THIS SPEC IS OBSOLETE

Spec No: 002-05535

Spec Title: AN205535 - FM0+ S6E100X POWER METER DEMO BOARD

Replaced by: None

AN205535

FM0+ S6E100x Power Meter Demo Board

Author: Mily Wang

Associated Part Family: S6E100X

Associated Code Examples: (FWSC)PM_ITRON_PDL

AN205535 documents the S6E100X power meter demo board solution and configuration. At the same time, the AN also provides source code for secondary development.

Contents

1 Introduction

1.1 Purpose

This application note describes the power meter demo solution, demo board configuration, and the source code for secondary development.

1.2 Definitions, Acronyms and Abbreviations

N/A

1.3 Document Overview

The rest of document is organized as the following:

Chapter 2 explains System Hardware Environment

Chapter 3 explains Development Environment

Chapter 4 explains Hardware Introduction

Chapter 5 explains System Function

Chapter 6 explains Firmware Introduction

Chapter 7 explains Demo System

2 System Hardware Environment

Table 1. Hardware Environment

Item	Details
MCU	S6E1003 On-chip Flash Memory: 512K + 48K BYTE On-chip SRAM: 60 + 4K BYTE
System Power Supply	Electric Supply(220V 50Hz) Battery Supply(3.6V)
Interface	JTAG
Board	PM-FM0P-SOLU_V1.1.0

3 Development Environment

Table 2. Firmware Environment

Item	Details
Operating System	Windows 7
Compiler	IAR 7.1 / Keil 7.1
Development language	C

4 Hardware Introduction

4.1 Hardware Overview

The Power Meter Demo Board development is intended to provide the useful and low layer development of Power Meter applications based on Cypress S6E100X Series MCUs which are embedded with ARM Cotex-M0 core.

The board supports SWD, UART, IC card, IR, ESAM interface. It contains LCD, LED, Key and Buzzer on the board. The power system includes primary electrical system and standby battery system.

Figure 1. Front View of Demo Board

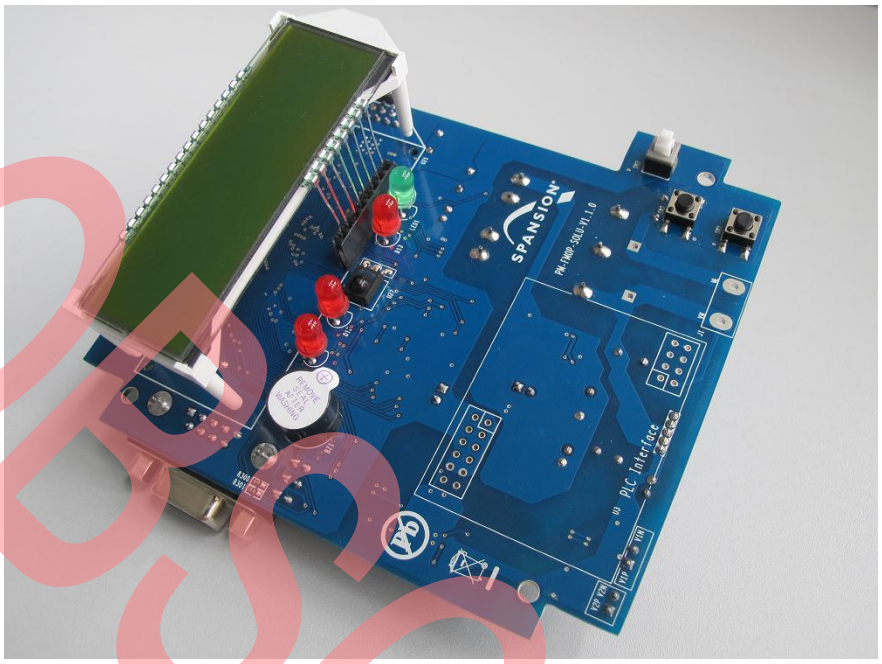
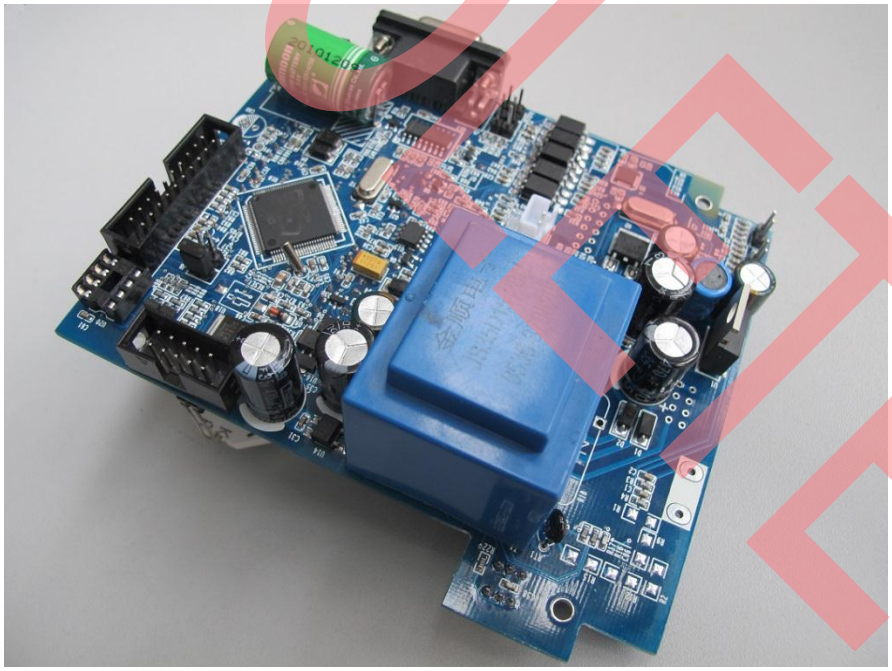


Figure 2. Rear View of Demo Board



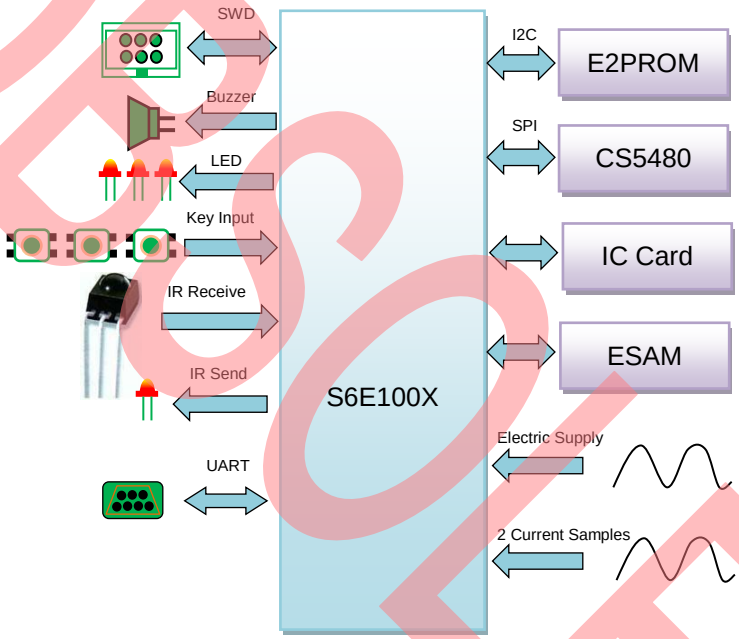
4.1.1 Features

The features of the development board include:

- Electric Supply: 220V 50Hz;
- Battery Supply: ER14250 (3.6V)
- Provide basic input/output (3 Keys, 3 LEDs, 8COM LCD, 1Buzzer);
- Provide multiple interfaces (1 UART, 1 SWD, 1 IR Receive, 1 IR Send, 1 IC Card, 1 ESAM);

4.1.2 System Block Diagram

Figure 3. Hardware System Block Diagram



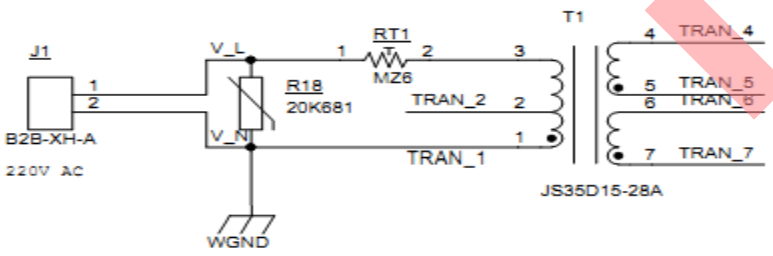
4.2 Power Supply

4.2.1 Electric Power Supply

Several different power sources of system are produced by the electric supply via the Power Supply module. It includes transformer, rectifier, LDO and DC-DC. Please refer to the following figure for detail.

