K148. 4-DIGIT TIMING MODULE with FIRMWARE MODULES

This kit is the hardware 'platform' that is the basis for a number of different timing modules. The **hardware** is the same for each kit – the only **change** is the **firmware** programmed into the microcontroller (uC) that controls it. Need another type of timer – simple, just change the uC chip. We can do custom changes.

At this time (8/2002), there are six timer firmware IC's available.

- **K148T1**. Simple Photographic Timer
- **K148T2**. Stopwatch with Pause function
- **K148T3**. 40KHz Auto Ranging Frequency Meter
- **K148T4** Programmable Down Timer counting down in minutes from a max of10,000 minutes
- **K148T5** Programmable Down Timer counting down in hours from a max. of 10,000 hours

Refer to the documentation for each specific module to see how they work. The sixth firmware IC **K148T0** for a Programmable Down Counter down-counting from a maximum of 10,000 seconds is supplied **with this kit**. The other firmware must be bought separately. Please note we do not provide the source code for any of our firmware.

First we describe the hardware module then on the last page we discuss the Programmable Down Timer **T0**.

HARDWARE SPECIFICATIONS

Supply voltage	9 to 12V DC
Supply current	30 to 50mA, depending on the number displayed.
Number and type of Inputs	3, active low, including hardware reset
Output	Open collector NPN transistor, 100mA @ 30V
Display	4-digit 7-segment with decimal point, 14mm RED LED
Physical size	51mm x 66mm (2.0" x 2.6")
Connection	10-way right-angle SIL header pins, 0.1" spacing

HARDWARE FEATURES

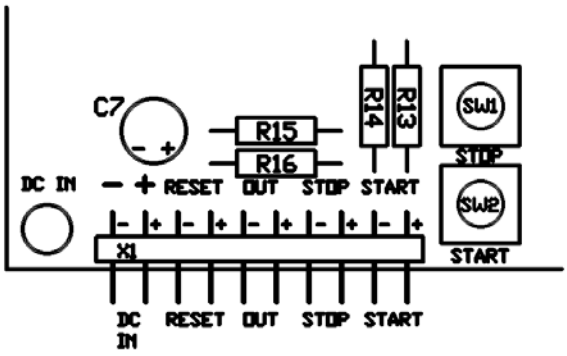
The kit features a 4-digit 7-segment display with decimal point. There is also an open collector output that can be used to operate a relay or sound a buzzer if required by your application.

There are three inputs to the kit. The RESET input is a hardware reset to the microcontroller. The other two inputs are labeled **START** and **STOP**. The actual function of these two inputs will vary between the different firmware used.

Inputs are normally pulled high and may be driven low by simple 'make' contacts from switches or relays or by an open collector output.

Two on-board zippy tact pushbutton switches are connected to the **START** and **STOP** inputs. These help to minimize the amount of external hardware required to get the kit 'up and running'.

All the inputs and output are organized as 'pairs' of pins, with each input or output having a corresponding ground pin, as shown in the following diagram.



The '+' sign indicates the actual input or output pin and the '-' sign indicates its associated ground pin.

Note: When using the output to switch a load (relay, buzzer, etc) connect the load between the output pin and a positive DC voltage. For example, if switching a 12V relay connect the relay between the output pin and +12V.

CIRCUIT DESCRIPTION

The circuit in this kit uses only one IC – the Atmel AT89C2051 microcontroller. It has 2K of Flash programmable and erasable memory and is compatible with the industry standard MCS-51™ instruction set. A data sheet can be downloaded from Atmel's website at "**www.atmel.com**"

You may ask why use an Atmel uC? Why not a PIC uC from Microchip? The answer is simple. Price. Here is Hong Kong where the kit is produced the price of the Atmel 89C2051 is one-third to one quarter the price of any Microchip uC. Had we used a PIC uC chip the price of this kit would have been an $US2 - $US3 more for you than it was and with no extra benefits whatsoever.

The IC is preprogrammed with firmware to provide each specific timer function. Using a microcontroller greatly reduces the component count while providing more features than using dedicated logic ICs. Cost is also lower. A 12.000MHz crystal provides a stable clock signal. This particular value was chosen because the microcontroller divides the crystal frequency by 12 to produce its own internal clock signal. This gives us an accurate 1uS timebase for elapsed time measurement.

