Application Report
SLAA203B - June 2005

Implementing An Electronic Watt-Hour Meter With The MSP430FE42x Devices

Stefan Schauer MSP430

ABSTRACT

This report shows how to implement an electronic watt-hour meter with the MSP430FE42x devices. It contains some guidelines and recommendations for the usage of the MSP430FE42x and shows a reference board including software demos.

Contents

List of Figures

1 Block Diagram for the Connection of a Shunt for Single Phase, 2-Wire 4

List of Tables

1 Introduction

This report shows a hardware reference design and software routines for implementing an electronic electricity meter with the MSP430FE42x devices. It is intended to be used as a supplement to the ESP430CE1 user's guide that describes the ESP430CE1 module.

The MSP430FE42x with the ESP430CE1 embedded signal processing for single-phase energy metering with integrated analog front-end and temperature sensor has been specifically developed for energy metering applications. The ESP430CE1 does most of the work for the energy measurement automatically without needing resources of the main CPU. This keeps the main CPU free for other tasks like communication. The ESP430CE1 offers wide flexibility for current sensors, so that it is possible to use shunt, current transformers (including dc-tolerant CTs with high phase shift) or Rogowski coils without additional hardware. All parameters can be adjusted via software and the calibration parameters can be stored in the MSP430 flash memory and passed to the ESP430CE1 during the system initialization.

2 Hardware

The reference board schematic and system block diagrams are shown below in Appendix A and discussed in the following sections. The reference board can be used with either current transformers or shunts and can be configured for a variety of configurations. The following block diagrams show the different sensor connections and configurations. A similar demo board is available for purchase from Softbaugh, part number DE427. Softbaugh can be contacted at www.softbaugh.com.

See the schematic in Appendix A for the V1, I1, and I2 channel connections for the reference board.

2.1 Getting Started with the Reference Board

The following sections will describe how to connect the reference board with each sensor. Also described are any software changes required based on the requirements of the system and the sensor(s). A flow for download of the software to the MSP430FE427 and an initial calibration follows the setup.

1. Connect the Board:

Connect the line voltage to the voltage sensor input and connect the current sensor to the current sensor input as shown in Appendix A. For initial testing an external supply can be used to power the board in the case there is an issue with the line voltage connections. If using an external line-independent supply, connect a DC power supply (using Galvanic isolation) to the external power supply input as shown in Appendix A. The supply voltage should be approximately 5V/10mA (current may be higher if additional external components are connected). Set the power supply jumper for an external DC supply as shown in Appendix A.

2. Adjust meter settings using the provided Parameter Spreadsheet (FE427_Setting.xls)

The output header file will be created in the same directory as the spreadsheet.

