Microprocessors are regular devils for work. In many cases, this involves gathering data, performing calculations and finally making the results visible. For this last, often very important task, a display is necessary. The circuit presented here shows that only a few components are needed to make an inexpensive display that has high contrast and is thus easy to read.

Design by I. Gerlach

versatile LED display

for use with many microprocessor systems

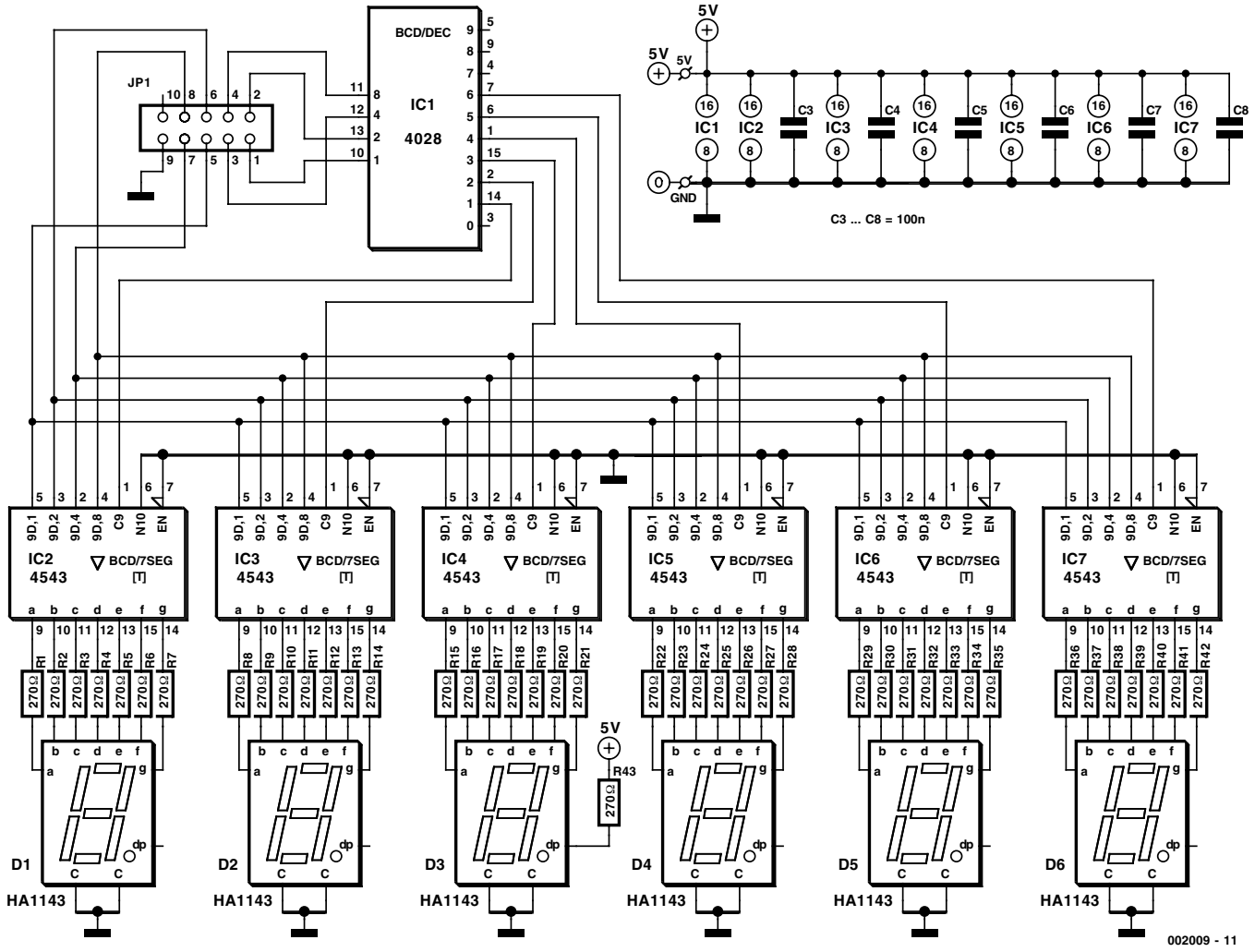

Figure 1. The schematic diagram of the LED display. Only standard components are used. The design of this display makes it very flexible, without imposing a heavy load on the processor.

