MCUs HEAT UP Digital Loop Control

By Tom Stamm, Applications Engineer, and **Vipin Bothra**, Power Electronics Applications Group Manager, STMicroelectronics, Schaumburg, Illinois

Conventional MCUs meet the digital-control requirements for ac-dc power supplies, including sufficient processing time to do the usual monitoring and communication tasks after the control-loop calculations.

Today's microcontroller units (MCUs) can implement the control functions and feedback loops of off-line ac-dc power converters, which do not require extremely fast control loops. As an example, a relatively simple off-line converter can employ a conventional microcontroller instead of specialized DSP, FPGA or chips containing complex programmable delay lines. Hardware engineers with little or no MCU experience will be surprised at how easy it is to develop such a system.

Response Time is Hardware Limited

Off-line power converters typically have response times limited by their output-filter components. Extremely fast response in the control circuit is not required, so a conventional MCU can operate the feedback loop.

Typically, the output filter (Fig. 1) is the limiting factor in the control-loop design. The inductor is selected to give reasonable ripple current in the output capacitor. In our example, load voltage is 12 V and load current is 4 A. The choke value would be set to give a peak-to-peak ripple current of one-fourth the rated load current, allowing continuous-mode operation down to one-eighth of the rated load current. The choke value is selected based on the ripple frequency. For our 150-kHz converter, the value is 50 μH.

A low-cost electrolytic typically would be selected for the output filter. The electrolytic's equivalent series resistance (ESR) sets the peak-to-peak ripple voltage. A typical ripple specification is 1% of the output voltage, or 0.12 V. ESR, therefore, must be less than 0.12 Ω. A low-

cost capacitor of 470 µF and 63 V was used in the demo. It has an ESR of about 0.06 Ω, giving a reasonable safety factor. The high-voltage rating gave some margin during debugging.

Consider the physical limits of the filter's input. Assume a maximum secondary voltage of 48 V set by the transformer turns ratio and source voltage. Since the forward converter has a maximum duty cycle of 50%, the average forcing voltage range for the converter is 0 V to 24 V.

Fig. 2 shows the best damped response time, 425 µs, given the range of forcing voltages. The damping resistor closely matches the impedance of the L-C circuit. In practice, the damping would be applied by the system feedback loop, with the input forcing voltage (duty cycle) reduced as the output voltage reaches its setpoint. The damping resistor serves the same function, in that it removes the excess current stored in the inductor as it approaches the setpoint. The response time cannot be improved with the components and forcing voltages selected. And, since design margins are important, the response time will be even slower.

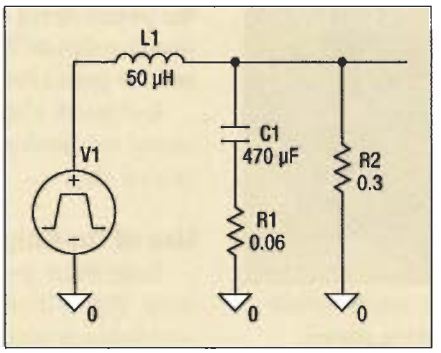

Fig. 1. *SPICE schematic for load-transient simulation. R2, the damping resistor, simulates the linear control loop.*

Demo Verification

To prove that a conventional MCU could do the control job, we constructed a demo. The demo contains an STM32F103 MCU (Table 1), which is largely underemployed. The demo (Fig. 3) consists of two pc boards, a MCU board and the power-converter board. The power converter is a two-switch forward converter, operating from the rectified ac line, feeding an isolated synchronous rectifier and L-C output filter. The system output is 12 V at 4 A. For the benefit of the software side of the development team, we referenced all control to the secondary side, isolated from the line.

