MICROCONTROLLER-BASED HEART-RATE METER

■ PROF. K. PADMANABHAN

Heart rate can be measured either by the ECG waveform or by the blood flow into the finger (pulse method). The pulse method is simple and convenient. When blood flows during the systolic stroke of the heart into the body parts, the finger gets its blood via the radial artery on the arm. The blood flow into the finger can be sensed photoelectrically.

To count the heart beats, here we use a small light source on one side of the finger (thumb) and observe the change in light intensity on the other side. The blood flow causes variation in light intensity reaching the light-dependent resistor (LDR), which results in change in signal strength due to change in the resistance of the LDR.

Circuit description

Fig. 1 shows the circuit of microcontroller-based heart-rate meter. The setup uses a 6V electric bulb for light illumination of flesh on the thumb behind the nail and the LDR as detector of change in the light intensity due to the flow of blood. The photo-current is converted into voltage and amplified by operational amplifier IC LM358 (IC1). The detected signal is given to the non-inverting input (pin 3) and its output is fed to another non-inverting input (pin 5) for squaring and amplification. Output pin 7 provides detected heartbeats to pin 12 of the microcontroller. Preset VR1 is used for sensitivity and preset VR2 for trigger-level settings.

Microcontroller IC AT89C2051 (IC2) is at the heart of the circuit. It is a 20-pin, 8-bit microcontroller with 2 kB of Flash programmable and erasable read-only memory (PEROM), 128 bytes of RAM, 15 input/output (I/O) lines, two 16-bit timer/counters, a five-vector two-level interrupt architecture, a full-duplex serial port, a precision analogue comparator, on-chip oscillator and clock circuitry.

Port-1 pins P1.7 through P1.2, and port-3 pin P3.7 are connected to input pins 1 through 7 of IC ULN2003 (IC3), respectively. These pins are pulled-up with 10-kilo-ohm resistor network RNW1. They drive all the segments of the 7-segment display with the help of inverting buffer IC3.

The displays are selected through port pins P3.0, P3.1 and P3.2 of the microcontroller (IC2). Port pins P3.0 down through P3.2 are connected to the base of transistors T3 through T1, respectively. Pin 6 of IC2 goes low to drive transistor T1 into saturation and provide supply to the common-anode pin (either pin 3 or pin 8) of DIS1. Similarly, transistors T2 and T3 drive common-anode pin 3 or 8 of 7-segment displays DIS2 and DIS3, respectively. Only three 7-segment displays are used.

IC2 provides segment-data and display-enable signals simultaneously in time-division-multiplexed mode for displaying a particular number on the 7-segment display unit. Segment-data and display-enable pulses for the display are refreshed every 5 ms. Thus the display appears to be continuous, even though it lights up one by one.

Switch S2 is used to manually reset the microcontroller, while the power-on reset signal for the microcontroller is derived from the combination of capacitor C4 and resistor R8. An 11.0592MHz crystal is used to generate the basic clock frequency for the microcontroller. The circuit is powered by a 6V battery.

Port pin P3.6 of the microcontroller is internally available for software checking. This pin is actually the output of the internal analogue comparator, which is available internally for comparing the two analogue levels at pins 12 and 13. As pins 12 and 13 of IC2 can work as an analogue comparator, these are used for sensing the rise and fall of the pulse waveform and thereby evaluate the time between two peaks and hence the beat rate.

The output of the pulse pick-up preamplifier is fed to pin 12 of the microcontroller. Pin 13 of the microcontroller is connected to the preset for reference-level setting of the comparator. Thus voltages at pins 12 and 13 are always compared. The signal rise and the fall at pin 12 are sensed

PARTS LIST

Semiconductors:	
IC1	- LM358 operational amplifier
IC2	- AT89C2051 microcontroller
IC3	- ULN2003 current buffer
T1-T3	- BC557 pnp transistor
D1	- 1N4007 rectifier diode
DIS1-DIS3	- LTS542 common-anode, 7-segment display
LED1, LED2	- 5mm LED
Resistors (all ¼-watt, ±5% carbon):	
R1, R8	- 10-kilo-ohm
R2	- 47-kilo-ohm
R3	- 100-kilo-ohm
R4, R5	- 1-kilo-ohm
R6, R7	- 330-ohm
R9-R11	- 1.2-kilo-ohm
RNW1	- 10-kilo-ohm resistor network
Capacitors:	
C1	- 470nF ceramic disk
C2, C5, C8	- 0.1µF ceramic disk
C3, C9	- 470µF, 16V electrolytic
C4	- 10µF, 16V electrolytic
C6, C7	- 22pF ceramic disk
Miscellaneous:	
S1, S3	- On/Off switch
S2	- Tactile switch
X_{TAL}	- 11.0592MHz crystal
BATT1, BATT2	- 6V battery

