Introduction

Our little town of Manitou Springs, Colorado takes it Carnival celebration seriously. It seems like half of the town is in the parade dressed in colorful and comic costumes and the other half is on the streets partying and having a good time. Beaded necklaces of all varieties are traded between the participants and many people have quite the collections. For next year I knew I had to come up with something special that no one else would have so I started thinking about making an electronic necklace of some kind. I wanted something that was unique, playful, personalized and inexpensive so I could give it away if I wished and so I wouldn't feel bad if it got broken during the celebration.

Having a basic idea of what I wanted I searched the Internet and found http://tinkerlog.com/howto/64pixels/ where Alex Weber had connected a 64 LED array to a ATtiny2313 micro controller to make a circuit capable of displaying static patterns, text strings and simple animations. This was basically what I was looking for. Subsequent searches turned up another site, https://sites.google.com/site/tinymatrix/, where tigeruppp (who also referenced Alex's work) built a 5x7 LED matrix and a ATtiny4313 into a necklace which was exactly what I wanted to do. Both Alex and tigeruppp provided C code for their projects so I had a lot of resources to draw upon.

In the end, I used tigeruppp's hardware configuration and modified his code to build what I'll refer to as a smart necklace. In this article I will show you how to:

1. Compile the C code for the smart necklace for the ATtiny4313 micro controller
2. Program (flash) the micro controller using an Arduino Uno board as a programmer
3. Build the smart necklace

When we are finished you will have a smart necklace with seven animations and 13 static display patterns available at the touch of a button. In addition, each necklace can be customized to display a persons name or other pertinent text message.

Prerequisites for building this smart necklace include:

1. Having a PC or MAC for compiling the code
2. Having some knowledge of C program development. The ability to edit a C program source file and run a Makefile from a command line window, for example.
3. Having a programmer for flashing the code into the controller (more on this shortly)
4. Having basic soldering skills for assembling the necklace.

After having built a few of these smart necklaces for my Mardi Gras friends, I realized they would also make great gifts for our young nieces and nephews; something they could show off to their friends. If you make these now you will be way ahead when it comes to Christmas shopping for the young ones, so lets get started.

The Hardware

Figure One shows the parts list for each necklace and where I bought the parts. Each necklace can be made for around $5 and, if you buy the parts in quantity, even less. Once you have built a couple, you can assemble the necklaces in about half an hour. Image One shows the parts involved. Figure Two is the schematic of the necklaces' electronics.

As you can see from the schematic, the circuit is very simple which makes it easy to build. What you cannot see from the schematic is that the middle 12 pins of the micro controller map directly to the 12 pins on the LED matrix. When we build the necklace we slide the LED matrix over the middle 12 pins of the controller and solder them in place. More on this latter as well.

The Programming Hardware and Software

When I decided to build these necklaces I knew I would need a method of programming (also called flashing) the ATtiny4313 chips. I knew an Arduino Uno (or other Arduino boards) can be used to program the chip by connecting the following wires.

Arduino Uno Pin	ATtiny4313 Pin
13	19
12	18
11	17
10	1
+ 5V	20
GND	10

These connections can be made on a breadboard for the lowest cost approach. I, however, wanted a real Atmel device programmer that would be capable of flashing Arduino boot loaders into new ATmega328 chips for other projects. I decided to buy the inexpensive ISP Shield kit and associated components from evilmadscientist.com. Figure Three shows the parts required and Image Two shows the assembled programmer hardware. The evilmadscientist.com site has the instructions for building the ISP Shield and the ATtiny target board. In this configuration the ISP Shield is plugged onto the Arduino Uno and the ISP Shield plugs into the ATtiny target board via a six pin ribbon cable. ATTiny4313 chips are placed in the ZIF (Zero Insertion Force) socket on the target board and the lever is lowered to lock the chip in place during the programming process.

Whether you use the dedicated wiring or the ISP Shield approach you will need to connect the Arduino Uno to your PC or MAC via a USB cable. You will also need to have the Arduino IDE installed on your computer. I used version 1.0.4 of the IDE. See http://arduino.cc for information about getting and installing the Arduino IDE if you haven't done so already.

