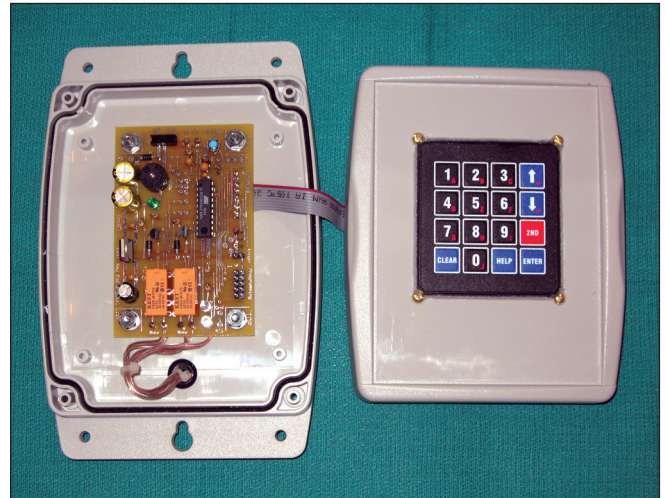


GARAGE ACCESS GOES DIGITAL

BY JAY CARTER, MD

An electric garage door opener is one of the greatest convenience devices ever created! Rain or snow, the weather doesn't matter. You hit a button from inside your car and the door instantly performs as commanded. By adding a digitally controlled keypad, one can make keyless entry easy for children, too.



■ PHOTO 1. The garage access keypad project shown completed and ready for installation. One chip — the ATtiny2313 — runs the show. It scans the keypad, validates an access code, and drives two relays, activating the garage doors. The weatherproof enclosure includes a rubber gasket.

This project uses a single chip — an ATMEL AVR ATtiny2313 microcontroller — to read a keypad and trigger activation of up to two garage doors. The four digit access code can be programmed using the keypad itself, allowing one to easily change the code as desired. A built-in safety feature will stop the door with the single press of any key following door activation. Using only a handful of components, this becomes a truly useful weekend project. Photo 1 shows the project assembled and ready for installation.

Garage Door Interfacing

Most garage doors can be activated via two methods: a doorbell type pushbutton switch or wirelessly. This project mimics the doorbell switch, so it keeps costs down and eliminates the need to know your garage door's wireless frequency and encoding. On the back of most garage door units are two terminals which are wired to the pushbutton switch. The microcontroller activates a small relay when the correct access code is entered. The relay contacts are tied to the same terminals as the pushbutton switch. Enter the code and the microcontroller effectively pushes the button for you.

Reading a Matrix Keypad

The circuit for this project (shown in Figure 1) is centered around the microcontroller. The selected keypad has 16 keys, connected in a 4 x 4 matrix arrangement. Four wires connect to the four rows of keys, while four more wires connect to the four columns of keys. Pressing a key shorts the associated row and column wires together. For example, pressing the "6" key ties the

row 2 and column 3 wires together.

The microcontroller can scan the keyboard and tell if any key has been pressed. To do this, the four column lines are pulled high to +5 volts by resistors R4-R7. The microcontroller outputs a 1, or +5V on each of the four row lines. Then, one row at a time, the microcontroller outputs a 0 on the row line, pulling it low. The microcontroller then reads in the state of each of the four column lines. If one of the keys in that row is pushed, its column will be low. If there are no keys in that row pushed, all four column inputs will read high, being pulled high by the pull-up resistors. As an example, if the "9" keypad is pressed, the row 3 and column 4 lines are tied together. When the microcontroller pulls row 3 low, columns 1, 2, and 3 will read high, while column 4 is now connected to the low level and is read as low.

The microcontroller stores the sequentially pressed keys in a first-in, first-out (FIFO) buffer. The access code is four digits long, followed by the "Enter" key. If more than four keys are pressed, the oldest key press is overwritten, with the program always having the last four keys in the buffer. When "Enter" is pressed, the program checks the last four key presses against the access code. If a valid code was entered, the microcontroller activates a relay and closes its contacts, mimicking a doorbell pushbutton switch being pressed.

Holding the relay closed for about 3/4 of a second works well on my door openers, but the value can be

easily changed within the software.

