltage is 15 V and the guaranteed static sink current is 3 mA.

The logic level of Rx is determined from the power supply voltage V_{CC} of the chip. Logic LOW is below 42 % of V_{CC}, and logic HIGH is above 58 % of V_{CC} (with a typical switching threshold of half V_{CC}).

Tx is an open-collector output without ESD protection diodes to V_{CC}. It may be connected via a pull-up resistor to a supply voltage in excess of V_{CC}, as long as the 15 V rating is not exceeded. It has a larger current sinking capability than a normal I²C-bus device, being able to sink a static current of greater than 30 mA, and typical 100 mA dynamic pull-down capability as well.

A logic LOW is only transmitted to Tx when the voltage at the I²C-bus pin (Sx) is below 0.6 V. A logic LOW at Rx will cause the I²C-bus (Sx) to be pulled to a logic LOW level in accordance with I²C-bus requirements (maximum 1.5 V in 5 V applications) but not low enough to be looped back to the Tx output and cause the buffer to latch LOW.

The minimum LOW level this chip can achieve on the I²C-bus by a LOW at Rx is typically 0.8 V.

If the supply voltage V_{CC} fails, then neither the I²C-bus nor the Tx output will be held LOW. Their open-collector configuration allows them to be pulled up to the rated maximum of 15 V even without V_{CC} present. The input configuration on Sx and Rx also present no loading of external signals even when V_{CC} is not present.

The effective input capacitance of any signal pin, measured by its effect on bus rise times, is less than 7 pF for all bus voltages and supply voltages including V_{CC} = 0 V.

Remark: Two or more Sx or Sy I/Os must not be interconnected. The P82B96 design does not support this configuration. Bidirectional I²C-bus signals do not allow any direction control pin so, instead, slightly different logic low voltage levels are used at Sx/Sy to avoid latching of this buffer. A 'regular I²C-bus LOW' applied at the Rx/Ry of a P82B96 will be propagated to Sx/Sy as a 'buffered LOW' with a slightly higher voltage level. If this

special 'buffered LOW' is applied to the Sx/Sy of another P82B96 that second P82B96 will not recognize it as a 'regular I²C-bus LOW' and will not propagate it to its Tx/Ty output. The Sx/Sy side of P82B96 may not be connected to similar buffers that rely on special logic thresholds for their operation, for example PCA9511, PCA9515, or PCA9518. The Sx/Sy side is only intended for, and compatible with, the normal I²C-bus logic voltage levels of I²C-bus master and slave chips, or even Tx/Rx signals of a second P82B96 if required. The Tx/Rx and Ty/Ry I/O pins use the standard I²C-bus logic voltage levels of all I²C-bus parts. There are **no** restrictions on the interconnection of the Tx/Rx and Ty/Ry I/O pins to other P82B96s, for example in a star or multipoint configuration with the Tx/Rx and Ty/Ry I/O pins on the common bus and the Sx/Sy side connected to the line card slave devices. For more details see *Application Note AN255*.

8. Limiting values

Table 4. Limiting values
In accordance with the Absolute Maximum Rating System (IEC 60134).
Voltages with respect to pin GND.

Symbol	Parameter	Conditions		Min	Max	Unit
V_{CC}	supply voltage	V_{CC} to GND		−0.3	+18	V
V_{Sx}	voltage on pin Sx	I²C-bus SDA or SCL		−0.3	+18	V
V_{Tx}	voltage on pin Tx	buffered output	[1]	−0.3	+18	V
V_{Rx}	voltage on pin Rx	receive input	[1]	−0.3	+18	V
I_n	current on any pin			-	250	mA
P_{tot}	total power dissipation			-	300	mW
T_j	junction temperature	operating range P82B96TD/S900		−40	+125	°C
T_{stg}	storage temperature			−55	+125	°C
T_{amb}	ambient temperature	operating		−40	+85	°C

[1] See also Section 10.2 "Negative undershoot below absolute minimum value".

9. Characteristics

Table 5. Characteristics

T_{amb} = +25 °C; voltages are specified with respect to GND with V_{CC} = 5 V, unless otherwise specified.

Symbol	Parameter	Conditions	T_{amb} = +25 °C			T_{amb} = −40 °C to +125 °C[1]		Unit
			Min	Typ	Max	Min	Max	
Power supply								
V_{CC}	supply voltage	operating	2.0	-	15	2.0	15	V
I_{CC}	supply current	buses HIGH	-	0.9	1.8	-	3	mA
		V_{CC} = 15 V; buses HIGH	-	1.1	2.5	-	4	mA
ΔI_{CC}	additional quiescent supply current	per Tx or Ty LOW	-	1.7	3.5	-	3.5	mA
Bus pull-up (load) voltages and currents								
V_{Sx}, V_{Sy}	maximum input/output voltage	open-collector; I²C-bus and V_{Rx}, V_{Ry} = HIGH	-	-	15	-	15	V
I_{Sx}, I_{Sy}	static output loading on I²C-bus	V_{Sx}, V_{Sy} = 1.0 V; V_{Rx}, V_{Ry} = LOW [2]	0.2	-	3	0.2	3	mA
I_{Sx}, I_{Sy}	dynamic output sink capability on I²C-bus	V_{Sx}, V_{Sy} = 2 V; V_{Rx}, V_{Ry} = LOW	7	18	-	7	-	mA
I_{Sx}, I_{Sy}	leakage current on I²C-bus	V_{Sx}, V_{Sy} = 5 V; V_{Rx}, V_{Ry} = HIGH	-	-	1	-	10	µA
		V_{Sx}, V_{Sy} = 15 V; V_{Rx}, V_{Ry} = HIGH	-	1	-	-	10	µA
V_{Tx}, V_{Ty}	maximum output voltage level	open-collector	-	-	15	-	15	V
I_{Tx}, I_{Ty}	static output loading on buffered bus	V_{Tx}, V_{Ty} = 0.4 V; V_{Sx}, V_{Sy} = LOW on I²C-bus = 0.4 V	-	-	30	-	30	mA
I_{Tx}, I_{Ty}	dynamic output sink capability, buffered bus	V_{Tx}, V_{Ty} > 1 V; V_{Sx}, V_{Sy} = LOW on I²C-bus = 0.4 V	60	100	-	60	-	mA
I_{Tx}, I_{Ty}	leakage current on buffered bus	V_{Tx}, V_{Ty} = V_{CC} = 15 V; V_{Sx}, V_{Sy} = HIGH	-	1	-	-	10	µA
Input currents								
I_{Sx}, I_{Sy}	input current from I²C-bus	bus LOW; V_{Rx}, V_{Ry} = HIGH	-	−1	-	-	−10	µA
I_{Rx}, I_{Ry}	input current from buffered bus	bus LOW; V_{Rx}, V_{Ry} = 0.4 V	-	−1	-	-	−10	µA
I_{Rx}, I_{Ry}	leakage current on buffered bus input	V_{Rx}, V_{Ry} = V_{CC}	-	1	-	-	10	µA
Output logic LOW level								
V_{Sx}, V_{Sy}	output logic level LOW on normal I²C-bus	I_{Sx}, I_{Sy} = 3 mA [3]	0.8	0.88	1.0	(see Figure 6)		V
		I_{Sx}, I_{Sy} = 0.2 mA [3]	670	730	790	(see Figure 5)		mV
dV_{Sx}/dT, dV_{Sy}/dT	temperature coefficient of output LOW levels	I_{Sx}, I_{Sy} = 0.2 mA [3]	-	−1.8	-	-	-	mV/K

Table 5. Characteristics *...continued*

T_{amb} = +25 °C; voltages are specified with respect to GND with V_{CC} = 5 V, unless otherwise specified.

Symbol	Parameter	Conditions		T_{amb} = +25 °C			T_{amb} = −40 °C to +125 °C[1]		Unit
				Min	Typ	Max	Min	Max	
Input logic switching threshold voltages									
V_{Sx}, V_{Sy}	input logic voltage LOW	on normal I²C-bus	[4]	-	640	600	(see Figure 7)		mV
V_{Sx}, V_{Sy}	input logic level HIGH threshold	on normal I²C-bus	[4]	700	650	-	(see Figure 8)		mV
dV_{Sx}/dT, dV_{Sy}/dT	temperature coefficient of input thresholds			-	−2	-	-	-	mV/K
V_{Rx}, V_{Ry}	input logic HIGH level	fraction of applied V_{CC}		$0.58V_{CC}$	-	-	$0.58V_{CC}$	-	V
V_{Rx}, V_{Ry}	input threshold	fraction of applied V_{CC}		-	$0.5V_{CC}$	-	-	-	V
V_{Rx}, V_{Ry}	input logic LOW level	fraction of applied V_{CC}		-	-	$0.42V_{CC}$	-	$0.42V_{CC}$	V
Logic level threshold difference									
V_{Sx}, V_{Sy}	input/output logic level difference	V_{Sx} output LOW at 0.2 mA – V_{Sx} input HIGH maximum	[3]	50	85	-	50	-	mV
Thermal resistance									
$R_{th(j-pcb)}$	thermal resistance from junction to printed-circuit board	SOT96-1 (SO8); average lead temperature at board interface		-	127	-	-	-	K/W
Bus release on V_{CC} failure									
V_{Sx}, V_{Sy}, V_{Tx}, V_{Ty}	V_{CC} voltage at which all buses are guaranteed to be released			-	-	1	(see Figure 9)		V
dV/dT	temperature coefficient of guaranteed release voltage			-	−4	-	-	-	mV/K
Buffer response time[5]									
$T_{fall\ delay}$ V_{Sx} to V_{Tx}, V_{Sy} to V_{Ty}	buffer time delay on **falling** input between V_{Sx} = input switching threshold, and V_{Tx} output falling 50 %	R_{Tx} pull-up = 160 Ω; no capacitive load; V_{CC} = 5 V		-	70	-	-	-	ns
$T_{rise\ delay}$ V_{Sx} to V_{Tx}, V_{Sy} to V_{Ty}	buffer time delay on **rising** input between V_{Sx} = input switching threshold, and V_{Tx} output reaching 50 % V_{CC}	R_{Tx} pull-up = 160 Ω; no capacitive load; V_{CC} = 5 V		-	90	-	-	-	ns

Table 5. Characteristics *...continued*

T_{amb} = +25 °C; voltages are specified with respect to GND with V_{CC} = 5 V, unless otherwise specified.

Symbol	Parameter	Conditions	T_{amb} = +25 °C			T_{amb} = −40 °C to +125 °C[1]		Unit
			Min	Typ	Max	Min	Max	
$T_{fall\ delay}$ V_{Rx} to V_{Sx}, V_{Ry} to V_{Sy}	buffer time delay on **falling** input between V_{Rx} = input switching threshold, and V_{Sx} output falling 50 %	R_{Sx} pull-up = 1500 Ω; no capacitive load; V_{CC} = 5 V	-	250	-	-	-	ns
$T_{rise\ delay}$ V_{Rx} to V_{Sx}, V_{Ry} to V_{Sy}	buffer time delay on **rising** input between V_{Rx} = input switching threshold, and V_{Sx} output reaching 50 % V_{CC}	R_{Sx} pull-up = 1500 Ω; no capacitive load; V_{CC} = 5 V	-	270	-	-	-	ns
Input capacitance								
C_i	input capacitance	effective input capacitance of any signal pin measured by incremental bus rise times	-	-	7	-	7	pF

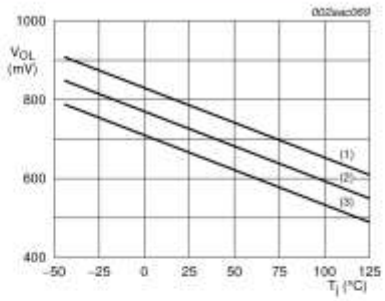
[1] Limit data for +125 °C applies to P82B96TD/S900 version. It is guaranteed by design/characterization, but not by 100 % test.

[2] The minimum value requirement for pull-up current, 200 µA, guarantees that the minimum value for V_{Sx} output LOW will always exceed the minimum V_{Sx} input HIGH level to eliminate any possibility of latching. The specified difference is guaranteed by design within any IC. While the tolerances on absolute levels allow a small probability the LOW from one Sx output is recognized by an Sx input of another P82B96, this has no consequences for normal applications. In any design the Sx pins of different ICs should never be linked because the resulting system would be very susceptible to induced noise and would not support all I²C-bus operating modes.

[3] The output logic LOW depends on the sink current. For scaling, see *Application Note AN255*.

[4] The input logic threshold is independent of the supply voltage.

[5] The fall time of V_{Tx} from 5 V to 2.5 V in the test is approximately 15 ns.
The fall time of V_{Sx} from 5 V to 2.5 V in the test is approximately 50 ns.
The rise time of V_{Tx} from 0 V to 2.5 V in the test is approximately 20 ns.
The rise time of V_{Sx} from 0.9 V to 2.5 V in the test is approximately 70 ns.



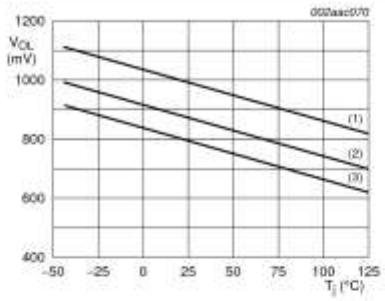
V_{OL} at Sx typical and limits over temperature

(1) Maximum

(2) Typical

(3) Minimum

Fig 5. V_{OL} as a function of junction temperature (I_{OL} = 0.2 mA)



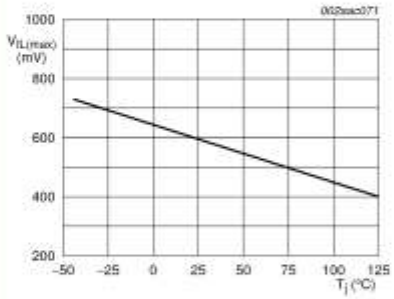
V_{OL} at Sx typical and limits over temperature

(1) Maximum

(2) Typical

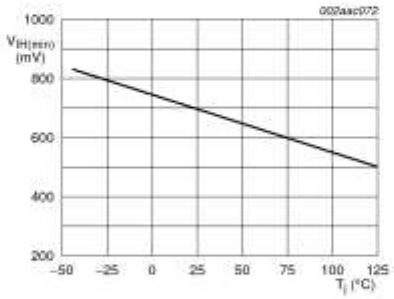
(3) Minimum

Fig 6. V_{OL} as a function of junction temperature (I_{OL} = 3 mA)



$V_{IL(max)}$ at Sx changes over temperature range

Fig 7. $V_{IL(max)}$ as a function of junction temperature



$V_{IH(min)}$ at Sx changes over temperature range

Fig 8. $V_{IH(min)}$ as a function of junction temperature

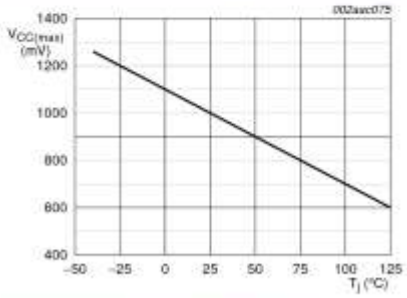


Fig 9. $V_{CC(max)}$ that guarantees bus release limit over temperature

10. Application information

Refer to *AN460* and *AN255* for more application detail.

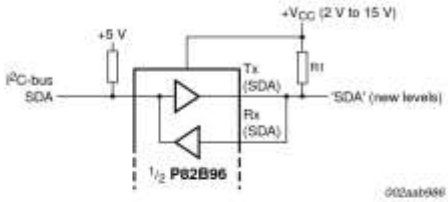


Fig 10. Interfacing an 'I²C' type of bus with different logic levels

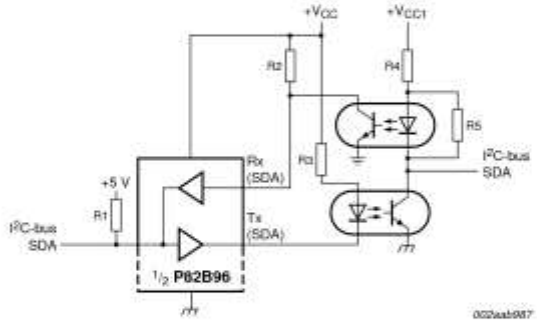


Fig 11. Galvanic isolation of I²C-bus nodes via opto-couplers

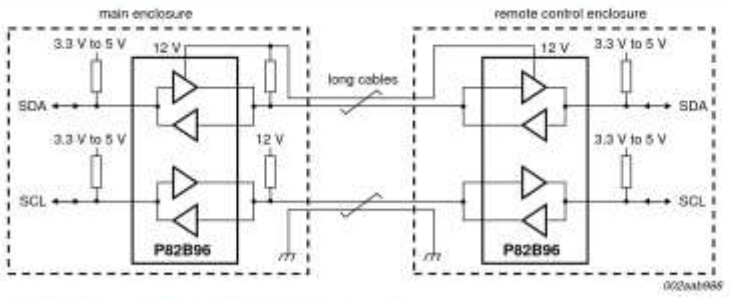


Fig 12. Long distance I²C-bus communications

Figure 13 shows how a master I²C-bus can be protected against short circuits or failures in applications that involve plug and socket connections and long cables that may become damaged. A simple circuit is added to monitor the SDA bus, and if its LOW time exceeds the design value, then the master bus is disconnected. P82B96 will free all its I/Os if its supply is removed, so one option is to connect its V_{CC} to the output of a logic gate from, say, the 74LVC family. The SDA and SCL lines could be timed and V_{CC} disabled via the gate if one or other lines exceeds a design value of 'LOW' period as in *Figure 28 of AN255*. If the supply voltage of logic gates restricts the choice of V_{CC} supply then the low-cost discrete circuit in Figure 13 can be used. If the SDA line is held LOW, the 100 nF capacitor will charge and the Ry input will be pulled towards V_{CC}. When it exceeds $0.5V_{CC}$ the Ry input will set the Sy input HIGH, which in practice means simply releasing it.

In this example the SCL line is made unidirectional by tying the Rx pin to V_{CC}. The state of the buffered SCL line cannot affect the master clock line which is allowed when clock-stretching is not required. It is simple to add an additional transistor or diode to control the Rx input in the same way as Ry when necessary. The +V cable drive can be any voltage up to 15 V and the bus may be run at a lower impedance by selecting pull-up resistors for a static sink current up to 30 mA. V_{CC1} and V_{CC2} may be chosen to suit the connected devices. Because DDC uses relatively low speeds (< 100 kHz), the cable length is not restricted to 20 m by the I²C-bus signalling, but it may be limited by the video signalling.

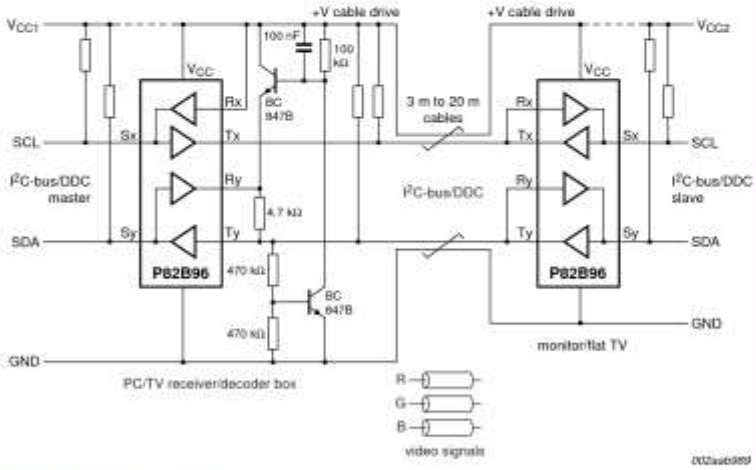


Fig 13. Extending a DDC bus

Figure 14 shows that P82B96 can achieve high clock rates over long cables. While calculating with lumped wiring capacitance yields reasonable approximations to actual timing, even 25 meters of cable is better treated using transmission line theory. Flat ribbon cables connected as shown, with the bus signals on the outer edge, will have a characteristic impedance in the range 100 Ω to 200 Ω. For simplicity they cannot be terminated in their characteristic impedance but a practical compromise is to use the minimum pull-up allowed for P82B96 and place half this termination at each end of the cable. When each pull-up is below 330 Ω, the rising edge waveforms have their first voltage 'step' level above the logic threshold at Rx and cable timing calculations can be based on the fast rise/fall times of resistive loading plus simple one-way propagation delays. When the pull-up is larger, but below 750 Ω, the threshold at Rx will be crossed after one signal reflection. So at the sending end it is crossed after 2 times the one-way propagation delay and at the receiving end after 3 times that propagation delay. For flat cables with partial plastic dielectric insulation (by using outer cores) the one-way propagation delays will be about 5 ns per meter. The 10 % to 90 % rise and fall times on the cable will be between 20 ns and 50 ns, so their delay contributions are small. There will be ringing on falling edges that can be damped, if required, by using Schottky diodes as shown.

When the Master SCL HIGH and LOW periods can be programmed separately, for example using control registers I2SCLH and I2SCLL of 89LPC932, the timings can allow for bus delays. The LOW period should be programmed to achieve the minimum 1300 ns plus the net delay in the slave's response data signal caused by bus and buffer delays. The longest data delay is the sum of the delay of the falling edge of SCL from master to slave and the delay of the rising edge of SDA from slave data to master. Because the buffer will 'stretch' the programmed SCL LOW period, the actual SCL frequency will be lower than calculated from the programmed clock periods. In the example for 25 meters the clock is stretched 400 ns, the falling edge of SCL is delayed 490 ns and the SDA rising edge is delayed 570 ns. The required additional LOW period is (490 ns + 570 ns) = 1060 ns and the I²C-bus specifications already include an allowance for a worst case bus rise time 0 % to 70 % of 425 ns. (The bus rise time can be 300 ns 30 % to 70 %, which means it can be 425 ns 0 % to 70 %. The 25 meter cable delay times as quoted already include all rise and fall times.) Therefore, the microcontroller only needs to be programmed with an additional (1060 ns − 400 ns − 425 ns) = 235 ns, making a total programmed LOW period 1535 ns. The programmed LOW will the be stretched by 400 ns to yield an actual bus LOW time of 1935 ns, which, allowing the minimum HIGH period of 600 ns, yields a cycle period of 2535 ns or 394 kHz.

Note that in both the 100 meter and 250 meter examples, the capacitive loading on the I²C-buses at each end is within the maximum allowed Standard mode loading of 400 pF, but exceeds the Fast mode limit. This is an example of a 'hybrid' mode because it relies on the response delays of Fast mode parts but uses (allowable) Standard mode bus loadings with rise times that contribute significantly to the system delays. The cables cause large propagation delays, so these systems need to operate well below the 400 kHz limit, but illustrate how they can still exceed the 100 kHz limit provided all parts are capable of Fast mode operation. The fastest example illustrates how the 400 kHz limit can be exceeded, provided masters and slaves have the required timings, namely smaller than the maximum allowed for Fast mode. Many NXP slaves have delays shorter than 600 ns and all Fm+ devices must be < 450 ns.

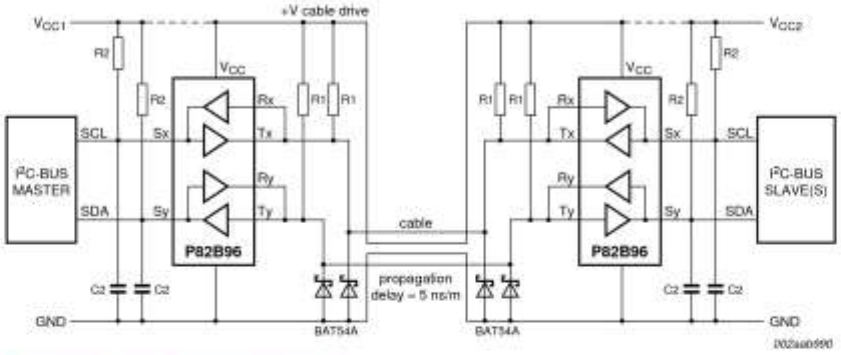
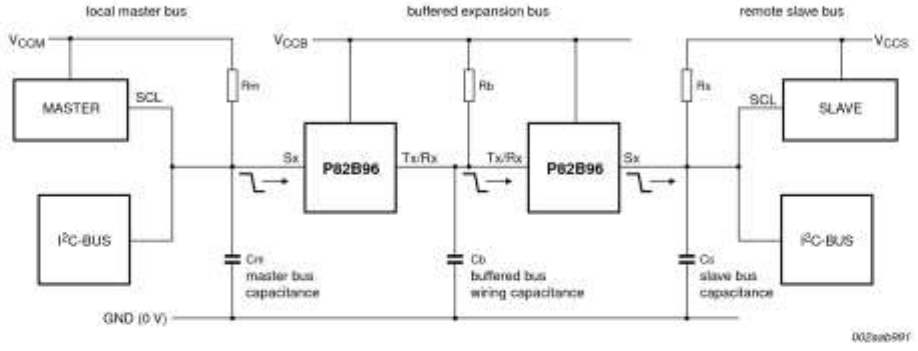


Fig 14. Driving ribbon or flat telephone cables

Table 6. Examples of bus capability
Refer to Figure 14.

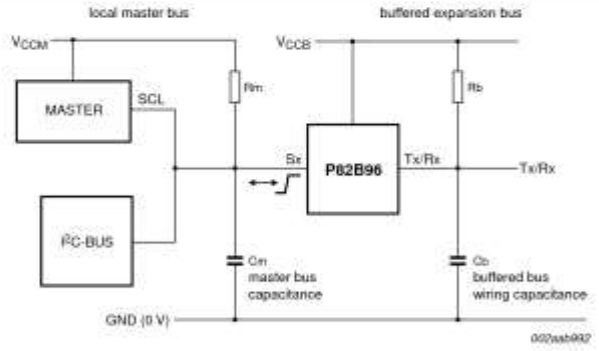
$+V_{CC1}$	+V cable	$+V_{CC2}$	R1 (Ω)	R2 (Ω)	C2 (pF)	Cable length	Cable capacitance	Cable delay	Set master nominal SCL		Effective bus clock speed	Maximum slave response delay
									HIGH period	LOW period		
5 V	12 V	5 V	750	2.2 k	400	250 m	n/a (delay based)	1.25 µs	600 ns	4000 ns	120 kHz	Normal spec. 400 kHz parts
5 V	12 V	5 V	750	2.2 k	220	100 m	n/a (delay based)	500 ns	600 ns	2600 ns	185 kHz	Normal spec. 400 kHz parts
3.3 V	5 V	3.3 V	330	1 k	220	25 m	1 nF	125 ns	600 ns	1500 ns	390 kHz	Normal spec. 400 kHz parts
3.3 V	5 V	3.3 V	330	1 k	100	3 m	120 pF	15 ns	600 ns	1000 ns	500 kHz	600 ns

10.1 Calculating system delays and bus clock frequency for a Fast mode system



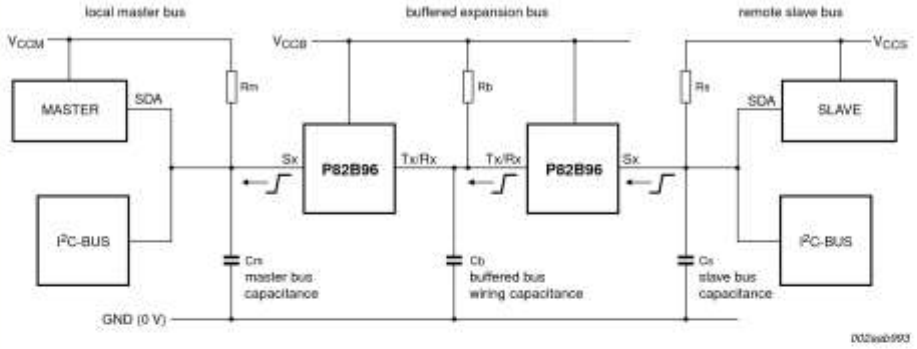
Effective delay of SCL at slave: $255 + 17V_{CCM} + (2.5 + 4 \times 10^9 C_b)V_{CCB} + 10V_{CCS}$ ns.
C = F; V = volts.

