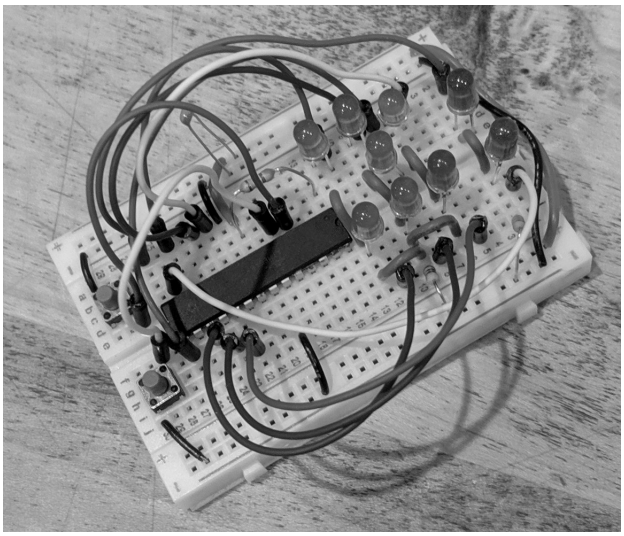
Electronic Dice

Based on the ATmega8a

From: Blinkenlights (www.youtube.com/blinkenlights)

by Schwager Electronics (www.schwager.com)



Thank you for your purchase of the ATmega8a-based electronic dice. We hope you find it enjoyable and interesting to put together, and may it bring you lots of (pseudo-random) luck!

The kit uses a breadboard as a base for construction. The breadboard has been around for decades and is an easy, versatile way to create prototype circuits. You can tear down the circuit and build it again, or build something new.

This electronic die is based on the popular ATmega series of 8-bit micro controller chips. It features a set of LEDs to display a random number from 1-6 in a die format, and additionally, a couple of LEDs that act as a “Heads/Tails” coin toss. There are pushbuttons that allow you to “roll” the die or “flip” the coin. This is a low-power, battery-powered project. Approximately 20 seconds after any action, the micro controller will put itself to sleep so as to preserve battery power. The current draw is so low in sleep mode that it’s not necessary to unplug or shut off the device.

Table of Contents

Kit Contents

The kit's contents include:

- 1 – 400 pin breadboard.
- 1 – ATmega8a "micro controller" computer chip. This comes preinstalled on the breadboard.
- 1 – 120 ohm resistor
- 1 – 1.5k ohm resistor (the k means 1000, so this is a 1500 ohm resistor)
- 1 – 4.7k ohm resistor (this is a 4700 ohm resistor)
- 1 – 0.1 uF disk capacitor
- 1 – 100 uF electrolytic capacitor
- 7 – Green LEDs
- 1 – Red LED
- 1 – Blue LED
- assorted 10cm (4") jumper wires
- 1 – long regular red wire.
- 1 – long regular black wire.
- 4 – short black wires.
- 1 – short red wire.

- 6 – short green wires.
- 1 – battery pack, for 3 – AA batteries.
- 2 – pushbutton switches

Source Code

Source code for this project is available at https://github.com/GreyGnome/dice/tree/master/dice_atmega8 . This project is distributed under the Apache 2.0 license. See the source code for details.

Photos

Photos of the kit in different stages of production are available at the website at http://www.schwager.com/Pages/Producs/dice_atmega8.html

The Breadboard

The instructions for the kit utilize the coordinate system on the 400-pin breadboard. At the top and bottom, the columns are denoted by the letters a-e and f-j. Along the left and right, the rows are denoted by the numbers 1-30. The first 5 rows are illustrated here:

	a	b	c	d	e		f	g	h	i	j
1											
2			**x**								
3									**X**		
4					**y**		**Y**				
5											
… etc (down to 30 rows) ...											

So, for example, if the instructions tell you to insert a wire in points 2c and 3h, you would insert one end where the **x** is in the table above, and the other end would go where the **X** is in the table above. Similarly, one end of a wire in point 4e would be inserted where the y is, and the other end in 4f would be inserted where the Y is in the table. And so on.

A Note About the Rows

You can think about each half of a row on the breadboard as a single point, or as a wire. That is, for any given row, points a-e are all connected to each other by a metal wire. Then there is a break in the middle, and to the right, points f-j are all connected to each other. This makes it easy to connect parts together.