2.2 Shunt as Current Sensor

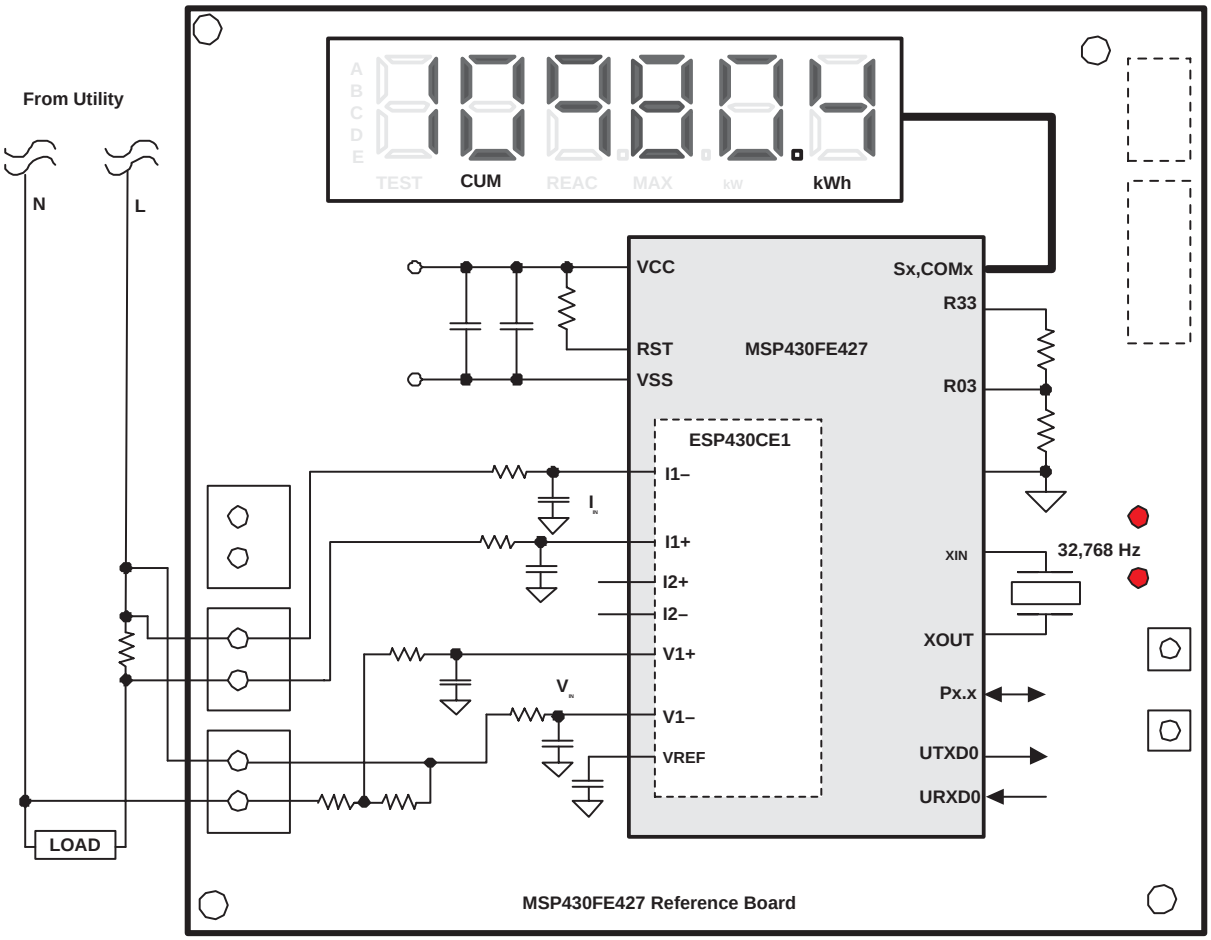


Figure 1. Block Diagram for the Connection of a Shunt for Single Phase, 2-Wire

2.3 CT as Current Sensor

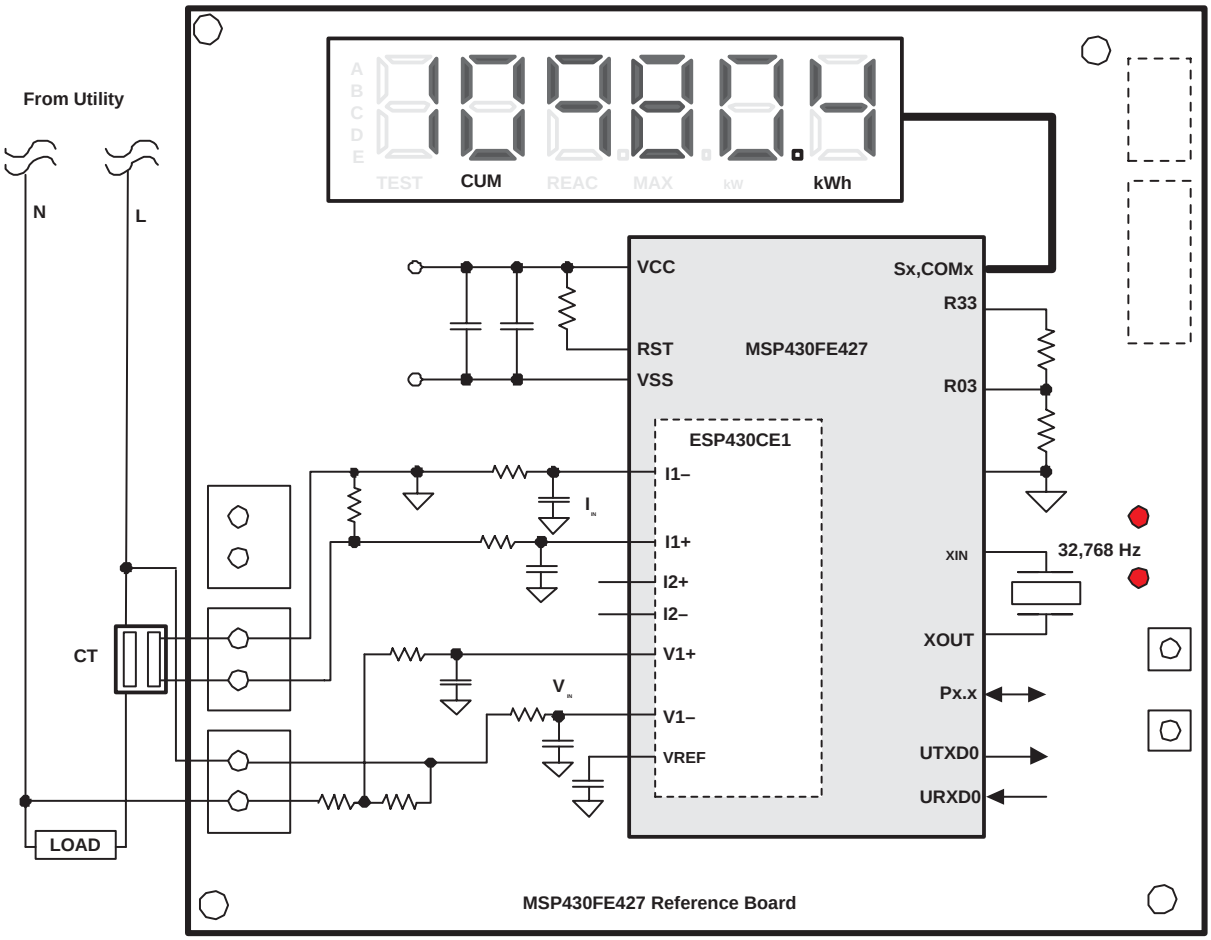


Figure 2. Block Diagram for the Connection of a CT for Single Phase, 2-Wire

2.4 CT and Shunt as Current Sensor for Tamper Detection

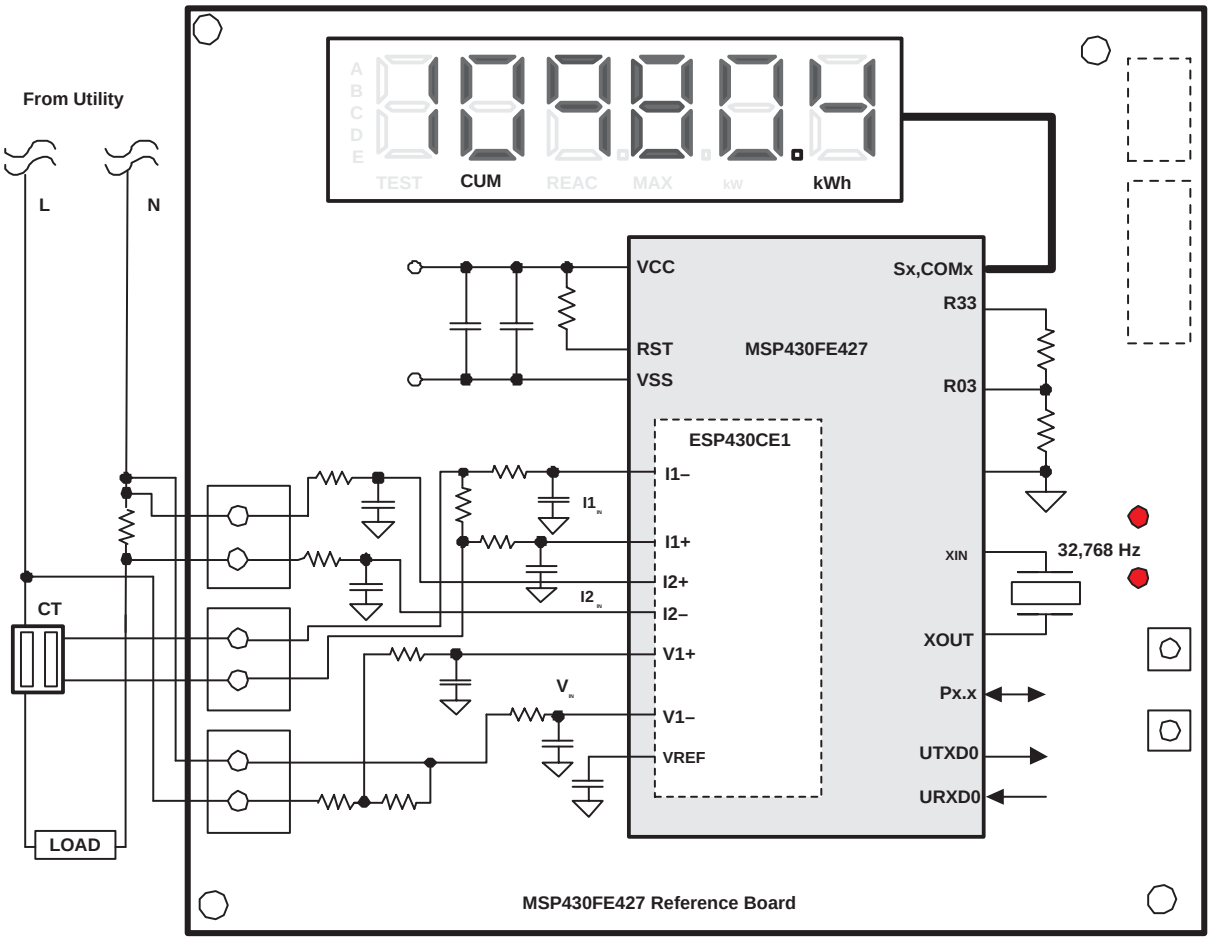


Figure 3. Block Diagram for the Connection of a Shunt and CT With Tamper Detection for Single Phase, 2-Wire

2.5 CT for the US 1-Phase 3-Wire E-Meter Solution

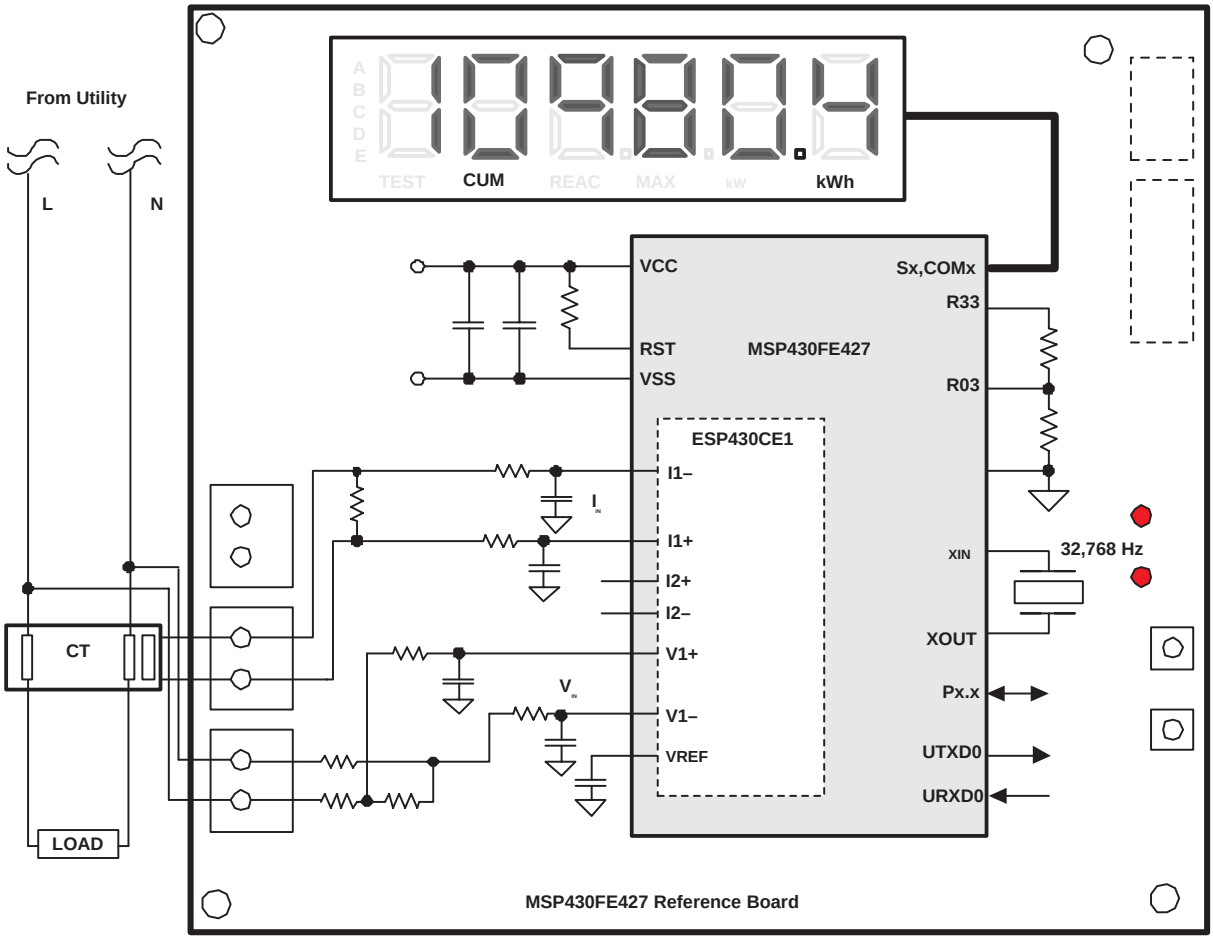


Figure 4. Block Diagram for the ANSI 1-Phase 3-Wire E–Meter Solution

2.6 Voltage Input Connections

The PCB is assembled with a voltage divider for an input voltage of 230 V (RMS). Also the protection circuit is designed for this input voltage.

The capacitor power supply is able to provide a current of ~4 mA. Care should be taken to not exceed this capacity. For example, a low-current LED was used on the reference design to help meet this requirement.

2.7 Current Input Connections

For the current path there is an SMD resistor footprint for the burden resistor of a current transformer (CT) but this resistor is not assembled.

NOTE: The burden resistor for a CT is **not** assembled. If a CT is connected, its burden resistor must be assembled. Otherwise, the MSP430 will be damaged.

2.8 Anti-Aliasing Filter

The recommended anti-aliasing filter is a 1-kΩ resistor in series to the ADC input and a 33-nF capacitor to the analog ground. To avoid external influences, it is recommended to have a filter in the positive and negative input channel of each used ADC input.

2.9 Unused ADC Inputs

Unused ADC inputs should be unconnected.

3 Calculation of the ESP430CE1 Meter Constants

The meter requires constants based on the voltage and current transformers / shunts. The following sections show how to calculate the ESP430CE1 meter constants.

3.1 Voltage Ratio

The voltage ratio, which calculates the real voltage from the ESP430CE1 voltage value, is calculated as follows:

V(inp.max) = VoltageGain x V (Line, Nom.) x sqrt(2) x R2 /(R1 + R2)

kV1 = Voltage (Line, nominal) x 2 x sqrt(2) / (2^{15} x (1- (Vref – V(inp.max) x 2)/Vref))

3.2 Current Ratio for Shunt

The current ratio for a shunt, which calculates the real current from the ESP430CE1 current value, is calculated as follows:

V(I, inp.max) = CurrentGain x Imax x R(Shunt) x sqrt(2)

kI1 = Current (Line, nominal) x 2 x sqrt(2) / (2^{15} x (1- (Vref – V(I, inp.max) x 2)/Vref))

3.3 Current Ratio for Current Transformer

The current ratio for a CT which calculates the real current from the ESP430CE1 current value, is calculated as follows:

V(I, inp.max) = CurrentGain x Imax / CTRatio x R(Burden) x sqrt(2)

kI1 = Current (Line, nominal) x 2 x sqrt(2) / (2^{15} x (1- (Vref – V(I, inp.max) x 2)/Vref))

3.4 Interrupt Level for Energy

The energy consumption interrupt level of the ESP430CE1 is calculated as follows:

InterruptLevel = Pulses/kWh x (1000 / 3600) x fADC / (kV1 x kI1 x 4096)

Pulses/kWh defines how many interrupts per consumed kWh should be generated.

4 Meter Calibration

Calibration of MSP430–based electricity meters with conventional calibration equipment using the same procedures as for Ferraris meters is possible, but not cost effective. The processing power of the MSP430 allows other methods shown below.

A basis calibration could be initiated with the UART c0 command. The execution of this command requires the input values which are defined in the parameter.h file:

- calVoltage
- calCurrent
- calPhi
- calCosPhi
- calFreq

The phase shift calibration between voltage and current should be done with an cos Phi of 0.5 as at this point the error of the phase shift resulting from the sensors does generate much higher errors and therefore a higher accuracy could be reached with less effort.

For the calibration of electricity meters it is necessary to separate the voltage and current path of the meters. This allows the calibration with low energy losses and a defined value for the voltage, current, and phase shift. Figure 5 shows the terminals of an electricity meter with the calibrating terminal switch open for the calibration.

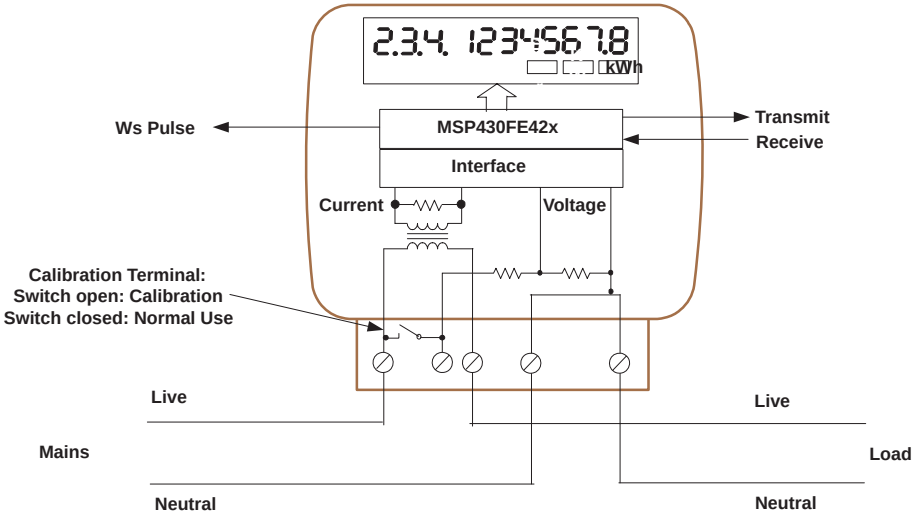


Figure 5. MSP430 Electronic Electricity Meter With External Terminals

4.1 Calibration With Continuous Measurement

The ESP430CE1 operation mode is set to electricity meter mode via the mailbox with the SetMode Command the CPU initializes the ESP430CE1 for normal measurement. The energy values written after each mains period to the value registers ActEnSPer1 (and ActEnSPer2 for two sensor systems) are converted by the CPU into a proportional, constant output frequency having only the information of the mean value of the measured energy. The Timer_A of the CPU may be used for the generation of an energy proportional output frequency.