Fig 15. Falling edge of SCL at master is delayed by the buffers and bus fall times



Effective delay of SCL at master: $270 + RmCm + 0.7RbCb$ ns.
C = F; R = Ω.

Fig 16. Rising edge of SCL at master is delayed (clock stretch) by buffer and bus rise times



Effective delay of SDA at master = 270 + 0.2RsCs + 0.7 (RbCb + RmCm) ns.
C = F; R = Ω.

Fig 17. Rising edge of SDA at slave is delayed by the buffers and bus rise times

Figure 15, Figure 16, and Figure 17 show the P82B96 used to drive extended bus wiring, with relatively large capacitance, linking two Fast mode I²C-bus nodes. It includes simplified expressions for making the relevant timing calculations for 3.3 V or 5 V operation. Because the buffers and the wiring introduce timing delays, it may be necessary to decrease the nominal SCL frequency below 400 kHz. In most cases the actual bus frequency will be lower than the nominal Master timing due to bit-wise stretching of the clock periods.

The delay factors involved in calculation of the allowed bus speed are:

A — The propagation delay of the master signal through the buffers and wiring to the slave. The important delay is that of the falling edge of SCL because this edge 'requests' the data or acknowledge from a slave. See Figure 15.

B — The effective stretching of the nominal LOW period of SCL at the master caused by the buffer and bus rise times. See Figure 16.

C — The propagation delay of the slave's response signal through the buffers and wiring back to the master. The important delay is that of a rising edge in the SDA signal. Rising edges are always slower and are therefore delayed by a longer time than falling edges. (The rising edges are limited by the passive pull-up while falling edges are actively driven). See Figure 17.

The timing requirement in any I²C-bus system is that a slave's data response (which is provided in response to a falling edge of SCL) must be received at the master before the end of the corresponding LOW period of SCL as appears on the bus wiring at the master. Since all slaves will, as a minimum, satisfy the worst case timing requirements of a 400 kHz part, they must provide their response within the minimum allowed clock LOW period of 1300 ns. Therefore in systems that introduce additional delays it is only necessary to extend that minimum clock LOW period by any 'effective' delay of the slave's response. The effective delay of the slaves response equals the total delays in SCL falling

edge from the master reaching the slave (Figure 15) minus the effective delay (stretch) of the SCL rising edge (Figure 16) plus total delays in the slave's response data, carried on SDA, reaching the master (Figure 17).

The master microcontroller should be programmed to produce a nominal SCL LOW period = (1300 + A − B + C) ns, and should be programmed to produce the nominal minimum SCL HIGH period of 600 ns. Then a check should be made to ensure the cycle time is not shorter than the minimum 2500 ns. If found necessary, just increase either clock period.

Due to clock stretching, the SCL cycle time will always be longer than (600 + 1300 + A + C) ns.

Example:

The master bus has an RmCm product of 100 ns and V_{CCM} = 5 V.

The buffered bus has a capacitance of 1 nF and a pull-up resistor of 160 Ω to 5 V giving an RbCb product of 160 ns. The slave bus also has an RsCs product of 100 ns.

The microcontroller LOW period should be programmed to ≥ (1300 + 372.5 − 482 + 472) ns, that is ≥ 1662.5 ns.

Its HIGH period may be programmed to the minimum 600 ns.

The nominal microcontroller clock period will be ≥ (1662.5 + 600) ns = 2262.5 ns, equivalent to a frequency of 442 kHz.

The actual bus clock period, including the 482 ns clock stretch effect, will be below (nominal + stretch) = (2262.5 + 482) ns or ≥ 2745 ns, equivalent to an allowable frequency of 364 kHz.

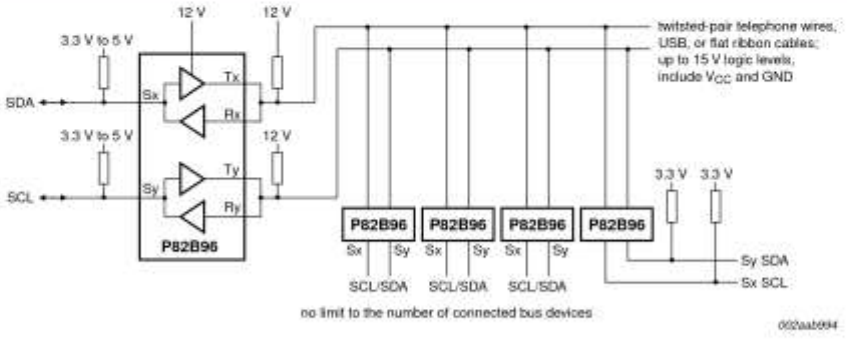
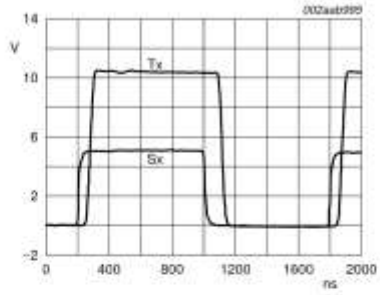
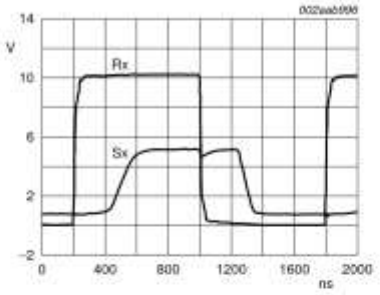


Fig 18. I²C-bus multipoint application



Frequency = 624 kHz

Fig 19. Propagation Sx to Tx (Sx pull-up to 5 V; Tx pull-up to V_{CC} = 10 V)



Ch1 frequency = 624 kHz

Fig 20. Propagation Rx to Sx (Sx pull-up to 5 V; Rx pull-up to V_{CC} = 10 V)

10.2 Negative undershoot below absolute minimum value

The reason why the IC pin reverse voltage on pins Tx and Rx in Table 4 "Limiting values" is specified at such a low value, −0.3 V, is **not** that applying larger voltages is likely to cause damage but that it is expected that, in normal applications, there is no reason why larger DC voltages will be applied. This 'absolute maximum' specification is intended to be a DC or continuous ratings and the nominal DC I²C-bus voltage LOW usually does not even reach 0 V. Inside P82B96 at every pin there is a large protective diode connected to the GND pin and that diode will start to conduct when the pin voltage is more than about −0.55 V with respect to GND at 25 °C ambient.

Figure 21 shows the measured characteristic for one of those diodes inside P82B96. The plot was made using a curve tracer that applies 50 Hz mains voltage via a series resistor, so the pulse durations are long duration (several milliseconds) and are reaching peaks of over 2 A when more than −1.5 V is applied. The IC becomes very hot during this testing but it was not damaged. Whenever there is current flowing in any of these diodes it is possible that there can be faulty operation of any IC. For that reason we put a specification on the negative voltage that is allowed to be applied. It is selected so that, at the highest allowed junction temperature, there will be a big safety factor that guarantees the diode will not conduct and then we do not need to make any 100 % production tests to guarantee the published specification.

For the P82B96, in specific applications, there will always be transient overshoot and ringing on the wiring that can cause these diodes to conduct. Therefore we designed the IC to withstand those transients and as a part of the qualification procedure we made tests, using DC currents to more than twice the normal bus sink currents, to be sure that the IC was not affected by those currents. For example, the Tx/Ty and Rx/Ry pins were tested to at least −80 mA which, from Figure 21, would be more than −0.8 V. The correct functioning of the P82B96 is not affected even by those large currents. The Absolute Maximum (DC) ratings are not intended to apply to transients but to steady state conditions. This explains why you will never see any problems in practice even if, during transients, more than −0.3 V is applied to the bus interface pins of P82B96.

Figure 21 "Diode characteristic curve" also explains how the general Absolute Maximum DC specification was selected. The current at 25 °C is near zero at −0.55 V. The P82B96 is allowed to operate with +125 °C junction and that would cause this diode voltage to decrease by 100 × 2 mV = 200 mV. So for zero current we need to specify −0.35 V and we publish −0.3 V just to have some extra margin.

Remark: You should not be concerned about the **transients** generated on the wiring by a P82B96 in normal applications and that is input to the Tx/Rx or Ty/Ry pins of another P82B96. Because not all ICs that may be driven by P82B96 are designed to tolerate negative transients, in Section 10.2.1 "Example with questions and answers" we show they can be managed if required.

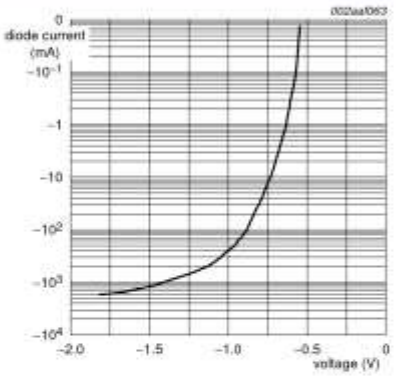


Fig 21. Diode characteristic curve

10.2.1 Example with questions and answers

Question: On a falling edge of Tx we measure undershoot at −800 mV at the linked Tx, Rx pins of the P82B96 that is generating the LOW, but the P82B96 data sheet specifies minimum −0.3 V. Does this mean that we violate the data sheet absolute value?

Answer: For P82B96 the −0.3 V Absolute Maximum rating is not intended to apply to transients, it is a DC rating. As shown in Figure 22, there is no theoretical reason for any undershoot at the IC that is driving the bus LOW and no significant undershoot should be observed when using reasonable care with the ground connection of the 'scope. It is more likely that undershoot observed at a driving P82B96 is caused by local stray inductance and capacitance in the circuit and by the oscilloscope connections. As shown, undershoot will be generated by PCB traces, wiring, or cables driven by a P82B96 because the allowed value of the I^2C-bus pull-up resistor generally is larger than that required to correctly terminate the wiring. In this example, with no IC connected at the end of the wiring, the undershoot is about 2 V.

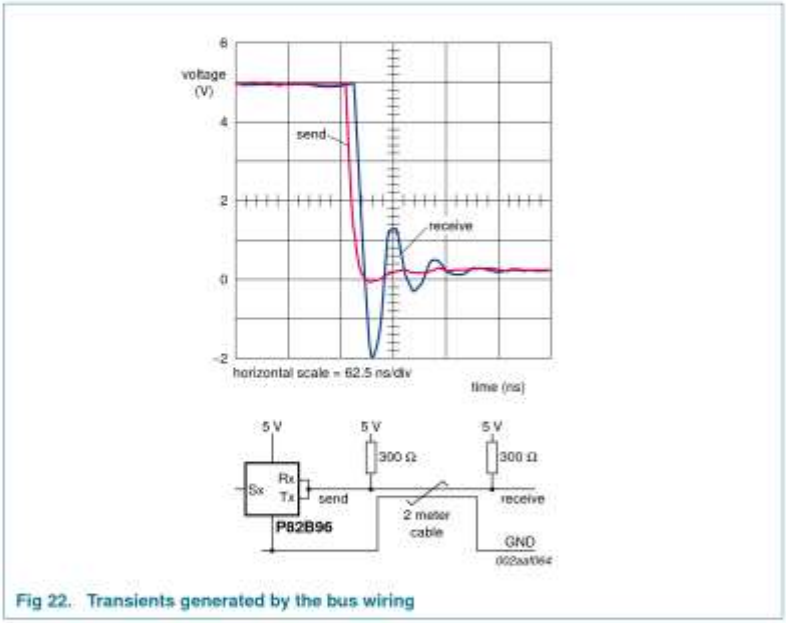


Fig 22. Transients generated by the bus wiring

Question: We have 2 meters of cable in a bus that joins the Tx/Rx sides of two P82B96 devices. When one Tx drives LOW the other P82B96 Tx/Rx is driven to −0.8 V for over 50 ns. What is the expected value and the theoretically allowed value of undershoot?

Answer: Because the cable joining the two P82B96s is a 'transmission line' that will have a characteristic impedance around 100 Ω and it will be terminated by pull-up resistors that are larger than that characteristic impedance there will always be negative undershoot generated. The duration of the undershoot is a function of the cable length and the input impedance of the connected IC. As shown in Figure 23, the transient undershoot will be limited, by the diodes inside P82B96, to around −0.8 V and that will not cause problems for P82B96. Those transients will **not** be passed inside the IC to the Sx/Sy side of the IC.

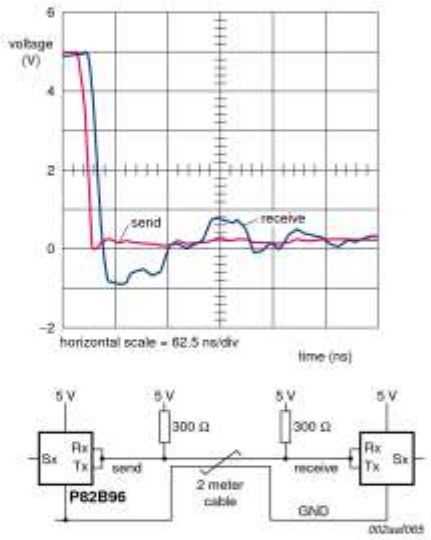


Fig 23. Wiring transients limited by the diodes in P82B96

Question: If we input 800 mV undershoot at Tx, Rx pins, what kind of problem is expected?

Answer: When that undershoot is generated by another P82B96 and is simply the result of the system wiring, then there will be no problems.

Question: Will we have any functional problem or reliability problem?

Answer: No.

Question: If we add 100 Ω to 200 Ω at signal line, the overshoot becomes slightly smaller. Is this a good idea?

Answer: No, it is not necessary to add any resistance. When the logic signal generated by Tx or Ty of P82B96 drives long traces or wiring with ICs other than P82B96 being driven, then adding a Schottky diode (BAT54A) as shown in Figure 24 will clamp the wiring undershoot to a value that will not cause conduction of the IC's internal diodes.

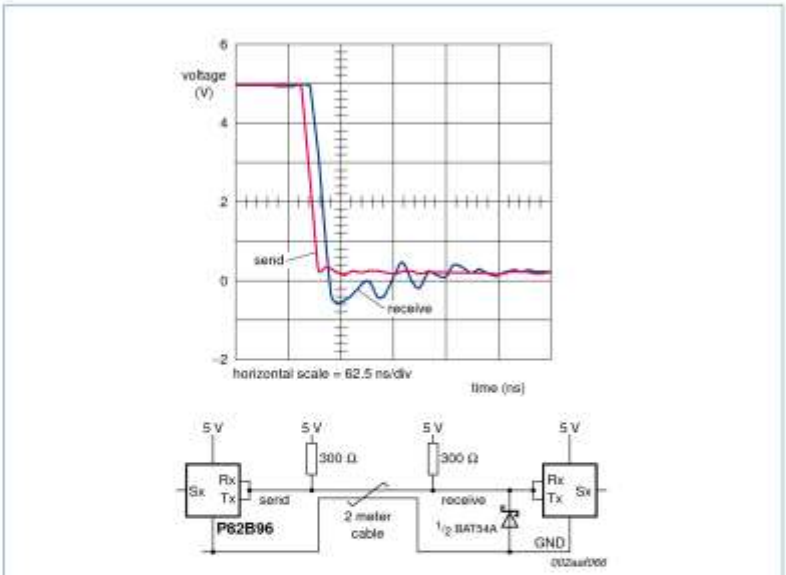
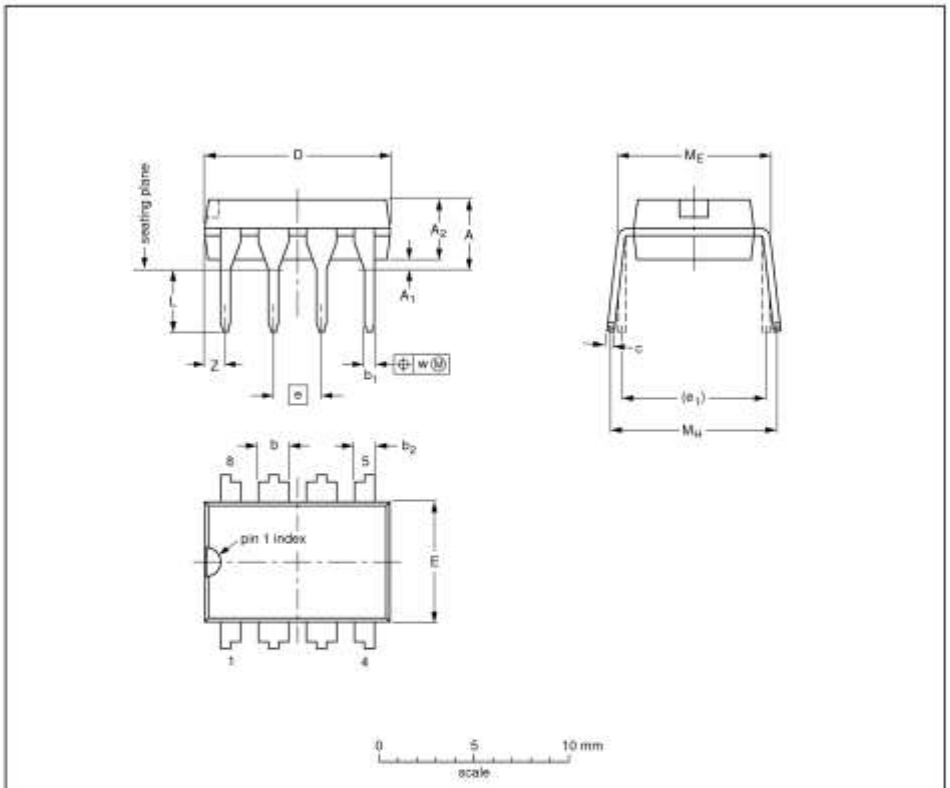


Fig 24. Wiring transients limited by a Schottky diode

11. Package outline

DIP8: plastic dual in-line package; 8 leads (300 mil) **SOT97-1**



DIMENSIONS (inch dimensions are derived from the original mm dimensions)

UNIT	A max.	A_1 min.	A_2 max.	b	b_1	b_2	c	$D^{(1)}$	$E^{(1)}$	e	e_1	L	M_E	M_H	w	$Z^{(1)}$ max.
mm	4.2	0.51	3.2	1.73 1.14	0.53 0.38	1.07 0.89	0.36 0.23	9.8 9.2	6.48 6.20	2.54	7.62	3.60 3.05	8.25 7.80	10.0 8.3	0.254	1.15
inches	0.17	0.02	0.13	0.068 0.045	0.021 0.015	0.042 0.035	0.014 0.009	0.39 0.38	0.26 0.24	0.1	0.3	0.14 0.12	0.32 0.31	0.39 0.33	0.01	0.045

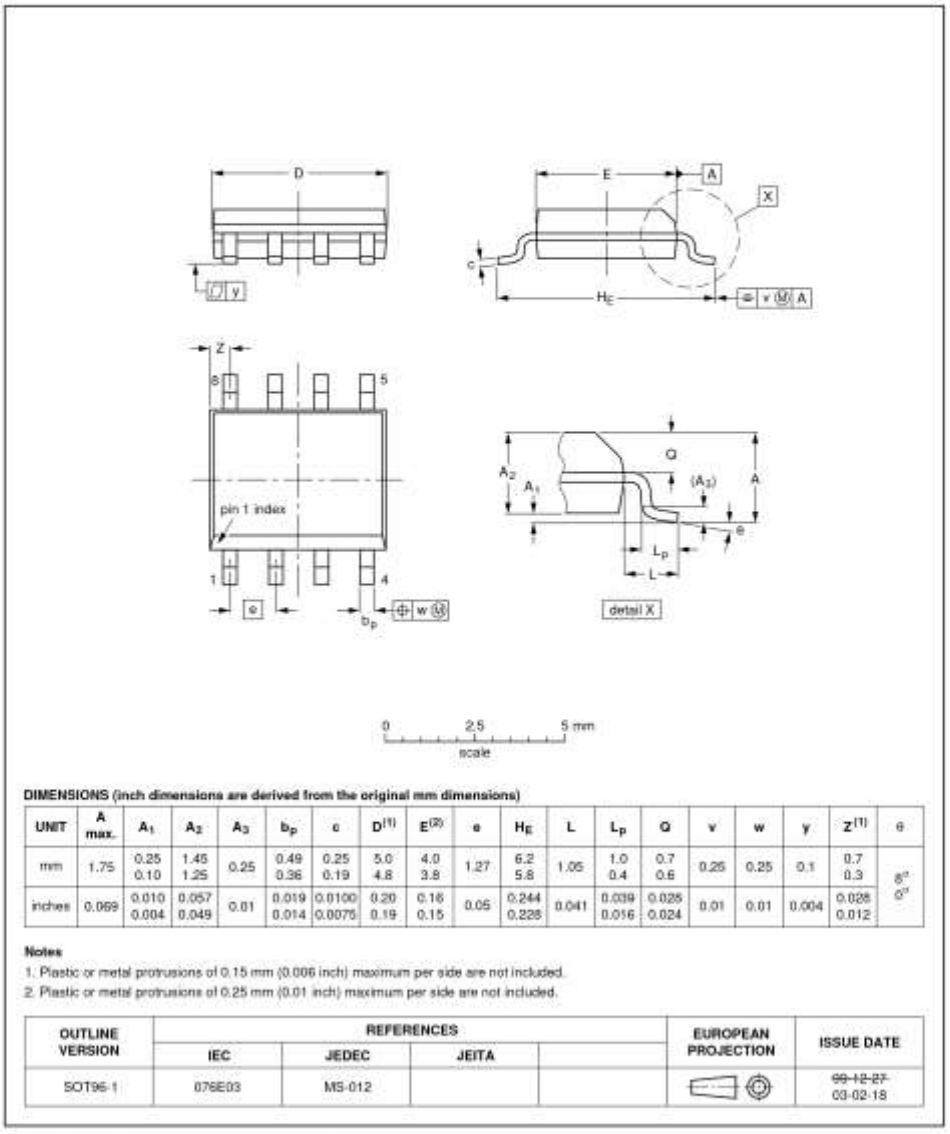
Note

1. Plastic or metal protrusions of 0.25 mm (0.01 inch) maximum per side are not included.

OUTLINE VERSION	REFERENCES				EUROPEAN PROJECTION	ISSUE DATE
	IEC	JEDEC	JEITA			
SOT97-1	050G01	MO-001	SC-504-8			~~99-12-27~~ 03-02-13

Fig 25. Package outline SOT97-1 (DIP8)

SO8: plastic small outline package; 8 leads; body width 3.9 mm **SOT96-1**



DIMENSIONS (inch dimensions are derived from the original mm dimensions)

UNIT	A max.	A_1	A_2	A_3	b_p	c	$D^{(1)}$	$E^{(2)}$	e	H_E	L	L_p	Q	v	w	y	$Z^{(1)}$	θ
mm	1.75	0.25 0.10	1.45 1.25	0.25	0.49 0.36	0.25 0.19	5.0 4.8	4.0 3.8	1.27	6.2 5.8	1.05	1.0 0.4	0.7 0.6	0.25	0.25	0.1	0.7 0.3	8° 0°
inches	0.069	0.010 0.004	0.057 0.049	0.01	0.019 0.014	0.0100 0.0075	0.20 0.19	0.16 0.15	0.05	0.244 0.228	0.041	0.039 0.016	0.035 0.024	0.01	0.01	0.004	0.028 0.012	

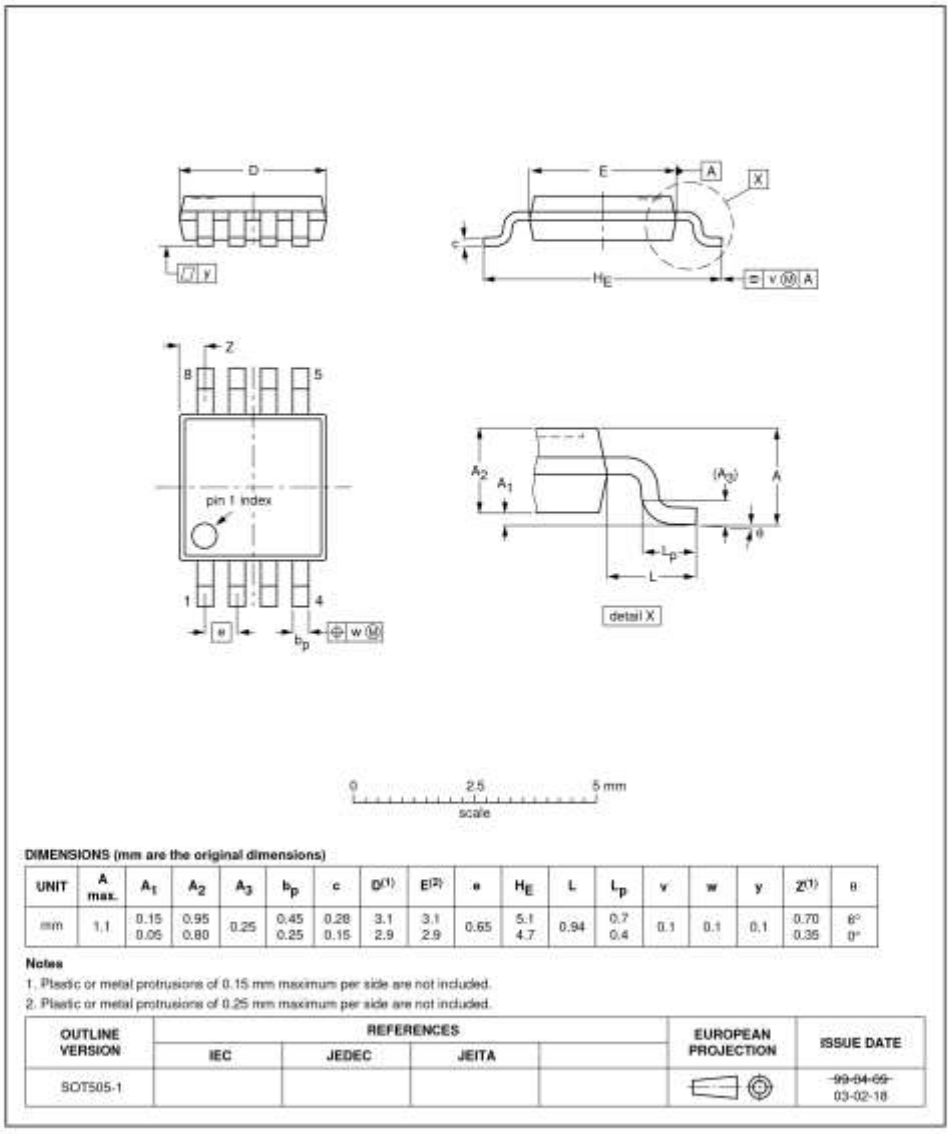
Notes

1. Plastic or metal protrusions of 0.15 mm (0.006 inch) maximum per side are not included.
2. Plastic or metal protrusions of 0.25 mm (0.01 inch) maximum per side are not included.

OUTLINE VERSION	REFERENCES				EUROPEAN PROJECTION	ISSUE DATE
	IEC	JEDEC	JEITA			
SOT96-1	076E03	MS-012				~~99-12-27~~ 03-02-18

Fig 26. Package outline SOT96-1 (SO8)

TSSOP8: plastic thin shrink small outline package; 8 leads; body width 3 mm **SOT505-1**



DIMENSIONS (mm are the original dimensions)

UNIT	A max.	A_1	A_2	A_3	b_p	c	$D^{(1)}$	$E^{(2)}$	e	H_E	L	L_p	v	w	y	$Z^{(1)}$	θ
mm	1.1	0.15 0.05	0.95 0.80	0.25	0.45 0.25	0.28 0.15	3.1 2.9	3.1 2.9	0.65	5.1 4.7	0.94	0.7 0.4	0.1	0.1	0.1	0.70 0.35	8° 0°

Notes

1. Plastic or metal protrusions of 0.15 mm maximum per side are not included.
2. Plastic or metal protrusions of 0.25 mm maximum per side are not included.

