Wireless church tower bell sensors using an inexpensive 433MHz OOK transmitter and receiver

This project was designed for the church of All Saints Basingstoke to provide a wireless communication link between optical tower bell sensors and a Windows PC in the ringing chamber. The primary motivation for the project was to dispense with the existing unsightly cabling and followed the augmentation of the bells from nine to ten in March 2016.

History

The church of All Saints Basingstoke has possessed a ring of nine bells since 1916. The original concept was to provide a solemn Dorian mode (minor) eight for use in penitential seasons e.g. Lent and a normal major eight at other times. In this centenary year (2016) the bell ringers decided to add an additional treble to create a major scale ring of ten.

The ringers use the Abel bell simulation program (*reference 1*) to provide a silent practice facility. Up till now it has been used in conjunction with eight optical sensors. Following the augmentation to ten bells two additional sensors were added. However the existing cable had insufficient cores for these additional sensors. This meant that we had either to replace or supplement the existing cable, or investigate the possibility of a wireless solution. In the event we chose a wireless solution because this meant that:

1) We could remove the unsightly cabling.
2) A wireless solution permits the flexible placement of the PC at any location within the ringing chamber.
3) A wireless solution increases the utility of our custom odd struckness meter[1], meaning that it can now be used in any part of the ringing chamber, or indeed in the belfry itself.

Notes:

1. An odd struckness meter is used to measure the time interval between a bell passing the exact bottom dead centre position (mouth down) and striking. This interval should be equal on each stroke. If it is not equal then the bell is said to be odd-struck.

Design Requirements

1) The solution should support up to 10 bell sensors, plus a sound volume sensor (for use with our existing odd struckness meter).
2) The solution should have adequate range to give reliable transmission between the belfry and the ringing chamber.
3) The solution should introduce minimal latency (of no more than a few milliseconds).
4) The solution should work with our current optical sensors.
5) The receiver should provide three 9 pin female DSUB outputs that are pin compatible with the Abel simulation software and the existing serial ports on the PC.
6) The receiver should be fully portable and have the possibility of being powered through internal batteries as well as via a wall-wart style USB charger or spare USB port on the PC.

7) The receiver should provide phantom power to our odd struckness meter.
8) The solution should be cheap!

The Design

Cheap RF modules

A search of the internet will reveal that simple 433 MHz transmitters and receivers are available as a pair for less than £2, including delivery. One particular design predominates and is often advertised as being suitable for providing a wireless link for Arduino and Raspberry-Pi applications. This commonly available product comprises two compact PCB boards, the receiver board being built with mostly surface mount components.

The transmitter board has 3 pins: a) data, b) power and c) ground, plus a solder pad for the antenna. The receiver board has 4 pins: a) power, b) data, c) data and d) ground, plus a solder pad for the antenna. The two data pins are in fact joined together on the PCB board so form one connection.

The transmitter can be used with a 3-12 volt supply. The receiver requires a 5 volt supply. These are extremely simple devices. The transmitter simply switches the 433 MHz carrier on or off according to whether a high or low voltage is present on its data pin. They are intended for simple OOK (On-Off Keying) data transmission.

Pulse width encoding

However for reliable data transmission the carrier must be present for a high percentage of the time. The reason for this is that the receiver with no carrier detected will after a short interval increase its gain in an attempt to receive something. If there is no signal it will therefore increase its gain to a point where it receives radio frequency "static" and this will tend to result in false high logic levels appearing on the receiver's data pins. For this reason the transmitter must be used with a data encoding scheme that turns the carrier on for both low and high logic levels, but for different durations. Software must then be added to the receiver side to distinguish a 0 from a 1 by measuring the length of each pulse.

There is a very useful article by Roman Black (*reference 2*) in which he evaluates these cheap RF modules.

Roman Black describes how at higher data rates the rise time of the pulses as detected by the receiver become progressively later. However the trailing edges are relatively unaffected. This led him to conclude that to achieve higher data rates a pulse width encoding scheme based on the duration between the *falling edges* of the pulses should be used. This is the encoding scheme that we have adopted for the All Saints project.

Commercially available encoders and decoders

Inexpensive off-the-shelf encoders and decoders are commercially available for OOK devices. However, it seems that they only work at low data rates and add unacceptably high latency to the data signals, of up to 30ms in some cases. This latency is impractical for an application that is required to deliver the state of 11 sensors in real time. The probable reason that commercial products are unsuitable is that they typically incorporate sophisticated features such as encryption, error correction and receiver addressing. These features add a considerable overhead to the transmitted data and are not required by this application.