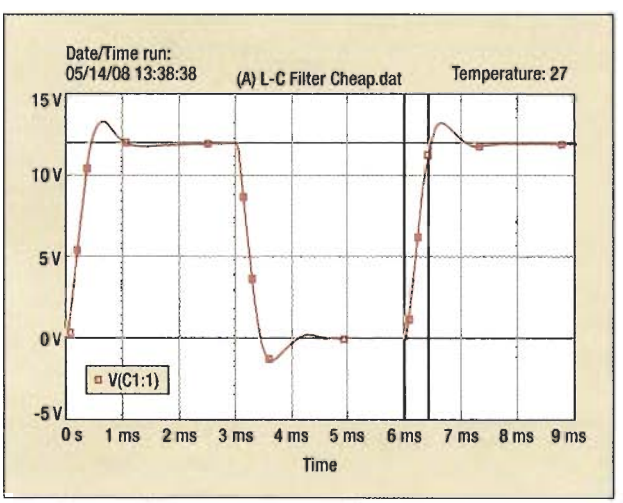

Fig. 2. *Damped transient response for a 6-ms square-wave input applied to the output filter in Fig. 1.*

The MCU demo board has many unused features. Only the MCU chip itself, the debug interface, the reset button and the 5-V to 3.3-V regulator are used for the control loop. All inputs and outputs are taken directly from the MCU pins, with level shifters added to allow interfacing with gate drivers.

Fig. 4 shows the system architecture and the general interfaces to the MCU. The system requires an isolated housekeeping supply, because the MCU cannot bootstrap its own power supply. Q1 and Q2 implement a 150-kHz two-switch forward converter driven in phase by a common transformer fed by the drive chip on the secondary side. (A 1-A fuse in series with the power-transformer primary helped the debugging phase by saving a lot of replacement parts.)

A current transformer isolates the primary current signal. The burden resistor directly feeds an analog comparator (not among the peripherals on the selected MCU die).

Schottky diodes shunt the synchronous rectifiers, which simplifies debugging. You can remove them once the dead-time is set.

The output ORing diode (D10) was used later to aid in debugging a current-sharing scheme using digital communication. Our planned PMBus sharing scheme will not be discussed here.

A voltage divider consisting of R13 and a resistor added to the MCU board senses the output voltage. There is no phase-lead network. Control-loop compensation is handled digitally.

Housekeeping and MCU Power

A simple 3-W flyback converter sources isolated +5 V and +15 V to the system. Its implementation is not impor-

• Core: ARM 32-bit Cortex-M3 CPU – 72-MHz maximum frequency, – Single-cycle multiplication and hardware division
• 32 kbytes to 128 kbytes of Flash memory and 6 kbytes to 20 kbytes of SRAM
• Clock, reset and supply management
• Low power with sleep, stop and standby modes
• 2 × 12-bit, 0 V to 3.6 V, 1-µs ADCs (up to 16 channels)
• 7-channel DMA controller supporting timers, ADCs, SPIs, I²Cs and USARTs
• Up to 80 fast I/O ports
• Debug mode
• Up to 7 timers
• Up to 9 communication interfaces
• CRC calculation unit, 96-bit unique ID

Table 1. *STMicroelectronics' STM32F103 features.*

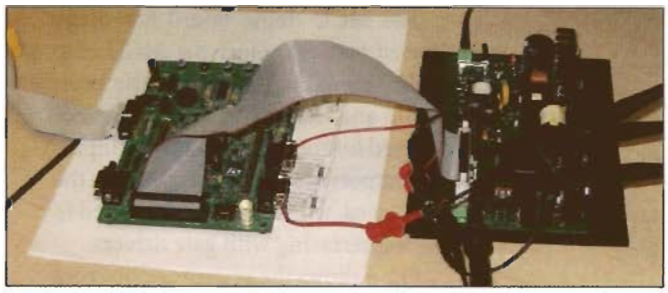

Fig. 3. *The power-converter demo consists of two printed-circuit boards. The STM32 MCU board is on the left and the power-converter board is on the right.*

tant to this discussion; it is part of the power board.

