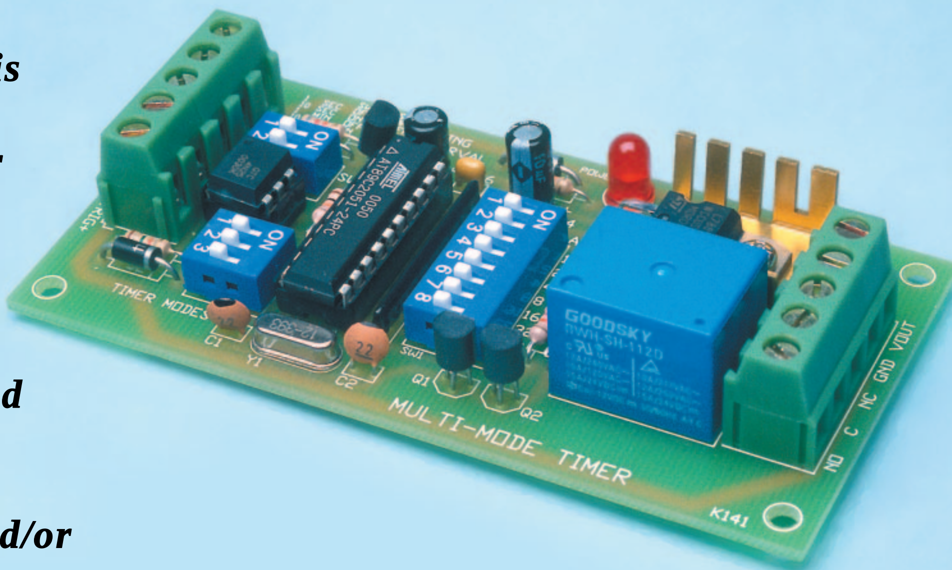
***Based on an Atmel microcontroller, this incredibly versatile timer is suitable for a wide range of applications. It's built on a compact PC board and both the trigger input and the output are fully isolated so that you can trigger from and/or switch high voltages.***




# Multi-Mode Timer

**By FRANK CRIVELLI & PETER CROWCROFT**

EVERYONE WHO BECOMES involved with electronics builds a timer at one stage or another. There are many, many thousands of designs using a variety of circuits, some of which have been around for decades. Witness the 555 timer IC, for example. This is longest surviving IC of all, being introduced over 20 years ago.

In the past, most timers were quite specialised in that they only performed one function – eg, an egg timer, a delayed timer, a timeout timer, a flasher, or a photographic timer, etc. Those days are now well and truly over – microcontroller ICs now allow us to easily design multi-purpose timers that can perform a variety of tasks, all at very low cost.

And that's exactly what you get with this new "Multi-Mode Timer". It supports no less than seven different timing modes using just one IC and a handful of other parts. In fact, it uses a lot less parts than many traditional "single-function" timers.

The various timing modes and delay ranges are selected using on-board DIP switches. You simply select the time delay you want and that's it – no further adjustments are required.

An optocoupler is used for the trigger input and this allows for complete electrical isolation between the trigger source and the remainder of the timer circuitry. This is important when high voltages are to be used for triggering the timer. An on-board relay provides electrical isolation of the output as well.

A number of triggering options are available, ranging from simple manual pushbutton triggering to electrically isolated voltage triggering. We'll take a closer look at the various triggering options that can be used later in this article.

As shown in the photos, all the parts are mounted on a single PC board, so it's really easy to build. Power supply requirements are quite modest and almost any 9-12V DC power source can be used. A 12VDC plugpack supply rated at 300mA will do the job quite nicely.

## Timer modes

OK, let's take a look at the various timing modes that are available from this circuit. There are currently seven timer modes defined – mode 8 is unused at present. If there's another timer variation you would like (or even a completely different set of timing modes), then let us know. After all, it's only a software change!

The various modes are as follows:

***Mode 1 – Instant On, Delayed Off, Level Triggered:*** a trigger signal operates the relay and starts the timing cycle. The relay then remains on for the selected delay time and then releases. A loss of the trigger signal also immediately ends the timing cycle and turns the relay off. The timer will then be ready for another trigger signal.

***Mode 2 – Instant On, Delayed Off, Edge Triggered:*** this is the same as Mode 1 except that loss of the trigger signal does not affect the timing cycle.

