Energy Monitoring & Power Factor Improvement System

Power System Laboratory

EEE-306

Bangladesh University of Engineering and Technology

Course No. EEE-306

Course Title: Power System Laboratory

A Report on

Energy Monitoring & Power Factor Improvement System

Prepared For

Md. Hasibul Alam
Assistant Professor

Md. Shahadat Hasan Sohel
Lecturer

Prepared By

Md. Meraz Al Nahian (1106123)
Homeira Kafi (1106124)
Shuvangkar Chandra Das (1106125)
Sakibul Alam (1106126)
Level **3** Term **1**
Section **B2**

Forwarding Letter

23 May, 2015

Md. Hasibul Alam
Md. Shahadat Hasan Sohel
Department of Electrical and Electronic Engineering,
Bangladesh University of Engineering & Technology, Dhaka.

Dear Sir,
It is an honor for us to have the opportunity of submitting the report titled "Energy Monitoring & Power Factor Improvement". We would like to take this occasion to express our sincere gratitude to you for the support and encouragement you have always extended so generously to us.

Power factor is an important parameter while supplying power by the generation side and is also important for the consumers. Due to low power factor the consumers are charged more than they consume. So it is customary to improve the power factor up to 1 so that the supplied power is consumed by the consumers. In this report we have explained a way to improve power factor by connecting parallel capacitors according to the amount of load using microcontroller. The report covers the circuit diagram along with total explanation of the code used in the microcontroller and the operation of the other IC‟s.

In this report we have left no stones unturned to analyze all the aspects regarding microcontroller based power factor improvement by connecting parallel capacitors. Finally we would like to express our sincere apologies for any inadvertent mistakes present in the report.

Sincerely yours,
Md. Meraz Al Nahian , Homeira Kafi, Shuvangkar Chandra Das, Sakibul Alam.

Contents

List of Illustrations

Introduction

Today the ever growing hunger for more power is pushing us to find more effective ways to curb our energy consumption and reduce energy loss. Energy Monitoring is crucial to modern electrical system and an integral part of today’s tech savvy smart homes. The purpose of such a system is to provide in depth data of the day to day energy consumption pattern and also locate losses in the system.

Another important requirement of today’s homes is a PFI system. With growing number of homes with elevators, Air Conditioners and other inductive loads, losses due to power factor is an important issue. So an in house power factor improvement system may reduce losses as well as clamp down energy bills and at the same time serve the overall grid efficiency.

Goal of this project is to build a home user targeted Energy Monitoring System with Power Factor Improvement Capabilities. Components used in this project were locally sourced and made to be cost effective as possible.

What is an Energy Monitoring System?

An Energy Monitoring system is a computer aided tool to monitor, analyze, control and optimize performance of an energy system. Such a purpose built system monitors the energy consumption a system and provides useful insights into the systems performance and also help locate faults or suboptimal performance.

Usage of Energy Monitoring System

1. Measure energy consumption in a building or organization.
2. Provides analytical data of energy consumption so that we can investigate on energy saving by replacing equipment.
3. Utility bill calculation.
4. Remote monitoring.
5. Automatically Power factor correction.
6. Can measure efficiency of a particular electric machine like transformer etc.
7. Analyzing data, we can make decision how to make best use of daylight.
8. Tracking progression of energy savings.
9. Reduce cost by keeping the energy consumption under the maximum unit limit which is cost effective.
10. Suppose a company develop an Inverter. And they want to supply their products among consumers and they also want to track the consumer misuses of the Inverter, Like not plugging with motors, Refrigerators etc. So that the developer company easily take decision which field they should pay attention.

What is Power Factor?

In AC electrical system Power Factor is the ratio of **Real Powe**r through a load, to the **Apparent Power** in the circuit. It signifies that the voltage and current waveforms are not in phase thus reducing the instantaneous power from the product of two waveforms. Power Factor **(PF)** ranges from -1 to 1. For pure resistive loads PF is 0 (zero). A negative PF signifies that the load is inductive while a positive PF signifies the load is capacitive.

An Analytical approach to PF is shown below

Instantaneous Power *p(t)* absorbed in an element is the product of the instantaneous voltage, *v(t)* & instantaneous, *i(t)*. If voltage & current waves are sinusoidal, then

$$Let, \quad v(t) = V_o sin(\omega t + \theta_v)$$

$$\& \quad i(t) = I_o sin(\omega t + \theta_i)$$

Where V_o & I_o are the peak values of the voltage & current wave respectively.

