Whistles from on high

Acoustic vertical speed indicator

Thierry Charlès

Let's not beat about the bush: the sole function of the acoustic vertical speed indicator (VSI) described here is to provide an RC model glider pilot with information about the rate of climb (or descent) of the craft.

Designed to detect very small variations in atmospheric pressure so as to enable the pilot of the model fitted with it to keep within an ascent, especially when these are infrequent, our acoustic vertical speed indicator is an example of some of the most complex electronics we're likely to come across, employing both analogue and digital technologies. The project combines RF and LF and for this reason, it calls for extremely careful construction, since the dynamic range of the signals that rub shoulders on the PCB reaches 120 dB.

General

All 'proper' aircraft are fitted with an instrument, the 'VSI' that indicates on a dial the 'rate' of climb or descent (in ft/mn = feet per minute or m/s = metres per second) — in other words, how fast it is climbing or descending. A model glider pilot stands behind a remote control, eyes glued to the sky, and when the air is still, it's sometimes hard to judge the vertical acceleration and catch that 'thermal' that's going to allow the model to soar aloft. Our vertical speed indicator for models is fitted with a pressure sensor. When the glider moves in the vertical plane, the measured atmospheric pressure will vary. The associated electronics converts this pressure change into a vertical acceleration figure that's then used to modulate an audio signal, transmitted via a radio link to the receiver, and thence to the pilot's ears. Climbing is indicated by a high-pitched signal, descending, by a low-pitched one. In each direction of variation (climb/descend), three vertical speed ranges produce three different frequencies. Below the lowest speed threshold, the audio frequency is modulated by the speed, changing from a continuous tone to an intermittent beep.

Target performance

The model flying zone lies at an altitude between 0 m (sea level) and 3,000 m. In order to 'catch a thermal' when the air is still (no wind), the VSI needs to be able to measure a low vertical speed: the target sensitivity is 10 cm/s (< 20 ft/mn).

It is also necessary to be able to pilot the model in real time, so it mustn't wait a whole second before informing the pilot — the reaction time of the circuit (pressure measurement + processing + audio generation + transmission) must be less than half a second.

Block diagram

As the block diagram in **Figure 1** shows, the unit consists of two modules: a transmitter installed in the glider (the top series of blocks) and a receiver on the ground (the bottom series of blocks).

The on-board part consists of an absolute pressure detector, a quad op. amp (the analogue processing chain), an 8-

pin microcontroller (taking care of the digital processing), and a miniature UHF transmitter. A compact power supply makes it possible to draw the power needed (> 200 mW) from a 7 V lithium polymer (LiPo) accumulator, though in fact any self-contained 6/12 V battery or rechargeable would do.

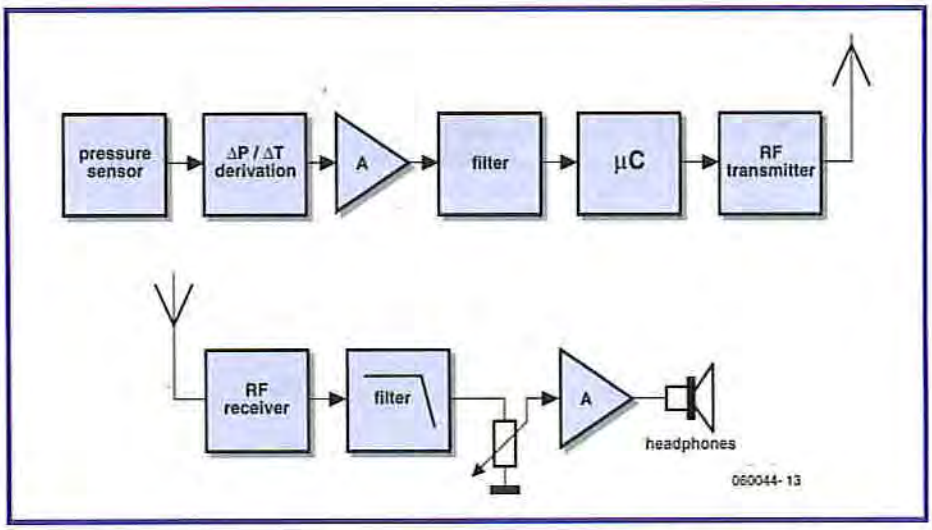


Figure 1. Block diagram of the two sub-assemblies that make up the acoustic VSI: transmitter (top) and receiver (below).

