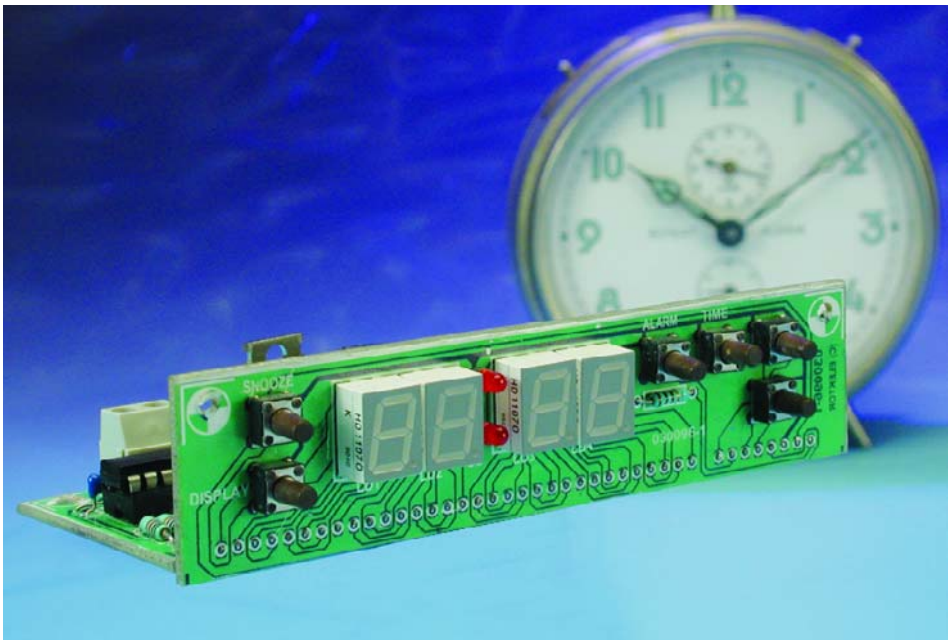
Digital Alarm Clock

Based on a PIC micro

Design by M. Conde de Almeida maconde@terra.com.br

Although digital alarm clocks have been around for years, most of today's off the shelf products suffer from serious design limitations. For example, many don't keep track of weekdays and can only store one alarm time. High time for a home-brew design that does a better job.



Off the shelf alarm clocks present a serious limitation if, for instance, you and your partner have different wake-up times or if you have to give or take medication at regular intervals. This limitation gets more serious if you want alarms to go off only on specific days of the week or only during work days. The time setting process on most clocks is also subject to improvement. In the majority of cases you adjust time by incrementing minutes and hours. If your clock is at, say, 06:15h and the correct time is 20:58h you need to keep a key pressed for quite some time and release it well before the desired hour/minute, switching to 'slow' setting. If not, you're past the desired time and have to start all over again.

This article proposes a solution for these design weaknesses by discussing a low-cost circuit based on the popular PIC 16F84A microcontroller from Microchip.

Our digital alarm clock keeps track of weekdays and has eight alarms that can be individually set to go off every day, only on working days or, if you want, on a specific day of the week.

The time setting process allows the adjustment of each digit of the clock separately by means of Up and Down (+ and –) keys. It also incorporates some other interesting features.

Like all good digital alarm clocks our project has a battery that keeps the clock ticking in the event of an AC power failure. In battery-powered mode the display is turned off to reduce energy consumption. However, if you want to check the time you may still enable the display by pressing a 'display on' button. In battery mode the alarms will continue to operate normally. When an alarm goes off the display will be turned on to show the current time.

The eight different alarm times are kept in the PIC's EEPROM. So, you won't have to adjust any of the alarms again even if the battery fails. A 'snooze' key will temporarily turn an alarm off. The alarm will be triggered again after a minute until it is definitely turned off. A master Alarm On/Off key will enable/disable all the alarms, independent of their individually set states. A bright LED display makes it easy to check the time from a distance or in the dark.

The 4-MHz crystal oscillator guarantees a pretty accurate timebase for the clock (error = 0.000427%).