Then, Power $p(t) = v(t)i(t)$

$$= V_o I_o sin(\omega t + \theta_v) sin(\omega t + \theta_i)$$

$$= \tfrac{1}{2} V_o I_o cos(\theta_v - \theta_i) - \tfrac{1}{2} V_o I_o cos(2\omega t + \theta_v + \theta_i)$$

A more practical and convenient approach is to measure the average power. Average power is time dependent and depends on the phase difference θ of the voltage and current waves

$$Average\ Power, P = \frac{1}{T}\int_0^T p(t)dt = \frac{1}{2} V_o I_o cos(\theta_v - \theta_i) = \frac{1}{2} V_o I_o cos\theta$$

In analysis the RMS values of voltage and current are generally used, $|\boldsymbol{V}| = \frac{V_o}{\sqrt{2}}$ and $|\boldsymbol{I}| = \frac{I_o}{\sqrt{2}}$,

In phasor form $V = |V|\angle\theta_v$ and $I = |I|\angle\theta_i$

Then, $P = \frac{1}{2} V_o I_o cos\theta = \frac{V_o}{\sqrt{2}} . \frac{I_o}{\sqrt{2}} . I_o cos\theta = |V||I|cos\theta = S \times PF$

Here $S = |V||I| = \boldsymbol{VI}^*$ is the complex power.

And Power factor $\boldsymbol{PF} = {}^{\boldsymbol{P}}/_{\boldsymbol{S}}$

Thus the power factor is the ratio of Real Power through the load to the Complex Power. For Sinusoidal waveforms $PF = cos\theta$

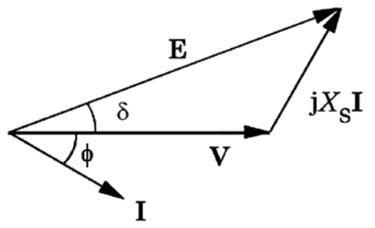


Figure 1 Lagging Power Factor

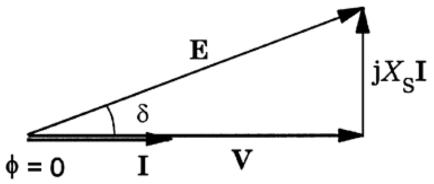


Figure 2 Unity Power Factor

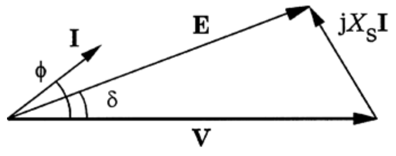


Figure 3 Leading Power Factor

For Inductive load PF is *'lagging'* as the current wave is lagging behind the voltage. For Capacitive load PF is *'leading'*. For purely resistive loads PF is *'unity'.*

Low Power Factor in the System

Power Factor is defined as the ratio of Real Power (**KW**) and Apparent Power (**KVA**). It is obvious that low power factor results when **KW** is small in relation to **KVA**. This occurs when Reactive Power (**KVAR**) in the system is large. Inductive loads like induction motor in elevators and Air Conditioners cause large **KVAR** in the system.

Reactive Power (**KVAR**) required by inductive loads increases the amount of apparent power (**KVA**) in the distribution system. This increase in reactive power and apparent power results in a larger angle θ (measured between KW and KVA). As θ increases, $cos\theta$ (or **PF**) decrease.

So, Inductive loads (with large **KVAR**) results in low power factor.

Why Improve Power Factor?

Lower Utility Fee

Inductive loads demands Reactive Power (**KVAR**) which causes increase in required Apparent Power (**KVA**), which is what utilities supply. So a lower power factor due to inductive loads demands more power generation and transmission capacity in order to meet extra demand. Improving power factor by raising it will lower the demand for power generation and transmission saving costs. Furthermore utilities usually charge customers additional fee when power factor is lower than a certain limit. Improving power factor one can avoid additional fees.

Increase in System Capacity

Improved PF will result in more real power supplied to the load by the system. For example a 1000KVA system running at 0.80 PF provides only 800 KW real power and 600 KVAR reactive power. By raising the power factor to 0.90 the real power capacity of the system increases to 900 KW and only 436 KVAR is supplied.

Reduction of System Loss

Lower PF in the system causes losses in the distribution system. Less real power reaches the load and efficiency drops. By correcting PF, apart from decreasing losses, additional loads can be added to the same system.

Means of Improving Power Factor

As sources of reactive power lower the power factor, consumers of power factor like capacitors, synchronous generators and synchronous motors can be used to improve the power factor. For large scale systems synchronous generators and motors may be used while in small scale systems capacitor banks are used to improve power factor.

By adding a capacitor, the reactive impedance, X can be made smaller. Thus the value of the phase angle of the impedance becomes smaller which results in a smaller phase difference between the voltage and current waves. As a result the power factor increases. Again by adding a capacitor the value of the phasor load current can be made small, causing relatively less line loss. So adding suitable capacitors to the line the power factor can be kept as close to unity ad possible. By choosing a suitable size of C, current wave can be kept in phase with oltage, implying **PF = 1.**

Analysis of the Circuit

DC Supply for MCU

The main line is stepped down to 12V using a 230V-12V transformer. The stepped down input is passed through a bridge rectifier using 4 diodes which causes a full wave rectification of the input and produces a DC supply. The output is fed to a 7805 (linear voltage regulator IC), which produces a constant 5V and drives the microcontroller and other components. Additional capacitors are added to the circuit to remove any remaining AC component and thus improving the output of 7805.

