A Digital Controller For a Unity Power Factor Converter [*]

A. H. Mitwalli[†] S. B. Leeb[†] G. C. Verghese[†] V. J. Thottuvelil[‡]
[†]Laboratory for Electromagnetic and Electronic Systems
Massachusetts Institute of Technology
and
[‡]Digital Equipment Corporation

Abstract

A digital controller for a Unity Power Factor AC-DC converter is designed, based on a linear large-signal model of the power supply. A hardware implementation of the design is presented and analyzed, along with simulations of the closed-loop system. Issues in digital control of power systems, such as quantization effects and fixed-point representation of system parameters, are examined in the context of this system. The experimental results are then compared with the simulations and used to evaluate the implementation.

1 Introduction

With the increasing need for high power factor converters, Unity Power Factor (UPF) supplies have rapidly gained popularity. It is essential that power supplies to be placed in computers and other equipment be robust, reliable, and responsive to perturbations in circuit parameters. Digital controllers hold promise in this regard. This paper describes the implementation of a digital controller for a UPF converter.

A popular scheme for achieving Unity Power Factor uses the circuit in Figure 1. The boost converter in the circuit receives as its input the rectified AC waveform. The inner (current) loop controls the source current to the shape and phase of the input voltage $v_{in}(t)$ by providing a switching sequence for the transistor that forces the inductor current $i_L(t)$ towards a desired current $i_p(t)$, which is in turn made proportional to the input voltage, $i_p(t) = kv_{in}(t)$. The outer (voltage) loop regulates the output voltage v_o to the desired reference voltage V_o by adjusting the proportionality constant k used to generate i_p every line cycle [4].

The next section of this paper describes a linear large-signal model of this power converter and develops a PI controller based on it, [3], [4]. Section 3 lays out the hardware system used for the implementation and details some of its relevant components. The software implementation of the PI controller and additional control features are presented in Section 4. Simulations and experimental results are illustrated and evaluated in Section 5. Section 6 concludes with a summary of the results of the research and an outline for future work.

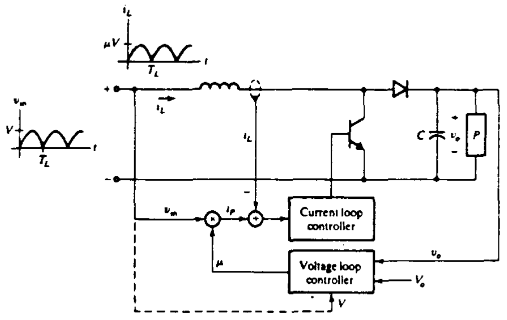


Figure 1: A high power factor ac/dc switching preregulator [3]

2 Modeling and Control Design

There are different approaches to modeling UPF power supplies and designing their control loops. Examples found in [1] and [5] are based on linearizations of nonlinear models of the converter circuits, and therefore their successful operation is guaranteed only in the vicinity of the operating point. A more recent paper [4] develops a linear large-signal model for the voltage loop. The model is based on a simple dynamic power balance equation for the circuit, with the squared output voltage as the state variable. A brief description of the model is presented below.

Let T_L denote the period of the rectified input voltage v_{in}. We assume that the current loop maintains $i_L(t) = i_p(t)$ and we ignore the switching ripple. In

[*]Supported by Digital Equipment Corp. and the MIT/Industry Power Electronics Collegium. We would like to thank A. Stankovic for his discussion and help. Address: Prof. G. Verghese, Rm 10-093, MIT, Cambrige, MA 02139.

the slow model used here, the proportionality factor $k(t)$ (In Figures 1 and 2 this corresponds to $\mu(t)$.) is varied once every rectified line cycle (faster variation of k can also be considered [11]). Modeling the load as a constant-power load that draws power P, the following power balance equation for the boost converter is obtained:

$$\frac{1}{2}C\frac{dv_o^2}{dt} = kv_{in}^2 - k^2\frac{1}{2}L\frac{dv_{in}^2}{dt} - P \tag{1}$$

where i_p has been substituted for i_L and i_{in}, and the relationship $i_p = kv_{in}$ has been used.

