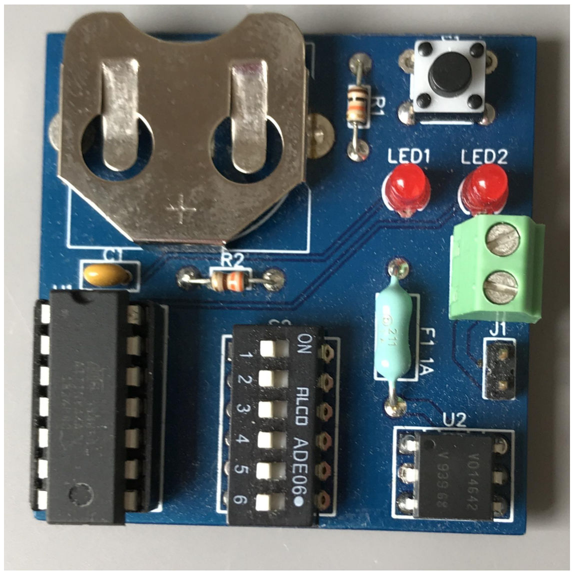
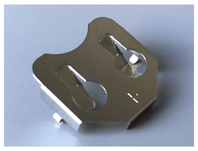
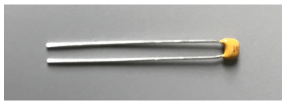
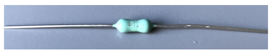
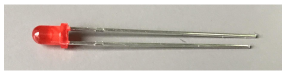
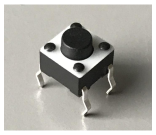
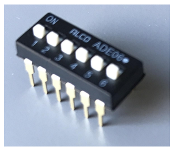
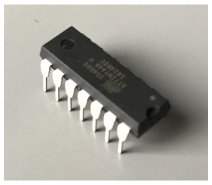
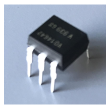
MG006 Programmable Time Switch

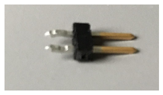
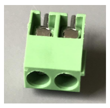
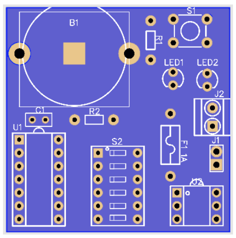
What is it?

MG006 is designed to switch a low voltage AC or DC line on and off at user-configurable time intervals. It is battery powered, and time intervals are set by means of a DIP switch. It is based on the ATTINY44A microcontroller, which can be removed and reprogrammed (using Arduino IDE) to accommodate different time periods, or even for a different use altogether.



What’s in the kit?

Name	Quantity	Description	Picture	Present?
B1	1	CR2032 battery holder		❟
C1	1	Capacitor, ceramic, 100 nF		❟
F1	1	Fuse, PCB Leaded, 1 A, 125 V		
LED1, 2	2	LED, Red, Through Hole, T-1 (3mm), 20 mA		❟
R1	1	Resistor, 10 kohm, 125 mW		❟
R2	1	Resistor, 330 ohm, 125 mW		❟
S1	1	Tactile switch		❟
S2	1	DIP switch		❟
U1	1	ATTINY44A		❟
U2	1	VO14642AT		❟

U1 socket	1	14-pin DIP socket		,
J1	1	Header, male, 1 x 2 pin, straight		,
J2	1	Terminal, screw		
PCB	1	MG006 PCB	 B1 S1 LED1 LED2 J2 C1 R2 U1 S2 J1	,

<u>What's not supplied?</u>
A CR-2032 battery.

<u>How do I build it?</u>
This kit assumes basic through hole soldering skills. If you're not too confident, I recommend searching for an online tutorial, there are some good ones on YouTube.

Recommended tools include:

- Soldering iron (ideally temperature controlled)
- Multicore solder
- Small snips to cut off leads
- Small pliers
- Desoldering pump and/or braid
- Anti-static wrist strap (or steer clear of materials that cause static and touch a grounded object every now and then).

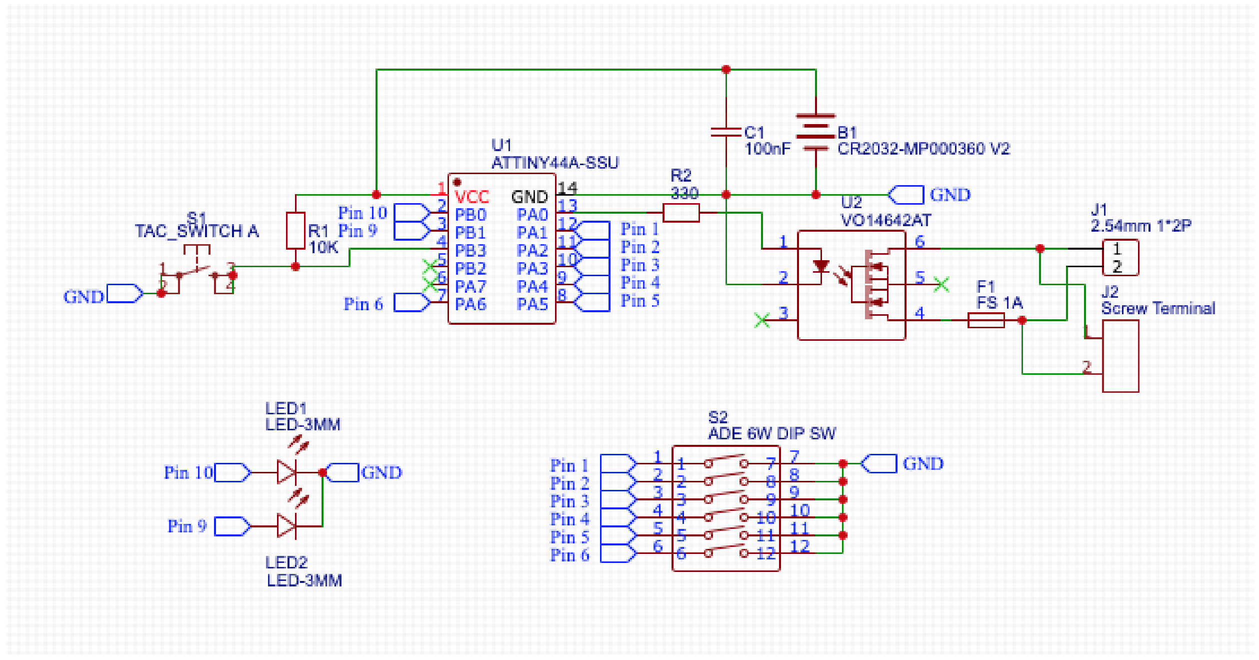
The normal rule of thumb is to solder the lowest height components first, working up (orientation doesn't matter unless stated):