Sampling of Voltage and Current

The sampling of the voltage and current waves are done through ADC of the microcontroller. As the microcontroller requires a positive input in the range of 0V to 5V, the input line was fed to a step down transformer and the voltage on the sampling side was brought down to 5V p-p. Then the now stepped down wave form was clamped above the negative range by offsetting it by 2.5V using a simple voltage divider. This Final wave form is a 5V p-p ranging from 0V to 5V. This was fed to the microcontroller ADC.

Similarly the Current was detected using Current Transformer (**CT**). The stepped down current was sampled from the voltage drop in an appropriately selected resistor. This resulted in a 5V p-p voltage ranging from 0V to 5V which was fed to another ADC of the microcontroller.

The ADC of the microcontroller takes analog reading of the instantaneous Voltage and Current Waves and determines phase difference of the waves in code.

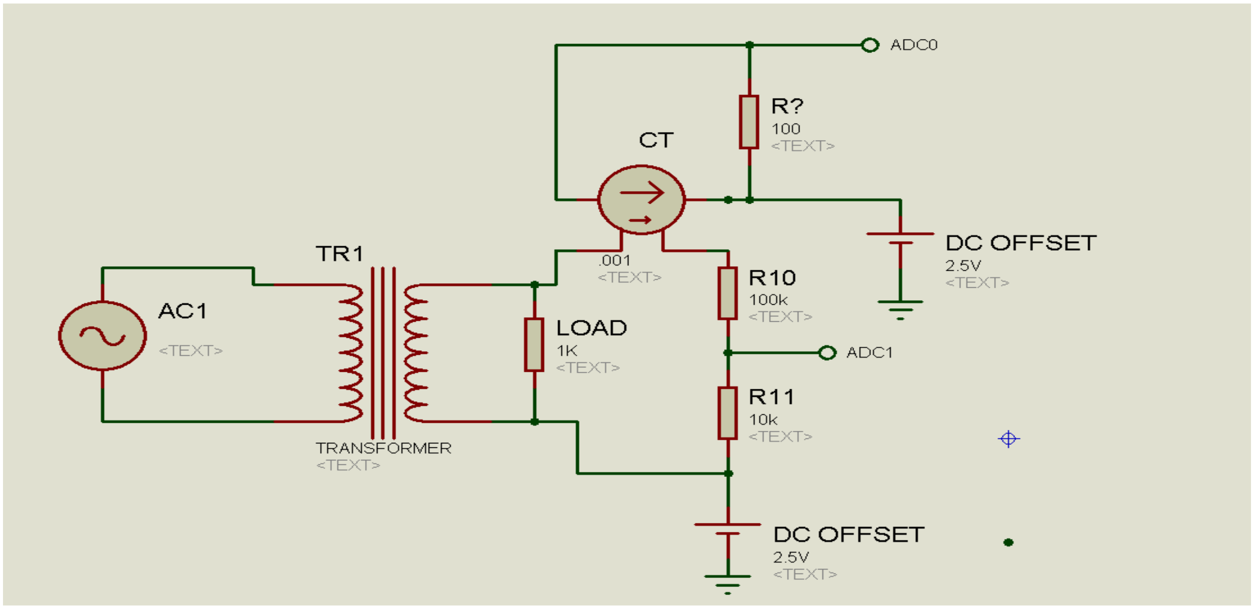


Figure 4 Sampling Circuit

Determination of the Type of Load

The microcontroller takes samples of the voltage and current waves through ADC. From the sampled signals calculations are done in code to determine which wave is lagging or leading. Based on the results the type of load is determined.

Microcontroller

A microcontroller was used as the control unit. It was used to analyze the sampled wave forms, log data for energy monitoring, and to switch the PFI capacitor banks. Our device of choice was ATMEGA32. It is a 40 pin, 8-Bit Microcontroller with 32K of program space, 32 I/O lines, 8 of which are 10bit Analog to Digital converter capable.