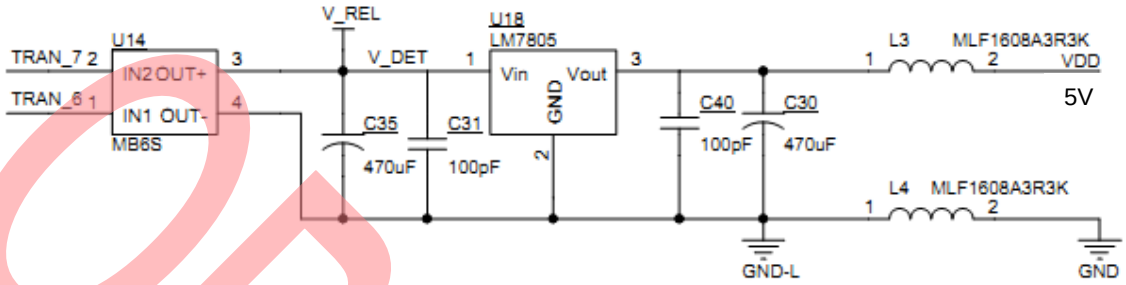
The transformer circuit is as shown in Figure 4. Transformer Circuit. It reduces the range of the electric supply for the power supply module.

Figure 4. Transformer Circuit



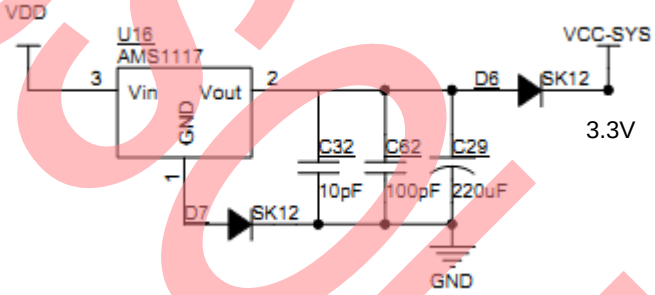
There are several independent power supplies generated by different power supply circuits. It includes VDD which is power source for IR module, Buzzer LED and PLC module. The circuit is as shown in Figure 5. Power Supply Circuit of VDD

Figure 5. Power Supply Circuit of VDD



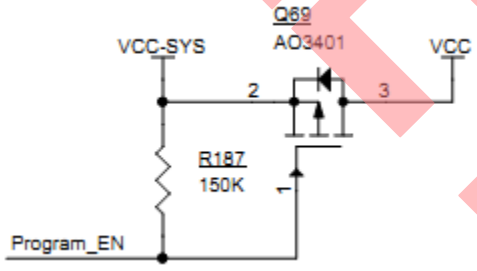
The VCC_SYS is power source for the whole minimum MCU system. Please refer to Figure 6. Power Supply Circuit of VCC_SYS. It supplies power for the VCC at the same time.

Figure 6. Power Supply Circuit of VCC_SYS



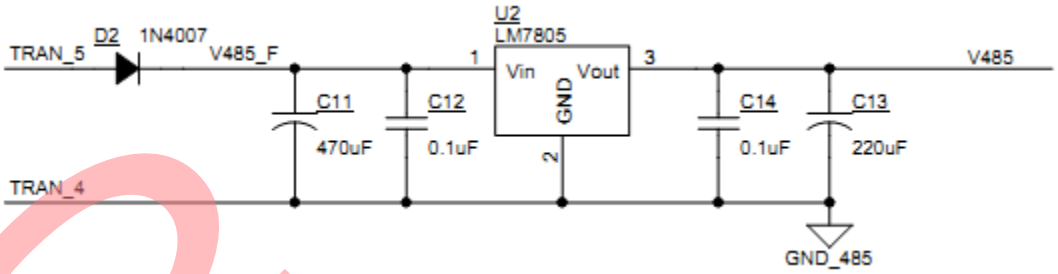
The VCC is power source for UART module, Program IF module, E2PROM, Key module, CS5480, 485, IC card module, RS485 module and ESAM module. The VCC can be controlled by a control circuit as shown in Figure 7. Control Circuit for VCC

Figure 7. Control Circuit for VCC



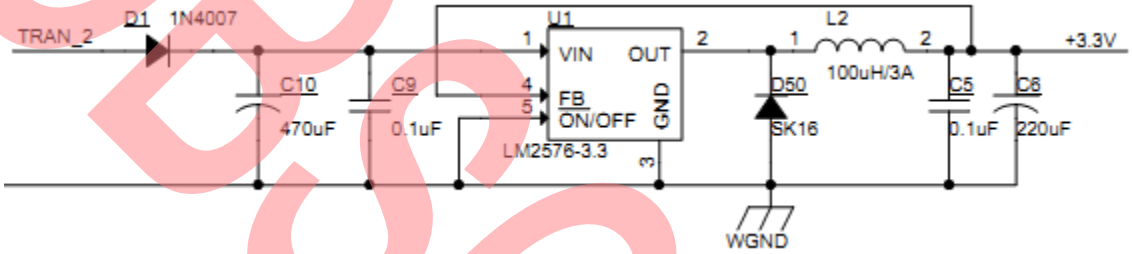
The V485 is the power source for RS485 module.

Figure 8. Power Supply Circuit of V485



The power source of +3.3V is power supply for CS5480 module.

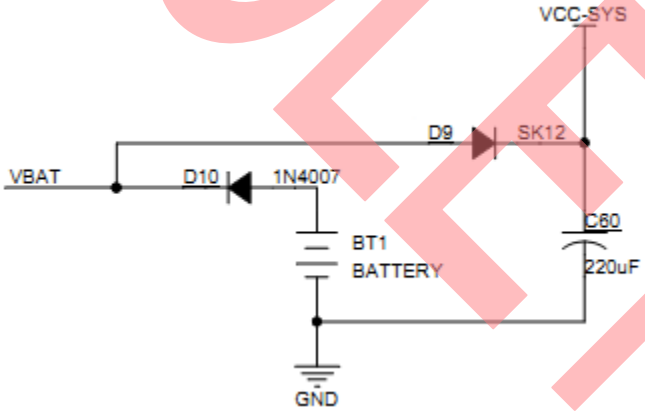
Figure 9. Power Supply Circuit of CS5480



4.2.2 Battery Supply

The battery supply is the power source of the system when primary electric supply failed.

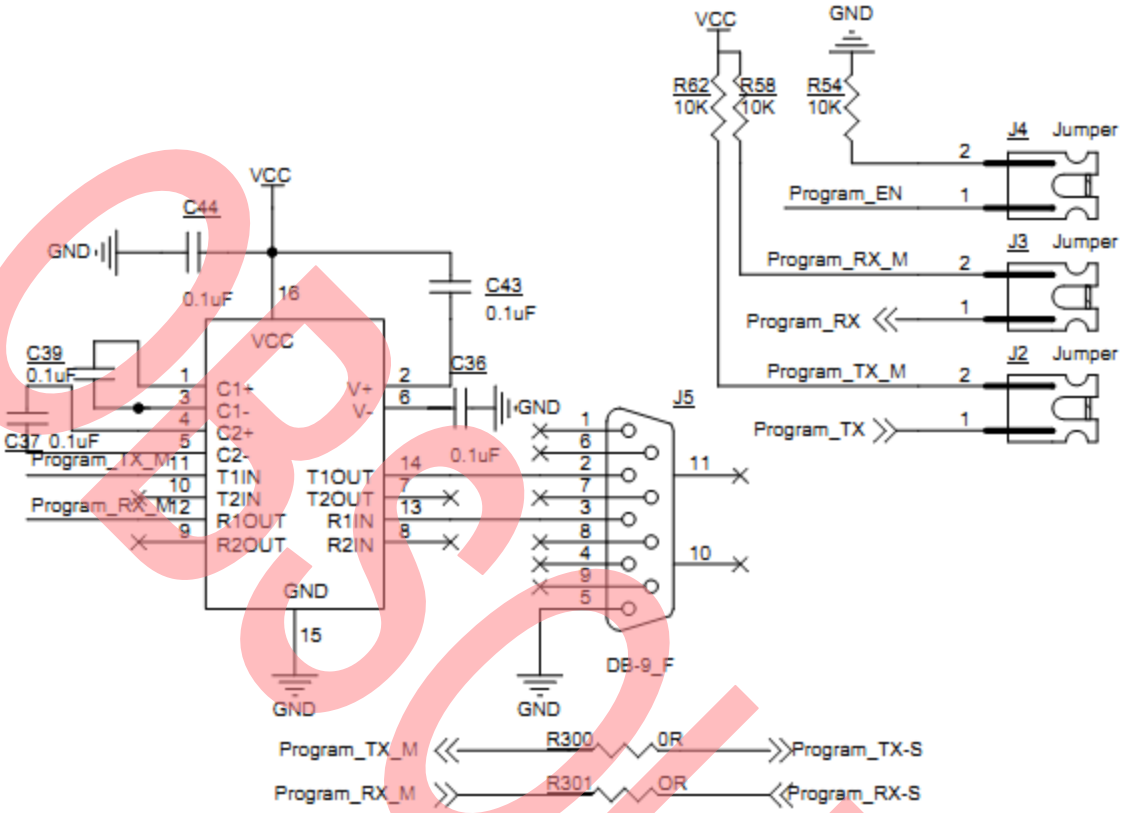
Figure 10. Battery Supply Circuit



4.3 UART Module

UART module has two channels. One channel includes two ports: Program_TX and Program_RX. It connects to P60 and P61 which is used for internal flash program. The other channel includes two ports: Program_TX_M and Program_RX_M. it connects to SOT3_2 and SIN3_2.

