Architectures and Circuits for Low-Voltage Energy Conversion and Applications in Renewable Energy and Power Management

by

Robert C. N. Pilawa-Podgurski

B.S. Physics, Massachusetts Institute of Technology (2005)
B.S. EECS, Massachusetts Institute of Technology (2005)
M.Eng. EECS, Massachusetts Institute of Technology (2007)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2012

Author .....
Department of Electrical Engineering and Computer Science
December 22, 2011

Certified by .....
David J. Perreault
Professor, Department of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by .....
Leslie A. Kolodziejski
Professor, Chair of the Committee on Graduate Students

Architectures and Circuits for Low-Voltage Energy Conversion and Applications in Renewable Energy and Power Management

by

Robert C. N. Pilawa-Podgurski

Submitted to the Department of Electrical Engineering and Computer Science
on December 22, 2011, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

In this thesis we seek to develop smaller, less expensive, and more efficient power electronics. We also investigate emerging applications where the proper implementation of these new types of power converters can have a significant impact on the overall system performance.

We have developed a new two-stage dc-dc converter architecture suitable for low-voltage CMOS power delivery. The architecture, which combines the benefits of switched-capacitor and inductor-based converters, achieves both large voltage step-down and high switching frequency, while maintaining good efficiency. We explore the benefits of a new soft-charging technique that drastically reduces the major loss mechanism in switched-capacitor converters, and we show experimental results from a 5-to-1 V, 0.8 W integrated dc-dc converter developed in 180 nm CMOS technology.

The use of power electronics to increase system performance in a portable thermophotovoltaic power generator is also investigated in this thesis. We show that mechanical non-idealities in a MEMS fabricated energy conversion device can be mitigated with the help of low-voltage distributed maximum power point tracking (MPPT) dc-dc converters. As part of this work, we explore low power control and sensing architectures, and present experimental results of a 300 mW integrated MPPT developed in 0.35 um CMOS with all power, sensing and control circuitry on chip.

The final piece of this thesis investigates the implementation of distributed power electronics in solar photovoltaic applications. We explore the benefits of small, intelligent power converters integrated directly into the solar panel junction box to enhance overall energy capture in real-world scenarios. To this end, we developed a low-cost, high efficiency (>98%) power converter that enables intelligent control and energy conversion at the sub-panel level. Experimental field measurements show that the solution can provide up to a 35% increase in panel output power during partial shading conditions compared to current state-of-the-art solutions.

Thesis Supervisor: David J. Perreault
Title: Professor, Department of Electrical Engineering and Computer Science

Acknowledgments

I want to thank Professor David Perreault, my thesis advisor, with whom I have worked for the last six years of graduate school. He has taught me most of what I know about power electronics, and have shown me, by example, what it means to be a great engineer, researcher, and educator. Many experiments and projects that I pursued during my time at MIT never made it into this thesis, but they enriched my experience, taught me important concepts, and gave me the breadth to understand where my research fits into the bigger picture. I am very grateful to Dave for letting me pursue many of these projects while simultaneously working on the projects contained in this thesis.

My thesis committee members provided valuable advice and guidance throughout the course of this work: Professor Charles Sullivan was the one I relied on for any magnetics-related problems, and he was also instrumental in shaping the direction and scope of the solar PV project. Professor Anantha Chandrakasan not only taught me most of what I know about low-power digital circuits, he also provided helpful advice about what problems to address in my research. Dr. Ivan Celanovic was a fantastic manager of the TPV project. In addition to his deep technical skills, he has a unique ability to bring together some fantastic people and have them work towards a common goal, which is not always easy at a place like MIT.

I would like to thank the rest of the Perreault research group at MIT, both past and present. It has been a joy to be exposed to all the different projects that they have been working on, and to learn from, and contribute to, all of their research.

My many friends of the round table at LEES have made my time in graduate school a very enjoyable experience: Olivia Leitermann, Anthony Sagneri, Jackie Hu, Al Avestruz, Shahriar Khushrushahi, Warit Wichakool, John Cooley, and many more.

In addition to being great friends, Brandon Pierquet and Bill Richoux have helped in many aspects of this work. I could count on Brandon for fun discussions about all things embedded and hands-on, while Bill was my source of information whenever the math got in the way of my understanding. He has a great gift for concisely explaining difficult (to me) mathematical concepts, which I often took advantage of. The lively lunch conversations with Brandon and Bill around the round table is one of the things I will miss the most about my time at LEES.

Walker Chan, Nathan Pallo, and Wei Li provided key contributions to the TPV project. Yogesh Ramadass patiently guided me in my first steps as an IC designer, and I appre-

ciate his willingess to teach me this new (to me) area. Juan Santiago contributed to the experimental solar PV setup, for which I am grateful.

I would also like to thank Dave Otten, who knows more than anyone else in our laboratory how to make things work, and how to make them work well. His practical experience and knowledge is truly a treasure, and I have enjoyed our many conversations in the early morning hours before the rest of the laboratory comes in.

I would like to thank my family for their love and support. My love for engineering comes from my father Peter, who is always building or repairing something, somewhere. My mother Maria made sure that my time at MIT was fun outside of the classroom as well, by constantly encouraging me to study less! My sisters Petra and Jana always remind me that engineering is but a small part of what makes this world work, something that is easy to forget when you are immersed in the culture that is MIT. My brother Daniel reminds me that we are all given different gifts in life, and his ability to overcome his challenges makes whatever I do pale in comparison.

Many sponsors have made this work possible. I acknowledge the National Science Foundation, the Interconnect Focus Center, and the MIT Institute for Soldier Nanotechnologies for their support. I would also like to take this opportunity to thank the foundation "Erik och Göran Ennerfelts Fond för Svensk Ungdoms Internationella Studier", whose generous financial assistance helped realize my dream of coming to MIT as an undergraduate.

Last but certainly not least, I would like to express my gratitude to my wife Brooke, to whom I dedicate this thesis. Her love and support throughout this (long) journey is something that I will always be grateful for, and I cherish our time together. Throughout our time in Cambridge, we also received our newest family addition, Nikolai, whose primary contribution to this thesis is delaying its completion. I could not ask for a better reason to delay it.

To Brooke

Contents

List of Figures

List of Tables

LIST OF TABLES

Chapter 1

Introduction

1.1 Introduction

WITH the continued downward scaling of semiconductor devices described by Moore's law, information processing circuits have achieved substantial reductions in size, and improvements in performance. The field of power electronics has not benefited to the same extent from Moore's law, owing to the fundamental differences between processing energy and processing information. As a result, modern electronic products are very often limited by the power electronics when it comes to size, weight, and cost. This thesis seeks to address this issue by exploring new architectures that enable drastically reduced size of the power electronics, while maintaining high efficiency and power density.

One application area that is investigated is that of CMOS power delivery. Specifically, the work presented here makes use of device characteristics in standard CMOS processes to implement architectures that enable large voltage step-down at high switching frequency. In addition, fundamental limits of switched-capacitor power converters are investigated, and methods to improve their efficiency are explored.

In addition to fundamental advancements of power electronics architectures themselves, this thesis seeks to identify and analyze applications where small, efficient, low-voltage power converters can improve system-level performance in energy conversion devices. Two applications that are explored in detail are thermophotovoltaic energy conversion and solar photovoltaics. Challenges associated with control losses and conversion efficiencies at low

power and voltage are investigated, and several experimental implementations are demonstrated that enable an increase in system-level efficiency.

1.2 Organization of Thesis

The challenge of low-voltage CMOS power delivery is addressed in Chapters 2, 3, and 4. A new power converter architecture (merged two-stage architecture) that leverages the properties of CMOS transistors and switched-capacitor power converters is introduced in Chapter 2, together with mathematical analysis and simulated performance comparison that highlights the strengths of the proposed architecture compared to state-of-the-art solutions. The chapter introduces the concept of *soft charging* operation, which recycles energy normally dissipated in switched-capacitor power converters, thereby increasing the overall converter efficiency, while also reducing the converter size.

Chapter 3 contains a description of the first experimental prototype that implements the soft charging techniques, implemented using discrete components. A comparison to regular operation is provided, together with experimental waveforms.

In Chapter 4 a fully integrated version of the merged two-stage architecture is presented, developed in a 180 nm CMOS process. All requisite components for soft charging operation are described, together with a startup scheme that solves practical implementation issues normally associated with switched-capacitor converters. A feed-forward control implementation that is a critical component of soft charging operation is presented, along with experimental results and a comparison to conventional solutions.

Chapter 5 presents the background information of a large interdisciplinary thermophotovoltaic generator project and the associated power electronics challenges. We introduce a distributed maximum power point tracking (MPPT) architecture that solves many of the mechanical challenges of the system by using intelligent power electronics. The distributed solution offers increased energy capture compared to conventional techniques, and can be

adapted to other energy sources in addition to the TPV system considered in this work.

A discrete prototype of the distributed MPPT system is presented in Chapter 6, along with techniques to achieve low power sensing and control and high power efficiency conversion. A description of suitable MPPT algorithms and their practical implementation are provided, as well as a discussion about tracking speed and precision trade-offs for this particular implementation. Furthermore, experimental results of the circuit operating with a TPV power generator are presented, and represents the first (to our knowledge) full system-level demonstration of a micro-TPV power generator.

Chapter 7 contains a detailed description of a custom CMOS implementation of the control and power stage of a distributed MPPT for the TPV power generator. A custom low-power loss-less current sensing ADC is presented, along with a fully digital implemenation of an MPPT algorithm in 0.35 μm CMOS. A soft-switching power stage and a detailed size/efficiency comparison are also presented in this chapter, as well as experimental results and characterization of the system.

Finally, Chapter 8 contains the last contribution of this thesis, which is a sub-module integrated MPPT converter for solar photovoltaic applications. An analysis of state-of-the-art PV power electronics solutions is provided, along with a survey of appropriate power converter topologies for distributed MPPT for solar PV applicatons. We address global and local control algorithms, and present an experimental prototype used to evaluate the control techniques. Finally, static and dynamic field testing is presented, which illustrate the benefits achieved by the distributed sub-module power electronics in solar PV applications.

Chapter 2

Merged Two-Stage Converter

2.1 Motivation

Shown in Figure 2.1 is a conventional method for delivering power from an input voltage to a low-voltage load such as CMOS circuitry. In this approach, to a CMOS load circuit, an off-chip converter takes the relatively high (e.g., 5-12 V) input voltage and performs a step-down to the low-voltage (e.g., 1-2 V) load. Because the power transistors in the off-chip converter must be rated for the full input voltage, they will have relatively high parasitic capacitance (for a reasonable on-state resistance) which will limit the achievable switching frequency. A low switching frequency will in turn necessitate large passive components, which contribute to an overall large power converter volume and cost. The low achievable switching frequency of the power transistors also lead to a control bandwidth that is low, and hence relatively large output capacitance is needed to handle transient events such as the CMOS load going from sleep mode to full active mode. The result of this power delivery architecture is that in today's electronics, the power converter can make up the majority of the weight and volume of the system, and is thus often the bottleneck to achieve miniaturization and integration.

For magnetics-based designs operating at low, narrow-range input voltages (e.g., 2 V in and 1 V out), it is possible to achieve extremely high switching frequencies (up to hundreds of MHz [1–3]), along with correspondingly high control bandwidths and small passive components (e.g., inductors and capacitors). It also becomes possible to integrate portions of the converter with a microprocessor load in some cases. These opportunities arise from

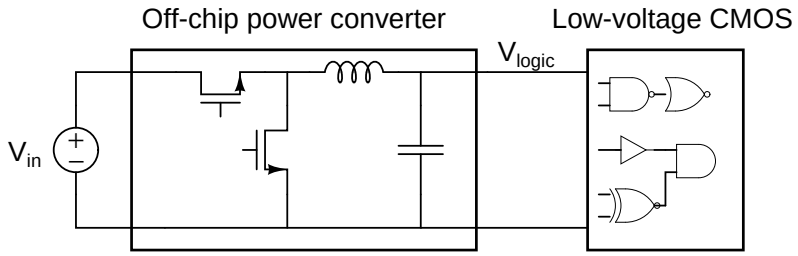


Figure 2.1: *Schematic drawing of conventional CMOS power delivery. The transistors of the off-chip power converter must block the full input voltage, resulting in the use of slow, high-voltage transistors that must operate at a low switching frequency.*

the ability to use fast, low-voltage, process-compatible transistors in the power converter. However, at higher input voltages and wider input voltage ranges, much lower switching frequencies (on the order of a few MHz and below) are the norm, due to the need to use slow extended-voltage transistors (on die) or discrete high-voltage transistors. This results in much lower control bandwidth, and large, bulky passive components (especially magnetics) which are not suitable for integration or co-packaging with the devices.

Another conversion approach that has received attention for low-voltage electronics is the use of switched-capacitor (SC) based dc-dc converters [4–10]. This family of converters is well-suited for integration and/or co-packaging of passive components with semiconductor devices, because they do not require any magnetic devices (inductors or transformers). A SC circuit consists of a network of switches and capacitors, where the switches are turned on and off periodically to cycle the network through different topological states. Depending on the topology of the network and the number of switches and capacitors, efficient step-up or step-down power conversion can be achieved at different conversion ratios.

There are, however, certain limitations of the SC dc-dc converters that have prohibited their widespread use. Chief among these is the relatively poor output voltage regulation in the presence of varying input voltage. The efficiency of SC converters drops quickly as the conversion ratio moves away from the ideal (rational) ratio of a given topology and operating mode. In fact, in many topologies the output voltage can only be regulated for a narrow range of input voltages while maintaining an acceptable conversion efficiency [6, 7, 11]. SC

dc-dc converters have been described in the literature [4–7, 9] for various conversion ratios and applications, and the technology has been commercialized. These types of converters have found widespread use in low-power battery-operated applications, thanks to their small physical size and excellent light-load operation.

Another disadvantage of early SC converters is discontinuous input current which has been addressed in [12,13]. These new techniques, however, still suffer from the same degradation of efficiency with improved regulation as previous designs.

One means to partially address these limitations is to cascade a SC converter having a fixed step-down ratio with a low-frequency switching power converter having a wide input voltage range [14] to provide efficient regulation of the output. Other techniques [15, 16] integrate a SC circuit within a buck or boost converter to achieve large conversion ratios. However, the regulation bandwidth of these techniques is still limited by the slow switching of the SC stage.

Another approach that has been employed is to use a SC topology that can provide efficient conversion for multiple specific conversion ratios (under different operating modes) and select the operating mode that gives the output voltage that is closest to the desired voltage for any given input voltage [7, 17]. However, none of these approaches are entirely satisfactory in achieving the desired levels of performance and integration.

2.2 Two-stage architecture

Fig. 2.2 shows a block diagram of a two-stage converter that combines a high efficiency switched-capacitor transformation stage with a high-frequency, low-voltage regulation stage. This strategy makes use of on-die device characteristics available in CMOS processes. As examined in [18], low-voltage submicron CMOS processes inherently provide far higher achievable switching frequencies than higher-voltage processes. In a given process, one often has access to both slow, moderate-blocking-voltage devices and fast, low-voltage devices.

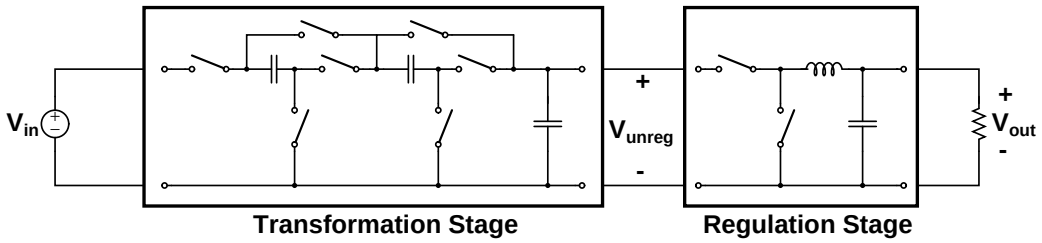


Figure 2.2: *Block diagram illustrating a two-stage converter. The transformation stage can be constructed using slow, high voltage devices and operated at a slow switching frequency, while the regulation stage can be constructed with fast, low-voltage devices and operated at a high switching frequency.*

The converter architecture of Fig. 2.2 is well-suited to the available devices in such a process: The SC transformation stage can achieve a large voltage step-down, and can be designed for very high power density and efficiency using slow, moderate-voltage devices at relatively low switching frequency. The unregulated voltage, V_{unreg} is low so that the regulating stage can utilize fast, low-voltage devices operating at a high switching frequency to provide high-bandwidth regulation and a small additional voltage step-down. Since the regulation stage operates at a high frequency, the size of its passive components can be made small. By separating the transformation and regulation stage in this manner, the benefits typically associated with SC converters (i.e. high efficiency, high power density) can be preserved, while the main drawback (poor regulation) is done away with by the use of a separate magnetic regulation stage. Furthermore, since the regulation stage only sees a very low voltage, it can operate at a much higher frequency and control bandwidth than a single, conventional switching power converter that needs to provide a large step-down in voltage.

It should also be noted that if a switched-capacitor stage capable of multiple conversion ratios (e.g. 1:1, 2:1, and 3:1) is utilized, one can dynamically change the conversion ratio of the transformation stage (e.g. as a function of input voltage). This enables such a system to function over a wide range of input voltages, while preserving a low and relatively narrow voltage range on the regulation stage.

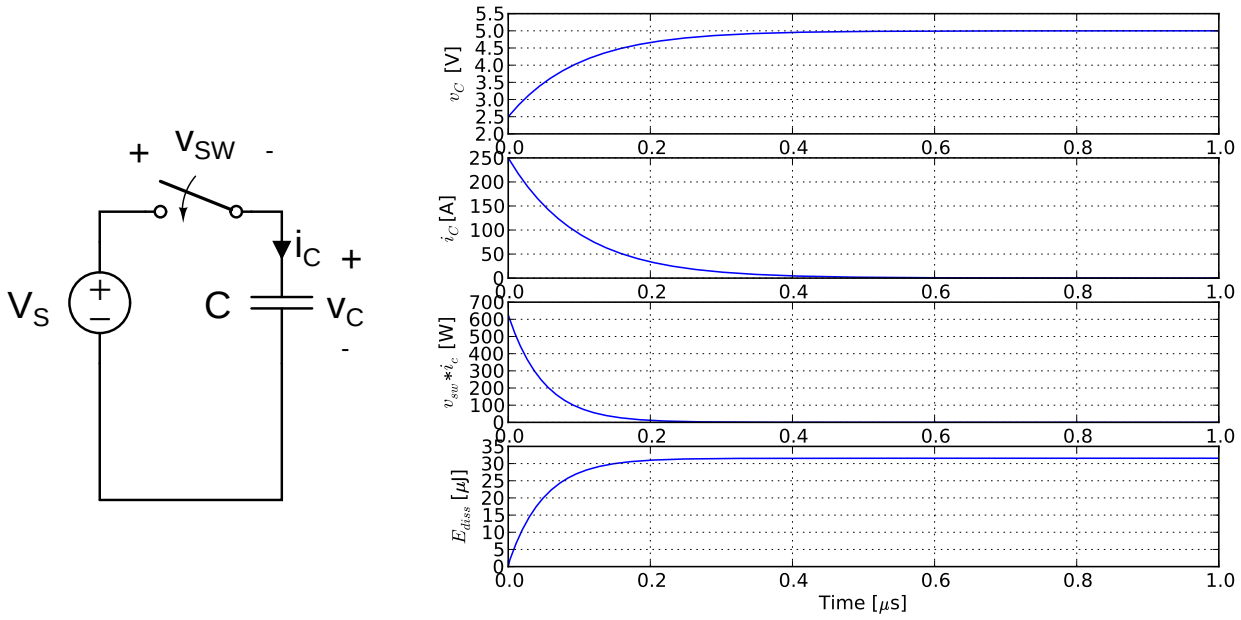
2.3 Merged Two-stage Converter

In addition to the benefits listed above, yet another advantage can be realized by a suitable implementation of the two-stage approach. To understand this concept, it is illustrative to first consider the fundamental trade-offs in efficiency, capacitance, and frequency in a conventional switched-capacitor converter.

The circuit shown in Fig. 2.3 is a simple example which illustrates the loss mechanism for charging of the capacitors in the SC stage. In this example, a single capacitor represents the "stack" of capacitors typically seen in a SC converter, while the switch likewise represents the total switch resistance of all transistors in the charging path.

Fig. 2.3 shows an example of the charging process of a capacitor C as is done in a conventional SC circuit. The switch has some small value of on-state resistance. The capacitor has an initial charge of $V_C = V_S - \Delta V$, and the switch is closed at $t = 0$. After $t = 0^+$, the difference between voltage V_S and the capacitor voltage at each instance in time appears across the switch. If charging is allowed to continue for a sufficient period of time (the so-called "slow switching limit"), the voltage across the capacitor will charge up to V_S, and the voltage across the switch will become 0 V. The instantaneous voltage across the switch and the current through it results in a power loss during the charging phase of the capacitor.

Figure 2.3b shows simulated waveforms, with an initial capacitor voltage of 2.5 V, a switch resistance of 10 mΩ, and a capacitor value of 10 μF. It can be seen from the plot that the capacitor voltage charges up to a final voltage of 5 V with the typical exponential characteristics of a first-order system with a time constant RC. The switch current i_C falls of with the same time constant, as the voltage across it decreases when the capacitor charges up. The high initial value is what is often-time a problem in SC converters, as small switch resistances can give rise to very large impulse-type currents. Also shown is the power dissipated in the switch, which is the product of v_{SW} and i_C.



(a) **(b)**

Figure 2.3: *Charging of capacitor from a constant voltage source. Schematic shown in (a), along with simulated waveforms ((b)0 for an initial capacitor voltage of 2.5 V, a switch resistance of 10 mΩ, and a capacitor value of 10 μF.*

It is important to note that this fixed charge-up loss associated with this power dissipation cannot be reduced by employing switches with lower on-state resistance. A lower parasitic R will only result in larger peak currents of shorter duration, but the total power loss in each charge cycle will remain the same. This can be clearly seen from Eq. 2.2, which shows that the energy lost in charging (E_{sw}) in the slow-switching limit is independent of the switch resistance.

$$E_{sw} = \int_{t=0}^{t=\infty} I_{sw} \times V_{sw}\, dt = \int_{t=0}^{t=\infty} C\frac{dV_C}{dt} \times (V_S - V_C)\, dt \tag{2.1}$$

$$E_{sw} = \int_{V_S-\Delta V}^{V_S} C \times (V_S - V_C)\, dV_C = \frac{1}{2}C(\Delta V)^2 \tag{2.2}$$

Thus, for a conventional SC circuit, a fixed amount of charge-up energy loss proportional to $\frac{1}{2}C(\Delta V)^2$ will result at each switch interval, where ΔV corresponds to the difference between the initial ($t = 0$) voltage of the capacitor and the final voltage. As seen from the plot showing total energy dissipated (E_{diss} of Fig. 2.3b), the total energy dissipated is 31.25 μJ, which is $\frac{1}{2}C(\Delta V)^2$ for the capacitor and voltage values of the simulation[1]. Consequently, conventional SC converters require either large capacitors or high switching frequencies to minimize ΔV (and the associated power loss), and achieve high efficiency and power levels [5, 19].

This coupling between switching frequency and capacitor size for conventional SC converters has been well explained in [20], where the concept of slow switching and fast switching limits were introduced to illustrate the concept. In the fast switching limit (FSL), the current flow between capacitor is constant, and the switching frequency is sufficiently high for a given capacitor size such that each capacitor does not reach equilibrium before the switch configuration is altered again. The switch on-state resistance and other circuit resistances are high enough such that the capacitor only incrementally charges and discharges, but

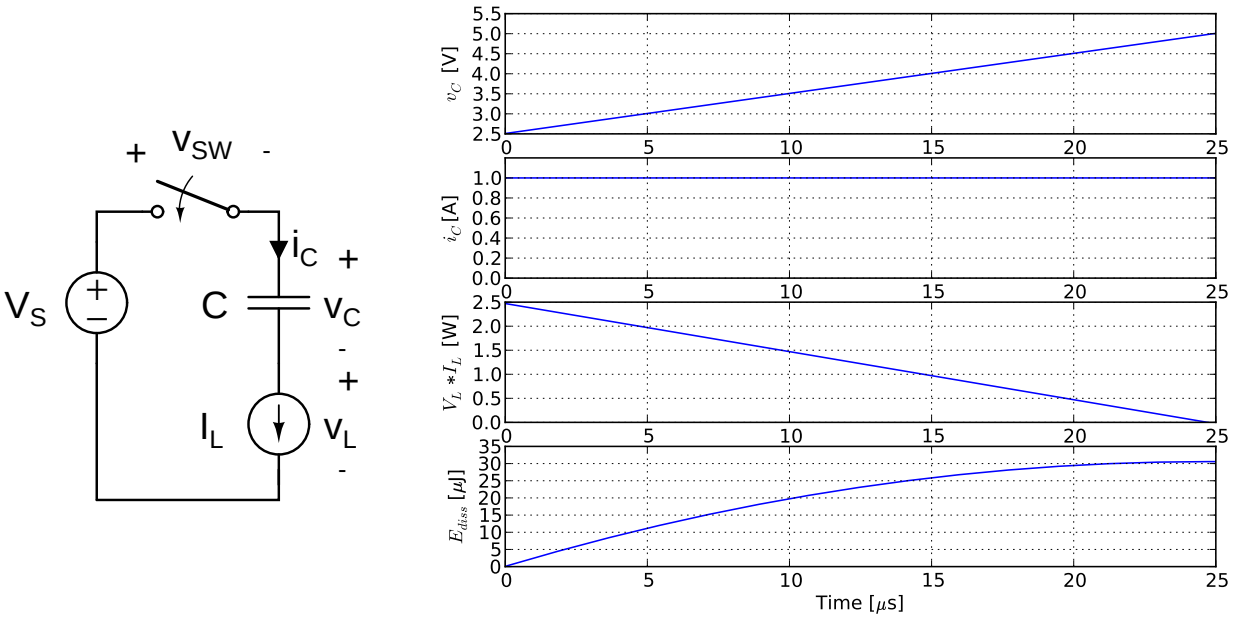
[1]Note that these and the following simulations will show a small discrepancy between the theoretical and simulated values, owing to the numerical precision of the SPICE tool.

never reaches steady-state for a given switch configuration. In FSL operation, the capacitor charging and discharging losses come entirely from losses in resistive elements. There is thus a lower bound on loss for a given circuit resistance, which cannot be made smaller by increasing the size of the capacitors or the switching frequency. It should be noted that in order to operate at the FLS, one must typically employ large capacitors (i.e. large converter volume) or high switching frequency (i.e. high switching loss).

Operation at the slow switching limit (SSL) corresponds to to the scenario described above, where $\frac{1}{2}C(\Delta V)^2$ of energy is lost in each switching cycle. While the authors of [20] uses charge multipliers rather than the detailed time-domain equations above, the resulting loss is the same.

However, as is shown below, this tight dependence of efficiency on capacitance and switching frequency in the SC converter can be mitigated through the appropriate merging between the SC (transformation) stage and the regulating stage in our two-stage converter. In this thesis we propose a method to achieve small loss levels associated with the FSL, while being able to operate at substantially lower switching frequency than what is established in [20] for conventional SC converters. Lower switching frequency has the benefit of reduced switching loss, and hence higher conversion efficiency.

As a means to better understand the proposed solution to the lossy charging of switched-capacitor converters, we will first consider the case shown in Fig. 2.4a. Here we have placed a current source in series with the charging path, which maintains the charging current at a steady value (I_L). As can be seen in Fig. 2.4b, the peak charging current has been greatly reduced, and is now constant throughout the charging process. Furthermore the plot of power dissipation ($V_L * I_L$) in the current load shows a linearly decreasing power loss. As can be seen from Eq. 2.5, almost all the energy loss is now in the current source, if the switch resistance is made low.



(a) **(b)**

Figure 2.4: *Current-load assisted charging of capacitor from a constant voltage source. Schematic shown in (a), along with simulated waveforms ((b)) for an initial capacitor voltage of 2.5 V, a switch resistance of 10 mΩ, and a capacitor value of 10 μF . The current load is constant at 1 A.*

$$E_{current-source} = \int_{t=0}^{t=\infty} I_L \times V_L \, dt = \int_{t=0}^{t=\infty} I_L \times (V_S - V_{sw} - V_C) \, dt \tag{2.3}$$

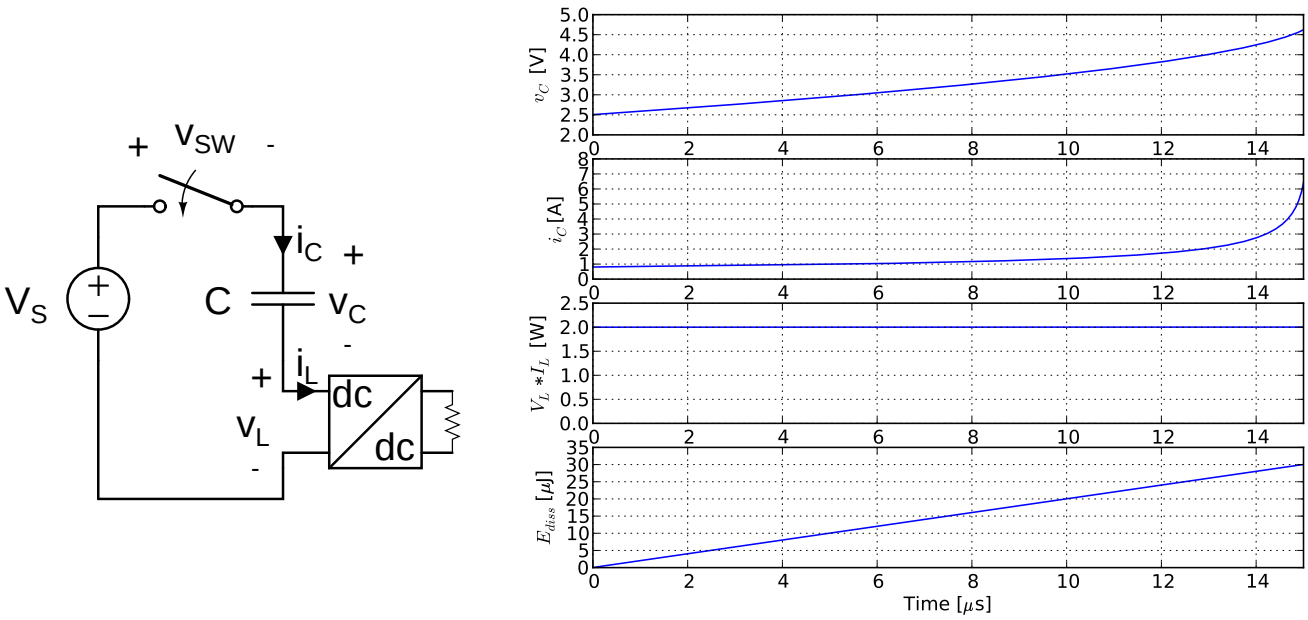
$$E_{current-source} = \int_{t=0}^{t=\infty} I_L \times (V_S - I_L R_{sw} - V_C) \, dt \simeq \int_{t=0}^{t=\infty} C \frac{dV_C}{dt} \times (V_S - V_C) \, dt \tag{2.4}$$

$$E_{current-source} \simeq \frac{1}{2} C (\Delta V)^2, \tag{2.5}$$

,

where the approximation has been made that the $I_L R_{sw}$ product is small, which can be ensured by the proper choice of switch resistance and/or current value. In the simulation example used, it can be seen from the bottom plot of Fig. 2.4b that the total dissipated power in the current load (E_{diss}) is very close to $\frac{1}{2} C (\Delta V)^2$, for the case when a current load value of 1 A is used together with the simulation parameters stated previously (and hence, the approximation of (2.4) is valid). By introducing a current source in the charging path we have moved the charging loss from the switch to the current source, but its value is still the same (again, assuming low switch resistance).

Fig. 2.5a illustrates the proposed method to improve the efficiency of the SC circuit. Here we replace the constant current source with a dc-dc converter. In this circuit, the dc-dc converter is operating at a much higher switching frequency than the SC stage, such that is appears to the SC stage as a constant power sink when the switch is closed. The system is designed such that the majority of the difference between source voltage V_S and the capacitor stack voltage V_C appears across the input of the dc-dc converter when the capacitor is charging. Instead of being dissipated as heat in the switch resistance, the energy associated with charging the capacitor stack is delivered to the output of the dc-dc converter.



(a) **(b)**

Figure 2.5: *Soft charging of capacitor with the help of dc-dc converter. Schematic shown in (a), along with simulated waveforms ((b)) for an initial capacitor voltage of 2.5 V, a switch resistance of 10 mΩ, and a capacitor value of 10 μF. The dc-dc converter is acting like a constant 2 W power sink.*

By using an auxiliary dc-dc converter to absorb the effective ΔV of the capacitor, the impulse-like charging current spikes typically associated with conventional SC converters are replaced with a smooth charging current, whose value is determined by the output power of the dc-dc converter and the value of ΔV. We term this technique *soft charging*[2]. With this technique, the $\frac{1}{2}C(\Delta V)^2$ energy that is typically lost at each switching interval in a conventional converter is instead captured and provided to the output.

In the plot of Fig. 2.5b, we can see that the input power to the dc-dc converter ($v_L * i_L$) is constant, and that the total energy delivered to the load (E_{diss}) is approximately $\frac{1}{2}C(\Delta V)^2$, in accordance with the mathematical analysis of (2.7).

$$E_{dc-dc} = \int_{t=0}^{t=\infty} I_L \times V_L \, dt = \int_{t=0}^{t=\infty} I_L \times (V_S - I_L R_{sw} - V_C) \, dt \tag{2.6}$$

$$E_{dc-dc} \simeq \int_{t=0}^{t=\infty} C\frac{dV_C}{dt} \times (V_S - V_C) \, dt \simeq \frac{1}{2}C(\Delta V)^2, \tag{2.7}$$

One important thing to note from Fig. 2.5b is the behavior of the circuit as the capacitor voltage (v_C) becomes close to the source voltage (V_S). In this case, the input voltage to the dc-dc converter (v_L) approaches zero, which means that the current increases rapidly (since the dc-dc converter maintains a constant input power). This behavior is undesirable, as it would lead to very high peak currents (and associated power loss). In addition, the approximation of Eq. 2.7 would no longer be correct, as the voltage drop across the switch resistance would no longer be negligible. For proper operation, the soft charging technique therefore must not allow the charging process to reach steady-state, but must instead operate the SC converter such that a sufficiently large voltage always appears across the input of the dc-dc converter.

[2]Likewise, we will use the term *hard charging* to denote a SC converter operating in a conventional manner.

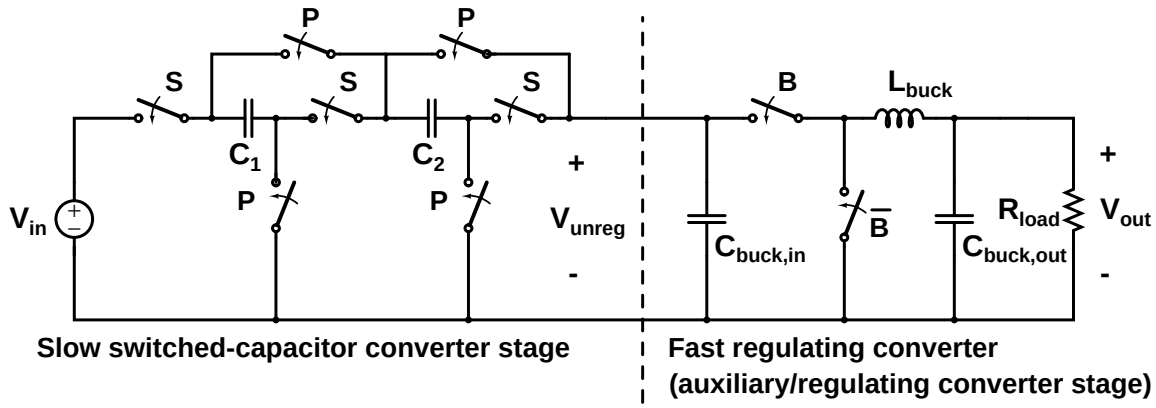


Figure 2.6: *Example of the switched-capacitor transformation stage circuit coupled with a fast regulating stage which provides soft charging of the switched-capacitor stage.*

This mode of operation is markedly different from traditional SC converters, and as we shall see it requires entirely new control techniques. The example above illustrates the key principle of the soft charging technique, but there are a number of practical issues to manage when this technique is employed in a full converter circuit. We now highlight some of these challenges and propose solutions that enable the use of this powerful concept.

Fig. 2.6 illustrates how soft charging can be implemented in the two-stage converter. The fast regulating converter (in this case a synchronous buck converter) serves as both the auxiliary dc-dc converter and the regulating converter stage for the system. It operates at a switching frequency much higher than that of the switched capacitor stage. As the capacitor $C_{buck,in}$ serves only as a filter and bypass for the fast regulating converter, its numerical value can be much smaller than the capacitors C_1 and C_2 of the SC stage.

When the SC stage is configured for charging of C_1 and C_2 (switches S closed), the difference between V_{in} and the sum of the voltages across capacitors C_1 and C_2 appears across the input terminal of the fast regulating converter. C_1 and C_2 thus charge with low loss (soft charging), and at a rate determined by the power drawn from the regulating converter to control the system output. Likewise, when the SC stage is configured for discharging C_1 and C_2 in parallel (switches P closed), the discharge is at a rate based on the power needed to regulate the output.

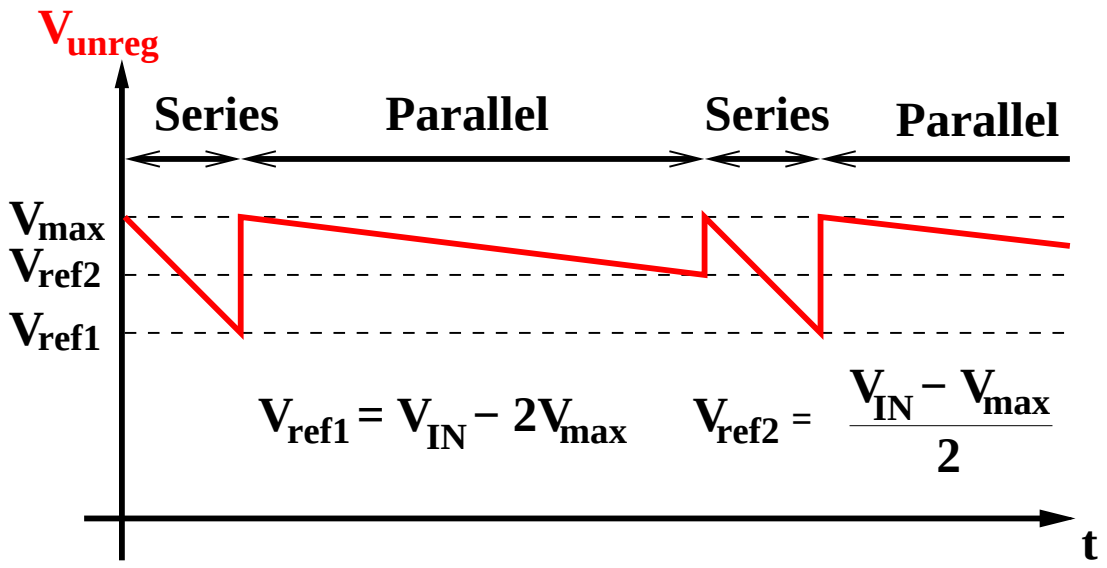


Figure 2.7: *Example of control strategy based on maximum input voltage of regulating converter.*

In operating the system, the SC stage can be controlled to provide a specified maximum voltage V_{unreg} at the input of the regulating converter. Fig. 2.7 illustrates a control strategy utilizing this technique, where two separate reference voltages are used to ensure that the input voltage of the auxiliary converter does not exceed $V_{unreg,max}$. The reference voltages can be expressed in terms of $V_{unreg,max}$ and V_{IN}:

$$V_{ref1} = V_{IN} - 2V_{unreg,max} \tag{2.8}$$

$$V_{ref2} = \frac{V_{IN} - V_{unreg,max}}{2} \tag{2.9}$$

In this example, the switches S of Fig. 2.6 are on (series charging of the capacitors) until V_X falls below V_{ref1}. At this time, switches S turn off, and switches P turn on (parallel discharging of capacitors), until V_X falls below V_{ref2}, at which time the cycle repeats.

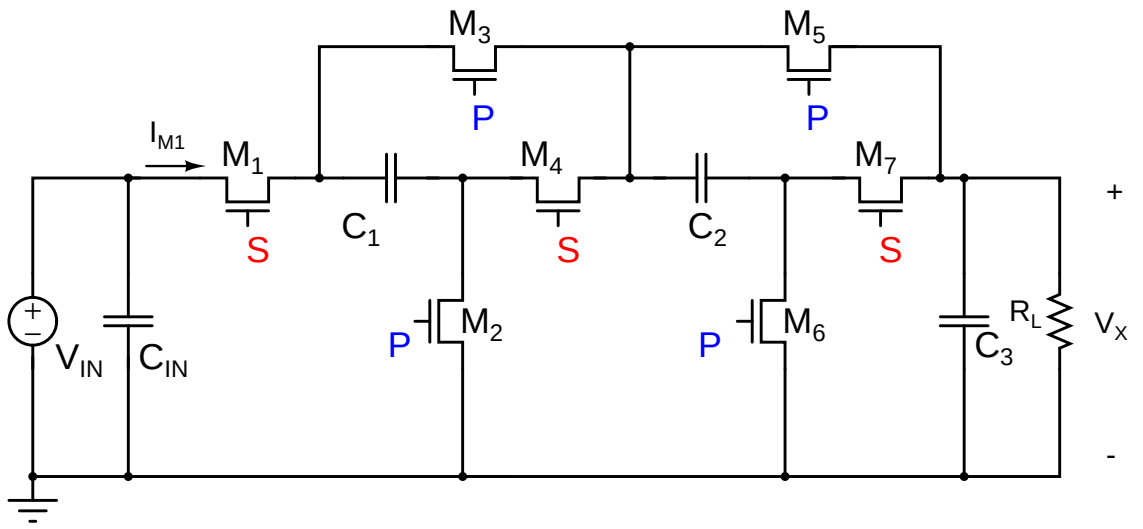


Figure 2.8: *Schematic diagram of simulation setup.*

2.3.1 Simulated Switched-Capacitor Results

To investigate the promise of the soft charging strategy, a 2 W, 3-to-1 switched-capacitor stage was simulated in SPICE, using device characteristics from a 90 nm CMOS process and discrete capacitors. In the analysis presented here, only the performance of the switched-capacitor stage is considered. Fig. 2.8 shows a schematic diagram of the simulated circuit, which consists of a a 3-to-1 stepdown SC stage with a resistive load. The input voltage is 5.5 V, R_{load}=1.68 Ω, and the switching frequency is 1 MHz. In a full two-stage converter, a regulating converter (switching at a frequency much higher than 1 MHz) would replace the load resistor.

For hard charging operation, capacitors $C_1 - C_3$ are all 10 μF, while for soft charging operation C_1 and C_2 are 1.5μF each, and C_3 is 0.01 μF. Table 2.1 presents one metric of the improvement offered by the merged two-stage converter. Listed is the required capacitance for a 98% efficient transformation stage, for both a conventional (hard charging) SC converter and one implementing the soft charging technique. For the same efficiency, the soft charging implementation enables a 10x reduction in required capacitance compared

Table 2.1: *Capacitance requirements for two-stage converter*

Converter	Hard Charging	Soft Charging
Total Capacitance	30 μF	3 μF (10%)
Total Capacitor Volume	3.072 mm^3	0.5 mm^3 (16%)
Total Capacitor Area	3.84 mm^2	1 mm^2 (26%)
Discrete Capacitor Sizes	3 x 0603	2 x 0402

to the hard charging transformation stage. If total capacitance is instead kept constant in the comparison, overall efficiency gains can be realized using the soft charging technique.

The soft charging characteristics of the merged two-stage converter is best illustrated by the waveform of the switch current. In a conventional SC converter operating in the slow-switching limit, this current will have a large, exponentially decaying peak on top of a steady-state charging current. This peak corresponds to capacitor charging loss, which can be a substantial part of the overall converter loss. Fig. 2.9 shows the switch current (I_{M1} of Fig. 2.8) for a conventional SC converter, and that for a converter utilizing the soft charging techniques. As is evident from the figure, the soft charging technique enables a drastic reduction in peak and rms switch current and the associated loss. The output voltage of the SC stage (V_X of Fig. 2.8) is shown in Fig. 2.10. The substantially larger voltage ripple associated with the soft charging technique is evident from the two waveforms. This would be undesirable in a SC converter operating as a single stage (whose output voltage is the system output voltage), but in the merged two-stage topology this voltage merely corresponds to an input voltage to the regulating converter that changes slowly (compared to the switching frequency of the regulating converter). The regulating stage is designed to provide a steady output voltage despite a time-varying input voltage such as that shown in Fig. 2.10.

This chapter has provided an introduction to, and overview of, the concept of *soft charging* switched-capacitor converters. The next two chapters present two experimental prototypes designed to operate with increased voltage ripple under soft charging conditions. We will illustrate some key benefits and drawbacks of the technique, and present solutions to some

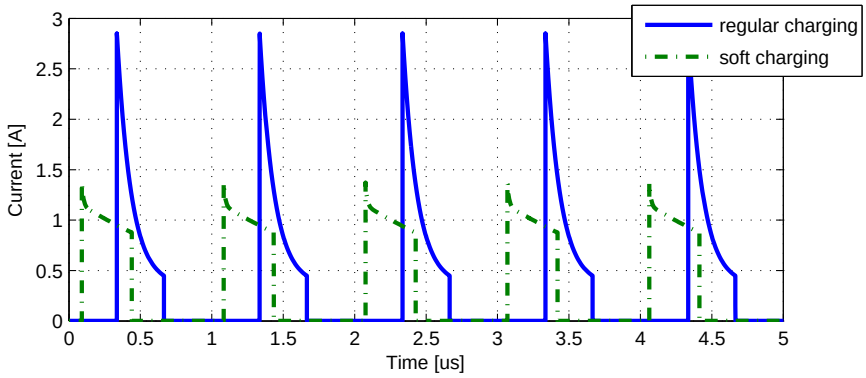


Figure 2.9: *Simulated current waveforms (I_{M1} of Fig. 2.8) for the SC converter stage, illustrating reduced peak currents (and, correspondingly, reduced loss) for the soft charging case.*

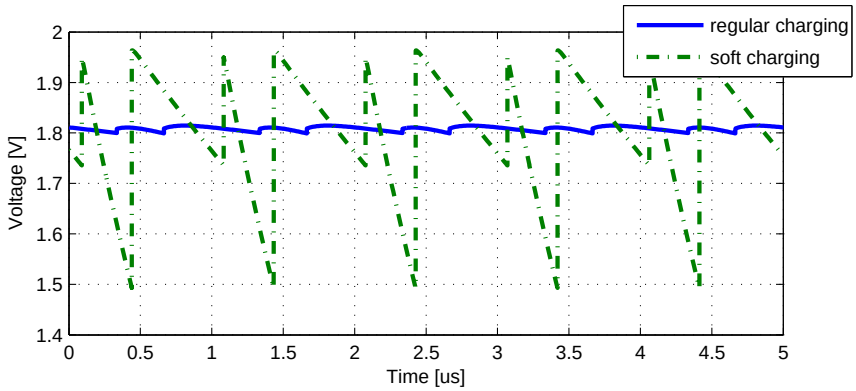


Figure 2.10: *Simulated output voltage waveforms (V_X of Fig. 2.8) for the SC converter stage, illustrating increased voltage ripple for the soft charging case.*

of the challenges involved in making this converter architecture work in practice.

Chapter 3

Discrete Implementation of Merged Two-Stage Power Converter

In order to verify the validity of the merged two-stage strategy, an experimental prototype was designed. It should be noted that while the ultimate target platform of the proposed converter is a low-voltage integrated process, this initial discrete prototype is not implemented in such a process, and thus does not achieve the design scaling and power density that is possible with this technique. The purpose of this discrete prototype is purely to validate the concept and provide insights for designs based on integrated processes.

Fig. 3.1 shows a photograph of the prototype converter, which consists of a 3-to-1 switched-capacitor stage coupled with a commercial synchronous buck converter (LTC3418). The SC stage is controlled in the manner described in Fig. 2.7 using a microcontroller (ATtiny24) with a built-in comparator to sense the different thresholds, and to provide the logic signals for the gate drive chips. These components are placed on the backside of the board (not shown). A schematic drawing of the converter is shown in Fig. 3.2, and component values are listed in Table 3.1. The relatively large values of C_{IN} and C_{OUT} are used to ensure steady input and output voltages for more precise efficiency measurements. Resistors R_1-R_4 are used to set the reference voltages V_{ref1} and V_{ref2} (Fig. 2.7) which determine the discharge level of the capacitors.

Appendix B provides the microcontroller code for the prototype, and Appendix A contains a complete Bill of Materials, full Eagle schematic, and PCB images.

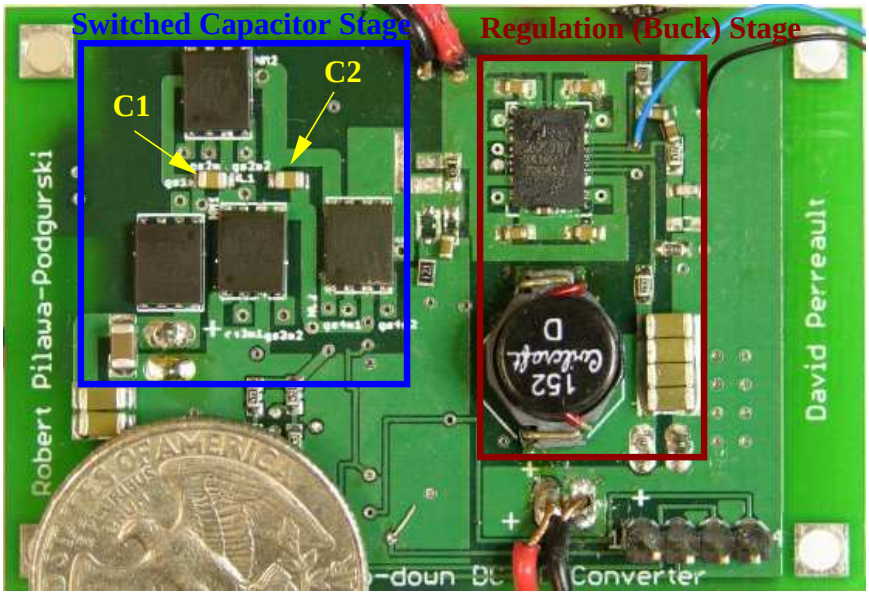


Figure 3.1: *Photograph of experimental prototype with switched-capacitor stage and regulation stage outlined. U.S. quarter shown for scale.*

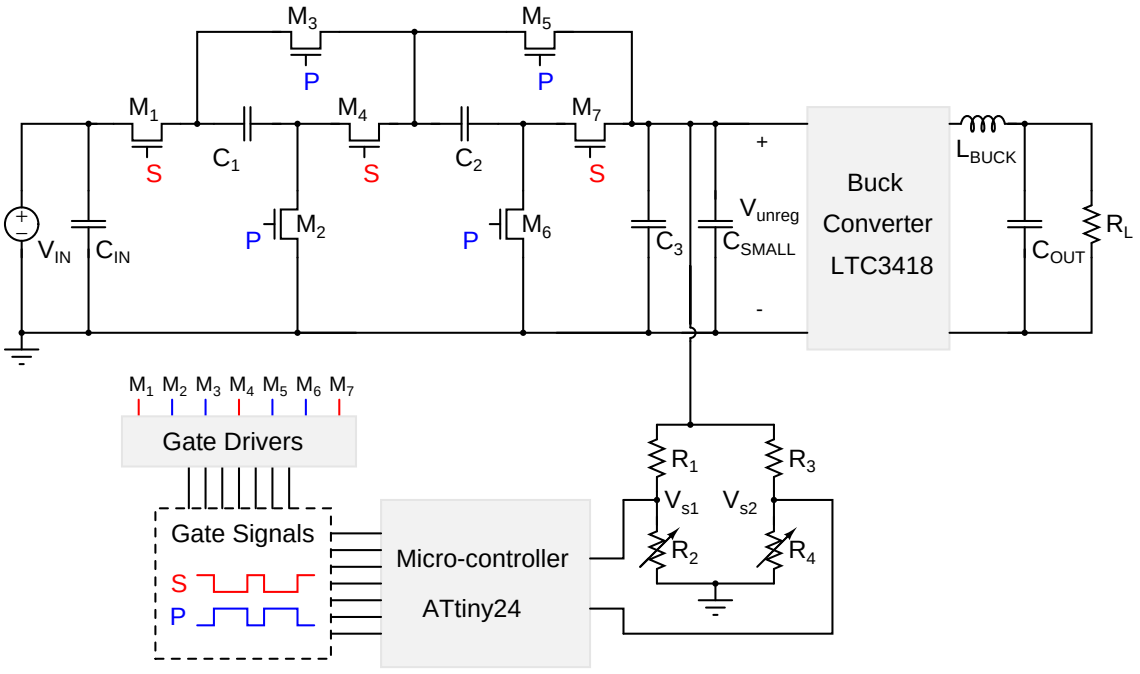


Figure 3.2: *Schematic of experimental prototype. The microcontroller samples the output voltage through the voltage dividers, and alternates the SC stage between parallel and series configuration.*

Table 3.1: *Component Values for Prototype Converter*

Component	Value
C_{IN}	5x22 μF
	1x2200 μF
M_1-M_7	Si7236DP
C_1	10 μF
C_2	10 μF
C_3 hard charging	10 μF
soft charging	1 μF
C_{SMALL}	4x0.1 μF
L_{BUCK}	1.5 μH
C_{OUT}	1x100 μF
	1x2200 μF
R_1 & R_2	47 kΩ
R_3 & R_4	0-50 kΩ
Gate Drive (high side)	LTC4440-5
Gate Drive (M_2 & M_6)	LM5111

The floating high-side gate drive chips are powered from the energy stored on the capacitors C_1 and C_2, while the low-side gate drive chips and the microcontroller are powered from a separate, low-voltage supply. In addition to implementing the control strategy, the microcontroller initiates the startup sequence, which coordinates the turn-on of the SC switches to provide power to the gate drive chips, and to ensure that the output voltage of this stage (V_{unreg}) stays below its allowed maximum value. The synchronous buck converter, LTC3418 from Linear Technology, is set to operate at a switching frequency of 1 MHz, and an output voltage of 2 V.

3.1 Hard and Soft Charging Comparison

To evaluate the merits of soft charging operation in the merged two-stage converter architecture, we compare it to traditional hard charging operation. By placing capacitor C_3 in the circuit of Fig. 3.2 that is the same size as C_1 and C_2 (10 μF), the SC stage implements regular hard charging operation. For soft charging operation, C_3 instead consists of a small

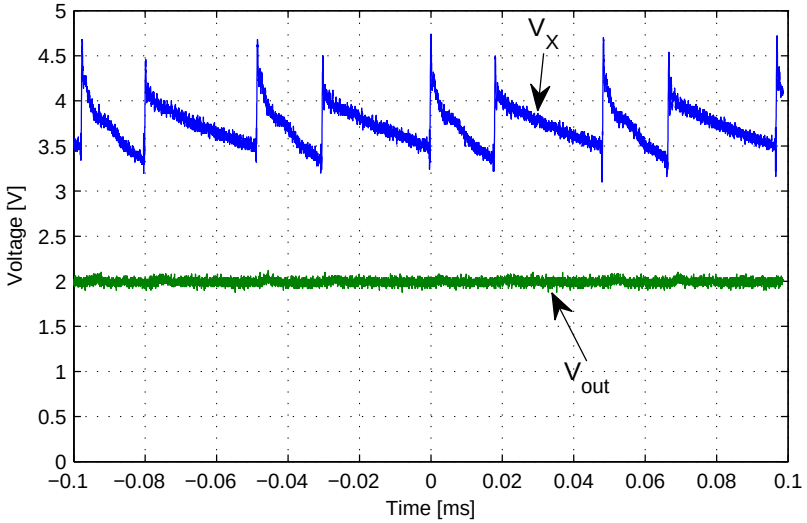
capacitor (1 μF) to filter the 1 MHz input current ripple of the LTC3418.

Fig. 3.3 shows the measured input and output voltage of the buck converter (corresponding to V_X in Fig. 2.6), for V_{IN}=12 V, V_{OUT}=2 V, and I_{OUT}=0.4 A for soft and hard charging operation. The switching of the SC stage can be clearly seen in the V_X waveform, with alternating series charging and parallel discharging of the capacitors. Since the switching frequency of the buck converter is much higher (1 MHz) than the frequency of the SC stage (∼20 kHz for this load), V_{OUT} can be well regulated with small ripple despite the large ripple seen at V_X, as illustrated by the V_{out} waveforms. Fig. 3.3 also illustrates the larger ripple of V_X for soft charging compared to hard charging, which is consistent with our earlier discussion. Note that in the case presented in Fig. 3.3, C_1 and C_2 have the same values for both soft and hard charging operation, and the frequencies of the two modes of operation are made to be approximately equal. In contrast, the simulated waveforms shown in Fig. 2.10 shows the case where C_1 and C_2 are drastically smaller in the soft charging case while overall efficiency is the parameter that is kept constant for the two modes of operation. Consequently, the voltage ripple at the input of the buck converter is significantly larger for soft charging compared to hard charging in Fig. 2.10.

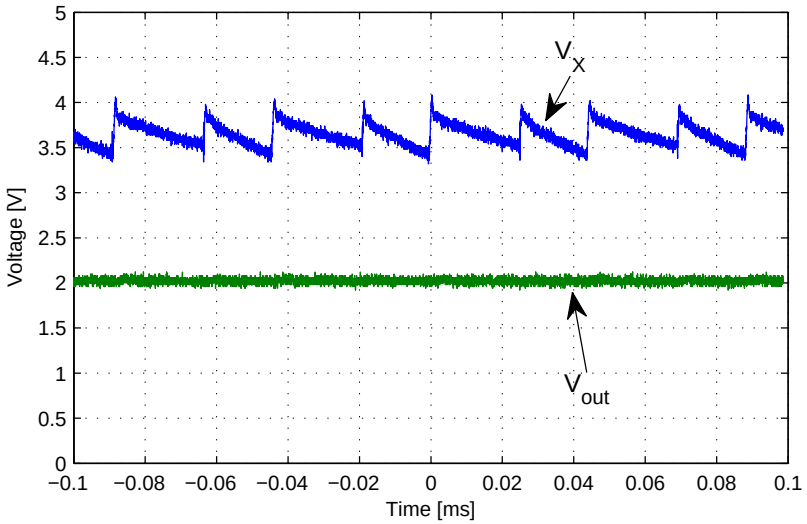
In addition to decreased capacitance requirement and reduced current spikes, efficiency improvement is a key benefit of soft charging operation. To estimate the efficiency gains realized by soft charging operation in the discrete implementation presented here, a comparison to hard charging operation was made over a wide load range. It is important to note that the objective of the discrete prototype presented here is not to obtain the highest efficiency achievable, but rather to investigate the feasibility of the soft charging architecture for cases where total capacitance is limited. Thus, absolute measures of efficiency is not the metric with which to evaluate the proposed converter, but rather the relative improvements offered by soft charging.[1]

The resulting efficiency measurements shown in Fig. 3.4 illustrate the efficiency improve-

[1] Absolute efficiency can, in this case, be improved by utilizing larger energy transfer capacitors ($C_1 - C_3$).



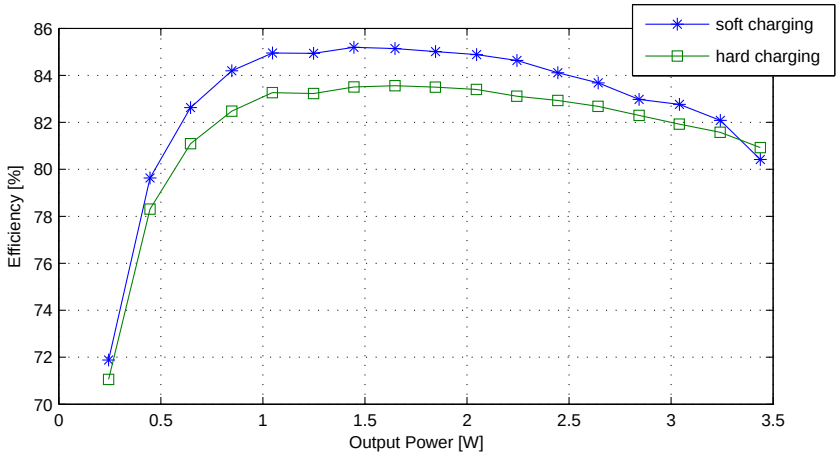
(a) Soft Charging.



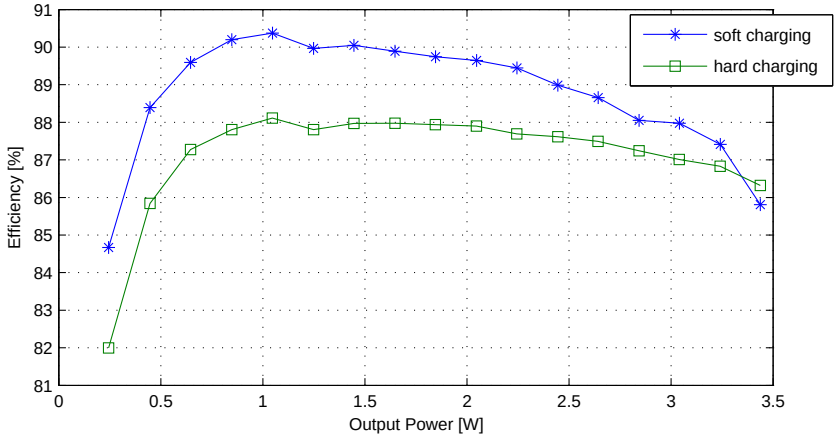
(b) Hard Charging.

Figure 3.3: *Measured input and output voltage of the buck converter in the experimental prototype for soft and hard charging implementation.*

ments offered by the soft charging implementation. The efficiency of the SC stage alone is estimated by measuring the efficiency of the buck converter across the load range, and subtracting its loss from the overall converter loss. The resulting estimated SC stage efficiency is shown in Fig. 3.4b. It is clear from this plot that soft charging offers a noticeable improvement in efficiency. At 1 W load, the estimated power loss in the SC stage is 25% higher for hard charging than for soft charging. This work, also presented in [18], sets the stage for development of an integrated merged two-stage power converter, which is described in the next chapter.



(a) Measured Overall Converter Efficiency.



(b) Estimated SC Efficiency.

Figure 3.4: *Efficiency measurements for discrete prototype converter for soft and hard charging operation.*

Chapter 4

180 nm CMOS Integrated Merged Two Stage Converter

The discrete implementation described in the previous chapter was designed in order to evaluate the soft charging switched-capacitor technique. A discrete design enables probing and replacement of components, and offers substantial flexibility. However, the merged two-stage architecture is perhaps better suited for integration using a CMOS process, where the designer can make use of the various transistor options available. The capability to tailor the type and sizing of the power transistors is especially beneficial in the merged two-stage converter, which makes use of the different types of CMOS transistors available in the process. Furthermore, whereas the number of transistors required for this power converter topology can be difficult to layout and populate in a discrete implementation (due to cost and interconnect requirements), it is not a problem in an integrated process. In a CMOS power converter, it is not the *number* of transistors that dictate cost, but rather the total die area consumed by the chip. In this chapter the design and experimental validation of a merged-two stage converter implemented in 180 nm CMOS is presented. The converter, with an input voltage of 5-5.5 V, and output voltage of 1-1.3 V, delivers 0.8 Watt to the output, and operates the low-votage regulation stage at 10 MHz, while achieveing a more than five-to-one step-down ratio. The peak efficiency is more than 80%, including all gate drive and control losses, as well as bond-wire and packaging losses. Technical solutions to various challenges in implementing an integrated merged-two-stage design are also presented. As will be seen, the viability and high performance of this approach is demonstrated.

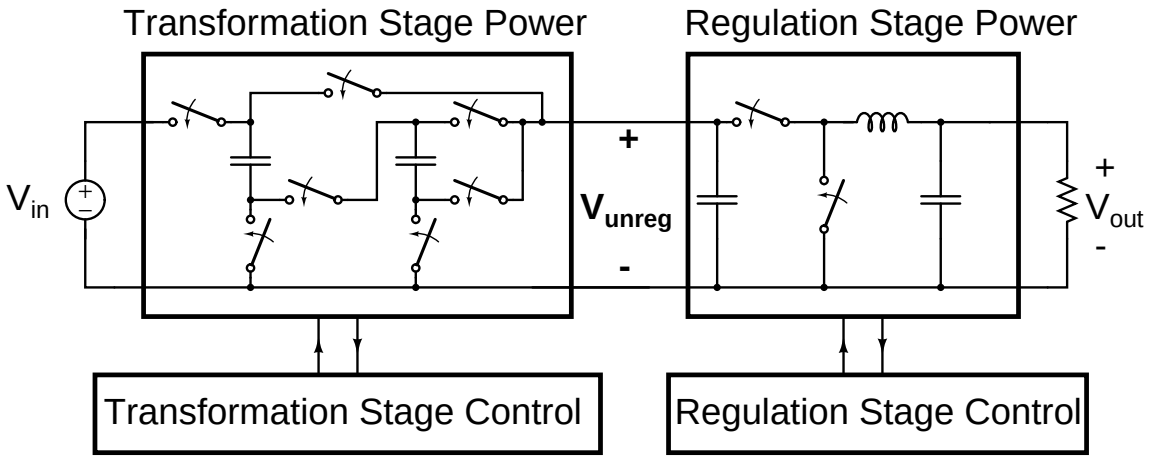


Figure 4.1: *Schematic drawing of the CMOS integrated merged two-stage converter.*

4.1 Converter Overview

The schematic diagram of Figure 4.1 illustrates the different parts of the merged two-stage converter, and how they relate to one another. The converter achieves both a large voltage step-down (5 V to 1 V) and high frequency operation (10 MHz) by utilizing slow, high-voltage (5 V) and fast, low-voltage (2 V) devices, both of which are available in the 180 nm CMOS process. The SC transformation stage employs the high-voltage switches, and operates at a relatively low switching frequency (< 200 kHz), while the synchronous buck regulation stage operates at a switching frequency of 10 MHz , thanks to use of low-voltage core transistors. The focus of this work (presented in [21]) was to demonstrate the implementation of the soft charging architecture in a CMOS process. In the following sections, we will illustrate some of the design choices challenges associated with implementing the architecture of Figure 4.1.

4.2 Transformation Stage Control

As part of this research a complete control implementation was developed and tested in 180 nm CMOS. The two-level hysteretic control strategy described in Chapter 2 (shown in

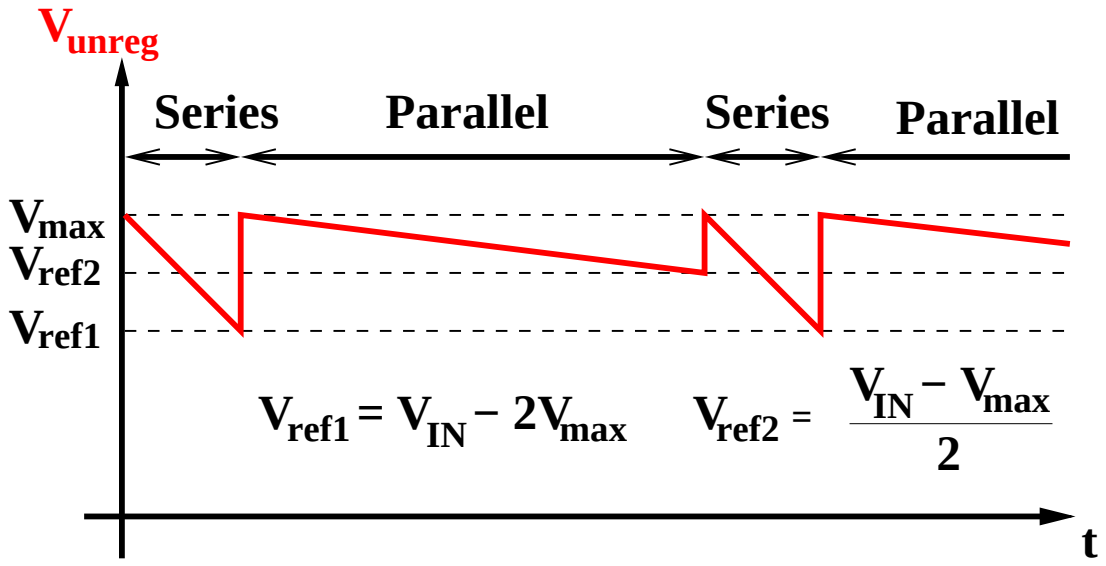


Figure 4.2: *Two-level hysteretic control strategy of SC transformation stage.*

Fig. 2.7, repeated here as Fig. 4.2) was implemented in CMOS for the switched-capacitor part of the two-stage converter. A schematic drawing of the control circuitry is shown in Fig. 4.3. The two different reference voltages (V_{ref1} and V_{ref1}) are provided from an off-chip source in this implementation, to allow flexibility in the characterization of the control technique. A flip-flop is used to keep state of the operation mode (series or parallel), and the inverted output controls a multiplexer such that the corresponding comparator output is used to trigger a change in series-parallel operation.

The one-shot circuitry (with details provided in Figure 4.4) is used to introduce a blanking time of approximately 0.18 μs immediately following a comparator transition. This is added as a safeguard against any oscillations caused by the other comparator. Without the blanking period, a high output of the other comparator could be propagated to the flip-flop when the multiplexer changes. The blanking period is chosen to be long enough to prevent this from accidentally happening (based on simulation), and must not be so long that it interferes with the correct switching operation (i.e. must be significantly shorter than a switching period of the SC stage).

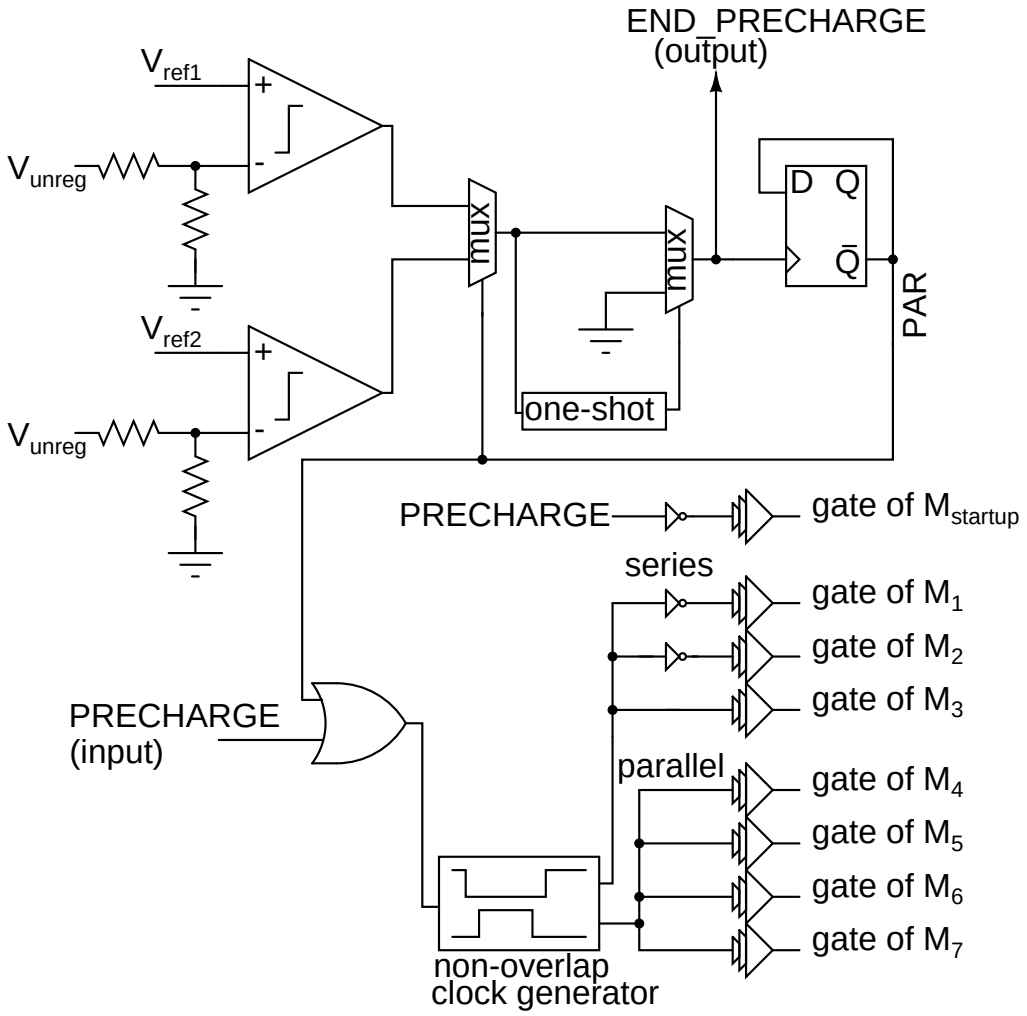


Figure 4.3: *Schematic drawing of the SC stage control implementation.*

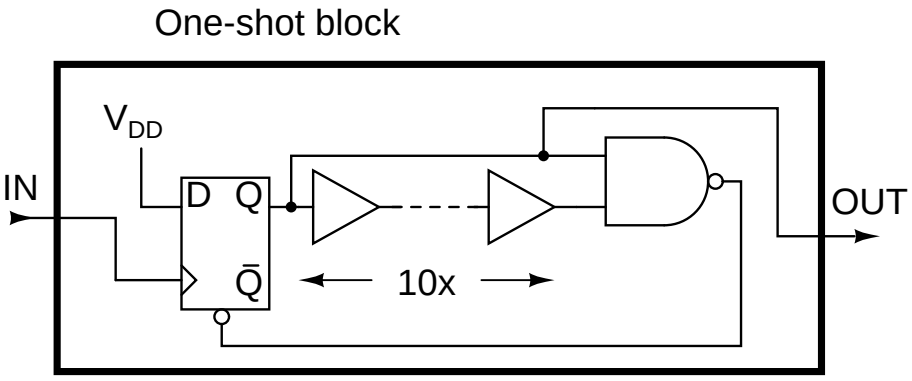


Figure 4.4: *Schematic drawing of the one-shot circuitry used in Figure 4.3. The one-shot circuit is used to introduce a blank-out period when the output of the comparator is not propagated to the rest of the control circuit.*

Finally, a programmable non-overlap generator (shown in Figure 4.5) is used to ensure that there is sufficient dead-time between the transitions of the series-parallel modes. All the control circuitry was implemented using low-voltage 180 nm transistors, while the final tapered gate drivers (discussed in Section 4.3.1) employed high-voltage devices. The transistors corresponding to the output of the tapered gate drivers are shown in the schematic drawing of Figure 4.6. Note that two of the SC power stage transistors (M_1 and M_2) are implemented as PMOS devices, and therefore requires inverted gate drive signals.

