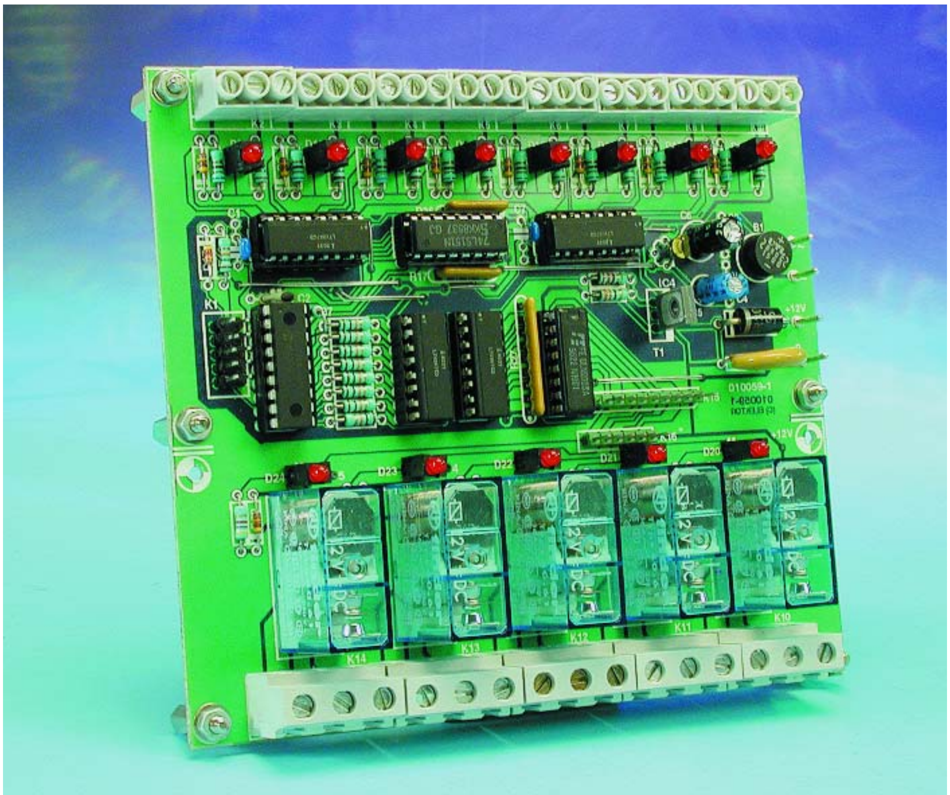
Pico PLC

microcontroller or Programmable Logic Controller?

Design by H Gebhard, DF2DS

There are certainly strong similarities between these two approaches to controlling equipment, and with the circuit described here we blur any remaining distinction still further. The Pico PLC employs only standard components and the necessary software development tools are available free on the Internet for private users.



A distinction is frequently drawn between microcontrollers and PLCs (programmable logic controllers) used in industrial control applications. The first thing that is apparent is that the PLC is designed for industrial use: in particular this usually means a supply voltage of 24 V. Also, the circuit will be robust against supply polarity reversal, voltage spikes and short circuits.

We find a further difference in the software. A PLC runs a monitor program which controls the sequence of events executed. This involves, among other things, the use of interrupts, software timers and dealing with various interface protocols (frequently RS-232 or CAN bus).

The user program always executes cyclically over a fixed period of time. This ensures that the timing behaviour of the controller is predictable. An important prerequisite here is the ban on 'busy wait' loops: only in the main program loop can the user test whether a particular event has occurred.

Various methods are suggested for programming PLCs, such as the 'ladder diagram' (which looks like a set of interconnected switches) or as an 'instruction list' (which looks more like assembler source code). The user program is stored in the PLC in flash memory and is run automatically when power is applied.

Is a microcontroller better than a PLC?