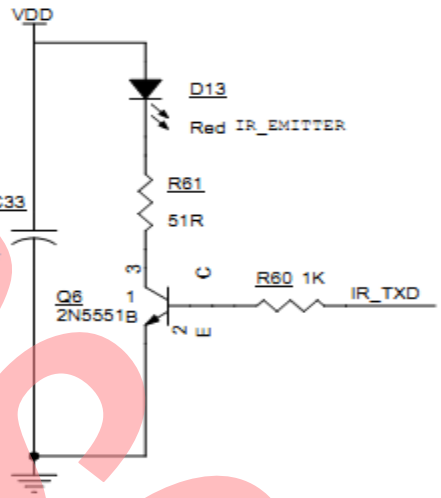
Figure 11.Circuit for UART



4.4 IR Module

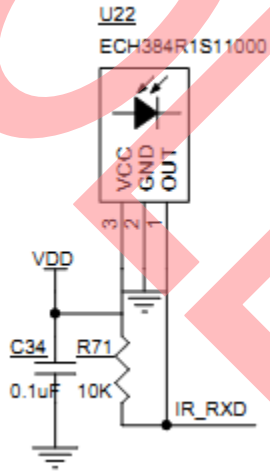
4.4.1 IR Send

Figure 12. IR Send Circuit



4.4.2 IR Receive

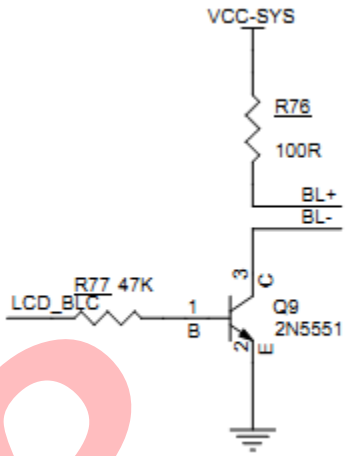
Figure 13. IR Receive Circuit



4.5 CL010 Module

4.5.1 LCD Backlight Control

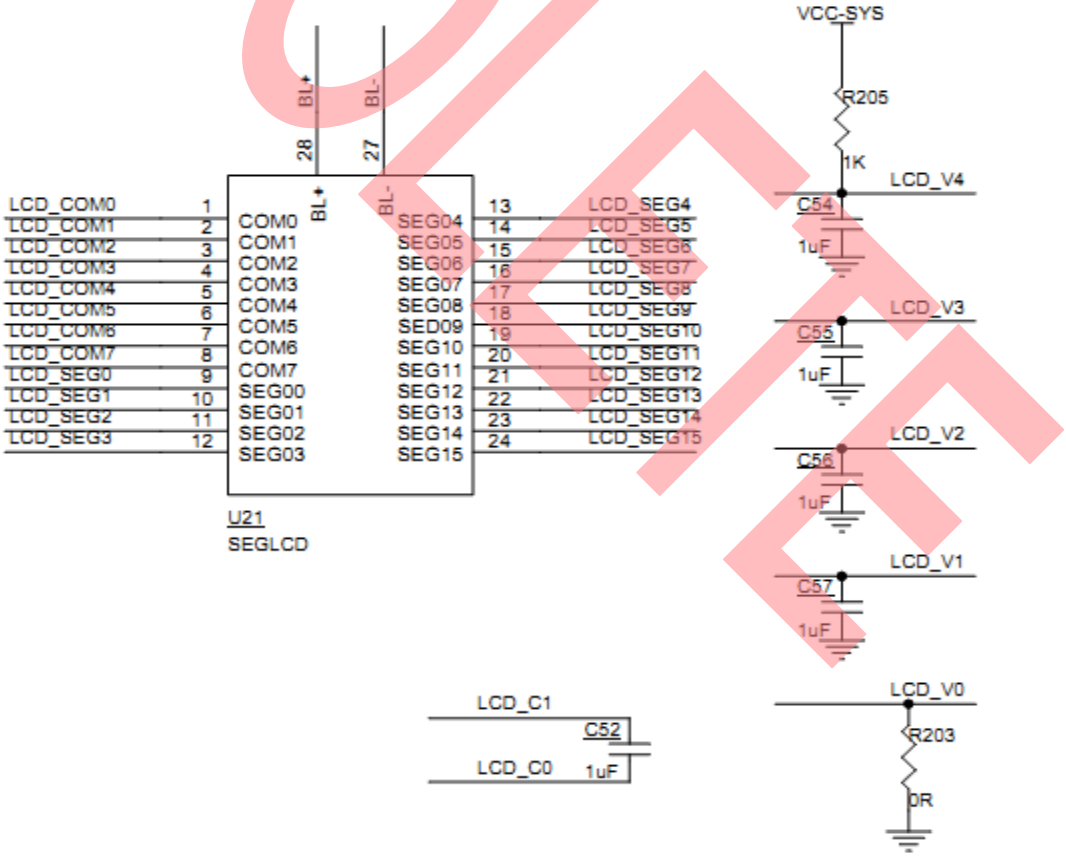
Figure 14. Backlight Control Circuit



4.5.2 LCD Display Control

To use the LCD boost function, R205 should be removed from the circuit of LCD control. If you do not use the LCD boost function, the component of C52 is not necessary for the circuit.

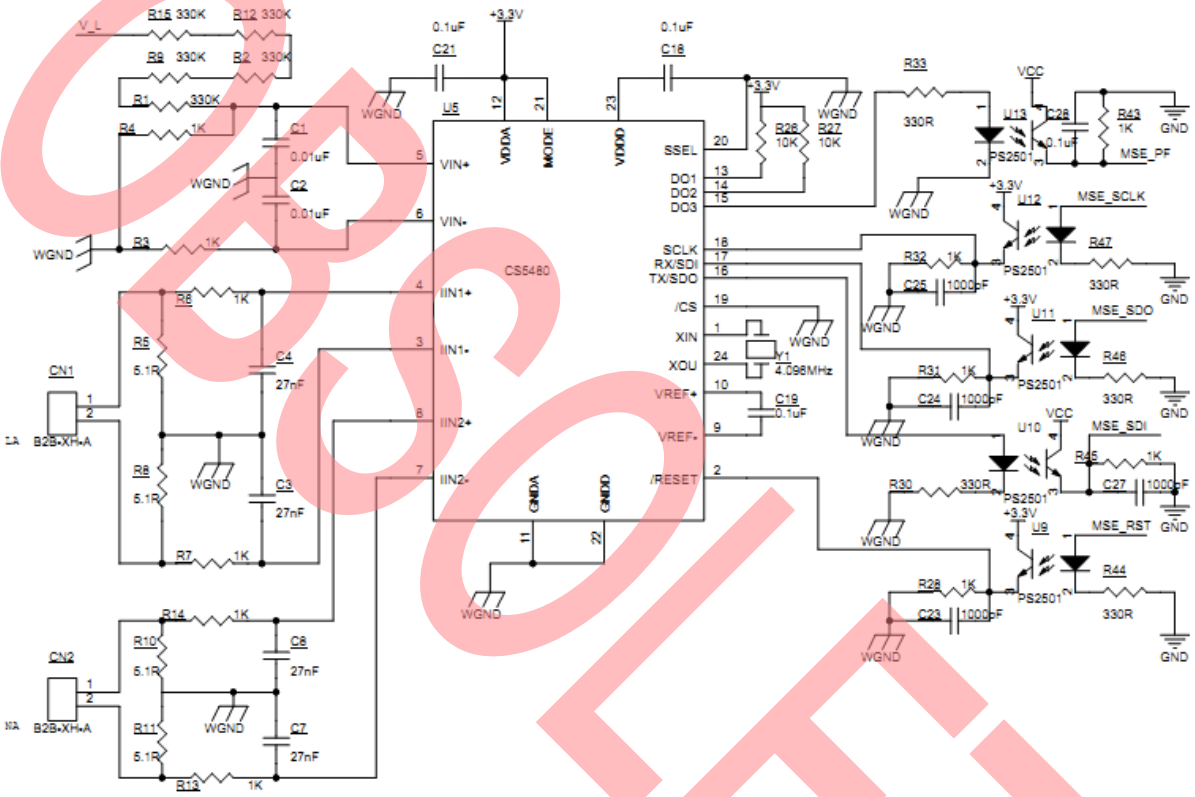
Figure 15. LCD Display Control Circuit



4.6 CS5480 Module

As shown in Figure 16. CS5480 Circuit, the V_L and WGND are electric supply input. LA and LB are current sample from the inductor. MSE_SCLK, MSE_SDO and MSE_SDI are interface between MCUs communicating through UART protocol. MSE_RST is hardware reset control pin which is controlled by a GPIO of MCU. MSE_PF is pulse out pin which outputs pulse from CS5480 when measuring function is on. Please note that the power source VCC and +3.3V are independent power source in the system. For details, please refer to 4.2.1.