For digital control, it is convenient to have a sampled-data model. Let the value of k in the nth cycle be $k[n]$. Now integrating (1) over T_L and denoting v_o^2 at the beginning of the nth cycle by $x[n]$, we get the following equation:

$$\frac{1}{2}C(x[n+1] - x[n]) = T_L(k[n]\frac{V^2}{2} - P) \tag{2}$$

or

$$x[n+1] = x[n] + \frac{T_L V^2}{C}k[n] - \frac{2T_L}{C}P \tag{3}$$

This is a state-space representation of a first-order LTI discrete-time system, with control input $k[n]$ and disturbance input P.

References [3] and [4] outline the design steps for a digital controller for the outer voltage loop. It is basically a discrete-time version of a PI controller, and is aimed at regulating $x = v_o^2$ to its reference value $X = V_o^2$, despite constant errors in the model and constant disturbances to the system. Figure 2 demonstrates this control method. Instead of the integrator, an accumulator is used. Its output in the nth cycle is the sum of all its past inputs. The closed-loop system will reach a constant steady state if it is stable and P is constant. In this steady state, the output of the accumulator is constant, meaning that its input is zero. The steady state reached is thus $v_o^2[k] = V_o^2$, i.e. one in which we have driven the output voltage to the desired reference point. Stability is obtained by proper choice of the gains h_1 and h_2. These gains have to be chosen so that the roots of the following characteristic polynomial have magnitude less than 1:

$$z^2 - (h_1 + 2)z + 1 + h_1 - h_2 \tag{4}$$

3 Hardware Implementation

A digital implementation of the controller can add to the robustness of the system by allowing a more complex and flexible design, and a highly repeatable implementation. The disadvantages, however, are higher cost, and performance degradation due to sampling and quantization [6]. In the following sections, the various components used to implement different sections of the layout in Figure 1 are presented, and the overall implementation discussed.

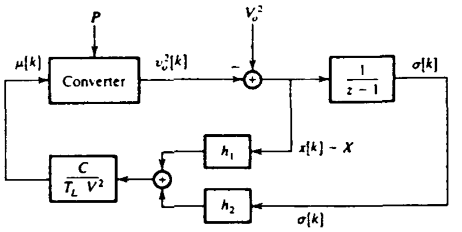


Figure 2: A discrete-time PI controller for the UPF converter [3].

3.1 Digital Voltage-Loop Controller

The voltage-loop controller in Figure 1 is to be implemented digitally. From the system model and control design above, the output of this loop only changes once every T_L seconds. This low-speed loop naturally lends itself to a digital implementation that is not too expensive. To implement the current-loop controller on a microprocessor would require more expensive technology. Moreover, the digital implementation of the voltage-loop controller provides most of the flexibilty and robustness that one could hope to achieve in this system. The extra cost required to include the current loop controller on the microprocessor is not warranted presently.

The ROMless 16-bit 80C196KB was chosen as the microprocessor for the application. Much less expensive than 32-bit microprocessors and DSP chips, this embedded microcontroller provides ample processing power for the system at hand. A straightforward implementation of the PI controller discussed above actually should not require as sophisticated or expensive a microprocessor as the 80C196KB. However, the difficulties and complications in design due to quantization may cause a simpler microprocessor to end up requiring a considerably more complex implementation. Moreover, to achieve high levels of performance and to explore the various control schemes and features made possible by operating in the digital domain, a reasonably powerful microprocessor is required.

The EV80C196KB software evaluation tool for the 80C196KB microcontroller, allows control and monitoring of the processor through an Embedded Controller Monitor (ECM) that supports basic debug facilities in the target system [12]. The ECM is broken into two programs, one executing in the EV80C196KB and the other in an IBM PC compatible. They communicate via an asynchronous serial channel to allow the downloading of microcontroller programs, execution of these programs, and monitoring the various processor states as the programs execute. The 80C196KB has a 16-bit CPU and 232 bytes of on-chip RAM. It is a register-to-register machine, so no accumulator is needed, and most operations can be directly performed from or to any of the registers. In addition, there are many peripherals that are directly controlled through register operations. Figure 3 illustrates the features of the evaluation board and the microprocessor necessary in the application.

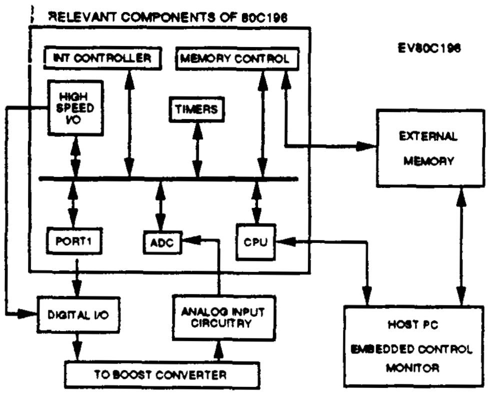


Figure 3: The Digital Development System.