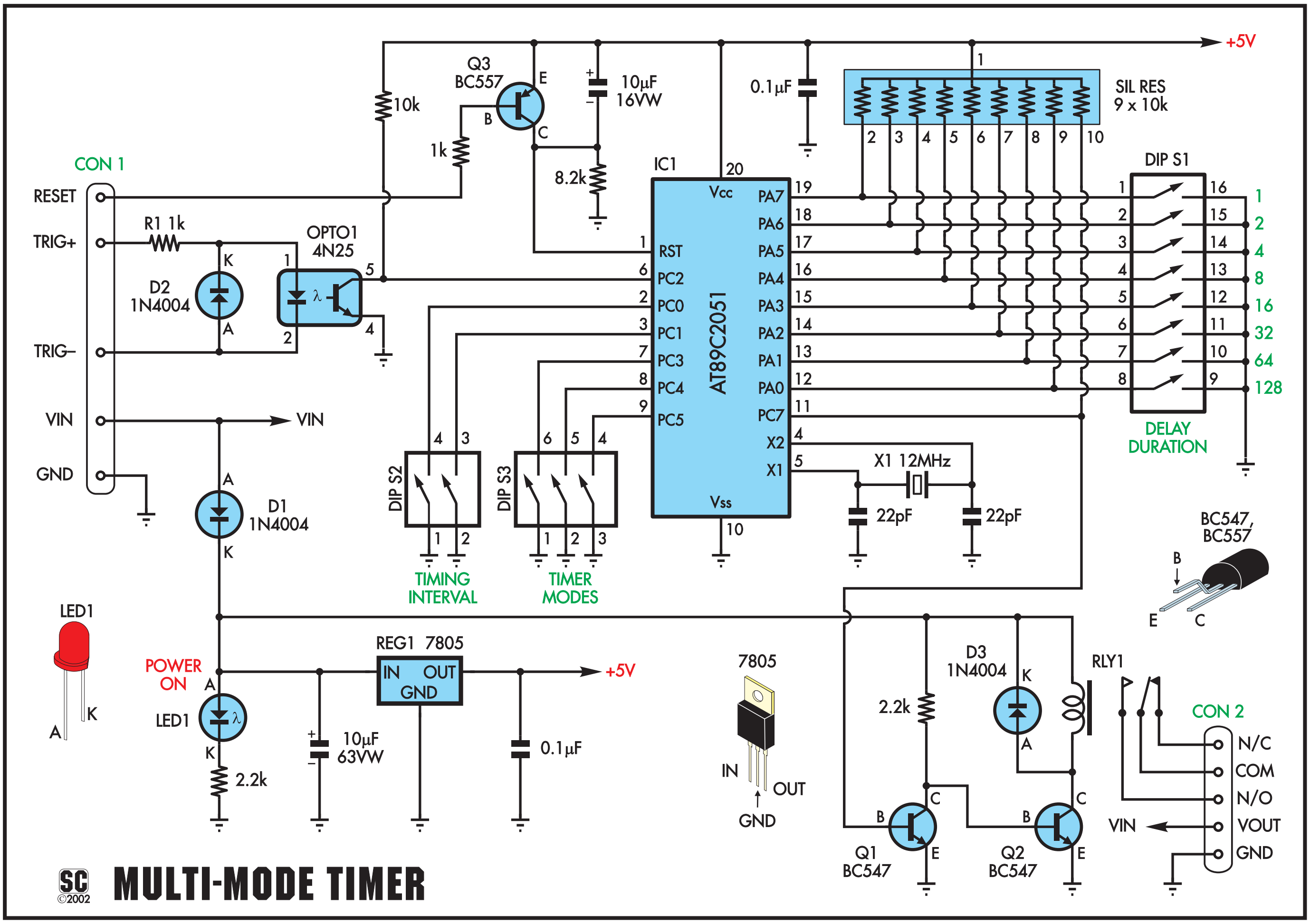



***Fig.1: the circuit for the Multi-Mode Timer is based on IC1, an Atmel 89C2051 microcontroller. This is preprogrammed with software to provide all the timing modes, which are set using DIP switch DIP3 (see Table 1). Triggering is via optocoupler OPTO1, while relay RLY1 isolates the timer's output.***

However, applying another trigger signal before the end of the timing cycle will restart the timer from zero. The effect is a "re-triggerable" timer.