The MCU operates from 3.3 Vdc, derived from a low-dropout regulator on its demo board. The 3.3-V supply is used as a reference voltage for the system; precision was not required for the demonstration, but a precise regulator could be used. The MCU only requires 30 mA to operate; supply loading does not compromise its precision.

The gate drivers used were far from optimum for a 150-kHz converter. They exhibit about a 500-ns delay, which gives excellent noise immunity in their intended motor control application, but the delay is a bad choice here. Both turn-on and turn-off times delay drive to the FETs. The turn-off delay creates some problems with current limiting at low duty cycles, with the output short circuited.

Comparator U7 generates a digital signal for cycle-by-cycle peak current-mode control or for current limiting in voltage-mode control. The comparison signal is an analog voltage derived from a 1-MHz pulse-width modulation (PWM) timer output from the MCU.

A ribbon cable carries all signals and power between the power board and the MCU board. Table 2 lists these interface signals. Fig. 5 shows the circuits between the MCU and the power converter's synchronous rectifiers.

It is worth noting that the 12-in. ribbon cable did not create *any* problems. Try *that* distance with an analog control chip.

Use of On-Chip MCU Peripheral Components

Only three peripherals implement the control functions. Fig. 6 illustrates the flow chart and the resulting waveforms produced by the MCU to drive the synchronous rectifiers.

Timer 4 generates a 1-MHz PWM signal that is filtered to a dc voltage for the peak current comparator. The PWM scheme is far from ideal for the task. The filter causes a significant delay after changes in the duty cycle. This could be reduced by having a digital-to-analog converter (DAC)

Signal	Type	Direction	Level
Primary gate drive	Digital	From MCU	0-3.3 V, level-shifted
Series rectifier gate drive	Digital	From MCU	0-3.3 V, level-shifted
Shunt rectifier gate drive	Digital	From MCU	0-3.3 V, level-shifted
Voltage feedback	Analog	To MCU	0-3.3 V from resistor divider
1-MHz PWM	PWM	From MCU	3.3 V, 0-100%
+5-V power	Power	To MCU	+5 V
+3.3-V power, reference	Power	From MCU	+3.3 V