3.1.1 Interrupt Handling

In a digital control system, some tasks such as sampling and output generation need to be synchronized to specific points in time. Others are event generated, such as shutdown or other precautionary measures taken when a certain condition is detected. Still other tasks, such as system operation services or computations carried out during time gaps free of other processing tasks, have no time restriction. For such a system, an interrupt oriented software structure is the best choice. In the UPF controller setup, the sampling and control output need to be carried out at specific points in time (8.3 ms apart). System shut-off, whether user generated or due to failure, is event-generated, and fine tuning of the system over several control cycles may be accomplished through estimation of circuit parameters to be computed over some time, whenever the processor is free. The event-generated tasks are carried out when an event occurs, and the time specific tasks are executed when timers on the microcontroller expire or reach a specified value. These events and timers generate interrupts, with the event-generated interrupts in this case having the higher priority. When an interrupt is generated, the CPU transfers control to the interrupt service routine (ISR) associated with the detected interrupt, as instructed by an interrupt vector. Tasks that have no timing constraints are not executed within an ISR and therefore have the lowest priority. On the 80C196KB, special function registers (SFR) and other devices handle interrupt generation, as shown in Figure 4.

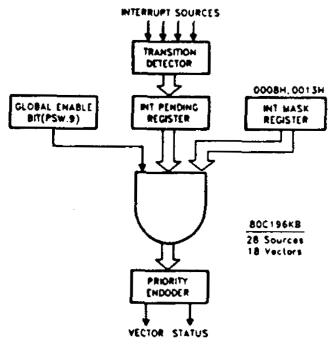


Figure 4: The 80C196 Interrupt Structure [13].

3.1.2 The A/D Converter

The 80C196KB has an analog to digital (A/D) converter system on board. The system has an 8-channel multiplexer to allow the sampling of 8 different analog signals at different times. The output of the multiplexer is passed through a sample and hold and then a 10-bit successive-approximation A/D converter. On the $12MHz$ version of the microprocessor, a conversion is completed in approximately $26\mu s$. As is the case with other peripherals, the A/D converter is SFR controlled. In this application, an A/D sample is acquired by writing a value to an SFR that specifies the channel to be sampled and initiates the acquistion. When the conversion is complete, an interrupt is generated. The results of the conversion (the value sampled and the number of the source channel) are found in other SFR's. [16] outlines in detail the A/D operation.

3.1.3 Timers

There are two timers on the 80C196KB. Timer1 is a 16-bit free-running timer incremented every 16 clock cycles, giving it a timing resolution of $1.33\mu s$. Timer2 is clocked externally through an input pin. Four "software timers" may be implemented using the high speed output unit (HSO) in conjunction with one of these timers. This is accomplished by programming the HSO to generate interrupts at preset times.

As each programmed time is reached by the specified timer, a Software Timer Interrupt is optionally generated. The timing of the digital controller implemented here is accomplished by writing to SFR's specifying Timer1 as the source of timing and enabling the Software Timer Interrupt. When the interrupt is generated, time-specific tasks are carried out. The source of the interrupt is identified from the contents of another register. This information is necessary when using more than one software timer (e.g. to implement a digital filter during testing, or for adaptive control which uses much faster sampling to estimate circuit parameters). The various SFR's used and the timing algorithm are described in detail in [16].

3.1.4 Output Ports

Ports 3 and 4 on the 80C196KB are output ports. On the EV80C196, however, these ports are used for memory access, and may not be used for output without off-board latches and decoding [12]. Port 1 and two pins on Port 2 are "quasi-bidirectional", and may be used as output pins [13]. In this implementation, the 8 lines on Port 1 are used for the output, as it turns out that 8 bits provide sufficient resolution. However, it should be noted that in an application not constrained by the evaluation board's use of resources, higher output resolution may be achieved, and control of more than one system on the same microcontroller is feasible. The output mechanism is discussed further in the next section.

