

Implementing An Electronic Watt-Hour Meter With MSP430FE42x(A)/FE42x2

Stefan Schauer, Kripasagar Venkat

MSP430 Applications

ABSTRACT

This report shows how to implement an electronic watt-hour meter with the MSP430FE42x(A)/FE42x2 devices. It contains guidelines and recommendations for use of the MSP430FE42x(A) and MSP430FE42x2 devices. In addition, a reference board with hardware details and software examples are included.

Contents

| | | |
|---|--|----|
| 1 | Introduction | 3 |
| 2 | Hardware | 3 |
| | 2.1 Getting Started with the Reference Board | 3 |
| | 2.2 Shunt as Current Sensor | 4 |
| | 2.3 CT as Current Sensor | 5 |
| | 2.4 CT and Shunt as Current Sensor for Tamper Detection (Not Applicable for FE42x2 Devices) | 6 |
| | 2.5 CT for the US 1-Phase 3-Wire E-Meter Solution | 7 |
| | 2.6 Voltage Input Connections | 7 |
| | 2.7 Current Input Connections | 7 |
| | 2.8 Anti-Aliasing Filter | 8 |
| | 2.9 Unused ADC Inputs | 8 |
| 3 | Calculation of the MSP430CE Meter Constants | 8 |
| | 3.1 Voltage Ratio | 8 |
| | 3.2 Current Ratio for Shunt | 8 |
| | 3.3 Current Ratio for Current Transformer | 8 |
| | 3.4 Interrupt Level for Energy | 8 |
| 4 | Meter Calibration | 9 |
| | 4.1 Calibration With Continuous Measurement | 10 |
| | 4.2 Calibration With a Host Computer | 12 |
| | 4.3 Self Calibration | 13 |
| 5 | Capacitor Power Supply | 14 |
| | 5.1 Power Line Voltage On/Off Detection | 14 |
| 6 | Layout Recommendations | 15 |
| | 6.1 Grounding | 15 |
| | 6.2 EMI Sensitivity | 16 |
| 7 | Demo Software | 17 |
| | 7.1 Analog Front-End Initialization | 17 |
| | 7.2 E-Meter Initialization | 17 |
| | 7.3 Demo 1 Software | 17 |
| | 7.4 Energy Pulse Generation | 18 |
| | 7.5 Temperature Compensation for CT Phase Shift | 19 |
| | 7.6 Controls | 19 |
| | 7.7 Demo 2 | 20 |
| 8 | References | 23 |
| | Appendix A Reference Board Schematic and Layout | 24 |

| | | |
|------------|----------------------------------|----|
| Appendix B | Frequently Asked Questions | 30 |
|------------|----------------------------------|----|

List of Figures

| | | |
|-----|---|----|
| 1 | Block Diagram for the Connection of a Shunt for Single Phase, 2-Wire | 4 |
| 2 | Block Diagram for the Connection of a CT for Single Phase, 2-Wire..... | 5 |
| 3 | Block Diagram for Connection of a Shunt and CT With Tamper Detection for Single Phase, 2-Wire | 6 |
| 4 | Block Diagram for the ANSI Single-Phase 3-Wire E-Meter Solution | 7 |
| 5 | MSP430 Electronic Electricity Meter With External Terminals | 9 |
| 6 | Electricity Meter Calibration With a Host Computer | 12 |
| 7 | Self-Calibration for Electricity Meters..... | 13 |
| 8 | Capacitor Supply | 14 |
| 9 | Power Detection..... | 14 |
| 10 | Analog-to-Digital Converter Grounding..... | 15 |
| 11 | Routing Sensitive to External EMI | 16 |
| 12 | Routing for Minimum EMI Sensitivity | 16 |
| 13 | Software Flow | 18 |
| A-1 | Components on the Reference Board..... | 24 |
| A-2 | Schematics (1 of 3) | 25 |
| A-3 | Schematics (2 of 3) | 26 |
| A-4 | Schematics (3 of 3) | 27 |
| A-5 | Components on Top Side | 28 |
| A-6 | Components on Bottom Side | 28 |

List of Tables

| | | |
|-----|-------------------------------|----|
| 1 | Demo 1 Files..... | 17 |
| 2 | Demo 2 Additional Files | 20 |
| A-1 | Bill of Materials | 28 |

1 Introduction

This application report shows a hardware reference design and software routines for implementing an electronic electricity meter with the MSP430FE42x(A)/FE42x2 devices. It is intended to be used as a supplement to the *ESP430CE1, ESP430CE1A, ESP430CE1B Peripheral Modules User's Guide (SLAU134)*, which describes the ESP430CE1/1A/1B modules.

The MSP430FE42x(A) and MSP430FE42x2 with the ESP430CE1(A) and ESP430CE1B respectively have been specifically developed for energy metering applications. The Embedded Signal Processor (ESP) with an integrated analog front end and temperature sensor is designed for single phase energy metering. The ESP430CE 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 such as communication. The ESP430CE offers wide flexibility for current sensors, so that it is possible to use shunts or current transformers (including dc-tolerant CTs with high phase shift) 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 ESP430CE during the system initialization.

2 Hardware

The reference board schematic and system block diagrams are shown 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.

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 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 MSP430FE42x(A)/FE42x2 and an initial calibration follows the setup.

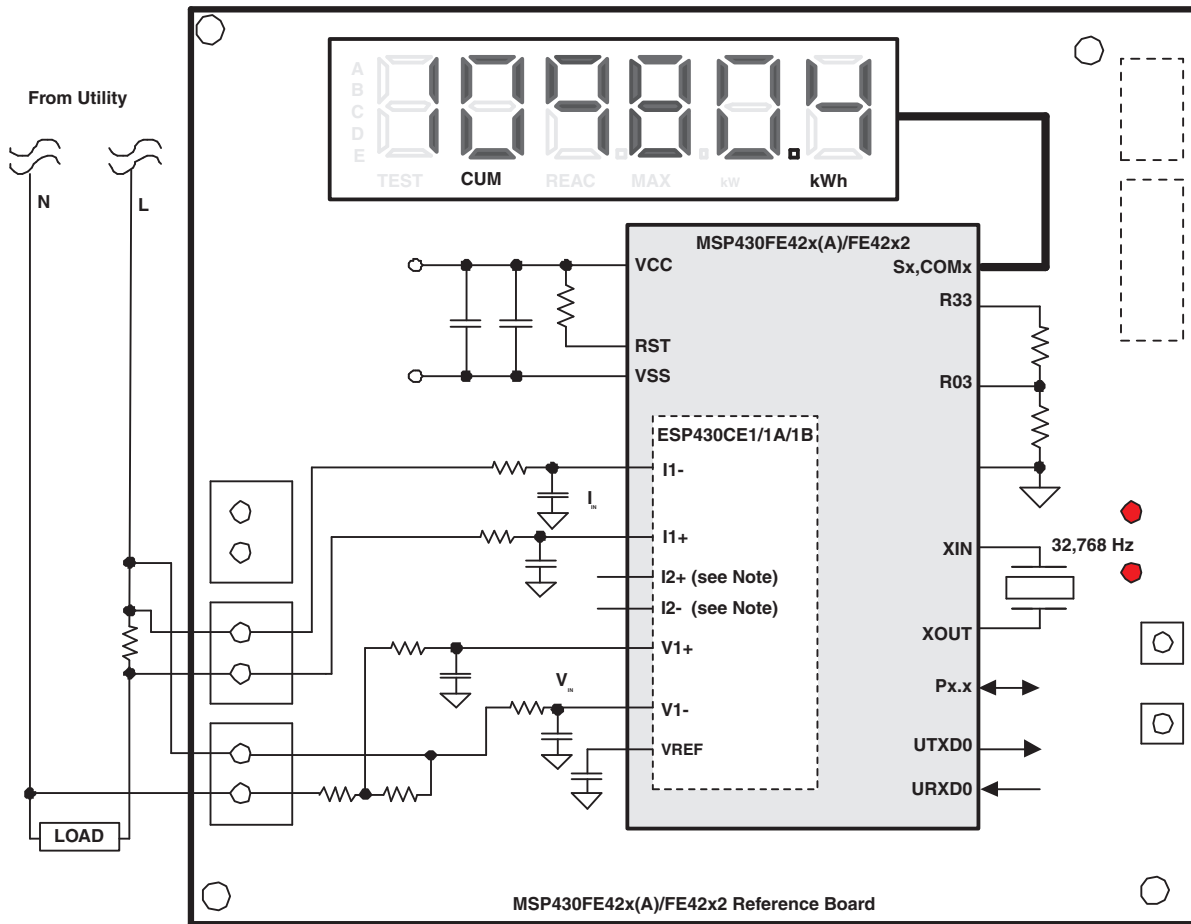
1. Connect the board

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 if 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 5 V/10 mA (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 (Settings.xls)