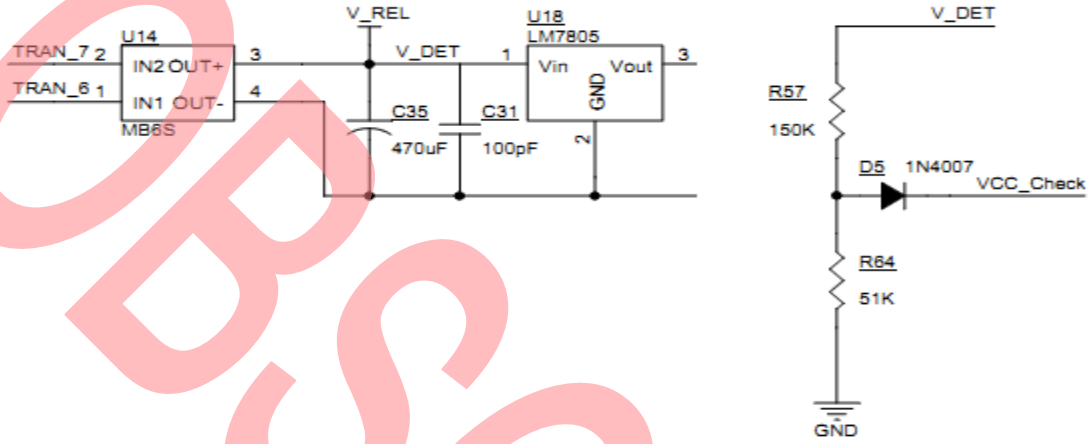
Figure 16. CS5480 Circuit



4.7 LVD Input

There are two channels of LVD in the power meter system. One channel named channel 0 detects the voltage of VCC-SYS. Another channel named channel 1 detects part of the voltage of electric supply, as shown in Figure 17. LVD1 Detect Circuit.

Figure 17. LVD1 Detect Circuit



5 System Function

5.1 IR Receive

IR receive function module receives IR data through the RC5 protocol.

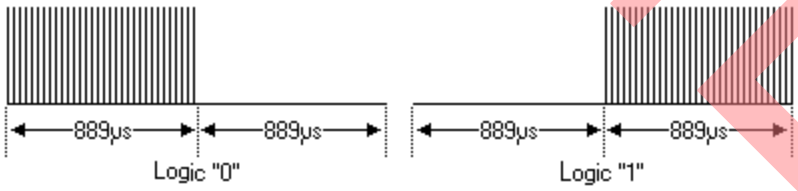
5.1.1 IR RC5 Protocol

The features of RC5 protocol are as below:

- 5-bit address and 6-bit command length (7 command bits for RC5X)
- Bi-phase coding (aka Manchester coding)
- Carrier frequency of 36kHz
- Constant bit time of 1.778ms (64 cycles of 36 kHz)

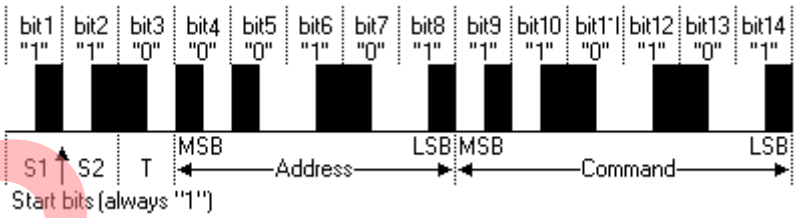
The data definition of each bit is as shown in Figure 18. RC5 Data Definition.

Figure 18. RC5 Data Definition



One frame of RC5 protocol data is as shown in Figure 19. RC5 Frame Data.

Figure 19. RC5 Frame Data



5.2 IR Send

IR Send function module sends IR signal through the RC5 protocol.

5.3 CS5480

CS5480 is an electric power measurement IC which supports SPI interface communication. This module's function is to communicate with Power Measure IC CS5480 via SPI interface and calculate the power consumption of the load circuit.

5.3.1 Specification of CS5480

The CS5480 contains 4 types host commands, such as reading/writing register and instruct the calculation engine. The two most significant bits (MSBs) of the host command define the function to be performed. The detail is shown inFigure 20. CS5480 Command type.

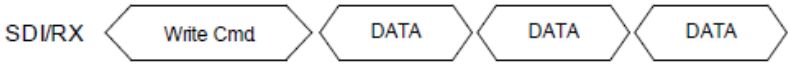
Figure 20. CS5480 Command type

Function	Binary Value	Note
Register Read	0 0 A_5 A_4 A_3 A_2 A_1 A_0	$A_{[5:0]}$ specifies the register address.
Register Write	0 1 A_5 A_4 A_3 A_2 A_1 A_0	
Page Select	1 0 P_5 P_4 P_3 P_2 P_1 P_0	$P_{[5:0]}$ specifies the page.
Instruction	1 1 C_5 C_4 C_3 C_2 C_1 C_0	$C_{[5:0]}$ specifies the instruction.

1. Page Select

2. Register Read

3. Register Write

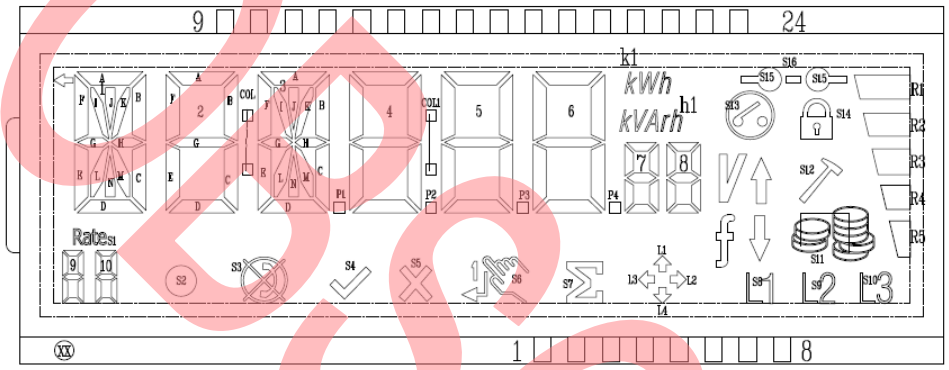


4. Send Command

5.4 CL010

This module is the driver of the 8COMs LCD for ITRON power meter. The LCD front specification and the relation of RAM table are as shown in Figure 21. LCD Specification.

Figure 21. LCD Specification



PIN	C1	C2	C3	C4	C5	C6	C7	C8
S1	⇦	1F	1G	1E	S1	9F	9G	9E
S2	1I	1J	1L	1D	9A	9B	9C	9D
S3	1A	1K	1N	1M	S2	10F	10G	10E
S4	1B	1H	1C	S3	10A	10B	10C	10D
S5	2F	2G	2E	S4	S5	S6	S7	L3
S6	2A	2B	2C	2D	L1	L4	S8	S9
S7	3F	3G	3E	COL	L2	f	⇩	S10
S8	3I	3J	3L	3D	V	⇧	S11	R5
S9	3A	3K	3N	3M	S13	S14	S12	R4
S10	3B	3H	3C	S16	S15	R1	R2	R3
S11	4F	4G	4E	P1	h	W	K1	k
S12	4A	4B	4C	4D	h1	r	A	V
S13	5F	5G	5E	COL	8A	8B	8C	P2
S14	5A	5B	5C	5D	8F	8G	8E	8D
S15	6F	6G	6E	P3	7A	7B	7C	7D
S16	6A	6B	6C	6D	7F	7G	7E	P4

5.5 RTC Function

The Real Time Count (RTC) function is based on the RTC module driver in the PDL project. The function will be provided to VBAT Switch sample. It counts the Year, Month, Day, Hour, Minute and Second.