Most microprocessor systems are difficult to use without a good display. Microprocessors are after all black boxes that can quickly perform a whole lot of computations, totally hidden from the outside world. Since computation is not an end in itself, and the user wants to see clear conclusions, many projects need displays. Liquid crystal (LC) displays are used more and more often for this purpose, for evident reasons:

- both letters and numbers can be displayed on a LC display,
- a built-in LCD controller makes the connection simple and minimises the load on the processor,
- LC displays use little energy,
- the displays are compact.

In addition to their advantages, LC displays naturally have disadvantages:

- they emit no light and thus need a backlight under conditions of poor illumination,
- they have a limited viewing angle,
- they can only be used over a limited temperature range,
- they have relatively low contrast,
- they are relatively expensive,
- different software may be required, depending on the type of display used,
- the dimensions of the display are limited,
- the useful lifetime is limited.

In summary, an LC display is sometimes a good choice, but in other cases an LED display may be a better option. In this article, we show that a six-character LED display can be built at a relatively low cost. There were several primary considerations in the development of this idea. In particular, the display must:

- use relatively few I/O lines,
- be inexpensive to build,
- require relatively little computing time from the processor,
- be flexible and thus extensible,
- be usable with practically every type of microprocessor.

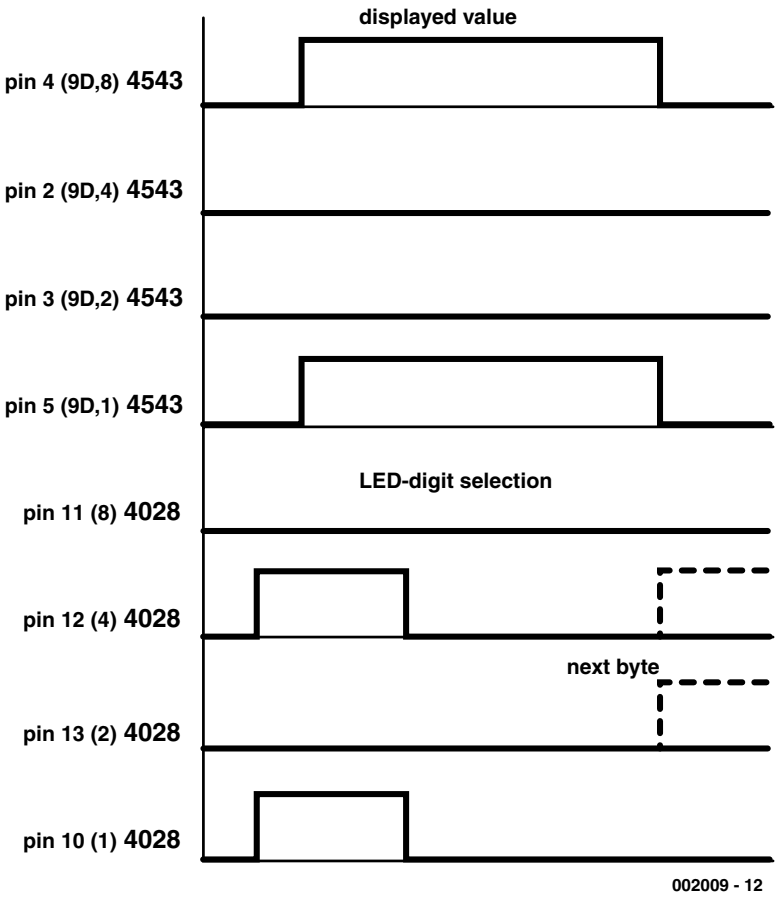

Figure 2. Timing diagram of the drive signals. The upper four signals determine the code for the display (9 in this case), while the lower four signals select the display IC (5 in this case).