Circuits

Having got an idea of the sub-assemblies that make up this project, now it's time to take a look at the circuit diagrams. To keep things simple, we've subdivided the project into two diagrams, one for the transmitter (**Figure 2**) and one for the receiver (**Figure 3**). Let's start with the most complex one, the transmitter.

Analogue

We opted for an MPX 5100AP sensor (from Motorola, since become Freescale) as it enables us to achieve the desired sensitivity, without being a miniaturized component; what's more, it is compatible with being used in a model, while being readily available at an affordable price (the datasheet can be downloaded from the site provided in reference [7]).

The MPX 5100AP is simple to use: it provides a DC output voltage with a slope of 45 mV/kPa, i.e. 512 µV per metre (at ground level @ 15 °C). So in order to react to 10 cm, the VSI electronics needs to have a sensitivity better than 50 µV.

The voltage provided by the sensor may vary from 0.5 V (0 mb) to 4.75 V (1,100 mb); now the dynamic range of our VSI is limited to between 0 m and 3,000 m; hence the useful output voltage is going to be from 4.75 to 2.95 V. Establishing a virtual earth V_{EE} at 2.5 V enables us to obtain a signed speed output directly from the analogue chain. In the absence of speed, the output is at 2.5 V (virtual earth zero); if the model climbs, the pressure reduces, the sensor output voltage falls, and the analogue chain output signal is negative w.r.t the virtual earth V_{EE}. During descent, the reverse happens.

Let's take a closer look at all that. R10, C7, IC4A are wired as a differentiator. R7 is a compensating resistor to stabilize the gain of the circuit, regardless of phase. Analysis of the circuit shows that it is also a filter with a lower cutoff frequency of 0.23 Hz; this filter eliminates rapid movements that tend to repeat (vibrations). The gain G of the differentiating circuit is set at

$$G = -6.8\ \mathrm{d}P\ /\ \mathrm{d}T$$

IC4B amplifies the signal with a gain $G = 56\times$.

The first-order low-pass filter R6/C8 cutting off at 156 Hz eliminates most of this noise. The function of the last opamp, IC4C, is to match the impedance between the filter and the microcontroller ADC input.

For a vertical speed of 10 cm/s, the wanted signal out of the pressure sensor ($\mathrm{d}P/\mathrm{d}T$) is 51.2 µV, which is detected and amplified by the analogue