Table 2. *Signals between power pc board and MCU.*

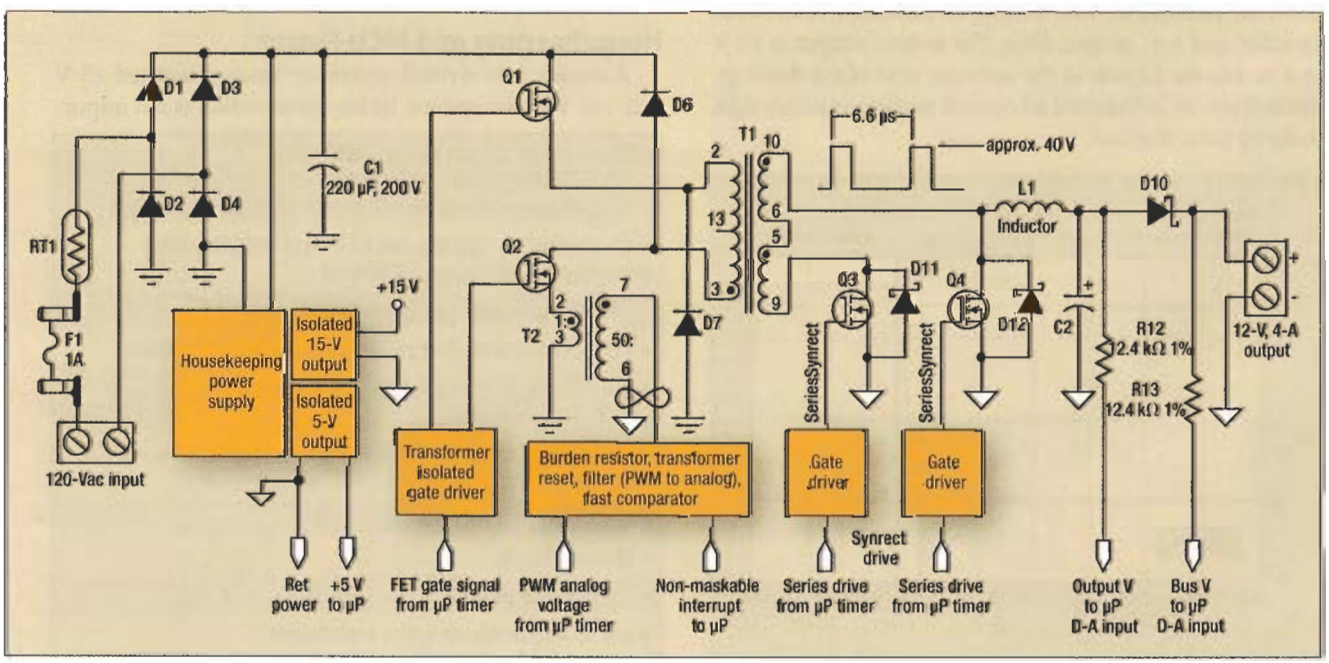

Fig. 4. *The power converter employs a two-switch forward converter using Q1 and Q2 that drives synchronous rectifiers Q3 and Q4 via transformer T1.*

loaded with a new value as each calculation cycle completes. (Planned versions of the MCU will have the digital to analog and an on-board comparator.) A 72-MHz system clock generates the PWM signal, giving a granularity of 1/72 of the maximum voltage. This could have been halved; a frequency doubler is available for the counter. It was not necessary, however, because the power-filter components still limit performance.

Timer 1 is a 16-bit timer that generates the complementary drive signals required by the synchronous rectifiers and primary MOSFETs. It includes four capture-compare (CCP) registers, of which only one is used in the demo. The timer and CCP generate a reference 150-kHz, 50% duty cycle waveform. The current comparator's changing state can also terminate the waveform's on time. A dead-time generator at the CCP output prevents overlap of the on time of the shunt and series synchronous rectifiers. You can optimize the dead-time for different conditions of load, line and temperature, or it can be adjusted by a self-optimizing background control loop.

While the use of this timer is gross overkill for setting the switching frequency, the fine granularity of the dead-time adjustment, 13.9 ns, was used to advantage. To maximize efficiency, the dead-time was adjusted to give minimum input power with a fixed 80% load.

The fine granularity of this timer also can be advantageous if voltage-mode control is used. Again, if necessary, the timer's granularity can be decreased to less than 7 ns by doubling the 72-MHz clock frequency. This could be useful in reducing audible noise from limit-cycle oscillations.

The MCU contains two 12-bit analog-to-digital converters (ADCs). One measures the system output voltage. Only one analog input is used of the 10 available. It is anticipated that another will be used when implementing a current-sharing scheme. The fast 1-µs conversion time allows a sample to be taken at a known point in the power cycle, where switching noise is at a minimum.

This processor core contains a 32-bit single-cycle hardware multiplier, which greatly reduces PID calculation time. Several multiplications are needed in the PID algorithm. It was not necessary to minimize their number by rearranging the algorithm.