With the IDE installed and the USB cable connected, load the ArduinoISP sketch. Then select the Arduino Uno from the IDE Tools/Board menu and click the upload button. If you are using the ISP Shield the pulse LED will slowly flash indicating you are ready to program the ATtiny4313. You can then close the Arduino IDE since the programming sketch has been loaded into the Arduino Uno. Actual programming of the ATtiny4313 will be done after we install the software development tools we discuss next.

Software Tools

You must install AVR software development tools on your computer to compile the smart necklace C code and to flash the compiled code into the micro controller chip. The controller chips are programmed before being assembled into a necklace.

CrossPack (http://www.obdev.at/products/crosspack/index.html) is the development environment for Atmel's AVR micro controllers running on Apple's OS X and is similar to WinAVR (http://sourceforge.net/projects/winavr/) for Windows. These free development environments consists of a GNU compiler suite, a C library for the AVR, the avrdude uploader/programmer and several other useful tools. Install the development tools for your computing environment as described on the appropriate website. At this time also download the file “smartnecklace.zip” from the Nuts and Volts website and unzip it into a directory of your choosing. This zip file contains the files TinyMatrix.c (the C source code for the smart necklace) and a Makefile for building and programming the micro controller.

The Makefile may need to be tailored to your computing environment. The Makefile I use on my Mac is shown below for reference. The line in bold is what will need to be changed if you are running on a PC. The -P command line switch specifies the serial port that will be used with the avrdude programmer. On the Mac the port is specified as /dev/tty.usb*, on a PC it would be changed to COMx where x is the number of the serial port used by the programmer. The -b command line switch specifies the baud rate of the serial connection. Edit this file if necessary for your computing environment.

```
# Makefile for TinyMatrix
```

```
NAME      = TinyMatrix
OBJECTS    = $(NAME).o
DEVICE     = attiny4313
CLOCK      = 4000000
FUSES      = -U hfuse:w:0xDF:m -U lfuse:w:0xE2:m
PROGRAMMER = avrisp
AVRDUDE = avrdude -c $(PROGRAMMER) -p t4313 -P /dev/tty.usb* -b 19200
COMPILE = avr-gcc -Wall -Os -DF_CPU=$(CLOCK) -mmcu=$(DEVICE)
# symbolic targets:
all:    $(NAME).hex
.c.o:
        $(COMPILE) -c $< -o $@
flash:  all
        $(AVRDUDE) -U flash:w:$(NAME).hex:i
fuse:
        $(AVRDUDE) $(FUSES)
clean:
        rm -f $(NAME).hex $(NAME).elf $(OBJECTS)
# file targets:
$(NAME).elf: $(OBJECTS)
        $(COMPILE) -o $(NAME).elf $(OBJECTS)
$(NAME).hex: $(NAME).elf
        rm -f $(NAME).hex
        avr-objcopy -j .text -j .data -O ihex $(NAME).elf $(NAME).hex
```

With the Makefile properly configured, you will use the following commands from a command line window or terminal shell.

Command	Result
make	Compiles the C code in the file TinyMatrix.c. Use this command to find any errors in the source code. Errors will be displayed in the command/terminal window.
make fuse	Programs the ATtiny4313 fuses as required for operation of the code. This command only needs to be executed once per micro controller chip.
make flash	Compiles the C code and if there are no errors causes the compiled code to be flashed into the micro controller. The ATtiny4313 can be reprogrammed many times without issue. Perform this operation as many times as necessary to get the code correct.

Command	Result
make clean	Deletes all temporary object and hex files so you can continue with a clean slate.

Smart Necklace Code

The C code for the smart necklace is contained in the file TinyMatrix.c. This amazingly concise code was written by tigeruppp with a modification I did to allow the smart necklace to display text messages. You will have to study the code if you wish to understand its operation in detail but here are some of the highlights:

* The controller runs on its internal clock at 4 MHz
* The bitmap two dimensional array in the code holds the data for the currently displayed frame.
* An interrupt service routine drives the display process which is capable of updating the LED matrix approximately 78 times a second. Fast enough to be flicker free.
* The code polls the pattern switch between display updates. If a switch click is detected the mode variable is updated which causes the next display pattern to be loaded into the bitmap array.

