

## FEATURE ARTICLE



Ultra-Low Power Flash MCU  
**MSP430**  
DESIGN CONTEST WINNER

Aleksey Britov, Sergei  
Khlebnikov, Vladimir  
Lomakovsky, Alexander  
Makeenok, & Vitaly  
Zakharchenko

# Automatic Blood Pressure Meter

Thanks to the popularity of TI's MSP430 and all of the entrants' ingenuity, the Ultra-Low Power Flash MCU MSP430 Design Contest was a success. Read on to find out how these five gentlemen from Ukraine captured Second Place in the stiff competition.



The BAT-31 blood pressure meter is intended for systolic and diastolic blood pressure measurement with simultaneous pulse rate measurement. For systolic and diastolic pressure measurement, the oscillometric method is used.

Take a look at Photo 1 for an overall view of the meter. The device consists of two parts: a cuff and an electronic unit. The cuff must be placed tightly around a patient's arm above his elbow and connected to the electronic unit via the attached air tube.

The cover of the electronic unit contains two buttons for device control and a transparent window through which the LCD screen is visible. The buttons are labeled "I/O" and "Start." The LCD shows numeric measurement results and various icons indicating the meter's state. The internal parts of the electronic unit are presented in Photo 2 and Photo 3.

A battery consisting of four AA cells powers the device; they're placed in a compartment that opens to the bottom side of the unit. Between measurements, the device is in Low Power mode and consumes less than 10  $\mu$ A.

To turn on the meter, you have to press the I/O button for a short period

of time. The device then wakes up and displays the LCD test when all of the screen segments are highlighted. After about 2 s, the LCD shows the maximum pressure to which the cuff will be inflated; it stays in this state for 2 s. If you don't press a button, the device goes into Ready To Measure mode, during which the current pressure in the cuff is displayed. At the same time, a buzzer produces a sound, which indicates that the device is ready for measurement.

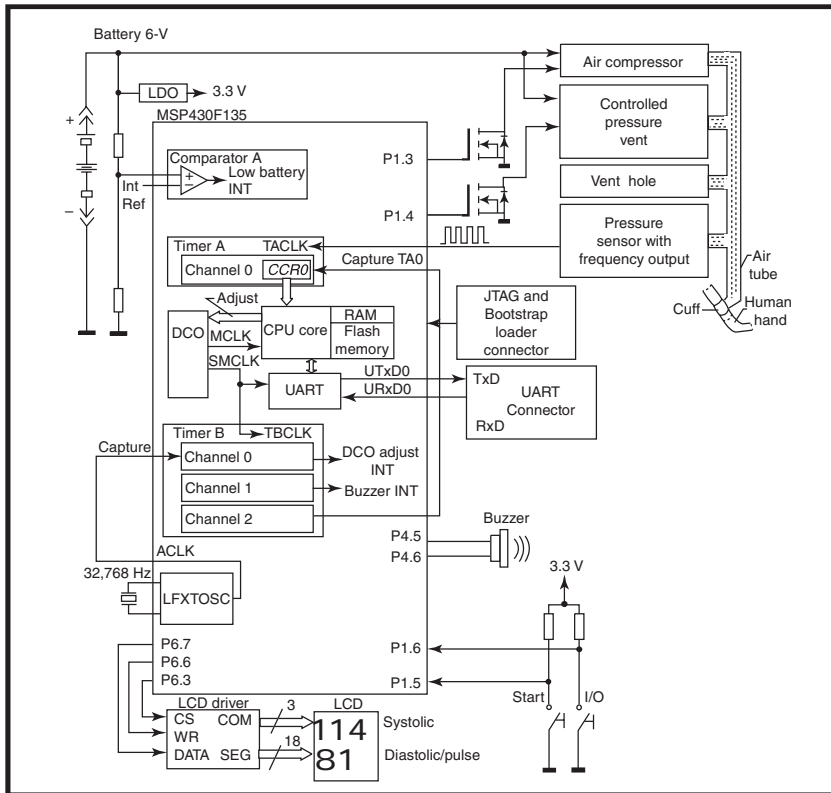
Every time you press the Start button while the maximum pressure is displayed, the value of the maximum pressure changes by 5 mmHg. When the Start button is held down, the pressure value is automatically incremented in steps of 5 mmHg. After the incremented value achieves 285 mmHg, it jumps down to 100 mmHg and then continues to increase. The moment the Start button is released, the last indicated value is stored for use in further measurements in current and succeeding measurement sessions. This value should be set 30 to 40 mmHg above an expected systolic pressure. If you set this value too low, the meter automatically increases it by 40 mmHg during a measurement, but the new value won't persist between measurement sessions.