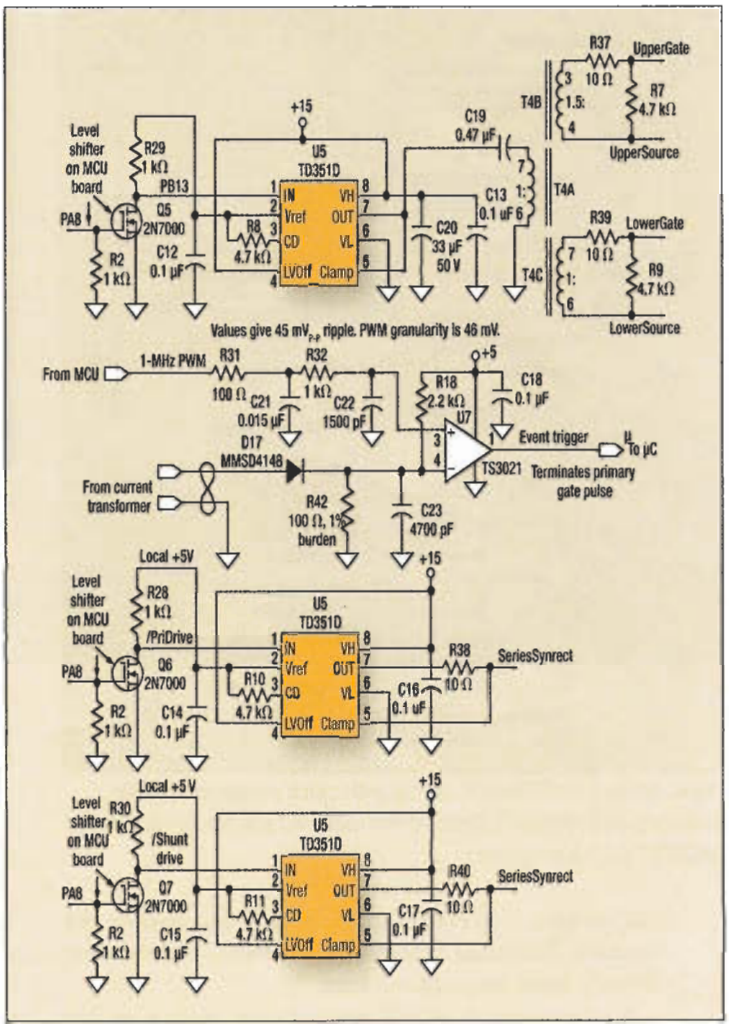

Fig. 5. *MCU board-to-power converter board interface components consist of 2N7000 transistor level shifters, three TD351D MOSFET 1-A sink and 0.75- A source-gate drivers (U4, U5 and U6) and the TS3021 U7 comparator along with resistors, capacitors and diodes.*

Firmware

The entire firmware development was done in C. The code is compiled in a laptop. Fig. 7 lists the PID algorithm's pseudo code, which occupies only six lines of C code.

Flash memory on the µC holds the program and initialization constants. When programming starts, the MCU stops and its output pins go to a high-impedance state, stopping the forward converter.

A debug pod handles the programming with the system powered and live. Programming takes only 15 seconds to 20 seconds, then the reset button on the MCU board is pressed and released. The program starts and the forward converter comes to life.

Initialization (Main Routine)

The two timers are set up first. The system clock frequency of 72 MHz increments the counters. Counter 4's terminal count is set to 72 to give a 1-MHz output. The threshold count is initially set to zero, to give 0 V out of the filter. The PID algorithm will modify this variable later.

The gate-drive outputs are held off until the last step of the initialization.

The initialization of Counter 1 is more complex. The timer is set to count up for 480 clock cycles and reset. The state of the gate-drive outputs must be programmed to zero for the reset state and the idle state, so the converter will not run until the control loop is ready.

When operating, CCP1's reference signal gives a rising edge at counter reset and a falling edge at a threshold count