OUTLINE VERSION	REFERENCES				EUROPEAN PROJECTION	ISSUE DATE
	IEC	JEDEC	JEITA			
SOT505-1						~~99-04-09~~ 03-02-18

Fig 27. Package outline SOT505-1 (TSSOP8)

12. Soldering of SMD packages

This text provides a very brief insight into a complex technology. A more in-depth account of soldering ICs can be found in Application Note *AN10365 "Surface mount reflow soldering description"*.

12.1 Introduction to soldering

Soldering is one of the most common methods through which packages are attached to Printed Circuit Boards (PCBs), to form electrical circuits. The soldered joint provides both the mechanical and the electrical connection. There is no single soldering method that is ideal for all IC packages. Wave soldering is often preferred when through-hole and Surface Mount Devices (SMDs) are mixed on one printed wiring board; however, it is not suitable for fine pitch SMDs. Reflow soldering is ideal for the small pitches and high densities that come with increased miniaturization.

12.2 Wave and reflow soldering

Wave soldering is a joining technology in which the joints are made by solder coming from a standing wave of liquid solder. The wave soldering process is suitable for the following:

- Through-hole components
- Leaded or leadless SMDs, which are glued to the surface of the printed circuit board

Not all SMDs can be wave soldered. Packages with solder balls, and some leadless packages which have solder lands underneath the body, cannot be wave soldered. Also, leaded SMDs with leads having a pitch smaller than ~0.6 mm cannot be wave soldered, due to an increased probability of bridging.

The reflow soldering process involves applying solder paste to a board, followed by component placement and exposure to a temperature profile. Leaded packages, packages with solder balls, and leadless packages are all reflow solderable.

Key characteristics in both wave and reflow soldering are:

- Board specifications, including the board finish, solder masks and vias
- Package footprints, including solder thieves and orientation
- The moisture sensitivity level of the packages
- Package placement
- Inspection and repair
- Lead-free soldering versus SnPb soldering

12.3 Wave soldering

Key characteristics in wave soldering are:

- Process issues, such as application of adhesive and flux, clinching of leads, board transport, the solder wave parameters, and the time during which components are exposed to the wave
- Solder bath specifications, including temperature and impurities

12.4 Reflow soldering

Key characteristics in reflow soldering are:

- Lead-free versus SnPb soldering; note that a lead-free reflow process usually leads to higher minimum peak temperatures (see Figure 28) than a SnPb process, thus reducing the process window
- Solder paste printing issues including smearing, release, and adjusting the process window for a mix of large and small components on one board
- Reflow temperature profile; this profile includes preheat, reflow (in which the board is heated to the peak temperature) and cooling down. It is imperative that the peak temperature is high enough for the solder to make reliable solder joints (a solder paste characteristic). In addition, the peak temperature must be low enough that the packages and/or boards are not damaged. The peak temperature of the package depends on package thickness and volume and is classified in accordance with Table 7 and 8

Table 7. SnPb eutectic process (from J-STD-020C)

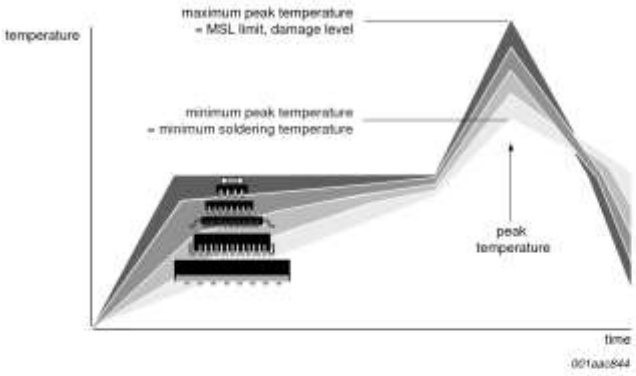
Package thickness (mm)	Package reflow temperature (°C)	
	Volume (mm³)	
	< 350	≥ 350
< 2.5	235	220
≥ 2.5	220	220

Table 8. Lead-free process (from J-STD-020C)

Package thickness (mm)	Package reflow temperature (°C)		
	Volume (mm³)		
	< 350	350 to 2000	> 2000
< 1.6	260	260	260
1.6 to 2.5	260	250	245
> 2.5	250	245	245

Moisture sensitivity precautions, as indicated on the packing, must be respected at all times.

Studies have shown that small packages reach higher temperatures during reflow soldering, see Figure 28.



MSL: Moisture Sensitivity Level

Fig 28. Temperature profiles for large and small components

For further information on temperature profiles, refer to Application Note *AN10365 "Surface mount reflow soldering description"*.

13. Soldering of through-hole mount packages

13.1 Introduction to soldering through-hole mount packages

This text gives a very brief insight into wave, dip and manual soldering.

Wave soldering is the preferred method for mounting of through-hole mount IC packages on a printed-circuit board.

13.2 Soldering by dipping or by solder wave

Driven by legislation and environmental forces the worldwide use of lead-free solder pastes is increasing. Typical dwell time of the leads in the wave ranges from 3 seconds to 4 seconds at 250 °C or 265 °C, depending on solder material applied, SnPb or Pb-free respectively.

The total contact time of successive solder waves must not exceed 5 seconds.

The device may be mounted up to the seating plane, but the temperature of the plastic body must not exceed the specified maximum storage temperature ($T_{stg(max)}$). If the printed-circuit board has been pre-heated, forced cooling may be necessary immediately after soldering to keep the temperature within the permissible limit.

13.3 Manual soldering

Apply the soldering iron (24 V or less) to the lead(s) of the package, either below the seating plane or not more than 2 mm above it. If the temperature of the soldering iron bit is less than 300 °C it may remain in contact for up to 10 seconds. If the bit temperature is between 300 °C and 400 °C, contact may be up to 5 seconds.

13.4 Package related soldering information

Table 9. Suitability of through-hole mount IC packages for dipping and wave soldering

Package	Soldering method	
	Dipping	Wave
CPGA, HCPGA	-	suitable
DBS, DIP, HDIP, RDBS, SDIP, SIL	suitable	suitable[1]
PMFP[2]	-	not suitable

[1] For SDIP packages, the longitudinal axis must be parallel to the transport direction of the printed-circuit board.

[2] For PMFP packages hot bar soldering or manual soldering is suitable.

14. Abbreviations

Table 10. Abbreviations

Acronym	Description
CDM	Charged Device Model
DDC	Display Data Channel
ESD	ElectroStatic Discharge
HBM	Human Body Model
IC	Integrated Circuit
I²C-bus	Inter IC bus
MM	Machine Model
SMBus	System Management Bus

15. Revision history

Table 11. Revision history

Document ID	Release date	Data sheet status	Change notice	Supersedes
P82B96_8	20091110	Product data sheet	-	P82B96_7
Modifications:	• Table 4 "Limiting values": added Table note [1]. • Added Section 10.2 "Negative undershoot below absolute minimum value".			
P82B96_7	20090212	Product data sheet	-	P82B96_6
P82B96_6	20080131	Product data sheet	-	P82B96_5
P82B96_5	20060127	Product data sheet	-	P82B96_4
P82B96_4 (9397 750 12932)	20040329	Product data	-	P82B96_3
P82B96_3 (9397 750 11351)	20030402	Product data	853-2241 29602 of 2003 Feb 28	P82B96_2
P82B96_2 (9397 750 11093)	20030220	Product data	853-2241 29410 of 2003 Jan 22	P82B96_1
P82B96_1 (9397 750 08122)	20010306	Product data	853-2241 25758 of 2001 Mar 06	-

16. Legal information

16.1 Data sheet status

Document status[1][2]	Product status[3]	Definition
Objective [short] data sheet	Development	This document contains data from the objective specification for product development.
Preliminary [short] data sheet	Qualification	This document contains data from the preliminary specification.
Product [short] data sheet	Production	This document contains the product specification.

[1] Please consult the most recently issued document before initiating or completing a design.

[2] The term 'short data sheet' is explained in section "Definitions".

[3] The product status of device(s) described in this document may have changed since this document was published and may differ in case of multiple devices. The latest product status information is available on the Internet at URL http://www.nxp.com.

16.2 Definitions

Draft — The document is a draft version only. The content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included herein and shall have no liability for the consequences of use of such information.

Short data sheet — A short data sheet is an extract from a full data sheet with the same product type number(s) and title. A short data sheet is intended for quick reference only and should not be relied upon to contain detailed and full information. For detailed and full information see the relevant full data sheet, which is available on request via the local NXP Semiconductors sales office. In case of any inconsistency or conflict with the short data sheet, the full data sheet shall prevail.

16.3 Disclaimers

General — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in medical, military, aircraft, space or life support equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors accepts no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Limiting values — Stress above one or more limiting values (as defined in the Absolute Maximum Ratings System of IEC 60134) may cause permanent damage to the device. Limiting values are stress ratings only and operation of the device at these or any other conditions above those given in the Characteristics sections of this document is not implied. Exposure to limiting values for extended periods may affect device reliability.

Terms and conditions of sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at http://www.nxp.com/profile/terms, including those pertaining to warranty, intellectual property rights infringement and limitation of liability, unless explicitly otherwise agreed to in writing by NXP Semiconductors. In case of any inconsistency or conflict between information in this document and such terms and conditions, the latter will prevail.

No offer to sell or license — Nothing in this document may be interpreted or construed as an offer to sell products that is open for acceptance or the grant, conveyance or implication of any license under any copyrights, patents or other industrial or intellectual property rights.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from national authorities.

16.4 Trademarks

Notice: All referenced brands, product names, service names and trademarks are the property of their respective owners.

I²C-bus — logo is a trademark of NXP B.V.

17. Contact information

For more information, please visit: http://www.nxp.com

For sales office addresses, please send an email to: salesaddresses@nxp.com

18. Contents

Date of release: 10 November 2009
Document identifier: P82B96_8

3.7. ADC MCP3421

MICROCHIP

MCP3421

18-Bit Analog-to-Digital Converter with I²C Interface and On-Board Reference

Features

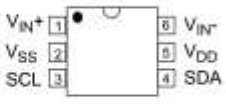
- 18-bit ΔΣ ADC in a SOT-23-6 package
- Differential input operation
- Self calibration of Internal Offset and Gain per each conversion
- On-board Voltage Reference:
  - Accuracy: 2.048V ± 0.05%
  - Drift: 5 ppm/°C
- On-board Programmable Gain Amplifier (PGA):
  - Gains of 1,2,4 or 8
- On-board Oscillator
- INL: 10 ppm of FSR (FSR = 4.096V/PGA)
- Programmable Data Rate Options:
  - 3.75 SPS (18 bits)
  - 15 SPS (16 bits)
  - 60 SPS (14 bits)
  - 240 SPS (12 bits)
- One-Shot or Continuous Conversion Options
- Low current consumption:
  - 145 µA typical (V_{DD}= 3V, Continuous Conversion)
  - 39 µA typical (V_{DD}= 3V, One-Shot Conversion with 1 SPS)
- Supports I²C Serial Interface:
  - Standard, Fast and High Speed Modes
- Single Supply Operation: 2.7V to 5.5V
- Extended Temperature Range: -40°C to 125°C

Typical Applications

- Portable Instrumentation
- Weigh Scales and Fuel Gauges
- Temperature Sensing with RTD, Thermistor, and Thermocouple
- Bridge Sensing for Pressure, Strain, and Force.

Package Types

SOT-23-6 Top View



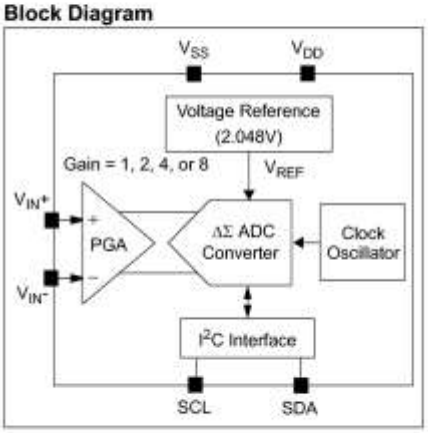
Description

The MCP3421 is a single channel low-noise, high accuracy ΔΣ A/D converter with differential inputs and up to 18 bits of resolution in a small SOT-23-6 package. The on-board precision 2.048V reference voltage enables an input range of ±2.048V differentially (Δ voltage = 4.096V). The device uses a two-wire I²C compatible serial interface and operates from a single 2.7V to 5.5V power supply.

The MCP3421 device performs conversion at rates of 3.75, 15, 60, or 240 samples per second (SPS) depending on the user controllable configuration bit settings using the two-wire I²C serial interface. This device has an on-board programmable gain amplifier (PGA). The user can select the PGA gain of x1, x2, x4, or x8 before the analog-to-digital conversion takes place. This allows the MCP3421 device to convert a smaller input signal with high resolution. The device has two conversion modes: (a) Continuous mode and (b) One-Shot mode. In One-Shot mode, the device enters a low current standby mode automatically after one conversion. This reduces current consumption greatly during idle periods.

The MCP3421 device can be used for various high accuracy analog-to-digital data conversion applications where design simplicity, low power, and small footprint are major considerations.

Block Diagram



MCP3421

1.0 ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings†

V_{DD} 7.0V
All inputs and outputs w.r.t V_{SS} ..... −0.3V to VDD+0.3V
Differential Input Voltage ..... $|V_{DD} - V_{SS}|$
Output Short Circuit Current ..... Continuous
Current at Input Pins ..... ±2 mA
Current at Output and Supply Pins ..... ±10 mA
Storage Temperature ..... -65°C to +150°C
Ambient Temp. with power applied ..... -55°C to +125°C
ESD protection on all pins ..... ≥ 6 kV HBM, ≥ 400V MM
Maximum Junction Temperature (T_J) ..... +150°C

†Notice: Stresses above those listed under "Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operational listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

ELECTRICAL CHARACTERISTICS

Electrical Specifications: Unless otherwise specified, all parameters apply for T_A = -40°C to +85°C, V_{DD} = +5.0V, V_{SS} = 0V, $V_{IN}+ = V_{IN}- = V_{REF}/2$. All ppm units use 2*$V_{REF}$ as full-scale range.

Parameters	Sym	Min	Typ	Max	Units	Conditions
Analog Inputs						
Differential Input Range		—	±2.048/PGA	—	V	$V_{IN} = V_{IN}+ - V_{IN}-$
Common-Mode Voltage Range (absolute) (Note 1)		V_{SS}-0.3	—	V_{DD}+0.3	V	
Differential Input Impedance (Note 2)	Z_{IND} (f)	—	2.25/PGA	—	MΩ	During normal mode operation
Common Mode Input Impedance	Z_{INC} (f)	—	25	—	MΩ	PGA = 1, 2, 4, 8
System Performance						
Resolution and No Missing Codes (Note 8)		12	—	—	Bits	DR = 240 SPS
		14	—	—	Bits	DR = 60 SPS
		16	—	—	Bits	DR = 15 SPS
		18	—	—	Bits	DR = 3.75 SPS
Data Rate (Note 3)	DR	176	240	328	SPS	S1,S0 = '00', (12 bits mode)
		44	60	82	SPS	S1,S0 = '01', (14 bits mode)
		11	15	20.5	SPS	S1,S0 = '10', (16 bits mode)
		2.75	3.75	5.1	SPS	S1,S0 = '11', (18 bits mode)
Output Noise		—	1.5	—	μV_{RMS}	T_A = 25°C, DR = 3.75 SPS, PGA = 1, V_{IN} = 0

Note 1: Any input voltage below or greater than this voltage causes leakage current through the ESD diodes at the input pins. This parameter is ensured by characterization and not 100% tested.
2: This input impedance is due to 3.2 pF internal input sampling capacitor.
3: The total conversion speed includes auto-calibration of offset and gain.
4: INL is the difference between the endpoints line and the measured code at the center of the quantization band.
5: Includes all errors from on-board PGA and V_{REF}.
6: Full Scale Range (FSR) = 2 x 2.048/PGA = 4.096/PGA.
7: This parameter is ensured by characterization and not 100% tested.
8: This parameter is ensured by design and not 100% tested.

ELECTRICAL CHARACTERISTICS (CONTINUED)

Electrical Specifications: Unless otherwise specified, all parameters apply for T_A = -40°C to +85°C, V_{DD} = +5.0V, V_{SS} = 0V, V_{IN}+ = V_{IN}- = V_{REF}/2. All ppm units use 2*V_{REF} as full-scale range.

Parameters	Sym	Min	Typ	Max	Units	Conditions
Integral Nonlinearity (Note 4)	INL	—	10	35	ppm of FSR	DR = 3.75 SPS (Note 6)
Internal Reference Voltage	V_{REF}	—	2.048	—	V	
Gain Error (Note 5)		—	0.05	0.35	%	PGA = 1, DR = 3.75 SPS
PGA Gain Error Match (Note 5)		—	0.1	—	%	Between any 2 PGA gains
Gain Error Drift (Note 5)		—	5	40	ppm/°C	PGA=1, DR=3.75 SPS
Offset Error	V_{OS}	—	15	40	µV	Tested at PGA = 1 V_{DD} = 5.0V and DR = 3.75 SPS
Offset Drift vs. Temperature		—	50	—	nV/°C	V_{DD} = 5.0V
Common-Mode Rejection		—	105	—	dB	at DC and PGA =1,
		—	110	—	dB	at DC and PGA =8, T_A = +25°C
Gain vs. V_{DD}		—	5	—	ppm/V	T_A = +25°C, V_{DD} = 2.7V to 5.5V, PGA = 1
Power Supply Rejection at DC		—	100	—	dB	T_A = +25°C, V_{DD} = 2.7V to 5.5V, PGA = 1
Power Requirements						
Voltage Range	V_{DD}	2.7	—	5.5	V	
Supply Current during Conversion	I_{DDA}	—	155	190	µA	V_{DD} = 5.0V
		—	145	—	µA	V_{DD} = 3.0V
Supply Current during Standby Mode	I_{DDS}	—	0.1	0.5	µA	
I²C Digital Inputs and Digital Outputs						
High level input voltage	V_{IH}	0.7 V_{DD}	—	V_{DD}	V	
Low level input voltage	V_{IL}	—	—	0.3V_{DD}	V	
Low level output voltage	V_{OL}	—	—	0.4	V	I_{OL} = 3 mA, V_{DD} = +5.0V
Hysteresis of Schmitt Trigger for inputs (Note 7)	V_{HYST}	0.05V_{DD}	—	—	V	f_{SCL} = 100 kHz
Supply Current when I²C bus line is active	I_{DDB}	—	—	10	µA	
Input Leakage Current	I_{ILH}	—	—	1	µA	V_{IH} = 5.5V
	I_{ILL}	-1	—	—	µA	V_{IL} = GND
Pin Capacitance and I²C Bus Capacitance						
Pin capacitance	C_{PIN}	—	—	10	pF	
I²C Bus Capacitance	C_b	—	—	400	pF	
Thermal Characteristics						
Specified Temperature Range	T_A	-40	—	+85	°C	
Operating Temperature Range	T_A	-40	—	+125	°C	
Storage Temperature Range	T_A	-65	—	+150	°C	

Note 1: Any input voltage below or greater than this voltage causes leakage current through the ESD diodes at the input pins. This parameter is ensured by characterization and not 100% tested.
2: This input impedance is due to 3.2 pF internal input sampling capacitor.
3: The total conversion speed includes auto-calibration of offset and gain.
4: INL is the difference between the endpoints line and the measured code at the center of the quantization band.
5: Includes all errors from on-board PGA and V_{REF}.
6: Full Scale Range (FSR) = 2 x 2.048/PGA = 4.096/PGA.
7: This parameter is ensured by characterization and not 100% tested.
8: This parameter is ensured by design and not 100% tested.

MCP3421

2.0 TYPICAL PERFORMANCE CURVES

Note: The graphs and tables provided following this note are a statistical summary based on a limited number of samples and are provided for informational purposes only. The performance characteristics listed herein are not tested or guaranteed. In some graphs or tables, the data presented may be outside the specified operating range (e.g., outside specified power supply range) and therefore outside the warranted range.

Note: Unless otherwise indicated, T_A = -40°C to +85°C, V_{DD} = +5.0V, V_{SS} = 0V, $V_{IN}^+ = V_{IN}^- = V_{REF}/2$.

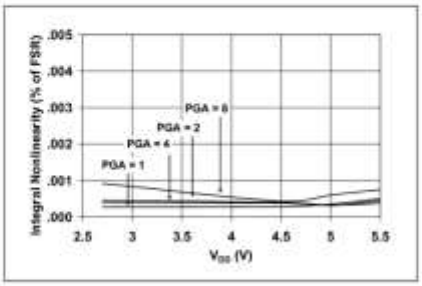


FIGURE 2-1: INL vs. Supply Voltage (V_{DD}).

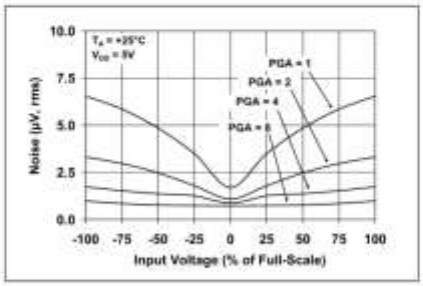


FIGURE 2-4: Noise vs. Input Voltage.

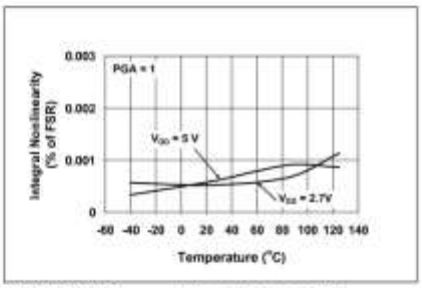


FIGURE 2-2: INL vs. Temperature.

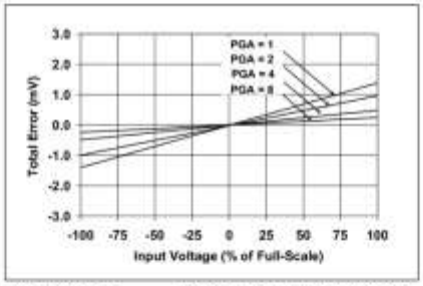


FIGURE 2-5: Total Error vs. Input Voltage.

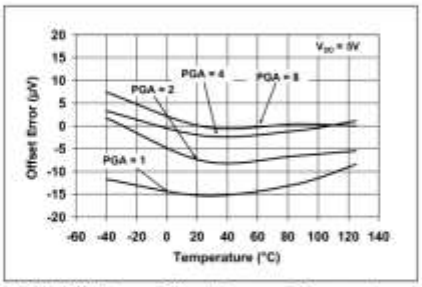


FIGURE 2-3: Offset Error vs. Temperature.

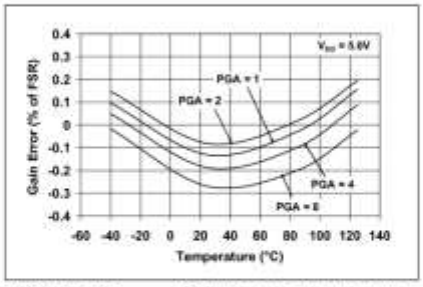


FIGURE 2-6: Gain Error vs. Temperature.

Note: Unless otherwise indicated, T_A = -40°C to +85°C, V_{DD} = +5.0V, V_{SS} = 0V, $V_{IN}+ = V_{IN}- = V_{REF}/2$.

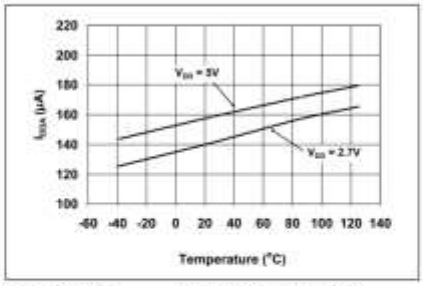


FIGURE 2-7: I_{DDA} vs. Temperature.

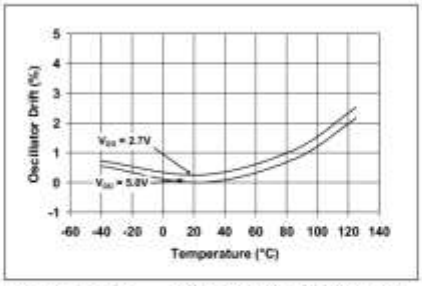


FIGURE 2-10: OSC Drift vs. Temperature.

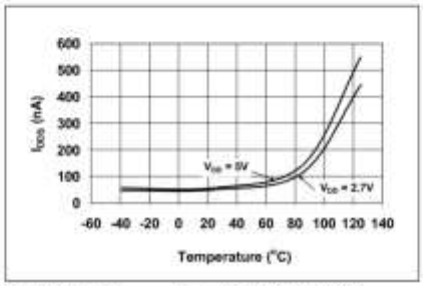


FIGURE 2-8: I_{DDS} vs. Temperature.

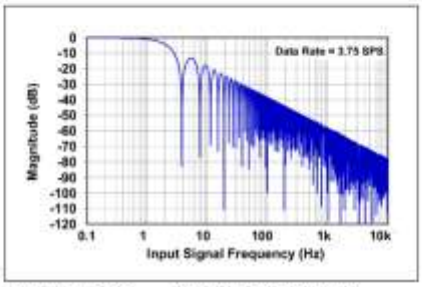


FIGURE 2-11: Frequency Response.

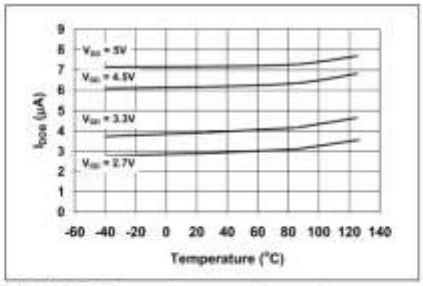


FIGURE 2-9: I_{DDB} vs. Temperature.

MCP3421

3.0 PIN DESCRIPTIONS

TABLE 3-1: PIN FUNCTION TABLE

Pin No	Sym	Function
1	V_{IN}+	Non-Inverting Analog Input Pin
2	V_{SS}	Ground Pin
3	SCL	Serial Clock Input Pin of the I^2C Interface
4	SDA	Bidirectional Serial Data Pin of the I^2C Interface
5	V_{DD}	Positive Supply Voltage Pin
6	V_{IN}-	Inverting Analog Input Pin

3.1 Analog Inputs (V_{IN}+, V_{IN}-)

V_{IN}+ and V_{IN}- are differential signal input pins. The MCP3421 device accepts a fully differential analog input signal which is connected on the V_{IN}+ and V_{IN}- input pins. The differential voltage that is converted is defined by $V_{IN} = (V_{IN}+ - V_{IN}-)$ where V_{IN}+ is the voltage applied at the V_{IN}+ pin and V_{IN}- is the voltage applied at the V_{IN}- pin. The input signal level is amplified by the programmable gain amplifier (PGA) before the conversion. The differential input voltage should not exceed an absolute of (2* V_{REF}/PGA) for accurate measurement, where V_{REF} is the internal reference voltage (2.048V) and PGA is the PGA gain setting. The converter output code will saturate if the input range exceeds (2* V_{REF}/PGA).

The absolute voltage range on each of the differential input pins is from V_{SS}-0.3V to V_{DD}+0.3V. Any voltage above or below this range will cause leakage currents through the Electrostatic Discharge (ESD) diodes at the input pins. This ESD current can cause unexpected performance of the device. The common mode of the analog inputs should be chosen such that both the differential analog input range and the absolute voltage range on each pin are within the specified operating range defined in **Section 1.0 "Electrical Characteristics"** and **Section 4.0 "Description of Device Operation"**.

3.2 Supply Voltage (V_{DD}, V_{SS})

V_{DD} is the power supply pin for the device. This pin requires an appropriate bypass capacitor of about 0.1 µF (ceramic) to ground. An additional 10 µF capacitor (tantalum) in parallel is also recommended to further attenuate high frequency noise present in some application boards. The supply voltage (V_{DD}) must be maintained in the 2.7V to 5.5V range for specified operation.