4.2.1 Startup

A key challenge in switched-capacitor converters is the issue of startup conditions. While it is true that the individual transistors and capacitors in a SC converter typically only see fractions of the input voltage in steady-state operation, large voltage stresses can develop across individual components during startup. In the merged two-stage converter, it is therefore critical to implement a startup sequence that ensures that the voltage across all transistors and capacitors remain below their rated voltage. Since the switched-capacitor transistors and capacitors are all rated for a voltage higher than (or equal to) the input voltage, the critical voltage that must be controlled is the output voltage of the SC stage

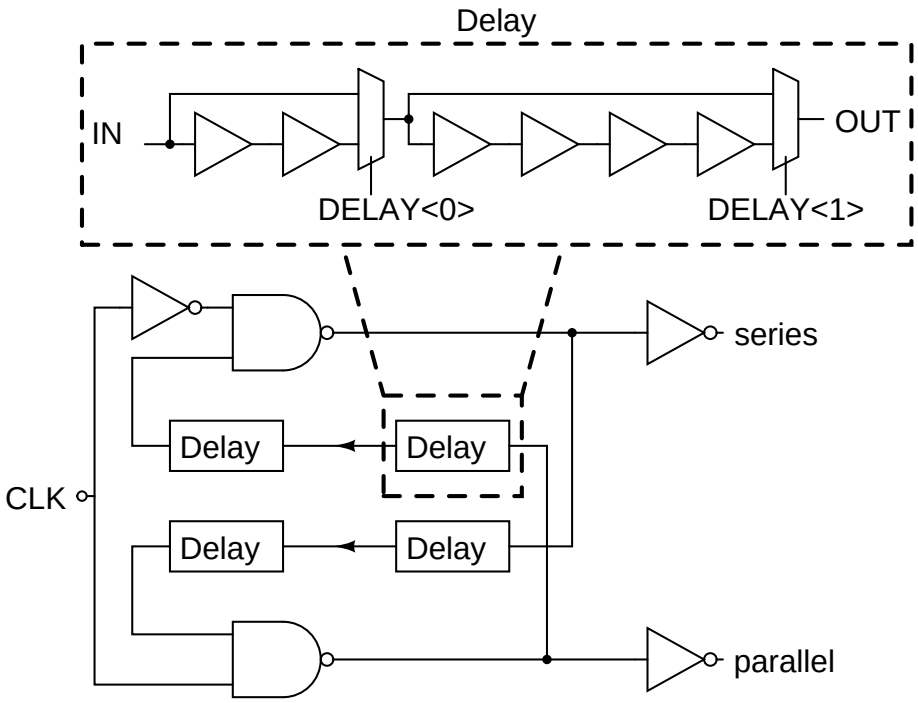


Figure 4.5: *Schematic drawing of the non-overlap generator with 2-bit programmable delay.*

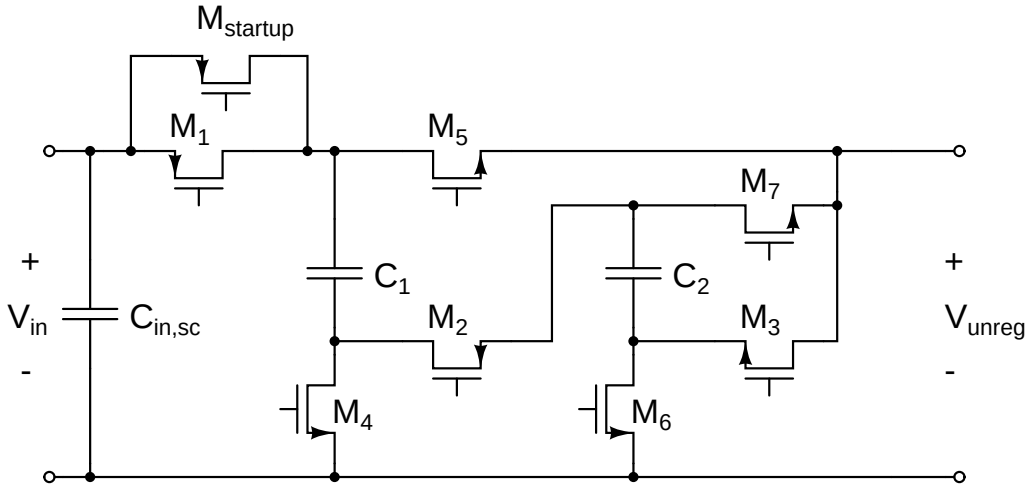


Figure 4.6: *Schematic drawing of the switched-capacitor transformation stage. The capacitors are off-chip, and the transistors are 5 V triple-well thick-oxide devices available in the 180 nm CMOS process.*

(V_{unreg}). It can be easily seen from Figure 4.6 that if capacitors C_1 and C_2 have no charge (which will be the case if V_{in} has been kept low for some time), and the SC stage is configured to operate in series mode (M_1, M_2, M_3 closed), the full input voltage (5 V) appears across the output terminals of the SC stage. Since these terminals are also connected to the input of the low-voltage (2 V devices) devices of the regulation stage (as shown in Figure 4.1), care must be taken to never allow V_{unreg} to go above 2 V.

Shown in Figure 4.7 is a schematic drawing of the startup circuitry. It employs a comparator that compares the output voltage of the SC stage to a reference voltage ($V_{ref-startup}$), which is lower than V_{ref1} and V_{ref2}. The AND logic block is used together with a slow clock to ensure that the the flip-flop will indeed trigger when the startup is detected. Since the flip-flop is of the edge-detect type, there could be a situation at startup where the comparator output is not detected if the flip-flop is not properly initialized before the signal arrives at the clock input. The slow clock and the AND block ensures that once the comparator has detected an under-voltage situation, this information will be captured by the flip-flop. Finally, the multiplexer is used together with an SC-ENABLE signal to ensure that the pre-charge signal is not initiated when the SC stage is not enabled.

The pre-charge signal is applied to the startup transistor ($M_{startup}$, as shown in Figures 4.3 and 4.6), which has a gate width many times smaller than the other power transistors. The pre-charge signal also drives the input node of the non-overlap clock generator high (through the OR block, as seen in Figure 4.3). This ensures that the SC stage remains in parallel mode while the precharge signal is high. Transistors M_4, M_5, M_6 and M_7 are thus on, and the output node V_{unreg} is slowly brought up from zero volts through the transistor $M_{startup}$. The pre-charge phase is turned-off by the END-PRECHARGE command from the circuit of Figure 4.3, which goes high once the voltage is high enough to trip one of the other two comparators. At this point, the control circuitry transitions to two-level hysteretic control of the SC stage. The advantage of this startup scheme is that it only requires one additional (small) power transistor, and a few additional analog and digital blocks. In regular operation, the $M_{startup}$ transistor is not used, and does not incur any additional

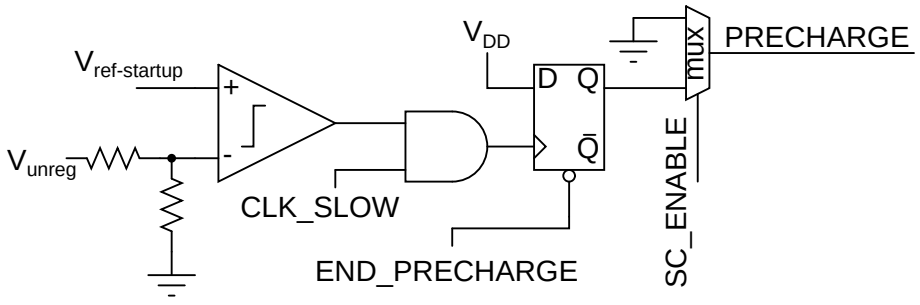


Figure 4.7: *Schematic drawing of the SC startup control circuitry. At startup, this control circuitry ensures that no node voltages exceed the ratings of the on-chip transistors.*

loss, as compared to some other series-connected startup schemes in the literature [22].

4.2.2 Comparators

Figure 4.8 shows a schematic drawing of the comparator stage, adapted from [23]. It is made of low-voltage (180 nm) transistors, and consists of an NMOS pre-amplification stage, a decision circuit with positive feedback, and an NMOS differential amplifier followed by an inverter. The inverter adds additional gain and also isolates the differential amplifier from any load capacitance. The focus on the design was to achieve a fairly wide input voltage operating range, whereas speed was not a critical consideration because of the low frequency operation of the SC stage. Any delay in the comparator would have the effect of adjusting the effective values of V_{ref1} and V_{ref2}, which can be compensated for in the choice of reference voltages.

4.3 Transformation Stage Power and Gate Drive Devices

A schematic drawing of the SC transformation stage is shown in Figure 4.6. The transistors in the SC stage are 5 V isolated triple-well thick-oxide devices with extended drain regions. The capacitors C_1 and C_2 are 22 μF off-chip ceramic (X5R) capacitors, and the transistor $M_{startup}$ is activated during startup (by the on-chip control circuitry) to ensure that the SC

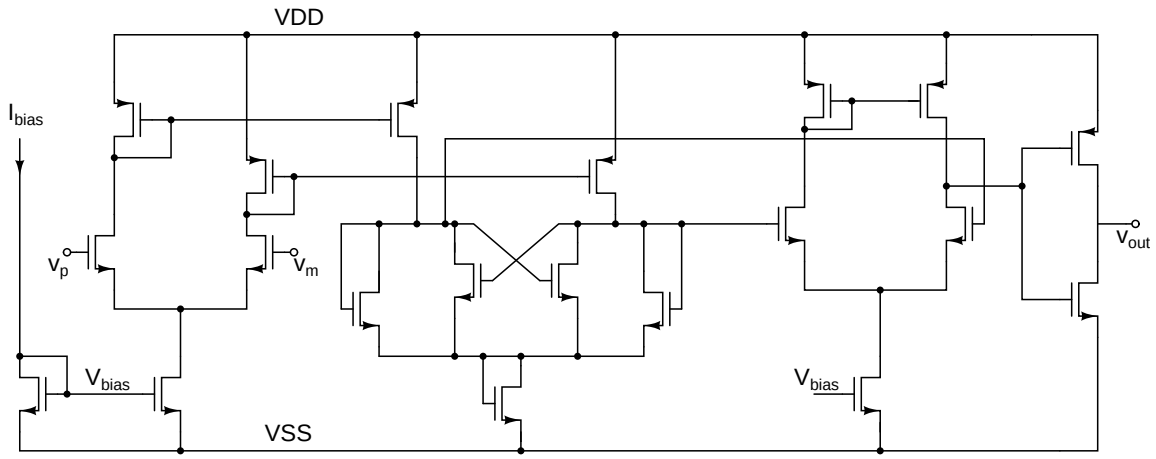


Figure 4.8: *Schematic drawing of the comparator used in the SC stage. The bias current in this design is 1 μA, and all transistors are low-voltage core logic devices, in the 180 nm CMOS process.*

output voltage never rises above 2 V (the maximum working voltage of the regulation stage transistors) by slowly charging capacitors C_1 and C_2. During regular operation, $M_{startup}$ remains off. The NMOS devices (M_3, M_4, M_5, M_6 and M_7) each have a gate length of 600 nm, and each have a total device width of 0.17 meters, layed out in a multi-fingered structure. The PMOS devices (M_1 and M_2) also have a gate length of 600 nm, and a total device width of 0.374 meters.

4.3.1 Tapered Gate Drives

In order to drive the large power devices of the SC stage at high speed, tapered gate drivers must be used. The gate drivers, also made from 600 nm gate-length 5 V devices, have six stages with a tapering factor of 10, which represents a good balance between switching speed and gate drive loss, as determined by simulation. Figure 4.9 shows a schematic drawing of the gate drive section, which consists of a level shifter driving a tapered buffer stage with a tapering factor of a (a = 10 in this design), followed by one of the power transistors. In this design, all gate drivers were powered from 5 V, and connected to the ground potential. For improved performance, it is beneficial to use flying (high-side) gate drivers for power

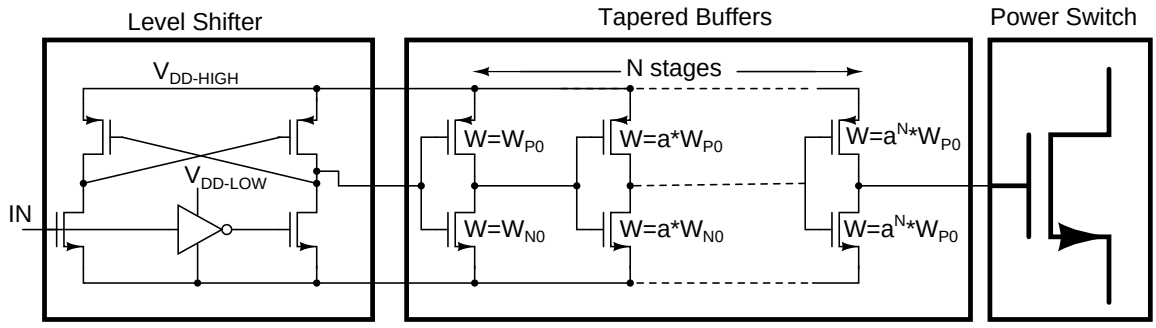


Figure 4.9: *Tapered gate drive circuit with a tapering factor of a (a=10 in this design). The level shifter interfaces the low-voltage control circuitry to the higher gate drive voltage. The implemented design uses N = 6 buffer stages.*

transistors that are not ground referenced (i.e. all transistors of Figure 4.6 except M_4 and M_6). While this would help decrease their on-state resistance by driving their gate-source voltage higher thus better enhancing the devices, it adds significant complexity to the gate drive circuit. For this initial prototype, we decided to keep the gate drive simple and slightly underdrive the transistors (using 0-5 V operating voltage), at the expense of somewhat lower efficiency.

4.4 Regulation Stage Control

The job of the regulation stage in the two-stage architecture is to keep the converter output voltage V_{out} steady at the desired value. The regulation stage must keep the output voltage within an acceptable range despite variations in converter input voltage and load current. Figure 4.10 shows a block diagram of a typical feedback implementation for a buck converter. Here ΔV_{osc} is the amplitude of the triangle-waveform used to generate the PWM signal, $G_{vs}(s)$ is the input-to-output voltage small signal transfer function (also known as *audio susceptibility*) of the buck converter, $H(s)$ is the sensor gain, and $G_c(s)$ is the transfer function for the compensator. For a buck converter using voltage mode control, the duty-

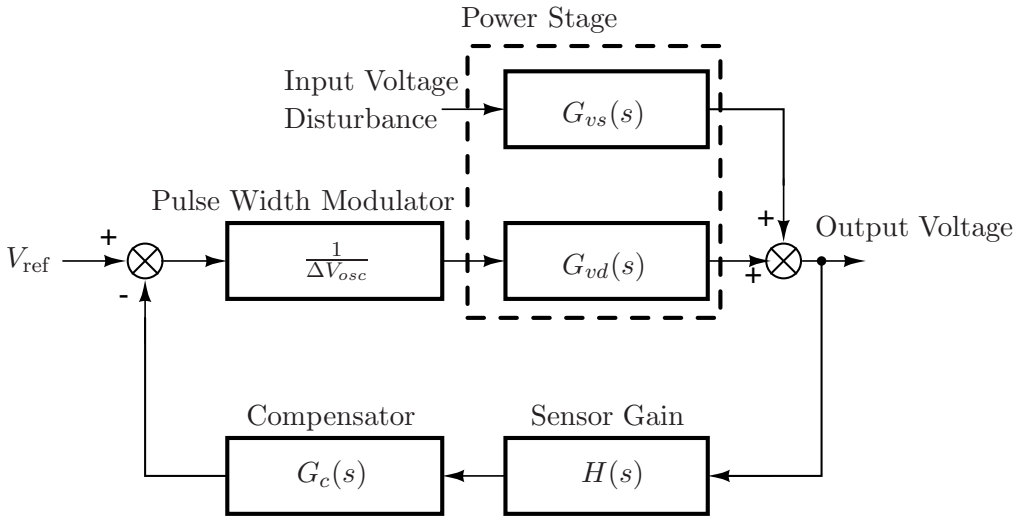


Figure 4.10: *Block diagram illustrating conventional feedback in buck converter.*

cycle-to-output-voltage transfer function $G_{vd}(s)$ can be approximated by [24], Chapter 8:

$$G_{vd}(s) = \frac{V_{OUT}}{D} \frac{1}{1 + s\frac{L}{R} + s^2 LC}, \tag{4.1}$$

where ideal components are assumed for simplicity. While it is possible to calculate the transfer function including parasitics, it is typically easier to complete the initial compensator design assuming no parasitics, and to later fine-tune the compensator by simulation, where parasitics are taken into account. At the frequency that we are operating, our buck converter output capacitor will be a ceramic capacitor with low ESR, which makes this assumption better than for designs that employ electrolytic capacitors whose large ESR can have a substantial impact on the transfer function.

Shown in Fig. 4.11 is a Bode plot of the system of in Fig. 4.10, with power stage transfer function as given in (4.1), with the parameters as listed in Table 4.1. All components are assumed ideal in this case. Also shown (labelled "Simulation") is a Bode plot of a simulation which incorporates parasitic resistances of 10 mΩ inductor resistance ($R_{L,esr}$) and 2 mΩ ESR on the capacitor($R_{C,esr}$). The simulation was done in Spectre on Cadence, and used the open-loop small signal model of Fig. 4.12, which was adapted from [25] (Chapter 2-3).

Table 4.1: *Plant model and simulation nominal values*

Component	Value
V_{IN}	1.2 V
V_{OUT}	1 V
D	0.83 V
L	6.25 nH
R	0.25 Ω
C	2 μF
$G_c(s)$	1
ΔV_{osc}	0.5 (i.e. $G_{pwm} = 2$)
$H(s)$	1
F_{sw}	30 MHz

The analog multiplier blocks were implemented in Verilog-A, and the large-valued capacitor and inductor are there to establish the appropriate operating conditions, while isolating the ac feedback loop so that the small-signal behavior can be observed. This enables us to observe the small signal response of the circuit (v_{observe}) to a small signal stimulus (v_{AC}), while ensuring that the bias point remains constant.

Shown in Figure 4.13 is a schematic drawing of a circuit-based implementation of the feedback control described in the block diagram of Figure 4.10. In analog control circuits, the pulse width modulator (PWM) is typically implemented with a triangle (or sawtooth) waveform and a comparator, to translate the error voltage into a series of pulses of appropriate width to drive the gate of the main switches at the desired duty ratio. The compensator comprises an error amplifier with a compensation network (not shown in Figure 4.13). The sensor gain is typically just a resistive voltage divider that attenuates the output voltage to a level suitable for the input voltage range of the error amplifier. In the next section, we will discuss the implementation of the error amplifier and the compensation network to achieve both a fast transient response and stability.

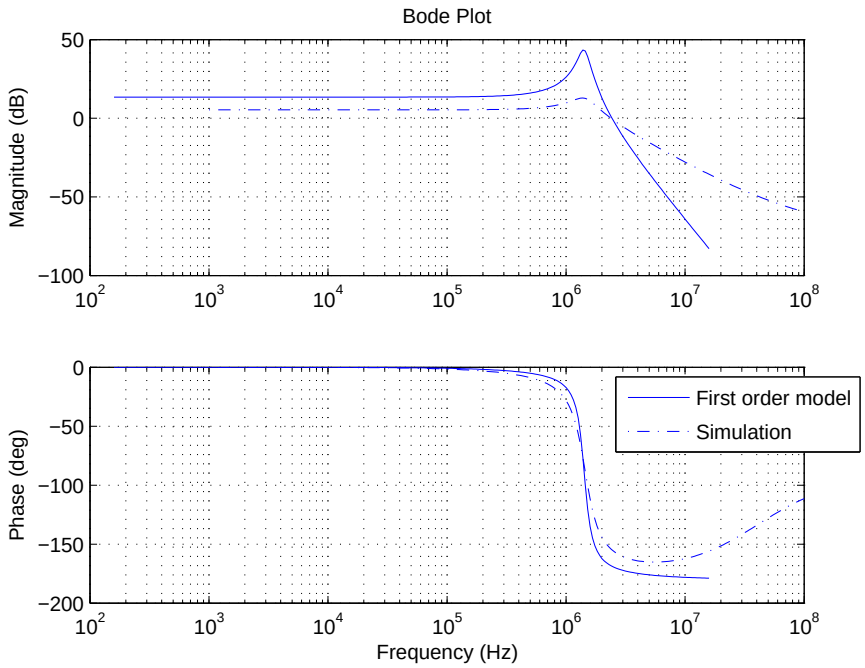


Figure 4.11: *Bode plots of a first order model uncompensated buck converter and a higher order simulation (as described in Fig. 4.12). At our desired crossover frequency (5 MHz), the simulated system has a gain of -15.5 dB, and a phase of -165°, thus requiring compensation to meet our performance goals.*

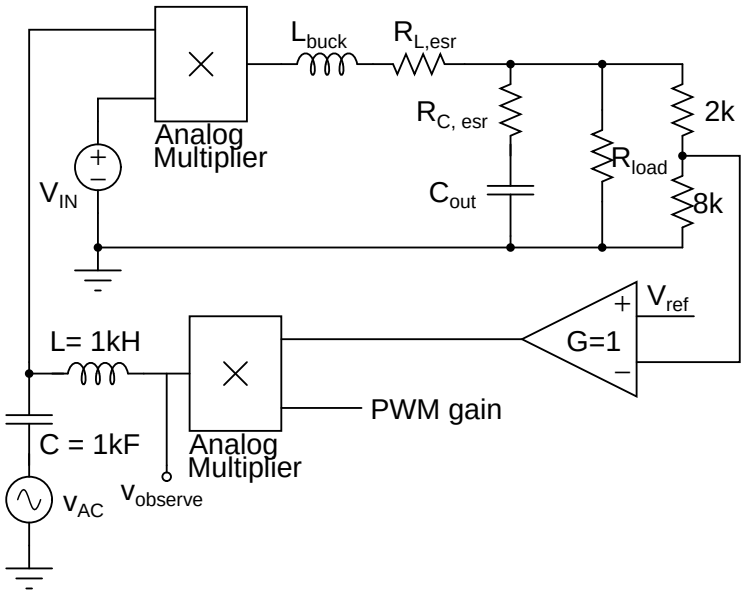


Figure 4.12: *Schematic drawing of small-signal buck converter model implemented in Spectre/Cadence to capture higher order behavior.*

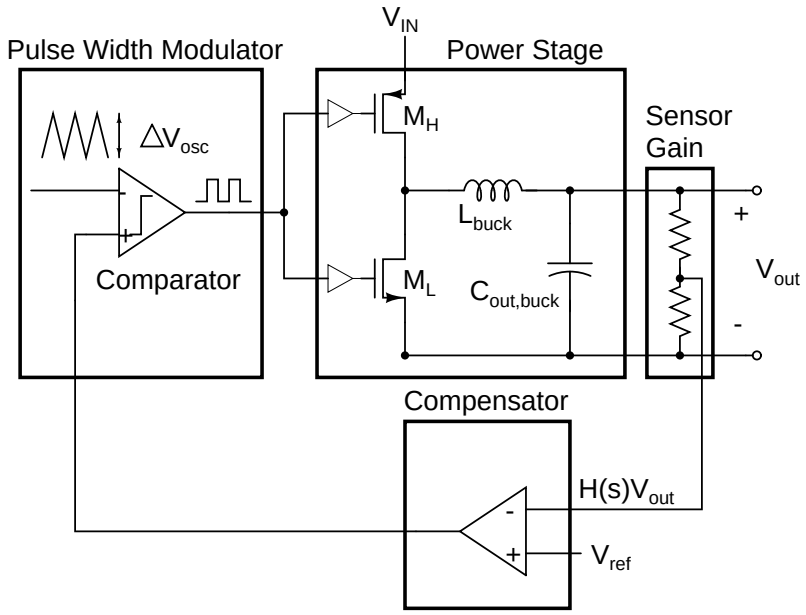


Figure 4.13: *Schematic drawing of a typical circuit implementation of the feedback control illustrated in Figure 4.10.*

4.4.1 Compensation Network

The soft-charging operation of the merged two-stage converter architecture achieves its low-loss operation at the cost of increased voltage ripple at the output of the SC stage. If the transformation stage of Figure 4.1 were to be implemented with a regular SC converter, there would be a large capacitor on the output of the SC stage to minimize the output voltage ripple, providing a stable input voltage of the regulation stage. However, with soft-charging operation, the large SC output capacitor is removed, and the output node of the SC stage (V_{unreg}) operates under large ripple, which can be problematic for the regulation stage if care is not taken in the design of the control stage for the buck converter.

Because the regulation stage is operating with substantial input voltage ripple, we seek to design our feedback control with a very high control bandwidth, so that we can reject these disturbances as much as possible. In practice, it means that our feedback loop compensation should have a high crossover frequency, while maintaining adequate gain and phase margins for stability. Type III compensation is often employed [26–29] in a buck converter to extend the crossover frequency by offsetting the double pole phase lag introduced by the L-C output filter. This is accomplished by utilizing two zeroes to provide a phase boost of 180 degrees. Figure 4.14 shows a schematic drawing of the Type III compensation network used in this network (corresponding to the "Compensator" block of Figure 4.13).

The method to achieve a desired performance using Type III compensation network is generally described in [26] and [25], Chapter 3. In our design, we wish to achieve a phase margin of 50 degrees, and a cross-over frequency of around 5 MHz (for a switching frequency of 30 MHz). This will require a large phase boost from the uncompensated system, and requires introduction of additional poles and zeros. The Type III compensation network places one origin pole for low DC error, a double zero to boost the phase before cross-over, and a double pole to bring down the high frequency gain. The exact placement of these poles and zeros can be computed using the k factor method [26], which provides an easy way to adjust the distance between the pole-zero pairs, and the corresponding R and C values

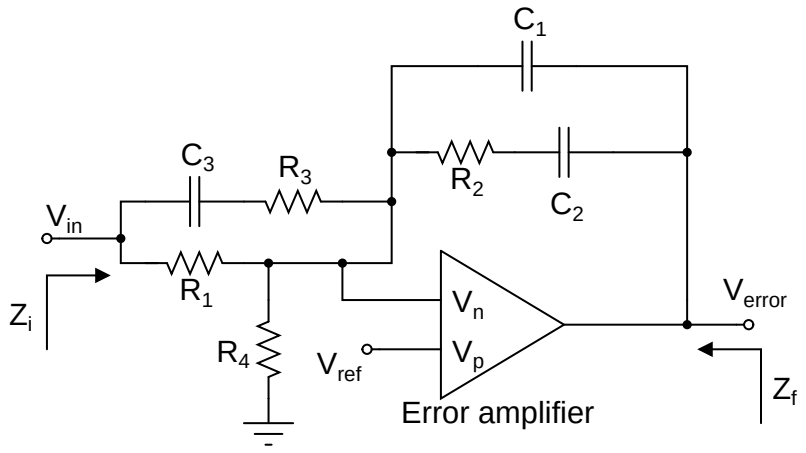


Figure 4.14: *Compensation network for Type III compensation of buck converter. Component values are listed in Table 4.2*

required. In our case, we require a phase boost of approximately 125 degrees (since our integrator will introduce an additional 90 degree phase shift from the open-loop system of Figure 4.11. Furthermore, to achieve a cross-over frequency around 5 MHz, we need to add a gain of around 15.5 dB (≈ 6). Using the k method, the R and C values of Figure 4.14 can be calculated to achieve our objectives. A MATLAB script that automatically calculates these values (based on [25], Chapter 3) is provided in Appendix D, for the interested reader.

The transfer function $G(s)$ of the compensation network of Figure 4.14 with an ideal amplifier can be expressed as:

$$G(s) = \frac{Z_f}{Z_i} = \frac{sR_2C_1 + 1}{sR_1(C_1 + C_2)\left(1 + sR_2\dfrac{C_1C_2}{C_1 + C_2}\right)} \frac{sC_3(R_1 + R_3) + 1}{sR_3C_3 + 1} \tag{4.2}$$

Figure 4.15 shows a Bode plot of this transfer function, for R and C values as calculated in Appendix D. This plot illustrates the additional phase and gain boost around our desired cross-over frequency.

The Bode plots of Figure 4.16 show the magnitude and phase of the small signal buck converter system as depicted in Figure 4.10 with parameter values as given in Table 4.1,

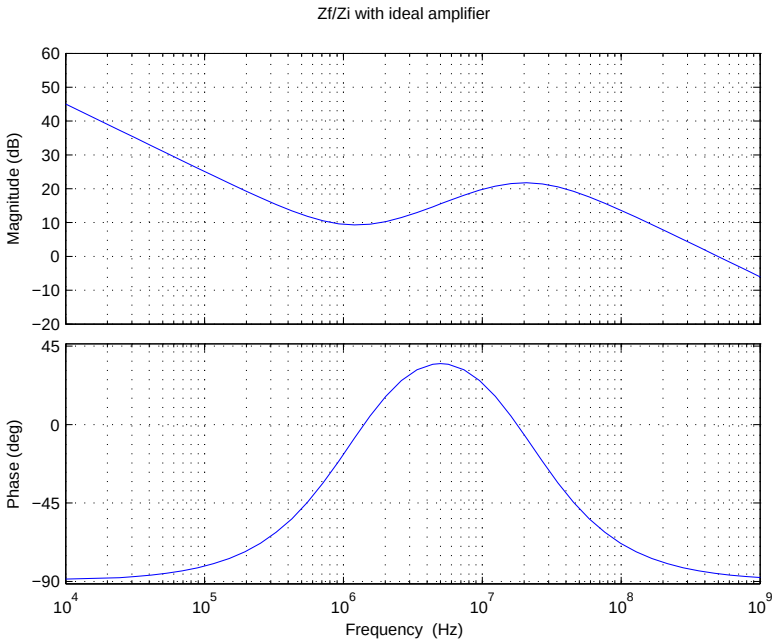


Figure 4.15: *Bode plot of transfer function of (4.2) with appropriately chosen values (as calculated in Appendix D).*

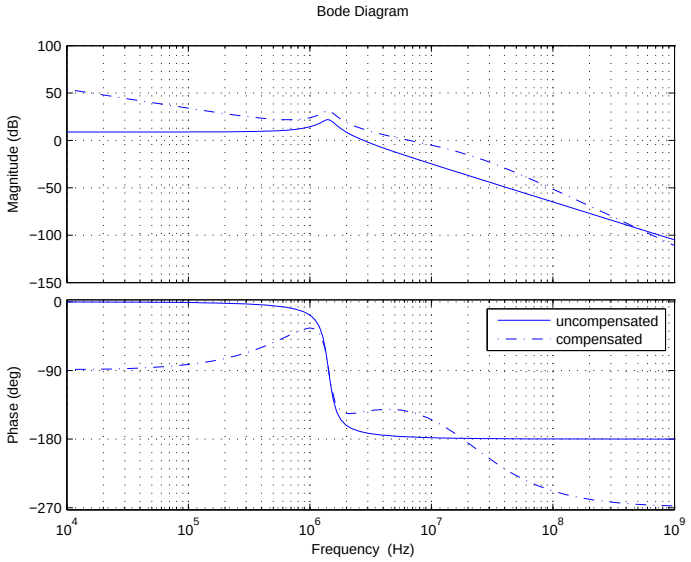


Figure 4.16: *Bode plot showing magnitude and phase of the* modelled *loop transfer function of first-order small-signal model of buck converter (as depicted in Figure 4.10), with and and without compensation network. Parameter values are listed in Table4.1.*

with and without compensation. The increased phase margin and cross-over frequency are apparent, but it should be noted that this plot only represents a starting point, as the parameters were derived from first-order models with ideal components. To mathematically model all parasitics of the buck converter and the non-idealities of the error amplifier is not feasible, but the initial compensation values can be fine-tuned in simulation, where high-level transistor model capture many of the non-idealities and parasitics associated with a circuit implementation.

A schematic drawing of the error amplifier used in the compensator is shown in Figure 4.17. It consists of a cascode current mirror, a PMOS input differential pair amplifier with current loads, and a common-source output stage to provide a large output voltage swing. It is worth noting that since this amplifier will be used only with our purpose-built compensation network, no dominant-pole compensation was used in the amplifier itself. In fact, a dominant-pole compensated amplifier would not be able to achieve the performance required for our desired cross-over frequency and phase margin in this CMOS process.

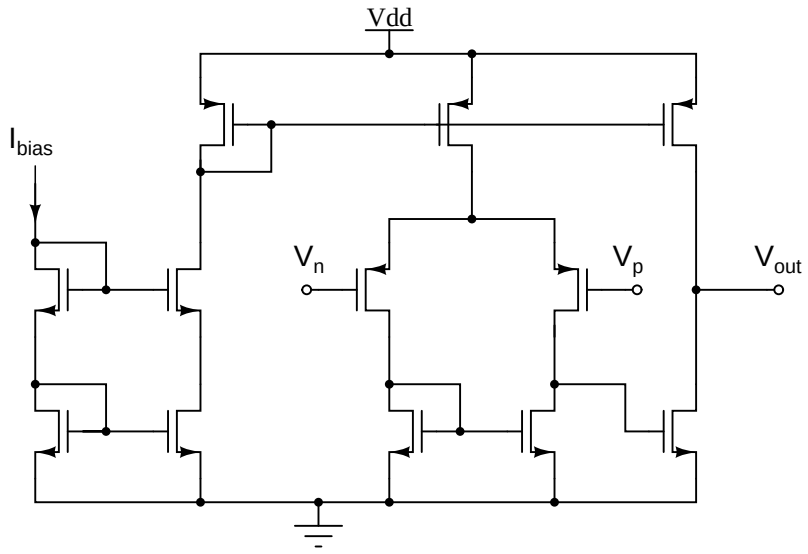


Figure 4.17: *Schematic drawing of the error amplifier used in the compensator of Figure 4.14.*

The small-signal circuit used to test the compensator with a full-circuit simulation of the error amplifier is shown in Figure 4.18. This circuit is similar to the circuit used to simulate the higher order small-signal transfer function of the buck converter (Figure 4.12), where the simple error amplifier has been replaced with our compensation network, and a circuit-level error amplifier. Note also that the compensator itself incorporates the resistive divider (comprising R_1 and R_4), to attenuate the output voltage. An AC analysis of this circuit was performed in Spectre/Cadence to ensure stability and good performance across the input voltage range (1.2-1.8 V), load range (0.2-2 W), and output voltage range (1-1.3 V). In this manner, the first-order compensation parameters calculated in Appendix D were fine-tuned to reach the final values, which are presented in Table 4.2.

Shown in Figure 4.19 is a Bode plot that shows the *simulated* transfer functions of the compensated and uncompensated systems (corresponding to Figure 4.18 and Figure 4.12, respectively), with the compensation network values of Table 4.2 and operating parameters of Table 4.1. We see that our compensation network provides a boost in phase and magnitude around the cross-over frequency (which is close to 5 MHz), and the new system phase margin is approximately 50 degrees. Additional time-domain simulation verified

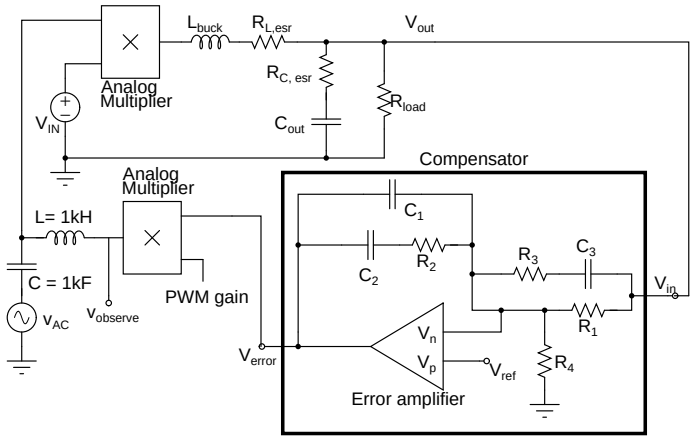


Figure 4.18: *Schematic drawing of circuit used to tune feedback compensation network, taking into account higher-order effects. The circuit schematic of the error amplifier is shown in Figure 4.17.*

Table 4.2: *Compensator Component Values*

Component	Value	Type
R_1	37.1 kΩ	P^+ Poly
R_2	32.4 kΩ	P^+ Poly
R_3	2.1 kΩ	P^+ Poly
R_4	148.3 kΩ	P^+ Poly
C_1	220 fF	Double Poly
C_2	3.88 pF	Double Poly
C_3	3.26 pF	Double Poly

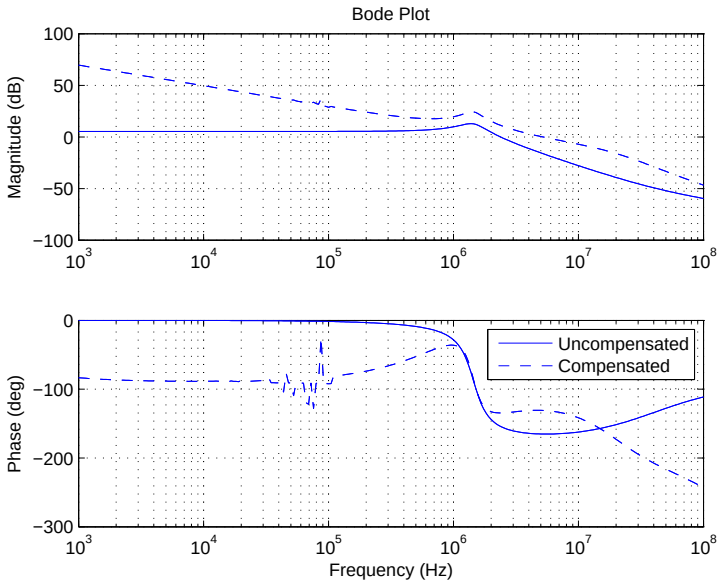


Figure 4.19: *Bode plot showing magnitude and phase of the* simulated *loop transfer function of the uncompensated system (Figure 4.12) and the compensated system (Figure 4.18). The compensation network provides a cross-over frequency of approximately 5 MHz, and a phase margin of 50 degrees.*

the stability and fast response of the feedback control implementation under a variety of conditions.

4.4.2 Feed-forward Control

A key enabler of the soft charging technique developed in this thesis is the frequency separation between the switched-capacitor transformation stage and the inductor-based regulation stage. Operating the regulation stage at a frequency that is much higher (>10x) than the transformation stage ensures that the capacitors in the the transformation stage experience soft charging and discharging. In this mode of operation, the regulation stage appears as a constant power load to the switched-capacitor circuit, and charges and discharges the capacitors with a controlled current.

A challenge associated with this mode of operation is the increased voltage ripple at the input of the regulation stage, as shown in Fig 4.2. Since the regulation stage operates at

a much higher switching frequency than the transformation stage, its control bandwidth can be made sufficiently high such that the input voltage appears as just a slowly changing voltage. The compensation network used to realize such a high control bandwidth was described in the previous section.

However, at the exact switching times of the transformation stage, the input voltage of the regulation stage changes abruptly. With a conventional feedback loop, this discontinuity in the input voltage will cause a corresponding change in output voltage (*audio susceptibility*) unless a very large output capacitor is used. This large step in input voltage can unfortunately not be adequately attenuated by a fast feedback loop alone, so we must seek alternative strategies to ensure that the large input step is not observed as an output voltage ripple. We choose to address this challenge using feed-forward control.

Shown in Figure 4.20 is the block diagram of the feed-forward control we use in this work. The feed-forward is implemented by having the gain of the PWM be inversely proportional to the input voltage (with an appropriate scaling factor). When there is an abrupt change in the input voltage (due to the transition of the SC stage), the controller is able to respond immediately since the gain of the PWM can change very rapidly, without affecting the stability of the feedback loop.

A circuit schematic of a high-level implementation of the feed-forward control is shown in Figure 4.21. In the case of the transformation stage control scheme we employ in this work (as seen in Figure 4.3), the buck converter input voltage *increases* in a step-wise manner. It is illustrative to see how the circuit of Figure 4.21 responds in this situation: With an abrupt increase in input voltage, the height of the triangle waveform input to the comparator increases as fast as the triangle waveform generator can respond. At this time, the compensator block has not noticeably changed its output voltage, since it has limited speed due to the need for guaranteed stability. The immediate impact is then that the comparator will output narrower pulses (smaller duty ratio D). This is indeed the exact behavior we desire, since for a buck converter, $V_{out} = DV_{in}$, and we wish to keep the output

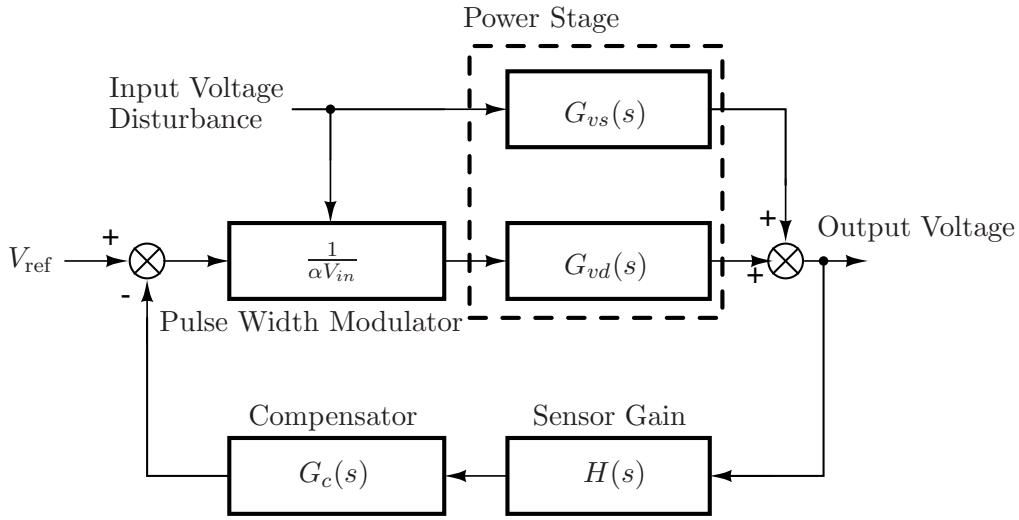


Figure 4.20: *Block diagram illustrating feed-forward control. The gain of the PWM block is inversely proportional to the input voltage, enabling cycle-by-cycle feed-forward control with fast response.*

voltage steady after an abrupt increase in input voltage.

A key circuit component of the feed-forward control technique is a circuit that can generate a fixed-frequency triangle waveform with an amplitude that is proportional to the converter input. Figure 4.23 shows the circuit used in this work to accomplish that task. The components in the amplitude control block set the height of the triangle waveform (proportional the input voltage), and the components in the frequency control block maintains a constant frequency (regardless of triangle waveform amplitude). In the amplitude control block, the input voltage is first converted to a current through the transconductance amplifier (shown in detail in Figure 4.22). This current is then fed as a bias current (I_{slope}) to the current-starved inverter of the adjustable slope generator, where it charges a capacitor. A larger bias current will make the slope of the generated triangle-waveform steeper. Figure 4.24 shows the circuit schematic of the adjustable slope generator used to provide the triangle waveform. The asymmetric buffer of Figure 4.21 sets the minimum voltage of the triangle waveform (set slightly higher than the minimum operating input voltage of the comparator of the feedback circuitry). When then triangle waveform reaches this value, the

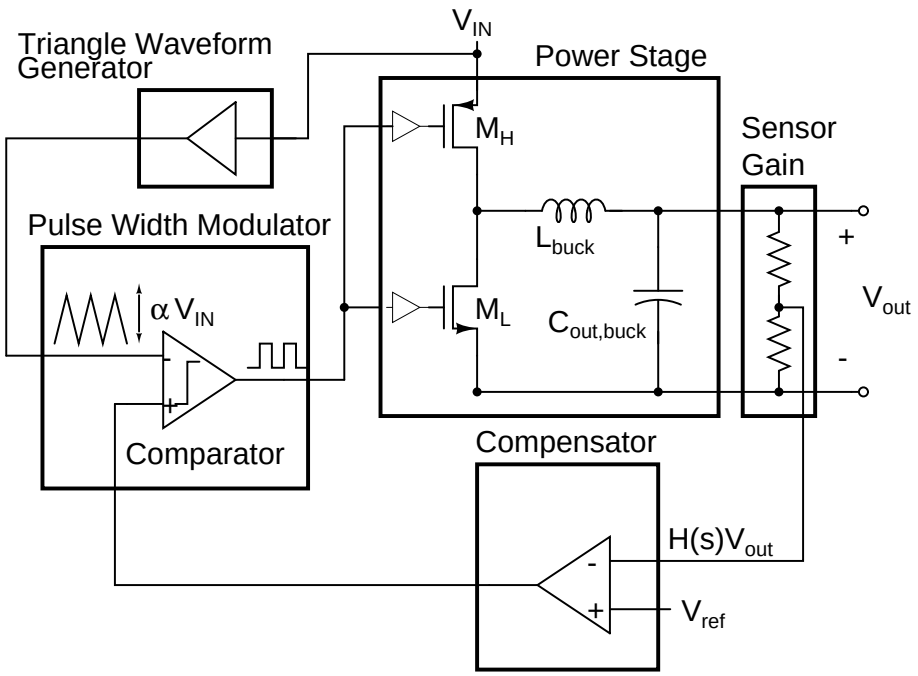


Figure 4.21: *Schematic drawing of a circuit implementation of the feed-forward combined with conventional feedback control illustrated in Figure 4.20.*

RS latch is set, and the *toggle* signal goes low, causing the adjustable slope generator to increase the triangle voltage. The Q output of the RS latch also starts the ramp generator in the frequency control block (shown in Figure 4.25). The ramp generator consists of a current-starved inverter that slowly charges a capacitor, and quickly discharges it when the input is toggled. A Schmitt trigger issues a timeout command when the ramp signal reaches the set value (1.4 V in this design). The timeout command in turn resets the RS latch in the amplitude block, causing the triangle waveform to change direction. In this manner, the frequency control blocks ensures that the triangle waveform is of constant frequency (set by the off-chip bias current I_{ramp}), and the amplitude block varies the height of the signal in proportion to the input voltage.

Shown in Figure 4.26 is a schematic drawing of all the pieces of the feedback and feed-forward control implementation for the regulation stage. The feedback loop keeps the output voltage steady during the (relatively) slow changes in input voltage caused by the

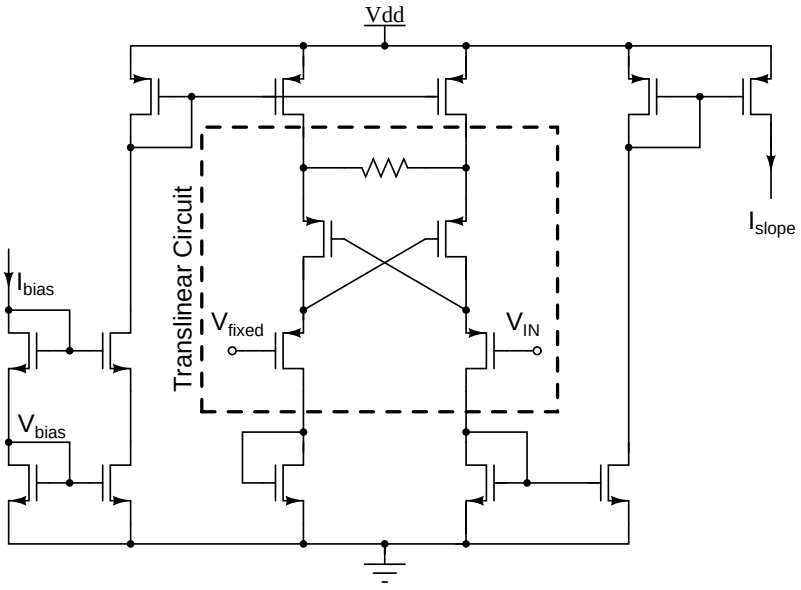


Figure 4.22: *Schematic drawing of the transconductance amplifier that converts the input voltage to a bias current. The circuit consists of a cascode input current mirror, and a translinear circuit that creates a differential current that is proportional to the differential voltage of the PMOS input transistors*

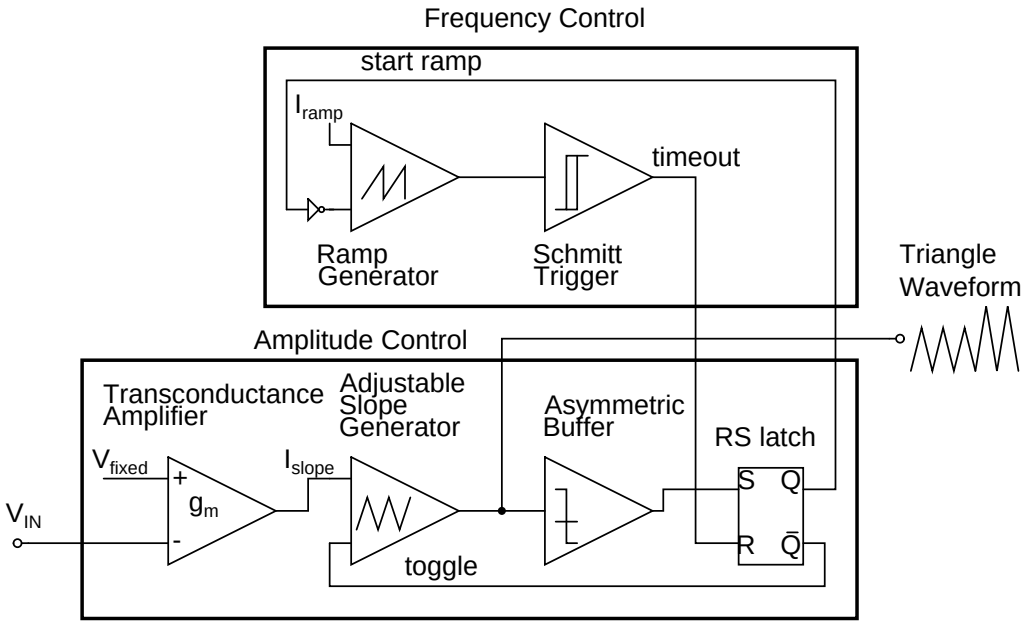


Figure 4.23: *High level schematic drawing of the components used to generate a fixed-frequency triangle waveform with an amplitude proportional to the buck converter input voltage.*

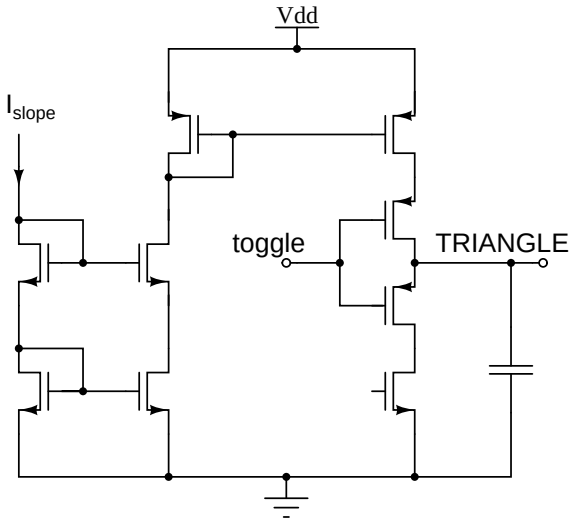


Figure 4.24: *Schematic drawing of the adjustable slope circuit block of Figure 4.23. The current starved inverter charges and discharges a capacitor, which generates a slope proportional to the bias current (I_{slope}).*

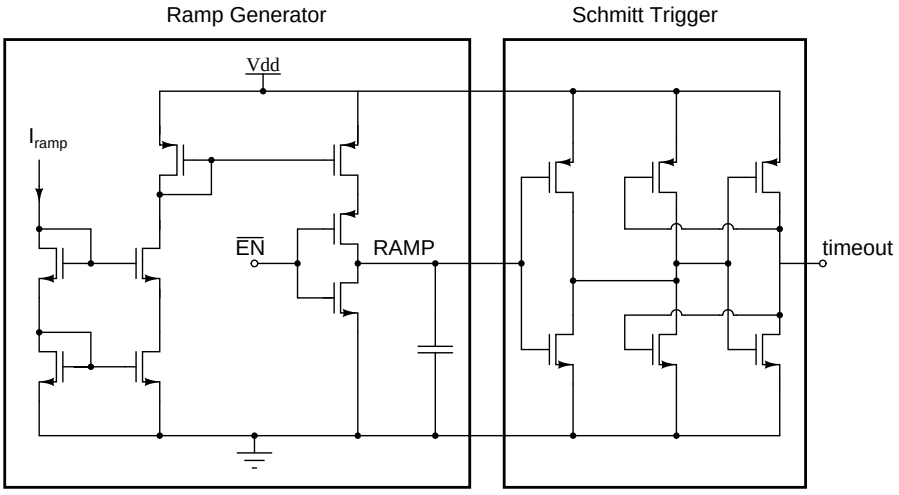


Figure 4.25: *Frequency control: A resettable ramp generator and a Schmitt trigger are used to control the switching frequency of the buck converter. The bias current I_{ramp} is is provided from off-chip, enabling a wide range of operating frequencies.*

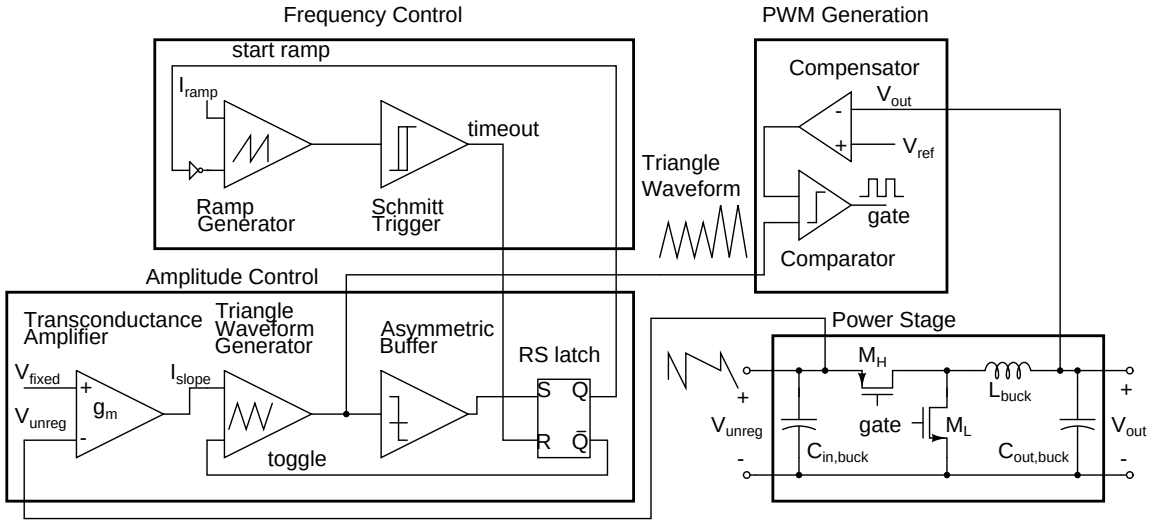


Figure 4.26: *Schematic drawing of feed-forward control of regulation stage to maintain steady output voltage despite the sharp transitions of the input voltage.*

capacitors discharging and charging in the SC stage. The feed-forward blocks ensures that the output voltage does not see large ripple even at the switch transitions of the SC stage (with corresponding step changes in the input voltage). It should be noted that the feedback loop can be made sufficiently slow to ensure stability, and that the only circuit block which requires very fast operation is the transconductance amplifier, which is implemented with a few fast low-voltage transistors.

The simulated waveforms of Figure 4.27 illustrate the operation of the feed-forward control. In this plot, the full circuit model of the buck converter (with operating parameters as given in Table 4.1, except that the input voltage is not fixed, but ramping as indicated in the figure) is simulated in Spectre/Cadence, with the feed-forward and feedback control enabled. The bias current to the ramp generator (I_{ramp} of Figure 4.26) is 30 μA, and the bias current to the transconductance amplifier (shown in Figure 4.22) is 5 μA.

In the simulation the input voltage is shaped like a ramp with sharp edges (just as would be expected if the input of the buck converter is connected to an SC transformation stage, as outlined previously). Also plotted in Figure 4.27 is the generated triangle waveform

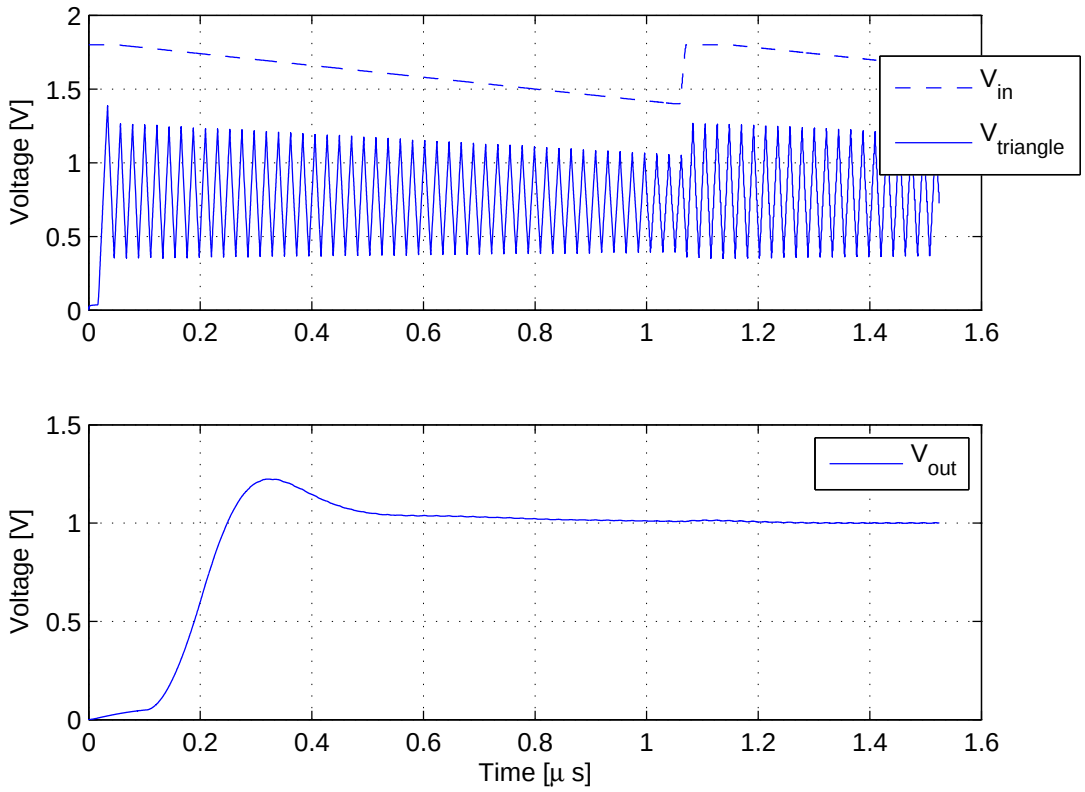


Figure 4.27: *Simulated waveforms showing the performance of the feed-forward and feedback control circuitry. The output remains at a steady 1 V despite large discontinuities in the input voltage*

($V_{triangle}$) that is an input to the comparator. It can be seen that the height of the triangle is proportional to the input voltage, and responds quickly to changes in input voltage. After a brief start-up transient, the output voltage settles to a steady 1 V output, despite a large input voltage ripple.

4.5 Regulation Stage Power Stage

Shown in Figure 4.28 is a schematic drawing of the power stage of the regulating converter, with associated power devices, tapered gate drives, and a non-overlapping clock generator.

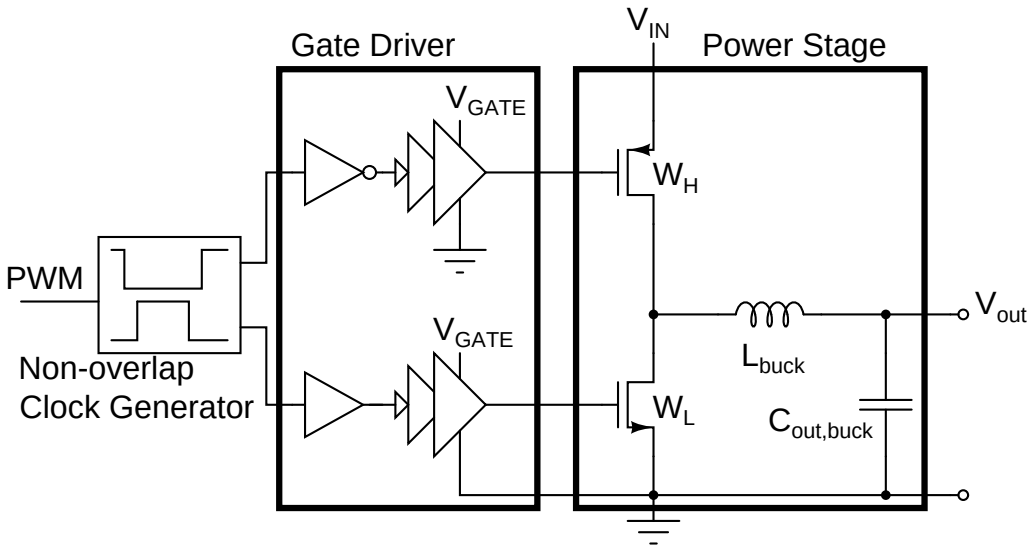


Figure 4.28: *Schematic drawing of power stage of the regulating converter. External component values are listed in Table 4.5.*

Appendix C provides a detailed description of the process of appropriately sizing the power devices (W_H and W_L). In our implementation the high-side PMOS device has a total device width $W_H = 28$ mm, and the low-side device has a width $W_L = 9.8$ mm. Both devices were implemented in the core low-voltage technology with a gate length of 180 nm.

The tapered gate drive, shown in Figure 4.29 comprises six stages with a tapering factor of 9, which represents a good balance between gate drive loss and device switching loss, as determined by simulation. The gate drive circuitry was powered from a separate low-voltage supply (1.8 V) in order to accurately measure the gate drive loss and its contribution to converter efficiency. While the gate drive circuitry can be powered directly from the input of the buck regulation stage, the large voltage swing at the input would result in a time-varying drive voltage for the power transistors, with resulting changes in efficiency. The non-overlapping clock generator of Figure 4.28 is the same programmable unit as was used in the switched capacitor stage (shown in Figure 4.5).

Table 4.3 provides a listing of the value and type of on-die capacitance that was added. The table only lists additional decoupling capacitance that was added as filler where space

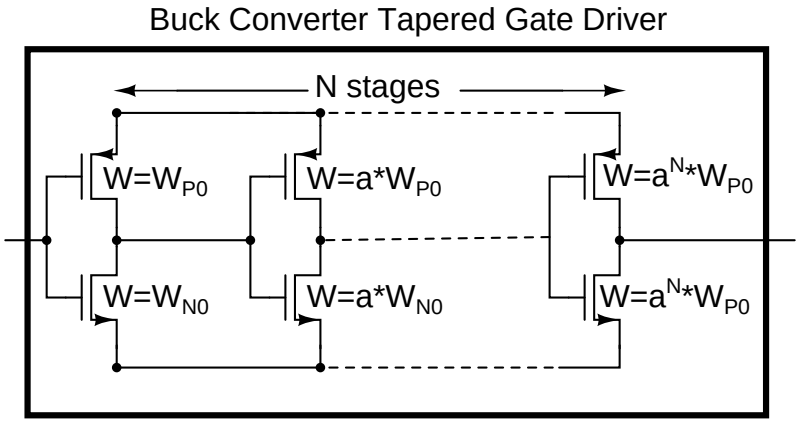


Figure 4.29: *Schematic drawing of the tapered gate driver for the buck converter. The tapering factor a is 9, and the total number of stages (N) is 6.*

Table 4.3: *On-die Decoupling Capacitance Values*

Component	Value	Type
$C_{gate,SC}$	280 fF	5V MOSCAP
$C_{gate,buck}$	840 fF	2V MOSCAP
C_{analog}	840 fF	2V MOSCAP
$C_{in,buck}$	840 fF	2V MOSCAP
$C_{gate,buck}$	840 fF	2V MOSCAP

allowed it, and does not include capacitance from other devices connected to the same node within the circuit.

4.6 Experimental Results

Shown in Figure 4.30 is a die-photo of a prototype merged two-stage converter fabricated in National Semiconductor's CMO9T5V 180 nm CMOS process, with 6 metal layers. Because of packaging restrictions, the die was connected through bond-wire (with considerably higher resistance than a flip-chip packaging method) to the LLP40 5x5 mm package. 1.3 mil gold bond-wire was used, with all pins processing power double-bonded to decrease the parasitic resistance. The layout was optimized to minimize bond-wire and on-chip metallization

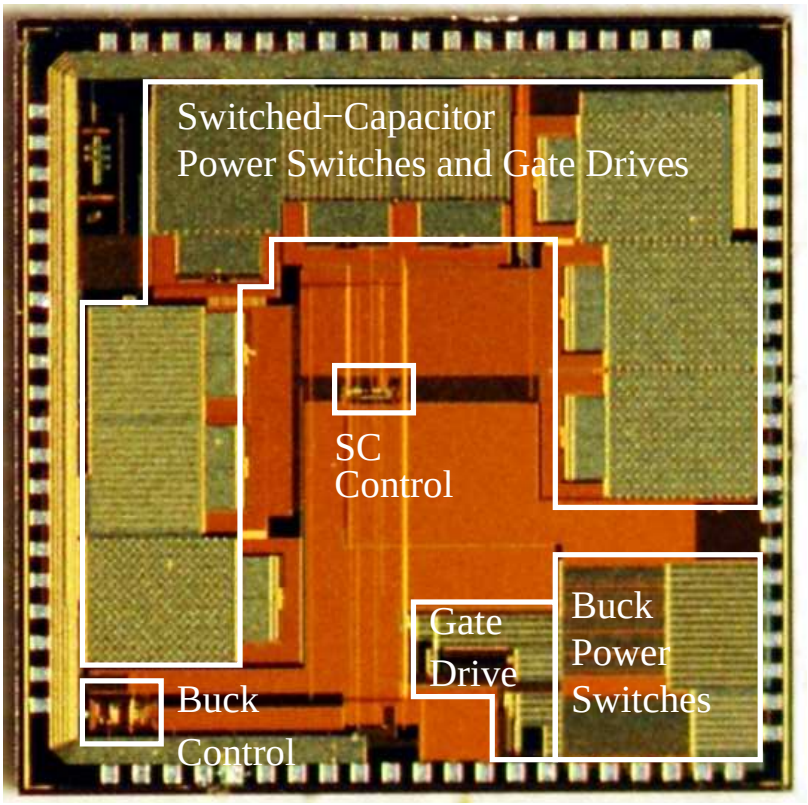


Figure 4.30: *Die photograph of a soft charging converter implemented in 180 nm CMOS technology. The total die area is 5x5 mm (not optimized for space).*

resistance as much as possible, which lead to an overall design with larger area than what was actually required by the power devices. As can be seen from the die-photo, all power devices are placed at the perimeter of the chip, and placed as to minimize interconnect distances.

The chip was mounted on a test PCB, as shown in Figure 4.31. In addition to the passive off-chip components directly required by the merged two-stage converter, the test PCB also contains a micro-controller (ATtiny861) that writes serial data to the chip for setting parameter values, as well as enable/disable select parts. A Python script on a lab bench computer communicates over serial interface with the micro-controller. The test PCB also contains tuning potentiometers for setting reference voltages, as well as floating current sources for biasing purposes.

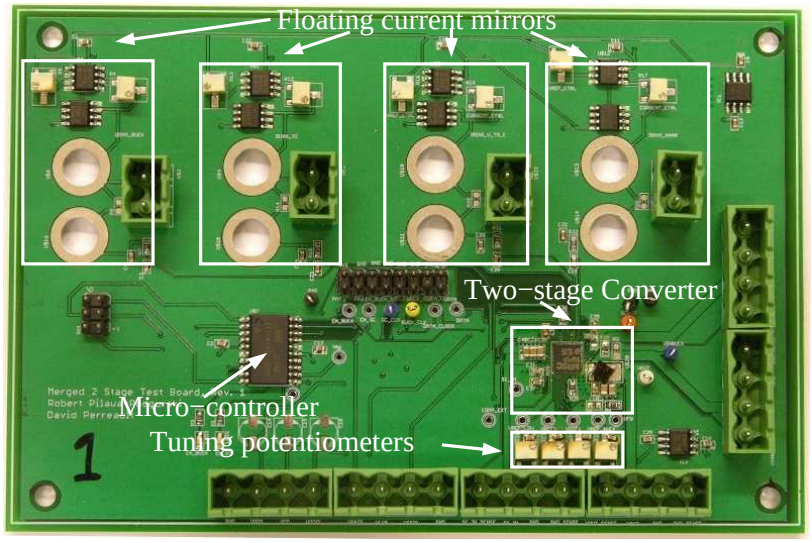


Figure 4.31: *Photograph of test PCB with bias current sources, reference voltages, and a micro-controller to write parameter settings to the chip.*

Figure 4.32 shows a zoomed-in photograph of the merged two-stage test chip, as well as the passive components. The SC stage external capacitors (C_1 and C_2) are placed on the bottom side of the PCB to minimize the loop area.

Despite our best efforts to keep packaging losses to a minimum, upon testing it was discovered that bond-wire resistance and on-die metallization resistance were significantly higher than anticipated. This has a particularly detrimental effect on efficiency at high output powers, where the ohmic losses dominate. For this reason, the output level at which we run our converter (up to 0.8 W) is lower than the 2 W that it was designed to handle. Furthermore, although the buck regulator stage was designed to operate at 30 MHz, the new lower power level required a decrease in switching frequency to 10 MHz to maintain satisfactory efficiency. This was due to the fact that the power switches in the buck regulator were sized for 2 W max output load, and at the lower output power the gating loss became a dominant loss that decreased efficiency. It should be mentioned that the buck regulator works at a switching frequency up to 30 MHz, at the expense of efficiency. For the experimental results shown in this work, the operation and characterization was done at a regulation stage frequency of 10 MHz.

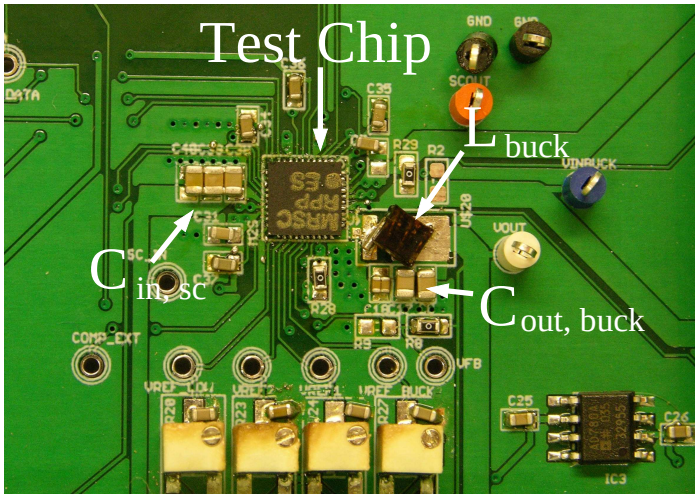


Figure 4.32: *Photograph of merged two-stage test chip mounted on PCB, along withi top-side passive components. Some capacitors were placed on the bottom side to minimize inductance.*

Table 4.4: *Converter Specifications*

Input Voltage Range	4.5-5 V
Output Voltage Range	1-1.3 V
Output Power Range	0.3-0.8 W
SC Switching Frequency	2-100 kHz (load dependent)
Buck Switching Frequency	10 MHz
Peak Efficiency	81%

Shown in Table 4.4 is a table that summarizes the operation region and performance of the experimental prototype. The efficiency includes all control and gating losses of the two converters.

Shown in Fig. 4.33 are experimental waveforms that illustrate the performance of the feed-forward control circuitry. It can be seen that despite large voltage swings at the input of the buck converter (>500 mV step of V_{unreg}), the output voltage remains stable (<50 mV ripple). It should be noted that this was accomplished without a large capacitor on the output of the buck regulator as can be seen in Table 4.5, which lists the external components used in the experimental prototype. The good attenuation of the input ripple can be attributed to the feed-forward control, which works well. For this measurement, the

Table 4.5: *External Component Values*

Component	Value	Type
$C_{in,SC}$	3 x 10 μF	X5R, 0603
C_1	22 μF	X5R, 0603
C_2	22 μF	X5R, 0603
$C_{in,buck}$	2 μF	X5R, 0402
$C_{out,buck}$	4.7 μF	X5R, 0402
L_{buck}	28 nH	Air core, Coilcraft B08T

input current to the ramp generator (I_{ramp}, which controls frequency), was set to 13.8 μA, and the bias current of the transconductance amplifier (I_{bias} of Figure 4.22) was 0.7 μA.

The performance of the converter was also evaluated during a load-step, as shown in Figure 4.34. Here the load was stepped repeatedly between 10% and 90% of full load, using a switchable external resistor load. This type of load behavior is possible when electronic circuit go in and out of sleep mode, for instance. It can be seen from the waveform that the control implementation maintains the output voltage at the desired operating point, despite both load steps and large buck converter input voltage. The light-load operation of the SC stage is also apparent in this plot, where the hysteretic controller increases the switching frequency of the SC stage at heavy load, and reduces it at light load (leading to lower loss at light load).

Measured efficiency for a few different output voltages are shown in Fig. 4.35. The efficiency measurements include all power losses associated with the control circuitry, as well as gating losses and all packaging and bond-wire losses. The decrease in efficiency at low input power is almost entirely due to the regulation stage, which was operated at a fixed frequency (10 MHz) at all times. Efficiency at low power levels can be increased with suitable light-load control schemes such as pulse-frequency modulation (PFM), if desired. The SC stage is inherently light-load efficient due to the hysteretic controller, which automatically operates at a lower switching frequency at low output power.