If for some reason the pressure in the cuff reaches 300 mmHg, the microcontroller opens the controlled air vent to rapidly deflate the cuff. By quickly pressing the I/O button, you can stop all of the processes and turn off the meter.

In Ready To Measure mode, pressing and holding the I/O button for 2 s brings the archive records to the



**Photo 1**—Take a look at the finished product. This view of the Automatic Blood Pressure Meter shows the electronic unit box lying on the plastic air cuff.



**Figure 1**—As you can see from the diagram, an MSP430F135 microcontroller performs parts control and data processing with extremely low power consumption.

LCD. Each record is displayed in two or three steps. First, the record number is displayed (zero corresponds to the most recent record). If the record contains an unsuccessful measurement, the error number is displayed after the record number. In addition, the systolic and diastolic pressures are displayed and followed by the pulse rate.

All of the values are shown in a cycle, each of which is presented on the LCD for 2 s. You can press the Start button to bring the next record to the display.

The blood pressure meter can function under PC control. For this purpose, a UART connector is installed; it should be connected to the computer's COM port by a cable with an RS-232 transceiver. The cable's end connector can be mated to the unit's UART connector through the rectangular hole in the battery compartment's base. In this mode, the unit is powered from an external power supply through the same UART connector.

You can set the maximum pressure value in the PC program's window to specify a name for the file to which the internal data will be written and to initiate measurement by pressing the Start button in the program's window. During a measurement, the PC program receives calibration data, measurement conditions, and raw pressure sensor output data from the unit. The received data may be analyzed for further algorithm enhancements. You can review the blood pressure meter's specifications in Table 1.

## PRINCIPLES OF OPERATION

Figure 1 shows a block diagram of the blood pressure meter. All of the device's functions are controlled by an MSP430F135 microcontroller.

The CPU core clock frequency is supplied by DCO, which is continuously adjusted by the Timer B, Channel 0 interrupt handler. The handler uses the stable 32,768-Hz signal from the LFX-TOSC crystal generator as a reference. Thus, stable MCLK and SMCLK clocks are provided, and they can be used for time interval measurements.

The pressure sensor is a capacitor with a moving plate. The distance between plates changes proportionally to the applied pressure. The capacitor is included in an RC generator, and the oscillating frequency also changes with applied pressure. Timer A and Timer B peripheral blocks accomplish sensor frequency measurements. Channel 2 of Timer B generates a timer window of 8 ms. Timer A counts pulses from the pressure sensor. During a Timer B output signal transition, a current value of Timer A's counter is captured to the CCR0 register. The instantaneous frequency is calculated from this value.

The instantaneous frequency needs smoothing as it experiences fluctuations. We used a first-order digital recursive filter. Its equation is:

$$f_{\text{smoothed}} = 2^{-n}f_{\text{instant}} + (f_{\text{smoothed}} - 2^{-n}f_{\text{smoothed}}) \quad [1]$$

where  $n$  is the number that determines the averaging period.

This filter requires only one memory cell, which is the advantage of such a filter over moving average filters because they require multiple cells. On the other hand, its coefficients are

| Systolic/diastolic pressure measurement range | 20–280 mmHg              |
|---|--------------------------|
| Maximum pressure                              | 300 mmHg                 |
| Pulse rate range                              | 30–180 pulses per minute |
| Pressure error bound                          | ±3 mmHg                  |
| Temperature range                             | 10–35°C                  |
| Power supply voltage                          | 4.1–6.4 V                |
| Current consumed from battery:                |                          |
| • during cuff inflation                       | 450 mA                   |
| • in Low Power mode                           | <10 µA                   |
| Size (without cuff)                           | 125 × 78 × 40 mm         |
| Weight (without batteries)                    | 250 g                    |

**Table 1**—Here are the basic specs for the Automatic Blood Pressure Meter.

degrees of two, so it doesn't require multiplication.