5.6 VBAT Switch Sample Source

The sample source depends on the two channel of LVD function. LVD0 channel detects voltage of VCC, which is indicated by solid line in Figure 22. VBAT Switch Sample Operation Flow. LVD1 channel detects voltage of electric supply, which is indicated by dotted line in Table 3. VBAT Switch Sample Trigger. The sample source of the VBAT switch is as shown in bellow figure and Table 3. VBAT Switch Sample Trigger

5.6.1 VBAT Switch Sample Operation Flow

Figure 22. VBAT Switch Sample Operation Flow

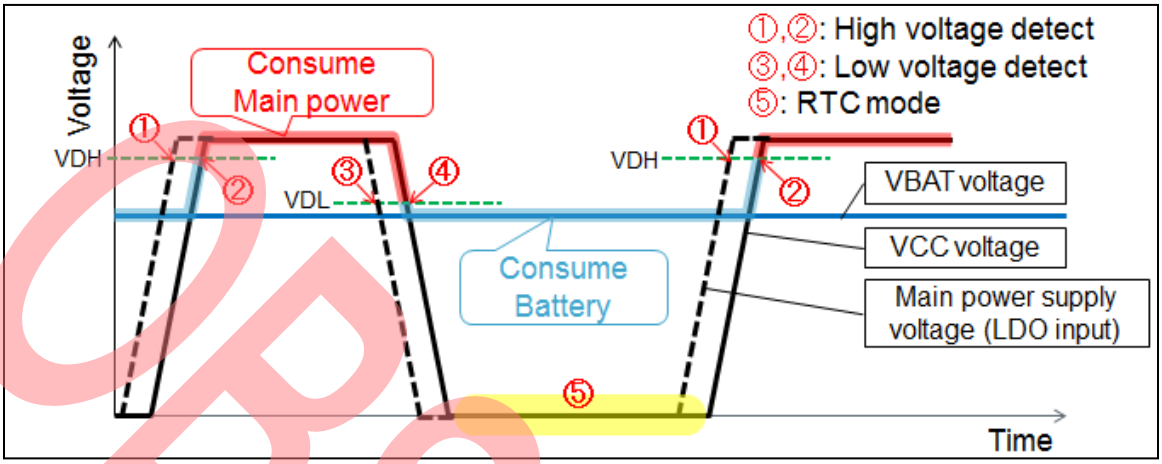


Table 3. VBAT Switch Sample Trigger

#	Trigger	Action
①	Detect high voltage at main power supply by LVD1	N.A.
②	Detect high voltage at VCC by LVD0	Turn off SW1 / Turn on SW2 automatic/software Wake up CPU Read out 300 bytes from EEPROM, UART 57600bps. Measure VBAT voltage by ADC Set suitable VDL based on 4. Display RTC time on LCD
③	Detect low voltage at main power supply by LVD1	Read, increment, write 300 bytes into EEPROM (Measure VBAT voltage by ADC) (Set suitable VDL based on 2.)
④	Detect low voltage at VCC by LVD0	Turn on SW1 / Turn off SW2 automatic/software Go to DS-RTC mode immediately
⑤	Wake up interrupt by button press	Display RTC time on LCD for 5 sec, then off LCD and go back into DS-RTC mode

6 Firmware Introduction

6.1 Firmware Overview

There are six folders in the structure of firmware which is indicated in Figure 23. Structure of the Firmware.

Figure 23. Structure of the Firmware

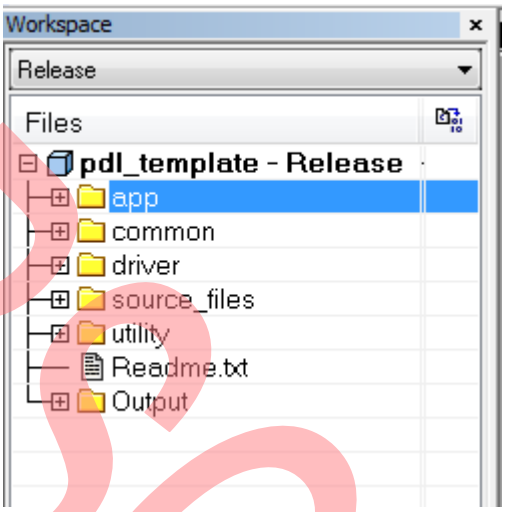


Table 4. Directory Description of Project

Folder	Description
app	It includes application layer files of power meter project.
common	It includes system file and global macro definition, register definition and FM0 system initialization functions
driver	It includes FM0 MCU peripheral driver library
utility	It includes application sample files.
readme.txt	It records the version log and upgrade information of firmware
Output	It includes .map and .out files.

The detailed file description for necessary file is shown in Table 5. File Description of Project.

Table 5. File Description of Project

Folder	File	Description	Mark
app	CS5480.c	Source file for CS5480 functions	N/A
	CS5480.h	Header file for CS5480 functions	N/A
app	ICCard.c	Source file for IC Card function	N/A
	ICCard.h	Header file for IC Card	N/A
app	IR_Rev.c	Source file for IR receive communication	N/A
	IR_Rev.h	Header file for IR communication	N/A
app	IR_Send.c	Source file for IR Send communication	N/A
	IR_Send.h	Header file for IR communication	N/A
app	LCDDisplay.c	Source file for LCD Display functions	N/A
	LCDDisplay h	Header file for LCD display function	N/A
app	rtc_device.c	Source file for RTC user interface	N/A
	rtc_device.h	Header file for RTC user interface	N/A
app	VbatSwitch.c	Source file for VBAT Switch sample	N/A
	VbatSwitch.h	Header file for VBAT Switch sample	N/A
source_files	main.c	Source code for main process	N/A
	main.h	Header file for main file	N/A
utility	cl010.c	CL010_7031 LCD display functions	N/A
	cl010.h	Header file for CL010_7031 LCD display functions	N/A
utility	i2c_polling_at24xxx.c	Driver for E2PROM	N/A
	i2c_polling_at24xxx.h	Header file for Driver of E2PROM	N/A

6.1.1 Initialization Flow

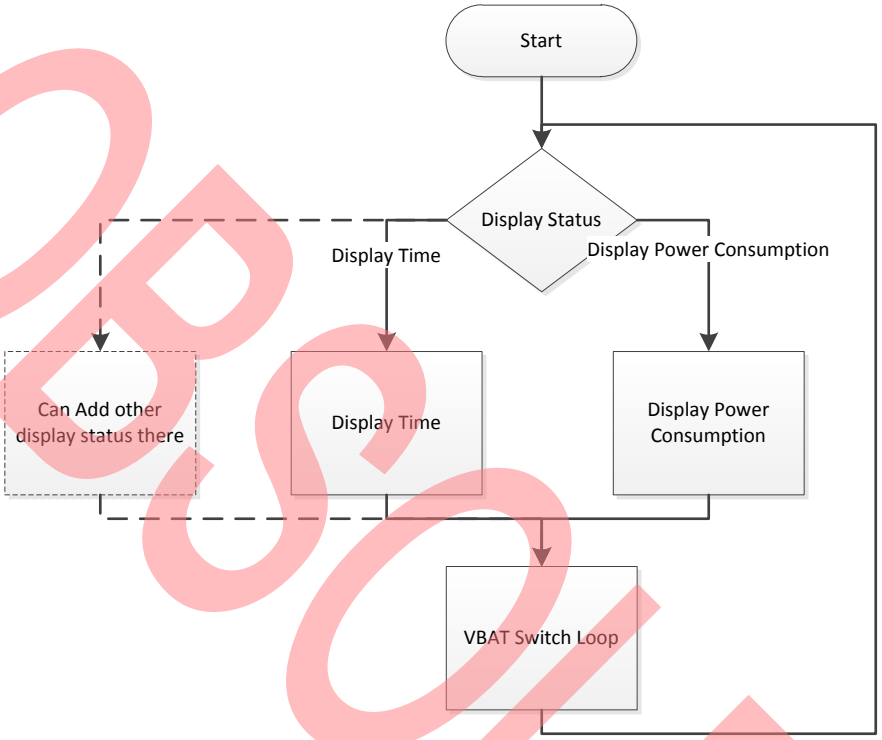
The system Initialization Flow is as below figure. Please be careful that IR receive and IR send Module can't run at the same time, because there will be serious noise on the hardware.

Figure 24. System Initialization Flow

6.1.2 Main Loop Function

Please see Figure 25. System Main Loop Flow for detail. More display status can be added in the main loop process as figure below.