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

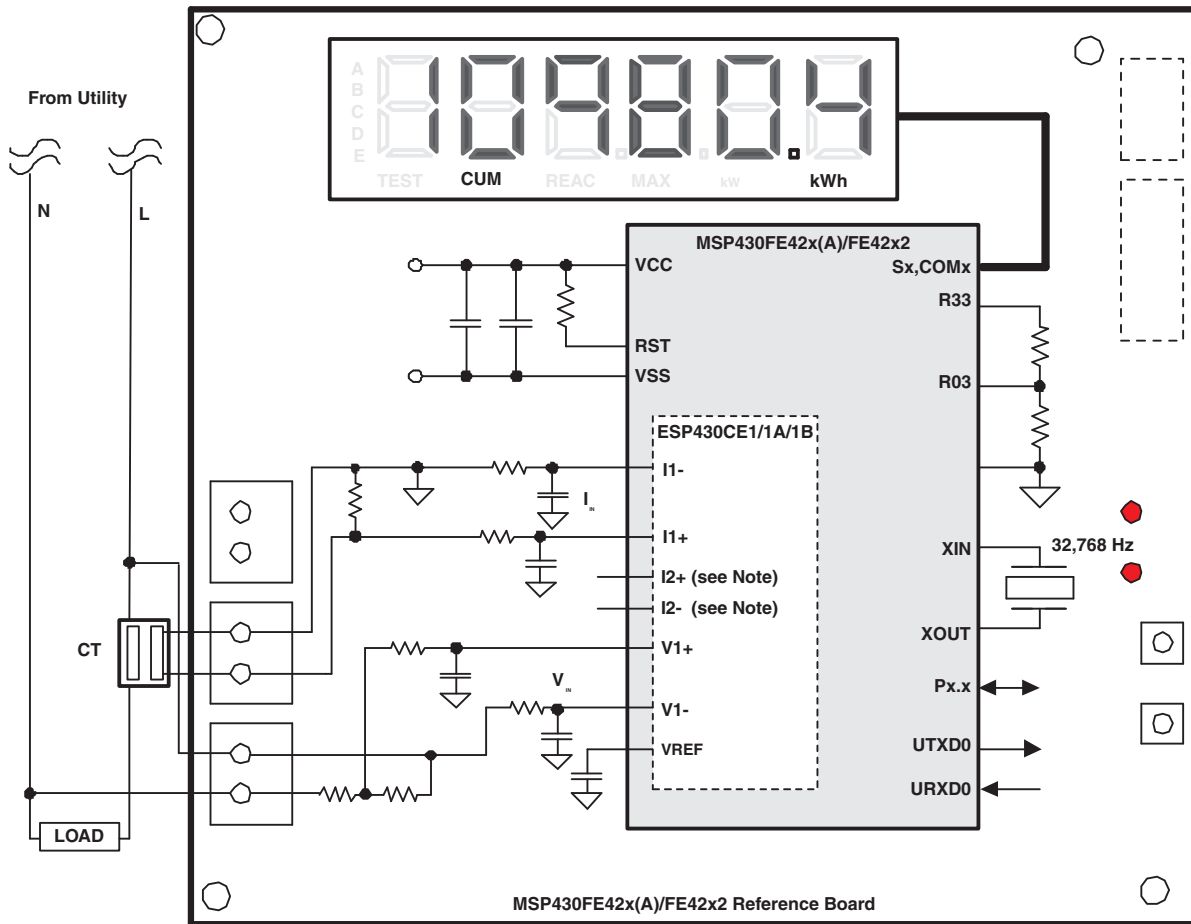
2.2 Shunt as Current Sensor



Note: I2+ and I2- are not present on the FE42x2 devices.

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

2.3 CT as Current Sensor



Note: I2+ and I2- are not present on the FE42x2 devices.

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 (Not Applicable for FE42x2 Devices)

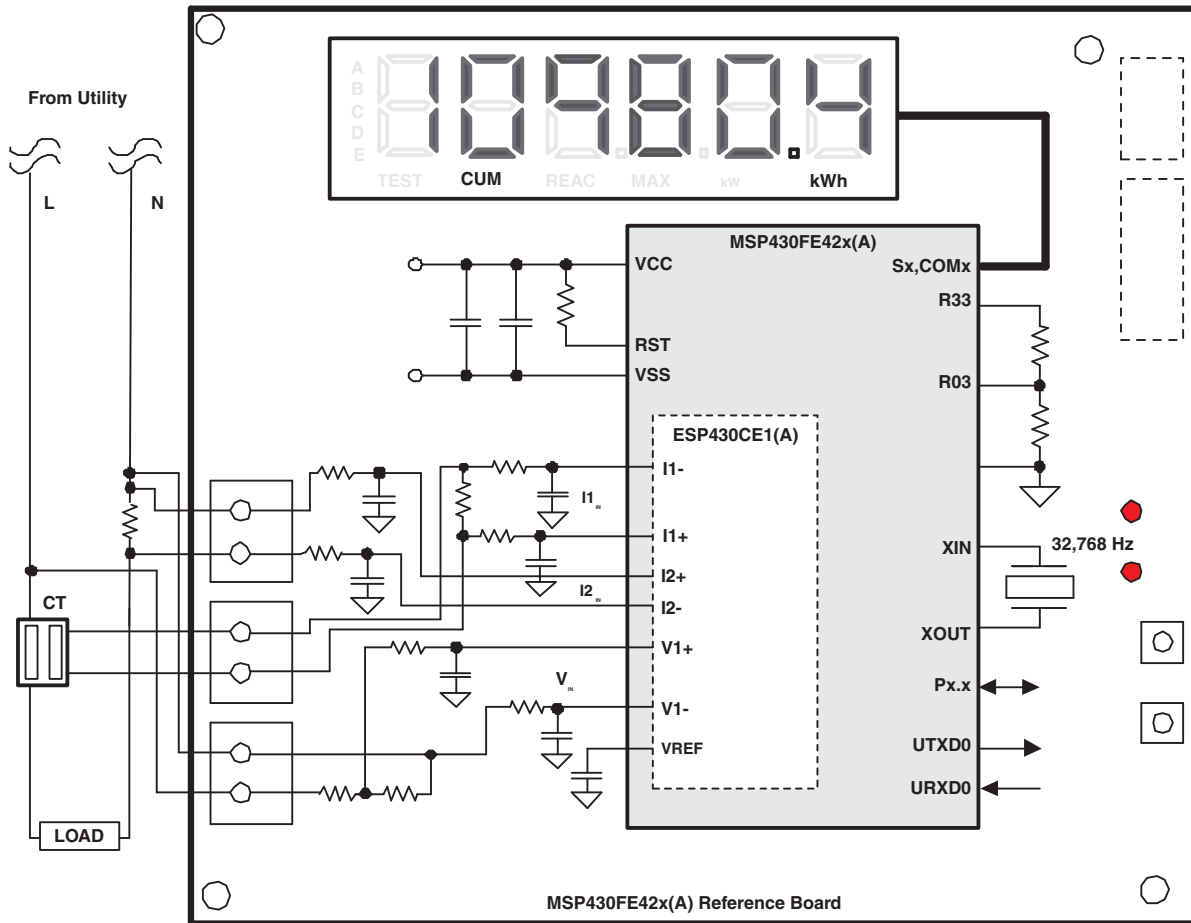
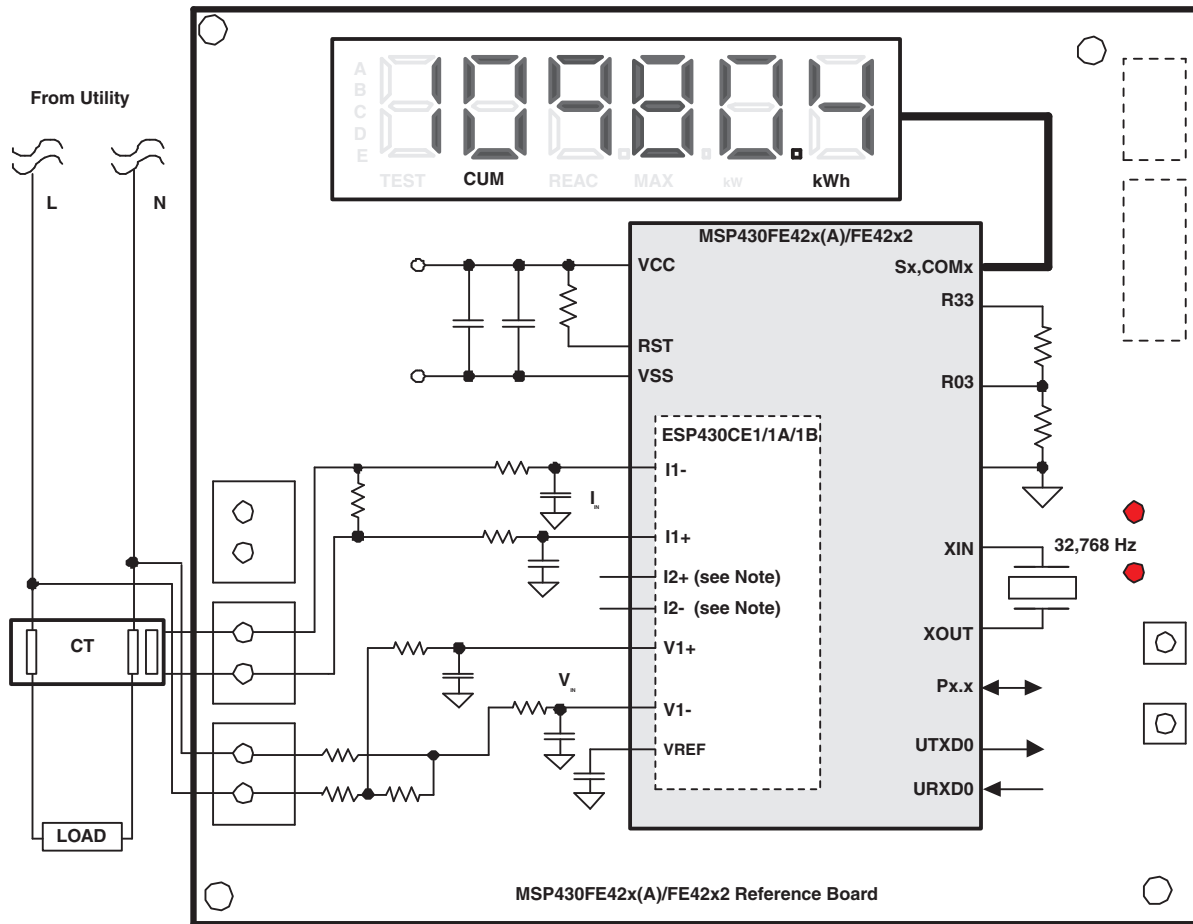