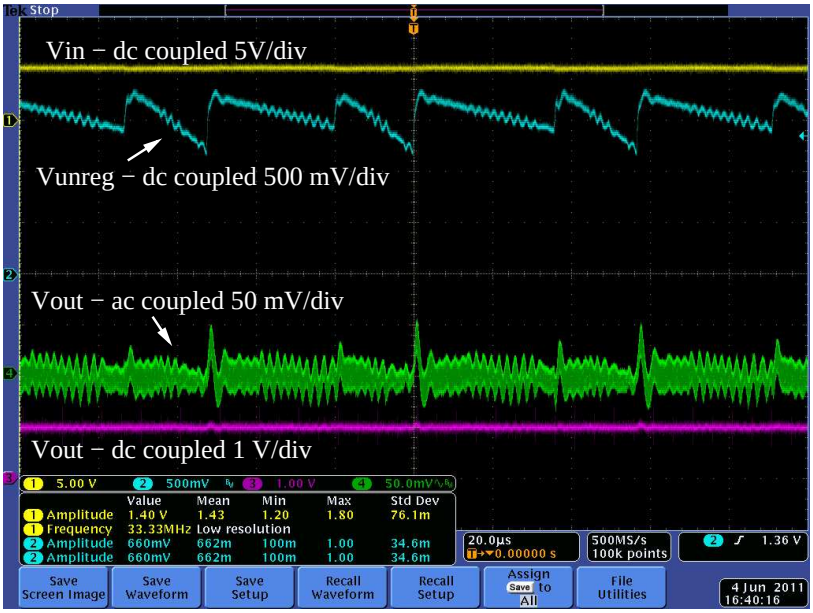


Figure 4.33: *Experimental waveforms showing converter operation. Note that the input voltage is 4.5 V, and the output voltage is steady at 1 V, despite the large voltage swings at the input of the buck converter (V_{unreg}).*

Table 4.6 shows an estimated breakdown of losses. A significant portion of the losses come from bond-wire resistance and on-chip metallization resistance, owing to the package used. There are well-known techniques to mitigate these losses (e.g thick top layer metallization, flip-chip technology). It is therefore expected that the overall converter efficiency can be significantly improved through appropriate packaging techniques.

Shown in Fig 4.36 is the measured efficiency of the merged two-stage converter together with *modelled* efficiency for a single-stage buck converter operating at 10 MHz. The single-stage buck converter is modelled with the same 5 V devices (and attendant packaging losses) that were used in the SC stage, and provides a benchmark for comparison. It should be noted that only gate drive and conduction losses (including packaging) were modelled, and that an experimental implementation would likely see an even lower measured efficiency than what is shown in Fig 4.36, owing to additional control and switching losses. Moreover, while a switching frequency of 10 MHz is approaching the practical limit of a single-stage 5-to-1 V buck converter, the merged two-stage converter can be operated at

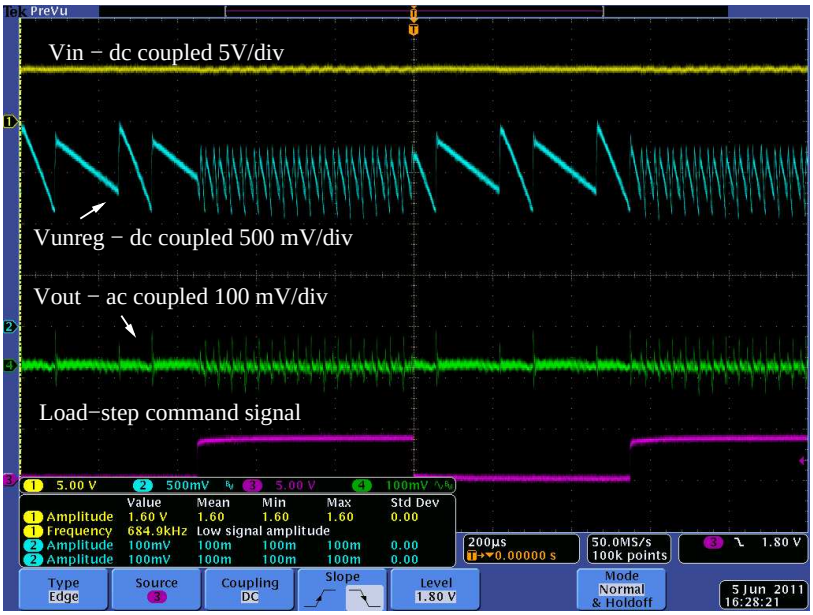


Figure 4.34: *Experimental waveforms showing converter operation performance during a load step between 10 and 90% of full load. The output voltage is steady, and the light-load behavior of the SC stage can be observed.*

even higher switching frequencies without difficulty, owing to its use of low-voltage devices in the regulation stage. The more important result, however, is that compared to a single-stage topology, the two-stage architecture that we have presented scales well to significantly higher switching frequencies than what was demonstrated here. Consequently, we expect the benefits in terms of size and efficiency of our proposed architecture to be even more apparent as higher switching frequencies are pursued.

Table 4.6: *Estimated Converter Loss Breakdown at* P_{out}*=0.8 W*

Bond-wire conduction loss	60 mW
Transistor gating loss	45 mW
On-die metallization conduction loss	40 mW
Transistor conduction loss	11 mW
Inductor loss	5 mW
Control losses	2 mW

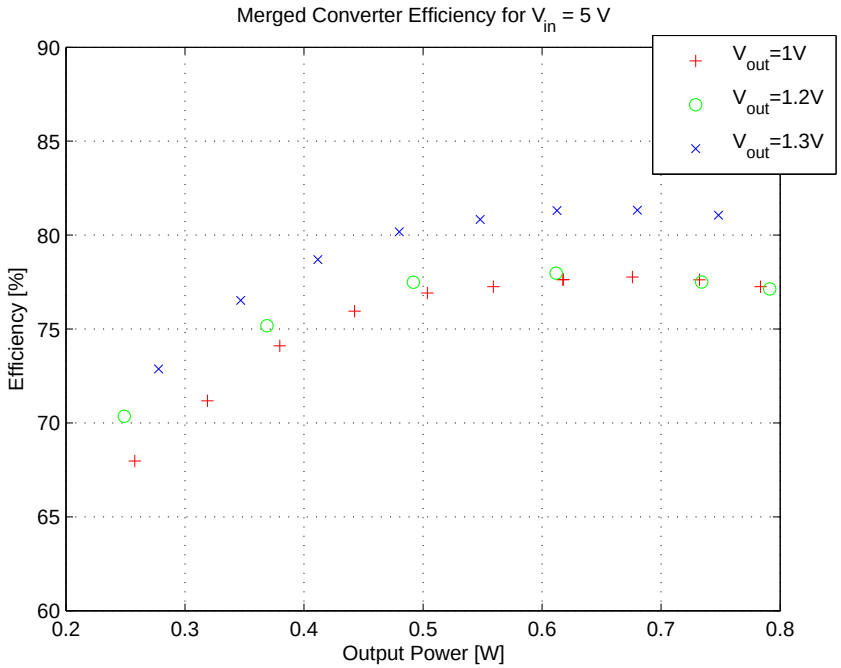


Figure 4.35: *Plot showing measured efficiency for the prototype merged two-stage converter across output power range. All control and gate drive losses are included in the efficiency measurement.*

4.7 Conclusions

We have presented a new power converter architecture that is suitable for large voltage step-down applications, where efficiency and size are important. The merged two-stage architecture makes use of the available CMOS device characteristics to offer both large voltage step-down and high frequency operation on a single die. Furthermore, we have illustrated that by properly merging the slow SC transformation stage with the fast synchronous buck regulation stage, we can achieve an improvement in energy density and/or efficiency of the SC stage through *soft charging* operation. In this mode of operation, the SC stage capacitors can operate at large voltage ripple without increased loss. This powerful technique does require some more advanced control techniques, and we have highlighted how this can be implemented in a 180 nm CMOS process

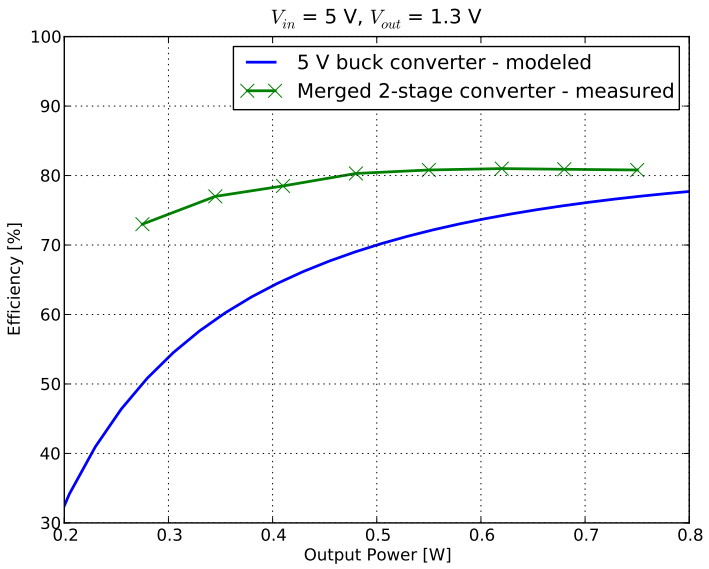


Figure 4.36: *Plot showing experimentally measured efficiency for the prototype merged two-stage converter compared to modelled efficiency of a single-stage 5-to-1 V buck converter using transistors from the same process.*

4.8 Future Work

The merged two-stage architecture is an entirely new kind of converter, and there are several areas where the work we have presented here can be improved.

4.8.1 Packaging

As described in section 4.6, a limiting factor in our CMOS prototype was packaging. Since we only had a bond-wire package available to us, a majority of our losses came from bond-wire resistance and metallization resistance. Much of these losses can be mitigated by the use of more advanced packaging techniques, such as flip-chip packages and solder bumps. Together with an interposer board, much of the off-chip interconnect resistance can be reduced, together with even higher packaging densities. We anticipate that the benefits of the merged two-stage architecture will be even more pronounced with the appropriate pack-

aging techniques. Aside from the challenge associated with implementing more advanced packaging techniques, it would require work in the area of optimum power device layout, as the vertical structure of solder bumps changes the direction of the current flows, and thus the optimum geometry of the power transistors.

4.8.2 On-chip Passive Components

A large part of the benefit of the merged two-stage converter is the ability to drastically reduce the size of the passive components, through higher frequency operation, and better energy utilization of the capacitors. A natural step forward is thus to integrate both the capacitors and inductors on die. While this removes many interconnect issues (in particular for large step-down ratio SC converters, which have many capacitors), a challenge is getting high enough energy density integrated capacitors, and on-die inductors with high quality factors. For lower power applications, the merged two-stage converter shows promise for achieving complete integration of the active and passive components on a single die, but much work remains to be done to realize this potential.

4.8.3 Improved Control

While the feed-forward technique presented in this thesis showed good performance, there are alternative ways to achieve the same results. One natural extension of this work is to implement the buck regular using current-mode control, which naturally offers the benefit of fast response to changes in input voltage. Current-mode control may in fact be more simple than the feed-forward technique presented here, and it would be worth pursuing in future research.

An advantage of the merged-two stage converter that we did not make use of in this work is the fact that the large voltage discontinuities on the input of the regulation stage are not caused by an external source, but by our own SC stage controller. It is therefore possible to

implement a buck regulation stage that has *a priori* knowledge about the sudden change in input voltage, such that it can begin to change its mode of operation in anticipation of this event. This can also be coupled with a digital controller, which can self-tune to achieve the appropriate response to minimize output voltage ripple.

4.8.4 Alternative SC Topologies

In this work we used a series-parallel switched-capacitor topology, which lends itself naturally to the soft charging technique. There are, however, many other possible switched-capacitor topologies that can also benefit from soft charging, and may in fact be more suitable for integration. This is a research area that could benefit from a theoretical survey and analysis of how soft charging can be implemented in other SC topologies.

Chapter 5

Thermophotovoltaic Power Generation

As part of this thesis, applications of low-voltage integrated power electronics are explored. One area where power electronics can provide substantial improvements in system performance is that of energy harvesting. In most practical energy harvesting systems, the characteristics of the energy source are distinctly different from those of the electric load. A common electric load in these system is an electric circuit (analog or digital, or both), which typically requires a well-behaved dc voltage level. Another common scenario is the use of temporary energy buffer at the output of the energy harvester, such as a battery or an ultracapacitor. These energy buffer operate at specific dc voltage levels, which most often do not match the characteristics of the energy harvester. For instance, in piezo-electric and many MEMS-based energy harvesters, the energy source often produces ac voltage and current waveforms, which need to be converted to a dc voltage of the correct value. Many other energy harvesters such as PV, TPV, and thermoelectric converters generate dc voltages and currents that are at much different levels than what is desired by the load. Moreover, most energy harvester systems require operation at a particular voltage and current level to generate maximum power, so it is desirable to employ intelligent electronics to ensure that the system operates at this point at all times, thereby extracting the most energy from the system.

In this section, a low-voltage, low-power maximum power point tracking dc-dc converter is presented that is intended to interface a portable thermophotovoltaic power generator with its load. The system described performs voltage conversion to a level more suitable for the load, and provides intelligent tracking of the most desirable operating point, ensuring

that all available energy is extracted from the power generator. It should be noted that the methods and components introduced in this section are suitable for a variety of other low-voltage energy harvesting applications in addition to the thermophotovoltaic application.

5.1 System Overview

The possibility of statically converting heat into electricity–without moving parts–has captured the imagination of scientists and engineers for nearly two centuries. Since the discovery of the thermoelectric, photovoltaic and thermionic effects, there have been significant efforts towards developing devices that can perform this conversion with good efficiencies. One of the promising technologies to convert heat (more precisely radiant heat) into electricity is thermophotovoltaics (TPV). TPV converts heat into thermal radiation photons that are in turn converted into electron current via the photovoltaic effect, as shown in the inset of Figure 5.1. While TPV power conversion is in many aspects similar to solar photovoltaics (PV), there are several key differences. The TPV emitter typically operates at temperatures between 1100K-1500K, and hence the peak of the radiated spectrum is shifted towards longer wavelengths. This is illustrated in Figure 5.1 which shows spectral irradiance of a blackbody emitter at 1100K that peaks around 2.6 μm; this is in stark contrast with the solar spectrum, which peaks around 480 nm. Indeed, TPV requires low-bandgap PV diodes such that the bandgap is better matched to the peak infrared (IR) radiation, since only photons with energies above the PV diode bandgap can generate electron-hole pairs, as represented by shaded area under the blackbody curve in Figure 5.1. Furthermore, a TPV thermal emitter and TPV diode are in close proximity, thereby enabling photon recycling; a process where photons reflected from the TPV diode can be reabsorbed by the emitter. Due to the close proximity, TPV cells operate at more than two orders of magnitude higher energy densities than solar PV (as shown in Figure 5.1). However, the TPV cells are exposed to spatially non-uniform incident photon flux, which can be challenging from a system design perspective and which motivates the distributed power conversion architecture utilized

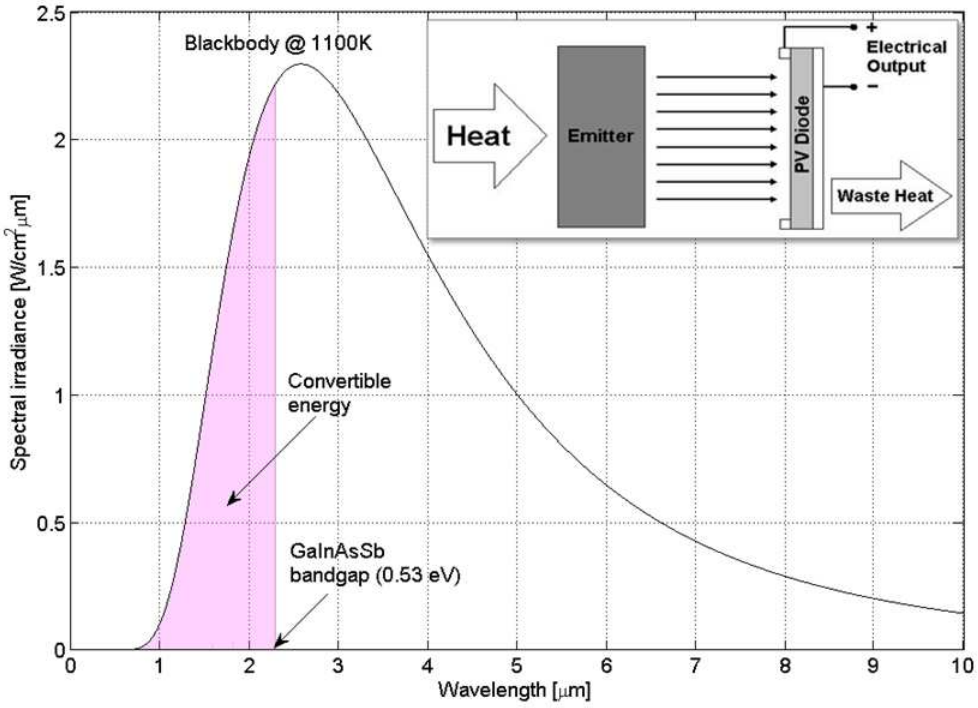


Figure 5.1: *Radiated spectral power distribution of a blackbody emitter at 1100K. Inset shows a block diagram of a thermophotovoltaic energy conversion process. Image courtesy of Ivan Celanovic.*

here.

The TPV concept was first proposed in the 1950s [30]. However, high-efficiency operation has only recently been enabled through scientific and technological advancements in two critical areas: low-bandgap semiconductor materials, and photonic crystals. High-performance low-bandgap semiconductor diodes such as GaInAsSb enable quantum efficiencies approaching unity for a wavelength range between 1 and 2.3 μm. The addition of photonic crystals (PhC) allows for spectral shaping of the thermal radiation so that its spectrum is almost perfectly matched to the diode electronic bandgap [31, 32]. These two technologies combined have brought TPV to the forefront of portable power generation, demonstrating above 20% efficiency in converting radiative heat into electricity [33]. With new PhC designs and

optimized TPV diodes 30% conversion efficiency is within reach.

In this work we focus on the low-power, micro-fabricated, butane powered TPV generator, as shown in Figure 5.2. It comprises a silicon micro-fabricated fuel reactor that acts as a radiant heat source [34], low-bandgap GaInAsSb PV diodes [35], and a low-power power electronics module. The key advantages of the TPV technology for micro-scale power generation are: high energy density, no moving parts, robust multi-fuel operation, and high efficiency. High energy density stems from the energy density of butane, which is almost two orders of magnitude higher than current Li-ion batteries.

Although significant headway has been made on the device level there have been very few attempts at complete TPV system level demonstrations. One of the critical components in a fully integrated micro-TPV system is the low-power power electronics converter. This work, to the best of our knowledge, is the first systematic and rigorous treatment of the design, optimization, and testing of a low-power maximum power point tracking (MPPT) converter for a TPV power generator system. To this end, we describe the power electronics subsystem for the TPV system of Figure 5.2, address some unique challenges associated with this application, and outline the solutions implemented to achieve a high performance overall system. Although our focus is on a micro-fabricated TPV generator, this approach is applicable to other TPV systems such as radioisotope powered TPV, and solar-TPV (where concentrated sun-light heats an element which re-radiates at longer wavelengths).

5.2 TPV Cell Characteristics

Shown in Figure 5.3 is the I-V characteristic for one TPV module, which consists of four series-connected GaInAsSb PV diodes [35]. The bottom graph of the figure shows the corresponding power versus voltage graph, which clearly shows a maximum power point (MPP) at approximately 0.85 V for this example. This point typically changes with operating conditions such as incident irradiation and cell junction temperature, and must therefore be

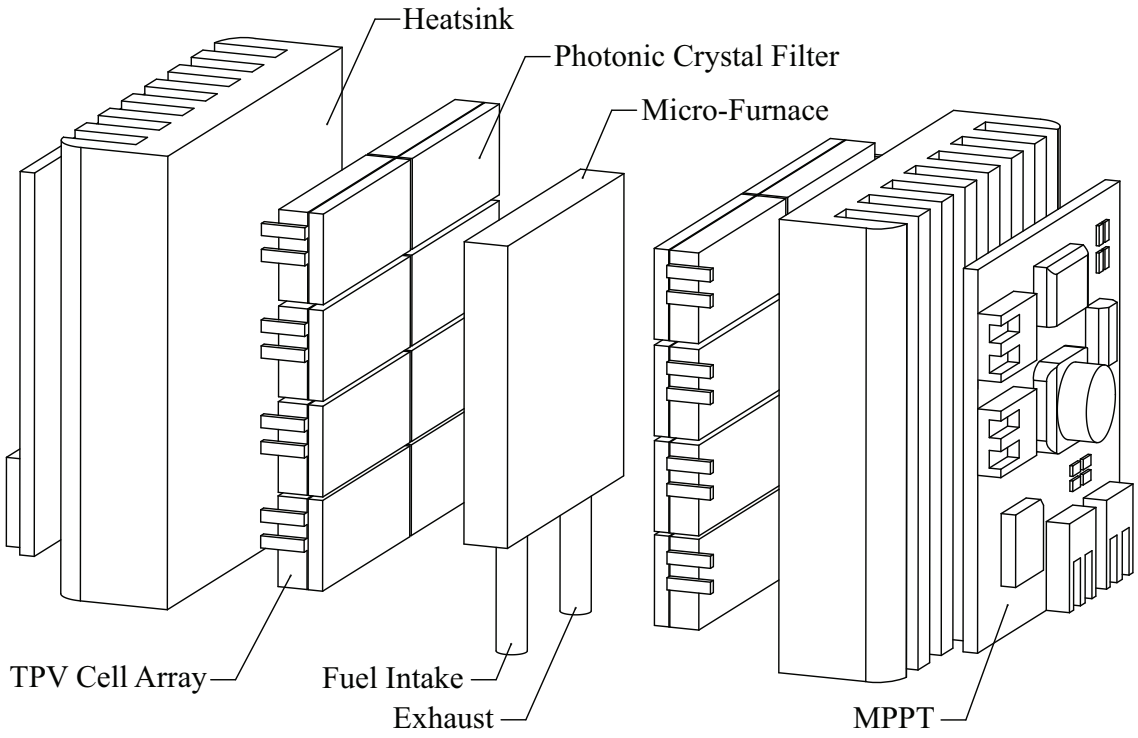


Figure 5.2: *Illustrative drawing of burner and TPV cells for portable power generation. Image courtesy of Nathan Pallo.*

continuously tracked to ensure that the maximum power is extracted from the cell. Shown in Figure 5.4 is a schematic drawing of the circuit model of the TPV cell. The photo current is modeled as a current source (I_{PH}), the P-N junction is represented by a diode (with ideality factor n and junction voltage scaled to fit the empirical data of Figure 5.3). The two resistors R_S and R_P represent series interconnect loss and leakage, respectively.

Figures 5.5a and 5.5b illustrate two common methods to connect photovoltaic cells to their loads. In Figure 5.5a all the cells are connected in series, and are directly connected to the load, a battery in this example. A diode is typically placed in series with the cells to prevent the battery from discharging through the cells during low light conditions. This approach, while simple, is typically very inefficient. Ignoring the small voltage drop across the diode, the string voltage V_{string} is restricted to be equal to the battery voltage V_{out} at all times, which is typically not the same as the MPP voltage (V_{MPP}). For a particular

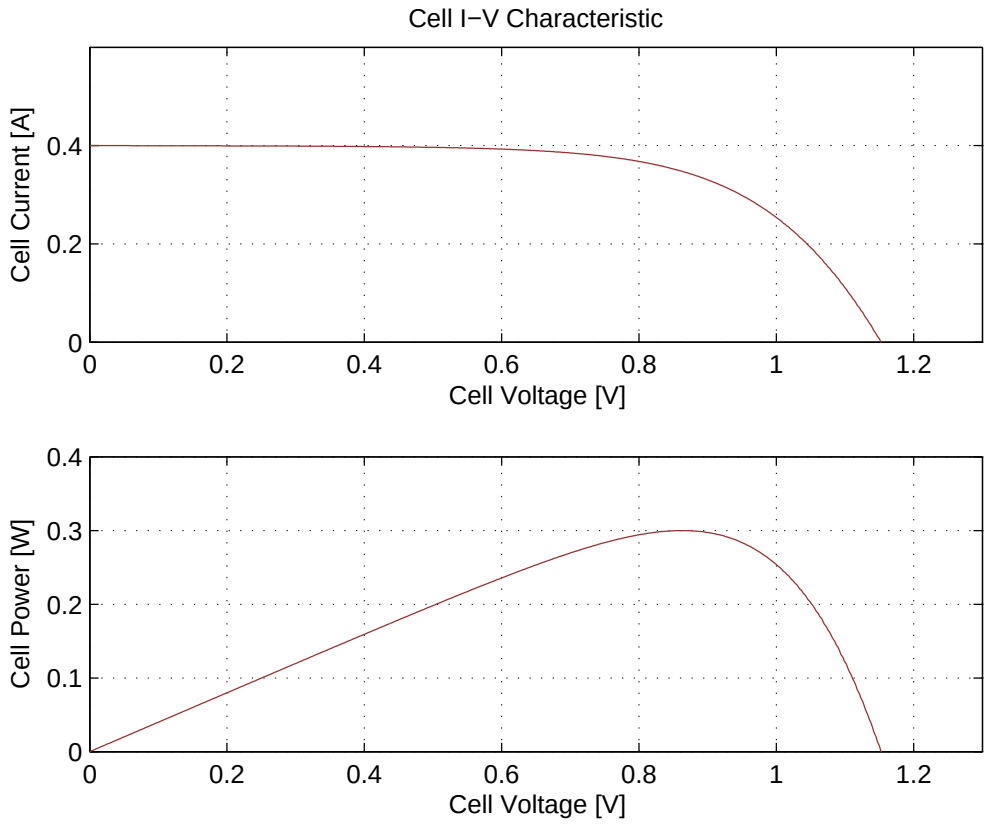


Figure 5.3: *I-V (top) and P-V (bottom) characteristic of TPV cell used in this work for a typical operating point.*

operating irradiation level and temperature, the series-connected cells' V_{MPP} may coincide with V_{out}, but at all other times, less than the maximum power is extracted from the cells. Figure 5.5b shows a method which is typically used to circumvent this limitation. By placing a dc-dc converter between the series-connected cells and the load, the string voltage V_{string} can be controlled to equal V_{MPP} at all times. The dc-dc converter, acting as a maximum power point tracker (MPPT), continuously tracks V_{MPP} by adjusting its conversion ratio in response to changes in operating conditions.

The method of Figure 5.5b is often adequate for solar photovoltaic applications, where the solar irradiation is a plane-wave, ensuring uniform illumination of all cells in the series

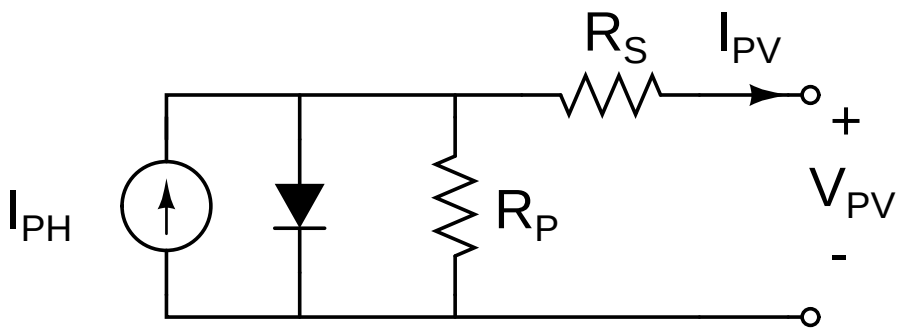


Figure 5.4: *Schematic drawing of the typical circuit model of a TPV cell.*

string. Provided the cells are properly matched in terms of their electrical characteristics, they will then produce equal currents. The situation is different in the TPV application considered here. Since the burner is positioned close to the TPV diode (2-3 millimeter separation), the irradiation is non-uniform and depends on the relative position of the diode with respect to the burner. In addition, the temperature distribution across the burner surface is non-uniform, and resonant cavity effects and reflections furthermore distort the uniformity of irradiation. This leads to mismatched cell photocurrents, with the cell receiving the most irradiation producing the most current. If a method similar to that of Figure 5.5b is employed in this situation, the string current I_{string} is limited to the value of the least irradiated cell. Thus, all other cells are operating at a cell current that is below their peak current, resulting in a total output power that can be substantially lower than the maximum achievable. The result is similar to that observed in solar panels with partial shading, as discussed in [36, 37]. The non-uniform irradiation in this application prevents efficient energy extraction with the stacking of many cells in series to achieve a high output voltage. In a stacked system, it is expected that that resulting mismatch would result in power reduction between 10 and 50%.

Figure 5.5c shows the architecture we propose to ameliorate these concerns. In this architecture, four diodes are connected in series and form a module. Each module is then connected to its own individual MPPT, and the outputs of all MPPTs are connected in parallel. The choice of four cells per module was made to provide a large enough working

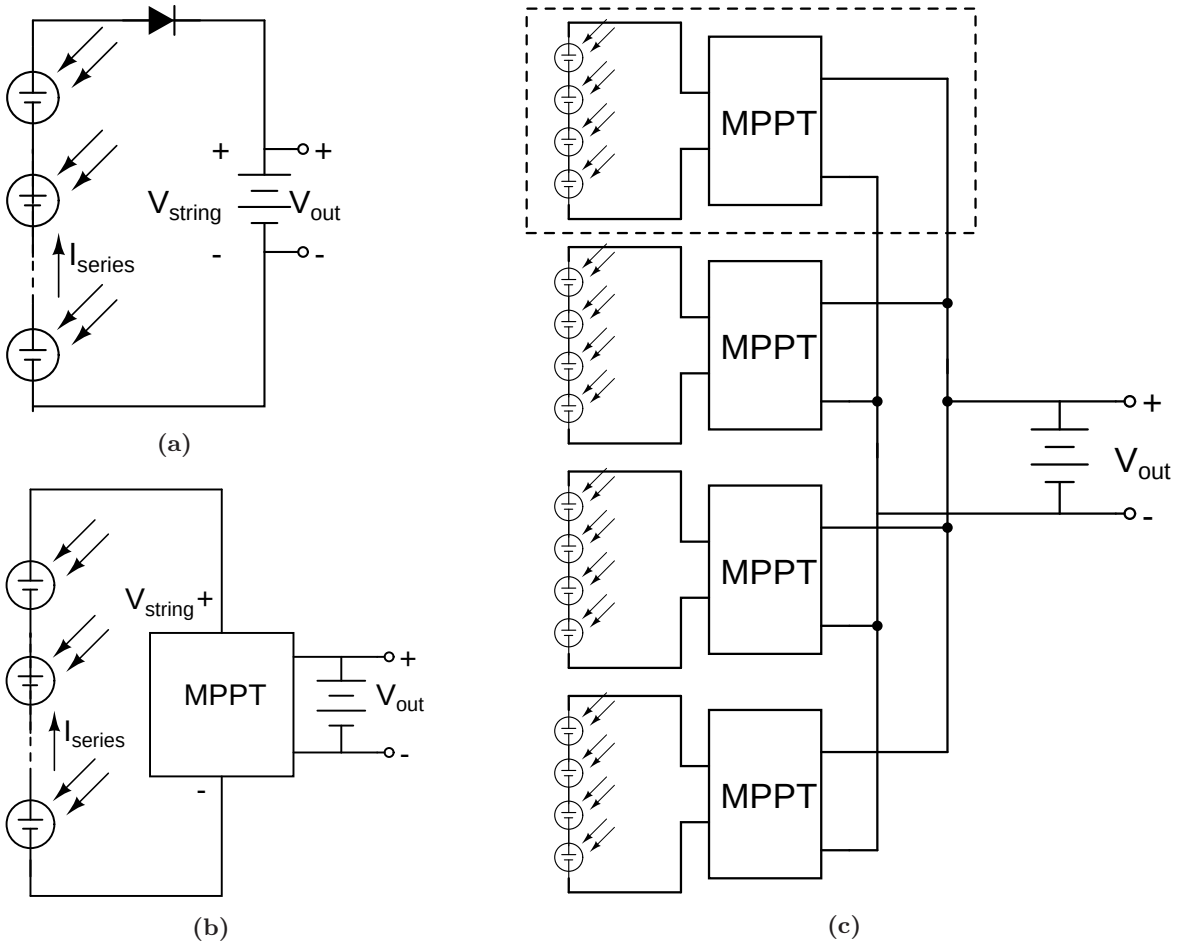


Figure 5.5: *(a) Simple cell connection, which does not extract the maximum power from the cell. (b) Conventional method with series-connected cells attached to MPPT. (c) Multi-MPPT method employed in this work.*

voltage (approximately 1 V) for the MPPTs to ensure efficient power conversion by the electronics. Using this architecture, current mismatch is limited to only four cells, all of which are placed in close proximity to each other, thereby minimizing the negative effects of non-uniform irradiation. The boxed area of Figure 5.5c highlights the system components that are considered in this work, which constitute four series-connected cells and one MPPT.

In the next two chapters, we will describe two implementations of low-power MPPTs for TPV energy harvesting, which work well with the architecture described in Figure 5.5c. While the specific application in both cases is the TPV system described in this chapter, many of the low-power techniques and analysis is applicable to other energy sources, such as solar PV, thermoelectric, and fuel cells.

Chapter 6

Discrete Implementation of a Distributed Maximum Power Point Tracking System for TPV

To ensure that the TPV cells in the architecture of Fig. 5.2 are each operated at the maximum power point (MPP), power electronics are often employed. By continuously tracking the MPP, more power can be extracted from a given cell. A schematic drawing of the discrete maximum power point tracker (MPPT) developed as part of this thesis is shown in Fig. 6.1, alongside the other system components. The power tracker consists of two primary structures: the control stage and the power stage. The task of the control stage is to provide the duty cycle command to the power converter to ensure that the TPV cell is operating at its most efficient point – the maximum power point. The task of the power stage is to provide efficient conversion between the optimum cell voltage (V_{mpp}) and the load voltage. This chapter demonstrates power conversion and control techniques suitable for providing high energy extraction from TPV (and other low-voltage energy sources) while minimizing overhead loss from the power electronics, using discrete semiconductor switches, gate drivers, and a microcontroller. As we will see, the low overall output power of the energy source requires careful consideration of all parasitic losses, and pushes the design to the limit of what is achievable with commercially available discrete components.

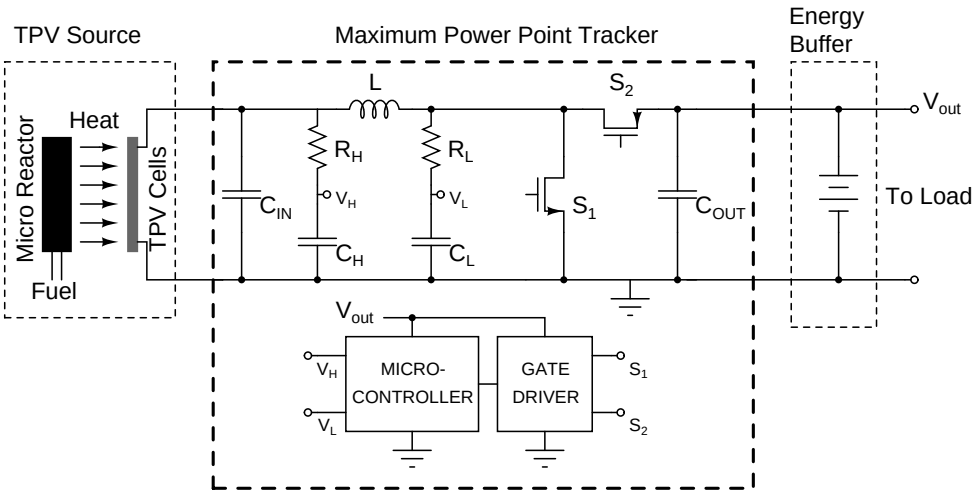


Figure 6.1: *Schematic drawing of the discrete implementation of the TPV maximum power point tracker.*

6.1 Control Algorithm and Implementation

The power stage of the maximum power point tracker of Figure 6.1 consists of a boost converter, which in addition to maintaining the PV cell voltage at the MPP voltage, also provides voltage conversion, to enable the low-voltage source (PV cell) to interface with the higher-voltage load (lithium ion battery). The boost converter shown in Figure 6.1 has an input/output voltage relationship given by:

$$V_{out} = \frac{V_{in}}{1 - D} \tag{6.1}$$

where D is the duty cycle of the bottom switch (S_L). In this synchronous rectification implementation, the top switch (S_H) is turned on when the bottom switch is off. The boost converter can be controlled to achieve peak power tracking by perturbing the duty cycle in a certain direction (increase or decrease), and observe whether the delivered power increased or decreased due to this perturbation. If the power increased, the controller continues to perturb the duty cycle in the same direction, but if the power decreased, the direction of the perturbation is changed. With this method, the controller eventually settles

on the peak power point of Figure 5.3, where it oscillates to within the finest resolutions of the duty cycle command and sensors. This method, often called hill climbing, or perturb and observe, [38] is one of the most common MPPT algorithms used to date. Figure 6.2 shows a flow chart of the MPPT algorithm. The initial starting point for the duty cycle is determined by performing a coarse sweep of the duty cycle at startup, and recording the duty cycle corresponding to the maximum output power observed. This approach ensures that the peak power tracker can quickly lock in on the maximum power point.

The algorithm described above is well-suited for an implementation in digital form, and we have chosen to use a microcontroller for our implementation. In addition to keeping state and running the tracking algorithm, the microcontroller can be used to perform analog to digital conversion, generate the PWM signals, perform temperature measurements, and handle communication. The ability of the microcontroller to handle a variety of functions is very beneficial in this low-power application, where the power loss of the auxiliary components must be kept to a minimum. An additional benefit of a multi-function chip such as the micro-controller is the significant space savings that can be realized compared to an implementation with discrete devices for each function.

6.1.1 Voltage and Current Measurement

In the general case, both current and voltage must be measured to find the maximum power point (see [36] for a discussion of cases where only one of the two needs to be measured). Typically, only the average values need to be measured, which reduces bandwidth requirements and enables the use of low-power analog to digital converter (ADC) architectures. Furthermore, the absolute value of current and voltage is not required, since the minimum or maximum power points are found relative to the other possible operating points. The ADC thus needs high resolution, but not high absolute precision, a characteristic that can be leveraged to obtain high performance while maintaining low power consumption.

The microcontroller used, the 8-bit ATtiny861 from Atmel, provides a multiplexed 10-

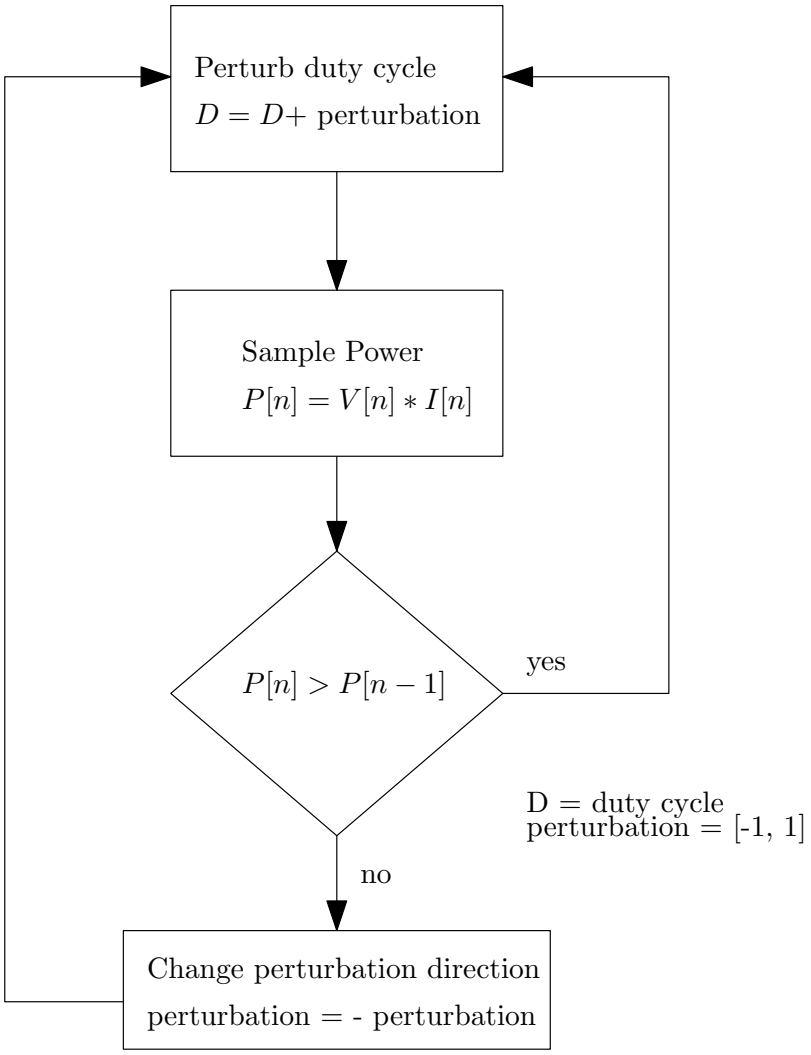


Figure 6.2: *Flow chart illustrating the operation of perturb and observe.*

bit ADC, along with an internal bandgap reference. The 10-bit precision can be further extended in the digital domain by oversampling and decimation [39]. The input and output voltages can thus easily be measured with this built-in ADC with sufficient resolution.

A more difficult challenge is that of current sensing, which is typically done with a current-sense resistor. The addition of a current-sense resistor in the current path introduces an undesired power loss, which decreases overall converter efficiency. For this reason, the current-sense resistor is typically made small, and the subsequently small voltage drop is sensed with a low-noise, high gain amplifier. In this application, with a total output power of less than 500 mW for an individual MPPT, the additional power consumption and area of a low-noise amplifier for current sensing, together with the added power loss of a current-sense resistor, was deemed too high, so alternative implementations were investigated.

Another current-sensing option is that of a hall-effect sensor, which measures the magnetic field associated with a current. With no added resistor in the current path, the only power loss is that of the magnetic sensing circuitry, which can unfortunately be quite large. Indeed, in this application it was found that the static power consumption of this method was much too large for acceptable system efficiency. As an example, the static power loss of one of the most popular low-power hall-effect sensors, the ACS712 from Allegro Microsystems, is 50 mW. This would represent a power loss of 10% in current-sensing alone for our 500 mW system, making this approach unacceptable.

Figure 6.3 illustrates the current-sensing technique used in the power tracker. To maximize overall system efficiency, loss-less current sensing [40] is used, where the average voltage drop across the inductor is measured. (By lossless, we mean incurring no *additional* loss beyond that already present in the circuit.) The relationship between inductor current I_L and sensed voltage ΔV is given by:

$$\langle I_L \rangle R_{esr} = \langle \Delta V \rangle = \langle V_H \rangle - \langle V_L \rangle, \tag{6.2}$$

where R_{esr} is the parasitic resistance of the inductor. The average voltages, $\langle V_H \rangle$ and $\langle V_L \rangle$

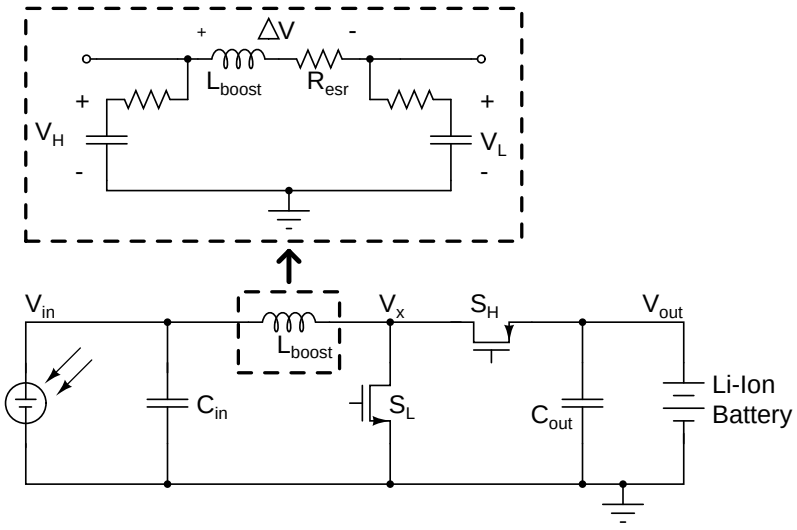


Figure 6.3: *Schematic drawing illustrating the loss-less current sensing technique used.*

are produced by first-order RC low-pass filters. These two voltages are then sampled by the differential ADC of the microcontroller with a built-in gain of 32, which gives a reading directly proportional to the inductor current. It should be noted that the common concern with this current sensing method, the tolerance and temperature coefficient of R_{esr}, is not a problem in this application. Since, for our tracking algorithm, we are only concerned with relative changes of the current, any static error in the assumed value R_{esr} has no effect on the peak power tracking. Furthermore, the time constant of any temperature-induced variation of the R_{esr} value is much larger than the chosen sampling time, so the tracking can be made insensitive to this variation as well. In our converter implementation, a relative change in current of less than 1 mA can be resolved using this method, as confirmed by experimental measurements. It should be noted that this current sensing is achieved without the need for a power-consuming series-sense resistors, and that the amplifier and ADC are built-in to the microcontroller, and thus consume negligible additional power and take up no additional area.

It should be emphasized that a key enabler to the use of this current-sensing technique is the fact that the application requires neither absolute accuracy of the current, nor instan-

taneous current values. Thus, the tolerance of the inductor resistance is not critical, and the low-pass filters can be designed to provide significant averaging over a relatively long time.

6.1.2 Tracking Precision and Speed Trade-offs

As in any MPPT application, there exists a trade-off between tracking speed and accuracy in our converter. By averaging many current and voltage samples, it is possible to achieve a very accurate power measurement. However, as the number of samples required for each decision increases, so does the minimum time between decisions, which affects the speed at which the converter responds to changes in the maximum power point.

For an N-bit analog-to-digital converter (ADC), the quantizer step size, Δ, is given by

$$\Delta = \frac{V_{ref}}{2^N}, \tag{6.3}$$

where V_{ref} is the analog reference voltage. In the conversion between continuous analog values and discrete digital codes, the ADC introduces a quantization error, e_q, given by :

$$|e_q| = \frac{\Delta}{2} \tag{6.4}$$

For many types of signals the quantization errors can be represented statistically. Reference [41] contains a description of this process and the assumptions that enable the statistical representation of quantization errors. We note that in our work, the sampling time of the ADC and the switching transitions of the power stage are not synchronized, making the assumption of uncorrelated error and signal sequence good. Furthermore, experimental work [42] has shown that as the signal complexity increases, the signal and quantization error become more uncorrelated, enabling a statistical representation of the errors. Thus, assuming that quantization error is a white noise process with zero mean and variance, the

average noise power can be calculated as:

$$\sigma_e^2 = \int_{-\frac{\Delta}{2}}^{\frac{\Delta}{2}} p(e_q)\, e_q^2\, de_q = \frac{\Delta^2}{12} \tag{6.5}$$

Furthermore, it can be shown [41] that if the input signal is oversampled by a factor or M, the noise power n_p is given by:

$$n_p = \frac{\Delta^2}{12M} \tag{6.6}$$

According to equation 6.6, for every multiple of 4 that we oversample the signal by, we achieve a 1-bit increase in the effective resolution. In the maximum power point tracking problem addressed in this work, the frequency of the signal of interest corresponds to the frequency of duty cycle changes (tracking frequency), and is determined by the MPPT algorithm. An upper bound of the MPPT tracking frequency is approximately 10 times lower than the switching frequency, to enable the converter to reach steady-state operation after a change in duty cycle. A lower bound of tracking frequency is determined by the system time constants associated with a change in the maximum power point location, which in this application are quite long (on the order of seconds). Equation 6.6 can thus be leveraged to improve the steady-state tracking efficiency. In this implementation, we oversample by a factor of 256, giving a maximum tracking frequency of approximately 500 Hz. This is still considerably faster than what is required by the application, and as we shall see, provides excellent tracking efficiency.

6.2 Discrete converter prototype

An experimental prototype of the MPPT converter has been developed and characterized. Figure 6.4 shows a photograph of the peak power tracker, and Table 6.1 lists the converter specifications; converter efficiency includes all control and gate driver losses. The efficiency measurement was taken at a load of 500 mW, and input voltage of 1 V, and an output voltage of 4 V. Table 6.2 lists the estimated loss-breakdown at this operating point. The

tracking efficiency is a measurement of how close the tracking algorithm operates to the true maximum power point, and is given by:

$$\eta_{tracking} = \frac{\langle P_{in} \rangle}{P_{MPP}}, \tag{6.7}$$

where P_{in} corresponds to the converter input power, and P_{MPP} is the output power of the TPV module at the maximum power point. Due to the low voltage point of the TPV module ($\sim$ 1 V), it is difficult to make a high precision input power measurement of the converter without also perturbing the actual operating point of the converter. An easier, but strictly speaking less accurate, approximation of the tracking efficiency can be found by calculating the ratio:

$$\eta_{tracking,approx.} = \frac{\langle P_{out} \rangle}{P_{out,max}}, \tag{6.8}$$

where $P_{out,max}$ corresponds to the maximum output power from the converter. This is only an approximation, and will over-estimate the tracking efficiency because $P_{out,max}$ will not correspond to the exact peak power point, owing to the finite resolution of the digital PWM implementation. However, with proper knowledge of the cell I-V curve (Figure 5.3) and the tracking algorithm step-size (PWM resolution is this implementation), one can find an upper bound on the error in the approximation given by 6.8, and from there calculate a minimum tracking efficiency. Using this technique, the tracking efficiency of the converter considered here was found to be above 99%.

The converter design was guided by the desire to achieve small system size and weight, while maintaining high efficiency. As can be seen in Figure 6.4, the majority of the circuit board area is taken up by connectors, while the converter core (switching devices, micro-controller, and passive components) take up a relatively small area. Figure 6.5 provides a detailed schematic drawing of the converter. As shown, the converter can be powered either from the Li-Ion battery output, or from an external power supply. Table 6.3 lists the components used in the experimental prototype. Appendix E provides a complete Bill of Materials listing, as well as a full schematic for the design. The microcontroller code used

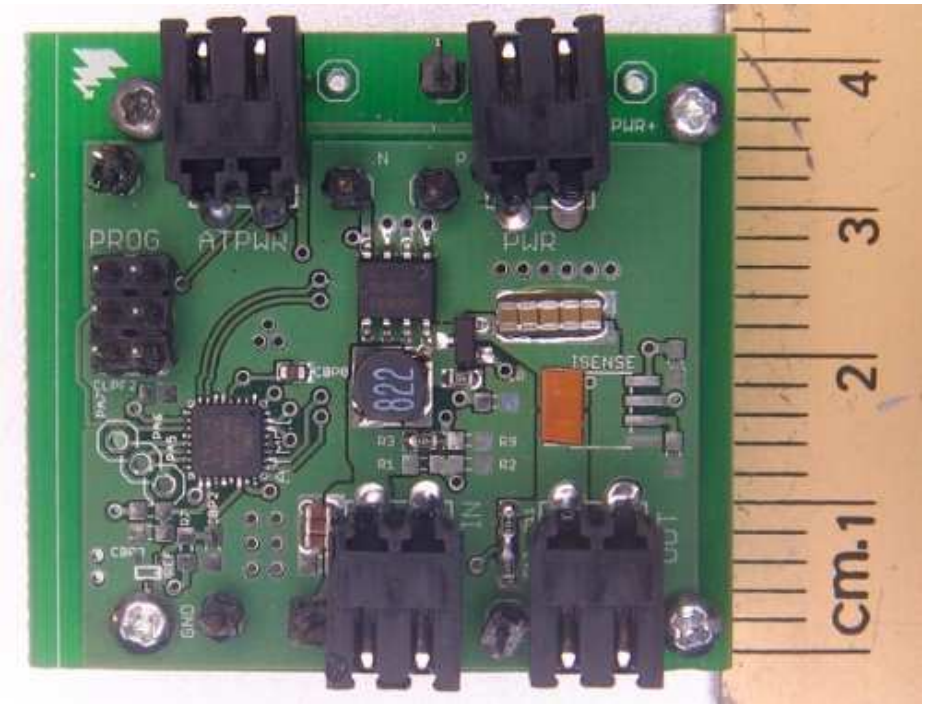


Figure 6.4: *Photograph of the peak power tracker.*

to perform the MPPT is provided in Appendix F.

6.3 Converter experimental verification

To evaluate the performance of the peak power tracker, the converter was initially connected to a PV diode illuminated by a quartz halogen lamp. The lamp brightness and distance from

Table 6.1: *Converter Specifications*

Input Voltage	0.3-1.1 V
Output Voltage	1.5-4.2 V
Output Power	500 mW
Switching Frequency	250 kHz
Converter Efficiency	90%
Tracking Efficiency	>99%

Table 6.2: *Converter Specifications*

Loss Components	Normalized Loss [%]
Transistor Conduction Loss	1 %
Transistor Switching Loss	2 %
Inductor Conduction Loss	1 %
Inductor Core Loss	< 0.5 %
Microcontroller Power Consumption	5 %

Table 6.3: *Component Listing. See Appendix E for additional information such as PCB image files and Eagle schematic drawings.*

Device	Model	Value	Manufacturer
S_1	BSO300N03S		Infineon Tech.
S_2	SI2351DS		Vishay Siliconix
L	MSS5131-822ML	8.2 μH	Coilcraft
R_H, R_L	0603	100 kΩ	Panasonic
C_H, C_L	0603	10 μF	Murata
C_{IN}	0805	3 μF	Murata
C_{OUT}	0805	50 μF	Murata
Microcontroller	ATtiny861		Atmel
Gate Driver	LM5111		National Semi.

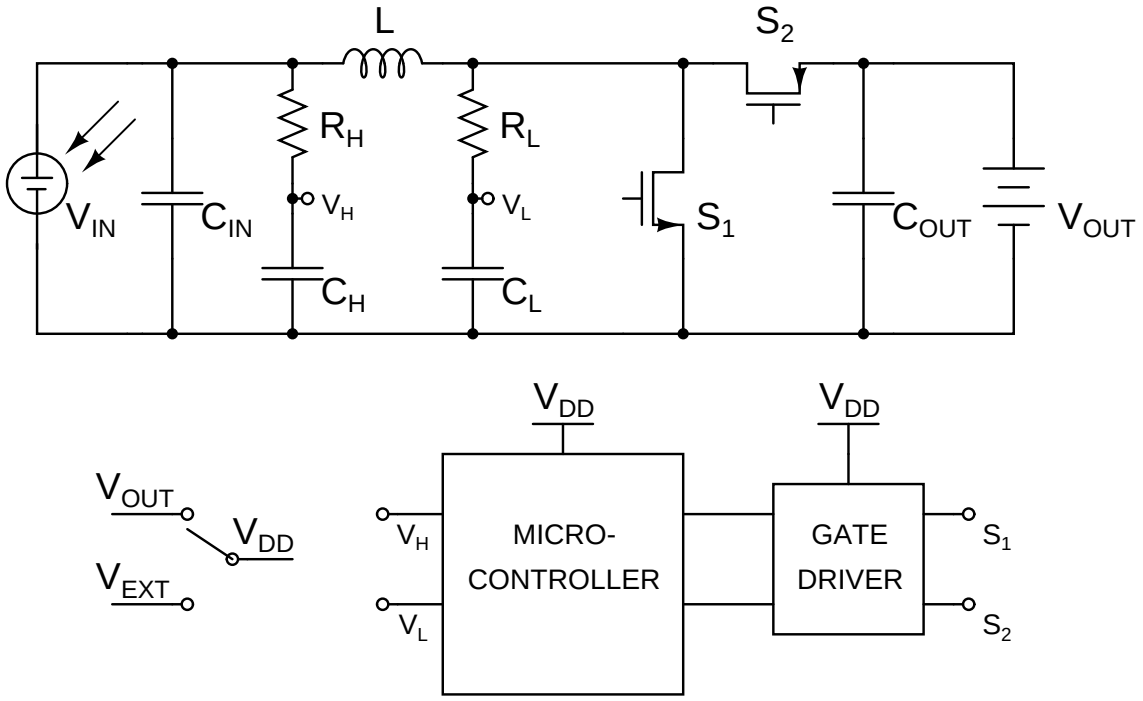


Figure 6.5: *Schematic drawing of converter.*

the cell was adjusted to match the expected power output from the cell when illuminated by the micro-reactor (500 mW). This enabled initial characterization of the converter without the added complexity of the micro-reactor dynamics. Figure 6.6 (top) shows the output power of the converter over time, and illustrates the MPPT startup algorithm for this experimental setup. Initially, the converter steps its duty cycle through a coarse sweep to find the approximate point of the MPP. The duty cycle corresponding to the maximum power observed is recorded, and once the sweep is concluded, the duty cycle is set to this value. At this point, the converter enters the hill-climbing phase (perturb and observe), and uses a fine step-size to reach the MPP. Note that the step-size of the hill-climbing algorithm is too small to be visible in the top plot.

The steady-state behavior of the hill-climbing algorithm is shown in the bottom of the figure, which shows the converter output power versus time in steady-state. This is a zoomed-in version of the top plot, and shows the discrete steps in power corresponding to

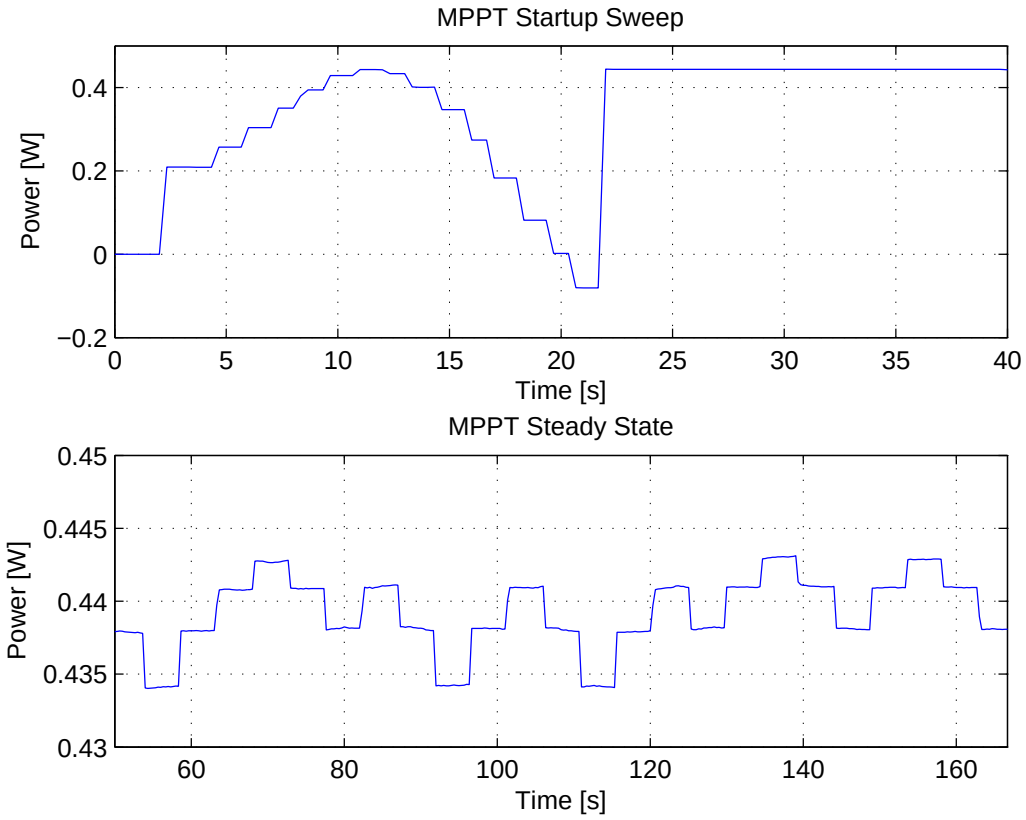


Figure 6.6: *Experimental data showing startup behavior of power tracker (top), and steady-state performance (bottom).*

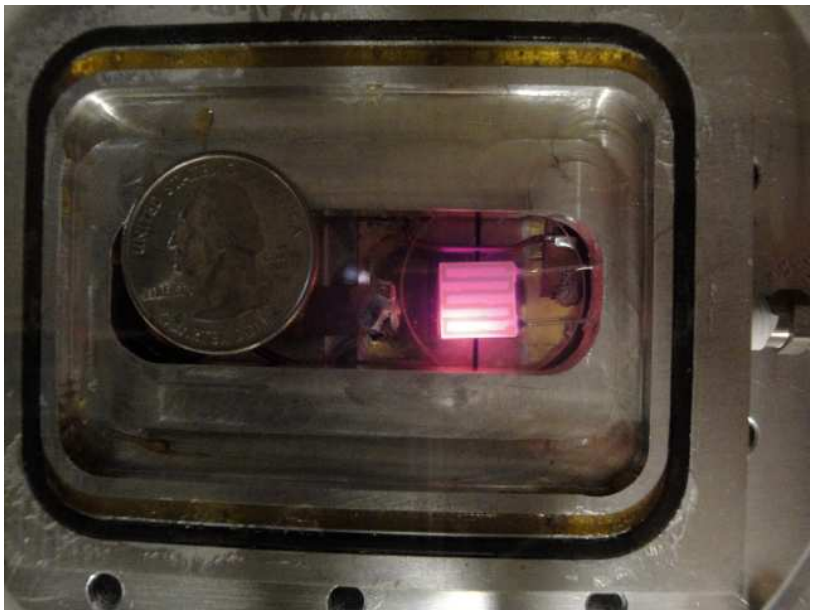


Figure 6.7: *Photograph of the experimental setup with the top two PV cells removed and a US quarter for scale. The MEMS burner and the bottom two PV cells are visible.*

a 1-bit change in duty cycle. The total PWM resolution of the micro-controller is 10 bits. The converter oscillates around the MPP to within the resolution of the PWM signal and the current and voltage sensors. Because the sensing and duty cycle control have similar resolution, the hill-climbing algorithm is limited by sensing noise, and occasionally takes one extra step in the wrong direction. It should be noted that the sampling interval for the MPPT algorithm has been set to several seconds, as seen in Figure 6.6 (bottom). This was done to enable high accuracy power measurements by the external instruments used to characterize the converter, and is not a fundamental limit of the converter itself. If desired, the MPPT algorithm can be set to sampling frequencies considerably higher (on the order of several kHz) without a noticeable impact on tracking efficiency. In this application, however, the system time constant of any change in maximum power point is long enough such that the sampling frequency of Figure 6.6 is sufficient to allow efficient energy extraction from the TPV module.

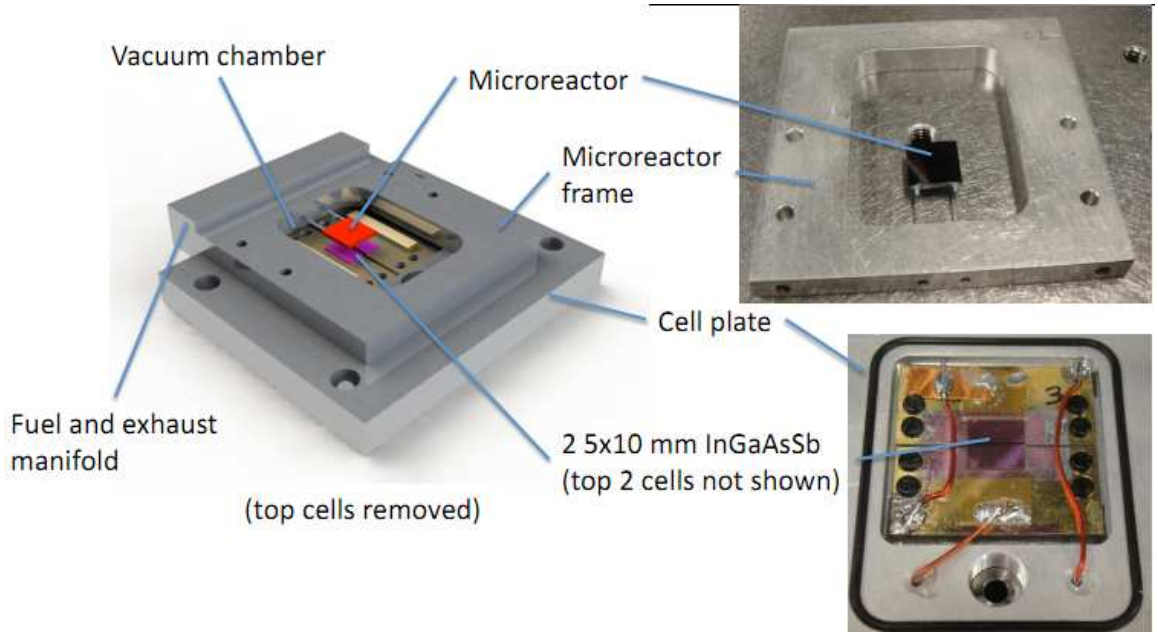


Figure 6.8: *System overview of the different components of the TPV micro-reactor system. Image courtesy of Walker Chan.*

6.4 Micro-reactor experimental results

In order to fully evaluate the MPPT converter performance in our complete system, we tested it with an experimental system setup similar to the one depicted in Figure 5.2. The PV cells were illuminated with the micro-reactor, shown in the photo of Figure 6.7. The reactor is a 10 mm by 10 mm by 1 mm silicon slab with a serpentine, platinum catalyst-loaded channel running through it [34]. Figure 6.8 shows the different components of the TPV micro-reactor [43] A mixture of butane and oxygen is fed into one end of the channel; carbon dioxide and water vapor are exhausted from the other end. With a butane flow of 8 sccm (standard cubic centimeters per minute) and 80 sccm of oxygen, the average surface temperature is 850°C. For reference, an ordinary pocket lighter burns 15 sccm of butane.

In the experimental setup, the two GaInAsSb PV cells are located directly above the burner and another two cells are located below the burner as shown in Figure 6.7. These four PV cells are connected in series and their output is connected to the MPPT converter.

Experimental data from the complete system setup is shown in Figure 6.9, which shows converter output power versus time. As expected, this plot looks similar to Figure 6.6, but there are some notable differences. This first generation micro-reactor assembly has a typical output power of 150 mW, due to the cell being placed at a distance from the burner that is too far for optimum power transfer. Despite this, the demonstrated system output power is more than two orders of magnitude higher than what has previously been achieved [44]. The measured energy density of this micro-TPV system is 75 mW/cm^2. For comparison, the best power densities reported for micro scale direct methanol fuel cell (DMFC), with comparable size to this TPV system, are in the range from 4 to 30 mW/cm^2 [45]. It should be noted that while this early burner prototype has a lower efficiency than the fuel cell presented in [45], previous TPV results [33] show that a comparable efficiency to that of a fuel cell system is achievable. With better system packaging and by further optimizing the system design we are targeting a micro-TPV system power density of 250-300 mW/cm^2.

One of the difficulties encountered during system testing was that the burner experiences occasional temperature fluctuations due to condensed butane entering the fuel supply. Butane is delivered to the burner as a gas but occasional droplets, representing additional fuel, can enter the inlet stream. When a droplet enters the burner, there is a sudden increase in temperature as it burns. Figure 6.9 captures such an event, which occurs slightly before time t=45 seconds, with a correspondingly large increase in output power, followed by an exponential decay back to steady-state. The time constant associated with this event is such that the MPPT algorithm may take one or two steps in the wrong direction during the increasing power phase, followed by a continuous change of direction during the exponential decay, since the output power at each sample time is lower than the previous sample. The result is that while the converter may operate slightly off of the peak power point during this transient event, it is guaranteed not to move more than a few steps in the wrong direction, ensuring a quick return to the maximum power point once the burner has returned to equilibrium.

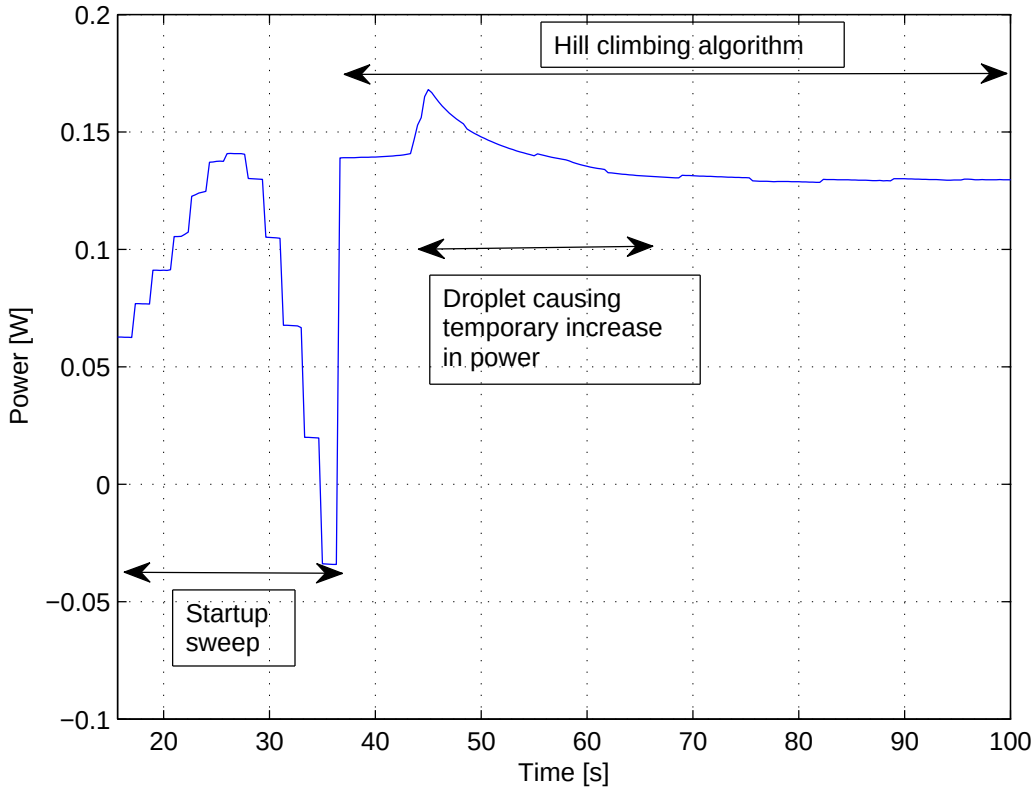


Figure 6.9: *Experimental data showing the output power of the MPPT as a function of time. The temporary increase in output power around time t=45 seconds is due to a butane droplet forming and causing an increase in burner temperature.*

Chapter 7

Integrated Distributed MPPT in 0.35μm CMOS

The previous chapter illustrated the benefits associated with a distributed MPPT architecture in our TPV application, along with experimental results from an MPPT implementation using commercially available, discrete semiconductors. While we achieved relatively good efficiency and small size, our discrete implementation approaches the limit of what is achievable using off-the-shelf discrete semiconductors and microcontrollers. As we seek to realize further improvements in efficiency as well as significant reductions in size, the best option is to pursue a fully integrated custom design in a low-voltage CMOS process.

A custom CMOS chip enables us to substantially decrease the converter parasitic power losses, thanks to two advantages offered in an integrated process:

1. A custom CMOS design enables the use of low-voltage power devices with optimal device widths for a specific operating frequency. Rather than being limited to the small selection of discrete transistors of suitable device widths (and often-time higher operating voltage than our desired 5 V), we can tailor each power transistor to just the right on-state resistance and parasitic capacitance trade-off that we desire for our operating power and frequency.

2. Custom control circuitry can achieve considerably lower power consumption than a full-fledged microcontroller, which is a general purpose device with many peripherals that consume power whether they are needed or not. By only implementing the mini-

mum required hardware to achieve our functionality, we can expect a drastic reduction in control losses. This is particularly important in our low-power applications, where even just a few mW of control losses has a big impact on overall efficiency.

In this chapter we seek to leverage the advantages of a custom design in a low-voltage process, and present solutions for achieving very low standby and control power, as well as a size/efficiency optimized power stage.

7.1 System Overview

The maximum power point tracker we have developed is illustrated in the schematic drawing of Fig. 7.1, alongside the other system components. The power tracker consists of two primary structures: the control stage and the power stage. The task of the control stage is to provide the duty cycle command (and associated gate drive signals) to the power devices to ensure that the TPV cell is operating at its most efficient point – the maximum power point. Many different techniques [38] have been proposed to implement the maximum power point tracking functionality. In this work, we use Perturb and Observe (P&O) [46]. Since the duty cycle (D) directly affects the input voltage (cell voltage) through the boost converter relationship $V_{in} = V_{out} * (1 - D)$, it is sufficient to perturb the duty cycle and observe the change in input power. The P&O technique is well-suited for digital implementation, which we have chosen for our 0.35 μm CMOS design. The details of the control stage are presented in section 7.2

The power stage comprises a CMOS integrated boost converter with an off-chip inductor and capacitors. The control stage and gate drivers are all powered from the intermediate energy buffer on the output, which is a lithium-ion battery in Fig. 7.1, but can be any charge storage device with suitable energy density and voltage range. A detailed description of the power stage and its operation is presented in section 7.3.

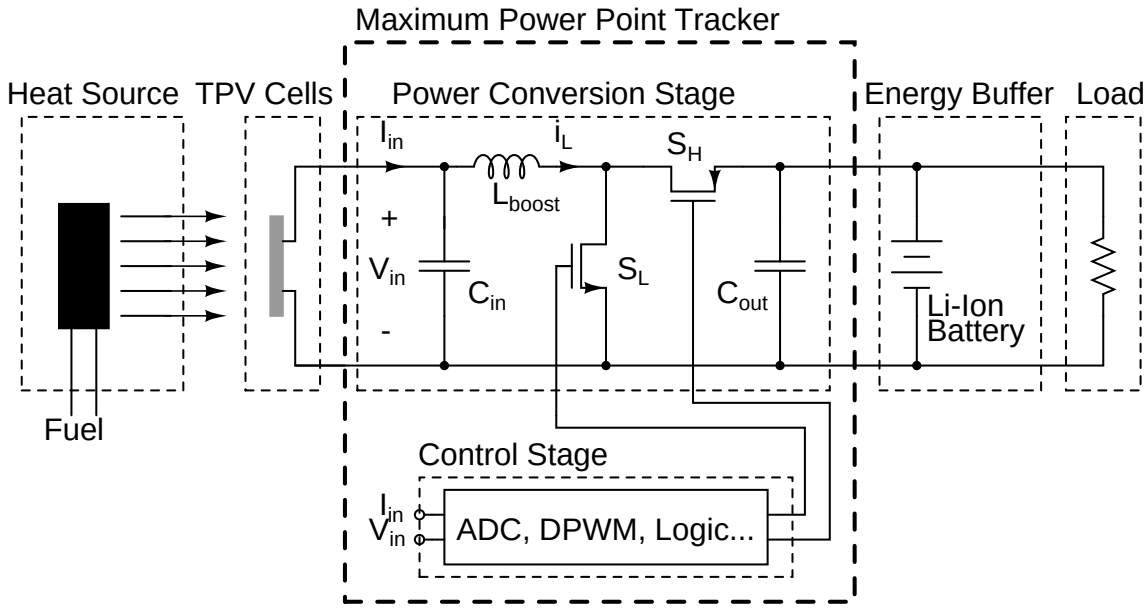


Figure 7.1: *Schematic drawing of the system architecture. The integrated maximum power point tracker consists of a boost converter power stage and a control stage, all implemented in a 0.35 µm CMOS process. The main boost inductor L_{boost} and input and output bulk capacitors are placed off-die, though there is significant on-die capacitance across the output of the boost converter for high-frequency switching currents.*

7.2 Control

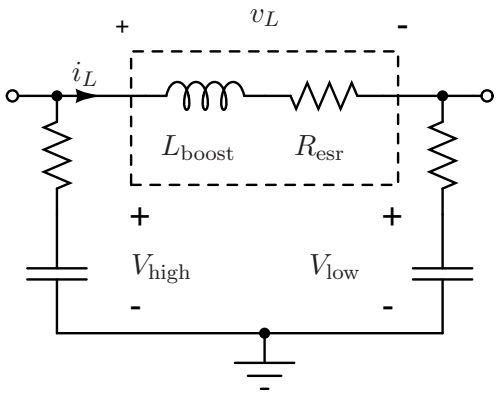
Here we introduce how the controls of our system are realized while achieving the goals of very low sensing and control loss and maximum extraction of available energy from the source.