Debouncing Switches

When one pushes a keypad or closes any switch for that matter, the switch contacts tend to bounce, making and breaking contact several times before assuming their new state. This 'contact bounce' is highly variable between switches, and even varies from activation to activation of the same switch. When viewed with an oscilloscope, one might see the switch open and close a half dozen times or more during its activation. The duration of the individual bounces is variable, and could be up to a millisecond or two, although shorter pulses are common. It may take up to 10 ms or even longer for the switch to assume a stable state. This bouncing occurs with both opening and closing mechanical contact switches.

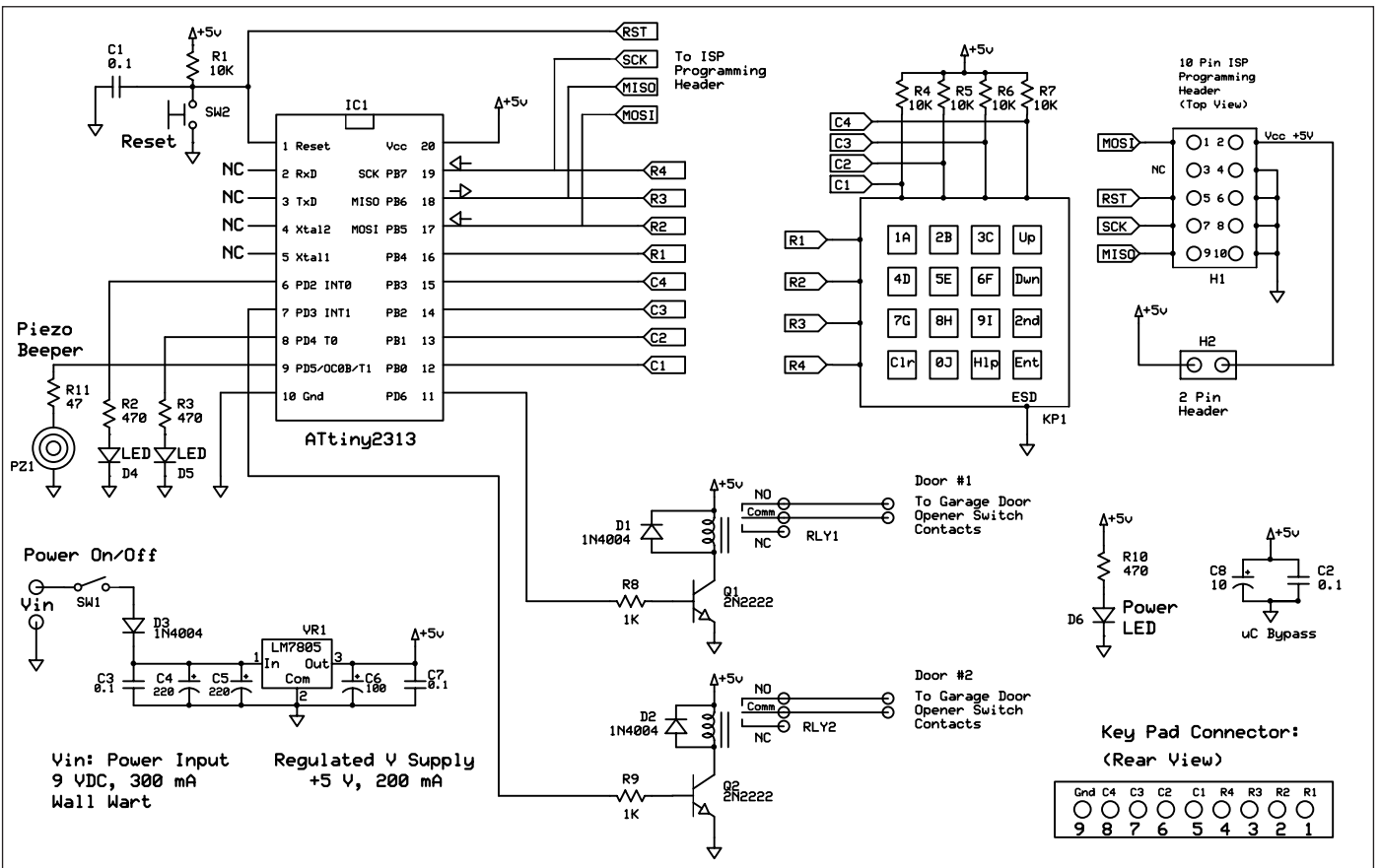
When one turns a ceiling light on or off, the bounce of the switch is imperceptible. When one is entering an access code, however, switch bounce is very important. If the circuitry and software do not account for switch bounce, the microcontroller might register one keypad press as multiple presses of the same key, making it impossible to correctly enter the code. Entering 1, 2, 3, 4 might be read by the microcontroller as if one had entered: 1, 1, 1, 2, 2, 2, 2, 2, 3, 3, 4, 4, 4, 4, 4.