There is no completely satisfactory answer to this question. Microcontroller-based systems are consider-

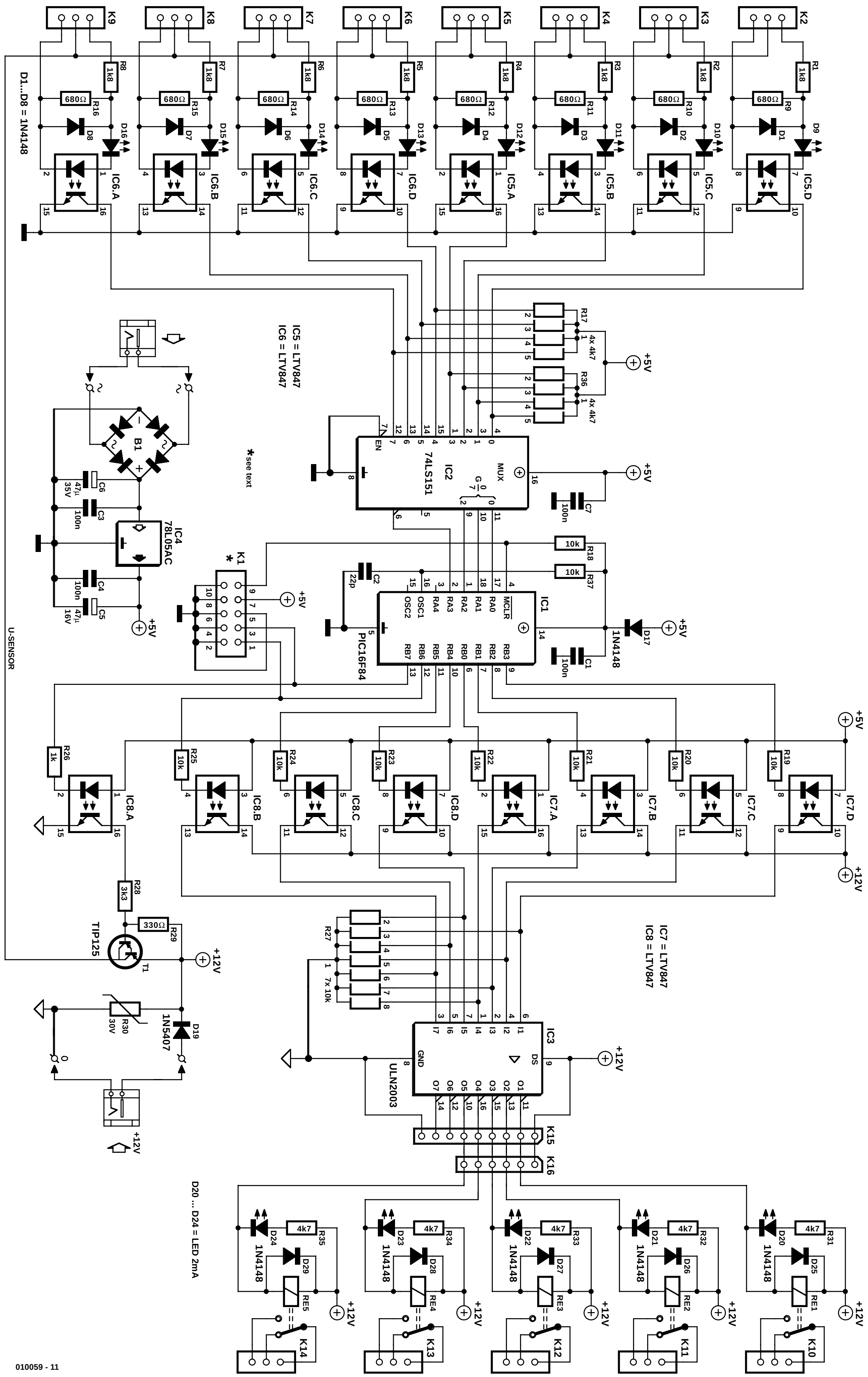


Figure 1. A microcontroller with isolated inputs and outputs.

ably more flexible, but require much more programming experience and discipline when used as PLCs. The possibilities are much wider, since the full range of assembly-code instructions is available for use. Almost always a high-level language compiler is available, generally C or BASIC, although sometimes Pascal or other more exotic languages are found.

It should however be clear to any developer that in using a compiler for a high-level language it will not be possible to squeeze the last clock cycle of performance out of the microcontroller. However, programming in a high-level language does confer the advantages of simplifying debugging and modification or adaptation of the code, when compared to using assembler.

The Pico PLC

The Pico PLC system can be used in both application areas. At the heart of the hardware is a Microchip PIC16F84 microcontroller [1]. This microcontroller offers 13 digital I/O port lines, 64 bytes of EEPROM and 1024 14-bit words of flash program memory. At first sight this seems laughably small compared to the amount of memory found in a PC, but thanks to the specialised highly-efficient instruction set it is nonetheless adequate for many applications.

If the microcontroller's tasks are not time-critical, the quartz crystal can be dispensed with and the processor clock generated using its internal RC oscillator (with R37 and C2).