7.2.1 Lossless Current Sensing

While voltage sensing is typically relatively easy to implement, sensing of current in a power converter is often more challenging. The current sensing method used in this work is shown in Fig. 7.2. It provides lossless sensing of the current by utilizing the parasitic resistance of the power inductor (L_{boost} of Fig. 7.1). (The approach is "lossless" in the sense that it does not introduce additional loss beyond what is already unavoidably present in the circuit.) This method results in overall increased conversion efficiency, since no additional sense resistors are introduced into the circuit, which would add power loss to the system. The average voltage across the inductor, $\langle v_L \rangle$, is directly proportional to the average inductor current, I_L, since in steady-state, $L\langle \frac{di_L}{dt} \rangle$ is zero by definition. The low-pass filtered differential voltage $V_{\text{high}} - V_{\text{low}}$ can thus be used to measure the average input current. This sensing method is well suited to this application as we only need to know *relative* currents (and powers), not absolute values. Variations in inductor ESR are thus not problematic. Furthermore, as was illustrated in the discrete implementation of Chapter 6, the time constant of any temperature-induced variation of the ESR value is much larger than the chosen sampling time, so it does not negatively affect tracking performance.

7.2.2 Analog to Digital Converter Overview

We implemented the ADC architecture of Fig. 7.3 to convert the analog low-pass filtered differential voltage of Fig. 7.2 to a digital value. The architecture provides inherent low-pass filtering through the counting stage, which is beneficial since it reduces the analog



$$v_L = L\frac{di_L}{dt} + i_L R_{esr}$$

$$\langle v_L \rangle = L\langle \frac{di_L}{dt} \rangle + \langle i_L \rangle R_{esr}$$

$$\langle v_L \rangle = \langle i_L \rangle R_{esr}$$

$$\langle v_L \rangle = V_{\text{high}} - V_{\text{low}}$$

Figure 7.2: *Schematic drawing of the lossless current sensing implementation. The voltage drop across the inductor parasitic resistance R_{esr} is extracted through low-pass filtering.*

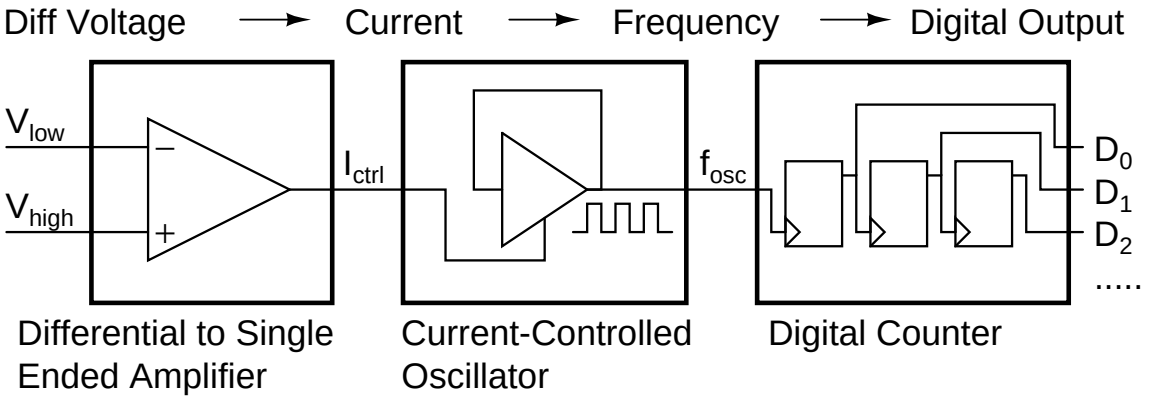


Figure 7.3: *Block diagram of the differential ADC architecture with inherent low-pass filtering and low power and area requirements.*

filtering requirements of the signal. This directly translates to a reduction in silicon area by the integrated filter resistors and capacitors. Other key characteristics of the architecture of Fig. 7.3 are low power consumption and very small area. The active area occupied by the two ADCs (for current and voltage measurement) is 0.083 mm^2, and the power consumption for two ADCs at a sampling rate of 100 Hz (much faster than what is required for the application) is 48 μW. Furthermore, the ADC architecture can be implemented as a single-ended ADC by connecting V_{low} to a fixed reference voltage. We use this strategy to measure the input voltage of the MPPT, with V_{low} tied to ground and V_{high} connected to the input voltage through a resistor divider.

Here we discuss the operation and design of the components of Fig. 7.3 in more detail:

7.2.3 Differential voltage to single-ended current converter

The conversion from differential voltage to single-ended current is performed by the circuit block shown in Fig. 7.4, which is a translinear amplifier adapted from [47]. The circuit operation can be analyzed by using the translinear principle [48, 49]:

$$V_{\text{low}} - V_{GS1} - V_{GS4} + V_R + V_{GS3} + V_{GS2} = V_{\text{high}} \tag{7.1}$$

Since the currents through M2 and M4 are the same, their corresponding V_{GS} values must also be the same. A similar argument holds for M1 and M3, resulting in:

$$V_{GS2} = V_{GS4}, \qquad V_{GS1} = V_{GS3} \tag{7.2}$$

Using the results of Eq. 7.2 in Eq. 7.1 gives the result:

$$V_R = V_{\text{high}} - V_{\text{low}}$$

$$i = \frac{V_{\text{high}} - V_{\text{low}}}{R}$$

The current $I_{\text{bias}} + i$ is mirrored to the output, and transistor M_{sub} is biased to subtract I_{bias}, leading to:

$$I_{\text{ctrl}} = i = \frac{V_{\text{high}} - V_{\text{low}}}{R}$$

Shown in Figure 7.5 is a plot of simulated performance of the voltage-to-current converter. It shows the output current (I_{ctrl}) versus differential input voltage (V_{low} is held at 500 mV while V_{high} is swept from 500 mV to 512 mV, corresponding to the expected maximum average inductor voltage drop of 12 mV). Also shown is a linear least-squares estimate, illustrating the good linearity of the converter. We can characterize the converter by its voltage to current coefficient, $K_{vi} = \frac{dI}{dV}$. In this example, K_{vi} is approximately 0.293 $\mu A/mV$. Much care was taken in the design of the converter to minimize linearity errors. The transistor M_{sub} is not set to subtract the entire 5 μA bias current, but only 4.5 μA to increase linearity, as determined by simulation.

7.2.4 Current-controlled oscillator

The output current of the circuit block of Fig. 7.4 is used to control the frequency of the current-controlled oscillator of Fig. 7.6. It comprises a bias network, current-starved inverter, an on-chip capacitor, and a Schmitt trigger to produce a square-wave output voltage whose frequency is dependent on the input current.

The oscillation frequency is given by:

$$f_{osc} = \frac{I_{\text{ctrl}}}{2\Delta V_{\text{Schmitt}} C_{\text{osc}}}, \tag{7.3}$$

where $\Delta V_{\text{Schmitt}}$ is the hysteretic voltage of the Schmitt trigger (which thus sets the amplitude of the triangle waveform), and C_{osc} is the capacitor value. The resulting waveform has a duty cycle of approximately 50%, owing to the fact that the charge and discharge transistors of the current-starved inverter are biased by the same current.

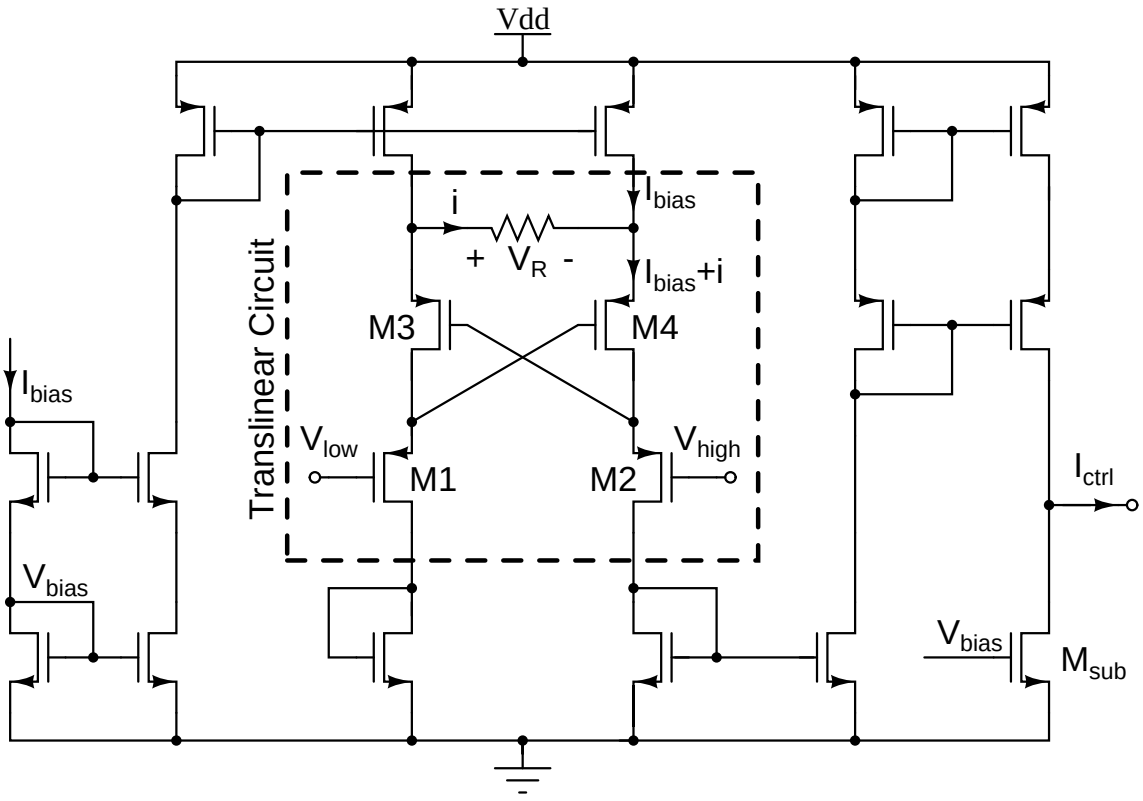


Figure 7.4: *Schematic diagram of differential voltage to single-ended current converter used as the first stage of the ADC architecture of Fig. 7.3.*

By changing the bias current, we can thus control the oscillation frequency in a linear matter. Since C_{osc} and $\Delta V_{\text{Schmitt}}$ are determined at design time, we can combine them into a single coefficient, K_{if} giving us the relationship

$$f_{osc} = K_{if} I_{bias}.$$

Figure 7.7 shows a simulated plot of the frequency versus control current characteristics for the Schmitt trigger oscillator, together with a linear least square error fit. From this, we we can deduce the proportionality constant K_{if} to be approximately 0.94 MHz/μA. We also see from the plot that the frequency and bias current are very well approximated by a linear relationship. In this simulation (and in the experimental prototype), C_{osc} has a value

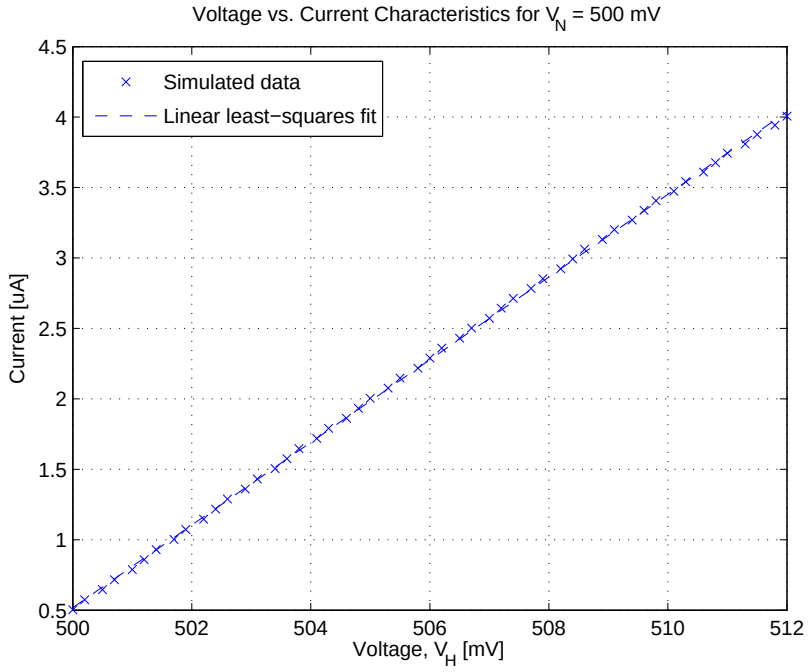


Figure 7.5: *Plot showing simulated performance of the voltage to current converter of fig:translinear, together with a linear least-squares estimate.* V_{low} *is held at 500 mV while* V_{high} *is swept from 500 mV to 512 mV, corresponding to the expected maximum average inductor voltage drop.*

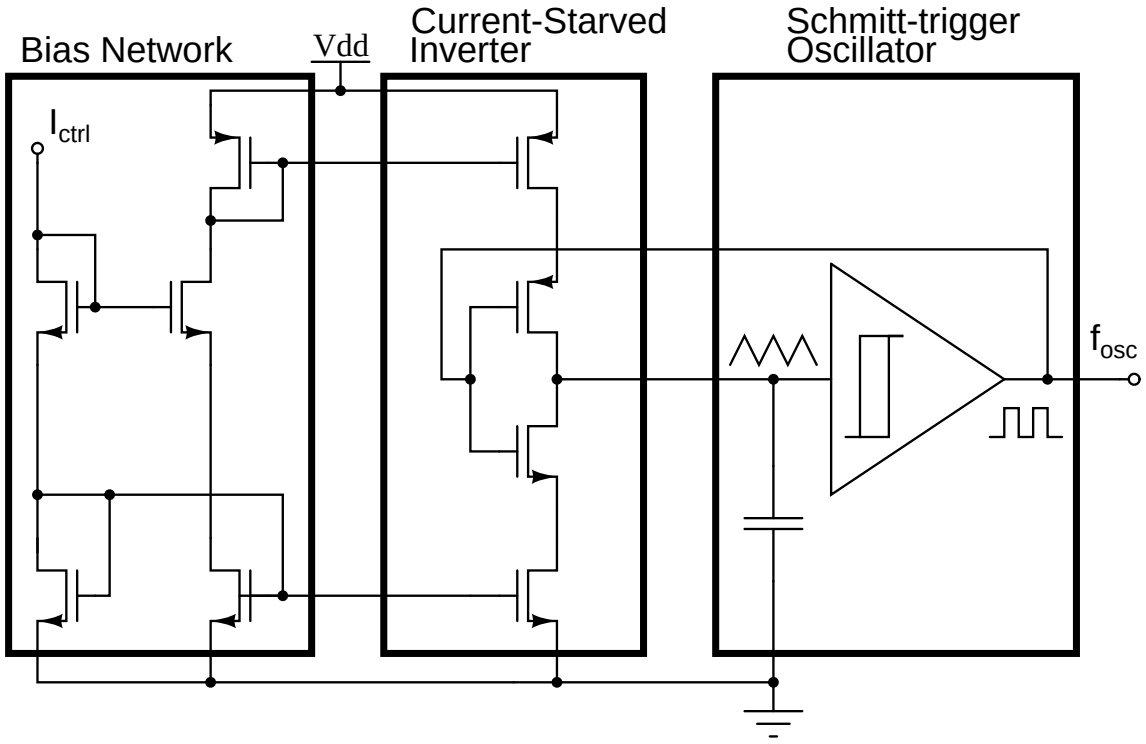


Figure 7.6: *Schematic diagram of current-controlled oscillator used in ADC architecture of Fig. 7.3*

of 273 fF, and the Schmitt trigger oscillator has a hysteretic voltage value of 1 V.

7.2.5 Digital Counter

The output of the current-controlled oscillator (f_{osc}) is fed into a digital counter to produce a value proportional to the differential input voltage. A schematic drawing of the 9-bit digital counter is shown in Fig. 7.8. The counter is resettable via the RESET command, followed by an ENABLE command that begins the counting phase.

The relationship between the count K, and our other parameters is is given by:

$$K = (V_{\text{high}} - V_{\text{low}})K_{vi}K_{if}T_{sample},$$

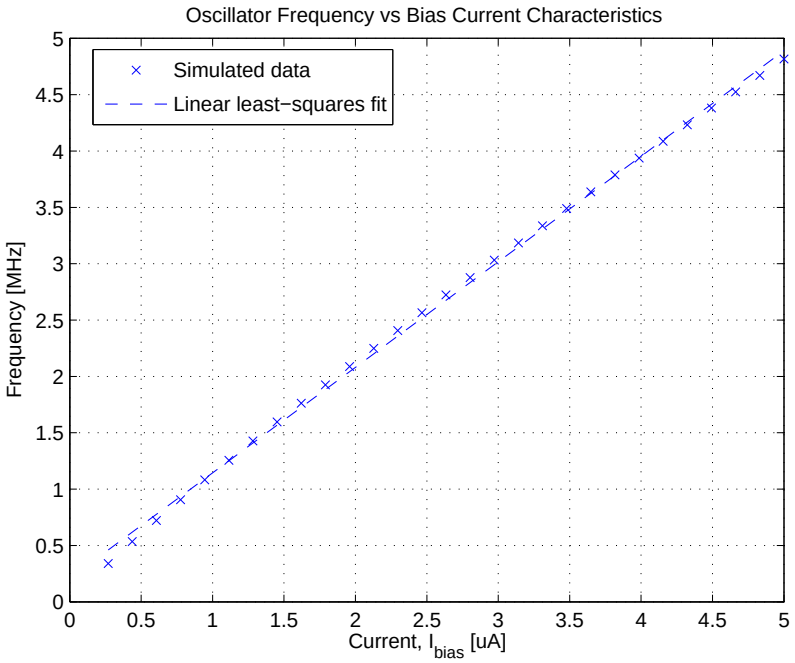


Figure 7.7: *Plot showing simulated control current to frequency relationship of the Schmitt trigger-based oscillator of Figure 7.6. Also shown is a linear approximation for the relationship.*

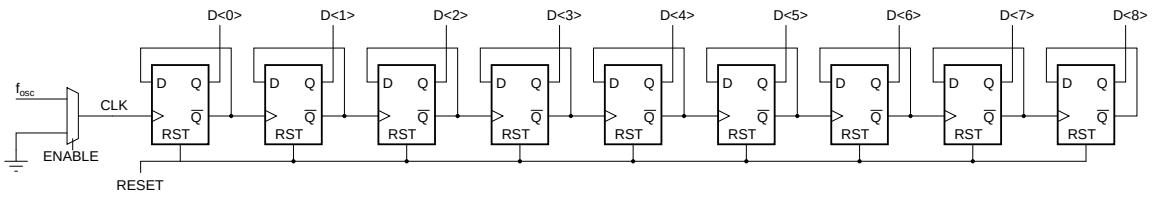


Figure 7.8: *Schematic diagram of the digital counting stage used in ADC architecture of Fig. 7.3.*

where T_{sample} is the sampling time[1], and the other parameters are as described previously. We denote K_H by the count we will get for the largest inductor current we see (400 mA in this application), and K_L by the count corresponding to the lowest inductor current we see (0 mA). We must then choose T_{sample} such that $K_H < 512$ (for a 9-bit counter) to prevent counter overflow. To keep the counter (and subsequent logic elements) relatively small, we also desire K_H to be as small as possible, given the constraint above. While it is tempting to try to design the system such that K_H is 512 and K_L is 0, this should typically be avoided, as it implies that the voltage to current converter needs to be linear all the way down to zero current, which is very difficult to achieve in practice.

In our TPV system, the largest inductor ESR that we expect to see is 30 mΩ, giving us a maximum average inductance voltage drop of 12 mV. From Figures 7.5 and 7.7, we can see that this corresponds to a maximum expected control current of 4 MHz. However, since Figure 7.5 7.7 were generated from typical transistor models under room temperature conditions, we also determined the maximum frequency under 80 degrees Celsius, with fast-fast corner transistors. Through simulation, we observe a maximum frequency of 4.3 MHz in that case, which will determine the appropriate sample time to ensure that the counter does not overflow. Our maximum sample time for a 9-bit counter is thus:

$$T_{sample} = \frac{C_H}{f_{osc,max}} = \frac{512}{4.3} = 119\mu s \tag{7.4}$$

[1]If possible, it would be preferable to make T_{sample} an integer number of switching periods to help cancel out the effect of the residual ripple. This is similar to the 60 Hz noise canceling technique commonly used in dual-slope ADCs. In this work, the sampling times is controlled from off-chip, so the added timing complexity of integer sampling made this a less attractive option.

A sampling time of 119 μs will correspond to an approximate K_L of 60, giving our ADC an effective resolution of $K_H - K_L = 512 - 60 = 452$. It should be pointed out that it is possible (through careful fine-tuning) to achieve an effective resolution of 9-bit in this ADC, despite the non-zero frequency associated with a zero voltage drop across the inductor. Implementing the resistive dividers used to sample V_{high} and V_{low} to provide a slight negative differential voltage would have the effect of decreasing K_L all the way to zero, if desired. While the non-linearity would suffer at very low counts, this may be a desirable trade-off, in particular if high resolution at higher currents is important.

In out TPV MPPT experimental prototype, we provide the sampling clock externally, to enable a wide range of tunable ADC resolutions for a variety of inductor ESRs and output powers. The digital counter was implemented using low-voltage transistors in the 0.35 μm process, which can operate at substantially higher frequencies than the maximum 4.3 MHz used here. For applications which use very high frequency sampling, care must be taken to employ flip-flop with sufficiently high operating frequency range.

7.2.6 Digital Logic

The MPPT algorithm was implemented in digital logic, and Fig. 7.9 shows a block diagram of the key components. The current and voltage measurements are provided as 9-bit values from the ADC, and the digital multiplier calculates the corresponding input power. This power is then compared to the last power sample, and if it is smaller, the perturbation direction is changed. Depending on the direction, the digitally-stored duty cycle command is either incremented or decremented in the accumulator, and the duty cycle command is translated to a time-domain waveform by the digital pulse-width modulator.

Through appropriate choice of sampling time and resistor dividers, the ADC and digital logic described in this work can be employed in a variety of output power applications. For the parameters calculated here, an expected power range of 0-500 mW with an effective sensing resolution higher than 8-bits can be achieved.

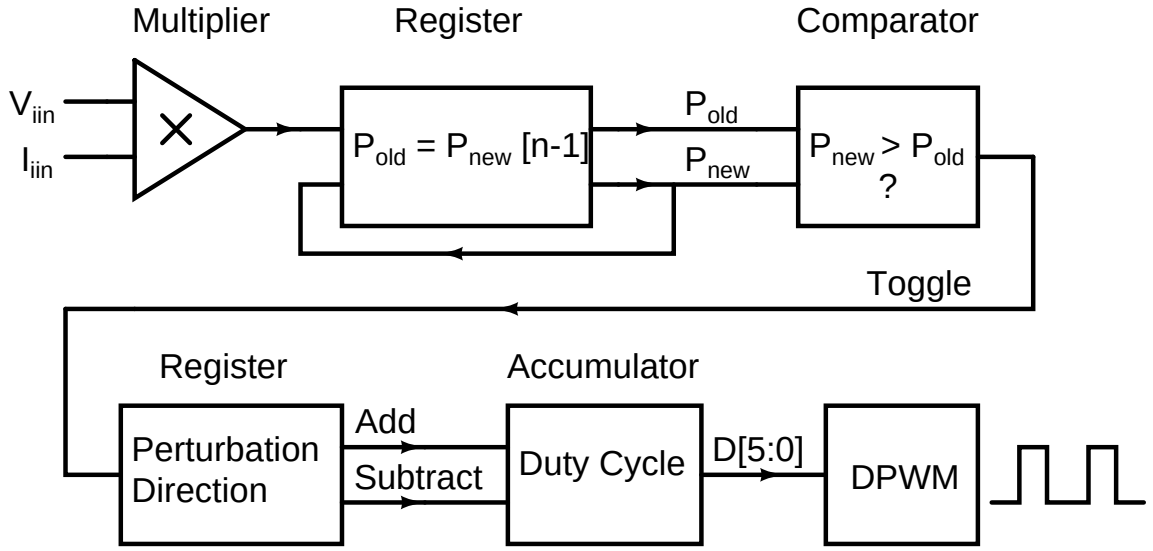


Figure 7.9: *Block diagram illustrating digital implementation of Perturb and Observe MPPT algorithm.*

7.2.7 ADC Power Consumption Discussion

A key metric in an ADC is the energy required per conversion. This is particularly important in this application, where the sampling rate is less important. We present here a discussion of power consumption and energy per conversion, as it pertains to the proposed ADC architecture.

Ignoring leakage currents, there is no static power consumption by the digital counter. Consequently, the energy consumed by the counter per conversion is fixed, and independent of the sampling time (but data-dependent, with higher counts consuming more energy due to the number of switched transistors). While less obvious, a similar result can be obtained for the Schmitt-trigger oscillator. The average current through the current-starved inverter leg of the oscillator is:

$$\langle I_{inv} \rangle = C_{osc} \frac{\Delta V_{Schmitt}}{T_{osc}/2} = 2C_{osc} \Delta V_{Schmitt} f_{osc} = I_{ctrl}, \tag{7.5}$$

where Eq. 7.3 is used to simplify the expression.

The average current drawn by the oscillator is then:

$$I_{avg} = I_{bias-network} + \langle I_{inv} \rangle + I_{SW,Schmitt} = 3I_{ctrl} + I_{SW,Schmitt}, \tag{7.6}$$

where $I_{bias-network}$ is the power consumed by the bias network of Figure 7.6 ($2I_{ctrl}$), and $I_{SW,Schmitt}$ is the dynamic current consumed at each switching interval in the Schmitt trigger itself. If we (for now) assume that $I_{SW,Schmitt}$ is negligible (compared to the dynamic and static currents from the other components of Figure 7.6), Eq. 7.6 can be used to express the approximate average power consumption of the ADC as:

$$P_{adc} = 3V_{DD}I_{ctrl}. \tag{7.7}$$

The energy per conversion required from the oscillator is simply the average power drawn times the sampling time:

$$E_{conv} = 3V_{DD}I_{ctrl}T_{sample} \tag{7.8}$$

$$E_{conv} = 3V_{DD}2C_{osc}\Delta V_{Schmitt}f_{osc}T_{sample} = 6V_{DD}C_{osc}\Delta V_{Schmitt}K_F, \tag{7.9}$$

where K_F is the final value of the digital counter (equal to $f_{osc}T_{sample}$) after the sampling is complete. We see that the energy per conversion of the oscillator is data-dependent, but it is not dependent on the sampling time. We should also point out that to minimize the energy required per conversion, it is desirable to keep C_{osc} and $\Delta V_{\text{Schmitt}}$ as small as possible. Reducing either of these two values too much, however, increases the noise of the oscillator. Making $\Delta V_{\text{Schmitt}}$ too small makes the transition times of the Schmitt trigger susceptible to small variations in transistor threshold values and increases the coupling between voltage noise on the capacitor and jitter in the frequency output. A small change (noise) in voltage can then have a large impact on the frequency output. Similarly, making the capacitor C_{osc} too small increases the $\dfrac{kT}{C}$ noise at the input of the Schmitt trigger.

It should be noted that if C_{osc} and $\Delta V_{\text{Schmitt}}$ are made sufficiently small, the approxi-

mation that $I_{SW,Schmitt}$ is negligible is no longer valid. For ultra low-power designs, low power Schmitt trigger circuits [50,51] must be investigated, or other more suitable oscillator circuit employed.

The voltage to current converter does in fact consume static power during a conversion, so the energy per conversion will depend on the sampling time[2]. The power consumed by the translinear voltage to current converter of Figure 7.4 is given by

$$P = 6V_{DD}I_{bias}. \tag{7.10}$$

The factor of 6 in this equation comes from the six branches that carry the bias current in Figure 7.4. This factor can be reduced by appropriately scaling the bias network transistors to carry only fractions of the required bias current for the translinear circuit, but the minimum value of the coefficient must be larger than 3 (for the two legs of the translinear circuit block, as well as the output current mirror, all of which must carry the full $I_{mathrmbias}$). In this work, all transistor were scaled to carry the same bias current. To reduce the power of the voltage-to-current converter, it is desirable to reduce I_{bias}. The energy required per conversion is given by

$$E_{conv} = 6V_{DD}I_{bias}T_{sample} = \frac{6V_{DD}I_{bias}K}{f_{osc}} = \frac{6V_{DD}I_{bias}K}{(V_{high} - V_{low})K_{vi}K_{if}}. \tag{7.11}$$

Therefore, to minimize the energy per conversion in the voltage-to-current converter, we want to make the parameter K_{vi} as large as possible, and I_{bias} as small as possible. K_{vi} can be made large by using a small value of R in the translinear amplifier. However, since we are limited by the fact that $I_{bias}/2$ must be larger than i, we can not increase K_{vi} arbitrarily, without also increasing $I_{bias}/2$. Thus, the best we can do is to try make i a large fraction of I_{bias}. However, as discussed earlier, as i becomes a larger fraction of I_{bias}, the linearity of the voltage to current converter deteriorates, which may decrease ADC

[2]For the analysis considered here, we assume that the ADC can be turned off (power gated) when a conversion is not taking place. For simplicity, such mechanisms were not implemented in the experimental prototype, but it can be done by the addition of just a few transistors.

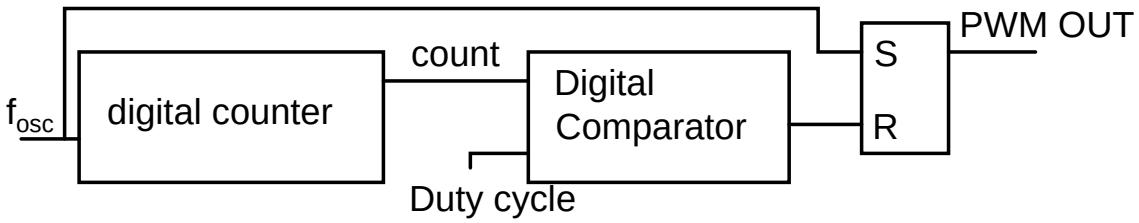


Figure 7.10: *High-level schematic drawing of counter-based digital pulse-width modulator implementation.*

overall performance. In a given design, there is thus a certain i to I_{bias} ratio that gives the best power efficiency to linearity trade-off. As the voltage-to-current converter stage is scaled with this ratio constant, the energy per conversion is yet again constant. This is because as the absolute values of I_{bias} and i are increased, the sampling time T_{sample} can be correspondingly reduced, leading to a constant energy per conversion.

We thus note that for a given design where the power/linearity trade-off has been made, the energy required per conversion for this ADC architecture is constant. However, at very low power levels the power losses that have been ignored in this analysis (e.g. counter leakage currents, Schmitt-trigger dynamic and static power consumption) will become large enough such that their contributions must be taken into account.

7.2.8 Digital Pulse Width Modulator

The digital pulse width modulator (DPWM) of Fig. 7.10 is used to convert the digital code held in the accumulator (of Fig 7.9) to a series of pulses of the correct width to drive the gates of the power MOSFETs. The design is a counter-based solution, which ensures monotonicity and achieves good linearity, while keeping the implementation area low. Because of the relatively low switching frequency and DPWM resolution (6-bit), the power consumption of the DPWM can be kept low. At a switching frequency of 1 MHz, the estimated (from simulation) power consumption of the DPWM is 0.45 mW.

Shown in Figure 7.11 is a detailed schematic drawing that illustrates how the DPWM

is implemented, while making use of many of the components already designed for other part of the converter. The frequency is controlled by an external bias current, which is fed to a current-to-frequency-converter (using the same design as the current-controlled oscillator of Figure7.6 which was previously designed for the ADC). The resulting high ($\sim$ 128 MHz) frequency clock is fed to a 9-bit counter (using the same design as the digital counting stage for the ADC, shown in Figure 7.8). The 7 lowest order bits from the counter is connected to a 7-bit comparator, which compares the counter output to the 7-bit duty code that the MPPT logic outputs. When the count exceeds the duty value, the output of the comparator is triggered, which resets the flip-flop. Whenever the counter has counted up to 128 (COUNT<7> goes high), the counter is reset, and the edge-triggered flip-flop sets its output (Q) high, so that the PWM output is high until the 7-bit comparator resets it again.

The counter-based DPWM is easy to implement, takes up a small amount of die area, and provides inherent monotonicity. The drawback of the architecture is high power consumption as the frequency and resolution is increased. The DPWM was implemented in 0.35 μm CMOS technology, which enables the relatively small digital blocks to operate at frequencies well in excess of what is required here. It is expected that the DPWM implementation of Figure 7.11 can operate with 7-bit resolution at switching frequencies well in excess of 10 MHz, with a power consumption penalty (e.g. 10x higher than what was observed in this work).

7.3 Power Stage

The power stage described in this section was designed with the help of my colleague Wei Li, who also provided assistance in the layout of the pad-ring for the final TPV MPPT chip.

The power stage of the TPV tracking system is an integrated synchronous dc-dc boost converter. In the maximum power operating condition, it converts 0.8-1.3 V from the output

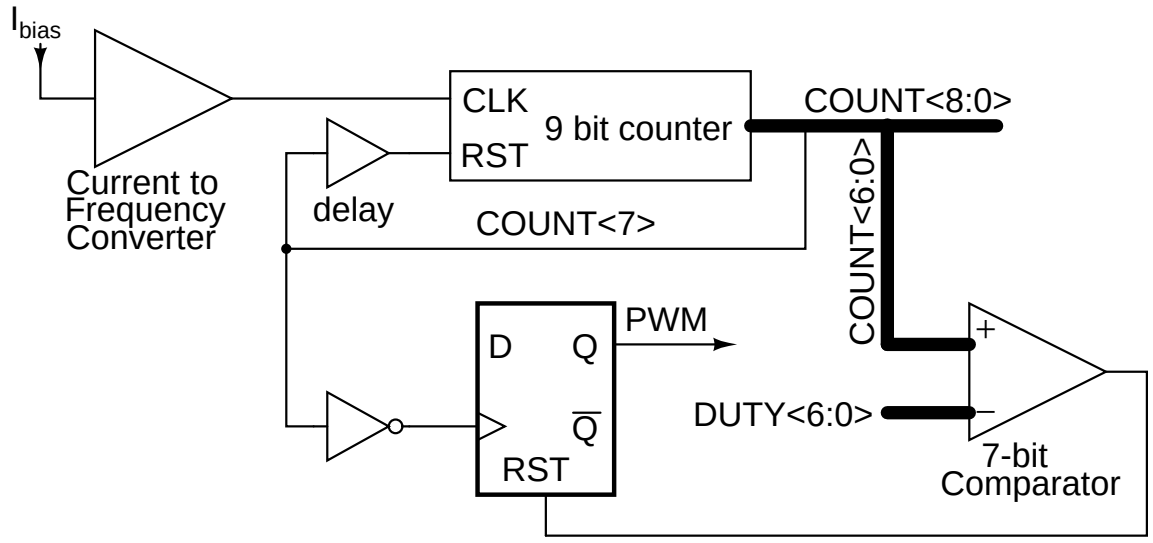


Figure 7.11: *Detailed schematic drawing of DPWM implementation.*

of the TPV cell to 3.6-4.2 V for battery charging. A TSMC 0.35 μm thick oxide device process is used to provide 5 V blocking voltage capability. Since the maximum power output of the TPV cells is approximately 300 mW (as seen in Fig. 5.3), the device sizes and gate driver taper factor are optimized for this power level, to balance the capacitive switching loss and conduction loss [52]. The IC power stage is designed to be flexible, enabling operation at switching frequencies to beyond 1.5 MHz, and with either hard-switching or high-ripple soft-switching operation [53].

Since the converter will operate at the optimal power output condition of the TPV unit most of the time, the system only needs to operate efficiently over a relatively narrow power range. This opens up the possibility of using high-ripple zero-voltage-switching (ZVS) soft-switched operation. Fig. 7.12 shows sample soft-switching waveforms of this mode of operation.

If the inductor current i_L has peak-to-peak current ripple over 200% of the average current, soft-switching can be implemented [54–56]. After the high-side device is turned off and before the low-side device is turned on, the inductor current will discharge the drain-source capacitance of the low-side device and charge the capacitance of the high-side device.

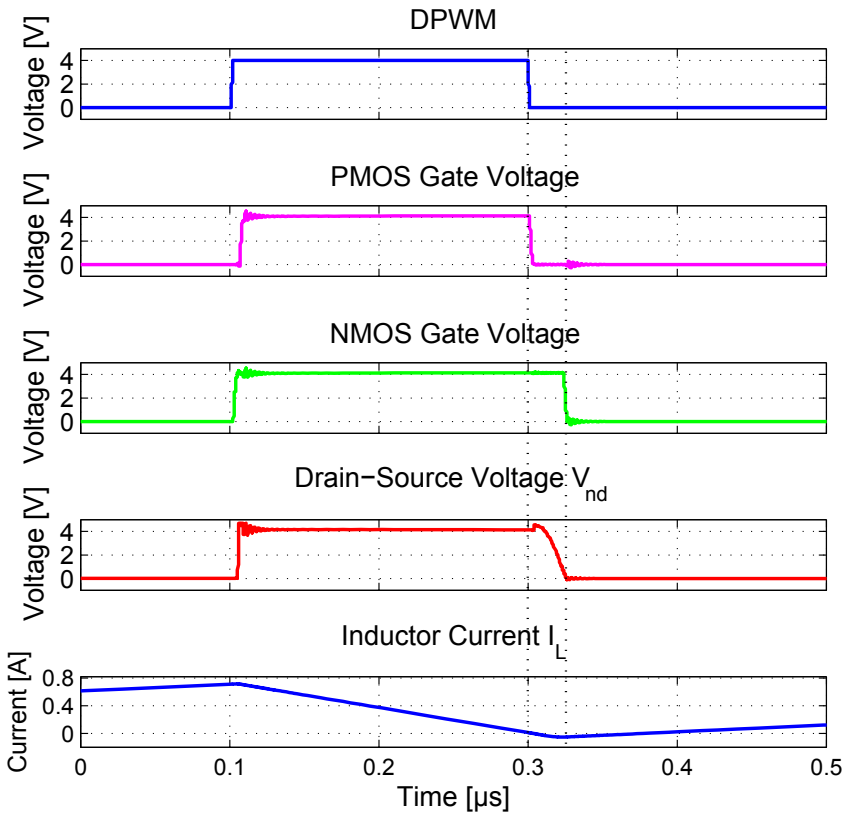


Figure 7.12: *Simulated waveforms to illustrate soft-switching operation. V_{in} = 0.9 V, V_{out} = 4 V and P_{out} = 300 mW.*

The converse can likewise be made to happen on the other transition. By adjusting the dead-times between the switching of the two devices carefully, ZVS can be achieved at the turn-on transition for both devices.

Self-adjusted digital dead-time control circuitry is introduced to provide enhanced performance in soft switching. This self-adjusted dead-time control circuit has several advantages, including simplicity, low power consumption, fast response to changes in operating condition, and the ability to extend the soft-switching operation range as compared to fixed dead-time control. Fig. 7.13 shows a simplified schematic of the dead-time control circuit. The self-adjusted dead-time circuit controls the dead-time based on the voltage level at the drain of the low-side device, V_{nd}. The low-side device will only be turned on once voltage V_{nd} drops below the dead-time logic threshold. Likewise, the high-side device will only be turned on after voltage V_{nd} rises above the dead-time threshold level for the high-side device turn-on. A Schmitt trigger is used to set the upper and lower switching threshold voltages and also provide stability improvement.

To address operating conditions when ZVS switching will not occur, an additional 28 ns dead-time limit is set. This enables hard-switching operation to be employed if desired, and also ensures correct operation under conditions (such as transients) that disrupts soft-switching operation. (This window size is determined by the longest required dead-time for ZVS with minimum inductor current ripple.)

The power stage design is compatible with both soft and hard switching operation. The final optimized size (device width) for the NMOS transistors is 118000 μm, and for the PMOS transistor is 121000 μm. A taper factor of 11 is chosen for the gate drivers to balance the gate drive loss and switching loss of the power devices. The dead-time control logic and gate drivers are powered by the output of the converter.

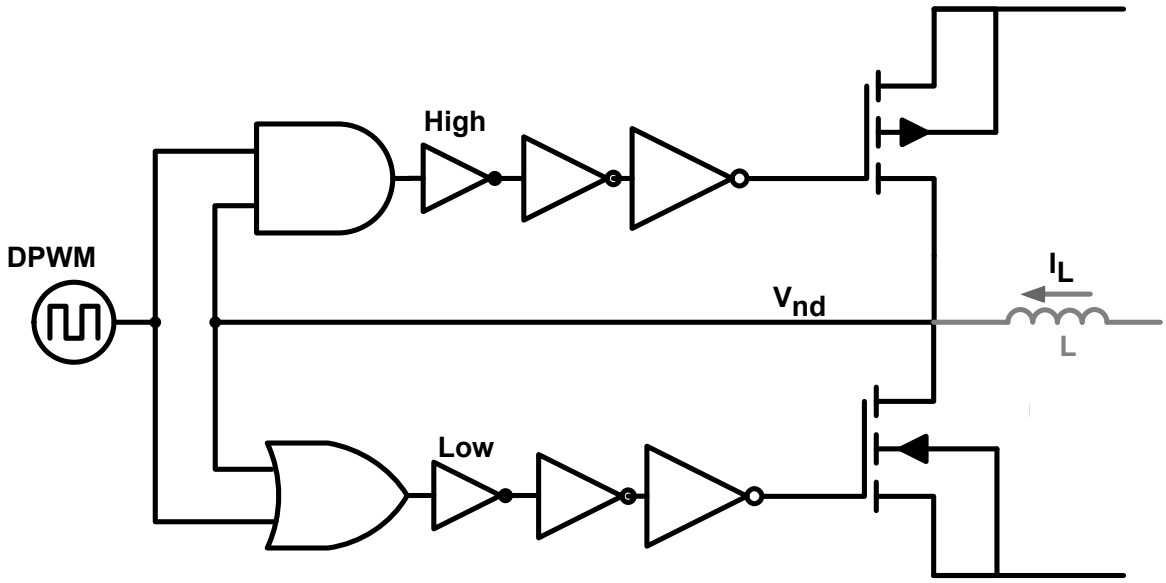


Figure 7.13: *Simplified schematic drawing of the self-adjusted dead-time control circuit used to achieve ZVS. Additional logic ensures switching even when soft switching is not realized.*

7.3.1 Experimental Results

The TPV tracking system was fabricated in a TSMC 0.35 μm CMOS process and mounted in a QFN40 package. An annotated die photo of the converter is shown in Fig. 7.14, and approximate silicon area breakdown is presented in Table 7.1. The converter specifications are shown in Table 7.2. Table 7.3 provides a listing of the passive component used in the design (notice that different inductors were used in size/efficiency analysis). An annotated photograph of the test-board where the TPV converter chip was mounted is shown in Figure 7.15.

Power Stage Characterization

Shown in Fig. 7.16 are experimental waveforms of the converter which illustrate soft-switching operation using a 0.9 μH inductor with 11-120-P material, and operating at an input voltage of 0.9 V, an output voltage of 4 V, and an output power of 300 mW. Hard-switching waveforms are also as would be expected. Measured converter efficiencies

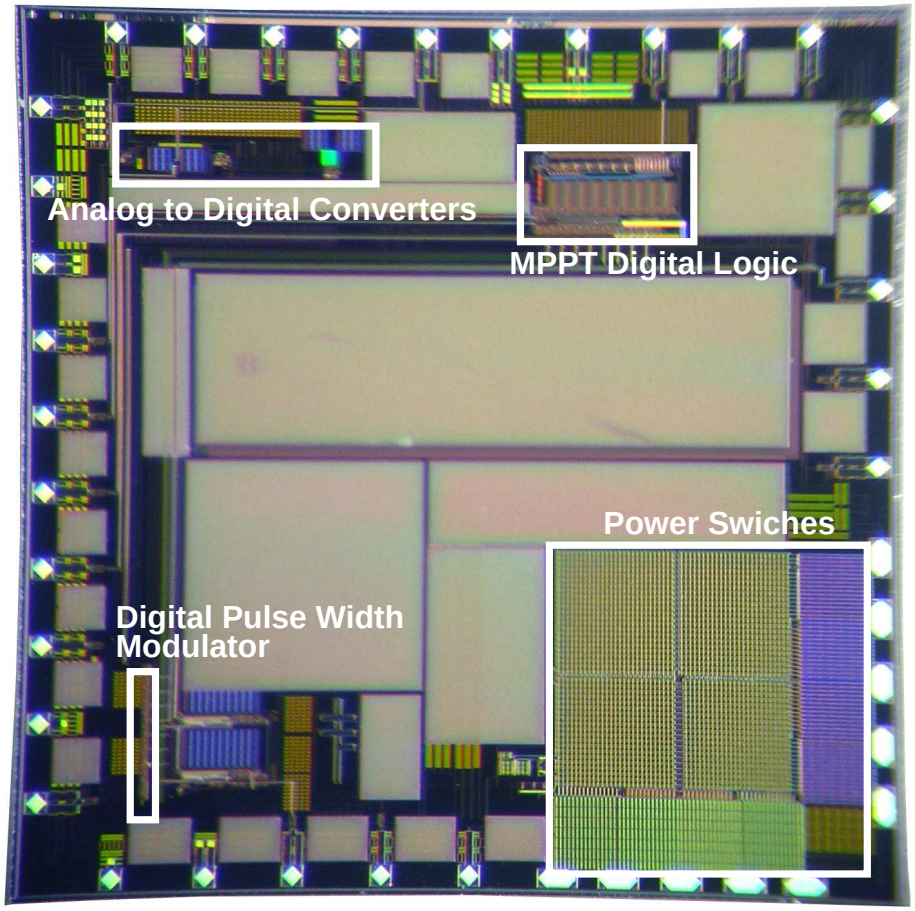


Figure 7.14: *Annotated die photo of the maximum power point tracker implemented in a 0.35 µm CMOS process. Total die area is 4x4 mm, with approximately 1.16 mm^2 of active area (see Table 7.1 for more details regarding area breakdown).*

Table 7.1: *Converter Area Breakdown*

Component	Area [mm^2]
ADCs (2)	0.083
Analog Bypass Capacitors (oversized)	0.131
MPPT Logic	0.192
DPWM	0.031
Digital Decoupling Capacitors (oversized)	0.134
Power Devices	0.752
Gate Drives	0.061
Dead-time Control	0.040
Output Capacitor	1.21
Total Active Area	1.159
Total Capacitor Area (oversized)	1.475

Table 7.2: *Converter Specifications*

Input Voltage Range	0.8-1.3 V (1 V Nominal)
Output Voltage Range	3.6-4.2 V (4 V Nominal)
Nominal Output Power	300 mW
Switching Frequency	500 kHz
Converter Peak Efficiency	95.4%
Tracking Efficiency	>98%

Table 7.3: *Component Listing*

Device	Model	Value	Manufacturer
L	SER1360-103KL	10 μH	Coilcraft
C_{OUT}	0603, X5R	2 x 1 μF	Murata
C_{OUT}	0402, X5R	4 x 0.1 μF	Murata
C_{IN}	0603, X5R	4 x 1 μF	Murata

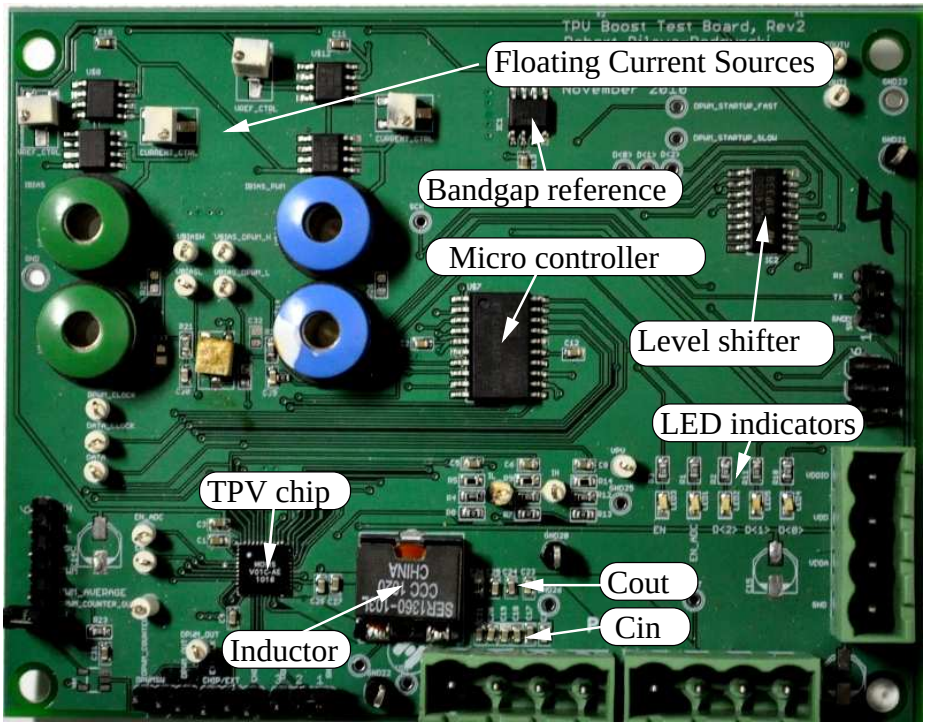


Figure 7.15: *Annotated photograph of printed circuit board used for testing the TPV converter. A detailed schematic of the test board is provided in Appendix G.*

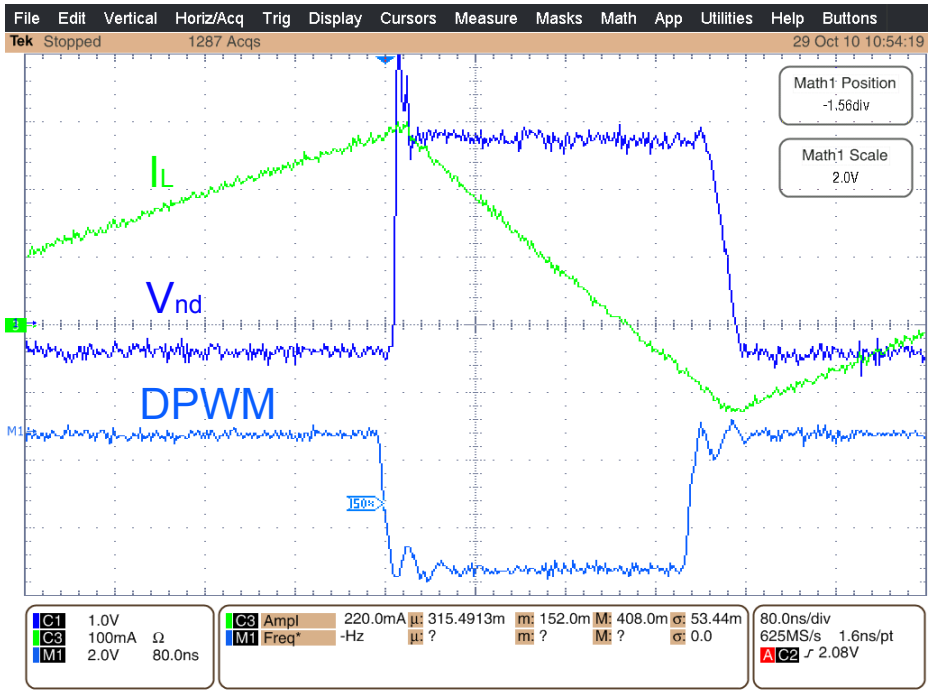


Figure 7.16: *Experimental waveforms of the power stage drain voltage and inductor current, as well as the DPWM signal. The dead-time control circuitry adjusts the timing of the gate signals to achieve ZVS. In this example, the input voltage is 0.9 V, the output voltage is 4 V, the inductor value is 0.9 μH, and the output power is 300 mW.*

for various power and voltage levels are shown in Fig. 7.17 for one power-stage implementation under hard-switched conditions. It can be seen that the converter has a peak efficiency of 95.4% with V_{in} = 1.3 V, V_{out} = 4 V and output power of 300 mW.

With the low output power and requirements of small size and high efficiency in this work, inductor size and converter performance trade-offs become important, especially as inductor size dominates the overall size of the converter (for most design conditions). Fig. 7.18 shows the measured converter performance for different frequencies, inductor designs and operating modes with a nominal input voltage of 1 V, output voltage of 4 V and output power of 300 mW. A picture of some of the inductors used in the experimental measurements is shown in Fig. 7.19. For reference, the TPV converter chip and a US penny are also shown in the picture, as well as a cm-scaled ruler.

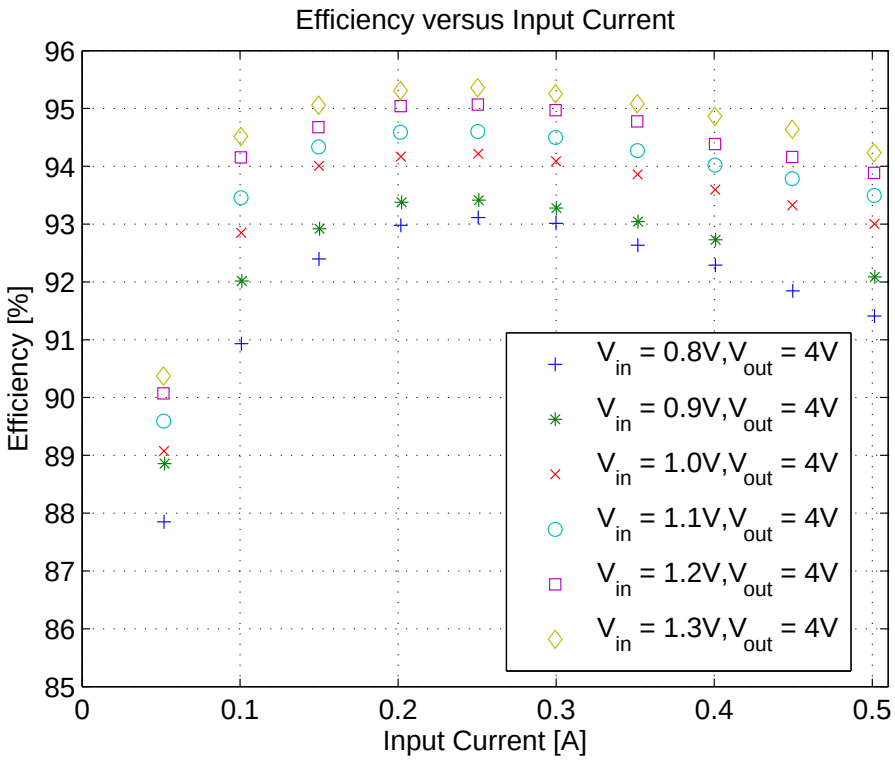


Figure 7.17: *Plot of measured power stage efficiency in hard-switching operation at $f_{sw} = 500$ kHz. Input capacitance is 4 μF, output capacitance is 4.8 μF, and the power inductor is 8 μH wound on a P9/5 3F3 core with 3 $\times$ 28 AWG.*

As part of evaluating our system, we undertook a detailed study of achievable efficiency as a function of inductor size, switching frequency and operating mode (hard switching vs. soft switching). This included modeling of system losses for numerous designs (using loss models of commercial inductors along with detailed models of our own converter IC) and experimental validation of a subset of designs. We considered operation at frequencies from 500 kHz to 1.5 MHz, with inductance values selected for both soft- and hard-switching operation.

At higher operating frequencies, designs can effectively use either high-permeability core materials or low-permeability core materials. An advantage of some low-permeability materials (e.g., NiZn ferrites) is that the effect of core loss can be reduced to an extent, benefiting the use of high-ripple soft switching. As illustrated in Fig. 7.18, at the lowest inductor volumes tested (≈ 80 mm^3), the achieved experimental efficiencies with soft switching and hard switching were very close. (The soft-switched design operated at 1.5 MHz, while the hard switching design of comparable efficiency operated at a reduced frequency of 1 MHz; considering only 1.5 MHz operation, soft switching was superior by more than 2% in efficiency.) However, our models suggest that with an appropriate customized low permeability core material (relative permeability of 20-30), a soft-switched implementation could perform significantly better than a hard-switched implementation at frequencies above 1 MHz. (Our experimental results were limited to available commercial cores, and did not include an appropriate custom core material.)

Figure 7.20 shows calculated converter efficiency as a function of inductor size for a wide variety of commercial cores and inductance values, for both hard and soft switching. Figure 7.21 overlays these calculated results with the experimental results from Fig. 7.18. It can be seen that the measured experimental results all fall in to the range expected from model calculations. Consequently, Figs. 7.18, 7.20, and 7.21 show the frontier of inductor size vs. conversion efficiency, at least for the types of core materials and inductor designs evaluated.

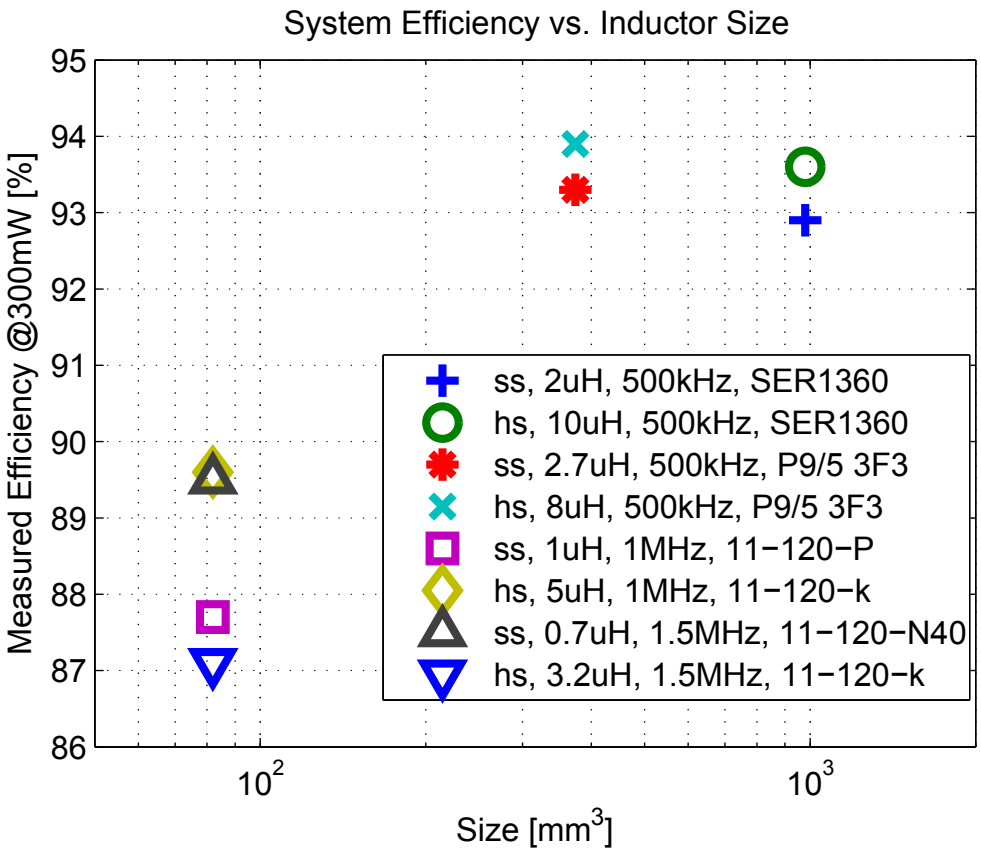


Figure 7.18: *Measured converter efficiency for various inductor sizes and values. Inductors are wound on selected available cores. "hs" stands for hard-switching and "ss" stands for soft-switching. Operation is for V_{in} = 1 V, V_{out} = 4 V, and P_{out} = 300 mW.*

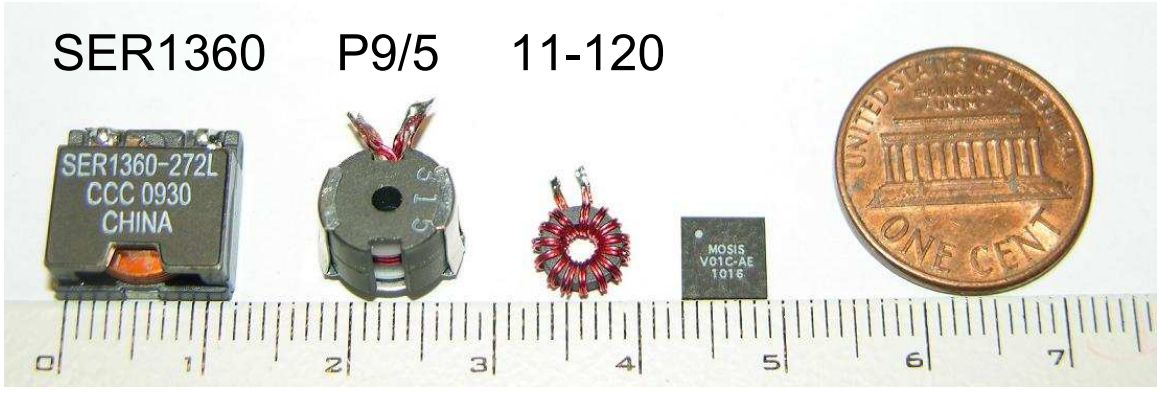


Figure 7.19: *Picture of some inductors used for the experiment. The packaged TPV converter chip and a US penny are shown for size reference, together with a cm-scale ruler.*

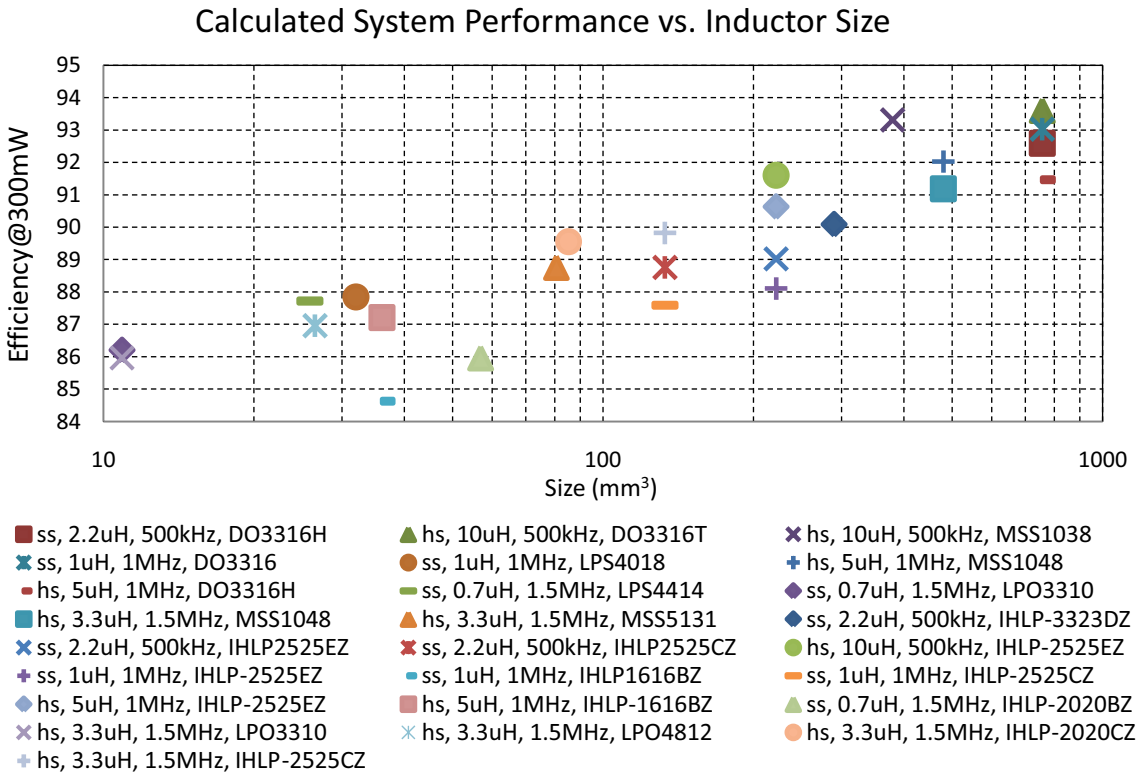


Figure 7.20: *Calculated converter efficiency versus inductor sizes. All inductors are commercially available from Coilcraft and Vishay. "hs" stands for hard-switching and "ss" stands for soft-switching.*

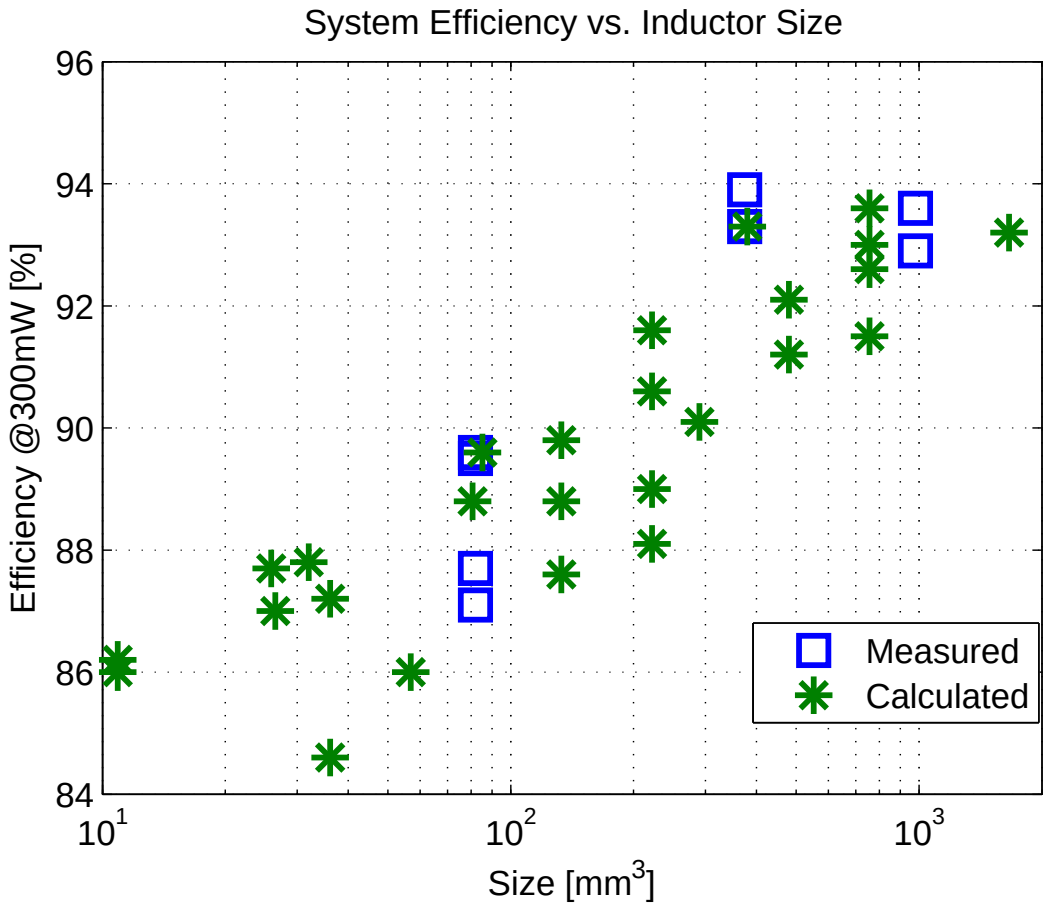


Figure 7.21: *Measured and calculated converter efficiency versus inductor sizes. The measured results agree well with calculated values. Operation is for V_{in} = 1 V, V_{out} = 4 V, and P_{out} = 300 mW.*

7.3.2 Tracking Performance

To evaluate the performance of the peak power tracker under repeatable conditions, the converter was attached to two crystalline Silicon series-connected solar cells illuminated by a halogen lamp to produce I-V characteristics similar to that produced by the micro-burner. This enabled characterization of the converter without the added complexity of the micro-reactor dynamics.

Shown in Fig. 7.22 are plots of power versus time, illustrating the peak power tracker performance. In the top plot, the tracker is started with a duty cycle set to operate at a voltage that is higher than V_{mpp}. The bottom plot shows the corresponding data when the starting voltage is set below V_{mpp}. In both cases, the converter correctly finds the maximum power point and tracks it to within the resolution of the duty cycle command and the noise in the power measurement. The tracking efficiency, η_{track}, is a measure of how precisely the MPP is tracked, and is given by: $\eta_{\text{track}} = \frac{\langle P_{in} \rangle}{P_{MPP}}$, and is above 98% in both cases in Fig. 7.22.

Fig. 7.23 shows a plot of converter input power versus input voltage, which illustrates the I-V characteristics of the source, which is similar to the plot shown in Fig. 5.3. In addition, the discretization of the input voltage illustrates the finite achievable voltage step-size. The minimum step-size is limited by the resolution of the digital pulse-width modulator.

7.3.3 Conclusions

We have presented a distributed MPPT architecture for use with a portable TPV power generator. By employing intelligent, local, tracking of the MPP, the overall energy of the system can be increased. A discrete power converter implementation has been designed and tested with the full TPV power generator, showing efficient power conversion and tracking of the optimum operating point of the TPV cells. To address the high control losses and non-optimum power transistor sizes associated with the discrete prototype, a fully

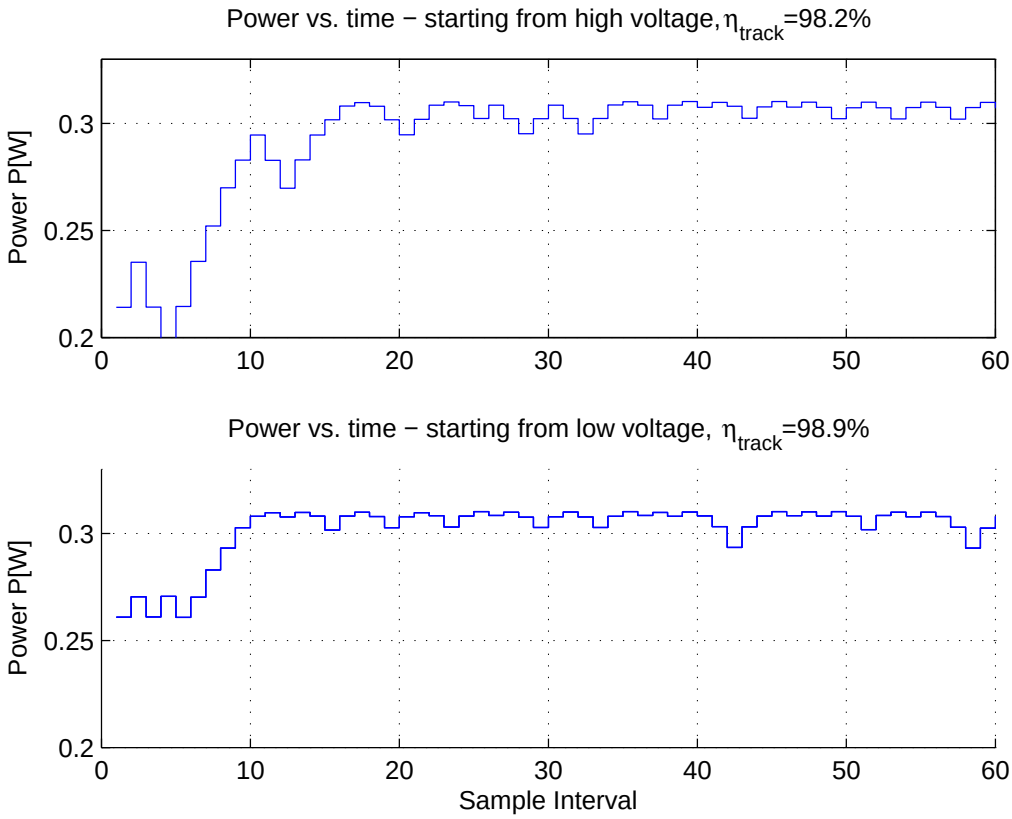


Figure 7.22: *Time-domain plot of the converter input power, showing maximum power point tracking.*

integrated design was developed in 0.35 μm CMOS technology. Custom low-power voltage and current sensing techniques were developed, together with a low-power conting-based ADC that is suitable for loss-less current sensing. A digital perturb and observe algorithm was implemented in CMOS logic, along with a counting-based DPWM and integrated gate drive circuitry and power transistors. We perform a detailed performance comparison for a variety of inductors and frequencies, and combine measured and modelled data to map out the possible size and efficiency trade-offs for the power stage. Finally, we show experimental results with excellent tracking of the MPP, along with high conversion efficiency and very low control losses.

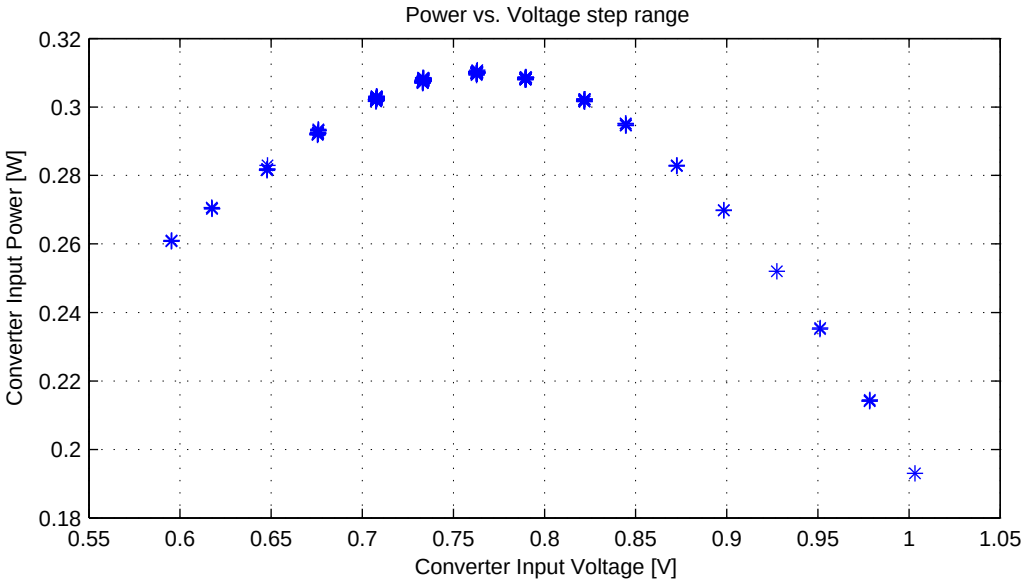


Figure 7.23: *Plot showing the power and voltage dependence of the experimental power source, using the same data as that which generated Fig. 7.22. The voltage step-size is limited by the resolution of the digital pulse-width modulator.*

7.3.4 Future Work

Here we outline a few areas that could benefit from additional research:

Low-Power High Frequency DPWM

In this work we employed a simple counting based DPWM for its simplicity and small area. As digital control becomes more prevalent in power electronics, more power efficient DPWM designs will be required. Many alternative DPWM implementations trade-off die-area for power loss, making a high-resolution, high-frequency, low-power DPWM take up considerable size. There is thus room for further innovation in this area to develop compact and efficient DPWM solutions digital control of power electronics.