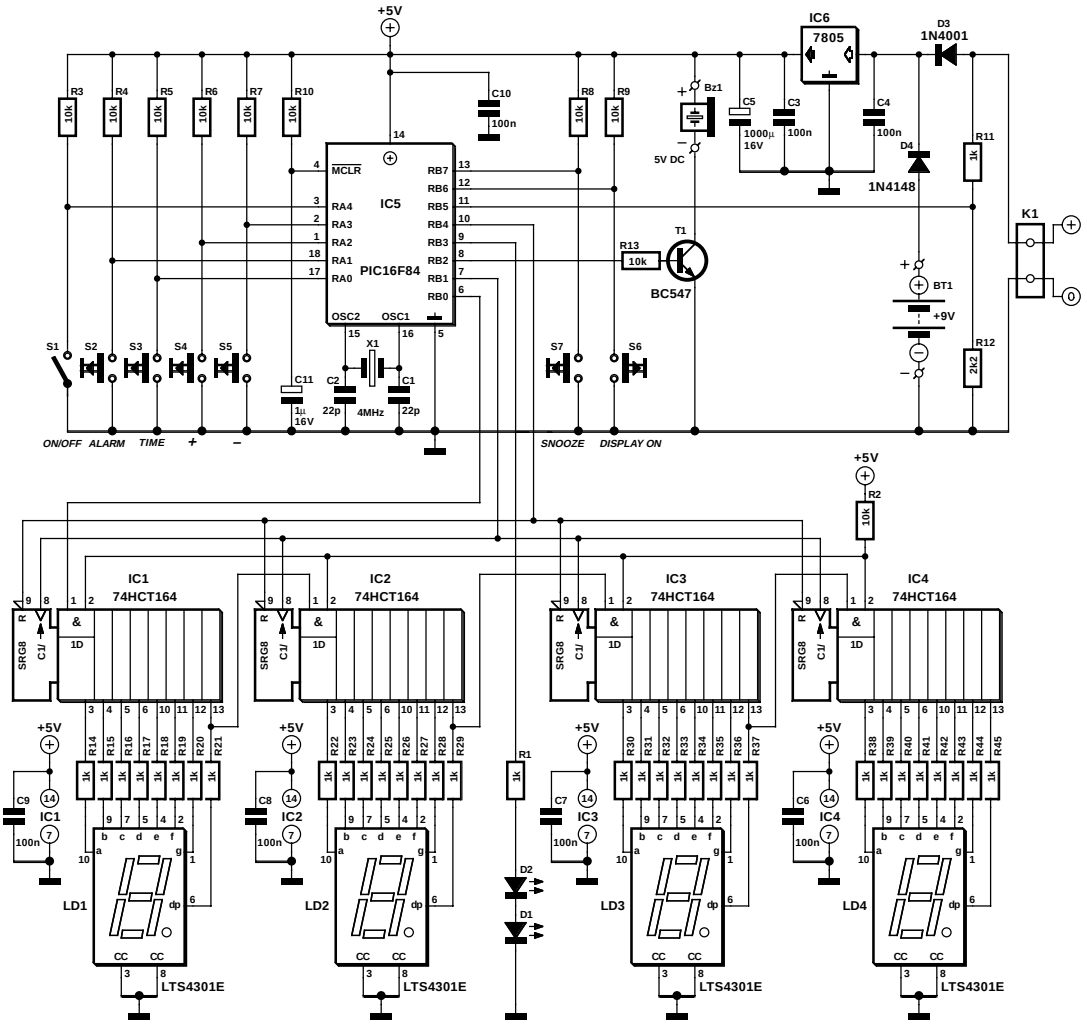


Figure 1. Circuit diagram of the PIC-based Digital Alarm Clock.

The hardware

Figure 1 shows the complete circuit diagram of the clock. All intelligence (and a lot of logic) is vested in the PIC16F84 MCU in position IC5. Using RA0-RA4 and RB5-RB7 as input port lines and RB0-RB4 as output port lines, a fair amount of executable code run from the on-chip memory is able to take total control of the circuit, requiring just a 5-V supply voltage and a clock signal generated with the aid of an external 4-MHz quartz crystal, X1.

Components R10 and C11 guarantee that the microcontroller is reset at power-on.

