Gecko "VAMPIRE"

by Fred Eady

"In Transylvania, cloves of garlic, sharp wooden stakes, hawthorn branches, and a cross are just a few of the necessary tools one needs to combat vampires ..."

Running water, daylight and holy silver bullets are also potent vampire killers. If your vampire pest was once a mortal human being (most all of them were), every vampire killing tool I've just called out will do the job. However, if your vampire smells of silicon and steel, forget about getting any killer results from the garlic and wood as only a steady stream of running water and a carefully placed bullet will kill a G203V Gecko Vampire stepper motor drive.

My soldering iron is still hot from our last project. So, put some water on your soldering iron tip cleaning sponge and get your Moto-Tool ready to grind and spin some drill bits. Fire up your soldering iron as this month we're going to scratch build a PIC-based step and direction controller that will act as an intelligent front end to a Gecko G203V stepper motor drive.

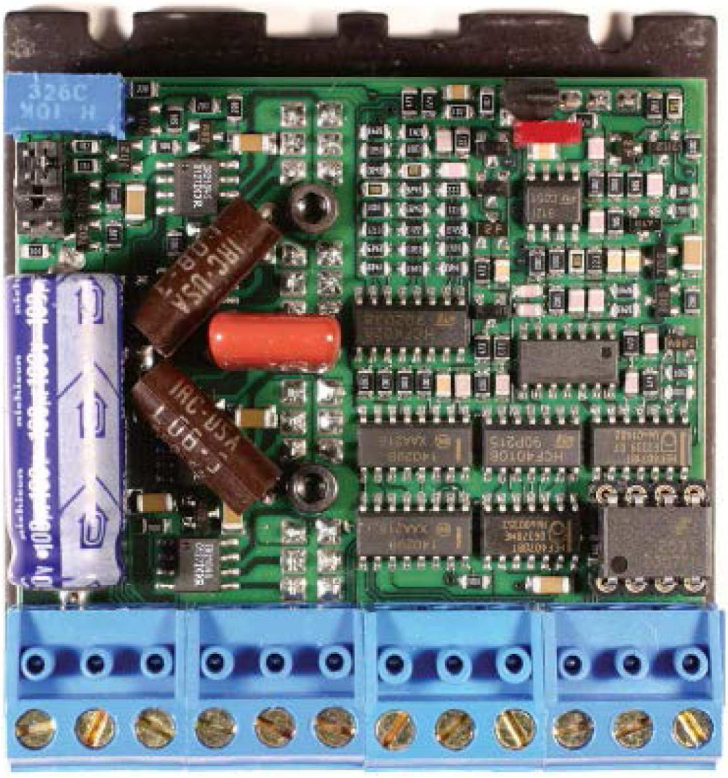


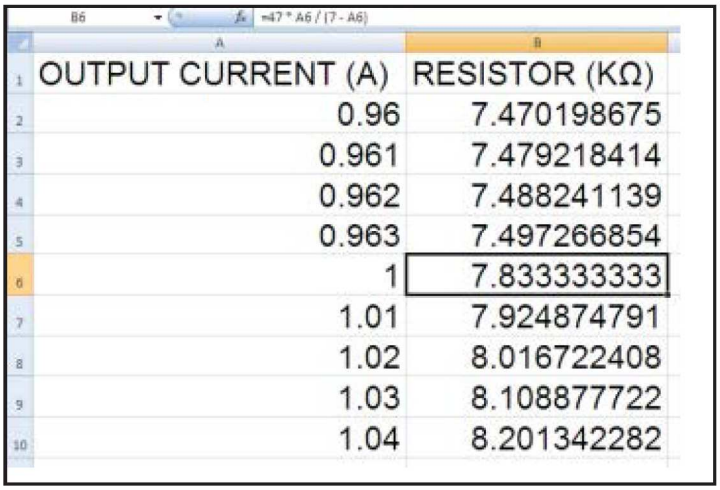
PHOTO 1. Discrete CMOS circuitry rules the G201. All of the Geckos are equipped with an integral heatsink, which has been machined for easy mounting to a larger heat-conducting surface.

Enter the Vampire

The "V" in G203V does indeed stand for "Vampire." The idea is that the Gecko G203V is virtually indestructible. To understand why, we must examine the G203V's ancestral roots. The Gecko G203V is really an armored Gecko G201. The G201 is the most basic and eldest of the line of GeckoDrives. Unlike its vampire cousin, you can kill the G201 if you are not careful.