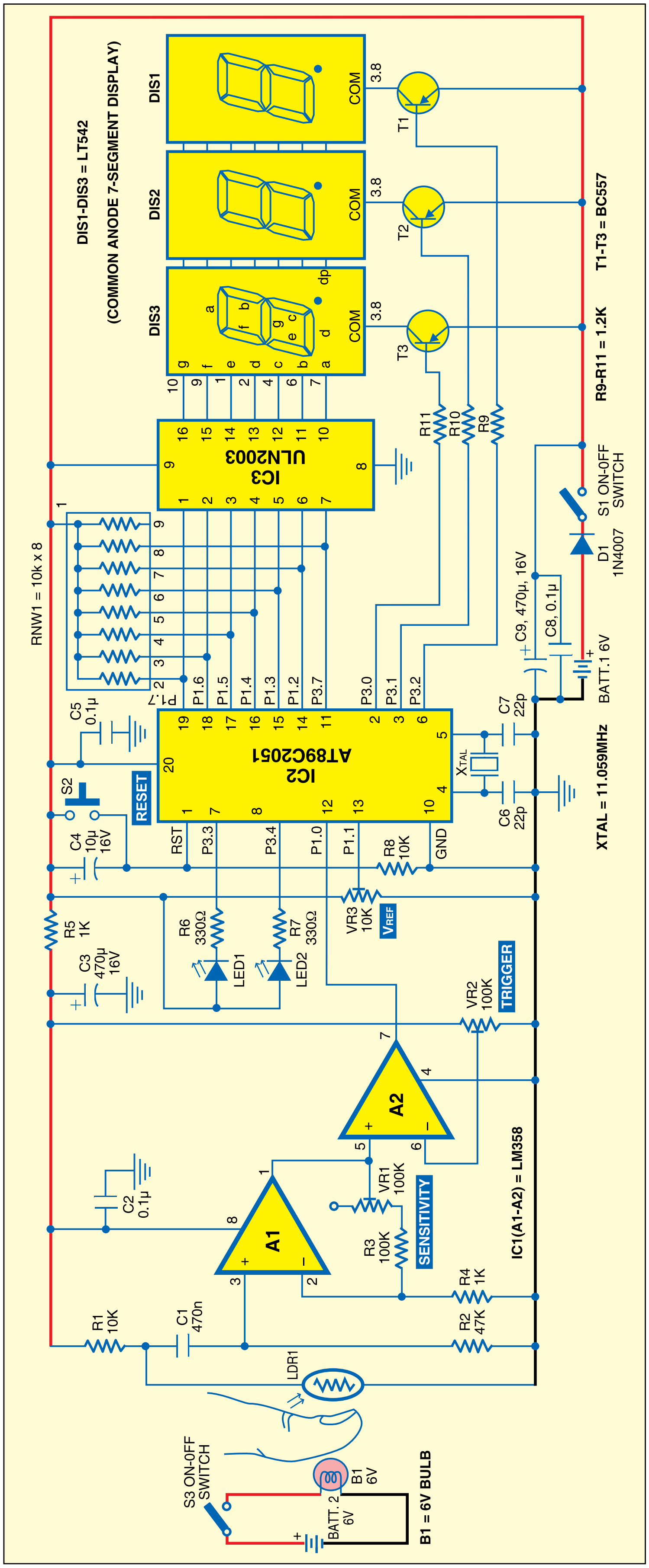

Fig. 1: Circuit diagram of microcontroller-based heart rate meter

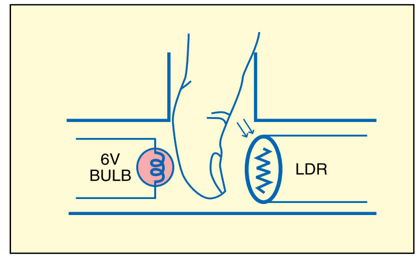

Fig. 2: 'T' tube with finger inserted

by the program.