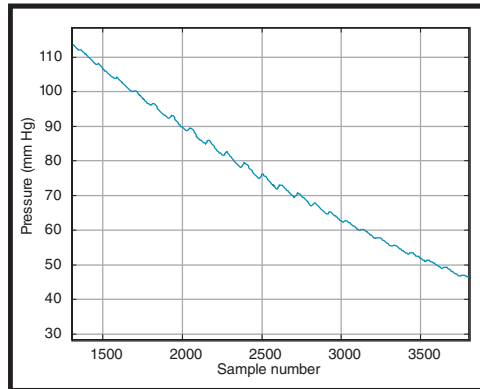
The current pressure value is calculated from the instantaneous frequency. Because the sensor's frequency-pressure response is nonlinear and changes from sample to sample, each device goes through a calibration procedure. The frequency is calculated by using a quadratic approximation of the sensor response, which you'll learn more about in the sensor calibration section of this article.

During cuff inflation, the device measures instantaneous pressure and displays it on the LCD. When the pressure achieves the specified maximum value, the controller turns off the air compressor. Then, the air steadily leaves the cuff through the vent hole, which limits the pressure drop rate to 2 to 5 mmHg per second. You can see a typical pressure curve in Figure 2. When the air pressure in the cuff drops down to the systolic pressure level, it becomes nonmonotonic and pulsates in accordance with blood pulses.

The beginning of each pulsation is indicated by a local minimum of pressure drop curve. Each pulsation is characterized by two parameters, amplitude and pressure difference, which must be stored for further processing. Also, each pulsation amplitude is stored in a byte with resolution of 0.0625 mmHg. Storing a full pressure value that corresponds to a pulsation beginning would require 2 bytes of memory per pulsation. To reduce memory requirements, we stored the difference between current pulsation pressure and the next pulsation pressure. A single byte is enough to store this difference with resolution of 0.5 mmHg. The pulsation parameters are illustrated in Figure 3.

As the pressure steadily drops with time, the amplitude of air pressure pulsations increases, reaches its maximum, and then decreases. After the air pressure drops below the diastolic pressure level, the pulsations almost disappear. By plotting the pulsation amplitude versus pressure, you'll obtain a bell-like curve (see Figure 4).

A counter of frequency samples represents current time from the begin-



**Figure 2**—This graph shows air pressure in the cuff versus time. As the air goes out of the cuff, the pressure goes down. The blood pulsations cause ripples of changing amplitude on the curve.

ning of the measurement. The sampling period is determined by the Timer B, Channel 2 output period; it is constant and equals 8 ms. Thus, any time interval may be determined with a resolution of 8 ms. The translation of a sample number into milliseconds is accomplished with three left shifts of the sample number.

When the program detects rising pulsation amplitudes, it stores the current value of the sample counter and begins to count detected pulsations. After diastolic pressure is bypassed, the program calculates the time interval as the difference between the current sample counter and stored value multiplied by the Timer B, Channel 2 output period. Then, the pulse rate is calculated as the number of detected pulsations divided by the time interval and converted to pulses per minute:

$$\text{Pulse rate} = \frac{60,000 \times \text{number\_of\_pulsations}}{\text{time\_period [ms]}} \quad [2]$$

The measurement ends when the amplitudes of a specified number of successive pulsations drop below a threshold. Take note that the threshold is determined by multiplying the maximum pulsation amplitude by a specified coefficient.

After that, the program moves on to the calculation of systolic and diastolic pressures. The systolic pressure is obtained at the point where

the curve crosses the systolic threshold on the right-hand side of the bell. The diastolic pressure is obtained at the point where the curve crosses the diastolic threshold on the left-hand side of the bell. If a crossing point does not coincide with an available pressure-amplitude pair, the linear interpolation is used to obtain a more accurate value. Note that the program will increase the maximum pressure value by 40 mmHg and go into cuff inflation if the first pulsation amplitude is greater than the systolic pressure. Then, the entire measurement procedure is repeated, and the obtained results are stored in the archive and displayed on the LCD.