Switches S1-S5 are connected to the microcontroller's PORTA pins (all configured as inputs). Resistors R3-R7 guarantee a High logic level at the PORTA pins when the switches are open. When closed, these keys will force a LOW state on the PORTA pin they are connected to, triggering the execution of specific clock control routines.

PORTB.6 and PORTB.7 are also configured as inputs. R8 and R9 guarantee a High logic level at these inputs when the associated switch S6 or S7 is open. S7, when closed, will activate the 'Snooze' function. Similarly, S6 will temporarily activate the display. This function is only available when the clock is operating in the battery mode.

Resistors R11 and R12 form a voltage divider fed by the main 12 VDC source. They guarantee a High logic level at PORTB.5 when the main supply voltage is available. In this situation the display will be permanently on. If for some reason the main power is not available (AC power failure) there will be a Low logic level at PORTB.5 and the entire display will be turned off by the software.

Looking at the output devices controlled by the PIC, the alarm buzzer Bz1 is driven via PORTB.2 while PORTB.3 controls LED1 and LED2, the Hour/Minute separator in the readout.

Four common-cathode 7-segment displays, LD1-LD4 constitute the clock readout. These displays are driven by four 74HCT164 shift registers (IC1-IC4) connected in series. R2 only serves to guarantee a High level to the enable pins of the shift registers. PORTB.1 drives the CLOCK pin (8) of the shift registers while PORTB.0 drives the DATA line of the first shift register in the chain (IC1, pin 1). The microcontroller program will clock 32-bit strings into the shift registers whenever a display update is necessary. PORTB.4 controls the RESET pin of the shift registers. This port line will be held Low when the display is to be turned off.

The regular 12 VDC power may be supplied by a standard AC or DC adapter. Diodes D3 and D4 will guarantee exclusive operation of either the 9 V battery or the 12 VDC power

source. They will also guarantee that R11 and R12 are only powered by the 12 VDC source.

The 5 V supply voltage for the logic circuitry is provided by a 7805 regulator (IC6). Capacitors C3, C4, C5 and C6-C9 help to keep the supply voltage as clean as possible.

Current consumption of the clock in normal use (i.e., powered by the mains adapter) is of the order of 75 mA. The current from the 9-V backup battery amounts to about 5 mA.

Control software

The entire clock program was written in the PIC Assembly Language using the MPLAB Integrated Development Environment (v. 5.70.40) supplied free of charge by Microchip.

The source code and Hex files containing the program ready to be flashed into the PIC microcontroller can be obtained free of charge from the Publisher's website, see 'the Free Downloads' inset. The file number is **030096-11**. For those without access to a PIC programmer, the microcontroller is also available ready-programmed under number **030096-41**.

The flowchart in **Figure 2** summarizes the operation of the program. After an initialisation routine where I/O pins, Interrupt register, Timer_0/Prescaler operation are configured and alarms information is read from the EEPROMs into the PIC data memory, the program enters a loop in which control keys are read and, based on their status, a specific branch is taken.