I've removed the cover from my G201 in Photo 1 to give you an idea of where the Gecko G203V came from. Note the heavy use of discrete CMOS logic within the innards of the G201. Despite CMOS circuitry being able to eat just about anything thrown at it, the STEP and DIRECTION inputs of the G201 are optically isolated. The idea is to sacrifice the Gecko and save the controller if something goes awry at the stepper motor end. You can easily see the G201's socketed eight-pin optoisolator

SCREENSHOT 1. If you haven't discovered the beauty of an Excel spreadsheet in an electronic application, behold. This took all of two minutes to formulate and compute.



	A	B
1	OUTPUT CURRENT (A)	RESISTOR (KΩ)
2	0.96	7.470198675
3	0.961	7.479218414
4	0.962	7.488241139
5	0.963	7.497266854
6	1	7.833333333
7	1.01	7.924874791
8	1.02	8.016722408
9	1.03	8.108877722
10	1.04	8.201342282

IC in the lower right corner of Photo 1.

A separate set of stepper motor power supply inputs are provided through the far left pair of removable screw terminals. Even though you see a 100 µF electrolytic capacitor in the proximity of the stepper motor power input pins, you must still install a 470 µF across the stepper motor power input pins if the bulk stepper motor power supply is more than one foot away from the G201. The stepper motor bulk power source must be an unregulated supply whose output falls between +24 VDC and +80 VDC.

When operated within the specified voltage range, the G201 can supply as little as 300 mA or as much as 7.0A to the stepper motor coils. The maximum amount of current delivered by the G201 is governed by a standard 0.25W 5% resistor, which connects between the pair of screw terminals located at the extreme right corner of Photo 1. The resistance values for a corresponding current output are silkscreened onto the G201's cover. However, you can calculate your own current limit resistor value using the following formula:

For output currents between 0.3A and 2.0A:

R = 47 * I /(2-I)
where R = resistance in KΩ
I = desired output current

For output currents between 1.0A and 7.0A:

R = 47 * I /(7-I)
where R = resistance in KΩ
I = desired output current

When it comes to math, I believe half of what I see and all that I can compute. So, it seems that if we calculate the resistor value for a motor current of 1.0A, we will get a different result depending on which of the current limit resistor formulae we use. For instance, let's calculate the current limit resistor for a 1.0A load using the 300 mA to 2.0A formula:

R = 47 * 1 / (2-1) = 47K

Now let's do the same using the 1.0A to 7.0A formula:

R = 47 * 1 / (7-1) = 7.83K

The 47K value is a standard off-the-shelf 5% resistor value. The 7.83K value is not. The next lower 5% value to 7.83K is 7.50K. Let's use the power of Bill's Excel to get an idea of how close the 7.50K resistor value is to our target motor current. Take a look at Screenshot 1, which is a run of the 1.0A to 7.0A formula around the center value of 7.83KΩ. I condensed the spreadsheet data output as we're only interested in obtaining a good resistor match for a 1.0A output current. The next common 5% resistor value above 7.83KΩ is 8.20KΩ. As you can see in Screenshot 1, you can select an output current value that is about 4 mA below the target 1.0A output current using a 7.50KΩ resistor.

The alternative option is to make the 1.0A target and overshoot it by 4 mA with an 8.20KΩ resistor. I know that I have a 7.50KΩ 5% resistor in my inventory. If you're not a spark head like me, you may not have a full set of the common 5% resistor values. You're more likely to find that 8.20KΩ resistor on the wall at RadioShack and I'll bet that's why the 8.20KΩ value is stamped on the G201 cover as the 1.0A current limit resistor.

PHOTO 2. Here's a look at the G201 as it would appear out of the box. The screw terminals are removable to aid in mounting the G201 to an external heatsink.

The G201 also has other knobs that you can turn. The potentiometer located at the top right of Photo 1 is used to dampen motor noise at low rotational speeds. Just below the potentiometer is a jumper block that is used to select normal current range (1.0A to 7.0A) and reduced current range (0.3A to 1.0A). Automatic current reduction is also in the jumper selections.