As can be seen from the circuit diagram in **Figure 1**, there are no secrets lurking inside the processor. The inputs are brought one-by-one to port pin RA3, which is configured as an input, via multiplexer IC2. The selection of the input is done using port pins RA0, RA1 and RA2. The inversion at the output of the multiplexer cancels out the inversion which occurs in the optocouplers.

The individual inputs are electrically isolated using optocouplers, as demanded by standard PLC hardware design practice. The potential dividers prevent too great a current flowing through the optocoupler with a high input voltage (here between 10 V and 30 V). The diode connected in antiparallel with the optocoupler is important: it protects the LED in the coupler from a reverse polarity input. The infrared LEDs used have a low maximum reverse voltage (6 V maximum for the PC847 [2] or LTV847).

A special feature is the *U sensor* connection, which is an output driven by the microcontroller. Using this the sensors can be supplied with power only when required. This power-saving feature is particularly useful in battery-operated systems.

The output drivers are also electrically isolated from the controller

COMPONENTS LIST

Resistors:
R1-R8 = 1kΩ8
R9-R16 = 680Ω
R17,R36 = 4-way 4kΩ7 SIL array
R18-R25 = 10kΩ
R26 = 1kΩ
R27 = 7-way (or 8-way) 10kΩ SIL array
R28 = 3kΩ3
R29 = 330Ω
R30 = varistor 30V, 600mW, diameter 15-17 mm (e.g., BC-Components # 2322 5953006)
R31-R35 = 4kΩ7
R37 = 10kΩ (4kΩ7)*

Capacitors:
C1,C3,C4,C7 = 100nF ceramic
C2 = 22pF
C5 = 47µF 16V radial
C6 = 47µF 35V radial

Semiconductors:
B1 = B80C1500 (80V piv, 1.5A), in round case
D1-D8,D17,D25-D29 = 1N4148