Figure 3. Block Diagram for Connection of a Shunt and CT With Tamper Detection for Single Phase, 2-Wire

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



Note: I2+ and I2- are not present on the FE42x2 devices.

Figure 4. Block Diagram for the ANSI Single-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 is 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.

CAUTION

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-k Ω resistor in series to the ADC input and a 33-nF capacitor to the analog ground. To avoid external influences, having filters in the positive and negative input channels of each ADC input is recommended.

2.9 Unused ADC Inputs

Unused ADC inputs should be unconnected.

3 Calculation of the ESP430CE Meter Constants

The meter requires constants based on the voltage and current transformers/shunts. The following sections describe how to calculate the ESP430CE meter constants.

3.1 Voltage Ratio

The voltage ratio, which calculates the real voltage from the ESP430CE voltage value, is calculated as follows:

$$V(\text{inp.max}) = \frac{\text{VoltageGain} \times V(\text{Line, Nom}) \times \sqrt{2} \times R2}{R1 + R2}$$

$$kV1 = \frac{\text{Voltage}(\text{Line, Nom}) \times 2 \times \sqrt{2}}{2^{15} \times \left(1 - \frac{V_{\text{ref}} - V(\text{inp.max}) \times 2}{V_{\text{ref}}}\right)}$$

3.2 Current Ratio for Shunt

The current ratio for a shunt, which calculates the real current from the ESP430CE current value, is calculated as follows:

$$V(I, \text{inp.max}) = \text{CurrentGain} \times I_{\text{max}} \times R(\text{Shunt}) \times \sqrt{2}$$

$$kI1 = \frac{\text{Current}(\text{Line, Nom}) \times 2 \times \sqrt{2}}{2^{15} \times \left(1 - \frac{V_{\text{ref}} - V(I, \text{inp.max}) \times 2}{V_{\text{ref}}}\right)}$$

3.3 Current Ratio for Current Transformer

The current ratio for a CT which calculates the real current from the ESP430CE current value, is calculated as follows:

$$V(I, \text{inp.max}) = \frac{\text{CurrentGain} \times I_{\text{max}}}{\text{CTRatio} \times R(\text{Burden}) \times \sqrt{2}}$$

$$kI1 = \frac{\text{Current}(\text{Line, Nom}) \times 2 \times \sqrt{2}}{2^{15} \times \left(1 - \frac{V_{\text{ref}} - V(I, \text{inp.max}) \times 2}{V_{\text{ref}}}\right)}$$

3.4 Interrupt Level for Energy

The energy consumption interrupt level of the ESP430CE is calculated as follows:

$$\text{InterruptLevel} = \frac{\text{Pulses}}{\text{kWh}} \times \frac{1000}{3600} \times \frac{f_{\text{ADC}}}{kV1 \times kI1 \times 4096}$$

Pulses/kWh defines how many interrupts or pulses 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.

A basic calibration can be initiated with the UART c0 command. The execution of this command requires these input values defined in the parameter.h file:

- calVoltage
- calCurrent
- calPhi
- calCosPhi
- calFreq

The phase shift calibration between voltage and current should be done with a $\cos\phi$ of 0.5 as at this point the error of the phase shift resulting from the sensors generates much higher errors and, therefore, a higher accuracy can be reached with less effort.

For the calibration of electricity meters, it is necessary to separate the voltage and current paths 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.