The display is a 4-digit, common anode, multiplexed, 7-segment display. This means the LEDs in a single digit share a common anode (positive) connection. The cathodes (negative) of each segment (a,b,c,etc) are

connected across the four digits, forming a matrix. This minimizes the number of pins needed to drive the display but requires a more complex method to do it.

Multiplexing is a technique where each digit is only 'on' for a short period of time. In this kit, each digit is turned ON for 1mS in every 8mS. There is an OFF time of 1mS between each digit being turned on. This is much faster than the human eye can distinguish so it looks like the display is constantly on. This effect is called persistence of vision.

Resistors R1-8 limit the maximum current that can flow through each segment, including the decimal point. PNP transistors Q1-4 provide power to each digit and are switched by a low signal from the microcontroller via resistors R9-11.

The **START** and **STOP** inputs are pulled high by 10K resistors and have a low pass filter formed by a 1K resistor and 1nF capacitor to filter out high frequency noise. This reduces the chance of false triggering. The filter's time constant is 1uS – pulses shorter than this won't make it to the microcontroller.

Power on reset is provided via capacitor C7. The microcontroller requires an active high reset signal but, in keeping with the other two inputs, we wanted an external active low input to reset the kit. Transistor Q5, a PNP device, is normally held off by the 10K pullup resistor R19 on its base. An external low reset input, via resistor R18, turns the transistor on which switches 5 volts to the reset input of the microcontroller.

Transistors Q6 and Q7, both NPN types, are used in conjunction to provide an active low, open collector output. Q7 is protected by zener diode Z1 which will break down and conduct if the voltage across Q5 exceeds 33V, or it will conduct if a negative voltage is applied to the collector. This is needed when driving inductive loads such as relays, as the back EMF generated by relays coils can exceed the transistor rating and damage it. At first glance you may wonder why we are using two transistors for the output. Why not eliminate one of the transistors and simply use an active HIGH signal from the microcontroller to switch the output transistor? 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 the I/O pin was connected directly to the output transistor then the output would be 'on' during reset. Of course it would switch 'off' after reset once the onboard firmware took over. However, the output would "flick" on momentarily during the reset period – not what we want. Using the extra transistor means we can use a LOW signal to turn the output on and a HIGH to turn it off - just right during reset! Pull-up resistor R17 ensures a 'solid' high level signal to turn the output off.
The circuitry is powered by IC2, a 78L05 low-current 5V voltage regulator. It requires about 2.2V of 'headroom' (input voltage – output voltage) to operate properly.

Diode D1 provides reverse polarity protection in case the power supply is connected the wrong way around. There is a 0.6V drop across the diode so the kit requires at least 8 volts (5V + 2.2V + 0.6V) to work.

Timing Accuracy: The crystals we use have a tolerance of +/- 30 ppm (parts per million). So the actual crystal frequency could vary by as much as 360Hz either side of 12.000MHz - an uncertainty of +/-0.003%. Over a 1 hour timing period this amounts to a maximum error of +/-0.108 seconds/hour. However, testing our prototypes showed the error was more like –1.25 seconds/hour (-0.035%) using the local telephone company's time information service. The reason for this (**assuming** that the software does not contain clock ticks unaccounted for) is that the oscillation frequency is dependent on the oscillator **circuit –** in this case a Pierce oscillator circuit with both load capacitors grounded **-** as well as the crystal. (And then there are other second order effects factors such as temperature, age and track layout but we will leave those out of the discussion.) If you want to get an accuracy below 0.01% then you must not only specify a frequency to the manufacturer (in this case 12.000MHz) but you must **also specify** a C_L oscillator load reactance of the crystal for the circuit it will be used in. C_L for these crystals is the default value 32pF.

This it also assumes we have the test gear to be able to measure down to 0.1Hz which we do not.

In conclusion, the kit as assembled will be accurate within +/-0.05% (or 1.8 seconds/hour.) Where possible do an accurate test of the kit over 24 hours (1440 minute) period using the telephone company's time service and determine the number of seconds gained or lost per hour for your particular kit. Then you can allow for this deviation when working out future timing requirements. If you wish you can experiment with changing the 2 load capacitors (say between 10pF and 56pF) to try reduce the timing uncertainty.

ASSEMBLY INSTRUCTIONS
The PCB pads are quite small and are relatively close to each other. It is recommended that you use a fine tipped soldering iron and thin solder. Do not use too much solder – it increases the risk of solder 'bridges' between adjacent pads. Use a good light so you can clearly see what you are doing.