The hardware

Figure 1 shows the schematic diagram of the display circuit. Only two types of IC are used in this project, both of which are inexpensively available. IC1 is a 4028, which is a simple demultiplexer. The remaining six ICs are 4543 types. This is a BCD decoder with a built-in latch and display driver.
The display components are seven-segment display ICs with common cathodes. This means that the CC pins are connected to earth. The remaining pins (a, b, … g) are directly connected to the corresponding outputs of their associated display controllers (IC2 through IC7). Finally, each display IC has one other pin, labelled 'dp'. This pin drives the decimal point of the display.
In this design, only the decimal point of the third display IC is activated, by connecting it to +5 V via a 270 Ω resistor (R43). If you want to have the decimal point displayed at some other location, it is easy to change this arrangement.
The entire display is connected to the microprocessor system via a header with two rows of five pins. Only nine of the ten pins are actually used. Since one of the pins is connected to earth, only eight processor I/O lines are needed to drive the display.
A 4-bit signal applied to pins 1 through 4 selects the desired display IC. The bit pattern on these four lines determines which output of IC1 is high. The pattern '0000' is not used, since no display IC is connected to

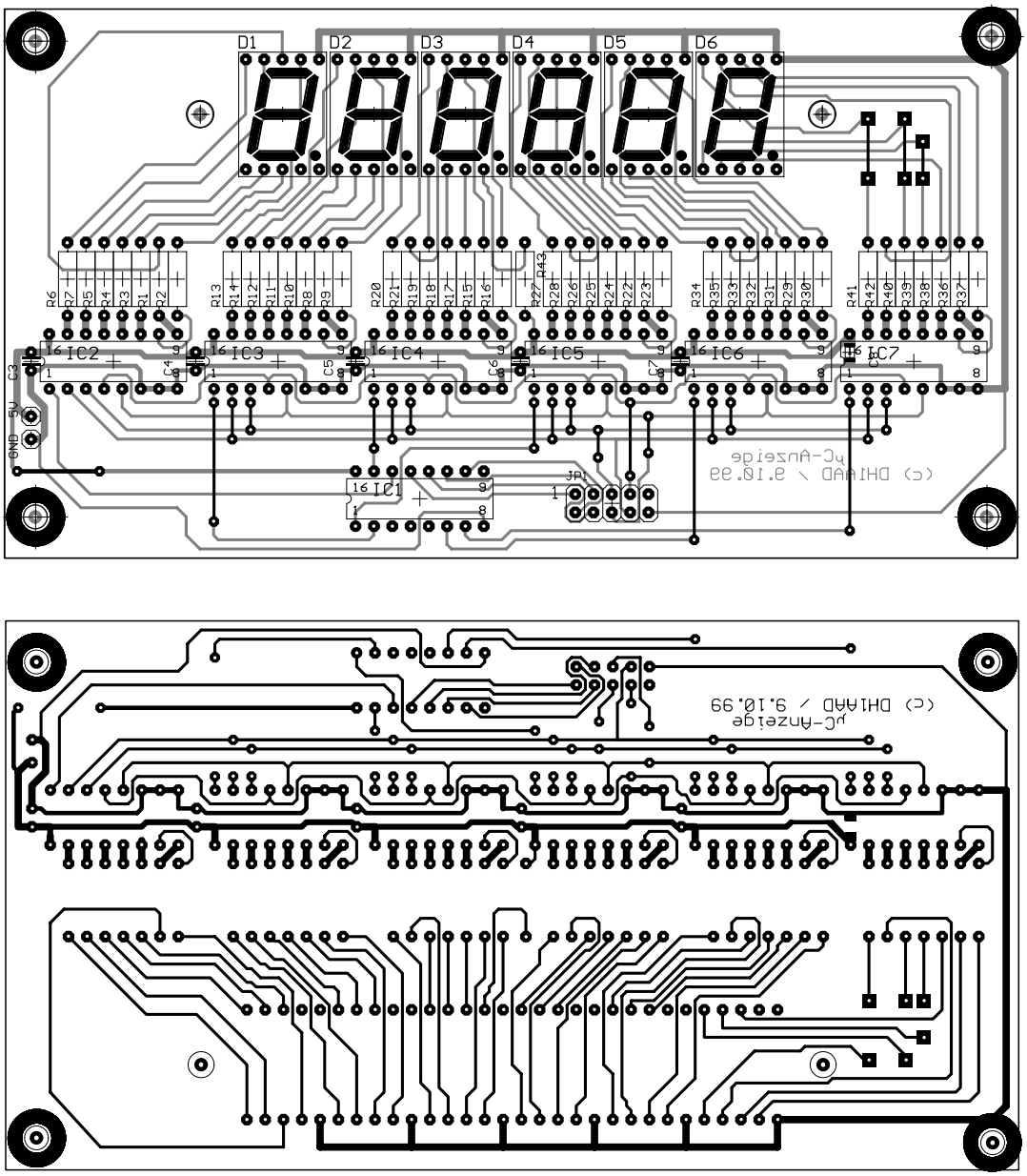

Figure 3. The copper track layout and component layout of a printed circuit board that can be used to build the LED display.