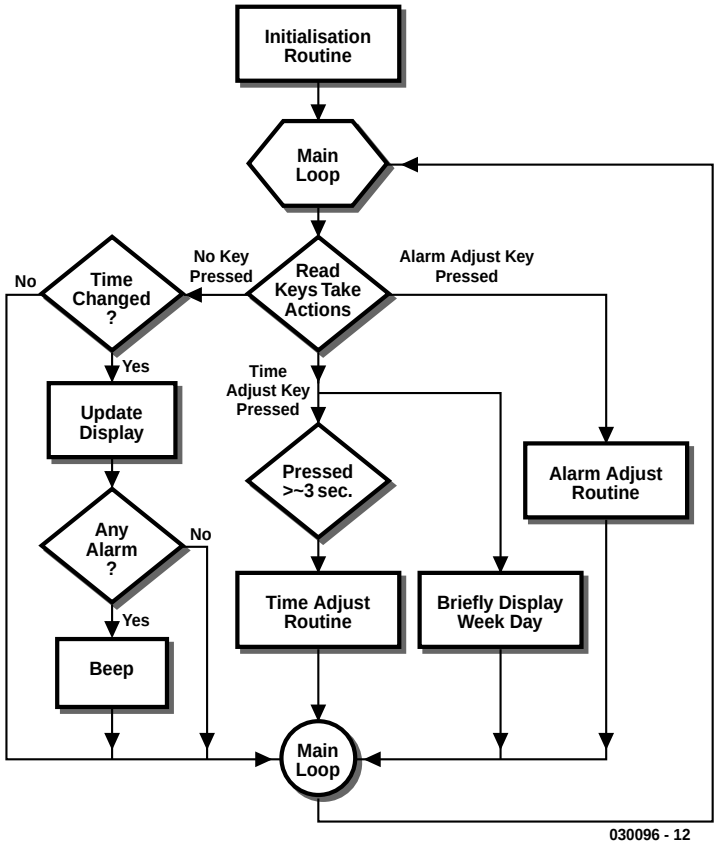
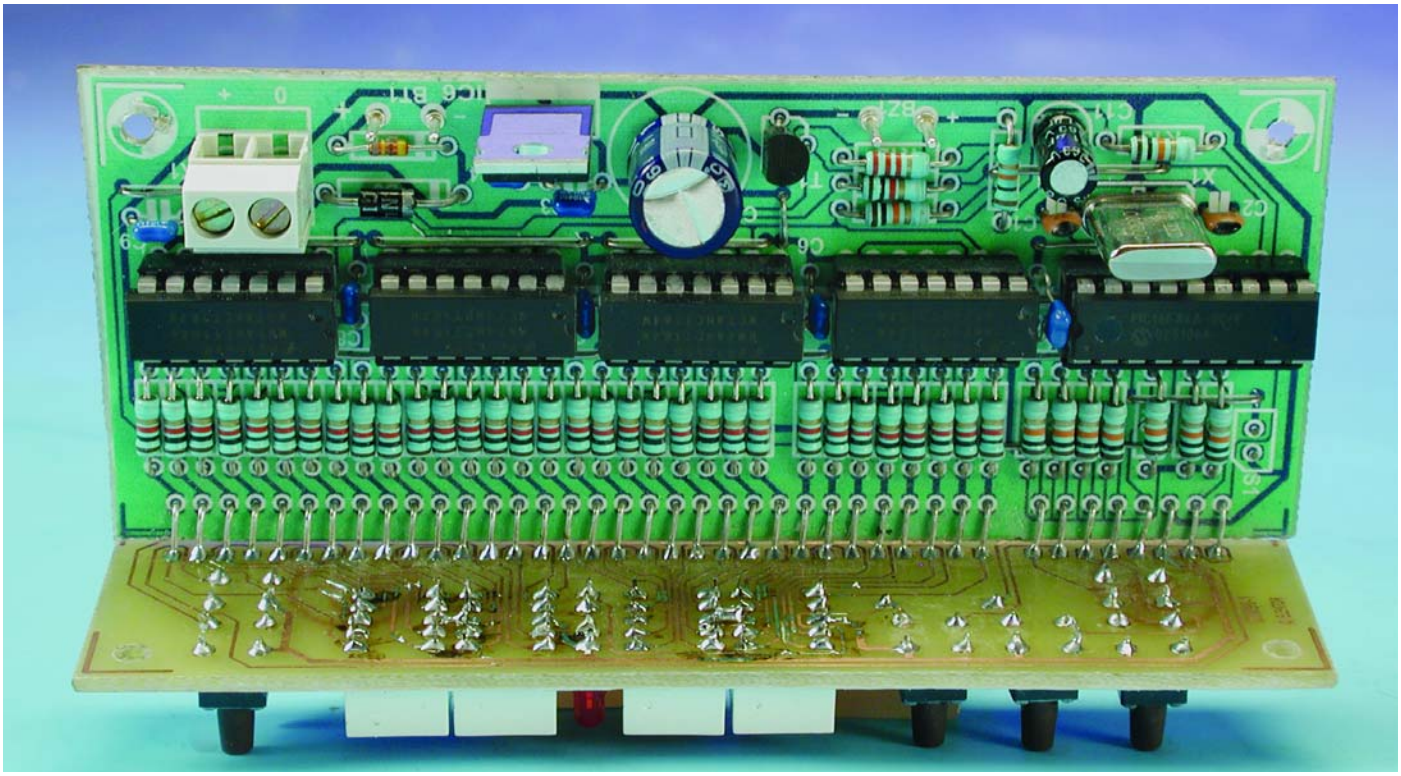


Figure 2. Flow chart of the program run by the PIC.

