VOLTAGE REGULATION OF A SERIES STACKED SYSTEM OF DIGITAL LOADS BY DIFFERENTIAL POWER PROCESSING

BY

DIPANJAN DAS

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2016

Urbana, Illinois

Adviser:

Professor Philip T. Krein

ABSTRACT

A modern high-end multi-core microprocessor has very stringent power supply requirements. It can draw hundreds of amperes of current at supply voltages as low as 0.8 V. As the supply voltages keep decreasing, the power delivery to meet the supply requirements is becoming increasingly difficult and inefficient. However, the presence of multiple cores in the microprocessor offers us a way to power it at a higher voltage by series-stacking the cores. Differential power processing has been shown to be an efficient way to series-stack server loads. In this work we study the dynamics of the element-to-element DPP topology implemented with bi-directional buck-boost converters. Some of its dynamic drawbacks are pointed out and a topological modification to counter those drawbacks is proposed. We then develop a linear control to regulate processor core voltages in a series stack of 4 cores. A hysteretic control to accommodate light load modes in the bi-directional regulating converters is also discussed. Both the linear and the hysteretic controller are implemented successfully in hardware and efficiency improvement due to light-load modes is demonstrated.

To my parents, for their love and support.

ACKNOWLEDGMENTS

I would like to thank my adviser, Professor Philip T. Krein, for giving me the opportunity to work under his guidance. He has always encouraged me and given me valuable insights and resources to solve problems. I would also like to thank Professor Robert Pilawa for providing me with valuable and enjoyable learning experiences in the last two years. I would also like to thank Srikanthan for motivating and providing me with energy during my first two years here.

I would like to thank the Strategic Research Initiative and Grainger Center for Electric Machinery and Electromechanics for funding this work.

Finally I would like to thank my family, who have always loved me and supported me unconditionally to pursue my goals.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

THEORY AND LITERATURE SURVEY

1.1 Processor Power Delivery Requirements

Ever since the first integrated circuit was developed, transistors have been growing exponentially ever smaller in agreement with Moore's law. As a direct consequence, computational power of microprocessors has been steadily increasing due to more transistors fitting into the same area. Until the mid 2000s, along with computational power, electrical power consumption of processors had also been steadily increasing (due to operation at increased clock frequencies). Increased power consumption implied increased heat dissipation per unit area, which became a serious limiting factor (processors running at more than 4 GHz were reported but never made it to the market because of low reliability). Power consumption of microprocessors has since then stagnated at around 100 W. However, to continue increasing computational power (the official term is *performance per watt*) of computers, multi-core processors started coming into the picture. Today we have high-end processors for PCs which have 8 or even more cores.

Power consumption and maximum clock frequencies of processors may have stagnated, but scaling of transistors has not stopped. As transistors keep getting smaller, the supply voltages keep decreasing. As a result, power supplies that power today's processors often have to supply more than a 100 A of current at extremely low voltages (as low as 0.7 V). So far the industry has been able to meet supply specifications using multiphase buck converters. However, as the core voltages continue to decrease, power supply designers are approaching a limit where multi-phasing buck converters or multiplying output capacitor count will simply not be able to meet the low-voltage high-current requirements of a microprocessor. An alternate series-stacked power delivery architecture to overcome this problem has been proposed before

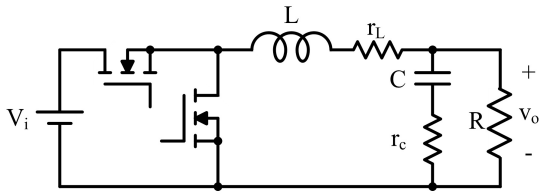


Figure 1.1: Buck converter with parasitic elements

[1]. Here, it has been shown how series-stacking processor cores has several advantages over the usual parallel connected core architecture. In this thesis we first consider the limitations of the parallel connected core architecture (in terms of its power supply requirements). Then we study the dynamics of a series-stacked architecture and develop a control for it, to improve its efficiency and transient performance.

1.2 Output Impedance of the Buck Converter

To design a power supply for a microprocessor load we have to take care that the output impedance of the supply is able to match the load. The load, i.e. the digital electronic circuit that forms the processor core, is usually considered to be resistive.[1]

As the voltage requirements of processors scale down while maintaining constant power consumption, the impedances of processor loads tend to go down by a factor of $\frac{1}{V_{DD}^2}$. To understand the corresponding impact on the design of power supplies let us take a look into a typical synchronous buck converter with parasitic impedances as shown in Figure 1.1.

The basic operation of the synchronous buck converter is simple [2]. The two switches alternately switch on and off to create a square wave at the

[1]Equation 1.1 gives the dependence of Power Consumed by a CMOS digital circuit with activity factor α, clock frequency f_{clk} and supply voltage V_{DD}. C_{dyn} is a fixed parameter dependent on the number of nodes in the CMOS circuit that undergoes switching during a particular clock cycle.

$$P_{dyn} = \alpha C_{dyn} f_{clk} V_{DD}^2 \tag{1.1}$$

The implication of this equation is that if clock frequency does not change during operation then ay CMOS digital circuit can be considered to be a resistive load (dependent on activity factor α) from a power supply designer's point of view. Modern processors modulate clock frequency as a function of activity factor so that processing does not slow down for heavy computational loads.

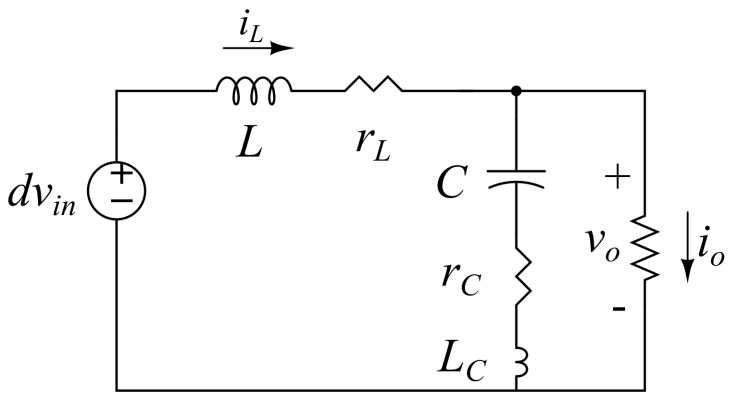


Figure 1.2: Buck converter averaged model

switching node N_{sw}. That square wave is filtered by the LC low-pass filter to provide a low-ripple dc voltage at the output terminal. The output voltage can be varied by varying the duty ratio of the square wave generated at the switching node. The output voltage generated if we assume lossless switches and inductor is given by

$$v_{out} = dv_{in} \tag{1.2}$$

This however only tells us about the static (or steady-state) behavior of the buck converter. A dynamical analysis has to be done to understand what the output voltage transients look like when there is a step (or ramp, etc.) change in output current or input voltage occurs. Averaged models are used to model the dc/dc converters as linear circuits [3]. One way to do the averaged modeling for the buck converter is to assume that the voltage at the switching node is dv_{in} (Figure 1.2).

If we neglect perturbations in the input voltage (i.e. $v_{in} = V_{in}$) it can be easily seen using the impedance divider expression that the transfer function of the output voltage with respect to duty ratio is

$$G_{vd} = \frac{v_o(s)}{d(s)} = \frac{V_{in}\left(1 + Cr_c s + L_c C s^2\right)}{1 + C\left(r_c + r_l\right)s + \left(L + L_c\right)Cs^2} \tag{1.3}$$

Now that we have a transfer function that defines the relation between output voltage and duty ratio variation we can do a closed loop control to regulate output voltage. Several types of controllers are used for closed loop control of buck converters. The most common among them is the 2-pole, 2-zero compensator combined with an integrator. The 2-pole, 2-zero

compensator is used to extend the bandwidth of the buck converter and stabilize it by improving its phase margin, while the integrator improves dc-gain to remove any steady state error. A typical design using L = 5 μH, C = 50 μF is shown in Figure 1.3.

Let us now try to find an expression for the output impedance of the buck converter. For simplicity first we find an expression for the open loop output impedance. Since it is open loop we can evaluate the output impedance simply by shorting the input ($d(s)v_{in}$ at the switching node) and observing the output voltage variation with current injected at the output terminal as shown in Figure 1.2. Following this method and neglecting the series inductance of the capacitor, the output impedance expression turns out to be:

$$Z_{out,ol} = Z_L || Z_C = \frac{(Ls + r_l)(1 + Cr_c s)}{1 + C(r_l + r_c)s + LCs^2} \tag{1.4}$$

We can use superposition to find an expression for the closed loop output impedance now. Assume that the feedback law is $dv_{in} = -H(s)v_o$ which is typical for a voltage mode controller with input voltage feed-forward (an alternative to current mode control for dealing with input voltage transients). With some algebraic manipulations the closed loop output impedance can be shown to be

$$Z_{out,cl} = \frac{Z_{out,ol}}{1 + G(s)H(s)} \tag{1.5}$$

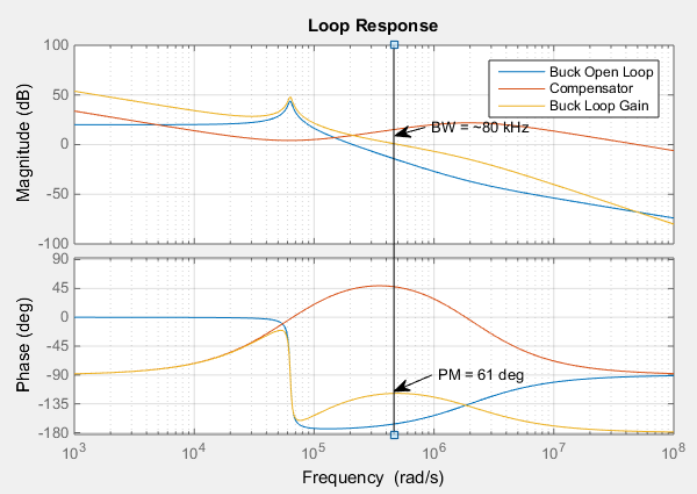


Figure 1.3: Compensation of buck converter

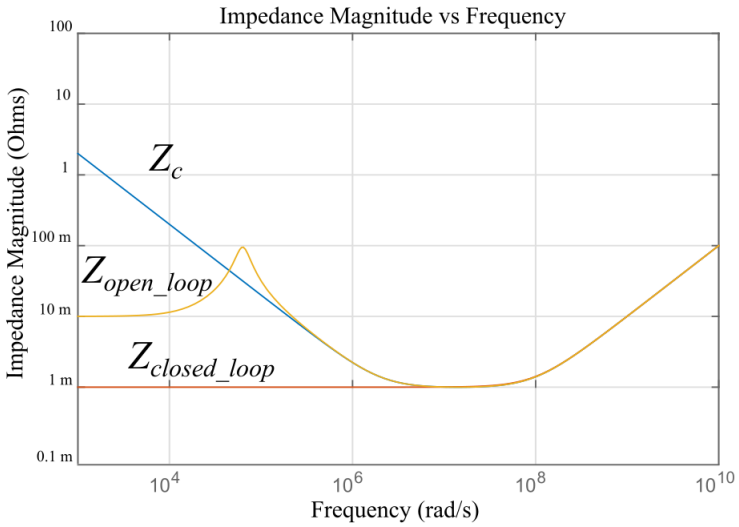


Figure 1.4: Open loop and closed loop output impedance of buck converter under droop control

In this expression the term in the denominator $G(s)H(s)$ is also called the loop-gain in control theory, and typically this term is what determines the bandwidth of the buck converter. At frequencies near and above the crossover frequency of the converter the output impedance of the converter can be approximated to be equal to the impedance of the paralleled capacitor network (Figure 1.4). Below the crossover frequency the output impedance is governed by the loop transfer function. Usually we see that below the crossover frequency, the output impedance decreases with decreasing frequency (with integral control), but controller design such that the output impedance remains constant below the crossover frequency is also considered desirable in certain cases [4], [5]. This type of design is also known as droop control and offers constant output impedance at frequencies lower than the crossover frequency. The magnitude of the constant output impedance determines the deviation of the output voltage from the reference voltage at steady state with load current. This type of control is known as droop control.

Our motive is to decrease closed loop output impedance of the converter without affecting its bandwidth (loop gain crossover frequency- ω_{co}). To accomplish that, first we have to make a few simplifying assumptions.

- Consider droop control, i.e. minimum output impedance in closed loop occurs at and below the crossover frequency. This simplifies calculations of output impedance. At the crossover frequency ω_{co}, the closed

loop output impedance equals the open-loop output impedance. So we only have to figure out a way to reduce output impedance at the crossover frequency.

- $\frac{1}{\sqrt{LC}} = k\omega_{co}$, i.e. the frequency of the double pole of the buck converter is a fixed fraction of the loop-bandwidth
- $\frac{\omega_{co}L}{r_l} = Q_L$, i.e. the quality factor of the inductor at the crossover frequency is fixed, and paralleling of the same component does not affect the quality factor
- $\frac{1}{\omega_{co}Cr_c} = Q_c$, i.e. the quality factor of the output capacitor is fixed at the crossover frequency

Manipulating the open-loop output impedance expression at the crossover frequency using these simplifying constraints,

$$|Z_{out,ol}| = \frac{1}{C\omega_{co}} \sqrt{\frac{\left(1+\frac{1}{Q_L^2}\right)\left(1+\frac{1}{Q_c^2}\right)}{\left(1-k^2\right)^2 + \left(\frac{k^2}{Q_c}+\frac{1}{Q_L}\right)^2}} \tag{1.6}$$

So from here we see that the output impedance of a buck converter scales with output capacitance as $\frac{1}{C}$. We have already seen before that the output impedance requirement scales with output voltage as $\frac{1}{V_{DD}^2}$. So projecting, the output capacitance requirement of a buck converter supplying a processor load will increase with decreasing output voltage; i.e., theoretically if the processor voltage scales down from V_{DD} to $\frac{1}{2}V_{DD}$, then the output capacitance of the buck converter has to be increased to 4 times its original value.

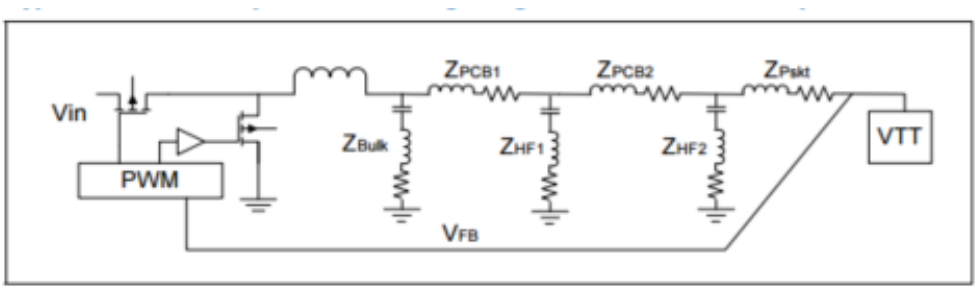


Figure 1.5: Typical buck converter output filter structure with parasitic components [5]

This however is a very optimistic evaluation. In effect an actual power supply supplying the processor load looks like the circuit shown in Figure 1.5. The buck converter is directly connected to some bulk capacitance (usually OSCON type capacitors having relatively high series resistance and inductance compared to ceramic capacitors). The power supply unit is connected to the microprocessor board where ceramic decoupling capacitors (Z_{HF1}) are provided to stabilize any instability present in the output of the power supply unit. We have to keep in mind that the very high currents and extremely low voltages make the drops across the PCB path parasitics Z_{PCB1}, Z_{PCB2} very real issues. After encountering the decoupling capacitors, the power goes to the separate supply voltage pins of the microprocessor. Each of the supply pins (or small clusters of supply voltage pins) has separate ceramic capacitors connected very close to it. These ceramic capacitors (called cavity capacitors) form the final filtering/stabilizing stage (Z_{HF2}). The impedance Z_{Pskt} is the effective impedance between the cavity capacitors and the supply pins of the microprocessors. To reduce the impact of this impedance, today's processors often have half or more of their total pin count dedicated to supply and grounding. The large degree of paralleling due to this helps keep the socket impedance in check.

Any practical capacitor will have parasitic resistances and inductances (effective series resistance and inductance) and are modeled as a series connection of a resistor, capacitor and inductor. So in essence a capacitor will behave like a capacitor only below the resonant frequency of the LCR series circuit. Above that, it will behave like an inductor. Paralleling of the same types of capacitors will not change the resonant frequency. It will only reduce the values of impedances at all frequencies. The frequency at which a capacitor or combination of capacitors start behaving like an inductor is known as the breakaway point.

OSCON capacitors are very energy dense in the sense that they can have large capacitances packed into small areas. But they also have significantly large ESR and ESL compared to ceramics. Therefore, OSCON capacitors tend to breakaway (start behaving like inductors) at significantly lower frequencies than ceramic capacitors (at least one decade below ceramics). One possible way obtain large capacitances and still maintain a high breakaway frequency is to connect ceramic capacitors in parallel with OSCON capacitors. (Special care has to be taken to set the ratio of the OSCON capacitor

to the ceramic capacitor correctly. Otherwise the paralleling may not prove to be effective at all.)

The effective impedances of a particular combination of bulk, decoupling and socket capacitors are plotted in Figure 1.6. The capacitor network impedance with the series parasitic elements (Z_{PCB1}, Z_{PCB2}, Z_{Pskt}) is shown in Figure 1.6 as well.

The effect of the series PCB impedance components is to offset the capacitor impedance curves to higher minimum values by adding double zeroes at multiple resonant frequencies near the breakaway frequencies of each types of capacitors (Figure 1.6). To compensate for this increase in output impedance, capacitor values have to be increased further. The impact of these parasitic series impedances is crippling. As more decoupling capacitors are added in parallel to each other in order to reduce the ESR, the effective capacitance network moves closer to a distributed model which can be simplistically represented as in Figure 1.7

It can be shown analytically that the impedance of the network shown in Figure 1.7 approaches a minimum value of $\sqrt{\frac{L}{C}}$. So in effect output impedance z_{out} of the power supply line scales as $\frac{1}{\sqrt{C}}$ instead of $\frac{1}{C}$. This

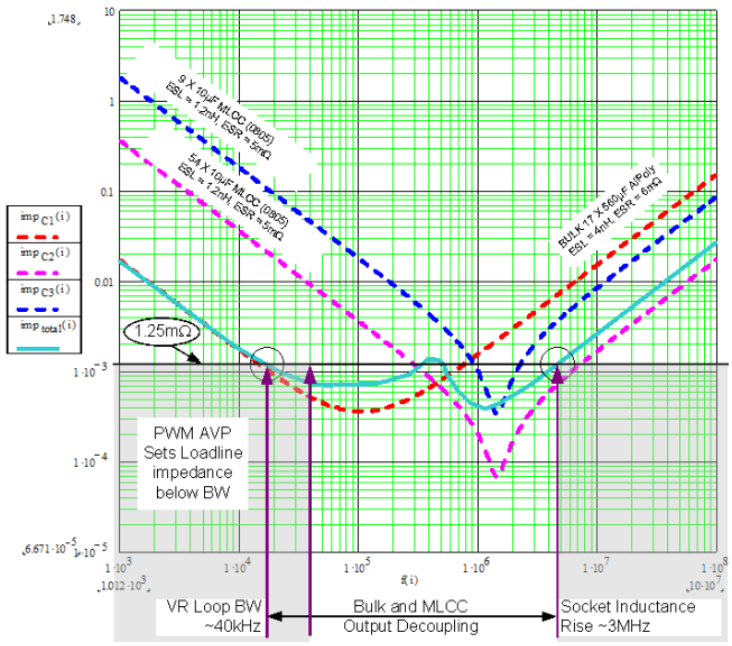


Figure 1.6: Typical impedance of a composite system of capacitors (bulk electrolytic, MLCC decoupling and MLCC cavity capacitors) - [5]

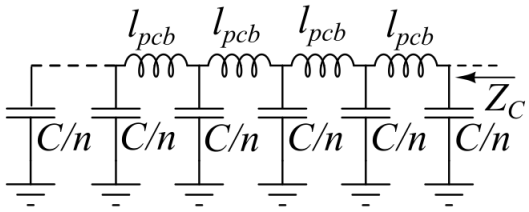


Figure 1.7: A distributed model approximation for evaluating capacitor impedance in 1.6

implies that the output capacitance requirement varies with supply voltage as $C \propto \frac{1}{V_{DD}^4}$. Empirically the capacitance requirement varies with output voltage as a function of $\frac{1}{V_{DD}^5}$ instead of $\frac{1}{V_{DD}^4}$ due to capacitance derating effects at lower voltages.[2]

The point of this exercise was to show that supplying microprocessor loads with buck converters as supply voltages decrease (keeping power constant) becomes impractical. The PCB impedances pose a limiting value to the obtainable output impedance from a power supply and after some point increasing output capacitances (or even decreasing the impedance of the inductive network Z_L by multi-phasing techniques, which we will see later in this chapter) will simply not be able to handle the output impedance requirements. As we have pushed towards lower supply voltages (supply at sub-threshold voltages less than 0.4 V may also become a possibility in the future) the number of supply pins has increased drastically (in order to reduce effective PCB impedance by achieving a high degree of paralleling).

To counter the PCB impedance effects at low voltages and high currents, most microprocessor manufacturers have now started to integrate buck converters inside the chip itself. This way the processor can be supplied at a relatively high voltage (hence reducing the output impedance requirement of the converter that supplies power to it) and the core voltage can be internally generated by high-frequency integrated buck converters. The logic behind integrating the buck converters on chip is that the output impedance requirement of the integrated converters is significantly lower because there are multiple converters that supply different cores. This architecture of power

[2] $C \propto \frac{1}{V_{DD}^5}$ is an overly pessimistic estimation which is approached when an impractical number of decoupling capacitors are paralleled in an attempt to reduce the effective ESR of the network. The actual factor at which capacitance requirement decreases with supply voltage is somewhere between $\frac{1}{V_{DD}^5}$ and $\frac{1}{V_{DD}^3}$

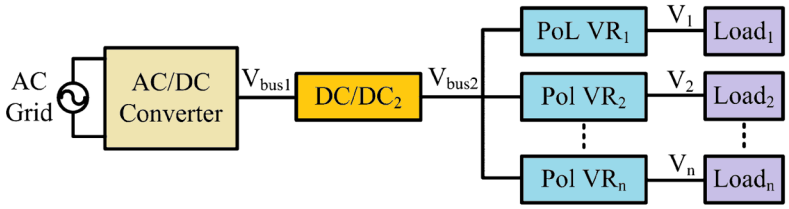


Figure 1.8: Intermediate bus architecture (IBA)

delivery is known as the intermediate bus architecture (IBA), shown in Figure 1.8.

Several other advantages of this IBA power delivery are evident. It has been shown that system-level energy/efficiency optimization is possible by regulating the core voltage as a function of activity factor of each core. If a certain core is operating at a low activity factor then its supply voltage can be reduced and operated at a lower frequency than other cores. This reduces the power consumption of that core without compromising throughput. This is called dynamic voltage scaling (DVS). Modern microprocessors use DVS very frequently. Processors usually identify the activity factor and appropriately adjust their clock frequency (low clock frequency when activity factor is low and vice versa) to maintain a constant throughput. Then they send out voltage identification (VID) bits to the voltage regulator modules to appropriately increase or decrease the supply voltage. With IBA power delivery the voltage levels of different cores can be adjusted independently of each other. This has been proven to be very useful for power consumption optimization. However, efficiency of integrated converters tends to be low because of high frequency switching. So, while we are saving power by DVS we are also losing power because of lower overall efficiency. So the benefits of this architecture are limited.

What we should take away from this development is that supplying power to the chip at higher voltages is always more desirable than supplying lower voltages (and correspondingly higher current). This motivates us to think about the possible gains that we may see if we connect processor cores in series instead of in parallel as is the usual practice. We will discuss the possibilities of series connection in more detail in section 1.3.

Now, let us take a look into the losses in buck converters. Consider the same simple model of the buck converter as in Figure 1.1. Conduction losses

during steady state operation are incurred mainly in the inductor and the two switches. Neglecting inductor current ripple,

$$P_{cond} =< i_l >^2 (r_L + r_{sw}) \tag{1.7}$$

Usually as the voltage scales down and current scales up, to avoid inductor saturation phases are added in the buck converter (Figure 1.9)

As a consequence of paralleling several buck converters, the effective r_L and r_{sw} scale down linearly with voltage (increase in current output implies more phases in the multi-phase buck converter). But since the conduction losses are dependent on i_L^2, the conduction losses in the converter increase as a function of $\frac{1}{v_o}$. The switching losses in the converter increase linearly with the number of phases because of additional switches. That is also a linear function of $\frac{1}{v_o}$, so losses in the converter increase as a function of $\frac{1}{v_o}$. Therefore, the efficiency of power converters will decrease with decreasing output voltage. However, better switching devices and inductor core materials are becoming readily available day by day and as such decreasing efficiency is not as big a limitation as the capacitor requirement for down-scaling of output voltage. Multi-phasing of buck converters also has a few other significant advantages that will be discussed in the next section.

Now suppose we have multiple PoL converters supplying different cores from the output of the multi-phase buck as seen in the IBA. The net efficiency of this combination will mainly be governed by the most inefficient of the power conversion stages (usually the PoL conversion stage because of its high frequency). We will see later in section 1.4 why the series connection of cores has a clear advantage here.

1.3 Transient Response Improvement, Efficiency Improvement

With processors now incorporating various power saving modes and voltage scaling, output impedance is not the only area requiring improvement for power supplies. According to the latest Intel VRM specifications [5] a power supply designer can now expect step load changes from 0 to full load current at a repetition rate of 50 kHz (Figure 1.10).

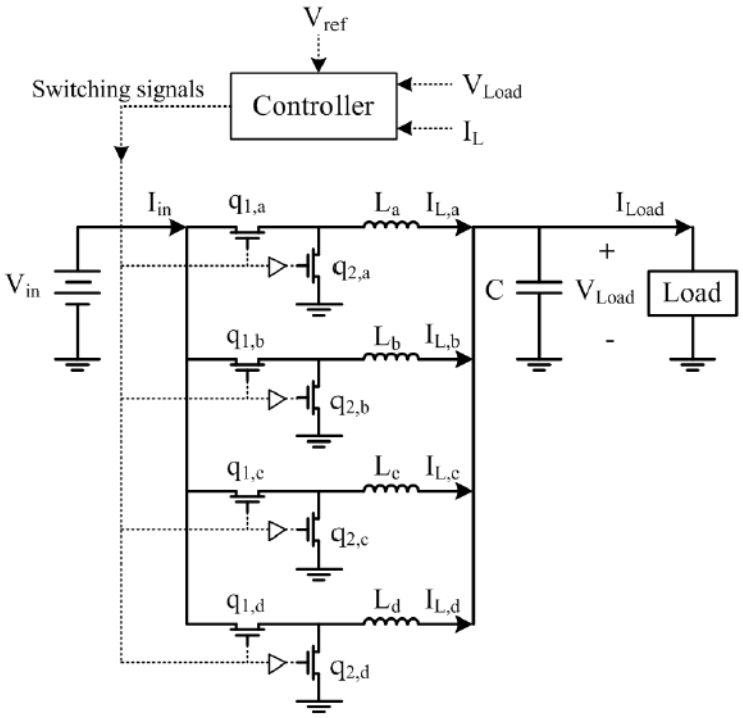


Figure 1.9: Multi-phase buck

The implication here is that the output voltage should stabilize within the very short periods (20 μs for a transition between 10% to 100% of full load) as specified by the repetition rates in the plot. The settling time of a buck converter is governed by the closed-loop bandwidth of the voltage regulator. Usually the obtainable closed-loop bandwidth for buck converters using a 2-pole, 2-zero compensator is a fraction (1/5 or less) of the switching frequency. The best transient response (bandwidth) that linear control can provide has been effectively studied by many researchers. One such instance is [6], although parameter variation plays a big role as we try to increase bandwidth using this approach. Also estimating/sensing capacitor current and utilizing it in feedback control have proven to be effective in improving bandwidth [7].

Linear control of power supplies in general does not give us optimal response times for transients and designers often end up overusing capacitors to keep transient voltages within limits. Time-optimal response of a buck converter (or bang-bang control) has been studied and targeted for many years. The effective bandwidth achieved with time-optimal response is nearly

equal to the switching frequency of the converter while the bandwidth usually obtained using linear control is only a fraction of the switching frequency. Current Mode control by using carefully matched or adaptive ramp compensation can provide near optimal results. Nonlinear control methods have been tried out successfully and are very useful as they provide uniform and predictable transient response to all kinds of load steps. A geometric method for obtaining near null (optimal) and null response (i.e. total negation of output voltage transient with load current stepping) has also been tested in the past [8], [9]. The latter method providing null response requires augmentation that makes the converter inefficient during transients. In a converter that is supplying fast varying loads such as microprocessors these are not very practical to implement.

Improving the efficiency of microprocessor power supplies has been studied mainly in the form of improving light-load efficiency. Microprocessor power supplies are almost exclusively made of multiphase buck converters (Figure 1.9). While all the phases available need to be operational at high current load to avoid inductor saturation, at lighter loads the multiple phases only cause increased switching losses. To improve light load efficiency, usually phases are shut down (phase-shedding) at lighter loads. Switching signals to the n phases of the multiphase buck converter can be interleaved so that the current ripples in the individual phases are complemented and lowered (or even eliminated) when they add up at the output capacitor. This helps in significantly lowering the switching noise in the output of the buck converter.

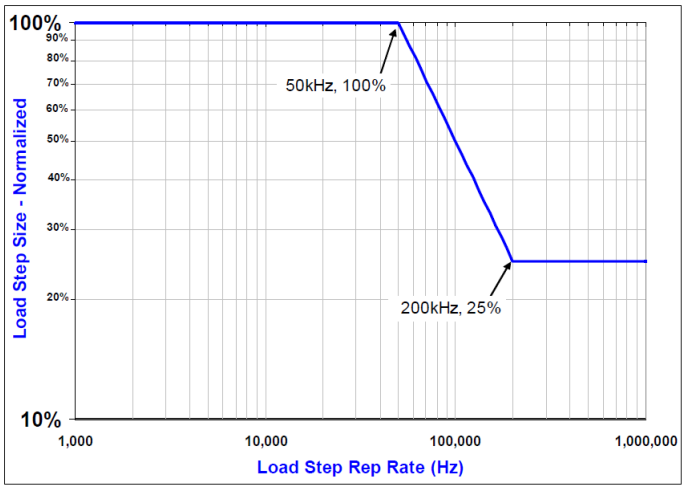


Figure 1.10: Transient repetition rate [5]

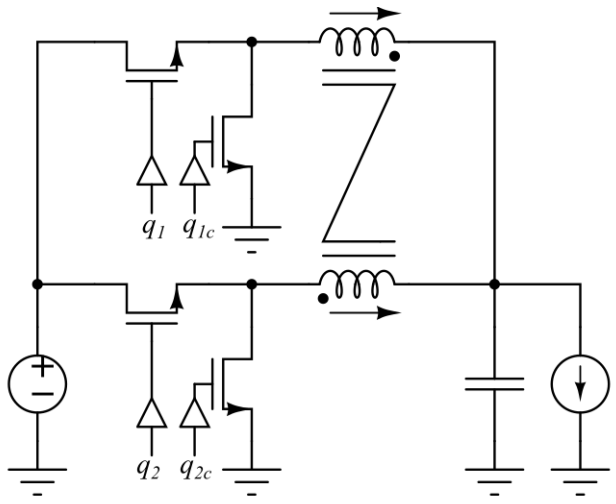


Figure 1.11: Coupled inductor 2-phase buck converter

At lighter loads, when multiple phases have been shed the output switching voltage noise increases, which may or may not be tolerated by the VRM restrictions [10]. Transient behavior due to shedding of a phase also has to be taken into account. Another slightly different and more effective approach to light load efficiency improvement in multiphase buck converters has been studied in [11]. However, this variation is more suitable for lower power applications where current sharing is not the primary reason for using a multiphase converter. A similar technique more suitable for higher power has been proposed in [12].

Improving transient response of power supplies has proven to be significantly more challenging with increasing step load requirements. The path to improving settling times of buck converters in general is to increase the number of phases so that individual phases have to handle smaller step currents. Decreasing the inductance value to improve settling times of converters is not a good solution since the inductors will saturate at lower average currents. An interesting solution to this problem was developed by coupling the output inductances as shown in Figure 1.11. Initially proposed for two phase converters with mutually coupled inductances [13], the coupling action enables the inductor current to slew at a rate inversely proportional to the leakage inductance of the coupled inductors (when the two phases are not interleaved) while rejecting current ripple proportionally to its magnetizing inductance when the phases are separated by 180^0. A number n of these two phase coupled units can be paralleled and interleaved to form a $2n$ phase

solution for a modern VRM. The idea has been extended to more than two phase converters by use of multi-phase coupled inductors with ladder type cores and has been shown to provide exceptional transient performance [14]. This modified multiphase buck converter is less prone to inductor current mismatches and the inductor also saturates at a higher average current due to lower ripple. Also, increased ripple rejection suggests that the switching frequency of the phases can be reduced to an large extent to improve efficiency substantially as has been mentioned in [14].

1.4 Series Connected Power Delivery and DPP

From the discussion on output impedance of buck converters, we can infer that maintaining constant power delivery at lower and lower voltages becomes exponentially difficult owing mainly to PCB impedances and parasitic inductances and resistances of capacitors. As supply voltages are expected to fall lower (even sub-threshold operation at supply voltages as low as 0.4 V is becoming more probable in the near future) we expect to see output impedance of converters to approach an asymptotic limit when increasing capacitance and number of supply pins may still not be enough to cope with the output impedance requirements of new processors. However, as supply voltages decrease and as we become able to multiply transistor count in upcoming processors, we have seen an increase in core count of processors and we can expect to see processors with more cores (16 or more) in the future. Since clock speeds have reached a certain limit, increasing core count is now the way to improve processing power (performance per watt) of processors. This opens up the possibility of connecting processor cores in series rather than in parallel.

Since we want to independently control the supply voltages of each core, we have to use additional power converters that supply the difference in currents between adjacent cores at each node (Figure 1.12). One may argue that the total converter count remains the same as the IBA architecture of Figure 1.8, but a closer look explains why the series architecture is better than the IBA architecture in terms of power delivery efficiency. The intermediate dc-dc buck converters of both the architectures have nearly the same efficiencies. And the PoL converters that regulate the core voltages also have

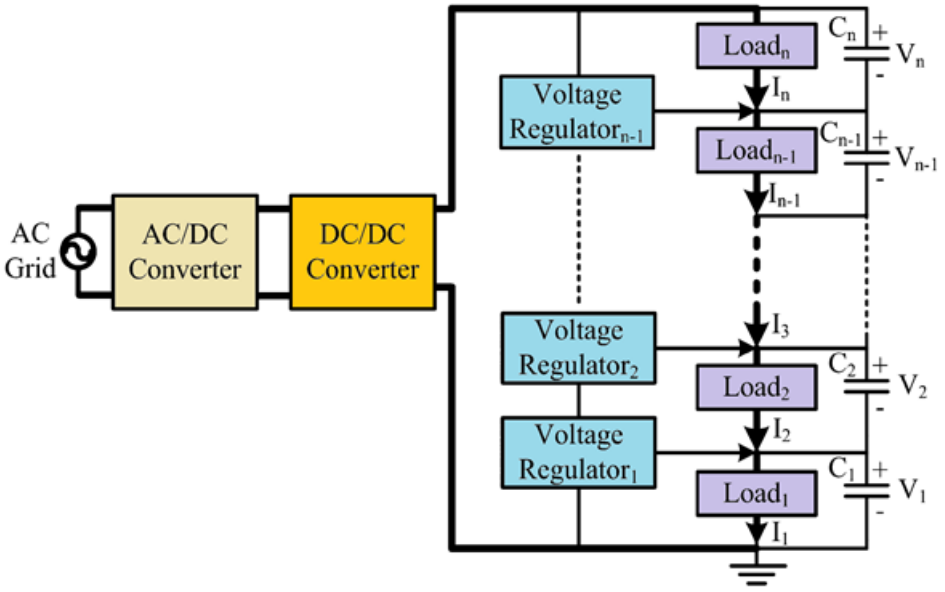


Figure 1.12: Series-connected power delivery

the same efficiency, but the PoL converters that regulate the stack voltages only process a small fraction of the net power of the CPU in the case of the series architecture. In the IBA architecture the PoL converters process the entire power of the CPU as demonstrated in Figure 1.13.

