SIMULATED ANNEALING AS AN OPTIMIZING TOOL IN POWER ELECTRONICS

Filip Ivanovski, Goce L. Arsov, Goce V. Shutinoski

Faculty of Electrical Engineering, SS Cyril and Methodius University, Karpos II b.b. P.O.Box 574, 1001 Skopje, Republic of Macedonia; E-mail: ivphilip@yahoo.com, g.arsov@ieee.org, sugo@cerera.etf.ukim.edu.mk

Summary *- An approach of using optimization techniques in designing power electronics systems is presented in the paper. The aim of this optimization is to constrain the THD of the input current, in the case of a Buck Converter fed by a rectifier bridge, to a level compliant to PFC standards. The common way to get ahead of the input current distortion is to utilize active or passive PFC pre-regulation techniques. Having this as a starting point, the accent in the paper is given to utilization of simulated optimization concept in controlling the converter's operation. The aim is to reveal a common pattern scheme of a bit-string governing the active PFC pre-regulator circuit to obtain lowered THD to the level compliant with the predefined conditions. In this paper optimization algorithms of several optimization techniques are presented and compared with the algorithm that simulates the converter itself.*

I. INTRODUCTION

Within the conversion of the AC to the DC power a well-known side effects are very often manifested. The side effects are the harmonic components that are well known generators of numerous problems such as: increase in the operating temperature, problems with electricity meters, malfunctioning of the equipment, discomfort (flickers, lighting). Due to the problems induced by the harmonic distortion the IEC 6100-3-2 [1] standard has been introduced.

Numerous methods for filtering and preregulation have already been applied [2]. The approach presented in this work is a combination of a preregulation technique and optimization of the preregulator by the bit-string governing scheme. The aim to lower the THD is accomplished by generating an optimizing bit-string switching pattern, a scheme that controls the work of the preregulator. This differs regarding the classic PWM approach, as many switching cycles can occur during one cycle of the reference sinewave while, due to the fixed frequency of the PWM only a set numbers of switching transitions can occur during each cycle of the fundamental [3]. Another important factor in the proposed switching scheme is the number of transients that will occur: the higher number of transients gives better results in lowering the lower harmonics, but at the same time the losses that are accounted to the switching transients are higher [4]. Trade off between these two factors is often applied.

The approach presented in this work can simply be described as a way of moving the lower harmonics from its scale to the higher order harmonics in order to be compliant with IEC 6100-3-2.

In order to implement this idea, we have developed a Matlab model of the power converter. Also we have programmed the Simulated Annealing optimization routine (written in Matlab as well) which is fully presented in the Appendix.

II. SIMULATED ANNEALING

Simulated Annealing (SA) is a well known optimization technique that have been inspired by the natural process of metal annealing and was applied as an optimization method for the first time in 1983 [5, 6, 7, 8]. It is regarded as a second generation of the stochastic optimization techniques.

The annealing algorithm simulates the way of processing metals by means of statistical mechanics. The process of annealing is divided in two routines; the first one is the heating of the metal until it melts down and the second is slowly cooling it down until the equilibrium point (crystallization) is reached. At the end of the second routine we gain a new molecular structure of the metal with lower energy state [5].

The same analogy is used in the process of optimization where lower energy state of the metal is presented as a state of the cost function. The whole system of statistical mechanics and its analogy in the filed of combinatorial optimization is presented in the next table.

Table 1 Conversion of thermodynamic system into combinatorial optimization

Thermodynamic Simulation	Combinatorial Optimization
System States	Feasible Solutions
Energy	Cost
Change of State	Neighboring Solutions
Temperature	Control Parameter
Frozen State	Heuristic Solution

Using these mappings any combinatorial optimization problem can be converted into an annealing algorithm [6].

The whole optimization algorithm has two routines, inner and outer. The inner optimization cycles calculates the energy state of the cost function and maps it with appropriate logical value. The cycle goes like this:

1. The process starts with initial energy state S_i that defines the initial cost function;
2. The starting initial state is stochastically changed to S_j and a new energy state is gained;
3. The energy difference between the two states is calculated ΔE;
4. The energy state is mapped with logical one or zero depending of the sign and value of ΔE and the Metropolis acceptance procedure is used as follows:
   a. If $\Delta E \leq 0$, the mapping value is equal to logical 1, meaning that the new energy state is accepted

b. If $\Delta E \geq 0$ and the Metropolis acceptance procedure hasn't accepted the new energy state the mapping value is equal to logical 0, meaning that the new energy state is not accepted;

5. The algorithm jumps back to the step 2 in order to calculate the next value of the cost function or it directs its way toward the outer routine.