However, when reduced current range is selected (no jumpers), you can't jumper automatic current reduction into the mix. Automatic current reduction reduces the motor phase current to 33% of the set value one second after the last step pulse when the motor is stopped. Our G201 is jumpered for normal current range operation with automatic current reduction enabled.

The G201's maximum step frequency is 200 kHz with steps generated on the falling edge of each step pulse. A G201 step pulse must be logically low for at least 0.5 µS and logically high for a minimum of 4 µS. Driving the G201's optically isolated control inputs requires a +5 VDC power source as the optoisolator LEDs are arranged in a common anode configuration.

The current sink drive requirement for each G201 optoisolated input is 16 mA. Any Microchip PIC can easily handle sinking this much current but other microcontrollers may have a problem with sinking 16 mA without a current-sinking buffer connected between the microcontroller's I/O pin and the G201's optoisolator input.

Photo 2 fills in the blanks. The location of the bipolar stepper motor phase connections is revealed, as well as the hardware logic behind the optoisolated control inputs. The G201 must be attached to an external heatsink if you intend to drive a stepper motor at 3.0A or more. The screw terminals are removable to accommodate attaching the G201's integral heatsink to a bigger chunk of metal. The G201 DISABLE input is not optoisolated and when grounded, forces all of the motor winding currents to zero.

The G201's DIR (DIRECTION) input requires some special handling. The direction can only be changed within a window that begins with the falling edge of the STEP pulse. Ideally, the direction should be changed simultaneously with the falling edge of the STEP pulse. This really isn't as tricky as it sounds. If a PIC is used to generate the G201 control signal and the G201 STEP input is driven with a PIC PWM subsystem signal, each time the PIC timer value matches the PWM period timer the PWM output goes logically low and an interrupt is generated. We can use the interrupt to trigger the execution of an interrupt handler that controls the G201 DIRECTION input.

Okay, now that you're checked out on the G201, please divert your attention to Photo 3, which happens to be the bowels of the Vampire. The very first noticeable difference is that the number of CMOS IC devices has been drastically reduced. Also, the G201's large thru-hole current sense resistors have been replaced in the G203V by physically smaller SMT current sense resistors.

Eight power MOSFETs are mounted under the printed circuit board (PCB) you see in Photo 3. You can mentally visualize the MOSFETs' orientation by following down the groups of six solder joints to the immediate right of the SMT current sense resistors. The MOSFETs are laid out in a left-to-right manner with their pins meeting at the six-pack solder pad spine that runs from the top to the bottom of the Gecko G203V PCB. If you look back at Photo 1, you'll see that the MOSFET layout you see here for the Gecko G203V is a holdover from the G201.

Another commonality of the G210 and G203V can be more easily seen in Photo 3 than in Photo 1. There are four IR2104S half-bridge drivers servicing the eight MOSFETs. Using my fingers and toes, that adds up to a pair of complete H-bridges with four MOSFETs and two IR2104S devices per H-bridge. The IR2104S devices are located directly to the left of the 120 µF electrolytic capacitor in the lower left corner of Photo 3.

The majority of the CMOS logic we saw in Photo 1 has been relegated to the Xilinx XC2C64A CoolRunner-II CPLD. We won't get heavy into CPLD theory here as I will be doing some Xilinx CPLD stuff over in *Nuts & Volts* (**www.nutsvolts.com**). Let's just say that the CoolRunner-II CPLD has been programmed to replace the logical work done by all of those CMOS ICs you see on the G201 PCB in Photo 1.

Now for the things we can't see. The Gecko G203V is short-circuit protected. You can even attach the stepper motor power backwards and not release the magic smoke. If you apply too high of a voltage or try to cook the Gecko G203V, it will go into survival mode to fight another day.