CPU core activity balancing has been a software focus of major processor and operating system researchers since the advent of multicore CPUs. If we take advantage of that fact and assume that at any time the PoL converters are not going to have to process more than x percent of the net processor power, then the minimum efficiency of the power delivery will be given by $\eta_{DPP} = \frac{\eta_{rect}\eta_{mpb}\eta_{int}}{x+(1-x)\eta_{int}}$. Compared to the IBA, whose efficiency can be written as $\eta_{IBA} = \eta_{rect}\eta_{mpb}\eta_{int}$, this is a very significant improvement. This expression validates the fact that if the activity factors of all the cores are matched exactly, then the relatively lower efficiency of the final PoL stage will not be a factor in the net power delivery efficiency of the system.

Our original motivation, managing output impedance without having to increase capacitances unreasonably, is also addressed under certain reasonable assumptions. The capacitance requirements at each node are mainly defined by the current consumption of each core. Under the assumption that core count increases proportionally with decrease in voltage, we see that net capacitor requirement will also increase proportionally ($\frac{1}{V_{DD}}$) with decreas-

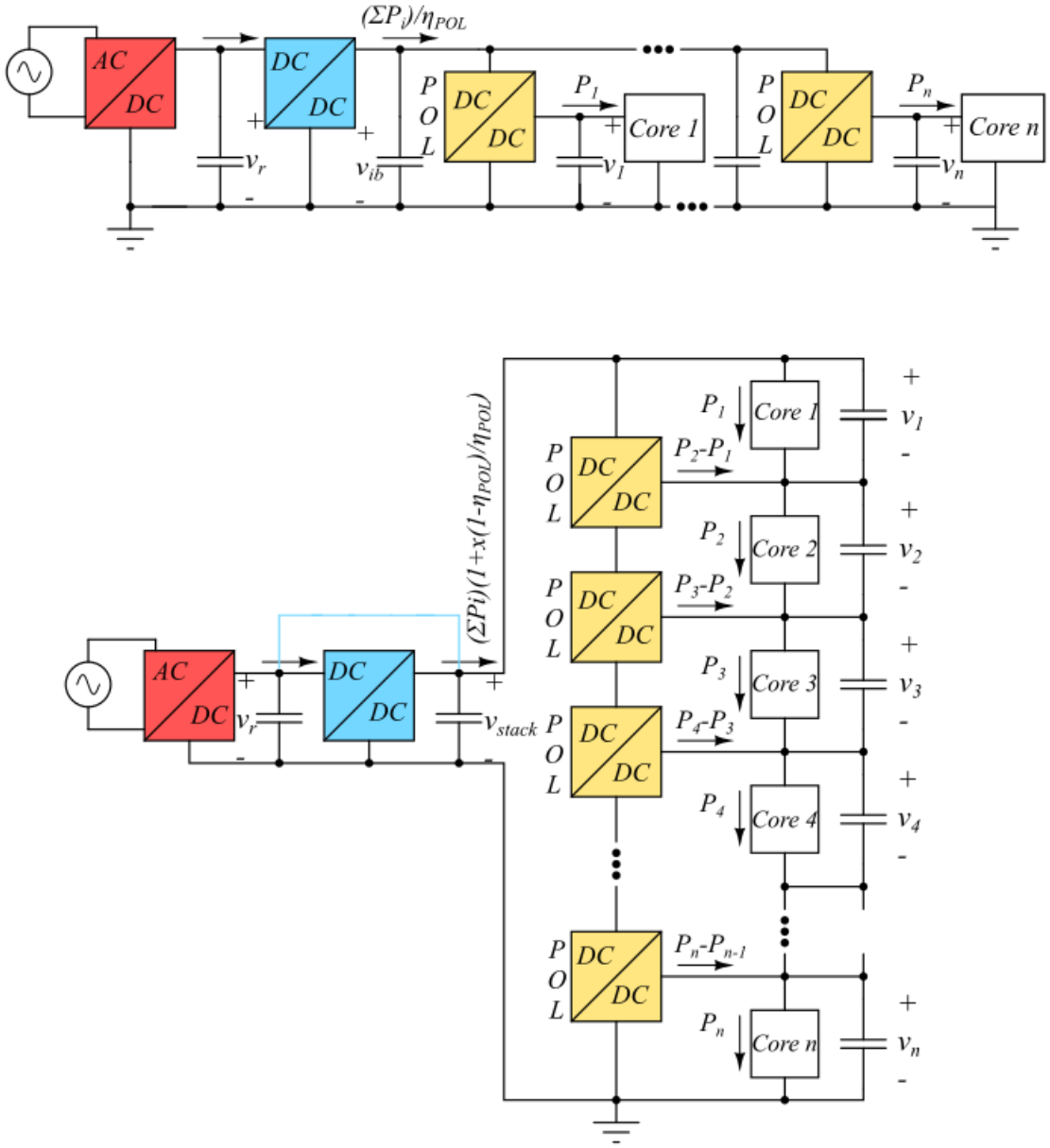


Figure 1.13: IBA vs. series-connected power delivery: The architecture shown at the top is the integrated bus architecture (IBA) and the lower one is a high level diagram of the series-connected power delivery architecture.

ing voltage unlike the discussion in section 1.2 where capacitance requirement was shown to increase at least as a function of $\frac{1}{V_{DD}^2}$. An additional advantage of using series connected power delivery is that dynamic voltage scaling of processor cores is implicitly possible.

Series connection or stacking is not a new concept. Differential power processing (DPP) is an efficient way to implement series stacking and also has been studied in the past under different names (partial power processing, charge recycling etc.). Series connection of solar panels using DC optimizers [15] is a popular solution for MPPT tracking for a series connection of a stack of photovoltaic panels. The series-connected solution with DPP [16] however, has been shown to be more efficient than the DC optimizer solution. The concept of DPP was also used for voltage balancing of battery systems. An example of this is the series connected battery charger proposed by Brainard [17]. Here a cascade of inverting buck-boost converters were used to equalize the voltages of a series battery string while charging the batteries. This circuit is the basis of the circuit we later use as voltage regulators for processor core voltage regulation. A switched capacitor method for charge balancing of a series connected string of batteries, which also relates well with the concept of DPP, was proposed by Pascual and Krein [18].

Series connected power delivery for digital loads is a much more recent concept and so far it has shown a lot of promise. The main applications have come up in the area of power delivery to data centers. In one application [19], the load balancing has been done entirely in software so that the external converters that process the differential powers between adjacent servers are not needed at all. Although this is what we should be targeting, it is somewhat unrealistic in general. Distributing the huge amount of computational load equally among all servers itself is a significant challenge. However, we still can achieve relatively small mismatches by following a less stringent computational load distribution and allowing the differential power processing converters to process a small amount of power. The results obtained here [20] confirm that extreme efficiencies can be achieved by this series power delivery scheme.

Compared to series-connected power delivery for server systems, research on series power delivery in processor cores has been much more limited due to the apparent difficulty in setting up test systems. Deciding on the level (motherboard level or digital circuit level) at which series stacking is to be

done is a challenge since processors are not the only components on the board and the voltage levels of all units have to be compatible. An early result using multipliers as digital loads and low dropout regulators (LDOs) as the DPP voltage regulators has shown that series power delivery is indeed possible and can meet necessary voltage regulation specifications [21]. As seen here, LDOs seem to be a good choice for the DPP converters because of their excellent bandwidth and input-to-output noise rejection but they tend to be inefficient. Typically if the supply voltage of each core is to be equalized then the LDOs will be only 50% efficient (ideally). This makes LDOs not suitable for high power processor cores where small relative computational mismatches can lead to significant amounts of injected currents at each node. A more efficient solution can be achieved by replacing LDOs with switch-mode power converters. As always there is an efficiency-bandwidth trade-off here. In [22] several architectures have been proposed that can be used to successfully achieve voltage regulation of a series stack of processor cores.

A more recent development in this area is the use of switched capacitor converters for voltage regulation [23]. Since the power processing requirement of the DPP converters is quite small, switched capacitor converters can be easily used for voltage regulation of the intermediate nodes. Improving power processing capability and efficiency of power conversion by using soft charging [24] can serve to improve both the power processing capacity and efficiency of power delivery. However, switched capacitor power delivery only proves effective if voltage equalization is our goal (and voltage droop is acceptable). More research has to done on switched capacitor DPP topologies before the advantages of voltage scaling are not compromised when using switched capacitor circuits for stack voltage regulation.

CHAPTER 2

DPP FOR PROCESSOR POWER DELIVERY

2.1 Selection of Topology

The general structure for series-connected power delivery has been shown in Figure 1.12. However, there are several different topologies that can be used for actually implementing the voltage regulators that process differential current between the different loads. All these topologies are fundamentally different in the way they transfer power to the load. Selecting the topology for a particular series-connected application depends on which parameter we are looking to optimize. First we look into the bus-to-element connected topology. Then we will look into the element-to-element topology which is less intuitive compared to the former. After that we will discuss the motivation for selection of the element-to-element topology for our processor core voltage regulation application.

2.1.1 The Bus-to-Element topology

The bus-to-element architecture transfers energy between the series elements and the main bus. A virtual bus or other storage element could also be used instead of the main bus. Various topologies and control strategies can be implemented with this architecture. The main benefit of this architecture is in the independence of converter states. A disadvantage is that the components have to be rated for the entire bus voltage. It is possible to have n or $n-1$ bus-to-element differential converters depending on converter topology and system objectives. One implementation of the bus-to-element architecture is shown in Figure 2.1. The average current provided by the differential converters can be determined by applying KCL at the intermediate voltage nodes.

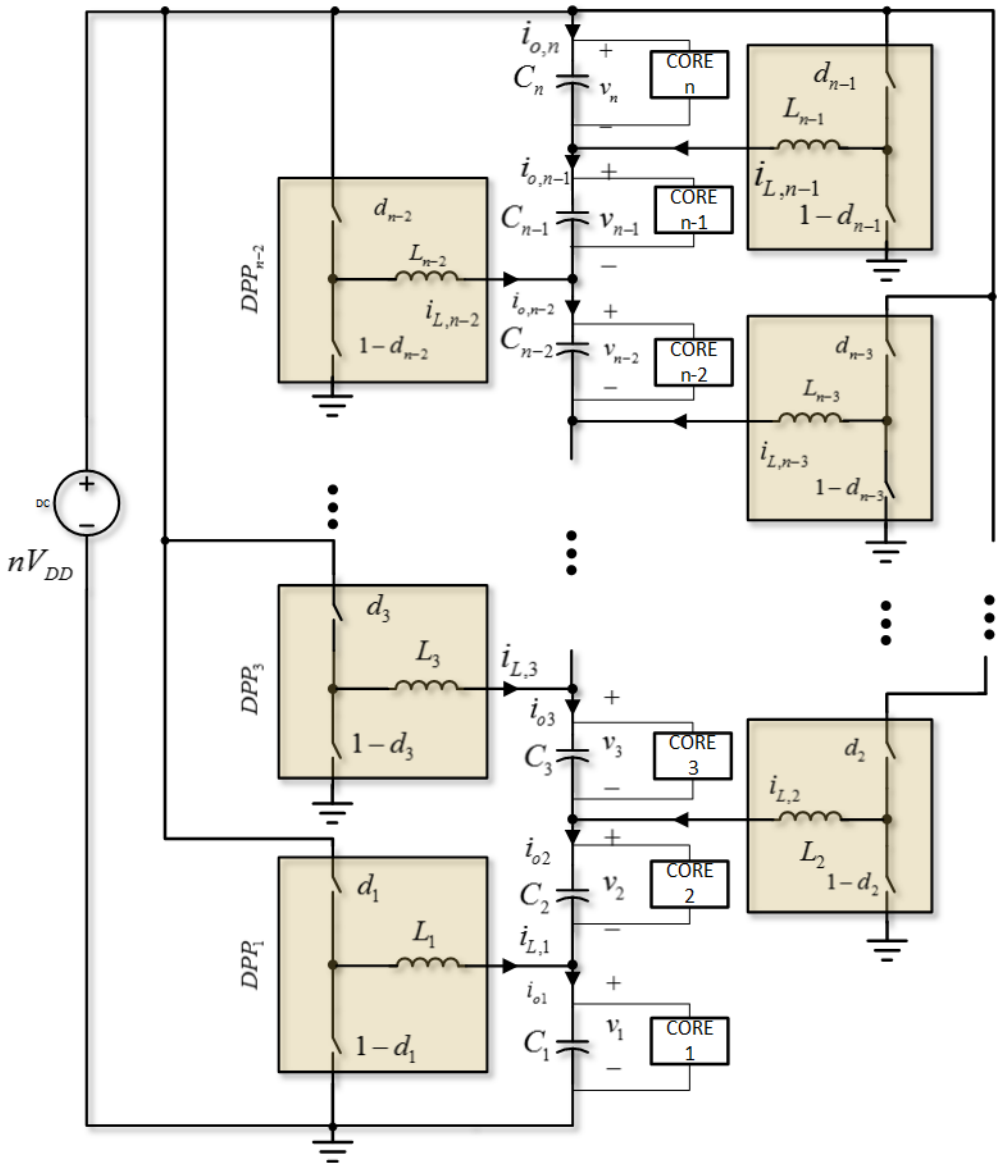


Figure 2.1: The bus-to-element DPP topology

The relation between the domain voltages and the duty ratios of the converters can be evaluated as

$$\begin{aligned}
D_1 V_{stack} &= V_1 \\
D_2 V_{stack} &= V_1 + V_2 \\
D_3 V_{stack} &= V_1 + V_2 + V_3 \\
&\vdots \\
D_{n-1} V_{stack} &= V_1 + V_2 + V_3 + \cdots + V_{n-1}
\end{aligned}$$

where

$$V_{stack} = V_1 + V_2 + V_3 + \cdots + V_n \tag{2.1}$$

With the bus-to-element converters, the steady state current equation for series load elements is simply

$$I_{L,k} = I_{o,k} - I_{o,k+1} \tag{2.2}$$

As we see the duty ratios of the individual converters in general decrease from high values at the top to low values at the bottom of the stack. If the stack has a large number of elements, the converters have to be designed with different values of inductances and output capacitances to maintain the same bandwidth at each node. Component sizing to ensure the same bandwidth of each converter becomes a problem as the number of elements in the stack increases.

To get around this particular disadvantage (i.e. to improve modularity and hence increase scalability) two particular modifications to this topology have been suggested. A very promising architecture uses n flyback converters (Figure 2.2), as suggested in [25]. Another similar topology that is derived from this topology is the virtual bus-to-element topology shown in Figure 2.3 [26]. Both of these topologies use n balancing converters as opposed to $n-1$, which may suggest that efficiency is compromised with respect to the original bus-to-load topology. However, a careful analysis done in [25] and [26] proves that the converters themselves are prone to process less differential power in the two isolated topologies.

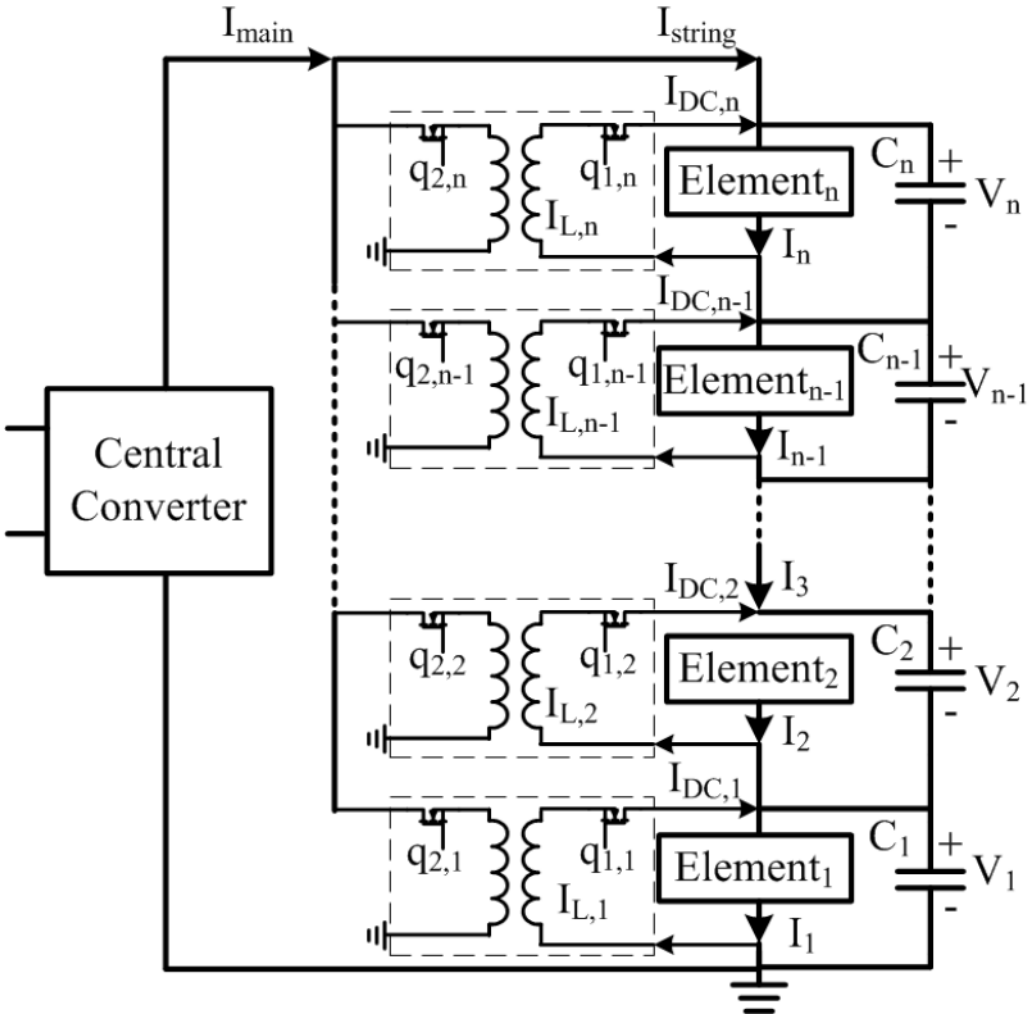


Figure 2.2: Series connected bus-to-element DPP using n isolated flyback converters [25]

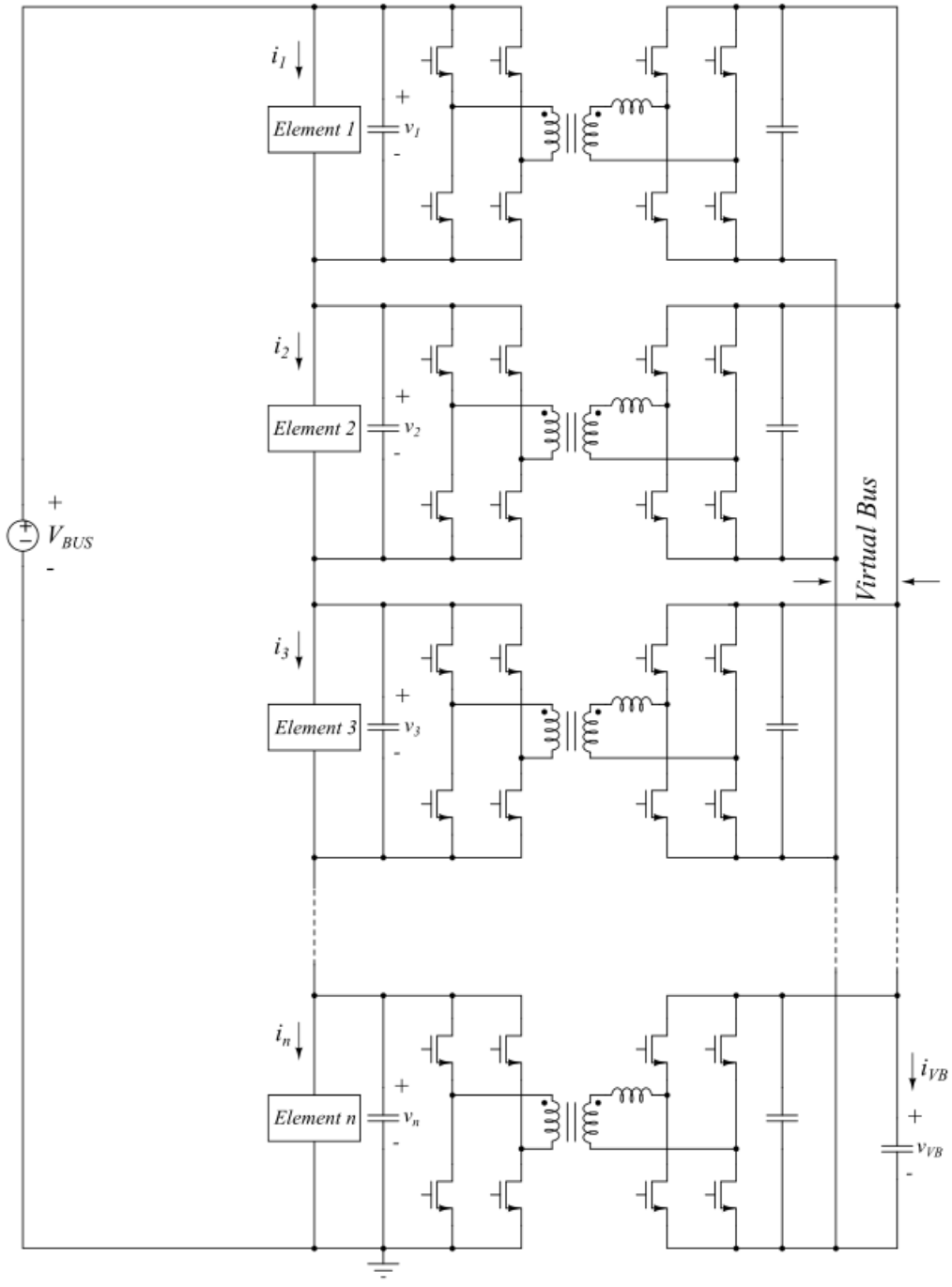


Figure 2.3: Virtual bus-to-element topology using full bridge isolated converters as DPP converters

2.2 The Element-to-Element Topology

A slightly different topology for voltage regulation of a series stack is the element-to-element topology shown in Figure 2.4.

The element-to-element architecture focuses on local converters that transfer energy between neighboring nodes. This architecture shown in Figure 2.4 has its differential converters in a buck-boost topology. The advantage of this approach is that the differential converters can be locally controlled and component voltage ratings can be comparatively low. The drawback is that the converter states are not independent. The inductor current of one DPP converter is dependent on the inductor currents flowing in the adjacent converters. By applying KCL at each node The steady state inductor currents and mismatch currents are observed to be related by

$$\begin{bmatrix} 1 & -(1-D_2) & 0 & \dots & 0 & 0 \\ -D_1 & 1 & -(1-D_3) & \ddots & \vdots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 & 0 \\ \vdots & \ddots & -D_{n-3} & 1 & -(1-D_{n-1}) & 0 \\ 0 & \dots & 0 & -D_{n-2} & 1 & 0 \\ 0 & \dots & \dots & 0 & -D_{n-1} & 1 \end{bmatrix} \begin{bmatrix} I_{L,1} \\ I_{L,2} \\ \vdots \\ I_{L,n-2} \\ I_{L,n-1} \\ I_n \end{bmatrix} = \begin{bmatrix} I_1 - I_2 \\ I_2 - I_3 \\ \vdots \\ I_{n-2} - I_{n-1} \\ I_{n-1} - I_n \\ I_n \end{bmatrix} \tag{2.3}$$

The converters can be interchangeably considered to behave as buck converters (stepping down $v_{k+1} + v_k$ to v_k) or boost converters (converse of the buck) or buck-boost converters (converting v_{k+1} to v_k). The relation between the duty ratios and the domain voltages can also be expressed as follows:

$$\begin{aligned} \frac{v_1}{v_2} &= \frac{D_1}{1-D_1} \\ \frac{v_2}{v_3} &= \frac{D_2}{1-D_2} \\ \frac{v_3}{v_4} &= \frac{D_3}{1-D_3} \\ &\vdots \\ \frac{v_{n-1}}{v_n} &= \frac{D_{n-1}}{1-D_{n-1}} \end{aligned}$$

Although the coupling of inductor currents poses a challenge in implement-

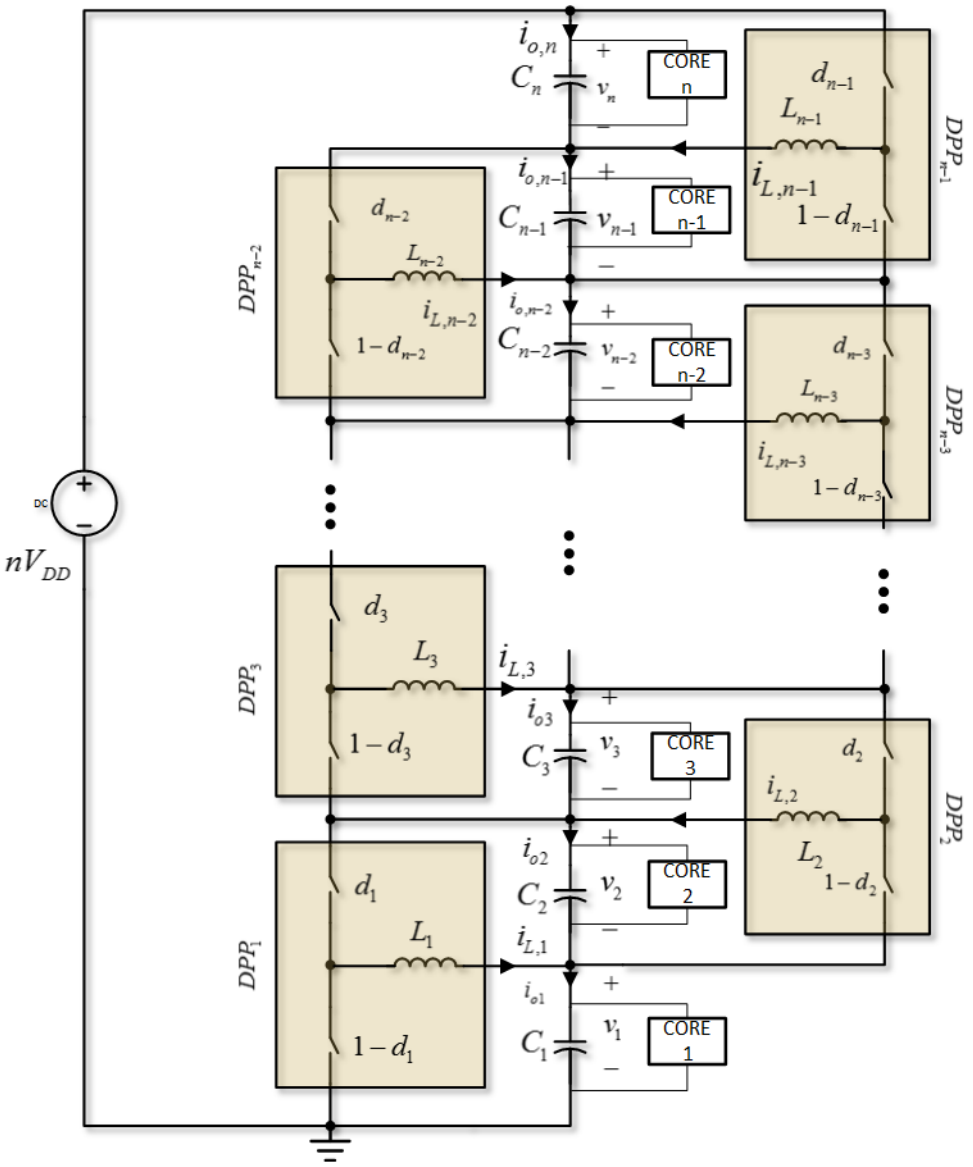


Figure 2.4: Element-to-element topology

ing closed loop control (as we are going to see later on in this chapter) the lower voltage ratings of the converter switches give this topology an advantage over the bus-to-element topology. The lower voltage ratings enable us to switch at high frequencies, lowering the inductor requirement for each DPP converter and improving bandwidth of the DPPs.

As we have mentioned before, selection of topology is largely application dependent. For example differential power processing for optimizing energy output of a series-connected array of solar panels calls for the element-to-element topology. The wide area over which panels are distributed makes it impractical to use the bus-to-element topology. Topology selection also depends on the level at which we want to do differential power processing. If we are connecting a rack of server loads in series, the power supply and the differential voltage regulation circuitry cannot be limited to a single board. Also, the comparatively large load and voltages suggest that we have to use a topology in which the DPPs process minimum power for a given degree of mismatch. A comparative analysis of the power processed by different topologies given a limit on the amount of mismatches between consecutive elements of the stack has been done in [25]. This shows that the isolated bus-to-load topology has a clear advantage in the amount of power processed. So isolated bus-to-load (or virtual bus-to-load) architectures find their application in rack level voltage regulation by DPP. For board-level applications like ours, the motive is to achieve fast transient responses and maintain very stringent voltage regulation. High-frequency on-chip DPP converters would be an ideal solution for our case. The element-to-element topology with buck-boost converters is a good choice for board-level DPP because of lower switch voltages. A switched capacitor equivalent of the element-to-element topology is also possible. Figure 2.5 shows a switched capacitor circuit that can be used for voltage balancing of a series stack. This topology is a variant of the series stacked battery charge balancing circuit proposed in [27] and shares the advantages of the element-to-element topology with buck-boost converters in terms of modularity and scalability. However, it is only suitable for low-power processor stacks because of its voltage droop characteristics (or cases where droop is a requirement). Further research is needed before the switched capacitor topology becomes an attractive solution for regulating supply voltages of a series stack of processor cores. A successful attempt to improve power density and efficiency of the switched capacitor

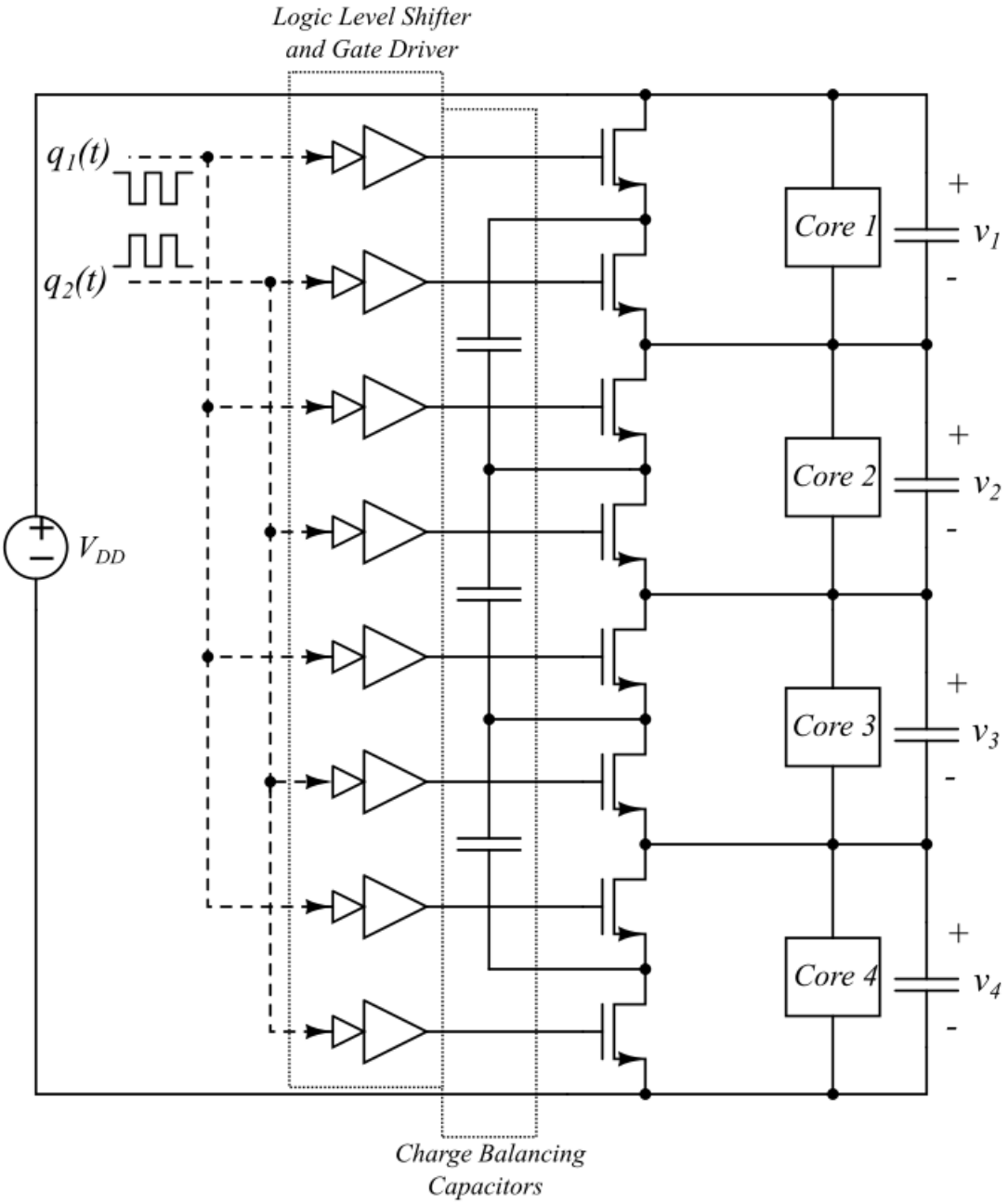


Figure 2.5: Switched capacitor voltage balancing circuit

ladder so that higher mismatch currents can be processed has been made by using a resonant ladder converter in [28]. The resonant switched capacitor ladder was used to regulate voltages of a series stack of 4 BeagleBone Black servers.

However, to implement a system of DPP converters for voltage regulation of a series stack of processor cores we have to study the system in more detail. Conventional multi-phase buck converters that supply modern processors have been modeled quite rigorously in the past. Averaged models of buck converters have been studied and methods to design linear control to improve their bandwidth have been researched since the advent of the first CPUs. To properly realize the benefits and consider the feasibility of using DPP to regulate core voltages, we have to study the dynamics of the element-to-element topology supplying a series string of processor loads [29].