The internal timer of the microcontroller is used to find the time taken for one wavelength. This time is converted into the heart beat rate in beats per minute by a pre-calculated look-up table. The program notes the time between the high-to-low and low-to-high transitions of the wave. This time in microseconds is converted in steps of 4 ms for comparison with the values already stored in the look-up table. This number is used to find (from the look-up table) the heart rate in beats per minute. The number so obtained is converted into a 3-digit number in binary-coded decimal (BCD) form. The same is output to the 7-segment LED displays in a multiplexed manner. The display shows the rate for a while and proceeds to another measurement. Thus beat rates obtained from time to time are visible on the display.

Construction and testing

The arrangement for heart beat rate detection is shown in Fig. 2. Purchase a plastic 'T' tube from an electrical parts shop. The tube should be about 5cm long and have a diameter of 1.5 cm. House the electric bulb into the left tube and the LDR (soldered on a small PCB) into the right tube. Fit shields on both sides of the tube to maintain darkness for better performance. Connect the 6V battery supply to the bulb and the LDR to the circuit board via a shielded cable.

For heart beat detection, which can be seen on a cathode ray oscilloscope (CRO), insert your thumb with the nail facing the LDR inside the T-tube. Shaking the thumb will change the level of signal from the previous

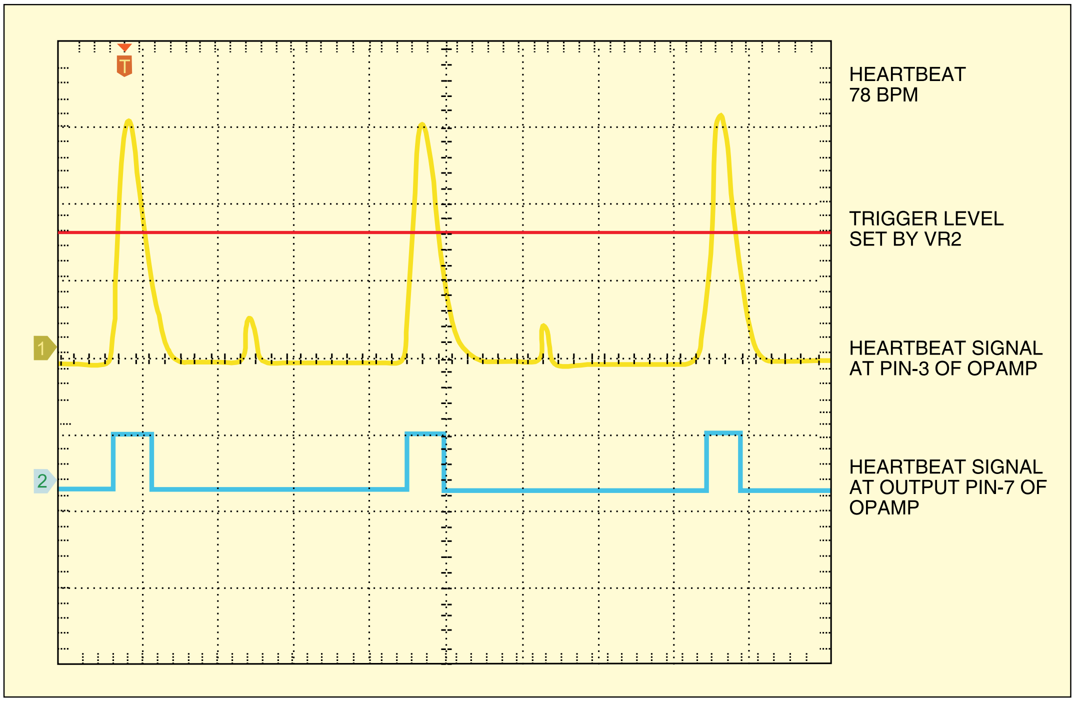

Fig. 3: Waveform of heartbeat detection

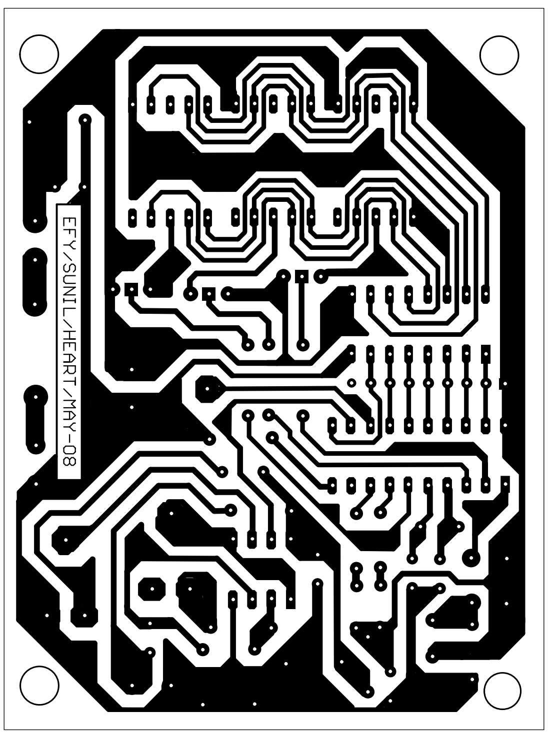

Fig. 4: A single-side, actual-size PCB layout for microcontroller-based heart-rate meter