***Mode 3 – Delayed On:*** a trigger signal starts the timing cycle. At the end of the delay time the relay operates and remains on until the trigger signal is removed or the timer is reset. Loss of the trigger signal during the delay time aborts the timing cycle.

***Mode 4 – Instant On and Hold, Delayed Off:*** a trigger signal turns on the relay but does not start the timing cycle. The relay then remains on while ever the trigger signal is present. Loss of the trigger signal then starts the timing cycle and the relay turns off at end of delay time.

***Mode 5 – Toggling:*** a trigger signal turns on the relay for the selected delay time. The relay then switches off for the same period. This cycle continues until loss of trigger signal or until a reset signal is applied.

***Mode 6 – Instant On, Delayed Off, With Pause:*** similar to Mode 1, a trigger signal operates the relay and starts the timing cycle. However, loss of trigger signal causes the timing cycle to pause and the relay remains on. Reapplying the trigger signal then restarts the delay time from the point where it was interrupted. At the end of the delay time, the relay turns off.

***Mode 7 – Delayed On with Pause:*** a trigger signal starts the timing cycle. At the end of the delay time the relay operates for 2 seconds and the timing cycle starts again. Loss of trigger signal causes the timing cycle to pause. Reapplying the trigger signal restarts the timing cycle from where it was stopped. Reset is the only way to exit this mode.

***Mode 8 – Not used.***

Important: note that for each of the

## SPECIFICATIONS

Operating Voltage .......... 12VDC (see text)
Trigger Voltage .......... 6-81V DC (see text)
Trigger Current .......... 5mA minimum; 80mA maximum (see text)
Trigger Pulse Width .......... 20ms minimum
Relay Contact Rating* .......... 10A @ 240V AC/DC max.
Timing Modes 8 .......... (see text)
Timing Ranges .......... 1-255s, 10-2550s, 1-255 minutes, 10-2550 minutes

**NB: although the relay contacts are rated at 240VAC, the relay should be limited to switching voltages up to about 40-50V DC or AC. DO NOT use the on-board relay to switch 240VAC (mains) voltages (see text).**

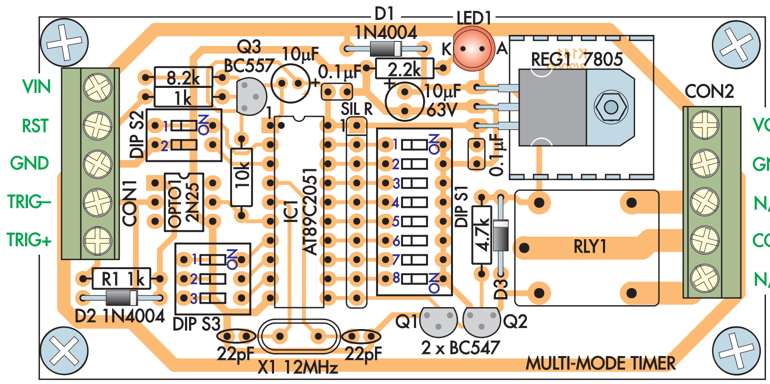



***Fig.2: install the parts on the PC board in the order listed in the text but don't install IC1 into its socket until the test procedure has been completed. A small mini-U heatsink is required for REG1.***

timer modes, a reset signal will stop the timing cycle immediately and reset the timer, ready for another trigger signal. The timer is reset by connecting the RST input to the GND input -– see Fig.1.

## *Circuit details*

Refer now to Fig.1 for the complete circuit details. At the heart of the circuit is IC1, an Atmel 89C2051 microcontroller. This is preprogrammed with software to provide all the timing functions.

A 12MHz crystal between pins 4 & 5 provides accurate timing and an easily divisible clock source for the internal hardware timers. Crystals are generally accurate to ±100ppm (parts per million) so, in this case, the actual crystal frequency could vary by as much as 1200Hz either side of 12MHz – an error of .01% maximum.

Over a period of 42.5 hours (2550 minutes, the maximum delay time this unit can be programmed for), this amounts to a maximum error of just ±15.3s.