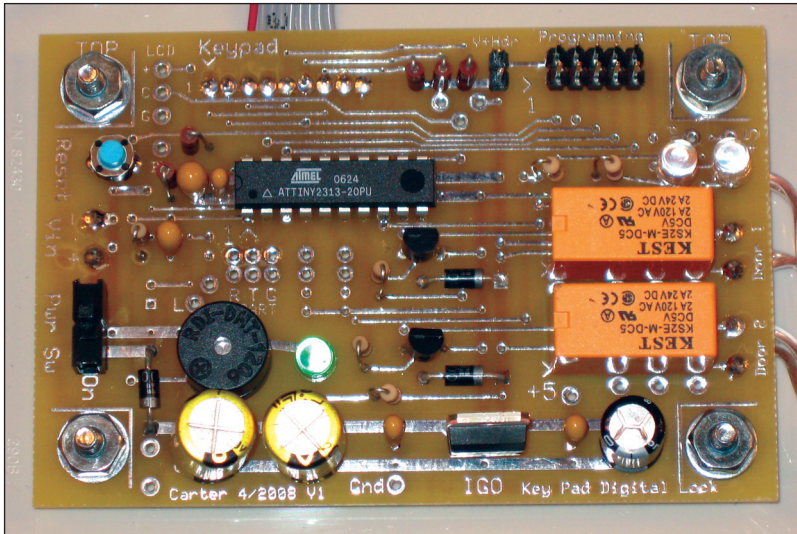
Although one can eliminate the bounce in hardware by placing a capacitor across the switch contacts, this

requires additional components and circuit board space. For high voltage circuits — where each bounce is associated with a small spark and the generation of electromagnetic interference — this approach may be needed. For this project, however, debouncing is done in software.

Many techniques exist for debouncing a switch in software. Perhaps the simplest technique — employed in this case — is a simple timer. Here, when a key is pressed its value is stored and the microcontroller just waits 75 ms before doing anything else. The switch may bounce numerous times, but these will be ignored by the microcontroller. After 75 ms, the microcontroller watches to see when the key is released. It again waits 75 ms to avoid detecting any switch bounce on the keypad release, and then resumes its main loop, watching for key presses. This may seem like a long time interval to wait to insure that one has skipped over any keypad switch bounces, but even taking 150 ms to read each key press would still allow the user to enter the access code at over six digits per second. This is much faster than the normal individual could possibly press the keys. A more efficient algorithm to detect key presses and eliminate the

■ **FIGURE 1.** The garage access keypad schematic. The keypad connects to port B on the microcontroller via a ribbon cable. The microcontroller interfaces with a matrix keypad, relays, LEDs, and a piezoelectric beeper. A generic five volt power supply rounds out the circuit.





■ **PHOTO 2.** A close up view of the ExpressPCB circuit board. The lower row of components comprise a regulated five-volt power supply. The two transistors seen below the microcontroller drive the two orange relays to their right. The programming header is located on the top right corner of the board.

switch bounce could be used if the microcontroller had other tasks to perform. In this case, no other tasks exist, and this simplistic approach works well.

Beep, Beep, Beep

Whenever a key is pressed, the microcontroller generates a short beep via a piezoelectric element. This provides feedback to the user, letting them know that the keypad press was detected. Piezoelectrics are powered with a square wave. This is easily generated by a microcontroller by toggling a digital output pin high and low, generating a string of pulses, ideally at their resonant frequency for clear tone generation. The volume of the tone generated diminishes as one moves off the resonant frequency. A 2 kHz tone played for 200 ms generates a nice feedback for the user. The frequency and duration can be easily changed in software to best match one's specific element and preferences.