2.3 Dynamics of the Element-to-Element Topology

To study the dynamics of the system, first we have to make a few simplifying assumptions. For simplicity we are going to assume that the core voltages

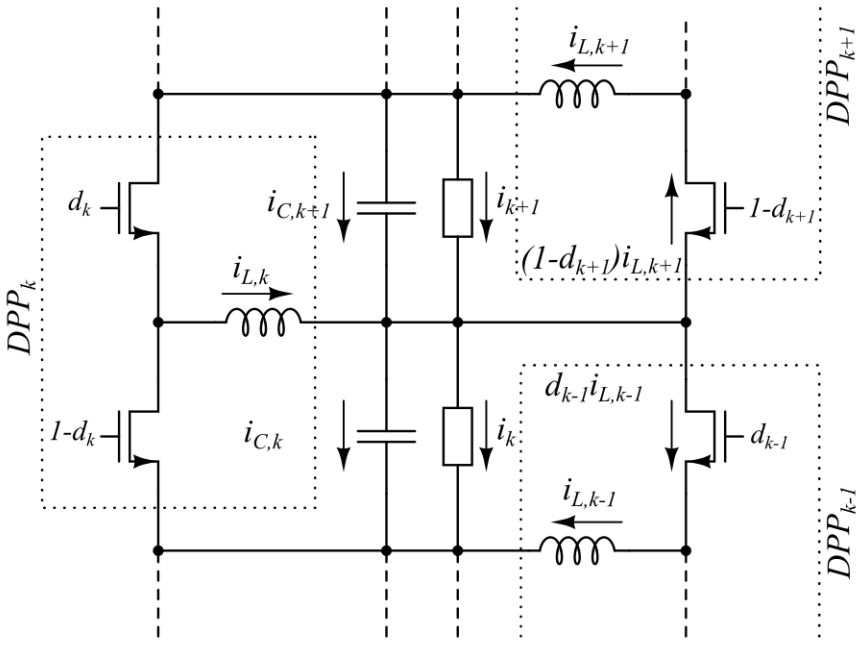


Figure 2.6: A single DPP converter with adjacent converter elements

are to be equalized. Although during actual operation we want the core voltages to be independently controlled according to the activity level of the core, this assumption has its validity. During actual operation usually the computational load of the processor is more or less equally distributed across all cores, so the voltage requirements of all cores rise or fall in close correlation with each other. Therefore, when we connect cores in parallel we expect the stack voltage to go down during low computational load and the differential power processing units will simply operate to equalize the core voltages. Now with voltage equalization in mind let us try to model the DPP system. Figure 2.6 shows a part of the entire stack from Figure 2.4.

Writing down the KCL equation at node N, we have

$$i_{c,k+1} + i_{k+1} - d_{k-1}i_{L,k-1} + i_{L,k} - (1 - d_{k+1})i_{L,k+1} - i_k - i_{c,k} = 0 \qquad (2.4)$$

Now recollecting $C\frac{dv_c}{dt} = i_c$ as the relation between capacitor voltage and current for an ideal capacitor and substituting $\Delta v_k = v_{k+1} - v_k$, we have

$$C\frac{d}{dt}\Delta v_k = d_{k-1}i_{L,k-1} - i_{L,k} + (1 - d_{k+1})i_{L,k+1} - (i_{k+1} - i_k) \qquad (2.5)$$

This equation is valid for all nodes except those at the two extremities of the stack. As seen from Figure 2.7, for the bottommost and topmost nodes we have respectively

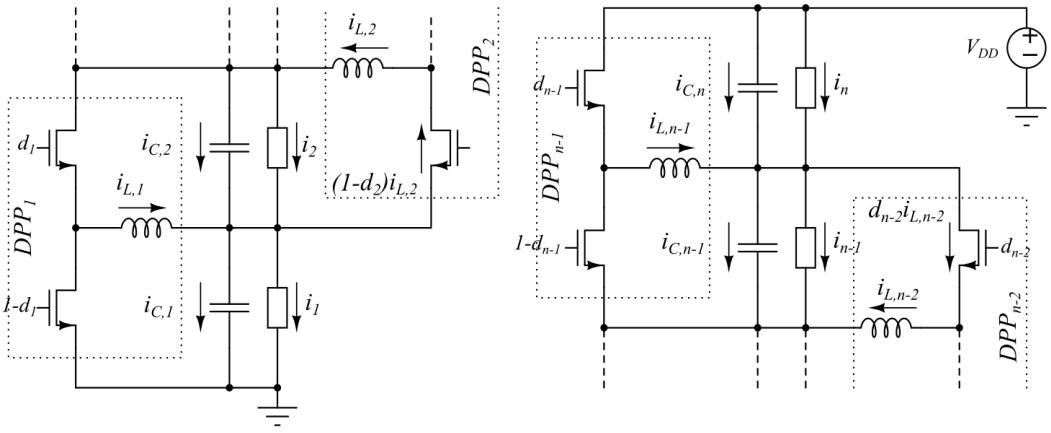


Figure 2.7: DPP converters at the extremities of the stack

$$
\begin{aligned}
C\frac{d}{dt}\Delta v_1 &= -i_{L,1} + (1-d_2)i_{L,2} - (i_2 - i_1) \\
C\frac{d}{dt}\Delta v_{n-1} &= d_{n-2}i_{L,n-2} - i_{L,n-1} - (i_n - i_{n-1})
\end{aligned}
\tag{2.6}
$$

This complete set of $n-1$ equations can now be linearized about operating points $i_{L,k} = I_{L,k} + \hat{i_{L,k}}$, $d_k = D_k + \hat{d}_k$ and $\Delta v_k = \Delta V_k + \Delta\hat{v}_k$. Also if we neglect perturbations in the mismatch currents $(i_{k+1} - i_k)$, then the above set of Equations 2.5 and 2.6 are simplified as

$$
\begin{aligned}
C\frac{d}{dt}\Delta\hat{v}_1 &= -\hat{i_{L,1}} + (1-D_2)\hat{i_{L,2}} - \hat{d}_2 I_{L,2} \\
C\frac{d}{dt}\Delta\hat{v}_k &= D_{k-1}\hat{i_{L,k-1}} - \hat{i_{L,k}} + (1-D_{k+1})\hat{i_{L,k+1}} + \hat{d_{k-1}}I_{L,k-1} - \hat{d_{k+1}}I_{L,k+1} \\
C\frac{d}{dt}\Delta\hat{v_{n-1}} &= D_{n-2}\hat{i_{L,n-2}} - \hat{i_{L,n-1}} + \hat{d_{n-2}}I_{L,n-2}
\end{aligned}
\tag{2.7}
$$

Apart from this set of equations there is another set of equations that define the inductor current dynamics in terms of duty ratios and domain voltages. Writing down those equations from Figure 2.6 we have

$$
L\frac{di_{L,k}}{dt} = d_k v_{k+1} - (1-d_k)v_k \tag{2.8}
$$

As described before, we have a simplifying assumption that voltage equalization is our objective. Because of that we have the nominal operating point of the duty ratio of the DPP converters at $D_k = \frac{1}{2}$ (for loss-less converters). Under this assumption, the above nonlinear set of equations reduces to

$$
L\frac{d\hat{i_{L,k}}}{dt} = \frac{1}{2}\Delta\hat{v}_k + \hat{d}_k\left(V_{k+1} + V_k\right) \tag{2.9}
$$

We can now write the complete set of $2\,(n-1)$ equations in state space form as

$$
\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u} \tag{2.10}
$$

where the state and input matrices A and B are given by

$$\mathbf{x} = \begin{bmatrix} \hat{\Delta v_1} & \hat{\Delta v_2} & \dots & \hat{\Delta v_{n-1}} & | & \hat{i_{L,1}} & \hat{i_{L,2}} & \dots & \hat{i_{L,n-1}} \end{bmatrix}'$$

$$\mathbf{u} = \begin{bmatrix} \hat{d_1} & \hat{d_2} & \hat{d_3} & \dots & \hat{d_{n-1}} \end{bmatrix}'$$

$$A = \left[\begin{array}{cccc|ccccc} 0 & 0 & \dots & 0 & | & -\frac{1}{C} & \frac{1}{2C} & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & | & \frac{1}{2C} & -\frac{1}{C} & \frac{1}{2C} & \dots & 0 \\ 0 & 0 & \dots & 0 & | & 0 & \frac{1}{2C} & -\frac{1}{C} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & | & \vdots & \vdots & \ddots & \ddots & \frac{1}{2C} \\ 0 & 0 & \dots & 0 & | & 0 & 0 & 0 & \frac{1}{2C} & -\frac{1}{C} \\ \hline \frac{1}{2L} & 0 & \dots & 0 & | & 0 & 0 & 0 & \dots & 0 \\ 0 & \frac{1}{2L} & \dots & 0 & | & 0 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & | & \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & \frac{1}{2L} & | & 0 & 0 & 0 & \dots & 0 \end{array}\right] \tag{2.11}$$

$$B = \begin{bmatrix} 0 & -\frac{I_{L,2}}{C} & 0 & \dots & 0 \\ \frac{I_{L,1}}{C} & 0 & -\frac{I_{L,3}}{C} & \dots & 0 \\ 0 & \frac{I_{L,2}}{C} & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & -\frac{I_{L,n-1}}{C} \\ 0 & 0 & 0 & \frac{I_{L,n-2}}{C} & 0 \\ \hline \frac{V_1+V_2}{L} & 0 & 0 & \dots & 0 \\ 0 & \frac{V_2+V_3}{L} & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & 0 \\ 0 & 0 & 0 & \dots & \frac{V_{n-1}+V_n}{L} \end{bmatrix}$$

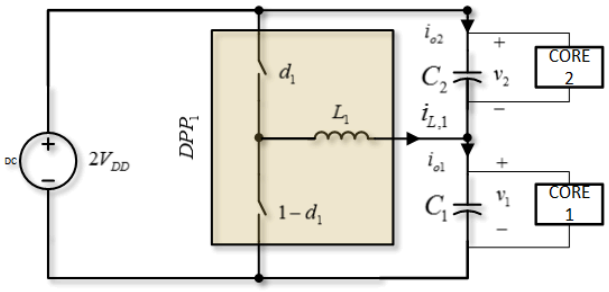


Figure 2.8: DPP converter module

The relationships between the voltage difference perturbations, $\hat{\Delta v_i}$, and the duty ratio perturbations, $\hat{d}_i$, are described by the above matrix equations. However it is still a bit unclear how the system behaves because of the cross terms in the top half of the B matrix. To maintain modularity, our motive is to develop a control for each DPP module that independently equalizes the voltages of the two domains it is attached to. This means that the controller for one DPP will only take in Δv_i as input and provide d_i as the output. The control design will have serious limitations if the voltage differences are also largely affected by the duty ratios of the adjacent DPP converters. The obvious way to decouple the adjacent DPP units is to increase the domain capacitances C.

Evaluating $(sI-A)^{-1}B$ with these matrices will give us an $(n-1)\times(n-1)$ matrix of transfer functions between $\mathbf{x}$ and $\mathbf{u}$. We are only concerned about the direct terms of that matrix since we are targeting a modular design. To observe the behavior of the direct terms of the transfer function matrix we are going to evaluate the transfer functions in MATLAB with varying number of stacked elements. For a 2 element stack (Figure 2.8) the transfer function $\frac{x_1(s)}{d_1(s)}$ is easy to derive

$$\frac{x_1(s)}{d_1(s)} = \frac{1}{1+2LCs^2} \tag{2.12}$$

Moving on to stacks containing more elements, we can see that the Bode plots of the transfer functions are as shown in Figure 2.9. The plots have been made in MATLAB using $L = 1\ \mu$H and $C = 100\ \mu$F, a small inductance ESR of 10 $m\Omega$ was also incorporated for damping. The effect of non-zero average inductor currents was neglected here since those effects can easily be desensitized (added zeros can be shifted to very high frequencies) by using

sufficiently large values of domain capacitances. The obtained Bode plots, even as the stack height (number of elements in the stack) increases, closely resemble the bode plot for the 2-domain case.

This suggests that the same transfer function of Equation 2.12 can be used to design closed loop control for the individual DPPs.

2.3.1 Closed Loop Control

The frequency response of transfer functions obtained suggests that modular design of converters with local control is possible. Each DPP converter can be modeled as a buck converter with 2 V_{DD} input and output V_{DD} as shown in Figure 2.9. The transfer function between duty ratio and voltage difference between the two domains is given by

$$G_{\Delta vd,i} = \frac{\Delta v_i}{d_i} = \frac{2V_{DD}\left(1 + 2r_c Cs\right)}{1 + \left(r_c + r_L\right)Cs + LCs^2} \tag{2.13}$$

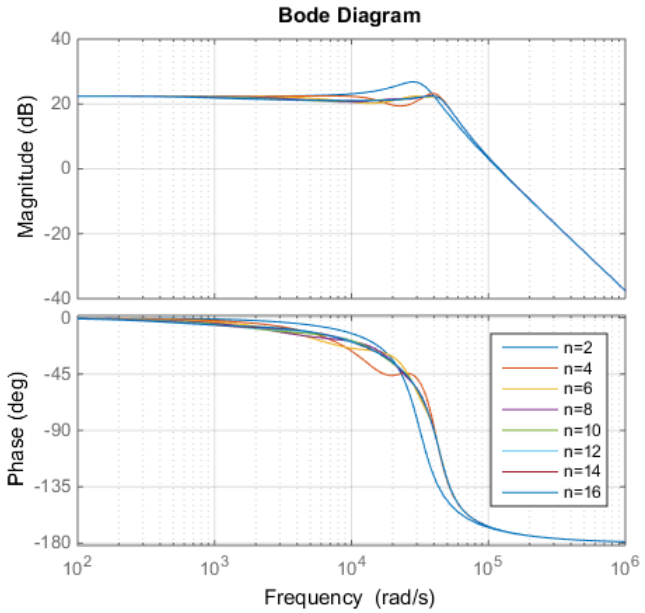


Figure 2.9: Frequency response of converters at the middle of the stack. n = stack height

Based on this model a 2-pole, 2-zero compensator of the form

$$C(s) = K\frac{(\omega_{z_1} + s)(\omega_{z_2} + s)}{s(\omega_{p_1} + s)(\omega_{p_2} + s)} \tag{2.14}$$

can be implemented to regulate the voltage difference to zero. To ensure that the controller is able to equalize the domain voltages as expected, let us simulate a few cases.

2.4 Simulation Results on Dynamic Performance

A test converter with $L = 5$ μH, $C = 20$ μF was considered with appropriate series resistances of 10 mΩ on both components and the compensator was designed so that we obtain 60 degree phase margin at a 100 kHz bandwidth. The regulated domain voltages are to be 5 V and transients of 0.5 A are considered. The simulation results for the 2-core case in Figure 2.10 verify that the closed loop control is effective in voltage equalization and reaches steady state in 20 μs which complies with the calculated bandwidth.

To see the impact of increasing the stack height, transient simulations with more cores in the stack have been carried out and the results are shown in Figure 2.11.

As we see from the simulations, the settling times of the voltages get severely degraded as the number of cores in the stack increases. So in effect to meet the performance specifications (bandwidth requirements) of processor cores, the switching frequency of the DPP converters has to be increased proportionally so that the degradation of bandwidth is compensated. This is undesirable as we are not getting the most bandwidth out of the designed converters and is one major disadvantage of this topology. It has been shown however that with global control and including inductor current information into our control design, improvement in performance can be achieved [30]. However, this will compromise our goal of achieving a modular design and the method will require large processing power for large core count. Instead, a topological modification to counter this problem of decreasing transient performance is proposed in the next section.

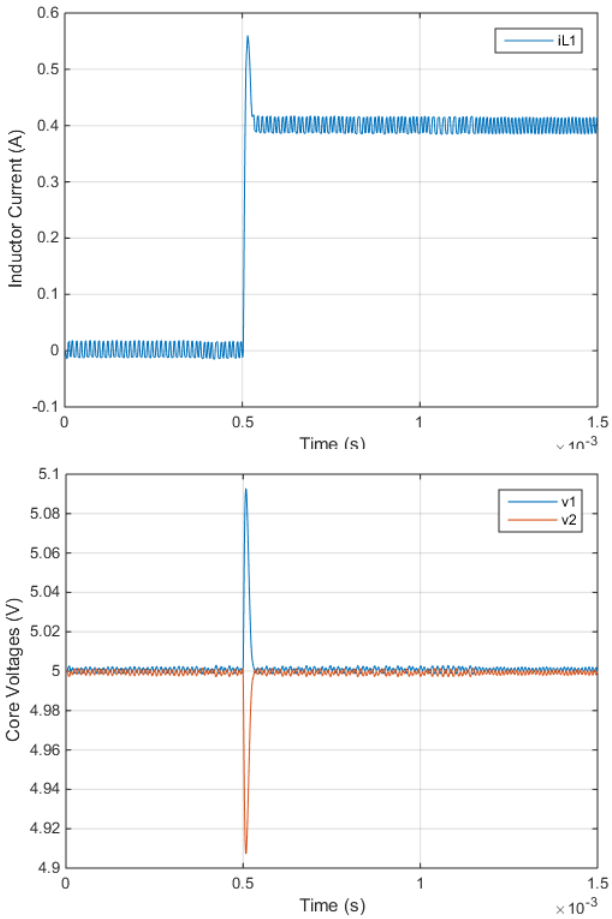
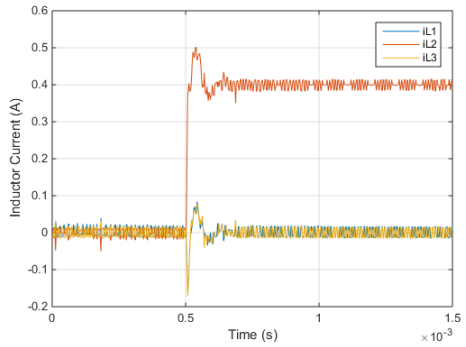
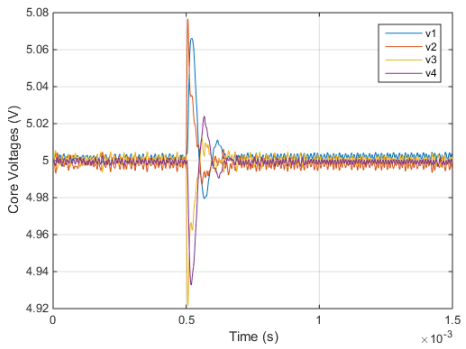
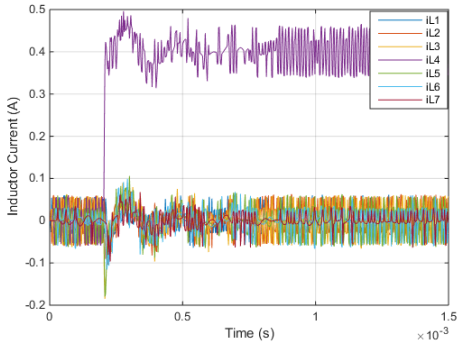


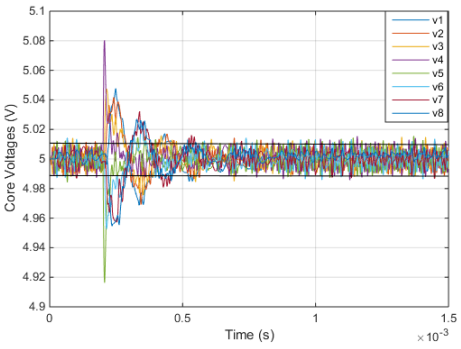
Figure 2.10: Load current step response, 2-core case



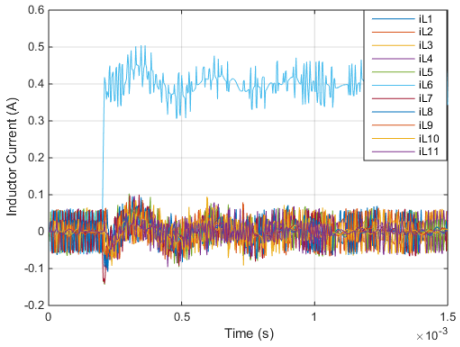


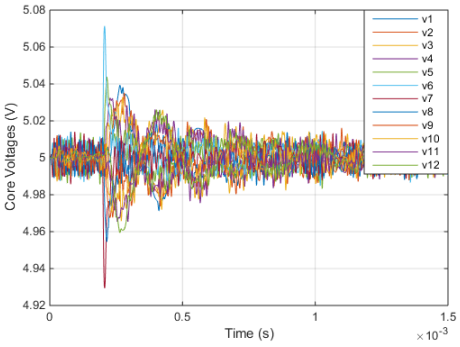
(a) 4 cores



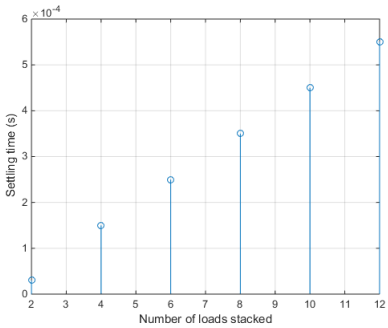


(b) 8 cores





(c) 12 cores



(d) Trend of settling times with core height

Figure 2.11: Simulated trend in settling times with increasing number of series-stacked elements

2.5 Hierarchical Element-to Element topology

A hierarchical element-to-element topology is now proposed that significantly improves transient performance for larger core counts. The topology, shown in Figure 2.12, is a direct implementation of the hybrid/nested topology suggested in [22]. In this topology, alternate DPP converters have been set to balance the voltages of two consecutive domains instead of 1. We can divide the DPP converters on the system into two categories: the inner DPP converters which regulate adjacent domain voltages and outer converters that regulate the voltages taken two domains at a time. The advantage that we obtain from this is that the inner converters are now completely decoupled from each other.

2.6 Motivation for Modified Topology

To understand why this topology was selected for improvement in transient performance we need to properly understand why the element-to-element topology performed worse with increasing core count. The bandwidth deterioration arises because of inductor current coupling exhibited in the capacitor charge balance equations. To maintain capacitor voltages at the set-points, the inductor currents of each DPP cannot change independently of the adjacent inductor currents. To look at the situation more intuitively, let us consider the path of the balancing current that must flow for voltage regulation (Figure 2.13). For an n element stack there are $n-1$ inductors in the balancing current path. The rate at which the average balancing current can increase in the event of a transient gets limited as we increase the number of DPP converters in the stack. This explanation also validates the linear increase in settling times that we observe in Figure 2.11 (d). The bus-to-load topology is not prone to the same problem.

The hierarchical element-to-element topology can be used to improve settling times for a large number of stacked loads. The difference between this topology and the original element-to-element topology is that this one has half the converters running in an outer loop (DPP_2,DPP_5...,DPP_{n-2}). This reduces the number of inductors in the balancing current path to half the number that was present in the original element-to-element topology. So in

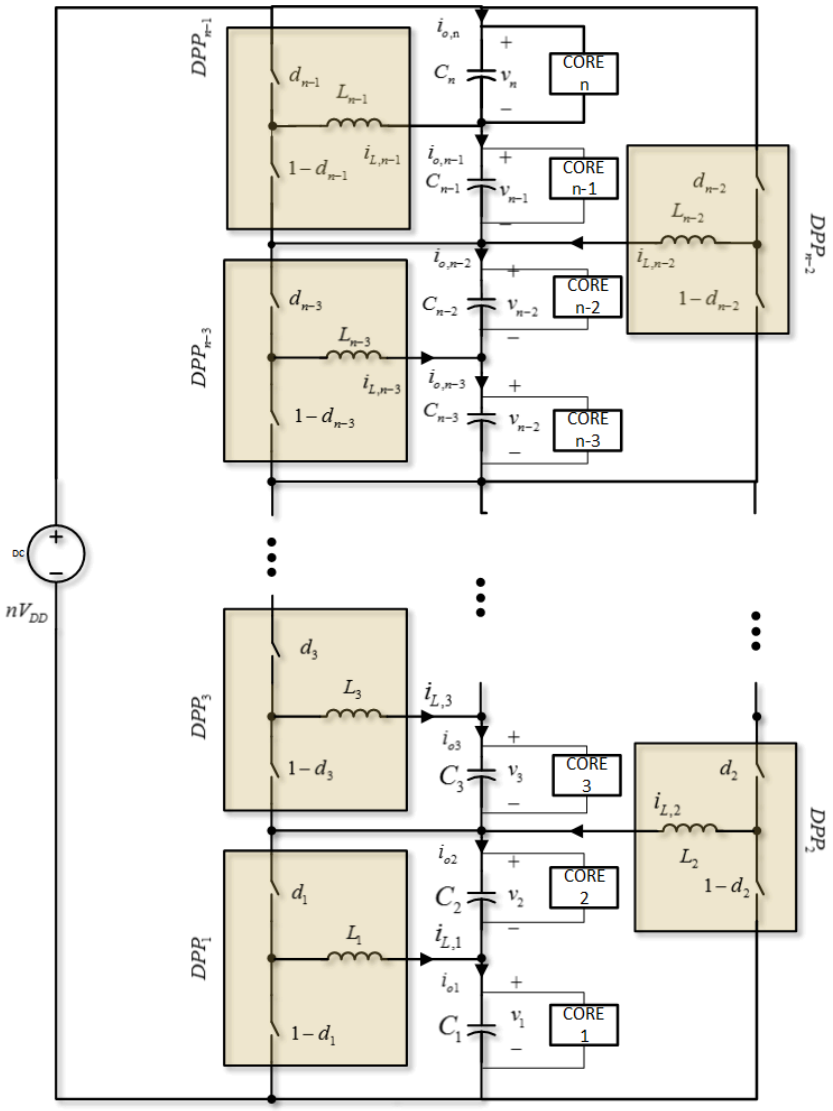


Figure 2.12: Hierarchical element-to-element DPP

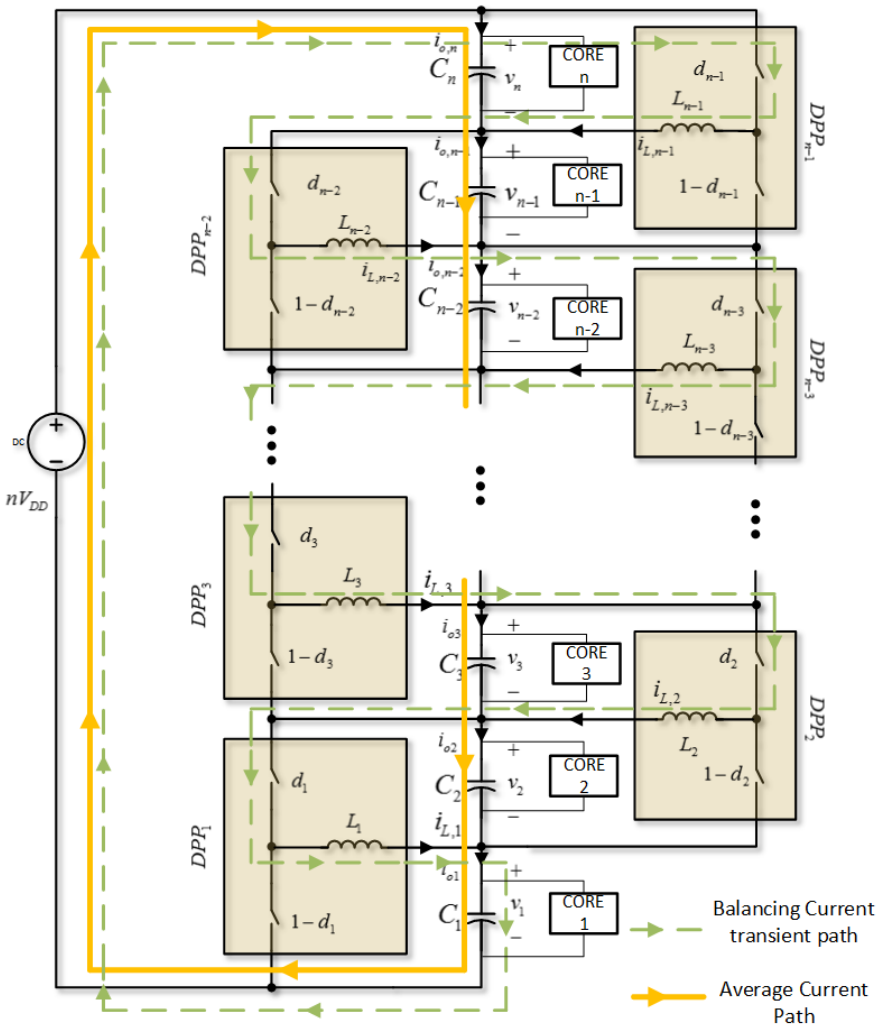


Figure 2.13: Balancing current path in element-to-element topology

effect a two-level (one inner loop and one outer loop) hierarchical element-to-element topology can be used to double the number of voltage domains in series without degrading transient performance. If even more cores need to be stacked, then the number of hierarchies in the topology can be increased.

One obvious disadvantage of the hierarchical topology is the higher switch voltages on the outer-loop converters. This implies that we will have higher switching losses in the outer-loop converters if the switching frequencies of the inner loop converters are kept equal. This is essential because the bandwidths of the outer loop converters define the bandwidth of the series-connected system. So the efficiency of the hierarchical topology will be slightly lower than that of the original element-to-element topology. This limitation is not

very serious since the converters process very little power and the impact of DPP converter efficiency on net efficiency is negligible for small mismatches.

2.7 Simulation Results on Improvement in Dynamic Performance

To validate improvement in dynamics we have to ensure that the bandwidths of the outer converters are the same as that of the inner loop converters. Since the input voltages of the outer converters have been doubled, the inductors in those converters have to be doubled as well. Matching bandwidths of inner loop converters and outer loop converters is not a necessity in an actual implementation. However, here we want to demonstrate that the improved transient response is a consequence of the modification of the topology only and not a consequence of improved DPP converter bandwidth. The simulation results for 8 stacked domains for the element-to-element and the hierarchical topologies are shown in Figure 2.14.

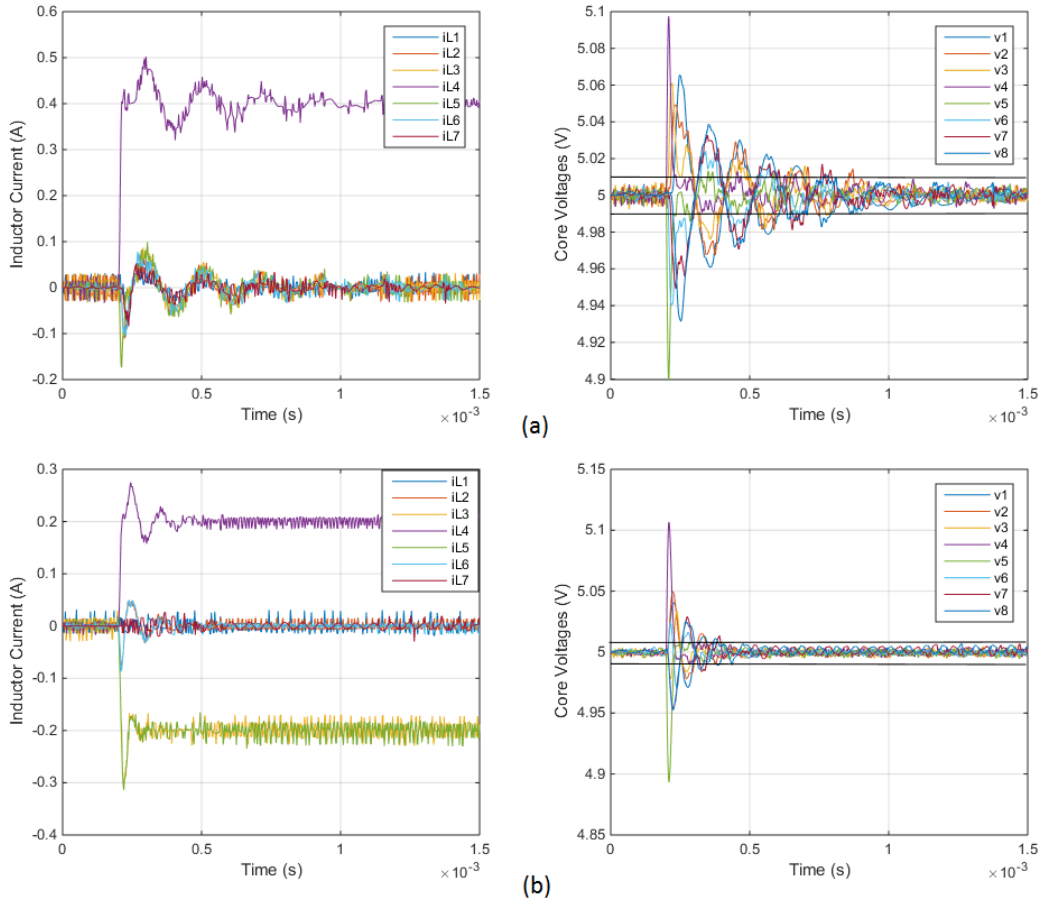


Figure 2.14: Improvement in transient performance:
(a) Element-to-element DPP
(b) Hierarchical element-to-element DPP

CHAPTER 3

HARDWARE VERIFICATION

A hardware setup with 4 series-connected voltage domains was prepared to test our theoretical results. The hardware was initially set up for powering a stack of 4 Beaglebone Black boards. So the converter was optimized for 3.3 V - 5 V output and a few (less than 10 W) of output power. However the Beaglebone Black boards do not consume nearly as much power for the switched inductor converter to be effective. The converter is bidirectional and the mismatch currents are too low. Effective efficiency evaluation with small Beaglebone loads would require setting a very high switching frequency (several MHz) and relatively large inductor value (since voltage is high, and following from our previous discussion on scaling). So for verification 4 electronic loads (HP 6060B) were connected to form a series stack. The hardware was set up such that both the original element-to-element and the hierarchical element-to-element topologies can be tested on the same board.

3.1 DPP Converter board

A high-level schematic of the board is shown in Figure 3.1. Each individual DPP unit has a 2-phase buck converter with integrated drivers. Both phases are not required considering that the power processed by the converters is expected to be low. This enables us to have 2 different values of inductors in the converter at the middle of the stack, one the same as the other two DPPs and another that is double of the other two. The switches J1A, J2A, J1B, and J2B enable us to switch between the element-to-element topology and the hierarchical element-to-element topologies and has been incorporated with an array of shorting resistors. This way, the number of wire connections external to the board has been kept minimal. Only connections to the individual loads and no additional connections to configure the board into the required

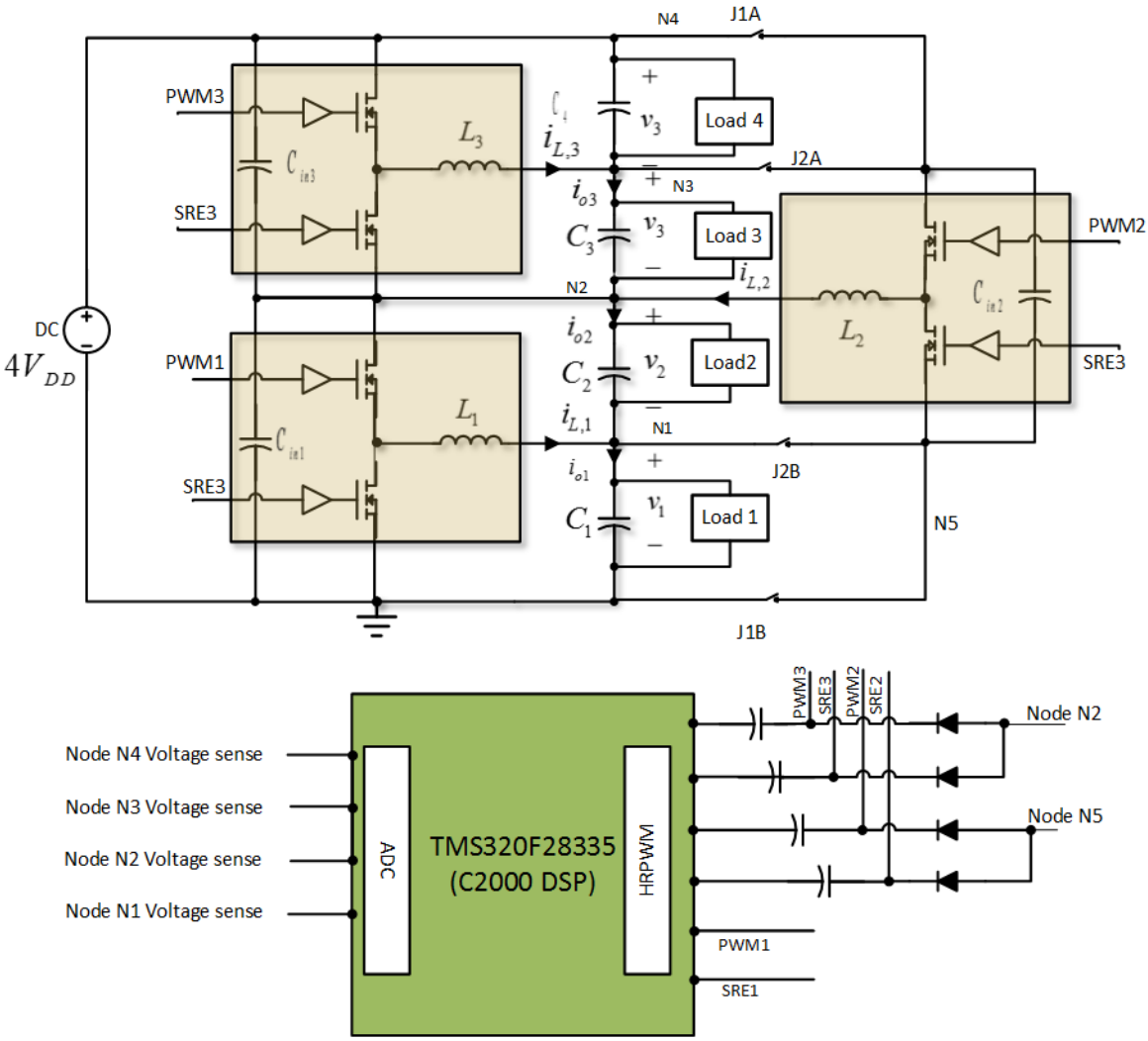


Figure 3.1: High-level schematic of the implemented hardware

topologies are necessary.