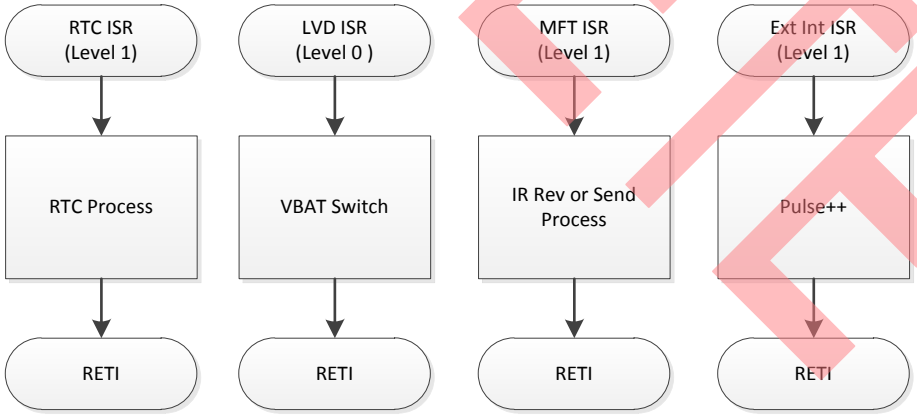
Figure 25. System Main Loop Flow



6.1.3 ISR Function Flow

The primary ISR used in the system is as shown in Figure 26. System ISR Function Flow.

Figure 26. System ISR Function Flow



6.2 IR Receive

6.2.1 Macro Definition

stc_IR_Rev_FIFO_t	typedef struct stc_IR_Rev_FIFO
	{
	uint32_t u32RevBuf[REV_FIFO_LEN]; //data received FIFO
	uint8_t u8WRIndex; //Write index of fifo
	uint8_t u8RDIndex; //Read index of fifo
	uint8_t u8EmptNum;
	}stc_IR_Rev_FIFO_t;
		/

stc_IR_Rev_Statu_t	typedef enum en_IR_RevStage{
	IR_REV_NONE = 0,
	IR_REV_S0_LOW = 1,
	IR_REV_S0_HIGH = 2,
	IR_REV_DATA = 3,
	}en_IR_RevStage_t;
	
	typedef struct stc_IR_Rev_Statu
	{
	en_IR_RevStage_t enRevStage;
	uint8_t u8DataNum; //data had been received(in bit)
	}stc_IR_Rev_Statu_t;
		/

stc_rc5_data_t	typedef struct stc_rc5_data { uint8_t u8Adr; uint8_t u8Cmd; boolean_t u8Repeat; }stc_rc5_data_t;	/
#define REV_FIFO_LEN 40	Indicate the max number of the storage data.	/

6.2.2 Global Data

Data Type	Definition	Description
stc_IR_Rev_FIFO_t	stcIRRevFIFO	Received data buffer and FIFO status

6.2.3 Function Details

1) Receive Initialization

[Syntax]

void IR_RevInit(void)

Parameters	In/Out	Type	Description
void	/	/	/

[Description]

Initialize the IR Receive status and ICU module to receive the IR signal.

2) Receive Stop

[Syntax]

void IR_RevDeinit(void)

Parameters	In/Out	Type	Description
void	/	/	/

[Description]

Disable the IR module and ICU module to disable receiving the IR signal.

3) Receive FIFO Initialization

[Syntax]

void IR_RevFifoIni(void)

Parameters	In/Out	Type	Description
void	**/**	**/**	**/**

[Description]

Initialize receive FIFO. It includes clearing the receive buffer and initializing the FIFO status.

4) Read Receive FIFO

[Syntax]

en_result_t IR_RDRevFIFO(stc_rc5_data_t* pstcDataRead)

Parameters	In/Out	Type	Description
pstcDataRead	In	stc_rc5_data_t*	The pointer of the buffer which stores the read data.
en_result_t	Out	en_result_t	Indicate that if read success

[Description]

Read one data from receive FIFO.

5) ICU Interrupt

[Syntax]

void MftIcuIntHandler(void)

Parameters	In/Out	Type	Description
void	/	/	/

[Description]

The ICU interrupt handle.

6.3 IR Send

6.3.1 Macro Definition

stc_IR_SendStatus_t	typedef enum en_IR_SendOnOff{
	IR_SEND_NONE = 0, // IR send is finished or not on
	IR_SEND_ON = 1, // IR Send is on
	}en_IR_SendOnOff_t;

	typedef enum en_IR_SendEven{
	IR_SEND_EVEN = 0,
	IR_SEND_ODD = 1,
	}en_IR_SendEven_t;
	
	typedef struct stc_IR_SendStatus
	{
	en_IR_SendOnOff_t enIRSendOnOff;
	uint8_t u8IRSendPhase; //to indicate current phase
	en_IR_SendEven_t enIRPhaseEven;//to indicate the state is even or odd in one phase
	uint8_t u8IRSendMaxPhase; //the total number of the Phase
	}stc_IR_SendStatus_t;

6.3.2 Global Data

Data Type	Definition	Description
stc_IR_SendStatus_t	stc_IRSendStd	Indicate the IR send status

6.3.3 Function Details

1. Send Initialization

[Syntax]

void IR_SendInit(void)

Parameters	In/Out	Type	Description
void	/	/	/

[Description]

Initialize the IR send status and OCU module to get ready for sending the IR signal.

2. Send Data

[Syntax]

en_result_t IR_Send_Data_RC5(uint8_t u8Adr, uint8_t u8Cmd, boolean_t bISRepeat)

Parameters	In/Out	Type	Description
u8Adr	In	uint8_t	The address to send.
u8Cmd	In	uint8_t	The command to send
bISRepeat	In	boolean_t	The repeated information in the send process
en_result_t	Out	en_result_t	Indicate that if send success

[Description]

Kick start the IR Send(RC5).

3. OCU Interrupt

[Syntax]

void Ocu1IntCallback(void)

Parameters	In/Out	Type	Description
void	/	/	/

[Description]

The OCU interrupt handle.

6.4 CS5480

6.4.1 Global Data

Data Type	Definition	Description
uint32_t	u32EnergyPulseCount	Pulse number

6.4.2 Static Definition

- Register Page Definition

```
/* Page Define */
#define Page0           0
#define Page16          16
#define Page17          17
#define Page18          18
```

- Register Definition

```
//Page0
#define Config0         0x00
#define Config1         0x01
#define Mask            0x03
#define PCreg           0x05
#define SerialCtrl      0x07
#define PulseWidth      0x08
#define PulseCtrl       0x09
#define Status0         0x17
#define Status1         0x18
#define Status2         0x19
#define RegLock         0x22
//Page16
#define Config2         0x00
#define RegChk          0x01
#define I1              0x02
#define V1              0x03
#define Power1          0x04
#define P1avg           0x05
#define I1rms           0x06
#define V1rms           0x07
#define I2              0x08
#define V2              0x09
#define Power2          0x0A
#define P2avg           0x0B
#define I2rms           0x0C
#define V2rms           0x0D
#define Q1avg           0x0E
#define Q1              0x0F
#define Q2avg           0x10
```

```
#define  Q2                0x11
#define  I1peak            0x12
#define  V1peak            0x13
#define  S1                0x14
#define  PF1               0x15
#define  I2peak            0x16
#define  V2peak            0x17
#define  S2                0x18
#define  PF2               0x19
#define  Temperature       0x1B
#define  Psum              0x1D
#define  Ssum              0x1E
#define  Qsum              0x1F
#define  I1dcoff           0x20
#define  I1gain            0x21
#define  V1dcoff           0x22
#define  V1gain            0x23
#define  P1off             0x24
#define  I1acoff           0x25
#define  Q1off             0x26
#define  I2dcoff           0x27
#define  I2gain            0x28
#define  V2dcoff           0x29
#define  V2gain            0x2A
#define  P2off             0x2B
#define  I2acoff           0x2C
#define  Q2off             0x2D
#define  Epsilon           0x31
#define  Ichlevel          0x32
#define  SampleCount       0x33
#define  Tgain             0x36
#define  Toff              0x37
#define  Emin              0x38
#define  Tsettle           0x39
#define  Loadmin           0x3A
#define  SysGain           0x3C
#define  SysTime           0x3D
//Page17
#define  V1Sagdur          0x00
#define  V1Saglevel        0x01
#define  I1Overdur         0x04
#define  I1Overlevel       0x05
#define  V2Sagdur          0x08
#define  V2Saglevel        0x09
#define  I2Overtdur        0x0C
#define  I2Overlevel       0x0D
//Page18
#define  PulseRate         0x1C
#define  INTgain           0x2B
#define  V1Swelldur        0x2E
#define  V1Swelllevel      0x2F
#define  V2Swelldur        0x32
#define  V2Swelllevel      0x33
#define  ZXlevel           0x3A
#define  CycleCount        0x3E
#define  Scale             0x3F
```

- Instruction Definition

```
#define  AFESWReset        0x01
#define  AFEStandby        0x02
#define  AFEWakeup         0x03
#define  SigneConv         0x14
#define  ContiConv         0x15
#define  StopConv          0x18
#define  CaliI1acoff       0x31
#define  CaliI2acoff       0x33
#define  CaliI1gn          0x39
#define  CaliV1gn          0x3A
#define  CaliI2gn          0x3B
#define  CaliV2gn          0x3C
```

6.4.3 Function Details

1. CS5480 Initialization

[Syntax]

void Cs5480_Init(void)

Parameters	In/Out	Type	Description
void	/	/	/

[Description]

Initialize CS5480. Call it to kick start the measuring operation.