V_{SS} is the ground pin and the current return path of the device. The user must connect the V_{SS} pin to a ground plane through a low impedance connection. If an analog ground path is available in the application PCB (printed circuit board), it is highly recommended that the V_{SS} pin be tied to the analog ground path or isolated within an analog ground plane of the circuit board.

3.3 Serial Clock Pin (SCL)

SCL is the serial clock pin of the I^2C interface. The MCP3421 acts only as a slave and the SCL pin accepts only external serial clocks. The input data from the Master device is shifted into the SDA pin on the rising edges of the SCL clock and output from the MCP3421 occurs at the falling edges of the SCL clock. The SCL pin is an open-drain N-channel driver. Therefore, it needs a pull-up resistor from the V_{DD} line to the SCL pin. Refer to **Section 5.3 "I^2C Serial Communications"** for more details of I^2C Serial Interface communication.

3.4 Serial Data Pin (SDA)

SDA is the serial data pin of the I^2C interface. The SDA pin is used for input and output data. In read mode, the conversion result is read from the SDA pin (output). In write mode, the device configuration bits are written (input) though the SDA pin. The SDA pin is an open-drain N-channel driver. Therefore, it needs a pull-up resistor from the V_{DD} line to the SDA pin. Except for start and stop conditions, the data on the SDA pin must be stable during the high period of the clock. The high or low state of the SDA pin can only change when the clock signal on the SCL pin is low. Refer to **Section 5.3 "I^2C Serial Communications"** for more details of I^2C Serial Interface communication.

4.0 DESCRIPTION OF DEVICE OPERATION

4.1 General Overview

The MCP3421 is a low-power, 18-Bit Delta-Sigma A/D converter with an I^2C serial interface. The device contains an on-board voltage reference (2.048V), programmable gain amplifier (PGA), and internal oscillator. The user can select 12, 14, 16, or 18 bit conversion by setting the configuration register bits. The device can be operated in Continuous Conversion or One-Shot Conversion mode. In the Continuous Conversion mode, the device converts the inputs continuously. While in the One-Shot Conversion mode, the device converts the input one time and stays in the low-power standby mode until it receives another command for a new conversion. During the standby mode, the device consumes less than 0.1 µA typical.

4.2 Power-On-Reset (POR)

The device contains an internal Power-On-Reset (POR) circuit that monitors power supply voltage (V_{DD}) during operation. This circuit ensures correct device start-up at system power-up and power-down events. The POR has built-in hysteresis and a timer to give a high degree of immunity to potential ripples and noises on the power supply. A 0.1 µF decoupling capacitor should be mounted as close as possible to the V_{DD} pin for additional transient immunity.

The threshold voltage is set at 2.2V with a tolerance of approximately ±5%. If the supply voltage falls below this threshold, the device will be held in a reset condition. The typical hysteresis value is approximately 200 mV.

The POR circuit is shut-down during the low-power standby mode. Once a power-up event has occurred, the device requires additional delay time (approximately 300 µs) before a conversion can take place. During this time, all internal analog circuitries are settled before the first conversion occurs. Figure 4-1 illustrates the conditions for power-up and power-down events under typical start-up conditions.

When the device powers up, it automatically resets and sets the configuration bits to default settings. The default configuration bit conditions are a PGA gain of 1 V/V and a conversion speed of 240 SPS in Continuous Conversion mode. When the device receives an I^2C General Call Reset command, it performs an internal reset similar to a Power-On-Reset event.

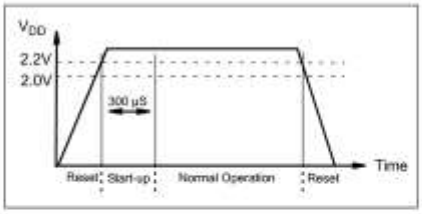


FIGURE 4-1: *POR Operation.*

4.3 Internal Voltage Reference

The device contains an on-board 2.048V voltage reference. This reference voltage is for internal use only and not directly measurable. The specifications of the reference voltage are part of the device's gain and drift specifications. Therefore, there is no separate specification for the on-board reference.

4.4 Analog Input Channel

The differential analog input channel has a switched capacitor structure. The internal sampling capacitor (3.2 pF) is charged and discharged to process a conversion. The charging and discharging of the input sampling capacitor creates dynamic input currents at the $V_{IN}+$ and $V_{IN}-$ input pins, which is inversely proportional to the internal sampling capacitor and internal frequency. The current is also a function of the differential input voltages. Care must be taken in setting the common-mode voltage and input voltage ranges so that the input limits do not exceed the ranges specified in **Section 1.0 "Electrical Characteristics"**.

4.5 Digital Output Code

The digital output code produced by the MCP3421 is a function of PGA gain, input signal, and internal reference voltage. In a fixed setting, the digital output code is proportional to the voltage difference between the two analog inputs.

The output data format is a binary two's complement. With this code scheme, the MSB can be considered a sign indicator. When the MSB is a logic '0', it indicates a positive value. When the MSB is a logic '1', it indicates a negative value. The following is an example of the output code:

(a) for a negative full-scale input voltage: 100...000

(b) for a zero differential input voltage: 000...000

(c) for a positive full-scale input voltage: 011...111.

The MSB is always transmitted first through the serial port. The number of data bits for each conversion is 18, 16, 14, or 12 bits depending on the conversion mode selection.

MCP3421

The output codes will not roll-over if the input voltage exceeds the maximum input range. In this case, the code will be locked at 0111...11 for all voltages greater than +(V_{REF} - 1 LSB) and 1000...00 for voltages less than -V_{REF}. Table 4-2 shows an example of output codes of various input levels using 18 bit conversion mode. Table 4-3 shows an example of minimum and maximum codes for each data rate option.

The output code is given by:

EQUATION 4-1:

$$Output\ Code = (Max\ Code + 1) \times \frac{(V_{IN}^{+} - V_{IN}^{-})}{2.048V}$$

The LSB of the code is given by:

EQUATION 4-2:

$$LSB = \frac{2 \times 2.048V}{2^N}$$

Where:

N = the number of bits

TABLE 4-1: LSB SIZE OF VARIOUS BIT CONVERSION MODES

Bit Resolutions	LSB (V)
12 bits	1 mV
14 bits	250 µV
16 bits	62.5 µV
18 bits	15.625 µV

TABLE 4-2: EXAMPLE OF OUTPUT CODE FOR 18 BITS

Input Voltage (V)	Digital Code
≥ V_{REF}	011111111111111111
V_{REF} - 1 LSB	011111111111111111
2 LSB	000000000000000010
1 LSB	000000000000000001
0	000000000000000000
-1 LSB	111111111111111111
-2 LSB	111111111111111110
- V_{REF}	100000000000000000
< -V_{REF}	100000000000000000

TABLE 4-3: MINIMUM AND MAXIMUM CODES

Number of Bits	Data Rate	Minimum Code	Maximum Code
12	240 SPS	-2048	2047
14	60 SPS	-8192	8191
16	15 SPS	-32768	32767
18	3.75 SPS	-131072	131071

Note: Maximum n-bit code = $2^{n-1} - 1$
Minimum n-bit code = $-1 \times 2^{n-1}$

4.6 Self-Calibration

The device performs a self-calibration of offset and gain for each conversion. This provides reliable conversion results from conversion-to-conversion over variations in temperature as well as power supply fluctuations.

4.7 Input Impedance

The MCP3421 uses a switched-capacitor input stage using a 3.2 pF sampling capacitor. This capacitor is switched (charged and discharged) at a rate of the sampling frequency that is generated by the on-board clock. The differential mode impedance varies with the PGA settings. The typical differential input impedance during a normal mode operation is given by:

$$Z_{IN}(f) = 2.25\ M\Omega/PGA$$

Since the sampling capacitor is only switching to the input pins during a conversion process, the above input impedance is only valid during conversion periods. In a low power standby mode, the above impedance is not presented at the input pins. Therefore, only a leakage current due to ESD diode is presented at the input pins.

The conversion accuracy can be affected by the input signal source impedance when any external circuit is connected to the input pins. The source impedance adds to the internal impedance and directly affects the time required to charge the internal sampling capacitor. Therefore, a large input source impedance connected to the input pins can increase the system performance errors such as offset, gain, and integral nonlinearity (INL) errors. Ideally, the input source impedance should be zero. This can be achievable by using an operational amplifier with a closed-loop output impedance of tens of ohms.

4.8 Aliasing and Anti-aliasing Filter

Aliasing occurs when the input signal contains time-varying signal components with frequency greater than half the sample rate. In the aliasing conditions, the device can output unexpected output codes. For applications that are operating in electrical noise environments, the time-varying signal noise or high frequency interference components can be easily added to the input signals and cause aliasing. Although the MCP3421 device has an internal first order sinc filter, its' filter response may not give enough attenuation to all aliasing signal components. To avoid the aliasing, an external anti-aliasing filter, which can be accomplished with a simple RC low-pass filter, is typically used at the input pins. The low-pass filter cuts off the high frequency noise components and provides a band-limited input signal to the MCP3421 input pins.

MCP3421

5.0 USING THE MCP3421 DEVICE

5.1 Operating Modes

The user operates the device by setting up the device configuration register and reads the conversion data using serial I^2C interface commands. The MCP3421 operates in two modes: (a) Continuous Conversion Mode or (b) One-Shot Conversion Mode (single conversion). The selection is made by setting the $\overline{O}/C$ bit in the Configuration Register. Refer to **Section 5.2 "Configuration Register"** for more information.

5.1.1 CONTINUOUS CONVERSION MODE ($\overline{O}/C$ BIT = 1)

The MCP3421 device performs a Continuous Conversion if the $\overline{O}/C$ bit is set to logic "high". Once the conversion is completed, the result is placed at the output data register. The device immediately begins another conversion and overwrites the output data register with the most recent data.

The device also clears the data ready flag ($\overline{RDY}$ bit = 0) when the conversion is completed. The device sets the ready flag bit ($\overline{RDY}$ bit = 1), if the latest conversion result has been read by the Master.

5.1.2 ONE-SHOT CONVERSION MODE ($\overline{O}/C$ BIT = 0)

Once the One-Shot Conversion (single conversion) Mode is selected, the device performs a conversion, updates the Output Data register, clears the data ready flag ($\overline{RDY}$ = 0), and then enters a low power standby mode. A new One-Shot Conversion is started again when the device receives a new write command with $\overline{RDY}$ = 1.

This One-Shot Conversion Mode is recommended for low power operating applications. During the low current standby mode, the device consumes less than 1 µA typical. For example, if user collects 18 bit conversion data once a second in One-Shot Conversion mode, the device draws only about one fourth of its total operating current. In this example, the device consumes approximately 39 µA (= ~145 µA/3.75 SPS), if the device performs only one conversion per second (1 SPS) in 18-bit conversion mode with 3V power supply.

5.2 Configuration Register

The MCP3421 has an 8-bit wide configuration register to select for: PGA gain, conversion rate, and conversion mode. This register allows the user to change the operating condition of the device and check the status of the device operation. The user can rewrite the configuration byte any time during the device operation. Register 5-1 shows the configuration register bits.

REGISTER 5-1: CONFIGURATION REGISTER

R/W-1	R/W-0	R/W-0	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0
$\overline{RDY}$	C1	C0	$\overline{O}$/C	S1	S0	G1	G0
1 *	0 *	0 *	1 *	0 *	0 *	0 *	0 *
bit 7							bit 0

* Default Configuration after Power-On Reset

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **$\overline{RDY}$:** Ready Bit

This bit is the data ready flag. In read mode, this bit indicates if the output register has been updated with a new conversion. In One-Shot Conversion mode, writing this bit to "1" initiates a new conversion.

Reading RDY bit with the read command:

1 = Output register has not been updated.
0 = Output register has been updated with the latest conversion data.

Writing $\overline{RDY}$ bit with the write command:

Continuous Conversion mode: No effect

One-Shot Conversion mode:
1 = Initiate a new conversion.
0 = No effect.

bit 6-5 **C1-C0:** Channel Selection Bits

These are the Channel Selection bits, but not used in the MCP3421 device.

bit 4 **$\overline{O}$/C:** Conversion Mode Bit

1 = Continuous Conversion Mode. Once this bit is selected, the device performs data conversions continuously.
0 = One-Shot Conversion Mode. The device performs a single conversion and enters a low power standby mode until it receives another write/read command.

bit 3-2 **S1-S0:** Sample Rate Selection Bit

00 = 240 SPS (12 bits),
01 = 60 SPS (14 bits),
10 = 15 SPS (16 bits),
11 = 3.75 SPS (18 bits)

bit 1-0 **G1-G0:** PGA Gain Selector Bits

00 = 1 V/V,
01 = 2 V/V,
10 = 4 V/V,
11 = 8 V/V

In read mode, the $\overline{RDY}$ bit in the configuration byte indicates the state of the conversion: (a) $\overline{RDY}$ = 1 indicates that the data bytes that have just been read were not updated from the previous conversion. (b) $\overline{RDY}$ = 0 indicates that the data bytes that have just been read were updated.

If the configuration byte is read repeatedly by clocking continuously after the first read (i.e., after the 5th byte in the 18-bit conversion mode), the state of the $\overline{RDY}$ bit indicates whether the device is ready with new conversion data. See Figure 5-2. For example, $\overline{RDY}$ = 0 means new conversion data is ready for reading. In this case, the user can send a stop bit to exit the current read operation and send a new read command to read out updated conversion data. See Figures 5-2 and 5-3 for reading conversion data. The user can rewrite the configuration byte any time for a new setting. Tables 5-1 and 5-2 show the examples of the configuration bit operation.

TABLE 5-1: CONFIGURATION BITS FOR WRITING

R/$\overline{W}$	O/$\overline{C}$	$\overline{RDY}$	Operation
0	0	0	No effect if all other bits remain the same - operation continues with the previous settings
0	0	1	Initiate One-Shot Conversion
0	1	0	Initiate Continuous Conversion
0	1	1	Initiate Continuous Conversion

TABLE 5-2: CONFIGURATION BITS FOR READING

R/$\overline{W}$	O/$\overline{C}$	$\overline{RDY}$	Operation
1	0	0	New conversion data in One-Shot conversion mode has been just read. The $\overline{RDY}$ bit remains low until set by a new write command.
1	0	1	One-Shot Conversion is in progress, The conversion data is not updated yet. The $\overline{RDY}$ bit stays high.
1	1	0	New conversion data in Continuous Conversion mode has been just read. The $\overline{RDY}$ bit changes to high after this read.
1	1	1	The conversion data in Continuous Conversion mode was already read. The latest conversion data is not ready. The $\overline{RDY}$ bit stays high until a new conversion is completed.

5.3 I²C Serial Communications

The MCP3421 device communicates with Master (microcontroller) through a serial I²C (Inter-Integrated Circuit) interface and supports standard (100 kbits/sec), fast (400 kbits/sec) and high-speed (3.4 Mbits/sec) modes. The serial I²C is a bidirectional 2-wire data bus communication protocol using open-drain SCL and SDA lines.

The MCP3421 can only be addressed as a slave. Once addressed, it can receive configuration bits or transmit the latest conversion results. The serial clock pin (SCL) is an input only and the serial data pin (SDA) is bidirectional. An example of a hardware connection diagram is shown in Figure 6-1.

The Master starts communication by sending a START bit and terminates the communication by sending a STOP bit. The first byte after the START bit is always the address byte of the device, which includes the device code, the address bits, and the R/$\overline{W}$ bit. The device code for the MCP3421 device is `1101`. The address bits (A2, A1, A0) are pre-programmed at the factory. In general, the address bits are specified by the customer when they order the device. The three address bits are programmed to "000" at the factory, if they are not specified by the customer. Figure 5-1 shows the details of the MCP3421 address byte.

During a low power standby mode, SDA and SCL pins remain at a floating condition.

More details of the I²C bus characteristic is described in **Section 5.6 "I²C Bus Characteristics"**.

5.3.1 DEVICE ADDRESSING

The address byte is the first byte received following the START condition from the Master device. The MCP3421 device code is `1101`. The device code is followed by three address bits (A2, A1, A0) which are programmed at the factory. The three address bits allow up to eight MCP3421 devices on the same data bus line. The (R/$\overline{W}$) bit determines if the Master device wants to read the conversion data or write to the Configuration register. If the (R/$\overline{W}$) bit is set (read mode), the MCP3421 outputs the conversion data in the following clocks. If the (R/$\overline{W}$) bit is cleared (write mode), the MCP3421 expects a configuration byte in the following clocks. When the MCP3421 receives the correct address byte, it outputs an acknowledge bit after the R/$\overline{W}$ bit. Figure 5-1 shows the MCP3421 address byte. See Figures 5-2 and 5-3 for the read and write operations of the device.

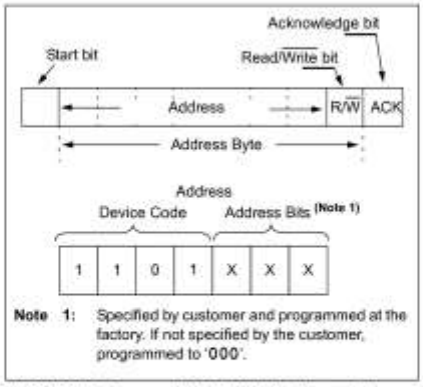


FIGURE 5-1: *MCP3421 Address Byte.*

5.3.2 READING DATA FROM THE DEVICE

When the Master sends a read command (R/$\overline{W}$ = 1), the MCP3421 outputs the conversion data bytes and configuration byte. Each byte consists of 8 bits with one acknowledge (ACK) bit. The ACK bit after the address byte is issued by the MCP3421 and the ACK bits after each conversion data bytes are issued by the Master.

When the device is configured for 18-bit conversion mode, the device outputs three data bytes followed by a configuration byte. The first 7 data bits in the first data byte are the MSB of the conversion data. The user can ignore the first 6 data bits, and take the 7th data bit (D17) as the MSB of the conversion data. The LSB of the 3rd data byte is the LSB of the conversion data (D0).

If the device is configured for 12, 14, or 16 bit-mode, the device outputs two data bytes followed by a configuration byte. In 16 bit-conversion mode, the MSB of the first data byte is the MSB (D15) of the conversion data. In 14-bit conversion mode, the first two bits in the first data byte can be ignored (they are the MSB of the conversion data), and the 3rd bit (D13) is the MSB of the conversion data. In 12-bit conversion mode, the first four bits can be ignored (they are the MSB of the conversion data), and the 5th bit (D11) of the byte represents the MSB of the conversion data. Table 5-3 shows an example of the conversion data output of each conversion mode.

The configuration byte follows the output data byte. The device outputs the configuration byte as long as the SCL pulses are received. The device terminates the current outputs when it receives a Not-Acknowledge (NAK), a repeated start or a stop bit at any time during the output bit stream. It is not required to read the configuration byte. However, the user may read the configuration byte to check the $\overline{RDY}$ bit condition to confirm whether the just received data bytes are updated conversion data. The user may continuously send clock (SCL) to repeatedly read the configuration bytes to check the $\overline{RDY}$ bit status.

Figures 5-2 and 5-3 show the timing diagrams of the reading.

5.3.3 WRITING A CONFIGURATION BYTE TO THE DEVICE

When the Master sends an address byte with the R/$\overline{W}$ bit low (R/$\overline{W}$ = 0), the MCP3421 expects one configuration byte following the address. Any byte sent after this second byte will be ignored. The user can change the operating mode of the device by writing the configuration register bits.

If the device receives a write command with a new configuration setting, the device immediately begins a new conversion and updates the conversion data.

TABLE 5-3: EXAMPLE OF CONVERSION DATA OUTPUT OF EACH CONVERSION MODE

Conversion Mode	Conversion Data Output
18-bits	MMMMMMMD16 (1st data byte) - D15 ~ D8 (2nd data byte) - D7 ~ D0 (3rd data byte) - Configuration byte
16-bits	MD14~D8 (1st data byte) - D7 ~ D0 (2nd data byte) - Configuration byte
14-bits	MMMD12~D8 (1st data byte) - D7 ~ D0 (2nd data byte) - Configuration byte
12-bits	MMMMMD10D9D8 (1st data byte) - D7 ~ D0 (2nd data byte) - Configuration byte
Note:	M is MSB of the data byte.

MCP3421

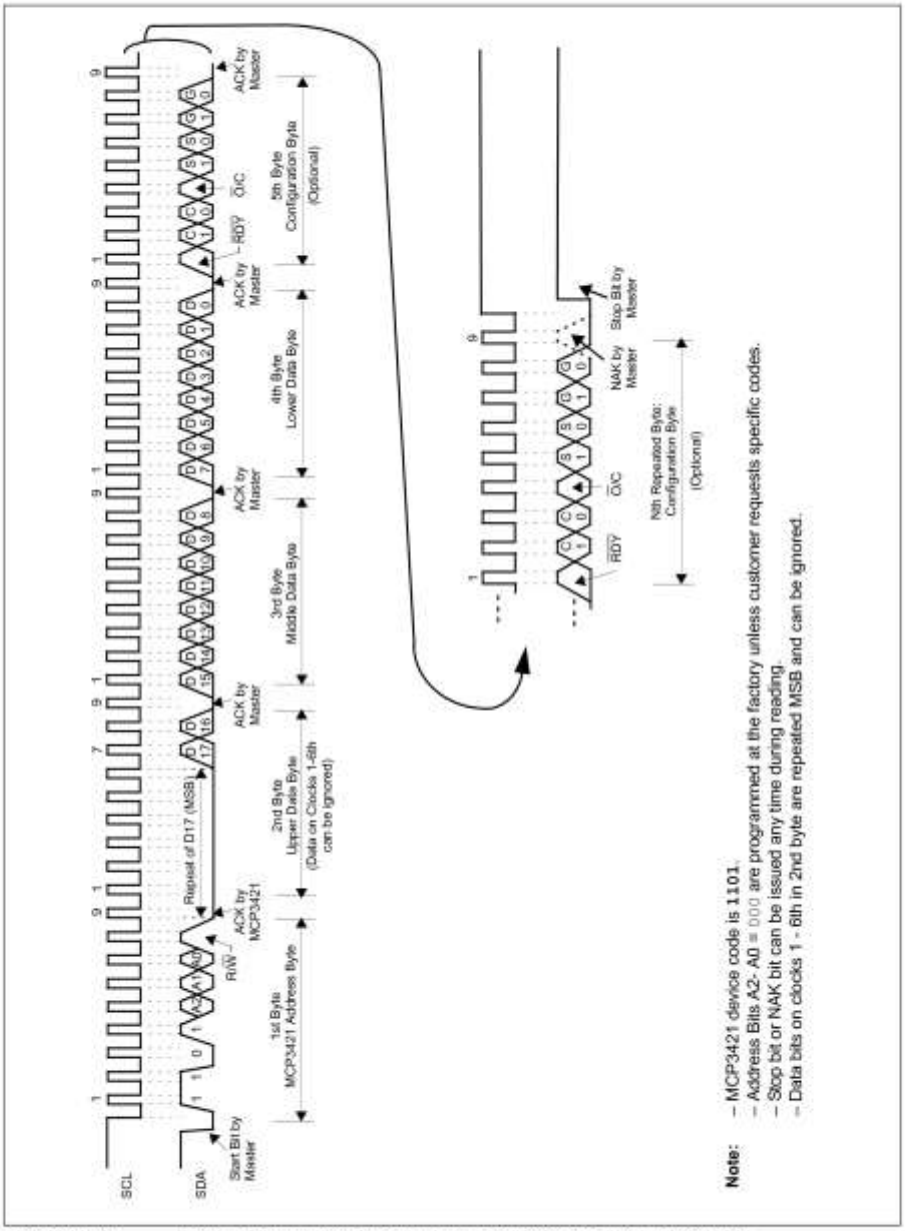


FIGURE 5-2: *Timing Diagram For Reading From The MCP3421 With 18-Bit Mode.*

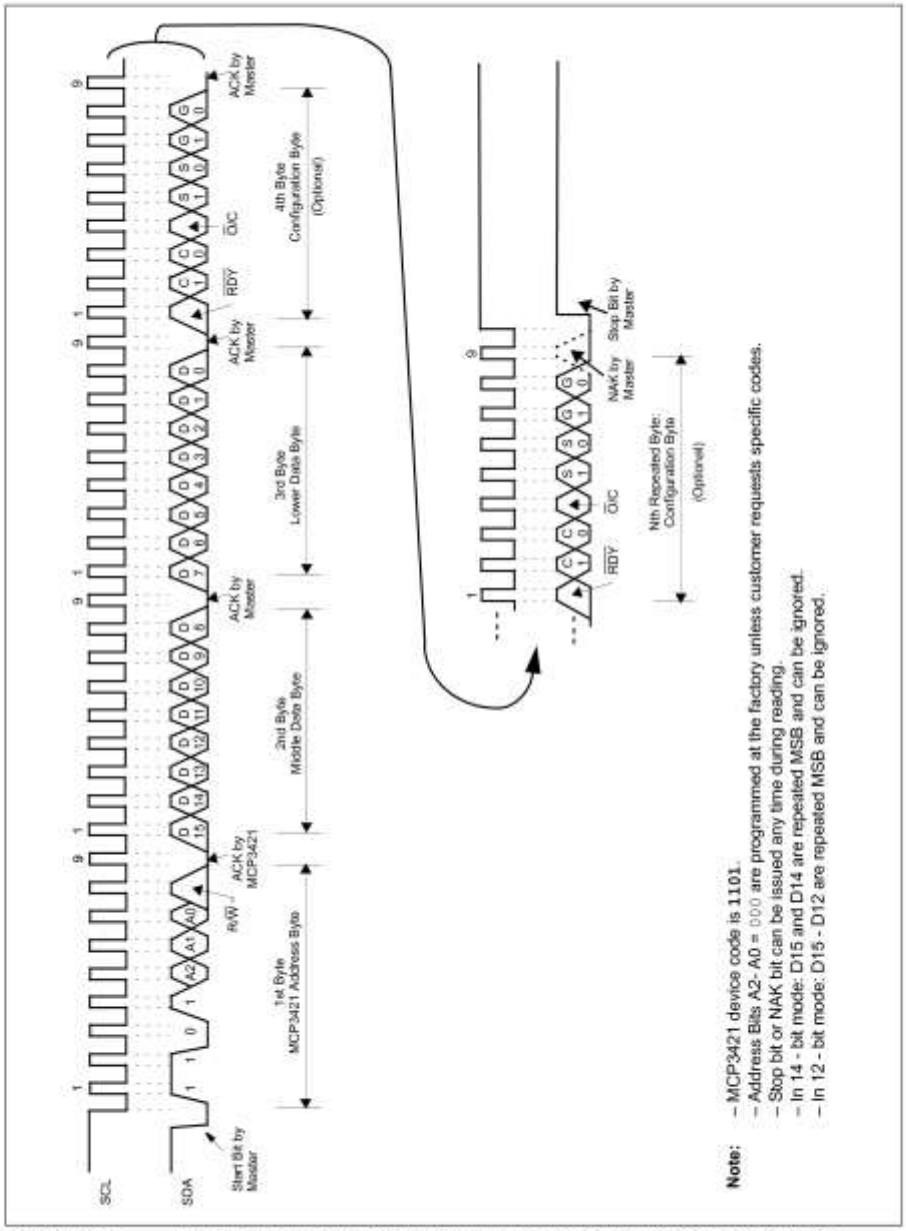


FIGURE 5-3: *Timing Diagram For Reading From The MCP3421 With 12-Bit to 16-Bit Modes.*

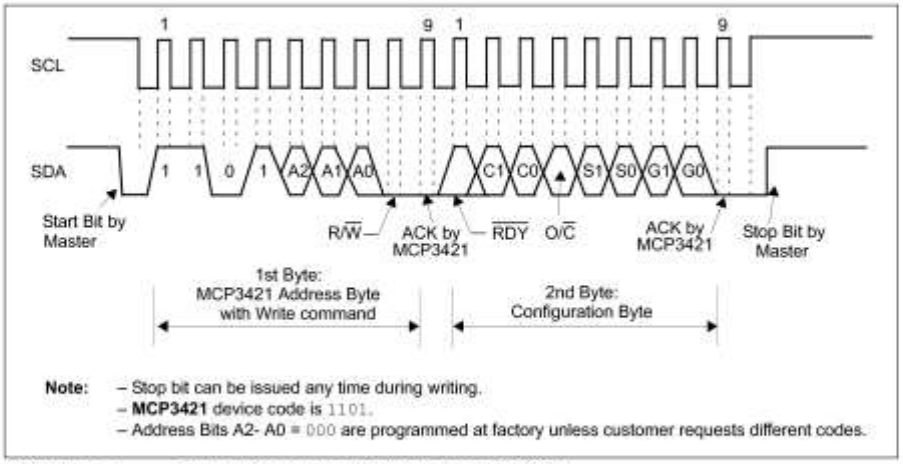


FIGURE 5-4: *Timing Diagram For Writing To The MCP3421.*

5.4 General Call

The MCP3421 acknowledges the general call address (0x00 in the first byte). The meaning of the general call address is always specified in the second byte. Refer to Figure 5-5. The MCP3421 supports the following general calls:

5.4.1 GENERAL CALL RESET

The general call reset occurs if the second byte is '00000110' (06h). At the acknowledgement of this byte, the device will abort current conversion and perform an internal reset similar to a power-on-reset (POR).