The calibration sequence is:

- The CPU sets the flags Curr_I1, Curr_I2 in control register 0 according to the measurement mode of the ESP430CE1.
- The parameter registers are initialized for the load point to be measured. This is made via the mailbox with the SET_PARAM Command.
- The ESP430CE1 mode is set to electricity meter mode via the mailbox with the mSet_Mode command
- The first result in addresses ActEnSPer1 (and ActEnSPer2 if current path 2 is enabled) is not used, due to the unknown start point inside the mains period.
- The next results in addresses ActEnSPer1 (and ActEnSPer2 if current path 2 is enabled) are valid and are used for calculations.
- The flag St_ZCld in status register 0 indicates, that with the next available samples (flag St_NEVal is set) new results for the last mains period are available in addresses ActEnSPer1 and ActEnSPer2.
- The CPU resets the flag St_NEVal with the Mailbox command mCLR_EVENT and processes the read results with the equations below.
- The last four steps are repeated if necessary (e.g. for a summing of the results of more than one mains period).

The above steps are repeated for the second calibration point.

Calibration of two current sensors should be done independently. The meter should be calibrated for one of the current sensors, while the current through the other is zero. A second calibration should then be run for the second current sensor, while the current through the first sensor is zero.

4.1.1 Formulas

Calibration is made for a single mains period (or during n_{per} mains periods) with the two currents I_{1HI} and I_{1LO}. The nominal energy results for the two calibration points are:

$$n_{HIcalc} = C_{Z1} \times I_{1HI} \times V_1 \times \cos\varphi_1 \times \frac{n_{per}}{f_{mains}} \times \frac{f_{ADC}}{4096} \quad [\text{steps}^2]$$

$$n_{LOcalc} = C_{Z1} \times I_{1LO} \times V_1 \times \cos\varphi_1 \times \frac{n_{per}}{f_{mains}} \times \frac{f_{ADC}}{4096} \quad [\text{steps}^2]$$

The resulting values for the slope and offset are:

Slope: $$\text{GainCorr1} = \frac{n_{HIcalc} - n_{LOcalc}}{n_{HImeas} - n_{LOmeas}} \times 2^{14}$$

Offset: $$\text{POffset1} = \frac{n_{HImeas} \times n_{LOcalc} - n_{LOmeas} \times n_{HIcalc}}{n_{HImeas} - n_{LOmeas}} \times \frac{f_{mains}}{n_{per}} \times \frac{4096}{f_{ADC}}$$

Where:	f_{mains}	Mains frequency	[Hz]
	f_{ADC}	ADC repetition frequency (4096 Hz normally)	[Hz]
	n_{per}	Number of mains periods used for calibration	[1]
	n_{HIcalc}	Calculated energy at the high current calibration point	[steps^2]
	n_{HImeas}	Measured energy at the high current calibration point	[steps^2]
	n_{LOcalc}	Calculated energy at the low current calibration point	[steps^2]
	n_{LOmeas}	Measured energy at the low current calibration point	[steps^2]

4.1.2 Calibration Example

The I_1 path of the electricity meter shown in figure 1 is calibrated with the following values:

V_1 = 230 V $_{1HI}$ = 20 A $_{1LO}$ = 1 A $\cos\varphi_1$ = 1 n_{per} = 1

f_{ADC} = 2048 Hz f_{mains} = 50 Hz

The nominal measurement results n_{HIcalc} and n_{LOcalc} are:

$$n_{HIcalc} = C_{Z1} \times I_{1HI} \times V_1 \times \cos\varphi_1 \times \frac{n_{per}}{f_{mains}} \times \frac{f_{ADC}}{4096} = 29{,}322.80806 \times 20 \times 230 \times 1 \times \frac{1}{50} \times \frac{2048}{4096}$$

$$n_{HIcalc} = 1{,}348{,}849.171 = 14{,}94\text{F1h}$$ [Steps^2]

$$n_{LOcalc} = C_{Z1} \times I_{1LO} \times V_1 \times \cos\varphi_1 \times \frac{n_{per}}{f_{mains}} \times \frac{f_{ADC}}{4096} = 29{,}322.80806 \times 1 \times 230 \times 1 \times \frac{1}{50} \times \frac{2048}{4096}$$

$$n_{LOcalc} = 67{,}442.458 = 1{,}0772\text{h}$$

The measurement results for the two calibration points I_{1LO} and I_{1HI} are:

$n_{1Himeas}$ = 14,6040h –1 % error compared to $n_{1Hicalc}$ = 14,94F1h)

$n_{1Lomeas}$ = 1,0CB7h (+2 % error compared to $n_{1Localc}$ = 1,0772h)

With the above four results the rounded slope in address GainCorr1 becomes:

$$\text{GainCorr1} = \frac{n_{HIcalc} - n_{LOcalc}}{n_{HImeas} - n_{LOmeas}} \times 2^{14} = \frac{14{,}94\text{F1h} - 1{,}0772\text{h}}{14{,}6040\text{h} - 1{,}0\text{CB7h}} \times 2^{14} = 1.01171 \times 2^{14} = 40\text{C0h}$$

The offset in addresses POffset1 and POffset1+2 becomes:

$$\text{POffset1} = \frac{n_{HImeas} \times n_{LOcalc} - n_{LOmeas} \times n_{HIcalc}}{n_{HImeas} - n_{LOmeas}} \times \frac{f_{mains}}{n_{per}} \times \frac{4096}{f_{ADC}}$$

$$\text{POffset1} = \frac{14{,}6040\text{h} \times 1{,}0772\text{h} - 1{,}0\text{CB7h} \times 14{,}94\text{F1h}}{14{,}6040\text{h} - 1{,}0\text{CB7h}} \times \frac{50 \times 4096}{1 \times 2048} = -215{,}489 = \text{FFFC, B63Fh}$$

NOTE: the calculated value for (POffset1) is the offset for each product $N_{V1} \times N_{I1}$, thus the relatively high value.

If the measured calibration points are corrected with the calculated slope and offset:

$$n_{corr} = \left(n_{meas} \times (\text{GainCorr1})\right) \times 2^{-14} + (\text{POffset1}) \times \frac{n_{per}}{f_{mains}} \times \frac{f_{ADC}}{4096}$$

$$n_{HIcorr} = 14{,}6040\text{h} \times 40\text{C0h} \times 2^{-14} + \text{FFFC, B63Fh} \times \frac{1 \times 2048}{50 \times 4096} = 1{,}348{,}890 = 14{,}951\text{Ah}$$

$$n_{LOcorr} = 1{,}0\text{CB7h} \times 40\text{C0h} \times 2^{-14} + \text{FFFC, B63Fh} \times \frac{1 \times 2048}{50 \times 4096} = 67{,}441 = 1{,}0771\text{h}$$

The resulting error for both corrections is +3.1E–5 which is 31 ppm.

4.2 Calibration With a Host Computer

Figure 6 shows a possible calibration environment for electronic electricity meters. The host computer is connected to the meters via the USART0 communication port running in SPI or UART mode. All necessary calibration calculations are made by the host; the MSP430 in each meter only stores the received correction values in its information memory or an external EEPROM.

The host controls the calibration equipment containing a voltage generator, a current generator and a phase shifter via the host interface. The host reads out the accumulated results of the multiplying of voltage and current ADC steps (or counts the Ws pulses of each electricity meter) and compares the equivalent energy with the energy equivalent to the reference pulses coming from a reference meter, which is part of the calibration equipment. The host calculates the meter error out of the energy amounts for (e.g., 100% Inom) or two load points (e.g., 100% Inom and Imax). With these errors the slope and offset of the load characteristic can be calculated individually and sent to the MSP430s in each meter.

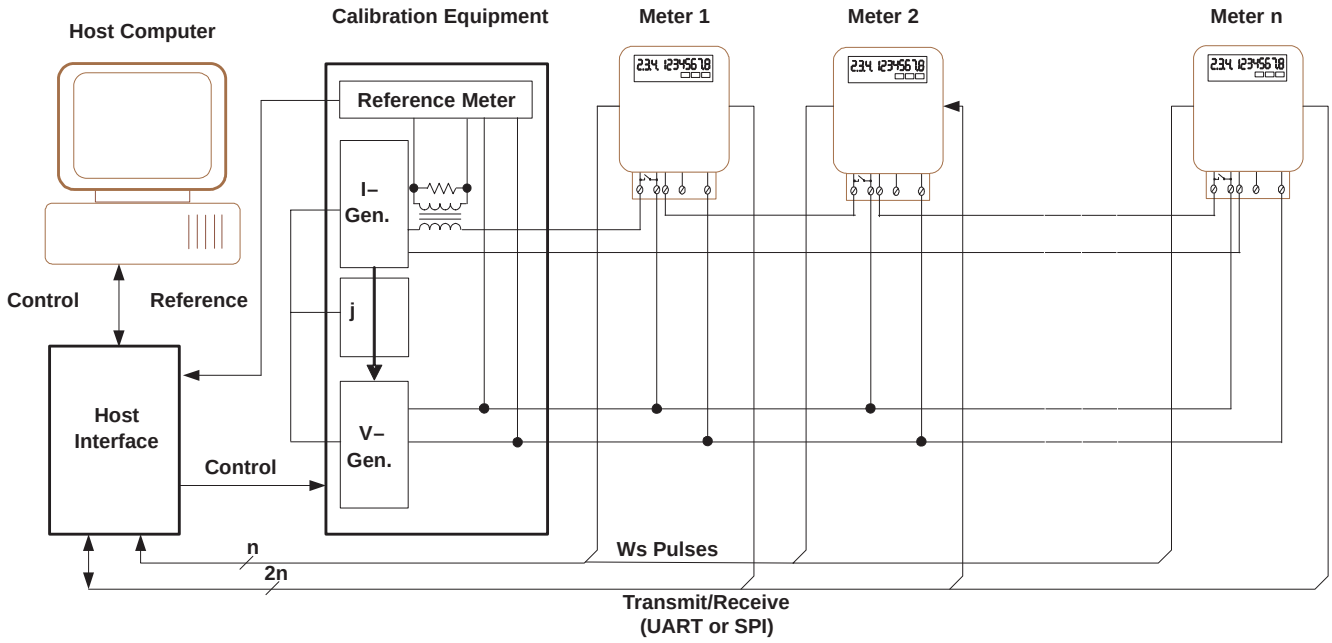


Figure 6. Electricity Meter Calibration With a Host Computer

The formulas for the calculation of the calibration values are shown in the ESP430CE1 module user's guide.

4.3 Self Calibration

Another calibration method uses the processing capability of each MSP430 in each meter. The main advantage of this calibration method is the simplicity: no wiring for the information transfer is necessary (see Figure 7). The error correction equations – used by the electricity meter under test – are the same ones as shown in the *Calibration With Continuous Measurement* section.