A Note About the Power Supply Rails

To the left and right of the main breadboard are two power supply rails. The holes in the red column, labelled "+" at the top and bottom, are connected together by a long wire. Similarly, the holes in the blue column, labelled "-", are connected together by a wire. There are no breaks in these wires. The top hole is connected all the way down to the bottom . It is convenient to connect our power packs to these columns and to use them as power for the components in our circuit. This is because there are often many connections to the two sides of a power supply in any given circuit, including this one. So having these rails makes things cleaner.

Both the left and the right side each have a positive rail, shown by the red "+" sign and the long red bar, and both the left and right side have a negative rail, shown by the blue "-" sign and long blue bar.

You can see a picture of the board with a couple of the power supply rails highlighted at http://www.schwager.com/Pages/Products/dice_atmega8a.html#A Note About the Power Supply Rails

Instructions

- Use the Wiring Instructions table and the below instructions as your guides.
- Starting from Step 1, install one side of each component in the point given in the "From" column. Install the other side in the point given in the "To" column.
- Indepth details, along with photos, can be found online at http://www.schwager.com/Pages/Products/dice_atmega8a.html

Steps 1-13: Install Wires

Install the Long and Short wires. Put one end of the wire in the point indicated by the "From" column of the table. Put the other end in the point indicated by the "To" column of the table.

These wires are different from the Jumper wires as they have no special plug on each end. See the photo to see how your board should look after step 13 at http://www.schwager.com/Pages/Products/dice_atmega8a.html#Steps 1-13

Steps 14-19: Install Resistors, Jumpers, and 2 LEDs

In this series of steps, you'll be installing your first components: the 1.5k ohm and 120 ohm resistors. It doesn't matter which end of each resistor goes in which hole. A resistor works the same in either direction. But it is important to use the correct resistor for each location. The 1.5k ohm resistor is colored with brown, green, red, and gold bands. The 120 ohm resistor is colored with brown, red, brown and gold bands. See the photo on the website at http://www.schwager.com/Pages/Products/dice_atmega8a.html#Steps 14-19

The 10 cm jumper wires have special tips at each end. See the photo at http://www.schwager.com/Pages/Products/dice_atmega8a.html#Steps 14-19

The color of the jumper wires doesn't matter. Unfortunately, although we like to reserve red and black for power rails, we may need to use some of those colored jumper wires for signals to the LEDs.

Look at the photo at http://www.schwager.com/Pages/Products/dice_atmega8a.html#Steps 14-19, continued to see how your board should look after this step, with the 120 ohm and 1.5k ohm resistors circled.

Steps 20 - 22: First Battery Test

Install the batteries in the battery pack, making sure to put the flat end of each battery (without the little bump) against the spring. Connect the battery pack: The red wire into one of the holes in the + column, the black wire into one of the holes in the – column. Since all the holes in a column are connected by a metal wire inside, it doesn't matter which hole you use.

After you plug in the battery, you should see the LEDs blink a few times. This is a self-test. If they do not blink, remove the battery power. Check your wiring and try again.

Whenever the battery pack is not plugged in, remove one of the batteries. This will prevent an accidental short circuit by touching the ends of the battery wires together. It is not necessary to remove them all.

Steps 23 – 30 Jumpers and Resistor

Now, starting with line 23, install more jumpers. See the photo at http://www.schwager.com/Pages/Products/dice_atmega8a.html#Steps 23-30 to see what the board will look like after line 25 has been completed.

Complete the 4 more jumper wires and 1 resistor in lines 26-30 for this section.

Steps 31 – 37 Dice LEDs

Now it's time to install the majority of the dice LEDs. Look at the wire leads ("legs") of the LEDs. One is longer than the other. Insert the LED with the long and short leads going into the holes indicated on the Wiring Instruction Table. For the LED on step 31, from 9d to 7e, it would be helpful to carefully bend the leads as show in the picture so it will be more straight. See http://www.schwager.com/Pages/Products/dice_atmega8a.html#Steps 31-37

Steps 38 - 40: Second Battery Test

Again, install the batteries in the battery pack, making sure to put the flat end of each battery (without the little bump) against the spring. Connect the battery pack: The red wire into one of the holes in the + column, the black wire into one of the holes in the – column.