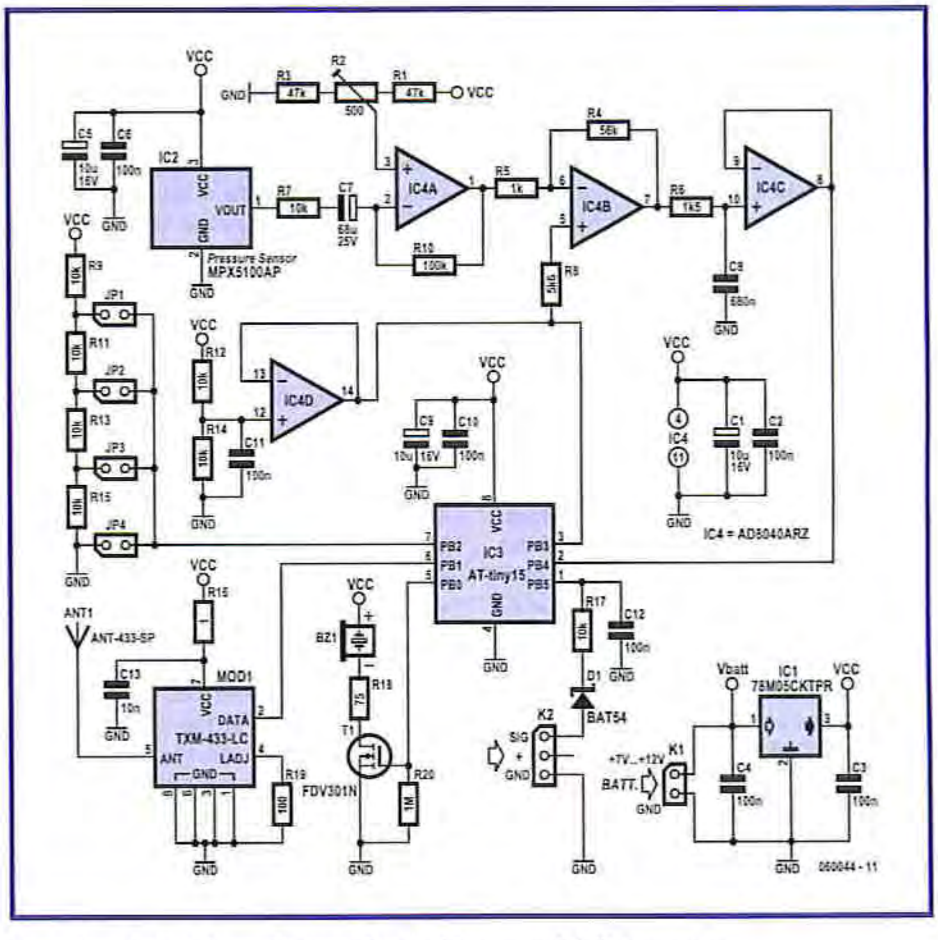


Figure 2. Transmitter circuit diagram. The integrated aerial and the sensor are the bulkiest components.

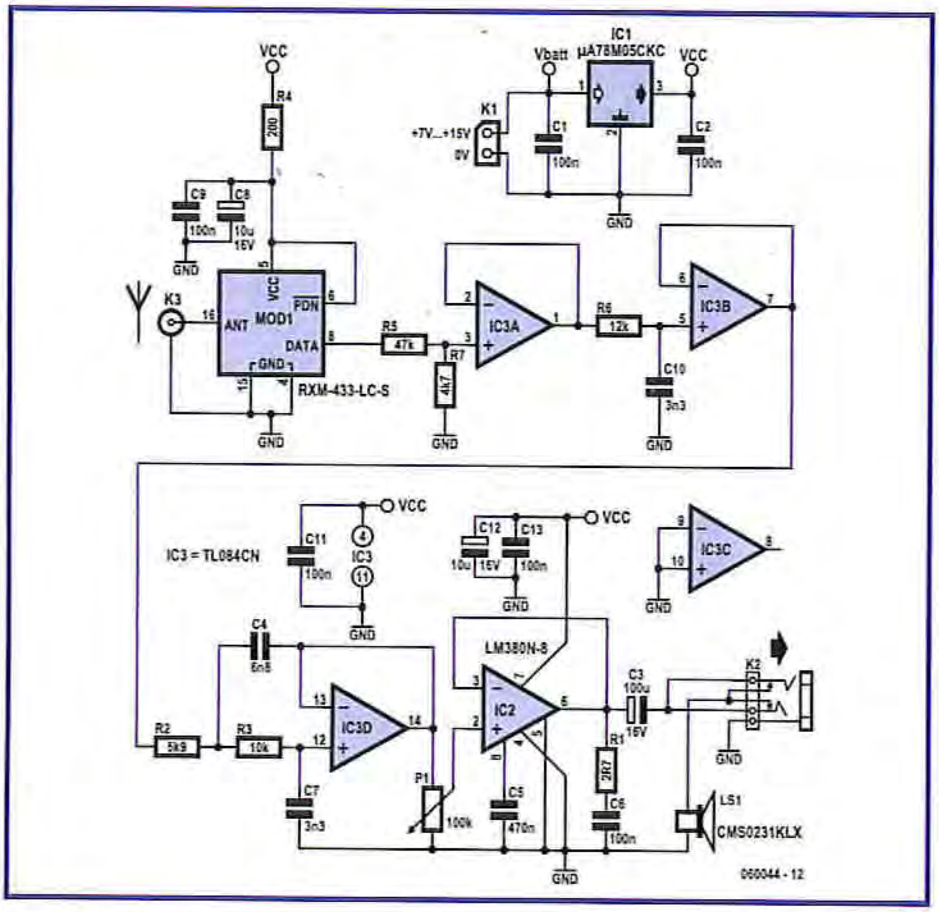


Figure 3. The receiver circuit amounts to very little really, as the integrated module does the lion's share of the work.

processing chain; at the input to the microcontroller this wanted signal (19.3 mV) is drowned in noise and offset by the offset voltage.