The code can be customized to display someones name or a text message by changing the following line in the code.

```
// Set the following to the text you wish to display
#define TEXT "Hello"
```

So if you wanted to change the displayed text from “Hello” to “Hello from Nuts and Volts Magazine” you would make the following change:

```
// Set the following to the text you wish to display
#define TEXT "Hello from Nuts and Volts Magazine"
```

and rebuild the code with the “make flash” command. Now when the smart necklace is powered up it will display this message over and over until the pattern switch is pressed to select a different display pattern.

If you are making a necklace for your nephew Bob, for example, you might change the code to:

```
// Set the following to the text you wish to display
#define TEXT "My name is Bob"
```

After you have made changes to the C code and have gotten it programmed into a ATtiny4313 micro controller you might want to test it before building a smart necklace just to be sure your changes work as expected. I would suggest breadboarding the circuit as shown in Image Three because once you build the Attiny4313 into a necklace it would be difficult to reprogram.

Building a Smart Necklace

Now with all of the background information out of the way it is time to build a smart necklace. Please examine Images four, five and six during the following discussion.

First, locate pin one of the ATtiny4313. It is marked with a small dot and a small indentation at the top of the DIP package. Next locate pin one of the LED matrix which isn't obvious because there is no marking I could find. Pin one is on the top left of the matrix when the part number is to the right and you are looking at the top of the device. Gently spread the leads of the LED matrix outward slightly. With both pin ones on the left, slide the matrix onto the controller so that pin one of the matrix contacts pin three of the micro controller. In other words the LED matrix connects to the middle 12 pins of the micro controller. Push the LED matrix down as far as it will easily go and solder one of the pins from the matrix to the micro controller to hold it in place. When you are satisfied with the placement of the matrix solder the remaining 11 pins to the controller chip. Cut two pieces of light gauge wire for the necklace to around 16 1/2” in length. I happen to have some red and black wire which made keeping the polarity straight easy. Position the 100K resistor on top of the controller chip and bend its leads so as to connect it between pins one and 20. Solder the resistor to pin one. Connect the positive necklace wire, which in my case was red, to pin 20 as well and solder it. Position the pattern display pushbutton switch at the opposite end of the chip and splay its leads so as to make contact to pins 10 and 11 of the micro controller. Solder pin 11. Connect the negative wire (black in my case) to pin 10 and solder. I then slid a 3/4” piece of heat shrink tubing over the wires and pressed it up against the chip before shrinking it into place. You may want to trim the micro controller pins at this time because they are awfully sharp.

I then slid another 3/4” piece of shrink tubing on the other end of wires which I will shrink shortly. As a strain relief, I looped the wires through the battery holder before soldering them to the appropriate terminals. Note the polarity. I then used some hot glue to anchor the wires to the battery holder. I then pushed the shrink tubing up against the battery holder and shrank it. With that, when you pop in the battery you should have an operational smart necklace. Click the pattern select switch to run through the various display patterns and animations that are available.

Possible Enhancements

Since there is quite a bit of memory left in the controller many more static display patterns and/or animations could be added once you understand how the code works. The code could also be altered to display different text messages with every press of the button.

Instead of the necklace configuration, the circuitry could be housed in small round or square boxes and used as Christmas tree decorations or sewn into the fabric of a shirt, vest or coat. It could also be used as an intelligent name tag displaying a person's name, company and job title. If you think about it, many other uses will come to mind as well.

Author BIO

Craig has been interested in the production of lights/sounds/music with computers for a very long time. His most recent book is “Digital Audio with Java” published by Prentice-Hall. He lives in the mountains of Colorado and can be contacted at calhjh@gmail.com. When not messing around with electronics, computer projects, wood working or beer brewing, he does a solo musical act around Colorado Springs.