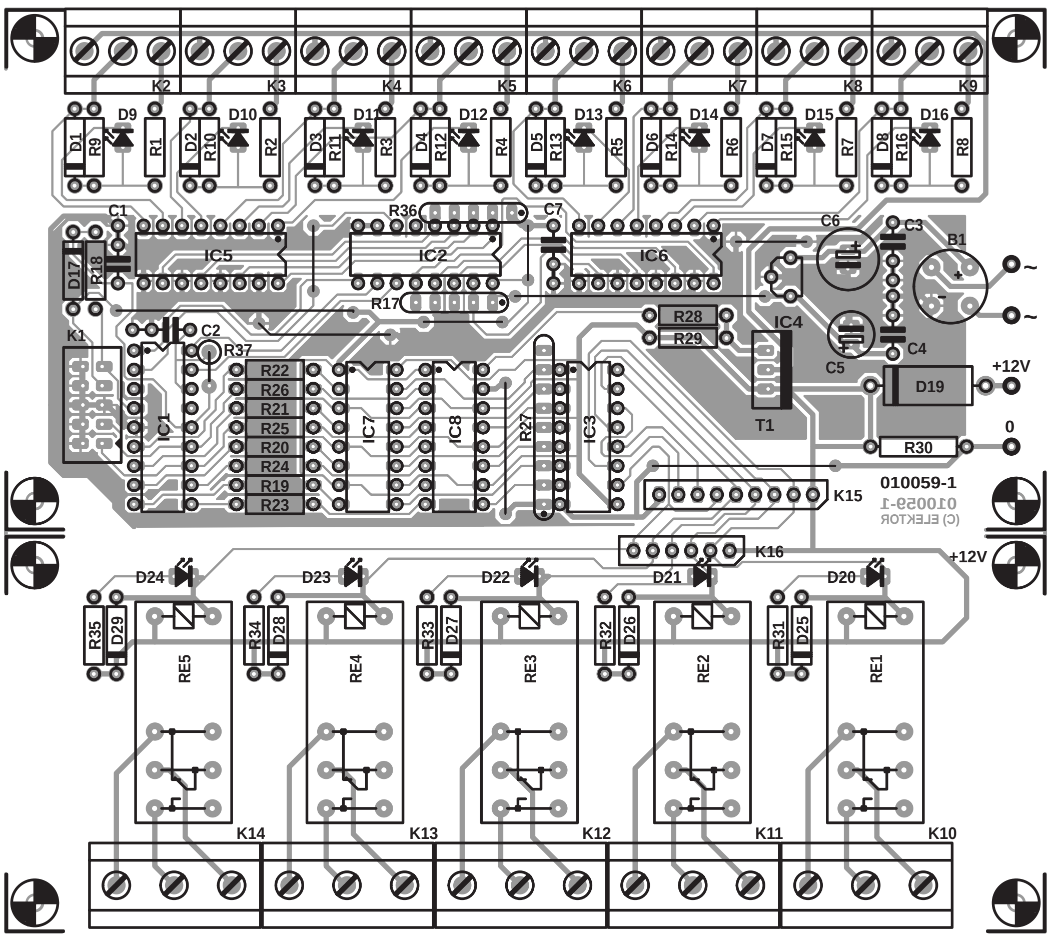


Figure 2. The part of the circuit board carrying the relays can be separated from the rest of the PLC.

using optocouplers. The desired output voltage can simply be connected up: it will generally be either 12 V or 24 V. IC3, a ULN2003, allows output voltages of up to 50 V, as long as the total output current does not exceed 500 mA.

Since it will be desired to switch greater loads than this, as well as mains voltages, relays are connected to the outputs of the power driver IC, each capable of switching up to 16 A at 250 V AC. This should be sufficient for almost all applications; and it is always possible to omit the relays if the drive of the ULN2003 is enough by itself.

The software

Any item of microprocessor-controlled hardware can only be as good as the software running on it, and the quality of that software depends in turn on the development tools used to write it. One of the main reasons for selecting the PIC16F84 was the availability of good-quality, low-cost (or even better, free) development tools. The manufacturer, in line with most of the competitors in the market, offers a suite of software development tools. MPLAB [4] from Microchip includes an assembler, a linker and a simulator, which allows the logical correctness of the software to be verified on a PC. Together, these tools allow a huge range of possible projects to be developed, as a glance at the Microchip website will clearly show.

If you would prefer to program the PIC in a high-level language, a suitable compiler will be required. Fortunately two good C compilers for the PIC are already available, namely C2C by Pavel Baranov [5] and CC5X from Knudsen Data [6]. The compilers differ slightly as regards processor-specific instructions, but otherwise are broadly as good as one another. Assembler source code is generated from a C source file; the result is automatically assembled and converted into a binary file (in Intel Hex format). Now we have just one problem.

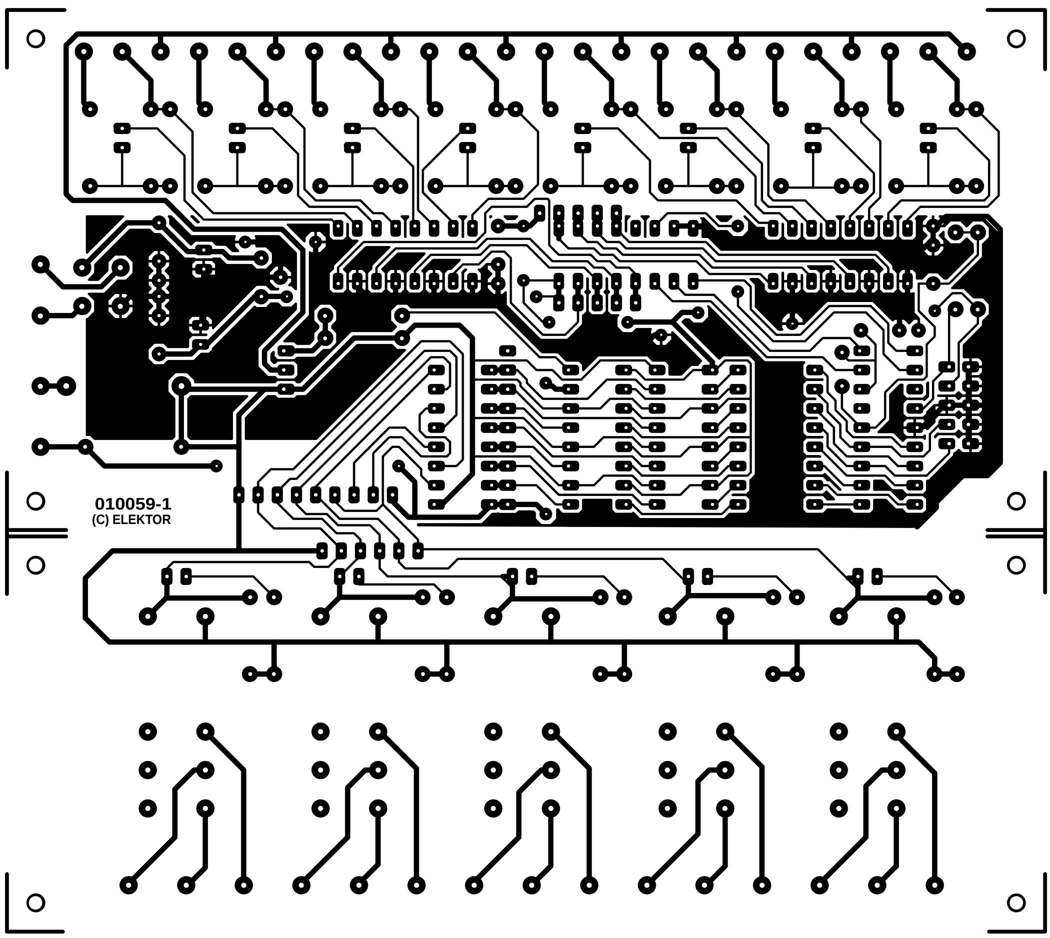
D9-D16,D20-D24 = LED. low current
D19 = 1N5407
IC1 = PIC16F84A-04/P
IC2 = 74LS151
IC3 = ULN2003
IC4 = 78L05AC
IC5-IC8 = LTV847 (Liteon), ILQ621 (Infineon) or PC847 (Sharp)