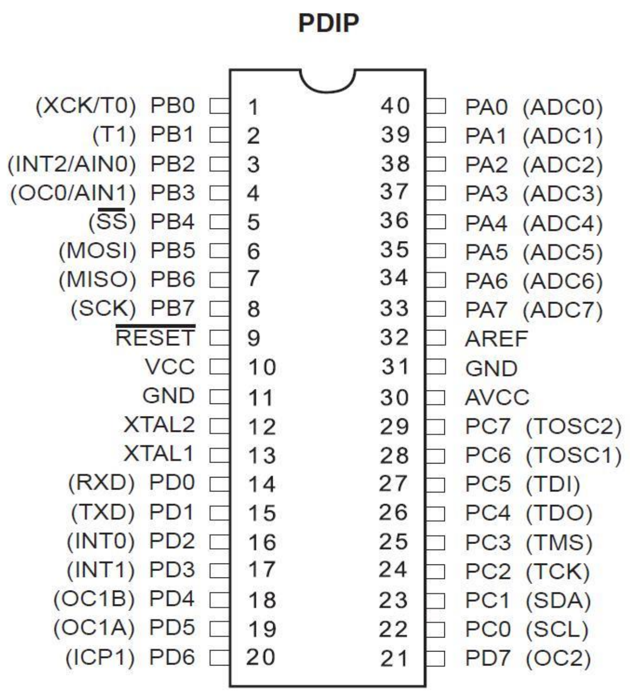


Figure 5 ATMEGA32 MCU

We used ADC pins 0 and 1 for sampling voltage and current respectively. We used Port B pin PB0 to drive the capacitor switching relay. We used to pins on Port D for serial Communication with a Computer through USART.

Activating Capacitors

Capacitors used to improve power factor are connected through relay switches. The relays are driven by 5V supplied by a control pin on the microcontroller. The relays are NC or normally not connected. When 5V is supplied to the relays, the coils in the