The piezoelectric element specified in the parts list draws about 90 mA when powered with a 5V square wave. The ATtiny2313 can source up to 40 mA per pin for driving external devices, (200 mA chip total). The series resistor — R11, 47 ohms — limits the current drawn to 10 mA. One could substitute a 22 ohm resistor, doubling the current, and increasing the volume, if desired.

Relay Driver

The relays selected for this project draw about 40 mA at 5V. This is right at the maximum current allowed per pin on the microcontroller. The microcontroller could power the relay directly, but for long term, reliable operation it is best to avoid operating the chip right at its maximum limits. Designing in a safety margin also protects the chip in case a particular relay actually draws more current than expected.

A 2N2222 transistor is used to drive each relay. A high on the microcontroller pin turns on the transistor

via its base resistor. The microcontroller pin now sources less than 5 mA when driving the transistor — well within the 40 mA limit. The transistor is operated in saturation mode as a switch, being either fully turned on or off. When on, the collector current activates the relay. This general-purpose NPN transistor can handle 600 mA of collector current. It is coasting along at 40 mA, and does not require a heat sink to keep it cool.

Each of the two relays have a reverse-biased diode across their coils (D1 and D2). This is a crucial item when driving an inductive load such as a relay coil. When the relay is turned off, the energy stored in the coil can generate a brief, high voltage spike across its terminals. This spike can easily damage the driver transistor. The diode clamps this spike, protecting the driver circuitry.

Flashing LEDs

The ATtiny2313 has far more digital input/output pins than are needed for this project. Two spare pins were used to drive two LEDs. The current software turns LED #1 or #2 on whenever Relay #1 or #2 are activated. They can be mounted on the project's front panel for additional user feedback, left on the circuit board, or they can be completely eliminated from the circuit. Being under software control, their actual usage is at the discretion of the programmer. The LEDs have a series resistor to limit their current to about 10 mA. This is within the limits of both the LEDs and the microcontroller pin source current maximum of 40 mA.

Microcontroller Clock

All microcontrollers require a 'clock' to run. This clock is actually a square wave which drives the internal CPU and causes the microcontroller to execute sequential instructions in its program. This particular microcontroller can run on either an internal oscillator and external crystal, or an external square wave clock signal. As there was no need for any precise timing for this project, the internal oscillator was selected. The microcontroller is set to use its internal 8 MHz clock source when it is programmed. If one were to modify the project to automatically upload the date, time, door, and access code to a computer logging program, using the microcontroller's serial port one would want to utilize an external crystal for precise timing and reliable serial communications.

Power Supply

The power supply voltage for the ATtiny2313 can range from 2.7 to 5.5 VDC. Five volts was selected for convenience, as 5V relays were used to interface to the garage door controllers. The circuit draws less than 20 mA in its usual state, awaiting someone to enter a keypad code. With all of the peripheral devices (two relays, two software controlled LEDs, the power supply's LED, and the piezoelectric beeper) turned on, the circuit could potentially draw a maximum of about 130 mA. In actual practice, the maximum current drawn by the circuit is less than one half of this value, about 65 mA when one relay is active. The LM7805 is a three-terminal positive voltage regulator which can supply up to 1A of current at 5V. At these low current levels, the regulator runs cool without an attached heatsink.

A 9 VDC, 350 mA wall-wart type power supply is used to power the circuit. The 9V feeds the voltage regulator which powers the remainder of the circuit. Older wall-warts were typically unregulated. Their usage required a voltage regulator to provide a fixed voltage to the microcontroller, and to prevent exposing the microcontroller to an over-voltage. Newer wall-wart power supplies are available which incorporate internal voltage regulators. Using one of that type would eliminate the need for the LM7805.

Outdoor Exposure

By its very nature, this project is mounted outdoors. This allows one to enter an access code and thereby raise the garage door for easy entry. The keypad and enclosure were both selected due to their indoor/outdoor weather rating. The plastic enclosure's top and bottom halves incorporate a rubber gasket to maintain a watertight seal. Additionally, a bead of silicone was placed around the keypad to keep the enclosure watertight. This design utilizes a plastic keypad, plastic enclosure, and exposed mounting screws. A higher level of security and vandalism protection could be provided by using an all metal keypad, metal enclosure, and inaccessible mounting hardware.