First identify each of the components against the parts list. Separate them into groups so they are easy to find when assembling. Follow the component legend on the PCB when placing components.

Install all the resistors first then the diodes. The cathode (striped) end of the diode should match the stripe on the PCB overlay. Next is the crystal. Note that it is located **inside** the IC socket. Make sure it is sitting down against the PCB before soldering.

K148. 4-DIGIT TIMING MODULE with FIRMWARE MODULES

Now comes the IC socket. Cut the 20 pin IC socket apart at each end. This is necessary because the crystal will not fit inside IC socket if it is not cut apart. Put in the crystal then solder in each cut half of the IC socket pressed up against the crystal. The 89C2051 will fit OK in the slightly moved-apart IC socket.

Next put in the capacitors, paying particular attention to C1 and C7. They are polarized and need to be inserted the correct way around. These are clearly marked on the PCB overlay. Now install the transistors and IC2. Don't get these confused. Transistors Q1-Q5 are BC557s and transistors Q6 and Q7 are BC547s. IC2 is a 78L05 voltage regulator. Use the outline on the PCB as a guide for orientation.

Push the transistors down as low as possible (without being too excessive) so that they are below the level of the display when it is inserted. You can temporarily insert the display to check this. This will help later on if you decide to mount the kit in a box. Double check that you don't have any solder bridges across the transistor pins as they are close together.

Install the LED display – the decimal points go towards the microcontroller as per the component overlay. Then install the 10-pin 90° header for the inputs and output. A male and female 10-way header strip , provides external connections from the kit. **Note:** you have a choice whether to use the male or the female header soldered onto the PCB. Consider how you want to use the Timer and decide which connector to solder in.

Finally, install the two tact push-button switches. **Do not install a microcontroller into the IC socket yet.**

Apply power to the board. Use a multimeter to measure the voltage across pins 10 (-) and 20 (+) of the IC socket. The voltage should be around 5 volts +/- a few millivolts.

If all is well remove power and carefully insert the microcontroller into its socket (noting its polarity). Check that the IC leads are actually in place and are not bent up under the body of the IC. After re-checking the board, apply power and the display will light. The digits displayed will depend on the specific timer kit. In most cases it will be all zeros.

Open Collector Output. For more information about what an open collector output is read the note at
www.kitsrus.com/zip/opencol.txt

PARTS LIST – K148

Resistors (0.25W carbon)

270R Red, Purple, Brown	R1-8	8
1K Brown, Black, Red	R13,14,18	3
4K7 Yellow, Purple, Red	R9-12,17,20	6
10K Brown, Black, Orange	R15,16,19	3

Capacitors

22pF ceramic	C3,4	2
1nF 102 ceramic	C5,6	2
100nF monobloc	C2	1
1uF 16V electrolytic	C7	1
10uF 25V electrolytic	C1	1

Semiconductors

33V 1W zener diode	Z1	1
1N4004	D1	1
BC557 transistor, PNP	Q1-5	5
BC547 transistor, NPN	Q6,7	2
78L05, 5V regulator	IC2	1
LED display, 4 digit, common anode, LN5644		1
AT89C2051-24PC '**T0'** firmware……	IC1	1

Miscellaneous

Zippy pushbutton switch	SW1,2	2
Crystal, 12MHz	Y1	1
20 pin IC socket for IC1		1
Male Header, 10 pin R	X1	1
Female Header, 10 pin R	X1 alternative	1
PCB, K148		1

IF IT DOESN'T WORK

Poor soldering ("dry joints") is the most common reason that the circuit does not work. Check all soldered joints carefully under a good light. Re-solder any that look suspicious. Are there any solder 'bridges or splashes' shorting out adjacent points on the PCB?

Check that all components are in their correct position on the PCB. Is the diode and electrolytic capacitors inserted the right way round?

What about the transistors? Q6 and Q7 are NPN types (BC547) while all the others are PNP types (BC557.) Did you get them mixed up? Did you confuse the 78L05 regulator with one of the transistors?

Use a multimeter to measure the DC input voltage to the kit. It should be at least 8 volts. Anything less and the 5V regulator, IC2, will not operate correctly.

Web Address & Email

This kit was designed and developed by Frank Crivelli at Ozitronics. If you want slightly altered firmware for your particular application then this is the man to contact. The source code for these kits is not available. You can email Frank at **frank@ozitronics.com**

You can email me at **peterhk@kitsrus.com**

See our website at **http://www.kitsrus.com** for information about our Kit 123, Atmel 89Cxxx Programmer, used to program the AT89C2051.