pin 3 (Q0) of IC1. The other four signal lines (pins 5 through 8) are used to generate the BCD codes that drive the display. As soon as the proper BCD code has been applied to these lines, the load input (LD, pin 1) of the controller for the desired display IC is set high. On the falling edge (from high to low) of this signal, the 4543 stores the code present on its inputs 1A through 1D (pins 2 through 5). The pattern of output pins (AS through G) that go high for a particular input code is determined by a matrix integrated into the decoder IC. The associated segments of the display IC are then illuminated.

The whole process is illustrated in the timing diagram shown in **Figure 2**. The four upper bits show that a BCD code of '1001' is applied to the inputs of the display controllers. After this, IC5 is activated by the 4028 decoding the selection code "0101' (decimal 5).

The approach taken here has two important advantages for the user. First, as long as the displayed value does not have to be changed, no new control signals have to be sent the display. Second, it is possible to modify the value displayed by a single display IC. These two features mean that the load on the processor is kept to a minimum.

There is little to say about the rest of the circuit. A simple 5-volt power supply is used. Each IC has a decoupling capacitor. The series resistors for the display segments are well chosen for the supply voltage used. If you want to make the display brighter, the value of these resistors (R1 through R43) can be reduced, for example to 180 Ω. If on the other hand you want less brightness, in the interest of reduced energy consumption, you can increase the value of these resistors.

Printed circuit board

In order to simplify the construction of the display, the author has designed a printed circuit board that is shown in **Figure 3**. This board is not available from Readers Services, so you will have to make it yourself. This should not be difficult, since the layout is simple and

the board is single-sided.
When assembling the board, start with the wire bridges. After this you can insert the remaining components. You should preferably use IC sockets, and solder carefully. For IC2 through IC7, the last resistor position always has no component number. This is the resistor for driving the decimal point (R43 in the schematic diagram). First decide if you want to have a decimal point, and then decide where it should be located. After this, all you have to do to activate the decimal point is to solder a 270-Ω series resistor in the proper location.

Sample software

In order to use the display with a microcontroller, you will naturally need some software. Consequently, we have included a sample program in the form of a machine-language routine that is suitable for Atmel AT90S1200 through AT90S8515 processors. We have chosen to use a separate driver to keep the application as simple as possible. This makes a flexible approach possible, and this driver can be called from an application. This means that the control software only has to be written once, after which it can be used in a variety of systems. **Listing 1** shows the design of the driver. Its has a quite straightforward construction, which

```
;*************************************************
;* File Name    :LEDDisp.inc
;* Title              :Driver for LED Display
;* Date         :Ingo Gerlach / 10.10.99
;* Version            :1.0 / 11.10.99
;* Version            :
;* µC                 :AT90S1200...8515
;*                    :
;* Changes            :
;*
;*************************************************
;
;
; Main program register variables
;-----------------------------------------
;.def   temp    = r16
; Registers  / LED
;-----------------------------------------
;.def   cntr    = r20   ; counter
;.def   dly     = r21   ; delay loop variable
;.def   pos     = r23   ; position
;.def   byte    = r24   ; byte

; Equates
;-----------------------------------------
;.equ   LED_qty = 6             ; number of LEDs
;.equ   LED_Del = 45            ; delay
;.equ   OutPort = PortB

; Functions
; LED_Blank     : switch display on
; LED_Null      : reset display 0 ( Null)
; LED_Show      : show bytes , transport byte (R24),
position (R23)

; **** Switch display off
;*************************
LED_Blank:      ldi     cntr,LED_QTY
LedLoop1:       ldi     temp,192
                add     temp,cntr
                out     OutPort,temp
                Rcall   Led_Delay
                dec     cntr
                brne    LedLoop1
                Ret

;*************************
; **** Reset display
;*************************
LED_Null: ldi   cntr,LED_QTY    ; load number of LEDs
LedLoop2: out   OutPort,cntr
                Rcall   Led_Delay
                dec     cntr
                brne    LedLoop2
                Out     OutPort,cntr
                Ret
;*************************
; **** Show byte
;*************************
LED_Show:   mov   temp,pos ; position in register
                out       OutPort,temp; activate BCD-
to-decimal decoder, LD 4543
                Rcall     Led_Delay ; short delay
                mov       temp,byte ; value in
                                      register
                swap      temp      ; value high
                                      nibble
                add       temp,pos  ; goto position
                out       OutPort,temp; 4028 / 4543
                          move value to display
                Rcall     Led_Delay ; short delay
                sub       temp,pos
                out       OutPort,temp; LD signal
off, store value
                Rcall     Led_Delay
                Ret
;*************************
; * Internal functions !!!

; *** Delay LED display
;**********************
LED_Delay:      ldi dly,LED_Del
LedLoop:        dec dly
                brne LEDLoop
                ret
```