Signal processing is performed digitally by the microcontroller.

Digital

The digital processing chain extracts the wanted signal from this noisy information, then calculates the relevant speed and delivers the audio signal.

The microcontroller we've chosen is the ATMEL ATtiny 15L, which is powerful enough despite its 8 pins, with four 10-bit ADC inputs, two programmable counters, and an EEPROM. What's more, it can operate without any additional components (crystal, voltage reference, Reset, etc.). Choosing V_{CC} as the reference voltage, the 10-bit ADC has a resolution of 2^{10} i.e. 5 V ÷ 1,024 = 4.88 mV, defining the circuit's sensitivity threshold at 12.8 µV, i.e., 2.5 cm/s (12.8 µV × 6.8 × 52).

An interrupt triggers the program every 5 ms. Each interrupt prepares an A-D conversion in the following order: the virtual earth, the speed, the absolute pressure, the battery voltage. Thus each of these parameters is encoded at 50 Hz.

To extract the wanted signal, the software performs three filtering operations. The first supplies the mean value of five consecutive measurements, thus providing a filtered speed across five samples at 10 Hz. The second filter performs a rolling average at 10 Hz of the last *n* (where *n* is configurable) values of the filtered speed. The speed value obtained at the output of this second filter stage is used to produce the audible signal.

The third stage of processing is enabled when the speed signal produced at the output of the second filter remains within the 'configurable' level flight limits (± 20 cm/s or ± 30cm/s, i.e. when the vertical speed is less than 3 or 4 cm/s) for 7.2 seconds at a stretch. This filtering derives the mean value of the residual noise, on the assumption that it in fact represents the offset at the output of the analogue chain; the relevant value is used to 'digitally offset' the virtual earth value. Thus the bias at the analogue chain output is compensated for in real time.

Together, all this processing increases the sensitivity dynamically to bring the noise amplitude on the wanted speed to ± 2 LSB (± 5 cm/s).

In the presence of this processing, having defined (in consultation with end users) the minimum detectable speed threshold at 20 cm/s (by clipping), on the bench the circuit does not produce any spurious audible signal, unless there's a forecast of storms....

Below the 'level flight' speed threshold, the software considers that the model is flying level.

To produce the audible signal, four speed values have been defined: 50 cm/s, 75 cm/s, 1 m/s, and 2 m/s. Below 1 m/s, the audio frequency is generated at 625 Hz descending and 1,000 Hz climbing, and is modulated by the speed, going from a keyed tone at the lowest speed to a continuous tone at 1 m/s. Between 1 and 2 m/s, the sound is continuous at 1,250 Hz climbing or 430 Hz descending. Beyond 2 m/s, a continuous tone of 1,650 Hz climbing and 310 Hz descending is produced.

The on/off modulation (keying) below 1 m/s increases below 75 and then 50 cm/s to enable variations at slower speeds to be detected.

Table

Speed	Climbing	Descending
<30 cm/s	Nothing	Nothing
<1 m/s	1,000 Hz keyed	625 Hz keyed
>1 m/s	1,250 Hz steady	430 Hz steady
>2 m/s	1,650 Hz steady	310 Hz steady

The audio generation needs to be 'decoupled' from the continuous, rapid speed variations.

Without this processing, the audio signal is constantly varying and hence difficult to interpret. To obtain 'meaningful' audio information, the audio generation processing takes the maximum speed value produced over three cycles (the maximum value measured every 300 ms).

To achieve maximum sensitivity in the analogue chain, the circuit needs to have reached thermal stability and the differentiating capacitor to 'have reached charge balance'.

In order to differentiate an almost continuous voltage (frequency < 0.1 Hz), the capacitor charging time constant is long. From a cold start, stable operation is reached after 20 minutes; from a 'hot start', the operating point is reached 3–5 minutes after power-up.

This difference is partly due to the temperature stabilization, as well as to the fact that when hot starting, the differentiating capacitor is already partially charged.