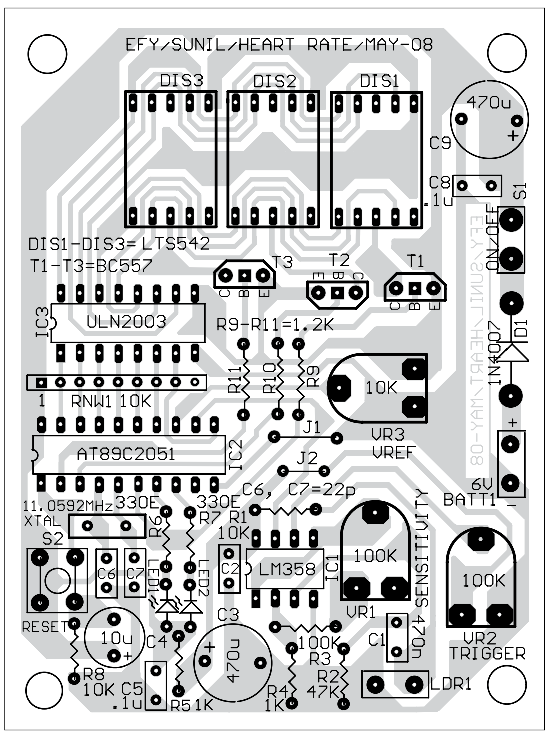

Fig. 5: Component layout for the PCB

value, and it will keep oscillating. Therefore you have to hold the thumb firmly between the light bulb and the LDR while the measurement is being made.

Place the circuit components and IC bases on the PCB board. Check the pulse pick-up through the CRO at output pin 7 of IC1 (refer Fig. 3). Insert the programmed microcontroller and other ICs into the IC bases. Set the levels of sensitivity, trigger and voltage reference for the comparator by using presets VR1, VR2 and VR3, respectively.

Hold the thumb steady and observe the heart beat rate on the display. The rate may vary and may not be exactly steady. For instance, normally, the rate can vary between 60 and 100.

Since this is a beat-to-beat measurement and not an average over a time period of one minute, variation is expected. However, when the reading shows high value at times, say, 140, it may be due to unusual mains hum picked up by the transducer. To suppress it, place a separate capacitor of 100 µF across the 5V supply.

An actual-size, single-side PCB for the microcontroller-based heart-rate meter is shown in Fig. 4 and its component layout in Fig. 5.

Software

The software is written in Assembly language and assembled using ASM51 cross-assembler. The Intel hex code is generated and burnt into the microcontroller chip by using a suitable programmer. The software is well commented and easy to understand.

The timer does the job of finding the time between two successive pulse waveform points. Since the comparator within the microcontroller IC knows the point of crossing of the wave with the DC line determined by preset VR3, the three crossings follow one after another and at the end of the third crossing the time is read from the time-count register. This time is then converted in terms of the number of 4ms intervals. From the number of such 4ms units, the number of beats per minute is determined from the look-up table already stored in the same memory starting from the label 'table' in the program listing.

EFY note. The source code and other relevant files of this article are included in this month's EFY-CD.