Listing 1. A machine-language driver for the LED display. This can be integrated into various applications.

makes its operation easy to follow.
Listing 2 contains a short demonstration program. This program first sets the display to all zeros and then displays the number '145675'. If the decimal point is set after the third segment, the value to be read from the display is thus 145.675.
You can easily experiment with this flexible LED display. If the code has to be translated for a different processor family, this can be easily done based on the information provided in the listings.

(002009-1)

Text (Dutch original): H. Steeman

```
;***************************************************
;* File Name    :LED.asm
;* Title              :Test program for LED display
;* Date         :Ingo Gerlach / 10.10.99
;* Version            :1.0 / 10.10.99
;* Version            :
;* µC                 :AT90S1200...8515
;*                    :
;* Changes :
;*
;***************************************************
;

;***** Directives


.device AT90S1200              ;device type
.NOLIST

.include "1200def.inc"


.list
.listmac



; Show data
; Structure of data

;       MSB                LSB
;       7 6 5 4        3 2 1 0
;       0 0 0 0        0 0 0 0
;
;       Select position 1 = 1. 2 = 2. etc
;       Number in BCD code
;
;e.g.   10010010b = 146d = pos. 2, value 9


;
; Main program register variables
;-----------------------------------------
.def    temp           = r16


; Registers  / LED
;-----------------------------------------
.def    cntr           = r20   ; counter
.def    dly            = r21   ; delay loop variable
.def    pos            = r23   ; position
.def    byte           = r24   ; byte


; Equates
;-----------------------------------------
.equ    LED_qty = 6            ; number of LEDs
.equ    LED_Del = 40           ; delay
.equ    OutPort = PortB


;***** Interrupt vector table

reset:
    rjmp  main  ; main routine
    reti        ; external interrupt0 handle
    reti        ; T/C0 overflow interrupt handle
    reti        ; analogue comparator interrupt handle

;***** Functions
***************************************************


;****************************************
;***** Main ******************************

main:
;   ldi   temp, LOW(RAMEND)    ; setup StackPointer
                                 for > 90S1200
;   out   SPL, temp            ; initialize SPL
;   ldi   temp, HIGH(RAMEND)
;   out   SPH, temp            ; initialize SPH


    ldi   temp,255             ; temp = 255
    out   ddrb,temp            ; port B output

    Rcall LED_Null             ; reset display

mainloop:

; Show  145.675

                ldi     Pos,1
                ldi     Byte,1
                Rcall   Led_Show

                ldi     Pos,2
                ldi     Byte,4
                Rcall   Led_Show


                ldi     Pos,3
                ldi     Byte,5
                Rcall   Led_Show

                ldi     Pos,4
                ldi     Byte,6
                Rcall   Led_Show

                ldi     Pos,5
                ldi     Byte,7
                Rcall   Led_Show

                ldi     Pos,6
                ldi     Byte,5
                Rcall   Led_Show


forever:        rjmp forever

; ***** End of main program ******************

; ***   Include Files  ***
.include "LEDDisp.inc"
```

Listing 2. A demonstration program that shows how the driver can be integrated with an application.