- The meters to be calibrated are put into the calibration mode via a hidden switch, the UART, a key or an input pulse, etc.
- The host switches on the calibration equipment and transmits a defined amount of energy – measured with the reference meter – to the electricity meters being calibrated.
- The electricity meters measure the transmitted energy and calculate the energy value WEM1 for the 100% Inom current.
- After this transmission of energy the calibration equipment is switched off (I = 0, U = 0). This allows the electricity meters to calculate or measure the ADC offsets if necessary.
- The host switches on the calibration equipment again and transmits a defined amount of energy (e.g. 5% Inom, 100% Vnom, cosφ = 1) to the electricity meters. After this transmission of energy the calibration equipment is switched off (I = 0, V = 0).
- The electricity meters measure the transmitted energy and calculate the energy value WEM0 for the 5% Inom current.
- With the two faulty energy values WEM1 and WEM0 found for the 100% and 5% Inom loading conditions the electricity meters calculate their individual offsets and slopes.

- A simple visual final test is possible after the calibration:
  - The electricity meters reset their display to zero
  - The calibration equipment transmits a precisely defined energy profile (different percentages of *Inom*, *Vnom* and two values of cosφ) to the electricity meters.
  - A visual check is made if the meters display the known amount of energy.
  - The MSP430 indicates in the LCD if the calculated slopes and offsets are within worst case limits.

EXAMPLE: the energy profile for the final check consists of

- 10,000 *Ws* (100% *Inom*, 100% *Vnom*, cosφ = 1)
- 5,000 *Ws* (100% *Inom*, 100% *Vnom*, cosφ = 0.5)

The calibrated electricity meters must display the number 15,900 *Ws* ± accuracy, otherwise the calibration failed.

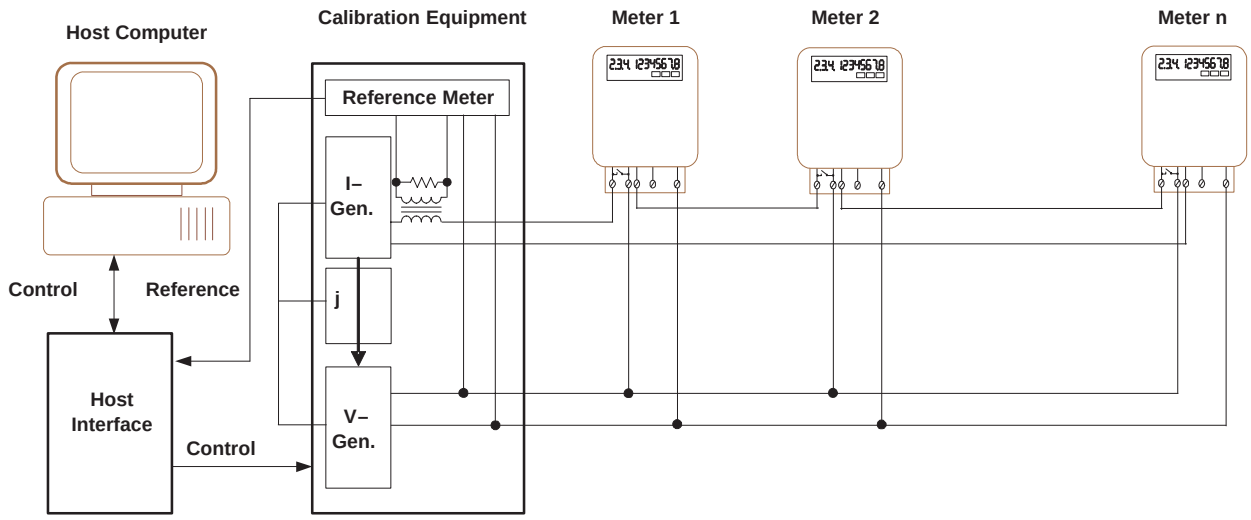


Figure 7. Self-Calibration for Electricity Meters

5 Capacitor Power Supply

Figure 8 shows a capacitor power supply for a single output voltage *Vcc* = +3 V. If the output current is not sufficient, an NPN output buffer may be used.

The design equations for the power supplies below are given in SLAA024, section *3.8.3.2 Capacitor Power Supplies*. This chapter also describes other kinds of power supplies with their design equations.

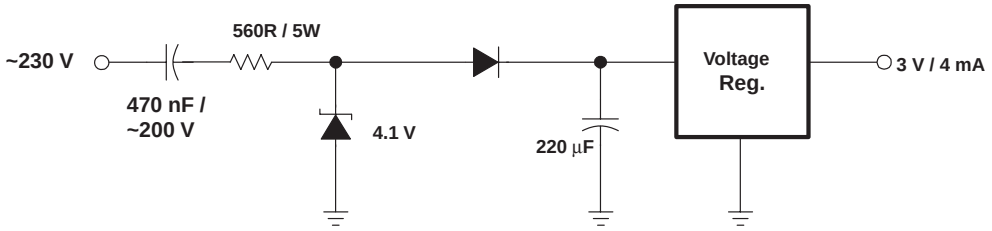


Figure 8. Capacitor Supply

5.1 Power Line Voltage On/Off Detection

Because the ESP430CE1 voltage drop detection is combined with the line cycle counter, the voltage drop detection of the ESP430CE1 does not function if the voltage drops to 0 V. To detect this condition, the VRMS voltage can be observed in dedicated time intervals or an external circuit may be used to detect when the power line is off. If an external circuit is used, the ESP430CE1 module can be switched off to conserve power.

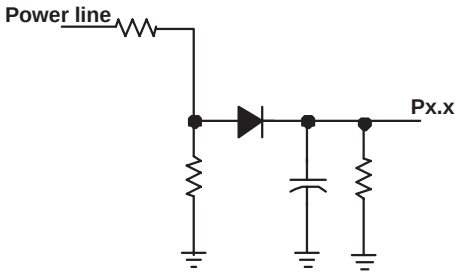


Figure 9. Power Detection

6 Layout Recommendations

6.1 Grounding

Good circuit-board layout is important for high resolution ADC systems. Following are some basic layout guidelines.

1. Use of a separate analog and digital ground plane wherever possible:
2. Thick traces from the battery to the DV_{SS}, AV_{SS}, DV_{CC}, and AV_{CC} terminals.
3. The decoupling capacitor at the AV_{SS} terminal is a star point for all analog ground connections. The decoupling capacitor at the DV_{SS} terminal is a star point for all digital ground connections.
4. The connections of the capacitor Cb are the star point of the complete system. This is due to the low impedance of this capacitor.
5. The AV_{SS} and DV_{SS} terminals must be connected together externally.
6. The AV_{CC} and DV_{CC} terminals must be connected together externally.
7. Battery and storage capacitor Cb should be close together. Two capacitors are connected across the digital (Cd) and the analog (Ca) supply terminals.
8. The coil L could be used to keep disturbances introduced from the digital supply away from the analog supply voltage. It is also possible to use a resistor for this. The coil brings additional advantage in filtering high frequency signals.
9. If a metal case is used around the printed circuit board containing the MSP430 then it should be connected to the ground potential (0 V) of the board.

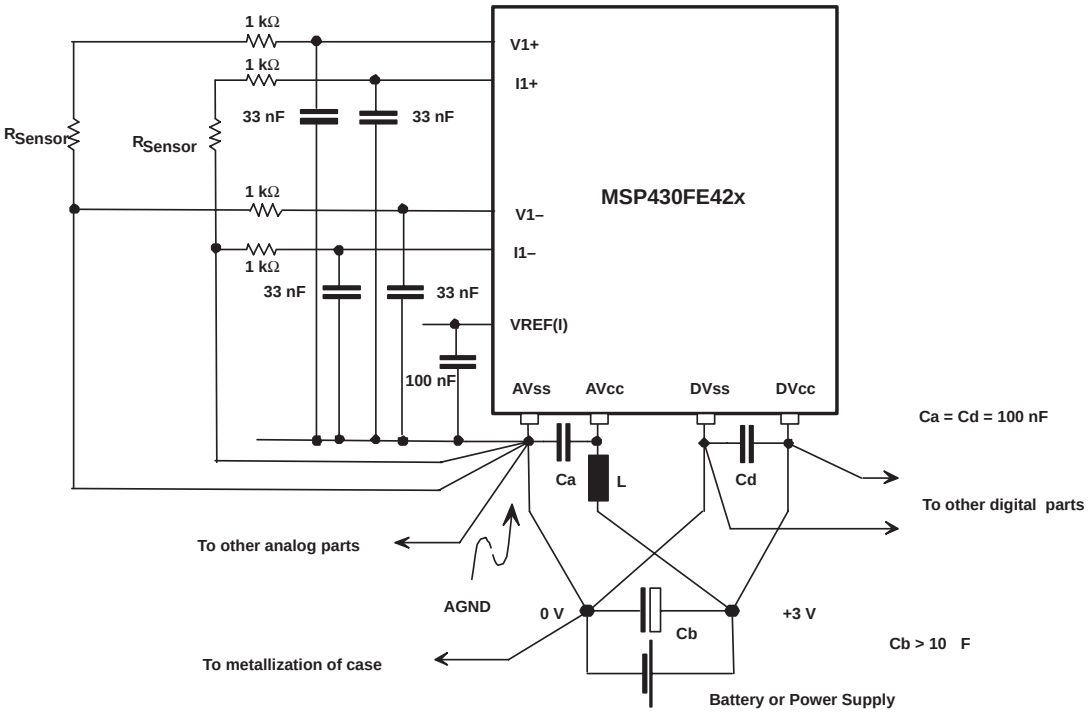


Figure 10. Analog-to-Digital Converter Grounding

6.2 EMI Sensitivity

Figure 11 shows a simplified way a routing that is not optimal: the gray areas receive EMI from external sources. For a minimum influence coming from external sources, these areas must be as small as possible.

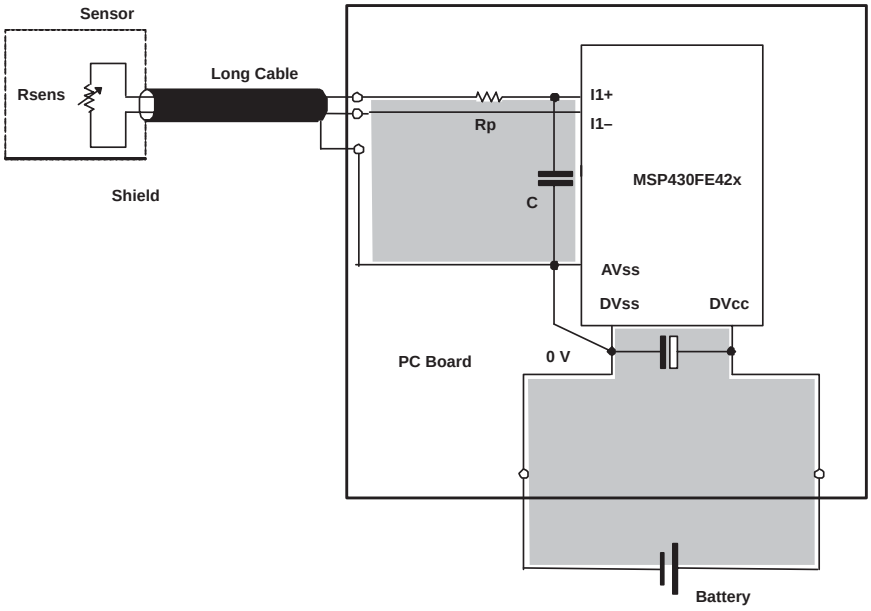


Figure 11. Routing that is Sensitive to External EMIT

Figure 12 shows an optimized routing. The areas that receive noise have a minimum loop area.