The outer cycle follows the inner only if the mapping value of ΔE is equal to 1or ΔE is equal to 0 but the Metropolis procedure has accepted the appropriate solution due to its probability (step 4 in the inner cycle):

1. The system temperature is changing, it is cooling or heating depending of the energy state or the cost function;
2. Checking whether the system is cold enough to stop the cycle or hot enough to start cooling it down with the predefined step;
   The cycle is continuing (if the system has not reached equilibrium or the system is not cold enough), and the new state S_j is memorized;
3. Jump to inner loop step 2

The presented algorithm can vary depending on how the parameters were defined. The inner and the outer optimization cycles are shown in figure 1. Certain Simulated Annealing algorithms follow different routine [7].

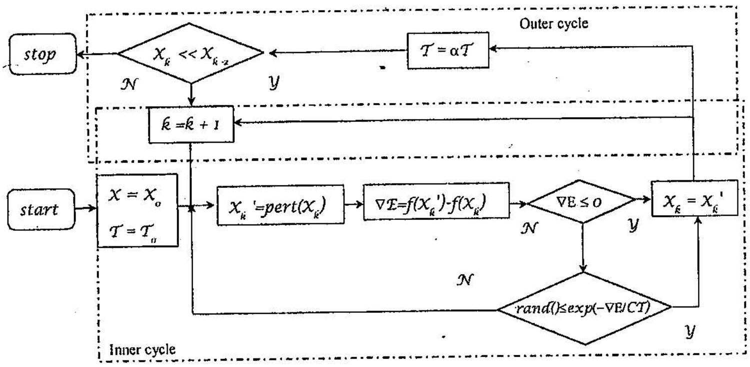
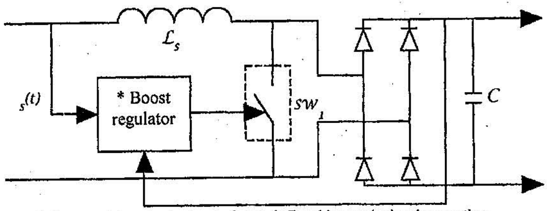


Fig. 1 The inner and outer optimization cycle

III. BUCK-CONVERTER WITH BOOST PREREGULATOR

The optimization algorithm outputs a bit-string that is used to generate a pulse pattern for driving the boost preregulator, figure 2. The Boost-Buck converter is fully modeled in Matlab and it is defined as a subroutine of the optimization cycle for calculating the cost function. The Boost-Buck is completely regulated by the optimization algorithm. The control law of the Boost is defined by the THD function as a cost function while the control law of the Buck as a function that depends of the output voltage. The complete circuit configuration is presented in figure 3.



* Preregulator, logical regulator defined by optimization cycles,

Fig. 2 Logical regulation of preregulator with the bit string defined scheme by the optimization cycles

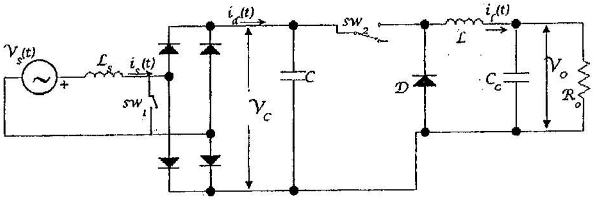


Fig. 3 Buck-converter with Boost preregulator

IV. SIMULATION RESULTS

Simulations using four different bit-string patterns for controlling the preregulator have been performed as follows: classical PWM, Simulated Annealing (SA), Monte Carlo (MC), Simplified Simulated Annealing (SSA) [9]. Some results are presented in Figs. 4 and 5, and in Table 2.

The optimization methods presented have several differences [9] but still they are closely related among each other. The main difference is in the way the optimization cycles are defined and in the Metropolis acceptance procedure that is only present in the Simulated Annealing algorithm.