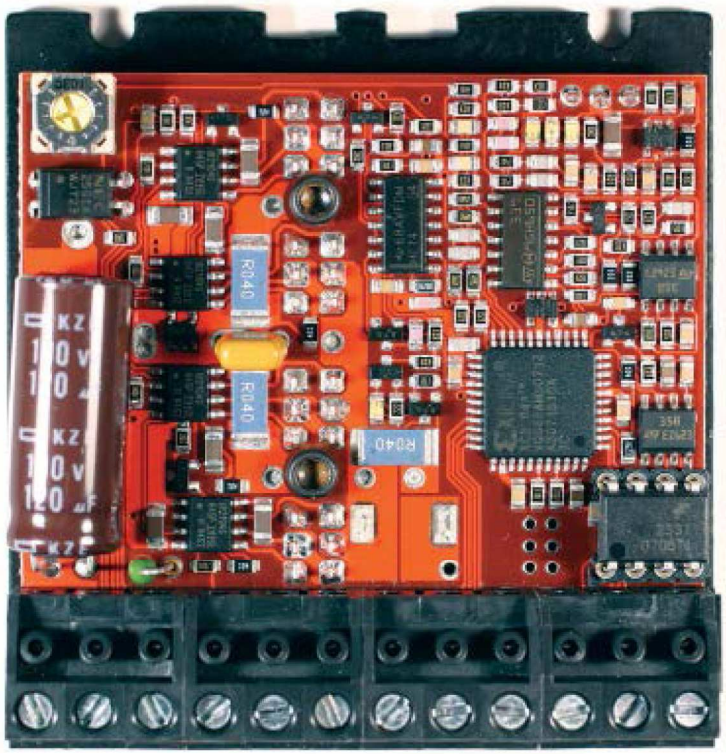


PHOTO 3. A Xilinx CoolRunner-II CPLD on the Gecko G203V printed circuit board has replaced the bulk of the CMOS ICs that were the mainstay of the G201 circuitry. Note the more ergonomical orientation of the motor dampening potentiometer.

The Gecko G203V control inputs are all optically isolated and referenced to the controller's common ground instead of the controller's common +5 VDC.

This control input arrangement relieves the burden on the microcontroller's I/O as the Gecko G203V inputs only require 2.5 mA of current to drive the optoisolator LEDs. Another advantage of the G203V's common cathode optoisolated input scheme is that the STEP, DIRECTION, and DISABLE inputs can operate at 2.5V, 3.3V, and 5.0V logic levels.

The G203V's maximum step frequency is 350 kHz versus the G201's 200 kHz. In addition, the direction change restriction has been lifted as the G203V can process a change direction command at any time with 200 ns setup and hold timing.

The G201 current limiting logic we discussed, as well as the minimum and maximum phase current output amperage, has been carried over into the G203V design. The same holds true for the dampening potentiometer, which has been reoriented on the G203V for easier user accessibility.

There is no need to add the external 470 µF electrolytic capacitor to a G203V design and the G203V's minimum stepper motor voltage is now +18 VDC instead of the +24 VDC minimum that is required by the G201.

A look at the upper left corner of Photo 3 reveals what looks to be an NEC PS2501A photocoupler, which is standing in place of the G210 jumper block we saw in Photo 1. The G203V has no adjustments or jumper settings other than the motor dampening potentiometer and the current limit resistor.

Hey, you're beginning to grow a tail! That's okay, as at this point we know enough about the G201 and G203V to put them both to use, if we desire. Since we have the beginnings of a Gecko look and a couple of GeckoDrives in our hands, let's do something with a soldering iron instead of selling car insurance on television.

Designing a G203V Controller

There is one fundamental design point that you must understand to design with a GeckoDrive. Take a look at the Gecko G203V shown in Photo 4. The G203V POWER GROUND input is not electrically connected to the optoisolated COMMON input. The idea is to provide an unregulated bulk power supply that is suitable to drive your stepper motor. Attach the unregulated bulk power supply to the G203V POWER GROUND and +18 TO 80 VDC terminals. The G203V COMMON input is meant to be attached to the ground system of the low-voltage power supply that is servicing the controller.

In our case, the controller is actually a PIC18F2620 and its associated power supply, clock circuitry, and I/O subsystem. Depending on your requirements, the PIC18F2620 can be replaced with a PIC18LF2620 to allow the PIC-based G203V controller to operate at voltages below +5 VDC.

Recall that the G203V's optoisolated inputs can work with logic rails that lie below the standard +5 VDC logic levels. With the G203V's COMMON attached to controller ground, the PIC18F2620's I/O pins need only source 2.5 mA of current to the DISABLE, DIR, AND STEP optoisolated inputs to control the stepper motor attached to the Gecko G203V's PHASE outputs.

I've laid out all of the G203V controller connections and components graphically in Schematic 1. The stepper motor bulk power supply can be customized to suit your stepper motor's needs. I used an AMVECO 62084 18 VAC @ 2.7A toroid transformer to drive the GBU601 full wave bridge and filter capacitor you see in Photo 5.