## SENSOR CALIBRATION

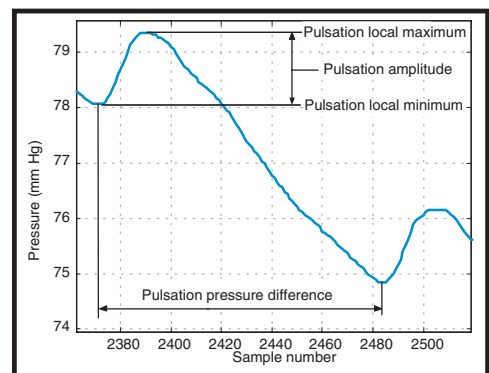
The capacitance of the pressure sensor is inversely proportional to the distance between its plates:

$$C = \frac{\epsilon S}{d_0 - kP} \quad [3]$$

where  $\epsilon$  is the dielectric constant,  $S$  equals plates squared,  $d_0$  is the initial distance between plates,  $k$  is the coefficient, and  $P$  represents the applied pressure. When this capacitance is used in an RC generator, the generator's output frequency is:

$$f = \frac{A}{RC} \quad [4]$$

Therefore, the generator's output frequency is a linear function of applied pressure:



**Figure 3**—The estimation of pulsation parameters is critical for systolic and diastolic pressure measurement.

$$f = \frac{A}{R \frac{eS}{d_0 - kP}} = f_0 - bP \quad [5]$$

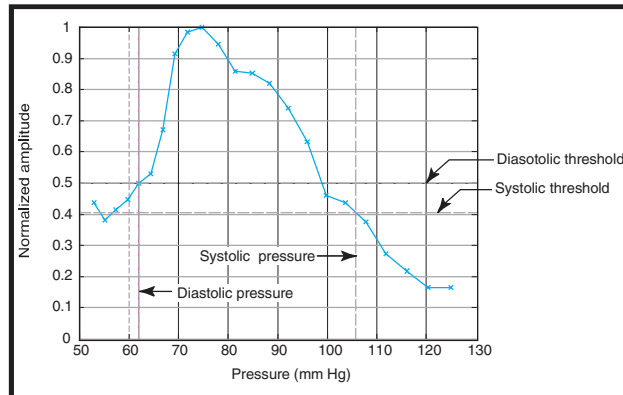
In reality, the manufacturing imperfections, stray resistances, and capacitances result in the pressure-frequency curve's deviation from the ideal straight line. Figure 5 illustrates the typical frequency-pressure dependency for different temperatures. Frequency values are normalized to the frequency at zero pressure and at a temperature of +25°C (77°F).

As you can see in Figure 5, it's clear that in order to achieve an accurate measurement, some sort of approximation of the sensor's response must be found. This approximation also must take into account the temperature dependency of the response. On the other hand, such an approximation should be based on a minimal number of calibration points. After investigating slice-linear and quadratic approximations, we chose a quadratic approximation (see Figure 6).

Calibration data consists of at least three pressure-frequency pairs: one for zero pressure, one for intermediate pressure, and one for maximum pressure. If there is more than one intermediate point, the *C* coefficient is calculated as the average of the values for each intermediate pressure.

For calibration, the blood pressure meter must be connected via an air tube to a pressure reference device. This process may be fully automatic if the reference device has an output signal indicating transitions from one pressure level to another. The device's firmware contains routines that implement calibration and store calibration data in internal MSP430 information flash memory.

The coefficients A, B, and C are calculated in advance, before going to measurement, and only the first formula is used in real-time calculations. It requires five floating-point arithmetic operations: one subtraction, one addition, and three multiplications.



**Figure 4**—When the air pressure in the cuff goes down to systolic pressure, pulsation amplitude begins to increase. After achieving its maximum, it decreases and the diastolic threshold becomes very low. The air pressures at which the amplitude crosses corresponding thresholds give the systolic and diastolic pressure values.