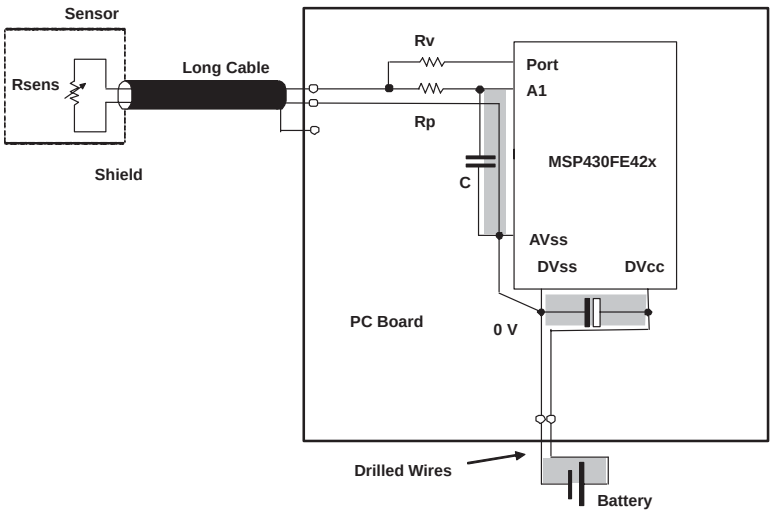


Figure 12. Routing for Minimum EMI Sensitivity

7 Demo Software

7.1 Analog Front-End Initialization

While the ESP430CE1 is off, the MSP430 CPU has access to the SD16 module. During this time, the MSP430 CPU should do the initialization of the analog front-end. This consists of setting the gain, oversampling ration, and clock source for the SD16 as shown in the function init_analog_front_end in the emeter.c source file.

7.2 E-Meter Initialization

The ESP430CE1 must be configured before use. A configuration example is shown in the subroutine init_esp_parameter in the source file emeter.c.

7.3 Demo 1 Software

Demo 1 is as a simple demo that initializes the ESP430CE1 for energy measurement and outputs the data on the display. The LED is also pulsed. This demo can be use with IAR Kickstart development tool. The files and contents of the Demo 1 software are:

File	Contents and Function
Main.c	Control of system Initialization and call of the functions for the display update once requested in the interrupt service routines: Init FLL and System Clock, Init Basic Timer and Real time Clock, Init LCD, Init analog front end, Init ESP430CE1 Parameters, Start Measurement
FET4xx_RTCwLCD.s43/.c	Basic routines for LCD & Real Time Clock with Basic Timer ISR.
FET4xx_RTCwLCD_sb.s43/.C	Basic routines for LCD and Real Time Clock with Basic Timer ISR for SoftBaugh FE development board
Display.c	High level routines for LCD
FLL.c	Routines to setup FLL and clock system
PortFunc.c	Interrupt service routine for Port1
TimerA.c	Initialization routine and interrupt service routine for Timer_A. The Timer_A is used to generate a pulse without flicker.
EMeter.c	EMeter.c contains the initialization routine for the analog front end, the ESP430CE1, and the interrupt service routine for the ESP430CE1.
FE427_Measure_v3.ewp/.eww	Project Files for IAR Workbench Version 3
FE427_Measure.ewp/.eww	Project Files for IAR Workbench Version 2
FE427_Measure.hzp/.hzs	Project Files for Rowley CrossStudio
bin2bcd16.s43/.c	Converts integer to BCD
bin2bcd32.s43/.c	Converts long to BCD
LCDdec16.s43/.c	Support to convert signed integer values for display to the LCD
LCDdecu16.s43/.c	Support to convert unsigned integer values for display to the LCD
LCDdec32.s43/.c	Support to convert signed long values for display to the LCD
LCDdecu32.s43/.c	Support to convert unsigned long values for display to the LCD
emeter_toolkit.h	DIsplay function support header file
device.h	Device-specific information header file
esp_Parameter.h	Meter parameter definitions for the ESP module (generated by FE427_Setting.xls)
Parameter.h	Global parameter definitions
Subroutines.h	Subroutine definitions
portfunc.c	Port handling source file
unused_Int.s43	Unused ISR support source file

The Demo software flow is shown in Figure 13.

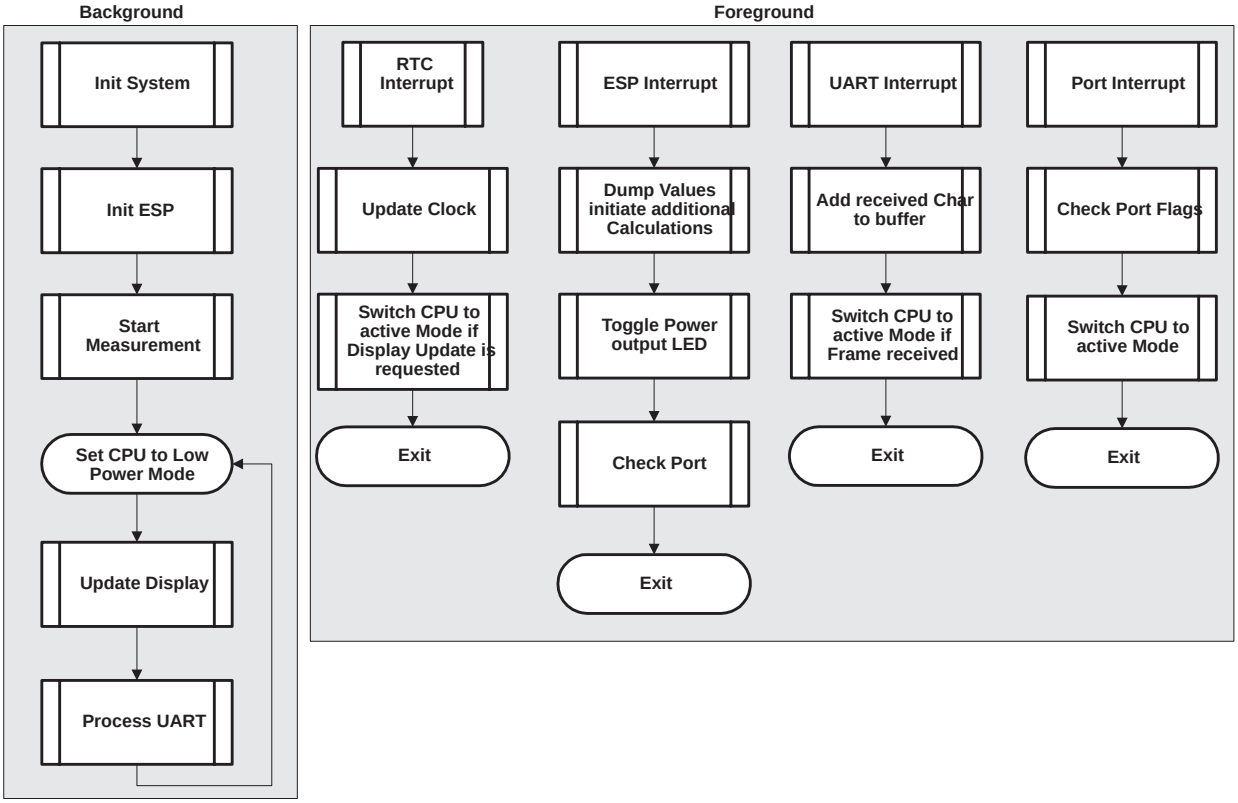


Figure 13. Software Flow

7.4 Energy Pulse Generation

A pulse can be output to indicate a specific energy level. Two methods can be used for the generation of the energy pulse output signal.

7.4.1 Direct Output With Interrupt Level

The first method is the direct interrupt source of the ESP430 module with the Interrupt level set to a specific energy level. The implementation is simple, does not need any additional hardware or software resources but, as the energy is the accumulation of a sine wave; this signal could have some jitter.

This method is active if the following are defined:

```
//#define TIMERA_PULSE_OUTPUT
```

7.4.2 Timer_A Output

The second method uses the Timer_A as a constant time basis to directly remove the jitter of the interrupt level method. This method is sufficient for pulse frequencies up to ~30 Hz. Here, the following setting could be used in the parameter.h file.

```
#define  TimerAClock          TASSEL_1    /* ACLK = 32kHz
#define  TACLOCK              32768ul
#define  CLOCKSPERPERIOD     (TACLOCK/defSET_NOMFREQ)
```

This is active if the following is defined:

```
#define TIMERA_PULSE_OUTPUT
```

7.5 Temperature Compensation for CT Phase Shift

The ESP430 can measure device temperature using the integrated temperature sensor. The MSP430 CPU can update the ESP430 parameters accordingly without stopping the operation of the ESP. This can be used to provide a change in the Phase Shift Parameter for a current transformer based on the measured temperature using a coefficient table.

This functionality is enabled in the demo software by defining:

```
#define with TempCorrection
```

in the parameter.h file and setting the correct values of the correction either with the function

```
#define PHI_Temp_Ratio 0.01  /* 0.01 Degree Phase shift per 1 Degree C */
                             /* From CT Datasheet */
                             /* if this is not linear it could be modified or
                             /* entered directly in the emeter.c file */
```

or by modifying the function in the emeter.c file.

7.6 Controls

The two buttons are used for following functions:

- S_A: Switch the ESP430CE1 off and set the MSP430 into low power mode. The real time clock continues to run.
- S_B: Toggle through the display modes

7.6.1 Parameter.h file

All configuration settings are done within the parameter.h file including the settings for the:

- Pulse output level
- Voltage and current ratio
- Configuration settings for the ESP430CE1

The #defines for withUARTComm, withCalibration, withDisplay allow scaling the code for different function and sizes. The code uses floating point functions for these functions and including one them will increase the code size.

Commenting or uncommenting of the following line in the parameter.h file configures the software to use a Shunt or CT on the I1 input:

```
#define shunt
```

To do a precalculation of the parameters, the following formulas can be used:

- defVRatio = kV1 / 1000 (for kV1 see the *Voltage Ratio* section)
- defIRatio = kI1 / 1000 (for kI1 see the *Current Ratio For Shunt* section)
- defEnergyRatio = (defVRatio x defIRatio)

For an easier calculation of the main parameters defined the parameter.h file, the Excel sheet FE427_Settings.xls could be used. After entering the required information into the white fields, the parameters are calculated and displayed. By clicking the *Save Parameter to File* button, the parameters are saved into the file ‘Test_Parameter.h’.

This file with the calculated parameters will be included into the source. The following section describes the usage of the FE427_Settings.xls file.

See the Excel sheet which is included with the source code.

7.6.2 Using FE427_Settings.xls

Enter values into the white and blue cells as needed to meet application–specific requirements for the meter. Blue cells are drop down boxes and contain fixed selections while yellow cells provide information and intermediate calculation results. Green cells provide the calculated parameters for use by the ESP430 based on the application–specific user inputs for the e–meter design.

Clicking the 'Save Parameters to File' button saves the parameters in green cells to the file 'ESP_Parameter.h'. This header file must be copied into the source file folder of the applicable MSP430FE42x demo project.

NOTE: If the Calculation values in the Hex columns are not updated in the spreadsheet, please verify that the "Analysis ToolPak" is installed and enabled under the Tools | Add–Ins... menu.