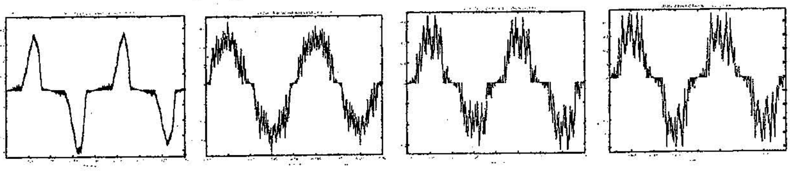


Figure 4 Input current waveforms obtained by four different bit sting patterns controlling the preregulator: PWM, SA, MC and SSA;

Some quantitative results are presented in table 2. The total harmonic distortion (*THD*) was calculated as ratio of rms value of all higher harmonic components and the rms value of the first harmonic component of the input ac current multiplied by 100; the efficiency (η) was calculated as ratio of the mean output dc power and mean input ac power multiplied by 100; ΔV_{out} is the relative difference between the maximal and the mean value of the output voltage multiplied by 100. It is easily seen that the SA optimization gives the lowest value for *THD,* and the highest values for the efficiency and ΔV_{out}.

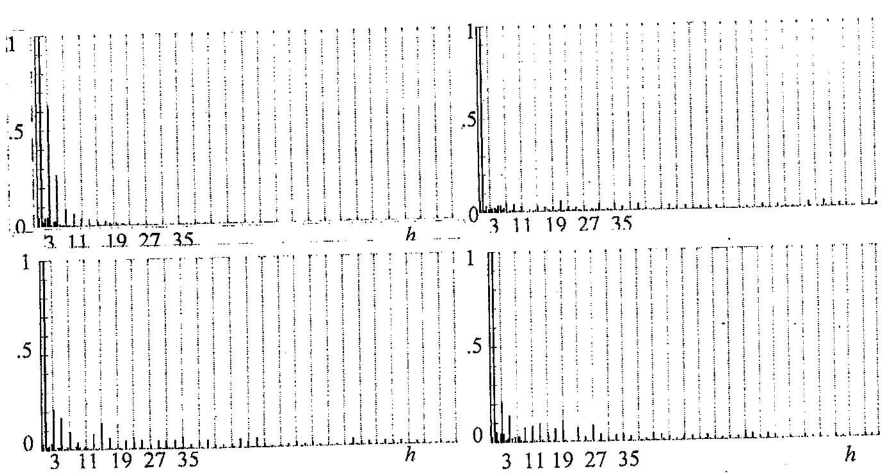


Figure 5 Plotted THD of the input current circuit obtained with four different bit sting patterns governing the preregulator. The bit patterns are as follows: PWM, SA, MC and SSA

Table 2 Comparative results using different methods of optimization and classical PWM

Bit string patterns, obtaining techniques	*THD* (%)	η (%)	ΔV_{out} (%)
PWM	71.1498	0.5014	0.8750
Simulated Annealing (SA)	23.9220	0.7534	0.7397
Monte Carlo (MC)	41.1036	0.5904	0.8810
Simplified Simulated Annealing (SSA)	40.3115	0.6843	0.7862

V. CONCLUSION

The bit-string switching sequence approach has been discussed and applied to the preregulator of a Bridge rectifier-Buck converter combination. In comparison with PWM, MC and SSA it has several advantages. The diagrams shown in Fig. 4 and 5 and the results given in table 2 show that the shape of the input current is closest to the sinusoidal, while the amplitude of the harmonics are the lowest for the case of Simulated Annealing used as an optimization method for generating bit-string patterns for controlling the preregulator. The values of THD should be also compared with those obtained for the non pre-regulator case (189.3%) and for non-optimized preregulator case (66.8%).

REFERENCES

[1] EN61000-3-2 BULLETIN, Amendment 14, November 2000

[2] Cobos J., Arsov G., "*Design for Compliance with Standards – PFC & EMC*", Tempus Phare Joint European Project: Development of Power Electronics Courses – 1999/2000. European Commission Contract No: S_JEP-12103-97

[3] Shaw S. R., Jackson D. K., Denison T. A., Leeb S. B. "Computer-Aided Design and Application of Sinusoidal Switching Patterns", *Compel '98*, Como, pp. 185-191.

[4] Mohan N., Undeland T. M., Robbins W. P., "*Power Electronics: Converters, Applications, and Design*" 2nd ed. New York: Wiley, 1995.

