Text at the Speed o

A secondary school project was the starting point for this ingenious creation, which uses just eight blue LEDs to produce a scrolling text display with 360-degree all-round visibility. A more economical way of building a giant display device would be hard to imagine...

Steffen Sorge

The idea for constructing this display arose at a brainstorming session on our project for the school-leaving examination, what we call an 'Abi-Denkmal' (literally 'exam memorial' in Germany. This advanced level final exam involves an assignment to create something entirely original, a task in fact that at the author's school was traditionally undertaken by the electronics students. The original stimulus arose from bedside alarm clocks that function along similar lines. However, these inexpensive products from the Far East indicate only the time, not any text.

The first prototype was created in just one day using the following parts: eight LEDs from the junk box, a switch for synchronising the start of the message display, an AT90S2313, a PP3 (6F22/R222) 9-volt battery and a few other scrap components were mounted on a small wooden batten. The whole affair was then attached to the drive shaft of a small DC motor and set in motion. After a little programming effort were we rewarded with a display reading out the word 'Hallo'.
However, this initial prototype (just like in 'Modding & Tweaking' projects — *Ed.*) had a number of shortcomings:
The LEDs were not bright enough
The life expectancy of the switch was alarmingly short (little wonder: it operated with each revolution of the motor)

Having a rotating battery was not the cleverest idea...

Problems exist to be solved. The first task was to source some ultra-bright blue LEDS with an intensity of 3500 mCd. Next problem solved was the switch; we obtained a Hall effect switch from an old printer. The battery was replaced by a proper mains power supply and a slip ring. And finally the AT90S2313 made way for an ATmega8.

A controller for eight LEDs

As the circuit diagram (**Figure 1**) indicates, the functional circuitry comprises just a microcontroller (IC1), which drives the eight LEDs and is linked to the Hall sensor (IC3). As regards external components, the controller requires only a 16 MHz crystal (X1) with