This wasn't necessary in my neighborhood.

Software and Features

The heart of any microcontroller project is its software program. AVR microcontrollers are most commonly

PARTS LIST

ITEM	QTY	DESCRIPTION
<input type="checkbox"/> IC1	1	ATtiny2313 (SparkFun)
<input type="checkbox"/> KP1	1	Keypad, 4x4, matrix format (All Elect: Cat# KP-23)
<input type="checkbox"/> VR1	1	LM7805 three terminal linear voltage regulator
<input type="checkbox"/> D4-D6	3	LEDs
<input type="checkbox"/> PZ1	1	Piezo beeper, 5V (All Elect: Cat# PE-52)
<input type="checkbox"/> Q1,Q2	2	2N2222 NPN transistor
<input type="checkbox"/> RLY1, RLY2	2	Relay, 5V coil, DPDT, 16 pin DIP profile (All Elect: Cat# RLY-625)
<input type="checkbox"/> D1-D3	3	1N4004 diode
<input type="checkbox"/> SW1	1	On/off switch, PCB mounted
<input type="checkbox"/> SW2	1	Reset switch, normally open, PCB mounted
<input type="checkbox"/> H1	1	10 pin programming header, PCB mounted
<input type="checkbox"/> H2	1	2 pin header, PCB mounted
<input type="checkbox"/> R1, R4-R7	5	10K
<input type="checkbox"/> R2, R3, R10	3	470Ω
<input type="checkbox"/> R8, R9	2	1K
<input type="checkbox"/> R11	1	47Ω
<input type="checkbox"/> C4, C5	2	220 μF 16V
<input type="checkbox"/> C6	1	100 μF 16V
<input type="checkbox"/> C8	1	10 μF 10V
<input type="checkbox"/> C1-C3, C7	4	0.1 μF 16V

Miscellaneous:

9V 300 mA wall wart power supply; printed circuit board; waterproof enclosure (PacTec OD56 Kit; gray); header shorting connector; grommet for wires to exit enclosure; board, keypad and enclosure mounting hardware; silicon sealant; twin lead wire to wall wart and garage door opener to switch contacts

Websites for further information

Atmel	www.atmel.com
AVR Freaks Forum	www.avrfreaks.net
Bascom-AVR	www.mcselec.com
ButtLoad	www.fourwalledcubicle.com
ExpressPCB	www.expresspcb.com
Nuts & Volts	www.nutsvolts.com
PacTec Enclosures	www.pactecenclosures.com
PonyProg	www.lancos.com
Spark Fun Electronics	www.sparkfun.com
Jay Carter	www.docjc.us

Footnotes

The PonyProg Serial Device Programmer is available from Claudio Lanconelli. The program and further information is available at www.lancos.com

ButtLoad (Butterfly software Loader) is available from Dean Camera. The program and further information are available at www.fourwalledcubicle.com

Parts Suppliers

All Electronics	www.allelectronics.com
Digi-Key Corp.	www.digikey.com
SparkFun Electronics	www.sparkfun.com

programmed in assembly language, C, or Basic. This project was programmed using Bascom-AVR Basic. Roughly 2/3's of the available memory is used by the program, allowing room for future revisions or enhancements.

The current software includes several convenience features. By first entering a "Programming Code," the microcontroller beeps three times and the user then enters the access code of their choice. This code is saved in the microcontroller's EEPROM memory for future use. The access code can be changed at any time by simply re-programming it.

Many programs contain a "back door" or secret access code, and this project is no exception. A secret access code also exists which will always work and can not be changed by user re-programming described above.

If the user presses the second key on the keypad prior to entering their access code, the second door (Door #2) is activated instead of Door #1. A single keypad and access code can therefore be used to open or close either door.

Safety First

Once a valid access code is entered and a door is activated to either open or close, one can press any key on the keypad to instantly stop the door. This feature is

active for 30 seconds after entering a valid access code. If the incorrect door was activated or if one needed to stop the door quickly, it can be easily done without remembering and re-entering the entire access code.