[5] Starck V., "*Implementation of Simulated Annealing optimization method for APLAC circuit simulator*", Master Thesis, Helsinki University of Technology, Oct. 26, 1996

[6] Kirkpatric S., Gelatt C.D. Jr., Vecchi M.P. "Optimization by Simulated Annealing" in "*Science*", 13 may 1983, vol.220, Number 4598
[7] Davis L., "*Genetic algorithms and simulated annealing*", Pitman 1987
[8] Spinellis D., Papadopoulos C., J. MacGregor S. "Large production line optimization using simulated annealing", *International Journal of Production Research*, 38(3): 509–541, February 2000.
[9] Hamill D. C., Bina M. T., "Optimizing a Discrete Switching Pattern Using Two Simulated Annealing Algorithams", *Compel '00*.

Appendix A - *Simulated Annealing Algorithm*

```
clear, clc
N=256;
% Simulated Annealing Parameters
Tinitial=1;      Tcool=0.99;     Theat=1.2;
Melted=0;        Melt_point=5;   Max_T=40;
x_now=[1 0];
for j =1:7
   x_now=[x_now x_now]; zlength=length(x_now);
end
x_best=x_now;
% Initialize annealing variables
ni_best=.7;
index=1:N;
THD_best=70;
THD_last=70;
DV_good=5;
DV_best=5;
T_schedule=[1.0 1.0 0.5 0.33];
% Main annealing LOOP
for h=1:length(T_schedule)
   T=Tinitial*T_schedule(h);
   for j=1:Max_T
      num_accepts=0;
      for i=1:N                    % Bit swapping loop
        % Reconfigure the bits in x.new
        x_new=x_now;
        z=floor(rand(1)*N/2)+1;
        o=floor(rand(1)*N/2)+1;
        in_x_new=1-x_new;
        Lx_new=logical(x_new);
        Lin_x_new=logical(in_x_new);
        i_ones=index(Lx_new);
        i_zeros=index(Lin_x_new);
        x_new(i_ones(o))=0;
        x_new(i_zeros(z))=1;
        kocka =rand(1);
        [delta_V, THD, ni] = buck(x_new);
        DTHD = THD - THD_last;
        if ( DTHD <= 0 )           % Accept -dV always
           THD_last=THD;
           DV_good=delta_V;
           x_now=x_new;
           num_accepts=num_accepts + 1;
        elseif (kocka < exp(-DTHD/T))
          THD_last=THD;
           DV_good=delta_V;
           x_now=x_new;
           num_accepts=num_accepts + 1;
         if (Melted & ((DTHD/THD) >
Melt_point/100) )
             Melted=1;             % Check if melted
             Tinitial=T;
          end
        end
        if (THD <=THD_best)              % Keep a
copy of the best
             x_best=x_new;
             THD_best=THD;
             x_best=x_new;
             DV_best=delta_V;
             ni_best=ni;
        end
        lin=sprintf('delta_V=%5.2f%',delta_V);
        lin=sprintf('%s   THD=%5.2f%',lin,THD);
        lin=sprintf('%s   THD_best=%5.2f%',lin,
THD_best);
        lin=sprintf('%s   DV_good=%5.2f%',lin,
DV_good);
        disp(lin);
      end               % End the bit swapping loop
      if (Melted==1); Melted=2; break; end
   if (num_accepts > 0)    % Cool or heat
        if(h==1)
          T=T*Theat;     % Heat it not Belted yet
        else
          T=T*Tcool;     % Cool if already melted
        end
      else  break; end     % Break out if accepts=0
      S=100;
      if(h==1)
        str=sprintf('Melting=%1.0f Tcyc=%3.0f',h,j);
      else
        str=sprintf('Coolcyc=%1.0f
Tcye=%3.0f',h,j);
      end
      str=sprintf('%s Acpts=%4.1f%', str,
num_accepts/N*S);
      str=sprintf('%s T=%5.2f%' ,str,T);
      str=sprintf('%s THD_best=%6.2f%'
,str,THD_best);
      str=sprintf('%s DV_good=%6.2f%'
,str,DV_good);
      str=sprintf('%s DV_best==%6.2f%'
,str,DV_best);
      str=sprintf('%s ni_best==%4.3f%' ,str,ni_best);
      disp(str);
      sprintf('%2.0f', x_best)
   end                  % End Temperature loo
end                     % End the Reheat loop
```