7.7 Demo 2

Demo 2 is setup as a complex application including UART communication and some auto-calibration routines that store the parameters back into the flash memory. For the energy calculation, the energy values reported from the ESP430CE1 are used instead of the interrupt level function. The initialization of the ESP430CE1, the data output on the display, and the LED usage is included as in the Demo 1. This demo is too large to be used with IAR Kickstart.

Demo 2 contains all the files in Demo 1 plus:

File	Contents and Function
UART.c	Interrupt handler for UART receive
Comms_UART.c	UART communication routines: – Init UART – UART Send Functions – UART Receive Function: Process_UART (This routine processes a received UART Command).
SendData.c	Conversion routines for the data which should be sent by the UART
Calibration.c	Some simple calibration functions which could be used to do a basic calibration. These functions are executed by commands sent via the UART.
flash_xxxx.c	Files used to erase, write and replace information in the MSP430 flash memory space

7.7.1 UART Communication

The baud rate is 57600, 8N1. Each command should be terminated with a carriage return ‘CR’.

The ’h’ command displays the help list in the terminal window, as shown below.

```
MSP430FE427 Firmware Version: xxxx

UART Commands:
 SHxx: set hour
 SMxx: set minutes
 SSxx: set seconds
 SDxx: set day
 SOxx: set month
 SYxx: set year
 SIxx: set calibration current
 SVxx: set calibration voltage

 Dx: Set Display mode
  1: Off
  2: Time
  3: Date
  4: Voltage (V)
  5: Current (A)
  6: Peak Voltage (V)
  7: Peak Current (A)
  8: Frequency (Hz)
  9: CosPhi
  10: Temp
  11: Power (kW)
  12: Energy (kWh)

 Vx: Value of single measurement
  1: Off
  2: Time
  3: Date
  4: Voltage (V)
  5: Current (A)
  6: Peak Voltage (V)
  7: Peak Current (A)
  8: Frequency (Hz)
  9: CosPhi
  10: Temp
  11: Power (kW)
  12: Energy (kWh)

 H : Show help test
 Tx: Set test dump mode
 Qx: Query dump
 R : Reset system
 Wxx: Write message to LCD display
 Mx: Execute calibration measurement over x*50 cycles
 I : Init
```

UART Commands continued:

```
C0: auto calibration of U / I / P / Phase Shift
C1: calibration of Interrupt Level
C2: calibration of Phase correction 1
C3: calibration of Phase correction 2
C4: calibration of V1 Offset
C5: calibration of I1 Offset
C6: calibration of I2 Offset
C7: calibration of Gain correction 1
C8: calibration of Gain correction 2
C9: save settings to flash
C10: calibration of V Ratio
C11: calibration of I Ratio
C12: calibration of Energy Ratio
C13: calibration of Power1 offset
C14: calibration of Power2 offset

+xxx: inc values for calibration
-xxx: dec values for calibration
```

7.7.2 Calibration

A basic calibration can be done with the UART command ‘C0’.

The execution of this command requires the input values defined in the parameter.h file:

- calVoltage
- calCurrent
- calPhi
- calCosPhi
- calFreq

With the UART command ‘C9’ the calculated values are stored into the flash.

Calibration of Demo2 Software

The following steps detail a method for calibrating the meter implementation for the Demo2 source code. This method makes a three-point calibration...

Step 1:
– adjust the values in FE427_Settings.xls for the sensor used
– generate the esp_Parameter.h source file and add to the Demo2 project folder
– save, compile and download the software to the ’FE427

Step2:
– attach the meter to an I/V generator adjusted to the calibration values defined in FE427_Settings.xls (e.g. 240 V / 10 A / 60° Phase Shift)

Step 3:
– using the provided UART connection to a PC, open a serial communication terminal window (i.e. HyperTerminal) and validate proper operation. For example, using mode t8 – the terminal window should display the pre-calibration meter results
– enter c0[ENTER] to perform an initial calibration (not required, adjusts displayed values only)

Step 4:
– set first calibration point with 0° phase shift (.e.g. 240V / 10 A / 0° phase shift)
– enter c7[ENTER] to adjust the channel 1 current gain for zero energy measurement error (correction is done with 0.1% of measured error, e.g. +5[ENTER] corrects an error of +0.5%)

Step 5:
– set second calibration point, .e.g. 240V / 10 A / 60° phase shift
– enter c2[ENTER] to adjust the channel 1 phase correction for zero energy measurement error and for a measured cos phi of –0.5.
(correction is done with the steps of the FE427 phaseCorr register, e.g. +5[ENTER])

Step 6: (optional 3rd calibration point)
– adjust the I/V generator to a low current load, e.g 0.5 A and 0° phase shift
– enter c13[ENTER] to adjust the power offset
(correction is done with steps of the FE427 power offset register, e.g. +5[ENTER])
use energy value of the FE427 displayed in the terminal with t5 mode:
correction value = –((CurrentEnergyValue * Error%) / 4096)

Step 7:
– enter c9[ENTER] to save the values to the info memory of the FE427

Using the UART interface provided, an alternative calibration scheme can also be realized. In this case, Gain and Phase Calibration can be combined into a single test point. This process is described in the following steps.

Step 1:
– adjust the values in FE427_Settings.xls for the sensor used
– generate the esp_Parameter.h source file and add to the Demo2 project folder
– save, compile and download the software to the ’FE427

Step2:
– attach the meter to an I/V generator adjusted to the calibration values defined in FE427_Settings.xls (e.g. 240 V / 10 A / 60° Phase Shift)

Step 3:
– using the provided UART connection to a PC, open a serial communication terminal window (i.e. HyperTerminal) and validate proper operation. For example, using mode t8 – the terminal window should display the pre-calibration meter results
– enter c0[ENTER] to perform an initial calibration (not required, adjusts displayed values only)
– enter c2[ENTER] to adjust the channel 1 phase correction for zero energy measurement error and for a measured cos phi of 0.5.
(correction is done with the steps of the FE427 phaseCorr register, e.g. +5[ENTER])
– enter c7[ENTER] to adjust the channel 1 current gain for zero energy measurement error
(correction is done with 0.1% of measured error, e.g. +5[ENTER] corrects an error of +0.5%)

Step 4: (optional 3rd calibration point)
– adjust the I/V generator to a low current load, e.g 0.5 A and 0° phase shift
– enter c13[ENTER] to adjust the power offset
(correction is done with steps of the FE427 power offset register, e.g. +5[ENTER])
use energy value of the FE427 displayed in the terminal with t5 mode:
correction value = –((CurrentEnergyValue * Error%) / 4096)

Step 5:
– enter c9[ENTER] to save the values to the info memory of the FE427

8 References

1. *MSP430FE42x Data Sheet* (SLAS396)
2. *ESP430CE1 Module User’s Guide* (SLAU134)
3. *MSP430x4xx Family User’s Guide* (SLAU056)
4. *MSP430 Family Application Report Book* (SLAA024)