- Before you solder anything, the small square pad that acts as the negative battery contact needs to be built up a little. Heat it up with a soldering iron, and apply a layer of solder sufficient to rise above the surrounding blue solder mask
- R1, R2
- F1
- C1
- U2. Be careful to solder the right way round. Solder two opposite corners, check the IC is flat on the board before continuing
- Socket for U1 (do not fit IC yet). Again, solder two opposite corners, check the socket is flat on the board before continuing. Make sure the notch at the end of the socket matches with the PCB graphics, to reduce the risk of installing the IC the wrong way round
- B1
- S1, S2. S2 needs to be the right way round, per the PCB graphic
- LED1, LED2. Be careful to solder in the right way round, per the PCB graphic
- J1
- J2

If you have flux cleaner, clean all joints. Now inspect them carefully for issues (a magnifying glass of some sort can be very helpful, the camera on some phones works quite well).

The final step prior to fitting the battery and testing is to fit the U1 into its socket. Pay attention to orientation.

How does it work?



U1 does the hard work, switching up to 1A through U2 and F1. S1 resets U1, with R1 holding the reset line high otherwise. S2 provides inputs to "Pins 1-6" of U1, which U1 reads to set on and off times. U1 controls LED1 and LED2 via "Pins 9 and 10" during startup to indicate the timings selected.

How do I use it?

The circuit to be controlled needs to be connected across J1 or J2 (polarity doesn’t matter).

S2 is read after power on or a reset and needs to be configured as follows to set on times:

Switch 1 (“ON”=0)	Switch 2 (“ON”=0)	Switch 3 (“ON”=0)	On time (mins)	LED 1 Flashes
0	0	0	1	1
0	0	1	2	2
0	1	0	4	3
0	1	1	8	4
1	0	0	16	5
1	0	1	32	6
1	1	0	64	7
1	1	1	128	8

Switches 4-6 and LED 2 work in the same way for off times.

When the battery is inserted, or reset pushed, LED 1 and then LED 2 flash to confirm the times set. Operation then starts with an “off” time, followed by “on”, then “off”, then “on” etc. until either the battery is removed (or is exhausted) or reset is pushed.

Acknowledgements/Legal

MG006 has been designed for hobbyist use only and is not to be used for safety or business critical applications.

MG006 is NOT designed to be used to control or be connected to any kind of dangerous voltage.

To program the ATTINY44A I used the ARDUINO IDE, and ATTINY core library located at:

https://github.com/damellis/attiny

The ATTINY can be reprogrammed to change operation by removing and connecting to an ARDUINO In-System-Programmer (ISP) and using the ARDUINO IDE.

```
[code]
/*
  MG006 Rev- Timer ATTINY 44A
  Timings 1,2,4,8,16,32,64,128 minutes
*/

int time = 500;
int ontime = 1;
int offtime = 1;
unsigned long onms = 60000;
unsigned long offms = 60000;

void setup() {
  // initialize outputs.
  pinMode(10, OUTPUT);
  pinMode(9, OUTPUT);
  pinMode(0, OUTPUT);
  // initialize inputs.
  pinMode(6, INPUT_PULLUP);
  pinMode(5, INPUT_PULLUP);
  pinMode(4, INPUT_PULLUP);
  pinMode(3, INPUT_PULLUP);
  pinMode(2, INPUT_PULLUP);
  pinMode(1, INPUT_PULLUP);
  // read in on switch values.
  ontime = ontime + digitalRead(3);
  ontime = ontime + digitalRead(2) * 2;
  ontime = ontime + digitalRead(1) * 4;
  offtime = offtime + digitalRead(6);
  offtime = offtime + digitalRead(5) * 2;
  offtime = offtime + digitalRead(4) * 4;
  // confirm switch values via LED flashes.
  for (int a = 1; a<= ontime; a++) {
    digitalWrite(10, HIGH);
    delay(time);
    digitalWrite(10, LOW);
    delay(time);
  }
   for (int a = 1; a<= offtime; a++) {
    digitalWrite(9, HIGH);
    delay(time);
    digitalWrite(9, LOW);
    delay(time);
  }
  // convert switch values to ms
  if (ontime > 1) {
    onms = (pow(2, (ontime-1))*60000);
  }
  if (offtime > 1) {
    offms = (pow(2, (offtime-1))*60000);
  }
}

void loop() {
  digitalWrite(0, LOW);
  delay(offms);
  digitalWrite(0, HIGH);
  delay(onms);
}
[/code]
```