3.2 D/A Conversion

The output of the digital voltage-loop controller needs to be fed to the current-loop controller as an analog signal. A natural implementation of the system computes the commanded current i_P in Figure 1 inside the microprocessor and then converts it to an analog signal to send to the current loop controller. This requires, in addition to computing $k[n]$, that the input waveform be sampled and reconstructed to provide the correct sinusoidal shape on the output. This extra sampling and processing is demanding in terms of microcontroller power and time. A better method utilizes a multiplying digital-to-analog converter (multiplying DAC). A multiplying DAC outputs a certain function of a digital and an analog input, usually a product. It is used to replace the multiplier in Figure 1 with inputs $k[n]$ and $v_{in}(t)$. Figure 5 illustrates the multiplying DAC in the implemented circuit configuration.

The latch at the input to the multiplying DAC is used to synchronize the signals to its pins. When used as outputs, the quasi-bidirectional pins will change state shortly after the system clock (CLKOUT) falls. The LS374 Flip-Flop shown clocks in the values of

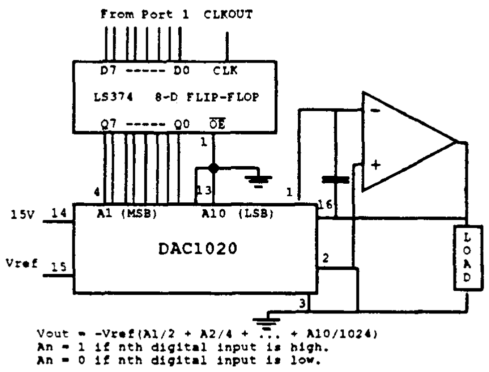


Figure 5: Output Port 1 and multiplying DAC configuration.

Port 1 pins to the multiplying DAC on the rising edge of CLKOUT.

3.3 Analog Current-Loop Controller

The current-loop controller is implemented using the Unitrode UC3854. (The UC3854 may be used in an analog setting to implement both the voltage and current control loops [15].) Three inputs to the chip, A, B, and C, are fed to a function block which outputs the function $\frac{AB}{C}$. This output is used as the current reference to which the input current is controlled through a current-mode controller on the chip, as shown in Figure 6. If one of the inputs is made to be the input voltage waveform, another a correction term based on the output voltage error, and the third (the one in the denominator) proportional to the square of the input voltage rms value, the system in Figure 1 is achieved [15]. For our digital implementation, inputs A and C are fixed in value and B is the output of the multiplying DAC from the previous section. This setup is particularly attractive for testing and development since it allows the flexible limiting of commanded power to safe values defined by A, C, and the upper limit on B.

3.4 A/D Resolution Enhancement

As it turns out, the 10-bit resolution on the A/D does not yield satisfactory results. Since only a small portion of the range of voltages (a section around steady state) needs to be resolved for successful PI control within the limits of the control command, it is possible to increase this resolution with the circuit in Figure 7. A range of voltages is mapped onto the 0-5

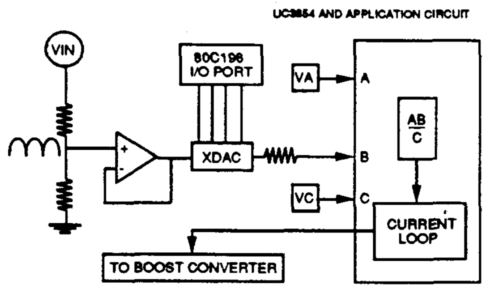
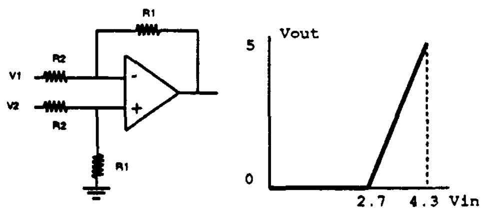


Figure 6: Implementation of the current loop.

volt range of the A/D converter, effectively increasing its resolution by a number of bits equal to $log_2(m)$, where m is the slope of the linear section of the curve.



Vout = (R2 / R1) x (V2 - V1)

Figure 7: Increasing Input Resolution.

3.5 The Complete System

Figure 8 shows a schematic of the overall system. The boost converter uses a load capacitor $C = 470\mu f$ and inductor $L = 1mH$. The capacitor is rated 450 volts. The load is composed of 3 light-bulbs in series, combining for a total of approximately 3200 ohms. The load may be effectively doubled by dropping an equal resistor in parallel with the light-bulbs. The inputs to the UC3854 in Figure 6 and the analog input to the multiplying DAC are chosen so that the maximum current command at 110 input voltage rms outputs 50 watts of power, which at the lower load level of 3200 ohms produces 400 volts across the load capacitor, sufficiently below its rated maximum.