Operating initialization takes place in three stages:

1. the voltage at the ADC falls rapidly from 2.5 V to 195.2 mV (the differential voltage between pins 3 and 2 of the microcontroller) in less than 3 minutes; this delay will varies, as it depends on the type of start: cold or hot. During this phase the audio output produces an 800 Hz signal on/off modulated; flying is possible, but strongly to be discouraged.
2. The ADC input voltage is below 190 mV, the level flight threshold is set at 30 cm/s and the microcontroller 'compensates' for the offset at its analogue input. This compensation is established in two ways: by adjusting the offset dynamically by means of regular self-calibration while the measured speed remains below ± 30 cm/s (assumed to mean level flight); and by superimposing on this first compensation inverse correction (predefined) for the differentiating capacitor charging. During this phase, flying is possible, albeit with slightly reduced performance: reduced sensitivity and a risk of the appearance of a slight bias on the audio output.
3. Once the offset correction calculated in phase 2 falls below ± 20 cm/s, the first successful self-calibration ends the operational initialization process and automatically selects a sensitivity of ± 20 cm/s.

Developing the software

The code was written in assembler, using the tools provided by ATMEL, Studio 3.5 then Studio 4, available on free download from the ATMEL website (Ref [8]). Once you've got the hang of this tool, writing, debugging and emulating become child's play.

The program VMR-0-4.hex will be loaded into Flash memory, using one of the suitable commercially-available tools; in developing this project, the STK 500 tool was used in conjunction with Studio 4.

After loading, the microcontroller has to be configured in order to function correctly: RESET = internal, BROWNOUT = 4 V, and it is preferable, though not essential, to use the internal clock calibration constant. Numerous examples of code (Assembler and C) are available on the ATMEL website, obviating the need to write everything from scratch (Ref. [9]). The documentation that enables us exploit the microcontroller's resources correctly is comprehensive and contains a wealth of information. You can also download the detailed spec. sheet for the microcontroller from the ATMEL website (Ref. [10]).

Once created and tidied up, the VMR-0-4 software occupies 99 % of the Flash memory and 40 % of the EEPROM.

The software runs at a rate determined Timer 0 which supplies an interrupt every 5 ms (processing time 200 µs). Every five cycles, the main process is run (process time < 400 µs).

The background task manages switching between the main task of processing, putting the microcontroller into standby, and the audio output; it takes less than 50 µs.

In assembler, processing times are extremely short.

For this application, the processing time is not a constraining factor, the main problem is posed by the EEPROM write time — as there is no RAM, use has to be made of the EEPROM resources to store the data. The duration of a write cycle is 8,192 machine cycles, i.e. 5.12 ms at a clock frequency of 1.6 MHz. The processing task performed at 50 Hz performs three writes and 16 reads; there is no constraint on the read duration.

This task requires a processing time of 3 x 5.1 ms = 15 ms, which is still compatible with the 20 ms cycle time. To limit the effects of an uncontrolled process, the interrupt generator that sets the processing rate to 5 ms is interrupted during this task.

RF section

The RF transmitter is the smallest possible, and the frequency the highest permitted, so as to ensure frequency-compatibility with the radio-control receiver and limit the size of the on-board aerial. We chose the TXM-433 from LINX Technologies. The summary, just about adequate, datasheet can be downloaded from the Internet (Ref. [3]). With a standard receiver, as well as with a dedicated receiver specially built for this application, the range is sufficient.

The 50 Ω transmitter aerial has been designed for the most even radiation pattern, whilst taking very little space. So we opted for a planar-type integrated aerial, the 433-SP2 'Splatch' from LINX (them again!). Opting for this type of aerial allows us to produce an extremely compact transmitter. Nothing overhangs the board, not even the pressure sensor.

This leaves the receiver to talk about. This version of the receiver, the circuit shown in **Figure 3**, is extremely simple: an RXM-433-LC-S RF receiver, also from LINX Technologies, a low-pass filter to suppress everything except the wanted audio signal, and an amplifier.

Power supply

The circuit could be powered from the model's own battery — to do so, all you need do is replace IC1 by a protective choke. Although this solution does make it possible to limit the space occupied and the weight, it is not really to be recommended. Models are powered from a battery whose voltage varies from 4.7 to 5.6 V at full charge. This supply voltage variation, along with the interference generated by the servo motors, is not conducive to correct operation of the analogue chain and achieving the required sensitivity.