1. Send Command to CS5480

[Syntax]

void Cs5480_SendCmd(uint8_t u8Cmd)

Parameters	In/Out	Type	Description
u8Cmd	In	uint8_t	Command to send

[Description]

Send command to CS5480.

2. Select Page

[Syntax]

void Cs5480_SelPage(uint8_t u8Page)

Parameters	In/Out	Type	Description
u8Page	In	uint8_t	Which page to select

[Description]

Select the page of CS5480.

3. Read Register of CS5480

[Syntax]

void Cs5480_ReadReg(uint8_t u8Addr,uint8_t *pu8Buff)

Parameters	In/Out	Type	Description
u8Addr	In	uint8_t	Which register to read
pu8Buff	In	uint8_t *	The point of the buffer which stores the data read from the register

[Description]

Read Register of CS5480. Make sure the page of the register has been configured by function Cs5480_SelPage.

4. Write Register of CS5480

[Syntax]

void Cs5480_WrtReg(uint8_t u8Addr,uint8_t u8High,uint8_t u8Mid,uint8_t u8Low)

Parameters	In/Out	Type	Description
u8High	In	uint8_t	CS5480 register value high byte
u8Mid	In	uint8_t	CS5480 register value middle byte
u8Low	In	uint8_t	CS5480 register value low byte

[Description]

Write Register of CS5480. Make sure the page of the register has been configured by function Cs5480_SelPage.

5. Reset the CS5480

[Syntax]

void Cs5480_Reset(void)

Parameters	In/Out	Type	Description
void	/	/	/

[Description]

Hardware reset the CS5480.

6. Enable Pulse Count

[Syntax]

void CS5480_EnablePulseCount(void)

Parameters	In/Out	Type	Description
void	/	/	/

[Description]

Enable the Extint function to start the pulse count.

7. ISR for Pulse

[Syntax]

void INT_09_Cs5480_ISR(void)

Parameters	In/Out	Type	Description
void	/	/	/

[Description]

Callback of the Extint interrupt for pulse.

6.5 CL010

6.5.1 Function Details

1. CL010 Initialization

[Syntax]

void CL010_Init(void)

Parameters	In/Out	Type	Description
void	/	/	/

[Description]

The function is Initialization of CL010 LCD.

2. CL010 Disenable

[Syntax]

void Cl010_DeInit(void)

Parameters	In/Out	Type	Description
void	/	/	/

[Description]

Close the LCDC module and close the backlight.

3. Display One Word

[Syntax]

en_result_t Cl010_DispLcdWord(uint8_t u8WordIndex)

Parameters	In/Out	Type	Description
u8WordIndex	In	uint8_t	Indicate that which word should be displayed. For the Word Index, please refer to the Spec of The CL010.
	return	en_result_t	Indicate whether the display succeeds. If Parameter out of the range. The FALSE will be returned. Oppositely the function will return TRUE.

[Description]

Display one word of the LCD.

4. Clear One Word

[Syntax]

en_result_t Cl010_ClrLcdWord(uint8_t u8WordIndex)

Parameters	In/Out	Type	Description
u8WordIndex	In	uint8_t	Indicate that which word should be cleared. For the Word Index, please refer to Spec of The CL010.
	return	en_result_t	Indicate that if clear operation successfully. If Parameter out of the range. The FALSE will be returned. Oppositely the function will return TRUE.

[Description]

Clear one word of the LCD.

5. Display One Number

[Syntax]

en_result_t Cl010_DispLcdCommonNum(uint8_t u8NumIndex, uint8_t u8DispVal)

Parameters	In/Out	Type	Description
u8NumIndex	In	uint8_t	Indicate which number should be displayed. For the Number Index, please refer to the Spec of The CL010.
u8DispVal	In	uint8_t	The display content.
	return	en_result_t	Indicate whether the display succeeds. If Parameter out of the range. The FALSE will be returned. Oppositely the function will return TRUE.

[Description]

Display one number of the LCD.

6. Clear One Number

[Syntax]

n_result_t Cl010_ClrLcdCommonNum(uint8_t u8NumIndex)

Parameters	In/Out	Type	Description
u8NumIndex	In	uint8_t	Indicate which number should be cleared. For the Number Index, please refer to the Spec of The CL010.
	return	en_result_t	Indicate whether clear successfully. If parameter is out of the range, the FALSE will be returned. Oppositely the function will return TRUE.

[Description]

Clear one number of the LCD.

7. Enable Backlight

[Syntax]

void Cl010_EnableLcdBackLight(void)

Parameters	In/Out	Type	Description
void	/	/	/

[Description]

Open the backlight of LCD.

8. Disable Backlight

[Syntax]

void Cl010_DisableLcdBackLight(void)

Parameters	In/Out	Type	Description
void	/	/	/

[Description]

Close the backlight of LCD.

9. Clear Screen of LCD

[Syntax]

void Cl010_ClrScreen(void)

Parameters	In/Out	Type	Description
void	/	/	/

[Description]

Clear LCD screen.

6.6 RTC Function

6.6.1 Global Data

Data Type	Definition	Description
uint32_t	u32SecCount	Count the second
stc_rtc_time_t	stcTimeDate	The current time, date and year information

6.6.2 Function Details

1. Initialize RTC Module

[Syntax]

en_result_t InitRtc(boolean_t bIniTime)

Parameters	In/Out	Type	Description
bIniTime	In	boolean _t	TRUE - - don't initialize RTC when RTC is running FALSE -- Initialize RTC any time
	return	en_result_t	Indicate that if initialize successfully.

[Description]

Initialize the RTC module. Include Configure the interrupt of RTC, configure the RTC register and kick start RTC module.

2. RT Interrupt Handle

[Syntax]

void SampleRtcOneSencondCb(void)

Parameters	In/Out	Type	Description
void	/	/	/

[Description]

The call back function of one second interrupt request.

6.7 LCD Display

6.7.1 Function Details

1. Display Time

[Syntax]

void DisplayTimeDate(stc_rtc_time_t *pstcTimeDate)

Parameters	In/Out	Type	Description
pstcTimeDate	In	stc_rtc_time_t *	The pointer of the time variable.

[Description]

Display current time on LCD.

2. Display Number

[Syntax]

void DisplayPulseNum(uint32_t u32Num)

Parameters	In/Out	Type	Description
u32Num	In	uint32_t	The number to display.

[Description]

Display a number on LCD.

6.8 VBAT Switch Sample

6.8.1 Function Details

1. RT Interrupt Handle

[Syntax]

void RtTrigIntHandler(void)

Parameters	In/Out	Type	Description
void	/	/	/

[Description]

Reload Timer interrupt handle. The function is for timing of 5 seconds. When times out, the system will return to the power consume consumption mode.

2. Initialize BT Module

[Syntax]

void InitBt(void)

Parameters	In/Out	Type	Description
void	/	/	/

[Description]

Initialize the Base Timer Module to kick start timing for 5 second. It will trigger a RT interrupt handle when times out.

3. Enter Deep RTC

[Syntax]

void EnterDeepRTC(void)

Parameters	In/Out	Type	Description
void	/	/	/

[Description]

The function enforces the system to enter Deep RTC mode. It takes some necessary steps before entering Deep RTC mode, for example, closing LCD, configuring the LVD module, and so on.

4. LVD0 IRQ Handle

[Syntax]

void LVD0_irq(void)

Parameters	In/Out	Type	Description
void	/	/	/

[Description]

Interrupt handle of LVD0 Module. LVD0 module is configured for monitoring the voltage of VCC. It is interrupted when VCC voltage changed across the detected voltage. It contains E2PROM Write and power source switch and ADC detection and so on.

5. LVD1 IRQ Handle

[Syntax]

void LVD1_irq(void)

Parameters	In/Out	Type	Description
void	/	/	/

[Description]

Interrupt handle of LVD1 Module. LVD1 module is configured for monitoring the voltage of VDD. It is interrupted when VDD voltage changed across the detected voltage. It contains E2PROM Write and power source switch and ADC detection and so on.

6. VBAT Switch Loop

[Syntax]

void Vbat_switchLoop(void)

Parameters	In/Out	Type	Description
void	/	/	/

[Description]

VBAT Switch Loop handle. It contains event process.

7. VBAT Switch Start

[Syntax]

void Vbat_switchStart(void)

Parameters	In/Out	Type	Description
void	/	/	/

[Description]

VBAT Switch Start handle. Please refer to Figure 4 1 VBAT Switch Start Process Flow.