The 2-phase converters have been implemented with UCD7242 drivers (with integrated power MOSFETs). Each UCD7242 chip contains 2 synchronous buck converters with their corresponding separate drivers. The advantage of using a synchronous buck converter with integrated drivers is that it keeps the length of connections between the driver and the MOSFET minimum. This ensures low parasitic inductance in that path and switching frequencies can be increased to a few MHz without significant switching losses. This particular device offers switching frequencies up to 2 MHz and was operated at 1 MHz and 500 kHz for our efficiency estimation. These drivers are useful for digital control and they offer internal average current

measurement and fault detection. Each UCD7242 chip has an internally generated gate-driver supply that generates optimized gate drive voltage for the particular value of input voltage used. The chip can operate from an input voltage higher than 4.7 V according to the datasheet, but there is a 0.5 V headroom. It was observed that the driver-MOSFET pair operates properly even when supplied with voltages as low as 4.2 V. So in effect the hardware can generate domain voltages that are as low as 2.1 V operating at 0.5 nominal duty ratio. There are 2 PWM inputs for each buck converter on the chip; the PWM input is the signal that drives the top MOSFET and the SRE pin input drives the bottom MOSFET. If the SRE pin is driven high all the time, then the converter operates in forced continuous conduction mode (FCCM). The driver internally generates the complementary drive signal and also incorporates a suitable dead time. If the SRE pin is driven low, then the lower switch of the converter operates in diode emulation mode, i.e., the bottom switch automatically shuts down if the current through it goes negative during its on time. However this mode is not useful for this application due to the needed bi-directional nature of the converter.

Optimizing the inductor value on each DPP converter is very important. Since we have seen that the transient response slows down as we increase the number of elements in the stack, it is tempting to improve the transient response by simply lowering the inductance and increasing capacitance per domain. However, if no light load mode is implemented, then at low loads the converters output only inductor current ripple and it is easy to see that large current ripples cycling through the stack of converters will cause high unwanted switching and conduction losses even at small mismatches. An inductor value of 4.7 μH was deemed suitable for this board. Operating at 1 MHz switching frequency, 50% duty ratio, and domain voltage of 3.3 V the current ripple will be only be 0.35 A peak-to-peak. Since we are going to test this converter for a maximum mismatch current of 5 A this current ripple value seems suitable. A 9.8 μH inductor and a 4.7 μH were connected to the two phases of the DPP in the middle for reasons mentioned before. Ceramic capacitors at 120 μF were used to form the output capacitance at each domain.

For control of the DPPs a single C2000 microcontroller was embedded on the board. The board was set up such that the microcontroller would derive its supply (1.8 V core and 3.3 V I/O) from the stack supply. The

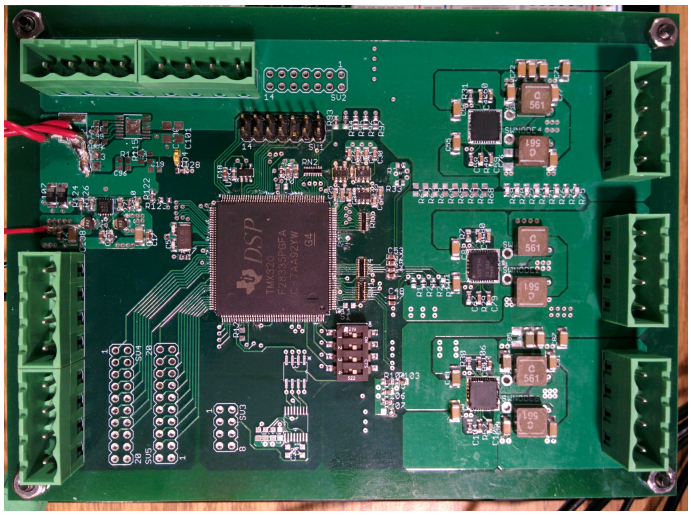


Figure 3.2: DPP board with 3 converters

internal ADCs of this particular DSP offer fast sampling (12.5 MSPS). So sampling the node voltages at N1, N2, N3, N4 would only take up 320 ns. This was one particular reason for choosing this DSP. Also the clock speed at which this DSP works is 150 MHz which enables us to obtain good PWM resolution (6 bits without using the high-resolution PWM feature and 11.1 bits with the HRPWM feature enabled) even at frequencies higher than 1 MHz. The closed loop control implementation from the last chapter requires computation of three 2 pole-2 zero compensator. Implemented in floating point this computation would take at least 120 clock cycles and it takes slightly less than 1 μs when implemented in this DSP. Accounting for other time delays encountered in interrupt response etc. the entire compensation and PWM generation scheme takes about 2 μs to take place.

Another aspect that has to be noted is that the individual PWM inputs of the second and third DPPs from the bottom require level shifting; i.e., since their ground references are not connected to the ground of the DSP level shifting is required to drive those pins. This has been implemented by a simple bootstrapping circuit implemented with a Schottky diode and a capacitor as shown in Figure 3.1. The diodes ensure that the nodes PWM2, SRE2, PWM3 and SRE3 always remain above the local ground voltages of DPP2 and DPP3, and the PWM outputs from the DSP simply makes the arrangement act like a charge pump. All 6 HRPWM channels offered by the TMS320F28335 DSP were used to control the three 2-phase converters on the board. Figure 3.2 shows a picture of the fully populated board. The entire

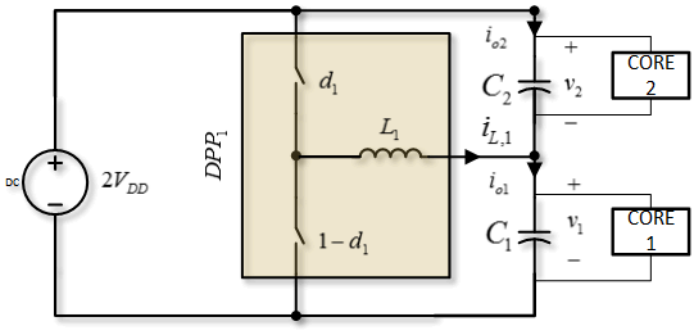


Figure 3.3: Individual DPP converter

schematics and the layout of the layers on the board are shown in Appendix A. An external JTAG emulator was used to program the DSP. The c-code used to program the DSP is shown in Appendix B.

3.2 Transient Performance Tests

Digitally implementing a controller like the one described in Section 2.3 involves a the discretization process. There are several ways to discretize a continuous Laplace domain transfer function to get a discrete transfer function. Numerous methods can be found in a digital control textbook [31]. The circuit parameters of the converter shown in Figure 3.3 were $L = 4.7$ μH, $C = 100$ μF, $r_L = 15$ mΩ, $r_C = 5$ mΩ, $r_{sw} = 10$ mΩ.

Plugging these values into the transfer function of the DPP converter previously derived in Chapter 2 (Equation 2.13), we obtain the transfer function that relates the duty ratio d_i with the voltage differences Δv_i. A 2-pole, 2-zero compensator was then designed to control the buck converter. Figure 3.4 shows the frequency response of the converter, the compensator and the loop gain.

Switching at a frequency of 1 MHz, the achievable bandwidth of a buck converter should usually be between 100 kHz and 200 kHz. However digital control provides several obstructions to achieving that closed loop bandwidth. The worst limitation that arises is because of the ADC delay and computation time. Incorporating the ADC and computation delay (τ) into the converter transfer function Equation 2.15 is modified as

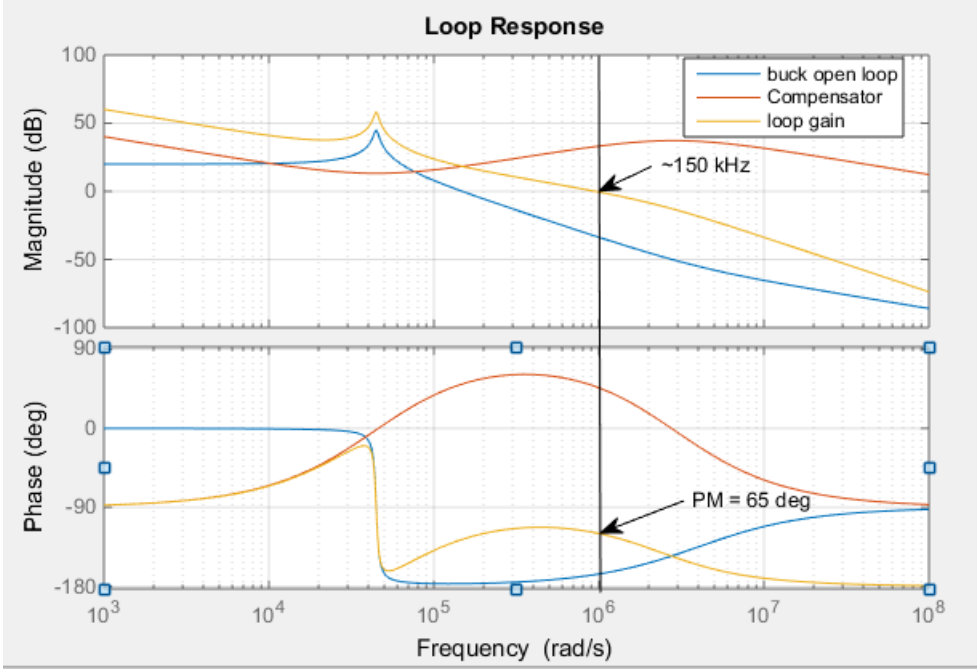


Figure 3.4: Compensation

$$G_{vd} = \frac{\Delta v_i(s)}{d_i(s)} = \left[\frac{2V_{DD}\,(1 + 2r_C Cs)}{1 + (r_L + r_{sw} + r_c)Cs + LCs^2}\right] e^{-\tau s} \tag{3.1}$$

Several approximations exist for simplifying the exponential term in the transfer function. In general this term behaves approximately like a right half-plane zero situated at $\omega_z = 1/\tau$. However, instead of going into the approximated transfer function we will simulate the actual transfer function in MATLAB. Figure 3.5 shows the impact of the delay. Since the computation time and ADC delay are slightly higher than 2 μs we see that the phase lag due to the delay term makes the previously designed closed loop system in Figure 3.4 unstable. In fact with a 2-pole, 2-zero compensator and 2 μs delay it is impossible to achieve a bandwidth of 100 kHz while maintaining a good phase margin. Ultimately we had to settle with a sampling frequency of 250 KHz (not the same as the switching frequency, which is still 1 MHz to reduce inductor current ripple) and a bandwidth of 25 kHz. There are several ways to avoid such a heavy bandwidth deterioration due to computation time [32]; however, since our motive is to verify the deterioration in settling time with stack height, those methods were not tried out here. The designed compen-

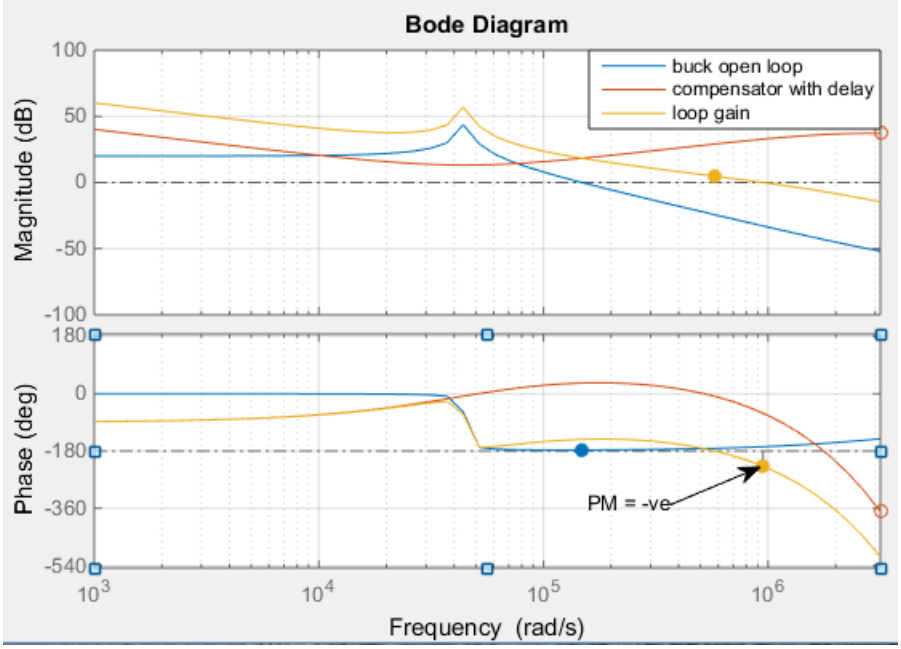


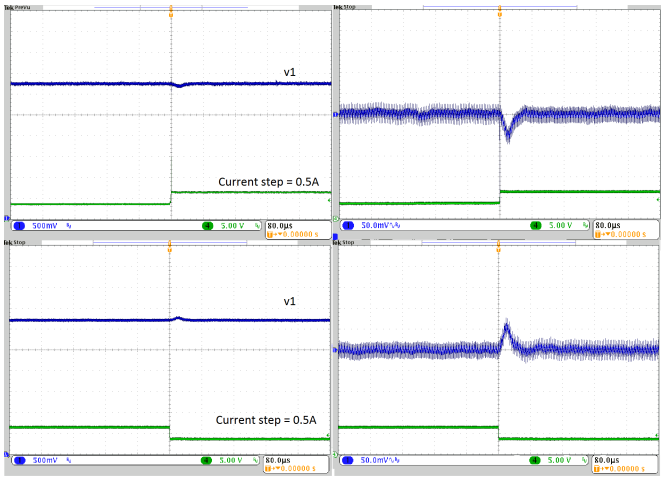
Figure 3.5: Delay effect

sator in Laplace domain was then discretized using a bi-linear transform[1] and the program in Appendix B was used to control the converters.

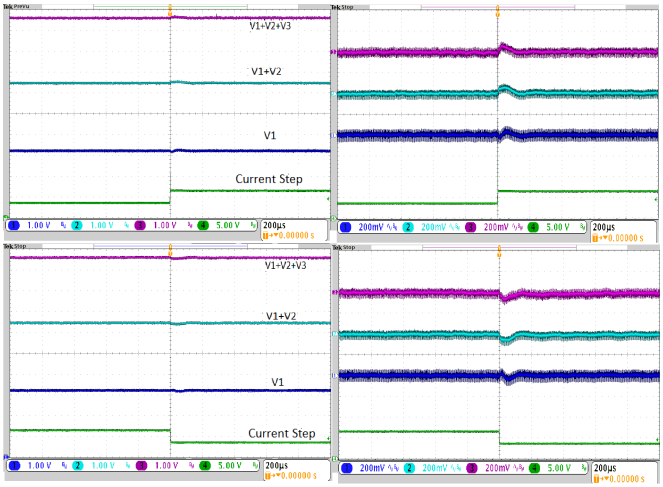
The transient performance was then tested with respect to a load current step of 1 A first when only 2 domains are present in the stack and then when 4 domains were present in the stack. The results shown in Figure 3.6 show the deterioration in settling time as expected from our simulations.

The configuration of the DPP converters was then changed to the hierarchical topology on the same board and the second phase of the converter in the middle of the stack was used to ensure that the bandwidth of the outer converter remains same as the other two converters. The effect of the reduction in inductor current coupling (as discussed in Chapter 2) is clearly seen in Figure 3.7. For the same current transients as in Figure 3.6 (b) we see that the settling time is 40 μs - the same as when we had only two series-connected voltage domains.

[1]bilinear transform: $s = \frac{2\left(1-z^{-1}\right)}{T\left(1+z^{-1}\right)}$



(a) 2 domains (settling time = $40\mu s$)



(b) 4 domains (settling time = $100\mu s$)

Figure 3.6: Transient performance tests for the basic element-to-element topology

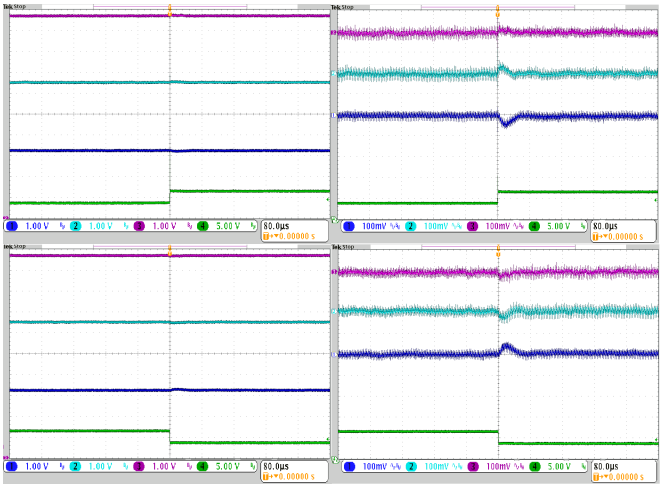


Figure 3.7: Transient performance test for the hierarchical topology

3.3 Efficiency

Our target has always been efficiency improvement. To evaluate the efficiency of the converter stack, digital loads were used and closed-loop control was set up to regulate the domain voltages at 2.5 V. The converters were tested with 1 μH output inductance and 4.7 μH output inductance at switching frequencies of 1 MHz and 500 kHz. The nominal current consumed by the loads at each domain was set at 5A (total load of 50W). The efficiencies were tested first at 10% mismatch (0.5 A deviation from 5 A) and then at 20% mismatch. Eighteen (exhaustive set of extreme mismatch conditions possible under 10% or 20% mismatch) sets of data points were taken and the average efficiencies measured are tabulated in Table 3.1.

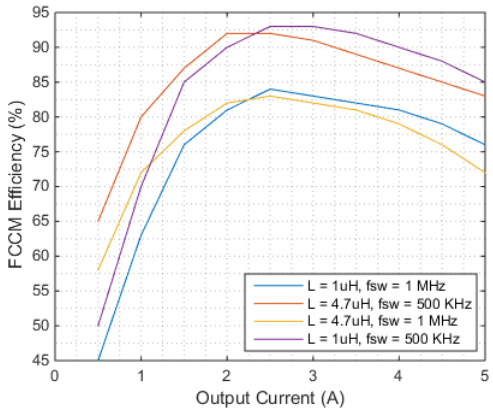


Figure 3.8: Individual DPP converter efficiency plot

Table 3.1: Averaged efficiency of the element-to-element topology, nominal stack current = 5 A

Maximum % mismatch	L=4.7 μH f_{sw} =500 kHz	L=4.7 μH f_{sw} =1 MHz	L=1 μH f_{sw} =500 kHz	L=1 μH f_{sw} =1 MHz
10%	96.2%	94.3%	94.8%	93.8%
20%	95.8%	94.1%	94.2%	92.7%

We can see from the Table 3.1 that efficiencies obtained using the lower inductance value are considerably lower than the efficiencies obtained using the higher inductance value, even though the series resistance of the 1 μH inductor is less. The setup using the larger inductor has considerably lower current ripple and the circulating ripple currents in the circuit under low mismatch conditions are much lower. This reduces the RMS and switching losses in the circuit at light loads. However, dynamically the lower inductance value provides more bandwidth, which is necessary for this series stacking architecture. Also even at 20% mismatch, the converters only process about 1 A of mismatch current, which as we can see from the efficiency plot, Figure 3.8, is not the optimal point for which the converter was designed. To optimize the converter such that the optimal point shifts toward a lower current value, the inductance has to be increased (which proves to be dynamically crippling) and the switching frequency has to be increased further which lowers the efficiency of our DPP converters further. This motivates us to employ a different method to improve the light load efficiency of the DPP converters without having to use a large inductance or having to increase the switching frequency unnecessarily. A method to do that will be discussed and implemented in the following chapters.

CHAPTER 4

IMPROVING EFFICIENCY AND DYNAMIC PERFORMANCE

As we have seen in the previous chapter, the coupled inductor currents in the load-to-load DPP topology cause the bandwidth of the balancing converters to degrade significantly as more cores are connected in series. To improve bandwidth of the balancing circuit, smaller inductance values can be used (with increased capacitance per voltage domain). This however comes with an efficiency decrease. Since it is expected that most of the converters are going to process small amounts of current, decreasing the inductance will increase the RMS copper losses significantly at light loads. This suggests that some sort of light-load mode (discontinuous conduction) has to be implemented to further improve efficiency of the balancing circuit. In this chapter implementation of a controller that performs these functions is discussed.

4.1 Light-Load Modes in a Buck Converter

Fixed frequency PWM operation in synchronous buck converters provides us with the advantage of using a simplified controller which does not have to reconfigure its gains based on load current. The transfer function of the buck converter remains the same at all load currents. However at light loads the synchronous buck converter becomes increasingly inefficient. At zero load the inductor in the synchronous buck converter carries only ripple current (nonzero RMS losses). Added to that are the switching losses (in the devices due to turn-on and turn-off times, gate drivers and the inductor core). A conventional buck converter with a diode as the low side switch, however, turns out to be more efficient in light load than a synchronous buck converter in FCCM because it enters discontinuous mode.

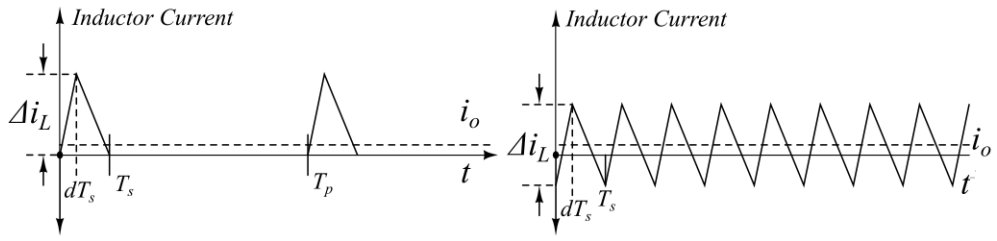


Figure 4.1: Single pulse PFM mode (left) and FCCM mode (right) inductor currents at same average output current

4.1.1 Discontinuous Conduction Mode Efficiency Improvement

The inductor current waveforms of a buck converter operating in pulse frequency modulation (PFM) mode and another converter in FCCM are shown in Figure 4.1. It can be shown that the RMS inductor current in PCM mode is

$$i_{L,rms} = \sqrt{\frac{2i_o\Delta i_L}{3}} \tag{4.1}$$

and the same under FCCM can be shown to be

$$i_{L,rms} = \sqrt{i_o^2 + \frac{\Delta i_L^2}{12}} \tag{4.2}$$

Comparing the two expressions we can see that if we want to have lower conduction losses it is only advantageous to go into PFM mode when the average output current is lower than $\frac{\Delta i_L}{6}$. However, at light loads, the losses in the converter are dominated by the switching losses rather than the conduction losses. The switching losses are mainly divided into gate driver losses, device switching losses due to turn-on and turn-off times, and the inductor core losses. An expression of the switching losses in a buck converter can be obtained as

$$P_{sw} = V_{in}I_{out}\left(t_{on} + t_{off}\right) \tag{4.3}$$

$$P_{gate_drv} = Q_{gate}V_{drv}f_{sw} \tag{4.4}$$

$$P_{core} = CB_{max}^{y}V_eF_{sw}^{x} \tag{4.5}$$

Here t_{on}, t_{off} are the turn-on and turn-off times of the MOSFETs, Q_{gate} is the sum of the gate charges of the high-side and the low-side switches and B_{max} is the difference between the maximum and minimum flux densities in the inductor core. The parameter x in the core loss expression is greater than 1, so the total switching loss in the buck converter is proportional to the switching frequency. Lowering the switching frequency in a buck converter at light loads by going into DCM is thus advantageous for improving its light load efficiency. Other forms of light load discontinuous modes include burst-mode (in which multiple pulses are used instead of a single pulse as in PFM) and pulse skipping mode (in which the controller decides if a pulse should be skipped to maintain a discontinuous mode at light load). All of these methods have inherently the same effect (i.e., reducing the switching frequency) on the converter operation and efficiency as the single pulse PFM method.

4.1.2 Diode Emulation

In terms of conduction losses, the synchronous buck converter is more efficient than a buck converter with a diode as the low-side switch. However, to improve light load efficiency in a synchronous converters it has to be able to enter DCM. To enable that, a method known as diode emulation is used. A general way to emulate a diode using external logic, current detection and a high-performance switch like a MOSFET was proposed in [33]. In the context of a synchronous buck converter, the lower switch can be operated in diode emulation mode by pulling down the switching signal operating it whenever the controller detects a negative drain current (for NMOS). Instead of the drain current, the inductor current can also be used for diode emulation of the lower switch.

4.2 Switching Boundary Controllers

A generalized linear controller which can operate both in CCM and DCM is difficult to design. Usually both the inductor current and the output voltage have to be used in the control (current mode control) to obtain a controller that is stable in both modes. Most light-load mode controllers switch from a linear controller to a hysteretic controller whenever light-load is detected. Devising a generalized controller which operates both in CCM and DCM for bi-directional current output is our goal. A class of controllers known as switching boundary controllers provides an attractive solution to this problem.

Switching boundary controllers are fundamentally different from conventional PWM controllers mainly in the generation method of the switching signals. A general switching boundary controller is usually defined by the following set of equations:

$$\begin{aligned} u = 1 &: \sigma(\mathbf{x}, \mathbf{t}) < 0 \\ u = 0 &: \sigma(\mathbf{x}, \mathbf{t}) > 0 \end{aligned} \tag{4.6}$$

The full state feedback nature of the controller (like current mode controllers) ensures better stability and control over transient response compared to conventional voltage mode controllers. This method is also more useful than voltage mode control for converters that have a right half plane zero (like the buck-boost DPP converters in our application if voltage regulation of each domain is our motive, and not voltage equalization of adjacent domains).

The parameter $\sigma(\mathbf{x}, \mathbf{t})$ is known as the switching surface and is a function of the states of the converter and time. The control is inherently non-linear in nature and the switching surface can be planar or curved or even time varying. Planar switching surfaces are common and they form the basis of hysteretic controllers. The comparison-with-zero nature of the controller defined by Equation 4.1 makes the switching frequency of the converter infinite at the equilibrium point. This is undesirable as infinite switching frequency is impractical and does not make sense for switch-mode power converters. To make this form of controller suitable for power electronics a slight modification shown in Figure 4.2 is made.

A planar sliding surface for a converter with two energy storage elements

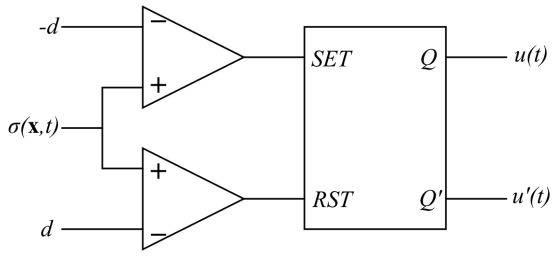


Figure 4.2: Sliding mode switching signal generation

is given by

$$\sigma\left(\mathbf{x},\mathbf{t}\right)=k_{v}\left(v_{ref}-v_{o}\right)+k_{i}\left(i_{L,ref}-i_{L}\right) \tag{4.7}$$

The sliding mode controller action usually regulates the sliding surface such that it remains within bounds set by d and $-d$ under certain conditions dependent on the dynamics of the converter itself [34]. For a buck converter, whose voltage needs to be regulated to v_{ref} without any steady state error, usually the term $i_{L,ref}$ is set as

$$i_{L,ref}=k_{i,v}\int\left(v_{ref}-v_{o}\right)dt \tag{4.8}$$

This forms the basis of current-mode hysteretic control which we will discuss later in this section. In general both hysteretic controllers (current mode and voltage mode with augmentation) perform very well dynamically and they also have the advantage of smoothly transitioning into light load modes. This is one reason to consider these types of controllers for our bi-directional light-load problem.

4.3 Voltage Mode Hysteretic Controllers

A buck converter with a hysteretic voltage mode controller is shown in Figure 4.3. The operation of the hysteretic comparator is as follows. Whenever the output voltage hits a lower limit the hysteretic comparator turns on the top switch and turns off the bottom switch after a specified amount of turn-on delay. Whenever the voltage hits a lower limit the reverse switching operation happens after a specified turn-off delay. The turn-on and turn-off delays are necessary for stability of the converter and arise because of

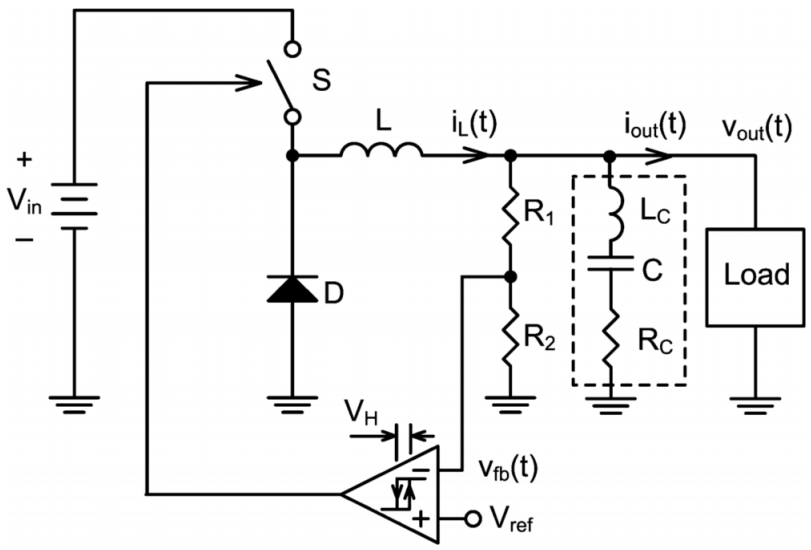


Figure 4.3: A buck converter with hysteretic control

the inherent phase delay between the switching signals and the capacitor voltage ripple of the buck converter. The exact expressions of the switching frequency of operation can be found in [35]. The converter automatically goes into a light-load mode if the inductor current is not allowed to go negative (using a diode as the lower switch in a buck converter ensures that). The switching frequency of the hysteretic regulator is however very sensitive to the capacitor parameters. The DC regulation of the regulator also varies slightly when transitioning from DCM to CCM as seen in Figure 4.4

To reduce the frequency variation of the hysteretic buck regulator a ramp is usually superposed on the original downscaled output voltage feedback. In the ramp generation method shown in Figure 4.5(a) an RC integrator is used to generate a triangular ramp by integrating the inductor voltage. Only the AC component of this ramp is then passed on to the feedback node by using the coupling capacitor C_{inj}. In the method shown in Figure 4.5(b) the entire ripple at the output voltage node is passed down to the feedback node. This method is particularly useful if the output capacitor already has significant ESR and the output voltage ripple is significant.

A slightly different method of generating the feedback voltage has been found to be useful as shown in Figure 4.6. If the time constants of the inductor (RL circuit) and the series RC branch are matched, then the voltage across the capacitor turns out to be equal to the voltage across the ESR of

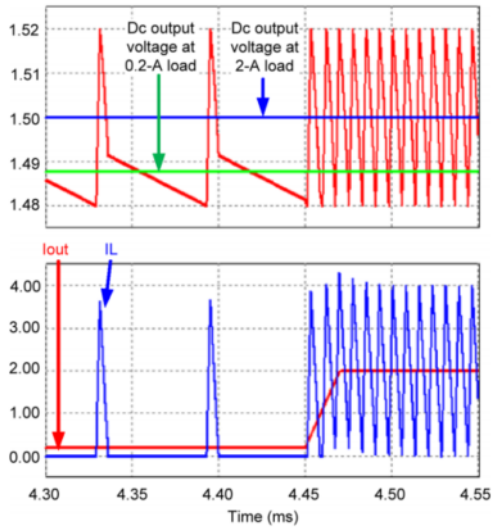
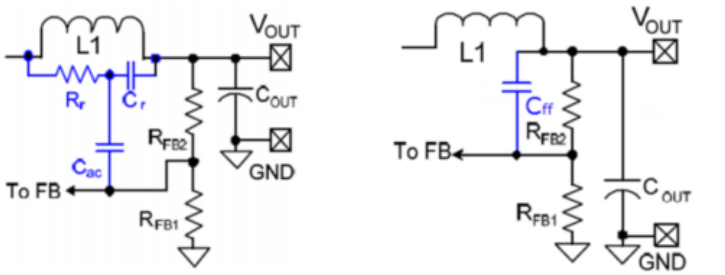


Figure 4.4: DC regulation of hysteretic buck



(a) Current ripple feedforward (b) Voltage ripple feedforward

Figure 4.5: Current and voltage ripple feedforward

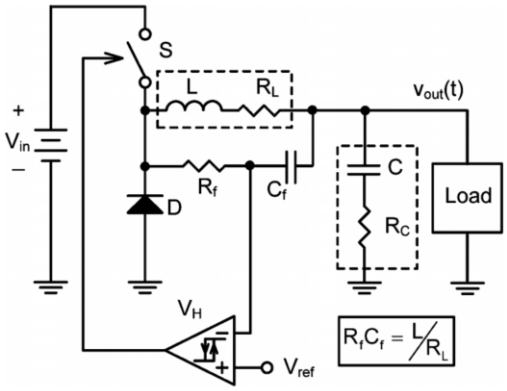


Figure 4.6: Droop control of hysteretic buck converter

the inductor. Consequently the voltage at the node between R_f and C_f is $v_o + i_L r_L$. The hysteretic controller thus regulates the output voltage to $v_{ref} - i_L r_L$. This implies that there is a droop in the average output voltage with increasing load current, which may or may not be desirable. Recent microprocessor chips allow voltage droop as load increases as a means to reduce power consumption of the cores (less heating as a result). The most recent VRM from Intel specifies that for high power processors to be used in servers 0.8 mV of voltage droop per ampere of current increase is allowed (specified as a DC impedance of 0.8 mΩ). The DC output impedance of the converter under this control is equal to the ESR of the inductor. To reduce this value (lower droop or no droop) an external slower PI loop may be implemented which controls the positive reference voltage of the controller according to the actual error between the output voltage and the reference voltage. Alternatively, putting a coupling capacitor between the sensing node and R_f and adding a large resistor in parallel with C_f will reduce the droop voltage to zero.