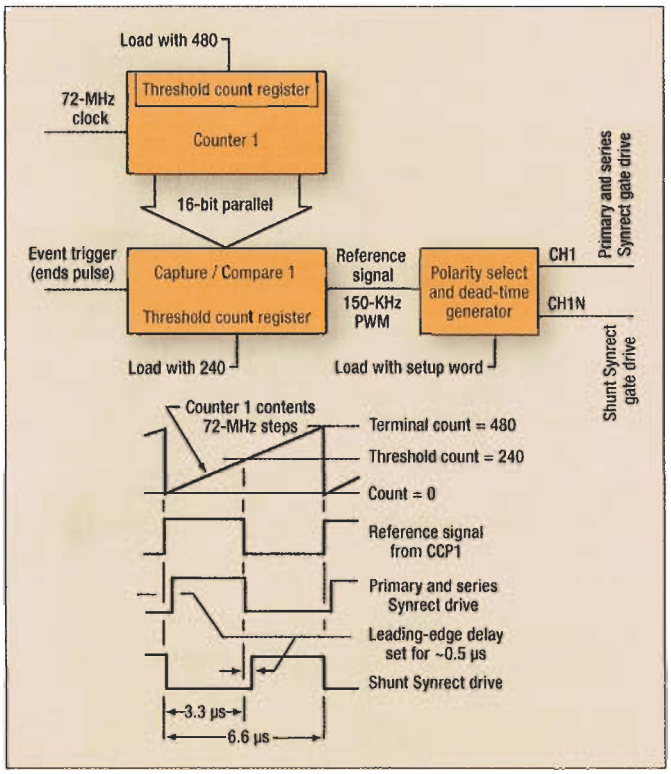

Fig. 6. *Timer 1 and Timer 4, along with clock pulses and other counters and registers, develop the required synchronous rectifier MOSFET gate waveforms.*

(240) to give 50% maximum duty cycle required by the converter. This could be increased somewhat, as dead-time subtracts from the actual on time.

The comparator on the power board feeds Counter 1's event trigger to terminate each cycle in advance of the threshold count. The event trigger is programmed to terminate the reference pulse on the first edge. Multiple transitions are not a problem.

In current-mode control, the current threshold and the rising current determine when the on period ends. In voltage-mode control, the current threshold is set above the maximum required load current. In either case the same hardware is used for peak current limiting.

PID Control Algorithm

The PID algorithm is implemented in integer math and runs once each 6.6-µs cycle, interrupt triggered by the Timer 1 terminal count reset. The execution time was measured at 4.4 µs, with the code written in C and not optimized. That leaves 2.2 µs still available in each cycle for other tasks, the equivalent of a 22-MHz processor. It is not even necessary to run the algorithm each cycle, if the control system can tolerate the added delay.

```
On timer interrupt, do:
THIS_ERROR = OUTPUT VOLTAGE MINUS SETPOINT
ERROR_PROPORTIONAL = GAIN_PROPORTIONAL × THIS_ERROR
ERROR_DERIVATIVE = GAIN_DERIVATIVE × (THIS_ERROR - LAST_ERROR)
ERROR_INTEGRAL = GAIN_INTEGRAL × (ERROR_INTEGRAL + THIS_ERROR)
TOTAL_ERROR_OUT = ERROR_PROPORTIONAL + ERROR_DERIVATIVE + ERROR_INTEGRAL
Adjust DC level (calculate FAST_TIMER_DUTY):
FAST_TIMER_DUTY = SCALE_CONSTANT × TOTAL_ERROR_OUT
LAST_ERROR = THIS_ERROR
```

Fig. 7. *The PID algorithm's pseudo code occupies only six lines of C code.*

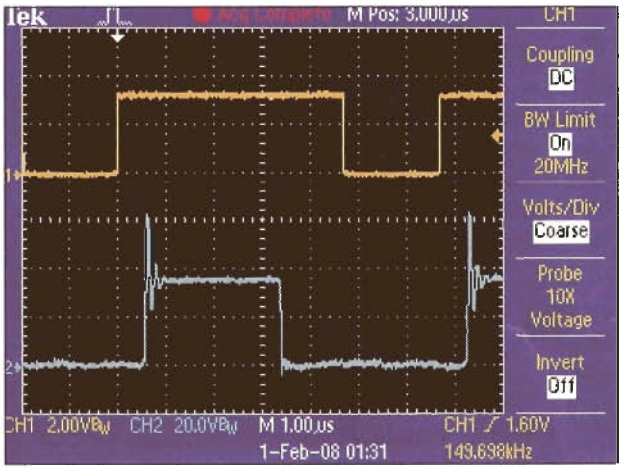

Fig. 8. *PID calculation time versus cycle time shows that the digital output (yellow) goes high at the beginning and low at the end of the PID interrupt service routine. The cyan square wave is the 6.6-µs power cycle.*