8. VBAT Switch Initialization

[Syntax]

void Vbat_SwitchIni(boolean_t bPolSet)

Parameters	In/Out	Type	Description
bPolSet	In	boolean_t	TRUE -- set the POL bit when configuring the LVD module FALSE – don't set POL bit when configuring the LVD module

[Description]

LVD and VBAT switch operation initialization

9. ADC Polling

[Syntax]

en_result_t Main_adc_polling(stc_adc_std_t* pstcAdcStd)

Parameters	In/Out	Type	Description
pstcAdcStd	In	stc_adc_std_t*	The pointer of the structure data for storing the ADC data.
	return	en_result_t	If detect is successful.

[Description]

Kick start and detect one ADC data. It is used for detecting the voltage of the battery.

10. E2PROM Initialization of the Buffer

[Syntax]

void E2pInitialWrbuf(void)

Parameters	In/Out	Type	Description
void	/	/	/

[Description]

Initialize the E2PROM Write buffer.

11. E2PROM Buffer Additional

[Syntax]

void E2pBufAddNum(void)

Parameters	In/Out	Type	Description
void	/	/	/

[Description]

Calculate the data in u16WrBuf for E2PROM write.

12. E2PROM Write Buffer

[Syntax]

void E2pWrBuf(void)

Parameters	In/Out	Type	Description
void	/	/	/

[Description]

Write data to E2PROM.

13. E2PROM Read Buffer

[Syntax]

void E2pRdBuf(void)

Parameters	In/Out	Type	Description
void	/	/	/

[Description]

Read data from E2PROM.

14. Enable UART

[Syntax]

void Enable_UART(void)

Parameters	In/Out	Type	Description
void	/	/	/

[Description]

Configure and enable UART function.

15. Disable UART

[Syntax]

void Disable_UART(void)

Parameters	In/Out	Type	Description
void	/	/	/

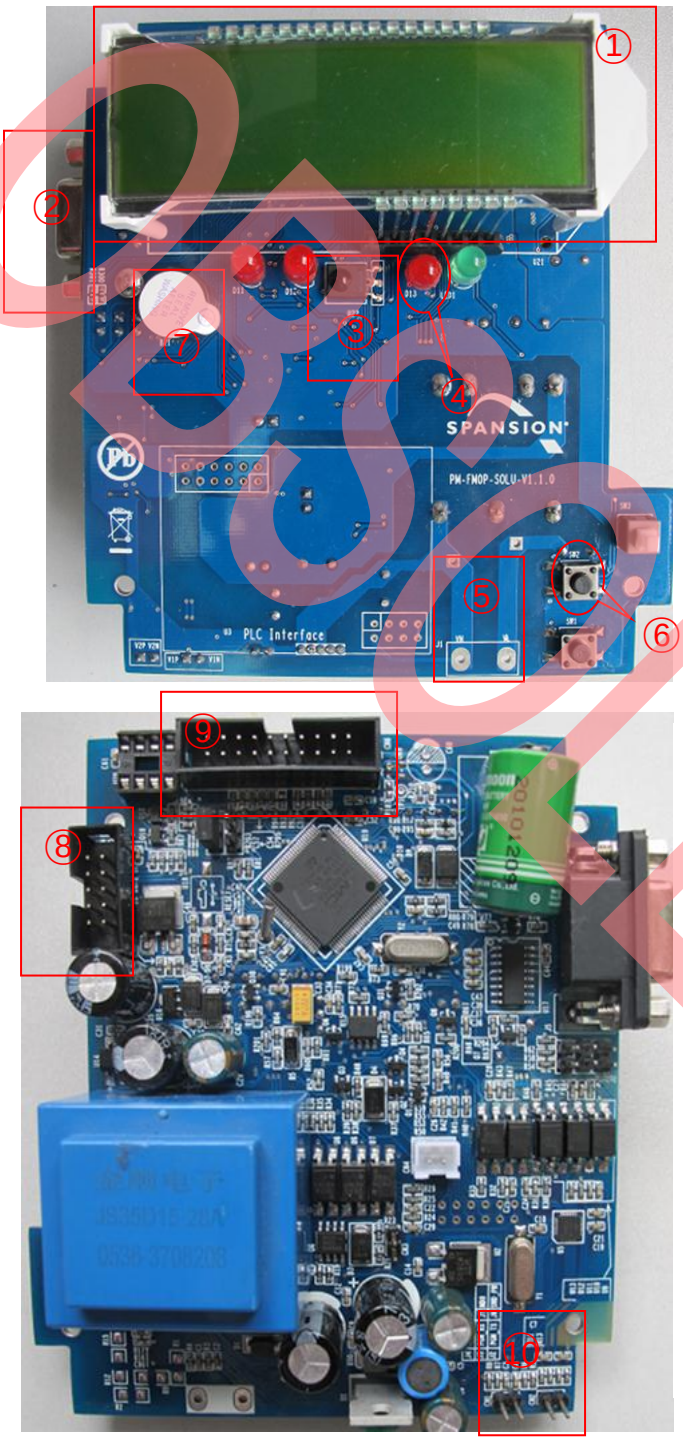
[Description]

Disable the UART function.

7 Demo System

7.1 Interface Description

Figure 7-1 Interface Description



1) LCD CL010
2) UART Interface
3) IR Incept Tube
4) IR Send Tube
5) Electric Supply Input
6) Display Key
7) Buzzer
8) IC Card Interface
9) J-Link Interface
10) Current Sample input

7.2 Jumper Description

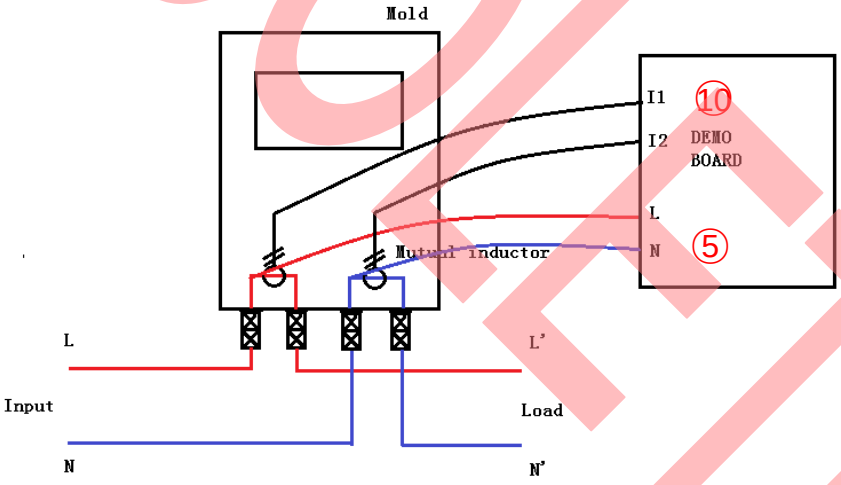
Table 7-1 Jumper Description

Jumper	Function	Setting
J2	MCU flash programme TX pin select	Open: Disable UART communication mode
		Close: Enable UART communication mode
J3	MCU flash programme RX pin select	Open: Disable UART communication mode
		Close: Enable UART communication mode
J4	MCU flash communication select	Open: Disable UART communication mode
		Close: Enable UART communication mode
J7	MD0 configuration of FM0P S6E100X	Open: User mode
		Close: Flash programming mode
J8	J-Link power supply select	Open: Disable power supply of J-link
		Close: Enable power supply of J-link

7.3 Connection Method

When connecting the DEMO board with electric supply, please refer to Figure 7-2 Connection for Power Meter. Two current samples and the VL and VN are the input of the system board.

Figure 7-2 Connection for Power Meter



8 Document History

Document Title: AN205535 - FM0+ S6E100X Power Meter Demo Board

Document Number: 002-05535

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	–	MIWA	07/22/2015	Initial Release.
*A	5031186	MIWA	12/02/2015	Migrated Spansion Application Note S6E100X_AN710-00010-1v0-E to Cypress format.
*B	5731032	HUAL	05/23/2017	Obsolete the document.

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

Automotive	cypress.com/go/automotive
Clocks & Buffers	cypress.com/go/clocks
Interface	cypress.com/go/interface
Lighting & Power Control	cypress.com/go/powerpsoc
Memory	cypress.com/go/memory
PSoC	cypress.com/go/psoc
Touch Sensing	cypress.com/go/touch
USB Controllers	cypress.com/go/usb
Wireless/RF	cypress.com/go/wireless

MMIC Solutions

http://www.spansion.com/Products/Pages/default.aspx

Cypress Developer Community

Community | Forums | Blogs | Video | Training

Technical Support

cypress.com/go/support

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
Phone : 408-943-2600
Fax : 408-943-4730
Website : www.cypress.com