f Light

rotating message display with LEDs and an AVR micro

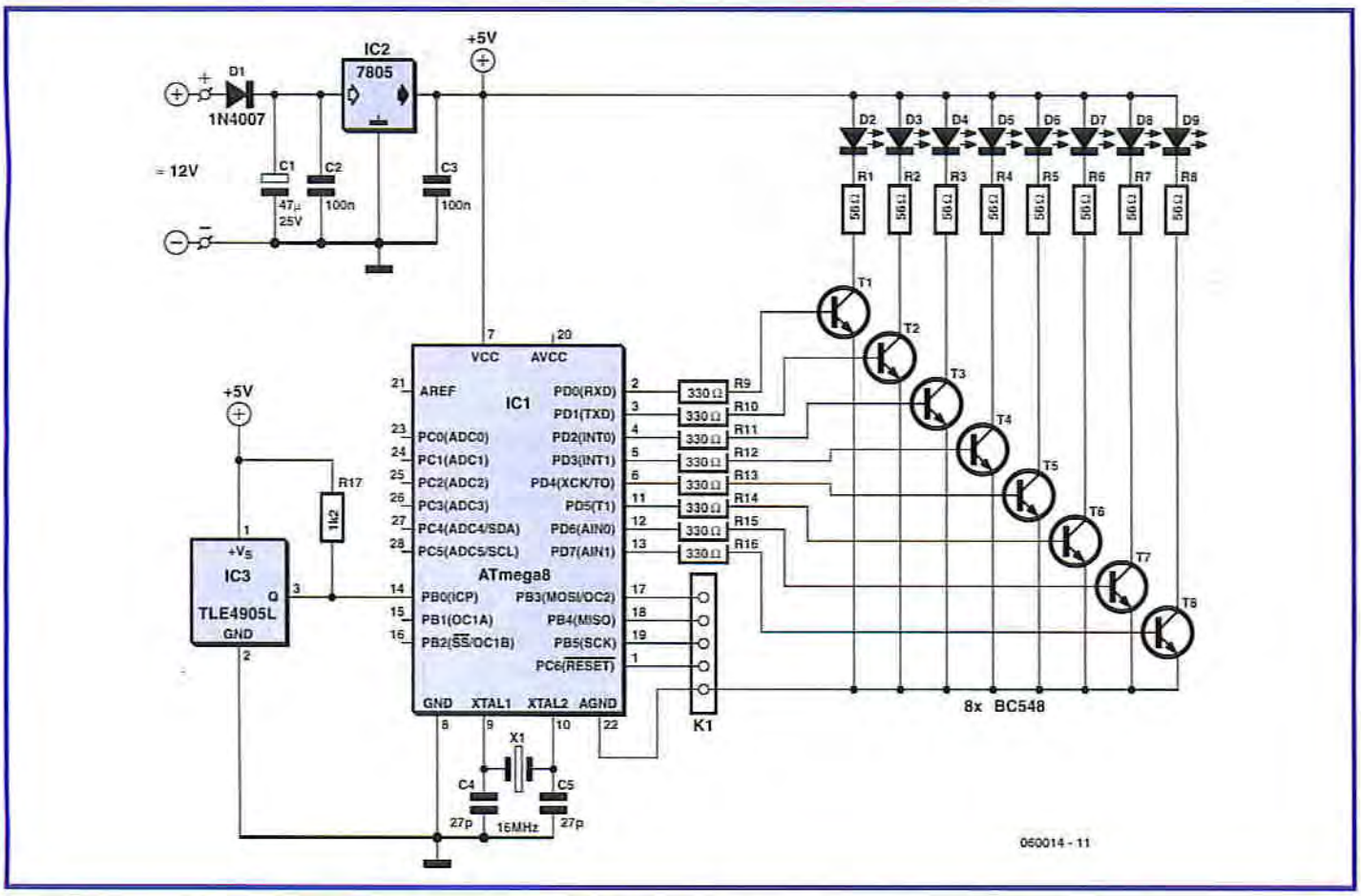

Figure 1. The circuit comprises fundamentally just an AVR controller linked to a Hall sensor (IC3) that switches the eight LEDs.

two resonating capacitors (C4 and C5). Voltage regulator is the trustworthy old 7805 (IC2).

As the microcontroller cannot source enough current for pulsing the hefty LEDs, a follower stage comprising transistors T1 to T8 is provided. The eight rotating LEDs 'write' an image built up from eight lines of light (not from left to right of course but in continuous 360-degree rotation) and for this reason the microcontroller needs to be synchronised with what in TV or computer displays we would call a frame pulse, a signal indicating the start of a new image. This task is handled by the Hall switch (IC3), which alters its output level when it detects a magnetic field (in the vicinity of a magnet in other words). Its normal state produces a 'high' output that goes 'low' on approaching a magnet. This change of level sets in motion the process of building up the image, which the software in the microcontroller looks after.

Software

The ATmega8 is an AVR controller by Atmel, belonging to the family of RISC controllers (with reduced control set). The program for the rolling text is not written in Assembler, however, but in the well-known BASCOM AVR-BASIC language.
This software, along with extremely very detailed annotations, can be downloaded freely from the Elektor Electronics website (data file 060014-11.zip).
The program produces small image elements representing text fragments relating to character strings stored in the controller. These text fragments are

What does AVR stand for?

We wondered too. Apparently Atmel says AVR is just a name and stands for nothing in particular. Another opinion states it's a play on the designers' names, an acronym for Alf (Egil Bogen) and Vegard (Wollan)'s RISC processor. The allegation that it stands for Advanced Virtual RISC sounds unlikely but you can take your choice!

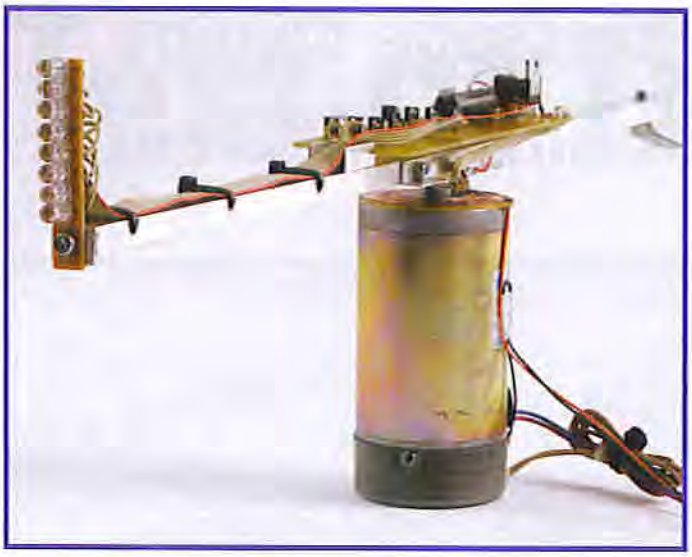

Figure 2. View of the rotating arm with the controller circuit board.