Theoretical Analys of Optimum Resolution of PWM and Sensing

While much work has been done to come up with different algorithms for performing maximum power point tracking, much less attention has been paid to the important area of how to implement these algorithms. Particularly in a fully integrated solution, where one has complete control of the resolution of the DPWM and the ADC, it is important to allocate the control power budget to the area where it provides the most benefit. Since once can easily trade-off ADC resolution/speed and power consumption, it is important to quantify what are the appropriate design parameters. A theoretical analysis of this trade-off would be highly valuable for many designers of MPPT circuitry.

Power MPPT from Low-Voltage Input

In this application, the MPPT is powered from the 4 V output voltage (owing to the existence of a voltage buffer on the output). In many other applications, such a voltage buffer may not exist, and the circuit needs to be powered from the low-voltage input. A boot-strap circuit that starts the circuit up from a low (< 1 V) input voltage would therefore be desirable.

Chapter 8

Solar Photovoltaic Applications

8.1 Motivation

With rising world-wide energy demands and soaring prices of fossil fuels, interest in renewable energy sources has increased. Among these, solar photovoltaic (PV) energy has seen a rapid growth in the last few years, resulting in decreased prices of PV cells as production capacity increases at a fast pace. As the PV cell prices decrease, the cost of the power electronics required to extract the maximum power of the PV modules and to interface the PV system to the grid is becoming a larger part of the overall system cost [57]. Much attention has therefore been given to the development of power electronics that enable a cost reduction of the overall system. In addition, much research is focused on increasing the efficiency of the power processing stage, as well as on improving the power yield of the overall system [58, 59]. This chapter investigates techniques for implementing low-voltage distributed power electronics in a solar photovoltaic system, and explores the achievable system output power improvements under real-world conditions.

8.2 PV Characteristics

Fig. 8.1 shows a schematic drawing of a PV system. The DC output voltage of the solar array is controlled by the maximum power point tracker (MPPT) to ensure optimum power extraction from the solar array. The maximum power point (MPP) changes with temperature and irradiation, so the MPPT dynamically adjusts the operating point of the array

to track these changes. The DC output of the MPPT is then fed to an inverter, which connects to the grid.

Unfortunately, the amount of power that can be extracted from MPP operation of a series- or parallel-connected set of PV cells may be substantially lower than the power that could be extracted if each cell were operated at its individual maximum power point. As shown in Fig. 8.2 (top), the output power is significantly higher when a PV cell receives full sunlight (1 kW/m^2) than when it receives 25% of full sunlight. Fig. 8.3 shows a drawing of a PV module, which typically consists of 36 to 72 series-connected PV cells. Because the cells are all connected in series, the module output current is limited by the weakest cells. The output current of each cell varies strongly with irradiation, as can be seen in Fig. 8.2 (bottom). The current also changes with manufacturing lot (sometimes also within a lot), temperature and age [61], so cell-current mismatch is a common phenomenon which reduces power yield. The most severe effects are seen when PV modules experience different irradiation levels across the module (typically due to partial shading). The shaded cells are reverse biased by the other series-connected cells, and can be driven into reverse conduction, acting as power loads, wasting power and incurring damage through localized dissipation at hot spots.

To prevent damage to the shaded cells by reverse current, bypass diodes are commonly employed, as shown in Fig. 8.4. In practice, one diode per 18 to 24 cells is typically used. When shading of one or more cells causes the bypass diode to conduct, the section of cells that is bypassed contributes no power to the output.

The importance of these effects on design should not be underestimated. Indeed, field

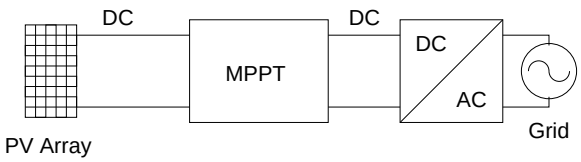


Figure 8.1: *Schematic drawing of a PV system.*

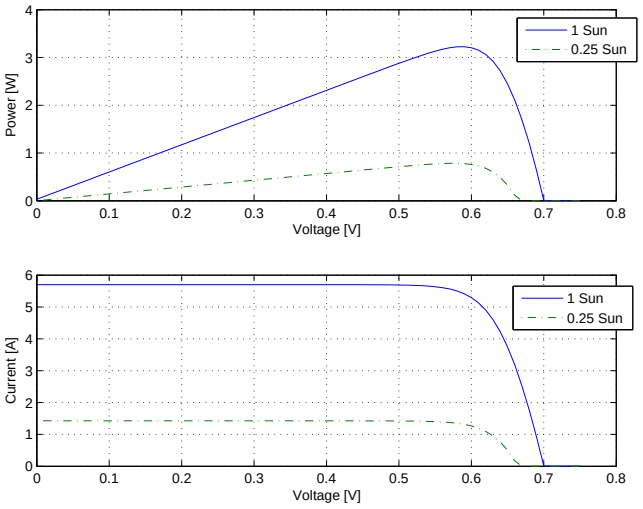


Figure 8.2: *Electrical characteristics of a single solar cell under varying irradiation levels (adapted from [60]). Peak output current (bottom) and power (top) is significantly reduced at lower irradiation levels.*

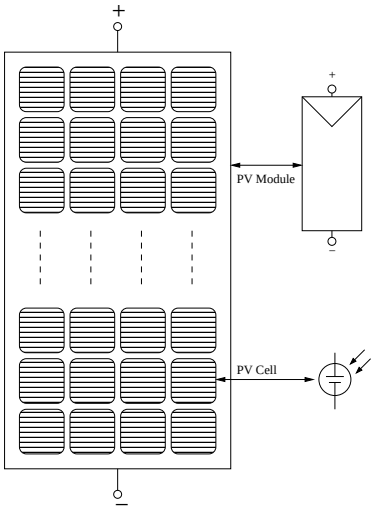


Figure 8.3: *Schematic drawing of PV module.*

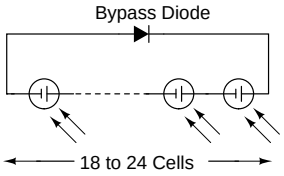
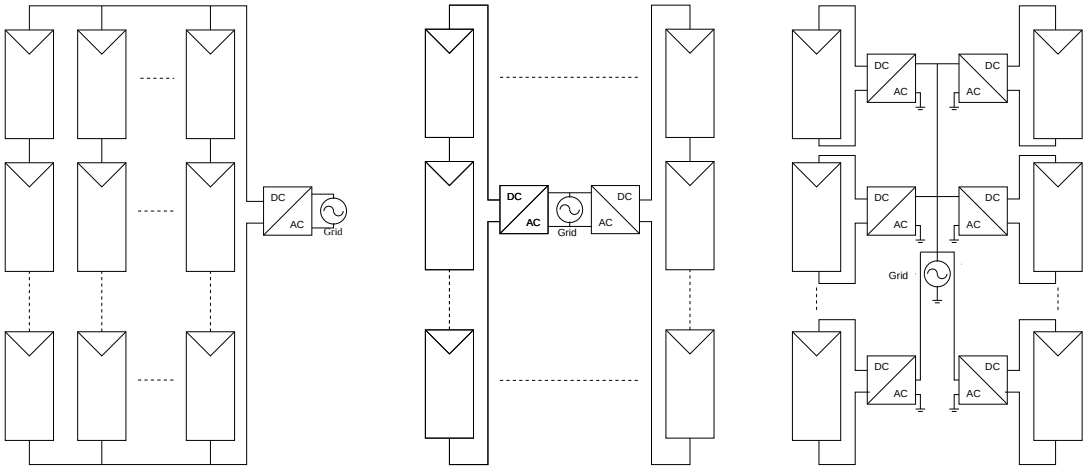


Figure 8.4: *Schematic diagram illustrating the use of bypass diodes to prevent damage to shaded cells.*

results from early residential photovoltaic installations incorporating long strings of cells showed a significantly lower total power yield than expected [62]. A large portion of the yield reduction can be attributed to the problem of partial shading of the solar panel from obstructions such as clouds, power lines, utility poles, trees, and dirt.

8.3 PV System Evolution

The problem of partial shading has led to the evolution of PV system architectures illustrated in Fig. 8.5. Most early installations used a central converter, as shown in Fig. 8.5a. In this architecture, a number of PV modules are connected in a series string to achieve a



(a) Central inverter. **(b)** String inverter. **(c)** Module inverter.

Figure 8.5: *Schematic diagram of PV system evolution.*

high output voltage. Multiple groups of these strings are connected in parallel to increase the power output. The advantage of this technique is the ability to use a single high-voltage, high-power central inverter that can be made very efficient. The disadvantage is that since all of the strings are constrained to operate at the same output voltage, some strings will not operate at their maximum power points (MPP) in case of uneven irradiation of the modules, or mismatched cells/modules. This can lead to large reductions in power yields from what is theoretically possible.

To mitigate problems with MPP mismatch, the string inverter concept was developed (Fig. 8.5b), in which each series-string of modules is connected to its own inverter. This enables each string to be operated at a voltage that coincides with its MPP, and thus improves power yield. One disadvantage of this approach is the need for several inverters of lower power than the central inverter system. This typically leads to a less efficient and more expensive power converter system [37]. Although each string of PV modules is operating at its MPP, total output power is still constrained by modules with reduced output capability. In the case where a module is sufficiently shaded, its bypass diodes conduct, and it absorbs power. In addition, shading of individual modules in the string can lead to a situation where the MPP tracking system settles on a local optimum power point that is less than

the global MPP [36].

To further improve power extraction, there has been movement towards architectures that provide MPP tracking at the individual *module* level [58,63]. For example, the module integrated inverter, shown in Fig. 8.5c, uses one grid-interfaced inverter per module, which enables each module to operate at its own MPP [63]. The disadvantage of this approach is the increased number of inverters, each of which operates at low power (e.g., 100-200 W) and large voltage transformation, leading to higher total system cost and lower conversion efficiency.

The evolution from tracking parallel strings to individual strings and finally to tracking individual modules stems from the desire to improve power yield by operating the PV cells as close to their MPP as possible. However, even when per-module tracking is used, not all of the available power is captured, since not every cell is operated at its MPP. A simple example is illustrated in Fig. 8.6, which shows a typical module with 72 cells and 3 bypass diodes, where a single cell is shaded (this could happen for instance by dirt accumulation, fallen leaves, or shading by a power line). The shaded cell causes the bypass diode to conduct, and all 24 cells are bypassed, contributing no power to the output. The total output power that can be extracted is thereby reduced by 33%. Power conversion systems providing cell-level power-point tracking have been proposed precisely to address this issue (e.g., [64,65]). However, the methods proposed to date are inherent costly and complex; they would be practical only in highly specialized applications. This thesis seeks to investigate alternative implementations where distributed power electronics integrated into the solar panel can improve overall energy capture.

8.4 Distributed Power Electronics Solutions

In order to increase the PV system output power in the face of partial shading and other mismatch-inducing phenomena, distributed power electronics such as those shown in Fig. 8.7

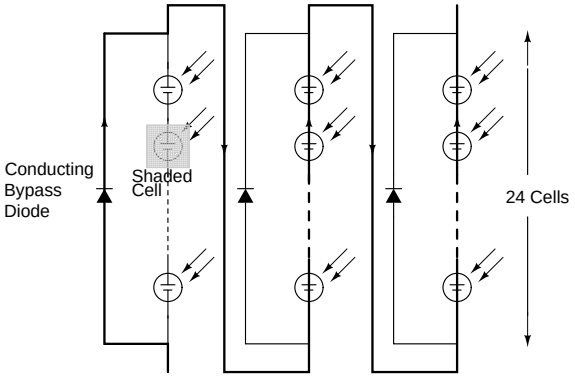
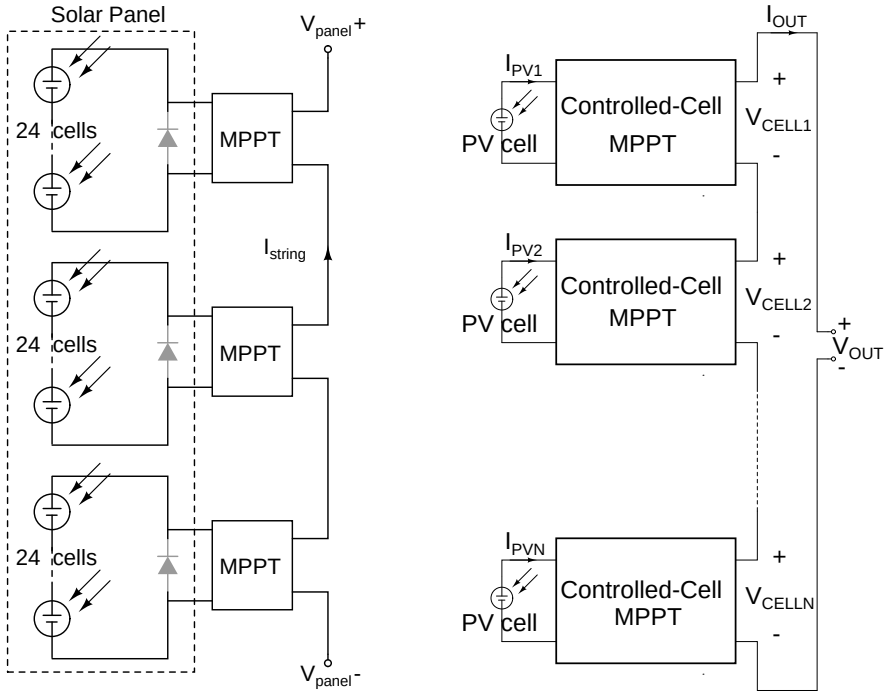


Figure 8.6: *Schematic drawing illustrating the effects of shading of one cell. The left-most bypass diode is conducting, and none of the power from the bypassed cells is contributed to the total output power, which is thereby reduced by 33%.*

can be employed. Fig. 8.7a shows an implementation where three MPPTs are connected to a single 72-cell panel. In this case, the inputs of each MPPT is connected to a sub-module of the PV panel[1], lending this technique to easy implementation in existing panels, where the connections to the sub-modules are accessible from the junction box. The output of the MPPTs are connected in series, which provides a large output voltage, while still enabling each sub-module to operate at its own unique MPP. However, the limitation of this implementation is that only mismatch between each set of 24 cells (a sub-module) can be compensated for. Therefore, any current mismatch between different cells in a given sub-module can not be mitigated using this approach. Despite this limitation, the distributed MPPT implementation of Fig. 8.7 can achieve increased energy capture compared to conventional techniques, as our experimental measurements show.

Shown in Fig 8.7b is an implementation where each solar cell is connected to an individual MPPT. In this case, the maximum available energy of the entire system can be captured, as each cell operates completely independently of the others. While the approach of Fig 8.7b has a much larger number of power converters, it should be noted that due to the low operating voltage of each cell (<0.6V), the power electronics can be implemented in a

[1]In this thesis, we will refer to all cells in a PV panel that are connected to the same bypass diode as a sub-module.



(a) Sub-Module MPPT **(b)** Single Cell MPPT.

Figure 8.7: *Distributed MPPT solutions to increase energy capture in a PV panel.*

low-voltage CMOS process, enabling high frequency operation with correspondingly small passive component size and low cost.

In this thesis, we will develop general control techniques and power converter design for implementing the distributed MPPT architecture of Figure 8.7, suitable for per-sub-module or per-cell tracking. Our experimental prototype will feature a sub-module tracking implementation, where care has been taken to make the approach useful for cell-level tracking as well.

8.4.1 System Advantages

The proposed architecture of Figure 8.7 is compatible with the use of efficient, centralized grid-tie inverter systems, and may also help increae the overall energy capture of module-level inverters (also known as micro-inverters), which today do not capture all the available energy in the case of partial shading of single panels. Moreover, the proposed system offers many benefits in terms of increased power yield, reduced cost, and improved reliability and flexibility. Here we discuss some of these potential benefits.

Increased Power Yield

Because the system can extract the maximum possible power from each cell or sub-module, the total power yield is greater than that of conventional systems, whose output power is limited by the weakest cell/sub-module. In installations where partial shading is common (e.g. building-integrated PV systems and residential installations) the resulting increase in power yields will be the most dramatic. However, power yield will also increase for PV installations where shading is not a big concern, since the total output power in the distributed MPPT system is not limited by cell or sub-module mismatch, differential ageing, and temperature variation, all of which reduce the power yield of systems used today. Another important aspect that increases the power yield is the ability to use a central,

high-voltage, high-power inverter which can be made more efficient than many smaller micro-inverters. The reduced power processing losses will thus contribute to an additional increase in power yield.

Reduced Cost

The proposed system has potential to decrease the cost of both manufacturing and installation of PV systems:

Manufacturing Cost

In order to obtain maximum power output per module, today's PV manufacturers take great care to place matching cells (with identical electrical characteristics) in each module. Each cell is measured and sorted into matching performance bins [66], and various algorithms are used to determine which cells are combined into a module [67]. With cell or sub-module level distributed power electronics, less strict binning is required, and a looser manufacturing tolerance can be used.

In addition, the system could have a substantial impact on the manufacturing cost of thin-film photovoltaic modules (e.g. amorphous silicon, cadmium telluride and copper indium gallium diselenide). These PV technologies are being pursued because of their material costs are potentially much lower than those of crystalline silicon. Thin-film modules are typically manufactured by depositing a thin layer of material onto a large area substrate. The panel is then scribed by a laser, which electrically separates the different parts of the panel into smaller cells. To produce a useful output voltage the cells are connected in series. Thus, for thin-film modules, there is no way to sort the cells by performance and accomplish current matching similar to that of crystalline modules. Therefore, in thin-film manufacturing, much care has to be taken to produce a very uniform deposition of material, which leads to increased cost and complexity [68]. The system proposed here would enable each sub-

module of cells to contribute its maximum achievable power, regardless of its performance relative to neighboring sub-modules. It is therefore expected that a thin-film panel using the distributed MPPT implementation could be manufactured with less stringent uniformity requirements, which would lead to reduced manufacturing cost.

Installation Cost

Because of the severe reduction in output power due to partial shading of PV modules, much care is typically taken at the time of installation to orient the modules in a system to minimize the negative effects of shading. In addition to long-term solar irradiation measurements, software can sometimes be used to achieve the optimum placement of PV modules [37]. Since partial shading does not have the same detrimental effect on output power in our proposed system, less time and effort need to be spent on achieving the optimum configuration of modules. As building-integrated PV systems become more prevalent, it is expected that the increased flexibility offered by the cell and sub-module based MPPT will greatly simplify the planning and installation process. Today, it is possible to choose the most favorable sites for PV installations. In the future, the ability to utilize other sites, such as those that have partial shading, will become more important.

8.4.2 Improved Reliability/Lifetime

The poor lifetime of electrolytic capacitors used in the power processing equipment (MPPT and inverter) is one of the limiting reliability factors of PV systems. This is of particular concern for installations that employ per-module tracking (Fig. 8.5c), as these micro-inverters are typically attached to the individual modules, where they are exposed to the harsh outdoor environment (in particular solar heating) which can drastically reduce their lifetime. To maintain adequate reliability and lifetime, expensive enclosures rated for outdoor use must therefore be used for each converter. In contrast, the distributed MPPT enables the use of a central inverter stage which can be located in an easily accessible indoor environ-

ment. The distributed dc-dc converters that are integrated into the panel do not require large electrolytic capacitors, since they do not need to buffer the 120 Hz power ripple which an inverter must handle in single-phase applications.

Finally, the usable lifetime of a PV installation can be increased with the proposed system. Over time, the solar cell electrical characteristics change due to, among other things, degradation of encapsulation material from ultraviolet light [61]. It has been shown [69] that cells age at different rates, leading to an increased cell mismatch over the lifetime of the PV system. In a conventional PV installation the cell that degrades the fastest limits the total system output power, leading to a system rate of degradation that is faster than that of the average cell. With our proposed system the lifetime of the PV system can be drastically increased, since degradation of individual cells has less of an impact on overall power output.

8.5 Suitable Circuit Topologies

The architecture shown in Fig. 8.7 can be implemented with several different circuit topologies. Previous work at the panel-level has employed boost converters [70], non-inverting buck-boost converters [71] and multi-stage choppers [72]. Each topology offers some advantages, and some topologies are more suitable than others. Here we outline a few important criteria that help guide the decision regarding what topology to use.

We have identified the following characteristics as desirable for any power converter employed in the distributed MPPT architecture of Figure 8.7.

A first requirement for the converter topology is that it should be able to modulate the cell current between zero and a value sufficient for MPP operation. It is also desirable that the topology be well adapted for current-source loading (for string connection) and that complete dc bypass be achievable without requiring continuous modulation (e.g., for the case of a broken or fully-shaded cell).

A second requirement relates to filtering. Solar cells mounted in an array typically exhibit capacitance to ground (e.g., owing to PV cell structure and mounting) [73]. It is therefore desirable to configure the converter to suppress common-mode switching currents to ground. Topologies with output inductors are useful in this regard, as the interconnect inductance can help accomplish this. Moreover, coupling between the top and bottom inductances of each converter in Fig. 8.7 could be used to further suppress common-mode currents through parasitic capacitances.

A third requirement is that the topology selected should be suitable for maintaining acceptable stresses on the low-voltage switches across the whole required operating range. In topologies imposing higher switch voltage stress, "stacking" two switches (e.g., with a "cascode" switch connection [74]) can be used to double the achievable blocking voltage. Low voltage stress (and subsequent rating) of the device is paramount to achieve efficient operation and small size and cost.

A fourth requirement is the ability to implement the converter design with few semiconductor devices, and few passive components, if possible. In a discrete implementation the number of components directly affect cost and size. In a fully integrated converter it is not the number of semiconductor devices, but the die area that must be kept low, so topologies that can achieve the same functionality while using less die-area (e.g. implemented with fewer low-voltage CMOS switching transistors) are preferred.

Shown in Figure 8.8 are schematic drawings of the converter topologies considered in this analysis: buck, boost, non-inverting buck-boost, zeta, and SEPIC. All converters are shown with power MOSFETs as their switching devices, since synchronous operation is desirable for high efficiency at the relatively low voltages involved.

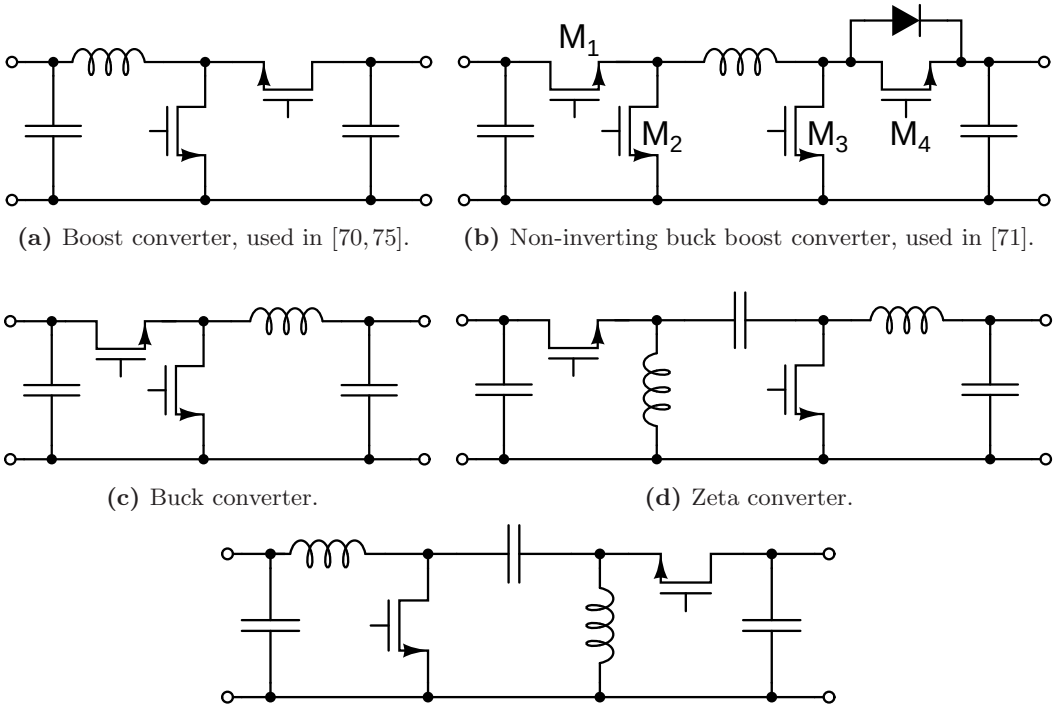
8.5.1 Boost Converter

The boost converter, shown in Figure 8.8a, has the advantage that it can provide an increase in output voltage, thus enabling a system with high output voltage with only a few converters. However, this comes at a big disadvantage: the boost converter can only step down current, so a poorly performing sub-module (low output current) can again bring down the entire string. As pointed out in [70], if one sub-module is sufficiently shaded such that its output current is lower than the string current, it may be necessary to completely bypass the weak sub-module when using boost converters.

As an example, consider a system with a nominal sub-module V_{mpp} of 12 V, and an I_{mpp} of 5 A. If one wishes to achieve a nominal voltage step-up of 2, the nominal duty cycle is 0.5. The nominal output (string) current is then 2.5 A. If any of the sub-modules experience shading (or other, static mismatch) that causes their I_{mpp} to drop below 2.5 A (consistent with slightly less than 50% shading, as shown by our experimental measurements of Section 8.10), the output current of the entire string must be reduced. If one attempts to reduce the nominal output current further (by increasing the nominal D) to mitigate this effect, one ends up running at a very large conversion ratio. The correspondingly large output voltage of the converter requires high-voltage power transistors with attendant parasitic losses and low achievable switching frequency. For this reason, existing boost-converter based module-level MPPTs in the literature [70, 75] have so far been limited by poor efficiency and large size.

8.5.2 Non-inverting Buck-boost Converter

The non-inverting buck-boost (also known as a cascaded buck-boost), shown in Figure 8.8b has been proposed as a suitable power converter topology for per-module distributed MPPT [71], thanks to its ability to provide both an increase and decrease in voltage and current. The diode of Figure 8.8b is added to provide a means for passive bypassing of the converter



(a) Boost converter, used in [70, 75].

(b) Non-inverting buck boost converter, used in [71].

(c) Buck converter.

(d) Zeta converter.

(e) SEPIC converter.

Figure 8.8: *Schematic drawings of possible distributed MPPT power converter topologies.*

in the case of failure.

The non-inverting buck-boost converter, with voltage conversion ratio given by:

$$\frac{V_{out}}{V_{in}} = \frac{D}{1-D}, \tag{8.1}$$

does not suffer from the same limits as the boost converter, as it can handle under-performing sub-modules without limiting the current in the entire string. As described in [71], in this application the non-inverting buck-boost converter is typically operated in only buck-mode (M_3 permanently off, M_4 permanently on, and M_1 and M_2 switching as a buck converter), in boost-mode (M_2 permanently off, M_1 permanently on, and M_3 and M_4 switching as a boost converter), or in bypass-mode (M_2 and M_3 permanently off, M_1 and M_4 permanently on). The non-inverting buck-boost converter thus offers a high degree of flexibility in terms of the achievable output voltages, but comes with significant down-sides, as discussed below.

One of the main disadvantages of the non-inverting buck-boost converter is the achievable efficiency. Since all transistors must be sized for switching operation (buck or boost type), their on-state resistance must be carefully balanced with their parasitic capacitance. The designer can thus not choose devices with very low on-state resistance (and correspondingly high capacitance), as switching losses will bring down the overall conversion efficiency when operating in buck or boost mode. At all times, however, there is at least one device (two in bypass mode) that is always on, contributing to the overall conduction loss. Thus, the overall efficiency of this topology is limited (as reported in [76], with a bypass efficiency of 98% and a buck and boost operating efficiency of approximately 95%[2].).

The other disadvantage of the non-inverting buck-boost converter is that it requires twice as many power devices as the buck and the boost topologies, which will contribute to an overall higher cost. With recent initiatives [77] seeking to reduce installed cost of solar

[2]The efficiency results given in [71] do not include gate drive and control losses. This impact (around 1% reduction in conversion efficiency) has been accounted for here based on information presented in [76].

PV to less than $1/Watt, it is critically important to keep the cost of the power electronic components to a minimum.

8.5.3 Buck Converter

The buck converter topology (shown in Figure 8.8c) is a well-known converter that provides the benefit of low component count and simple control. By stepping up the input current (and stepping down the input voltage), the buck converter topology can ensure that even the weakest sub-modules can contribute whatever power they produce to the string without negatively affecting the performance of the other converters in the same string. Using this topology, all distributed MPPTs will produce the same output current, but the weaker sub-modules will have a lower output voltage.

A distributed MPPT architecture employing buck converters can thus not provide an increase in voltage, but will typically produce an output voltage that is slightly lower than a PV panel without distributed MPPTs. The exception would be for severe shading situations, where the conventional panel would have one or more bypass diodes conducting (making those sub-panels contribute no power, and add no voltage to the string voltage). In this scenario, the distributed MPPTs would ensure that even the weak sub-modules produce some power (and voltage), meaning that the overall string voltage would be higher than for a conventional panel.

Another benefit of the buck converter topology is that the highest voltage observed is the open circuit voltage (V_{oc}) of the sub-module. The converter can thus employ switches that are rated for this relatively low voltage, enabling high frequency operation while maintaining high efficiency. Both the boost and non-inverting buck-boost require that the switches must be rated for the (higher) output voltage, and thus require slower, high voltage devices, which lead to increased converter size and cost.

8.5.4 Alternative Converter Topologies

More advanced power converter topologies (such as the Zeta and SEPIC converters, shown in Figures 8.8d and 8.8e) can be employed, and offer some advantages, such as a voltage conversion ratio of:

$$\frac{V_{out}}{V_{in}} = \frac{D}{1-D}, \tag{8.2}$$

which can provide an output voltage that is higher or lower than the input voltage. In many aspects, these two converters offer many of the same benefits as the non-inverting buck-boost converter, with a different trade-off. The Zeta and SEPIC converters require only two power switches (compared to four for the non-inverting buck-boost), but these switches see a higher voltage stress ($V_{in} + V_{out}$), necessitating the use of high-voltage transistor with attendant parasitic losses. Furthermore, both the Zeta and the SEPIC converter make use of two inductors (which may be coupled), which is undesirable, since they are often-time the limiting component in terms of size and possibility of integration in a power converter.

Another, more subtle disadvantage of both the Zeta and the SEPIC is their inability to perform a native dc-bypass of the converter. The buck, boost, and non-inverted buck-boost all employ only switches and inductors in the main current-path, and can thus be operated in bypass-mode by turning on certain switches, and incur only conduction loss. This could be done, for instance, for the strongest sub-module (or set of sub-modules) to reduce switching losses when the controller detects that bypass-mode produces an overall increase in output power. Since neither the Zeta nor the SEPIC has a dc-path between input and output (owing to the charge transfer capacitor), an additional high-side switch would have to be used if bypass-mode is desired.

For our purposes, the Zeta converter is a more attractive candidate than the SEPIC, owing to the use of an output inductor, which is beneficial for common-mode filtering of the switching currents, and the ability to employ interconnect inductance for this purpose (particularly at very high switching frequencies, where small inductors can be employed).

8.5.5 System-level Considerations

Because individual solar cells operate at very low voltage (typically < 0.7 V), one must stack a large number of cells in series in order to realize the high voltages desired for efficient interface to the grid and for buffering of energy. While the synchronous buck topology is extremely simple and very effective in this application, it does not contribute any voltage gain (which would reduce the number of "controlled cells" or sub-modules that must be series connected).

In the case where the system-level implementation puts a premium on achieving high voltage gain with a small number of cells or sub-modules, the non-inverting buck-boost topology and the Zeta converter both offer voltage gain and the ability to handle under-performing sub-modules gracefully (which the boost converter cannot, owing to its inability to step-up the input current). If one strives to minimize semiconductor count in this scenario, the Zeta converter is a good choice, whereas the non-inverting boost has the advantage of using a single inductor, which may be important in some implementation. Both converter unfortunately will have difficulty achieving high efficiency at high switching frequency, owing to the switch sizing requirements previously discussed.

In our scenario, the buck converter is the most attractive choice, as it enables both high switching frequency (important for small size, low cost) and high efficiency. In most residential and utility-based installations there are a sufficient number of PV panels to provide for the inherent stacking of voltages without requiring the additional step-up from the power converter. By not tasking the power stage with providing additional voltage step-up, it can be optimized for size, cost, and efficiency. The synchronous buck converter then becomes a good choice, and as our experimental results indicate, offer size, cost, and efficiency that greatly surpass that of previous work which employed other topologies.

8.6 Discrete Hardware Implementation of Sub-Module Distributed MPPT

Chapter 7 of this thesis illustrated that low-voltage CMOS MPPT implementations with small size and high efficiency are indeed possible. While the input voltage in that case (0.8-1.3 V) was slightly higher than the working voltage of a single solar cell (0.5-0.6 V), many of the techniques presented in Chapter 7 are directly applicable to the problem of single-cell distributed MPPT. Even though power point tracking of individual cells in a PV panel has the potential to extract the most energy out of the system, the associated increase in cost and manufacturing complexity present significant challenges that remain unsolved to this date. It is clear, however, that the trend in solar PV is towards more and more localized control, and single-cell control represents the last step in that direction.

In order to explore different control strategies for the architectures of Fig. 8.7, a sub-module MPPT implementation with appropriate communication hardware has been implemented. A sub-module implementation shares many characteristics with a single-cell tracking system, so many of the techniques and control solutions apply equally well to both single-cell and sub-module tracking. In fact, the current and voltage relationships between the two approaches are similar, where the voltage in the sub-module case (for a 3-module 72-cell PV panel) is 24 times higher than the per-cell approach, and the current is the same in both approaches.

Figure 8.9 shows a schematic drawing of the sub-module MPPT implementation designed as part of this thesis. The system comprises a synchronous buck converter power stage controlled by a microcontroller to achieve local MPP operation. The microcontroller can sense voltage, and also employs lossless current sensing [78] for algorithms that also require current information. Each converter employs an isolated I2C communication interface, which enables bidirectional information transfer to a master node, which can be a dedicated microcontroller or a computer. Table 8.1 provides a listing of the components used in the

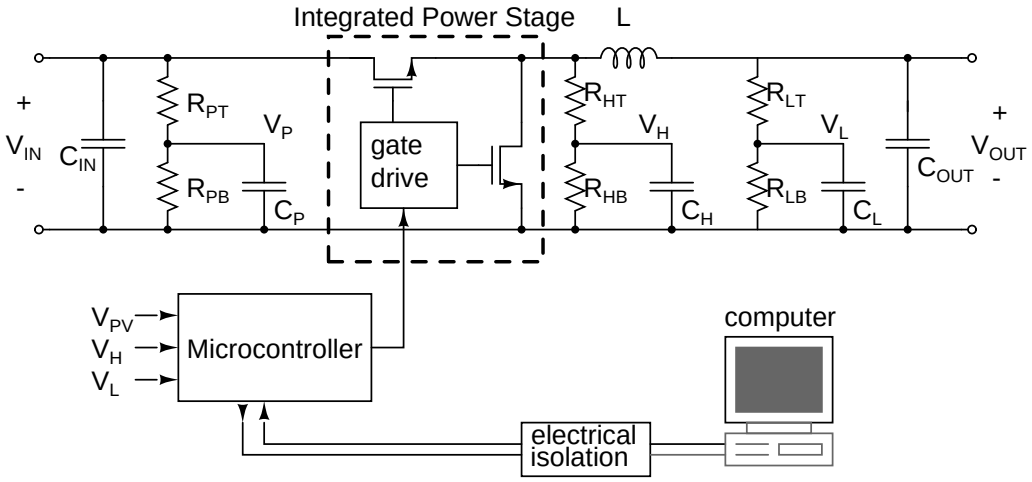


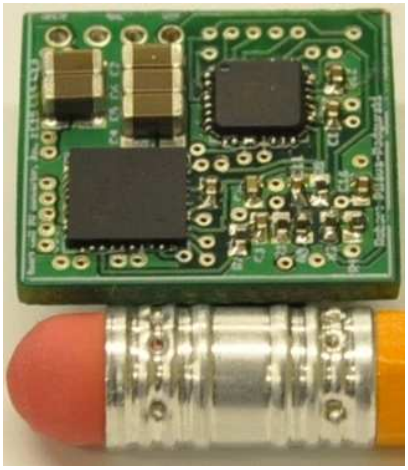
Figure 8.9: *Schematic drawing of the sub-module MPPT, developed using discrete components. Component values are provided in Table 8.1*

design. For a complete bill-of-material and cost analysis, please see Appendix H.

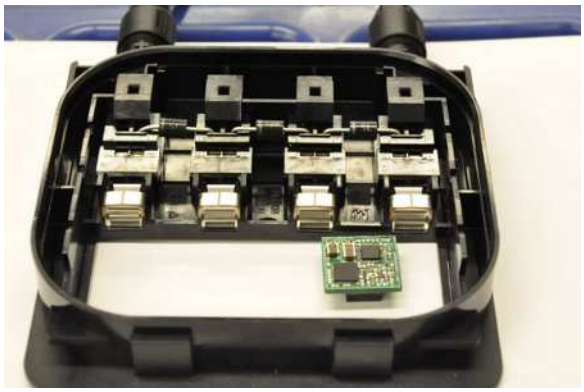
The focus of the design was to achieve a small enough converter footprint to fit into the junction box on the back of off-the-shelf PV panels. By utilizing the existing weather-resistant junction box as an enclosure, significant cost savings can be realized. The Integrated Power Stage is a combined gate-drive and power MOSFET chip (FDMF6704A), which also incorporates a 5 V linear regulator, enabling the converter to be completely powered from the sub-module. The FDMF6704A is intended for microprocessor VRM ap-

Table 8.1: *Component Listing*

Device	Model	Value	Manufacturer
Integrated Power Stage	FDMF6704A		Fairchild
L	SER1360-103KL	10 μH	Coilcraft
R_{HT}, R_{LT}, R_{PT}	0402	100kΩ	Panasonic
R_{HB}, R_{LB}, R_{PB}	0402	10kΩ	Panasonic
C_H, C_L, C_P	0402	1 μF	Murata
C_{IN}	1206, X5R, 25V	3 x 10 μF	Murata
C_{OUT}	1206, X5R, 25V	2 x 10 μF	Murata
Microcontroller	ATtiny861		Atmel



(a) Photograph of the sub-module MPPT converter with pencil shown for scale. The power inductor is on the bottom side of the PCB.



(b) Photograph showing discrete implementation of the power converter together with a solar panel junction box.

Figure 8.10: *Photographs of sub-module MPPT hardware.*

plications, and thus is optimized for a large conversion ratio (typically 12 V to 1-2 V). In our application, however, we would prefer the power switches to be optimized for a small conversion ratio, since that will give high efficiency when there is no shading (i.e. the buck converter operates near a duty cycle of 1). Nevertheless, for this experimental prototype, the significantly smaller footprint and performance of the FDMF6704A made it compare favorably to solutions which required separate gate driver chips, power transistors, and linear regulators, making it our preferred choice.

Shown in Fig. 8.10a is a photograph of the complete converter prototype, together with a pencil for scale. The PCB is a regular, low-cost 2-layer FR4 board available as a standard order from many commercial vendors. Shown in Fig. 8.10b is one of the MPPTs placed in a typical solar panel junction box. In a full installation, three converters are attached, one in parallell to each bypass diode. It is evident that the converter fits in the junction box, with plenty of space to spare, and room for connectors and sufficient air-flow.

Shown in Fig. 8.11 is an annotated photograph of the experimental test board with four converters and associated isolated I2C communication interfaces. In a typical PV panel

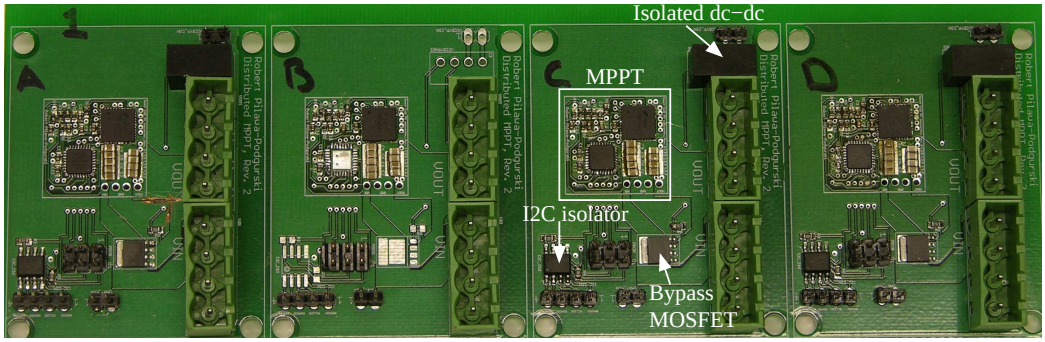


Figure 8.11: *Annotated photograph of the experimental prototype converters with isolated communication. The bypass MOSFET (powered by the isolated dc-dc converter) enables the MPPT to be bypassed entirely (for evaluation purposes).*

with three bypass diodes, only three of the converters would be used. In our prototype, each converter can be individually addressed and controlled, and information regarding operating voltage and current can be transferred to a master node for detailed analysis of tracking performance. It should be noted that each converter can operate as a stand-alone unit, without any I2C interface or computer interaction. The communication interface was implemented to enable extensive diagnostic and data collection, which helps in the development and evaluation of local and global tracking algorithms. Detailed PCB layout images and bill-of-materials for the MPPTs and test setup boards are provided in Appendix H.

The communication between the computer and the distributed MPPT was implemented using the Aardvark I2C Host Adapter from Total Phase. The host adapter provides bidirectional translation of the commands from the USB port to the I2C bus. Custom control software was written in Python to communicate with each MPPT, execute the tracking algorithms, and store data with information about operating voltage, current, and duty cycle of each converter for tracking analysis. Example Python scripts are provided in Appendix J.

The test setup of Fig. 8.11 also provides for complete bypassing of any MPPT through a low on-state resistance MOSFET. The MOSFET can be turned on by an isolated 5V-5V dc-dc converter that is powered directly through the USB port, as seen in Fig. 8.12. A 3.3 V general purpose port on the I2C host adapter is used in conjunction with a P-channel

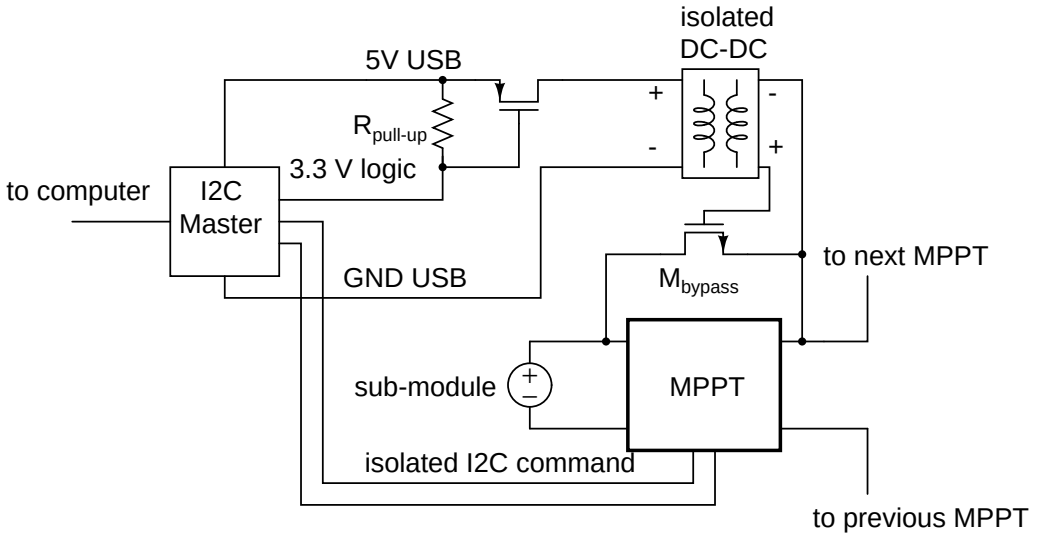


Figure 8.12: *Schematic drawing of the MPPT bypass circuit which is powered through the USB port, and controlled by a general purpose pin on the I2C host adapter.*

MOSFET to provide power to the dc-dc converter. Powering the bypass switch directly from the USB port of the control computer minimizes the required hardware and cabling. In addition, the ability to control the bypass switch directly through the I2C host adapter ensures that all bypass-commands can be synchronized and issued from within the main Python program, thus providing very accurate timing of the data capture.

8.7 Control Implementation

One challenge that must be addressed is the control of the distributed MPPT system to achieve the desired maximum power extraction. A centralized inverter system typically implements MPPT control at the "string level". Here, we can implement it in a distributed fashion at the cell or sub-module level, providing a variety of system-level control opportunities. From a control perspective, the sub-module and cell-level distributed MPPT architectures are very similar, since neither employs bypass diodes (the sub-module has a bypass diode across the entire sub-module, but there are no bypass diodes within a single sub-

mdoule itself). There are thus no local maximas caused by conducting bypass diodes, and the sub-module control case is essentially the same as the cell-level case, but with a higher operating voltage, owing to the stacking of the cells within the sub-module. In this work, we will describe a sub-module control algorithm and implementation, but the approach can be implemented at the cell-level as well, with suitable scaling of voltages.

In order to extract maximum energy from a PV installation with sub-module power tracking, each MPPT must continuously operate its sub-module at the correct current and voltage, while also allowing all other MPPTs do the same for their individual sub-modules. We must thus design a control algorithm that ensures that each sub-module operates at its *local* MPP, while also ensuring that the overall system operates at the *global* MPP (i.e., the overall string voltage and current are such that all sub-modules are operating at their respective MPPs).

8.7.1 Local MPPT algorithm

Since the outputs of the individual power trackers are connected in series (as seen in Figure 8.7a), all of them share the same output current (I_{string}). If the number of series-connected converters is large (which is typically the case in a system installation, where a large output voltage is desired), the string current (from the perspective of a single MPPT) can be considered constant. With a constant output current, each converter can then maximize its own output power by maximizing its output voltage. It thus follows that a local MPPT algorithm can be implemented by driving the local output voltage to its maximum value. As will be shown in the experimental section, this control algorithm works well with as few as three converters in series. Shown in Figure 8.13 is a flow chart diagram of the local MPPT algorithm.

In order to quickly locate the approximate location of the MPP, the converter starts by performing a coarse sweep of its duty cycle, and measuring the corresponding values of output voltage. The duty cycle corresponding to the maximum voltage observed is recorded,

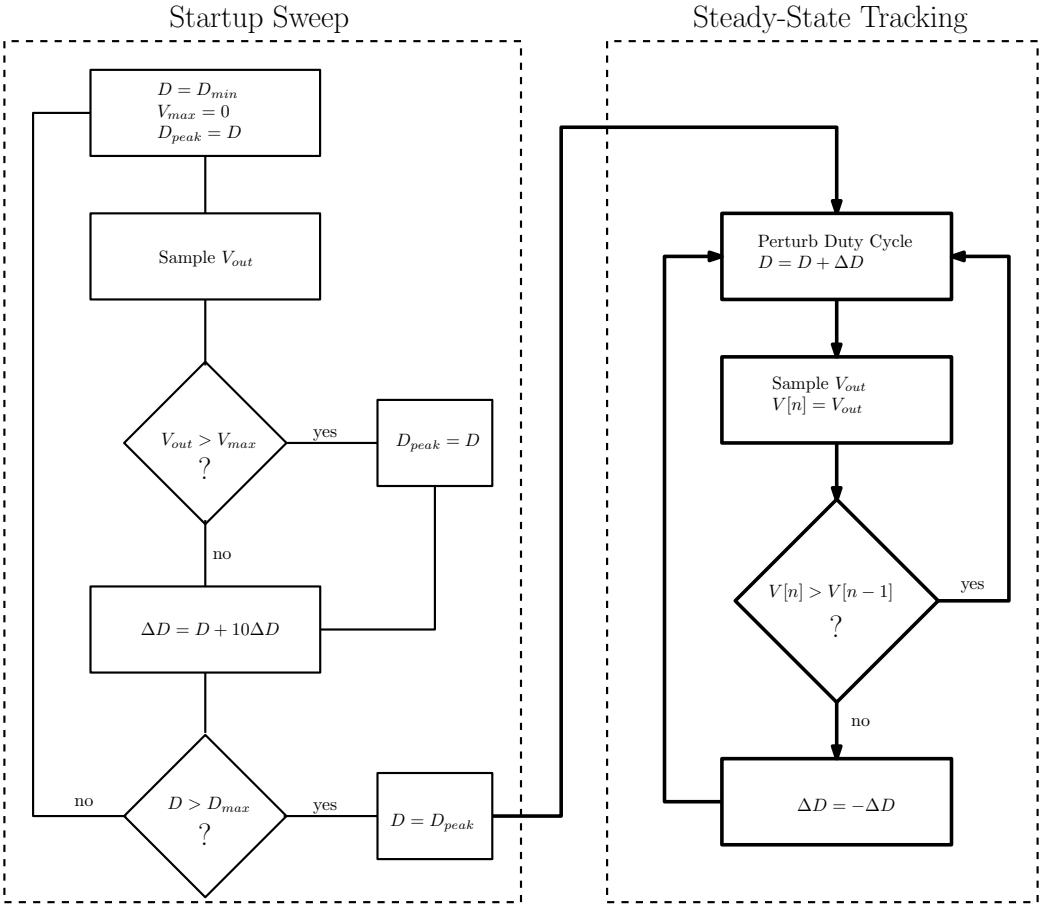


Figure 8.13: *Flow chart diagram illustrating the local MPPT algorithm. The approximate MPP is first found via a coarse startup script, followed by a perturb and observe algrithm that strives to maximize converter output voltage.*

and at the end of the startup sweep the converter is set to operate at this duty cycle. At this point, the steady-state tracking algorithm begins, which uses a perturb and observe algorithm which aims to maximize the converter output voltage by making small changes (ΔD) to the duty cycle (D). In this manner, the sub-module MPPT will continuously track the MPP, and oscillate around it to within the finite precision of its voltage sensing and duty cycle control. Table 8.3 provides information about our sensing and PWM resolution and step-size in the experimental prototype of this work.

8.7.2 Global MPPT algorithm

By adjusting the duty ratio, the local MPPT can autonomously achieve MPP operation so long as the cell or sub-module current at its MPP is equal to or less than that of the string[3]. Thus, to achieve overall MPP operation, each cell/sub-module controller adjusts its duty ratio for MPP operation (e.g.,in a "fast" loop) based on the string current, while the system level controller (typically implemented by the grid-interface inverter) adjusts the string current (in a "slow" loop) such that there is just sufficient string current available for the cell/sub-module with the highest MPP current. In this manner, the control problem can be separated into a local MPPT control for each cell, along with a single global loop that only requires limited information.

1-bit Feedback Global Algorithm

One method to ensure that the overall system is operating at the global MPP is to signal to the global ("slow") loop controller when one of the local MPPTs operate at its maximum permitted duty cycle. At this point, the system loop controller may not decrease the current (I_{string}) any further, as the strongest MPPT would then not be operating at its MPP. This 1-bit feedback signal can be implemented either using a very simple single-interconnect or

[3]This constraint is due to the chosen power converter topology (buck converter), where the power stage can only increase the output current.

zero-interconnect communications link, or by encoding the information to communicate it directly via the series string interconnect. (We note that such methods are well known in other types of distributed power conversion systems [79–81] and can be implemented without significant expense in this application.) One disadvantage of this method is that it would require the global controller (the inverter in a typical installation) to implement this functionality, such that separate dedicated hardware and firmware is required at the inverter level.

Communication-less Global Algorithm

It is also conceivable that with the appropriate cell or module-level control, global maximum power point operation can be ensured without *any* communication between individual converters, or between converters and the string-level inverter. All PV inverters used with conventional solar panels today already implement a maximum power point tracking functionality. It would be highly desirable to leverage this existing infrastructure to achieve both global and local optimization with existing inverter hardware.

If the global MPPT controller draws too little current, the strongest MPP will operate at its maximum duty cycle, and its sub-module will deliver I_{string}, which will be less than its I_{mpp}. Since this sub-module is no longer operating at its individual MPP, the overall output power of the string will decrease. When the global controller detects this decrease in power, it will act to reverse this change, thus increasing the string current. The global MPPT algorithm itself can thus ensure that the string current is not operating at a current that is lower than the highest I_{mpp} of the sub-modules.

The buck-topology can theoretically produce any output current that is higher than its input current (although there are certainly practical limits such as device parasitics, duty cycle resolution, and loss mechanisms that limit the maximum output current). In a real converter, the conduction losses in the MOSFETs, inductor, and wiring will increase as the output current is increased, leading to lower conversion efficiency at very high currents. A

lower conversion efficiency in the sub-module MPPTs will lead to lower string power, which can be detected by the global MPPTs algorithm if the output current is increased too much. It should be noted that this effect (decrease in output power by reduced conversion efficiency) is much less pronounced than the relatively sharp drop-off in power observed in a regular PV panel when it operates away from the MPP. The distributed MPPT thus have the effect of significantly "flattening" the power versus voltage (or current) characteristics of the system. The advantage of this is that the central inverter can operate at many different voltage and current levels (while drawing near maximum power from the system). There is, however, a risk that the central inverter may not be able to detect the small changes in power associated with the change in sub-module MPPT efficiency, and may continuously wander across a wide current and voltage range as it searches for the global MPPT.

In the experimental measurements of Section 8.10, we will see the results of a control mechanism that makes use of the 1-bit feedback global MPPT algorithm. The flattening effect of the distributed MPPTs will also be observed, and we can quantify the resolution required to implement the communication-less global MPPT algorithm in practice.

8.8 Power Stage Characterization

In order for the distributed MPPT architecture of Fig. 8.7 to be effective, it is important that the additional power captured by more localized control is not wasted by low conversion efficiency of the power electronics. Much care was thus taken in this work to achieve high efficiency operation, both through the choice of topology and passive components, as well as the implementation of sensing and control. In this section we characterize the discrete sub-module MPPT in terms of efficiency and output power.

Figure 8.14 is a schematic drawing showing the efficiency measurement setup. Four digital multimeters (HP34401A) were used to sense input and output voltage and current, while an electronic load (HP6060B) was used to vary the load characteristics. The input voltage

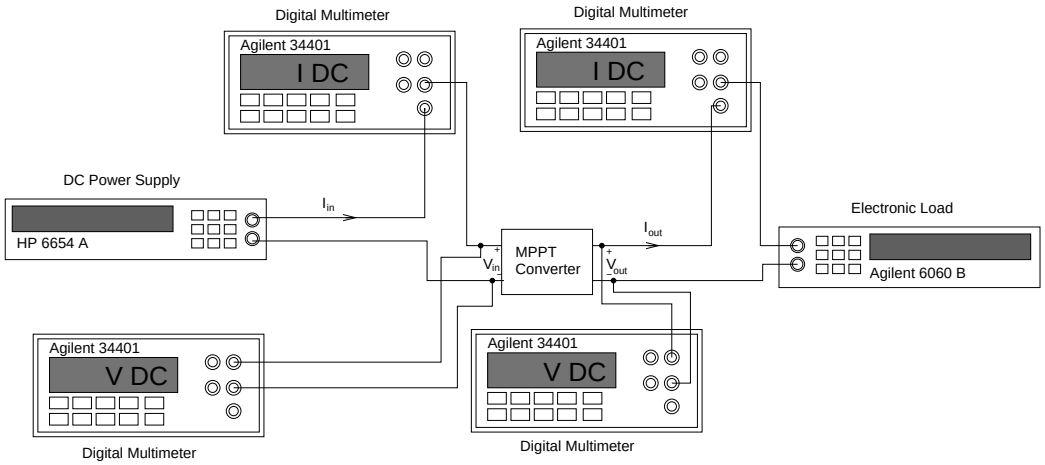


Figure 8.14: *Schematic drawing of efficiency measurement setup. All meters are triggered at the same time over GPIB, and their values read from a computer.*

Table 8.2: *Converter Specifications*

Input Voltage Range	5-27 V
Output Voltage Range	0.8-20
Max Output Power	80 W
Switching Frequency	250 kHz
Converter Peak Efficiency	98.2%

was provided by a a 60V, 9A power supply (HP6054A), with remote sensing of the input voltage at the input terminals of the converter under test, to account for any voltage drops due to wiring. The electronic load and the four multimeters were connected together over a common GPIB bus, and were controlled through a Python script on the lab bench computer. For efficiency measurements, all four meters were triggered at the same time over the GPIB bus, and data was read out sequentially and stored on the computer. In order to handle the relatively large output current rating of our converter (up to 8 A), the two current-sensing multimeters (fused at 3 A) employed a high precision shunt resistor (HP34330A, rated at 15 A continuous). Shown in Table 8.2 is an overview of the specifications of the converter, along with a performance summary.

A detailed efficiency and power characterization of the MPPT converter has been carried

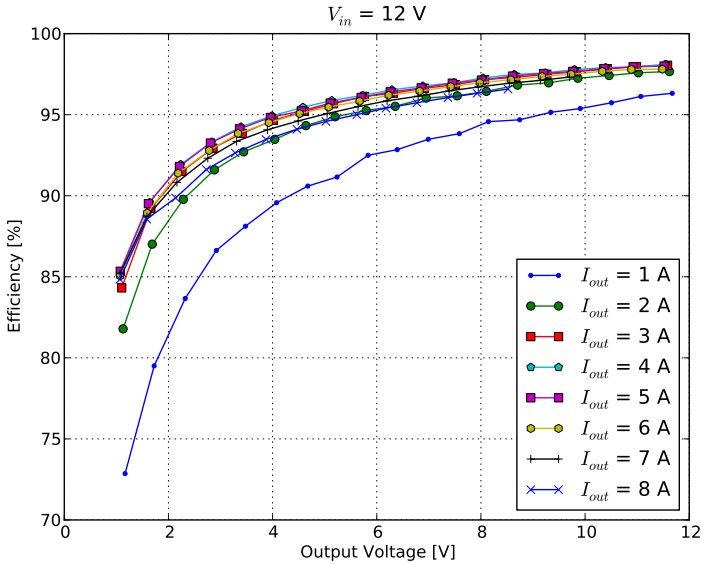


Figure 8.15: *Measured efficiency versus output voltage, parameterized by output current. A lower output voltage corresponds to a shaded sub-module, while a lower output current signifies a string with less insolation.*

out to measure performance across a wide load and output voltage range. Figure 8.15 shows a plot of efficiency versus output voltage, parameterized by output current, for a fixed input voltage of 12 V (with a current limit of 5 A, which is a typical maximum current for a sub-module). The converter would operate at lower output voltages if it suffers from more shading relative to the other converters in the string. A low output current would signify that the insolation of the entire string of MPPTs is relatively low. Given these characteristics of the system, it is important to achieve high efficiency at high power levels (for maximum total energy capture), as well as at operating points where the converter is expected to spend significant time in real-world scenarios. In Figure 8.15, this would correspond to high output voltage (no or little shading) and high current (>5 A, corresponding to high insolation). We see from the plot that we achieve an efficiency above 97% under these conditions. It should be noted that all efficiency measurements include all sensing, gate drive and control losses, as the converter itself is powered from the input voltage.

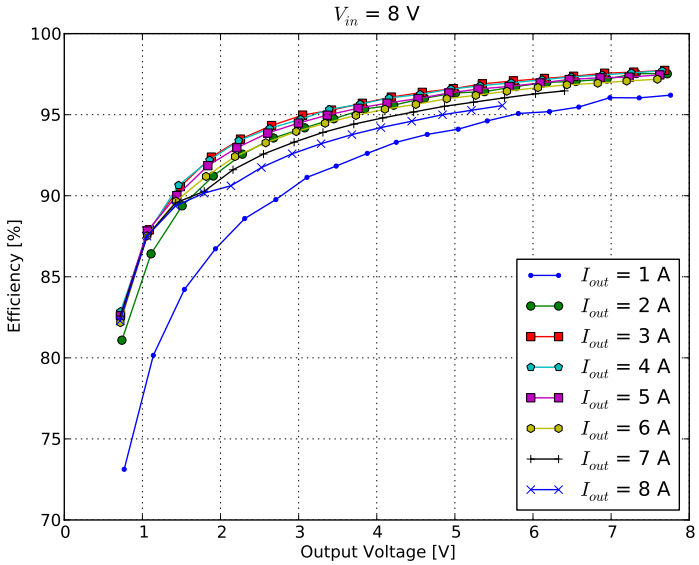


Figure 8.16: *Measured efficiency versus output voltage, parameterized by output current for an input voltage of 8 V.*

Figures 8.16 and 8.17 show the corresponding efficiency versus output voltage for input voltages of 8 V and 16 V, respectively. The input voltage is primarily a function of the number of cells in a given panel, and the voltage range we investigate here (8-16 V) is selected to match that of sub-modules for commercially available solar PV panels.

8.9 Experimental Laboratory Results

In order to properly test the distributed MPPT architecture, the laboratory test setup of Figure 8.18 was constructed. It comprises 18 halogen work lights suspended over the solar panel, with a total electric power output rating of 10 kW. Combined, the lights were able to produce enough irradiation in the wavelengths of interests to approximate the effect of a full sun. For our testing purposes, we need continuous irradiation for long periods of time to evaluate the MPPT algorithms and the power architecture. Conventional test platforms used to evaluate PV panels are typically of the flash type, which only provide a brief light

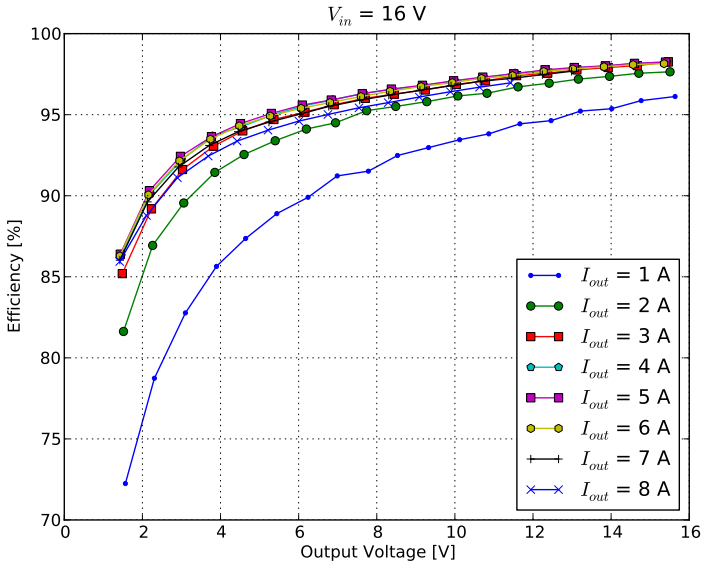


Figure 8.17: *Measured efficiency versus output voltage, parameterized by output current for an input voltage of 16 V.*

of high intensity (and of a spectrum carefully designed to match that of the sun) to evaluate the instantaneous efficiency and power output of the panel. The test platform of Figure 8.18 is designed only to provide enough total irradiation to test the power electronics, without trying to mimic the spectrum of the sun. In fact, our test setup emits significantly more irradiation in the infra-red regime than the sun, requiring additional fan cooling to keep the PV panel temperature down. In addition, since the test setup of Figure 8.18 was connected to 3-phase AC power with individual lamps allocated to a particular single phase, a 120 Hz AC power ripple could be observed when performing high accuracy measurements of the MPPT output power, so this system is not suitable for measuring very high MPPT tracking efficiency.

Shown in Figure 8.19 is a schematic drawing of a PV solar panel, which indicates the locations of each of the three sub-modules. Also indicated is the shading method used to simulate a cell that performs worse than the others due to issues such as: aging, soiling, shading, manufacturing defect, or external damage.

Figure 8.18: *Photograph of the bench setup for testing of shading effects on solar panel output power. The setup enables repeatable adjustment of light intensity and shading pattern, as well as easy access to measurement instruments.*

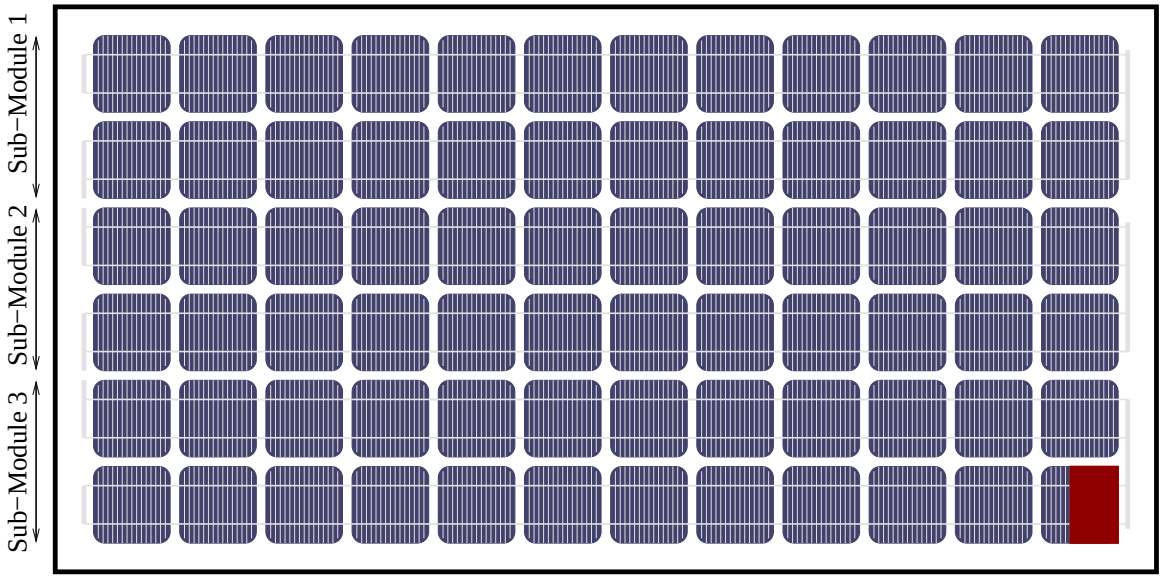


Figure 8.19: *Drawing of the solar panel illustrating the physical location of the three sections that are accessible through the junction box (corresponding to the electrical wiring schematic shown in Fig. 8.7a). The bottom right cell in Sub-Module 3 is partially shaded in this experiment. The solar panel used in this experiment was the STP175S-24/Ab01 72-cell monocrystalline Si panel from Suntech*

Shown in Figure 8.20 is a power versus string current plot for an experiment when one cell in sub-module 3 (as shown in Figure 8.19) is shaded. Table 8.3 provides the MPPT tracking parameters used for this and all subsequent MPPT tests. The minimum achievable duty cycle step-size with the hardware we implemented was 0.1%, but the 0.6% step-size provided a good trade-off between conversion speed and steady-state accuracy. The effect of the partial shading of a cell in sub-module 3 of the panel can be clearly seen in the reduced output power of sub-module 3. Note also that due to the non-uniform irradiation of our test setup, sub-module 1 produces less power than sub-module 2, even though both of them remain un-shaded in this experiment. The solid turquoise line shows the resulting output power when all three sub-modules are connected in series, as would be done in a conventional solar panel. The effect of the bypass diodes conducting can be clearly seen by the three local maxima in the P-I plot. The maximum output power of the panel configured conventionally is approximately 70 W as seen in the plot.

Table 8.3: *MPPT Tracking Parameters*

MPPT Duty Cycle Step-Size	0.6%
MPPT Startup Sweep Step-Size	5%
Minimum Duty Cycle	10%
Maximum Duty Cycle	99 %
ADC Resolution	10 bit
ADC Samples Per Measurement (Overampling)	100

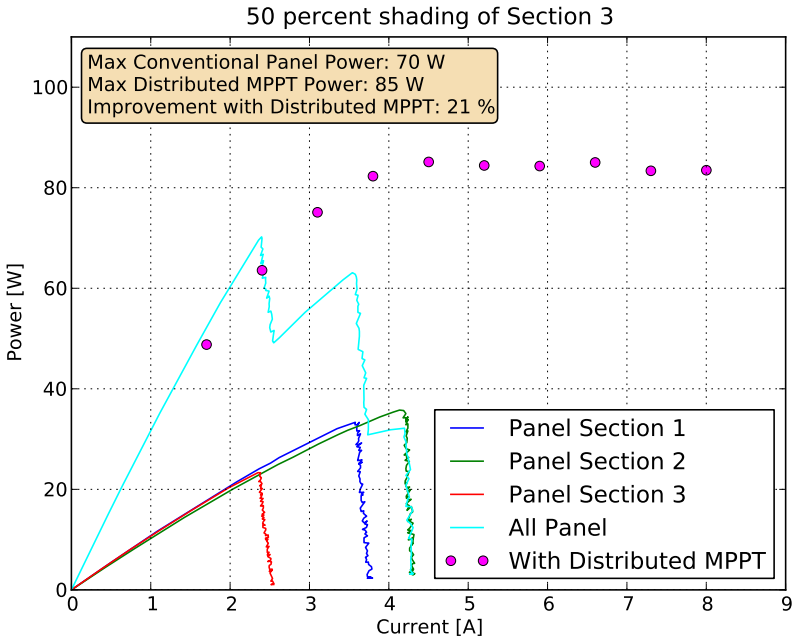


Figure 8.20: *Plot showing power versus current characteristics of the different panel sections (as shown in Fig. 8.19) under partial shading conditions. In this case, a single cell of sub-module 3 was shaded by 50%. The solid turqoise line shows the maximum output power of a conventional panel, where the effects can be clearly seen in the multiple local maxima (caused by conducting bypass diodes). Also shown is the experimentally-measured output power when the distributed MPPTs of Fig. 8.11 are used, which show increase in output power of approximately 15 watts (21%) for this particular shading scenario.*

Also shown in Figure 8.20 is the experimentally measured output power when the converters of Fig. 8.11 are connected to the panel in a distributed MPPT architecture (as previously described in Figure 8.10b). In this case, the distributed MPPTs allow each of the sub-modules to operate at their individual maximum power points, irrespective of the operating currents of the other sub-modules of the panel. This enables a substantial increase in output power, as seen in the plot. We also note that near maximum power can be extracted across a wide range of string currents.

Shown in Figure 8.21 is data from each MPPT during the experiment that generated Figure 8.20. The top plot shows how the load current is stepped in time, and correspond to the discrete current measurements of Figure 8.20. The middle plot shows the corresponding change in duty ratio, as the converters begin with startup sweeps, and adjust their operation each time the load current changes. In this implementation, each MPPT performs two startup sweeps, since the operating points of each MPPT is affected by the other converters. By running two staggered startup sweeps each converter finds the approximate MPP before the perturb and observe algorithm is started. It can be seen in the middle plot that MPPT 2 reaches its maximum duty cycle (1000) first, at a load current of slightly below 4 A. This is consistent with the I-V sweeps of Figure 8.20, which indicates that I_{MPP} of sub-module 2 (the strongest sub-module) is slightly above 4 A. When the string current is below this value, the converter will hit its maximum duty cycle, and no longer operates at the MPP.

The bottom plot of Figure 8.21 shows the output power of each MPPT over time. At each time when the load current changes, the MPPTs adjust their duty cycles to find the new MPP, as can be seen from the increasing ramp waveforms after each current step change. Note also that as the string current is reduced below 4 A, the maximum output powers of MPPT 1 and 2 are reduced, as they cannot operate at their MPP past this point, since their I_{MPP} is higher than 4 A.

A listing of the microcontroller code used for this (and all following) experiments is provided in Appendix I. The Python code used for performing the MPPT algorithm and

data-logging for the experiment described in this section can be found in Appendix J (`mppt_switching_1bit_feedback.py`).

8.10 Field Measurement Experimental Results

In order to fully evaluate the distributed MPPT system in a real setting, we chose to perform outdoor field experiments. Figure 8.22 shows an annotated photograph of the field setup. A south-facing PV panel (the STP175S-24/Ab01 72-cell monocrystalline Si panel from Suntech) was mounted on the roof of building 26 at the campus of the Massachusetts Institute of Technology, together with test equipment as shown. The camera was used to produce time-lapse photos of the shading pattern of the panel. The photos were synchronized with the output power measurement, which provides a visual check to discern shading patterns related to panel I-V characteristics. The distributed MPPTs were connected across each sub-module (in parallel with the existing junction diodes, as shown in Figure 8.7a), and their output connected to the electronic load (HP6060B). The electronic load was controlled through the GPIB interface by a small netbook computer that recorded all data.