## HARDWARE

The meter's hardware is based on the MSP430F149 (prototype version) or MSP430F135 (current version) microcontroller and uses few additional parts. Because these microcontrollers contain almost all of the peripheral blocks necessary for the device, its schematic is rather simple. You may download the schematic from the *Circuit Cellar* ftp site. The best way to study the schematic is to look at it along with the diagram illustrated in Figure 1.

The VD1 diode protects the electronic circuit against inverse battery polarity. The air compressor and controlled air vent currents don't flow through this diode, so the voltage drop across it is small and allows the circuit to function down to the deep battery discharge.

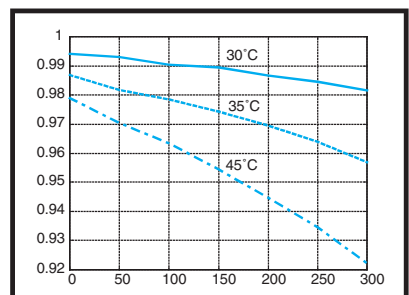
The LDO regulator DA1 provides a stable power voltage for the MSP430 and pressure sensor generator (DD1) independently of the battery discharge level. Because of the MSP430's extremely low current consumption while in Active mode, as well as the fact that it usually stays in Low Power mode LPM3, the power dissipated on the LDO regulator is insignificant.

The implementation of the X1 (JTAG) and X3 (bootstrap loader) connector as a PCB edge connector eliminates the necessity of installing additional connectors, simplifies PCB assembly, and reduces cost. The LCD

driver (DD3) is inevitable with the MCUs currently in use, but the developers hope that the appearance of new MSP430F43x family of chips would make it unnecessary.

The MOSFET switches with extremely low on-resistance DA2 turn the air compressor on and off. In addition, they make the controlled air vent open and close under the MSP430's control. Because all of the MCU's ports are inputs after reset, the switch's gates are tied to ground by resistors R2 and R3 to ensure their off state.

The circuit consisting of resistor R1 and diode VD2 reduces current consumed by the controlled air vent during cuff deflation. When the MCU turns on the air compressor, the current flowing through VD2 simultaneously makes the controlled air vent close. After the air pressure in the cuff reaches the prescribed level, the MSP430 turns the second MOSFET on and the first MOSFET off. The current flowing through the second switch is limited by the resistor and holds the air vent closed. This current is much less than the current necessary to close the vent. Because the air vent stays closed and consumes current throughout the measurement cycle, this technique provides significant battery charge saving.



**Figure 5**—The pressure sensor's output signal frequency is a nonlinear function of pressure and shows significant temperature dependence. The frequency decreases as the temperature goes up. The vertical axis refers to normalized frequency, and the horizontal axis to pressure in millimeters of mercury.

The pressure-sensitive element is the capacitor C1. The distance between the plates of the capacitor C1 is proportional to the applied pressure. The capacitor is included in the loop-back circuit of the RC generator, which is built on DD1 gates. The oscillating frequency fluctuates in the range from 2 to 4 MHz (at zero pressure) down to 1 to 2 MHz (at 300 mmHg) depending on the sensor sample.

R10, C10, R12, R11, C11, and R13 circuits are RC filters suppressing the bouncing of buttons when pressed and released. MSP430 I/O ports allow for not only the reading of the current button state, but also can fix the state change between the moments when the program analyzes its state.

The CPU's internal comparator is accurate enough to be used for battery voltage monitoring. One of its inputs is tied to the internal voltage divider, which provides the voltage of  $V_{CC}/2$  (1.65 V). R15, R16, and C12 form a low-current voltage divider

with an output voltage that drops below  $V_{CC}/2$  for a battery voltage less than 4.1 V. The R15 and R16 values are a trade-off between low-current consumption and the error introduced by voltage drop caused by the comparator's input leakage current. When the battery voltage drops below 4.1 V, the comparator generates an interrupt, which is handled by the program.