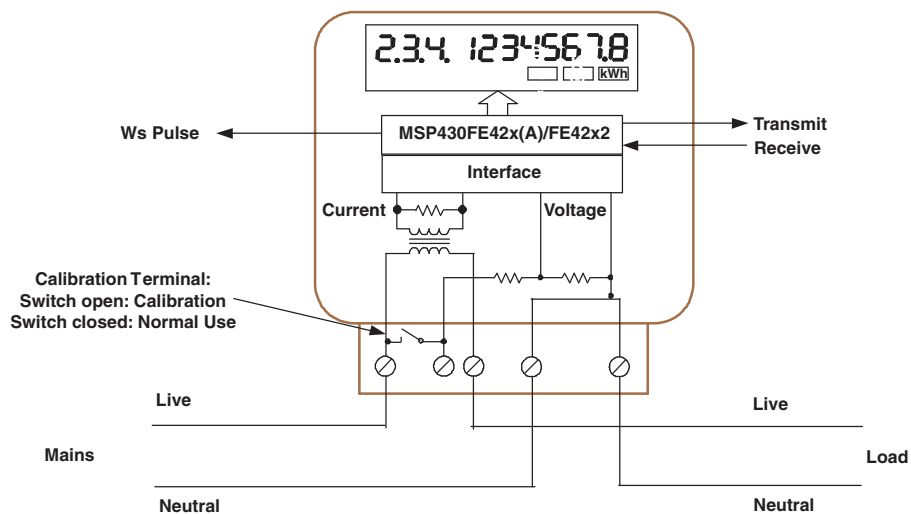


Figure 5. MSP430 Electronic Electricity Meter With External Terminals

4.1 Calibration With Continuous Measurement

The ESP430CE operation mode is set to electricity meter mode sending commands via the mailbox. The Set Mode command to the CPU initializes the ESP430CE for normal measurement. The energy values written after each mains period to the value registers ACTENPER1_LO, ACTENPER1_HI (and ACTENPER2_LO, ACTENPER2_HI 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 module 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 (not valid for FE42x2) in control register 0 according to the measurement mode of the ESP430CE.
- The parameter registers are initialized for the load point to be measured.
- The ESP430CE mode is set to electricity meter mode via the mailbox with the mSET_MODE command
- The first result in addresses ACTENPER1_LO, ACTENPER1_HI (and ACTENPER2_LO, ACTENPER2_HI, if current path 2 is enabled) is not used, due to the unknown start point inside the mains period.
- The next results in addresses ACTENPER1_LO, ACTENPER1_HI (and ACTENPER2_LO, ACTENPER2_HI, if current path 2 is enabled) are valid and are used for calculations.
- The flag ZXLDIFG in status register 0 indicates that with the next available samples (flag WFSRDYFG is set) new results for the last mains period are available in addresses ACTENPER1_LO, ACTENPER1_HI, ACTENPER2_LO, and ACTENPER2_HI.
- The CPU resets the flag WFSRDYFG with the Mailbox command mCLR_EVENT and processes the read results with the equations shown in [Section 4.1.1](#).
- 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 $I_{1\text{HI}}$ and $I_{1\text{LO}}$. The nominal energy results for the two calibration points are:

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

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

The resulting values for the slope and offset are:

$$\text{Slope : GAINCORR1} = \frac{n_{\text{HIcalc}} - n_{\text{LOcalc}}}{n_{\text{HI meas}} - n_{\text{LO meas}}} \times 2^{14}$$

$$\text{Offset : POFFSET1} = \frac{n_{\text{HI meas}} \times n_{\text{LOcalc}} - n_{\text{LO meas}} \times n_{\text{HIcalc}}}{n_{\text{HI meas}} - n_{\text{LO meas}}} \times \frac{f_{\text{mains}}}{n_{\text{per}}} \times \frac{4096}{f_{\text{ADC}}}$$

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 [steps²]

$n_{\text{HI meas}}$ = Measured energy at the high current calibration point [steps²]

n_{LOcalc} = Calculated energy at the low current calibration point [steps²]

$n_{\text{LO meas}}$ = Measured energy at the low current calibration point [steps²]

4.1.2 Calibration Example

The I_1 path of the electricity meter shown in [Figure 1](#) is calibrated with the following values:

$$V_1 = 230 \text{ V}, I_{1\text{HI}} = 20 \text{ A}, I_{1\text{LO}} = 1 \text{ A}, \cos\phi_1 = 1, n_{\text{per}} = 1, f_{\text{ADC}} = 2048 \text{ Hz}, f_{\text{mains}} = 50 \text{ Hz}$$

The nominal measurement results n_{Hlcalc} and n_{LOcalc} are shown below.

The meter constants are calculated using the formulas in [Section 4.1.1](#).

$$C_{Z1} = f_{\text{ADC}} / (\text{kV1} \times \text{kI1} \times 4096)$$

This example uses a previously known meter constant.

$$n_{\text{Hlcalc}} = C_{Z1} \times I_{1\text{HI}} \times V_1 \times \cos\phi_1 \times \frac{n_{\text{per}}}{f_{\text{mains}}} \times \frac{f_{\text{ADC}}}{4096} = 29,322.80806 \times 20 \times 230 \times 1 \times \frac{1}{50} \times \frac{2048}{4096}$$

$$n_{\text{Hlcalc}} = 1,348,849.171 = 14,94\text{F1h} \quad [\text{steps}^2]$$

$$n_{\text{LOcalc}} = C_{Z1} \times I_{1\text{LO}} \times V_1 \times \cos\phi_1 \times \frac{n_{\text{per}}}{f_{\text{mains}}} \times \frac{f_{\text{ADC}}}{4096} = 29,322.80806 \times 1 \times 230 \times 1 \times \frac{1}{50} \times \frac{2048}{4096}$$

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

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

$$\text{GAINCORR1} = \frac{n_{\text{Hlcalc}} - n_{\text{LOcalc}}}{n_{\text{Hlmeas}} - n_{\text{LOmeas}}} \times 2^{14} = \frac{14,94\text{F1h} - 1,0772\text{h}}{14,6040\text{h} - 1,0\text{CB7h}} \times 2^{14} = 1.01171 \times 2^{14} = 40\text{C0h}$$

The offset in addresses POFFSET1 and POFFSET1+2 becomes:

$$\text{POFFSET1} = \frac{n_{\text{Hlmeas}} \times n_{\text{LOcalc}} - n_{\text{LOmeas}} \times n_{\text{Hlcalc}}}{n_{\text{Hlmeas}} - n_{\text{LOmeas}}} \times \frac{f_{\text{mains}}}{n_{\text{per}}} \times \frac{4096}{f_{\text{ADC}}}$$

$$\text{POFFSET1} = \frac{14,6040\text{h} \times 1,0770\text{h} - 1,0\text{CB7h} \times 14,94\text{F1h}}{14,6040\text{h} - 1,0\text{CB7h}} \times \frac{50}{1} \times \frac{4096}{2048} = -215,489 = \text{FFFC, B63Fh}$$

Note: The calculated value for (POFFSET1) is the offset for each product $\text{NV1} \times \text{NI1}$, thus the relatively high value.

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

$$n_{\text{corr}} = (n_{\text{meas}} \times \text{GAINCORR1}) \times 2^{-14} + \text{POFFSET1} \times \frac{n_{\text{per}}}{f_{\text{mains}}} \times \frac{f_{\text{ADC}}}{4096}$$

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

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

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% I_{nom}) or two load points (e.g., 100% I_{nom} and I_{max}). With these errors, the slope and offset of the load characteristic can be calculated individually and sent to the MSP430s in each meter.

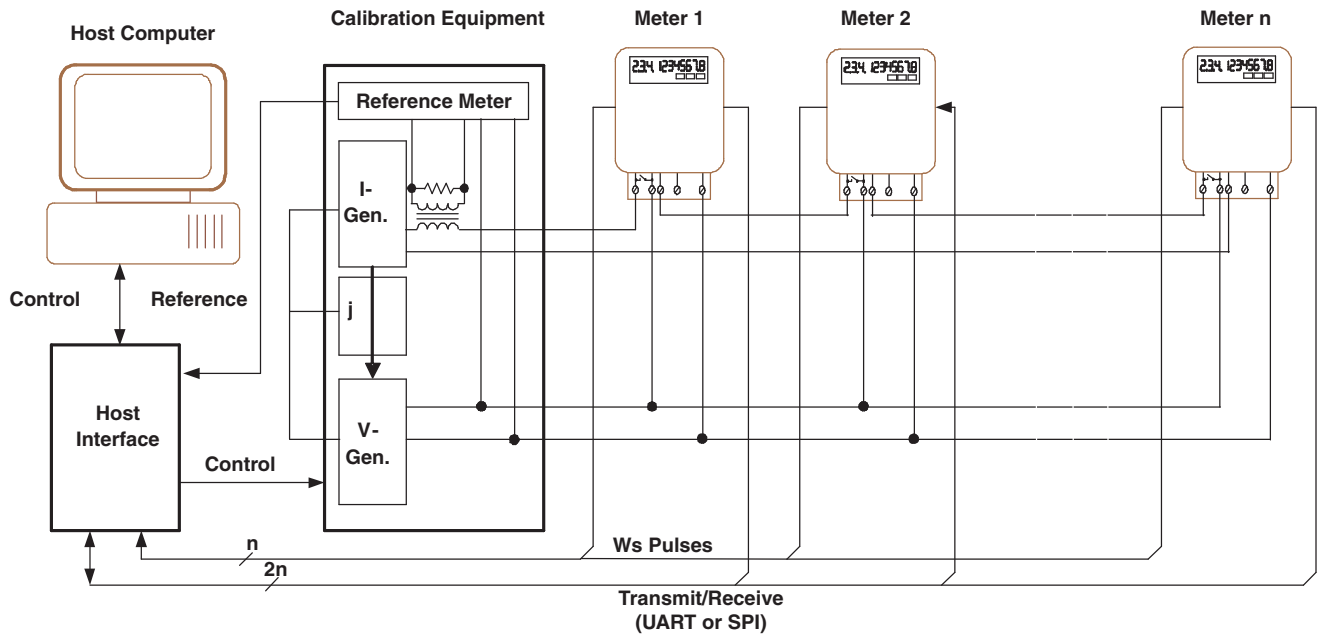


Figure 6. Electricity Meter Calibration With a Host Computer

The formulas for the calculation of the calibration values are shown in the *ESP430CE1, ESP430CE1A, ESP430CE1B Peripheral Modules User's Guide* ([SLAU134](#)).