8.10.1 Static Performance Evaluation

Shown in Figure 8.23 is a plot of measured panel output power versus load current for a completely un-shaded panel. The solid blue line represents the measurement when the panel was connected directly to the electronic load, without distributed MPPTs (the converters were bypassed). The green circles represent discrete data-points collected with the distributed MPPT enabled. After the electronic load has stepped its regulating current, enough time is allowed to pass (a few seconds) to ensure that the distributed MPPTs have reached their steady-state points after their start-up sweep. As can be seen from Figure 8.23, when there is no shading of the panel, our distributed MPPT architecture introduces a 2% power loss (consistent with our measured converter efficiency of 98%). For a perfectly matched panel

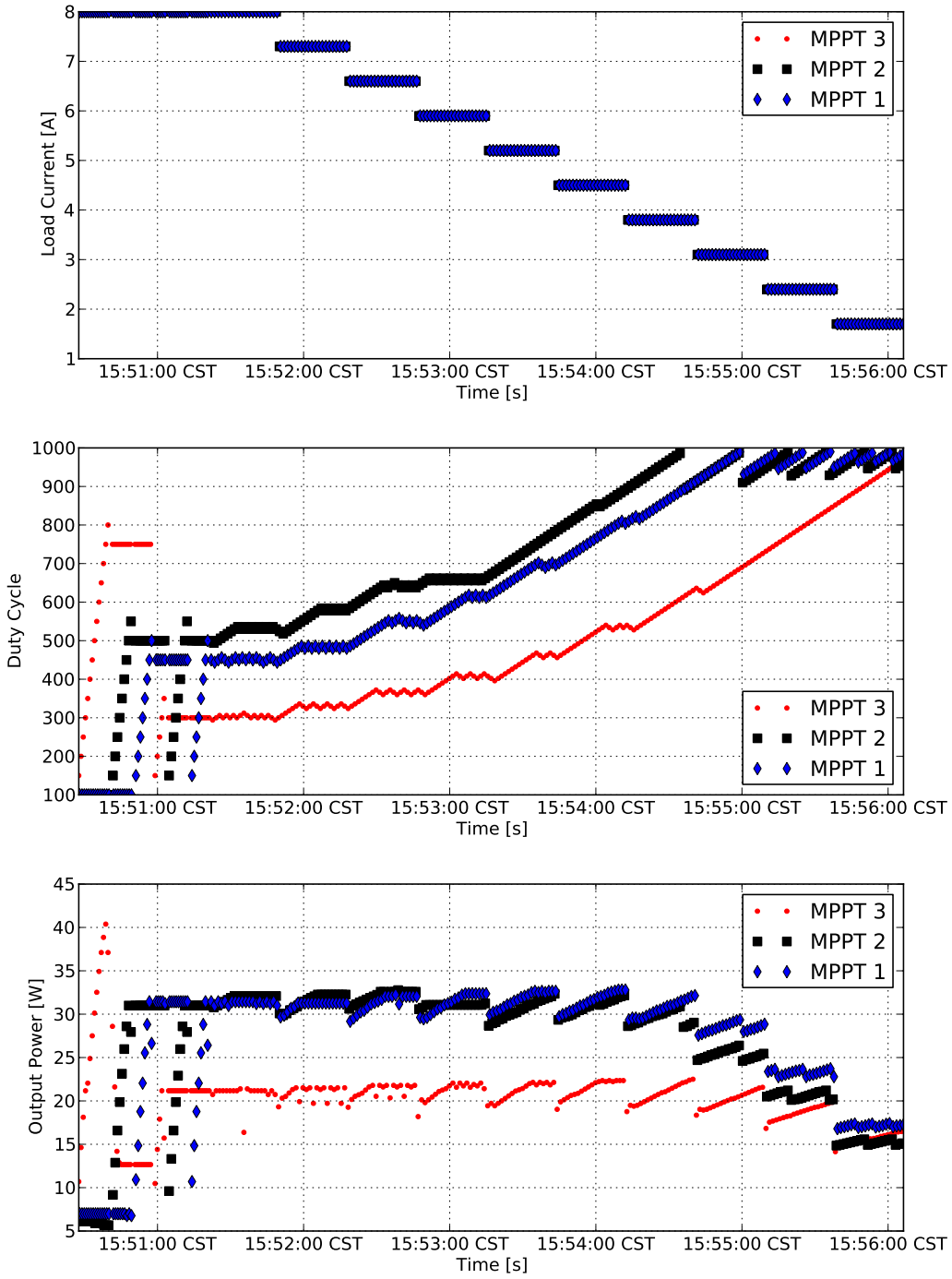


Figure 8.21: *Data from individual MPPTs for the experiment shown in Figure 8.20. (Note that a single cell of sub-module 3 was shaded by 50% for this experiment.) The dual startup sweeps can be clearly observed, as well as the individual MPPT controllers finding the MPP after each time the string current is stepped.*

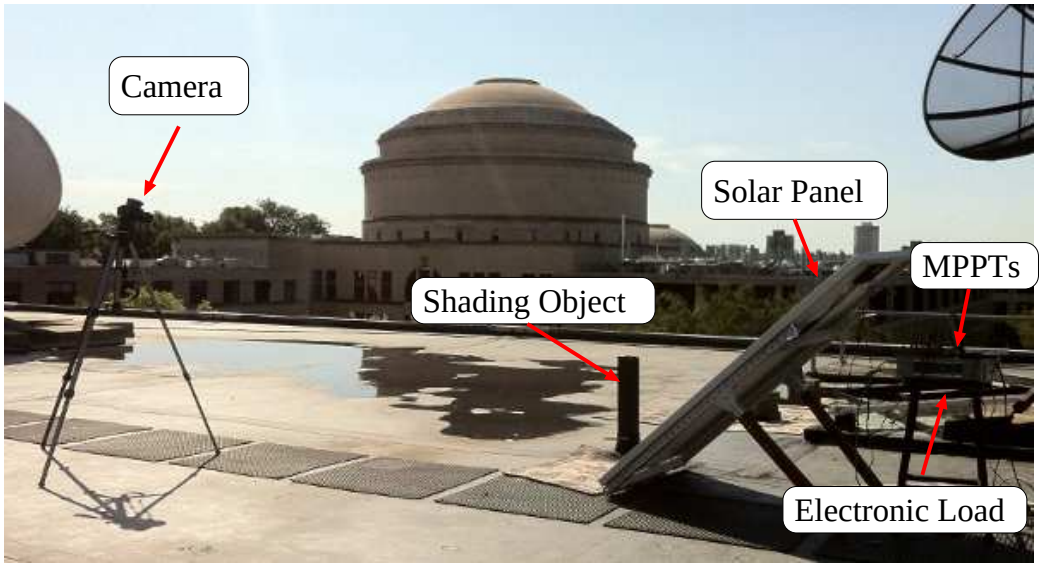


Figure 8.22: *Annotated field experiment setup on the roof of Building 26 of the Massachusetts Institue of Technology.*

with no shading throughout the day, our proposed system would thus not be beneficial, which comes as no surprise. It should be pointed out, however, that it is fairly trivial to implement a bypass-mode in the MPPTs themselves, such that during times of no shading the MPPTs are bypassed altogether, and thus not contributing any loss. This bypass-mode can be implemented in firmware only (turning the top MOSFET on permanently, with some additional conduction loss in the switch and inductor), or with one additional MOSFET with low on-state resistance (this approach will give the lowest loss in no-shading situations). The Python code used to perform these measurements (and to collect the data) is provided in Appendix J (`mppt_automatic_shading_patterns.py`).

Shown in Figure 8.24 is a plot of output power versus current when a single cell is shaded by 25%. In this case, the electronic load was first connected to each individual sub-module, to generate a plot of power versus output current. It can be clearly seen that sub-module 3 has a lower maximum output current (and hence power) due to the single shaded cell. Furthermore, from the plot showing the full panel power, two maximum power points can be seen. This is due to the bypass diode connected to sub-module 3 conducting when

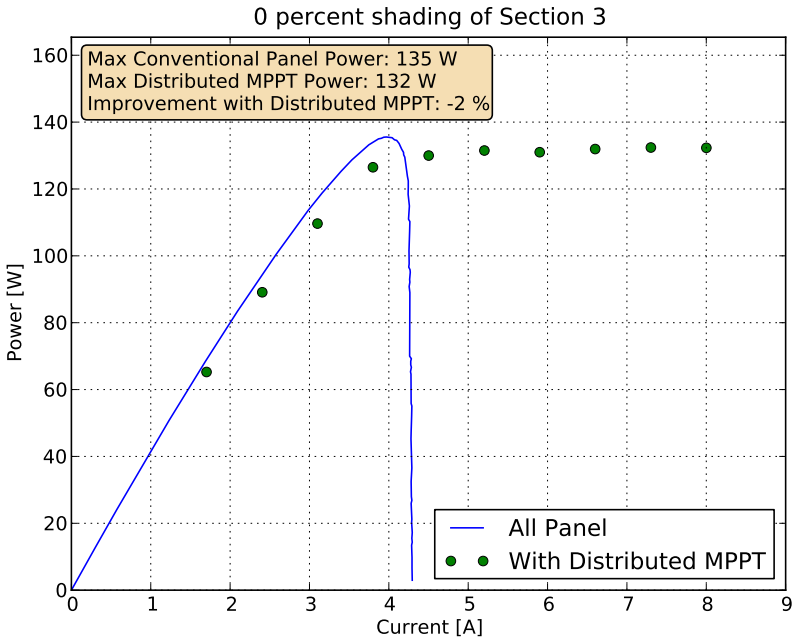


Figure 8.23: *Plot of power versus current with and without distributed MPPT, when there is no shading. A decrease in power of 2% is observed with distributed MPPTs, consistent with the 98% efficiency of the sub-module MPPT converters. When there is no shading, the MPPTs should be bypassed, which would eliminate this 2% loss. This data was taken on a October 6th, 2011, a very sunny day.*

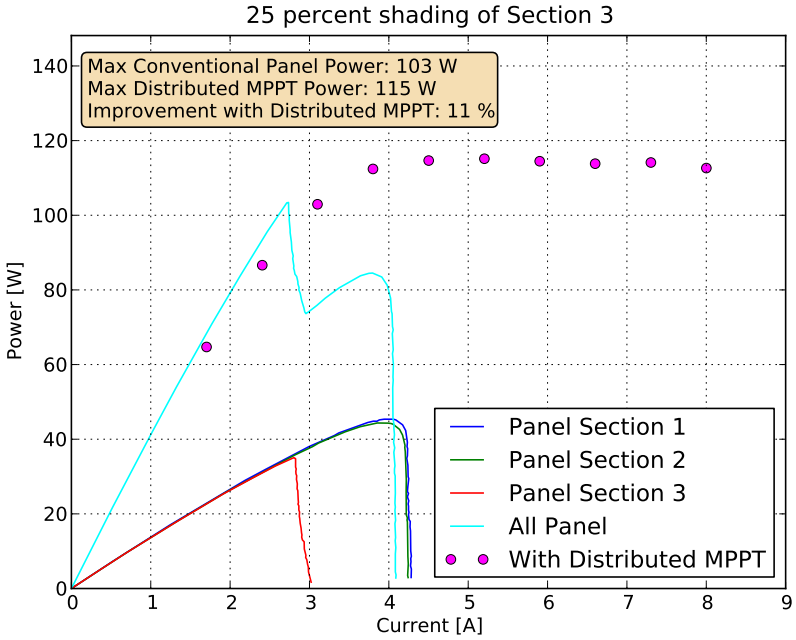


Figure 8.24: *Plot of power versus current with and without distributed MPPT, for 25% shading of one cell in sub-module 3. A power increase of 11% is observed by the use of the sub-module MPPTs. This data was taken on a October 6th, 2011, a very sunny day.*

the electronic load is drawing more current than the maximum current available from sub-module 3. In this case, it can be seen that the global maximum power point is the case where the bypass diode is not conducting, whereas the other point is a local maximum power point. Situations like this present problems for the MPPT algorithms in central and micro-inverters, as they can easily get stuck on the local maximum power point. The discrete data point collected with the distributed MPPT enabled in this scenario illustrates the benefit of our approach. In this case, an 11% increase in power output can be observed. Furthermore, there is only a single maximum power point, and the panel produces close to its maximum power across a broad range of output current, enabling the system's central inverter to operate across a wide voltage and current range.

Figures 8.25 and 8.26 show the measured data when the single cell is shaded by 50 and 75%, respectively. In both these cases, the global maximum power point of the regular panel

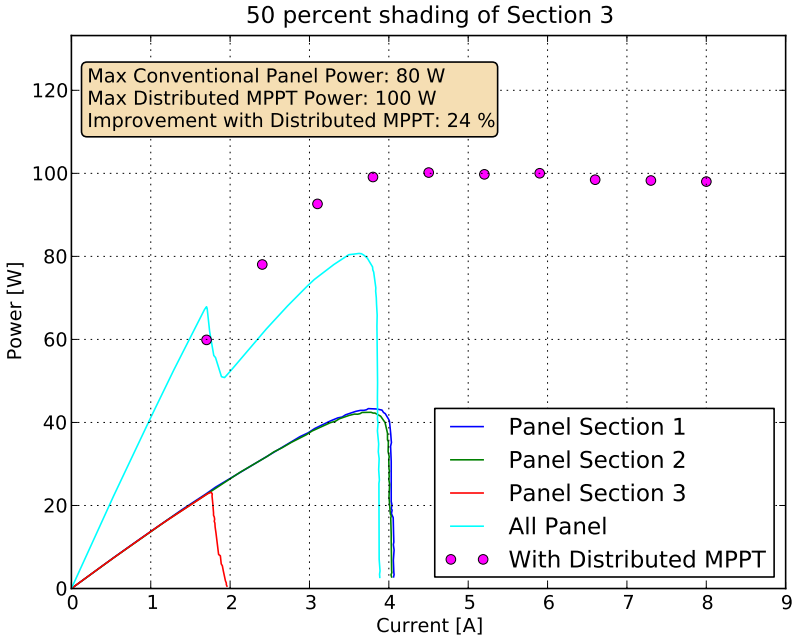


Figure 8.25: *Plot of power versus current with and without distributed MPPT, for 50% shading of one cell in sub-module 3. A power increase of 24% is observed by the use of the sub-module MPPTs. This data was taken on a October 6th, 2011, a very sunny day.*

is with the bypass diode of sub-module 3 conducting. It can be seen that the distributed MPPT system yields a power improvement of 24% when the cell is 50% shaded, and 11% increase in power when the cell is 75% shaded.

8.10.2 Dynamic Performance Evaluation

To evaluate the performance of the sub-panel distributed MPPT architecture under dynamic partial shading conditions, we performed the following experiment:

The panel was placed near a small metal chimney, so that only a small number of cells were shaded, as illustrated in Figure 8.27. As the sun moves throughout the day, the location of the shadow on the panel will move as well, and cover different sections of the panel, and to varying degrees. This situation is very similar to what would happen in

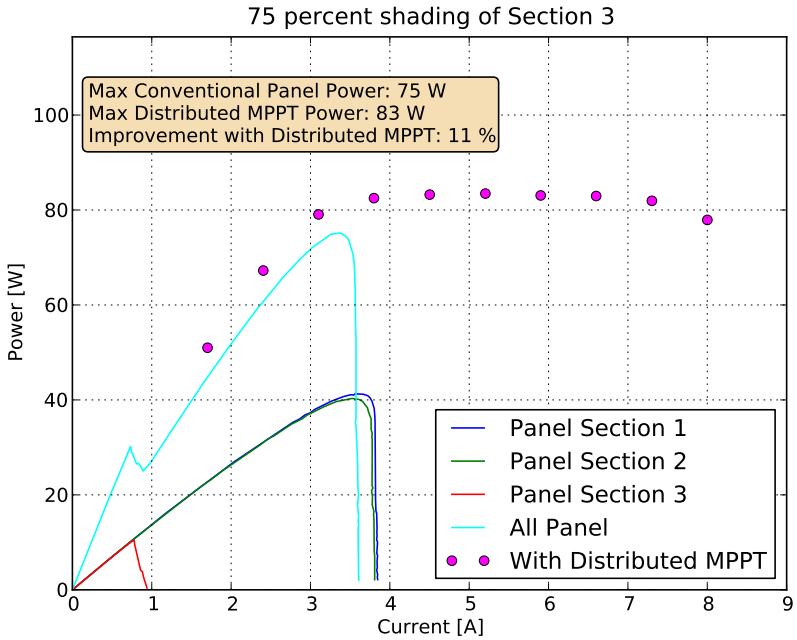


Figure 8.26: *Plot of power versus current with and without distributed MPPT, for 75% shading of one cell in sub-module 3. A power increase of 11% is observed by the use of the sub-module MPPTs. This data was taken on a October 6th, 2011, a very sunny day.*

Figure 8.27: *Photograph illustrating the shading (owing to a protruding pipe) that moves across the panel for the dynamic performance experiment.*

residential installations, where chimneys, power lines, trees, antennas, and other structures block parts of the panel throughout the day.

The system was set up such that approximately every minute it would switch between bypassing the distributed MPPTs, and connecting them to the panel. When the MPPTs are bypassed (i.e. the panel is configured just like a conventional panel) the electronic load performs a full I-V sweep of the panel, and the highest power is recorded. When the MPPTs are connected, the electronic load starts at a current (6 A) that is higher than the panel short-circuit current (5.2 A), and waits for the MPPT outputs to reach steady-state (a few seconds). It then decreases the current, at each time waiting for the MPPTs to settle again. It continues to decrease the current until one of the MPPTs (the one connected to the strongest sub-module) reaches its maximum allowed duty cycle (0.99). At this time any

further decreases in panel output current will mean that at least one of the MPPTs is not operating at the sub-module MPP, so the sweep is stopped, and the highest output power recorded. The electronic load thus performs a global MPPT tracking algorithm with 1-bit of feedback from the sub-module MPPT. In the implementation performed here, the computer receives the 1-bit feedback signal over the I2C communication link, but this control method could be implemented in a variety of ways, as was discussed in Section 8.7.2.

Another option that could have been used to perform the dynamic evaluation would be to use two panel, one with distributed MPPT, and one without. The problem with this approach is that it is difficult to ensure that both panels are perfectly matched in terms of their nominal output, and that they have identical shading patterns. By toggling between the two system on a single panel, these sources of bias are removed. By taking a very large number of samples, errors introduced by rapidly changing insolation between measurements can be averaged out, since it is expected that such insolation changes are equally likely to occur between both types of measurements. The long-term average error should thus be zero.

Shown in Figure 8.28 is a plot of panel output power versus time, with and without the distributed MPPT electronics, as discussed above. These measurements were taken at MIT's campus on a very sunny day (Oct 6, 2011) at the times indicated in the plot. It can be seen that at all times during the measurement period, the distributed MPPT system generated more power from the panel than what a conventional panel would generate, thanks to the mitigation of sub-module current mismatch owing to partial shading.

Shown in Figure 8.29 is the accumulated energy extracted from the panel during the measurement time, and it shows that the distributed MPPT system collects more than 20% more energy throughout the course of this experiment.

The data plotted in Figure 8.30 show the instantaneous power measured for the system during a day with more cloud cover (October 3, 2011), as can be seen by the rapidly changing output power (both with and without MPPT) when clouds move in. At times

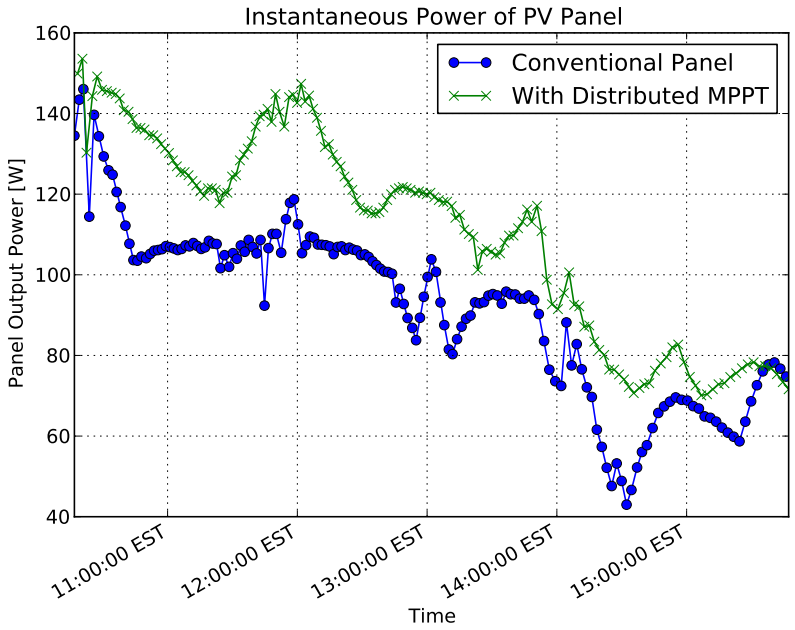


Figure 8.28: *Instantaneous measured power versus time for a sunny day (October 6, 2011) for a conventional panel, as well as with the distributed MPPT employed. Up to a 30% increase in captured power is observed.*

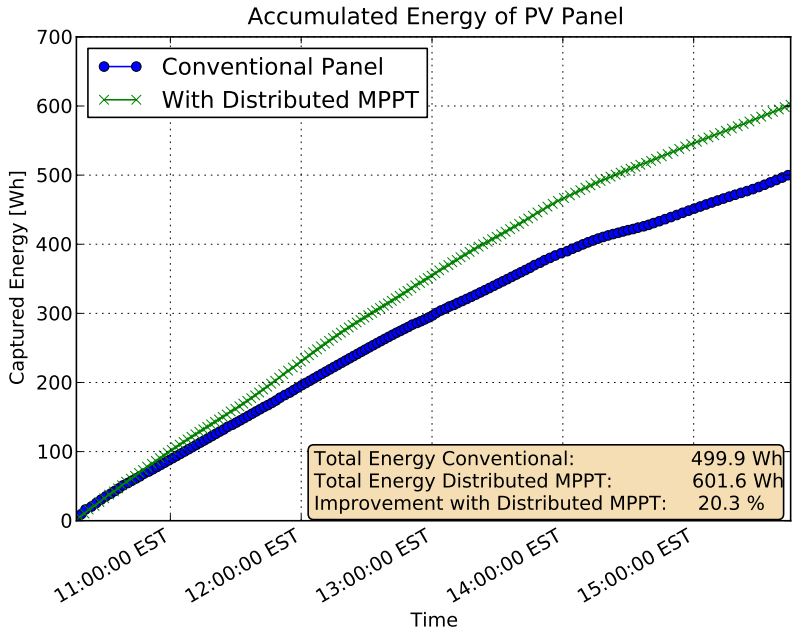


Figure 8.29: *Accumulated energy versus time for a sunny day (October 6, 2011) for a conventional panel, as well as with the distributed MPPT employed. The distributed MPPT system collects more than 20% additional energy over a conventional panel.*

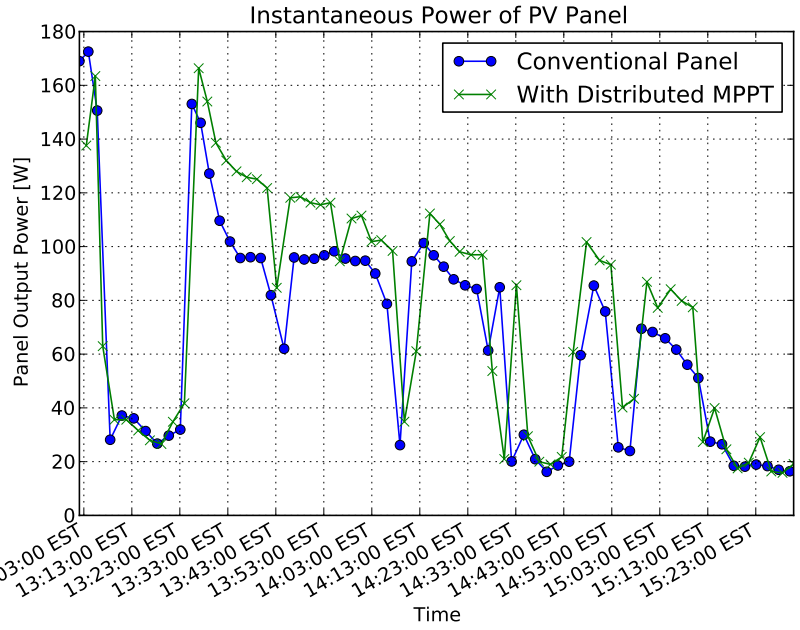


Figure 8.30: *Instantaneous measured power versus time for a day with moving cloud cover (October 3, 2011) for a conventional panel, as well as with the distributed MPPT employed.*

when the power output is changing rapidly between two measurement points, the relative performance differences between the two systems is more due to changing insolation than any change in shading pattern. The overall trend is nevertheless clear, with the distributed MPPT system generating more power for the majority of the time.

Figure 8.31 shows the accumulated energy for the system, where the distributed MPPTs collect more than 10% additional energy over a conventional panel. It is to be expected that the performance gains are smaller in this scenario than for a very sunny day, since the diffused light of cloudy days do not produce a pronounced shading pattern, and thus less sub-module mismatch.

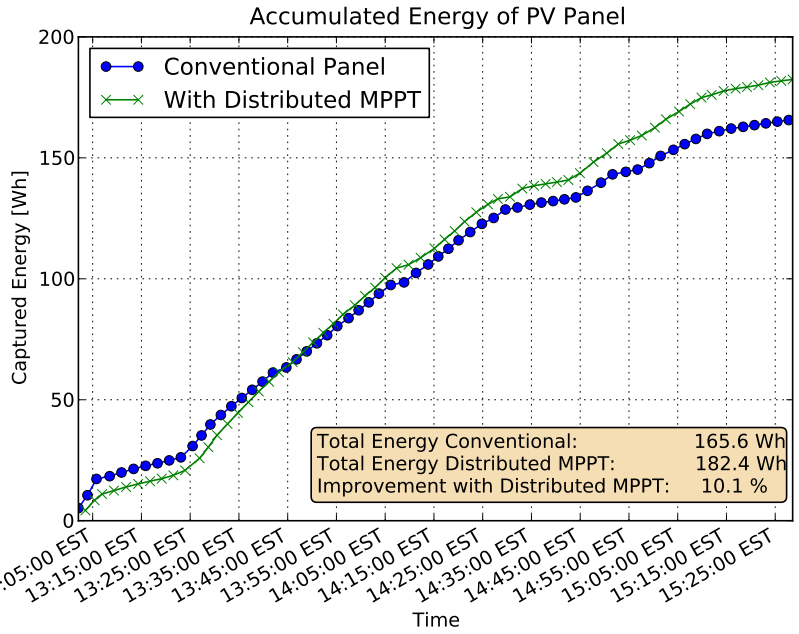


Figure 8.31: *Accumulated energy versus time for a day with moving cloud cover (October 3, 2011) for a conventional panel, as well as with the distributed MPPT employed. The distributed MPPT system collects more than 10% additional energy over a conventional panel.*

8.11 Performance Comparison

The previous section illustrated the improvements in overall energy capture that can be realized with the use of the distributed MPPT architecture, and the sub-module hardware implemented in this work. In solar PV applications, which are very cost sensitive, it is illustrative to perform a cost analysis, to quantify the added financial burden for this increase in power. A small increase in output power that comes at a large added system cost is clearly not worth it, and in this section we provide a quantitative analysis of this trade-off, based on the empirical data captured in our experiment.

Shown in Table 4.6 is a comparative chart of our work, previous academic work, as well as two selected commercial solutions. The topology, cost, power density, efficiency, and a figure of merit (discussed below) are listed. It should be noted that aside from the work presented here, none of the other solutions provide sub-module tracking, but only account for mismatch at the panel level. As was shown in the experimental section, sub-module mismatch can contribute to significant energy loss (up to 20%), which cannot be mitigated by the other solutions.

8.12 Figure of Merit

The merits of distributed MPPT in any solar PV system is entirely dependent of the particular installation. Some installations may benefit greatly from added power electronics, whereas others may see no improvement in overall energy capture (e.g., perfectly matched panels on a completely flat surface with no external objects that can cause shading). Due to the very site-specific circumstances, it is therefore difficult to quantify exactly how much a typical residential installation may benefit from our approach. It is, however, possible to quantify the relative merits of the power electronics itself, compared to other similar solutions. This is done in Table 8.4, where we have introduced a figure of merit that aims to capture some of the cost/benefit trade-off with this approach. It should be pointed out

that this figure of merit is a crude estimate of the relative performance between different solutions, and it should not be used as an absolute metric to judge whether distributed MPPT will pay off or not.

The figure of merit attempts to capture the incremental cost for the added average power to the PV system (given as $/Watt). It calculates the expected additional average power captured by the system (accounting for the electrical conversion losses of the MPPTs in each case), for a given nominal power increase factor (α). This increase factor represents the fractional increase in average output power that can be expected with the distributed MPPT system, and as such, is highly installation dependent. For our analysis, an α of 0.1 is chosen for per-panel MPPT, and 0.15 for sub-module MPPT (this is a modest 5% increase for sub-module MPPT compared to per-panel MPPT, keeping in mind that we experimentally measured between a 10% and 20% increase in captured energy for the sub-module case versus regular panel-based MPPT in our field experiments). The Figure of Merit is given by:

$$FOM = \frac{cost}{\langle P_{\text{added}} \rangle}, \tag{8.3}$$

where

$$\langle P_{\text{added}} \rangle = \eta_{MPPT} P_{\text{rated}} (1 + \alpha) - P_{\text{rated}}, \tag{8.4}$$

and η_{MPPT} is the electrical conversion efficiency of the MPPTs, and P_{rated} is the rated power of the MPPT. The FOM should be compared to the typical installed cost of solar PV systems, which was estimated to be around $6/W in 2010 [82]. In order for the distributed MPPT system to be cost effective, the FOM must be below the installed cost of the PV system, for a given installation. We see that for our assumptions of a 10% and 15% improvement in average power due to module and sub-module tracking, respectively, the cost benefit of many of the solutions of Table 8.4 are marginal. As the installed cost of solar PV continues to decrease, even further price pressure on the power electronics is expected. In light of this, our calculated FOM of 0.50 $/Watt makes our solution cost competitive

Table 8.4: *DC-DC Optimizer Performance Comparison*

Work	[71]	[70]	National	Azuray	This work
Type	Academic	Academic	Commercial	Commercial	Academic
Topology	Buck-Boost	Boost	Unknown	Unknown	Buck
Sub-Module Tracking	No	No	No	No	Yes
Volume [cm^3]	255 cm^3	unknown (big)	680 cm^3	740 cm^3	12 cm^3
Cost	\$20[4]	unknown (high)	\$150	\$90	\$12.80
Power [W]	85 W	60 W	230 W	300 W	200 W
Cost/Power [\$/W]	0.24 \$/W	high	0.65 \$/W	0.3 \$/W	0.064 \$/W
Efficiency [%]	95%	93%	98.5%	97.6%	98%
FOM [\$/(added W)]	5.22 \$/W		7.81\$/W	4.07 \$/W	0.50 \$/W

today, and for some time in the future.

It should be pointed out again that the FOM is highly dependent on the parameter α, which attempts to quantify the performance improvements offered by distributed MPPT. It is certainly possible to better quantify this improvement with a more detailed Figure of Merit that models the length of shading (in time), additional panels, and weather data. Our attempt here was merely to elucidate some of the trade-offs in terms of cost and performance, with rough estimates guided from our empirical data.

8.13 Conclusions

We have presented a distributed MPPT architecture for solar PV applications, which enables more energy to be extracted from the system. By employing low-voltage sub-module converters, a high frequency, very high efficiency power stage can be used, and miniaturized to the point where it fits into the existing junction box, thereby greatly reducing cost. We have provided a survey of potential power topologies, and have implemented a hardware prototype synchronous buck converter for use in sub-module tracking of a PV panel. Two different global control algorithms are proposed, and experimental measurement of static and dynamic shading patterns are investigated. We measure up to a 20% improvement in

overall energy capture compared to per-panel MPPT implementation, using field experiments with a partial shading obstacle. Finally, we compare our implementation to other, state-of-the-art commercial and academic solutions, and find that the proposed solution is by far the most cost-effective.

8.14 Future Work

While this work demonstrated one of the first cost-effective distributed MPPT solutions, there are a number of improvements that can be made to further reduce overall cost and improve performance. Here we list some areas that could benefit from further investigations

8.14.1 Cell-level Tracking

The control implementation and system architecture presented here are also suitable for cell-level tracking. While there are certainly additional challenges associated with a low-voltage cell-level converter (both from a performance and cost perspective), the methods outlined here provide a good starting point for work towards that goal. In addition, the flexible hardware implementation here can be used as a test platform for different kind of tracking algorithms, and to simulate cell-level converter operation.

8.14.2 Bypass-mode for Improved No-Shading Efficiency

One of the easiest improvements that can be made to our system is that of bypass-mode detection. We observed a 2% decrease in our overall power capture when there was no partial shading, due to the 98% efficiency of the power converters. It is relatively straightforward to implement a function in the MPPT code to detect when the power produced from bypassing is higher than that of regular operation. The MPPT can then be bypassed, either through a dedicated bypass-MOSFET, or through the top-level switch and inductor.

This would greatly enhance the performance during no-shading situations.

8.14.3 Verification of Communication-less Global MPPT Algorithm

One benefit of the distributed MPPT architecture is that the global controller (often-time implemented in the string inverter) can operate at a wide range of currents and voltages, while still extracting near maximum power from the system (as observed for example in Figure 8.20). As discussed in Section 8.7.2, it would be attractive to rely on the existing MPPT algorithm in the central inverter to find the global MPPT. As was observed in our experimental measurements, the distributed MPPT flattens the P-I and P-V curve, such that there is a broad region where near MPP operation is possible. One useful extension of this work would thus be to test the distributed MPPT architecture with a central MPPT converter that employs the communication-less global MPPT algorithm. It would be particularly illustrative to do this with a commercial PV inverter (or micro-inverter) and observe whether the inverter locates the MPPT, despite the relative flatness.

8.14.4 Reduction in Hardware Components/Reduction in Cost

In the current implementation, three identical MPPTs are employed, one for each bypass diode. There is thus a substantial replication of functionalities, in particular in terms of computation and sensing. Significant cost savings can be realized if, for example, a single microcontroller would control three power stages, and perform all sensing and control. The MPPT algorithm itself is not very computationally intensive, so it would be relatively simple for a single microcontroller (with a sufficient number of I/O pins) to control three power stages. One challenge that would need to be addressed is that of appropriate level-shifting of the signals, since each power stage is at a different voltage level. Another advantage of a single microcontroller is that each power stage can be easily phase-shifted relative to the others, and some current ripple cancellation could be achieved, enabling smaller inductors and/or fewer output capacitors. As with our previous demonstration of integrated

power point tracking control in Chapter 7, it could also be highly advantageous to develop dedicated integrated tracking control ICs, reducing both logic / control power, cost, and component count.

8.14.5 Other Applications

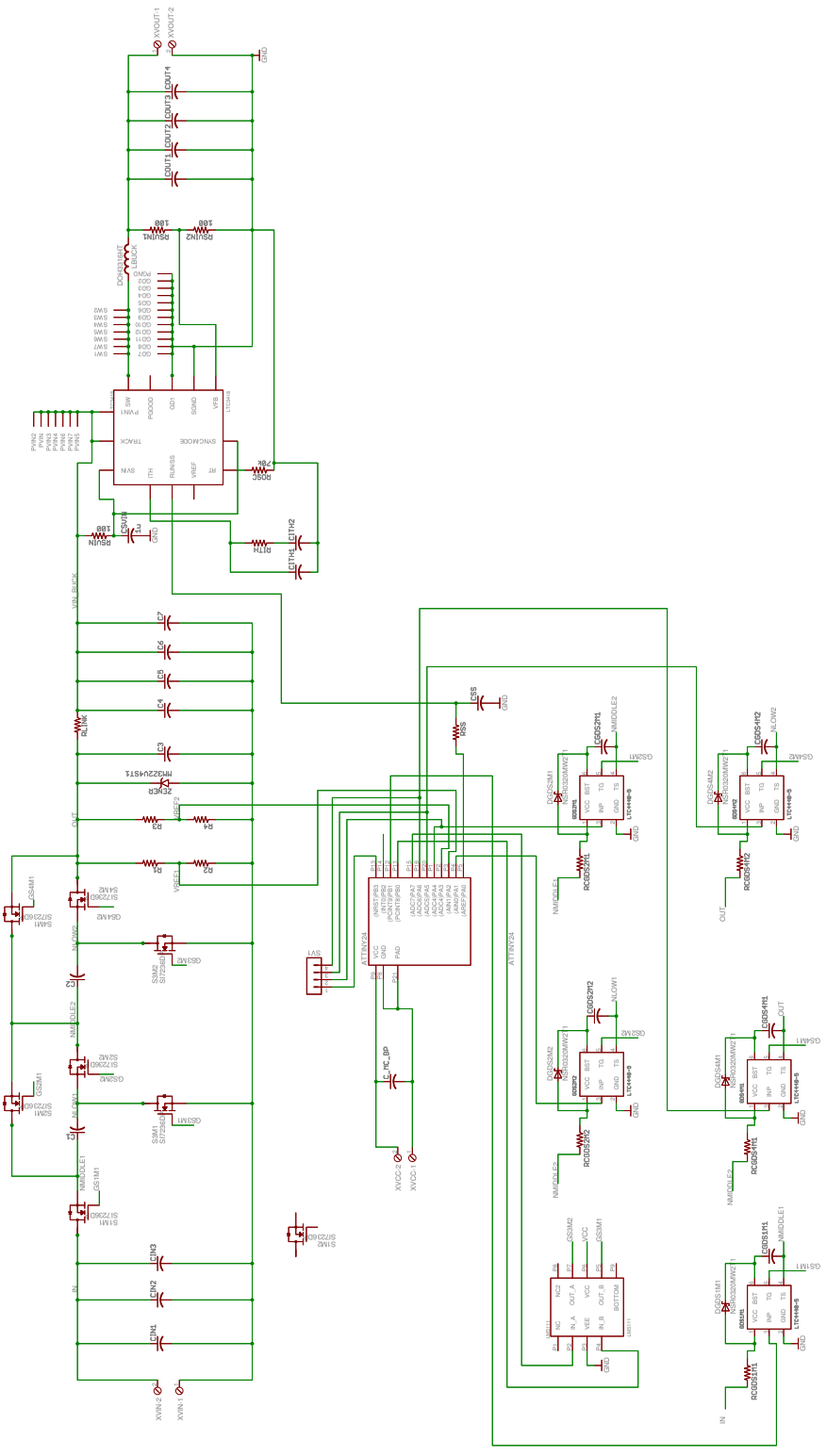
There are a number of other applications where a distributed power electronics architecture such as the one presented here would be beneficial. Thermoelectric power generation and fuel-cells are two examples that suffer from similar challenges in terms of current mismatch, and the need for voltage stacking to achieve a high voltage output from many low voltage sources. There is also an opportunity to employ this architecture in reverse, where a high voltage source (e.g. the power grid) needs to power many low voltage loads, such as in CPUs and LED lighting applications, to name just two examples.

Appendix A

PCB Layout, Detailed Schematic, and Bill of Materials for the Discrete Merged Two-Stage Converter

---

This appendix provides schematic and images of the PCB layout for the discrete merged two-stage converter prototype, as well as bill of materials. The PCB layout was made using EAGLE™ Layout Editor from Cadsoft Computer , Inc. Note that all PCB images here are scaled from their original size to provide better details.



12/7/11 9:56 PM f=0.52 /home/pilawa/work/eagle/SC/3to1_rev1/3to1SC/3to1SC_mirror_rev2.sch (Sheet: 1/1)

Figure A.1: *Converter schematic drawing. Tables A.1 and A.2 contains a components listing.*

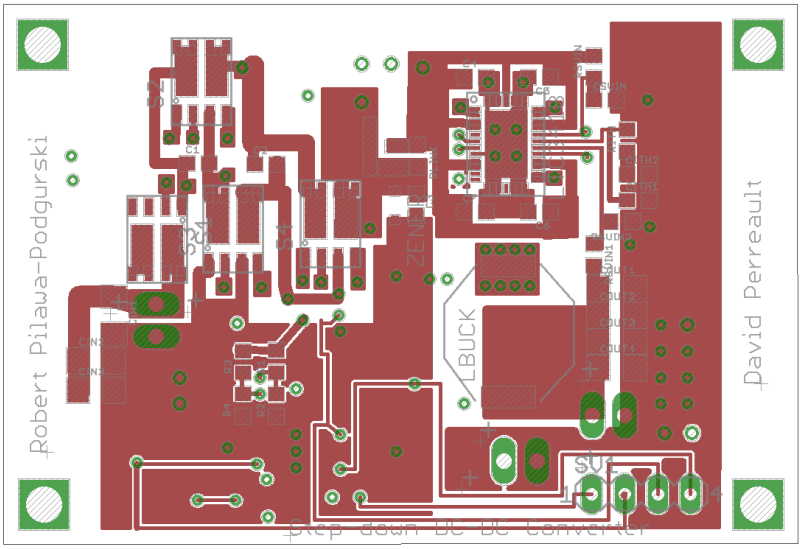


Figure A.2: *Converter PCB layout, top copper, silkscreen, and solder stop layers.*

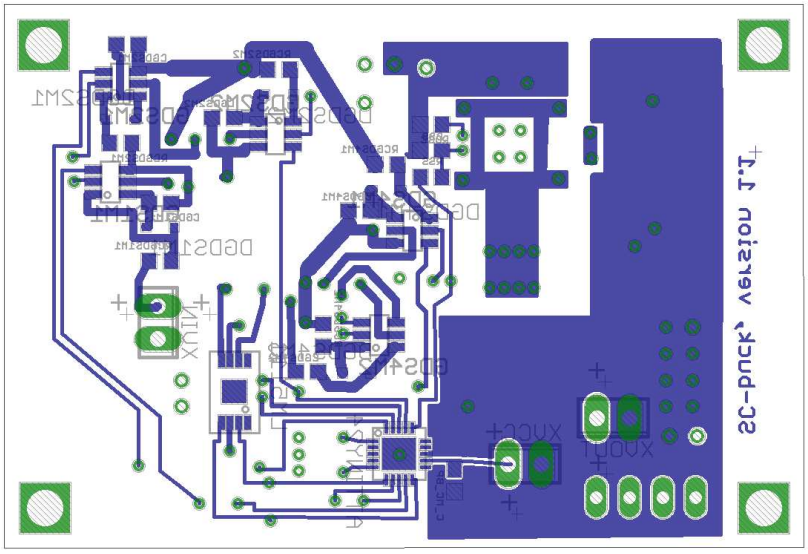


Figure A.3: *Converter PCB layout, bottom copper, silkscreen, and solder stop layers.*

Table A.1: *Bill of Materials for the discrete merged two-stage converter*

Part	Part Number	Package	Description
ATTINY24	ATTINY24	SOIC16	microcontroller,
C1		C0603	capacitor
C2		C0603	capacitor
C3		C0603	capacitor
C4		C0603	capacitor
C5		C0603	capacitor
C6		C0603	capacitor
C7		C0603	capacitor
CGDS1M1		C0603	capacitor
CGDS2M1		C0603	capacitor
CGDS2M2		C0603	capacitor
CGDS4M1		C0603	capacitor
CGDS4M2		C0603	capacitor
CIN1		C1206	capacitor
CIN2		C1206	capacitor
CIN3		C1206	capacitor
CITH1		C0603	capacitor
CITH2		C0603	capacitor
COUT1		C1206	capacitor
COUT2		C1206	capacitor
COUT3		C1206	capacitor
COUT4		C1206	capacitor
CSS		C0603	capacitor
CSVIN		C-USC0603	capacitor
C_MC_BP		C0603	capacitor
DGDS1M1	NSR0320MW2T1	SOD323	Schottky Diode
DGDS2M1	NSR0320MW2T1	SOD323	Schottky Diode
DGDS2M2	NSR0320MW2T1	SOD323	Schottky Diode
DGDS4M1	NSR0320MW2T1	SOD323	Schottky Diode
DGDS4M2	NSR0320MW2T1	SOD323	Schottky Diode
GDS1M1	LTC4440-5	SOT23-6	High side gate driver
GDS2M1	LTC4440-5	SOT23-6	High side gate driver
GDS2M2	LTC4440-5	SOT23-6	High side gate driver
GDS4M1	LTC4440-5	SOT23-6	High side gate driver
GDS4M2	LTC4440-5	SOT23-6	High side gate driver
LBUCK	DOH3316HT	DOH3316HT	Inductor
LM5111	LM5111	MSOP8	Low side gate driver
LTC3418	LTC3418		synchronous buck

Table A.2: *Bill of Materials for the discrete merged two-stage converter*

Part	Part Number	Package	Description
R1		R0603	resistor
R2		R0603	resistor
R3		R0603	resistor
R4		R0603	resistor
RCGDS1M1		R0603	resistor
RCGDS2M1		R0603	resistor
RCGDS2M2		R0603	resistor
RCGDS4M1		R0603	resistor
RCGDS4M2		R0603	resistor
RITH		R0603	resistor
RLINK		R0603	resistor
ROSC		R0603	resistor
RSS		R0603	resistor
RSVIN		R0603	resistor
RSVIN1		R0603	resistor
RSVIN2		R0603	resistor
S1	SI7236DP	POWERPAK_SOIC8	N-Channel 20V dual Mosfet
S2	SI7236DP	POWERPAK_SOIC8	N-Channel 20V dual Mosfet
S3	SI7236DP	POWERPAK_SOIC8	N-Channel 20V dual Mosfet
S4	SI7236DP	POWERPAK_SOIC8	N-Channel 20V dual Mosfet
ZENER	MM3Z2V4ST1	SOD323	Zener diode

Appendix B

Microcontroller C Code for Discrete Merged Two-Stage Converter

---

Listing B.1: *m2s_code/hard_tiny24_new.c*

```
1  #define F_CPU 8000000UL
2  #include <avr/io.h>
3  #include <util/delay.h>
4  #include <avr/interrupt.h>
5  /*DEFINES */
6
7  #define all_off_a        PORTB=0x80 //turn off all gate drives
8  #define all_off_b        PORTB=0x01 //turn off all gate drives
9  #define series_a         PORTA = 0xA2 //charge in series MLF
10 #define series_b         PORTB = 0x03 //charge in series MLF
11 #define parallel_a       PORTA = 0x50 //discharge in parallel MLF
12 #define parallel_b       PORTB = 0x00 //discharge in parallel MLF
13 int const delay_overlap = 1;
14 int const toggle_delay = 3;
15 int const switch_delay = 50; //max 96 useconds
16
17 void initports (void)
18 {
19 /* Set PORT directions, PA2, PA3 inputs, rest outputs*/
20    DDRA = 0xF3;
21    DDRB = 0xFF;
22
23 //Comparator setup
```

```
24
25 ACSR|=
26     (0<<ACD) |    //Comparator ON
27
28     (1<<ACBG) |   //Connect 1.23V reference to AIN0
29
30     (1<<ACIE) |   //Comparator Interrupt enable
31
32     (0<<ACIC) |   //input capture disabled
33
34     (1<<ACIS1) | //set interrupt on rising output edge
35
36     (1<<ACIS0);
37          SFIOR|=(1<<ACME);
38          ADCSRA&=~(1<<ADEN); //Make sure ADC is turned off
39          ADMUX = ADMUX=0x02; //Poll from ADC2 initially
40
41
42    /* Initialise value */
43          all_off_a;
44          all_off_b;
45
46 }
47 void startup (void)
48
49 {
50
51    // Startup sequence
52    PORTA = 0x00; //turn on s3m2 to charge s1m1 gate driver
53                  //s3m2 is inverted (PA7)
54    _delay_us(switch_delay);
55    _delay_us(switch_delay);
56    _delay_us(switch_delay);
57    _delay_us(switch_delay);
58    PORTA = 0x80; //turn off s3m2
```

```
59      PORTB = 0x03; //turn on s1m1
60      _delay_us(switch_delay);
61      _delay_us(switch_delay);
62      _delay_us(switch_delay);
63      _delay_us(switch_delay);
64            sei(); //enable interrupts
65  }
66
67  int main(void)
68  {
69            initports();
70            startup();
71
72
73      /* Run forever - "for(;;)" is the same as "while(1)" */
74      for (;;) {
75                  // Write value to Port B
76        /*  _delay_us(switch_delay);
77          _delay_us(switch_delay);
78            all_off_a;
79            all_off_b;
80          _delay_us(delay_overlap); //avoid shoot-through
81          //PORTA = 0x20; //charge in series DIP
82            series_a;
83            series_b;
84          _delay_us(switch_delay);
85            all_off_a;
86            all_off_b;
87                  _delay_us(delay_overlap); //avoid shoot-through
88          //PORTA=0x50; //discharge in parallel DIP
89            parallel_a;
90            parallel_b;
91            _delay_ms(500);
92            series_b;
93  //        all_off_a;
```

```
94            _delay_ms(500);
95  */
96
97          }
98  }
99
100 ISR(ANA_COMP_vect)
101 {
102   /*if (MUX) {
103     MUX=1;
104     PORTD=0xF0;
105   }
106   else {
107   MUX=0;
108   PORTD=0x0F;
109   }*/
110   if(ADMUX==0x02) //ADC2 input
111     {
112          series_a;
113          series_b;
114         ADMUX=0x03;
115          _delay_us(toggle_delay);
116
117     }
118   else {
119          parallel_a;
120          parallel_b;
121         ADMUX=0x02;
122          _delay_us(toggle_delay);
123   }
124
125 }
```

Appendix C

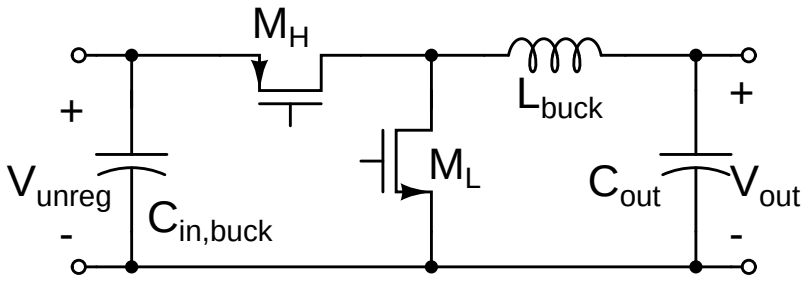
Regulation Stage Device Sizing

This appendix provides a detailed explanation of the design choices in the sizing of regulation stage power switches. The design equations will follow the nomenclature introduced in [74], which considered the optimal design widths of buck converters in integrated CMOS.

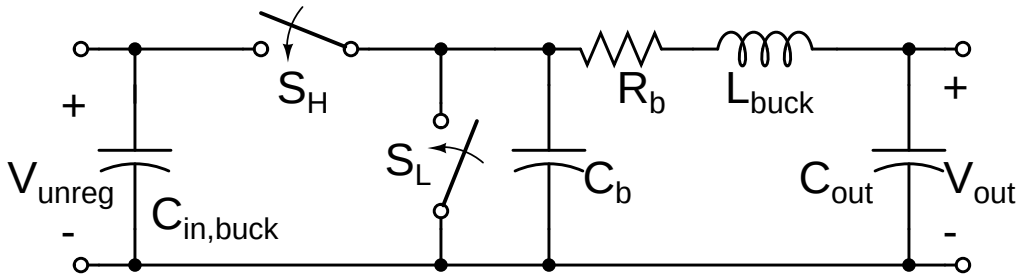
Shown in Figure C.1a is a schematic drawing of the regulation stage, which comprises a buck converter utilized with a PMOS high-side switch (M_H), and an NMOS low-side switch (M_L). While NMOS transistors typically have better intrinsic device performance in most CMOS processes, a PMOS device was used in this work for the top switch to simplify gate drive requirements. An NMOS high-side switch would require a voltage higher than the input voltage to ensure that the device turns on. In conventional converters, this is typically accomplished using a boot-strap circuit with a flying capacitor. It should be noted, however, that the merged two-stage topology does inherently provide voltages that are higher than the input voltage of the regulation stage (in the transformation stage). Thus, it is possible to use one of these voltages to provide power to a high-side gate driver circuitry, thereby removing the need for a flying capacitor and bootstrap circuitry. In this work, however, we chose to employ a PMOS high side switch for simplicity.

Figure C.1b shows an equivalent model of the synchronous buck regulator, which employs ideal switches and two additional components, R_b and C_b which are used to model the total device resistance and capacitance, respectively. The value of R_b is given by [74]:

$$R_b = \frac{R_{L0}}{W_L}(1-D) + \frac{R_{H0}}{W_H}D, \tag{C.1}$$



(a) Schematic drawing of the regulation stage, which utilizes a synchronous buck converter topology.



(b) Model of buck converter for device width optimization.

Figure C.1: *Schematic drawing of synchronous buck converter (a) and equivalent model to compute optimum device widths (b).*

where R_{L0} and R_H) are the effective on-state resistances of the low and high side switch, respectively (often given in units of $\Omega * m$). These can be found for a given CMOS process through simulation, where one measures the normalized (by width) resistance of a transistor in the linear region. We are not able to use the actual values for our process in this discussion as they are proprietary to the commercial foundry. W_L and W_H are the widths of the bottom and top transistor, respectively, and the duty cycle is the duration that the top side switch is on (also given by $D = V_{out}/V_{unreg}$). As can be seen from (C.1), the effective resistance R_b takes into account how long each switch is on, and gives a time-averaged resistance, which corresponds well to loss calculations.

Similarly C_b is given by:

$$C_b = W_L C_{L0} + W_H C_{H0}, \tag{C.2}$$

where C_{L0} and C_{H0} are the parasitic capacitance per unit width (often given in units of F/m). In a CMOS process, the gate capacitance is typically the dominant dynamic loss mechanism, so often only parasitic gate capacitance is considered. The effective gate capacitance can be modelled using process parameters, or extracted from layout. In our case, where different type of transistors are used for the high and low side switch, the values of C_{L0} and C_{H0} are different, as are R_{L0} and R_{H0}.

As outlined in [18], the optimal width ratio of the high-side device width to the low-side device width (α) [74] can be found by a constrained minimization of C_b at a constant R_b, yielding

$$\alpha = \frac{W_H}{W_L} = \sqrt{\frac{D R_{H0} C_{L0}}{(1-D) R_{L0} C_{H0}}}\,. \tag{C.3}$$

The total power loss is a combination of the static loss (P_{res}) and switching loss (P_{cap});

$$P_{loss} = P_{cap} + P_{res} = W_b C_0 V_{IN}^2 f_S + \frac{R_0}{W_b} I_{OUT}^2 \tag{C.4}$$

where

$$C_0 = \frac{C_b}{W_b} = \frac{C_{L0} + C_{H0}\alpha}{1+\alpha} \tag{C.5}$$

and

$$R_0 = R_b W_b = (1+\alpha)\left[(1-D)R_{R0} + \frac{DR_{H0}}{\alpha}\right] . \tag{C.6}$$

assuming that the power loss is only in the MOSFETs. The optimal bridge width (W_{opt}) can be found by minimizing P_{loss} (C.4) and is

$$W_{opt} = \frac{I_{OUT}}{V_{IN}}\sqrt{\frac{R_0}{C_0 f_S}} \tag{C.7}$$

From (C.7), the individual device widths can be found as:

$$W_L = \frac{W_{opt}}{1+\alpha} \tag{C.8}$$

$$W_H = W_L \alpha \tag{C.9}$$

It should be noted that his analysis has not considered additional losses such as inductor core and resistive losses, which will increase overall converter power losses. However, as as shown in [74], these additional losses do not affect the optimum device widths as calculated above.

Appendix D

Type III Compensation Network Calculation

```
%%Type III compensation network calculation, by Robert Pilawa
clear all
hold off
%%%%%Power converter parameters
L=6.25e-9;
C=2e-6; %output capacitor
R=0.25; %load capacitor
Rc=2e-3; %esr of load capacitor
Rl=10e-3; %esr of inductor
D=0.86; %duty ratio
Vin=1.2;
Vout=1;
%%%%%%%%%%%%%%%%%%%%%%%%%%
Vn=0.8; %voltage of negative input terminal of error amplifier, used to calculate Rbias.
P=bodeoptions;
P.FreqUnits = 'Hz';
s=tf('s');
Gvd= Vin/(D*(1+s*L/R+s^2*L*C)); %first order loop gain of LC filter
Gvd2=Vin*(1+s*C*Rc)/(D*(1+s*(C*(Rc+Rl)+L/R)+s^2*L*C)); %second order effects taken into acc
Gc=1;
```

```
Gpwm=2;
T=Gc*Gpwm*Gvd;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Parameters from cadence simulation:
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Gfc_db=-15.52 %current gain at desired cross-over frequency
fc=5e6; %Desired cross-over frequency
PS=-165; %Open loop phase shift at crossover we measure from cadence
PM=50; %Phase margin we want
G=10^(-Gfc_db/20); %calculate required gain boost, used to calculate C2, placing our origin
Boost=PM-PS-90;
Boost_rad=Boost/180*pi;
k=(tan(Boost_rad/4+pi/4))^2;
R1=40e3
Rbias=Vn*R1/(Vout-Vn)
C2=1/(2*pi*fc*G*R1)
C1=C2*(k-1)
R2=sqrt(k)/(2*pi*fc*C1)
R3=R1/(k-1)
C3=1/(2*pi*fc*sqrt(k)*R3)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Figure 1
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Zf=1/(s*C2)*(R2+1/(s*C1))/(1/(s*C2)+R2+1/(s*C1))
Zi=R1*(R3+1/(s*C3))/(R1+R3+1/(s*C3))
Tideal=Zf/Zi;
figure(1)
bodeplot(Tideal,P)
```

```
title('Zf/Zi with ideal amplifier')
grid on

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Figure 4
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
figure(4)
hold on
Tsystem_ideal=Gvd*Gpwm*Tideal;
bode(Gvd*Gpwm,P,'-');
bode(Tsystem_ideal,P,'-.');
legend('uncompensated', 'compensated')
%bodeplot(Tsystem_ideal,P);
%Tsystem_real=Gvd2*Gpwm*T;
%bodeplot(Tsystem_real,P);
grid on;
```

Appendix E

PCB Layout, Detailed Schematic, and Bill of Materials for the Discrete TPV MPPT

This appendix provides schematic and images of the PCB layout for the discrete TPV MPPT converter prototype, as well as bill of materials. The PCB layout was made using EAGLE™ Layout Editor from Cadsoft Computer, Inc. Note that all PCB images here are scaled from their original size to provide better details.

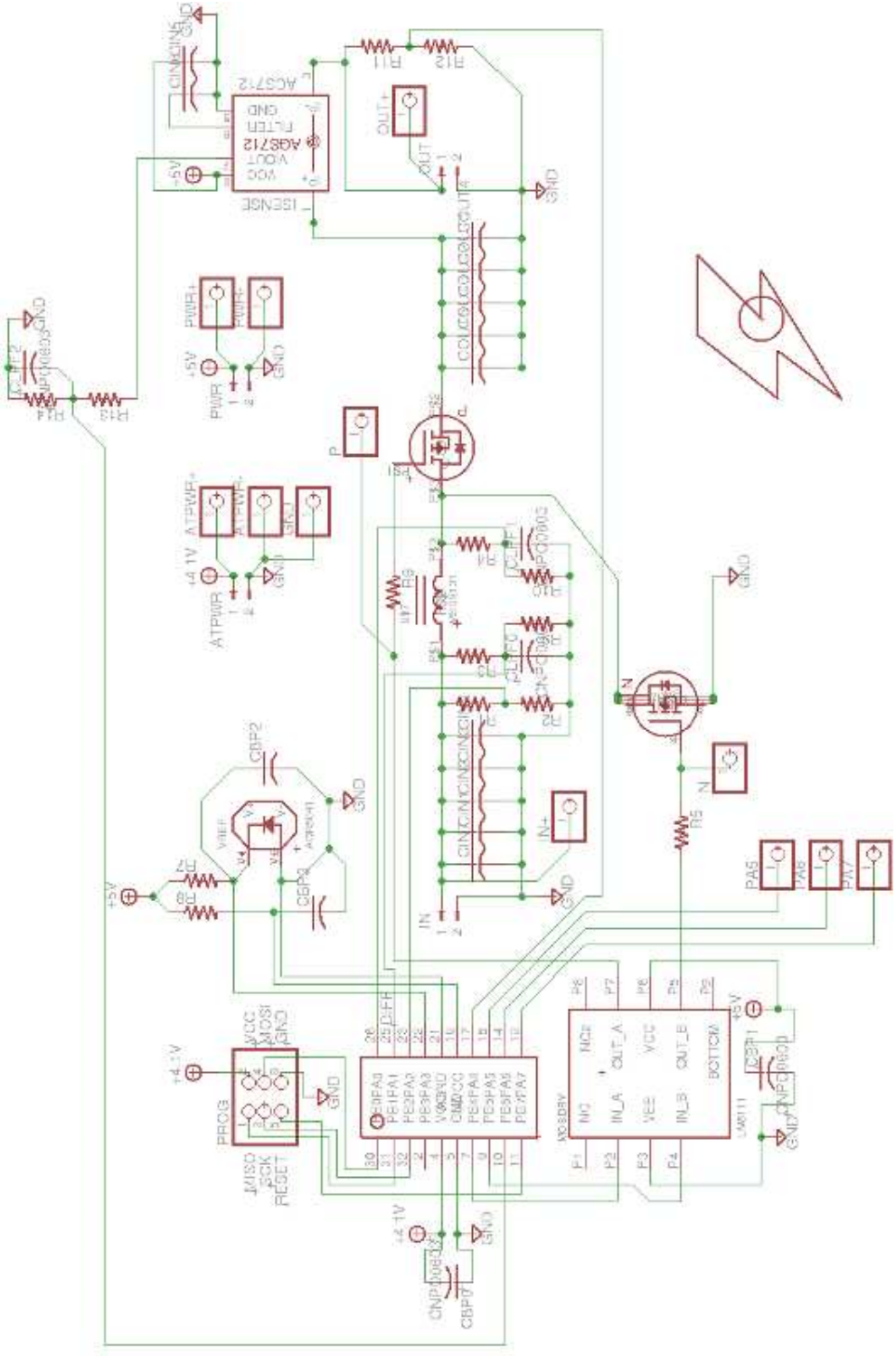


Figure E.1: *Converter schematic drawing. Note that the schematic contains many components that were not implemented. Table 6.3 of Chapter 6 contains a component listing of the experimental prototype.*

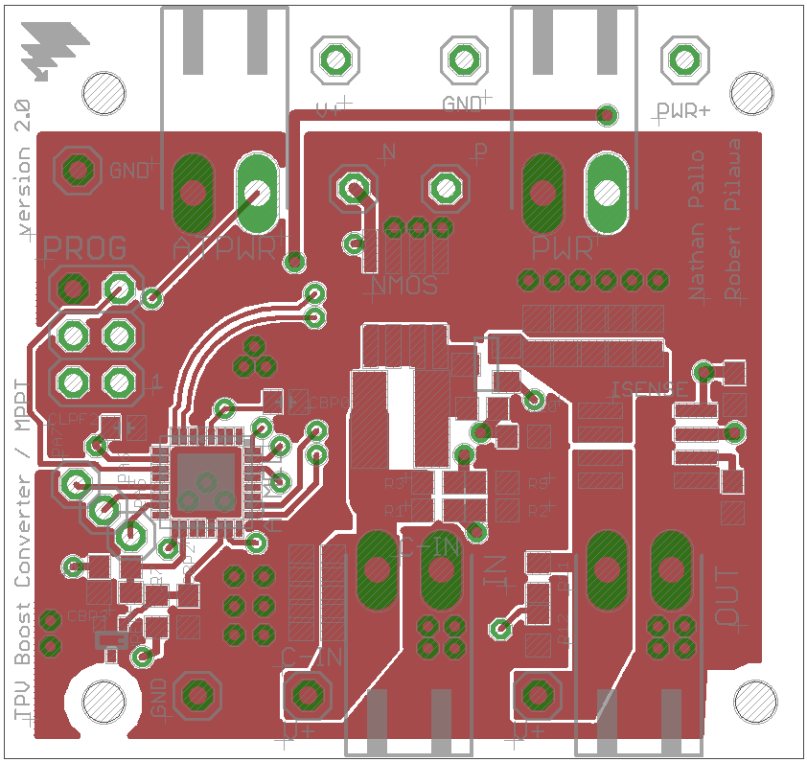


Figure E.2: *Converter PCB layout, top copper, silkscreen, and solder stop layers.*

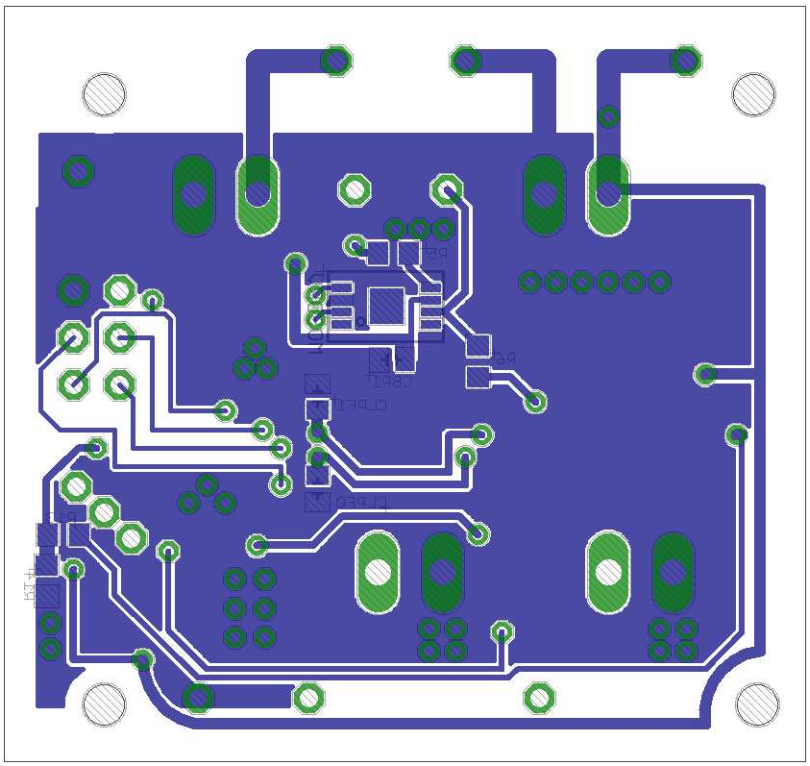


Figure E.3: *Converter PCB layout, bottom copper, silkscreen, and solder stop layers.*

Appendix F

Microcontroller C Code for Discrete TPW MPPT

Listing F.1: *tpv_discrete_code/mppt.c*

```
1  #include "mppt.h"
2  void delayms(uint16_t millis) {
3    while ( millis ) {
4      _delay_ms(1);
5      millis--;
6    }
7  }
8  int main(void)
9  {
10 delayms(2000);
11 SEND_STRING("\n\r");
12 SEND_STRING("MPPT rev 3, by Robert Pilawa\n\r");
13 MPPTInit();
14 SEND_STRING("MPPTInit Completed");
15 SEND_STRING("\n\r");
16 unsigned int test1=620;
17 unsigned long test2=619;
18 unsigned long test=test1*test2;
19    for (;;) // Loop forever
20    {
21                  MPPTSweep();
22                  delayms(5000);
23                  CurrentTracking();
```

```
24      }
25  }
26  void MPPTInit(void)
27  {
28  //USI_UART_Initialise_Transmitter();
29  //USARTInit();
30  SEND_STRING("USI_UART_Init_complete_\n_\r");
31  ADCInit();
32  //CMPInit();
33  PWMInit();
34  //duty_ratio= .5*TIMER1_TOP;
35  direction=-1; //start out by decreasing duty ration
36  sei();
37  //SEND_STRING("MPPTInit complete \n \r");
38  }
39
40
41  void MPPTSweep(void)
42  {
43
44  //Sweep duty cycle and record/spit out duty cycle and power
45  //unsigned int iout_peak=0;
46  unsigned long pin_peak=0;
47
48  PWMWriteDuty(DUTY_MAX);
49  //DUTY_COUNT=DUTY_MAX;
50  PWMEnable();
51  delayms(4000);
52  unsigned int zeropower=0;
53  unsigned int duty_peak=0;
54
55  while (PWMReadDuty() > DUTY_MIN)
56  {
57          iin=ADCReadValueDifferential(1);
58          vin=ADCReadValue(VIN_MUX,1);
```

```
59          pin=vin*iin;
60          //iout=ADCReadValue(IOUT_MUX,20);
61
62          if (pin>pin_peak)
63          {
64          pin_peak=pin;
65          duty_peak=PWMReadDuty();
66          //duty_peak=DUTY_COUNT;
67          }
68          if (pin<1)
69          {
70          zeropower++;
71          }
72          if (zeropower==1)
73          {
74          SEND_STRING("zeropower\n\r");
75          break;
76          }
77          SEND_DATA("  v: ",vin);
78          SEND_DATA("  i: ",iin);
79          SEND_DATA("  p: ",pin);
80          SEND_DATA("  p_pk: ", pin_peak);
81          SEND_DATA("  d: ", PWMReadDuty());
82          SEND_DATA("  d_pk: ",duty_peak);
83          SEND_STRING("\n\r");
84          PWMWriteDuty(PWMReadDuty()-5);
85          //DUTY_COUNT=DUTY_COUNT-5;
86          delayms(2000);
87  }
88  PWMWriteDuty(duty_peak);
89  }
90
91  void CurrentTracking(void)
92  {
93  delayms(1000);
```

```
94  SEND_DATA("  DC: ",PWMReadDuty());
95  SEND_STRING("\n\r");
96  vin=ADCReadValue(VIN_MUX,30)/30;
97  iin=ADCReadValueDifferential(30)/30;
98  vout=ADCReadValue(VOUT_MUX,1);
99  pin=vin*iin;
100 //To setup initial condition, move in same direction for first step.
101 pin_old=pin*0.9;
102 int count=0;
103 int direction_sum=0;
104 for (;;) //Loop into MPPT mode
105         {
106                 delayms(4000);
107         //for(int q=0;q++;q<10);
108         //{
109                 vin=ADCReadValue(VIN_MUX,16);
110                 iin=ADCReadValueDifferential(32768)/256;
111                 pin=vin*iin;
112                 vout=ADCReadValue(VOUT_MUX,1);
113                 vin_adjusted=vin*VREF/ADC_MAX*VIN_DIVIDER*100;
114                 vout_adjusted=vout*VREF/ADC_MAX*VOUT_DIVIDER*100;
115                 //iout=ADCReadValue(IOUT_MUX,30);
116                 count++;
117                 if(pin < pin_old) //if the new power we measure it smaller
                        than old one, we're going the wrong way.
118                         {
119                         direction=-direction; //change direction
120                         }
121
122         //}
123         if(count>0)
124                 {
125                 count=0;
126                 SEND_DATA("vout: ",vout_adjusted);
127                 SEND_DATA("  iin: ",iin/128);
```

```
128                     //SEND_DATA("    iout: ",iout);
129                     //SEND_DATA("    iout_old: ",iout_old);
130                     SEND_DATA("  pin: ",pin);
131                     SEND_DATA("  pin_old: ",pin_old);
132                     SEND_DATA("  DC: ",DUTY_COUNT);
133                     SEND_DATA("  dir: ",direction);
134                     SEND_STRING("\n\r");
135                     }
136             //iout_old=iout;
137             pin_old=pin;
138             PWMWriteDuty(PWMReadDuty()+1*direction);
139             }
140 }
```

Listing F.2: *tpv_discrete_code/mppt.h*

```
1  #ifndef MPPT_H
2  #define MPPT_H
3
4  #include <avr/io.h>
5  #include <util/delay.h>
6  //#include <stdlib.h>
7  #include <avr/interrupt.h>
8  #include <avr/pgmspace.h>
9  #include <avr/sleep.h>
10 #include <avr/wdt.h>
11 #include "USI_UART_TINY861.h"
12 //#include "USART.h"
13 #include "AVR035.h"
14 #include "ADC.h"
15 #include "PWM.h"
16 #include "CMP.h"
17
18 //Must set one on and one off, changes ADCReadValue
19 #define DIGITAL_ERROR_CORRECTION_OFF 1
20 #define DIGITAL_ERROR_CORRECTION_ON 0
```

```
21
22 #define VOLTAGE_REGULATION_MINIMUM 10 //To what ADC value should our regulated
       voltage be accurate
23
24 //From TPV
25 #define VOUT_TARGET 4
26 #define VOUT_DIVIDER 4
27 #define VIN_DIVIDER 1
28 #define VIN_MIN 0.3
29 #define VOUT_MAX 10
30 #define VOUT_MIN 0.4
31 #define VIN_MUX 1 //Same as VL in this case
32 //#define VIN_MUX 3 //VIN is sampled on PA4, which is ADC3
33 #define VOUT_MUX 3 //VOUT is sampled on PA4, which is ADC3
34 #define IOUT_MUX 8 //IOUT is sampled by PB5, which is ADC8. IOUT is sampled by
       hall effect sensor
35 #define VREF 1.1//2.54
36 #define ADC_CENTER 512
37 #define ADC_MAX 1024
38 #define DUTY_MAX 0.9*TIMER1_TOP
39 #define DUTY_MIN 0.1*TIMER1_TOP
40 #define F_PLL 64000000
41 #define TIMER1_TOP F_PLL/FS //maximum value is 1024 in this implementation, we
       're doing 10-bit.
42 #define FS 300000 //Converter switching frequency, change here. Don't go below
       251kHz
43 //Note, VL and VH are swapped on board, this may not be the case in the future
44 #define VL_MUX 1 //Inductor VL is sampled by PA1, which is ADC1
45 #define VH_MUX 0 //Inductor VH is same as VIN, which is ADC0s
46 #define DIRECTION 0 //status_char bit 0 will be used to determine direction. 0
       is down, 1 is up.
47 #define SEND_STRING USI_UART_Transmit_String //Let us change to other string
      sending function at a later time.
48 //#define SEND_STRING EmptySendString
49 #define SEND_DATA USI_UART_Transmit_Data
```

```
50 //#define SEND_DATA EmptySendData
51 #define MIN_ADC_DIFF 4 //used in ADCReadValue for digital error correction.
       The smallest value two subsequent readings may differ before we discard
       both of them.
52 unsigned int over_voltage;
53 volatile unsigned long vout;
54 volatile unsigned long vin;
55 volatile unsigned long vout_adjusted;
56 volatile unsigned long vin_adjusted;
57 volatile unsigned long iin;
58 volatile unsigned long iout;
59 volatile unsigned long iout_old;
60 volatile unsigned long pin_old;
61 volatile unsigned long pin;
62 volatile signed int direction; //1 is increase duty ratio, -1 is decrease
63 volatile unsigned int duty_count;
64 void MPPTInit(void);
65 void MPPTSweep(void);
66 void CurrentTracking(void);
67 #endif
```

Appendix G

PCB Layout, Detailed Schematic, and Bill of Materials for the Integrated TPV MPPT and Associated Test Board

---

This appendix provides schematic and images of the PCB layout for the integrated TPV MPPT test PCB board, as well as bill of materials. The PCB layout was made using EAGLE™ Layout Editor from Cadsoft Computer , Inc. Note that all PCB images here are scaled from their original size to provide better details.