Be sure to choose a working voltage for C4 and C5 that safely exceeds the maximum output voltage at the bridge rectifier. My bulk power supply delivers +28.8 VDC unloaded. The AMVECO toroid can deliver its rated current with no problems. So, be very careful with the bulk supply as it will leave a mark if you short it out.

Note that our linear bulk power supply only feeds the G203V's POWER GROUND and +18 to 80 VDC power input terminals. The power for the PIC-based controller can be had directly from a regulated +5 VDC wall wart or indirectly through the LM2940 fed from a clean +9 VDC to +12 VDC power source. Naturally, all of the power for this project can come from a bank of batteries.

As with the bulk power supply, you can use a microcontroller of your choice. I'm using the PIC18F2620 here because it does not require an external crystal, has

PHOTO 4. The G203V Vampire is pin-compatible with the G201 with the exception of the COMMON terminal, which is referenced to the controller's ground on the Vampire. Note also that the PHASE names are more stepper motor datasheet correct than the G201's PHASE A, PHASE B, PHASE C, and HASE D nomenclature.

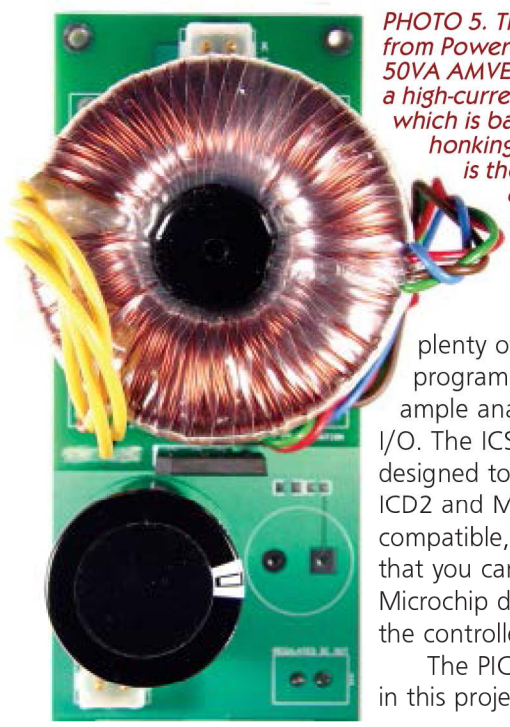


PHOTO 5. This design is taken from Power Supply 101. A 50VA AMVECO toroid is feeding a high-current bridge rectifier, which is backed up by a big honking filter capacitor. This is the meat and potatoes of electronic gear. AC goes in at the top and DC comes out at the bottom.

plenty of SRAM and program Flash, and offers ample analog and digital I/O. The ICSP connection is designed to be Microchip ICD2 and Microchip REAL ICE compatible, which means that you can use regulation Microchip development tools in the controller design process.