The trigger signal is applied to the input of OPTO1, a 4N25 optocoupler. As previously mentioned, using an optocoupler allows the trigger signal to be electrically isolated from the timer circuit. This is especially useful if triggering the unit from high voltages. Diode D2 protects the optocoupler's input from damage due to reverse voltages, while the 1kΩ resistor provides current limiting.

Normally, the optocoupler output is high (ie, at 5V) and goes low (to 0V) when triggered. In this case, the load resistor is 10kΩ, which means that we need a current of 0.5mA through it for the output of the optocoupler to go to 0V.

From the 4N25's data sheet, the input current required is 10 times the output current. This means that we need a minimum input current of 5mA to trigger the timer.

The voltage across the optocoupler's internal LED, Vf, is typically 1V and remains fairly constant regardless of input current. Therefore, the minimum input voltage necessary to trigger the timer is given by:

$$Vin = (Iin \times R1) + Vf$$
$$= (5mA \times 1k\Omega) + 1V = 6V$$

If lower trigger voltages are required, then it's necessary to reduce the value of R1.

The maximum optocoupler input current is 80mA, which means that the maximum trigger voltage is (80mA x 1kΩ) + 1V = 81V. Of course you should allow for a safety margin of say 5-10mA. If higher trigger voltages are to be used, you will need to either increase R1 or add an external resistor.

The output from the optocoupler is used to trigger the microcontroller, IC1. This works in conjunction with its internal software program and DIP switches DIP1-DIP3 which are connected to ports A & C of IC1. The internal software reads the DIP switch settings and sets the timing mode and duration accordingly.

IC1's output appears at pin11 and drives transistors Q1 and Q2, which in turn operate the relay. So why are two transistors used here instead of just one? It's all to do with what happens on reset.

On reset, the microcontroller's I/O ports are configured as inputs (via internal hardware) and "float" high. If only one transistor was used, the relay would be activated during reset. Of course, the relay would be released after reset once the onboard software took over but that's not what we want.

By using two transistors, we can use a low output to operate the relay and a high to release it. And that means that the relay doesn't turn on during reset!

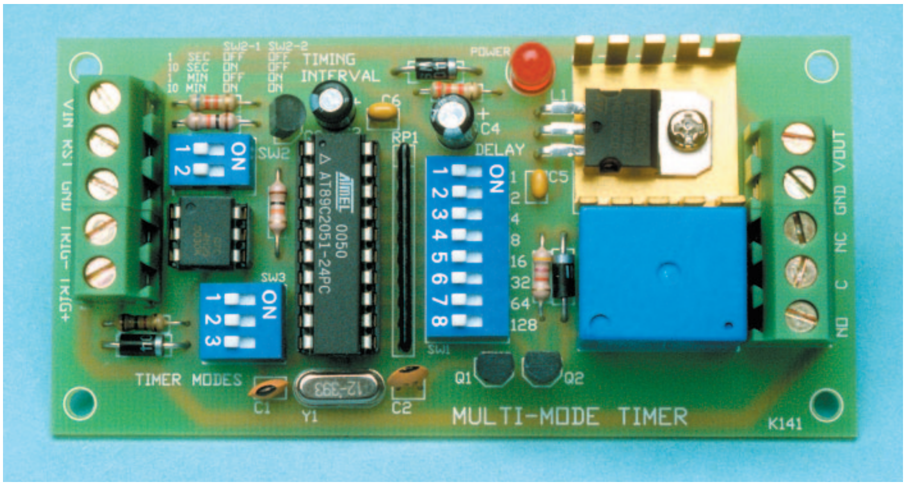
## *Power supply*

The timer requires a nominal 12VDC power supply; eg, a plugpack supply or a 12V battery. The incoming voltage is fed to REG1, a 7805 3-terminal regulator, to derive a regulated +5V rail which is then used to power IC1 & IC2. Diode D1 protects against reverse polarity connection of the power supply, while LED1 provides power-on indication.