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 Section 4.1.

- 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% I_{nom} 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% I_{nom} , 100% V_{nom} , $\cos\phi = 1$) 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% I_{nom} current.
- With the two faulty energy values WEM1 and WEM0 found for the 100% and 5% I_{nom} 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 I_{nom} , V_{nom} , and two values of $\cos\phi$) to the electricity meters.
 - A visual check is made to determine 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% I_{nom} , 100% V_{nom} , $\cos\phi = 1$)
- 5,000 Ws (100% I_{nom} , 100% V_{nom} , $\cos\phi = 0.5$)

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

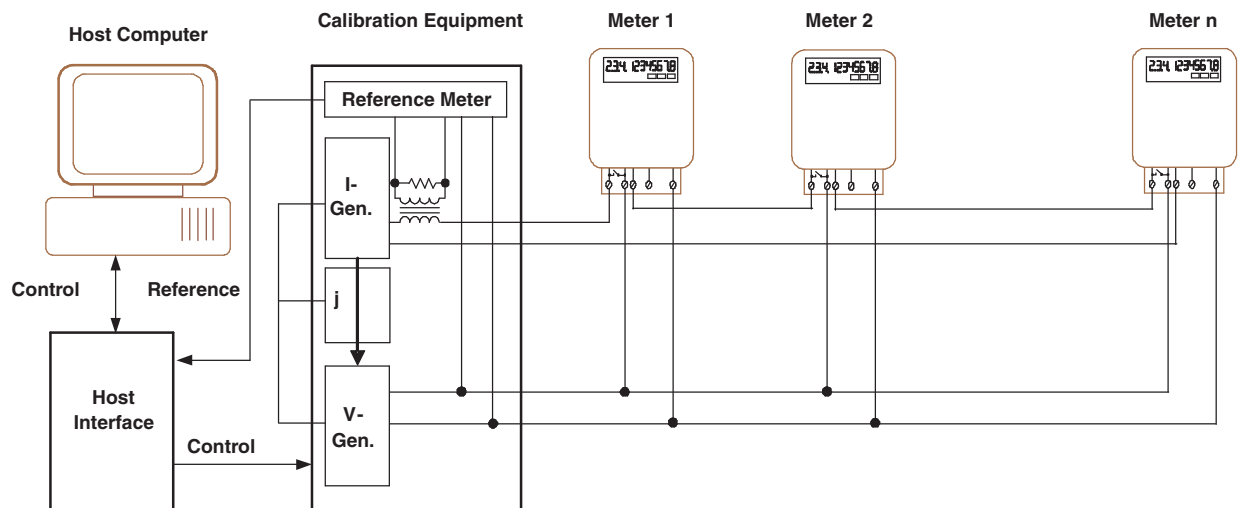


Figure 7. Self-Calibration for Electricity Meters

5 Capacitor Power Supply

Figure 8 shows a capacitor power supply with example values for a single output voltage up to $V_{CC} = 3.6$ 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 *MSP430 Family Mixed-Signal Microcontroller Application Reports (SLAA024)* in the *Capacitor Power Supplies* section. This section also describes other kinds of power supplies with their design equations.

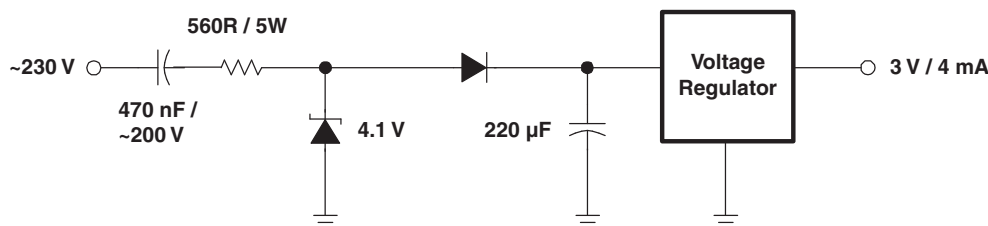


Figure 8. Capacitor Supply

5.1 Power Line Voltage On/Off Detection

Because the ESP430CE voltage drop detection is combined with the line cycle counter, the voltage drop detection of the ESP430CE 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 ESP430CE module can be switched off to conserve power.

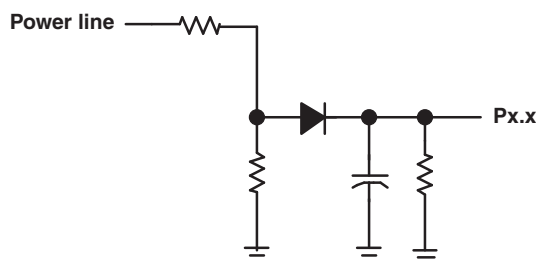


Figure 9. Power Detection

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 C_b 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 C_b should be close together. Two capacitors are connected across the digital (C_d) and the analog (C_a) supply terminals.
8. The coil L can 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, it should be connected to the ground potential (0 V) of the board.

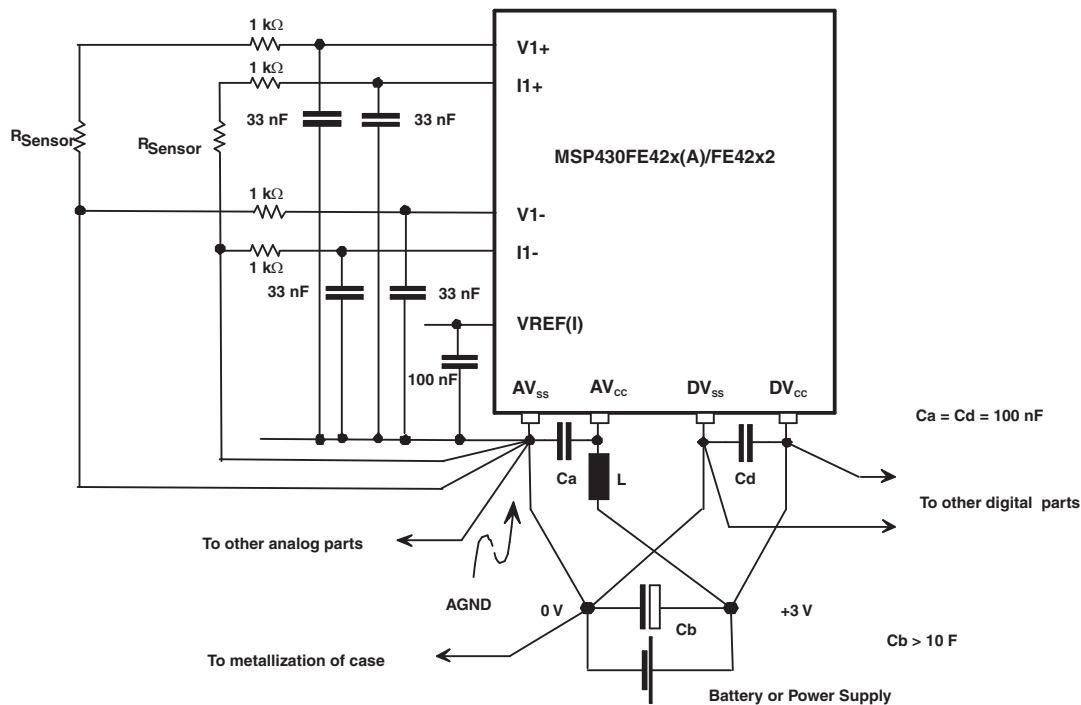


Figure 10. Analog-to-Digital Converter Grounding

6.2 EMI Sensitivity

Figure 11 shows a 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.