The processing chain needs a stabilised, regulated 5 V supply, and the virtual earth must be 'centred' with an error of less than 1 mV. In order to avoid potential interference that could disrupt the operation of the VSI or the model, the circuit is designed to be supplied from a dedicated battery.

Construction

The two PCBs, whose component overlays are shown in **Figures 4** and **5**, require careful construction, and the

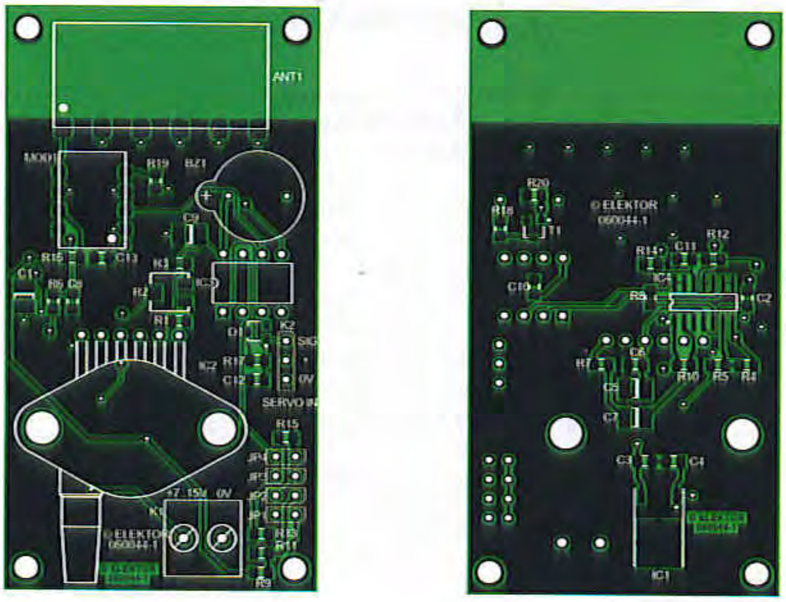


Figure 4. Transmitter component overlay.

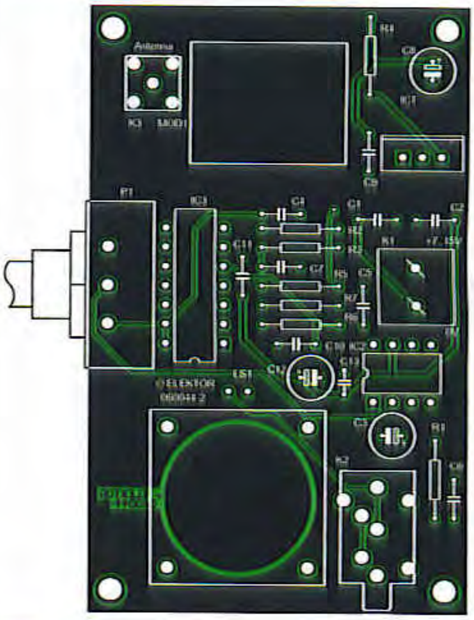


Figure 5. Receiver component overlay.

components must be chosen carefully to minimize the noise level.

The two boards are available from the usual addresses.

On the PCB, we find the five blocks from the block diagram: power supply, RF transmitter, microcontroller, op. amp, and sensor. Each of these blocks has its own decoupling capacitors, and the shortest possible track lengths.

The PCB is double-sided and through-hole plated, fitted with SMD components on both sides…

The recommended order for fitting the components to the PCB is as follows:

- Stick or position the SMDs, then solder. Take care to check the value of each component before fitting it. These components are not at all difficult to fit, it's 'almost' as easy as fitting the discrete components. One tip is to use a

Components list

Transmitter (060044-1)

Resistors

(all 0805 SMDs)
R1,R3 = 47kΩ
R2 = 500Ω SMD preset (4 mm)
R4 = 56kΩ
P5 = 1kΩ
P6 = 1kΩ5
R7,R9,R11–R15,R17 = 10kΩ
P8 = 5kΩ6
R10 = 100kΩ
R19 = 100Ω
R16 = 1Ω
R18 = 75Ω
R20 = 1MΩ