For example, if the Alarm Adjust key is pressed and held down longer than about 3 seconds, the program will execute the Alarm Adjust Routine and, once finished, return to the main loop.



COMPONENTS LIST

Resistors:
R1, R11,R14-R45 = 1kΩ
R2...R10,R13 = 10kΩ
R12 = 2kΩ2

Capacitors:
C1,C2 = 22pF
C3,C4,C6-C10 = 100nF
C5 = 1000µF 16V radial
C11 = 1µF 16V radial

Semiconductors:
D1,D2 = LED, 3mm, red, low-current
D3 = 1N4001
D4 = 1N4148
T1 = BC547
IC1-IC4 = 74HCT164
IC5 = PIC16F84-04/p, programmed, order code **030096-41**
IC6 = 7805

Miscellaneous:
K1 = 2-way PCB terminal block, lead pitch 5mm
S1 = on/off switch
S2-S7 = miniature PCB mount pushbutton, type DTS65N
LD1-LD4 = LTS4301E (Lite-On)
BZ1 = 5VDC buzzer (active)
X1 = 4MHz quartz crystal
BT1 = 9V battery with and clip-on lead
PCB, available from **The PCBShop**
Disk, PIC source and hex (object) code files, order code **030096-11** or Free Download

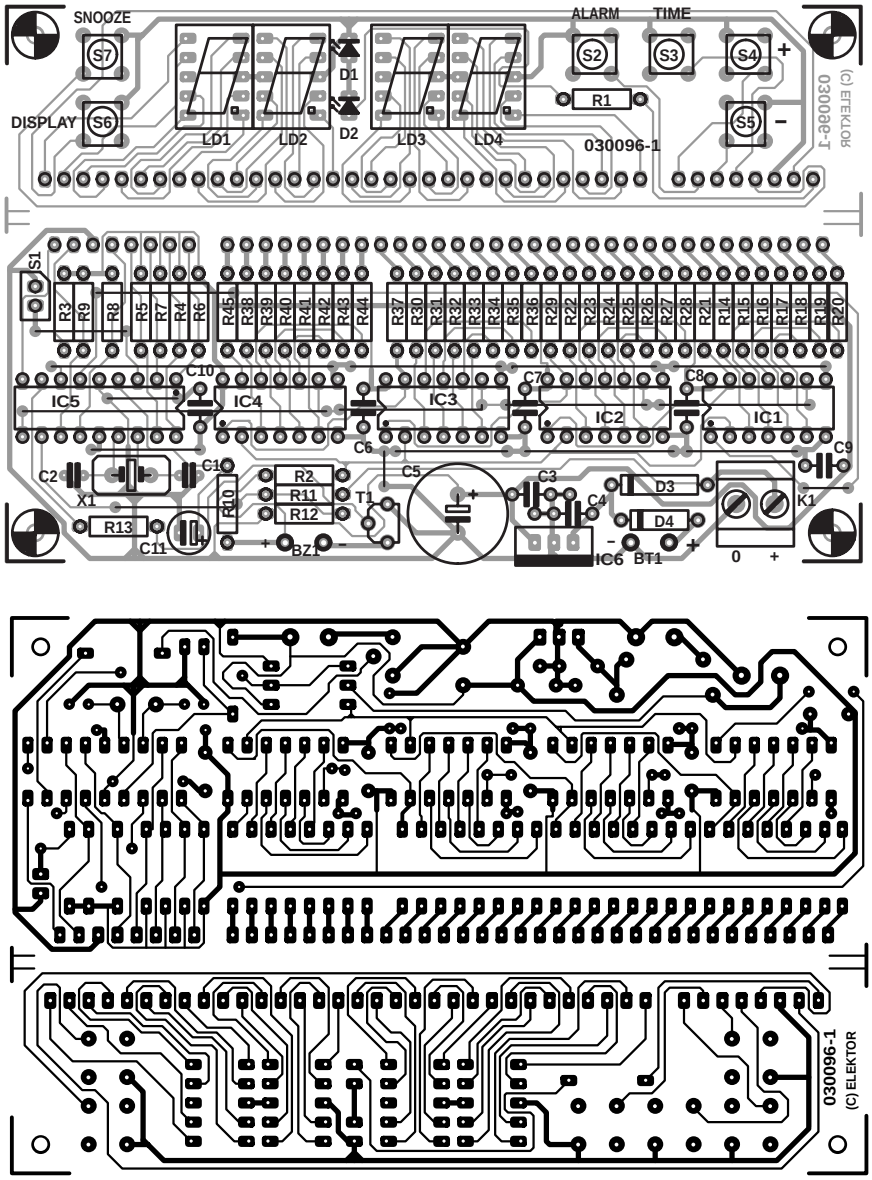


Figure 3. Copper track layout and component mounting plan of the PCB designed for the clock (board available from The PCBShop).

If the Time Adjust key is held depressed for less than about 3 seconds, the display will show the current weekday. If the key remains pressed longer than about 3 seconds, the program will execute the Time Adjust routine and, once time and weekday are adjusted, return to the main loop.