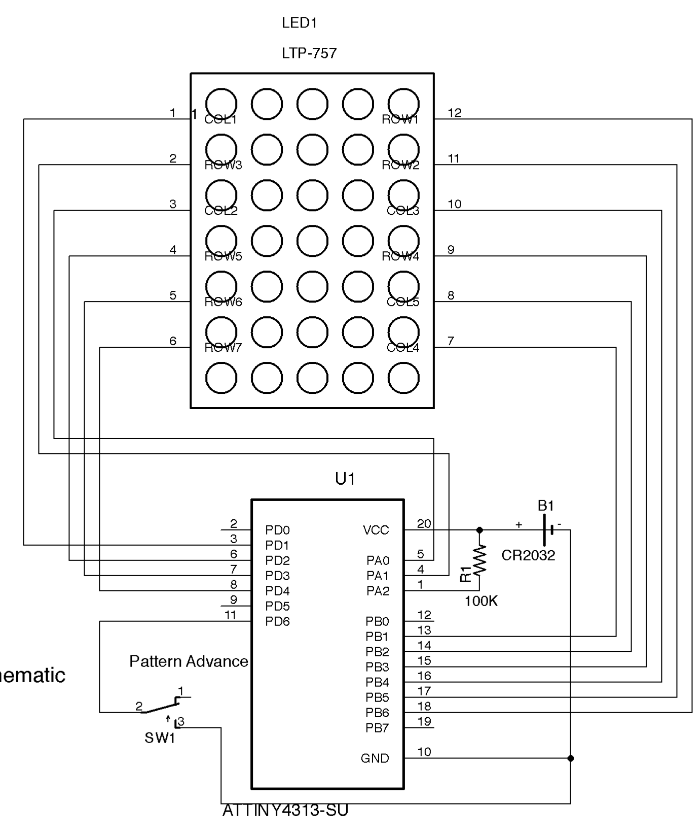
Figure One

Smart Necklace Parts List

Designation	Part Number	Source / Price
U1	ATtiny4313 micro controller	digikey.com - $1.51
LED1	LiteOn LTP-757 LED matrix	jameco.com - $0.89
R1	100K Ohm 1/4 watt resistor	evilmadscientist.com – 10 for $0.40
SW1	Tactile button switch	evilmadscientist.com – 5 for $2.00
B1	2032 coin battery	Many places - ~1.00
BH1	Coin battery holder	jameco.com - $0.79
N.A.	Hookup wire, heat shrink tubing	Your work shop

Figure Two

Smart Necklace Schematic



Smart Necklace Schematic

Figure Three

Programming Hardware Parts List

Part Number	Source / Price
ISP Shield	evilmadscientist.com - $12.95
Arduino Uno (if you don't have one)	evilmadscientist.com - $29.95
ATtiny target board	evilmadscientist.com – $3.00
20 pin ZIF socket	evilmadscientist.com – $1.40
6 pin DIL ribbon cable	evilmadscientist.com - $2.95
6 pin DIL header	evilmadscientist.com - $0.70
USB male A to male B cable for connecting your computer to the Arduino Uno.	evilmadscientist.com - $2.95