HEARTRT.ASM

```
$mod51
ORG 0H
         AJMP 30H
ORG 0BH            ;TIMER  0  INTERRUPT
VECTOR
         AJMP TIM0ISR       ;Timer 0 In-
terrupt service routine address
ORG 30H
         MOV SP,#60H        ;set  stack
pointer
         MOV P3,#0FFH       ;set   all
port 3 bits high to enable inputs also
         MOV P1,#03         ;set port 1
to all zeros expect bits 0,1
         MOV TMOD,#01100001B
;TIMER 1 - MODE 2 COUNTER,TIMR-0 TO MODE
1
BEG:     MOV TH0,#0f0H         ;TIMER
REG.0 IS SET TO foo0, GIVES 4ms
         MOV TL0,#0         ; timer low
reg. is also so
         mov r6,#255
         clr 20h            ;  flag  to
know time between beats exceeded
         mov r2,#0
         setb et0
         setb ea
PULSECHK:
         jb p3.6,$          ;  look  for
pulse at lowlevel _____
                  call delay2
                  jnb p3.6,$
;look for pulse high ----
                  setb tr0
;yes, pulse gone up, start timer
         call delay2
back1:            jb p3.6,$
; let waveform go low ____
                  call delay2
                  jnb p3.6,$         ;
look for next pulse high -----
                  clr tr0 ; stop timer
                  mov a,r2
                  cjne r2,#0,brady   ;
too low rate! brady-cardia
read_time:  mov a,r6
                  cpl a
                  mov dptr,#table    ;
table for rate calculated and kept
                           ; read value
in R6 which gives in steps of 4ms
                  clr c
                  subb a,#80
                  jc tachy
;rate too fast so tachy-cardia
lookup:           mov a,r6
                  cpl a
                  movc A, @a+dptr    ;
table looked up
                  MOV R2,A           ;
rate is now in r2
                  MOV R1,#0          ;
high byte is zero
                  call hex2bcd       ;
make it in BCD format
                  call disp1         ;
show the value on LED
                  mov 50h,#100       ;
refresh a 100 times (.5 sec)
REFR:                  CALL REFRESH1
                       djnz 50h,REFR      ;
so many times
clrint:                clr et0
                       clr ea             ;no
more interrupts
                       jmp beg
tachy:                 clr p3.4
; to show    on LED pin 8 that  rate is
too high
                       jmp beg
brady:
                  clr p3.3           ;  show  too
low beat at p3.3 LED
                       JMP beg
;16 Bit Hex to BCD Conversion for 8051
Microcontroller
 ;This routine is for 16 bit Hex to BCD
conversion;;;;;;;;;;;;;;;;;;;;
;Accepts a 16 bit binary number in R1,R2
and returns 5 digit BCD in
;R7,R6,R5,R4,R3(upto 64K )
 Hex2BCD: ;r1=high byte
         ;r7 most significant digit
         ;R2 = LSByte
         MOV R3,#00D
         MOV R4,#00D
         MOV R5,#00D
         MOV R6,#00D
         MOV R7,#00D
         MOV B,#10D
         MOV A,R2
         DIV AB
         MOV R3,B
         MOV B,#10          ;
R7,R6,R5,R4,R3
         DIV AB
         MOV R4,B
         MOV R5,A
         CJNE  R1,#0H,HIGH_BYTE  ; CHECK
FOR HIGH BYTE
         SJMP ENDD
         HIGH_BYTE: MOV A,#6
         ADD A,R3
         MOV B,#10
         DIV AB
         MOV R3,B
         ADD A,#5
         ADD A,R4
         MOV B,#10
         DIV AB
         MOV R4,B
         ADD A,#2
         ADD A,R5
         MOV B,#10
         DIV AB
         MOV R5,B
         CJNE R6,#00D,ADD_IT
         SJMP CONTINUE
         ADD_IT: ADD A,R6
         CONTINUE: MOV R6,A
         DJNZ R1,HIGH_BYTE
         MOV B, #10D
         MOV A,R6
         DIV AB
         MOV R6,B
         MOV R7,A
         ENDD: ret
DISP1:
REFRESH: ; content of 18 to 1B memory
locations are output on LEDs
         ;only numbers 0 to 9 and A to
F are valid data in these locations
                  MOV 18H,r3         ;
least significant digit
                  MOV 19H,r4         ;
next significant digit
                  MOV 1AH,r5
                  MOV 1BH,R6         ;
most
                                     ;
significant digit   (max:9999)