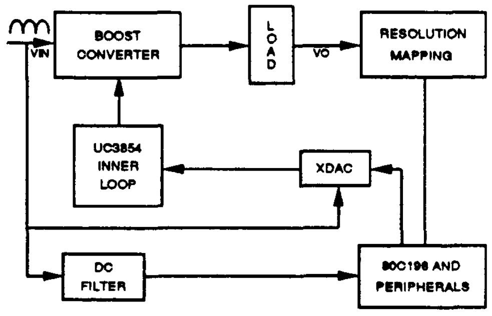


Figure 8: Overall structure of hardware system.

4 Implementation of Control Algorithm in Software

The software implementation of the controller uses fixed-point arithmetic to avoid the extra cost in terms of processing time and memory incurred by including a floating-point library. Care also needs to be taken to avoid extraneous gains from changing the characteristics of the closed loop system. These gains are introduced by the hardware interfaces or by scaling to enable the use of fixed-point representation. Timing constraints are also of extreme importance when working with sampled-data models. Timing as well as parameter quantization and scaling are at the core of the software design problem.

4.1 Timing

Timing for the controller is implemented using the HSO unit and Timer1 in the manner described above. To implement the discrete-time PI controller, the sampling and control output frequencies need to be $120Hz$. To start the system up, Software Timer 0 is set to expire after $8.3ms$. It is extremely important to maintain this frequency as accurately as possible, so the first instruction in the ISR for the software timer interrupt is to reload the timing register with the same value for the next cycle. In this manner, an $8.3ms$ period is insured every cycle, to within the resolution of the timer. The time-specific tasks of this controller, i.e. control output and sampling, are carried out next, before any further processing is executed. In doing so, uncertainty in the time periods between samples or outputs due to varying processing time from cycle to cycle (caused by conditionals in the code, for instance) is avoided. Once all of the necessary variables are sampled, the PI algorithm may be carried out, and the only time restriction on this task is that it be completed within one sampling cycle. It is therefore also included within the software timer ISR. Note that the 8.3ms time period is ample time for the controller implemented here. A simple PI computation requires less than 1ms to complete. The extra time is then left for estimation and adaptive (or more complex) control, or to be used by the operating system. Shutdown generates an interrupt of higher priority that transfers control to its own ISR when detected. Figure 9 illustrates the timing of the implementation.

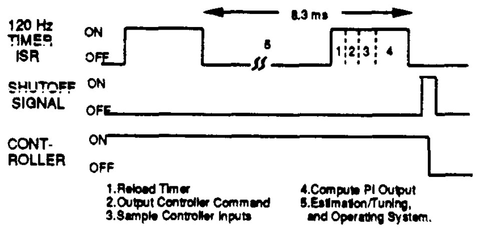

8.3 ms
120 Hz TIMER ISR
ON
OFF
5
1 2 3 4
SHUTOFF SIGNAL
CONT-ROLLER
1.Reload Timer
2.Output Controller Command
3.Sample Controller Inputs
4.Compute PI Output
5.Estimation/Tuning, and Operating System.

Figure 9: System timing.

4.2 Scaling and Digital Parameters

In order to achieve the correct implementation of the PI controller, the end result of our computation should be, as in Figure 2 above:

$$k[n] = \frac{C}{T_L V^2}(h_1(v_o^2 - V_o^2) - h_2\sigma[n]) \qquad (5)$$

Several problems arise in the digital implementation of this equation. Quantization of inputs to the controller occurs as a result of A/D conversion, and output quantization is a function of the number of bits that are feeding the input to the multiplying DAC. The equality above is therefore never met, and the desired zero-error steady state is not achieved. Instead, the output voltage and input current exhibit cyclic behavior around the "infinite-precision" or(analog) steady-state. This behavior is shown in the simulations in Figure 10. The system poles have been placed at .5 for fast transient decay. Although the quantization in voltage is not significant, the current (the envelope of which is proportional to k, since $i_P = kv_{in}$)is oscillating far from its average value. To improve the situation, the system is slowed down and the resolution is made higher, resulting in the working system discussed in Sections 5 and 6 [16].

The use of fixed-point arithmetic also dictates that many of the parameters of our system be scaled to be represented by an integer (as many of them are smaller than 1). This scaling, as well as other extraneous gains due to hardware, A/D conversion, or D/A conversion enter the loop in the transfer relation between the inputs to the controller and its outputs.