Image One

All of the required parts

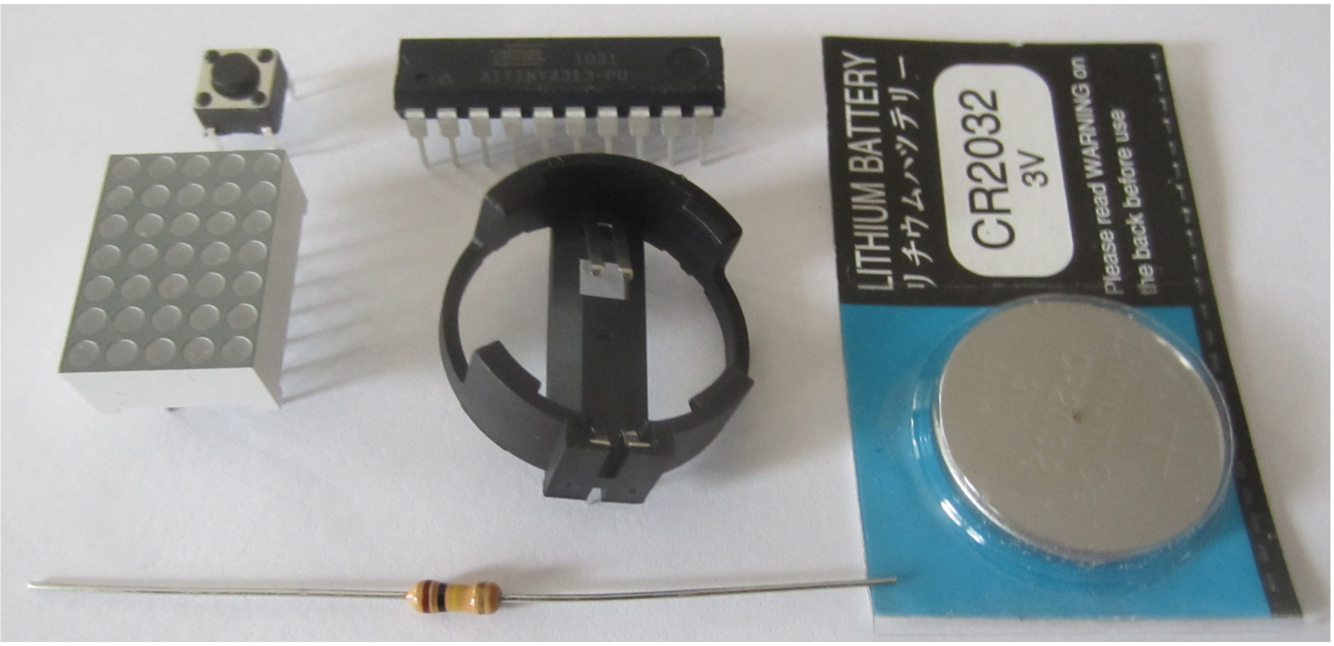


Image Two

The Programming Hardware

Arduino Uno, ISP Shield, ATtiny Target Board with connecting ribbon cable

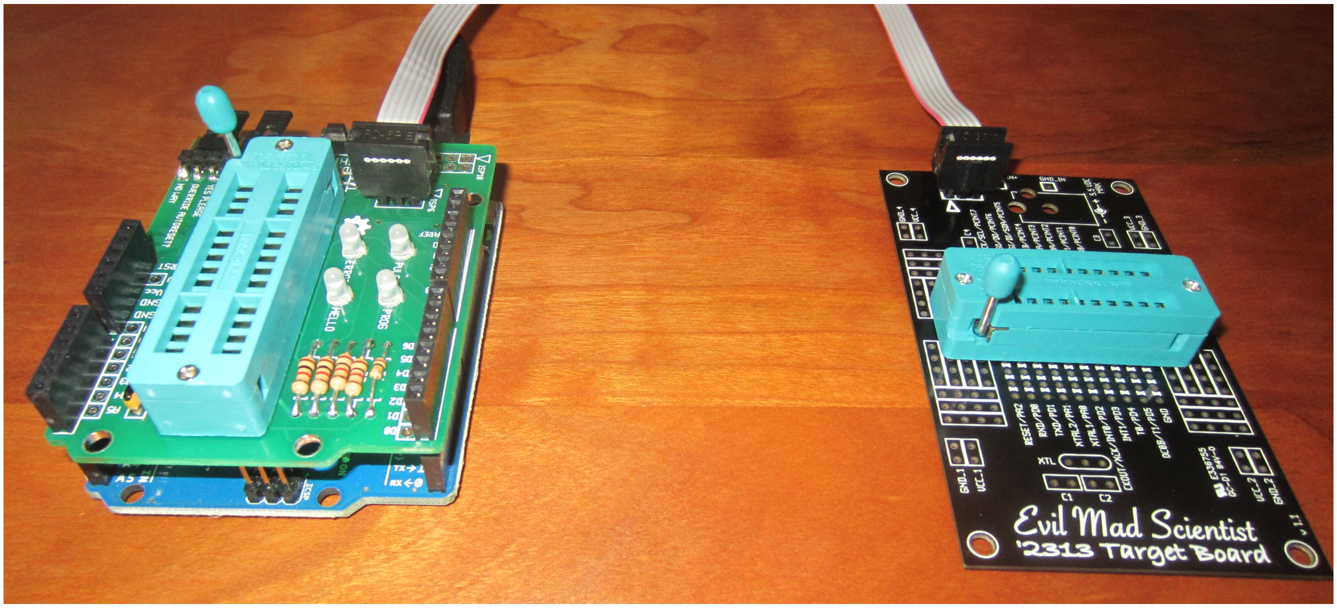


Image Three