4.4 Extending Light Load Operation to Bi-directional DPP Converters

Extending the voltage mode hysteretic controllers discussed in the previous section to control our bi-directional DPP converters in light load is possible by an augmentation of the switching signals, as shown in Figure 4.7. When the load current is positive we do not allow the inductor current to go neg-

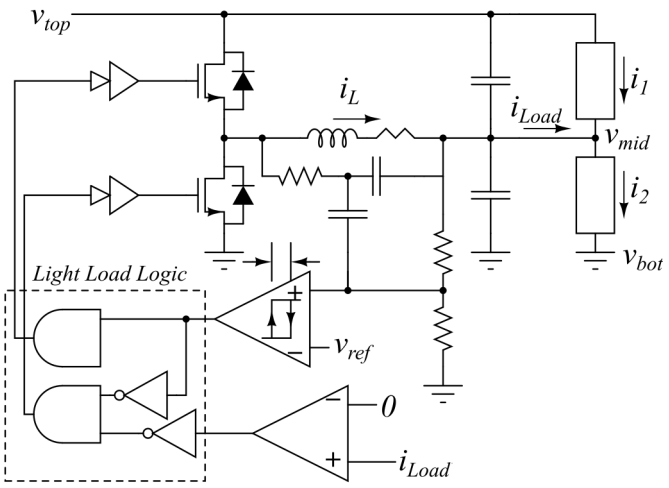


Figure 4.7: Bidirectional DPP converter with hysteretic control

ative by either pulling down switching signals at the lower switch to ground or operating it in diode-emulation mode (DEM). When the load current is negative we operate the top switch in DEM or simply pull down switching signals at the gate of the top switch. The main concern here is the detection of the load current direction. Using a sense resistor between the output node and the load will dissipate power and cause either droop in the output voltage or extra damping in regulation. A lossless method of estimating the load current direction is by sensing the capacitor currents and the inductor currents as explained in the next subsection.

4.4.1 Capacitor Current Based Detection of Load Current

Capacitor current sensing is very similar to DCR sensing of inductor current. If the time constants of the bulk capacitance branch and the sense RC branch are matched, then the voltage across the sense resistor will be proportional to the instantaneous currents through the capacitors. To ensure that we obtain a significant signal, the quality factor of the capacitor in the sense branch has to be significantly higher than that of the bulk capacitance branch.

Arranging the sense resistors in the manner shown in Figure 4.8 ensures that the common modes of all three (inductor and capacitor currents) sense nodes are at the same value (v_{out}). A summing amplifier can then be used and the gains tuned to obtain an estimate of the load current. Simulation results in Figure 4.9 show that indeed this method of estimating load current

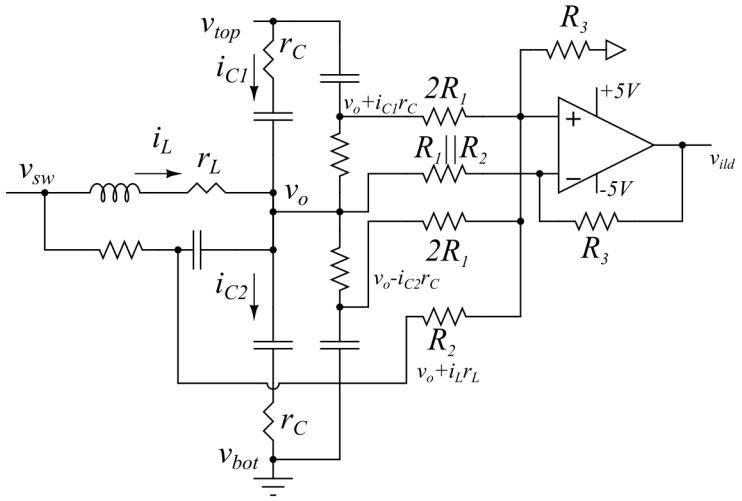


Figure 4.8: Load current estimation of DPP converters

and using it for control works satisfactorily to a certain extent on a single DPP converter. This sensing arrangement is however very sensitive to noise, since the signals are in the range of millivolts (as ESR of several paralleled ceramic capacitors are in the range of a few milliohms). High-frequency noise due to presence switching of several converters on the same board causes this measurement method to fail.

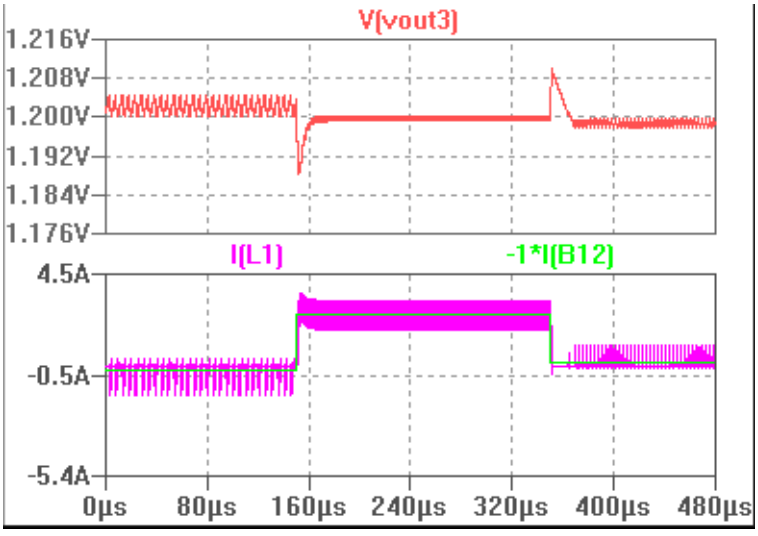


Figure 4.9: Hysteretic control with load current sensing

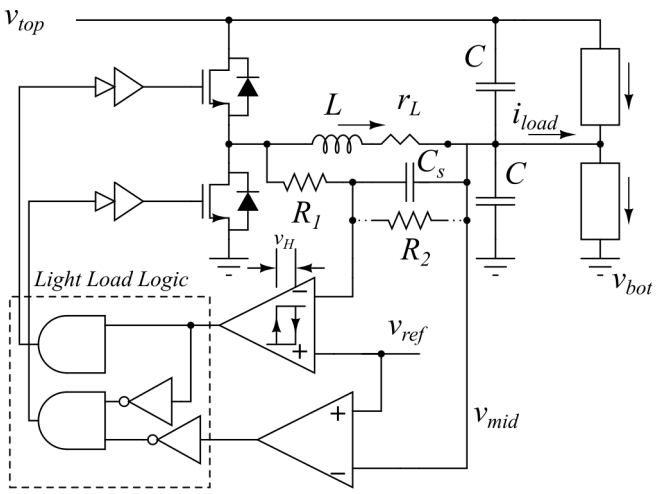


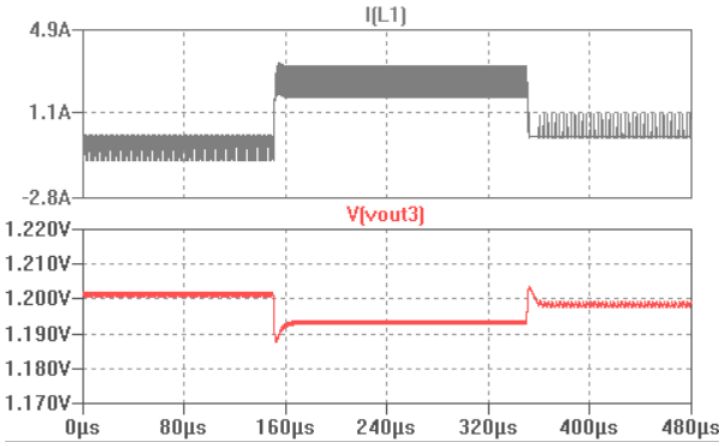
Figure 4.10: Droop controlled DPP

4.4.2 Droop Based Detection of Current Direction

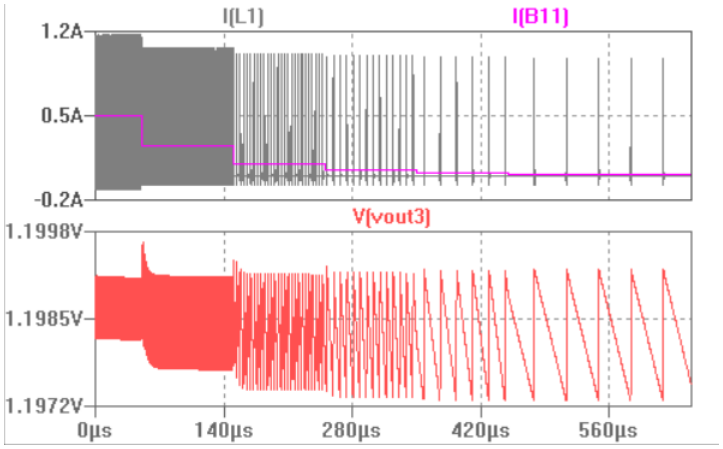
A different way to sense the output current direction is by causing a droop in output voltage. A hysteretic controller which inherently causes a droop in output voltage with load current was discussed in a previous section. Figure 4.10 describes the general theme of the control method. Depending on the sign of the voltage droop with respect to the reference voltage, the direction of load current is determined.

The method works perfectly for a single DPP converter regulating the voltages of 2 stacked voltage domains (Figure 4.11) until a certain limit of light load. If the load current is low enough that the droop voltage is lower than the ripple voltage, the light-load operation of the converter becomes irregular (Figure 4.11(b)). The limit at which this happens can be lowered by decreasing the output voltage ripple (increasing output capacitance) or by using another low-pass filter for droop voltage sensing. Both of these methods have adverse impact on the dynamic performance of the converter; however, this method does not fail to maintain regulation even at arbitrarily light loads.

The droop voltage may be reduced by using a resistor R_2 in parallel with the current sensing capacitor. Instead of the output voltage settling at $V_{ref} - i_L r_L$, the output voltage will settle at $V_{ref} - i_L r_L \frac{R_2}{R_1+R_2}$. Droop based load current detection may be useful for a few reasons in case of our series connected system of processors. The droop based control scheme gives



(a) Droop hysteretic control transient response



(b) Droop hysteretic control light-load (0.5A to 10mA)

Figure 4.11: Hysteretic droop control

us an easy way of deciding which processor cores are consuming more power than the others. In a series-stack of multiple processor cores and with a balancing circuit such as the load-to-load DPP circuit, the supply voltage of the core which consumes the highest current will be the lowest. A computational load balancing circuit may use this information to balance power consumption more efficiently. If the droop is not desirable it can be eliminated by using an external PI controller to adjust the reference voltage. The deviation of v_{ref} from the reference at zero load current can be used as an estimate of the load current. The droop control shown here can be represented as a sliding mode controller with a planar surface (Equations 4.2 and 4.3). The sliding surface is

$$\sigma(\mathbf{x}) = k(v_{ref} - v_o - i_L r_L) = k(v_{ref} - v_o) - k r_L i_L \tag{4.9}$$

Eliminating the droop with a PI controller will modify the above equation to

$$\sigma(\mathbf{x}) = k(v_{ref} - v_o) + k r_L \left(k_{i,v} \int (v_{ref} - v_o) - i_L \right) \tag{4.10}$$

However, instead of eliminating the droop in this way hysteretic current mode control offers a more straightforward solution. It also offers us a way to decouple the current and voltage gains in the expression of Equation 4.5 and makes it independent of r_L.

4.4.3 Hysteretic Current Mode Control

Hysteretic current mode control, where both the peak and valley limits of inductor currents are regulated, is a suitable controller that can be used for our purposes. Unlike peak current mode control, hysteretic current mode control does not require slope compensation for stability. A control current value is generated from the output voltage error of the converter and the switching takes place when the inductor current touches the hysteresis band values around that control current. If the control current is positive then the switching signal to the bottom switch is kept pulled down and vice versa. The controller is described in Figure 4.12. Although the DCR sensing method used in the droop control circuit (Figure 4.10) could be used to sense

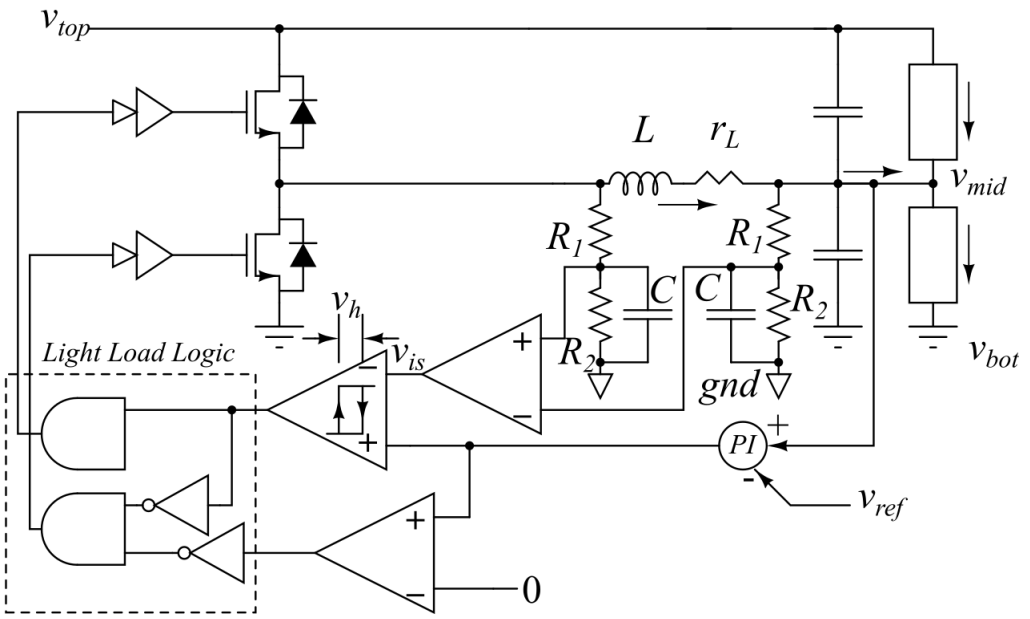


Figure 4.12: Current hysteretis controlled DPP

the inductor current differentially the circuit in Figure 4.12 has several advantages. The low-pass filtering action at both the switching and the output node cuts off any high-frequency noise from entering into the amplifier. Since there is more than one DPP converter connected to the output, the low-pass filtering at the output node is very helpful in obtaining a clean current sense output. Another advantage obtained from using this configuration is that the common mode voltage of the sensing nodes can be reduced so that we can place all current sensing amplifiers on the same voltage domain. The current sense output obtained from the amplifier is given by

$$V_{is} = \frac{i_L R_2 g\,(r_L + Ls)}{(R_1 + R_2)\,(1 + R_1||R_2 Cs)} = A i_L \frac{\left(1 + \frac{L}{r_L}s\right)}{(1 + R_1||R_2 Cs)} \tag{4.11}$$

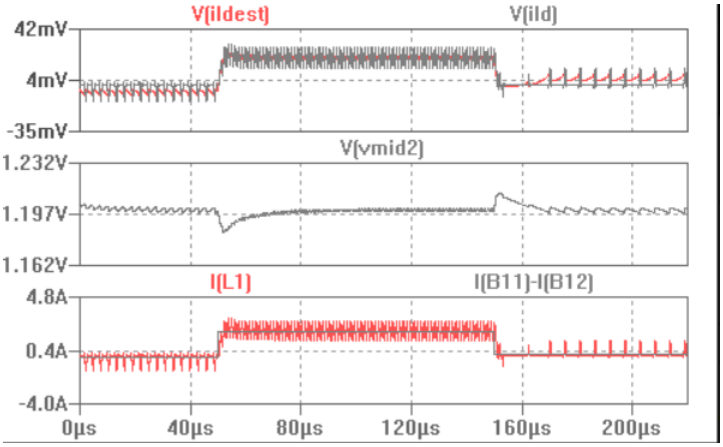
where g is the amplifier gain.

The pole and zero in the expression can be canceled out to obtain the exact inductor current waveform. An interesting aspect of this controller is that it is not necessary to have an exact sensing of the inductor current. The pole and zero can also be placed to improve the stability of the controller. The advantage of using current mode control is that the relationship between inductor current and the output voltage (capacitor voltage) is first order (second order if the ESL of the capacitor is considered). So even a

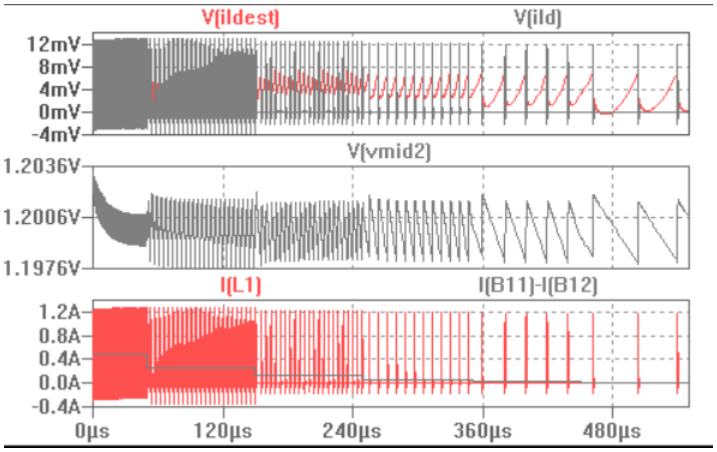
PI controller can provide exceptional transient response. Assuming that we sense the exact inductor current by matching the time constants of the RC integrator and the inductor, the sliding surface obtained by using this control method can be written as

$$\sigma(\mathbf{x}) = k_p(v_{ref} - v_o) + A\left(\frac{k_{i,v}}{A}\int(v_{ref} - v_o) - i_L\right) \tag{4.12}$$

Figure 4.13 shows simulation results for bidirectional light load operation of a single DPP converter under hysteretic current mode control.



(a) Transient response



(b) Light-load (0.5A to 25mA)

Figure 4.13: Hysteretic current mode control

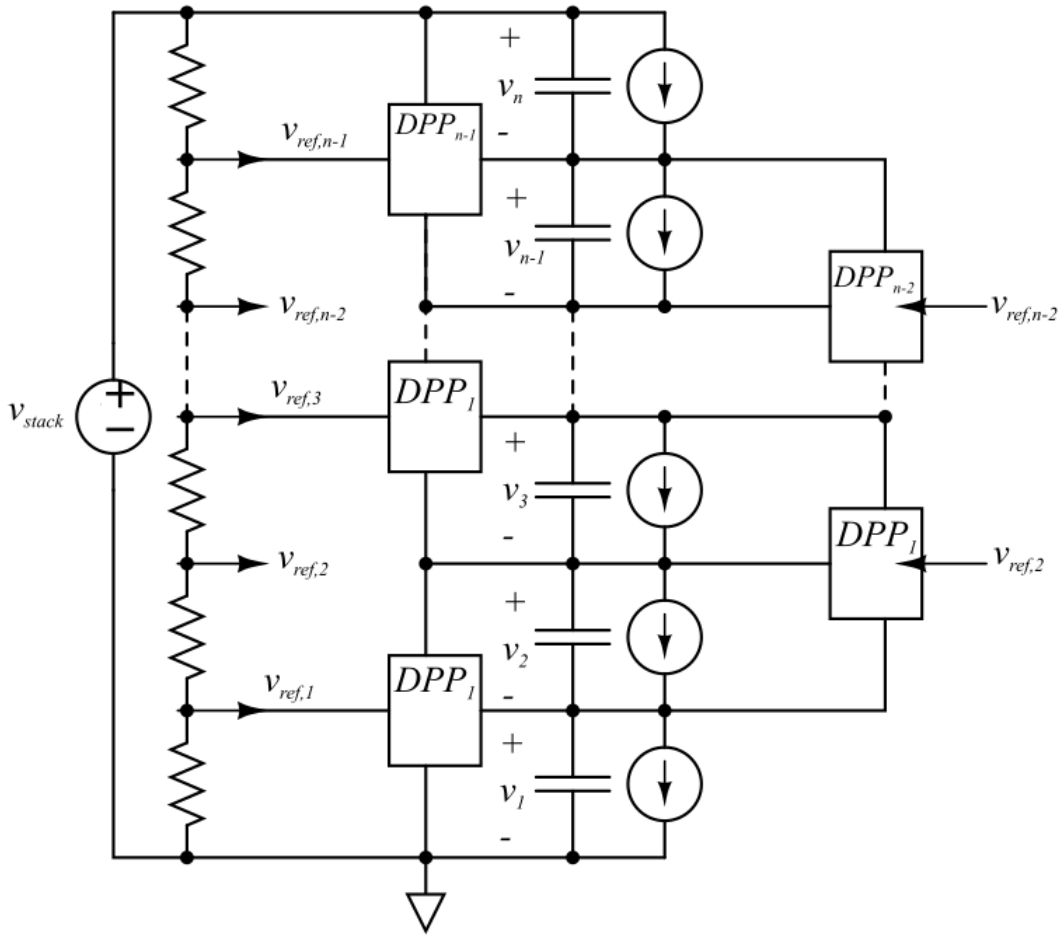


Figure 4.14: Current hysteretis controlled stack

4.5 Improved Voltage Regulation of the Stack

Current mode control provides us with improved input noise rejection due to the full-state feedback nature of the controller. It also improves voltage regulation of converters with right half-plane zeros, as is the case of our DPP converters. Taking advantage of that fact we can now set up the voltage regulators to regulate the voltages at each node to which they are connected to a fixed value, rather than to equalize the voltages of adjacent voltage domains. The new control scheme is shown in Figure 4.14.

The improved transient responses obtained with hysteretic current mode control (without light-load modes and with a high static gain proportional only current estimator) are shown in Figure 4.15. A 1 μH inductance in each DPP converter and 100 μF of capacitance per voltage domain were used in the simulations. A step transient (mismatch current) was created at the mid-

dle of the stack for 6 different stack heights. The proportional gain gives rise to voltage droops across domains (steady state error). The noticeable aspect here is that the deterioration of bandwidth previously obtained with voltage mode control as the number of domains in the stack was increased is not observed here. The domain voltage droops cause input voltage to the DPPs to be different and this causes frequency variation on our DPP converters. Some beats due to converters operating at slighty different frequencies can be observed in the the 4-domain case. The PI controller does not give us the same problem, but now the current estimator bandwidth becomes lower than with the proportional controller. The simulations shown in Figure 4.16 show elimination of droop and beats and slower transient response. The regulator still settles the domain voltages within 10 μs. From the discussion on transient repetition rates in Chapter 1 we can estimate that the maximum repetition rates at which the DPP converters need to step up current outputs may range in hundreds of kHz. Depending on the maximum possible mismatches the converter inductances and domain capacitances can be appropriately scaled.

Further, the current hysteretic controller with a PI current estimator used here can be used to operate the converters in bi-directional light-load modes. Figure 4.17 shows bi-directional light load modes operating in stacks of increasing number of elements. It can be seen that the transient responses are comparable to what was obtained in FCCM. Significantly reduced switching action can be seen when the mismatches are low. With increasing number of elements in the stack, however, the DCM behavior becomes more irregular, and can be taken up as a subject of further research.

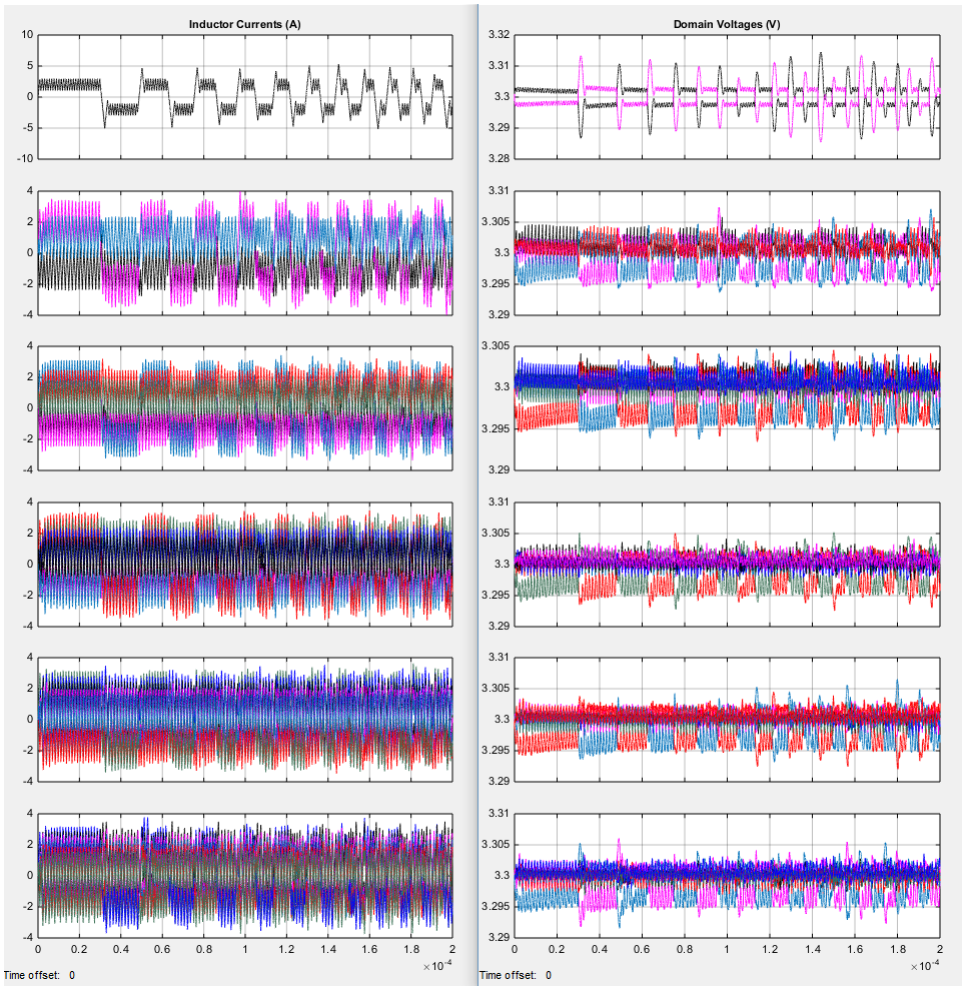


Figure 4.15: Current hysteretis controlled stack - high static gain current estimator; number of stacked elements increases from top to bottom (2 at top, 12 at the bottom)

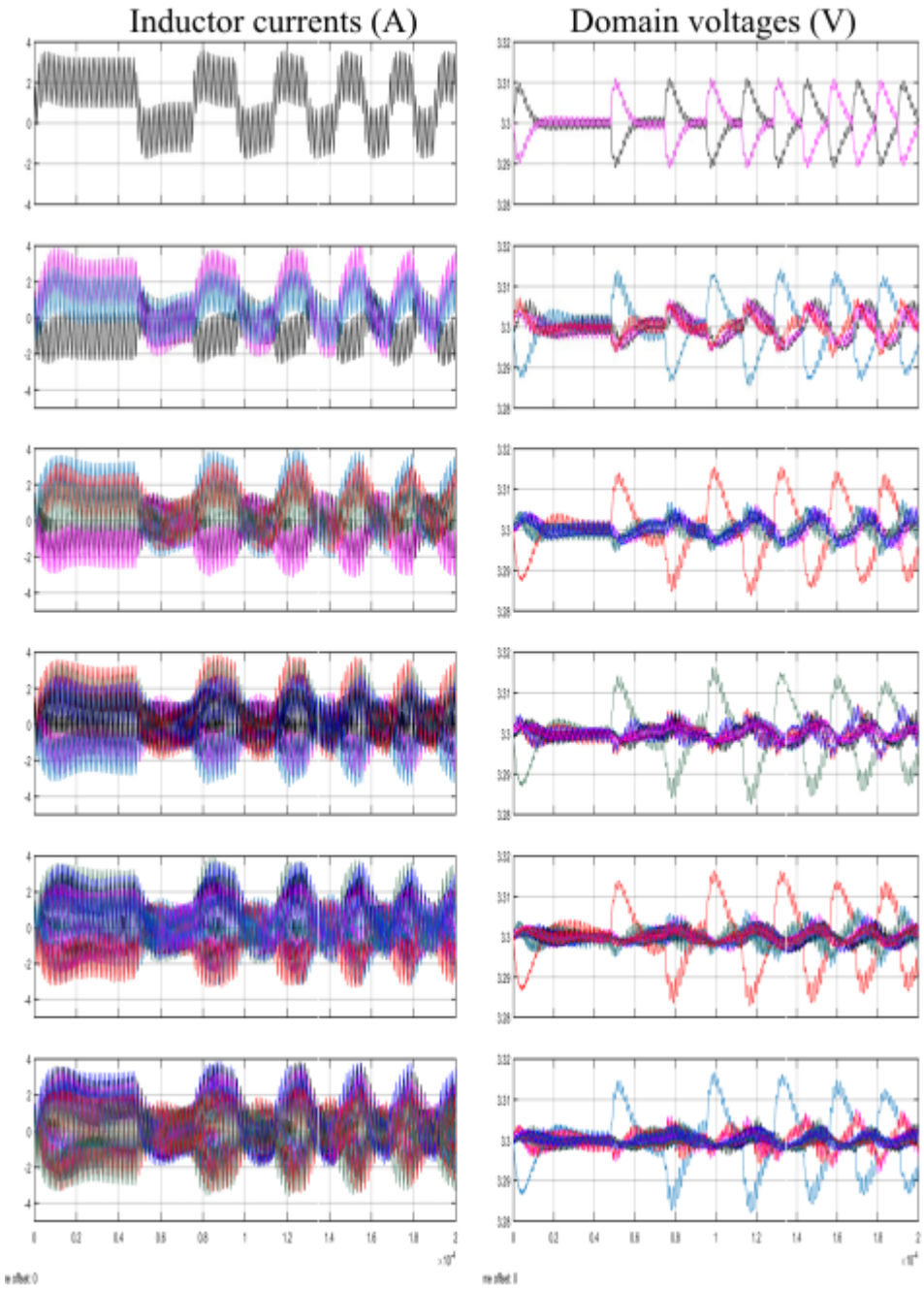


Figure 4.16: Transient responses to a step in mismatch current for a current hysteretis controlled stack in FCCM - PI controller in current estimator; number of stacked elements increases from top to bottom (2 at top, 12 at the bottom)

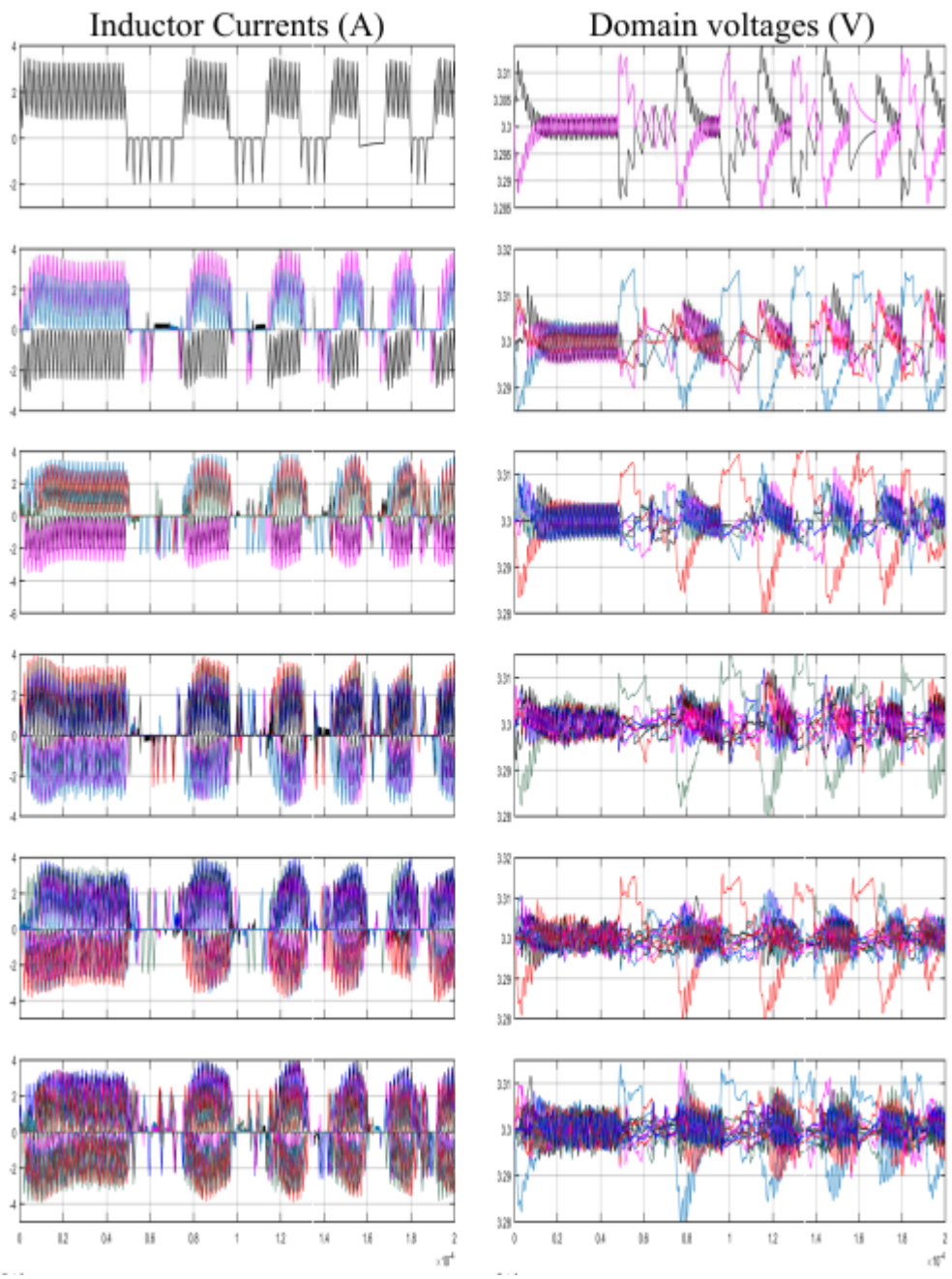


Figure 4.17: Transient responses to a step in mismatch current for a current hysteretis controlled stack - bidirectional light-load modes enabled; number of stacked elements increases from top to bottom (2 at top, 12 at the bottom)

CHAPTER 5

HARDWARE VERIFICATION OF HYSTERETIC CONTROL

The hardware to test hysteretic current mode control on DPP converters with 4 stacked domains was set up. The power circuit was set up as in Chapter 3 with slight differences. With the UCD7242 driver with integrated MOSFETs we did not have the capability to operate in negative light load (only FCCM or diode emulation in the lower switch was possible). The UCD74111 driver used in this design was suitable for our application because it allows independent control of the high and low side gates. Dead-time has to be programmed in software since the DrMOS does not provide any anti-cross-conduction circuitry when the independent gate drive operation is selected. The TMS320F28335 DSP that was used in the previous circuit was replaced by a TMS320F28377S DSP as it offers several analog comparator channels. Each of the comparator input pins in the DSP is connected to 2 comparators. The comparators can either derive their negative input from the negative input pins or from internal 12-bit DACs. The node voltages in the stack are sensed by 4 ADC channels (simultaneous sampling is possible since there are two separate ADC modules in the DSP) at a sampling rate of 1 MHz. The PI controller generates a reference value and writes it into the DAC of the comparator module. The comparator module outputs and PWM are set up in the same configuration as Figure 4.2. Two separate SR latches generate the PWM signals on channel A and channel B. If the reference value generated by the PI controller is positive then the PWM on channel B is reconfigured so that it remains pulled down. The same operation is done on channel A if the reference generated by the PI controller is negative. The C-code used to program the microcontroller is provided in Appendix C. A high-level schematic of the hardware setup is shown in Figure 5.1. Figure 5.2 shows the top and bottom layers of the PCB.