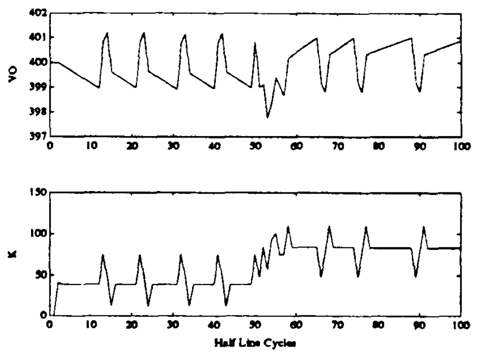


Figure 10: Closed-loop behavior with 8-bit input and output quantization.

The software must be designed to undo any such undesirable gains and ensure the relationship in (5) as the controller function [16].

4.3 Other Controller Features

Other features of the implementation serve to enhance its performance. Integrator windup is a problem that arises in PI controllers when the controller output is saturated. When saturation occurs, the integrator is effectively isolated from the closed loop system, and is therefore unstable [10]. Its value may increase excessively during that period, and it may take some time to "unwind" it from this state, which may result in undesirable behavior of the system (in this case voltage overshoot). As an anti-windup mechanism, the accumulator is disabled when the output of the digital controller is saturated. Simulations of the transient response of the system with and without the anti-windup implemented are shown later.

Other features on the controller include soft-startup and quantization error elimination in the steady state. The soft-startup is implemented by initializing the reference voltage at a low value and stepping it up to its final value. Simulations and results of this are demonstrated later. The quantization error elimination is necessary for the implementation of adaptive control, which requires the estimation of circuit parameters based on the ripple in the output voltage. Quantization error in this system is on the order of the ripple amplitude, and therefore makes it impossible to detect this information. The elimination is achieved through averaging of the output command, and is discussed in detail in [16].

Other features that are in the development stage or have been used in testing are digital filtering of the output voltage, and adaptive control. Both require the use of separate software timers and are discussed in [16] as well.

4.4 Overall Software Implementation

A simplified overall view of the software implementation is shown in Figure 11. The timing is accomplished as explained earlier, and the software is interrupt driven. The diagram shows the general operation of the code as well as an example of how extraneous gains produced in the hardware are offset in the software. Many of the details have been left out here, but are described in [16].

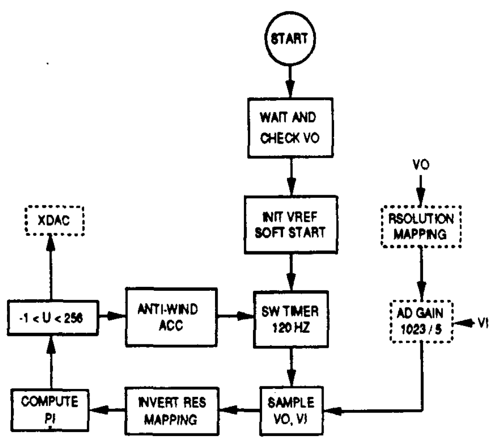


Figure 11: Software structure.

5 System Simulation and Experimental Results

Using the large-signal model and PI controller developed above, Matlab simulations [1] are constructed for the closed-loop system. With the general parameters of the development system and techniques presented earlier, simulations of the digital system are also defined. Titled simulations of different variations of the system are shown below.

Figure 12 shows the response of an analog system with the same poles as the system in Figure 10, but without quantization. Figure 13 shows a simulation of of the working digital system after slowing it down by moving the poles closer to the unit circle (around .91) and improving the input resolution using the resolution mapping circuit discussed earlier. The effects of quantization are negligible in this design, which was used for the final implementation.

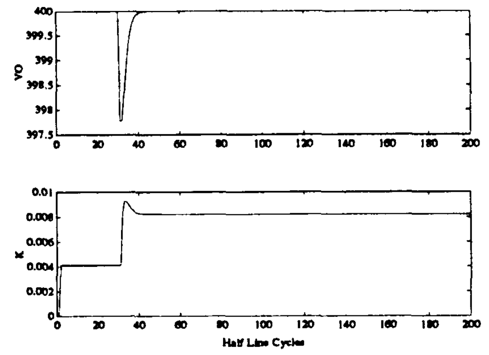


Figure 12: Analog system response to load transient with poles at .5.