Note, however, that the relay requires a 12V supply and so it is connected directly to the VIN supply input, rather than to the 5V rail (as is transistor Q1). This also minimises any switching noise on the +5V supply rail to IC1 when the relay turns on and off. Diode D3 is there to prevent

## *Table 1: Resistor Colour Codes*

| ❑ | No. | Value | 4-Band Code (5%) | 5-Band Code (1%) |
|---|---|---|---|---|
| ❑ | 1 | 10kΩ | brown black orange gold | brown black black red brown |
| ❑ | 1 | 8.2kΩ | grey red red gold | grey red black brown brown |
| ❑ | 1 | 4.7kΩ | yellow violet red gold | yellow violet black brown brown |
| ❑ | 1 | 2.2kΩ | red red red gold | red red black brown brown |
| ❑ | 2 | 1kΩ | brown black red gold | brown black black brown brown |




***This slightly larger-than-life view shows just how compact this versatile timer really is. A 9-12VDC plugpack supply rated at 300mA can be used to power the unit.***

back EMF from damaging Q2 when the relay releases.

Power on reset is provided by R2 and C3 (the 89C2051 microcontroller has an active high reset signal). In addition, transistor Q3 allows the user to use a low-going signal to reset the timer; eg, by connecting the RESET terminal on connector X1 to the GND terminal via a simple pushbutton switch.

### *Putting it together*

It's a cinch to put together – all you have to do is solder all the parts to the PC board as shown in Fig.2.

Install the resistors and diodes first, then install LED1, transistors Q1-Q3 and the electrolytic capacitors. Make sure that all the polarised parts are oriented correctly and double-check that Q3 is the BC557.

Take particular care when installing the SIL resistor pack (RP1). Pin 1 is identified by a dot at one end of its body and this goes towards the adjacent 0.1µF capacitor.

The DIP switches and relay RLY1 can go in next, followed by the 3-terminal regulator (REG1). The latter is mounted flat against the PC board together with a small U-shaped heatsink. That means that you have to bend REG1's leads down at right angles before fitting it to the board.

The best way to do that is to loosely attach the regulator to the board using a 3mm machine screw and then grip its three leads with needle-nose pliers. The screw can then be removed, the regulator lifted clear and its leads bent down through 90°. That done, REG1 and its heatsink can be fastened

## Parts List

1 PC board, code K141
1 12MHz crystal (X1)
1 12V relay, RWH-SH-112D (RLY1)
2 3-way PC-mount screw terminal blocks (5mm pitch)
2 2-way PC-mount screw terminal blocks (5mm pitch)
1 8-way DIP switch (DIP1)
1 2-way DIP switch (DIP2)
1 3-way DIP switch DIP3)
1 6-pin IC socket
1 20-pin IC socket
1 3mm x 8mm-long machine screw
1 3mm nut

**Semiconductors**
1 4N25 optocoupler (IC1)
1 AT89C2051 programmed Atmel microcontroller (IC2)
3 1N4004 diodes (D1,D2,D3)
2 BC547 NPN transistors (Q1,Q2)
1 BC557 PNP transistor (Q3)
1 7805 3-terminal regulator (REG1)
1 5mm red LED (LED1)

**Capacitors**
1 10µF 63VW PC electrolytic
1 10µF 16VW electrolytic
2 0.1µF MKT
2 22pF ceramic

**Resistors** (0.25W, 5%)
1 10kΩ 1 2.2kΩ
1 8.2kΩ 1 1kΩ
1 4.7kΩ
1 9 x 10kΩ 10-pin SIL resistor network (RP1)

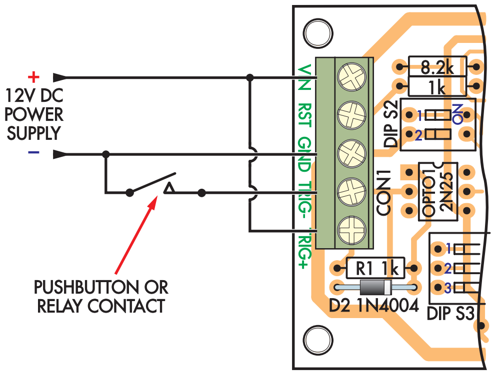



*Fig.3: using a pushbutton or relay contacts to trigger the timer.*

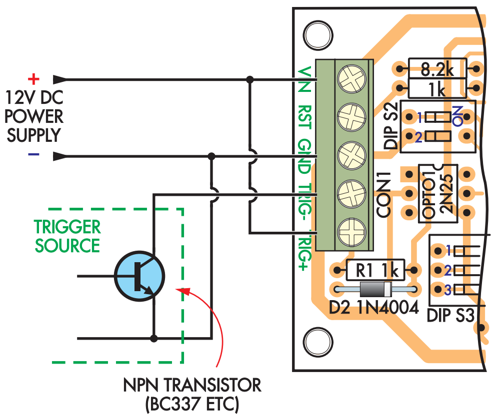



*Fig.4: triggering the timer using the open collector output of an NPN transistor.*