Appendix A Reference Board Schematic and Layout

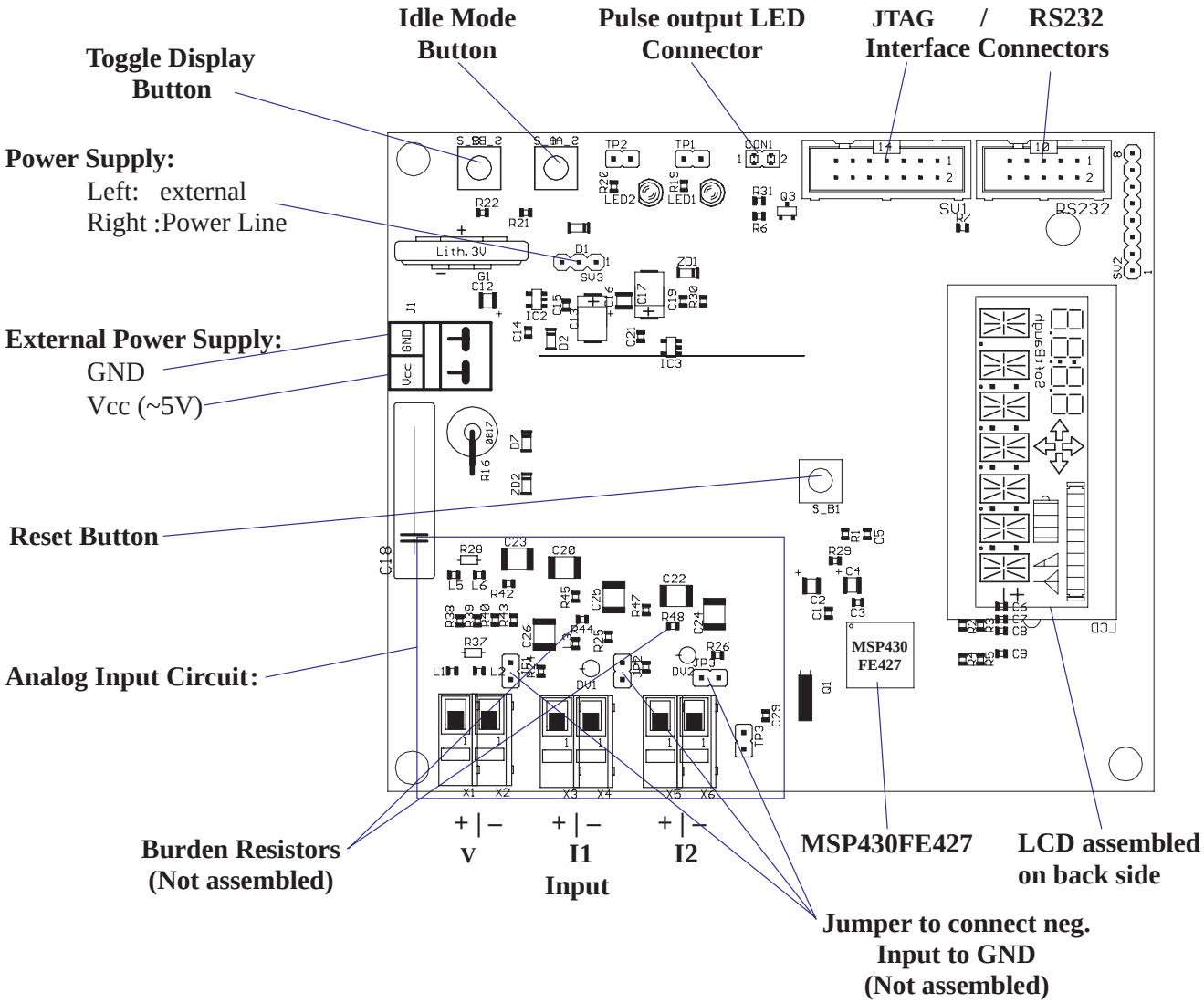


Figure A–1. Components on the Reference Board

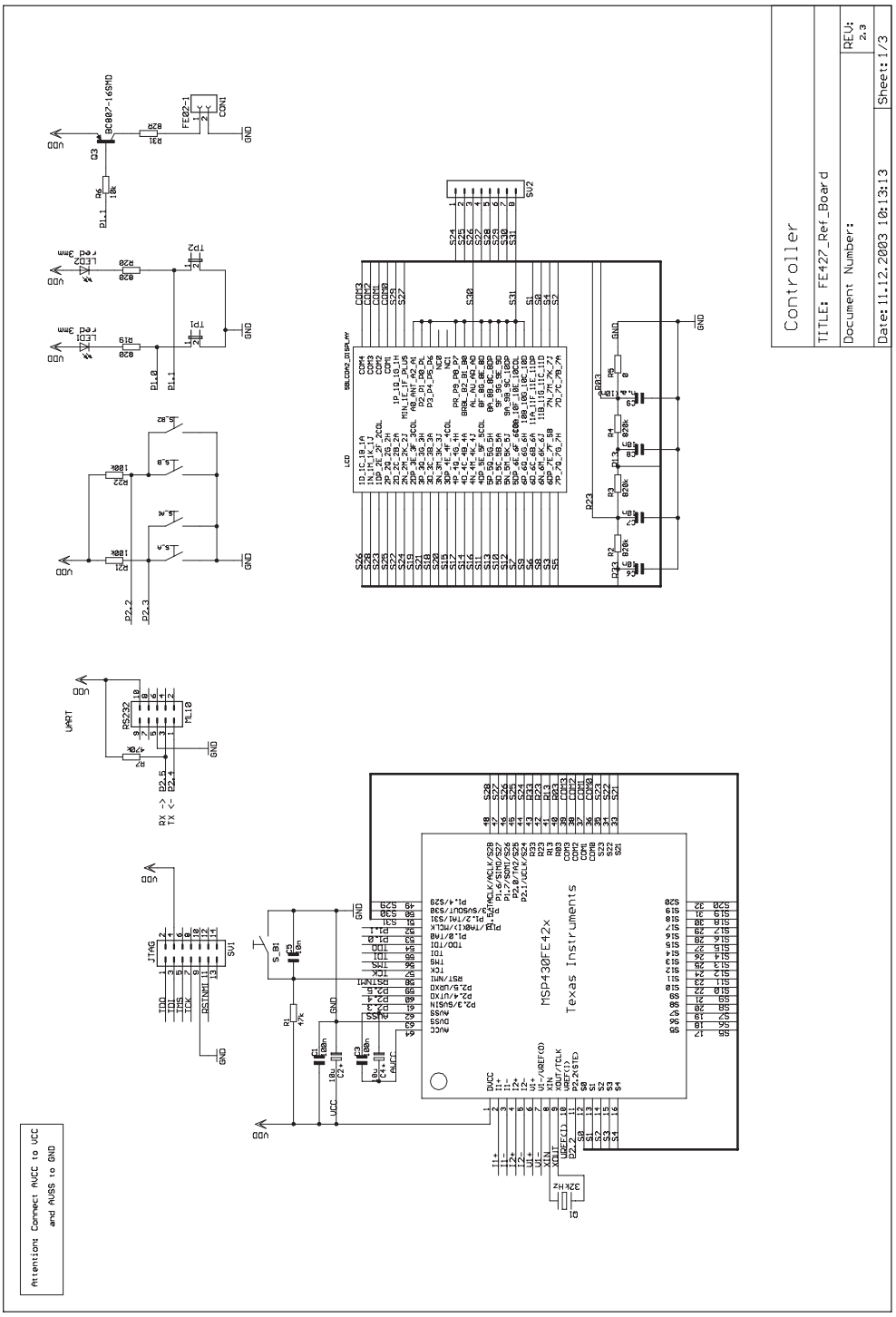
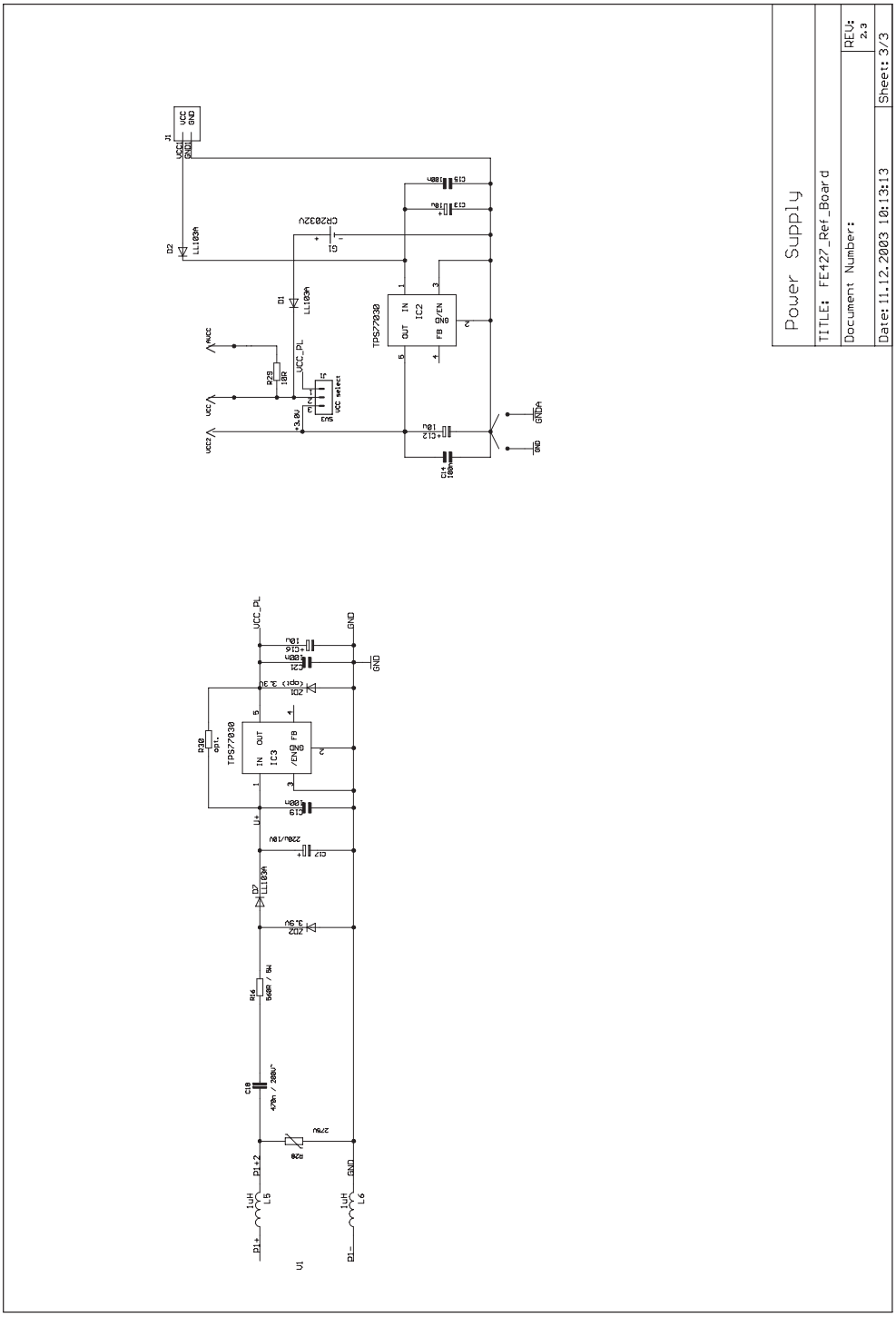
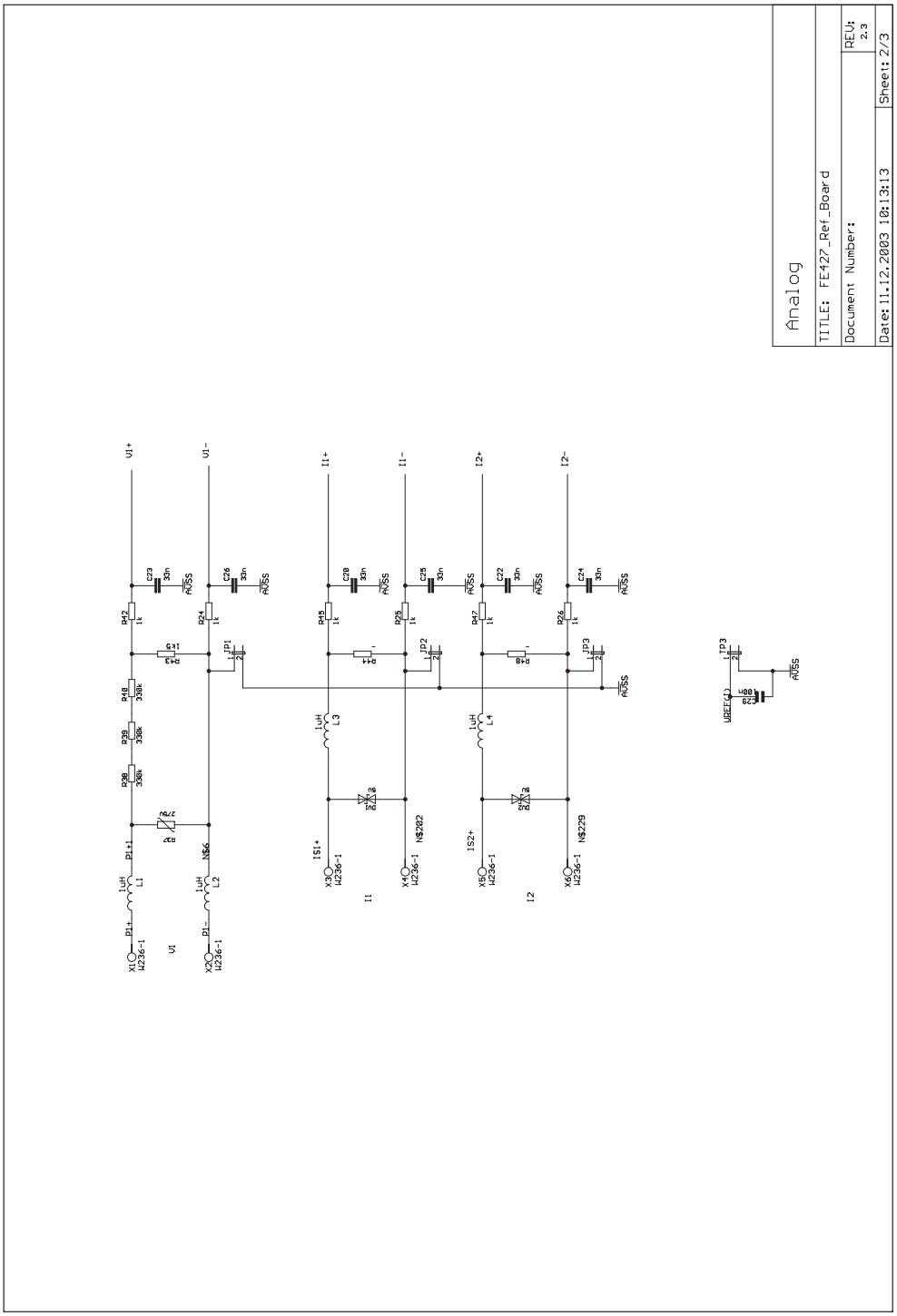


Figure A–2. Schematics


Power Supply
TITLE: FE422_Ref_Board
Document Number:
REV: 2.3
Date: 11.12.2003 10:13:13
Sheet: 3/3
J1
VCC
GND
D2
LL103A
CR2032V
G1
C15
100n
C13
10uF
D1
LL103A
TPS77030
IC2
IN
OUT
FB
/EN
VCC_PL
R28
18R
S2
VCC select
+3.6V
VCC
VCC2
C12
10u
C14
100n
GNDA
GND
VCC_PL
GND
C16
10u
C21
100n
ZD1
(opt) 3.9V
R30
0R1
TPS77030
IC3
IN
OUT
FB
/EN
C19
100n
U+
C12
220u/10V
D7
LL103A
ZD2
3.9V
R16
560R / 5W
C18
470n / 250V~
F28
27mU
D1+2
L5
1uH
L6
1uH
D1+
D1-
U1