When no key is pressed the program keeps updating the display, checking the current time against the alarm entries in memory and triggering the beeper if a matching times are found.

Although it has not indicated by flowchart, the display updating process is affected by the status of the PORTB.5 line (12 VDC supply monitor) while the alarm verification and triggering processes are subject to the status of Alarm On/Off and Snooze keys.

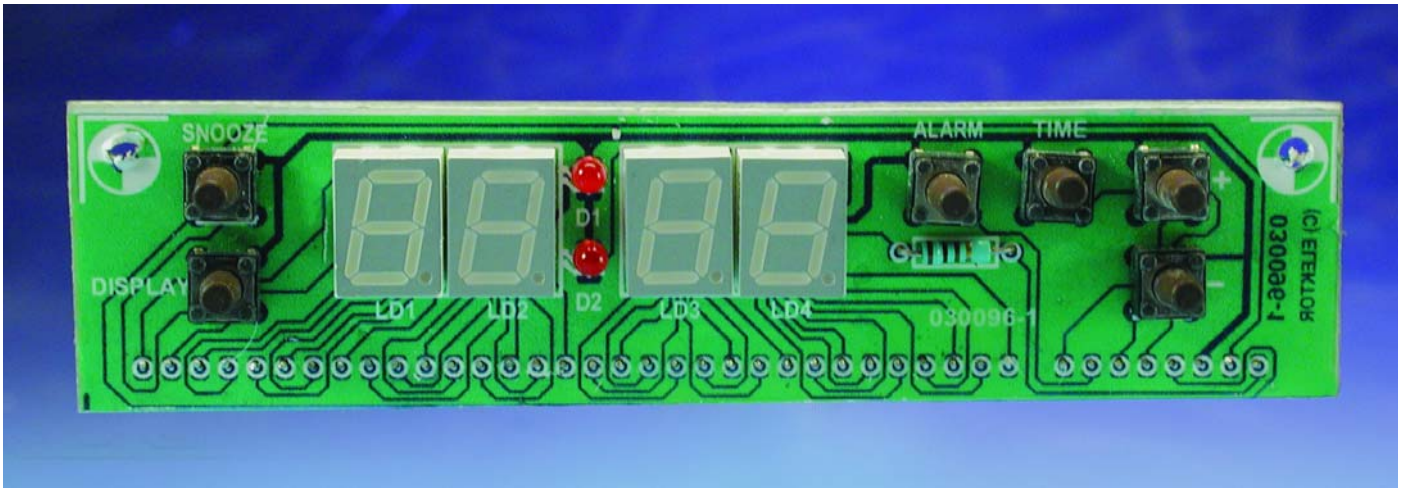
One important piece of the software is the time counting process. It is based on the microcontroller TIMER_0 which has the ability to interrupt the main program and deviate its execution to an interrupt handling routine whenever its count overflows (FFh → 00h).

The microcontroller is configured so that the TIMER_0 overflow will take place every 2.048 ms. This is accomplished by setting the prescaler to divide the TIMER_0 clock by eight.

Whenever called, the interrupt handling routine, will increment a 16-bit counter. This counter has a maximum count of 29,297 (7271h). This means that every minute (29,297 × 2.048 ms = 60,000.256 ms or very close to one minute) it will overflow.