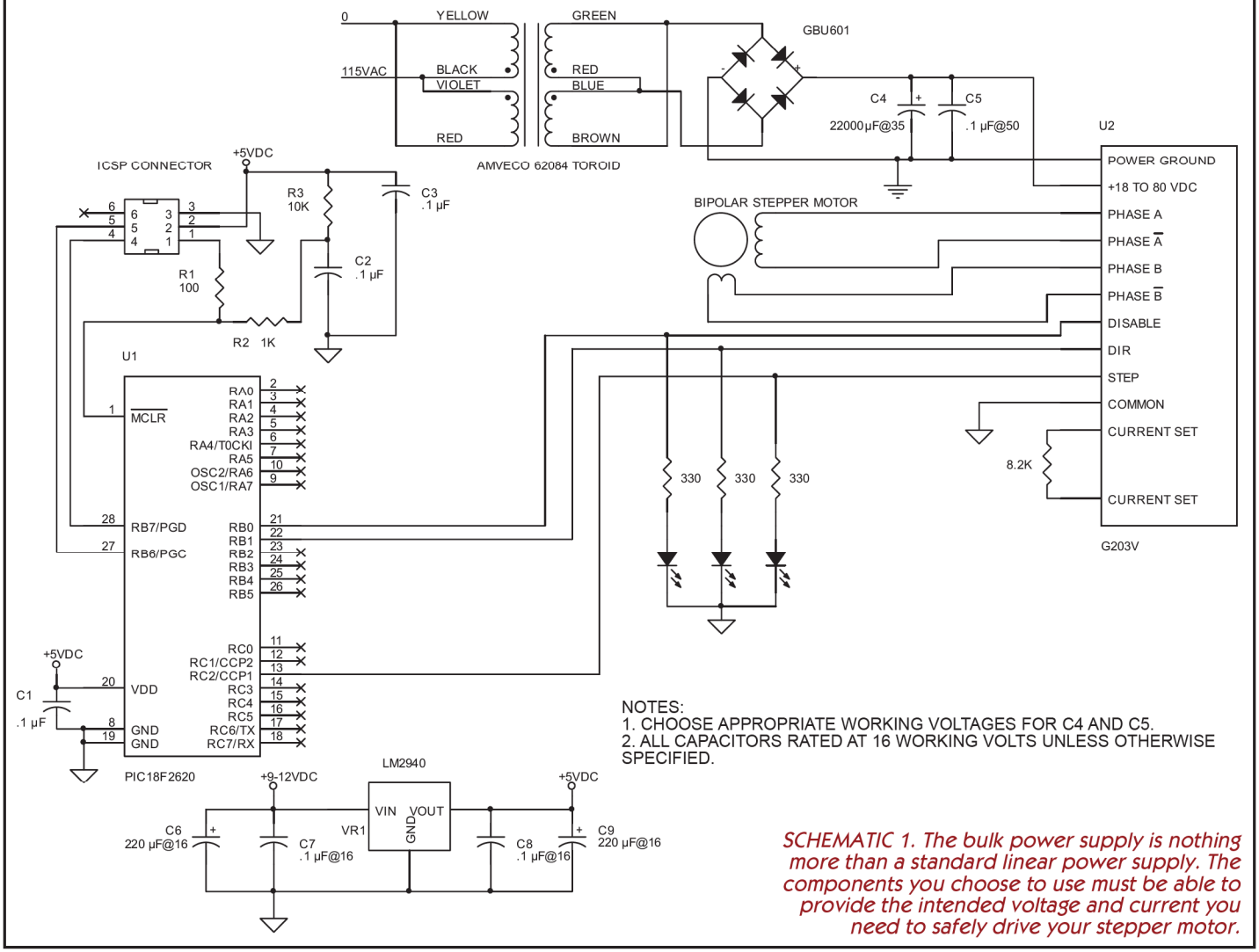
The PIC18F2620 I'm using in this project is socketed. So, if you have a PIC programmer that can program the PIC18F2620, you can eliminate all of the ICSP programming /debugging hardware from your hardware build.

Everything else you see in Schematic 1 should not be a surprise. If it is, I always answer reader email messages. Let's build this thing.

Scratching Up a G203V Controller

The slab of FR4 supporting this project is a custom perfboard design of mine that is available from ExpressPCB. I've provided the custom perfboard's ExpressPCB layout for you as a download from the *SERVO* website (**www.servomagazine.com**). The customization of the perfboard PCB design is really not customization at all. I simply order a PCB full of 0.1 inch centered plated-through holes. The plated-through holes make it really easy to mount components on the top or bottom of the board and make a connection on either side.

I used the plated-through holes to my advantage when assembling the Gecko G203V-to-controller plug-in interface. In Photo 6, I used 5 mm terminal block headers to hook the



SCHEMATIC 1. The bulk power supply is nothing more than a standard linear power supply. The components you choose to use must be able to provide the intended voltage and current you need to safely drive your stepper motor.