Analog
TITLE: FE427_Ref_Board
Document Number:
REV: 2.3
Date: 11.12.2003 10:13:13
Sheet: 2/3

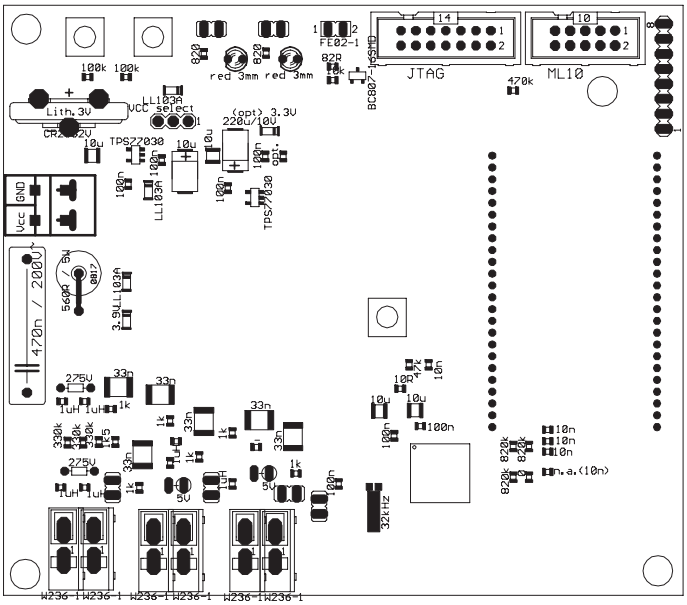


Figure A–3. Components on Top Side

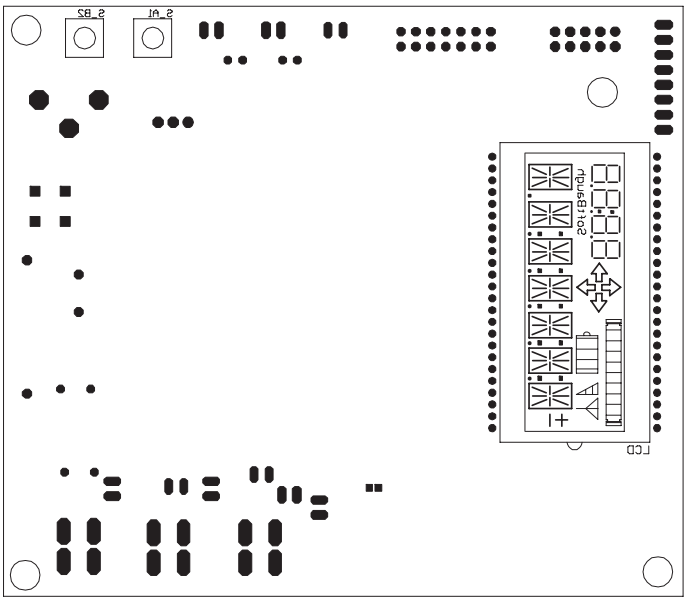


Figure A–4. Components on Bottom Side

Table A–1. Bill of Materials

QTY	PARTS	VALUE	DEVICE	MANUFACTURER
4	C1, C3, C19, C21	100 nF	CSMD0805	
1	C13	10 µF	ELKO	multicomp (10 µF/ 35 V)
3	C14, C15, C29	100 nF	C	
1	C17	220 µF/10 V	ELKO	
1	C18	470 nF / 200 V~	C–EU225–062X268	Phicomp
4	C2, C4, C12, C16	10 µF	ELKO_1210	10 µF/10 V
6	C20, C22, C23, C24, C25, C26	33 nF	C–EUC4532	AVX
1	C27	22 µF/25 V	CPOL–EUD/7343–31R	AVX Typ: TAJ
4	C5, C6, C7, C8	10 nF	CSMD0805	
1	C9	N.A. (10 nF)	CSMD0805	
2	D2, D3	LL103A	BAS32	
2	D7, D10	LL103A	D	
2	DV1, DV2	5 V	D_SUPPRESS	
1	G1	CR2032V	CR2032V	Varta
2	IC2, IC3	TPS77030	TPS77001	Texas Instruments
1	J1	SUP_CON	SUP_CON	
6	L1, L2, L3, L4, L5, L6	1 µH	L–US08050805	
1	LCD	SBLCDA2_DISPLAY	SBLCDA2_DISPLAY	Softbaugh
2	LED1, LED2	Red 3 mm	LED3MM	Multicomp (Low Power)
1	LED3	SFH486	LED5MM	Infinion
1	MSP1		FE427_CHIP	Texas Instruments
1	Q1	32 kHz	QUARZ32K	
1	Q3	BC807–16SMD	BC807–16SMD	Philips (5Ap)
1	R1	47 kΩ	R_0805	
1	R16	560R / 5 W	R–EU_0817/7V	
2	R19, R20	820	R_0805	
3	R2, R3, R4	820 kΩ	R_0805	
2	R21, R22	100 kΩ	R_0805	
1	R23	0R	R_0805	
6	R24, R25, R26, R42, R45, R47	1 kΩ	R_0805	
2	R28, R37	275 V	VARISTOR–2,5	EPCOS
1	R29	10R	R_0805	
1	R30	opt.	R_0805	
1	R31	82R	R_0805	
3	R38, R39, R40	330 kΩ	R_0805	
1	R43	1k5	R_0805	
2	R44, R48	–	R_0805	
1	R5	0	R_0805	
1	R6	10 kΩ	R_0805	
1	R7	470 kΩ	R_0805	
1	RS232	ML10	ML10	
5	S_A, S_A1, S_B, S_B1, S_B2	Switch	Switch	
1	SV1	JTAG	ML14	
1	SV2		MA08–1	

QTY	PARTS	VALUE	DEVICE	MANUFACTURER
1	SV3	VCC select	MA03–1@2	
3	TP1, TP2, TP3		JP1E	Connector row
6	X1, X2, X3, X4, X5, X6	W236–1	W236–1	Wago 256
1	ZD1	(opt) 3.3 V	D	
1	ZD2	3.9 V	D	

Appendix B Frequently Asked Questions

1) Is it possible to implement temperature compensation with the ESP.

Yes, the ESP430 can measure ambient temperature with the integrated temperature sensor. Using this result, the MSP430 CPU can modify parameters in the ESP430 accordingly. This can also be performed without stopping the ESP, on the fly. For example, to change the phase shift parameter for a current transformer based on ambient temperature, a parameter lookup table can be applied to the ESP setup by the MSP430 CPU to improve the temperature dependent error performance of the CT.

2) Reactive power appears to be inaccurate at low current loads.

The reactive power is calculated using the power triangle method using the active and apparent power measured and therefore may have some variances at very low currents.

3) How is the reactive power measured?

The reactive power is not measured by the ESP, it is calculated using the active and apparent power results. Because the discrete SD16 values are available to the MSP430 CPU, the reactive power can be calculated by doing a phase shift of the voltage or current by 90° with user software. In this case it is also possible to apply filtering for harmonics which are required in some meters.

4) Energy pulses which are directly generated by the ESP430 interrupt level function seems to exhibit jitter. Is the measurement unstable?

The ESP430 interrupt level flag is set as soon as the preset energy is accumulated in the buffer. With each SD16 conversion result a certain amount of energy is added to the internal buffer. Since this energy is based on a sin^2 function, sometimes a very small or very large amount can be accumulated. Each time the value in the buffer exceeds the limit set by the interrupt level the flag is set and the interrupt service is requested.

5) When negative energy is measured by the meter, ILREACHED is not launching an event message as it does when the meter measures positive energy. I believe ILREACHED should be causing an interrupt and a message each time the amount of negative energy is accumulated by the meter, but it is not doing this. I've tried NEx bit settings 00, 01, and 10 in the ESP_CTRL0 register without any success.

The ILREACHED function for energy mode does the processing in real-time. This means with each value which comes from the SD16 converter the actual measured energy is added to the internal buffer and compared with the preset level. This is the reason why it only works with positive active energy for the comparison.

However, the correct energies according to the NEx bits are always available in the energy return registers within the ESP430 module and can be accessed anytime by the CPU:

```
#define ACTENERGY1_LO  RET8   /* Active energy I1 Low Word */
#define ACTENERGY1_HI  RET9   /* Active energy I1 High Word */
#define ACTENERGY2_LO  RET10  /* Active energy I2 Low Word */
#define ACTENERGY2_HI  RET11  /* Active energy I2 High Word*/
#define ACTENSPER1_LO  RET16  /* Active energy I1 for last mains period LSW */
#define ACTENSPER1_HI  RET17  /* Active energy I1 for last mains period MSW */
#define ACTENSPER2_LO  RET18  /* Active energy I2 for last mains period LSW */
#define ACTENSPER2_HI  RET19  /* Active energy I2 for last mains period MSW */
```

6) I am intending to process the arrival of messages from the ESP in an interrupt service routine, using IN0IFG as the interrupt source. Based on the ESP430CE1 user's guide, the mEVENT message is sent when a flag in STAT0 is set if the corresponding flag in EVENT is also set. The message also includes the value of STAT0 in MBIN1. However, the description of many of the bits in STAT0 say that the bit is reset if an event message has been sent.

1. An event occurs in the ESP and the bit is set in STAT0.
2. The corresponding bit in EVENT is set, therefore:
   a: STAT0 is written to MBIN1
   b: mEVENT is written to MBIN0
   c: the bit is cleared in STAT0

In Addition:

a. If two events occur at the same time, does the ESP send just one message for all the events, or does it sent a separate message for each event?

The events are synchronized to the ADC samples and sent once new ADC samples have been processed.

b. When an event message is sent, how quickly must the CPU access the mailbox registers to prevent a following event message from being lost?

The event flags are set at the rate of the ADC samples. If the new event message could not be sent in time, the prior flag will not be cleared and will also be sent with the next event message.

7) When a temperature measurement command (mTEMP message) is sent to the ESP430, the temperature measurement is performed on the next zero-crossing of the voltage input, which could be up to 10ms after the command was sent. During the time between when the command was sent the temperature result is returned, is it possible to send other messages to the ESP, such as a request to read a parameter register? If this is possible, what happens if the replies to both messages are ready at the same time?

Yes, it is possible to send the additional message during this time. Messages will be read and processed in the order they are received by the ESP. User software should check that the mailbox is ready to receive a new message prior to accessing the out-going mailbox registers – this will reflect if the ESP has read the prior message.

8) In our design we only use one current channel but we would like use the 2nd current channel for a separate measurement (e.g. battery voltage). What settings should be made in order to do this?

It is possible to request a measurement on the ADC current input channel (I2) using the ESP but the limitation is that it cannot be sampled as often as the I/V meter input channels.

There is a message command available which is not currently included in the user's guide and in the header file which requests an single measurement on the I2 channel and returns the value back via the mailbox to the CPU.

```
/* Message to ESP */
#define mREAD_I2      (0x000F)  /* Sample request I2 Channel result */
/* Message from ESP */
#define mI2RDY        (0x000B) /* I2 value ready */
```

After sending the sample request message to the ESP430 module the I2 channel will be enabled and after 4 measurement cycles the result will be sent via the mailbox to the MSP430 CPU.

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
Interface	interface.ti.com	Digital Control	www.ti.com/digitalcontrol
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
		Telephony	www.ti.com/telephony
		Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments
Post Office Box 655303 Dallas, Texas 75265