The DPP modules were set up to output 3.3 V at each voltage domain in the series stack. A 1 μH inductor with 15 mΩ of series inductance was used in

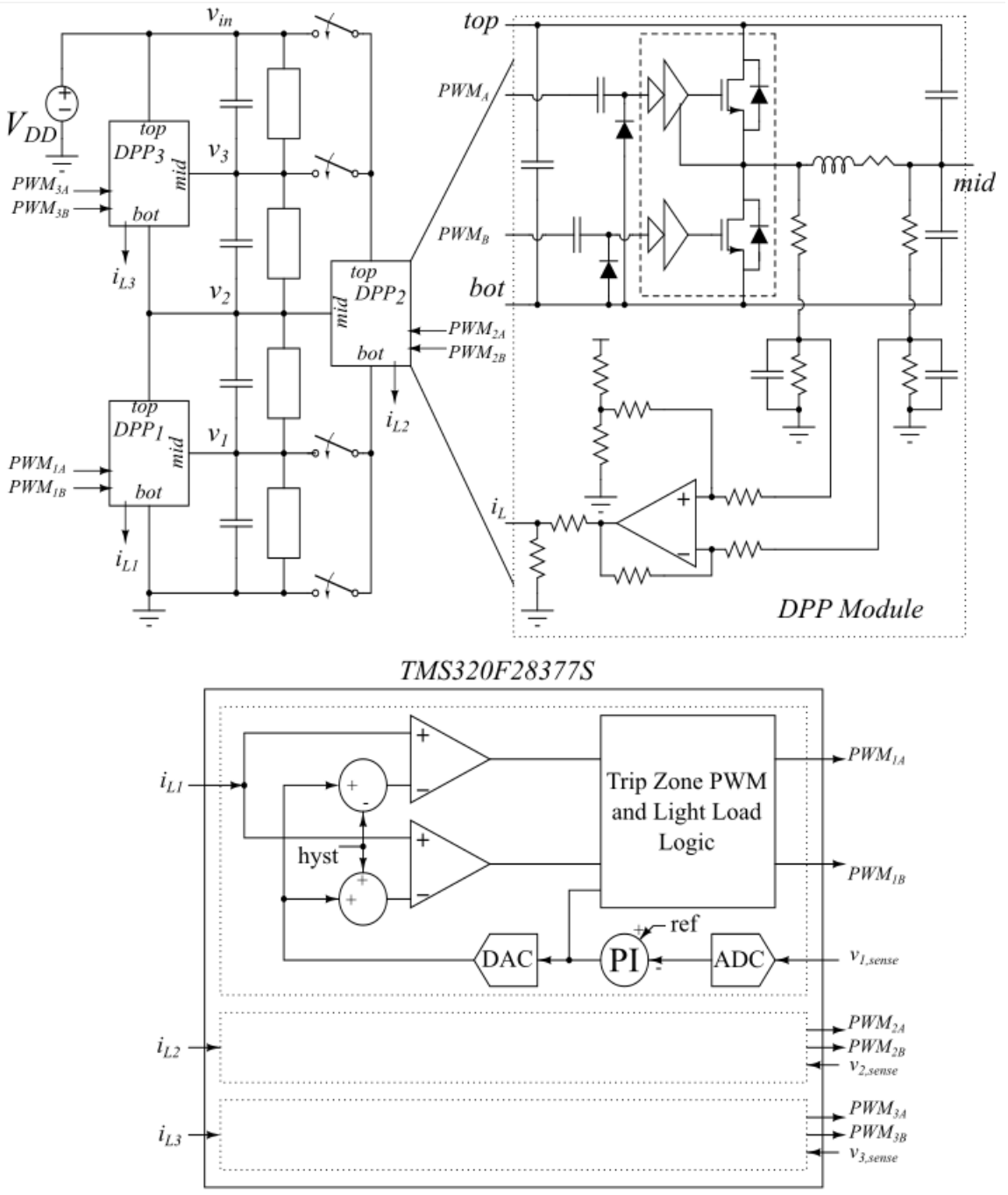
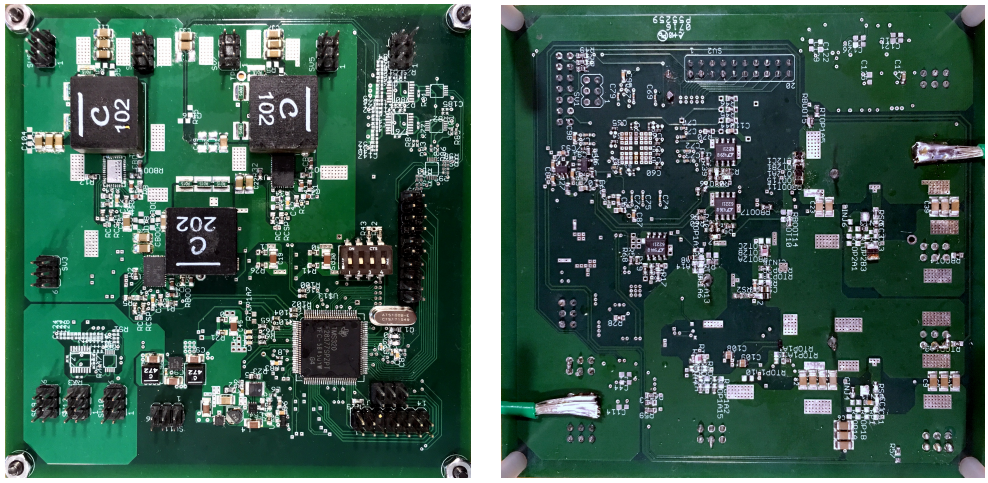


Figure 5.1: Hysteretic current mode control hardware setup



(a) Top layer (b) Bottom layer

Figure 5.2: DPP board with 3 converters

each DPP module. Ceramic capacitance at 66 μF (rated at 16 V and derates to about 50 μF at 3.3 V) was used at each voltage domain. Film capacitors were used for the RC integrators at the current sensing nodes. The common mode voltages at the inputs of the three current sense amplifiers were set to 1.1 V, 2.2 V, and 3.3 V by using the resistor divider configuration. The current sensing amplifier was set to provide a gain of 25 so that the output of the current sense amplifiers give us $\frac{i_L}{8}$ with a 2.5 V offset. The time constants of the integrators were set up such that the inductor current ripple to average current ratio is 0.5 times that at the sensed node. This gives us twice the comparator resolution (since we are using a DAC to compare the inductor current) at the cost of slightly slower transient response. The efficiency of a single DPP converter operating in light load enabled mode (with either the high side or low side switch as a diode) and the same converter under FCCM (both switches MOSFETs) is shown in Figure 5.3. Transient responses and bidirectional light-load behavior of a single DPP converter are shown in Figure 5.4. We can see that the light-load efficiency of the converter is significantly improved with respect to that obtained with FCCM (under 2 A), even with body diode conduction. But body diode conduction lowers the efficiency of the converter at higher load current. Implementing diode emulation was not possible in this simple test setup because it would require additional switch current zero crossing detection circuits on both the high side and the low side switch, which are difficult to implement without an

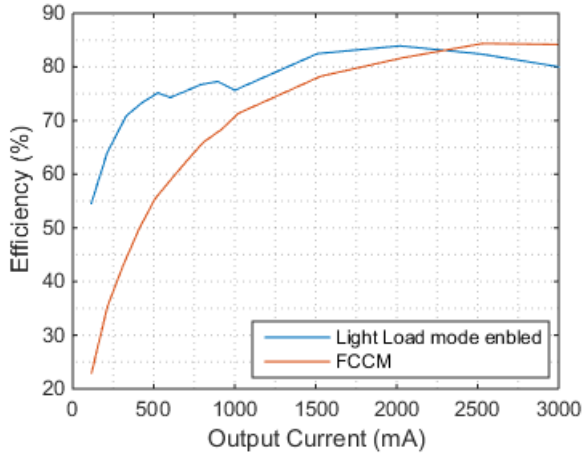


Figure 5.3: Individual DPP converter efficiency plot in FCCM and with light-load mode enabled, $L = 1$ μH, $f_{sw} = 500$ kHz

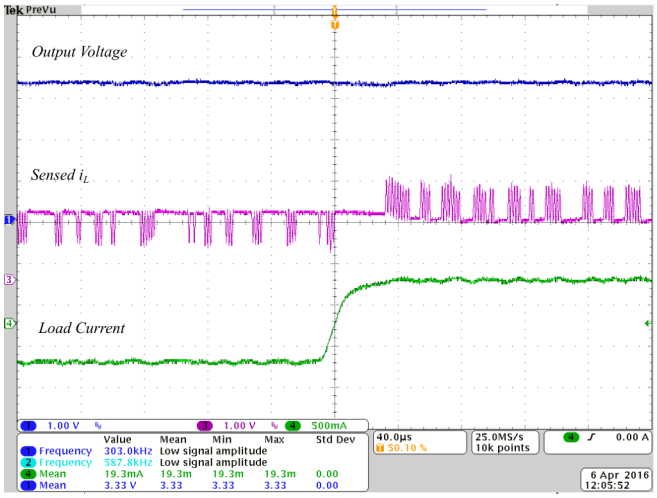
integrated solution.

The nominal stack current was set at 5 A and the currents in the domains were operated first at a mismatch of 10% (0.5 A deviation from 5 A) and then at 20%(1 A deviation from 5 A). The overall system level efficiencies obtained with FCCM and light-load mode enabled, under the two mismatch conditions mentioned above, are shown in Table 5.1.

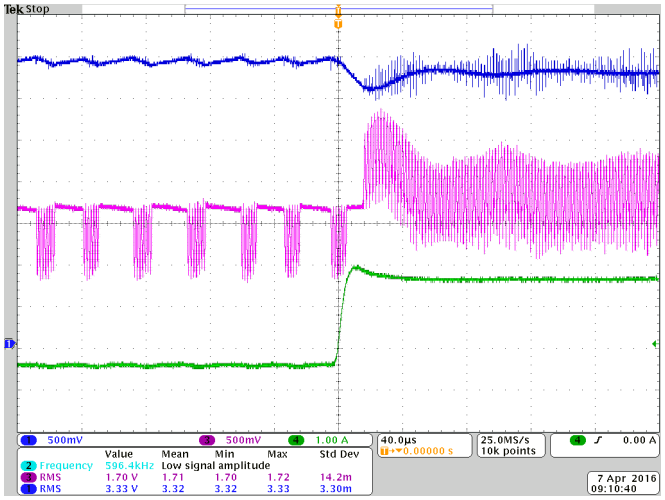
Transient responses validating operation of converters in bi-directional light load enabled mode for a system of 4 series-stacked domains are shown in Figure 5.5. It can be seen that the transients are not as smooth as was obtained when a single converter was tested. The reconfiguration of PWM channels in the DSP takes several clock cycles and when three PWM channels have to be reconfigured simultaneously the operation becomes even slower which causes an instability in the stack voltage regulation. However, if a dedicated controller is built for this operation, PWM reconfiguration can be implemented with logic gates only and this sort of glitch will not occur. Steady state operation under a few test cases is shown in Figure 5.6.

Table 5.1: Averaged efficiency of the element-to-element topology, nominal stack current = 5 A

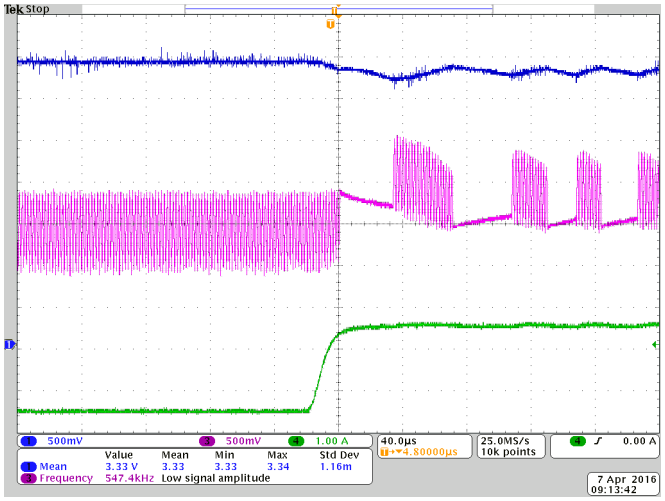
Maximum % mismatch	FCCM f_{sw} =500 kHz	Light-load f_{sw} =500 kHz
10%	95.2	97.3
20%	94.8	96.5



(a) Load current change from -0.5A to 0.5A

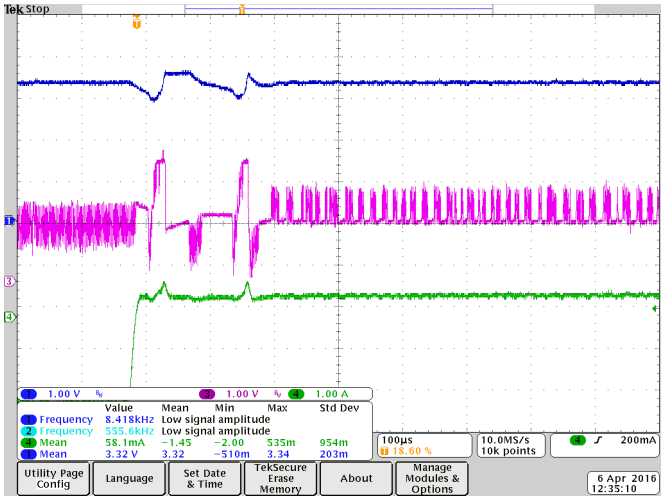


(b) Load current change from -0.5A to 1.5A

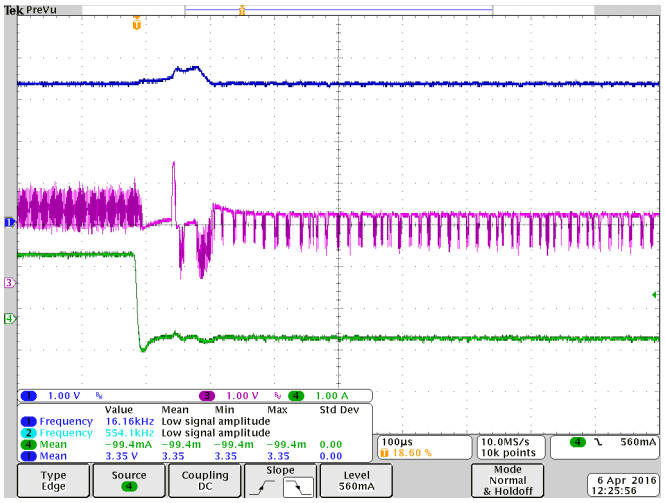


(c) Load current change from -1.5A to 0.5A

Figure 5.4: Bidirectional light-load modes in a single DPP converter

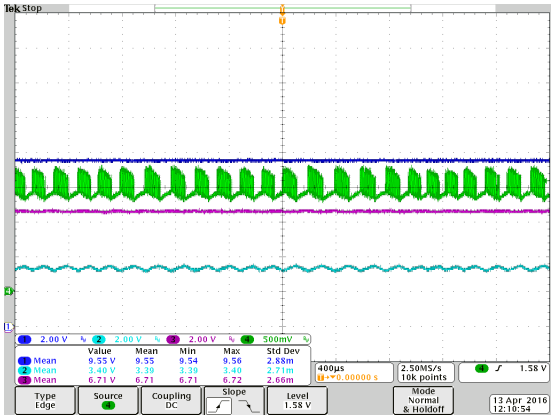


(a) Load current change from -1.5A to 0.5A

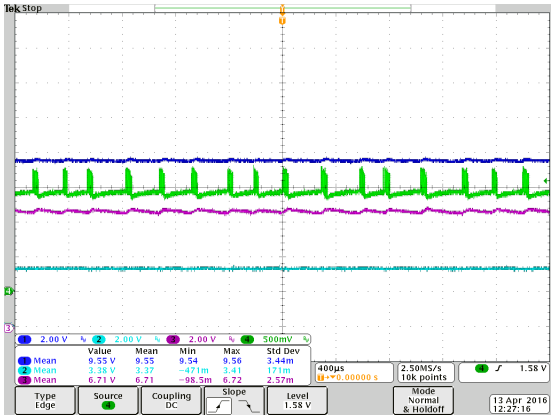


(b) Load current change from 1.5A to -0.5A

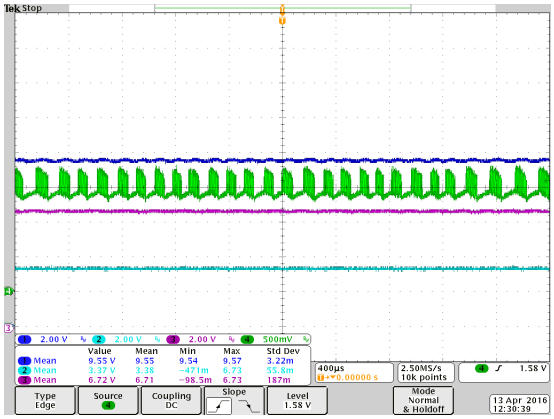
Figure 5.5: Transient response of a single converter (bottommost) operating in a stack of 4 voltage domains



(a) Lowermost converter operating in DCM, green: sensed inductor current of the lowermost converter



(b) Topmost converter in negative DCM and converter at the middle of the stack in DCM, green: sensed inductor current of the latter



(c) Topmost converter in DCM, green: sensed inductor current of the topmost converter

Figure 5.6: Light-load modes operating in a stack of 4 voltage domains

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 Conclusion

This work demonstrates that voltage regulation of a series-stacked system of processor cores with bi-directional buck-boost converters is indeed an attractive solution to replace conventional parallel-connected processor core systems. Efficiencies obtained by series-stacking (93%-96%) even without any light-load control or converters optimized for low currents (high switching frequency) were at least on a par with conventional multi-phase converter power supplies. The dynamic models obtained can be used to develop linear or nonlinear control to achieve the stringent voltage regulation parameters required by modern processors. A hysteretic controller which is capable of driving the buck-boost DPP converters in bidirectional light load was developed and efficiency improvement was verified.

6.2 Future Work

To truly determine the gains of series-stacking, an actual microprocessor load has to be powered using the element-to-element topology. A series stack of FPGA loads can serve as an appropriate digital load although multiple voltage domains higher than the core voltage will complicate the design. The modularity of the element-to-element topology has also to be demonstrated as an attractive feature with regard to core voltage regulation. For this, voltage regulation in a larger stack has to be demonstrated. Suitability of other modular DPP architectures or in core voltage regulation should be investigated further. Multiphase buck converters with coupled inductors can also be used in as the DPP converters. Utilizing the nominal 0.5 duty ratio

of the DPP converters the multiphase buck converters will prove to be much more effective in reducing/eliminating switching ripple from the stack nodes. This also opens up the possibility of getting into light load modes without generating extra switching noise. Bidirectional light load was demonstrated in Chapters 4 and 5 using a current mode hysteretic controller. However this method can be extended to general sliding mode controllers, and non-planar switching surfaces can be considered to improve transient response of the DP converters. This is essential so that we may be able to stack more elements in series without compromising dynamics.

APPENDIX A

SCHEMATICS, LAYOUT PICTURES AND BILL OF MATERIALS

This Appendix includes lists of components, in Tables A.1 and A.2, used in the two PCBs built for hardware verification in Chapters 3 and 5. The schematics and layout pictures of the PCBs built are also provided in Figures A.1 through A.15.

Table A.1: List of components used in the first PCB (linear controller)

Part Description	Part Number	Value
Schottky Diode	1SS416CT	
Clamping Diodes	NUO420MR6	
C2000 DSP	TMS320F28335	
3.3V and 1.8V dual buck regulator	TPS62400	
7-17V in, 5V out buck regulator	LM43600	
3.3V power good monitor	TPS3828-33	
Inductors for TPS62400	VLF3010A	2.2 μH
Dual synchronous buck DrMOS	UCD7242	
DPP converter inductances(1)	XAL6060-472 MEB	4.7 μH
DPP converter inductances(2)	XAL6060-102 MEB	1 μH
DPP converter output capacitors	GRM31CR61C226ME15L	22 μF
33 Ω resistor networks	8R-NEXB2HV-33	33 Ω
50 Ω resistor networks	8R-NEXB2HV-50	50 Ω
Digital Isolator IC	ISO7221C	
SPI communications IC	MAX3221E	
Surface mount Crystal 15Mhz	NX5032GA	
DSP supply decoupling capacitors	CL05B104KO5NNNC	100 nF
DSP supply ferrite bead	732-6708-2-ND(Digikey)	60 Ω (100 MHz)
DIL Switch	CTS-219-04J	
Other 0603 SMD components		

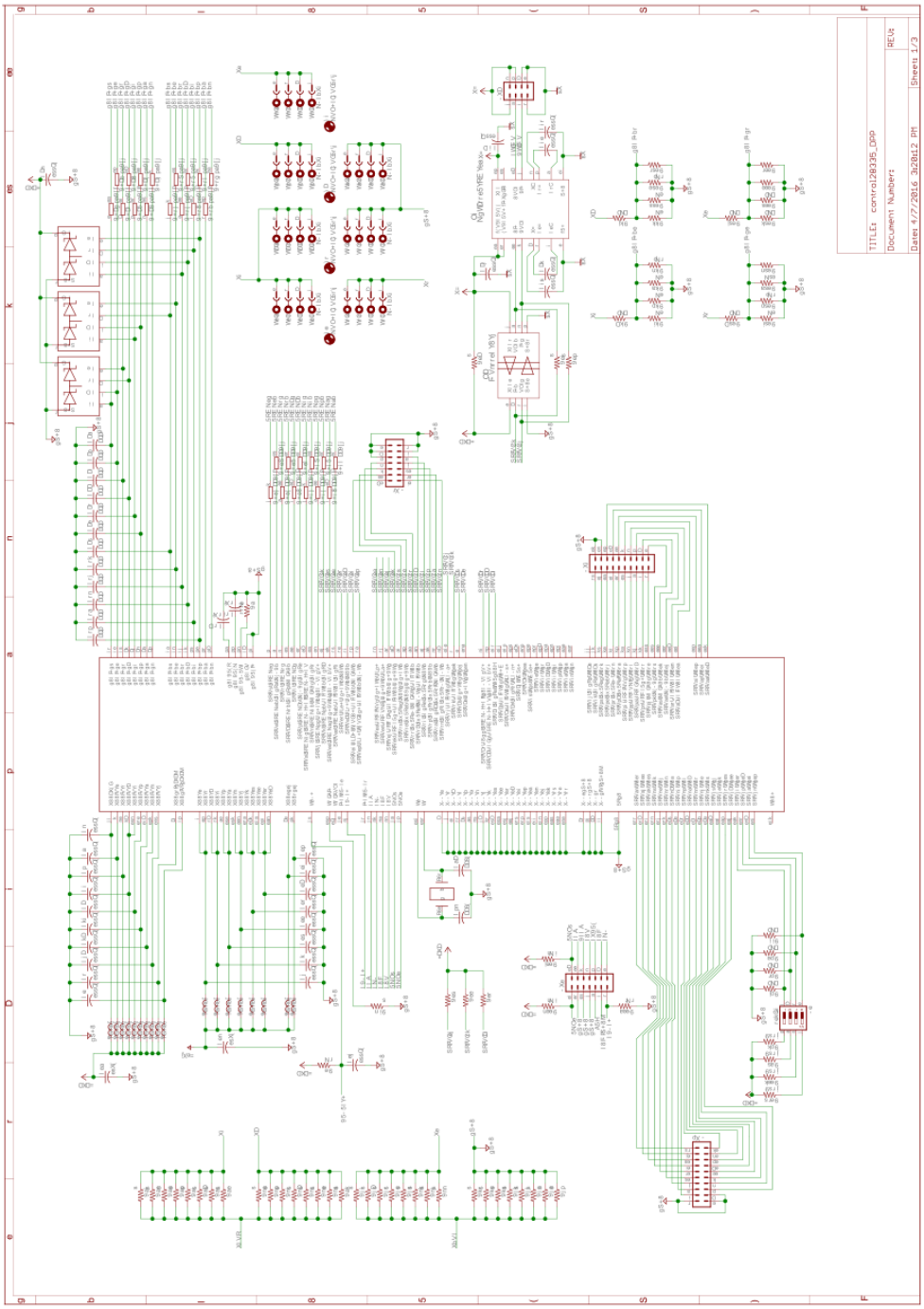


Figure A.1: Linear control board for DPP schematic, microcontroller

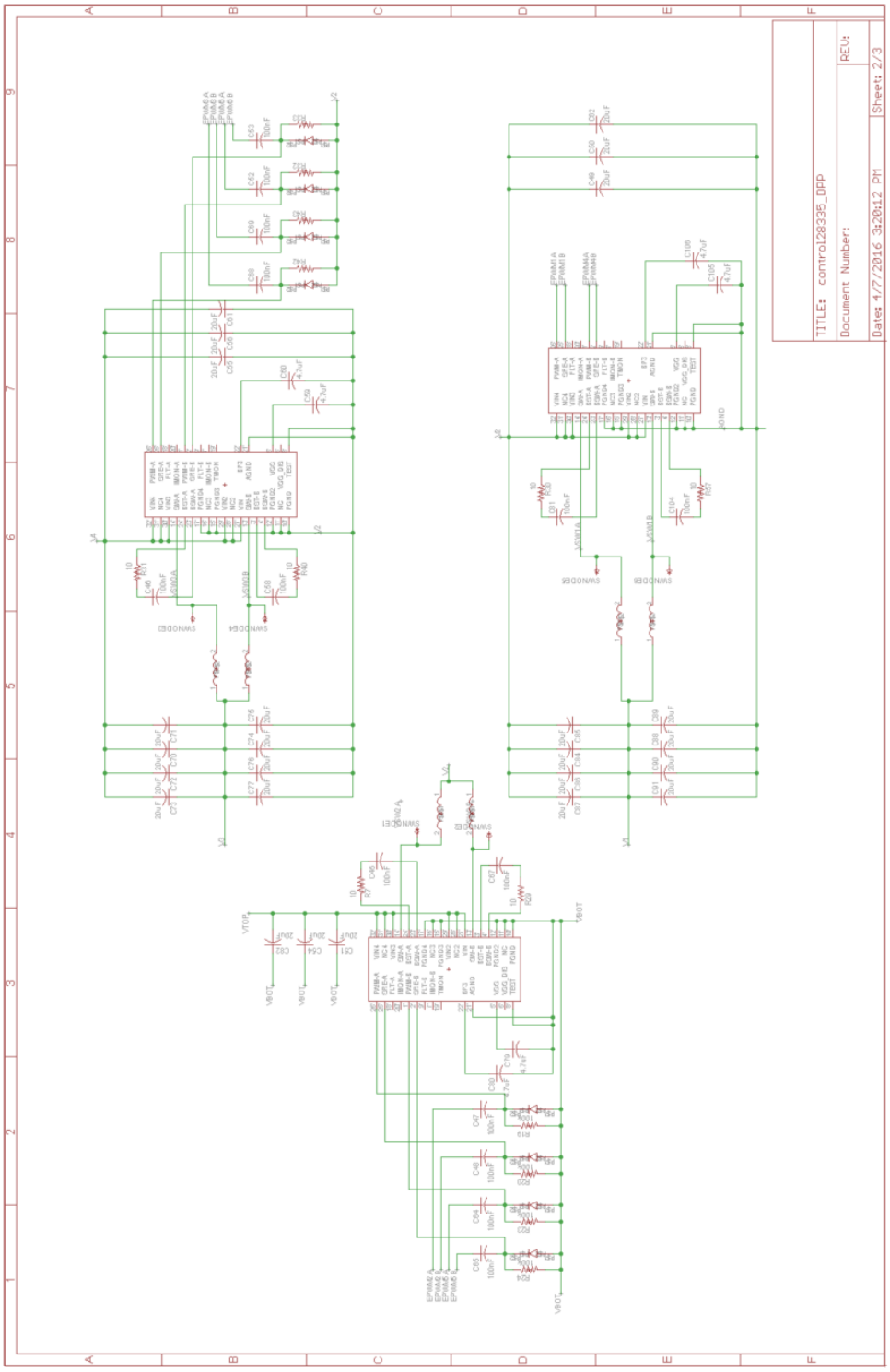


Figure A.2: Linear control board for DPP schematic, DPP converters

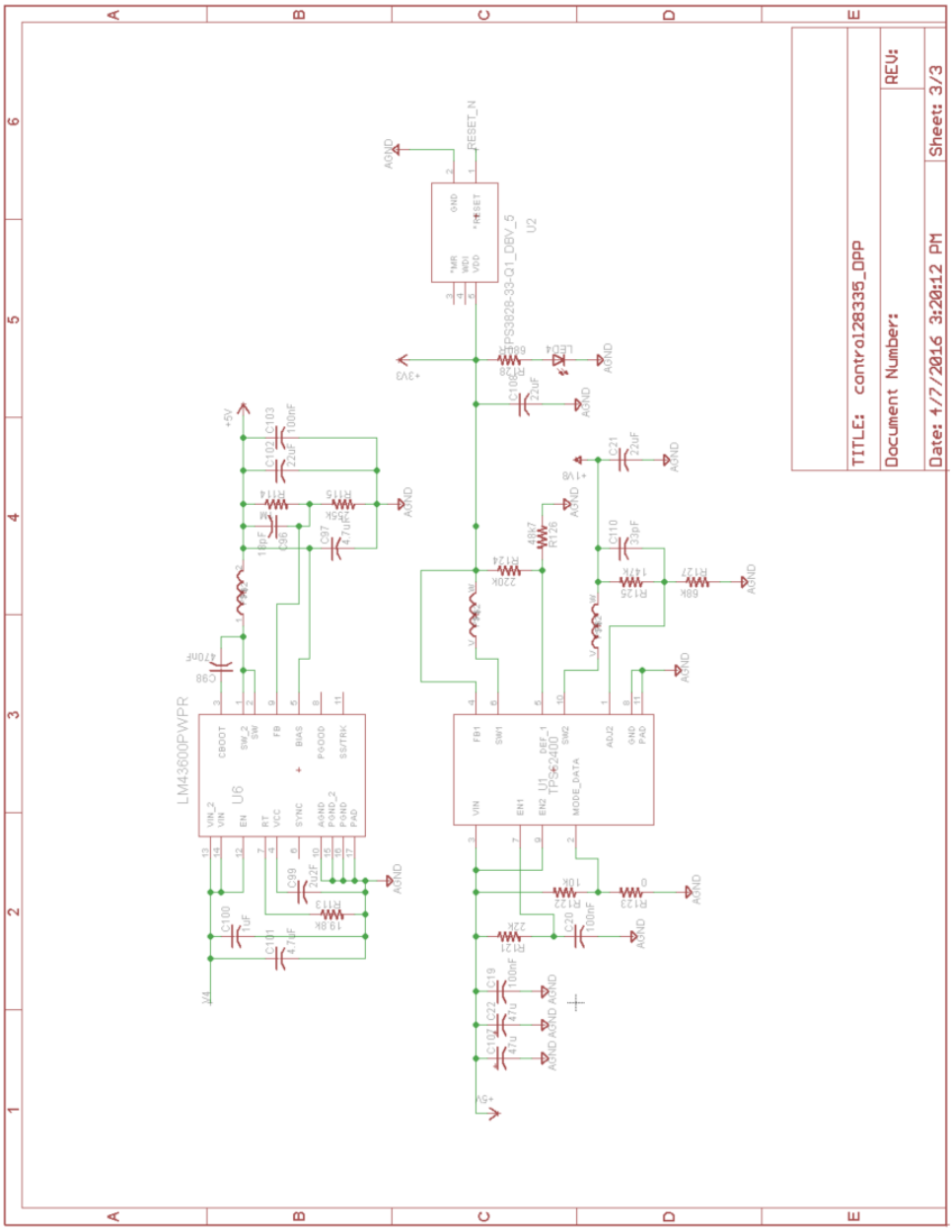


Figure A.3: Linear control board for DPP schematic, microcontroller supply

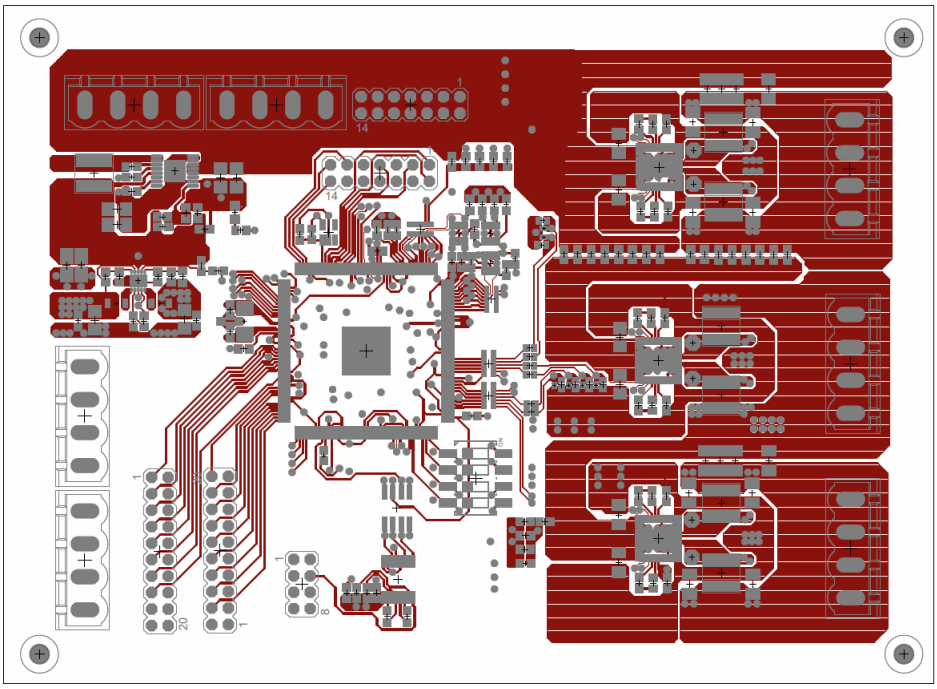


Figure A.4: Linear control board for DPP layout, top layer

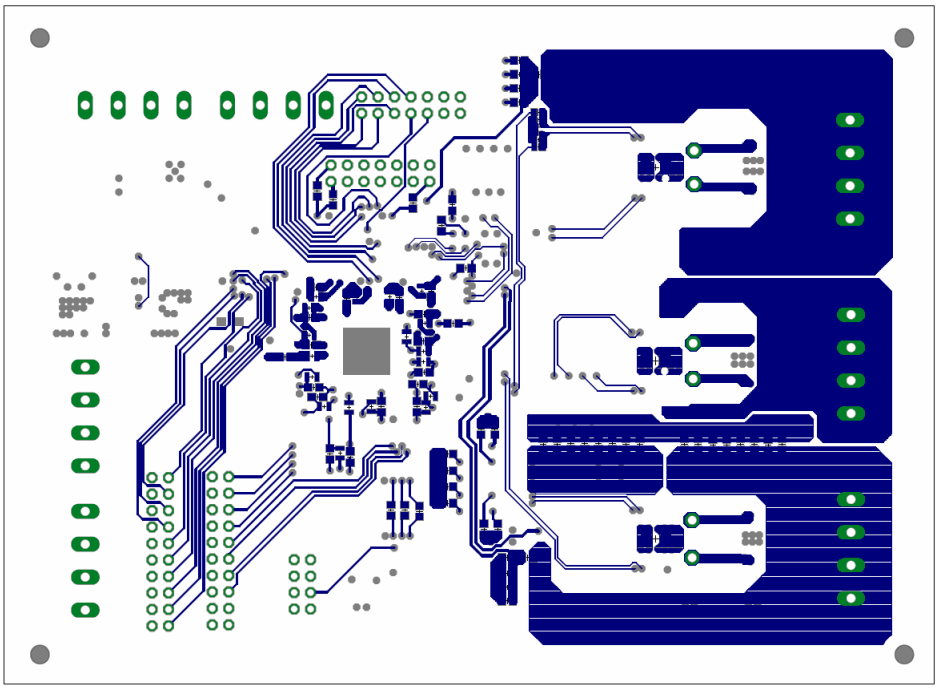


Figure A.5: Linear control board for DPP layout, bottom layer

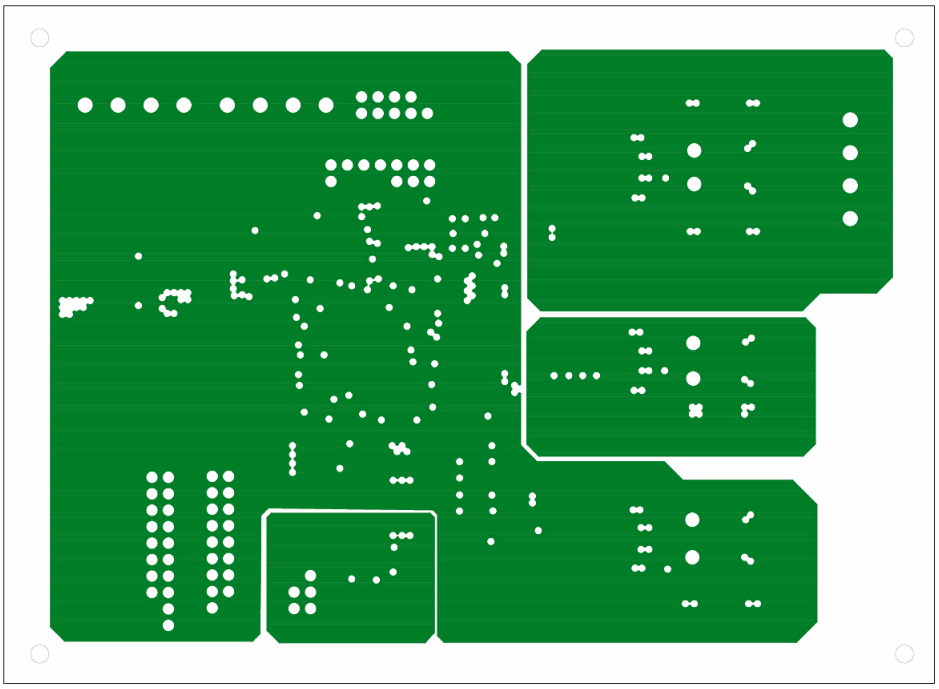


Figure A.6: Linear control board for DPP layout, layer 2 (gnd planes)