Again, you should see the LEDs blink a few times. This is a self-test. If they do not blink, remove the battery power. Check your wiring and try again. Especially check the LEDs; if you put them in

backwards, they won't light. You might try reversing one of the LEDs to see if maybe you put any in backwards. Unplug one of the battery wires and plug it in again. This will make the circuit go through its self-test again, and it will blink all of the LEDs.

Always be careful to properly plug the black and red wires into the proper column on the power rails. The red goes into the red rail, the black goes into the blue rail.

Look at the photo at http://www.schwager.com/Pages/Products/dice_atmega8a.html#Steps 38-40 to see what your circuit will look like after plugging in all the LEDs.

Steps 41-47: Capacitors, Switches, and Final Test!

Install the remaining parts, from step 41 to 45. The electrolytic capacitor has a positive lead and a negative lead. The negative lead is labeled. See http://www.schwager.com/Pages/Products/dice_atmega8a.html#Steps 41-47

It's important to insert it in the proper direction. Make sure the "-" side goes into the blue column on the power supply rails.

Insert the switches.

Congratulations, the circuit is all wired up! See the photo of the completed kit at http://www.schwager.com/Pages/Products/dice_atmega8a.html#Steps 41-47

Using the Dice

Plug in the battery pack again. Insert all the batteries. The LEDs should blink again, then all will be off.

Press one of the switches. You should see the dice blink in a pattern, then- depending on which switch you pressed- settle down to look like the face of a die, with a number 1-6 lit on the front, or one of the two LEDs at the top will light, to indicate either a "heads" or a "tails".

You can roll or flip as many times as you want! Play games with your friends! Impress your parents! Go online and read all about microcontrollers, and study the source code! Become a computer scientist! The possibilities are endless.

Support

Having trouble getting it to work? Send email to dicesupport@schwager.com. Tell me what's going wrong, and I'll try to help you figure it out. A picture would probably be helpful, if you can send one.

If you have any missing parts please send a list along with your address and I'll send you replacements.

Theory of Operation

Your ATmega8a dice kit uses a “microcontroller” as the brains of its operation. A microcontroller is a small, inexpensive computer capable of being programmed to do many things. It has built-in memory and a number of helper circuits so that it can read switches or other indicators, it can output signals to LEDs or other digital devices, and it can control motors or create sounds or perform any of a number of other tasks limited only by your imagination and the size of the built-in memory.

For this kit, the program is located at https://github.com/GreyGnome/dice/tree/master/dice_atmega8 . You are free to download or view the source code, and use it for any purpose.

After pressing the “roll” button, the dice works by using a pseudo-random number generator to choose a number between 1 and 6. A pseudo-random number generator is a mathematical operation or series of operations that produce numbers time and time again that (hopefully) look completely random with respect to each other. This project uses a pseudo-random number generator that’s delivered with the C language which was used to write the program. Although modern pseudo-random number generators are generally pretty good, all pseudo-random generators come with issues: You must “seed” them with a value that is itself hopefully random, and which is used as a basis for starting to generate pseudo-random numbers again and again. Because pseudo-random number generators are mathematical operations, they will create the same sequence of numbers every time a certain seed is used. This is a significant drawback to pseudo-random number generation, and it’s an issue that engineers go to great pains to try and compensate for. Using the same seed every time the device starts means that you’ll see the same sequence of numbers- hardly random at all!

So generating a random seed is critical. If it were easy, then there would be no need for pseudo-random number generators. This kit generates a random seed as follows: Inside the microcontroller is a counter that goes from 0 to 255 very quickly- it counts up 4000 times per second. Picking an exact number would be almost impossible. So as soon as you press a “roll” or “flip” switch, the counter is read. This number is used as a seed. So now we’ll have one of 256 sequences of numbers- pretty good, but remember, it’s only 256. Can we do better?

Yes. The very next time you press a button, the next counter number is read. This number is then combined with the previous number in a rather complicated way *, to get a seed that’s between 0 and 65535.

This repeats two more times, the first time the seed is one of 16 million values, and finally the second time the pseudo-random number generator is seeded with one of 4 billion values! 4 billion (actually 4,294,967,296) is the largest number possible to use for a seed. But that’s a lot of numbers, and would impossible to guess which was used. So the sequence of numbers created by the pseudo-random number generator should be good and non-guessable.

* How is it done? Technically, it’s shifted left by 8 (which is equivalent to multiplying by 256), then added to the current reading to get a number between 0 and 65535.