The overflow of the 16-bit counter increments the minute-counter (modulo-10) which, after overflowing, increments the tens-of-minutes counter (modulo-6) which increments the hour-counter and sure it will finally increment the tens-of-hours counter.

As shown above, the clock's minute is slightly longer than it should be. This error, amounting to 0.000427%, will cause a time difference of less than two minutes per year which is pretty low compared to what is achieved by most the off-the-shelf clocks.



Construction

The artwork of the PCB designed for the clock is shown in **Figure 3**. The PCB has to be cut in two to separate the main board from the display board.

Start by soldering the wire links onto the main board. Then follow the five IC sockets and all low-profile components. The rest of the construction should be mostly plain sailing as only regular components are used. As always, take care with the polarity of electrolytic components like capacitors, diodes and (yes!) the displays. The display board is secured to the main board at an angle of 90 degrees. The interconnections between the boards are made using angled SIL pin-headers or 40 short pieces of stiff, bare wire. The pushbutton actuator rods will protrude a little above the display tops to make them accessible from the outside once the PCB assembly has been mounted behind the front panel. A red bezel may be used as a finishing touch to enhance the appearance of the clock.

Using a multimeter, make sure that there's no short circuits between the +5 V supply rail and ground.

When inserting the integrated circuits always check the correct 'pin 1' position. Finally, avoid ESD damage to your circuit by keeping yourself grounded by means of wrist straps.

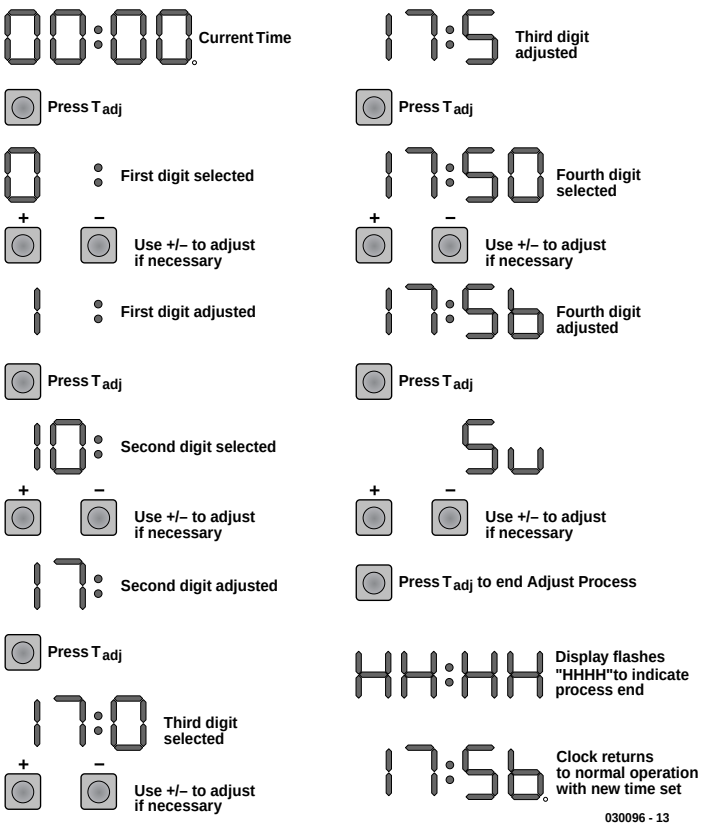


Figure 4. Time adjustment procedure.

Clock operation

The Time adjustment is shown pictorially in **Figure 4**. Other drawings are available but not printed here for lack of space. They include:

– clock display;
– keys and their functions;
– alarm on/off key operation;
– buzzer control procedure;
– operation in battery mode (mains power failure);
– alarm adjustment,

The pictures are supplied in the form of a (zipped) pdf document — see this month's Free Downloads, number **030096-12**.

(030096-1)

Free Downloads

PIC source and hex (object) code. File number: **030096-11.zip**
Clock operation pictograms. File number: **030096-12**
PCB layout in PDF format. File number: **030096-1.zip**
www.elektor-electronics.co.uk/dl/dl.htm, select month of publication.