K148. 4-DIGIT TIMING MODULE with FIRMWARE MODULES

Programmable Down Timer

With this kit we have supplied a uC which contains the program for a 4 digit programmable down timer with output and reset. The uC is marked '**T0**'. Timing is in **seconds** with a maximum programmable time of 10,000 seconds (0000) which is 2 hours, 46 minutes and 40 seconds. It has **five operating modes** that control the output function.

Timing range	0000 to 9999 seconds Starting at 0000 gives 10,000 seconds (2h 46m 40s)
Inputs	Start, Stop and Reset Active low (see text)
Output	Open collector NPN transistor, 100mA @ 30V
Supply voltage	9 to 12V DC
Supply current	30 to 50mA, depending on the number displayed.
Display	4-digit 7-segment with decimal point, 14mm RED LED
Physical size	51mm x 66mm (2.0" x 2.6")
Connection	10-way right-angle SIL header pins 0.1" spacing. Male or female

After you place the uC in the socket and apply power the display will show **0000**

PROGRAMMING THE TIMER

The two buttons marked S**TART** and **STOP** are used to **program the starting time** (4 digits) first, and **select the operating mode** (1 digit) second.

When you apply power you will see **0000** in the display.

If you press the **START** button now you will start to count down from 10,000 secs. So do not do that. If you did, reconnect the power and start again at **0000**

Programming the start value is done **one digit at a time** starting with the leftmost thousands digit. The decimal points are used to indicate which digit to its left is being set. Press the **STOP** button once to enter programming mode. The leftmost decimal point will come and show **0.000** Now use the START button to set the value required from 0-9. When you have selected the thousands digit you want (say 5) press the STOP button and you will see **50.00** The 5 has been set and you are now ready to program the hundreds digit. Press the START button to set the hundreds digit, say 4. Press the STOP. You will see **540.0**

Repeat the steps above for the tens and units digits.

After you set the units digit and press STOP the display will switch to allow the **operating mode** to be set. The current operating mode, probably **1** will be displayed. Use the START button to set which of the 4 operating modes you want (see below for a description of each.) After you have selected the mode you want then press the **STOP** button. The display will blank momentarily to indicate that programming mode has ended and the 4 digits you set to count down from will be displayed. For example **5460** if you want to count down from 91 minutes (60 x 91 = 5460 seconds.) The timer is now programmed ready to go.

STARTING THE TIMER

Press the **START** button or apply a ground to the start input and the timer will start counting down towards zero. **NOTE**: the STOP button or input has no affect while the timer is counting down.

STOPPING THE TIMER

The only way to stop the timer is via the reset input. Short the Reset pin to ground. The timer will reset to its programmed value - the operating mode is not affected. If the timer loses power it will restart in Mode 1 with a preset value of **0000** (10,000 seconds).

OPERATING MODES

There are five operating modes that control the timer and the output. The RESET input does not affect the operating mode.

Mode 1. Timer Stop, Output Hold (default)

This is the default mode at power up. The timer stops when it reaches zero and the Output goes low and stays low. You have to press Reset (short the Reset pin to ground) to continue.

Mode 2. Timer Overrun, Output Hold

Same as Mode 1 except the timer continues counting down past zero, wraps around to 9999 and starts counting down from there. The Output pin goes low and stays low. Short the Reset pin to ground to return to the preset timer value.

Mode 3. Auto Reset, Pulse Output

When the timer reaches zero the Output pulses low for 20mS and the timer resets itself to the programmed value and stays there. You can start to count down again from the preset value by pressing START.

Mode 4. Timer Overrun, Pulse Output

Same as Mode 2 except the output pulses low for 20msec instead of staying low. Counting wraps to 9999 and starts counting down. Short the Reset pin to ground to return to the preset timer value.

Mode 5. Output On, Timer Run

Pressing START turns on the output and starts the timer. When the timer reaches zero the output turns off and the timer resets to the preset value where it stays until START is pressed agan.

If the counter has stopped counting down you can reset the timer value by pressing STOP and resetting the time and the mode.

Accuracy. Timing will be accurate to within +/- 1.8 sec per hour (3600 seconds).

K148. 4-DIGIT TIMING MODULE with FIRMWARE MODULES