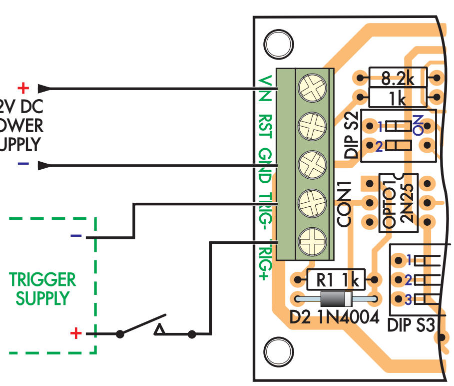



*Fig.5: use this circuit for fully-isolated triggering. Note that the trigger source must not connect to the timer's power supply if you want complete isolation.*

to the PC board using a machine screw and nut and the leads soldered.

Make sure that the heatsink is correctly aligned before tightening the screw, so that is doesn't foul the relay.

Now for the two 5-way screw terminal blocks. These are made by fitting together a 2-way block and 3-way block – just slide the raised edge on the side of one block into the matching groove of the other block. Each 5-way block is then installed on the PC board with the wire entry points facing outwards.

Don't install the ICs yet – that step comes later, after some initial tests. Just install their sockets for the time being, making sure that the notched end of each socket is positioned as shown on Fig.2.

## Testing

Apply power to the board – the RED power LED should be come on and the relay should remain off. Now use a multimeter to check the voltage between pins 20 & 10 of IC1's socket – you should get a reading of 5V. If this checks out, connect a short length of wire between pins 10 & 11. The relay should immediately operate.

If all is well, remove power and install the ICs in their sockets. Make sure that both ICs are correctly oriented and that none of their pins are "bent under" as you insert them.

## Setting the timer mode

The timer mode is set using DIP switch DIP3, as shown in Table 1. You will have to carefully read the details for the various timing modes at the start of this article before making your selection.

Note that mode 8 is unused, as mentioned previously.

## Setting the delay

DIP switches DIP2 & DIP1 together set the time delay. DIP2 set the base timing interval, while DIP1 sets the multiplier (ie, Delay Time = base timing interval x multiplier).

**TABLE 1: MODE SELECTION**

| Mode | DIP3-1 | DIP3-2 | DIP3-3 |
|---|---|---|---|
| 1 | On | Off | Off |
| 2 | Off | On | Off |
| 3 | On | On | Off |
| 4 | Off | Off | On |
| 5 | On | Off | On |
| 6 | Off | On | On |
| 7 | On | On | On |
| 8 | Off | Off | Off |

*Table 1: the timing mode required is selected using DIP switch DIP3. Note that mode 8 is not used.*

Tables 2 & 3 shows the possible settings for these two DIP switches.

An example will illustrate how this all works. Let's say that DIP1-8, DIP1-3 & DIP1-2 are ON and that the rest of DIP1's switches are off. In this case, the multiplier is 128 + 4 +2 = 134.

This means that the Delay time will be 134 x base timing interval. So if the base timing interval is 10 seconds, for example, the Delay Time is 134 x 10 seconds = 1340 seconds, or 22 minutes 20 seconds.

If DIP1-7 is also turned ON, then this adds 64 to the delay factor making it 134 + 64, or 198. The maximum delay factor is with all switches ON; ie, 255.

Setting all the DIP1 switches to the OFF position is invalid and the timer will not function.

Note that there is some overlap between the timing intervals. For example, you can get a 10-minute delay by selecting a 1-minute timing interval and setting the delay factor to 10 or by selecting a 10-minute timing interval and setting the delay factor to 1.

In summary, here are the time delays possible:

- 1 - 255s in 1s steps;
- 10 - 2550 seconds (42min 30sec) in 10s steps;
- 1- 255 minutes in 1- minute steps;
- 10 - 2550 minutes (42hr 30min) in 10-minute steps.