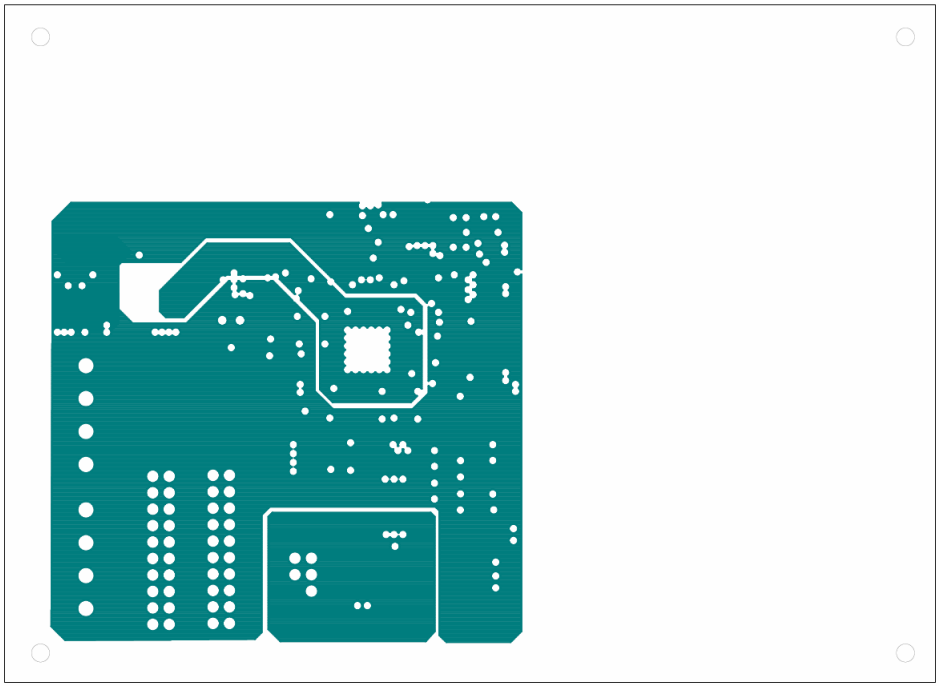


Figure A.7: Linear control board for DPP layout, layer 3 (microcontroller supply)

Table A.2: List of components used in the second PCB (hysteretic controller)

Part Description	Part Number	Value
Schottky Diode	1SS416CT	
C2000 DSP	TMS320F28377S	
3.3 V to 1.2 V buck regulator	TPS62080	
5 V to 3.3 V buck regulator	TPS62162	
5 V to $\pm$5 V regulator	TPS65133	
Inductor for TPS62080	ULQH3NPN1R0NJ0L	1 μH
Inductor for TPS62162	VLF4012A	2.2 μH
Inductors for TPS65133	XFL4020	4.7μH
Supply decoupling chip inductor	BLM15PD600SN1D	60 Ω
Supply decoupling chip inductor	BKP1005EM221-T	220 Ω
Synchronous buck DrMOS	UCD74111	
DPP converter inductances(1)	XAL1580-102 MEB	1 μH
DPP converter inductances(2)	XAL1580-202 MEB	2 μH
DPP converter output capacitors	GRM31CR61C226ME15L	22 μF
Reference generating IC	REF3030	
Opamp	OPA320	20 MHz
Crystal	CTX919-ND	10 MHz
Inductor current sensing opamp	LT6221	60 MHz
Load current sensing opamp	LT6232	200 MHz
Load current comparators	LT1715	2 ns
Thin film resistors	PRL1632	15 mΩ
Film capacitors (1)		0.1 μF
Film capacitors (2)		1 μF
DIL switch	CTS-219-04J	
Other 0603 SMD components		

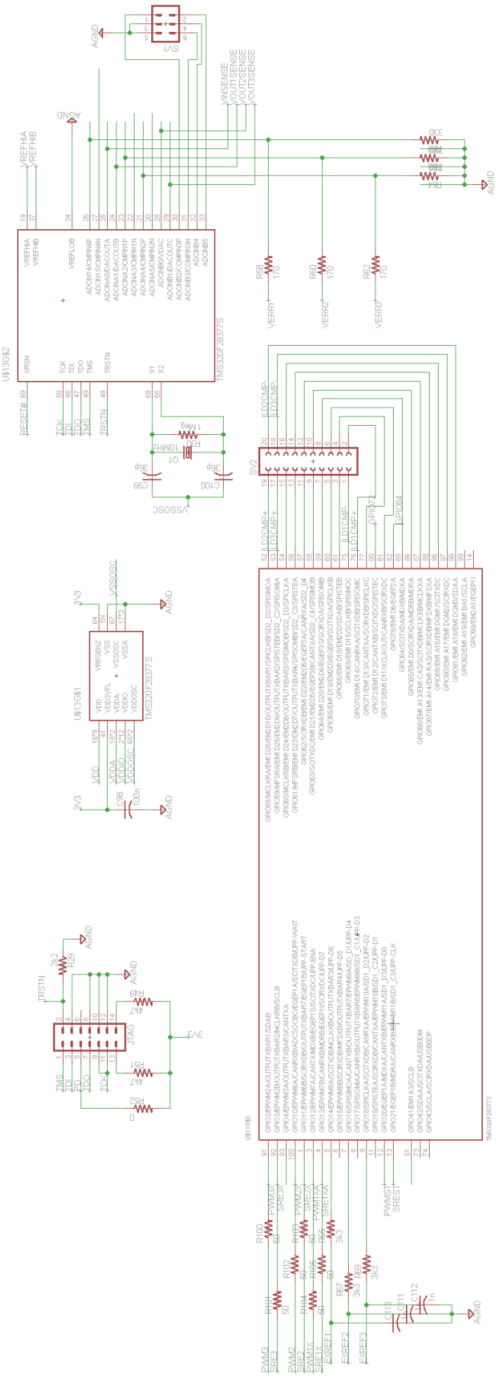


Figure A.8: Hysteretic control board for DPP schematic, microcontroller

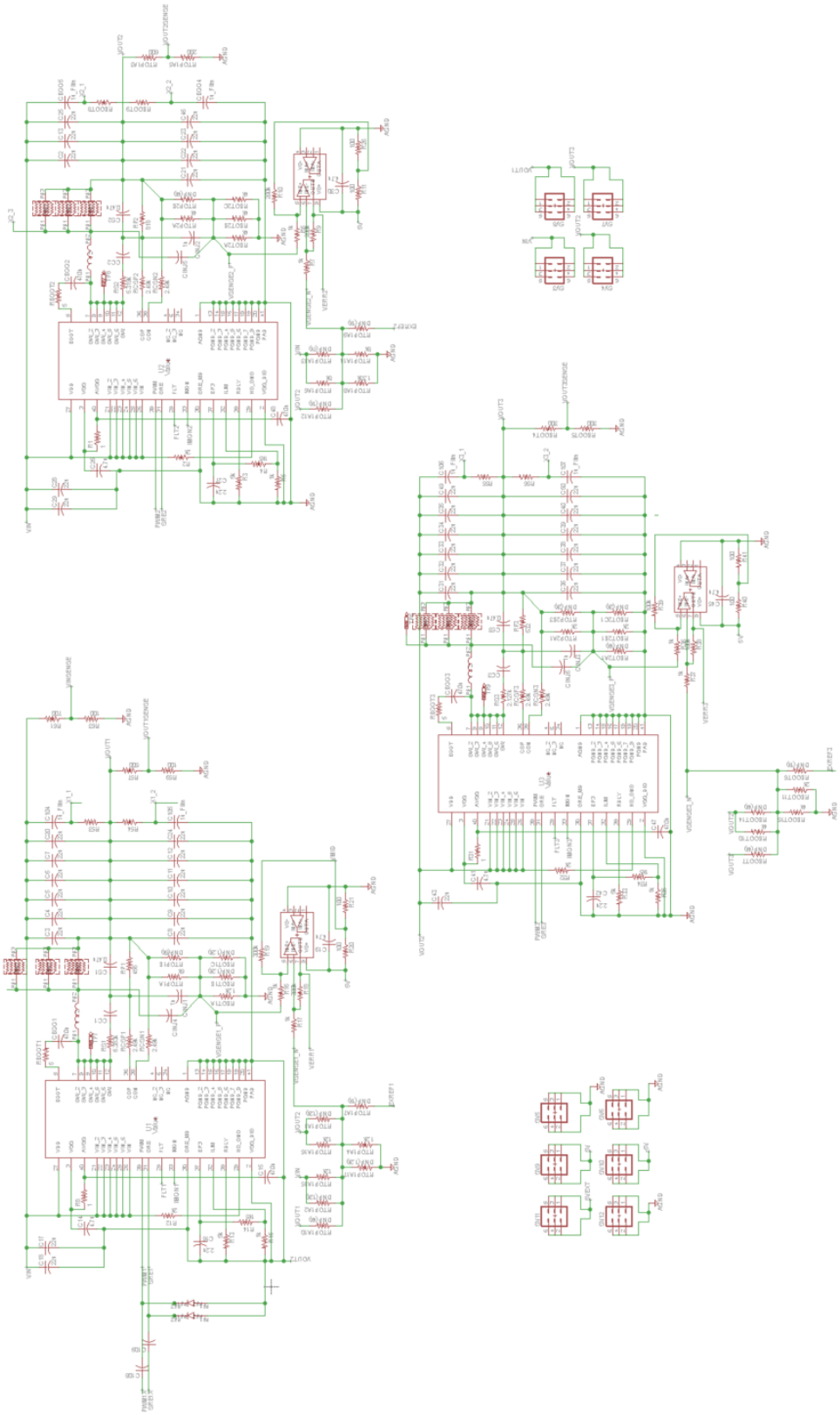


Figure A.9: Hysteretic control board for DPP schematic, DPP converters

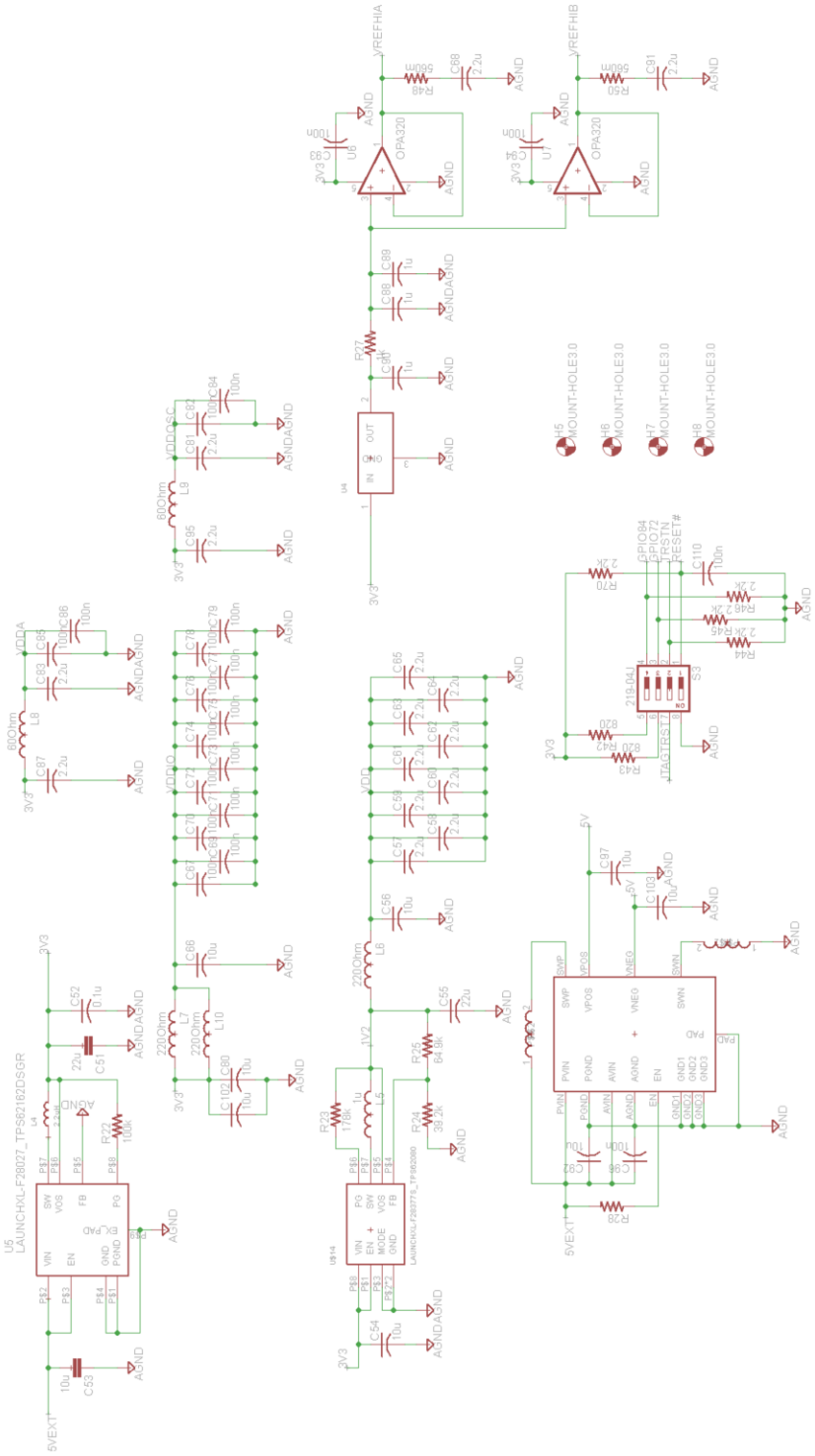


Figure A.10: Hysteretic control board for DPP schematic, microcontroller supply

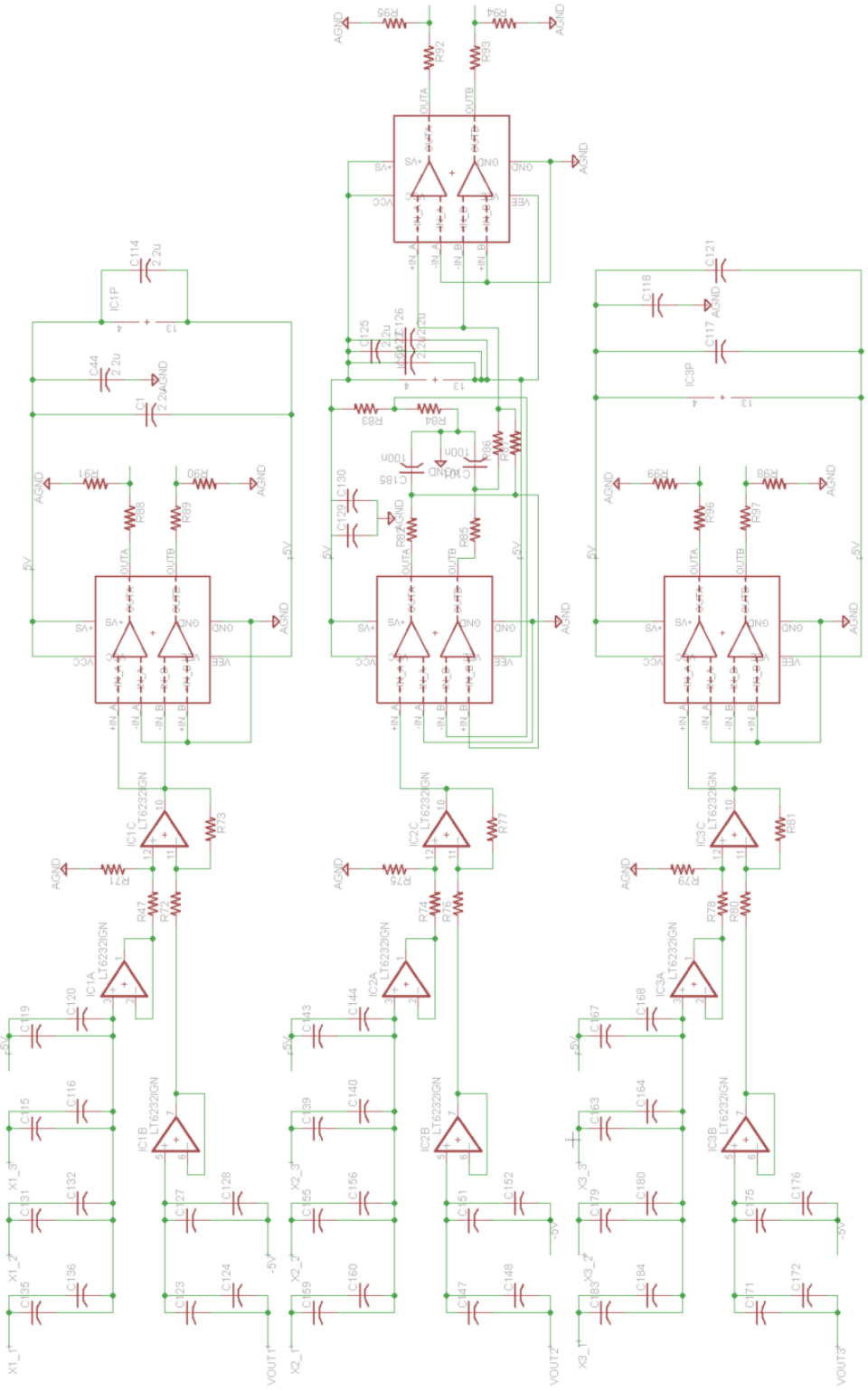


Figure A.11: Hysteretic control board for DPP schematic, microcontroller supply

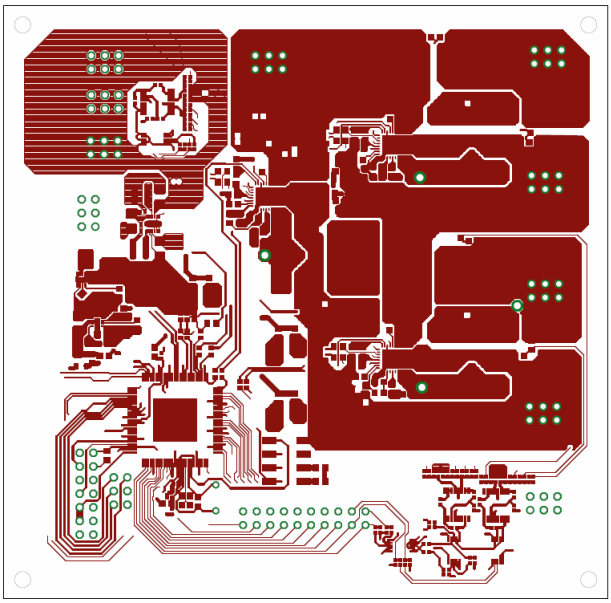


Figure A.12: Hysteretic control board for DPP layout, top layer

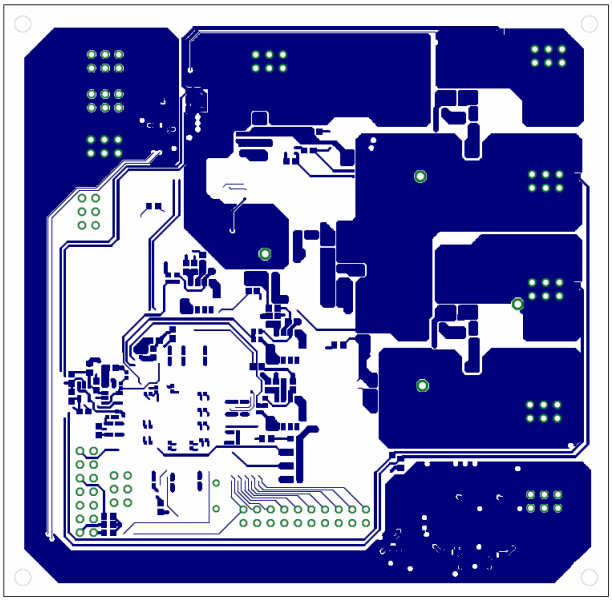


Figure A.13: Hysteretic control board for DPP layout, bottom layer

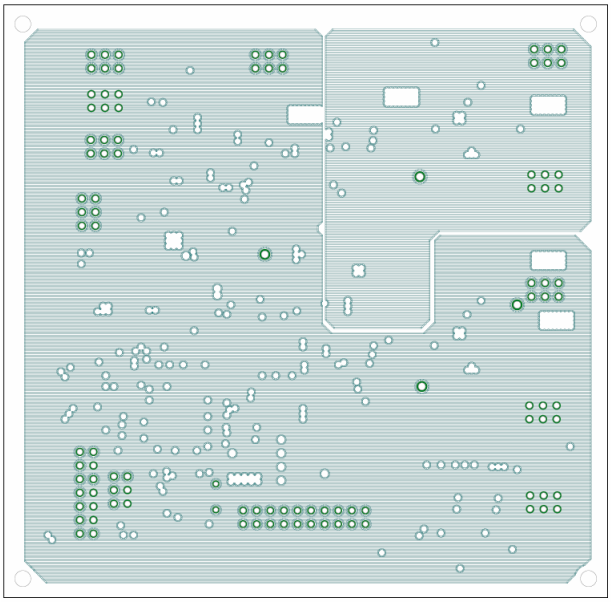


Figure A.14: Hysteretic control board for DPP layout, layer 2 (gnd planes)

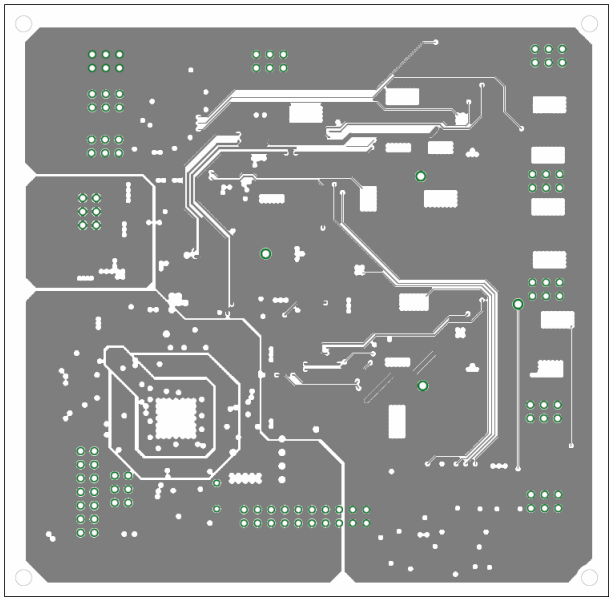


Figure A.15: Hysteretic control board for DPP layout, layer 3 (microcontroller supply)

APPENDIX B

MICROCONTROLLER C CODES

B.1 C Code Used to Program the TMS320F28335 for Voltage-Mode Control

```
#include "DSP28x_Project.h"
#include "DSP2833x_EPwm_defines.h"

#define HR_Disaable 0x0
#define HR_REP 0x1       // Rising Edge Position
#define HR_FEP 0x2       // Falling Edge Position
#define HR_BEP 0x3       // Both Edge position
#define HR_CMP 0x0       // CMPAHR controlled
#define HR_PHS 0x1       // TBPHSHR controlled
#define HR_CTR_ZERO 0x0  // CTR = Zero event
#define HR_CTR_PRD 0x1   // CTR = Period event


// Declare your function prototypes here
void HRPWM1_Config(int);
void HRPWM2_Config(int);
void HRPWM3_Config(int);
//void HRPWM5_Config(int);

__interrupt void adc_isr(void);

// General System nets - Useful for debug
Uint16 i,j,duty, DutyFine, n, update, status;
float32 v1, v1s, v2, v2s, v3, v3s, v4, v4s, vdom1, vdom2, vdom3, vdom4;
float32 vdiff1,vdiff2, vdiff3, vdiff1a, vdiff1b, vdiff2a;
float32 vdiff2b, vdiff3a, vdiff3b=0.0;
```

```
float32 vadd1, vadd2, vadd3, duty1, duty2, duty3, duty1a, duty2a;
float32 duty3a, ki, kp;
float32 idiff1, idiff2, idiff3, ilim, frac;
Uint32 temp, prd, hprd1, hprd2, hprd3, cl;
long cmpa_reg1, cmpa_reg2, cmpa_reg3, cmpahr1, cmpahr2, cmpahr3;

void main(void)
{
// Step 1. Initialize System Control:
// PLL, WatchDog, enable Peripheral Clocks
// This example function is found in the DSP2833x_SysCtrl.c file.

InitSysCtrl();
EALLOW;
#if (CPU_FRQ_150MHZ)     // Default - 150 MHz SYSCLKOUT
#define ADC_MODCLK 0x3//HSPCLK=SYSCLKOUT/2*ADC_MODCLK2=150/(2*3)=25.0 MHz
#endif
#if (CPU_FRQ_100MHZ)
#define ADC_MODCLK 0x2//HSPCLK=SYSCLKOUT/2*ADC_MODCLK2=100/(2*2)=25.0 MHz
#endif
EDIS;
prd = 100;
hprd1 = 50;
hprd2 = 50;
hprd3 = 50;
ki=0;
kp=0;
ilim=10000;
cl=0;
EALLOW;
SysCtrlRegs.HISPCP.all = ADC_MODCLK;
EDIS;

// Step 3. Clear all interrupts and initialize PIE vector table:
// Disable CPU interrupts
DINT;

// Initialize the PIE control registers to their default state.
```

```
// The default state is all PIE interrupts disabled and flags
// are cleared.
// This function is found in the DSP2833x_PieCtrl.c file.
InitPieCtrl();

// Disable CPU interrupts and clear all CPU interrupt flags:
IER = 0x0000;
IFR = 0x0000;

InitPieVectTable();
EALLOW;
PieVectTable.ADCINT = &adc_isr;
EDIS;



InitAdc();

InitEPwm4Gpio();
InitEPwm5Gpio();
InitEPwm6Gpio();
//   InitEPwm5Gpio();

// Initialize the PIE vector table with pointers to the shell Interrupt
// Service Routines (ISR).
// This will populate the entire table, even if the interrupt
// is not used in this example.  This is useful for debug purposes.
// The shell ISR routines are found in DSP2833x_DefaultIsr.c.
// This function is found in DSP2833x_PieVect.c.

PieCtrlRegs.PIEIER1.bit.INTx6 = 1;
IER |= M_INT1; // Enable CPU Interrupt 1
EINT;          // Enable Global interrupt INTM
ERTM;          // Enable Global realtime interrupt DBGM
// Step 4. Initialize all the Device Peripherals:
// This function is found in DSP2833x_InitPeripherals.c
// InitPeripherals();  // Not required for this example

// For this example, only initialize the ePWM
```

```
// Step 5. User specific code, enable interrupts:

update =1;
DutyFine =0;

EALLOW;
SysCtrlRegs.PCLKCR0.bit.TBCLKSYNC = 0;
EDIS;

//  Some useful Period vs Frequency values
//  SYSCLKOUT =     150MHz         100 MHz
//  ----------------------------------------
// Period         Frequency      Frequency
// 1000 150 kHz    100 KHz
// 800 187 kHz    125 KHz
// 600 250 kHz    167 KHz
// 500 300 kHz    200 KHz
// 250 600 kHz    400 KHz
// 200 750 kHz    500 KHz
// 100 1.5 MHz    1.0 MHz
// 50 3.0 MHz    2.0 MHz
// 25 6.0 MHz    4.0 MHz
// 20 7.5 MHz    5.0 MHz
// 12 12.5 MHz    8.33 MHz
// 10 15.0 MHz    10.0 MHz
// 9 16.7 MHz    11.1 MHz
// 8 18.8 MHz    12.5 MHz
// 7 21.4 MHz    14.3 MHz
// 6 25.0 MHz    16.7 MHz
// 5 30.0 MHz    20.0 MHz

//=========================================================================
// ePWM and HRPWM register initialization
//=========================================================================
HRPWM1_Config(prd);
HRPWM2_Config(prd);
HRPWM3_Config(prd);
//   HRPWM5_Config(prd);
```

```
AdcRegs.ADCTRL2.bit.EPWM_SOCA_SEQ1 = 1;//Enable SOCA for starting SEQ1
AdcRegs.ADCTRL2.bit.INT_ENA_SEQ1 = 1; // Enable SEQ1 interrupt
AdcRegs.ADCTRL3.bit.ADCCLKPS = 6; //HSPCLK : 6 --> 75 MHz / 6 = 12.5 MHz
AdcRegs.ADCTRL1.bit.ACQ_PS = 0x06; //T Sampling = ((ACQ_PS + 1) * AdcCLK)
/* Conversion mode configuration */
AdcRegs.ADCTRL3.bit.SMODE_SEL = 1; /* simultaneous mode */
AdcRegs.ADCMAXCONV.bit.MAX_CONV1 = 1;
AdcRegs.ADCTRL1.bit.SEQ_CASC = 0;

AdcRegs.ADCCHSELSEQ1.bit.CONV00 = 0x01; /* Input ADCIN A0 - B0 */
AdcRegs.ADCCHSELSEQ1.bit.CONV01 = 0x02; /* Input ADCIN A1 - B1 */

EALLOW;
SysCtrlRegs.PCLKCR0.bit.TBCLKSYNC = 1;
EDIS;

for(;;)
{
i++;
}

}


void HRPWM1_Config(period)
{
// EPwm4 register configuration with HRPWM
// EPwm4A toggle low/high with MEP control on Rising edge

EPwm4Regs.TBCTL.bit.PRDLD = TB_IMMEDIATE;// set Immediate load
EPwm4Regs.TBPRD = period - 1;      // PWM frequency = 1 / period
EPwm4Regs.CMPA.half.CMPA = period / 2;   // set duty 50% initially
EPwm4Regs.CMPA.half.CMPAHR = (1 << 8);   // initialize HRPWM extension
EPwm4Regs.CMPB = period / 2;          // set duty 50% initially
EPwm4Regs.TBPHS.all = 0;
EPwm4Regs.TBCTR = 0;
```

```
EPwm4Regs.TBCTL.bit.CTRMODE = TB_COUNT_UP;
EPwm4Regs.TBCTL.bit.PHSEN = TB_DISABLE;  // EPwm4 is the Master
EPwm4Regs.TBCTL.bit.SYNCOSEL = TB_SYNC_DISABLE;
EPwm4Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1;
EPwm4Regs.TBCTL.bit.CLKDIV = TB_DIV1;

EPwm4Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO;
EPwm4Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO;
EPwm4Regs.CMPCTL.bit.SHDWAMODE = CC_IMMEDIATE;
EPwm4Regs.CMPCTL.bit.SHDWBMODE = CC_IMMEDIATE;

EPwm4Regs.AQCTLA.bit.ZRO = AQ_CLEAR;     // PWM toggle low/high
EPwm4Regs.AQCTLA.bit.CAU = AQ_SET;
EPwm4Regs.AQCTLB.bit.ZRO = AQ_SET;
EPwm4Regs.AQCTLB.bit.CAU = AQ_CLEAR;

EALLOW;
EPwm4Regs.HRCNFG.all = 0x0;
EPwm4Regs.HRCNFG.bit.EDGMODE = HR_REP; //MEP control on Falling edge
EPwm4Regs.HRCNFG.bit.CTLMODE = HR_CMP;
EPwm4Regs.HRCNFG.bit.HRLOAD  = HR_CTR_ZERO;
EDIS;

EPwm4Regs.ETSEL.bit.SOCAEN       // Enable SOC on A group
EPwm4Regs.ETSEL.bit.SOCASEL = 6; // Select SOC from from CMPA on upcount
EPwm4Regs.ETPS.bit.SOCAPRD = 3;  // Generate pulse on 1st event
}

void HRPWM2_Config(period)
{
//    EPwm5 register configuration with HRPWM
//    EPwm5A toggle low/high with MEP control on Rising edge

EPwm5Regs.TBCTL.bit.PRDLD = TB_IMMEDIATE;    // set Immediate load
EPwm5Regs.TBPRD = period - 1;    // PWM frequency = 1 / period
EPwm5Regs.CMPA.half.CMPA = period / 2; // set duty 50% initially
EPwm5Regs.CMPA.half.CMPAHR = (1 << 8); // initialize HRPWM extension
EPwm5Regs.CMPB = period / 2;        // set duty 50% initially
```

```
EPwm5Regs.TBPHS.all = 0;
EPwm5Regs.TBCTR = 0;

EPwm5Regs.TBCTL.bit.CTRMODE = TB_COUNT_UP;
EPwm5Regs.TBCTL.bit.PHSEN = TB_DISABLE;// EPwm5 is the Master
EPwm5Regs.TBCTL.bit.SYNCOSEL = TB_SYNC_DISABLE;
EPwm5Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1;
EPwm5Regs.TBCTL.bit.CLKDIV = TB_DIV1;

EPwm5Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO;
EPwm5Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO;
EPwm5Regs.CMPCTL.bit.SHDWAMODE = CC_IMMEDIATE;
EPwm5Regs.CMPCTL.bit.SHDWBMODE = CC_IMMEDIATE;

EPwm5Regs.AQCTLA.bit.ZRO = AQ_CLEAR;     // PWM toggle low/high
EPwm5Regs.AQCTLA.bit.CAU = AQ_SET;
EPwm5Regs.AQCTLB.bit.ZRO = AQ_SET;
EPwm5Regs.AQCTLB.bit.CAU = AQ_CLEAR;

EALLOW;
EPwm5Regs.HRCNFG.all = 0x0;
EPwm5Regs.HRCNFG.bit.EDGMODE = HR_REP;  //MEP control on Rising edge
EPwm5Regs.HRCNFG.bit.CTLMODE = HR_CMP;
EPwm5Regs.HRCNFG.bit.HRLOAD  = HR_CTR_ZERO;

EDIS;

}

void HRPWM3_Config(period)
{
// EPwm6 register configuration with HRPWM
// EPwm6A toggle high/low with MEP control on falling edge

EPwm6Regs.TBCTL.bit.PRDLD = TB_IMMEDIATE;// set Immediate load
EPwm6Regs.TBPRD = period - 1;     // PWM frequency = 1 / period
EPwm6Regs.CMPA.half.CMPA = period / 2;  // set duty 50% initially
EPwm6Regs.CMPA.half.CMPAHR = (1 << 8);  // initialize HRPWM extension
```

```
EPwm6Regs.CMPB = period / 2;         // set duty 50% initially
EPwm6Regs.TBPHS.all = 0;
EPwm6Regs.TBCTR = 0;

EPwm6Regs.TBCTL.bit.CTRMODE = TB_COUNT_UP;
EPwm6Regs.TBCTL.bit.PHSEN = TB_DISABLE;  // EPwm6 is the Master
EPwm6Regs.TBCTL.bit.SYNCOSEL = TB_SYNC_DISABLE;
EPwm6Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1;
EPwm6Regs.TBCTL.bit.CLKDIV = TB_DIV1;

EPwm6Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO;
EPwm6Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO;
EPwm6Regs.CMPCTL.bit.SHDWAMODE = CC_IMMEDIATE;
EPwm6Regs.CMPCTL.bit.SHDWBMODE = CC_IMMEDIATE;

EPwm6Regs.AQCTLA.bit.ZRO = AQ_CLEAR;   // PWM toggle high/low
EPwm6Regs.AQCTLA.bit.CAU = AQ_SET;
EPwm6Regs.AQCTLB.bit.ZRO = AQ_SET;
EPwm6Regs.AQCTLB.bit.CAU = AQ_CLEAR;

EALLOW;
EPwm6Regs.HRCNFG.all = 0x0;
EPwm6Regs.HRCNFG.bit.EDGMODE = HR_REP; //MEP control on falling edge
EPwm6Regs.HRCNFG.bit.CTLMODE = HR_CMP;
EPwm6Regs.HRCNFG.bit.HRLOAD  = HR_CTR_ZERO;
EDIS;

}

__interrupt void  adc_isr(void)
{
v2 = (int)((AdcRegs.ADCRESULT0) >>3);
v4 = (int)((AdcRegs.ADCRESULT1) >>2);
v1 = (int)((AdcRegs.ADCRESULT2) >>4);
v3 = (int)((AdcRegs.ADCRESULT3) >>3)+(int)((AdcRegs.ADCRESULT3) >>4);

//voltage difference
vdiff1 = v2-2*v1;
```

```
vdiff2 = v3-2*v2+v1;
vdiff3 = v4-2*v3+v2;

idiff1=idiff1+vdiff1;
idiff2=idiff2+vdiff2;
idiff3=idiff3+vdiff3;

//integrator saturation
if(idiff1<-ilim)
idiff1=-ilim;
else if(idiff1>ilim)
idiff1=ilim;
else
{}

if(idiff2<-ilim)
idiff2=-ilim;
else if(idiff2>ilim)
idiff2=ilim;
else
{}

if(idiff3<-ilim)
idiff3=-ilim;
else if(idiff3>ilim)
idiff3=ilim;
else
{}

//compensation
duty1=((kp*vdiff1+ki*idiff1)/150+0.5)*32768;
duty2=((kp*vdiff2+ki*idiff2)/150+0.5)*32768;
duty3=((kp*vdiff3+ki*idiff3)/150+0.5)*32768;



if(cl==1)
{
```

```
cmpa_reg1 = ((long)duty1*(prd-1))>>15;
temp = ((long)duty1 * (prd-1));
temp = temp - ((long)cmpa_reg1 << 15);
cmpahr1 = (temp*52) >> 15;
cmpahr1 = cmpahr1 << 8;
cmpahr1 += 0x0180;

EPwm4Regs.CMPA.all = (long)cmpa_reg1 <<16 | cmpahr1;

cmpa_reg2 = ((long)duty2*(prd-1))>>15;
temp = ((long)duty2*(prd-1));
temp = temp - ((long)cmpa_reg2 << 15);
cmpahr2 = (temp*52) >> 15;
cmpahr2 = cmpahr2 << 8;
cmpahr2 += 0x0180;

EPwm5Regs.CMPA.all = (long)cmpa_reg2 <<16 | cmpahr2;

cmpa_reg3 = ((long)duty3*(prd-1))>>15;
temp = ((long)duty3*(prd-1));
temp = temp - ((long)cmpa_reg3 << 15);
cmpahr3 = (temp*52) >> 15;
cmpahr3 = cmpahr3 << 8;
cmpahr3 += 0x0180;

EPwm6Regs.CMPA.all = (long)cmpa_reg3 << 16 | cmpahr3;
}

else
{
EPwm4Regs.CMPA.half.CMPA = hprd1;
//EPwm4Regs.CMPA.half.CMPAHR = (duty1-(int)(duty1))*0.17;
EPwm4Regs.CMPB = hprd1;

EPwm5Regs.CMPA.half.CMPA = hprd2;
//EPwm5Regs.CMPA.half.CMPAHR = (duty2-(int)(duty2))*0.17;
EPwm5Regs.CMPB = hprd2;
```

```
EPwm6Regs.CMPA.half.CMPA = hprd3;
//EPwm6Regs.CMPA.half.CMPAHR = (duty3-(int)(duty3))*0.17;
EPwm6Regs.CMPB = hprd3;

}


// Reinitialize for next ADC sequence
AdcRegs.ADCTRL2.bit.RST_SEQ1 = 1;       // Reset SEQ1
AdcRegs.ADCST.bit.INT_SEQ1_CLR = 1;     // Clear INT SEQ1 bit
PieCtrlRegs.PIEACK.all = PIEACK_GROUP1; // Acknowledge interrupt to PIE

return;
}
```