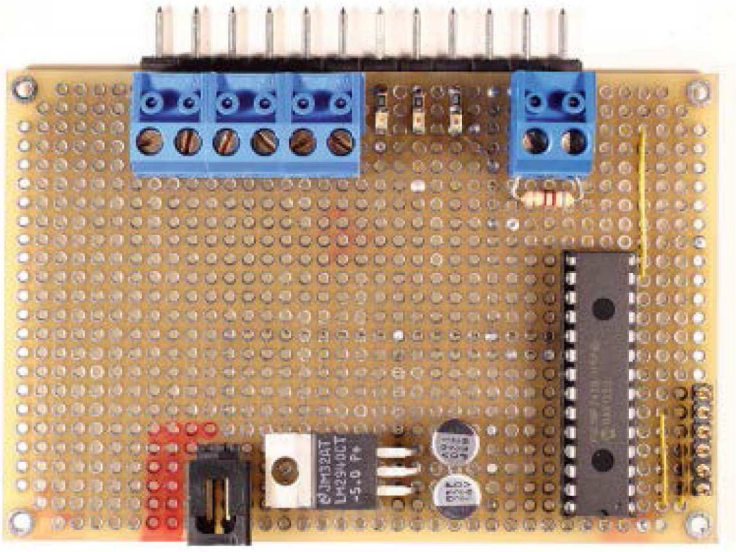


PHOTO 6. The highly integrated Gecko G203V allows this controller project to be built on a simple perfboard with a very small number of supporting components. Most all of the SMT capacitors and resistors (with the exception of the status LEDs and their associated current limiting resistors) are mounted on the bottom side of the perfboard.

G203V up to my PIC18F2620-based Gecko controller. The 5 mm terminal block headers I used just happen to be the same terminal block headers used by the G203V. You can get the 5 mm terminal block headers in 24-pin breakable form from Digi-Key by ordering part number ED1682-ND.

I designed in an ICSP (In Circuit Serial Programming) adapter for the Microchip ICD2 and Microchip REAL ICE that allows the use of the RJ-12 cable that is common to both of the Microchip development tools. I'll make sure to post the adapter ExpressPCB layout file for you on the *SERVO* website.

With the exception of the DISABLE, DIR, and STEP status LEDs and their current limiting resistors, all of the SMT capacitors and resistors are mounted on the underside of the FR4 slab you see in Photo 6. The 220 µF capacitors are actually surface-mount units. I carefully straightened the leads from their original 90° bends and discarded the plastic mounting plates. The newly parallel capacitor leads fit perfectly into the 0.1 inch centered holes of my plated-through perfboard.

To the immediate left of the 220 µF capacitor pair lies the LM2940 low dropout +5V regulator. The LM2940's feeding tube is in the form of a two-pin male BERG connector, which lies just beyond the LM2940's heatsink tab. Note that the 8.20K current select resistor is screwed down with a 5 mm terminal block.

The PIC18F2620 and its ICSP portal are shown at the far right in Photo 6. Once I got everything wired in and tested, I screwed down the Gecko G203V and installed the ICSP adapter. The finished G203V controller and its slave Gecko G203V are shown in Photo 7 along with the PIC18F2620's ICSP adapter.

Since the Gecko G203V is optically isolated from the PIC18F2620-based controller, we can test the controller firmware without having to attach the controller to the Gecko G203V. If you've been wondering what the DISABLE, DIR, and STEP status LEDs were for, there's your answer. We can generate the G203V control signals with the PIC18F2620 and view the results via the status LEDs, which are attached to the actual G203V control input pins. Let's spin up some G203V controller firmware.

Lizard Code

Coding the G203V is just as easy as designing its interface hardware. Let's begin by defining the PIC18F2620-to-G203V interface:

```
//*******************************************************
//*     GECKO CONTROL INTERFACE DEFINITIONS
//*******************************************************
#define DISABLE                 LATB0
#define DIR                     LATB1
#define CW                              1
#define CCW                             0
#define ON                              1
#define OFF                             0
```

There's no rocket science here. The DISABLE and DIR definitions are taken directly from Schematic 1. I've also defined some convenience terms so that our HI-TECH PICC-18 C source code "speaks" to all that we must read and understand.

We can manually generate steps for the G203V. However, that would be a waste as the PIC18F2620 has some very nice timing subsystems. So, we will activate TIMER1 and its interrupt structure for use as our general-purpose delay timer. TIMER2 will be used to support the generation and timing of the G203V step PWM signal. This may be a good time to download the controller source code from the SERVO website as the TIMER1, TIMER2, and interrupt source is listed there for you. What you will see in the Gecko controller