The X2 connector is intended for connection to a personal computer and entering a self-test mode. One of the self-test modes provides a calibration procedure. The meter goes into Self-Test mode when the RXD input is held low at the moment when the device is turned on by pressing the I/O button. In addition, R17, R18, R19, and R20 provide ESD protection and the necessary input-voltage bias when the UART connector is disconnected.

$$P = A[f(0) - f_c] \left[ B + A[f(0) - f_c] C \right]$$

$$A = \left( 1 + K \frac{f(0) - f_0}{f(0) - f(P_{MAX})} \right)$$

$$C = \frac{P_{MAX}}{[f(0) - f(P_{MAX})] [f(P_{INT}) - f(P_{MAX})]} - \frac{P_{INT}}{[f(0) - f(P_{INT})] [f(P_{INT}) - f(P_{MAX})]}$$

$$B = \frac{P_{MAX}}{[f(0) - f(P_{MAX})]} - C[f(0) - f(P_{MAX})]$$

where:

- f0 Frequency at zero pressure during calibration is at about 25°C
- f(0) Frequency at zero pressure measured at a measurement session beginning
- f<sub>c</sub> Current frequency
- P<sub>MAX</sub> Maximum calibration pressure (should be close to 300 mm Hg)
- P<sub>INT</sub> Intermediate calibration pressure (in the range from 0 to P<sub>MAX</sub>)
- K Temperature coefficient

Figure 6—You can use this quadratic approximation to determine the sensor's response.

A Piezoelectric buzzer is used to indicate the Ready To Measure mode and pulsation detection during measurement by sound. Because the circuit's supply voltage is low, in order to make the sounds louder, the buzzer is connected in Differential mode. The signals on its plates produced by MSP output port change in opposite phase so that the whole voltage change across the buzzer is twice the supply voltage.

## FIRMWARE

The blood pressure meter's firmware is written in MSP430 assembly language. For its development, the MSP430 Kickstart tool and the FPP430 floating-point library were used.

You may download the main program flowchart from the *Circuit Cellar* ftp site. Because the program is large, it's implemented as a module set. Each module is implemented as a separate source file with the corresponding header file. All of the files in a module are gathered in a separate directory whose name is the same as the module's name. The purpose of each module is described in a comment at the beginning of each file. The most important comments were translated into English. The rest of comments are in Cyrillic and may be displayed improperly. The modules are listed in Table 2.

If you want to compile the source code, all necessary files are in the Project\_TI090Source\_code directory.


| Module name  | Module purpose   |
|--------------|--|
| ARXIV.S43    | Store the last 10 results in internal archive in the information flash memory; display current results and results from archive  |
| CLOCK.S43    | Control MSP430 clock system; adjust DCO frequency (uses 32,768-Hz crystal as reference); implement program delay of <i>N</i> ms  |
| FPPV4IAR.S43 | MSP430 floating point package  |
| I2C.S43      | Implement I <sup>2</sup> C interface (required only when Philips LCD driver is used)   |
| IEEE_MSP.S43 | Convert number from IEEE STD 754 format to MSP430 floating-point format and visa versa   |
| IZMEREN.S43  | Set maximum pressure before measurement; implement Manometer mode before measurement cycle; inflate cuff before measurement; measure systolic and diastolic pressures and pulse rate                                     |
| LCD.S43      | Control LCD driver on logic level (independent of LCD driver type); driver type is selected in header file MAKET.S4H. An appropriate control file (see below) is included in this file by the #include directive.        |
| PCF8566.S43  | Control Philips LCD driver on the physical level   |
| HT1621B.S43  | Control Holtek LCD driver on the physical level  |
| 430F4xx.S43  | Control MSP430 internal LCD driver (not yet developed)   |
| MAIN_PRG.S43 | Main program; call to all other functions; switch the pressure meter to Low Power mode after measurement; implement interrupt vector table   |
| MAKET.S43    | Define all macros, names, constants and masks for hardware control; initialize MSP430 hardware; erase/write to MSP430 information flash memory; define all control constants for the pressure meter program              |
| SENSOR.S43   | Store pressure sensor calibration data; calculate coefficients for sensor response approximation; receive signal from pressure sensor; calculate and low-pass filter instantaneous frequency; calculate instant pressure |
| TESTS.S43    | Implement self-tests and calibration routine (runs as one of the tests)  |
| UART_FON.S43 | Initialize and control MSP430 UART hardware; receive packets from the PC; transmit packets to the PC   |

Table 2—Because the program is so large, it's implemented as a firmware module set. As you can see, each module has its own purpose.