B.2 C Code Used to Program the TMS320F28377S for Current Hysteretic Control

```
#include "F28x_Project.h"// Device Headerfile and Examples Include File

void InitEPWM2(void);
void InitEPWM6(void);
void InitEPWM7(void);
void InitCMPSS1(void);
void InitCMPSS2(void);
void InitECapture(void);
void ConfigureADC(void);
void ConfigureEPWM(void);
void SetupADCEpwm(void);
void startupseq(void);
void error(void);

__interrupt void ecap1_isr(void);
__interrupt void adca1_isr(void);
// Maximum Dead Band values
#define EPWM2_MAX_DB   0x03FF
```

```
#define EPWM2_MIN_DB   0

//definitions for selecting DACH reference
#define REFERENCE_VDDA     0
#define REFERENCE_VDAC     1

//definitions for COMPH input selection
#define NEGIN_DAC          0
#define NEGIN_PIN          1

//definitions for CTRIPH/CTRIPOUTH output selection
#define CTRIP_ASYNCH       0
#define CTRIP_SYNCH        1
#define CTRIP_FILTER       2
#define CTRIP_LATCH        3

//definitions for selecting output pin
#define GPIO_CTRIPOUT1_PIN_NUM      60 //OUTPUTXBAR3 is mux'd with GPIO14
#define GPIO_CTRIPOUT2_PIN_NUM      61 //OUTPUTXBAR4 is mux'd with GPIO15
#define GPIO_CTRIPOUT1_PER_NUM      5
#define GPIO_CTRIPOUT2_PER_NUM      5
#define RESULTS_BUFFER_SIZE 10

Uint16 v3[RESULTS_BUFFER_SIZE]={2300};
float kp, ki , sum;

int comp_mean2_2, comp_mean2_0, comp_mean2_1 = 2068;
int comp_dev2 = 400, hval, lval;
int dac1, dac=0;
int mode, i=0;
int err_av, ierr, err[10];
int prd = 100, start=0;
int intcnt=0;
int Tst1, Tst2, Tst3, Tst4, Prd1, Prd2, Prd3, Prd_av;

//
#define EPWM2_MIN_DB   0
```

```
void main(void)
{
// Step 1. Initialize System Control:
// PLL, WatchDog, enable Peripheral Clocks
// This example function is found in the F2837xS_SysCtrl.c file.
InitSysCtrl();

// Step 2. Initialize GPIO:
// This example function is found in the F2837xS_Gpio.c file and
// illustrates how to set the GPIO to its default state.
InitGpio();
// These functions are in the F2837xS_EPwm.c file
InitEPwm8Gpio();
GPIO_SetupPinMux(16, GPIO_MUX_CPU1, 5);
GPIO_SetupPinMux(18, GPIO_MUX_CPU1, 5);
// Disable CPU interrupts
DINT;

// Initialize the PIE control registers to their default state.
// The default state is all PIE interrupts disabled and flags
// are cleared.
// This function is found in the F2837xS_PieCtrl.c file.
InitPieCtrl();

// Disable CPU interrupts and clear all CPU interrupt flags:
IER = 0x0000;
IFR = 0x0000;

InitPieVectTable();

EALLOW;// This is needed to write to EALLOW protected registers
PieVectTable.ADCA1_INT = &adca1_isr;
EDIS;// This is needed to disable write to EALLOW protected registers

IER |= M_INT1;

EINT;   // Enable Global __interrupt INTM
```

```
ERTM;   // Enable Global realtime __interrupt DBGM

PieCtrlRegs.PIEIER1.bit.INTx1 = 1;

//
InitCMPSS2();
InitEPWM2();
InitEPwm2Gpio();
InitECapture();
ConfigureADC();
ConfigureEPWM();
SetupADCEpwm();


GPIO_SetupPinMux(60, GPIO_MUX_CPU1, 5);
GPIO_SetupPinMux(60, GPIO_MUX_CPU1, 5);
//   InitECap1Gpio(60);
//   GPIO_SetupPinOptions(60, 0, 0x3);

start=1;
kp=10;
ki=0;
mode=0;

while(1)
{

if(start==1)
{
EPwm2Regs.TBPRD = 1000;  // Set timer period
EPwm2Regs.AQCTLA.bit.CAU  = AQ_NO_ACTION;
EPwm2Regs.AQCTLA.bit.PRD  = AQ_NO_ACTION;
EPwm2Regs.AQCTLB.bit.CAU  = AQ_NO_ACTION;
EPwm2Regs.AQCTLB.bit.PRD  = AQ_NO_ACTION;
start=0;
}

}
```

```
// Step 6. IDLE loop. Just sit and loop forever (optional):
}

void InitCMPSS2(void)
{
EALLOW;
//Enable CMPSS
Cmpss2Regs.COMPCTL.bit.COMPDACE            = 0x1;
//NEG signal of High comparator comes from DAC
Cmpss2Regs.COMPCTL.bit.COMPHSOURCE         = 0x0;
//NEG signal of Low comparator comes from DAC
Cmpss2Regs.COMPCTL.bit.COMPLSOURCE         = 0x0;

//Use VDDA as the reference for DAC
Cmpss2Regs.COMPDACCTL.bit.SELREF           = 0x0;
// Load DACxVALA from its shadow registor (not the ramp generator)
Cmpss2Regs.COMPDACCTL.bit.DACSOURCE        = 0x0;
// Load DACxVALA immediately after loading its shadow registor
Cmpss2Regs.COMPDACCTL.bit.SWLOADSEL        = 0x0;

//Set DAC voltage level
//High comparator get upper limit, so its output is normally low
Cmpss2Regs.DACHVALS.bit.DACVAL             = comp_mean2_0+comp_dev2;
//Low comparator get lower limit, so its output is normally high
Cmpss2Regs.DACLVALS.bit.DACVAL             = comp_mean2_0-comp_dev2;

//invert Low comparator signal since we use high to trigger PWM event
Cmpss2Regs.COMPCTL.bit.COMPLINV         = 0x1;
//do not invert Low comparator signal
Cmpss2Regs.COMPCTL.bit.COMPHINV         = 0x0;
Cmpss2Regs.COMPHYSCTL.bit.COMPHYS           = 0x2;

// Configure compare result output path
//Asynch output feeds CTRIPH and CTRIPL
Cmpss2Regs.COMPCTL.bit.CTRIPHSEL        = 0x0;
Cmpss2Regs.COMPCTL.bit.CTRIPLSEL        = 0x0;
```

```
//Asynch output feeds CTRIPOUTH and CTRIPOUTL
Cmpss2Regs.COMPCTL.bit.CTRIPOUTHSEL        = 0x0;
Cmpss2Regs.COMPCTL.bit.CTRIPOUTLSEL        = 0x0;

// Configure CTRIPH output to ePWM X-BAR logic
//Configure TRIP4 to be CMPSS5 CTRIPH (select MUX8.1)
EPwmXbarRegs.TRIP4MUX0TO15CFG.bit.MUX2 = 0; //select .1 input
//Configure TRIP5 to be CMPSS5 CTRIPL (select MUX9.1)
EPwmXbarRegs.TRIP5MUX0TO15CFG.bit.MUX3 = 0; //select .1
//Enable TRIP4 Mux for Output
EPwmXbarRegs.TRIP4MUXENABLE.bit.MUX2   = 1;
//Enable TRIP5 Mux for Output
EPwmXbarRegs.TRIP5MUXENABLE.bit.MUX3   = 1;

//Configure CTRIPOUTH output pin
//Configure XTRIPOUT3 to be CTRIPOUT1H
OutputXbarRegs.OUTPUT3MUX0TO15CFG.bit.MUX2 = 0;
OutputXbarRegs.OUTPUT4MUX0TO15CFG.bit.MUX3 = 0;
//Enable XTRIPOUT3 Mux for Output
OutputXbarRegs.OUTPUT3MUXENABLE.bit.MUX2   = 1;
OutputXbarRegs.OUTPUT4MUXENABLE.bit.MUX3   = 1;

EDIS;
}



void InitEPWM2(void)
{
EALLOW;
//Configure EPWM to run at SYSCLK
ClkCfgRegs.PERCLKDIVSEL.bit.EPWMCLKDIV = 0;
EPwm2Regs.TBCTL.bit.CLKDIV             = 0;
EPwm2Regs.TBCTL.bit.HSPCLKDIV          = 0;


EPwm2Regs.TBCTL.bit.PHSEN = TB_DISABLE;    // Disable phase loading.
EPwm2Regs.TBPHS.bit.TBPHS = 0x0000;        // Phase is 0
```

```
EPwm2Regs.TBPRD = prd;  // Set timer period
EPwm2Regs.CMPA.bit.CMPA = prd/2 ;

EPwm2Regs.AQCTLA2.bit.T1U = AQ_CLEAR;      // clear PWM8A upon T1 event
EPwm2Regs.AQCTLA2.bit.T2U = AQ_SET;        // set PWM8A upon T2 event
EPwm2Regs.AQCTLB2.bit.T1U = AQ_SET;        // clear PWM8A upon T1 event
EPwm2Regs.AQCTLB2.bit.T2U = AQ_CLEAR;      // set PWM8A upon T2 event


EPwm2Regs.AQCTLA.bit.CAU = AQ_CLEAR;
EPwm2Regs.AQCTLA.bit.PRD = AQ_SET;
EPwm2Regs.AQCTLB.bit.CAU = AQ_SET;
EPwm2Regs.AQCTLB.bit.PRD = AQ_CLEAR;

// Set current hysteresis actions
EPwm2Regs.AQTSRCSEL.bit.T1SEL = 0x0;//select DCAEVT1 as T1 event source
EPwm2Regs.AQTSRCSEL.bit.T2SEL = 0x2;//select DCBEVT1 as T2 event source

//enable DCAEVT1, DCAEVT2 sync to clear the counter
EPwm2Regs.DCACTL.bit.EVT1SYNCE = 0x1;
EPwm2Regs.DCBCTL.bit.EVT1SYNCE = 0x1;



// Set digital compare and trip zone
// determine the input-ouptput logic of the Digital Comparator
EPwm2Regs.TZDCSEL.bit.DCAEVT1          = TZ_DCAL_HI_DCAH_LOW;
//generate DCAEVT1 (current signal hit upper limit)
EPwm2Regs.TZDCSEL.bit.DCAEVT2          = TZ_DCBL_HI_DCBH_LOW;
//generate DCAEVT2 (current signal hit lower limit)
//when DCA low input (trip 4) is high, high input (trip 5) is low
// be careful how to connect trip input to DC later
EPwm2Regs.TZDCSEL.bit.DCBEVT1          = TZ_DCBL_HI_DCBH_LOW;
//generate DCBEVT1 when DCB low input is high, high input is low
EPwm2Regs.TZDCSEL.bit.DCBEVT2          = TZ_DCAL_HI_DCAH_LOW;
//generate DCBEVT2 (current signal hit upper limit)

//Configure DCA input
```

```
EPwm2Regs.DCTRIPSEL.bit.DCALCOMPSEL    = 0x3; // DCA low input is trip 4
EPwm2Regs.DCTRIPSEL.bit.DCAHCOMPSEL    = 0x4; // DCA high input is trip 4
//Configure DCB input
EPwm2Regs.DCTRIPSEL.bit.DCBLCOMPSEL    = 0x4; // DCB low input is trip 5
EPwm2Regs.DCTRIPSEL.bit.DCBHCOMPSEL    = 0x3; // DCB high input is trip 4


//Configure DCA path to be unfiltered & async
EPwm2Regs.DCACTL.bit.EVT1SRCSEL        = DC_EVT1;
EPwm2Regs.DCACTL.bit.EVT1FRCSYNCSEL    = DC_EVT_ASYNC;
//Configure DCB path to be unfiltered & async
EPwm2Regs.DCBCTL.bit.EVT1SRCSEL        = DC_EVT1;
EPwm2Regs.DCBCTL.bit.EVT1FRCSYNCSEL    = DC_EVT_ASYNC;

// Enable TZ1 as one cycle-by-cycle trip sources
EPwm2Regs.TZCTL.bit.TZA                = TZ_NO_CHANGE;
EPwm2Regs.TZCTL.bit.TZB                = TZ_NO_CHANGE;
EPwm2Regs.TZCTL.bit.DCAEVT1            = TZ_NO_CHANGE;
EPwm2Regs.TZCTL.bit.DCBEVT1            = TZ_NO_CHANGE;

// Active high complementary PWMs - Setup the deadband
// EPwm2Regs.DBCTL.bit.OUT_MODE = DB_FULL_ENABLE;
// EPwm2Regs.DBCTL.bit.POLSEL = DB_ACTV_HIC;
// EPwm2Regs.DBCTL.bit.IN_MODE = DBA_ALL;
// EPwm2Regs.DBRED.bit.DBRED = 2;
// EPwm2Regs.DBFED.bit.DBFED = 2;

//SOC generation
//  EPwm2Regs.DCACTL.bit.EVT1SOCE = 1;      // Enable DCAEVT1 SOC
// EPwm2Regs.ETSEL.bit.SOCAEN = 1;     // Enable SOC on A group
// EPwm2Regs.ETSEL.bit.SOCASEL = 0;     // Select SOC on DCAEVT1
// EPwm2Regs.ETPS.bit.SOCAPRD = 1;     // Generate pulse on 1st event

//  EPwm2Regs.DCBCTL.bit.EVT1SOCE = 1;    // Enable DCAEVT1 SOC
// EPwm2Regs.ETSEL.bit.SOCBEN = 1;     // Enable SOC on A group
// EPwm2Regs.ETSEL.bit.SOCBSEL = 0;     // Select SOC on DCAEVT1
// EPwm2Regs.ETPS.bit.SOCBPRD = 1;     // Generate pulse on 1st event
```

```
//  EPwm2Regs.ETPS.bit.SOCPSSEL = 0;

//  Enable PWM
EPwm2Regs.TBCTL.bit.CTRMODE = TB_COUNT_UP; // Count up

EDIS;
}


//ecap used for pwm frequency measurement
void InitECapture()
{
ECap1Regs.ECEINT.all = 0x0000;    // Disable all capture __interrupts
ECap1Regs.ECCLR.all = 0xFFFF;     // Clear all CAP __interrupt flags
ECap1Regs.ECCTL1.bit.CAPLDEN = 0; // Disable CAP1-CAP4 register loads
ECap1Regs.ECCTL2.bit.TSCTRSTOP = 0; // Make sure the counter is stopped

ECap1Regs.ECCTL1.bit.CAP2POL = 0x0;
ECap1Regs.ECCTL1.bit.CAP3POL = 0x0;
ECap1Regs.ECCTL1.bit.CAP4POL = 0x0;
ECap1Regs.ECCTL1.bit.CTRRST1 = 0x0;
ECap1Regs.ECCTL1.bit.CTRRST2 = 0x0;
ECap1Regs.ECCTL1.bit.CTRRST3 = 0x0;
ECap1Regs.ECCTL1.bit.CTRRST4 = 0x0;
ECap1Regs.ECCTL1.bit.CAPLDEN = 0x1;
ECap1Regs.ECCTL1.bit.PRESCALE = 0x0;
ECap1Regs.ECCTL2.bit.CAP_APWM = 0x0;
ECap1Regs.ECCTL2.bit.CONT_ONESHT = 0x0;
ECap1Regs.ECCTL2.bit.SYNCO_SEL = 0x2;
ECap1Regs.ECCTL2.bit.SYNCI_EN = 0x0;
ECap1Regs.ECCTL2.bit.TSCTRSTOP = 0x1;

ECap1Regs.ECCTL2.bit.TSCTRSTOP = 1;        // Start Counter
ECap1Regs.ECCTL2.bit.REARM = 1;            // arm one-shot
ECap1Regs.ECCTL1.bit.CAPLDEN = 1;   // Enable CAP1-CAP4 register loads
ECap1Regs.ECEINT.bit.CEVT4 = 1;          // 4 events = __interrupt
}
```

```
__interrupt void ecap1_isr(void)
{
Tst1 = ECap1Regs.CAP1;
Tst2 = ECap1Regs.CAP2;
Tst3 = ECap1Regs.CAP3;
Tst4 = ECap1Regs.CAP4;

Prd1 = Tst2-Tst1;     //compute time differences between ecap events
Prd2 = Tst3-Tst2;
Prd3 = Tst4-Tst3;

Prd_av = (Prd1+Prd2+Prd3)/3;
//computed period averaged over 3 cycles
sum(z)/prd_av(z)=1/(3-z^(-1))

ECap1Regs.ECCLR.bit.CEVT4 = 1;
ECap1Regs.ECCLR.bit.INT = 1;
ECap1Regs.ECCTL2.bit.REARM = 1;

//   comp_dev=233-kp*(prd-sum)/10;
//   to deccrease thefrequency variation of the hysteretic controller
//   useful only in VM hysteretic control

// Acknowledge this __interrupt to receive more __interrupts from group 4
PieCtrlRegs.PIEACK.all = PIEACK_GROUP4;
}

void ConfigureADC(void)
{
EALLOW;

//write configurations
AdcaRegs.ADCCTL2.bit.PRESCALE = 6; //set ADCCLK divider to /4
AdcbRegs.ADCCTL2.bit.PRESCALE = 6;
AdcSetMode(ADC_ADCA, ADC_RESOLUTION_12BIT, ADC_SIGNALMODE_SINGLE);
AdcSetMode(ADC_ADCB, ADC_RESOLUTION_12BIT, ADC_SIGNALMODE_SINGLE);

//Set pulse positions to late
```

```
AdcaRegs.ADCCTL1.bit.INTPULSEPOS = 1;
AdcbRegs.ADCCTL1.bit.INTPULSEPOS = 1;

//power up the ADC
AdcaRegs.ADCCTL1.bit.ADCPWDNZ = 1;
AdcbRegs.ADCCTL1.bit.ADCPWDNZ = 1;

//delay for 1ms to allow ADC time to power up
DELAY_US(1000);
EDIS;
}

void SetupADCEpwm(void)
{
Uint16 acqps;

//determine minimum acquisition window (in SYSCLKS) based on resolution
if(ADC_RESOLUTION_12BIT == AdcaRegs.ADCCTL2.bit.RESOLUTION){
acqps = 14; //75ns
}
else { //resolution is 16-bit
acqps = 63; //320ns
}

//Select the channels to convert and end of conversion flag
EALLOW;

AdcaRegs.ADCSOC0CTL.bit.CHSEL = 1;
//SOC0 will convert pin A1, result stored in AdcaRegs.ADCRESULT0
AdcaRegs.ADCSOC0CTL.bit.ACQPS = acqps;
//sample window is 100 SYSCLK cycles
AdcbRegs.ADCSOC0CTL.bit.CHSEL = 1;
//SOC0 will convert pin B1, result stored in AdcbRegs.ADCRESULT0
AdcbRegs.ADCSOC0CTL.bit.ACQPS = acqps;//sample window is 100 SYSCLK cycles
AdcaRegs.ADCSOC0CTL.bit.TRIGSEL = 5;      //SOC0 trigger on ePWM1 SOCA
AdcbRegs.ADCSOC0CTL.bit.TRIGSEL = 5;      //SOC0 trigger on ePWM1 SOCA

AdcaRegs.ADCINTSEL1N2.bit.INT1SEL = 0; //end of SOC0 will set INT1 flag
```

```
AdcaRegs.ADCINTSEL1N2.bit.INT1E = 1;   //enable INT1 flag
AdcaRegs.ADCINTFLGCLR.bit.ADCINT1 = 1; //make sure INT1 flag is cleared

}

void ConfigureEPWM(void)
{
EALLOW;
// Assumes ePWM clock is already enabled
ClkCfgRegs.PERCLKDIVSEL.bit.EPWMCLKDIV = 0;
EPwm1Regs.TBCTL.bit.CLKDIV             = 0;
EPwm1Regs.TBCTL.bit.HSPCLKDIV          = 0;

EPwm1Regs.ETSEL.bit.SOCAEN = 1;         // Enable SOC on A group
EPwm1Regs.ETSEL.bit.SOCASEL = 4;         // Select SOC on up-count
EPwm1Regs.ETSEL.bit.SOCBEN = 1;         // Enable SOC on A group
EPwm1Regs.ETSEL.bit.SOCBSEL = 4;         // Select SOC on up-count

EPwm1Regs.ETPS.bit.SOCAPRD = 1;         // Generate pulse on 1st event
EPwm1Regs.ETPS.bit.SOCBPRD = 1;         // Generate pulse on 1st event
EPwm1Regs.CMPA.bit.CMPA = 49;        // Set compare A value to 2048 counts
EPwm1Regs.CMPB.bit.CMPB = 49;        // Set compare A value to 2048 counts
EPwm1Regs.TBPRD = 50;      // Set period to 4096 counts
EPwm1Regs.TBCTL.bit.CTRMODE = 0;     //unfreeze, and enter up count mode
EPwm1Regs.TBCTL.bit.CTRMODE = TB_COUNT_UP; // Count up

EDIS;
}

__interrupt void adca1_isr(void)
{
// intcnt++;
v3[0]  = AdcbResultRegs.ADCRESULT0;//ADCINB1 pin
// err[2] = err[1]*3;
// err[1] = err[0];
err[0] = 2252-(int)v3[0];

// comp_mean2_2 = comp_mean2_1;
```

```
comp_mean2_1 = comp_mean2_0*5;
comp_mean2_0 = comp_mean2_1/10+3*err[0];

// dac  = comp_mean2_0;

//    lval =
//    hval =

if(comp_mean2_0>0)
{
EPwm2Regs.AQCTLA2.bit.T1U = AQ_CLEAR;      // clear PWM8A upon T1 event
EPwm2Regs.AQCTLA2.bit.T2U = AQ_SET;        // set PWM8A upon T2 event
EPwm2Regs.AQCTLB2.bit.T1U = AQ_CLEAR;      // clear PWM8A upon T1 event
EPwm2Regs.AQCTLB2.bit.T2U = AQ_CLEAR;      // set PWM8A upon T2 event
}
else if(comp_mean2_0<0)
{
EPwm2Regs.AQCTLA2.bit.T1U = AQ_CLEAR;      // clear PWM8A upon T1 event
EPwm2Regs.AQCTLA2.bit.T2U = AQ_CLEAR;      // set PWM8A upon T2 event
EPwm2Regs.AQCTLB2.bit.T1U = AQ_SET;        // clear PWM8A upon T1 event
EPwm2Regs.AQCTLB2.bit.T2U = AQ_CLEAR;      // set PWM8A upon T2 event
}

Cmpss2Regs.DACHVALS.bit.DACVAL    = 2048+comp_mean2_0+comp_dev2;
Cmpss2Regs.DACLVALS.bit.DACVAL    = 2048+comp_mean2_0-comp_dev2;

AdcaRegs.ADCINTFLGCLR.bit.ADCINT1 = 1; //clear INT1 flag
PieCtrlRegs.PIEACK.all = PIEACK_GROUP1;
}


void error (void) {
ESTOP0;         // Stop here and handle error
}
```

REFERENCES

[1] P. S. Shenoy, S. Zhang, R. A. Abdallah, P. T. Krein, and N. R. Shanbhag, "Overcoming the power wall: Connecting voltage domains in series," in *Proc. IEEE Int. Conf. Energy Aware Computing*, 2011, pp. 1–6.

[2] P. T. Krein, *Elements of Power Electronics*. Oxford University Press New York, 1998, vol. 126.

[3] R. W. Erickson and D. Maksimovic, *Fundamentals of Power Electronics*. Springer Science & Business Media, 2007.

[4] J. Wei and F. C. Lee, "An output impedance-based design of voltage regulator output capacitors for high slew-rate load current transients," in *Proc. IEEE Applied Power Electron. Conf.*, 2004.

[5] Intel Corp., "Voltage regulator module (vrm) and enterprise voltage regulator-down (evrd) 11.1 design guidelines," Sep. 2009. [Online]. Available: http://www.intel.com/assets/PDF/designguide/321736.pdf.

[6] S. Kapat and P. T. Krein, "PID controller tuning in a DC-DC converter: A geometric approach for minimum transient recovery time," in *Proc. IEEE Workshop Control Modeling Power Electron.*, 2010, pp. 1–6.

[7] S. Kapat and P. T. Krein, "Improved time optimal control of a buck converter based on capacitor current," *IEEE Trans. Power Electron.*, vol. 27, no. 3, pp. 1444–1454, 2012.

[8] S. Kapat, P. S. Shenoy, and P. T. Krein, "Near-null response to large-signal transients in an augmented buck converter: A geometric approach," *IEEE Trans. Power Electron.*, vol. 27, no. 7, pp. 3319–3329, 2012.

[9] S. Kapat and P. T. Krein, "Null response to a large signal transient in an augmented dc-dc converter: A geometric approach," in *Proc. IEEE Workshop Control Modeling Power Electron.*, 2010, pp. 1–6.

[10] A. Pascual, G. Eirea, and E. Ferreira, "A control strategy for multi-phase buck converters under dynamical selection of active phases," in *Proc. IEEE Energy Convers. Congr. and Expo.*, 2009, pp. 1112–1117.

[11] Z. Lukić, Z. Zhao, A. Prodić, and D. Goder, "Digital controller for multi-phase dc-dc converters with logarithmic current sharing," in *Proc. IEEE Power Electron. Spec. Conf.*, 2007, pp. 119–123.

[12] J. Abu-Qahouq, H. Mao, and I. Batarseh, "Multiphase voltage-mode hysteretic controlled dc-dc converter with novel current sharing," *IEEE Trans. Power Electron.*, vol. 19, no. 6, pp. 1397–1407, 2004.

[13] P.-L. Wong, P. Xu, B. Yang, and F. C. Lee, "Performance improvements of interleaving vrms with coupling inductors," *IEEE Trans. Power Electron.*, vol. 16, no. 4, pp. 499–507, 2001.

[14] J. Li, A. Stratakos, A. Schultz, and C. R. Sullivan, "Using coupled inductors to enhance transient performance of multi-phase buck converters," in *Proc. IEEE Applied Power Electron. Conf.*, vol. 2, 2004, pp. 1289–1293.

[15] G. R. Walker and P. C. Sernia, "Cascaded dc-dc converter connection of photovoltaic modules," *IEEE Trans. Power Electron.*, vol. 19, no. 4, pp. 1130–1139, 2004.

[16] T. Shimizu, M. Hirakata, T. Kamezawa, and H. Watanabe, "Generation control circuit for photovoltaic modules," *IEEE Trans. Power Electron.*, vol. 16, no. 3, pp. 293–300, 2001.

[17] G. L. Brainard, "Non-dissipative battery charger equalizer," U.S. Patent 5 479 083, Dec. 26, 1995.

[18] C. Pascual and P. T. Krein, "Switched capacitor system for automatic battery equalization," U.S. Patent 5 710 504, Jan. 20, 1998.

[19] J. McClurg, R. C. N. Pilawa-Podgurski, and P. S. Shenoy, "A series-stacked architecture for high-efficiency data center power delivery," in *Proc. IEEE Energy Convers. Congr. and Expo.*, 2014, pp. 170–177.

[20] E. Candan, P. S. Shenoy, and R. C. Pilawa-Podgurski, "A series-stacked power delivery architecture with isolated differential power conversion for data centers," *IEEE Trans. Power Electron.*, vol. 31, no. 5, pp. 3690–3703, 2016.

[21] S. Rajapandian, K. L. Shepard, P. Hazucha, and T. Karnik, "High-voltage power delivery through charge recycling," *IEEE J. Solid-State Circuits*, vol. 41, no. 6, pp. 1400–1410, 2006.

[22] P. S. Shenoy and P. T. Krein, "Differential power processing for dc systems," *IEEE Trans. Power Electron.*, vol. 28, no. 4, pp. 1795–1806, 2013.

[23] S. K. Lee, T. Tong, X. Zhang, D. Brooks, and G.-Y. Wei, "A 16-core voltage-stacked system with an integrated switched-capacitor dc-dc converter," in *Proceedings of the IEEE Symposium on VLSI Circuits (VLSIC15)*, 2015.

[24] A. Stillwell and R. C. Pilawa-Podgurski, "A resonant switched-capacitor converter with GaN transistors for series-stacked processors with 99.8% power delivery efficiency," in *Proc. IEEE Energy Convers. Congr. and Expo.*, 2015, pp. 563–570.

[25] P. Shenoy, "Improving performance, efficiency, and reliability of dc-dc conversion systems by differential power processing," Ph.D. dissertation, University of Illinois at Urbana-Champaign, 2012.

[26] E. Candan, "A series-stacked power delivery architecture with isolated converters for energy efficient data centers," M.S. thesis, University of Illinois, Urbana-Champaign, 2014.

[27] C. Pascual and P. T. Krein, "Switched capacitor system for automatic series battery equalization," in *Applied Power Electronics Conference and Exposition, 1997. APEC'97 Conference Proceedings 1997., Twelfth Annual*, vol. 2, 1997, pp. 848–854.

[28] A. Stilwell, "Differential power processing for series-stacked processors," M.S. thesis, University of Illinois, Urbana-Champaign, 2015.

[29] D. Das and P. T. Krein, "Voltage regulation of a series-stacked system of processors by differential power processing," in *Proc. IEEE Workshop Control Modeling Power Electron.*, 2015, pp. 1–7.

[30] C. Schaef and J. T. Stauth, "Efficient voltage regulation for microprocessor cores stacked in vertical voltage domains," *IEEE Trans. Power Electron.*, vol. 31, no. 2, pp. 1795–1808, 2016.

[31] G. F. Franklin, J. D. Powell, and M. Workman, *Digital Control of Dynamic Ssystems*. Ellis-Kagle Press, 1998.

[32] A. Prodić and D. Maksimović, "Design of a digital PID regulator based on look-up tables for control of high-frequency dc-dc converters," in *Proc. IEEE Workshop Comput. in Power Electron.*, 2002, pp. 18–22.

[33] P. T. Krein and R. M. Bass, "Autonomous control technique for high-performance switches," *IEEE Trans. Ind. Electron.*, vol. 39, no. 3, pp. 215–222, 1992.

[34] S. Banerjee and G. C. Verghese, *Nonlinear Phenomena in Power Electronics*, 1999.

[35] R. Redl and J. Sun, "Ripple-based control of switching regulators - An overview," *IEEE Trans. Power Electron.*, vol. 24, no. 12, pp. 2669–2680, 2009.