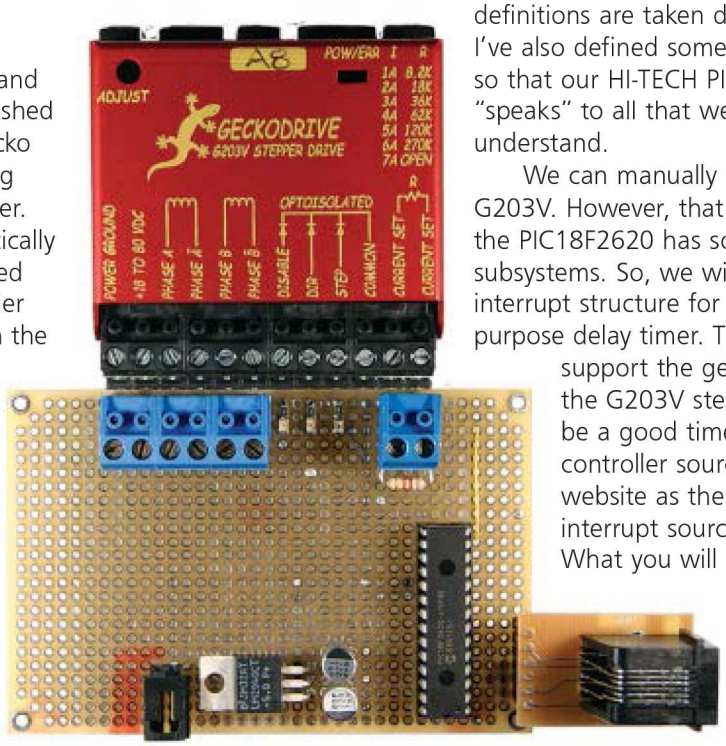


PHOTO 7. All that's left to do before performing the final smoke test is screw in the bulk power supply and the stepper motor.

Resources

GECKODRIVE — **www.geckodrive.com**
GeckoDrive G201; GeckoDrive G203V

Microchip — **www.microchip.com**
PIC18F2620; MPLAB ICD2; MPLAB REAL ICE

HI-TECH Software — **www.htsoft.com**
HI-TECH PICC-18 C Compiler

source code is that a millisecond timer is established using TIMER1, while TIMER2 is configured to support the PIC18F2620's PWM engine.

The PIC18F2620's internal CPU clock is set up to run at 8 MHz sans PLL. Here's a peek at the PIC18F2620 clock and I/O initialization C statements:

```
//*******************************************************
//*      INITIALIZE CLOCK AND IO PORTS
//*******************************************************
        OSCCON = 0x70;
        PLLEN = 0;
        TRISA = 0b11111111;
        TRISB = 0b11111100;
        TRISC = 0b11111011;
```


Another look at Schematic 1 reveals the fact that our PWM signal emanates from CCP1. Thus, it is the only connection and only output pin used on PORTC. Likewise, outputs RB0 and RB1 are the sole Gecko-oriented PORTB I/O pins. That leaves a bunch of communications, and analog and digital I/O for you to play with in your personal Gecko controller application.

If you've got the complete set of Gecko controller code in front of you, you can see that we simply check off the subsystems in the initialization function one by one. We make sure the analog-to-digital converter is disabled as we are not utilizing it. Then, we configure the PWM registers before activating TIMER1 and enabling all of the PIC18F2620's interrupts. Once all of the necessary PIC18F2620 subsystem preparations are complete, we execute the main program code:

```
void main(void)
{
//*******************************************************
//*      INITIALIZE
//*******************************************************
        init();
        DISABLE = OFF;
        DIR = CW;
        sdelay1(2);
        TIMER2ON;
//*******************************************************
//*      MAIN SERVICE LOOP
//*******************************************************
        do{
                sdelay1(5);
                DIR ^= 1;


        }while(1);
}
```

The main loop enables the G203V's PHASE outputs (DISABLE = 0FF), sets the initial stepper motor direction to clockwise (DIR = CW), and enables the PWM (TIMER2ON). The green status LED glows inside the G203V and the stepper motor spins, changing direction every five seconds. What a wonderful world!

That Tail is Getting Longer

FeDEx says use them when it absolutely, positively has to be there overnight. Use a GeckoDrive when your stepper motor absolutely, positively has to turn reliably each and every time.

There's plenty of PIC18F2620 I/O, program Flash, and SRAM left. You've earned the right to call yourself a Gecko expert. So, now it's time for you to put some pushbuttons, potentiometers, and optical switches to work with the G203V and its companion PIC18F2620-based controller and move something. See you next time! SV

Fred Eady can be reached via email at fred@edtp.com