Table G.1: *Bill of Materials for TPV integrated test board*

Ref Des	Part No.	Description	Function
U$12, U$8	AD8276	Low power unity-gain difference amplifier	Ibias
U$9, U$15	AD8655	IC OPAMP R-R CMOS 28MHz	Ibias
IC1	AD780ARZ	IC REFERENCE PREC 2.5/3.0V 8SOIC	Voltage reference
VREF_CTRL	3223W-1-504E	TRIMPOT 500K 2MM TOP ADJ SMD	Current control
CURRENT_CTRL	3223W-1-504E	TRIMPOT 500K 2MM TOP ADJ SMD	Current control
LED1-5	LTST-C191KRKT	LED Super RED CLR THIN	Red LED
C1-16		Capacitor, 1uF,25V, X5R	Bypass caps
X1, X2, X3	1755752	CONN HEADER VERT 4POS 5.08MM	Connector
R20		0 Ohm resistor	
R8,R7,R13		0 Ohm resistor	
R4, R6, R12		30k resistor	
R5, R9, R14		20k resistor	

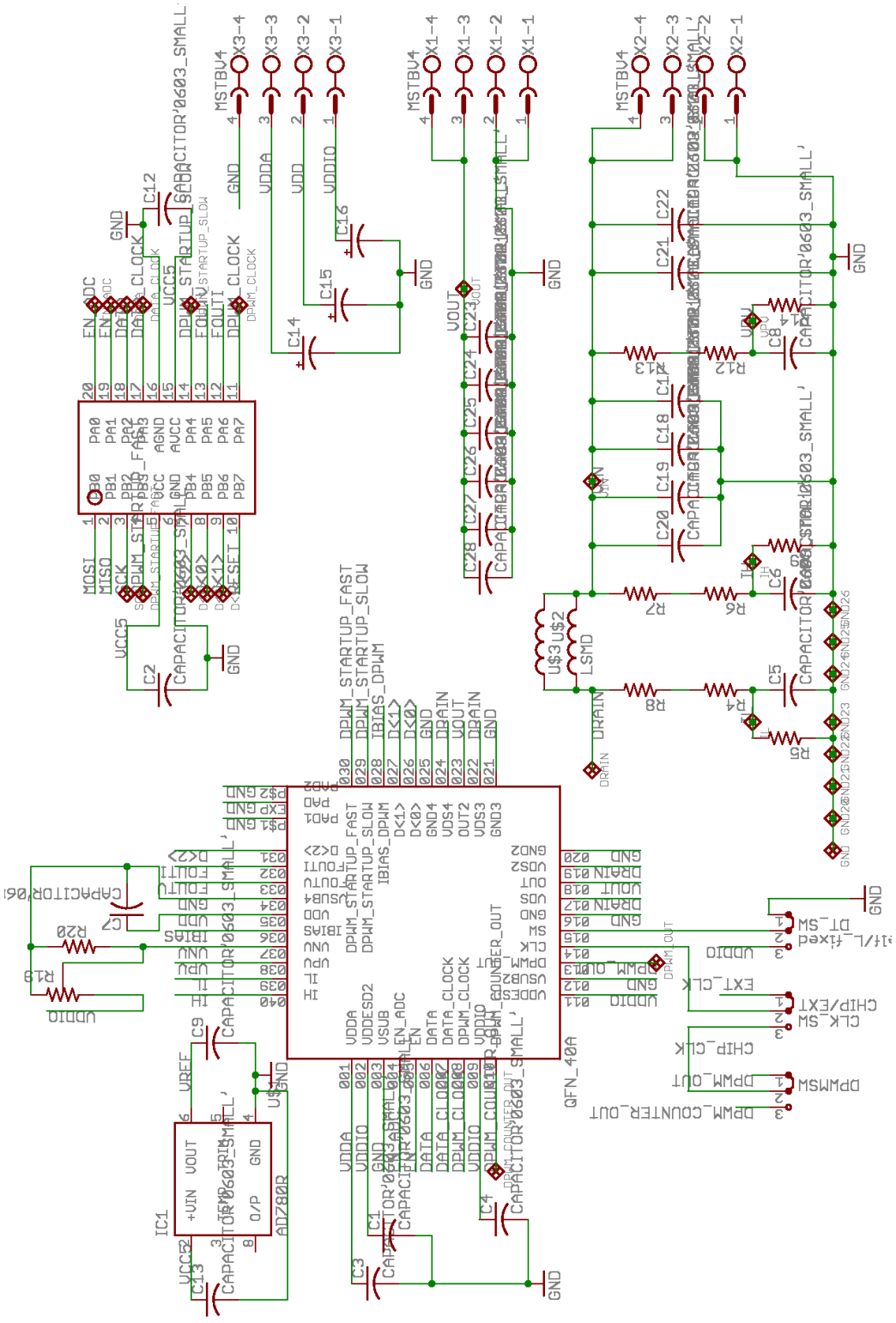


Figure G.1: *Eagle schematic drawing of TPV test board, sheet 1.*

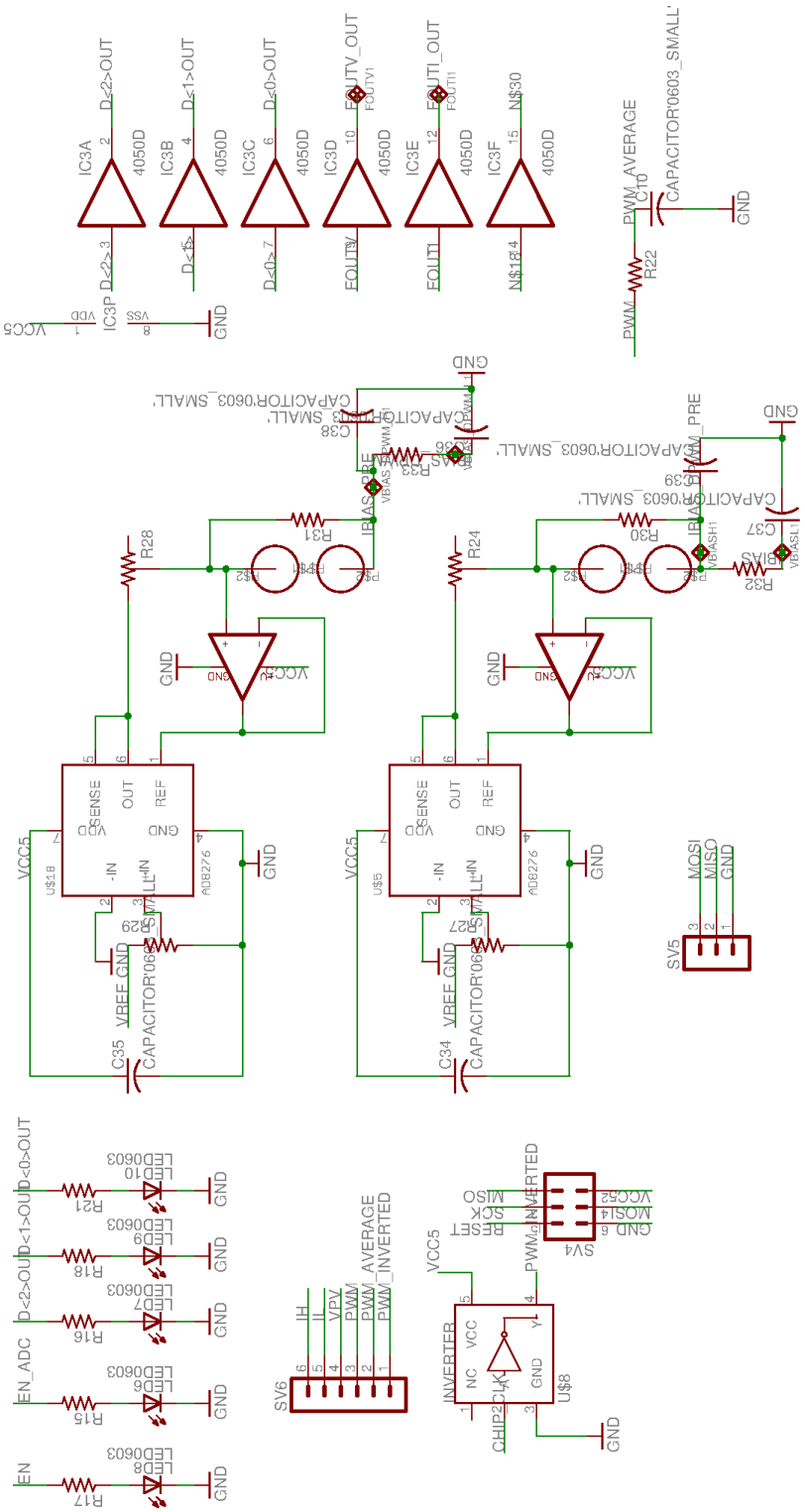


Figure G.2: *Eagle schematic drawing of TPV test board, sheet 2.*

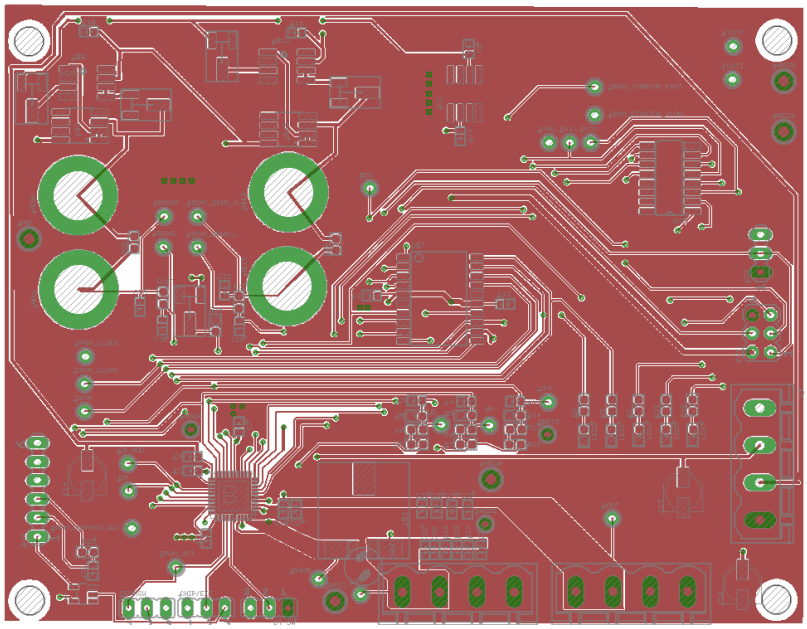


Figure G.3: *Converter PCB layout, top copper, silkscreen, and solder stop layers.*

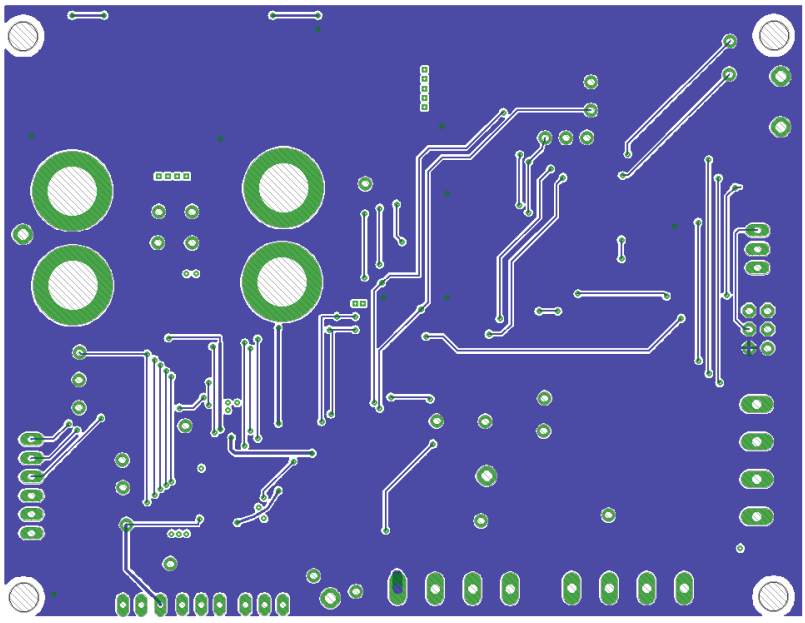


Figure G.4: *Converter PCB layout, bottom copper, silkscreen, and solder stop layers.*

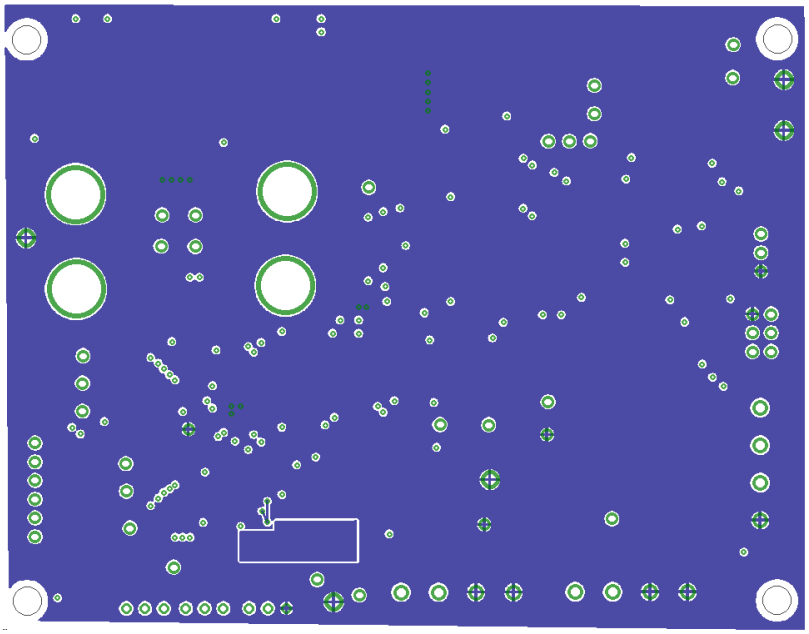


Figure G.5: *Converter PCB layout, layer 2 copper.*

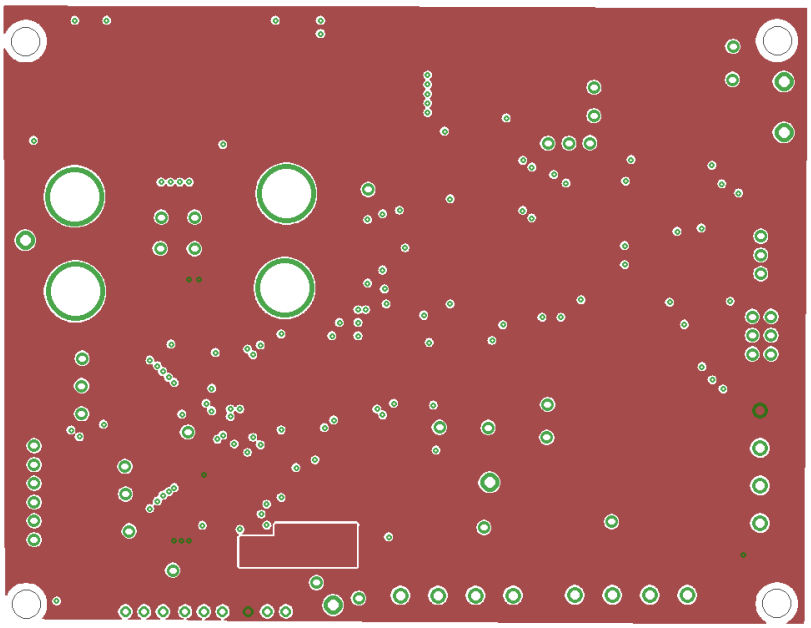


Figure G.6: *Converter PCB layout, layer 3 copper.*

Appendix H

PCB Layout, Detailed Schematic, and Bill of Materials for Distributed MPPT Hardware

---

This appendix provides schematic and images of the PCB layout for the distributed MPPT converter prototype, as well as bill of materials. The PCB layout was made using EAGLE™ Layout Editor from Cadsoft Computer , Inc. Note that all PCB images here are scaled from their original size to provide better details. The cost was for quantities of less than 5,000, either as listed directly from the manufacturer, or from Digi-Key.

Table H.1: *MPPT Bill of Materials and Cost*

Ref.	Type	Value	Package	Part Number	Cost
C4	Ceramic 16V X5R	10uF	0805	GRM21BR61C106KE15L	0.060
C5	Ceramic 16V X5R	10uF	0805	GRM21BR61C106KE15L	0.060
C6	Ceramic 16V X5R	10uF	0805	GRM21BR61C106KE15L	0.060
C13	Ceramic 16V X5R	10uF	0805	GRM21BR61C106KE15L	0.060
C14	Ceramic 16V X5R	10uF	0805	GRM21BR61C106KE15L	0.060
L	Ferrite	10 uH	SMD	SER1360-103KL_	0.530
PWM_IC	DrMOS		Power 56	FDM6704A	1.620
C7	Ceramic 6.3 V	4.7uF	0603	GRM155R60J475ME87D	0.070
C9	Ceramic 16V X5R	1uF	0402	C1005X5R1C105M	0.010
AVR	Microcontroller		QFP32	ATTiny861	1.670
C12	Ceramic 16V X5R	1uF	0402	C1005X5R1C105M	0.010
C1	Ceramic 16V X5R	1uF	0402	C1005X5R1C105M	0.010
R7	Resistor, 1%	100k	0402	ERJ-2RKF1003X	0.004
R1	Resistor, 1%	10k	0402	ERJ-2GEJ103X	0.002
R8	Resistor, 1%	100k	0402	ERJ-2RKF1003X	0.004
R9	Resistor, 1%	10k	0402	ERJ-2GEJ103X	0.002
R3	Resistor, 1%	100k	0402	ERJ-2RKF1003X	0.004
R4	Resistor, 1%	10k	0402	ERJ-2GEJ103X	0.002
C3	Ceramic 16V X5R	1uF	0402	C1005X5R1C105M	0.010
C11	Ceramic 16V X5R	1uF	0402	C1005X5R1C105M	0.010
C16	Ceramic 16V X5R	1uF	0402	C1005X5R1C105M	0.010
MPPT Cost					4.268

Table H.2: *MPPT Communication Components Bill of Materials*

Ref.	Type	Value	Package	Part Number
VCCBYPASS	DC-DC isolated	VBSD1-S5-S5-SIP	SIP	VBSD1-S5-S5-SIP
R6	SMT	220 Ohm	0603	
I2C_ISO	I2C isolator			Si8400
R2, R5	SMT	4.7k	0603	
BYPASS	MOSFET	Si4448DY	SO-8	SI4448DY-T1-E3

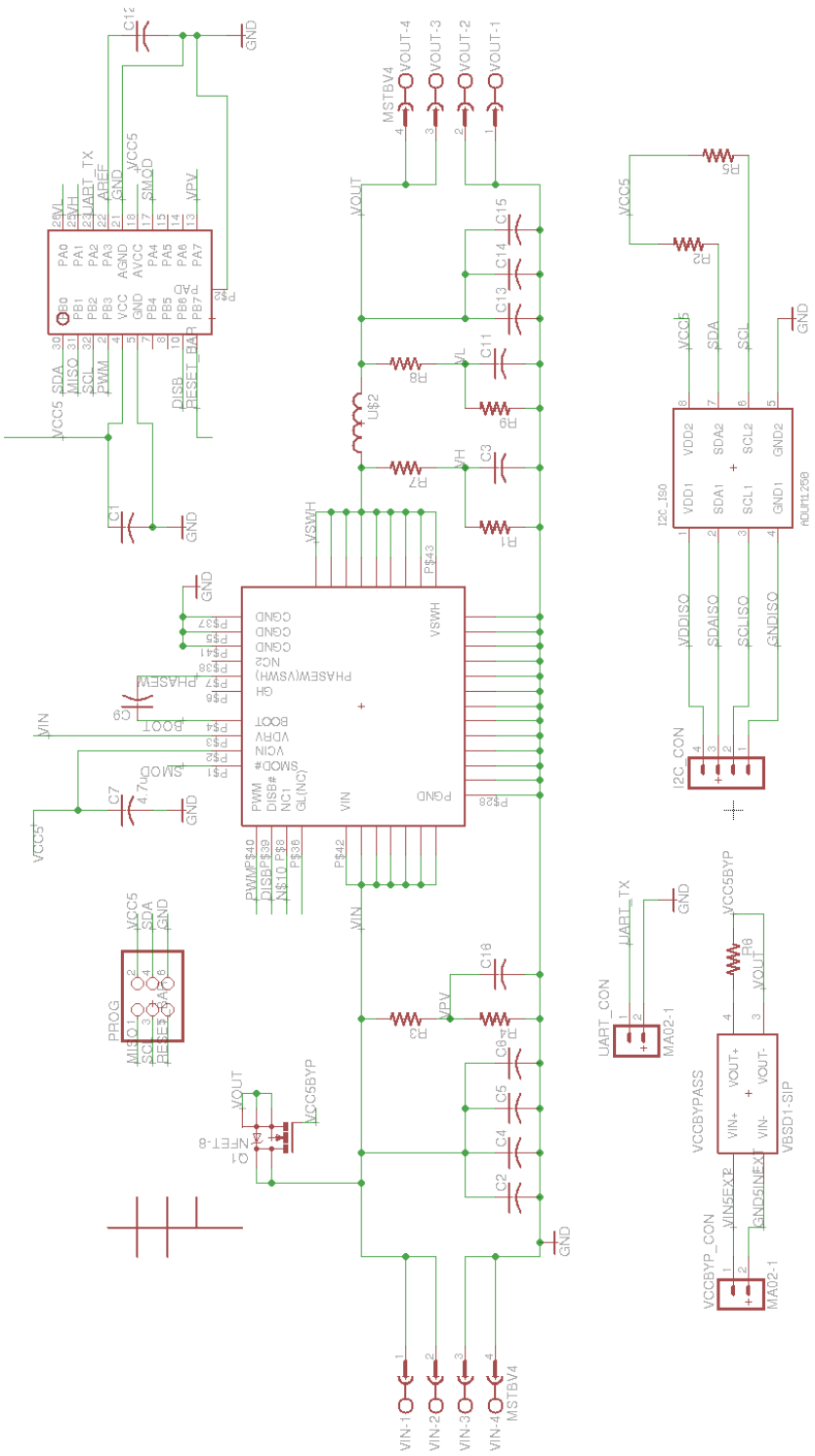


Figure H.1: *Converter schematic drawing.*

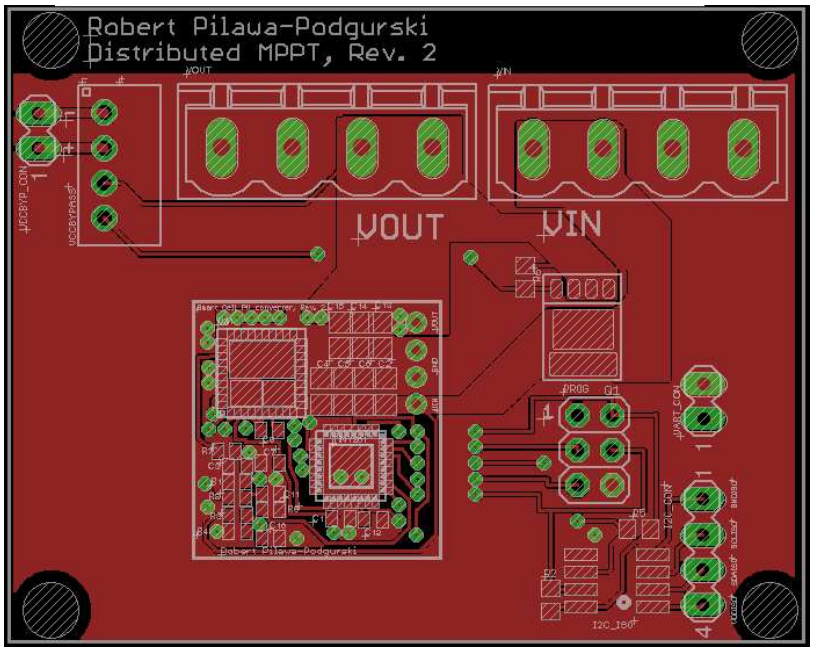


Figure H.2: *Converter PCB layout, top copper, silkscreen, and solder stop layers.*

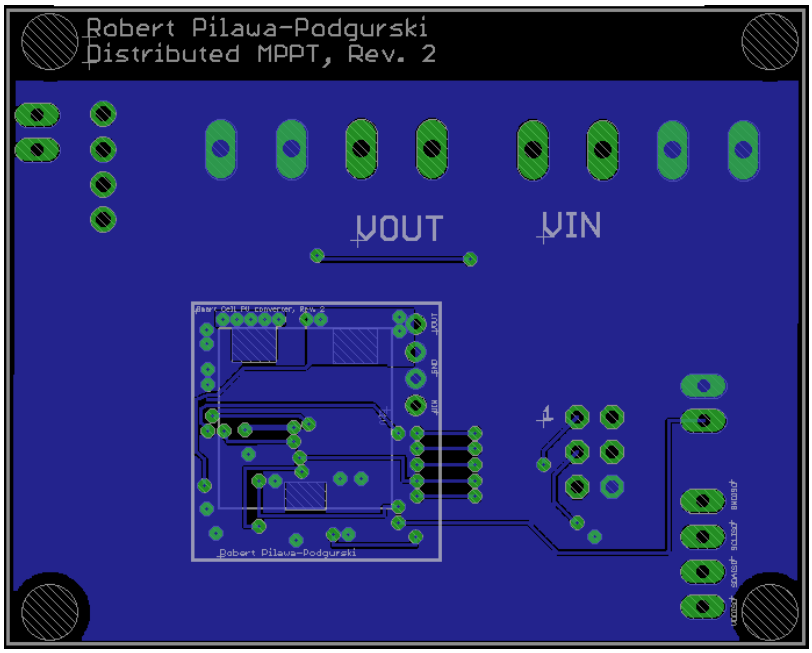


Figure H.3: *Converter PCB layout, bottom copper, silkscreen, and solder stop layers. The SER1360 inductor from coilcraft is the only component on the bottom side.*

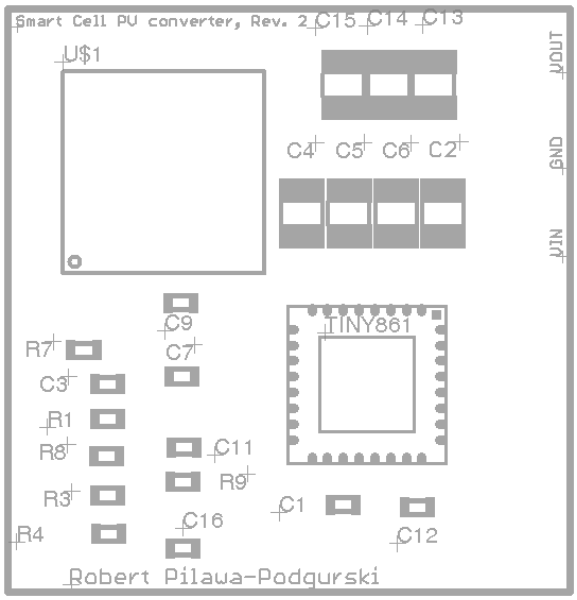


Figure H.4: *Converter top-layer silkscreen for MPPT only (without I2C chip, bypass-transistors, and connectors).*

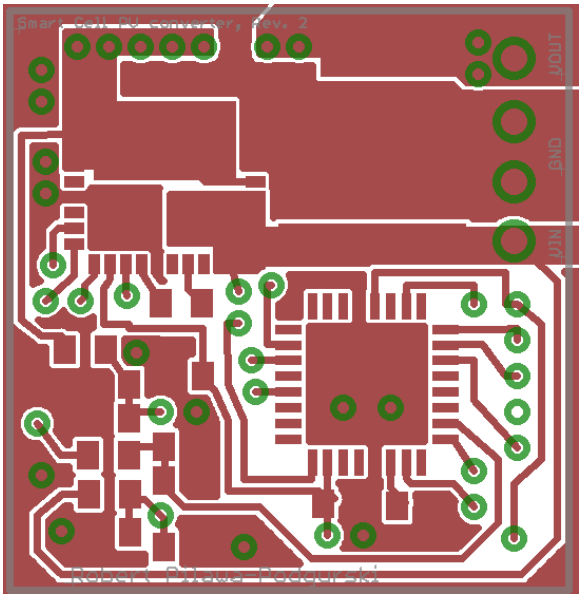


Figure H.5: *Converter top-layer copper, pads, and vias for MPPT only (without I2C chip, bypass-transistors, and connectors).*

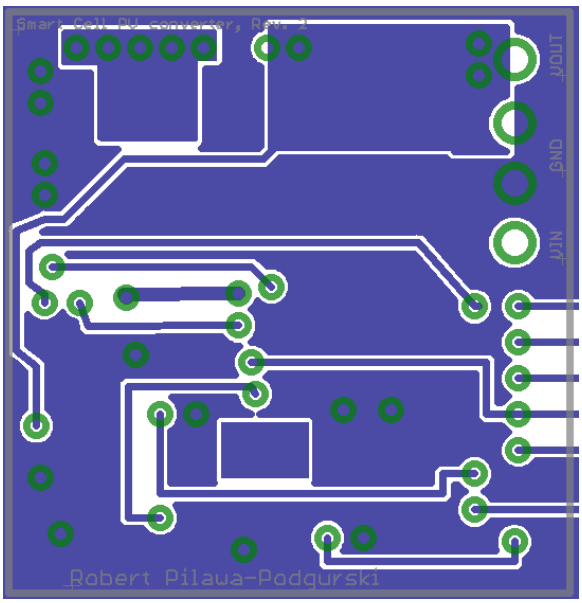


Figure H.6: *Converter bottom-layer copper, pads, and vias for MPPT only (without I2C chip, bypass-transistors, and connectors).*

Appendix I

Microcontroller C Code for Distributed MPPT

Listing I.1: *solar_code/mppt.c*

```
1  #include "mppt.h"
2  void delayms(uint16_t millis) {
3    while ( millis ) {
4      _delay_ms(1);
5      millis--;
6    }
7  }
8  //uint8_t EEMEM chip_id;
9  //unsigned char status_byte=0;
10 int main(void)
11 {
12 //Uncomment this out to assign identifier to chip.
13 ////////////////EEPROM programming///////////////
14 /*
15 uint8_t identifier_write=1;
16 uint8_t rev_write=1;
17 uint8_t i2c_address=24;
18
19 eeprom_write_byte((uint8_t*)EEPROM_CHIP_ID, identifier_write);
20 eeprom_write_byte((uint8_t*)EEPROM_REV, rev_write);
21 eeprom_write_byte((uint8_t*)EEPROM_I2CADDRESS, i2c_address);
22 */
23 ////////////////////////////////////////////////
```

```
24
25 //Handle watchdog timer immediately
26 wdt_reset();
27 wdt_enable(WDTO_8S);
28 wdt_reset();
29 unsigned int mcusr_mirror;
30 mcusr_mirror = MCUSR;
31 MCUSR = 0;
32 wdt_reset();
33 delayms(200);
34 chip_id_read=eeprom_read_byte((uint8_t*)EEPROM_CHIP_ID);
35 rev_read=eeprom_read_byte((uint8_t*)EEPROM_REV);
36 myI2Caddress=eeprom_read_byte((uint8_t*)EEPROM_I2CADDRESS);
37 //myI2Caddress=0x22;
38 delayms(200);
39 wdt_reset();
40 #ifdef UART
41 SEND_STRING("Single-Cell MPPT rev 1, by Robert Pilawa\n\r");
42 SEND_DATA("i2c address:",myI2Caddress);
43 SEND_STRING("\n\r");
44 #endif
45 MPPTInit();
46 USI_TWI_Slave_Initialise(myI2Caddress);
47 //sei();
48 delayms(500);
49 wdt_reset();
50 //SEND_DATA("myI2Caddress:",myI2Caddress);
51 unsigned int loopcount=0;
52 //unsigned char tempbyte=0;
53 #ifdef DEBUG
54 SEND_STRING("Single-Cell MPPT rev 1, by Robert Pilawa\n\r");
55 #endif
56 #ifdef UART
57 SEND_STRING("Single-Cell MPPT rev 1, by Robert Pilawa\n\r");
58 SEND_DATA("i2c address:",myI2Caddress);
```

```
59  SEND_STRING("\n\r");
60  #endif
61  SEND_DATA("mcusr:",mcusr_mirror);
62      for (;;) // Loop forever
63      {
64                      loopcount++;
65                      delayms(1);
66                      wdt_reset();
67                      //SEND_STRING("Single-Cell MPPT rev 1, by Robert Pilawa\n\r");
68                      if (loopcount>MAXLOOPCOUNT) //Overflowing should mean that the
                            device is not receiving data as fast as it should, so we
                            need to reinitialize.
69                      {
70                              SEND_STRING("Loopcount overflow\n\r");
71                              USI_TWI_Slave_Initialise(myI2Caddress);
72                              loopcount=0;
73                              delayms(50);
74                              PWMDisable();
75                      }
76
77                      #ifdef I2C
78
79                      if ( I2CRead()==1) //I2CRead() returns 1 after a successful
                            read
80                              {
81                              loopcount=0;
82                              }
83
84                      #endif
85
86      }
87  }
88
89  void MPPTInit(void)
90  {
```

```
91  ////////////////////////////
92  //Setup IO Ports
93  ////////////////////////////
94  //PORTB output ports are PB3 (PWM), PB6 (DISB)
95  DDRB=(1<<DDB3)|(1<<DDB6);
96  //PORTA output pots are PA2 (UART_TX), PA4 (SMOD)
97  DDRA=(1<<DDA2)|(1<<DDA4);
98  //Unused PORTA pins are PA5, PA6. Setup as inputs with pull-up resistor
        enabled
99  PORTA= (1<<PA5)|(1<<PA6);
100 //Unused PORTB pins are PB4, PB5. Setup as inputs with pull-up resistor
        enabled
101 PORTA= (1<<PB4)|(1<<PB5);
102 //Setup SMOD to enable both bottom and top switch to turn on/off:
103 PORTA |= (1<<PA4);
104 ADCInit();
105 PWMInit();
106 PWMDisable();
107 #ifdef I2C
108 USI_TWI_Slave_Initialise(myI2Caddress);
109 #endif
110 PWMWriteFrequency(200);
111 sei();
112 }
113
114 #ifdef I2C
115 void I2CWrite(void)
116 {
117 char send_byte=0x33;
118 for(;;)
119         {
120         USI_TWI_Transmit_Byte(send_byte);
121         delayms(1000);
122         }
123 }
```

```
124
125 unsigned int I2CRead(void) //return 1 if successful read, return 0 is
        unsuccessful
126 {
127 unsigned int timeout_counter=0;
128
129 unsigned char received_bytes[PACKET_SIZE-1];
130 unsigned char received_counter=0; //if all bytes are received, timeout_counter
        will be 0 at end of reception
131 unsigned char transmit_byte1=0; //MSB
132 unsigned char transmit_byte2=0; //LSB
133 unsigned int duty;
134 unsigned int parameter=0;
135 if( USI_TWI_Data_In_Receive_Buffer() ) //Master always send two times three
        bytes at a time. Each set of three bytes is repeating
136         //It is the job of the receiver check that bytes 1-3==3-6
137
138     {
139         //The first byte is what type of command, the next two bytes is
              command-specific parameters. MSB first
140
141
142         while ( (received_counter < PACKET_SIZE) && (timeout_counter <
              TIMEOUT_MAX) )
143                 {
144                 //byte_index++;
145                 if( USI_TWI_Data_In_Receive_Buffer() )
146                         {
147                         received_bytes[received_counter]=USI_TWI_Receive_Byte
                              ();
148                         received_counter++;
149                         }
150                 timeout_counter++;
151                 }
152         #ifdef DEBUG
```

```
153         SEND_DATA("outside of while loop, received_counter: ",received_counter
                );
154         SEND_STRING("\n\r");
155         #endif
156         if (received_counter < PACKET_SIZE) //We lost some messages
157                 {
158                 #ifdef DEBUG
159                         SEND_DATA("timed_out, received_counter: ",
                                received_counter);
160                         SEND_STRING("\n\r");
161                 #endif
162                 return 0; //Return
163                 }
164         if ( (received_bytes[0] != received_bytes[3]) || (received_bytes[1] !=
                received_bytes[4]) || (received_bytes[2] != received_bytes[5]) )
165                 {
166                 #ifdef DEBUG
167                         SEND_STRING("Corrupt message received");
168                         SEND_STRING("\n\r");
169                 #endif
170                 USI_TWI_Transmit_Byte(ERROR_BYTE);
171                 USI_TWI_Transmit_Byte(ERROR_BYTE);
172                 return 0;
173                 }
174         parameter = received_bytes[2];
175         parameter |= received_bytes[1] << 8;     //MSB
176         #ifdef DEBUG
177         SEND_DATA("parameter: ", parameter);
178         SEND_STRING("\n\r");
179         #endif
180
181         if (received_bytes[0]==DUTY_COMMAND) //Duty cycle command
182                 {
183                 #ifdef DEBUG
184                 SEND_STRING("Setting duty cycle");
```

```
185                 SEND_STRING("\n\r");
186                 #endif
187
188                 duty=parameter;
189                 PWMWriteDuty(duty);
190                 #ifdef DEBUG
191                 SEND_DATA("PWMWriteDuty :", duty);
192                 SEND_STRING("\n\r");
193                 #endif
194
195                 transmit_byte1=received_bytes[1];
196                 transmit_byte2=received_bytes[2];
197                 }
198         else if (received_bytes[0]==VOLTAGE_COMMAND) //Read back voltage
199         {
200                 #ifdef DEBUG
201                 SEND_STRING("Reading back voltage");
202                 SEND_STRING("\n\r");
203                 #endif
204                 vout=ADCReadValue(VOUT_MUX, parameter);
205                 transmit_byte1=(vout>>8); //higher order bits
206                 transmit_byte2=(unsigned char)vout;
207                 #ifdef DEBUG
208                 SEND_DATA(" transmit_byte1: ", transmit_byte1);
209                 SEND_DATA(" transmit_byte2: ", transmit_byte2);
210                 SEND_DATA(" read voltage: ", vout);
211                 SEND_STRING("\n\r");
212                 #endif
213                 }
214
215     else if (received_bytes[0]==CURRENT_COMMAND) //Read back current
216                 {
217                 #ifdef DEBUG
218                 SEND_STRING("Reading back current");
219                 SEND_STRING("\n\r");
```

```
220                 #endif
221                 iout=ADCReadValueDifferential(parameter);
222                 transmit_byte1=(iout>>8); //higher order bits
223                 transmit_byte2=(unsigned char)iout;
224                 }
225         else if (received_bytes[0]==CLEAR_BUFFER_COMMAND) //Clear buffers
226                 {
227                 #ifdef DEBUG
228                 SEND_STRING("Clearing buffer");
229                 SEND_STRING("\n\r");
230                 #endif
231
232                 Flush_TWI_Buffers();
233                 USI_TWI_Slave_Initialise(myI2Caddress);
234                 return 0;
235                 }
236         else if (received_bytes[0]==ENABLE_CONVERTER) //Enable converter
237                 {
238                 PWMEnable();
239                 transmit_byte1=received_bytes[1];
240                 transmit_byte2=received_bytes[2];
241                 }
242         else if (received_bytes[0]==VOLTAGE_IN_COMMAND) //Read back input
                    voltage
243         {
244                 #ifdef DEBUG
245                 SEND_STRING("Reading back voltage");
246                 SEND_STRING("\n\r");
247                 #endif
248                 vin=ADCReadValue(VIN_MUX, parameter);
249                 transmit_byte1=(vin>>8); //higher order bits
250                 transmit_byte2=(unsigned char)vin;
251                 #ifdef DEBUG
252                 SEND_DATA(" transmit_byte1: ", transmit_byte1);
253                 SEND_DATA(" transmit_byte2: ", transmit_byte2);
```

```
254                 SEND_DATA(" read input voltage: ", vin);
255                 SEND_STRING("\n\r");
256                 #endif
257                 }
258
259         else if (received_bytes[0]==PING_COMMAND) //Pinged by master, respond
260                 {
261                 transmit_byte1=0;
262                 transmit_byte2=0;
263                 }
264
265         else if (received_bytes[0]==BYPASS_ENABLE_COMMAND) //Bypass, shut off
                converter completely
266                 {
267                 PWMDisable();
268                 }
269         else if (received_bytes[0]==BYPASS_DISABLE_COMMAND) //Disable bypass
270                 {
271                 PWMDisable();
272                 //status_byte=STATUS_BYTE_BYPASS;
273                 }
274
275         else
276                 {
277                 SEND_STRING("Unknown message received");
278                 SEND_STRING("\n\r");
279                 transmit_byte1=255;
280                 transmit_byte2=255;
281                 return 0;
282                 }
283
284         #ifdef DEBUG
285         SEND_STRING("Sending back data");
286         SEND_STRING("\n\r");
287         #endif
```

```
288         USI_TWI_Transmit_Byte(received_bytes[0]);
289         USI_TWI_Transmit_Byte(transmit_byte1);
290         USI_TWI_Transmit_Byte(transmit_byte2);
291         USI_TWI_Transmit_Byte(received_bytes[0]);
292         USI_TWI_Transmit_Byte(transmit_byte1);
293         USI_TWI_Transmit_Byte(transmit_byte2);
294         return 1;
295         }
296 //Flush_TWI_Buffers();
297 return 0;
298
299 }
300
301 #endif
302
303 void MPPTTrack(void)
304 {
305 //unsigned int vout_old;
306 vout=ADCReadValue(VOUT_MUX,VOUT_READINGS);
307 vout_old=vout*0.9;
308 signed char direction=1;
309 PWMWriteDuty(500);
310 PWMWriteFrequency(200);
311 delayms(5000);
312 for(;;)
313         {
314         vout=ADCReadValue(VOUT_MUX,VOUT_READINGS);
315
316         if(vout <= vout_old) //if the new voltage we measure it smaller than
              old one, we're going the wrong way.
317             {
318             direction=-direction; //change direction
319             }
320         vout_old=vout;
321         PWMWriteDuty(PWMReadDuty()+6*direction);
```

```
322             delayms(1);
323             }
324 }
```

Listing I.2: *solar_code/mppt.h*

```
1  #ifndef MPPT_H
2  #define MPPT_H
3
4  #include <avr/io.h>
5  #include <util/delay.h>
6  //#include <stdlib.h>
7  #include <avr/interrupt.h>
8  #include <avr/pgmspace.h>
9  #include <avr/eeprom.h>
10 #include <avr/sleep.h>
11 #include <avr/wdt.h>
12 #include "USI_UART_TINY861_nopb3.h"
13 //#include "USART.h"
14 #include "AVR035.h"
15 #include "ADC.h"
16 #include "PWM.h"
17 //=============================================================
18 // PINOUT
19 //
20 //PB0   SDA (I2C)
21 //PB1   12VENABLED (digital input that is high when 12V is on, and converter
       should be bypassed, enter sleep mode)
22 //PB2   SCL (I2C)
23 //PB3   PWM (drive buck, this is OC1B)
24 //PB4   CLK_BAR (for dickson charge pump)
25 //PB5   CLK (for dickson charge pump)
26 //PB6   DISB (for disabling buck chip)
27 //PB7   RESET_BAR (charged through RC circuit)
28
29 //PA0   VL (low side of inductor averaged sample)
```

```
30 //PA1   VH (high side of inductor averaged sample)
31 //PA2   Empty
32 //PA3   AREF (decoupled with capacitor)
33 //PA4   SMOD (turn off bottom switch always, use integrated diode only) set to
       high to run sync. rect.
34 //PA5   VPV (input voltage)
35 //PA6   Empty
36 //PA7   Empty
37
38 ////////////////////////////////////////////////////////////////////////
39 //DEBUG STATEMENTS
40 ////////////////////////////////////////////////////////////////////////
41 //#define UART_DEBUG 1
42 //#define DEBUG 1
43 #undef DEBUG //remember to slow i2c communication if you do this!
44 //only one of I2C and UART can be defined at the same time
45 #define I2C 1
46 #define UART 1
47 #define BOARD_REV1 1
48 ////////////////////////////////////////////////////////////////////////
49 //UART
50 ////////////////////////////////////////////////////////////////////////
51 #ifdef UART
52 #include "BBUART.h"
53 #include "BBUARTasm.h"
54 //#include "USI_UART_TINY861_nopb3.h"
55 #endif
56
57 #ifdef I2C
58 #include "USI_TWI_Slave_tiny861.h"
59 #endif
60 ////////////////////////////////////////////////////////////////////////
61 //Sampling Constants
62 ////////////////////////////////////////////////////////////////////////
63 #define IOUT_READINGS 256
```

```
64 #define IIN_READINGS 256
65 #define VOUT_READINGS 256
66 #define VIN_READINGS 1
67 ////////////////////////////////////////////////////////////////////////
68 //EEPROM Locations
69 ////////////////////////////////////////////////////////////////////////
70 #define EEPROM_CHIP_ID 5
71 #define EEPROM_CAN_ID 6
72 #define EEPROM_I2CADDRESS 7
73 #define EEPROM_M 8
74 #define EEPROM_REV 9
75 ////////////////////////////////////////////////////////////////////////
76 //Sensing and ADC Settings
77 ////////////////////////////////////////////////////////////////////////
78 #define VOUT_DIVIDER 11 //100k and 10k resistor
79 #define VIN_DIVIDER 11
80 #define VOUT_MUX 0 //VOUT is sampled on PA0, which is ADC0
81 #define VIN_MUX 6 //VIN is sampled on PA7, which is ADC6s
82 #define VL_MUX 0 //Inductor VL is sampled by PA0, which is ADC0
83 #define VH_MUX 1 //Inductor VH is sampled by PA1, which is ADC1
84 #define VREF 2.54
85 #define ADC_CENTER 512
86 #define ADC_MAX 1024
87 #define TIMEOUT_MAX 1000
88 #define ERROR_BYTE 0xFF
89 #define PACKET_SIZE 6
90 #define MAXLOOPCOUNT 6000 //we pause for 1 ms in main loop, so 2000 will
       correspond to roughly 2 seconds
91 #define DUTY_COMMAND 1
92 #define VOLTAGE_COMMAND 2
93 #define CURRENT_COMMAND 3
94 #define ENABLE_CONVERTER 4
95 #define CLEAR_BUFFER_COMMAND 5
96 #define VOLTAGE_IN_COMMAND 6
97 #define BYPASS_ENABLE_COMMAND 7
```

```
98  #define BYPASS_DISABLE_COMMAND 8
99  #define PING_COMMAND 9
100 #define STATUS_BYTE_STARTUP 0
101 #define STATUS_BYTE_MPPT 1
102 #define STATUS_BYTE_BYPASS 2
103 //extern unsigned char status_byte;
104 unsigned char chip_id_read;
105 unsigned char rev_read;
106 unsigned char i2c_address;
107 unsigned char myI2Caddress;
108 unsigned long vout;
109 unsigned long vout_old;
110 unsigned long vin;
111 unsigned long vout_adjusted;
112 unsigned long vin_adjusted;
113 unsigned long iin;
114 unsigned long iout;
115 double iout_old;
116 void MPPTInit(void);
117 void MPPTTrack(void);
118 unsigned int I2CRead(void);
119 void I2CWrite(void);
120
121 #ifdef UART
122 #define SEND_STRING BBUART_Transmit_String //Let us change to other string
        sending function at a later time.
123 #define SEND_DATA BBUART_Transmit_Data
124 #else
125 #define SEND_STRING Empty_Transmit_String //Let us change to other string
        sending function at a later time.
126 #define SEND_DATA Empty_Transmit_Data
127 #endif
128
129 #endif
```

Listing I.3: *solar_code/ADC.c*

```
1  #include "ADC.h"
2
3  volatile unsigned int adcvalue=0;
4  //volatile unsigned int adccomplete=0;
5
6  void ADCInit(void)
7  {
8
9  //PA0 is inductor high side, PA1 is inductor low side.
10 //see page 157 of datasheet for clock pre-scaler.
11 ADCSRA |= (1 << ADPS2) | (1 << ADPS1) | (0 << ADPS0); // Set ADC prescaler to
       8 - 125KHz sample rate @ 1 MHz clock frequency. We need ADC frequency to
       be no higher than 200 kHz for 10 bit precision
12 //1 1 0 gives a 64 division factor from clock frequency, giving 125 kHz on a 8
       MHz clock
13 //ADMUX |= (0 << REFS2) | (1 <<REFS1) | (0 << REFS0); // Set ADC reference to
       1.1V.s
14 ADMUX |= (1 <<REFS1) | (1 << REFS0); // Set ADC reference to 2.54V with
       external bypass cap, see page 154 of datasheet
15 ADCSRB |= (1 << REFS2);
16
17 //ADMUX |= (1 << ADLAR); // Left adjust ADC result to allow easy 8 bit reading
18
19 // No MUX values needed to be changed to use ADC0, look up on page 257 in
       datasheet if other ADC channels should be sampled
20
21 //ADCSRA |= (1 << ADFR);  // Set ADC to Free-Running Mode
22 //ADCSRA |= (1 << ADEN);  // Enable ADC
23 //ADCSRA |= (1 << ADIE);  // Enable ADC Interrupt
24
25
26 }
27
28 void ADCSetMux(char Bitmask)
```

```
29 {
30 //Only applies to single-ended conversion, when the lower four bits set the
       mux input
31 ADMUX &= 0xF0; //this will clear the lower four bits of ADMUX, while keeping
       the top four the same
32 CLEARBIT(ADCSRB,MUX5);
33 CLEARBIT(ADMUX,MUX4);
34 ADMUX |= Bitmask;
35
36 }
37
38 void ADCSetMuxDifferential()
39 {
40 //ADMUX &= 0xF0; //this will clear the lower four bits of ADMUX, while keeping
        the top four the same
41 //setup for PA1 being positive differential input, and PA0 being negative
       differential input. See table on page 156 of datasheet.
42 //MUX5..0 100010
43 //VH is PA1 and VL is PA0, so measure PA1-PA0 differentially
44
45
46 SETBIT(ADCSRB, MUX5);
47 CLEARBIT(ADMUX, MUX4);
48 CLEARBIT(ADMUX, MUX3);
49 CLEARBIT(ADMUX, MUX2);
50 SETBIT(ADMUX, MUX1);
51 CLEARBIT(ADMUX, MUX0);
52
53 /*
54 SETBIT(ADCSRB, MUX5);
55 CLEARBIT(ADMUX, MUX4);
56 CLEARBIT(ADMUX, MUX3);
57 SETBIT(ADMUX, MUX2);
58 CLEARBIT(ADMUX, MUX1);
59 CLEARBIT(ADMUX, MUX0);
```

```
60 */
61
62
63 //setup a gain of 32x
64 SETBIT(ADCSRB, GSEL);
65 //SETBIT(ADCSRB, BIN);
66 }
67
68 unsigned int ADCSample(void)
69 {
70
71 ADCSRA |= (1 << ADSC);  // Start A2D Conversions
72 while (CHECKBIT(ADCSRA, ADSC)) //ADSC will go zero when conversion is complete
       .
73         {}
74 unsigned int temp;
75 temp=ADCL;
76 temp += (ADCH<<8);
77 return temp;
78
79 }
80
81 //Unsigned long is 2^32, which is very large. Maximum reading is 1024=2^10. So
       we can take 2^21 readings without problems
82 //Readings is the number of reading we would like to sum, don't make this
      larger than 2^21=2097152, to ensure that it doesn't overflow
83 unsigned long ADCReadValueDifferential(unsigned int readings)
84 {
85 ADCSetMuxDifferential();
86 unsigned long var_sum=0;
87 unsigned int count=0;
88 SETBIT(ADCSRA, ADEN);
89 while (count<readings)
90         {
91         //ADCSample();
```

```
92          var_sum=var_sum+(long)ADCSample(); //must do a type cast to make sure
                the result is long
93          count++;
94          }
95  CLEARBIT(ADCSRA, ADEN);
96  return var_sum;
97
98  }
99
100 //Readings is the number of reading we would like to sum, don't make this
        larger than 2^21=2097152, to ensure that it doesn't overflow
101 unsigned long ADCReadValue(char muxvalue, unsigned int readings)
102 {
103 ADCSetMux(muxvalue);
104 unsigned long var_sum=0;
105 unsigned int count=0;
106 SETBIT(ADCSRA, ADEN);
107 while (count<readings)
108         {
109         var_sum=var_sum+(long)ADCSample();
110         count++;
111         }
112 CLEARBIT(ADCSRA, ADEN);
113 return var_sum;
114
115 }
```

Listing I.4: *solar_code/ADC.h*

```
1  #ifndef ADC_H
2  #define ADC_H
3
4  #include <avr/io.h>
5  #include <avr/interrupt.h>
6  #include "mppt.h"
7
```

```
8  void ADCInit(void);
9  void ADCSetMux(char Bitmask);
10 void ADCSetMuxDifferential(void);
11 unsigned int ADCSample(void);
12 unsigned long ADCReadValue(char, unsigned int);
13 unsigned long ADCReadValueDifferential(unsigned int);
14
15
16
17 #endif
```

Listing I.5: *solar_code/PWM.c*

```
1  #include "PWM.h"
2  void PWMInit(void)
3  {
4  ////////////////////////////////////////
5  // We want to setup the PLL to get a 64 MHz PWM clock, which will give us 10-
       bit PWM resolution
6  ////////////////////////////////////////
7  //Setup PWMH and PWML as output pins
8  SETBIT(PWMPORTDDR, PWML1);
9  //Startup with gate driver disabled.
10 CLEARBIT(PWMPORT, PWMSHUTDOWN); //remember, active low in this implementation
11 //Enable PLL, page 89 of 861 datasheet
12 CLEARBIT(PLLCSR, LSM); //Do not run in low-speed mode (32MHz)
13 SETBIT(PLLCSR, PLLE); //Enable the PLL
14 //Wait 100 us for PLL to stabilize
15 _delay_us(100);
16 //Poll the PLOCK bit until it is set
17 while (CHECKBIT(PLLCSR, PLOCK)==0) {}; //Hang out until PLOCK is set
18 //Set the PCKE bit in the PLLCSR register which enables the asynchronous mode
19 SETBIT(PLLCSR, PCKE); //Run in asynchronous mode, which is running from PLL
       clock (64 MHz)
20 //Setup pre-scaler. Run at full speed, which is 0001 for the TCCR1B bits. See
       page 90 for table.
```

```
21 //TCCR2B is initialized to 0
22 CLEARBIT(TCCR1B, CS13);
23 CLEARBIT(TCCR1B, CS12);
24 CLEARBIT(TCCR1B, CS11);
25 SETBIT(TCCR1B, CS10);
26 //Setup compare output mode to fast pwm mode. OC1B is PWML1. TCCR1C=Timer/
       Counter Control Register 1. See page 113 in datasheet
27 //Not using top mosfet: Page 115, table 16-12
28 //This combo will clear OC1B on compare match, which makes D being the time
       when top switch is on.
29 CLEARBIT(TCCR1A, COM1B0);
30 SETBIT(TCCR1A, COM1B1);
31 //WGM11..10 is set to 00 as default, which corresponds to fast pwm, so no need
        to change this in TCCR1D
32 CLEARBIT(TCCR1D, WGM11);
33 CLEARBIT(TCCR1D, WGM10);
34 //TOP is stored in OCR1C, and max value is 1024. Must do a 10-bit operation.
35 int top_value=TIMER1_TOP;
36 TC1H=(top_value>>8);
37 OCR1C=(unsigned char)top_value;
38 //OCR1C=255;
39 }
40
41 void PWMEnable(void)
42 {
43 //Enable PWM mode based on comparator OCR1D, new for ATtiny861
44 SETBIT(TCCR1A, PWM1B);
45 SETBIT(PWMPORT, PWMSHUTDOWN); //active low, so setting it will enable pwm
46 }
47
48 void PWMDisable(void)
49 {
50 CLEARBIT(TCCR1A, PWM1B); //Shut off PWM clock
51 CLEARBIT(PWMPORT, PWMSHUTDOWN); //Disable gate drive chip
52 CLEARBIT(PWMPORT, PWML1); //Set PWML1 low
```

```
53 }
54
55 unsigned int PWMReadDuty(void)
56 {
57 //From page 111 on data sheet. TC1H is shared MSB register for top 2 bits
58 double duty_return=0;
59 unsigned duty;
60 duty=DUTY_COUNT;
61 duty |= ((unsigned int)TC1H << 8);
62 unsigned int top_count=OCR1C;
63 top_count |= ((unsigned int)TC1H << 8); //top count now holds the 10-bit value
       that is in OCR1C
64 duty_return = duty*1024.0/top_count;
65 return (unsigned int) duty_return;
66 }
67 //set the PWM value from 0 to 1024 and scale such that when the counter value
      is less than 1024, we keep the ratio the same
68 void PWMWriteDuty(unsigned int duty)
69 {
70
71 //From page 112 on data sheet. TC1H is shared MSB register for top 2 bits
72 unsigned int top_count=OCR1C;
73 top_count |= ((unsigned int)TC1H << 8); //top count now holds the 10-bit value
       that is in OCR1C
74 double new_duty=0;
75 new_duty=duty/1024.0*top_count;
76 //USI_UART_Transmit_Data("new duty:",new_duty);
77 //unsigned int new_duty=31;
78 TC1H=((unsigned int)new_duty>>8);
79 DUTY_COUNT=(unsigned char)new_duty;
80 }
81
82
83 void PWMWriteFrequency(unsigned int freq)
84 {
```

```
85  //Set OCR1C to correspont to frequency, which is given in kHz. OCR1C is a 10-
        bit register, so the minimum frequency is 62.5 kHz
86  unsigned int old_duty=PWMReadDuty();
87  unsigned int top_value=F_PLL/1000.0/freq;
88  TC1H=(top_value>>8);
89  OCR1C=(unsigned char)top_value;
90  //Ensure that duty cycle stays the same after we change the frequency
91  PWMWriteDuty(old_duty);
92  }
93
94  unsigned int PWMReadFrequency(void)
95  {
96  unsigned int freq=OCR1C;
97  freq |= ((unsigned int)TC1H << 8);
98  return freq;
99
100 }
```

Listing I.6: *solar_code/PWM.h*

```
1   #ifndef PWM_H
2   #define PWM_H
3
4   #include <avr/interrupt.h>
5   #include "mppt.h"
6
7   #define PWMPORTDDR DDRB
8   #define PWMPORT PORTB
9   #define PWML1 3
10  //#define PWMH1 4
11  #define PWMSHUTDOWN 6 //active low shutdown pin
12  #define DUTY_COUNT OCR1B
13  #define DUTY_MAX 0.99*TIMER1_TOP
14  #define DUTY_MIN 0.01*TIMER1_TOP
15  #define F_PLL 64000000 //Run at 6.4 MHz for 8 bit precision when switching at
        25 kHZ
```

```
16 #define TIMER1_TOP F_PLL/FS //Maximum is 1024, since we are doing 10-bit duty
       cycle access. For 64MHz PLL, and FS=250k, get 256, which is 8-bit
       precision
17 #define FS 250000 //Converter switching frequency, change here.
18 void PWMInit(void);
19 void PWMEnable(void);
20 void PWMDisable(void);
21 unsigned int PWMReadDuty(void);
22 void PWMWriteDuty(unsigned int);
23 void PWMWriteFrequency(unsigned int);
24 unsigned int PWMReadFrequency(void);
25
26 #endif
```

Listing I.7: *solar_code/BBUART.c*

```
1  #include <avr/io.h>
2  #include <util/delay.h>
3  #include "BBUART.h"
4  #include "BBUARTasm.h"
5
6
7  //extern void putchar(void);
8  extern void putchar(uint8_t byte);
9  void BBUART_Transmit_String(const char StringPtr[])
10 {
11     while (*StringPtr != 0x00)
12     {
13         //BBUART_Transmit_Byte(*StringPtr);
14             putchar(*StringPtr);
15         StringPtr++;
16     }
17 }
18
19 void BBUART_Transmit_Byte(unsigned char data)
20 {
```

```
21 putchar(data);
22 }
23
24 void BBUART_Transmit_Data(const char StringPtr[], unsigned long data)
25 {
26 char data_string[10];
27 ltoa(data,data_string,10);
28 //dtostrf(data,5,4,data_string);
29 BBUART_Transmit_String(StringPtr);
30 BBUART_Transmit_String(data_string);
31 }
```

Listing I.8: *solar_code/BBUART.h*

```
1  #ifndef BBUART_H
2  #define BBUART_H
3
4  #include <avr/io.h>
5  #include <stdlib.h>
6  #include "mppt.h"
7
8  //extern void putchar(uint8_t byte);
9  void BBUART_Transmit_String(const char StringPtr[]);
10 void BBUART_Transmit_Byte(unsigned char data);
11 void BBUART_Transmit_Data(const char[], unsigned long);
12
13 #endif
```

Listing I.9: *solar_code/BBUARTasm.h*

```
1  #ifdef __ASSEMBLER__
2
3  #  define bitcnt        r31
4  #  define temp          r30
5  #  define Txbyte    r24 //even though r25 is the first parameter passed, if it
       's an 8-bit one, it's stored in r24
```

```
6
7  #else  /* !ASSEMBLER */
8
9  #include <stdint.h>
10
11
12 #endif /* ASSEMBLER */
13
14 //BBUART_Transmit_String(const char[]);
15 //BBUART_Transmit_Byte(unsigned char);
```

Listing I.10: *solar_code/USI_TWI_Slave_tiny861.c*

```
1  //Adapted from Atmel AppNote AVR312, with specific changes to accommodate the
       ATtiny861 micro by Robert Pilawa
2  //#include <ioavr.h>
3  //#include <inavr.h>
4  #include "USI_TWI_Slave_tiny861.h"
5
6  /*! Static Variables
7   */
8
9  static unsigned char TWI_slaveAddress;
10 static volatile unsigned char USI_TWI_Overflow_State;
11
12
13 /*! Local variables
14  */
15 static uint8_t TWI_RxBuf[TWI_RX_BUFFER_SIZE];
16 static volatile uint8_t TWI_RxHead;
17 static volatile uint8_t TWI_RxTail;
18
19 static uint8_t TWI_TxBuf[TWI_TX_BUFFER_SIZE];
20 static volatile uint8_t TWI_TxHead;
21 static volatile uint8_t TWI_TxTail;
22
```

```
23 /*! \brief Flushes the TWI buffers
24  */
25 void Flush_TWI_Buffers(void)
26 {
27     TWI_RxTail = 0;
28     TWI_RxHead = 0;
29     TWI_TxTail = 0;
30     TWI_TxHead = 0;
31 }
32
33 //********** USI_TWI functions **********//
34
35 /*! \brief
36  * Initialise USI for TWI Slave mode.
37  */
38 void USI_TWI_Slave_Initialise( unsigned char TWI_ownAddress )
39 {
40   Flush_TWI_Buffers();
41
42   TWI_slaveAddress = TWI_ownAddress;
43
44   PORT_USI |=  (1<<PORT_USI_SCL);                    // Set SCL
         high
45   PORT_USI |=  (1<<PORT_USI_SDA);                    // Set SDA
         high
46   DDR_USI  |=  (1<<PORT_USI_SCL);                    // Set SCL
         as output
47   //DDR_USI  &= ~(1<<PORT_USI_SCL);                    // Set SCL
          as input, pilawa
48   DDR_USI  &= ~(1<<PORT_USI_SDA);                    // Set SDA
         as input
49   USICR    =  (1<<USISIE)|(0<<USIOIE)|               // Enable
         Start Condition Interrupt. Disable Overflow Interrupt.
50             (1<<USIWM1)|(0<<USIWM0)|                 // Set USI
                  in Two-wire mode. No USI Counter overflow prior
```

```
51                                                                  // to first Start Condition ( potentail failure )
52              (1<<USICS1)|(0<<USICS0)|(0<<USICLK)|                // Shift Register Clock Source = External, positive edge
53              (0<<USITC);
54    USISR     = 0xF0;                                             // Clear all flags and reset overflow counter
55  }
56
57
58  /*! \brief Puts data in the transmission buffer, Waits if buffer is full.
59  */
60  void USI_TWI_Transmit_Byte( unsigned char data )
61  {
62      unsigned char tmphead;
63
64      tmphead = ( TWI_TxHead + 1 ) & TWI_TX_BUFFER_MASK;          // Calculate buffer index.
65      while ( tmphead == TWI_TxTail );                            // Wait for free space in buffer.
66      TWI_TxBuf[tmphead] = data;                                  // Store data in buffer.
67      TWI_TxHead = tmphead;                                       // Store new index.
68  }
69
70  /*! \brief Returns a byte from the receive buffer. Waits if buffer is empty.
71   */
72  unsigned char USI_TWI_Receive_Byte( void )
73  {
```

```
74      unsigned char tmptail;
75      unsigned char tmpRxTail;                                    // Temporary
            variable to store volatile
76      tmpRxTail = TWI_RxTail;                                     // Not necessary
            , but prevents warnings
77      while ( TWI_RxHead == tmpRxTail );
78      tmptail = ( TWI_RxTail + 1 ) & TWI_RX_BUFFER_MASK;          // Calculate
            buffer index
79      TWI_RxTail = tmptail;                                       // Store new
            index
80      return TWI_RxBuf[tmptail];                                  // Return data
            from the buffer.
81  }
82
83  /*! \brief Check if there is data in the receive buffer.
84   */
85  unsigned char USI_TWI_Data_In_Receive_Buffer( void )
86  {
87      unsigned char tmpRxTail;                                // Temporary variable
            to store volatile
88      tmpRxTail = TWI_RxTail;                                 // Not necessary, but
            prevents warnings
89      return ( TWI_RxHead != tmpRxTail );                     // Return 0 (FALSE) if
             the receive buffer is empty.
90  }
91
92  /*! \brief Usi start condition ISR
93   * Detects the USI_TWI Start Condition and intialises the USI
94   * for reception of the "TWI Address" packet.
95   */
96
97  //#pragma vector=USI_START_VECTOR
98  //__interrupt void USI_Start_Condition_ISR(void)
99  ISR(USI_START_VECTOR)
100 {
```

```
101     unsigned char tmpUSISR;                                         //
            Temporary variable to store volatile
102     tmpUSISR = USISR;                                               // Not
            necessary, but prevents warnings
103 // Set default starting conditions for new TWI package
104         unsigned timeout_counter=0;
105     USI_TWI_Overflow_State = USI_SLAVE_CHECK_ADDRESS;
106     DDR_USI  &= ~(1<<PORT_USI_SDA);                                 // Set SDA
             as input
107     //while ( (PIN_USI & (1<<PORT_USI_SCL)) & !(tmpUSISR & (1<<USIPF) ) );
            // Wait for SCL to go low to ensure the "Start Condition" has
            completed.
108         //pilawa, if somehow SCL stays high before we are able to detect this,
                we may wait here forever, no?
109          while ( (PIN_USI & (1<<PORT_USI_SCL)) & !(tmpUSISR & (1<<USIPF)) && (
                timeout_counter < 254) )
110                 {
111                 timeout_counter++;   // Wait for SCL to go low to ensure the "
                        Start Condition" has completed. pilawa added
                        timeout_counter to see if it helps with freezing in code.
                        doesn't seem to be it.
112                 }

                    // If a Stop condition arises then leave the interrupt to
                    prevent waiting forever.
113     USICR    =    (1<<USISIE)|(1<<USIOIE)|                          // Enable
            Overflow and Start Condition Interrupt. (Keep StartCondInt to detect
            RESTART)
114                  (1<<USIWM1)|(1<<USIWM0)|                          // Set USI
                         in Two-wire mode.
115                              //(1<<USIWM1)|(0<<USIWM0)|
                                                        // Set USI in Two-
                                 wire mode. pilawa, page 133, don't hold SCL low
                                  when a counter overflow happens. This lead to
                                 only address being received
```

```
116                     (1<<USICS1)|(0<<USICS0)|(0<<USICLK)|                    // Shift
                            Register Clock Source = External, positive edge
117                     (0<<USITC);
118         USISR   =   (1<<USI_START_COND_INT)|(1<<USIOIF)|(1<<USIPF)|(1<<USIDC)|
                    // Clear flags
119                     (0x0<<USICNT0);                                         // Set USI
                            to sample 8 bits i.e. count 16 external pin toggles.
120 }
121
122
123 /*! \brief USI counter overflow ISR
124  * Handels all the comunication. Is disabled only when waiting
125  * for new Start Condition.
126  */
127 //#pragma vector=USI_OVERFLOW_VECTOR
128 //__interrupt void USI_Counter_Overflow_ISR(void)
129 ISR(USI_OVERFLOW_VECTOR)
130 {
131   unsigned char tmpTxTail;      // Temporary variables to store volatiles
132   unsigned char tmpUSIDR;
133
134
135   switch (USI_TWI_Overflow_State)
136   {
137     // ---------- Address mode ----------
138     // Check address and send ACK (and next USI_SLAVE_SEND_DATA) if OK, else
            reset USI.
139     case USI_SLAVE_CHECK_ADDRESS:
140       //if ((USIDR == 0) || (( USIDR>>1 ) == TWI_slaveAddress))
141           if ((( USIDR>>1 ) == TWI_slaveAddress)) //pilawa, removed UISDR == 0
                  check. This may have caused the code to hang.
142       {
143         if ( USIDR & 0x01 )
144           USI_TWI_Overflow_State = USI_SLAVE_SEND_DATA; //pilawa, if direction
                  bit is 1, it's a read from the master.
```

```
145         else
146           USI_TWI_Overflow_State = USI_SLAVE_REQUEST_DATA; //pilawa, master
                requests a write, that's why we acknowledge that we got the
                request.
147           SET_USI_TO_SEND_ACK();
148       }
149       else
150       {
151         SET_USI_TO_TWI_START_CONDITION_MODE();
152       }
153       break;
154
155     // ----- Master write data mode -------
156     // Check reply and goto USI_SLAVE_SEND_DATA if OK, else reset USI.
157     case USI_SLAVE_CHECK_REPLY_FROM_SEND_DATA:
158       if ( USIDR ) // If NACK, the master does not want more data.
159       {
160         SET_USI_TO_TWI_START_CONDITION_MODE();
161         return;
162       }
163       // From here we just drop straight into USI_SLAVE_SEND_DATA if the
            master sent an ACK
164
165     // Copy data from buffer to USIDR and set USI to shift byte. Next
          USI_SLAVE_REQUEST_REPLY_FROM_SEND_DATA
166     case USI_SLAVE_SEND_DATA:
167
168       // Get data from Buffer
169       tmpTxTail = TWI_TxTail;             // Not necessary, but prevents
            warnings
170       if ( TWI_TxHead != tmpTxTail )
171       {
172         TWI_TxTail = ( TWI_TxTail + 1 ) & TWI_TX_BUFFER_MASK;
173         USIDR = TWI_TxBuf[TWI_TxTail];
174       }
```

```
175         else // If the buffer is empty then:
176         {
177             SET_USI_TO_TWI_START_CONDITION_MODE();
178             return;
179         }
180         USI_TWI_Overflow_State = USI_SLAVE_REQUEST_REPLY_FROM_SEND_DATA;
181         SET_USI_TO_SEND_DATA();
182         break;
183
184       // Set USI to sample reply from master. Next
              USI_SLAVE_CHECK_REPLY_FROM_SEND_DATA
185       case USI_SLAVE_REQUEST_REPLY_FROM_SEND_DATA:
186         USI_TWI_Overflow_State = USI_SLAVE_CHECK_REPLY_FROM_SEND_DATA;
187         SET_USI_TO_READ_ACK();
188         break;
189
190       // ----- Master read data mode -------
191       // Set USI to sample data from master. Next
              USI_SLAVE_GET_DATA_AND_SEND_ACK.
192       case USI_SLAVE_REQUEST_DATA:
193         USI_TWI_Overflow_State = USI_SLAVE_GET_DATA_AND_SEND_ACK;
194         SET_USI_TO_READ_DATA();
195         break;
196
197       // Copy data from USIDR and send ACK. Next USI_SLAVE_REQUEST_DATA
198       case USI_SLAVE_GET_DATA_AND_SEND_ACK:
199         // Put data into Buffer
200         tmpUSIDR = USIDR;               // Not necessary, but prevents warnings
201         TWI_RxHead = ( TWI_RxHead + 1 ) & TWI_RX_BUFFER_MASK;
202         TWI_RxBuf[TWI_RxHead] = tmpUSIDR;
203
204         USI_TWI_Overflow_State = USI_SLAVE_REQUEST_DATA;
205         SET_USI_TO_SEND_ACK();
206         break;
207   }
```

```
208 }
```

Listing I.11: *solar_code/USI_TWI_Slave_tiny861.h*

```
1  //Adapted from Atmel AppNote AVR312, with specific changes to accommodate the
       ATtiny861 micro by Robert Pilawa
2  #include <avr/io.h>
3  #include <avr/interrupt.h>
4  #include "mppt.h"
5
6
7  void Flush_TWI_Buffers(void);
8  void           USI_TWI_Slave_Initialise( unsigned char );
9  void           USI_TWI_Transmit_Byte( unsigned char );
10 unsigned char USI_TWI_Receive_Byte( void );
11 unsigned char USI_TWI_Data_In_Receive_Buffer( void );
12 void           Timer_Init(void);
13
14 #define TRUE                1
15 #define FALSE               0
16
17 typedef     unsigned char       uint8_t;
18
19 ////////////////////////////////////////////////////////////////
20 ///////////////// Driver Buffer Definitions ////////////////////
21 ////////////////////////////////////////////////////////////////
22 // 1,2,4,8,16,32,64,128 or 256 bytes are allowed buffer sizes
23
24 #define TWI_RX_BUFFER_SIZE  (16)
25 #define TWI_RX_BUFFER_MASK ( TWI_RX_BUFFER_SIZE - 1 )
26
27 #if ( TWI_RX_BUFFER_SIZE & TWI_RX_BUFFER_MASK )
28         #error TWI RX buffer size is not a power of 2
29 #endif
30
31 // 1,2,4,8,16,32,64,128 or 256 bytes are allowed buffer sizes
```

```
32
33 #define TWI_TX_BUFFER_SIZE  (16)
34 #define TWI_TX_BUFFER_MASK ( TWI_TX_BUFFER_SIZE - 1 )
35
36 #if ( TWI_TX_BUFFER_SIZE & TWI_TX_BUFFER_MASK )
37         #error TWI TX buffer size is not a power of 2
38 #endif
39
40
41
42 #define USI_SLAVE_CHECK_ADDRESS                (0x00)
43 #define USI_SLAVE_SEND_DATA                    (0x01)
44 #define USI_SLAVE_REQUEST_REPLY_FROM_SEND_DATA (0x02)
45 #define USI_SLAVE_CHECK_REPLY_FROM_SEND_DATA   (0x03)
46 #define USI_SLAVE_REQUEST_DATA                 (0x04)
47 #define USI_SLAVE_GET_DATA_AND_SEND_ACK        (0x05)
48
49
50 //! Device dependent defines
51 //added by pilawa
52
53 //#if defined(__ATtiny261__) | defined(__ATtiny461__) | defined(__ATtiny861__)
54     #define DDR_USI             DDRB
55     #define PORT_USI            PORTB
56     #define PIN_USI             PINB
57     #define PORT_USI_SDA        PORTB0
58     #define PORT_USI_SCL        PORTB2
59     #define PIN_USI_SDA         PINB0
60     #define PIN_USI_SCL         PINB2
61     #define USI_START_COND_INT  USISIF
62     #define USI_START_VECTOR    USI_START_vect
63     #define USI_OVERFLOW_VECTOR USI_OVF_vect
64 //#endif
65
66
```

```
67
68 //! Functions implemented as macros
69 #define SET_USI_TO_SEND_ACK()
       \
70 {
       \
71         USIDR    =  0;                                          /* Prepare
               ACK                          */ \
72         DDR_USI  |=  (1<<PORT_USI_SDA);                         /* Set SDA
               as output                    */ \
73         USISR    =  (0<<USI_START_COND_INT)|(1<<USIOIF)|(1<<USIPF)|(1<<USIDC)|
              /* Clear all flags, except Start Cond  */ \
74                     (0x0E<<USICNT0);                           /* set USI
                         counter to shift 1 bit. */ \
75 }
76
77 #define SET_USI_TO_READ_ACK()
       \
78 {
       \
79         DDR_USI &=  ~(1<<PORT_USI_SDA);                         /* Set SDA
               as intput */                 \
80         USIDR    =  0;                                          /* Prepare
               ACK       */                 \
81         USISR    =  (0<<USI_START_COND_INT)|(1<<USIOIF)|(1<<USIPF)|(1<<USIDC)|
              /* Clear all flags, except Start Cond  */ \
82                     (0x0E<<USICNT0);                           /* set USI
                         counter to shift 1 bit. */ \
83 }
84
```

```
85  #define SET_USI_TO_TWI_START_CONDITION_MODE()
        \
86  {
        \
87    USICR    =  (1<<USISIE)|(0<<USIOIE)|                    /* Enable Start
          Condition Interrupt. Disable Overflow Interrupt.*/  \
88                (1<<USIWM1)|(0<<USIWM0)|                    /* Set USI in
                    Two-wire mode. No USI Counter overflow hold.      */  \
89                (1<<USICS1)|(0<<USICS0)|(0<<USICLK)|        /* Shift
                    Register Clock Source = External, positive edge       */  \
90                (0<<USITC);
                    \
91    USISR    =  (0<<USI_START_COND_INT)|(1<<USIOIF)|(1<<USIPF)|(1<<USIDC)|  /*
          Clear all flags, except Start Cond                          */ \
92                (0x0<<USICNT0);
                    \
93  }
94
95  #define SET_USI_TO_SEND_DATA()
        \
96  {
        \
97      DDR_USI |=  (1<<PORT_USI_SDA);                          /* Set SDA
            as output                    */ \
98      USISR    =  (0<<USI_START_COND_INT)|(1<<USIOIF)|(1<<USIPF)|(1<<USIDC)|
                /* Clear all flags, except Start Cond */ \
99                  (0x0<<USICNT0);                           /* set USI
                        to shift out 8 bits         */ \
100 }
```

```
101
102 #define SET_USI_TO_READ_DATA()
        \
103 {
        \
104     DDR_USI &= ~(1<<PORT_USI_SDA);                    /* Set SDA
          as input                        */ \
105     USISR    =  (0<<USI_START_COND_INT)|(1<<USIOIF)|(1<<USIPF)|(1<<USIDC)|
            /* Clear all flags, except Start Cond */ \
106                 (0x0<<USICNT0);                       /* set USI
                    to shift out 8 bits          */ \
107 }
```