5.4.2 GENERAL CALL CONVERSION

The general call conversion occurs if the second byte is '00001000' (08h). All devices on the bus initiate a conversion simultaneously. For the MCP3421 device, the configuration will be set to the One-Shot Conversion mode and a single conversion will be performed. The PGA and data rate settings are unchanged with this general call.

Note: The I²C specification does not allow to use "00000000" (00h) in the second byte.

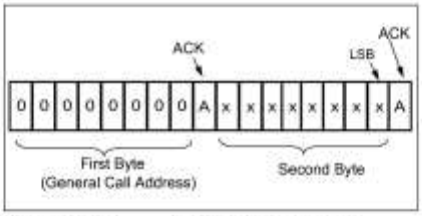


FIGURE 5-5: *General Call Address Format.*

For more information on the general call, or other I²C modes, please refer to the Phillips I²C specification.

5.5 High-Speed (HS) Mode

The I²C specification requires that a high-speed mode device must be 'activated' to operate in high-speed mode. This is done by sending a special address byte of 00001XXX following the START bit. The XXX bits are unique to the High-Speed (HS) mode Master. This byte is referred to as the High-Speed (HS) Master Mode Code (HSMMC). The MCP3421 device does not acknowledge this byte. However, upon receiving this code, the MCP3421 switches on its HS mode filters and communicates up to 3.4 MHz on SDA and SCL. The device will switch out of the HS mode on the next STOP condition.

For more information on the HS mode, or other I²C modes, please refer to the Phillips I²C specification.

5.6 I²C Bus Characteristics

The I²C specification defines the following bus protocol:

- Data transfer may be initiated only when the bus is not busy.
- During data transfer, the data line must remain stable whenever the clock line is HIGH. Changes in the data line while the clock line is HIGH will be interpreted as a START or STOP condition.

Accordingly, the following bus conditions have been defined using Figure 5-6.

5.6.1 BUS NOT BUSY (A)

Both data and clock lines remain HIGH.

5.6.2 START DATA TRANSFER (B)

A HIGH to LOW transition of the SDA line while the clock (SCL) is HIGH determines a START condition. All commands must be preceded by a START condition.

5.6.3 STOP DATA TRANSFER (C)

A LOW to HIGH transition of the SDA line while the clock (SCL) is HIGH determines a STOP condition. All operations can be ended with a STOP condition.

5.6.4 DATA VALID (D)

The state of the data line represents valid data when, after a START condition, the data line is stable for the duration of the HIGH period of the clock signal.

The data on the line must be changed during the LOW period of the clock signal. There is one clock pulse per bit of data.

Each data transfer is initiated with a START condition and terminated with a STOP condition.

5.6.5 ACKNOWLEDGE

The Master (microcontroller) and the slave (MCP3421) use an acknowledge pulse as a hand shake of communication for each byte. The ninth clock pulse of each byte is used for the acknowledgement. The acknowledgement is achieved by pulling-down the SDA line "LOW" during the 9th clock pulse. The clock pulse is always provided by the Master (microcontroller) and the acknowledgement is issued by the receiving device of the byte (Note: The transmitting device must release the SDA line ("HIGH") during the acknowledge pulse.). For example, the slave (MCP3421) issues the acknowledgement (bring down the SDA line "LOW") after the end of each receiving byte, and the master (microcontroller) issues the acknowledgement when it reads data from the Slave (MCP3421).

When the MCP3421 is addressed, it generates an acknowledge after receiving each byte successfully. The Master device (microcontroller) must provide an extra clock pulse (9th pulse of each byte) for the acknowledgement from the MCP3421 (slave).

The MCP3421 (slave) pulls-down the SDA line during the acknowledge clock pulse in such a way that the SDA line is stable low during the high period of the acknowledge clock pulse.

During reads, the Master (microcontroller) can terminate the current read operation by not providing an acknowledge bit on the last byte that has been clocked out from the MCP3421. In this case, the MCP3421 releases the SDA line to allow the master (microcontroller) to generate a STOP or repeated START condition.

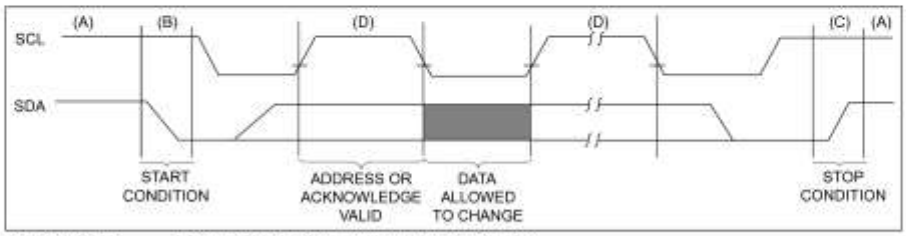


FIGURE 5-6: *Data Transfer Sequence on the Serial Bus.*

MCP3421

TABLE 5-4: I²C SERIAL TIMING SPECIFICATIONS

Electrical Specifications: Unless otherwise specified, all limits are specified for T_A = -40 to +85°C, V_{DD} = +2.7V, +3.3V or +5.0V, V_{SS} = 0V, $V_{IN}+ = V_{IN}- = V_{REF}/2$.

Parameters	Sym	Min	Typ	Max	Units	Conditions
Standard Mode						
Clock frequency	f_{SCL}	0	—	100	kHz	
Clock high time	T_{HIGH}	4000	—	—	ns	
Clock low time	T_{LOW}	4700	—	—	ns	
SDA and SCL rise time (Note 1)	T_R	—	—	1000	ns	From V_{IL} to V_{IH}
SDA and SCL fall time (Note 1)	T_F	—	—	300	ns	From V_{IH} to V_{IL}
START condition hold time	$T_{HD:STA}$	4000	—	—	ns	After this period, the first clock pulse is generated.
Repeated START condition setup time	$T_{SU:STA}$	4700	—	—	ns	Only relevant for repeated Start condition
Data hold time (Note 3)	$T_{HD:DAT}$	0	—	3450	ns	
Data input setup time	$T_{SU:DAT}$	250	—	—	ns	
STOP condition setup time	$T_{SU:STO}$	4000	—	—	ns	
STOP condition hold time	$T_{HD:STD}$	4000	—	—	ns	
Output valid from clock (Notes 2 and 3)	T_{AA}	0	—	3750	ns	
Bus free time	T_{BUF}	4700	—	—	ns	Time between START and STOP conditions.
Fast Mode						
Clock frequency	T_{SCL}	0	—	400	kHz	
Clock high time	T_{HIGH}	600	—	—	ns	
Clock low time	T_{LOW}	1300	—	—	ns	
SDA and SCL rise time (Note 1)	T_R	20 + 0.1Cb	—	300	ns	From V_{IL} to V_{IH}
SDA and SCL fall time (Note 1)	T_F	20 + 0.1Cb	—	300	ns	From V_{IH} to V_{IL}
START condition hold time	$T_{HD:STA}$	600	—	—	ns	After this period, the first clock pulse is generated
Repeated START condition setup time	$T_{SU:STA}$	600	—	—	ns	Only relevant for repeated Start condition
Data hold time (Note 4)	$T_{HD:DAT}$	0	—	900	ns	
Data input setup time	$T_{SU:DAT}$	100	—	—	ns	
STOP condition setup time	$T_{SU:STD}$	600	—	—	ns	
STOP condition hold time	$T_{HD:STO}$	600	—	—	ns	
Output valid from clock (Notes 2 and 3)	T_{AA}	0	—	1200	ns	
Bus free time	T_{BUF}	1300	—	—	ns	Time between START and STOP conditions.
Input filter spike suppression (Note 5)	T_{SP}	0	—	50	ns	SDA and SCL pins

Note 1: This parameter is ensured by characterization and not 100% tested.

2: This specification is not a part of the I²C specification. This specification is equivalent to the Data Hold Time ($T_{HD:DAT}$) plus SDA Fall (or rise) time: $T_{AA} = T_{HD:DAT} + T_F$ (OR T_R).

3: If this parameter is too short, it can create an unintended Start or Stop condition to other devices on the bus line. If this parameter is too long, Clock Low time (T_{LOW}) can be affected.

4: For Data Input: This parameter must be longer than t_{SP}. If this parameter is too long, the Data Input Setup ($T_{SU:DAT}$) or Clock Low time (T_{LOW}) can be affected.
For Data Output: This parameter is characterized, and tested indirectly by testing T_{AA} parameter.

5: This parameter is ensured by characterization and not 100% tested. This parameter is not available for Standard Mode.

TABLE 5-4: I²C SERIAL TIMING SPECIFICATIONS (CONTINUED)

Electrical Specifications: Unless otherwise specified, all limits are specified for T_A = -40 to +85°C, V_{DD} = +2.7V, +3.3V or +5.0V, V_{SS} = 0V, $V_{IN}^+ = V_{IN}^- = V_{REF}/2$.

Parameters	Sym	Min	Typ	Max	Units	Conditions
High Speed Mode						
Clock frequency	f_{SCL}	0 —	—	3.4 1.7	MHz MHz	C_b = 100 pF C_b = 400 pF
Clock high time	T_{HIGH}	60 120	—	—	ns ns	C_b = 100 pF C_b = 400 pF
Clock low time	T_{LOW}	160 320	—	—	ns	C_b = 100 pF C_b = 400 pF
SCL rise time (Note 1)	T_R	—	—	40 80	ns	From V_{IL} to V_{IH}, C_b = 100 pF C_b = 400 pF
SCL fall time (Note 1)	T_F	—	—	40 80	ns	From V_{IH} to V_{IL}, C_b = 100 pF C_b = 400 pF
SDA rise time (Note 1)	$T_{R:DAT}$	—	—	80 160	ns	From V_{IL} to V_{IH}, C_b = 100 pF C_b = 400 pF
SDA fall time (Note 1)	$T_{F:DATA}$	—	—	80 160	ns	From V_{IH} to V_{IL}, C_b = 100 pF C_b = 400 pF
START condition hold time	$T_{HD:STA}$	160	—	—	ns	After this period, the first clock pulse is generated
Repeated START condition setup time	$T_{SU:STA}$	160	—	—	ns	Only relevant for repeated Start condition
Data hold time (Note 4)	$T_{HD:DAT}$	0 0	—	70 150	ns	C_b = 100 pF C_b = 400 pF
Data input setup time	$T_{SU:DAT}$	10	—	—	ns	
STOP condition setup time	$T_{SU:STO}$	160	—	—	ns	
STOP condition hold time	$T_{HD:STD}$	160	—	—	ns	
Output valid from clock (Notes 2 and 3)	T_{AA}	—	—	150 310	ns	C_b = 100 pF C_b = 400 pF
Bus free time	T_{BUF}	160	—	—	ns	Time between START and STOP conditions.
Input filter spike suppression (Note 5)	T_{SP}	0	—	10	ns	SDA and SCL pins

Note 1: This parameter is ensured by characterization and not 100% tested.

2: This specification is not a part of the I²C specification. This specification is equivalent to the Data Hold Time ($T_{HD:DAT}$) plus SDA Fall (or rise) time: $T_{AA} = T_{HD:DAT} + T_F$ (OR T_R).

3: If this parameter is too short, it can create an unintended Start or Stop condition to other devices on the bus line. If this parameter is too long, Clock Low time (T_{LOW}) can be affected.

4: For Data Input: This parameter must be longer than t_{SP}. If this parameter is too long, the Data Input Setup ($T_{SU:DAT}$) or Clock Low time (T_{LOW}) can be affected.
For Data Output: This parameter is characterized, and tested indirectly by testing T_{AA} parameter.

5: This parameter is ensured by characterization and not 100% tested. This parameter is not available for Standard Mode.

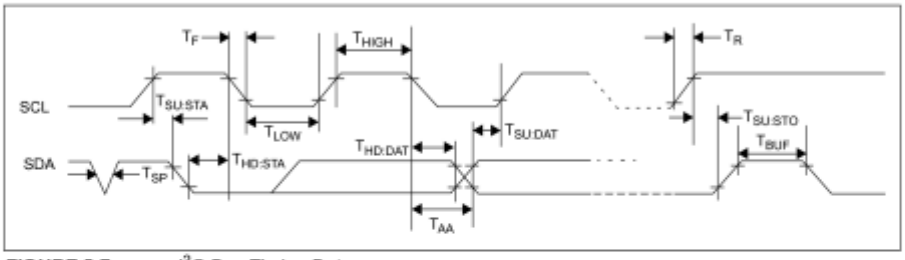


FIGURE 5-7: *I^2C Bus Timing Data.*

6.0 BASIC APPLICATION CONFIGURATION

The MCP3421 device can be used for various precision analog-to-digital converter applications. The device operates with very simple connections to the application circuit. The following sections discuss the examples of the device connections and applications.

6.1 Connecting to the Application Circuits

6.1.1 INPUT VOLTAGE RANGE

The fully differential input signals can be connected to the V_{IN}+ and V_{IN}- input pins. The input range should be within absolute common mode input voltage range: V_{SS} - 0.3V to V_{DD} + 0.3V. Outside this limit, the ESD protection diode at the input pin begins to conduct and the error due to input leakage current increases rapidly. Within this limit, the differential input V_{IN} (= V_{IN}+ - V_{IN}-) is boosted by the PGA before a conversion takes place. The MCP3421 can not accept negative input voltages on the input pins. Figures 6-1 and 6-2 show typical connection examples for differential inputs and a single-ended input, respectively. For the single-ended input, the input signal is applied to one of the input pins (typically connected to the V_{IN}+ pin) while the other input pin (typically V_{IN}- pin) is grounded. The input signal range of the single-ended configuration is from 0V to 2.048V. All device characteristics hold for the single-ended configuration, but this configuration loses one bit resolution because the input can only stand in positive half scale. Refer to **Section 1.0 "Electrical Characteristics"**.

6.1.2 BYPASS CAPACITORS ON V_{DD} PIN

For accurate measurement, the application circuit needs a clean supply voltage and must block any noise signal to the MCP3421 device. Figure 6-1 shows an example of using two bypass capacitors (a 10 µF tantalum capacitor and a 0.1 µF ceramic capacitor) in parallel on the V_{DD} line. These capacitors are helpful to filter out any high frequency noises on the V_{DD} line and also provide the momentary bursts of extra currents when the device needs from the supply. These capacitors should be placed as close to the V_{DD} pin as possible (within one inch). If the application circuit has separate digital and analog power supplies, the V_{DD} and V_{SS} of the MCP3421 should reside on the analog plane.

6.1.3 CONNECTING TO I²C BUS USING PULL-UP RESISTORS

The SCL and SDA pins of the MCP3421 are open-drain configurations. These pins require a pull-up resistor as shown in Figure 6-1. The value of these pull-up resistors depends on the operating speed (standard, fast, and high speed) and loading capacitance of the I²C bus line. Higher value of pull-up resistor consumes less power, but increases the signal transition time (higher RC time constant) on the bus. Therefore, it can limit the bus operating speed. The lower value of resistor, on the other hand, consumes higher power, but allows higher operating speed. If the bus line has higher capacitance due to long bus line or high number of devices connected to the bus, a smaller pull-up resistor is needed to compensate the long RC time constant. The pull-up resistor is typically chosen between 1 kΩ and 10 kΩ ranges for standard and fast modes, and less than 1 kΩ for high speed mode in high loading capacitance environments.

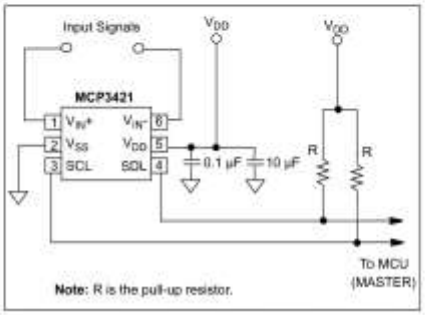


FIGURE 6-1: *Typical Connection Example for Differential Inputs.*

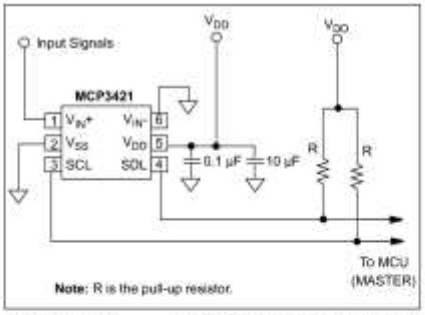


FIGURE 6-2: *Typical Connection Example for Single-Ended Input.*

The number of devices connected to the bus is limited only by the maximum bus capacitance of 400 pF. The bus loading capacitance affects on the bus operating speed. For example, the highest bus operating speed for the 400 pF bus capacitance is 1.7 MHz, and 3.4 MHz for 100 pF. Figure 6-3 shows an example of multiple device connections.

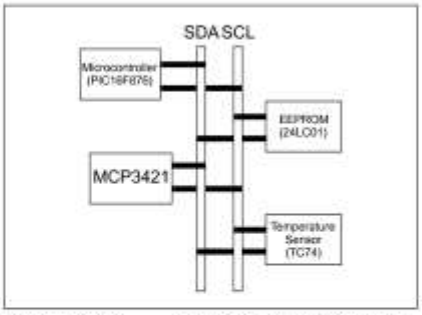


FIGURE 6-3: *Example of Multiple Device Connection on I^2C Bus.*

6.2 Device Connection Test

The user can test the presence of the MCP3421 on the I^2C bus line without performing an input data conversion. This test can be achieved by checking an acknowledge response from the MCP3421 after sending a read or write command. Here is an example using Figure 6-4:

(a) Set the R/$\overline{W}$ bit "HIGH" in the address byte.

(b) The MCP3421 will then acknowledge by pulling SDA bus LOW during the ACK clock and then release the bus back to the I^2C Master.

(c) A STOP or repeated START bit can then be issued from the Master and I^2C communication can continue.

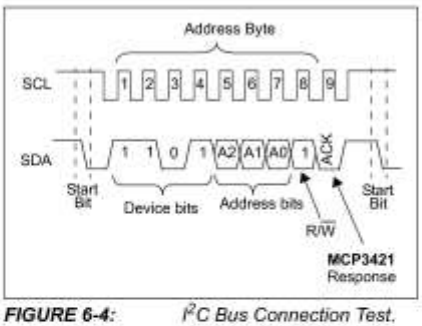


FIGURE 6-4: *I^2C Bus Connection Test.*

6.3 Application Examples

The MCP3421 device can be used in a broad range of sensor and data acquisition applications. Figure 6-5, shows an example of interfacing with a bridge sensor for pressure measurement.

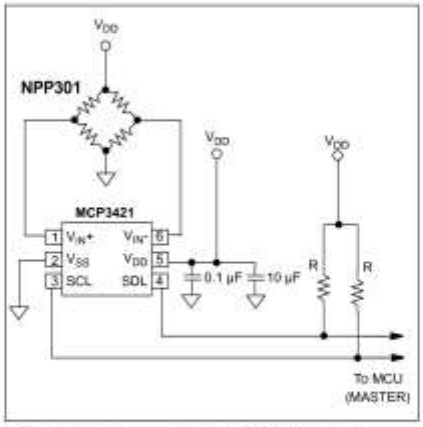


FIGURE 6-5: *Example of Pressure Measurement.*

In this circuit example, the sensor full scale range is ±7.5 mV with a common mode input voltage of V_{DD} / 2. This configuration will provide a full 14-bit resolution across the sensor output range. The alternative circuit for this amount of accuracy would involve an analog gain stage prior to a 16-bit ADC.

Figure 6-6 shows an example of temperature measurement using a thermistor. This example can achieve a linear response over a 50°C temperature range. This can be implemented using a standard resistor with 1% tolerance in series with the thermistor. The value of the resistor is selected to be equal to the thermistor value at the mid-point of the desired temperature range.

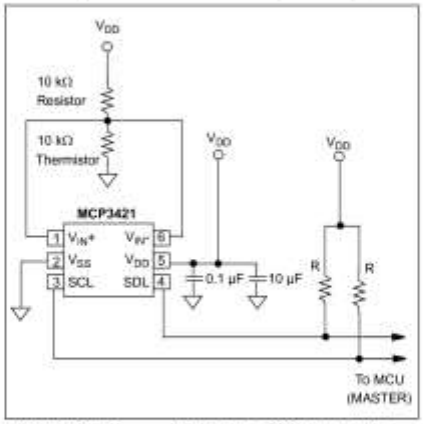
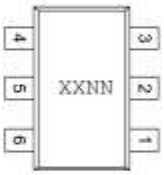


FIGURE 6-6: *Example of Temperature Measurement.*

7.0 PACKAGING INFORMATION

7.1 Package Marking Information

6-Lead SOT-23



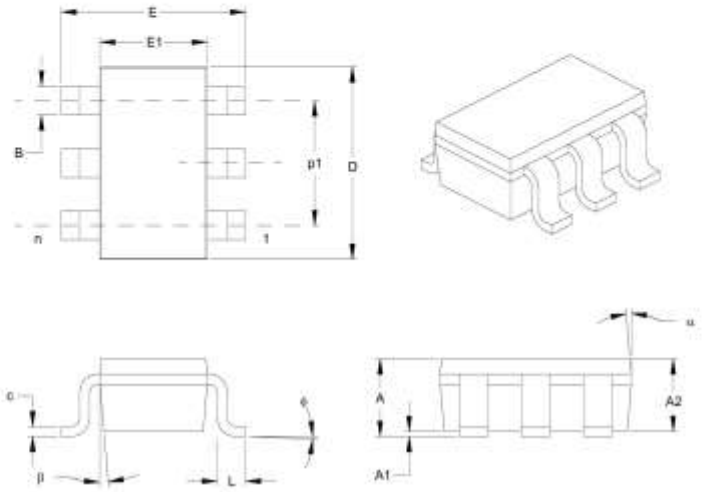
Example

Legend:	XX...X	Customer-specific information
	Y	Year code (last digit of calendar year)
	YY	Year code (last 2 digits of calendar year)
	WW	Week code (week of January 1 is week '01')
	NNN	Alphanumeric traceability code
	(e3)	Pb-free JEDEC designator for Matte Tin (Sn)
	*	This package is Pb-free. The Pb-free JEDEC designator (e3) can be found on the outer packaging for this package.
Note:	In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line, thus limiting the number of available characters for customer-specific information.	

MCP3421

6-Lead Plastic Small Outline Transistor (OT) (SOT-23)

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Units		INCHES*			MILLIMETERS		
Dimension Limits		MIN	NOM	MAX	MIN	NOM	MAX
Number of Pins	n		6			6	
Pitch	p		.038 BSC			0.95 BSC	
Outside lead pitch	p1		.075 BSC			1.90 BSC	
Overall Height	A	.035	.046	.057	0.90	1.18	1.45
Molded Package Thickness	A2	.035	.043	.051	0.90	1.10	1.30
Standoff	A1	.000	.003	.006	0.00	0.08	0.15
Overall Width	E	.102	.110	.118	2.60	2.80	3.00
Molded Package Width	E1	.059	.064	.069	1.50	1.63	1.75
Overall Length	D	.110	.116	.122	2.80	2.95	3.10
Foot Length	L	.014	.018	.022	0.35	0.45	0.55
Foot Angle	φ	0	5	10	0	5	10
Lead Thickness	c	.004	.006	.008	0.09	0.15	0.20
Lead Width	B	.014	.017	.020	0.35	0.43	0.50
Mold Draft Angle Top	α	0	5	10	0	5	10
Mold Draft Angle Bottom	β	0	5	10	0	5	10

* Controlling Parameter

Notes:

Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .005" (0.127mm) per side.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

See ASME Y14.5M

JEITA (formerly EIAJ) equivalent: SC-74A

Drawing No. C04-120

Revised 09-12-05

APPENDIX A: REVISION HISTORY

Revision B (December 2006)

- Changes to Electrical Characteristics tables
- Added characterization data
- Changes to I^2C Serial Timing Specification table
- Change to Figure 5-7.

Revision A (August 2006)

- Original Release of this Document.

NOTES:

PRODUCT IDENTIFICATION SYSTEM

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.

PART NO.	XX	X	/XX
Device	Address Options	Temperature Range	Package

Device: MCP3421T: Single Channel ΔΣ A/D Converter (Tape and Reel)

Address Options:

XX		A2	A1	A0
A0 *	=	0	0	0
A1	=	0	0	1
A2	=	0	1	0
A3	=	0	1	1
A4	=	1	0	0
A5	=	1	0	1
A6	=	1	1	0
A7	=	1	1	1

* Default option. Contact Microchip factory for other address options

Temperature Range: E = -40°C to +125°C

Package: OT = Plastic Small Outline Transistor (SOT-23-6), 6-lead

Examples:

a) MCP3421A0T-E/OT: Tape and Reel, Single Channel ΔΣ A/D Converter, SOT-23-6 package.