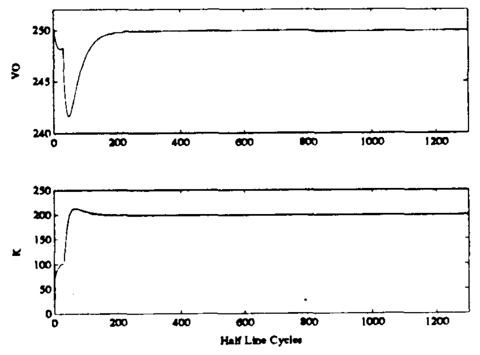


Figure 13: Simulation of implemented digital system.

The anti-windup and soft-startup features of the controller discussed earlier are also simulated. Figure 14 shows the startup procedure without either of these features. The output voltage overshoots to $420V$ as a result of the accumulator winding up during the large startup transient. The input current (proportional to k in the figure) saturates and does not show a smooth transition. Figure 15 illustrates the behavior of the circuit on startup with the anti-windup and soft-startup schemes simulated. The voltage no longer overshoots and the current is much smoother.

Experimental results on the startup, load transient response, and steady state operation of the physical system are shown below. They agree with the simulations and demonstrate satisfactory performance for the UPF. Quantization error can be seen in the current waveform and not so much in the voltage wave-

[1] Matlab is a trade mark of The Mathworks Inc.

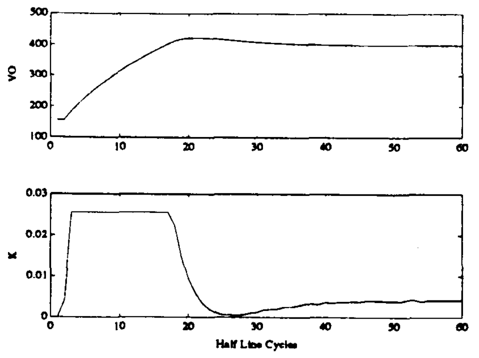


Figure 14: Startup without Anti-windup or Soft-start.

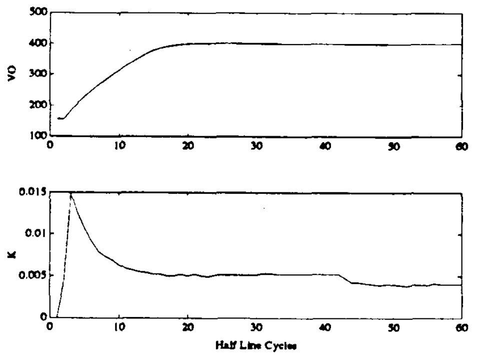


Figure 15: Startup with Anti-windup and Soft-start.

form, due to scale. The sinusoidal current yields a near unity power factor.

6 Conclusions and Future Work

The experimental results match the simulations well, verifying that the large-signal model is fairly accurate. Moreover, no load regulation and no steady state error are evident. A Unitrode application circuit built around the UC3854 to implement total control for the UPF exhibits load regulation. Although an analog implementation of a PI controller should in theory have zero steady-state error, in reality this is not the case. True PI control, with infinite gain at DC, is not possible due to parasitic resistances. The digital controller suffers from no such parasitics.

The basic closed-loop controller was successfully implemented, with a few additional features that were included with relative ease, and that illustrate the potential for rather complex control. The microprocessor used is relatively inexpensive and quite powerful, leaving room to implement more complex algorithms or to decrease the cost of the implementation. With the decreasing cost of semiconductors and their increasing functionality, the possibilities are even greater. Adaptive control is currently being developed for this system to achieve higher level performance and self-tuning (by adapting to changing values of the load capacitance). As power requirements become more stringent, this higher performance becomes more and more important. Digitally controlled power converters are better suited to meet these needs.