Fig. 8 shows a digital output (yellow) programmed to go high at the beginning and low at the end of the PID interrupt service routine, compared to the 6.6-µs power cycle (cyan). The output voltage is sampled just before the turn-on transient (first rising edge of cyan trace) in the dead-time interval.

The PID C routine was later compiled with the "Optimize for Speed" option set. The calculation time dropped from 4.4 µs to 1.8 µs.

Debugging

The timer routines were debugged without the power board. Everything possible was checked before connecting the two boards. In addition, a removable fuse was added to one of the power transformer primary connections. With the fuse removed, all the gate signals could be safely tested with power applied to the board.

The ADC was checked with an external power supply. The current-limit comparator and PWM current compare source were checked in the same way.

The fuse was then inserted, and the system debugged in operation. Several fuses and a few synchronous rectifier

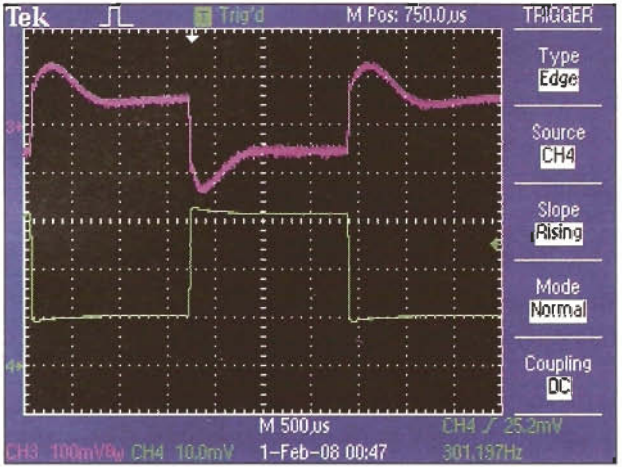

Fig. 9. *The demo unit's transient response is shown for a 2-A transient on a 1-A load. Load current (green trace) is measured on a 1-A/div. scale, while load voltage (magenta trace) is measured on a 100-mV/div. scale.*

MOSFETs were sacrificed in learning about the gate-driver delay time.

The PID routine's gain constants were adjusted using the seat-of-the-pants method. The gain margin had to be increased to accommodate the discontinuous-conduction mode at low load current. This could have been done in software, but we haven't gotten there yet. Fig. 9 shows the demo's transient response with a 2-A transient on a 1-A load.

Two factors contribute to poor current limiting at low output voltages (near short circuit): delay in the gate-driver chips (mentioned previously) and leading-edge spike on current waveform at the burden resistor.

The system is less stable when in discontinuous-conduction mode. This still needs work.

Overshoot on turn-on was a serious problem. It was solved by adding a conditional call to the PID algorithm. The method used is to call a separate routine to drop and slowly raise the output-voltage setpoint if the output voltage is measured to be below a percentage (90%) of the operating setpoint.

The output voltage is measured each cycle. If it is below 90% of the setpoint, the setpoint is reduced to 10% of the correct value, and incremented periodically by 10% until the output is within the correct range. Overshoot still exists, but it's smaller; ugly, but effective. **PETech**

References

1. UM0427 STM32 user manual.
2. STM32F103x6, STM32F103x8, STM32F103xB data sheet.
3. RM0008 reference manual.
4. AN2580 STM32F10xxx TIM1 application examples.
5. UM0426 STM3210B-EVAL evaluation board user manual.