Miscellaneous:
K1 = 10-way boxheader
K2-K9 = 3-way PCB terminal block, lead pitch 5mm
K10-K14 = 3-way PCB terminal block, lead pitch 7.5mm
K15 = 9-way pinheader
K16 = 6-way pinheader
RE1-RE5 = relay, 16 A/250 VAC (Finder # 40.61, coil 12VDC, 220Ω; or Omron # G2R-1-E 12VDC; or Schrack # RP310012)
PCB, order code **010059-1** (see Readers Services page)
Disk, test program, order code 010059-11 or Free Download from www.elektor-electronics.co.uk

How do we get the program into the PIC?

The PIC microcontrollers are popular not least because of their ease of programming. Since PICs can be programmed serially the interface is relatively straightforward and in principle a simple cable is all that is required to do the job. But even a more luxurious programmer does not involve much hardware: for example, a classical 'low cost' development system was presented in the July/August 1998 edition of *Elektor Electronics*. The tiny evaluation board described there provided a stabilised power supply and a switchable and variable clock oscillator, as well as a small prototyping area, where for example LEDs could be fitted to observe the levels on the output ports.

An alternative programmer circuit board can be found fully documented in [3]. The necessary (DOS) program can also be obtained from the website.



Construction and installation

The construction of the Pico PLC on the circuit board shown in **Figure 2** should present no problems. Components are only fitted on one side, and all components are of the normal through-hole type rather than surface mount devices. If the relays are not required the circuit board can be cut between K15 and K16 and the PLC can then be built into a smaller enclosure. It is also possible to fit the part of the circuit board carrying the relays elsewhere in the enclosure, connecting it to the PLC via K15 and K16. If the relays are fitted, the safety precautions for class 2 devices must be observed at the outputs.

The ICs should of course be fitted in sockets and good-quality components should be used. To avoid loose connections, use tension clamp mounting terminals on the inputs and outputs.

No special software is provided for the Pico PLC though Readers' Services other than a short test program. The program was created using the free CC5X tools and MPLAB, and creates a kind of running light display from D20 to D24 and back — assuming the hardware is working correctly. Then D24 starts to blink, and the state of the inputs is displayed. While D24 is extinguished D20 to D23 show the states of the four lower-numbered input bits, while when D24 is lit, they show the states of the four higher-numbered inputs.

There are many applications for the Pico PLC. The original use was to control the drive mechanism for an amateur radio antenna mast. Other possibilities include controlling the shutters or Venetian blinds for a window, heating control or automating mechanical toys and models.

(010059)

References and links

[1] Microchip: www. microchip. com

[2] Datasheet available (for example) at www.sharpmeg.com/products/opto/pdf/pc847x.pdf

[3] Madsen programming adapter www.jdm.homepage.dk/newpic.htm

[4] MPLAB: development environment for Microchip microcontrollers www.microchip.com/1000/pline/tools/picmicro/devenv/mplabi/index.htm

[5] Baranov, Pavel: C2C-Compiler www.geocities.com/SiliconValley/Network/3656/c2c/c.html

[6] B Knudsen Data: CC5X www.bknd.com/cc5x/index.shtml