Capacitors

C1,C5,C9 = 10µF
C2,C3,C4,C6,C10,C11,12 = 100nF
C7 = 68µF
C8 = 680nF
C13 = 10nF

Semiconductors

D1 = BAT54
T1 = FDV301N
IC1 = 78M05CKTPR
IC2 = MPX5100AP (ABS pressure sensor, 16.68 psi MAX.)
IC3 = ATtiny-15 (Atmel), programmed, order code **060044-41**)
IC4 = AD8040ARZ (SMD)

Miscellaneous:

JP1–JP4 = 2-way SIL pinheader
K1 = 2-way PCB terminal block, 5 mm lead pitch
K2 = 3-way SIL pinheader
ANT1 = Splatch SMD aerial (ANT-433-SP-ND from LINX)
BZ1 = 5 V sounder, 12 mm diameter, e.g. PB-12N23P-03Q (Mallory 12 mm)
MOD1 = TXM-433-LC (LINX)
PCB, order code **060044-1** from Elektor SHOP

Receiver (060044-2)

Resistors

R1 = 2Ω7
R2 = 5kΩ9
R3 = 10kΩ
R4 = 200Ω
R5 = 47kΩ
R6 = 12kΩ
R7 = 4kΩ7
P1 = 100kΩ potentiometer

Capacitors

C1,C2,C6,C9, C11 = 100nF
C3 = 100µF
C4 = 6n8F
C5 = 470nF
C7,C10 = 3nF3
C8,C12 = 10µF
C13 = 100nF

Semiconductors

IC1 = µA78M05CKC
IC2 = LM380N-8
IC3 = TL084CN

Miscellaneous

ANT1 = ANT-433-CW-HWR-RPS (LINX)
K1 = 2-way PCB terminal block, 5 mm lead pitch
K2 = 3.5 mm stereo jack socket
K3 = SMA socket, Digikey #ACX1231-ND
LS1 = miniature loudspeaker (CMS0231KLX)
MOD1 = RXM-433-LC-S
PCB, order code **060044-2** from Elektor SHOP
Software, file # 060044-11.zip from www.elektor-eletcronics.co.uk

pair of cross-tipped tweezers for holding the SMDs in place while you solder them.

- Fit the aerial and solder it to the six connection points provided.
- Before applying power to the circuit, use a multimeter to check the supply rails connections to the relevant device pins.
- Position the sensor, taking care that the fixing holes are correctly aligned. To ensure good mechanical fixing, you can if you wish insert two flat washers between the sensor and the PCB and then fix it using two PTFE screws; then solder it.
- Add a cable-tie if necessary onto one of the fixing screws so as to 'lace' the wiring to the circuit.

Solder the op. amp on the underside of the board, then solder the eight contacts (once again, it's hard to talk of 'pins'!) of the RF transmitter.

Don't plug the microcontroller into its socket yet, but instead connect an oscilloscope between pins 2 and 3 of its socket.

Note: the sounder BZ1 could be fitted in the space provided for it — it will be able to be used to identify the model, once the software is modified for this purpose (function not yet available at time of writing).

Now let's move on to constructing the receiver. Here again, we have a through-hole plated, double-sided PCB. Most of the components here are conventional devices, except for the receiver module itself, which is an SMD. The layout is relatively dense. You should start by fitting the SMD module, then the smaller components. The rest of the components don't need any special comment. The miniature speaker will be positioned on the underside. You might also consider fixing it onto the cover of the case, linking it to the points provided using two short lengths of flexible wire.

Initial testing

After checking the quality of your construction, connect up the battery (take care if you are using an earthed bench supply with the oscilloscope, which could take the virtual earth down to true earth). The output signal is saturated for a few tens of seconds, then it will gradually go down to virtual earth (0 V on the screen, 2.5 V in absolute terms).

Turn off the power, load the software into the microcontroller using a suitable tool (or buy the microcontroller ready programmed from Elektor), and insert it into its socket. Before powering up, set the VSI on a firm support so it can't move. Close doors and windows (to avoid detecting draughts and limit interference from ambient pressure changes).

After power on, the software starts up in 'operational initialization' mode — in this state it emits an intermittent 800 Hz audio signal. This state is maintained for less than three minutes, as long as the microcontroller does not detect a change to below 1 m/s over a period of 3 seconds. Then the sound stops, and the VSI is all ready to fly.