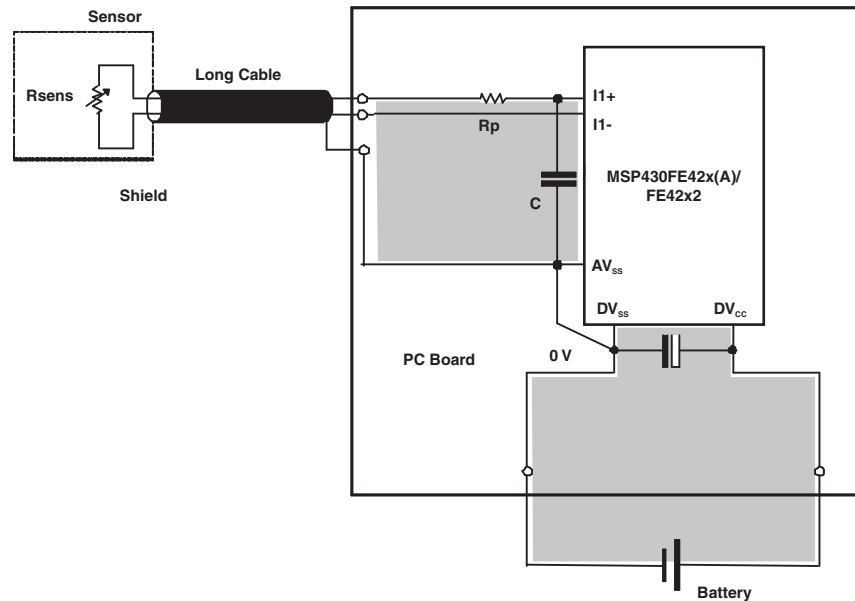


Figure 11. Routing Sensitive to External EMI

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

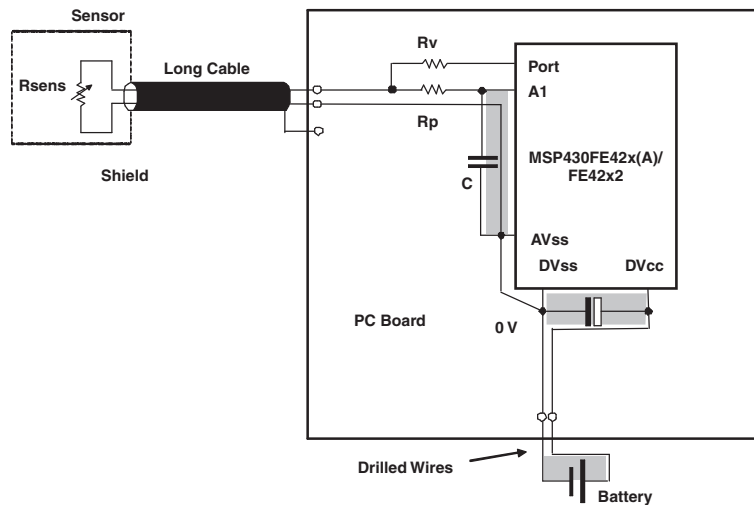


Figure 12. Routing for Minimum EMI Sensitivity

7 Demo Software

7.1 Analog Front-End Initialization

While the ESP430CE 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 ratio, 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 ESP430CE 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 ESP430CE 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:

Table 1. Demo 1 Files

| File ⁽¹⁾ | 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 ESP430CE parameters, start measurement |
| FET4xx_rtcwlcd.s43/.c | Basic routines for LCD and real time clock with basic timer ISR. |
| 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 ESP430CE, and the interrupt service routine for the ESP430CE. |
| FE427_Demo1_V3.ewp/.eww | Project files for IAR Workbench Version 3 |
| FE427_Demo1_V4.ewp/.eww | Project files for IAR Workbench Version 4 |
| 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 Settings Excel file) |
| parameter.h | Global parameter definitions |
| Subroutines.h | Subroutine definitions |
| Portfunc.c | Port handling source file |
| Unused_int.s43 | Unused ISR support source file |

⁽¹⁾ The file names change in accordance with the device chosen. For example, FE427 becomes FE427A or FE4272.

The demo software flow is shown in [Figure 13](#).

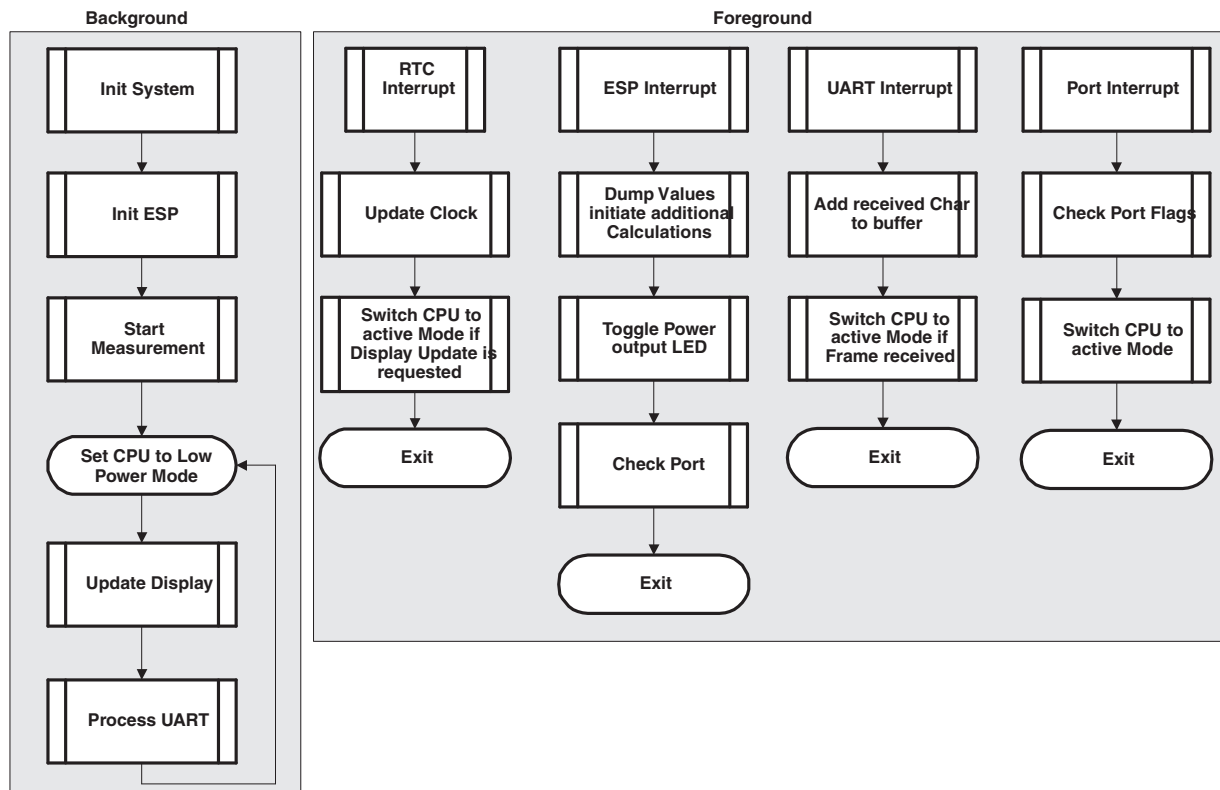


Figure 13. Software Flow

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 and does not need any additional hardware or software resources but, because the energy is the accumulation of a sine wave, this signal could have some jitter.