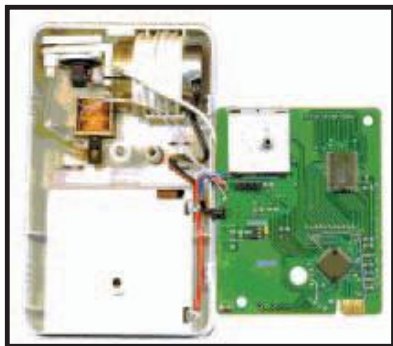


**Photo 2**—We took off the blood pressure meter's front cover so you can see the placement of the electronic block's main parts. You can also see the top of the PCB.

To compile the project, you have to make Project\_TI090\Source\_code a current directory, load the bp\_meter.prj project file into MSP430 KickStart, and execute the menu command Project\Build All. The program occupies 8798 bytes of flash memory and requires 478 bytes of RAM.

For more information on our project, stop by our web site at [cad.ntu-kpi.kiev.ua/~dsplab](http://cad.ntu-kpi.kiev.ua/~dsplab). 

*Aleksey Britov is a researcher in the CAD department of NTUU-KPI in Kiev, Ukraine. He received an engineering diploma from the National Technical University of Ukraine (NTUU-KPI). His technical interests include the design and implementation of hardware and software digital signal processing, and DSP and microcomputer-based embedded systems design.*



**Photo 3**—Take a look at the blood pressure meter's electronic unit when the PCB is put aside (upside down).

*Sergei Khlebnikov is a researcher in the CAD department of NTUU-KPI in Kiev, Ukraine. He earned his engineering diploma from National Technical University of Ukraine (NTUU-KPI) in 1993. His interests include the hardware design of embedded systems based on DSP and microprocessors, and software utilities for CAD PCB layout systems.*

*Vladimir Lomakovsky is currently the technical director of REBUS in Kiev, Ukraine. He received his engineering diploma from the Lvov Polytechnical Institute. Vladimir is interested in the design of medical devices, as well as the development of pneumoautomatic parts, constructions, and algorithms.*

*Alexander Makeenok is a senior scientific researcher at the International Research and Training Centre in Kiev, Ukraine. He holds an engineering diploma and PhD from the National Technical University of Ukraine (NTUU-KPI). His interests include the design of CAD systems for DSP and microcomputers, and control and telemetry over the Internet.*

*Vitaly Zakharchenko is the director of REBUS in Kiev, Ukraine. He received an engineering diploma from National Technical University of Ukraine (NTUU-KPI). His primary technical interests involve the development and production of medical diagnostic equipment.*

You may reach any of the authors at [dsplab@cad.ntu-kpi.kiev.ua](mailto:dsplab@cad.ntu-kpi.kiev.ua).

## SOFTWARE

To download the code, go to [ftp.circuitcellar.com/pub/Circuit\\_Cellar/2002/146/](http://ftp.circuitcellar.com/pub/Circuit_Cellar/2002/146/).

## SOURCES

**MSP430F135 Microcontroller, Kickstart**  
Texas Instruments, Inc.  
(800) 336-5236  
[www.ti.com](http://www.ti.com)

# PCB LAYOUT

SOFTWARE FOR WINDOWS - FREE



Download our FREE layout software  
 Design your two-sided plated-through PCB  
 Send us your design with just a click  
 Receive top quality PCBs in a few business days  
 often under \$100




## CIRCUIT BOARDS

# \$62

Select our MiniBoard service and get  
 three top quality 2.5" x 3.8" PCBs  
 for \$62 shipping included!

## expresspcb.com



For a  
 Discount  
 on your next  
 order use the  
 Code: PCS112

- No tooling charge!
- Lot charges start at \$80
- Simple order process
- Quickturn, low quantities

TWO SERVICES FOR CIRCUIT BOARDS




INSTANT ON-LINE QUOTES!

(Design required)

www.pcbpro.com

- Lowest Price Comparisons
- 60 days options and added features
- Prototype & production quantities