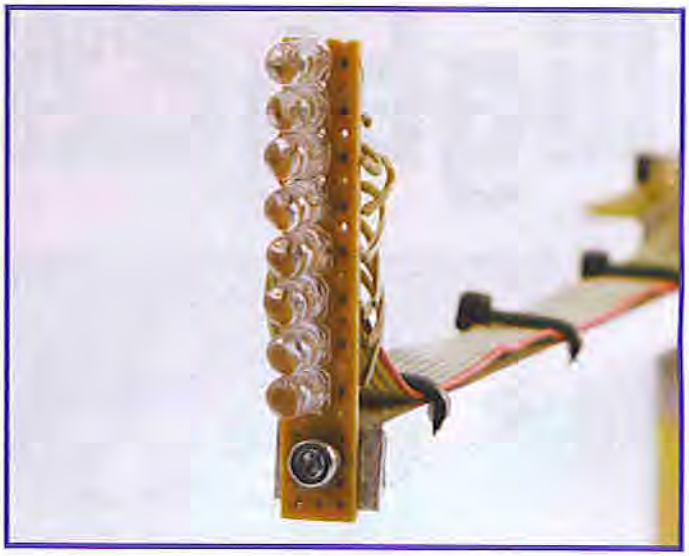

Figure 3. Configuration of the eight LEDs at the end of the rotor.

then re-written as columns and fed out to the LEDs in a defined time frame. The rotation of the motor creates the impression of text written around the periphery of a cylinder.
The controller can accept a maximum of 255 characters for the scrolling text. These characters are stored in the INSTRING array, which contains three elements (three segments of text) that are read out and displayed sequentially.
Readout of these characters is synchronised by the Hall switch already mentioned, which also provides the information whether the motor is in operation (when the motor is static the LEDs all remain unlit). The length of the rolling text in the 'display' depends on the radius of the circle created by the rotating LEDs, hence on the length of the rotor arm. The set-up illustrated in the photos operates with a text length of 30 characters. The program operates as an endless loop that is synchronised by the Hall switch and delivers 30 characters continuously.
The software produces the following characters: capital letters from A to Z (including the accented letters Ä, Ö and Ü), a null or space symbol, question mark, exclamation mark, full stop, comma and hyphen.
A subroutine is programmed for each character, making it straightforward to substitute or alter these symbols. A subroutine consists simply of multiple write commands to Port D, which controls the LEDs. Each write command is followed by a brief pause created by a Wait loop.

Each Write command creates one column of the relevant character, which is composed of several of these columns each having eight pixels vertically. This makes it easy to design a character on squared paper and transpose the columns from left to right into Bytes. If you visualise the Wait loop (subroutine 'Wa') in the source code, it is easy to work out how a character will actually. For example:

```
Sub A
Portd = &B11000000
Portd = &B00110000
Portd = &B00101110
Portd = &B00100001
Portd = &B00101110
Portd = &B00110000
Portd = &B11000000
Portd = &B00000000
Wa
Wa
End Sub
```

The last write command always has a zero value in order to extinguish the LEDs. The wait duration of the subroutine 'Wa' must be configured according to the rotational speed of the motor (just test out various values to determine the best one for you).
Playout begins with the first 30 characters of the first text segment held in INSTRING. After four revolutions of the motor (monitored by the variable 'Laufer1') the text is incremented and displaced by one character (the start position of the text is monitored in 'Laufer'). Once the end of the first text segment is reached, the next segment is selected (variable 'Strz'). At the end of the third text segment the display reverts to the first segment of text and begins again.

If you significantly vary parameters such as rotational speed and diameter of the rotating LEDs, then a few alterations need to be made to the program: Using a different radius (rotor arm length) the number of characters visible in the 'display' (diverging from 30) must be adjusted (at two locations in the program).
Alteration to the number of revolutions per minute requires adjustment of the pixel length (subroutine 'Wa').
The text readout speed (the rate at which the text scrolls) can be varied.
The number of text segments can be increased (this requires a controller with greater memory storage, however).