Programming the Microcontroller

The operating program – written in Basic – must be compiled to generate a hex file which can be loaded into the ATtiny2313 microcontroller. Both versions of the program are available on the *Nuts & Volts* website (www.nutsvolts.com). If you are familiar with programming AVRs, you are all set. A programmer is used to load the hex file into the microcontroller and many different programmers exist. A low cost approach is the AVR STK Serial Port Dongle Programmer available from SparkFun Electronics (~\$12.95). This device is used with the free PonyProg software. An ATMEL Butterfly demonstration board (~\$21.00) can be easily converted into an AVR ISP programmer using the free ButtLoad program. The ATMEL STK500 (~\$85.00) development board/programmer provides extensive testbed and programming capabilities.

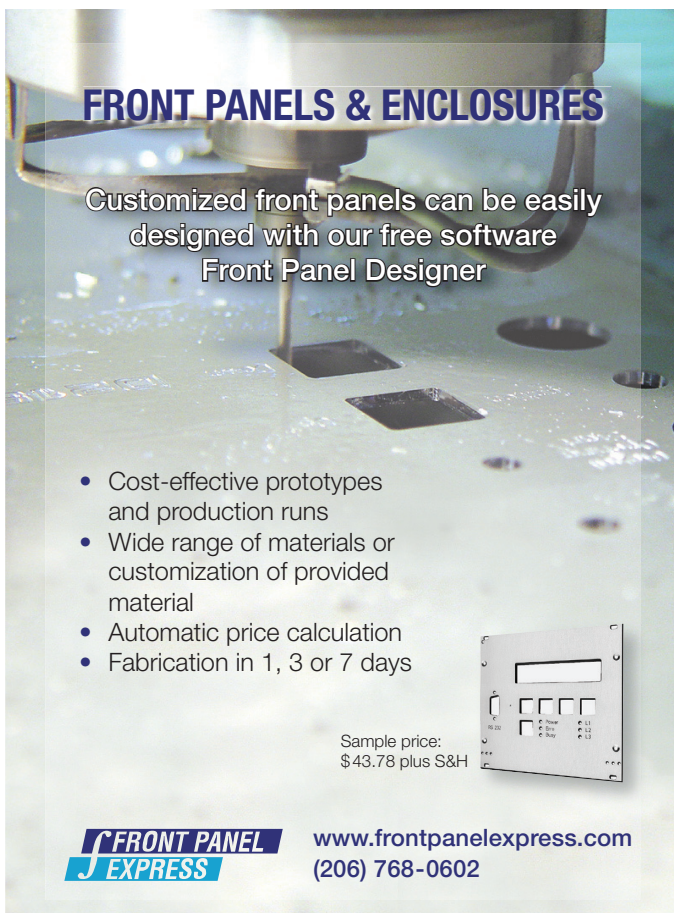
Additional options and further information on programming can be found on the ATMEL and AVR Freaks websites given in the parts list. Instructions on downloading the program (hex file) into the microcontroller chip are provided with each of the above programmers, and in tutorials on the ATMEL, AVR Freaks, and SparkFun Electronics websites.

A circuit board was developed for this project using ExpressPCB and their software. The board layout is available and is posted with the software. The board includes a 10 pin programming header for use with the above programmers. Additional pads are provided on the board for adding an external crystal, its capacitors, an LCD contrast potentiometer, and for connecting to the USART pins. These are unused in this project, but are useful for experimentation.

Let Me In!

This project demonstrates the incredible power and capability of small, inexpensive microcontrollers to perform everyday tasks and make our lives easier. With just a handful of components, a truly useful and convenient project can be assembled in a weekend. It demonstrates interfacing to a low cost matrix keypad and the software techniques required to read and debounce the keypad input. Additionally, the project demonstrates interfacing a microcontroller to LEDs, relays, and piezoelectric beepers.

Keypad access is both a great convenience and security measure. This project shows the ease of its implementation. But stay tuned for Garage Access Version 2, using ATMEL's biometric fingerprint Fingerchip sensor to make keypads a relic from the past! **NV**



FRONT PANELS & ENCLOSURES

Customized front panels can be easily designed with our free software
Front Panel Designer

- Cost-effective prototypes and production runs
- Wide range of materials or customization of provided material
- Automatic price calculation
- Fabrication in 1, 3 or 7 days

Sample price:
\$43.78 plus S&H



FRONT PANEL EXPRESS www.frontpanelexpress.com
(206) 768-0602