The operational initialization continues transparently for as long as the microcontroller input offset measured by the self-calibration process is not within the level flight range (± 20 cm/s). It is advisable to leave the circuit permanently powered so as to help the zero seeking by the software. Breaks in power and battery disconnections must be spaced apart by two minutes.

To test operation, all you need do is open or close the door of the room — the VSI will detect the pressure variation caused by opening or closing the door.

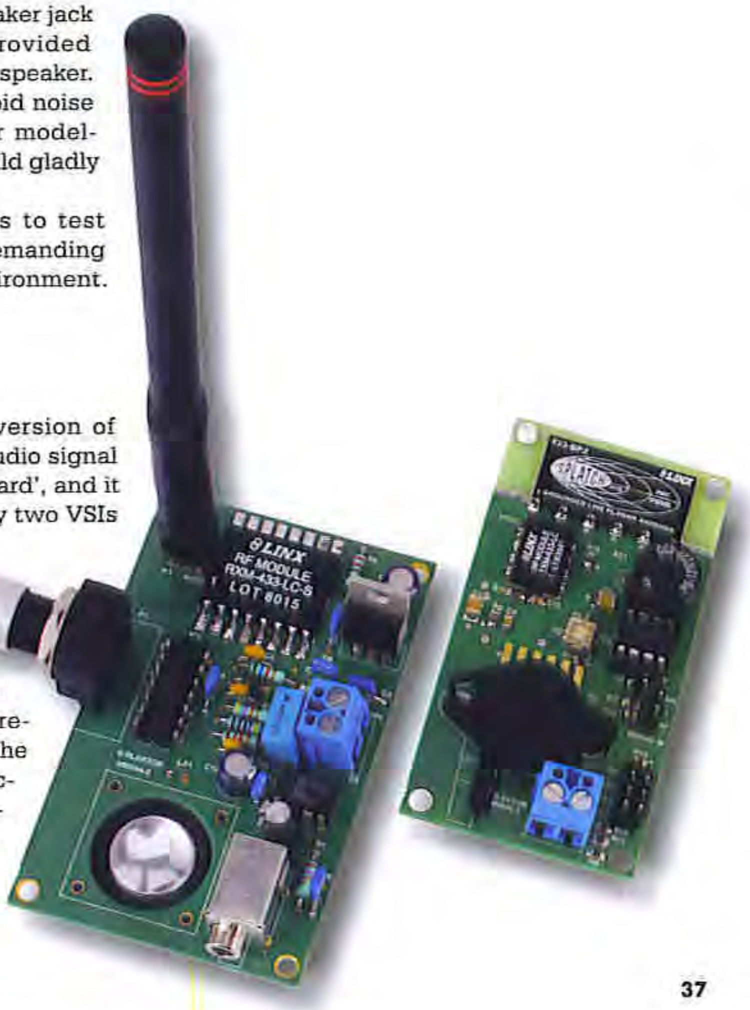
Plugging a loudspeaker jack into the socket provided cuts off the internal speaker. That way you'll avoid noise pollution that your modelling colleagues would gladly do without.

All that remains is to test the circuit in the demanding aeromodelling environment. Good luck!

Future

In the 'VMR 0-4' version of the software, the audio signal is generated 'on board', and it is not possible to fly two VSIs at the same time near one another, as the RF signals interfere with each other upon reception. However, the equipment architecture has been designed to allow five units to be flown simultaneously; this functionality could be available in a later version of the software.

(060044-I)



Web links

[1] www.freescale.com/files/sensors/doc/app_note/AN1646.pdf

[2] www.freescale.com/files/sensors/doc/

[3] www.linxtechnologies.com/Documents/TXM-xxx-LC_Data_Guide.pdf

[4] http://membres.lycos.fr/cepls/plan.html

[5] http://courelectr.free.fr/AOP/COURS.HTM

[6] www.freescale.com/files/sensors/doc/app_note/AN1100.pdf

[7] www.freescale.com/files/sensors/doc/data_sheet/MPX5100.pdf

[8] www.atmel.com/dyn/products/tools_card.asp?tool_id=2724

[9] www.atmel.com/dyn/products/app_notes.asp?family_id=607

[10] www.atmel.com/dyn/resources/prod_documents/doc1187.pdf