This method is active if the following is 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 CLOCKSPEEDPERIOD (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 withTempCorrection
```

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 ESP430CE off and set the MSP430 into low power mode. The real-time clock continues to run.
- S_B: Selects one of various evaluated parameters by the ESP430 for display.

7.6.1 Parameter.h file

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

- Pulse output level
- Voltage and current ratio
- Configuration settings for the ESP430CE

The #defines for withUARTComm, withCalibration, and withDisplay allow scaling the code for different function and sizes. The code uses floating point functions for these functions and including any one of them increases 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 [Section 3.1](#))
- defIRatio = $kI1 / 1000$ (for kI1 see [Section 3.2](#))
- defEnergyRatio = (defVRatio \times defIRatio)

For an easier calculation of the main parameters defined in the parameter.h file, the Excel files FE427_Settings.xls, FE427A_Settings.xls, or FE4272_Settings.xls could be used. After entering the required information into the white colored cells, the parameters are calculated and displayed. By clicking the Save Parameter to File button, the parameters are saved into the file 'ESP_Parameter.h'.

This file with the calculated parameters should be included in the source directory. [Section 7.6.2](#) describes the usage of the Settings Excel file (see the Excel sheet that is included with the source code).

7.6.2 Using Settings Excel file

Enter values into the white and blue colored cells as needed to meet application-specific requirements for the meter. Blue cells are dropdown boxes that contain fixed selections. 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 energy 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 MSP430FE427(A)/FE4272 demo project.

Note: If the Calculation values in the Hex columns are not updated in the spreadsheet, 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 ESP430CE are used instead of the interrupt level function. The initialization of the ESP430CE, the data output on the display, and the LED usage is included as in the Demo 1. Demo 2 is too large to be used with IAR KickStart version, and the full version must be used.

Demo 2 contains all the files in Demo 1 plus the files shown in [Table 2](#):

Table 2. Demo 2 Additional Files

| File | Contents and Function |
|---------------|---|
| UART0.c | Interrupt handler for UART receive |
| comms_uart.c | UART communication routines <ul style="list-style-type: none"> • 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/FE427A/FE4272 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
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 Demo 2 Software

From the zip file, extract and open the directory for the chosen device. The following steps detail a method for calibrating the meter implementation for the Demo 2 source code. This method makes a three-point calibration.

Step 1

- Adjust the values in FE427/FE427A/FE4272_Settings.xls for the sensor used.
- Generate the ESP_Parameter.h source file and add to the Demo 2 project folder.
- Save, compile and download the software to the appropriate target device.

Step 2

- Attach the meter to an I/V generator adjusted to the calibration values defined in the Settings Excel file (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., 240 V / 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., 240 V / 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(A)/FE42x2 PHASECORRx register; e.g., +5[ENTER]).

Step 6

- 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(A)/FE42x2 POFFSETx register; e.g., +5[ENTER]) use energy value of the FE42xA/FE42x2 displayed in the terminal with t5 mode:

$$\text{correctionvalue} = - \frac{\text{CurrentEnergyValue} \times \text{Error}\%}{4096}$$

Step 7

- Enter c9[ENTER] to save the values to the info memory of the FE42xA/FE42x2.

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/FE427A/FE4272_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 appropriate target device.

Step 2

- Attach the meter to an I/V generator adjusted to the calibration values defined in the Settings Excel file (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(A)/FE42x2 PHASECORRx 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

- 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(A)/FE42x2 POFFSETx register; e.g., +5[ENTER]) use energy value of the FE42xA/FE42x2 displayed in the terminal with t5 mode:

$$\text{correctionvalue} = - \frac{\text{CurrentEnergyValue} \times \text{Error\%}}{4096}$$

Step 5

- Enter c9[ENTER] to save the values to the info memory of the device.

8 References

1. MSP430FE42x Mixed Signal Microcontroller data sheet ([SLAS396](#))
2. MSP430FE42xA Mixed Signal Microcontroller data sheet ([SLAS588](#))
3. MSP430FE42x2 Mixed Signal Microcontroller data sheet ([SLAS616](#))