relay are energized and the line is closed thus connecting the capacitor in parallel to the load.

Data Logging

A crucial component of the system is to monitor and log energy consumption. The microcontroller takes reading continuously and relevant data, for example Voltage, Current, Power and PF are continuously reported back to a computer. Furthermore the output of the PFI may also be shown on the LCD display.

Simulation Circuit

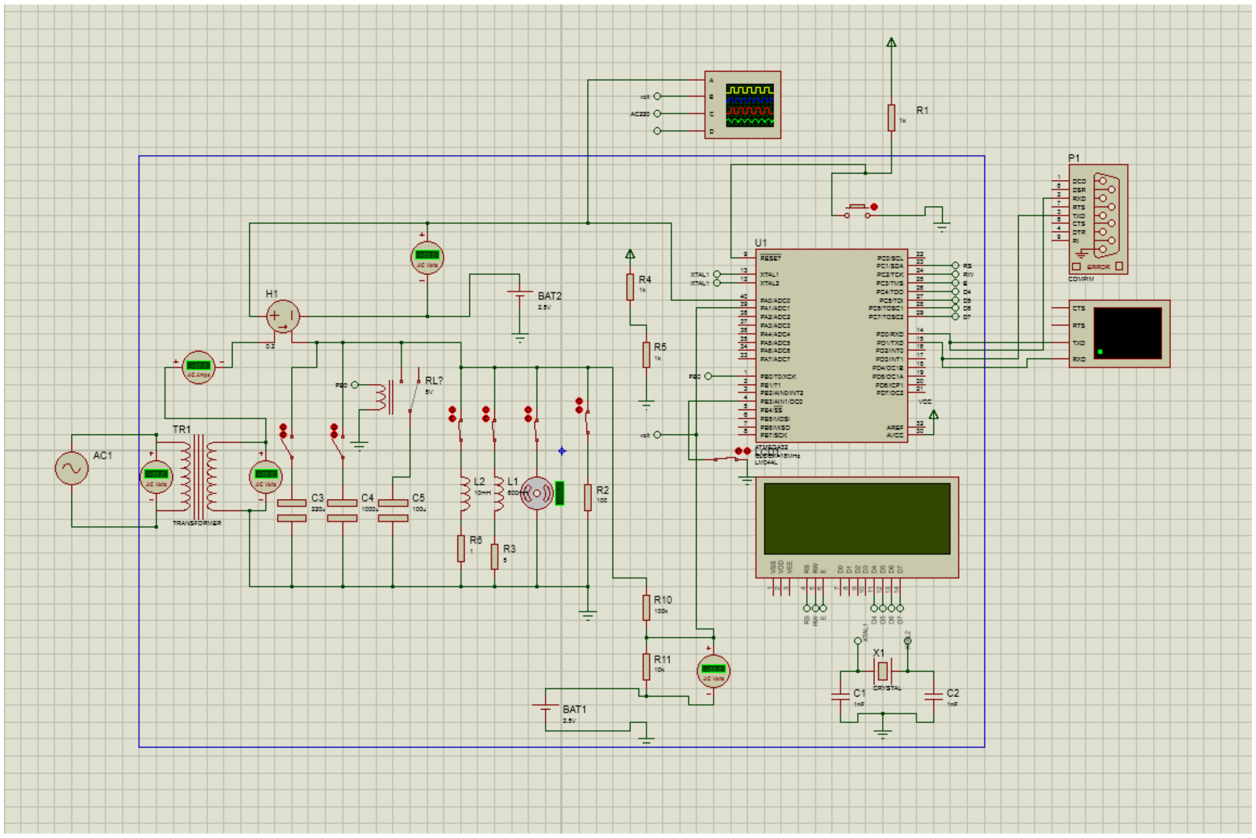
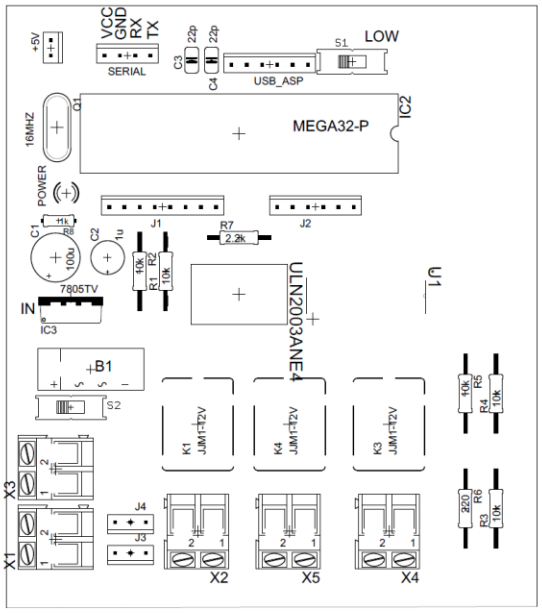
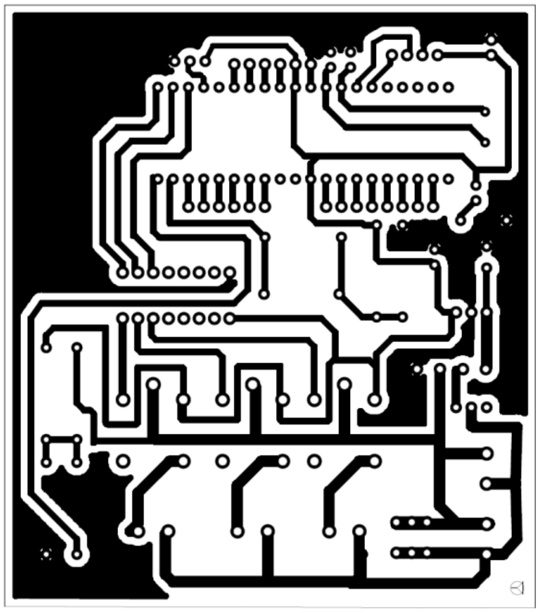