NOTES:

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

MICROCHIP

WORLDWIDE SALES AND SERVICE

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
http://support.microchip.com
Web Address:
www.microchip.com

Atlanta
Alpharetta, GA
Tel: 770-640-0034
Fax: 770-640-0307

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Farmington Hills, MI
Tel: 248-538-2250
Fax: 248-538-2260

Kokomo
Kokomo, IN
Tel: 765-864-8360
Fax: 765-864-8387

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608

Santa Clara
Santa Clara, CA
Tel: 408-961-6444
Fax: 408-961-6445

Toronto
Mississauga, Ontario,
Canada
Tel: 905-673-0699
Fax: 905-673-6509

ASIA/PACIFIC

Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon
Hong Kong
Tel: 852-2401-1200
Fax: 852-2401-3431

Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing
Tel: 86-10-8528-2100
Fax: 86-10-8528-2104

China - Chengdu
Tel: 86-28-8665-5511
Fax: 86-28-8665-7889

China - Fuzhou
Tel: 86-591-8750-3506
Fax: 86-591-8750-3521

China - Hong Kong SAR
Tel: 852-2401-1200
Fax: 852-2401-3431

China - Qingdao
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

China - Shanghai
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

China - Shenyang
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

China - Shenzhen
Tel: 86-755-8203-2660
Fax: 86-755-8203-1760

China - Shunde
Tel: 86-757-2839-5507
Fax: 86-757-2839-5571

China - Wuhan
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

China - Xian
Tel: 86-29-8833-7250
Fax: 86-29-8833-7256

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-4182-8400
Fax: 91-80-4182-8422

India - New Delhi
Tel: 91-11-4160-8631
Fax: 91-11-4160-8632

India - Pune
Tel: 91-20-2566-1512
Fax: 91-20-2566-1513

Japan - Yokohama
Tel: 81-45-471- 6166
Fax: 81-45-471-6122

Korea - Gumi
Tel: 82-54-473-4301
Fax: 82-54-473-4302

Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 or
82-2-558-5934

Malaysia - Penang
Tel: 60-4-646-8870
Fax: 60-4-646-5086

Philippines - Manila
Tel: 63-2-634-9065
Fax: 63-2-634-9069

Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

Taiwan - Hsin Chu
Tel: 886-3-572-9526
Fax: 886-3-572-6459

Taiwan - Kaohsiung
Tel: 886-7-536-4818
Fax: 886-7-536-4803

Taiwan - Taipei
Tel: 886-2-2500-6610
Fax: 886-2-2508-0102

Thailand - Bangkok
Tel: 66-2-694-1351
Fax: 66-2-694-1350

EUROPE

Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

UK - Wokingham
Tel: 44-118-921-5869
Fax: 44-118-921-5820

10/19/06

3.8. TRANSCEPTOR CAN

Product Folder | Order Now | Technical Documents | Tools & Software | Support & Community

TEXAS INSTRUMENTS

SN65HVD233, SN65HVD234, SN65HVD235

SLLS557H – NOVEMBER 2002 – REVISED NOVEMBER 2016

SN65HVD23x 3.3-V CAN Bus Transceivers

1 Features

- Single 3.3-V Supply Voltage
- Bus Pins Fault Protection Exceeds ±36 V
- Bus Pins ESD Protection Exceeds ±16 kV HBM
- Compatible With ISO 11898-2
- GIFT/ICT Compliant
- Data Rates up to 1 Mbps
- Extended –7 V to 12 V Common Mode Range
- High-Input Impedance Allows for 120 Nodes
- LVTTL I/Os are 5-V Tolerant
- Adjustable Driver Transition Times for Improved Emissions Performance
- Unpowered Node Does Not Disturb the Bus
- Low Current Standby Mode, 200-µA (Typical)
- SN65HVD233: Loopback Mode
- SN65HVD234: Ultra Low Current Sleep Mode
  - 50-nA Typical Current Consumption
- SN65HVD235: Autobaud Loopback Mode
- Thermal Shutdown Protection
- Power up and Down With Glitch-Free Bus Inputs and Outputs
  - High-Input Impedance With Low V_{CC}
  - Monolithic Output During Power Cycling

2 Applications

- Industrial Automation, Control, Sensors, and Drive Systems
- Motor and Robotic Control
- Building and Climate Control (HVAC)
- Backplane Communication and Control
- CAN Bus Standards such as CANopen, DeviceNet, CAN Kingdom, NMEA 2000, SAE J1939

3 Description

The SN65HVD233, SN65HVD234, and SN65HVD235 are used in applications employing the controller area network (CAN) serial communication physical layer in accordance with the ISO 11898 standard. As a CAN transceiver, each provides transmit and receive capability between the differential CAN bus and a CAN controller, with signaling rates up to 1 Mbps.

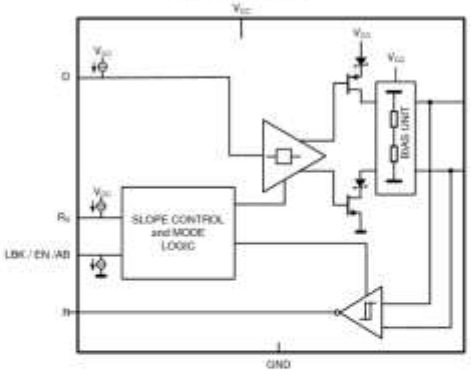
Designed for operation in especially harsh environments, the devices feature cross-wire protection, overvoltage protection up to ±36 V, loss of ground protection, overtemperature (thermal shutdown) protection, and common-mode transient protection of ±100 V. These devices operate over a wide –7 V to 12 V common-mode range. These transceivers are the interface between the host CAN controller on the microprocessor and the differential CAN bus used in industrial, building automation, transportation, and automotive applications.

Device Information(1)

PART NUMBER	PACKAGE	BODY SIZE (NOM)
SN65HVD233	SOIC (8)	4.90 mm × 3.91 mm
SN65HVD234		
SN65HVD235		

(1) For all available packages, see the orderable addendum at the end of the datasheet.

Block Diagram



An IMPORTANT NOTICE at the end of this data sheet addresses availability, warranty, changes, use in safety-critical applications, intellectual property matters and other important disclaimers. PRODUCTION DATA.

Table of Contents

4 Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from Revision G (January 2015) to Revision H — **Page**

- Deleted: "ISO 11783" from the last Application Bullet ... 1

Changes from Revision F (August 2008) to Revision G — **Page**

- Added *Pin Configuration and Functions* section, *ESD Ratings* table, *Feature Description* section, *Device Functional Modes*, *Application and Implementation* section, *Power Supply Recommendations* section, *Layout* section, *Device and Documentation Support* section, and *Mechanical, Packaging, and Orderable Information* section ... 1
- Changed the Functional Block Diagrams ... 4
- Added the THERMAL SHUTDOWN paragraph to the Application Information section ... 21
- Changed the BUS CABLE paragraph to BUS LOADING, LENGTH AND NUMBER OF NODES paragraph in the Application Information section ... 24
- Added the CAN TERMINATION paragraph to the Application Information section ... 24

Changes from Revision E (October 2007) to Revision F — **Page**

- Changed Figure 17, Receiver Test Circuit and Voltage Waveform. From: C_L = 50 pF ±20% to: C_L = 15 pF ±20% ... 13

Changes from Revision D (June 2005) to Revision E — **Page**

- Added 60-Ω load test condition to Figure 3 ... 10
- Deleted INTEROPERABILITY WITH 5-V CAN SYSTEMS section ... 26
- Added ISO 11898 COMPLIANCE OF SN65HVD230 FAMILY OF 3.3-V CAN TRANSCEIVERS section ... 26

Changes from Revision C (March 2005) to Revision D	Page
• Added Features Bullet: GIFT/ICT Compliant (SN65HVD234)	1

Changes from Revision B (June 2003) to Revision C	Page
• Added I_O, Receiver output current to the Abs Max Table	5

Changes from Revision A (March 2003) to Revision B	Page
• Changed the data sheet from Product Preview to Production for part number SN65HVD234 and SN65HVD235.	1
• Changed the APPLICATION INFORMATION section.	23

Changes from Original (November 2002) to Revision A	Page
• Changed the data sheet from Product Preview to Production for part number SN65HVD233	1

5 Description (continued)

Modes: The R_S pin (pin 8) of the SN65HVD233, SN65HVD234, and SN65HVD235 provides three modes of operation: high-speed, slope control, and low-power standby mode. The high-speed mode of operation is selected by connecting pin 8 directly to ground, allowing the driver output transistors to switch on and off as fast as possible with no limitation on the rise and fall slope. The rise and fall slope can be adjusted by connecting a resistor between the R_S pin and ground. The slope will be proportional to the pin's output current. With a resistor value of 10 kΩ the device driver will have a slew rate of ~15 V/µs and with a value of 100 kΩ the device will have ~2.0 V/µs slew rate. For more information about slope control, refer to *Feature Description*.

The SN65HVD233, SN65HVD234, and SN65HVD235 enter a low-current standby (listen only) mode during which the driver is switched off and the receiver remains active if a high logic level is applied to the R_S pin. If the local protocol controller needs to transmit a message to the bus it will have to return the device to either high-speed mode or slope control mode via the R_S pin.

Loopback (SN65HVD233): A logic high on the loopback (LBK) pin (pin 5) of the SN65HVD233 places the bus output and bus input in a high-impedance state. Internally, the D to R path of the device remains active and available for driver to receiver loopback that can be used for self-diagnostic node functions without disturbing the bus. For more information on the loopback mode, refer to *Feature Description*.

Ultra Low-Current Sleep (SN65HVD234): The SN65HVD234 enters an ultra low-current sleep mode in which both the driver and receiver circuits are deactivated if a low logic level is applied to EN pin (pin 5). The device remains in this sleep mode until the circuit is reactivated by applying a high logic level to pin 5.

Autobaud Loopback (SN65HVD235): The AB pin (pin 5) of the SN65HVD235 implements a bus listen-only loopback feature which allows the local node controller to synchronize its baud rate with that of the CAN bus. In autobaud mode, the bus output of the driver is placed in a high-impedance state while the bus input of the receiver remains active. There is an internal D pin to R pin loopback to assist the controller in baud rate detection, or the autobaud function. For more information on the autobaud mode, refer to *Feature Description*.

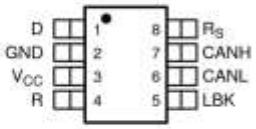
6 Device Comparison Table(1)

PART NUMBER	LOW POWER MODE	SLOPE CONTROL	DIAGNOSTIC LOOPBACK	AUTOBAUD LOOPBACK
SN65HVD233D	200-µA standby mode	Adjustable	Yes	No
SN65HVD234D	200-µA standby mode or 50-nA sleep mode	Adjustable	No	No
SN65HVD235D	200-µA standby mode	Adjustable	No	Yes

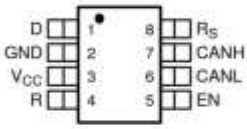
(1) For the most current package and ordering information, see *Mechanical, Packaging, and Orderable Information*, or see the TI web site at www.ti.com.

7 Pin Configuration and Functions

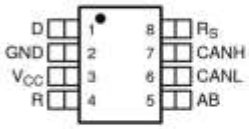
SN65HVD233D (Marked as VP233) (TOP VIEW)



SN65HVD234D (Marked as VP234) (TOP VIEW)



SN65HVD235D (Marked as VP235) (TOP VIEW)



Pin Functions

PIN NAME	PIN NO.	TYPE	DESCRIPTION
D	1	I	CAN transmit data input (LOW for dominant and HIGH for recessive bus states), also called TXD, driver input
GND	2	GND	Ground connection
V_{CC}	3	Supply	Transceiver 3.3-V supply voltage
R	4	O	CAN receive data output (LOW for dominant and HIGH for recessive bus states), also called RXD, receiver output
LBK	5	I	SN65HVD233: Loopback mode input pin
EN		I	SN65HVD234: Enable input pin. Logic high for enabling a normal mode (high speed or slope control) mode. Logic low for sleep mode.
AB		I	SN65HVD235: Autobaud loopback mode input pin
CANL	6	I/O	Low level CAN bus line
CANH	7	I/O	High level CAN bus line
R_S	8	I	Mode select pin: strong pulldown to GND = high speed mode, strong pullup to V_{CC} = low power mode, 10-kΩ to 100-kΩ pulldown to GND = slope control mode

8 Specifications

8.1 Absolute Maximum Ratings(1)(2)

over operating free-air temperature range unless otherwise noted

		MIN	MAX	UNIT
V_{CC}	Supply voltage	–0.3	7	V
	Voltage at any bus terminal (CANH or CANL)	–36	36	V
	Voltage input, transient pulse, CANH and CANL, through 100 Ω (see Figure 18)	–100	100	V
V_I	Input voltage, (D, R_S, EN, LBK, AB)	–0.5	7	V
V_O	Output voltage	–0.5	7	V
I_O	Receiver output current	–10	10	mA
	Continuous total power dissipation	See *Power Dissipation Ratings*		
T_J	Operating junction temperature		150	°C
T_{stg}	Storage temperature		125	°C

(1) Stresses beyond those listed under *Absolute Maximum Ratings* may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under *Recommended Operating Conditions* is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.
(2) All voltage values, except differential I/O bus voltages, are with respect to network ground pin.

8.2 ESD Ratings

				VALUE	UNIT
$V_{(ESD)}$	Electrostatic discharge	Human body model (HBM), per ANSI/ESDA/JEDEC JS-001[1]	CANH, CANL and GND	±16000	V
			All pins	3000	
		Charged-device model (CDM), per JEDEC specification JESD22-C101[2]		±1000	

(1) JEDEC document JEP155 states that 500-V HBM allows safe manufacturing with a standard ESD control process.
(2) JEDEC document JEP157 states that 250-V CDM allows safe manufacturing with a standard ESD control process.

8.3 Recommended Operating Conditions

			MIN	MAX	UNIT
V_{CC}	Supply voltage		3	3.6	V
	Voltage at any bus terminal (separately or common mode)		–7	12	
V_{IH}	High-level input voltage	D, EN, AB, LBK	2	5.5	
V_{IL}	Low-level input voltage	D, EN, AB, LBK	0	0.8	
V_{ID}	Differential input voltage between CANH and CANL		–6	6	
	Resistance from R_S to ground		0	100	kΩ
$V_{I(Rs)}$	Input Voltage at R_S for standby		0.75 V_{CC}	5.5	V
I_{OH}	High-level output current	Driver	–50		mA
		Receiver	–10		
I_{OL}	Low-level output current	Driver		50	mA
		Receiver		10	
T_J	Operating junction temperature	HVD233, HVD234, HVD235		150	°C
T_A	Operating free-air temperature[1]	HVD233, HVD234, HVD235	–40	125	°C

(1) Maximum free-air temperature operation is allowed as long as the device maximum junction temperature is not exceeded.

8.4 Thermal Information

over operating free-air temperature range (unless otherwise noted)

PARAMETERS		TEST CONDITIONS	VALUE	UNIT
$R_{\theta JA}$	Junction-to-ambient thermal resistance[1]	Low-K[2] board, no air flow	185	°C/W
		High-K[3] board, no air flow	101	
$R_{\theta JB}$	Junction-to-board thermal resistance	High-K[3] board, no air flow	82.8	°C/W
$R_{\theta JC}$	Junction-to-case thermal resistance		26.5	°C/W
$P_{(AVG)}$	Average power dissipation	R_L = 60 Ω, R_S at 0 V, input to D a 1-MHz 50% duty cycle square wave V_{CC} at 3.3 V, T_A = 25°C	36.4	mW
$T_{(SD)}$	Thermal shutdown junction temperature		170	°C

(1) See SZZA003 for an explanation of this parameter.
(2) JESD51-3 low effective thermal conductivity test board for leaded surface mount packages.
(3) JESD51-7 high effective thermal conductivity test board for leaded surface mount packages.

8.5 Power Dissipation Ratings

PACKAGE	CIRCUIT BOARD	$T_A \le 25°C$ POWER RATING	DERATING FACTOR[1] ABOVE T_A = 25°C	T_A = 85°C POWER RATING	T_A = 125°C POWER RATING
D	Low-K	596.6 mW	5.7 mW/°C	255.7 mW	28.4 mW
D	High-K	1076.9 mW	10.3 mW/°C	461.5 mW	51.3 mW

(1) This is the inverse of the junction-to-ambient thermal resistance when board-mounted and with no air flow.

8.6 Electrical Characteristics: Driver

over operating free-air temperature range (unless otherwise noted)

PARAMETER			TEST CONDITIONS	MIN	TYP(1)	MAX	UNIT
$V_{O(D)}$	Bus output voltage (Dominant)	CANH	D at 0 V, R_S at 0 V, See Figure 12 and Figure 13	2.45		V_{CC}	V
		CANL		0.5		1.25	
V_O	Bus output voltage (Recessive)	CANH	D at 3 V, R_S at 0 V, See Figure 12 and Figure 13		2.3		V
		CANL			2.3		
$V_{OD(D)}$	Differential output voltage (Dominant)		D at 0 V, R_S at 0 V, See Figure 12 and Figure 13	1.5	2	3	V
			D at 0 V, R_S at 0 V, See Figure 13 and Figure 14	1.2	2	3	
V_{OD}	Differential output voltage (Recessive)		D at 3 V, R_S at 0 V, See Figure 12 and Figure 13	−120		12	mV
			D at 3 V, R_S at 0 V, No Load	−0.5		0.05	V
$V_{OC(pp)}$	Peak-to-peak common-mode output voltage		See Figure 21		1		V
I_{IH}	High-level input current	D, EN, LBK, AB	D = 2 V or EN = 2 V or LBK = 2 V or AB = 2 V	−30		30	µA
I_{IL}	Low-level input current	D, EN, LBK, AB	D = 0.8 V or EN = 0.8 V or LBK = 0.8 V or AB = 0.8 V	−30		30	µA
I_{OS}	Short-circuit output current		V_{CANH} = −7 V, CANL Open, See Figure 26	−250			mA
			V_{CANH} = 12 V, CANL Open, See Figure 26			1	
			V_{CANL} = −7 V, CANH Open, See Figure 26	−1			
			V_{CANL} = 12 V, CANH Open, See Figure 26			250	
C_O	Output capacitance		See receiver input capacitance				
$I_{(Rs(s))}$	R_S input current for standby		R_S at 0.75 V_{CC}	−10			µA
I_{CC}	Supply current	Sleep	EN at 0 V, D at V_{CC}, R_S at 0 V or V_{CC}		0.05	2	µA
		Standby	R_S at V_{CC}, D at V_{CC}, AB at 0 V, LBK at 0 V, EN at V_{CC}		200	600	
		Dominant	D at 0 V, No Load, AB at 0 V, LBK at 0 V, R_S at 0 V, EN at V_{CC}			6	mA
		Recessive	D at V_{CC}, No Load, AB at 0 V, LBK at 0 V, R_S at 0 V, EN at V_{CC}			6	

(1) All typical values are at 25°C and with a 3.3-V supply.

8.7 Electrical Characteristics: Receiver

over operating free-air temperature range (unless otherwise noted)

PARAMETER			TEST CONDITIONS		MIN	TYP(1)	MAX	UNIT
V_{IT+}	Positive-going input threshold voltage		AB at 0 V, LBK at 0 V, EN at V_{CC}, See Table 1			750	900	mV
V_{IT-}	Negative-going input threshold voltage				500	650		
V_{hys}	Hysteresis voltage ($V_{IT+} - V_{IT-}$)					100		
V_{OH}	High-level output voltage		I_O = –4 mA, See Figure 17		2.4			V
V_{OL}	Low-level output voltage		I_O = 4 mA, See Figure 17				0.4	
I_I	Bus input current		CANH or CANL at 12 V	Other bus pin at 0 V, D at 3 V, AB at 0 V, LBK at 0 V, R_S at 0 V, EN at V_{CC}	150		500	µA
			CANH or CANL at 12 V, V_{CC} at 0 V		200		600	
			CANH or CANL at –7 V		–610		–150	
			CANH or CANL at –7 V, V_{CC} at 0 V		–450		–130	
C_I	Input capacitance (CANH or CANL)		Pin-to-ground, V_I = 0.4 sin (4E6πt) + 0.5 V, D at 3 V, AB at 0 V, LBK at 0 V, EN at V_{CC}			40		pF
C_{ID}	Differential input capacitance		Pin-to-pin, V_I = 0.4 sin (4E6πt) + 0.5 V, D at 3 V, AB at 0 V, LBK at 0 V, EN at V_{CC}			20		
R_{ID}	Differential input resistance		D at 3 V, AB at 0 V, LBK at 0 V, EN at V_{CC}		40		100	kΩ
R_{IN}	Input resistance (CANH or CANL) to ground				20		50	
I_{CC}	Supply current	Sleep	EN at 0 V, D at V_{CC}, R_S at 0 V or V_{CC}			0.05	2	µA
		Standby	R_S at V_{CC}, D at V_{CC}, AB at 0 V, LBK at 0 V, EN at V_{CC}			200	600	
		Dominant	D at 0 V, No Load, R_S at 0 V, LBK at 0 V, AB at 0 V, EN at V_{CC}				6	mA
		Recessive	D at V_{CC}, No Load, R_S at 0 V, LBK at 0 V, AB at 0 V, EN at V_{CC}				6	

(1) All typical values are at 25°C and with a 3.3-V supply.

8.8 Switching Characteristics: Driver

over operating free-air temperature range (unless otherwise noted)

PARAMETER		TEST CONDITIONS	MIN	TYP(1)	MAX	UNIT
t_{PLH}	Propagation delay time, low-to-high-level output	R_S at 0 V, See Figure 15		35	85	ns
		R_S with 10 kΩ to ground, See Figure 15		70	125	
		R_S with 100 kΩ to ground, See Figure 15		500	870	
t_{PHL}	Propagation delay time, high-to-low-level output	R_S at 0 V, See Figure 15		70	120	ns
		R_S with 10 kΩ to ground, See Figure 15		130	180	
		R_S with 100 kΩ to ground, See Figure 15		870	1200	
$t_{sk(p)}$	Pulse skew ($\lvert t_{PHL} - t_{PLH} \rvert$)	R_S at 0 V, See Figure 15		35		ns
		R_S with 10 kΩ to ground, See Figure 15		60		
		R_S with 100 kΩ to ground, See Figure 15		370		
t_r	Differential output signal rise time	R_S at 0 V, See Figure 15	20		70	ns
t_f	Differential output signal fall time		20		70	
t_r	Differential output signal rise time	R_S with 10 kΩ to ground, See Figure 15	30		135	ns
t_f	Differential output signal fall time		30		135	
t_r	Differential output signal rise time	R_S with 100 kΩ to ground, See Figure 15	350		1400	ns
t_f	Differential output signal fall time		350		1400	
$t_{en(S)}$	Enable time from standby to dominant	See Figure 19 and Figure 20		0.6	1.5	µs
$t_{en(Z)}$	Enable time from sleep to dominant			1	5	

(1) All typical values are at 25°C and with a 3.3-V supply.

8.9 Switching Characteristics: Receiver

over operating free-air temperature range (unless otherwise noted)

PARAMETER		TEST CONDITIONS	MIN	TYP(1)	MAX	UNIT
t_{PLH}	Propagation delay time, low-to-high-level output	See Figure 17		35	60	ns
t_{PHL}	Propagation delay time, high-to-low-level output			35	60	
$t_{sk(p)}$	Pulse skew ($\lvert t_{PHL} - t_{PLH} \rvert$)			7		
t_r	Output signal rise time			2	5	
t_f	Output signal fall time			2	5	

(1) All typical values are at 25°C and with a 3.3-V supply.

8.10 Switching Characteristics: Device

over operating free-air temperature range (unless otherwise noted)

PARAMETER			TEST CONDITIONS	MIN	TYP(1)	MAX	UNIT
$t_{(LBK)}$	Loopback delay, driver input to receiver output	HVD233	See Figure 23		7.5	12	ns
$t_{(AB1)}$	Loopback delay, driver input to receiver output	HVD235	See Figure 24		10	20	ns
$t_{(AB2)}$	Loopback delay, bus input to receiver output		See Figure 25		35	60	ns
$t_{(loop1)}$	Total loop delay, driver input to receiver output, recessive to dominant		R_S at 0 V, See Figure 22		70	135	ns
			R_S with 10 kΩ to ground, See Figure 22		105	190	
			R_S with 100 kΩ to ground, See Figure 22		535	1000	
$t_{(loop2)}$	Total loop delay, driver input to receiver output, dominant to recessive		R_S at 0 V, See Figure 22		70	135	ns
			R_S with 10 kΩ to ground, See Figure 22		105	190	
			R_S with 100 kΩ to ground, See Figure 22		535	1000	

(1) All typical values are at 25°C and with a 3.3-V supply.

8.11 Typical Characteristics

Rs, LBK, AB = 0 V; EN = V_{CC}

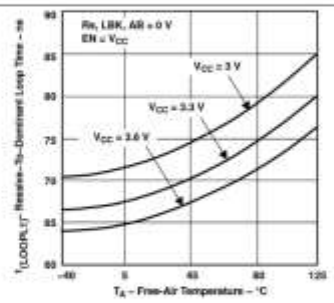


Figure 1. Recessive-to-Dominant Loop Time vs Free-Air Temperature

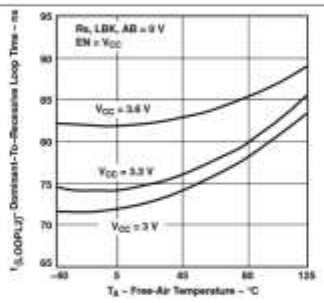
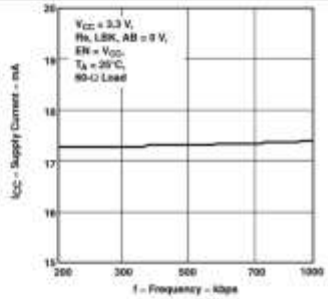
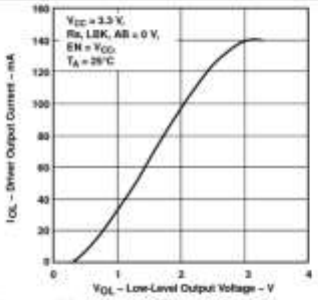


Figure 2. Dominant-to-Recessive Loop Time vs Free-Air Temperature



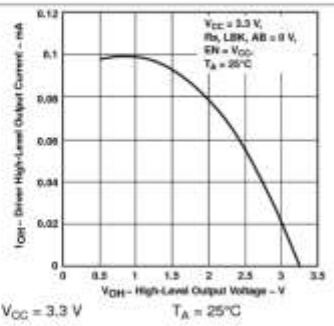
V_{CC} = 3.3 V T_A = 25°C 60-Ω Load

Figure 3. Supply Current vs Frequency



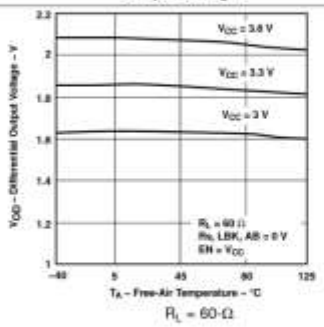
V_{CC} = 3.3 V T_A = 25°C

Figure 4. Driver Low-Level Output Current vs Low-Level Output Voltage



V_{CC} = 3.3 V T_A = 25°C

Figure 5. Driver High-Level Output Current vs High-Level Output Voltage

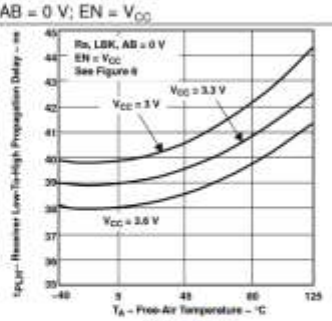


R_L = 60-Ω

Figure 6. Differential Output Voltage vs Free-Air Temperature

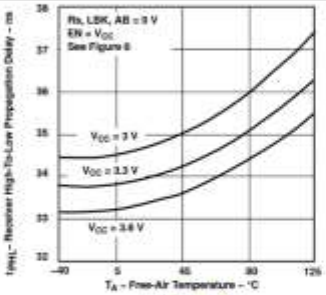
Typical Characteristics (continued)

R_S, LBK, AB = 0 V; EN = V_{CC}



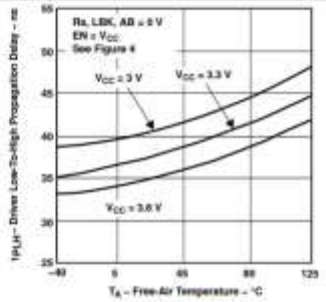
See Figure 3

Figure 7. Receiver Low-to-High Propagation Delay vs Free-Air Temperature



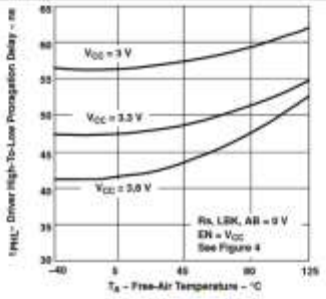
See Figure 3

Figure 8. Receiver High-to-Low Propagation Delay vs Free-Air Temperature



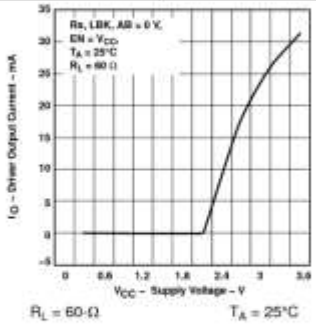
See Figure 1

Figure 9. Driver Low-to-High Propagation Delay vs Free-Air Temperature



See Figure 1

Figure 10. Driver High-to-Low Propagation Delay vs Free-Air Temperature



R_L = 60-Ω T_A = 25°C

Figure 11. Driver Output Current vs Supply Voltage

9 Parameter Measurement Information

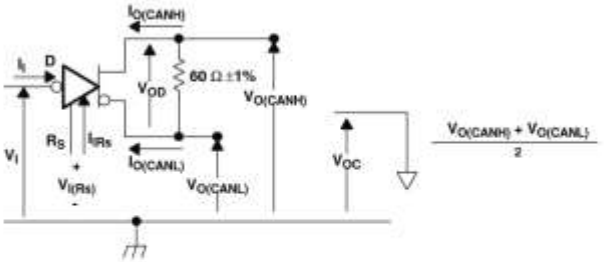


Figure 12. Driver Voltage, Current, and Test Definition

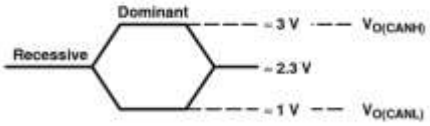


Figure 13. Bus Logic State Voltage Definitions

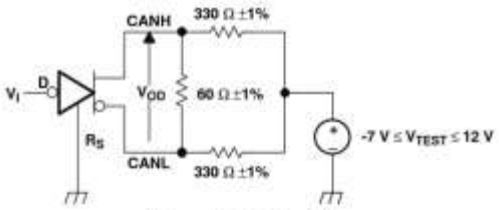
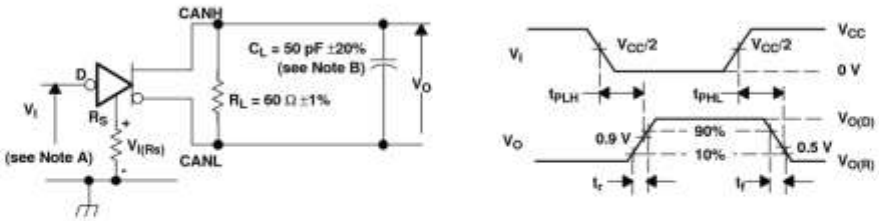


Figure 14. Driver V_{OD}



A. The input pulse is supplied by a generator having the following characteristics: Pulse repetition rate (PRR) ≤ 125 kHz, 50% duty cycle, t_r ≤ 6 ns, t_f ≤ 6 ns, Z_O = 50 Ω.