Circuit construction

The circuitry is relatively easy to construct by following the circuit diagram. As the DIL version of the ATmega8 is used (ATmega8-16PU), there are no surface-mounted devices or other problematic issues to contend with. Resistors R9 to R16 must be selected according to the nominal current of the LEDs used. They can be calculated simply as series resistors for 5 V. The sample set-up used a value of 56 Ω.

To assist construction you can download the author's circuit board layout and component plan together with the component list from the *Elektor Elec-*

Figure 4. A slip ring provides the power supply to the rotational circuitry.

tronics website (January 2007 items). You do, however, need to note that the designations used on the component plan may either vary from those shown in the circuit diagram (Figure 1) or else may not be shown at all. However, the circuit board does correspond with the diagram electrically and can always be cross-checked in case of doubt.

A rectangular format was selected for the circuit board, matching the dimensions of the rotor arm, with a separate smaller board provided at the end of the arm for the LEDs mounted one above the other (see photos). To make the 5-mm LEDs align with the hole spacing of normal 2.54 mm-spaced perf-board, the LED bodies need to be flattened a little with a file on two opposite sides (don't overdo this). The LED board is connected to the other board using a length of 9-conductor flat ribbon cable. The anodes of the LEDs are connected with one another and to +5 V, whilst the cathodes are wired to the series resistors provided (R9 to R16). D2 (linked to R1) is the uppermost LED on the board and D9 (linked to R8) is the bottom LED. If the LED array is wired the wrong way round (D9 on top, D2 below), the text will appear upside down!

Programming

The free download **060014-11.zip** contains .bas, .hex and .bin files. The .bas file holds the source code in 'Bascom' format. The .hex or .bin data can be flashed direct into the ATMega8 and is then ready to run. The data supplied will produce scrolling text of the demonstration message programmed by the author.
If you would like to program your own text you will need Bascom. For this open the file Schrift.bas and change the contents of the array INSTRING. All that remains to do is press F7 to compile the new source code.
After connecting the operating voltage (9 to 12 V) the controller can be programmed. To do this you link the programming connections (K1 on the circuit diagram) via a suitable programmer (PonyProg for example) with the PC and flash the file Schrift.bin or Schrift.hex into the chip (set fusebits externally to 16 MHz).

If you would like to try compiling with the free BASCOM-AVR DEMO 1.11.8.3 software you will need to restrict the length of your text and avoid filling the INSTRING array with three texts, since the demonstration version will not compile more than 4 kB of code.

Mechanical matters

For the 'rotor' an arm made of sheet aluminium about 50 cm long was used. At the centre of the arm you need to fix a small bush that needs to be connected with good electrical conductivity to the drive shaft of the motor. The return wire is screwed or otherwise clamped to the casing of the motor. A slip ring of clean printed circuit board material (not the sort that is pre-coated with photosensitive lacquer) is fixed to the upper surface of the motor and wired up to the positive (plus) side of the voltage source. Above the slip ring a small carbon brush is fixed on the rotor arm in an insulated mounting to avoid contact with the aluminium sheet. Finally a magnet is fixed to the motor casing in such a position that the Hall sensor mounted on the rotor arm passes through its magnetic field at each revolution and is thus activated.

The circuit board is now fixed to the rotor arm (ideally without putting it out of balance). Power supply arrangements involve wiring the return (ground) connection on the circuit board to the bush on the drive shaft and the positive connection to the carbon brush. If the arm remains unbalanced this can be corrected by using compensating weights.
The power source (a 9 to 12 V plug-in power supply) can now be connected up to the slip ring (positive connection) and the motor casing (ground return). The current drawn by the circuit is set mainly by the total draw of the LEDs, to which you must add that of the motor if the same power supply is used for both.

As soon as the power is connected and the arm rotates scrolling text should be visible on the 'display'—just like in the photo at the beginning of the article...

(060014-I)

PS

The author was successful not just with this special assignment but also with his final exams, and is now studying Electronics and Information Technology at the West Coast Technical College in Heide, Germany.