Figure 6 Simulation Circuit Diagram

Proposed PCB



TOP SILK



BOTTOM COPPER

Figure 7 Proposed PCB

Implemented Algorithm

Hardware Side

The voltage of the line was stepped down to 5V p-p wave form ranging from -2.5V to 2.5V. This is not suitable for microcontrollers as it cannot read negative voltage.

So, the resulting wave form was offset by 2.5V using a simple voltage divider to divide the 5V supply of the MCU system. This resulting in a 5V p-p wave ranging from 0V to 5V (only positive side).

Similarly the current taken through a current transformer is connected to a suitable resistance. The voltage drop through the resistance is a 5V p-p voltage wave dependent on the current wave in the line. This resultant wave form is offset by 2.5V to generate a 5V p-p signal ranging from0V to 5V.

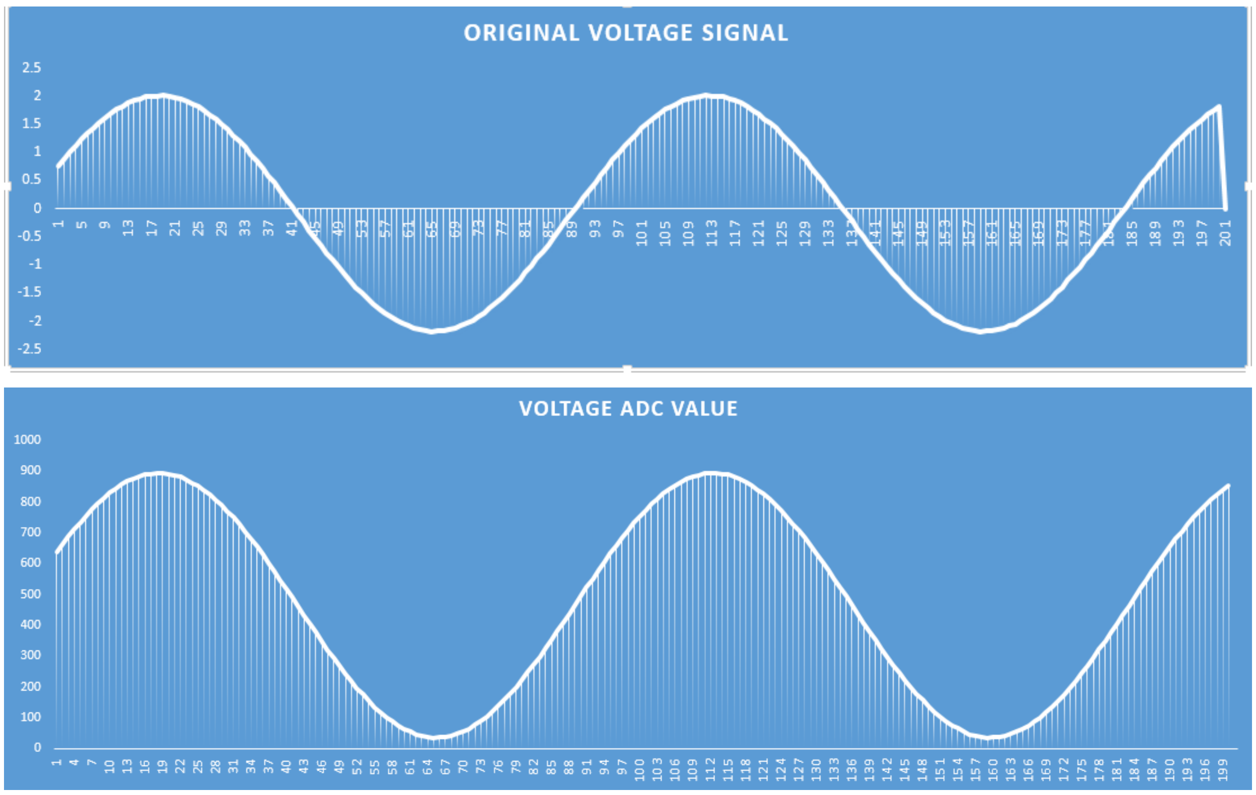


Figure 8 Offsetting input wave form

Software Side

Determining Power Factor

The instantaneous values of the voltage and current wave forms are sampled through the microcontroller's ADC pins. Several sample reading are taken (in our case 200 reading was taken) which is then analyzed to find PF.

The working formula to find PF is

$$\frac{1}{T}\int_0^T v(t)xi(t)dt = V_{rms}I_{rms}cos\theta$$

$$\text{So, } cos\theta = \frac{\frac{1}{T}\int_0^T v(t)xi(t)dt}{V_{rms}I_{rms}}$$

So we need Average Power and Apparent Power. Since we are taking discrete readings, this equations may be better written as

$$V_{rms} = \sqrt{\frac{(v_1^2 + v_2^2 + v_3^2 + \ \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots v_N)}{N}}$$

$$I_{rms} = \sqrt{\frac{(i_1^2 + i_2^2 + i_3^2 + \ \dots\dots\dots\dots\dots\dots\dots\dots\dots i_N)}{N}}$$

$$P_{average} = \frac{1}{N}\sum_{k=0}^{N} v(k)xi(k)$$

$$Power\ factor = \frac{P_{average}}{V_{rms}xI_{rms}}$$

This values are easily calculated from the 200 samples taken.

Determining Type of Load
We need to find out which wave is lagging. To do this we analyze samples taken of the voltage and current wave.

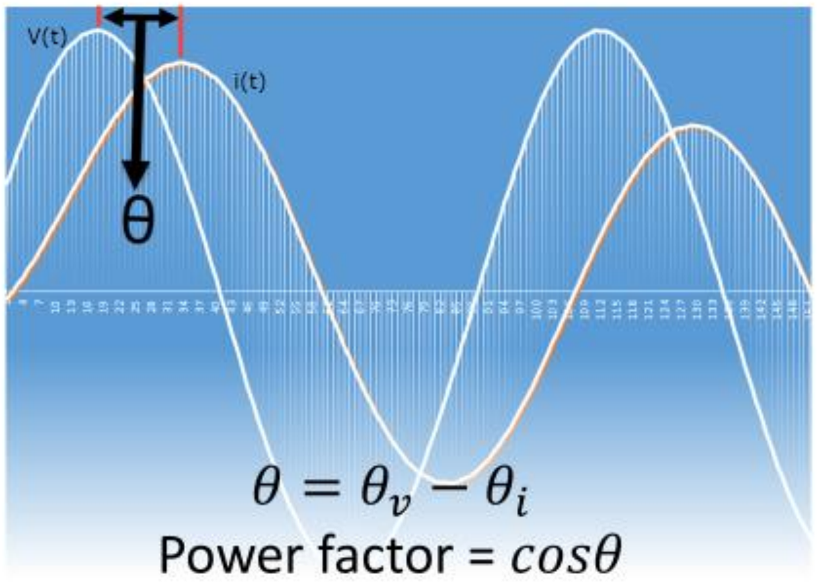


Figure 9 Power Factor from wave forms

In order to do this we find the peak of two wave forms (voltage & current wave). But the peaks may contain ripples and spikes so it is not reliable to depend on the peak of the wave forms. For a more decisive result we determine the zero crossing moment of the waves after a peak is reached. Comparing this moment of the two waves we are able to determine the lagging wave.