4. *ESP430CE1, ESP430CE1A, ESP430CE1B Peripheral Modules User's Guide* ([SLAU134](#))
5. *MSP430x4xx Family User's Guide* ([SLAU056](#))
6. *MSP430 Family Application Report Book* ([SLAA024](#))

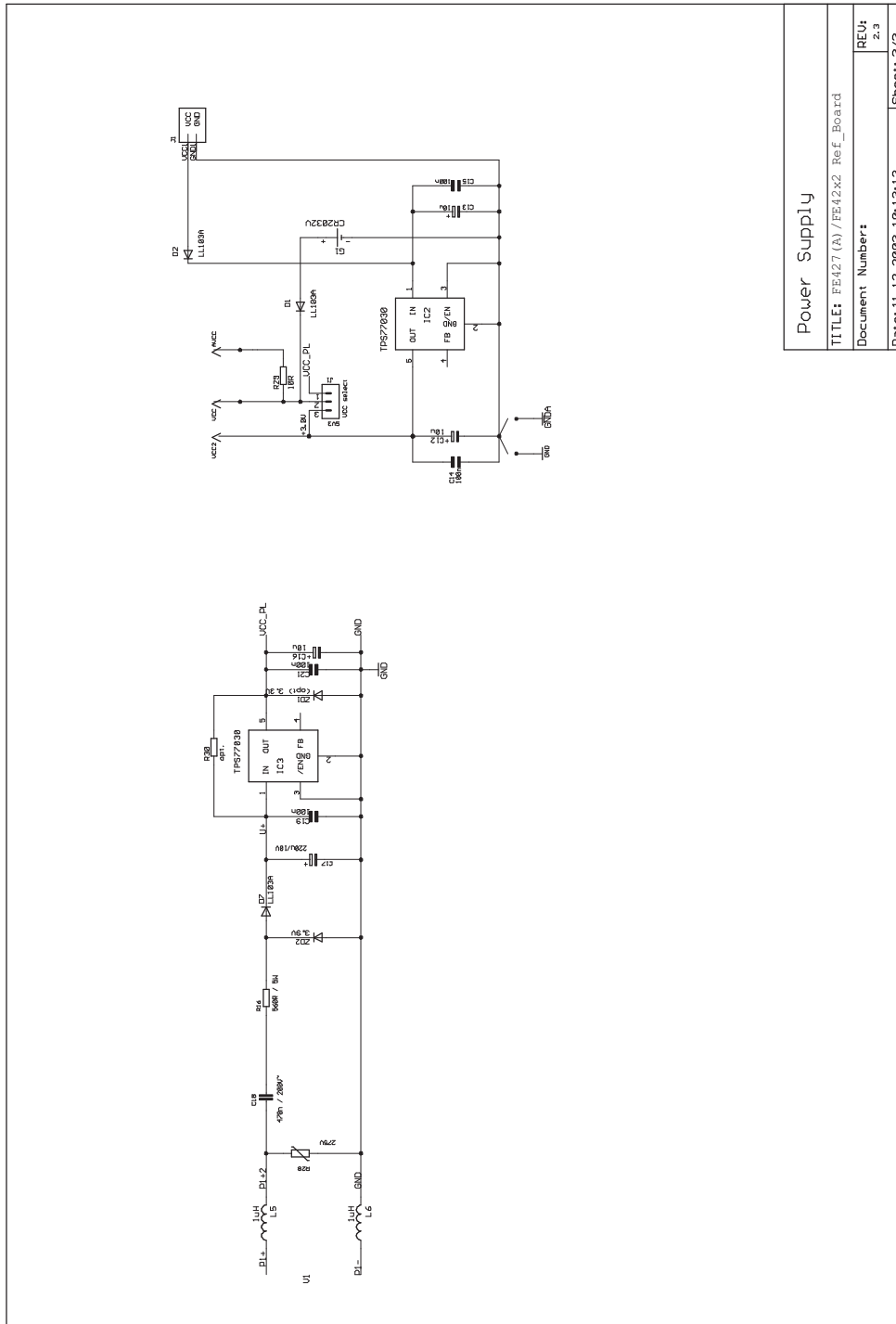
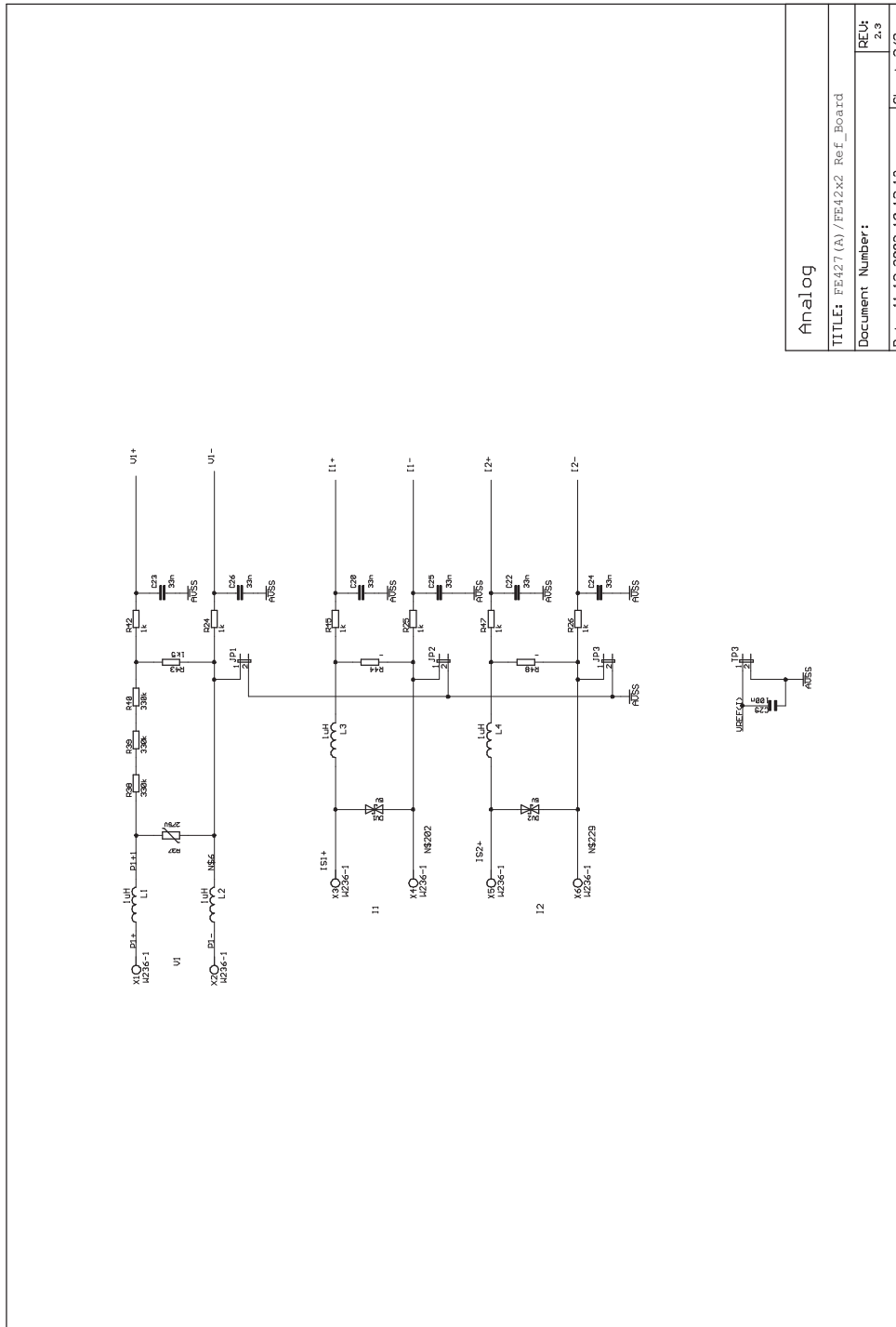


Figure A-3. Schematics (2 of 3)



| | |
|-------------------------------------|-------------|
| Analog | |
| TITLE: FE427 (A) / FE42x2 Ref_Board | |
| Document Number: | REU: 2,3 |
| Date: 11.12.2003 10:13:13 | Sheet: 2/3 |

Figure A-4. Schematics (3 of 3)

Table A-1. Bill of Materials (continued)

| Quantity | Parts | Value | Device | Manufacturer |
|----------|------------------------------|---------------------------|------------------------------------|-----------------------|
| 1 | C27 | 22 μ F / 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 | 3.0 V, 3.3 V, or variable | TPS77030, TPS77033, or TPS77001 | Texas Instruments |
| 1 | J1 | SUP_CON | SUP_CON | |
| 6 | L1, L2, L3, L4, L5, L6 | 1 μ H | 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 | | MSP430FE42x(A) or MS430FE42x2 | Texas Instruments |
| 1 | Q1 | 32 kHz | QUARZ32K | |
| 1 | Q3 | BC807-16SMD | BC807-16SMD | Philips (5Ap) |
| 1 | R1 | 47 k Ω | R_0805 | |
| 1 | R16 | 560R / 5 W | R-EU_0817/7V | |
| 2 | R19, R20 | 820 | R_0805 | |
| 3 | R2, R3, R4 | 820 k Ω | R_0805 | |
| 2 | R21, R22 | 100 k Ω | R_0805 | |
| 1 | R23 | 0R | R_0805 | |
| 6 | R24, R25, R26, R42, R45, R47 | 1 k Ω | 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 k Ω | R_0805 | |
| 1 | R43 | 1k5 | R_0805 | |
| 2 | R44, R48 | - | R_0805 | |
| 1 | R5 | 0 | R_0805 | |
| 1 | R6 | 10 k Ω | R_0805 | |
| 1 | R7 | 470 k Ω | 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 | |
| 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 on the MSP430FE42x.

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?

FE42x

The reactive power is not measured by the ESP430CE1; 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.

FE42xA and FE42x2

The ESP430CE1A/B calculates the reactive power independently of the active and apparent power results directly from the current and voltage samples. Refer to the ESP430 User's Guide ([SLAU134](#)) for exact formulae.

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) With the FE42x, 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 ESP430CE 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

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.

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 is not cleared and is also 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 are 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 outgoing mailbox registers; this reflects whether or not the ESP has read the prior message.

8) In our design we only use one current channel, but we would like use the second 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 that is not currently included in the user's guide and in the header file which requests a 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 is enabled, and after four measurement cycles the result is sent via the mailbox to the MSP430 CPU.

9) I would like to know if my load is capacitive/inductive and hence use the CAPIND register. It works when my current is measured using I1 but does not update or give me an expected result when current I2 is used for measurements.

The CAPIND register is present only in the FE42x (ESP430CE1) and not on the FE42xA (ESP430CE1A) and the FE42x2 (ESP430CE1B) devices. In the FE42x CAPIND functionality has not been implemented for current channel I2.

In the FE42xA and FE42x2 devices, there is no CAPIND register implementation. However, the reactive energy is now signed and can be used to indicate capacitive or inductive load, which now works for both currents I1 and I2. Refer to the ESP430 User's Guide ([SLAU134](#)) for load type interpretations.

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 TI 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. Information of third parties may be subject to additional restrictions.

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.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

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

Products

| | |
|-----------------------------|--|
| Amplifiers | amplifier.ti.com |
| Data Converters | dataconverter.ti.com |
| DLP® Products | www.dlp.com |
| DSP | dsp.ti.com |
| Clocks and Timers | www.ti.com/clocks |
| Interface | interface.ti.com |
| Logic | logic.ti.com |
| Power Mgmt | power.ti.com |
| Microcontrollers | microcontroller.ti.com |
| RFID | www.ti-rfid.com |
| RF/IF and ZigBee® Solutions | www.ti.com/lprf |

Applications

| | |
|--------------------|--|
| Audio | www.ti.com/audio |
| Automotive | www.ti.com/automotive |
| Broadband | www.ti.com/broadband |
| Digital Control | www.ti.com/digitalcontrol |
| Medical | www.ti.com/medical |
| Military | www.ti.com/military |
| Optical Networking | www.ti.com/opticalnetwork |
| 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
Copyright © 2009, Texas Instruments Incorporated