Breadboard of the Smart Necklace Circuit

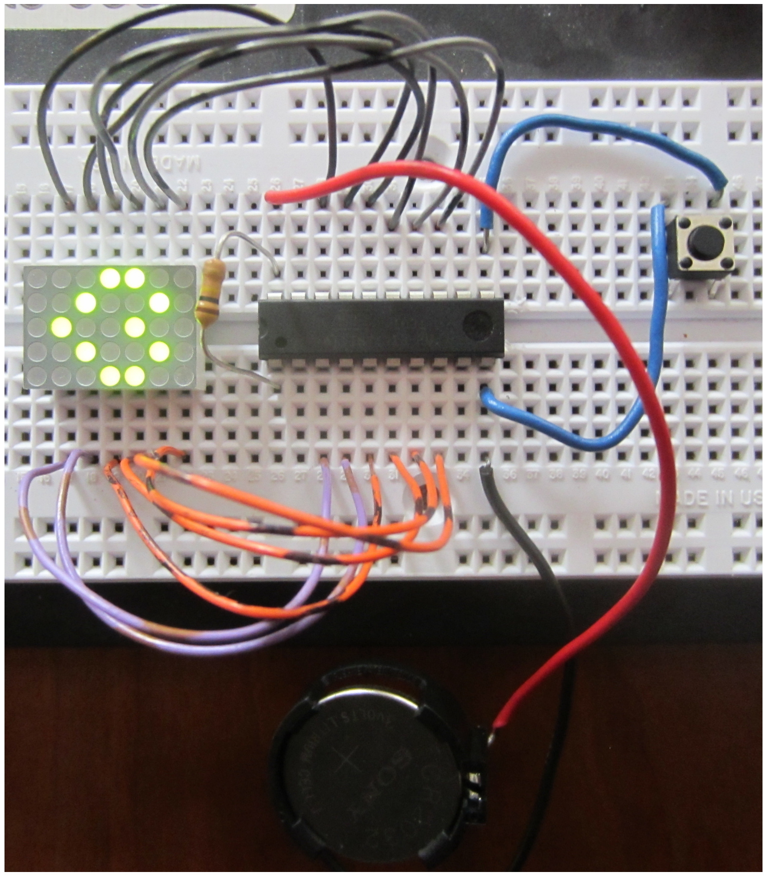


Image Four

Top View of Smart Necklace

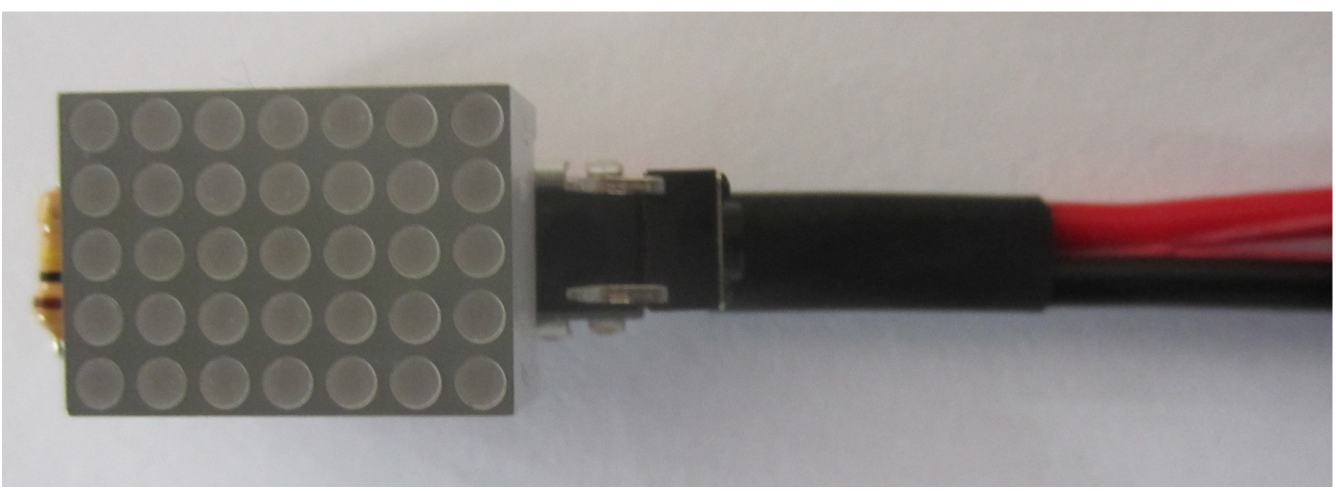


Image Five

Closeup of Smart Necklace

Image Six

Construction Detail of Smart Necklace