If time required by current wave to reach zero after the peak is t_i and that required by voltage wave is t_v.

Then, If $t_i < t_v$ => Load is lagging

If $t_v < t_i$ => Load is leading

Microcontroller Code

Main

```c
/*
 * PFI_test1.c
 *
 * Created: 4/21/2015 9:02:08 PM
 *  Author: Shuvangkar Shuvo
 */


#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
#include <math.h>
#include "ADC_single_conversion.h"
#include "usart.h"
#define current_pin 1
#define volt_pin 0
#define number_of_samples 200
#define sample_delay 700
#define enable_adc() ADCSRA |=(1<<ADEN)
#define disable_adc() ADCSRA &= ~(1<<ADEN)
#define F_CPU 16000000

int voltage_adc[number_of_samples]={0},current_adc[number_of_samples]={0},index_volt=0,index_current=0;;
uint8_t i=0,j=0,k=0,l=0,n=0;
double power_factor=0;
double  average_volt=0,average_current=0,actual_volt=0,actual_volt_1=0;actual_current=0,Vrms=0,Irms=0,V_avg=0,I_avg=0,volt_square_sum=0,current_squared_sum=0,real_power=0;

int main(void)
{
    setup();
    while(1)
    {
        calculate_power_factor();

        index_volt = voltage_zero_cross_time();
        index_current = current_zero_cross_time();

        if((index_current&&index_volt)==0)
        {
            index_volt = voltage_zero_cross_time();
            index_current = current_zero_cross_time();
        }

        UWriteString("\n V_avg= ");
        print_double(V_avg,4,3);
        UWriteString("\n I_avg=");
        print_double(I_avg,4,3);
        UWriteString("\n voltage index=");
        print_integer(index_volt);
        UWriteString("\n current index=");
        print_integer(index_current);
        //_delay_ms(20000);
        if (power_factor<0.9)
```

```
          PORTB |=1<<PB1;
        //enable_adc();
    }

}

void setup() {

        ADC_Initialize();
        Init_timer1_normal_mode_with_overflow();
        USARTInit(25);
        DDRB |=1<<PB0;
        DDRB |= 1<<PB1;
        DDRB |= 1<<PB2;
        _delay_ms(10);
}

ISR(TIMER1_OVF_vect) {
        PORTB ^=1<<PB0;
}

void voltage_current_read() {

    for (i=0;i<number_of_samples;i++)
    {
        voltage_adc[i]=ADC_Read(volt_pin);
        //_delay_us(700);;
        current_adc[i]= ADC_Read(current_pin);
        //_delay_us(700);
    }
}

void print_voltage_current()
{
    voltage_current_read();
    UWriteString("\n voltage \n");
    for (i=0;i<number_of_samples;i++)
    {
        print_integer(voltage_adc[i]);
        UWriteString("\t");
    }
    UWriteString("\n Current \n");
    for (i=0;i<number_of_samples;i++)
    {
        print_integer(current_adc[i]);
        UWriteString("\t");
    }
    disable_adc();
}

Init_timer1_normal_mode_with_overflow()
{
    TCCR1B |=(1<<CS10)|(1<<CS11); //timer 1 is activated with prescaler 64
    TIMSK |= 1<<TOIE1; //Overflow interrupt enalbe, so when timer reaches its maximum value, it overflows and overfl
ow interrupt vector routine will be executed
    sei(); // enable global interrupt
}