The timing accuracy for all modes is .01%.

## Triggering the timer

As discussed earlier, the input trigger voltage needs to be in the range of

## WHERE TO BUY A KIT

Kits for the "K141 Multi-Mode Timer" are available from Ozitronics (**www.ozitronics.com**) for $32.45 (incl. GST) plus postage.

You can email the authors at **peter@kitsrus.com** if you have any suggestions. Information on other kits in the range is available from **http://kitsrus.com**

If you have any technical problems or questions, you can contact the kit developer at **frank@ozitronics.com**

Note: copyright of the PC board and the source code for the Atmel microcontroller is retained by the author.

| TABLE 2: BASE TIMING INTERVAL | | |
|---|---|---|
| **Interval** | **DIP2-1** | **DIP2-2** |
| 1 second | On | Off |
| 10 seconds | Off | On |
| 1 minute | On | On |
| 10 minutes | Off | Off |

***Table 2 (left): DIP switch DIP2 sets the "base timing interval". This value is multiplied by the "multiplier" (set by DIP1 – see Table 3) to give the Delay Time for the timer.***

**TABLE 3: INTERVAL MULTIPLIER**

| DIP1 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |
|---|---|---|---|---|---|---|---|---|
| Value | 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |

***Table 3: DIP switch DIP1 sets the interval multiplier. Note that if more than one switch is set to ON, the multiplier values are added together; eg, if DIP1-8, DIP1-3 & DIP1-2 are ON, the multiplier is 128 + 4 +2 = 134.***

6-81V, although this can be varied by changing the value of R1 (see earlier text). Just how the trigger voltage is applied will depend on your application and the trigger source available.

Figs.3-5 show the triggering options available.

Probably the most common device used for triggering the timer will be a simple "make" contact, either from a pushbutton switch or relay contacts. Fig.3 shows the idea.

All you have to do is connect the TRIG+ terminal to the VIN terminal and connect the switch or relay contacts between the TRIG- and GND terminals. When the contact closes, the circuit path is complete and current flows, thus triggering the timer.

Fig.4 shows how to trigger the timer using the open collector output of an NPN transistor (this can either be a discrete transistor or incorporated into an IC package). Basically, the transistor takes the place of the switch shown in Fig.4. When the transistor turns on, the TRIG- input is pulled low and the timer triggers, as before.

Note that you can connect multiple open collector outputs in parallel, together with a common pull-up resistor; eg, if you want to trigger the timer from more than one source. That way, one or more of the open collector outputs can go low without causing damage to the others.

In both the previous two triggering methods, the trigger source ground is connected to the timer ground. This is often referred to as "commoning" and is done to provide a common reference point between the two circuits. However, this bypasses the electrical isolation on the timer's input because one side of the optocoupler's input is now connected to ground.

Fig.5 shows the circuit to use if you want complete electrical isolation. Note that, to ensure isolation, the trigger source must drive the input without any connection to the timer's power supply.

## Relay outputs

The relay's NO, NC & C (normally open, normally closed & common) contacts are brought out to CON2 and can be used to switch external loads or other relays. In addition, VOUT and GND are provided as convenient connection points for powering external devices.

The relay outputs can be used to switch voltages up to about 40-50V. ***However, don't try to use the relay outputs to switch 240VAC mains voltages – that would be much too dangerous, especially given the proximity of the ground track to the relay outputs.***

If you do want to switch mains voltages, you can use the on-board relay to switch an external relay that's adequately rated for the job. ***Don't do this unless you are experienced know exactly what you are doing – mains voltages can be lethal!***

## Troubleshooting

Poor soldering ("dry joints") is the most common reason for the circuit not working. If you strike problems, the first thing to do is to check all soldered joints carefully under a good light and resolder any that look suspicious.

You should also carefully check that the parts are in their correct positions and that all parts are correctly oriented. Check also to ensure that the ICs have been correctly installed and that none of the pins have been bent under their bodies.

Finally, check that REG1's output is at 5V? If there is no voltage at the output of this regulator, check the voltage at its input.

If there's no voltage here, then it's possible that D1 has been installed the wrong way around – either that or you've inadvertently reversed the supply leads. SC