Appendix J

Python Control Code for Distributed MPPT

Listing J.1: *solar_code/mppt_automatic_shading_patterns.py*

```
1  #!/bin/env python
2  #=============================================================================
3  # This file automates the process of capturing various shading patterns
4  # It first runs three separate sweeps across each diode, instructing the user
5  # between each time to reconfigure the wires.
6  # it then begins the following sweep for desired degrees of shading (0,25, 50,
       75, 100%)
7  # repeat for each shading pattern:
8  #    it runs an IV sweep across the entire panel, with MPPT bypassed
9  #    This is followed by a sweep of load current with MPPTs running.
10 #=============================================================================
11
12 #=============================================================================
13 # IMPORTS
14 #=============================================================================
15 import sys
16 import time
17 import datetime
18 #from aardvark32.aardvark_py import *
19 from aardvark64.aardvark_py import *
20 from array import array
21 from pilawa_instruments import *
22 import os
```

```
23 import glob
24 #=========================================================================
25 # CONSTANTS
26 #=========================================================================
27 #BUFFER_SIZE = 2048
28 I2C_BITRATE =  400
29 PORT  = 0
30 ADDR1=21
31 ADDR2=22
32 ADDR3=23
33 ADDR4=24
34 PACKET_LENGTH=6
35 CLEAR_BUFFER_MESSAGE=99
36 MAX_RESENDS=3 #number of times to resend an i2c message if the count read is
       not correct
37 #READ_DELAY=0.1 #minimum seems to be 0.05 to prevent hangups
38 READ_DELAY=0.1 #minimum seems to be 0.05 to prevent hangups
39 VOUT_DIVIDER=(100+10)/10.0
40 VIN_DIVIDER=(100+10)/10.0
41 VREF=2.54 #microcontroller vref value
42 ADC_MAX=1024
43 #debug=True
44 DUTY_COMMAND=1
45 VOLTAGE_COMMAND=2
46 CURRENT_COMMAND=3
47 ENABLE_COMMAND=4
48 CLEAR_BUFFER_COMMAND = 5
49 VOLTAGE_IN_COMMAND = 6
50 BYPASS_ENABLE_COMMAND = 7
51 BYPASS_DISABLE_COMMAND = 8
52 PING_COMMAND = 9
53 command_string=("NO_COMMAND","DUTY_COMMAND","VOLTAGE_COMMAND","CURRENT_COMMAND
       ","ENABLE_COMMAND","CLEAR_BUFFER_COMMAND","VOLTAGE_IN_COMMAND","
       BYPASS_ENABLE_COMMAND","BYPASS_DISABLE_COMMAND", "PING_COMMAND")
```

```
54 MPPT_SWEEP_STEP = 50 #how many steps (out of 1000) do we take when we perform
       MPPT sweep
55 MPPT_STEP_SIZE = 6 #how big are our MPPT steps (out of 1000)
56 MPPT_MAX_ITERATIONS=20.0 #how many times do we call the MPPT algorithm at each
       curren level.
57 MPPT_MAX_PEAK_OFFSET = 200.0 #not in use
58 MPPT_PEAK_OFFSET_COEFFICIENT = MPPT_MAX_PEAK_OFFSET/12000.0 #not in use
59 NUMREADS = 100 # number of reads for the ADC on each sampling interval
60 DUTY_MIN = 100 #minimum duty cycle
61 DUTY_MAX = 990 #maximum duty cycle
62
63
64 #switch operation constants
65 STARTVOLTAGE=45.0 #voltage to start the IV sweep at.
66 STOPVOLTAGE=0.0 #MUST HAVE THE .0 AT THE END!
67 NUMSTEPS=200.0 #how many steps do we take, higher number will give better
       resolution
68 FINISHTIME=2000 #i.e 20:00 hours, 8 pm for Americans.
69 NUMSTEPS_CURRENT=10.0 #how many current values
70 STARTCURRENT=8 #what current do we start running at
71 STOPCURRENT=1 #what current do we stop at
72 # on Saleae logic, set for 7-bit address display only.
73
74 #==========================================================
75 # CLASSES
76 #==========================================================
77
78 class converter:
79     def __init__(self, handle, addr=1, duty=500, numreads=10,debug=False):
80         self.handle=handle
81         self.addr=addr
82         self.duty=duty
83         self.numreads=numreads
84         self.debug=debug
85         self.direction=1
```

```
86          self.vout=0
87          self.vout_old=0
88          self.pout_old=0
89          self.iout=0
90          self.vin=0
91          self.peaked=False
92
93          #self.initialize()
94
95      def sendMessage(self, command, parameter):
96          command_byte=command
97          transmit_byte_1=parameter>>8 #higher order bits
98          transmit_byte_2=parameter & 0xFF
99          send_array=array('B',[command_byte, transmit_byte_1,transmit_byte_2,
                command_byte, transmit_byte_1, transmit_byte_2])
100
101         if self.debug: print "sending command: %s send_array: %s to addr: %s"
                % (command_string[command_byte], send_array, self.addr)
102         count = aa_i2c_write(self.handle, self.addr, AA_I2C_NO_FLAGS,
                send_array)
103         if ( count != (len(send_array)) ):
104             print "error sending, addr: %s, command %s, receive count: %d" % (
                    self.addr,command_string[command_byte],count)
105             return (0,0)
106         if self.debug: print "send count: %d" % count
107         time.sleep(READ_DELAY)
108         if (command==CLEAR_BUFFER_COMMAND):
109             return (1,1)
110         (count, data_in) = aa_i2c_read(self.handle, self.addr, AA_I2C_NO_FLAGS
                ,PACKET_LENGTH)
111         if self.debug: print "data_in: %s" % data_in
112         if ( count != (PACKET_LENGTH) ):
113             print "error receiving, addr: %s, command %s, receive count: %d" %
                    (self.addr,command_string[command_byte],count)
114             return (0,0)
```

```
115         if ((data_in[0] == data_in[3]) and (data_in[1] == data_in[4]) and (
                data_in[2] == data_in[5])):
116             value=data_in[1]*256+data_in[2]
117             return (1,value)
118         return (0,0)
119
120     def writeDuty(self, duty):
121         (return_value, readback_duty) = self.sendMessage(DUTY_COMMAND, duty)
122         if (return_value == 1) and (readback_duty == duty):
123             self.duty=duty # update internal duty if message was successful
124             return 1 #to indicate that the command was executed properly
125         else:
126             print "writeDuty_error"
127             return 0
128
129     def readVoltage(self,numreads):
130         (return_value, voltage_read) = self.sendMessage(VOLTAGE_COMMAND,
                numreads)
131         if return_value!=1:
132             print "readVoltage_error"
133         self.vout=(voltage_read/numreads)*VREF/ADC_MAX*VOUT_DIVIDER
134         return self.vout
135
136     def readInputVoltage(self,numreads):
137         (return_value, voltage_read) = self.sendMessage(VOLTAGE_IN_COMMAND,
                numreads)
138         if return_value!=1:
139             print "readInputVoltage_error"
140         self.vin=(voltage_read/numreads)*VREF/ADC_MAX*VIN_DIVIDER
141         return self.vin
142
143
144     def readCurrent(self, numreads):
145         (return_value, current) = self.sendMessage(CURRENT_COMMAND, numreads)
146         if return_value!=1:
```

```
147             print "readCurrent error"
148         self.iout=current
149         return current
150 
151     def clearBuffer(self):
152         self.sendMessage(CLEAR_BUFFER_COMMAND, 0)
153 
154     def ping(self):
155         return self.sendMessage(PING_COMMAND, 0)
156 
157     def bypassEnable(self):
158         #self.enable()
159         #self.writeDuty(1000)
160         #bring a pin down
161         #set the slave select pin low, this will turn on the pmos attached to
                the 5V bus on separate usb cable, and power isolated 5V supply to
                gate of bypass mosfet
162         #aa_gpio_set(handle, 0)
163         self.sendMessage(BYPASS_ENABLE_COMMAND, 0)
164         aa_gpio_set(handle, 0)
165         time.sleep(READ_DELAY)
166 
167     def bypassDisable(self):
168         self.sendMessage(BYPASS_ENABLE_COMMAND, 0)
169         aa_gpio_set(handle, AA_GPIO_SS)
170         time.sleep(READ_DELAY)
171 
172     def enable(self):
173         (return_value, readback_enable) = self.sendMessage(ENABLE_COMMAND, 0)
174         if (return_value == 1):
175             return 1 #to indicate that the command was executed properly
176         else:
177             print "enable error"
178             return 0
179 
```

```
180
181     def MPPTrack(self, f, current):
182         self.readVoltage(self.numreads)
183         self.readCurrent(self.numreads)
184         self.readInputVoltage(self.numreads)
185         pout=self.vout*1.0
186         #header_string_mppt=("addr", "time", "vout", "vin", "duty", "direction
                \n")
187         timestamp = datetime.datetime.now()
188         f.write("%s%s%s%s%s%s%s%s%s%s%s%s%s%s\n" % (self.addr, delimiter,
                timestamp.strftime("%H%M%S%f"), delimiter,      self.vout,
                delimiter, self.vin, delimiter, self.duty, delimiter, self.
                direction, delimiter, current))
189
190         #pout=self.vout*self.iout
191         if (pout <= self.pout_old):
192             self.direction=-1*self.direction #change direction if we're going
                    the wrong way
193             print "addr: %s changed direction" % self.addr
194         self.pout_old = pout #update pout_old with new value
195         newduty=self.duty+self.direction*MPPT_STEP_SIZE
196         if (newduty > DUTY_MIN) and (newduty < DUTY_MAX):
197             self.writeDuty(newduty)
198
199     def MPPTSweep(self, f, current):
200         numreads=100
201     #     if converter.debug:
202         print "MPPTSweep entered"
203         self.writeDuty(DUTY_MIN)
204         self.enable()
205         time.sleep(0.5)
206         vout=self.readVoltage(self.numreads)
207     #     iout=self.readCurrent(numreads)
208         iout=1
209         pout_peak=vout*iout
```

```
210         duty_peak=self.duty
211         while (self.duty < DUTY_MAX):
212             newduty=self.duty + MPPT_SWEEP_STEP
213             self.writeDuty(newduty)
214     #         if self.debug:
215
216             time.sleep(.0001)
217     #         time.sleep(1)
218             vout=self.readVoltage(self.numreads)
219             vin=self.readInputVoltage(self.numreads)
220             print "Converter %s, writing duty: %s, vout: %s" % (self.addr,
                    newduty, vout)
221             timestamp = datetime.datetime.now()
222             f.write("%s%s%s%s%s%s%s%s%s%s%s%s%s%s\n" % (self.addr, delimiter,
                    timestamp.strftime("%H%M%S%f"), delimiter, self.vout,
                    delimiter, self.vin, delimiter, self.duty, delimiter,      self
                    .direction, delimiter, current))
223             #iout=self.readCurrent(numreads)
224             iout=1
225             pout=vout*iout
226             if (pout >= pout_peak):
227                 duty_peak=self.duty
228                 pout_peak=pout
229             else:
230                 self.writeDuty(duty_peak)
231                 return
232     #converter.writeDuty(duty_peak)
233
234
235 #=============================================================================
236 # FUNCTIONS
237 #=============================================================================
238
239 def MPPTrack(converter, f, current):
240     vout = converter.readVoltage(converter.numreads)
```

```
241     #converter.readCurrent(converter.numreads)
242     adc_current = converter.readCurrent(converter.numreads)
243     vin = converter.readInputVoltage(converter.numreads)
244     pout = vout*1.0
245     #header_string_mppt=("addr", "time", "vout", "vin", "duty", "direction\n")
246     timestamp = datetime.datetime.now()
247     f.write("%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s\n" % (converter.addr, delimiter,
            timestamp.strftime("%H%M%S%f"), delimiter, vout, delimiter, vin,
            delimiter, converter.duty, delimiter, converter.direction, delimiter,
            current, delimiter, adc_current))
248     if (converter.addr == 21):
249         print "vin: %s" % vin
250     if (pout <= converter.pout_old):
251         converter.direction=-1*converter.direction #change direction if we're
                going the wrong way
252         print "addr: %s changed direction" % converter.addr
253     converter.pout_old = pout #update pout_old with new value
254     #print "pout_old: %s " % converter.pout_old
255     newduty=converter.duty+converter.direction*MPPT_STEP_SIZE
256     if (newduty > DUTY_MIN) and (newduty < DUTY_MAX):
257         converter.writeDuty(newduty)
258
259 def MPPTrackPeak(converter, f, current):
260     vout = converter.readVoltage(converter.numreads)
261     adc_current = converter.readCurrent(converter.numreads)
262     vin = converter.readInputVoltage(converter.numreads)
263     pout = vout*1.0
264     #header_string_mppt=("addr", "time", "vout", "vin", "duty", "direction\n")
265     timestamp = datetime.datetime.now()
266     f.write("%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s\n" % (converter.addr, delimiter,
            timestamp.strftime("%H%M%S%f"), delimiter, vout, delimiter, vin,
            delimiter, converter.duty, delimiter, converter.direction, delimiter,
            current, delimiter, adc_current))
267     if (converter.addr == 21):
268         print "vin: %s" % vin
```

```
269     if (pout <= converter.pout_old):
270         converter.direction=-1*converter.direction #change direction if we're
                going the wrong way
271         print "addr: %s changed direction" % converter.addr
272     converter.pout_old = pout #update pout_old with new value
273     #print "pout_old: %s " % converter.pout_old
274     newduty=converter.duty+converter.direction*MPPT_STEP_SIZE
275     if newduty > DUTY_MAX:
276         converter.writeDuty(newduty - int(MPPT_PEAK_OFFSET_COEFFICIENT*
                adc_current))
277         converter.direction=-1*converter.direction
278     elif (newduty > DUTY_MIN):
279         converter.writeDuty(newduty)
280
281 def MPPTSweep(converter, f, current):
282     numreads=100
283 #     if converter.debug:
284     print "MPPTSweep entered"
285     converter.writeDuty(DUTY_MIN)
286     converter.enable()
287     time.sleep(0.5)
288     vout=converter.readVoltage(converter.numreads)
289 #     iout=converter.readCurrent(numreads)
290     iout=1
291     pout_peak=vout*iout
292     duty_peak=converter.duty
293     while (converter.duty < DUTY_MAX):
294         newduty=converter.duty + MPPT_SWEEP_STEP
295         converter.writeDuty(newduty)
296 #         if converter.debug:
297
298         time.sleep(.0001)
299 #         time.sleep(1)
300         vout=converter.readVoltage(converter.numreads)
301         vin=converter.readInputVoltage(converter.numreads)
```

```
302             print "Converter %s, writing duty: %s, vout: %s" % (converter.addr,
                    newduty, vout)
303             timestamp = datetime.datetime.now()
304             f.write("%s%s%s%s%s%s%s%s%s%s%s%s%s%s\n" % (converter.addr, delimiter,
                    timestamp.strftime("%H%M%S%f"), delimiter, converter.vout,
                    delimiter, converter.vin, delimiter, converter.duty, delimiter,
                    converter.direction, delimiter, current))
305             #iout=converter.readCurrent(numreads)
306             iout=1
307             pout=vout*iout
308             if (pout >= pout_peak):
309                 duty_peak=converter.duty
310                 pout_peak=pout
311             else:
312                 converter.writeDuty(duty_peak)
313                 return
314         #converter.writeDuty(duty_peak)
315
316 def MPPTSweepList(converter_list, f, current):
317 #     if converter.debug:
318         print "MPPTSweepList entered"
319
320         for converter in converter_list:
321             converter.writeDuty(DUTY_MIN)
322             time.sleep(0.5)
323             vout=converter.readVoltage(converter.numreads)
324         #     iout=converter.readCurrent(numreads)
325             adc_current = converter.readCurrent(converter.numreads)
326             iout=1
327             pout_peak=vout*iout
328             duty_peak=converter.duty
329             while (converter.duty < DUTY_MAX):
330                 newduty=converter.duty + MPPT_SWEEP_STEP
331                 converter.writeDuty(newduty)
332         #         if converter.debug:
```

```
333                 time.sleep(.0001)
334         #         time.sleep(1)
335                 vout=converter.readVoltage(converter.numreads)
336                 vin=converter.readInputVoltage(converter.numreads)
337                 print "Converter %s, writing duty: %s, vout: %s" % (converter.addr
                        , newduty, vout)
338                 timestamp = datetime.datetime.now()
339                 for record_converter in converter_list:
340                     timestamp = datetime.datetime.now()
341                     record_converter.readVoltage(record_converter.numreads)
342                     converter.readInputVoltage(record_converter.numreads)
343                     f.write("%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s\n" % (record_converter
                            .addr, delimiter, timestamp.strftime("%H%M%S%f"),
                            delimiter, record_converter.vout, delimiter,
                            record_converter.vin, delimiter, record_converter.duty,
                            delimiter, record_converter.direction, delimiter, current,
                             delimiter, adc_current))
344                 #iout=converter.readCurrent(numreads)
345                 iout=1
346                 pout=vout*iout
347                 if (pout >= pout_peak):
348                     duty_peak=converter.duty
349                     pout_peak=pout
350                 else:
351                     converter.writeDuty(duty_peak)
352                     break
353         #converter.writeDuty(duty_peak)
354 
355 
356 def MPPTPing(converter_list):
357         converter_alive=0
358         while converter_alive < len(converter_list):
359             converter_alive=0
360             for x in converter_list:
361                 #x.clearBuffer()
```

```
362             time.sleep(0.2)
363             print "Pinging converter with addr: %s converter_alive: %s len (
                    converter_list): %s" % (x.addr, converter_alive, len(
                    converter_list))
364             time.sleep(0.2)
365             #x.ping()
366             (return_code, return_value) = x.ping()
367             if return_code == 1:
368                 converter_alive = converter_alive + 1
369             print "return_code: %s" % return_code
370
371 def IVSweep(startvoltage, stopvoltage, file_handle):
372     f = file_handle
373     print "doing regular sweep"
374     sweep_voltage=startvoltage
375     step_size=(stopvoltage-startvoltage)/NUMSTEPS #will be negative if we stop
            lower than we start
376     eload.setMode("VOLT")
377     eload.setSlew(2000000)
378     eload.setValue(sweep_voltage)
379     MPPTPing(converter_list)
380     time.sleep(0.5)
381     for x in converter_list:
382         x.bypassEnable()
383     while (sweep_voltage > stopvoltage): #change here is wanting to go
            different direction
384         eload.setValue(sweep_voltage)
385         eload_current=eload.readCurrent()
386         eload_voltage=eload.readVoltage()
387         timestamp = datetime.datetime.now()
388         f.write("%s%s%s%s%s\n" % ( timestamp.strftime("%H%M%S%f"), delimiter,
                eload_current, delimiter, eload_voltage))
389         sweep_voltage=sweep_voltage+step_size  #step down in voltage to keep
                mppt electronics up (for bypass purposes)
390     print ("bypass sweep done")
```

```
391
392 def MPPTCurrentSweep(filehandle_mppt, filehandle_track):
393     f_mppt = filehandle_mppt
394     f_track = filehandle_track
395     #bypass disabled
396     aa_gpio_set(handle, AA_GPIO_SS)
397     #let things settle for a bit
398     time.sleep(0.1)
399     sweep_current=STARTCURRENT
400     current_step_size=(STOPCURRENT-STARTCURRENT)/NUMSTEPS_CURRENT
401     print "current_step_size: %s" % current_step_size
402     eload.setMode("CURR")
403     eload.setValue(sweep_current)
404     eload.setSlew(2000000)
405     print "About to start MPPT sweep"
406     time.sleep(0.5)
407     MPPTPing(converter_list)
408     for converter in converter_list:
409         converter.writeDuty(DUTY_MIN)
410         converter.enable()
411         time.sleep(0.2)
412     time.sleep(0.5)
413     MPPTSweepList(converter_list, f_mppt, sweep_current)
414     MPPTSweepList(converter_list, f_mppt, sweep_current)
415     while (sweep_current > STOPCURRENT): #only need to change here if sweeping
            from high to low, stepsize is negative if going from high to low
416 #         while (sweep_current < STOPCURRENT):
417         print "setting current value to: %s" % sweep_current
418         eload.setValue(sweep_current)
419         mppt_iterator=0
420         print "starting tracking"
421         max_eload_current = 0
422         max_eload_voltage = 0
423         max_eload_power = 0
424         eload_currents=[]
```

```
425         eload_voltages=[]
426         while (mppt_iterator < MPPT_MAX_ITERATIONS):
427             for x in converter_list:
428                 #x.MPPTrack(f_mppt, sweep_current)
429                 #MPPTrack(x, f_mppt, sweep_current)
430                 MPPTrackPeak(x, f_mppt, sweep_current)
431             mppt_iterator=mppt_iterator+1
432             eload_current = float(eload.readCurrent())
433             eload_voltage = float(eload.readVoltage())
434             eload_power = float(eload_current) * float(eload_voltage)
435             eload_currents.append(eload_current)
436             eload_voltages.append(eload_voltage)
437             if (eload_power > max_eload_power):
438                 max_eload_power = eload_power
439                 max_eload_current = eload_current
440                 max_eload_voltage = eload_voltage
441         sweep_current=sweep_current + current_step_size
442         timestamp = datetime.datetime.now()
443         average_eload_current=sum(eload_currents)/len(eload_currents)
444         average_eload_voltage=sum(eload_voltages)/len(eload_voltages)
445         f_track.write("%s%s%s%s%s\n" % ( timestamp.strftime("%H%M%S%f"),
                delimiter, average_eload_current, delimiter, average_eload_voltage
                ))
446     print "done tracking"
447
448 #=============================================================
449 # MAIN PROGRAM
450 #=============================================================
451 if (len(sys.argv) < 2):
452     print "usage: mppt_single filename"
453     print "  'filename' is the root filename of where data will be saved"
454
455 #     sys.exit()
456
457 filename_root = sys.argv[1]
```

```
458
459 handle = aa_open(PORT)
460 if (handle <= 0):
461     print "Unable to open Aardvark device on port %d" % PORT
462     print "Error code = %d" % handle
463     sys.exit()
464
465 # Ensure that the I2C subsystem is enabled, also do I2C
466 #aa_configure(handle,  AA_CONFIG_SPI_I2C)
467 aa_configure(handle,  AA_CONFIG_GPIO_I2C)
468
469
470 #this will enable slave_select GPIO as output pin. See page 57 of aardvark
        datasheet
471 aa_gpio_direction(handle, AA_GPIO_SS)
472
473 aa_gpio_pullup(handle, AA_GPIO_SS)
474
475 #set slave select high, this will turn off bypass power (set the pmos gate
        high)
476 aa_gpio_set(handle, AA_GPIO_SS)
477
478
479 # Enable the I2C bus pullup resistors (2.2k resistors).
480 # This command is only effective on v2.0 hardware or greater.
481 # The pullup resistors on the v1.02 hardware are enabled by default.
482 aa_i2c_pullup(handle, AA_I2C_PULLUP_BOTH)
483
484 # Enable the Aardvark adapter's power supply.
485 # This command is only effective on v2.0 hardware or greater.
486 # The power pins on the v1.02 hardware are not enabled by default.
487 aa_target_power(handle, AA_TARGET_POWER_BOTH)
488
489 # Set the bitrate
490 bitrate = aa_i2c_bitrate(handle, I2C_BITRATE)
```

```
491 print "Bitrate set to %d kHz" % bitrate
492
493 delimiter=', '
494 t=time.strftime('%Y%m%d')
495 current_directory=os.curdir
496 print "Current directory: %s" % current_directory
497 foldername=current_directory + "/data/" + t
498 print foldername
499 if not os.path.exists(foldername):
500     os.makedirs(foldername)
501 #f=open(filename,"w")
502 #header_string=("addr", "vout", "iout", "vin", "pout", "pout_old", "direction
      ", "duty", "\n")
503 #header_string=("%addr", "time", "duty", "vin", "vout", "iout\n")
504 #f.write(delimiter.join(header_string))
505
506
507 #pseudo-code:
508 #go through each device in a list, read its duty cycle, vin, vout, il
509
510 addrList = [ADDR1,ADDR2,ADDR3,ADDR4]
511 #addrList=[ADDR1]
512 #debug=True
513 debug=False
514 numreads=10
515 converter1=converter(handle, ADDR1, 500, NUMREADS, debug)
516 converter2=converter(handle, ADDR2, 500, NUMREADS, debug)
517 converter3=converter(handle, ADDR3, 500, NUMREADS, debug)
518 converter4=converter(handle, ADDR4, 500, NUMREADS, debug)
519
520 converter_list=[converter1, converter2, converter3]
521 #converter_list=[converter3]
522
523 gpib = prologix_serial(port="/dev/ttyUSB0", baud=115200, debug=False, timeout
      =5)
```

```
524 eload= prologix_6060b(prologix=gpib, addr=5, mode="VOLT", rang="20", debug=
        False)
525 #eload.setMode("CURR")
526 #ping each converter and continue only if all of them are responding.
527 MPPTPing(converter_list)
528 #header strings for the various files
529 header_string=("time", "iload", "vload\n")
530 header_string_mppt=("addr", "time", "vout", "vin", "duty", "direction", "
        current, adc_current\n")
531
532 filecounter=0
533 time_counter=0
534 #pseudo-code
535 # It first runs three separate sweeps across each diode, instructing the user
536 # between each time to reconfigure the wires.
537 # it then begins the following sweep for desired degrees of shading (0, 25,
        50, 75, 100%)
538 # repeat for each shading pattern:
539 #   it runs an IV sweep across the entire panel, with MPPT bypassed
540 #   This is followed by a sweep of load current with MPPTs running.
541 shading_pattern = ('0', '25', '50', '75', '100')
542 yes = set(['yes','y', 'ye'])
543 for shading_percentage in shading_pattern:
544     #check to see if we want to do individual diode sweep
545     print "Perform individual diode sweep for %s percent shade? [y/n]" %
            shading_percentage
546     response = raw_input()
547     if response in yes:
548         for x in converter_list:
549             x.bypassDisable()
550         for diode in range(1,4):
551             print "Performing individual diode IV sweeps"
552             print "Connect electronic load across diode %s, then press Enter"
                    %diode
553             raw_input()
```

```
554                 filename= "%s_shading%s_diode%s.dat" % (filename_root,
                        shading_percentage, diode) #this file will save the measured
                        power and time
555                 f=open(foldername + "/" + filename,"w")
556                 #f=open(filename,"w")
557                 f.write(delimiter.join(header_string))
558                 IVSweep(STARTVOLTAGE/3, STOPVOLTAGE, f)
559                 f.close()
560         #perform regular IV sweep
561         print "Connect MPPTs, and connect electronic load to MPPT outputs, then
                press Enter"
562         raw_input()
563         print "Performing electronic load IV sweep (bypassed)"
564         filename= "%s_shading%s_bypass1.dat" % (filename_root, shading_percentage)
                #this file will save the measured power and time
565         f=open(foldername + "/" + filename,"w")
566         #f=open(filename,"w")
567         f.write(delimiter.join(header_string))
568         IVSweep(STARTVOLTAGE, STOPVOLTAGE, f)
569         f.close()
570         print "Performing tracking current sweep"
571         #MPPT section
572         filename_mppt= "%s_shading%s_mppt.dat" % (filename_root,
                shading_percentage) #this file is for storing the operation of the
                MPPTs for debugging and checking their operation
573         filename_track= "%s_shading%s_track.dat" % (filename_root,
                shading_percentage)#this file will save the measured power and time
574 #         f_mppt=open(filename_mppt,"w")
575         f_mppt=open(foldername + "/" + filename_mppt,"w")
576         timestamp = datetime.datetime.now()
577         f_mppt.write("time: %s MPPT_STEP_SIZE: %s MPPT_MAX_ITERATIONS: %s NUMREADS
                : %s DUTY_MIN: %s DUTY_MAX: %s\n" %(timestamp, MPPT_STEP_SIZE,
                MPPT_MAX_ITERATIONS,NUMREADS,DUTY_MIN, DUTY_MAX))
578         f_mppt.write(delimiter.join(header_string_mppt))
579         f_track=open(foldername + "/" + filename_track,"w")
```

```
580 #     f_track=open(filename_track,"w")
581       f_track.write(delimiter.join(header_string))
582       MPPTCurrentSweep(f_mppt, f_track)
583       f_track.close()
584       f_mppt.close()
585       print "Performing electronic load IV sweep (bypassed)"
586       filename= "%s_shading%s_bypass2.dat" % (filename_root, shading_percentage)
              #this file will save the measured power and time
587       f=open(foldername + "/" + filename,"w")
588 #     f=open(filename,"w")
589       f.write(delimiter.join(header_string))
590       IVSweep(STARTVOLTAGE, STOPVOLTAGE, f)
591       f.close()
592
593 aa_close(handle)
```

Listing J.2: *solar_code/mppt_switching_1bit_feedback.py*

```
1  #!/bin/env python
2  #This file will switch between IV sweep and MPPT operation, for long-term
       field measurements.
3  #The MPPT part employs 1-bit feedback, where we start out at a current higher
       than Impp,
4  #and reduces it, until one of the buck converter hits its maximum duty cycle.
       We then stop.
5  #
6  #usage: python mppt_switching_1bit_feedback.py root_filename
7
8  #=============================================================================
9  # IMPORTS
10 #=============================================================================
11 import sys
12 import time
13 import datetime
14 #from aardvark32.aardvark_py import *
15 from aardvark64.aardvark_py import *
```

```
16 from array import array
17 from pilawa_instruments import *
18 import os
19 import glob
20 #=============================================================
21 # CONSTANTS
22 #=============================================================
23 #BUFFER_SIZE = 2048
24 I2C_BITRATE =  400
25 PORT  = 0
26 ADDR1=21
27 ADDR2=22
28 ADDR3=23
29 ADDR4=24
30 PACKET_LENGTH=6
31 CLEAR_BUFFER_MESSAGE=99
32 MAX_RESENDS=3 #number of times to resend an i2c message if the count read is
       not correct
33 #READ_DELAY=0.1 #minimum seems to be 0.05 to prevent hangups
34 READ_DELAY=0.1 #minimum seems to be 0.05 to prevent hangups
35 VOUT_DIVIDER=(100+10)/10.0
36 VIN_DIVIDER=(100+10)/10.0
37 VREF=2.54 #microcontroller vref value
38 ADC_MAX=1024
39 #debug=True
40 DUTY_COMMAND=1
41 VOLTAGE_COMMAND=2
42 CURRENT_COMMAND=3
43 ENABLE_COMMAND=4
44 CLEAR_BUFFER_COMMAND = 5
45 VOLTAGE_IN_COMMAND = 6
46 BYPASS_ENABLE_COMMAND = 7
47 BYPASS_DISABLE_COMMAND = 8
48 PING_COMMAND = 9
```

```
49 command_string=("NO_COMMAND","DUTY_COMMAND","VOLTAGE_COMMAND","CURRENT_COMMAND
       ","ENABLE_COMMAND","CLEAR_BUFFER_COMMAND","VOLTAGE_IN_COMMAND","
       BYPASS_ENABLE_COMMAND","BYPASS_DISABLE_COMMAND", "PING_COMMAND")
50 MPPT_SWEEP_STEP = 100
51 MPPT_STEP_SIZE = 6
52 MPPT_MAX_ITERATIONS=20.0
53 NUMREADS = 100 # number of reads for the ADC on each sampling interval
54 DUTY_MIN = 100
55 DUTY_MAX = 990
56
57
58 #switch operation constants
59 STARTVOLTAGE=45.0
60 STOPVOLTAGE=0.0 #MUST HAVE THE .0 AT THE END!
61 NUMSTEPS=50.0
62 FINISHTIME=2000 #i.e 20:00 hours, 8 pm for Americans.
63 NUMSTEPS_CURRENT=5.0
64 STARTCURRENT=5.5
65 STOPCURRENT=1.0
66 # on Saleae logic, set for 7-bit address display only.
67
68 #=============================================================================
69 # CLASSES
70 #=============================================================================
71
72 class converter:
73     def __init__(self, handle, addr=1, duty=500, numreads=10,debug=False):
74         self.handle=handle
75         self.addr=addr
76         self.duty=duty
77         self.numreads=numreads
78         self.debug=debug
79         self.direction=1
80         self.vout=0
81         self.vout_old=0
```

```
82          self.pout_old=0
83          self.iout=0
84          self.vin=0
85
86          #self.initialize()
87
88      def sendMessage(self, command, parameter):
89          command_byte=command
90          transmit_byte_1=parameter>>8 #higher order bits
91          transmit_byte_2=parameter & 0xFF
92          send_array=array('B',[command_byte, transmit_byte_1,transmit_byte_2,
                command_byte, transmit_byte_1, transmit_byte_2])
93
94          if self.debug: print "sending command: %s send_array: %s to addr: %s"
                % (command_string[command_byte], send_array, self.addr)
95          count = aa_i2c_write(self.handle, self.addr, AA_I2C_NO_FLAGS,
                send_array)
96          if ( count != (len(send_array)) ):
97              print "error sending, addr: %s, command %s, receive count: %d" % (
                    self.addr,command_string[command_byte],count)
98              return (0,0)
99          if self.debug: print "send count: %d" % count
100         time.sleep(READ_DELAY)
101         if (command==CLEAR_BUFFER_COMMAND):
102             return (1,1)
103         (count, data_in) = aa_i2c_read(self.handle, self.addr, AA_I2C_NO_FLAGS
                ,PACKET_LENGTH)
104         if self.debug: print "data_in: %s" % data_in
105         if ( count != (PACKET_LENGTH) ):
106             print "error receiving, addr: %s, command %s, receive count: %d" %
                    (self.addr,command_string[command_byte],count)
107             return (0,0)
108         if ((data_in[0] == data_in[3]) and (data_in[1] == data_in[4]) and (
                data_in[2] == data_in[5])):
109             value=data_in[1]*256+data_in[2]
```

```
110             return (1,value)
111         return (0,0)
112
113     def writeDuty(self, duty):
114         (return_value, readback_duty) = self.sendMessage(DUTY_COMMAND, duty)
115         if (return_value == 1) and (readback_duty == duty):
116             self.duty=duty # update internal duty if message was successful
117             return 1 #to indicate that the command was executed properly
118         else:
119             print "writeDuty_error"
120             return 0
121
122     def readVoltage(self,numreads):
123         (return_value, voltage_read) = self.sendMessage(VOLTAGE_COMMAND,
                numreads)
124         if return_value!=1:
125             print "readVoltage_error"
126         self.vout=(voltage_read/numreads)*VREF/ADC_MAX*VOUT_DIVIDER
127         return self.vout
128
129     def readInputVoltage(self,numreads):
130         (return_value, voltage_read) = self.sendMessage(VOLTAGE_IN_COMMAND,
                numreads)
131         if return_value!=1:
132             print "readInputVoltage_error"
133         self.vin=(voltage_read/numreads)*VREF/ADC_MAX*VIN_DIVIDER
134         return self.vin
135
136
137     def readCurrent(self, numreads):
138         (return_value, current) = self.sendMessage(CURRENT_COMMAND, numreads)
139         if return_value!=1:
140             print "readCurrent_error"
141         self.iout=current
142         return current
```

```
143
144     def clearBuffer(self):
145         self.sendMessage(CLEAR_BUFFER_COMMAND, 0)
146
147     def ping(self):
148         return self.sendMessage(PING_COMMAND, 0)
149
150     def bypassEnable(self):
151         #self.enable()
152         #self.writeDuty(1000)
153         #bring a pin down
154         #set the slave select pin low, this will turn on the pmos attached to
                the 5V bus on separate usb cable, and power isolated 5V supply to
                gate of bypass mosfet
155         #aa_gpio_set(handle, 0)
156         self.sendMessage(BYPASS_ENABLE_COMMAND, 0)
157
158     def bypassDisable(self):
159         self.sendMessage(BYPASS_ENABLE_COMMAND, 0)
160         #aa_gpio_set(handle, AA_GPIO_SS)
161
162     def enable(self):
163         (return_value, readback_enable) = self.sendMessage(ENABLE_COMMAND, 0)
164         if (return_value == 1):
165             return 1 #to indicate that the command was executed properly
166         else:
167             print "enable error"
168             return 0
169
170
171     def MPPTrack(self, f, current):
172         self.readVoltage(self.numreads)
173         self.readCurrent(self.numreads)
174         self.readInputVoltage(self.numreads)
175         pout=self.vout*1.0
```

```
176         #header_string_mppt=("addr", "time", "vout", "vin", "duty", "direction
                \n")
177         timestamp = datetime.datetime.now()
178         f.write("%s%s%s%s%s%s%s%s%s%s%s%s%s%s\n" % (self.addr, delimiter,
                timestamp.strftime("%H%M%S%f"), delimiter,      self.vout,
                delimiter, self.vin, delimiter, self.duty, delimiter, self.
                direction, delimiter, current))
179
180         #pout=self.vout*self.iout
181         if (pout <= self.pout_old):
182             self.direction=-1*self.direction #change direction if we're going
                    the wrong way
183             print "addr: %s changed direction" % self.addr
184         self.pout_old = pout #update pout_old with new value
185         newduty=self.duty+self.direction*MPPT_STEP_SIZE
186         if (newduty > DUTY_MIN) and (newduty < DUTY_MAX):
187             self.writeDuty(newduty)
188
189     def MPPTSweep(self, f, current):
190         numreads=100
191     #     if converter.debug:
192         print "MPPTSweep entered"
193         self.writeDuty(DUTY_MIN)
194         self.enable()
195         time.sleep(0.5)
196         vout=self.readVoltage(self.numreads)
197     #     iout=self.readCurrent(numreads)
198         iout=1
199         pout_peak=vout*iout
200         duty_peak=self.duty
201         while (self.duty < DUTY_MAX):
202             newduty=self.duty + MPPT_SWEEP_STEP
203             self.writeDuty(newduty)
204     #         if self.debug:
205
```

```
206                 time.sleep(.0001)
207         #         time.sleep(1)
208                 vout=self.readVoltage(self.numreads)
209                 vin=self.readInputVoltage(self.numreads)
210                 print "Converter %s, writing duty: %s, vout: %s" % (self.addr,
                        newduty, vout)
211                 timestamp = datetime.datetime.now()
212                 f.write("%s%s%s%s%s%s%s%s%s%s%s%s%s%s\n" % (self.addr, delimiter,
                        timestamp.strftime("%H%M%S%f"), delimiter, self.vout,
                        delimiter, self.vin, delimiter, self.duty, delimiter,      self
                        .direction, delimiter, current))
213                 #iout=self.readCurrent(numreads)
214                 iout=1
215                 pout=vout*iout
216                 if (pout >= pout_peak):
217                     duty_peak=self.duty
218                     pout_peak=pout
219                 else:
220                     self.writeDuty(duty_peak)
221                     return
222         #converter.writeDuty(duty_peak)
223
224
225 #=============================================================================
226 # FUNCTIONS
227 #=============================================================================
228
229 def MPPTrack(converter, f, current):
230         vout = converter.readVoltage(converter.numreads)
231         #converter.readCurrent(converter.numreads)
232         vin = converter.readInputVoltage(converter.numreads)
233         pout = vout*1.0
234         #header_string_mppt=("addr", "time", "vout", "vin", "duty", "direction\n")
235         timestamp = datetime.datetime.now()
```

```
236         f.write("%s%s%s%s%s%s%s%s%s%s%s%s%s%s\n" % (converter.addr, delimiter,
                timestamp.strftime("%H%M%S%f"), delimiter, vout, delimiter, vin,
                delimiter, converter.duty, delimiter, converter.direction, delimiter,
                current))
237
238         #pout=converter.vout*converter.iout
239         print "pout: %s pout_old: %s direction: %s" % (pout, converter.pout_old,
                converter.direction)
240         if (pout <= converter.pout_old):
241             converter.direction=-1*converter.direction #change direction if we're
                    going the wrong way
242             print "addr: %s changed direction" % converter.addr
243         converter.pout_old = pout #update pout_old with new value
244         print "pout_old: %s " % converter.pout_old
245         newduty=converter.duty+converter.direction*MPPT_STEP_SIZE
246         if (newduty > DUTY_MIN) and (newduty < DUTY_MAX):
247             converter.writeDuty(newduty)
248
249
250 def MPPTSweep(converter, f, current):
251     numreads=100
252 #     if converter.debug:
253     print "MPPTSweep entered"
254     converter.writeDuty(DUTY_MIN)
255     converter.enable()
256     time.sleep(0.5)
257     vout=converter.readVoltage(converter.numreads)
258 #     iout=converter.readCurrent(numreads)
259     iout=1
260     pout_peak=vout*iout
261     duty_peak=converter.duty
262     while (converter.duty < DUTY_MAX):
263         newduty=converter.duty + MPPT_SWEEP_STEP
264         converter.writeDuty(newduty)
265 #         if converter.debug:
```

```
266
267             time.sleep(.0001)
268     #         time.sleep(1)
269             vout=converter.readVoltage(converter.numreads)
270             vin=converter.readInputVoltage(converter.numreads)
271             print "Converter %s, writing duty: %s, vout: %s" % (converter.addr,
                    newduty, vout)
272             timestamp = datetime.datetime.now()
273             f.write("%s%s%s%s%s%s%s%s%s%s%s%s%s\n" % (converter.addr, delimiter,
                    timestamp.strftime("%H%M%S%f"), delimiter, converter.vout,
                    delimiter, converter.vin, delimiter, converter.duty, delimiter,
                    converter.direction, delimiter, current))
274             #iout=converter.readCurrent(numreads)
275             iout=1
276             pout=vout*iout
277             if (pout >= pout_peak):
278                 duty_peak=converter.duty
279                 pout_peak=pout
280             else:
281                 converter.writeDuty(duty_peak)
282                 return
283         #converter.writeDuty(duty_peak)
284
285     def MPPTSweepList(converter_list, f, current):
286     #     if converter.debug:
287         print "MPPTSweepList entered"
288
289         for converter in converter_list:
290             converter.writeDuty(DUTY_MIN)
291             time.sleep(0.5)
292             vout=converter.readVoltage(converter.numreads)
293         #     iout=converter.readCurrent(numreads)
294             iout=1
295             pout_peak=vout*iout
296             duty_peak=converter.duty
```

```
297             while (converter.duty < DUTY_MAX - MPPT_SWEEP_STEP): #don't want to go
                    all the way up
298                 newduty=converter.duty + MPPT_SWEEP_STEP
299                 converter.writeDuty(newduty)
300         #           if converter.debug:
301                 time.sleep(.0001)
302         #           time.sleep(1)
303                 vout=converter.readVoltage(converter.numreads)
304                 vin=converter.readInputVoltage(converter.numreads)
305                 print "Converter %s, writing duty: %s, vout: %s" % (converter.addr
                        , newduty, vout)
306                 timestamp = datetime.datetime.now()
307                 for record_converter in converter_list:
308                     timestamp = datetime.datetime.now()
309                     record_converter.readVoltage(record_converter.numreads)
310                     converter.readInputVoltage(record_converter.numreads)
311                     f.write("%s%s%s%s%s%s%s%s%s%s%s%s%s\n" % (record_converter.
                            addr, delimiter, timestamp.strftime("%H%M%S%f"), delimiter
                            , record_converter.vout, delimiter, record_converter.vin,
                            delimiter, record_converter.duty, delimiter,
                            record_converter.direction, delimiter, current))
312                 #iout=converter.readCurrent(numreads)
313                 iout=1
314                 pout=vout*iout
315                 if (pout >= pout_peak):
316                     duty_peak=converter.duty
317                     pout_peak=pout
318                 else:
319                     converter.writeDuty(duty_peak)
320                     break
321         #converter.writeDuty(duty_peak)
322
323
324 def MPPTPing(converter_list):
325     converter_alive=0
```

```
326     while converter_alive < len(converter_list):
327         converter_alive=0
328         for x in converter_list:
329             #x.clearBuffer()
330             time.sleep(0.2)
331             print "Pinging converter with addr: %s converter_alive: %s len  (
                    converter_list): %s" % (x.addr, converter_alive, len(
                    converter_list))
332             time.sleep(0.2)
333             #x.ping()
334             (return_code, return_value) = x.ping()
335             if return_code == 1:
336                 converter_alive = converter_alive + 1
337             print "return_code: %s" % return_code
338
339
340 #================================================================
341 # MAIN PROGRAM
342 #================================================================
343 #if (len(sys.argv) < 2):
344 #    print "usage: aai2c_file  filename"
345 #    print "  'filename' is the filename where to send processed data"
346
347 #    sys.exit()
348
349 root_filename = sys.argv[1]
350
351 handle = aa_open(PORT)
352 if (handle <= 0):
353     print "Unable to open Aardvark device on port %d" % PORT
354     print "Error code = %d" % handle
355     sys.exit()
356
357 # Ensure that the I2C subsystem is enabled, also do I2C
358 #aa_configure(handle,  AA_CONFIG_SPI_I2C)
```

```
359 aa_configure(handle,  AA_CONFIG_GPIO_I2C)
360
361
362 #this will enable slave_select GPIO as output pin. See page 57 of aardvark
        datasheet
363 aa_gpio_direction(handle, AA_GPIO_SS)
364
365 aa_gpio_pullup(handle, AA_GPIO_SS)
366
367 #set slave select high, this will turn off bypass power (set the pmos gate
        high)
368 aa_gpio_set(handle, AA_GPIO_SS)
369
370
371 # Enable the I2C bus pullup resistors (2.2k resistors).
372 # This command is only effective on v2.0 hardware or greater.
373 # The pullup resistors on the v1.02 hardware are enabled by default.
374 aa_i2c_pullup(handle, AA_I2C_PULLUP_BOTH)
375
376 # Enable the Aardvark adapter's power supply.
377 # This command is only effective on v2.0 hardware or greater.
378 # The power pins on the v1.02 hardware are not enabled by default.
379 aa_target_power(handle, AA_TARGET_POWER_BOTH)
380
381 # Set the bitrate
382 bitrate = aa_i2c_bitrate(handle, I2C_BITRATE)
383 print "Bitrate set to %d kHz" % bitrate
384
385 delimiter=', '
386 t=time.strftime('%Y%m%d')
387 current_directory=os.curdir
388 print "Current directory: %s" % current_directory
389 foldername=current_directory + "/data/" + t
390 print foldername
391 if not os.path.exists(foldername):
```

```
392         os.makedirs(foldername)
393     #f=open(filename,"w")
394     #header_string=("addr", "vout", "iout", "vin", "pout", "pout_old", "direction
            ", "duty", "\n")
395     #header_string=("%addr", "time", "duty", "vin", "vout", "iout\n")
396     #f.write(delimiter.join(header_string))
397
398
399     #pseudo-code:
400     #go through each device in a list, read its duty cycle, vin, vout, il
401
402     addrList = [ADDR1,ADDR2,ADDR3,ADDR4]
403     #addrList=[ADDR1]
404     #debug=True
405     debug=False
406     numreads=10
407     converter1=converter(handle, ADDR1, 500, NUMREADS, debug)
408     converter2=converter(handle, ADDR2, 500, NUMREADS, debug)
409     converter3=converter(handle, ADDR3, 500, NUMREADS, debug)
410     converter4=converter(handle, ADDR4, 500, NUMREADS, debug)
411
412     converter_list=[converter1, converter2, converter3]
413     #converter_list=[converter2]
414
415     gpib = prologix_serial(port="/dev/ttyUSB0", baud=115200, debug=False, timeout
            =5)
416     eload= prologix_6060b(prologix=gpib, addr=5, mode="VOLT", rang="20", debug=
            True)
417     #eload.setMode("CURR")
418
419
420     MPPTPing(converter_list)
421
422
423     #PSEUDO-CODE
```

```
424
425 #in directory, create subdirectory with today's date.
426 #operation: toggle between MPPT operation and regular operation. Number
        filenames, all ending in odd are regular, all even are distributed MPPT
427 #setup MPPT communication
428 header_string=("time", "iload", "vload\n")
429 header_string_mppt=("addr", "time", "vout", "vin", "duty", "direction", "
        current\n")
430 existing_files_list = glob.glob("./data/%s/data*.dat" %t)
431 #if directory exists, then counter = number of files named test*.txt
432 i = len(existing_files_list)
433 filecounter=0
434 time_counter=0
435 duty_overflow = False #indicator used for 1-bit feedback to let us know that
        load current should not be reduced further.
436 keeprunning = True
437 while keeprunning is True:
438 #while (time_counter < FINISHTIME): #run this loop until a pre-set finish time
        .
439     filename= "%s_data%s.dat" % (root_filename, filecounter) #this file will
            save the measured power and time
440     f=open(foldername + "/" + filename,"w")
441     timestamp = time.strftime('%Y%m%d%H%M%S')
442     f.write("time: %s MPPT_STEP_SIZE: %s MPPT_MAX_ITERATIONS: %s NUMREADS: %s
            DUTY_MIN: %s DUTY_MAX: %s\n" %(timestamp, MPPT_STEP_SIZE,
            MPPT_MAX_ITERATIONS,NUMREADS,DUTY_MIN, DUTY_MAX))
443     f.write(delimiter.join(header_string))
444     #%%%%%%%%%%%%%
445     #SWEEP SECTION
446     #%%%%%%%%%%%%%
447     if filecounter%2==0: #even number, do a regular sweep
448         print "doing regular sweep, filecounter: %s" % filecounter
449         sweep_voltage=STARTVOLTAGE
450         step_size=(STOPVOLTAGE-STARTVOLTAGE)/NUMSTEPS #will be negative if we
                stop lower than we start
```

```
451 #          print "step_size %s" % step_size
452             eload.setMode("VOLT")
453             eload.setSlew(2000000)
454             eload.setValue(sweep_voltage)
455             MPPTPing(converter_list)
456             time.sleep(0.5)
457             for x in converter_list:
458                 x.bypassEnable()
459                 #bypass enabled
460                 aa_gpio_set(handle, 0)
461                 time.sleep(READ_DELAY)
462             while (sweep_voltage > STOPVOLTAGE): #change here is wanting to go
                    different direction
463 #                for x in converter_list:
464 #                    x.bypassEnable()
465                 eload.setValue(sweep_voltage)
466                 eload_current=eload.readCurrent()
467                 eload_voltage=eload.readVoltage()
468                 timestamp = datetime.datetime.now()
469                 f.write("%s%s%s%s%s\n" % ( timestamp.strftime("%H%M%S%f"),
                        delimiter, eload_current, delimiter, eload_voltage))
470                 sweep_voltage=sweep_voltage+step_size  #step down in voltage to
                        keep mppt electronics up (for bypass purposes)
471             print ("bypass_sweep_done")
472             f.close()
473
474         #%%%%%%%%%%%%%%
475         #MPPT SECTION
476         #%%%%%%%%%%%%%%
477         else: #odd number, do a sweep with distributed MPPTs
478             filename_mppt="%s_mppt%s.dat" % (root_filename, filecounter) #this
                    file is for storing the operation of the MPPTs for debugging and
                    checking their operation
479             f_mppt=open(foldername + "/" + filename_mppt,"w")
```

```
480             f_mppt.write("time: %s MPPT_STEP_SIZE: %s MPPT_MAX_ITERATIONS: %s
                    NUMREADS: %s DUTY_MIN: %s DUTY_MAX: %s\n" %(timestamp,
                    MPPT_STEP_SIZE,MPPT_MAX_ITERATIONS,NUMREADS,DUTY_MIN, DUTY_MAX))
481             f_mppt.write(delimiter.join(header_string_mppt))
482             print "doing MPPT sweep, filecounter: %s" % filecounter
483             #bypass disabled
484             aa_gpio_set(handle, AA_GPIO_SS)
485             #let things settle for a bit
486             time.sleep(0.1)
487             sweep_current=STARTCURRENT
488             current_step_size=(STOPCURRENT-STARTCURRENT)/NUMSTEPS_CURRENT
489             print "current_step_size: %s" % current_step_size
490             eload.setMode("CURR")
491             eload.setValue(sweep_current)
492             eload.setSlew(2000000)
493             print "About to start MPPT sweep"
494             time.sleep(0.5)
495             MPPTPing(converter_list)
496 #            for x in converter_list:
497  #                MPPTPing(converter_list)
498 #                MPPTSweep(x, f_mppt, sweep_current)
499                 #x.MPPTSweep(f_mppt, sweep_current)
500             for converter in converter_list:
501                 converter.writeDuty(DUTY_MIN)
502                 converter.enable()
503                 time.sleep(0.2)
504             time.sleep(0.5)
505             MPPTSweepList(converter_list, f_mppt, sweep_current)
506             MPPTSweepList(converter_list, f_mppt, sweep_current)
507             duty_overflow = False
508             while ((sweep_current > STOPCURRENT) and (duty_overflow is False)): #
                    only need to change here if sweeping from high to low, stepsize is
                     negative if going from high to low
509 #            while (sweep_current < STOPCURRENT):
510                 print "setting current value to: %s" % sweep_current
```

```
511             eload.setValue(sweep_current)
512             mppt_iterator=0
513             print "starting tracking"
514             max_eload_current = 0
515             max_eload_voltage = 0
516             max_eload_power = 0
517             while (mppt_iterator < MPPT_MAX_ITERATIONS):
518                 for x in converter_list:
519                     #x.MPPTrack(f_mppt, sweep_current)
520                     MPPTrack(x, f_mppt, sweep_current)
521                     #1 bit feedback, check for duty cycle maxed-out
522                     if x.duty >= ( DUTY_MAX - MPPT_STEP_SIZE):
523                         duty_overflow = True
524                         print "Duty overflow, done with current sweep"
525 
526                 mppt_iterator=mppt_iterator+1
527                 eload_current = eload.readCurrent()
528                 eload_voltage = eload.readVoltage()
529                 eload_power = float(eload_current) * float(eload_voltage)
530                 if (eload_power > max_eload_power):
531                     max_eload_power = eload_power
532                     max_eload_current = eload_current
533                     max_eload_voltage = eload_voltage
534             sweep_current=sweep_current + current_step_size
535             timestamp = datetime.datetime.now()
536             f.write("%s%s%s%s%s\n" % ( timestamp.strftime("%H%M%S%f"),
                    delimiter, max_eload_current, delimiter, max_eload_voltage))
537             print "done tracking"
538             #print "iload:%s" % eload_current ,
539             #print "vload:%s" % eload_voltage
540         f.close()
541     time_counter=int(time.strftime('%H%M'))
542     filecounter=filecounter+1
543 
544 
```

```
545 # Close the device
546 aa_close(handle)
```

Bibliography

[1] P. Hazucha, G. Schrom, J. Hahn, B. Bloechel, P. Hack, G. Dermer, S. Narendra, D. Gardner, T. Karnik, V. De, and S. Borkar, "A 233-MHz 80%-87% efficient four-phase dc-dc converter utilizing air-core inductors on package," *IEEE Journal of Solid-State Circuits*, vol. 40, no. 4, pp. 838–845, April 2005.

[2] J. Sun, J.-Q. Lu, D. Giuliano, T. P. Chow, and R. J. Gutmann, "3D power delivery for microprocessors and high-performance ASICs," in *Applied Power Electronics Conference*, Feb./March 2007, pp. 127–133.

[3] G. Schrom, P. Hazucha, F. Paillet, D. J. Rennie, S. T. Moon, D. S. Gardner, T. Kamik, P. Sun, T. T. Nguyen, M. J. Hill, K. Radhakrishnan, and T. Memioglu, "A 100MHz eight-phase buck converter delivering 12A in 25mm^2 using air-core inductors," in *Applied Power Electronics Conference*, Feb./March 2007, pp. 727–730.

[4] P. Lin and L. Chua, "Topological generation and analysis of voltage multiplier circuits," *IEEE Transactions on Circuits and Systems*, vol. 24, no. 10, pp. 517–530, Oct 1977.

[5] S. Cheong, S. Chung, and A. Ioinovici, "Development of power electronics converters based on switched-capacitor circuits," in *IEEE International Symposium on Circuits and Systems*, vol. 4, 3-6 May 1992, pp. 1907–1910vol.4.

[6] K. Ngo and R. Webster, "Steady-state analysis and design of a switched-capacitor dc-dc converter," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 30, no. 1, pp. 92–101, Jan. 1994.

[7] M. Makowski and D. Maksimovic, "Performance limits of switched-capacitor dc-dc converters," in *Power Electronics Specialists Conference*, vol. 2, June 1995, pp. 1215–1221.

[8] M. Makowski, "Realizability conditions and bounds on synthesis of switched-capacitor dc-dc voltage multiplier circuits," *IEEE Transactions on Circuits and Systems-I*, vol. 44, no. 8, pp. 684–691, Aug. 1997.

[9] B. Arntzen and D. Maksimovic, "Switched-capacitor dc/dc converters with resonant gate drive," *IEEE Transactions on Power Electronics*, vol. 13, no. 5, pp. 892–902, Sept. 1998.

[10] M. Seeman and S. Sanders, "Analysis and optimization of switched-capacitor dc-dc converters," in *IEEE Computers in Power Electronics Workshop*, 16-19 July 2006, pp. 216–224.

[11] S. Cheong, H. Chung, and A. Ioinovici, "Inductorless dc-to-dc converter with high power density," *IEEE Transactions on Industrial Electronics*, vol. 41, no. 2, pp. 208–215, Apr 1994.

[12] H.-H. Chung, S. Hui, and S. Tang, "Development of a multistage current-controlled switched-capacitor step-down dc/dc converter with continuous input current," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 47, no. 7, pp. 1017–1025, July 2000.

[13] G. Zhu, H. Wei, I. Batarseh, and A. Ioinovici, "A new switched-capacitor dc-dc converter with improved line and load regulations," in *IEEE International Symposium on Circuits and Systems*, vol. 5, 30 May-2 June 1999, pp. 234–237.

[14] J. Sun, M. Xu, Y. Ying, and F. Lee, "High power density, high efficiency system two-stage power architecture for laptop computers," in *Power Electronics Specialists Conference*, June 2006, pp. 1–7.

[15] O. Abutbul, A. Gherlitz, Y. Berkovich, and A. Ioinovici, "Step-up switching-mode converter with high voltage gain using a switched-capacitor circuit," *IEEE Transactions on Circuits and Systems I.*, vol. 50, no. 8, pp. 1098–1102, August 2003.

[16] B. Axelrod, Y. Berkovich, and A. Ioinovici, "Single-switch single-stage switched-capacitor buck converter," in *Proc. of NORPIE 2004, 4th Nordic workshop on Power and Industrial Electronics*, June 2004.

[17] Y. Ramadass and A. Chandrakasan, "Minimum energy tracking loop with embedded dc-dc converter delivering voltages down to 250mv in 65nm cmos," in *Proc. Digest of Technical Papers. IEEE International Solid-State Circuits Conference ISSCC 2007*, 2007, pp. 64–587.

[18] R. Pilawa-Podgurski, D. Giuliano, and D. Perreault, "Merged two-stage power converter architecture with soft charging switched-capacitor energy transfer," in *39th IEEE Power Electronics Specialists Conference*, 2008.

[19] C. Tse, S. Wong, and M. Chow, "On lossless switched-capacitor power converters," *IEEE Transactions on Power Electronics*, vol. 10, no. 3, pp. 286–291, May 1995.

[20] M. D. Seeman and S. R. Sanders, "Analysis and optimization of switched-capacitor dc–dc converters," vol. 23, no. 2, pp. 841–851, 2008.

[21] R. C. Pilawa-Podurski and D. J. Perreault, "Merged two-stage power converter with soft charging switched-capacitor stage in 180 nm cmos," in *37th European Solid State Circuits Conference (ESSCIRC)*, 2011.

[22] H.-P. Le, M. Seeman, S. R. Sanders, V. Sathe, S. Naffziger, and E. Alon, "A 32nm fully integrated reconfigurable switched-capacitor dc-dc converter delivering 0.55w/mm2 at 81% efficiency," in *Proc. IEEE Int. Solid-State Circuits Conf. Digest of Technical Papers (ISSCC)*, 2010, pp. 210–211.

[23] R. J. Baker, *CMOS Circuit Design, Layou, and Simulation*, S. K. Tewksbury and J. E. Brewer, Eds. IEEE Press, 2008.

[24] R. Erickson and D. Maksimovic, *Fundamentals of Power Electronics*. Kluwer Academics, 2000.

[25] C. Basso, *Switch-Mode Power Supplies - Spice Simulations and Practical Designs*. McGraw-Hill, 2008.

[26] D. H. Venable, "Practical techniques for analyzing, measuring, and stabilizing feedback control loops in switching regulators and converters," in *Proceedings of the Seventh National Power Conersion Conference, POWERCON 7*, 1980, pp. I2–1 to I2–17.

[27] A. I. Pressman, *Switching Power Supply Design*, 2nd ed. McGraw-Hill, 1998.

[28] P. Y. Wu, S. Y. S. Tsui, and P. K. T. Mok, "Area- and power-efficient monolithic buck converters with pseudo-type iii compensation," vol. 45, no. 8, pp. 1446–1455, 2010.

[29] D. Mattingly, "Designing stable compensation networks for single phase voltage mode buck regulators," Intersil," Technical Brief, 2003.

[30] H. Kolm, "Solar-battery power source, Quarterly Progress Report, Solid State Research, Group 35, MIT-Lincoln Laboratory," 1956.

[31] I. Celanovic, N. Jovanovic, and J. Kassakian, "Two-dimensional tungsten photonic crystals as selective thermal emitters," *Applied Physics Letters*, vol. 92, no. 19, pp. 193 101–+, May 2008.

[32] I. Celanovic, F. O'Sullivan, M. Ilak, J. Kassakian, and D. Perreault, "Design and optimization of one-dimensional photonic crystals for thermophotovoltaic applications," *Optics Letters*, vol. 29, p. 863, 2004.

[33] B. Wernsman, R. Siergiej, S. Link, R. Mahorter, M. Palmisiano, R. Wehrer, R. Schultz, G. Schmuck, R. Messham, S. Murray, C. Murray, F. Newman, D. Taylor, D. DePoy, and T. Rahmlow, "Greater than 20 percent radiant heat conversion efficiency of a thermophotovoltaic radiator/module system using reflective spectral control," *IEEE Transactions on Electron Devices*, vol. 51, no. 3, pp. 512– 515, 2004.

[34] B. Blackwell, "Design, fabrication, and characterization of a micro fuel processor," Ph.D. dissertation, Massachusetts Institute of Technology, 2008.

[35] M. W. Dashiell, J. F. Beausang, H. Ehsani, G. J. Nichols, D. M. Depoy, L. R. Danielson, P. Talamo, K. D. Rahner, E. J. Brown, S. R. Burger, P. M. Fourspring, W. F. TopperJr., P. F. Baldasaro, C. A. Wang, R. K. Huang, M. K. Connors, G. W. Turner, Z. A. Shellenbarger, G. Taylor, J. Li, R. Martinelli, D. Donetski, S. Anikeev, G. L. Belenky, and S. Luryi, "Quaternary InGaAsSb thermophotovoltaic diodes," *IEEE Transactions on Electron Devices*, vol. 53, no. 12, pp. 2879–2891, Dec. 2006.

[36] C. Sullivan and M. Powers, "A high-efficiency maximum power point tracker for photovoltaic arrays in a solar-powered race vehicle," in *Proc. 24th Annual IEEE Power Electronics Specialists Conference PESC '93 Record*, 1993, pp. 574–580.

[37] A. Woyte, J. Nijs, and R. Belmans, "Partial shadowing of photovoltaic arrays with different system configurations: literature review and field test results." *Solar Energy*, vol. 74, pp. 217–233, 2003.

[38] T. Esram and P. Chapman, "Comparison of photovoltaic array maximum power point tracking techniques," *IEEE Transaction on Energy Conversion*, vol. 22, no. 2, pp. 439–449, 2007.

[39] Atmel, "AVR121: Enhancing ADC resolution by oversampling, Application Note," 2005.

[40] X. Zhou, P. Xu, and F. Lee, "A novel current-sharing control technique for low-voltage high-current voltage regulator module applications," vol. 15, no. 6, pp. 1153–1162, 2000.

[41] A. Oppenheim and R. Schafer, *Discrete-time Signal Processing*, 2nd ed., A. Oppenheim, Ed. Prentice Hall, 1999.

[42] W. Bennet, "Spectra of quantized signals," Bell System technical Journal, Tech. Rep., 1948.

[43] W. Chan, P. Bermel, R. Pilawa-Podgurski, C. Marton, K. Jensen, M. Soljacic, J. Joannopoulos, and I. Celanovic, "A high-efficiency millimeter-scale thermophotovoltaic generator," in *TPV-9 World Conference at the 25th European Photovoltiac Solar Energy Conference and Exhibition*, 2010.

[44] O. Nielsen, L. Arana, C. Baertsch, K. Jensen, and M. Schmidt, "A thermophotovoltaic micro-generator for portable power applications," in *TRANSDUCERS, Solid-State Sensors, Actuators and Microsystems, 12th International Conference on, 2003*, vol. 1, June 2003, pp. 714–717 vol.1.

[45] A. Kamitani, S. Morishita, H. Kotaki, and S. Arscott, "Miniaturized microdmfc using silicon microsystems techniques: performances at low fuel flow rates," *J. Micromech. Microeng.*, vol. 18, p. 125019, 2008.

[46] O. Wasynczuk, "Dynamic behavior of a class of photovoltaic power systems," vol. PAS-102, no. 9, pp. 3031–3037, 1983.

[47] K. Lundberg, *Become one with the transistor for solid-state circuits.* MIT Department of Electrical Engineering and Computer Science, 2006.

[48] B. Gilbert, "Translinear circuits: a proposed classification," *Electronics Letters*, vol. 11, no. 1, pp. 14–16, 1975.

[49] T. Serrano-Gotarredona, B. Linares-Barranco, and A. G. Andreou, "A general translinear principle for subthreshold mos transistors," vol. 46, no. 5, pp. 607–616, 1999.

[50] S. Al-Sarawi, "Low power schmitt trigger circuit," *Electronics Letters*, vol. 38, no. 18, pp. 1009–1010, 2002.

[51] Y.-F. Chou, "Schmitt-trigger circuit with low power consumption," 2000.

[52] T. Takayama and D. Maksimovic, "A power stage optimization method for monolithic dc-dc converters," in *Power Electronics Specialists Conference*, 2006.

[53] K. Liu, R. Oruganti, and F. Lee, "Quasi-resonant converters – topologies and characteristics," *IEEE Transactions on Power Electronics*, 1987.

[54] V. Vorperian, "Quasi-square-wave converters: topologies and analysis," vol. 3, no. 2, pp. 183–191, 1988.

[55] C. P. Henze, H. C. Martin, and D. W. Parsley, "Zero-voltage switching in high frequency power converters using pulse width modulation," in *Proc. 1988. Third Annual IEEE Applied Power Electronics Conf and Exposition APEC '88*, 1988, pp. 33–40.

[56] D. M. Sable, F. C. Lee, and B. H. Cho, "A zero-voltage-switching bidirectional battery charger/discharger for the NASA EOS satellite," in *Applied Power Electronics Conf and Exposition*, 1992, pp. 614–621.

[57] "Trends in photovoltaic applications. survey report of seleted IEA countries between 1992 and 2006." International Energy Agency Photovoltaic Power Systems, Tech. Rep. IEA-PVPS T1-16:2007, 2007, [Online] www.iea-pvps.org. [Online]. Available: www.iea-pvps.org

[58] S. Kjaer, J. Pedersen, and F. Blaabjerg, "A review of single-phase grid-connected inverters for photovoltaic modules," vol. 41, no. 5, pp. 1292–1306, 2005.

[59] J. Myrzik and M. Calais, "String and module integrated inverters for single-phase grid connected photovoltaic systems - a review," in *Proc. IEEE Bologna Power Tech*, June 2003.

[60] *Sunpower 210 Solar Panel Datasheet*, Sunpower Corporation, [Online] Available at www.sunpowercorp.com.

[61] M. A. Quintana, D. L. King, T. J. McMahon, and C. R. Osterwald, "Commonly observed degradaion in field-aged photovoltaic modules," in *IEEE 29th Photovoltaic Specialists Conference*, May 2002, pp. 1436–1439.

[62] B. Decker and U. Jahn, "Performance of 170 grid connected pv plants in northern germany - analysis of yields and optimization potentials," *Solar Energy*, vol. 59, no. 4-6, pp. 127–133, 1997.

[63] Q. Li and P. Wolfs, "A review of the single phase photovoltaic module integrated converter topologies with three different dc link configurations," vol. 23, no. 3, pp. 1320–1333, 2008.

[64] R. Rohrig and J. Steger, "Circuit arrangements for photovoltaic system," United States Patent Application Publication US 2005/0172995 A1, August 2005.

[65] P. Wolfs and L. Tang, "A single cell maximum power point tracking converter without a current sensor for high performance vehicle solar arrays," in *Proc. IEEE 36th Power Electronics Specialists Conference PESC '05*, 2005, pp. 165–171.

[66] K. Wilson, D. De Ceuster, and S. R. A., "Measuring the effect of cell mismatch on module output," in *4th IEEE World Conference on Photovoltaic Energy Conversion*, vol. 1, May 2006, pp. 916–919.

[67] H. Field and A. Gabor, "Cell binning method analysis to minimize mismatch losses and performance variation in si-based modules," in *Conference Record of the Twenty-Ninth IEEE Photovoltaic Specialists Conference*, 2002, pp. 418–421.

[68] V. G. Karpov, A. D. Compaan, and D. Shvydka, "Effets of nonuniformity in thin-film photovoltaics," *Applied Physics Letters*, vol. 80, no. 22, 2002.

[69] N. Kaushika and A. K. Rai, "An investigation of mismatch losses in solar photovoltaic cell networks," *Energy*, vol. 32, no. 5, pp. 755–759, May 2007.

[70] G. R. Walker and P. C. Sernia, "Cascaded dc-dc converter connection of photovoltaic modules," vol. 19, no. 4, pp. 1130–1139, 2004.

[71] L. Linares, R. W. Erickson, S. MacAlpine, and M. Brandemuehl, "Improved energy capture in series string photovoltaics via smart distributed power electronics," in *Proc. Twenty-Fourth Annual IEEE Applied Power Electronics Conf. and Exposition APEC 2009*, 2009, pp. 904–910.

[72] T. Shimizu, M. Hirakata, T. Kamezawa, and H. Watanabe, "Generation control circuit for photovoltaic modules," *Power Electronics, IEEE Transactions on*, vol. 16, no. 3, pp. 293 –300, may 2001.

[73] S. Araujo, P. Zacharias, and B. Sahan, "Novel grid-connected non-isolated converters for photovoltaic systems with grounded generator," in *IEEE Power Electronics Specialists Conference*, 2008.

[74] G. Schrom, P. Hazucha, F. Paillet, D. Gardner, S. Moon, and T. Karnik, "Optimal design of monolithic integrated dc-dc converters," in *Proc. IEEE International Conference on Integrated Circuit Design and Technology ICICDT '06*, 2006, pp. 1–3.

[75] E. Roman, R. Alonso, P. Ibanez, S. Elorduizapatarietxe, and D. Goitia, "Intelligent pv module for grid-connected pv systems," *Industrial Electronics, IEEE Transactions on*, vol. 53, no. 4, pp. 1066 –1073, june 2006.

[76] L. Linares, "Design and implementation of module integrated converters for series connected photovoltaic strings," Master's thesis, Unisversity of Colorado, 2009.

[77] U. D. of Energy, "Sunshot initative," 2011. [Online]. Available: http://www1.eere.energy.gov/solar/sunshot/index.html

[78] R. C. N. Pilawa-Podgurski, N. A. Pallo, W. R. Chan, D. J. Perreault, and I. L. Celanovic, "Low-power maximum power point tracker with digital control for thermophotovoltaic generators," in *Proc. Twenty-Fifth Annual IEEE Applied Power Electronics Conf. and Exposition (APEC)*, 2010, pp. 961–967.

[79] D. Perreault and J. Kassakian, "Distributed interleaving of paralleled power converters," vol. 44, no. 8, pp. 728–734, 1997.

[80] D. Perreault, J. Selders, R.L., and J. Kassakian, "Frequency-based current-sharing techniques for paralleled power converters," vol. 13, no. 4, pp. 626–634, 1998.

[81] D. Perreault, K. Sato, J. Selders, R.L., and J. Kassakian, "Switching-ripple-based current sharing for paralleled power converters," vol. 46, no. 10, pp. 1264–1274, 1999.

[82] G. Barbose, N. Darghouth, R. Wiser, and J. Seel, "Tracking the sun iv - a historical summary of the installed cost of photvoltaics in the united states from 1998 to 2010," Lawrence Berkeley National Laboratory, Tech. Rep., 2011.