```

```


void calculate_power_factor()
{
   voltage_current_read();
   ADCSRA &= ~(1<<ADEN);
   //print_voltage_current();
   //Average voltage and current calculation
   for (n=0;n<number_of_samples;n++)
   {
      average_volt += (5.00*voltage_adc[n])/1024.00;
      average_current += (5.00*current_adc[n])/1024.00;

   }
   average_volt = average_volt/number_of_samples;
   average_current = average_current/number_of_samples;
   UWriteString(" \n Average voltage = ");
   print_double(average_volt,4,3);
   UWriteString(" \n Average current = ");
   print_double(average_current,4,3);

   UWriteString(" \n Actual voltage \n");
   volt_square_sum = 0;
   current_squared_sum=0;
   V_avg=0;
   for (n=0;n<number_of_samples;n++)
   {
      actual_volt = ((5.00*voltage_adc[n])/1024.00)-average_volt;
      V_avg = V_avg+actual_volt;
      volt_square_sum = volt_square_sum+pow(actual_volt,2);
      print_double(actual_volt,4,3);
      UWriteString("\t");

   }
   UWriteString(" \n Actual current \n");
   I_avg=0;
   for (n=0;n<number_of_samples;n++)
   {
      actual_current = ((5.00*current_adc[n])/1024.00)-average_current;
      //I_avg=I_avg+actual_current;
      current_squared_sum = current_squared_sum+pow(actual_current,2);
      print_double(actual_current,4,3);
      UWriteString("\t");


   }

   Vrms = volt_square_sum/number_of_samples;
   Vrms=sqrt(Vrms);
   Irms = current_squared_sum/number_of_samples;
   Irms=sqrt(Irms);
   UWriteString("\n Vrms = ");
   print_double(Vrms,4,3);
   UWriteString("\n Irms = ");
   print_double(Irms,4,3);

   //Instantaneous power and real power
   real_power = 0;
```

```
    for (n=0;n<number_of_samples;n++)
    {
        actual_volt = ((5.00*voltage_adc[n])/1024.00)-average_volt;
        actual_current = ((5.00*current_adc[n])/1024.00)-average_current;
        real_power = real_power+(actual_volt*actual_current);

    }
    real_power = real_power/number_of_samples;
    power_factor = real_power/(Vrms*Irms);
    UWriteString("\n Power factor = ");
    print_double(power_factor,4,3);

}
int voltage_zero_cross_time()
{
}


int current_zero_cross_time()
{
    int temp = 0,current_peak=0;
    for(i=0;i<number_of_samples;i++)
    {
        temp=current_adc[i];
        if(temp < current_peak) //Max value found
        {
            for (j=i-1;j<number_of_samples;j++)
            {
                actual_current = ((5.00*current_adc[j])/1024.00)-average_current;
                if (actual_current<0)
                {
                    if (j==0)
                    {
                        enable_adc();
                        continue;

                    }
                    return(j);
                    j=number_of_samples;
                    i=number_of_samples;
                }
            }

        }
        else
        {
            current_peak = temp;

        }
    }

}

```

ADC Conversion

```c
/*
 * ADC_single_conversion.c
 *
 * Created: 4/18/2015 12:55:23 PM
 *  Author: Shuvangkar Shuvo
 */
#include <avr/io.h>
#include "ADC_single_conversion.h"


void ADC_Initialize() {
   ADCSRA |= 1<<ADEN; //ADC enable. ADC doesn't consume power when ADEN is cleared
   ADCSRA |=(1<<ADPS0)|(1<<ADPS1)|(1<<ADPS2); // prescaler=128 means X_cpu/128=7.8125KHz
   ADMUX |= 1<<REFS0; //Reference: AVCC (AVCC must be connected with VCC)
}

unsigned int ADC_Read(uint8_t channel) {
   int adc_value = 0;
   //ADMUX |= (ADMUX&0b11100000)|(channel&0b00011111); // First five bits of ADMUX for selecting channel
   ADMUX = (ADMUX&0xf0)|(channel&0x0f);
   ADCSRA |=1<<ADSC; //Conversion will start writing this bit to one.
   while(!(ADCSRA&(1<<ADIF))); //ADIF will set when conversion complete and loop breaks.
   ADCSRA|= 1<<ADIF; //ADIF must be cleared(1) to trigger a new conversion next time
   adc_value = ADCW;
   return(adc_value);
}
```

Parts List

Control Section

1. ATMEGA32 MCU
2. 16x2 LCD Display
3. Bread Board (2pcs)
4. Resistor : 100R,1k,220R,10R,10k,100k,-3 watt
5. 22pf capacitor 2pcs
6. Rail: L rail
7. Connector: Male to female, male to male, female to female
8. Current sensor: 2 leg (model: TA21CM)
9. AC socket (Green)
10. Crystal 16Mhz
11. Variable resistor - 10k
12. 7805-4\5

Load Section

1. Inductor
2. Capacitor: 100uf, 2200uf, 10uf, 47uf
3. Resistor: 10R, 100R, (10watt)
4. Slide Switch (5pcs)
5. SPDT Switch: (3\/4 pcs)
6. AC Relay (4 pcs)
7. BJT, 1A, npn

Discussion

In order to find a suitable solution to measure Power Factor we employed several methods. We tried to use a Zero Crossing detection method using op-amps but failed to get desired results. Using the MCU ADC also resulted in problems as the wave forms we not being sampled correctly or only half of the wave form was detected. The ATMEGA32 ADC has some limitations that we were unware of. Using a dedicated ADC chip we could have taken better readings. Through trial and error we were able to find a suitable solution to using the MCU ADC.

We also had trouble determining load type due to spikes in the input signal. We are able to detect 200 samples and use software corrections to get desired results.

We were unable to use multiple capacitor solution or capacitor banks as we were pressed for time and unable to build required circuit in breadboard. So we opted for a single capacitor to show proof of concept.

We designed PCB to compact the circuit and make it more universal but sadly our PCBs were poorly made and we were unable to transfer the design to PCB.

In conclusion, we were able to overcome many pitfalls we encountered and finish the project and output desired result.