refresh1:         MOV R0,#18h        ;
1b,1a,19,18, holds values for 4 digits
                  MOV R4,#4          ;
pin p3.2_ 0 made low one by one starts
wth 18
         ; mov r7,#2       ;  decimal
pt.on third digit from left (2 nd from-
right)
PQ2:         CALL SEGDISP
                  INC R0
                  clr c
                  mov a,r4
                  rrc a
                  mov r4,a
                  jnc pq2
    PV3:
                  RET
  SEGDISP:
                  mov dptr,#ledcode
                  MOV A,@R0
                  ANL A,#0FH
                  MOVC A,@A+dptr
         ; k:  djnz r7,segcode
         ;yesDP:
          ; orl a,#01   ; add a dec. pt.
where it should be
segcode:
                 MOV R5,A
                 ORL A,#03H          ;
WE  WANT TO USE PORT 1 BITS 0 AND 1 FOR
INPUT ANLOG

; so retain them high
S3:      MOV P1,A           ;  SEGMENT_
PORT
S1:      ; MOV A,R4         ; get digit
code from r4
                  ;  rrc a
                  ;  jc s6
                  mov a,r5
                  rrc a
                  rrc a
                  mov p3.7,c         ;
segment' a on p3.7 pin
                 mov a,r4
                ; mov r4,a
                 cpl a
                 rrc a
                 mov p3.0,c
                 rrc a
                 mov  p3.1,c
                 rrc a
                 mov p3.2,c
S5:
S4:      ACALL DELAY1      ;  let  it
burn for some time
```

```
         ;MOV A,#07H
                 ;MOV P3,A ;
                 setb p3.0
;extinguish the digit after that time
                 setb p3.1           ;to
prevent shadow
                 setb p3.2
s6:             RET
ledcode:
DB   7EH,0CH,0B6H,9EH,0CCH,0DAH,0FAH
DB   0EH,0FEH,0CEH,0EEH,0F8H,72H,0BCH,0
F6H,0E2H
                           ;these  are
code for the numbers 0 to 9 and A to F
 DELAY2:  mov 51h,#80       ;80ms
delaywait:   call till20ms
                  djnz 51h,delaywait
                  ret
delay1:
till20ms:     MOV R1,#0ffH
          N:      NOP
                  nop
                  nop
                  DJNZ R1,N
                  ret
tim0isr:          push psw
                  push acc
                  MOV TH0,#0f0H
;AUTO RELOAD VALUE
                  mov tl0,0
                  DJNZ R6,K1A        ;r6
WAS FFH, SO 256 TIMES 4 ms GIVES 1 s
                  MOV R6,#255        ;
11.059 MHz 226 for it;   use 244 for 12
MHz crystal
                  MOV A,R2
                  ADD A,#1
;ADD 1 TO SECONDS
                  DA A
                  MOV R2,A
                  setb 20h
; seconds over
  K1A:
                  pop acc
                  pop psw
                  RETI
;INTERRUPT RETURN INSTRUCTION
table:
db  255,255,255,255,255,255,255,255,255,
255,255,255,255 ;
db  255,255,255,255,255,255,255,255,255,
255,255,255,255;
db  255,255,255,255,255,255,255,255,255,
255,255,255,255;
db  255,255,255,255,255,255,255,255,255,
255,255,255,255,255;
db  251,246,242,237,233,229,226,222,218,
215,211,208,205,202;
db   199,196,193,190,188,185,180,178,176
,173,171;
db  169,167,165,163,161,159,157,155,154,
152,150,149;
db 147 , 145 , 144 , 142 , 141 , 139 ,
138 , 136 , 135 , 134 , 132 , 131;
db  130 , 129 , 127 , 126 , 125 , 124 ,
123 , 122 , 121 , 120 , 118 , 117;
db 116 , 115 , 114 , 113 , 113 , 112 ,
111 , 110 , 109 , 108 , 107 , 106;
db 105 , 105 , 104 , 103 , 102 , 101 ,
101 , 100 , 99 , 98 , 98 , 97;
db  96 , 96 , 95 , 94 , 94 , 93 , 92 ,
92 , 91 , 91 , 90 , 89;
db 89 , 88 , 88 , 87 , 86 , 86 , 85 ,85
, 84 , 84 , 83 , 83;
db 82 , 82 , 81 , 81 , 80 , 80 , 79 , 79
, 78 , 78 , 77 , 77;
db 77 , 76 , 76 , 75 , 75 , 74 , 74 , 74
, 73 , 73 , 72 , 72;
db 72 ,71 , 71 , 70 , 70 , 70 , 69 , 69
, 69 , 68 , 68 , 68;
db  67 , 67 , 67 , 66 , 66 , 66 , 65 ,
65 , 65 , 64 , 64 , 64;
db 63 , 63 , 63 , 63 , 62 , 62 , 62 ,61
, 61 , 61 , 61 , 60;
db  60 , 60 , 60 , 59 , 59 , 59 , 58 ,
58 , 58 , 58 , 57 , 57;
db 57 , 57 , 56 , 56 , 56 , 56 , 56 , 55
, 55 , 55 , 55 , 54;
db  54 , 54 , 54 , 54 , 53 , 53 , 53
, 53;
 END
```