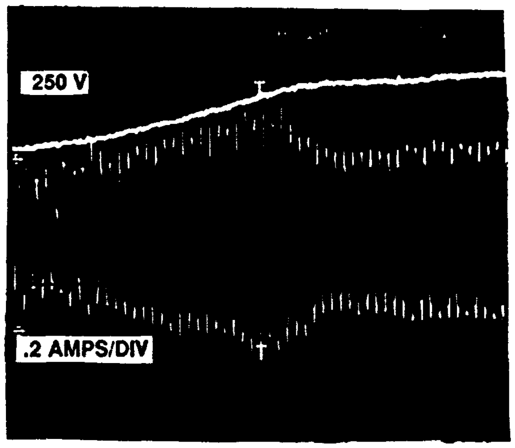


Figure 16: Startup in Implemented System

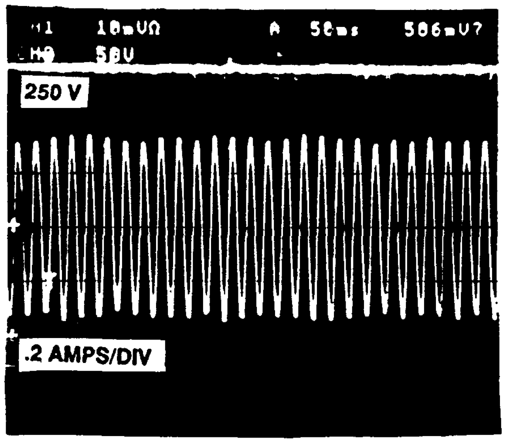


Figure 17: Steady-State in Imlpemented System

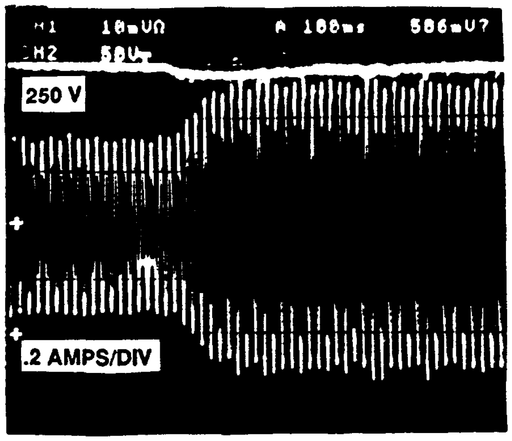


Figure 18: Response of Implemented System to Load Transient

References

[1] J.B. Williams, "Design of Feedback Loop in Unity Power Factor AC to DC Converter", PESC, 1989, pp.959-967.

[2] B. Sharifipour, P. Cacciola, and J. Maddox, "Designing a 1200 Watt Multiple Output Modular Power System with High Power Utilization for the Workstation Environment", APEC, 1989, pp. 439 - 444.

[3] J. G. Kassakian, M. F. Schlect, G. C. Verghese, "*Principles of Power Electronics*", Addison-Wesley, 1991, pp. 395-399.

[4] K. Mahabir, G. Verghese, J. Thottuvelil, and A. Heyman, "Linear Averaged And Sampled Data Models for Large Signal Control of High Power Factor AC-DC Converters", PESC, 1990, pp. 291 - 299.

[5] C.P. Henze and N. Mohan, "A Digitally Controlled AC to DC Power Conditioner that Draws Sinusoidal Input Current", PESC, 1986, pp531-540.

[6] K. Ogata, "*Discrete-Time Control Systems*", Prentice Hall Inc., 1987, pp. 1 - 36.

[7] B. Orlik, H. Weh, "Microprocessor-Controlled Three-Phase Motors With High Resolution Digital Pulse Width Modulator for High Pulse Frequencies", EPE, 1985, pp. 3.39 - 3.44.

[8] C. Bergman, P. Goureau, "Direct Digital Control of a Self-Controlled Synchronous Motor with Permanent Magnet", EPE, 1985, pp. 3.269 - 3.273.

[9] J. Siebert, "Freely Programmable Digital Open-loop/Closed-loop Control System for Converters and Converter Drives", EPE, 1985, pp.5.7 - 5.10.

[10] K. Astrom, B. Wittenmark, "*Computer-Controller Systems, Theory and Design*", Prentice Hall Inc., 1990, pp. 224 - 226.

[11] A. Stankovic, G. Verghese, X. Liu, and J. Thottuvelil, "Fast Controllers for High-Power-Factor AC-DC Converters", European Power Electronics, 1990.

[12] 'EV80C196KB Microcontroller Evaluation Board User's Manual, Release 001', INTEL Corporation, February 20, 1989.

[13] '80C196KB User's Guide', INTEL Corporation, October 1990.

[14] Y. Guijun, L. Norum, "Low Cost Digital Controller for PWM Converter", International Federation for Automatic Control Low Cost Automation, 1990.

[15] 'Power Factor Correction with the UC3854 - Application Note', UNITRODE Integrated Circuits,

[16] Mitwalli, Ahmed "A Digital Controller for a Unity Power Factor Power Converter", Master Thesis, MIT, Dec. 1992.