B. C_L includes fixture and instrumentation capacitance.

Figure 15. Driver Test Circuit and Voltage Waveforms

Parameter Measurement Information (continued)

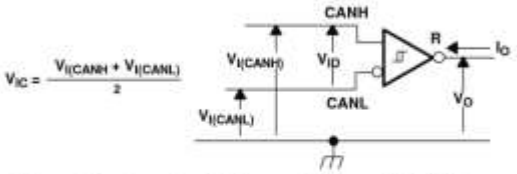
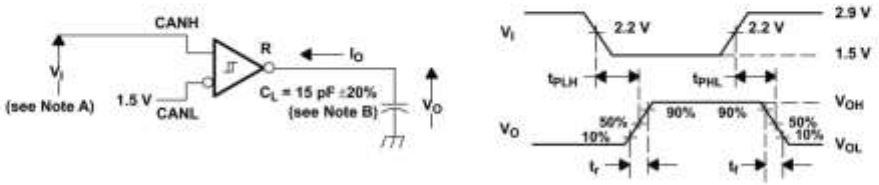


Figure 16. Receiver Voltage and Current Definitions



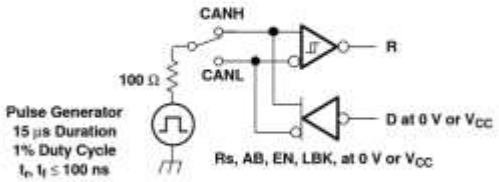
A. The input pulse is supplied by a generator having the following characteristics: Pulse repetition rate (PRR) ≤ 125 kHz, 50% duty cycle, t_r ≤ 6 ns, t_f ≤ 6 ns, Z_O = 50 Ω.

B. C_L includes fixture and instrumentation capacitance.

Figure 17. Receiver Test Circuit and Voltage Waveforms

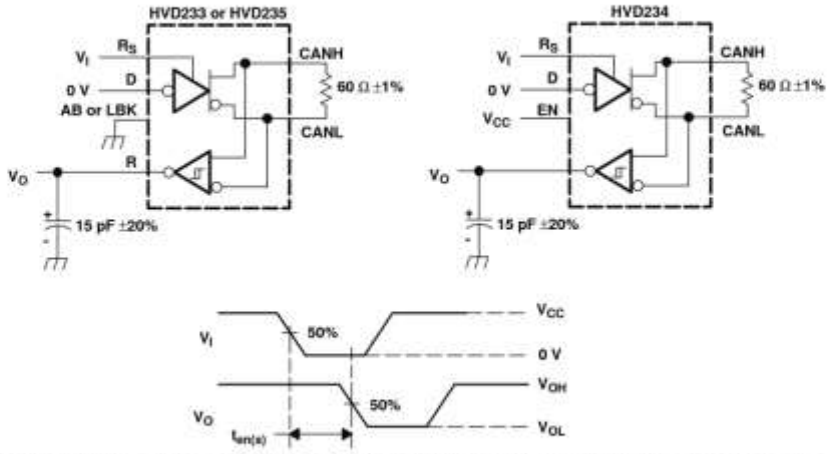
Table 1. Differential Input Voltage Threshold Test

INPUT		OUTPUT		MEASURED
V_{CANH}	V_{CANL}	R		$\|V_{ID}\|$
–6.1 V	–7 V	L	V_{OL}	900 mV
12 V	11.1 V	L		900 mV
–1 V	–7 V	L		6 V
12 V	6 V	L		6 V
–6.5 V	–7 V	H	V_{OH}	500 mV
12 V	11.5 V	H		500 mV
–7 V	–1 V	H		6 V
6 V	12 V	H		6 V
Open	Open	H		X



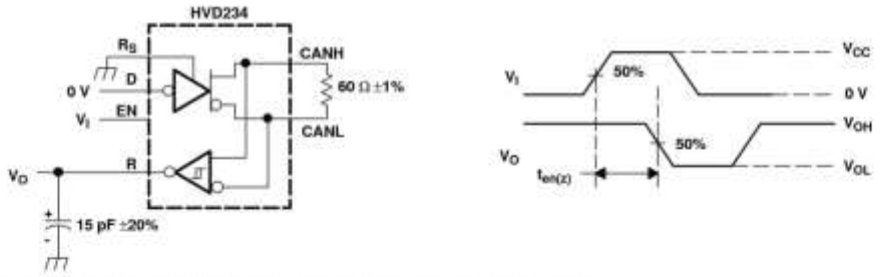
NOTE: This test is conducted to test survivability only. Data stability at the R output is not specified.

Figure 18. Test Circuit, Transient Overvoltage Test



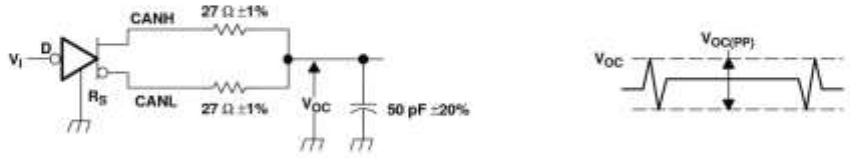
NOTE: All V_I input pulses are supplied by a generator having the following characteristics: t_r or $t_f \leq 6$ ns, pulse repetition rate (PRR) = 125 kHz, 50% duty cycle.

Figure 19. $T_{en(x)}$ Test Circuit and Voltage Waveforms



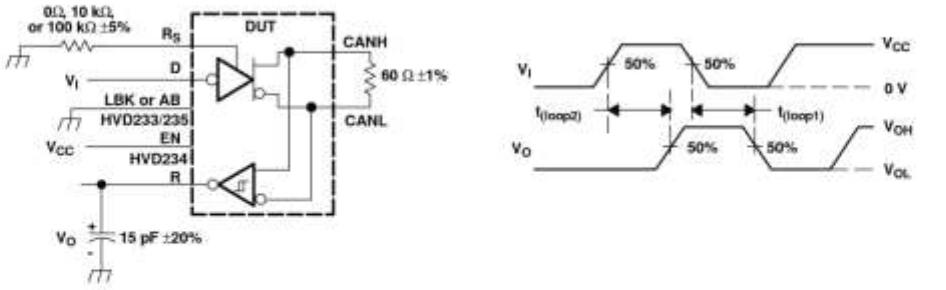
NOTE: All V_I input pulses are supplied by a generator having the following characteristics: t_r or $t_f \leq 6$ ns, pulse repetition rate (PRR) = 50 kHz, 50% duty cycle.

Figure 20. $T_{en(z)}$ Test Circuit and Voltage Waveforms



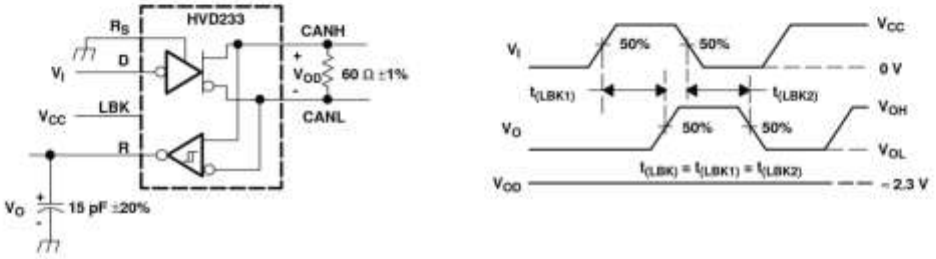
NOTE: All V_I input pulses are supplied by a generator having the following characteristics: t_r or $t_f \leq 6$ ns, pulse repetition rate (PRR) = 125 kHz, 50% duty cycle.

Figure 21. $V_{OC(pp)}$ Test Circuit and Voltage Waveforms



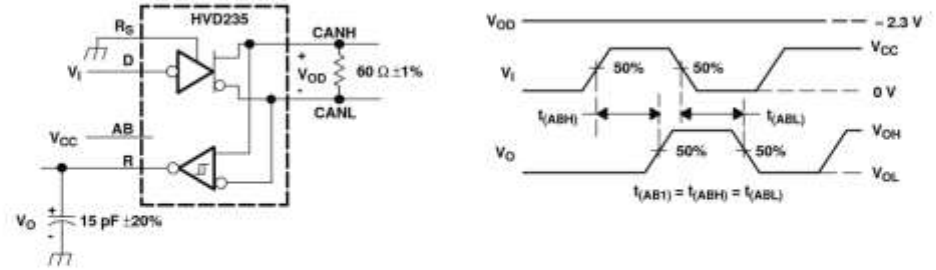
NOTE: All V_I input pulses are supplied by a generator having the following characteristics: t_r or $t_f \leq 6$ ns, pulse repetition rate (PRR) = 125 kHz, 50% duty cycle.

Figure 22. $T_{(loop)}$ Test Circuit and Voltage Waveforms



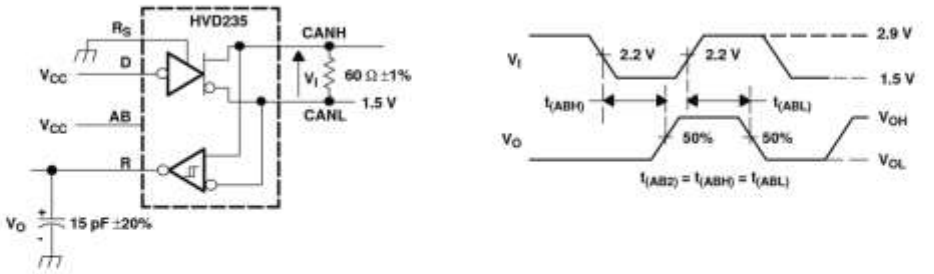
NOTE: All V_I input pulses are supplied by a generator having the following characteristics: t_r or $t_f \leq 6$ ns, pulse repetition rate (PRR) = 125 kHz, 50% duty cycle.

Figure 23. $T_{(LBK)}$ Test Circuit and Voltage Waveforms



NOTE: All V_I input pulses are supplied by a generator having the following characteristics: t_r or $t_f \leq 6$ ns, pulse repetition rate (PRR) = 125 kHz, 50% duty cycle.

Figure 24. $T_{(AB1)}$ Test Circuit and Voltage Waveforms



NOTE: All V_I input pulses are supplied by a generator having the following characteristics: t_r or $t_f \leq 6$ ns, pulse repetition rate (PRR) = 125 kHz, 50% duty cycle.

Figure 25. $T_{(AB2)}$ Test Circuit and Voltage Waveforms

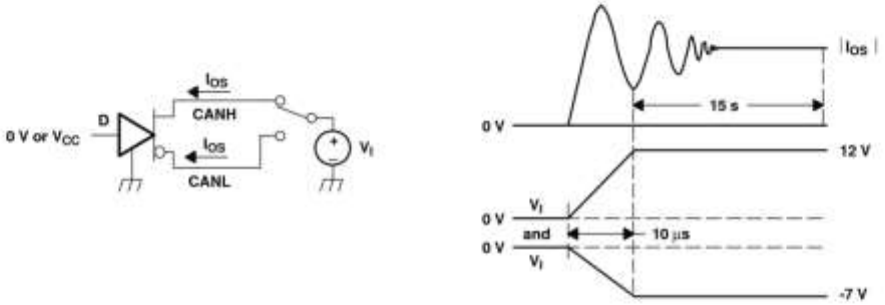
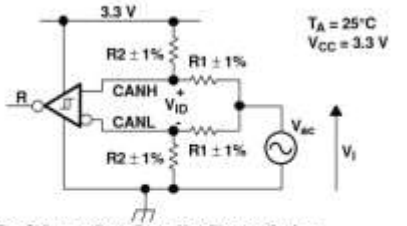
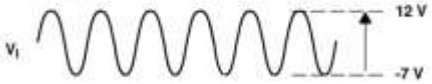


Figure 26. I_{OS} Test Circuit and Waveforms



The R Output State Does Not Change During Application of the Input Waveform.

V_{ID}	R1	R2
500 mV	50 Ω	280 Ω
900 mV	50 Ω	130 Ω



NOTE: All input pulses are supplied by a generator with f ≤ 1.5 MHz.

Figure 27. Common-Mode Voltage Rejection

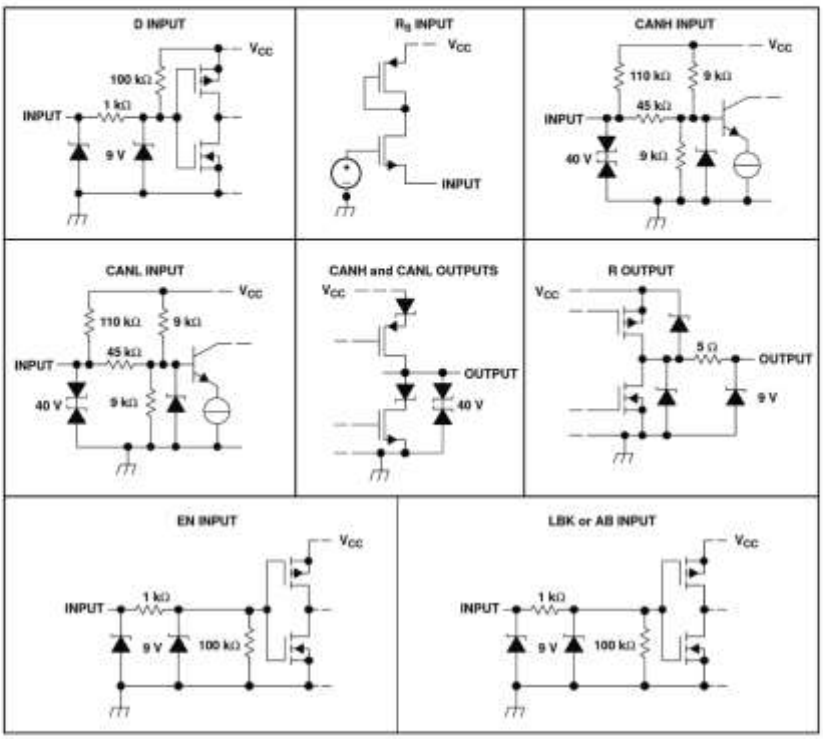


Figure 28. Equivalent Input and Output Schematic Diagrams

10 Detailed Description

10.1 Overview

This family of CAN transceivers is compatible with the ISO11898-2 High-Speed CAN (controller area network) physical layer standard. They are designed to interface between the differential bus lines in CAN and the CAN protocol controller at data rates up to 1 Mpbs.

10.2 Functional Block Diagrams

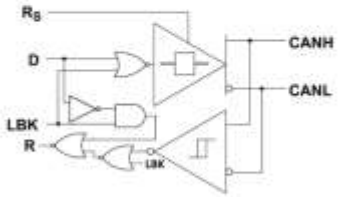


Figure 29. SN65HVD33 Functional Block Diagram

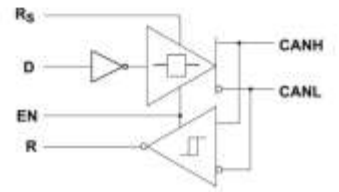


Figure 30. SN65HVD34 Functional Block Diagram

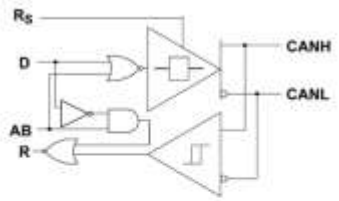


Figure 31. SN65HVD35 Functional Block Diagram

10.3 Feature Description

10.3.1 Diagnostic Loopback (SN65HVD233)

The diagnostic loopback or internal loopback function of the SN65HVD233 is enabled with a high-level input on pin 5, LBK. This mode disables the driver output while keeping the bus pins biased to the recessive state. This mode also redirects the D data input (transmit data) through logic to the received data output pin), thus creating an internal loopback of the transmit to receive data path. This mimics the loopback that occurs normally with a CAN transceiver because the receiver loops back the driven output to the R (receive data) pin. This mode allows the host protocol controller to input and read back a bit sequence or CAN messages to perform diagnostic routines without disturbing the CAN bus. A typical CAN bus application is displayed in Figure 36.

Feature Description (continued)

If the LBK pin is not used it may be tied to ground (GND). However, it is pulled low internally (defaults to a low-level input) and may be left open if not in use.

10.3.2 Autobaud Loopback (SN65HVD235)

The autobaud loopback mode of the SN65HVD235 is enabled by placing a high level input on pin 5, AB. In autobaud mode, the driver output is disabled, thus blocking the D pin to bus path and the bus transmit function of the transceiver. The bus pins remain biased to recessive. The receiver to R pin path or the bus receive function of the device remains operational, allowing bus activity to be monitored. In addition, the autobaud mode adds an internal logic loopback path from the D pin to R pin so the local node may transmit to itself in sync with bus traffic while not disturbing messages on the bus. Thus if the local node's CAN controller generates an error frame, it is not transmitted to the bus, but is detected only by the local CAN controller. This is especially helpful to determine if the local node is set to the same baud rate as the network, and if not adjust it to the network baud rate (autobaud detection).

Autobaud detection is best suited to applications that have a known selection of baud rates. For example, a popular industrial application has optional settings of 125 kbps, 250 kbps, or 500 kbps. Once the SN65HVD235 is placed into autobaud loopback mode the application software could assume the first baud rate of 125 kbps. It then waits for a message to be transmitted by another node on the bus. If the wrong baud rate has been selected, an error message is generated by the local CAN controller because the sample times will not be at the correct time. However, because the bus-transmit function of the device has been disabled, no other nodes receive the error frame generated by this node's local CAN controller.

The application would then make use of the status register indications of the local CAN controller for message received and error warning status to determine if the set baud rate is correct or not. The warning status indicates that the CAN controller error counters have been incremented. A message received status indicates that a good message has been received. If an error is generated, the application would then set the CAN controller with the next possibly valid baud rate, and wait to receive another message. This pattern is repeated until an error free message has been received, thus the correct baud rate has been selected. At this point the application would place the SN65HVD235 in a normal transmitting mode by setting pin 5 to a low-level, thus enabling bus-transmit and bus-receive functions to normal operating states for the transceiver.

If the AB pin is not used it may be tied to ground (GND). However, it is pulled low internally (defaults to a low-level input) and may be left open if not in use.

10.3.3 Slope Control

The rise and fall slope of the SN65HVD233, SN65HVD234, and SN65HVD235 driver output can be adjusted by connecting a resistor from the Rs (pin 8) to ground (GND), or to a low-level input voltage as shown in Figure 32.

The slope of the driver output signal is proportional to the pin's output current. This slope control is implemented with an external resistor value of 10 kΩ to achieve a ~15 V/µs slew rate, and up to 100 kΩ to achieve a ~2.0 V/µs slew rate . A typical slew rate verses pulldown resistance graph is shown in Figure 33. Typical driver output waveforms with slope control are displayed in Figure 39.

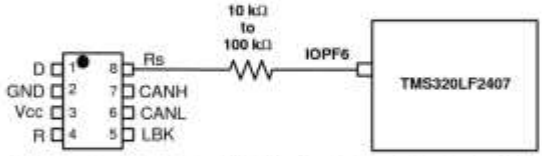


Figure 32. Slope Control/Standby Connection to a DSP

Feature Description (continued)

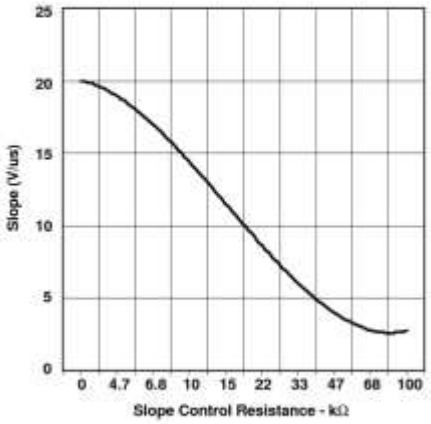


Figure 33. HVD233 Driver Output Signal Slope vs Slope Control Resistance Value

10.3.4 Standby

If a high-level input (> 0.75 V_{CC}) is applied to R_S (pin 8), the circuit enters a low-current, *listen only* standby mode during which the driver is switched off and the receiver remains active. If using this mode to save system power while waiting for bus traffic, the local controller can monitor the R output pin for a falling edge which indicates that a dominant signal was driven onto the CAN bus. The local controller can then drive the R_S pin low to return to slope control mode or high-speed mode.

10.3.5 Thermal Shutdown

If the junction temperature of the device exceeds the thermal shut down threshold the device turns off the CAN driver circuits thus blocking the D pin to bus transmission path. The shutdown condition is cleared when the junction temperature drops below the thermal shutdown temperature of the device. The CAN bus pins are high impedance biased to recessive level during a thermal shutdown, and the receiver to R pin path remains operational.

10.4 Device Functional Modes

10.4.1 Driver and Receiver

Table 2. Driver (SN65HVD233 or SN65HVD235)

INPUTS			OUTPUTS		
D	LBK/AB	R_s	CANH	CANL	BUS STATE
X	X	> 0.75 V_{CC}	Z	Z	Recessive
L	L or open	≤ 0.33 V_{CC}	H	L	Dominant
H or open	X		Z	Z	Recessive
X	H	≤ 0.33 V_{CC}	Z	Z	Recessive

Table 3. Receiver (SN65HVD233)

INPUTS				OUTPUT
BUS STATE	$V_{ID} = V_{(CANH)} - V_{(CANL)}$	LBK	D	R
Dominant	$V_{ID} \geq 0.9$ V	L or open	X	L
Recessive	$V_{ID} \leq 0.5$ V or open	L or open	H or open	H
?	0.5 V < V_{ID} <0.9 V	L or open	H or open	?
X	X	H	L	L
X	X		H	H

Table 4. Receiver (SN65HVD235)(1)

INPUTS				OUTPUT
BUS STATE	$V_{ID} = V_{(CANH)} - V_{(CANL)}$	AB	D	R
Dominant	$V_{ID} \geq 0.9$ V	L or open	X	L
Recessive	$V_{ID} \leq 0.5$ V or open	L or open	H or open	H
?	0.5 V < V_{ID} <0.9 V	L or open	H or open	?
Dominant	$V_{ID} \geq 0.9$ V	H	X	L
Recessive	$V_{ID} \leq 0.5$ V or open	H	H	H
Recessive	$V_{ID} \leq 0.5$ V or open	H	L	L
?	0.5 V < V_{ID} <0.9 V	H	L	L

(1) H = high level; L = low level; Z = high impedance; X = irrelevant; ? = indeterminate

Table 5. Driver (SN65HVD234)

INPUTS			OUTPUTS		
D	EN	R_s	CANH	CANL	BUS STATE
L	H	≤ 0.33 V_{CC}	H	L	Dominant
H	X	≤ 0.33 V_{CC}	Z	Z	Recessive
Open	X	X	Z	Z	Recessive
X	X	> 0.75 V_{CC}	Z	Z	Recessive
X	L or open	X	Z	Z	Recessive

Table 6. Receiver (SN65HVD234)(1)

INPUTS			OUTPUT
BUS STATE	$V_{ID} = V_{(CANH)} - V_{(CANL)}$	EN	R
Dominant	$V_{ID} \geq 0.9$ V	H	L
Recessive	$V_{ID} \leq 0.5$ V or open	H	H
?	0.5 V < V_{ID} <0.9 V	H	?
X	X	L or open	H

(1) H = high level; L = low level; Z = high impedance; X = irrelevant; ? = indeterminate

11 Application and Implementation

NOTE

Information in the following applications sections is not part of the TI component specification, and TI does not warrant its accuracy or completeness. TI's customers are responsible for determining suitability of components for their purposes. Customers should validate and test their design implementation to confirm system functionality.

11.1 Application Information

The CAN bus has two states during powered operation of the device; *dominant* and *recessive*. A dominant bus state is when the bus is driven differentially, corresponding to a logic low on the D and R pin. A recessive bus state is when the bus is biased to V_{CC} / 2 via the high-resistance internal resistors R_{IN} and R_{ID} of the receiver, corresponding to a logic high on the D and R pins. See *Figure 34* and *Figure 35*.

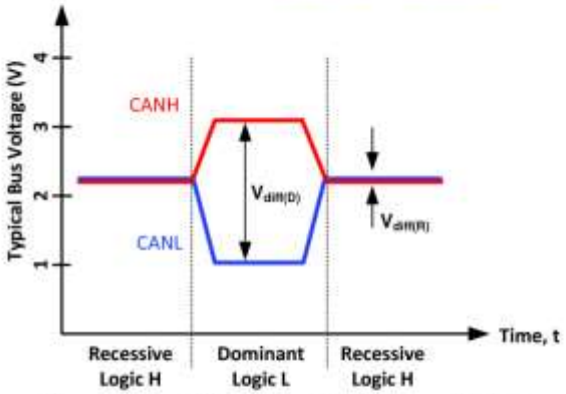


Figure 34. Bus States (Physical Bit Representation)

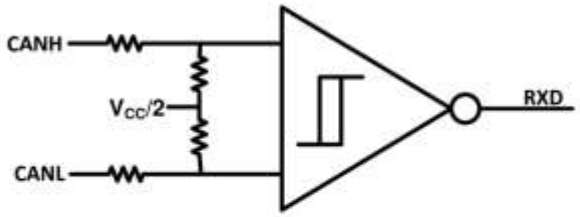


Figure 35. Simplified Recessive Common Mode Bias and Receiver

These CAN transceivers are typically used in applications with a host microprocessor or FPGA that includes the link layer portion of the CAN protocol. The different nodes on the network are typically connected through the use of a 120-Ω characteristic impedance twisted-pair cable with termination on both ends of the bus.

11.2 Typical Application

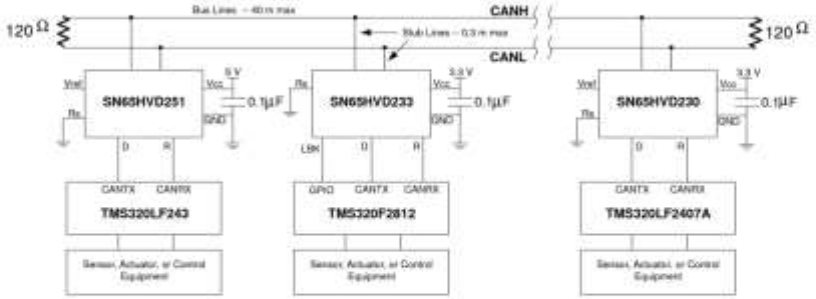


Figure 36. Typical HVD233 Application

11.2.1 Design Requirements

11.2.1.1 Bus Loading, Length and Number of Nodes

The ISO 11898 Standard specifies up to a data rate of 1 Mbps, maximum CAN bus cable length of 40 m, maximum drop line (stub) length of 0.3 m and a maximum of 30 nodes. However, with careful network design, the system may have longer cables, longer stub lengths, and many more nodes to a bus. Many CAN organizations and standards have scaled the use of CAN for applications outside the original ISO 11898 standard. They have made system level trade-offs for data rate, cable length, and parasitic loading of the bus. Examples of some of these specifications are ARINC825, CANopen, CAN Kingdom, DeviceNet and NMEA200.

A high number of nodes requires a transceiver with high input impedance and wide common mode range such as the SN65HVD23x CAN family. ISO 11898-2 specifies the driver differential output with a 60-Ω load (two 120-Ω termination resistors in parallel) and the differential output must be greater than 1.5 V. The SN65HVD23x devices are specified to meet the 1.5-V requirement with a 60-Ω load, and additionally specified with a differential output voltage minimum of 1.2 V across a common mode range of –2 V to 7 V through a 330-Ω coupling network. This network represents the bus loading of 120 SN65HVD23x transceivers based on their minimum differential input resistance of 40 kΩ. Therefore, the SN65HVD23x supports up to 120 transceivers on a single bus segment with margin to the 1.2-V minimum differential input voltage requirement at each node.

For CAN network design, margin must be given for signal loss across the system and cabling, parasitic loadings, network imbalances, ground offsets and signal integrity thus a practical maximum number of nodes may be lower. Bus length may also be extended beyond the original ISO 11898 standard of 40 m by careful system design and data rate tradeoffs. For example, CANopen network design guidelines allow the network to be up to 1 km with changes in the termination resistance, cabling, less than 64 nodes and significantly lowered data rate.

This flexibility in CAN network design is one of the key strengths of the various extensions and additional standards that have been built on the original ISO 11898 CAN standard.

11.2.1.2 CAN Termination

The ISO 11898 standard specifies the interconnect to be a twisted-pair cable (shielded or unshielded) with 120-Ω characteristic impedance (Z_O). Resistors equal to the characteristic impedance of the line should be used to terminate both ends of the cable to prevent signal reflections. Unterminated drop lines (stubs) connecting nodes to the bus should be kept as short as possible to minimize signal reflections. The termination may be on the cable or in a node, but if nodes may be removed from the bus the termination must be carefully placed so that it is not removed from the bus.

Typical Application (continued)

11.2.2 Detailed Design Procedure

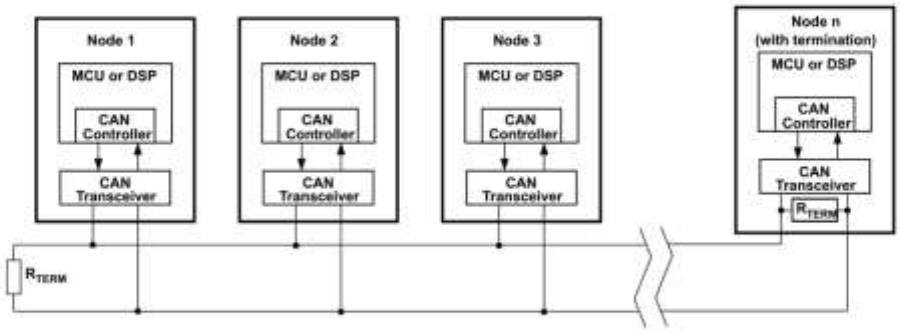


Figure 37. Typical CAN Bus

Termination is typically a 120-Ω resistor at each end of the bus. If filtering and stabilization of the common mode voltage of the bus is desired, then split termination may be used (see Figure 38). Split termination uses two 60-Ω resistors with a capacitor in the middle of these resistors to ground. Split termination improves the electromagnetic emissions behavior of the network by eliminating fluctuations in the bus common mode voltages at the start and end of message transmissions.

Care should be taken in the power ratings of the termination resistors used. Typically the worst case condition would be if the system power supply was shorted across the termination resistance to ground. In most cases the current flow through the resistor in this condition would be much higher than the transceiver's current limit.

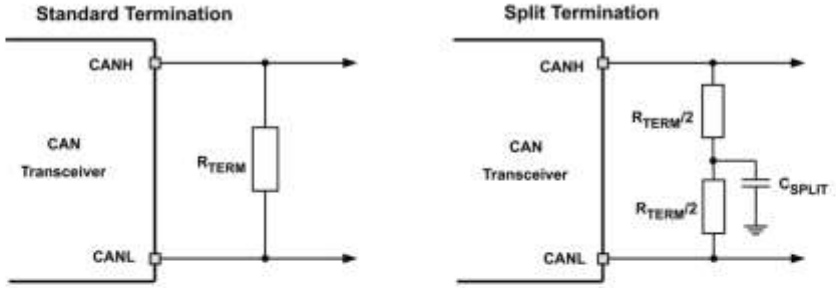


Figure 38. CAN Bus Termination Concepts

11.2.3 Application Curve

Figure 39 shows 3 typical output waveforms for the SN65HVD233 device with three different connections made to the R_S pin. The top waveform show the typical differential signal when transitioning from a recessive level to a dominant level on the CAN bus with R_S tied to GND through a 0-Ω resistor. The second waveform shows the same signal for the condition with a 10-kΩ resistor tied from R_S to ground. The bottom waveform shows the typical differential signal for the case where a 100-kΩ resistor is tied from the R_S pin to ground.

Typical Application (continued)

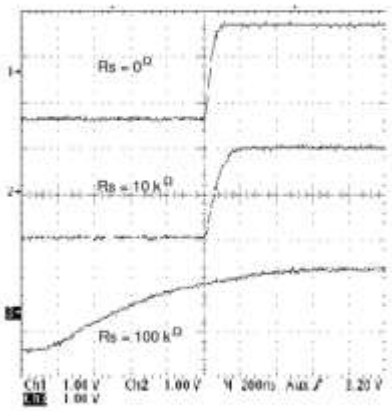


Figure 39. Typical SN65HVD233 Output Waveforms With Different Slope Control Resistor Values

11.3 System Example

11.3.1 ISO 11898 Compliance of SN65HVD23x Family of 3.3-V CAN Transceivers

11.3.1.1 Introduction

Many users value the low power consumption of operating their CAN transceivers from a 3.3-V supply. However, some are concerned about the interoperability with 5 V supplied transceivers on the same bus. This report analyzes this situation to address those concerns.

11.3.1.2 Differential Signal

CAN is a differential bus where complementary signals are sent over two wires and the voltage difference between the two wires defines the logical state of the bus. The differential CAN receiver monitors this voltage difference and outputs the bus state with a single ended logic level output signal.

System Example (continued)

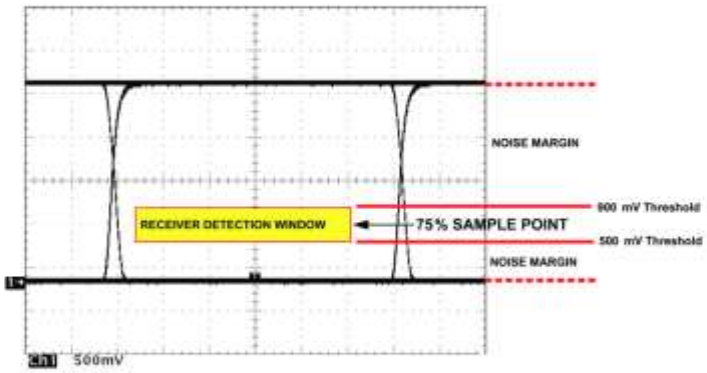


Figure 40. Typical SN65HVD230 Differential Output Voltage Waveform

The CAN driver creates the differential voltage between CANH and CANL in the dominant state. The dominant differential output of the SN65HVD23x is greater than 1.5 V and less than 3 V across a 60 ohm load as defined by the ISO 11898 standard. These are the same limiting values for 5 V supplied CAN transceivers. The bus termination resistors drive the recessive bus state and not the CAN driver.

A CAN receiver is required to output a recessive state when less than 500 mV of differential voltage exists on the bus, and a dominant state when more than 900 mV of differential voltage exists on the bus. The CAN receiver must do this with common-mode input voltages from –2 V to 7 V. The SN65HVD23x family receivers meet these same input specifications as 5 V supplied receivers.

11.3.1.3 Common-Mode Signal

A common-mode signal is an average voltage of the two signal wires that the differential receiver rejects. The common-mode signal comes from the CAN driver, ground noise, and coupled bus noise. Since the bias voltage of the recessive state of the device is dependent on V_{CC}, any noise present or variation of V_{CC} will have an effect on this bias voltage seen by the bus. The SN65HVD23x family has the recessive bias voltage set higher than $0.5*V_{CC}$ to comply with the ISO 11898-2 CAN standard. The caveat to this is that the common mode voltage will drop by a couple hundred millivolts when driving a dominant bit on the bus. This means that there is a common mode shift between the dominant bit and recessive bit states of the device. While this is not ideal, this small variation in the driver common-mode output is rejected by differential receivers and does not effect data, signal noise margins or error rates.

11.3.1.4 Interoperability of 3.3-V CAN in 5-V CAN Systems

The 3.3-V supplied SN65HVD23x family of CAN transceivers are fully compatible with 5-V CAN transceivers. The differential output voltage is the same, the recessive common mode output bias is the same, and the receivers have the same input specifications. The only slight difference is in the dominant common mode output voltage which is a couple hundred millivolts lower for 3.3-V CAN transceiver than 5-V supplied transceiver.

To help ensure the widest interoperability possible, the SN65HVD23x family has successfully passed the internationally recognized GIFT/ICT conformance and interoperability testing for CAN transceivers. Electrical interoperability does not always assure interchangeability however. Most implementers of CAN buses recognize that ISO 11898 does not sufficiently specify the electrical layer and that strict standard compliance alone does not ensure full interchangeability. This comes only with thorough equipment testing.

12 Power Supply Recommendations

To ensure reliable operation at all data rates and supply voltages, each supply should be decoupled with a 100-nF ceramic capacitor located as close to the V_{CC} supply pins as possible. The TPS76333 is a linear voltage regulator suitable for the 3.3 V supply.

13 Layout

13.1 Layout Guidelines

In order for the PCB design to be successful, start with design of the protection and filtering circuitry. Because ESD and EFT transients have a wide frequency bandwidth from approximately 3 MHz to 3 GHz, high frequency layout techniques must be applied during PCB design. On chip IEC ESD protection is good for laboratory and portable equipment but is usually not sufficient for EFT and surge transients occurring in industrial environments. Therefore robust and reliable bus node design requires the use of external transient protection devices at the bus connectors. Placement at the connector also prevents these harsh transient events from propagating further into the PCB and system.

Use V_{CC} and ground planes to provide low inductance.

NOTE

High frequency current follows the path of least inductance and not the path of least resistance.

Design the bus protection components in the direction of the signal path. Do not force the transient current to divert from the signal path to reach the protection device.

An example placement of the Transient Voltage Suppression (TVS) device indicated as D1 (either bi-directional diode or varistor solution) and bus filter capacitors C8 and C9 are shown in Figure 41.

The bus transient protection and filtering components should be placed as close to the bus connector, J1, as possible. This prevents transients, ESD and noise from penetrating onto the board and disturbing other devices.

Bus termination: Figure 41 shows split termination. This is where the termination is split into two resistors, R5 and R6, with the center or split tap of the termination connected to ground via capacitor C7. Split termination provides common mode filtering for the bus. When termination is placed on the board instead of directly on the bus, care must be taken to ensure the terminating node is not removed from the bus as this will cause signal integrity issues of the bus is not properly terminated on both ends. See the application section for information on power ratings needed for the termination resistor(s).

Bypass and bulk capacitors should be placed as close as possible to the supply pins of transceiver, examples C2, C3 (V_{CC}).

Use at least two vias for V_{CC} and ground connections of bypass capacitors and protection devices to minimize trace and via inductance.

To limit current of digital lines, serial resistors may be used. Examples are R1, R2, R3 and R4.

To filter noise on the digital IO lines, a capacitor may be used close to the input side of the IO as shown by C1 and C4.

Since the internal pullup and pulldown biasing of the device is weak for floating pins, an external 1-kΩ to 10-kΩ pullup or pulldown resistor should be used to bias the state of the pin more strongly against noise during transient events.

Pin 1: If an open drain host processor is used to drive the D pin of the device an external pull-up resistor between 1 kΩ and 10 kΩ and V_{CC} should be used to drive the recessive input state of the device.

Pin 8: is shown assuming the mode pin, RS, will be used. If the device will only be used in normal mode or slope control mode, R3 is not needed and the pads of C4 could be used for the pulldown resistor to GND.

13.2 Layout Example

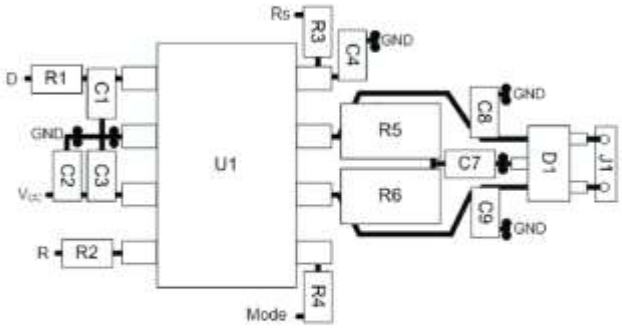


Figure 41. Layout Example Schematic

14 Device and Documentation Support

14.1 Related Links

The table below lists quick access links. Categories include technical documents, support and community resources, tools and software, and quick access to sample or buy.

Table 7. Related Links

PARTS	PRODUCT FOLDER	SAMPLE & BUY	TECHNICAL DOCUMENTS	TOOLS & SOFTWARE	SUPPORT & COMMUNITY
SN65HVD233	Click here	Click here	Click here	Click here	Click here
SN65HVD234	Click here	Click here	Click here	Click here	Click here
SN65HVD235	Click here	Click here	Click here	Click here	Click here

14.2 Receiving Notification of Documentation Updates

To receive notification of documentation updates, navigate to the device product folder on ti.com. In the upper right corner, click on *Alert me* to register and receive a weekly digest of any product information that has changed. For change details, review the revision history included in any revised document.

14.3 Community Resources

The following links connect to TI community resources. Linked contents are provided "AS IS" by the respective contributors. They do not constitute TI specifications and do not necessarily reflect TI's views; see TI's Terms of Use.

TI E2E™ Online Community ***TI's Engineer-to-Engineer (E2E) Community.*** Created to foster collaboration among engineers. At e2e.ti.com, you can ask questions, share knowledge, explore ideas and help solve problems with fellow engineers.

Design Support ***TI's Design Support*** Quickly find helpful E2E forums along with design support tools and contact information for technical support.

14.4 Trademarks

E2E is a trademark of Texas Instruments.
All other trademarks are the property of their respective owners.

14.5 Electrostatic Discharge Caution

These devices have limited built-in ESD protection. The leads should be shorted together or the device placed in conductive foam during storage or handling to prevent electrostatic damage to the MOS gates.

14.6 Glossary

SLYZ022 — *TI Glossary*.

This glossary lists and explains terms, acronyms, and definitions.

15 Mechanical, Packaging, and Orderable Information

The following pages include mechanical, packaging, and orderable information. This information is the most current data available for the designated devices. This data is subject to change without notice and revision of this document. For browser-based versions of this data sheet, refer to the left-hand navigation.

TEXAS INSTRUMENTS

www.ti.com

PACKAGE OPTION ADDENDUM

20-Nov-2018

PACKAGING INFORMATION

Orderable Device	Status (1)	Package Type	Package Drawing	Pins	Package Qty	Eco Plan (2)	Lead/Ball Finish (6)	MSL Peak Temp (3)	Op Temp (°C)	Device Marking (4/5)	Samples
SN65HVD233D	ACTIVE	SOIC	D	8	75	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM	-40 to 125	VP233	Samples
SN65HVD233DG4	ACTIVE	SOIC	D	8	75	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM	-40 to 125	VP233	Samples
SN65HVD233DR	ACTIVE	SOIC	D	8	2500	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM	-40 to 125	VP233	Samples
SN65HVD233DRG4	ACTIVE	SOIC	D	8	2500	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM	-40 to 125	VP233	Samples
SN65HVD234D	ACTIVE	SOIC	D	8	75	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM	-40 to 125	VP234	Samples
SN65HVD234DR	ACTIVE	SOIC	D	8	2500	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM	-40 to 125	VP234	Samples
SN65HVD234DRG4	ACTIVE	SOIC	D	8	2500	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM	-40 to 125	VP234	Samples
SN65HVD235D	ACTIVE	SOIC	D	8	75	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM	-40 to 125	VP235	Samples
SN65HVD235DG4	ACTIVE	SOIC	D	8	75	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM	-40 to 125	VP235	Samples
SN65HVD235DR	ACTIVE	SOIC	D	8	2500	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM	-40 to 125	VP235	Samples

(1) The marketing status values are defined as follows:
ACTIVE: Product device recommended for new designs.
LIFEBUY: TI has announced that the device will be discontinued, and a lifetime-buy period is in effect.
NRND: Not recommended for new designs. Device is in production to support existing customers, but TI does not recommend using this part in a new design.
PREVIEW: Device has been announced but is not in production. Samples may or may not be available.
OBSOLETE: TI has discontinued the production of the device.

(2) **RoHS:** TI defines "RoHS" to mean semiconductor products that are compliant with the current EU RoHS requirements for all 10 RoHS substances, including the requirement that RoHS substance do not exceed 0.1% by weight in homogeneous materials. Where designed to be soldered at high temperatures, "RoHS" products are suitable for use in specified lead-free processes. TI may reference these types of products as "Pb-Free".
RoHS Exempt: TI defines "RoHS Exempt" to mean products that contain lead but are compliant with EU RoHS pursuant to a specific EU RoHS exemption.
Green: TI defines "Green" to mean the content of Chlorine (Cl) and Bromine (Br) based flame retardants meet JS709B low halogen requirements of <=1000ppm threshold. Antimony trioxide based flame retardants must also meet the <=1000ppm threshold requirement.

(3) MSL, Peak Temp. - The Moisture Sensitivity Level rating according to the JEDEC industry standard classifications, and peak solder temperature.

TEXAS INSTRUMENTS

www.ti.com

PACKAGE OPTION ADDENDUM

20-Nov-2018

(4) There may be additional marking, which relates to the logo, the lot trace code information, or the environmental category on the device.

(5) Multiple Device Markings will be inside parentheses. Only one Device Marking contained in parentheses and separated by a "~" will appear on a device. If a line is indented then it is a continuation of the previous line and the two combined represent the entire Device Marking for that device.

(6) Lead/Ball Finish - Orderable Devices may have multiple material finish options. Finish options are separated by a vertical ruled line. Lead/Ball Finish values may wrap to two lines if the finish value exceeds the maximum column width.

Important Information and Disclaimer: The information provided on this page represents TI's knowledge and belief as of the date that it is provided. TI bases its knowledge and belief on information provided by third parties, and makes no representation or warranty as to the accuracy of such information. Efforts are underway to better integrate information from third parties. TI has taken and continues to take reasonable steps to provide representative and accurate information but may not have conducted destructive testing or chemical analysis on incoming materials and chemicals. TI and TI suppliers consider certain information to be proprietary, and thus CAS numbers and other limited information may not be available for release.

In no event shall TI's liability arising out of such information exceed the total purchase price of the TI part(s) at issue in this document sold by TI to Customer on an annual basis.

OTHER QUALIFIED VERSIONS OF SN65HVD233, SN65HVD234, SN65HVD235 :

- Automotive: SN65HVD233-Q1, SN65HVD234-Q1, SN65HVD235-Q1
- Enhanced Product: SN65HVD233-EP

NOTE: Qualified Version Definitions:

- Automotive - Q100 devices qualified for high-reliability automotive applications targeting zero defects
- Enhanced Product - Supports Defense, Aerospace and Medical Applications

TEXAS INSTRUMENTS

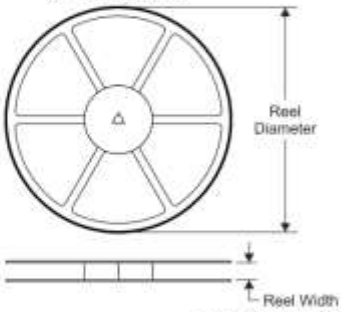
PACKAGE MATERIALS INFORMATION

www.ti.com

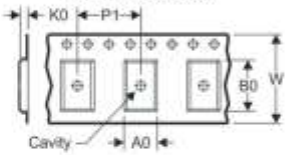
20-Nov-2018

TAPE AND REEL INFORMATION

REEL DIMENSIONS

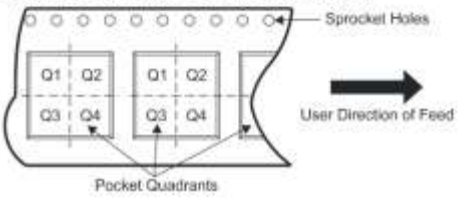


TAPE DIMENSIONS



A0	Dimension designed to accommodate the component width
B0	Dimension designed to accommodate the component length
K0	Dimension designed to accommodate the component thickness
W	Overall width of the carrier tape
P1	Pitch between successive cavity centers

QUADRANT ASSIGNMENTS FOR PIN 1 ORIENTATION IN TAPE



*All dimensions are nominal

Device	Package Type	Package Drawing	Pins	SPQ	Reel Diameter (mm)	Reel Width W1 (mm)	A0 (mm)	B0 (mm)	K0 (mm)	P1 (mm)	W (mm)	Pin1 Quadrant
SN65HVD233DR	SOIC	D	8	2500	330.0	12.4	6.4	5.2	2.1	8.0	12.0	Q1
SN65HVD234DR	SOIC	D	8	2500	330.0	12.4	6.4	5.2	2.1	8.0	12.0	Q1
SN65HVD235DR	SOIC	D	8	2500	330.0	12.4	6.4	5.2	2.1	8.0	12.0	Q1

Pack Materials-Page 1

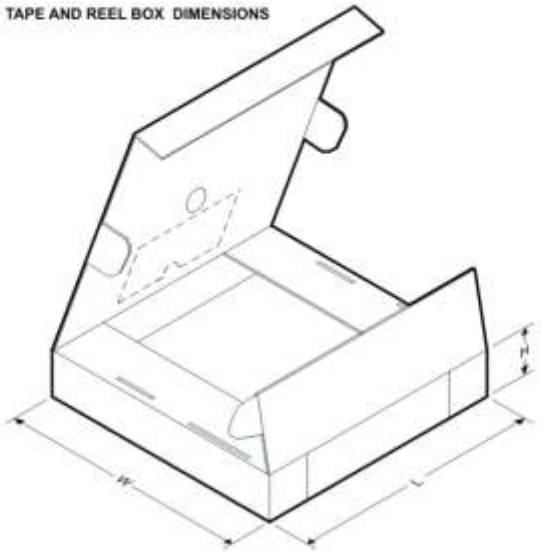
TEXAS INSTRUMENTS

PACKAGE MATERIALS INFORMATION

www.ti.com

20-Nov-2018

TAPE AND REEL BOX DIMENSIONS



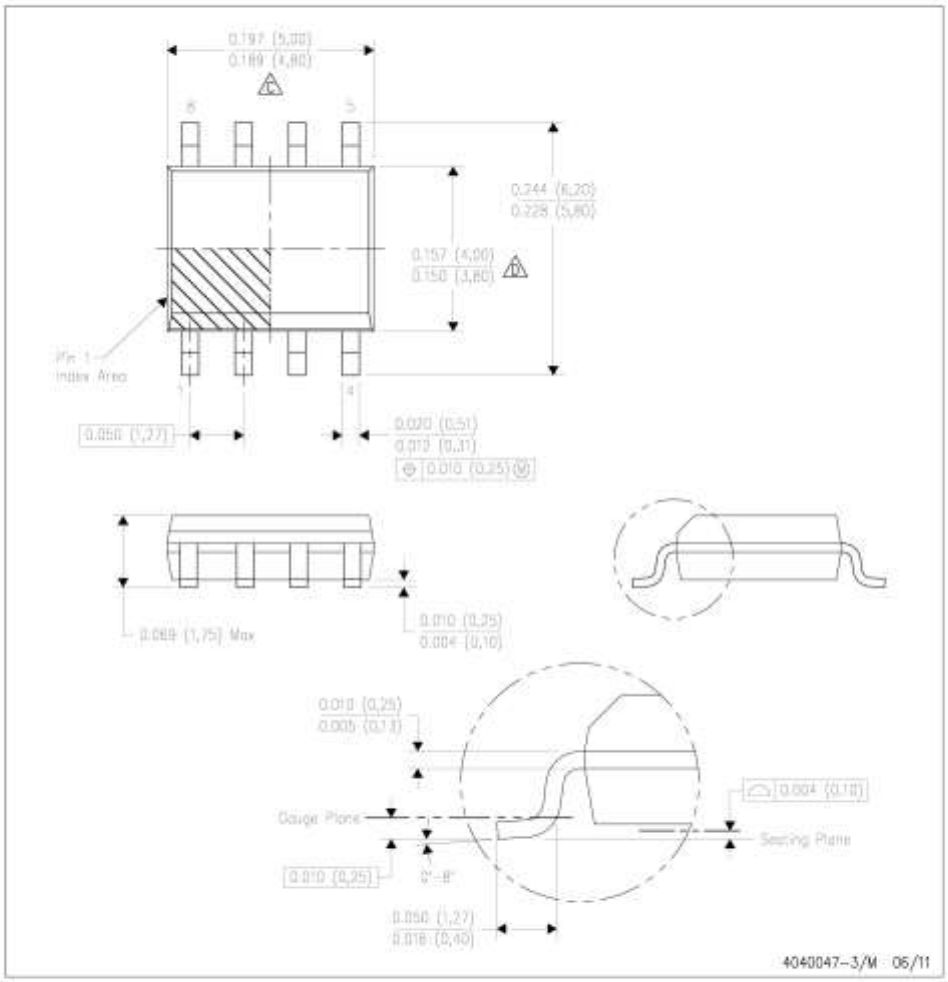
*All dimensions are nominal

Device	Package Type	Package Drawing	Pins	SPQ	Length (mm)	Width (mm)	Height (mm)
SN65HVD233DR	SOIC	D	8	2500	340.5	338.1	20.6
SN65HVD234DR	SOIC	D	8	2500	340.5	338.1	20.6
SN65HVD235DR	SOIC	D	8	2500	340.5	338.1	20.6

Pack Materials-Page 2

MECHANICAL DATA

D (R–PDSO–G8) PLASTIC SMALL OUTLINE



NOTES:
A. All linear dimensions are in inches (millimeters).
B. This drawing is subject to change without notice.
C. Body length does not include mold flash, protrusions, or gate burrs. Mold flash, protrusions, or gate burrs shall not exceed 0.006 (0,15) each side.
D. Body width does not include interlead flash. Interlead flash shall not exceed 0.017 (0,43) each side.
E. Reference JEDEC MS–012 variation AA.

TEXAS INSTRUMENTS
www.ti.com

Relación de documentos

(_) Memoria ..... 163 páginas

(X) Anexos ..... 275 páginas

(X)Esquemas y planos ..... 8 páginas

La Almunia, a 27 de noviembre del 2018

Firmado: Alfonso Mareca Miralles