The PIC software

In his article Roman Black presents example programs for an encoder and decoder written in C for the Microchip 18F PIC microprocessor. The encoder function sends 1 byte of data each time it is called. The receiver function waits until 10 bytes of data have been received and stores this as a packet.

The All Saints software follows a similar principle but is written in assembly language rather than C and for a different PIC processor (i.e. the p16f628). These choices were based on the fact that 1) the author is familiar with the p16f628 instruction set and 2) his Velleman PIC programmer only supports devices that are programmable in assembler, not C. However, the p16f628 processor is very suitable for the application and is readily available on eBay for £2, including postage.

The All Saints software uses the p16f628's 4MHz internal oscillator. The 4MHz oscillator gives a clock speed of 1MHz.

The encoder

The encoder operates in an infinite loop continuously scanning ports RA0 to RA3 and RB0 to RB6 for data. This is done as quickly as the encoding scheme and the 1MHz clock speed will allow. The encoded data stream is written to port RB7 which in turn is connected to the data pin of the OOK transmitter.

The approximate pulse durations, as measured on a Rigol DS1052E digital oscilloscope, are 300uS for the start bit, 152uS for data zero and 200uS for data one. Since data is being transmitted continuously no data preamble is required. The receiver will set its gain to an optimal level for the strength of the received signal, and this level will remain constant.

During ringing the average latency will be 300 + (10 * 152) + 200 = 2ms. This assumes that at any instant only one bell is indicating "bottom dead centre" and that the "strike" signal is low (which is the normal case during a tied bell simulator practice[2]). This gives an average data rate of 5445 bits per second.

Notes:

2. During a practice session with the bell simulation software the clappers are tied so that the bells can make no sound. As each bell reaches bottom dead centre (mouth down) a pulse is sent to the computer. The computer then adds a configurable delay before playing a bell sound through the computer's speakers. A different note is played for each bell, thus simulating a real ring of bells but without disturbing the neighbours.

Ports RA0 to RA3 use Schmitt Trigger logic levels

A slight complication with the chosen PIC device is that ports RA0 to RA3 have so called Schmitt Trigger logic levels, i.e. a voltage of 0.8 x Vcc and above is used to represent logic level 1, whereas ports RB0 to RB6 use 2 volts or above to represent logic level 1. This necessitated the inclusion of an inexpensive CMOS buffer chip (MM74HCT244) between the optical sensors and ports RA0 to RA3. Only 4 of the buffers in the chip were used. The other 4 were tied to ground. This device costs around £1.

The decoder

The decoder does the reverse of the encoder. It “listens” for data on port RB7 which in turn is connected to the data pin of the receiver. It decodes each bit in the stream to produce the output on ports RA0 to RA3 and RB0 to RB6. I.e. the reverse of the encoder. This means that a signal set on port RA0 of the encoder will appear on port RA0 of the decoder, and so on for all the other inputs.

As with Roman Black’s software, a feature of the decoder is that it must process a start bit followed by a complete set of data bits before the state of the output pins is changed. In our case we expect a start bit to be followed by 11 data bits rather than a byte. If an invalid packet is received then we discard it and await the receipt of a new start bit. This prevents bad data randomly changing the state of the output pins and keeps the receiver in constant synchronisation with the transmitter.

Constructional points

The transmitter

Our solution relies on power being available in the belfry. As part of this project we also constructed a simple 1A, 5 volt DC regulated power supply. The power supply is connected to a small plastic “patch box”. The patch box has inputs for 10 optical sensors and 1 sound sensor. The inputs are standard stereo 3.5mm jacks, with power sent to the sensors on the plug tip. The other two connections are used for the return signal and ground.

Contained within the box are a) the PIC encoder, b) the buffer chip for bells 1-4, c) some additional circuitry for the sound sensor, d) the OOK transmitter and e) a ¼ wave “loaded” antenna. The antenna follows the design by Ben Schueler (*reference 3*).

The inputs for the optical sensors are pulled to ground potential using 10K resistors. This is to allow the unit to be used without all the sensors connected. However, this arrangement does result in the creation of a potential divider on each input, so lowering the available signal voltage. In retrospect weaker pull down resistors would have been more appropriate. However, in practice this is not a problem.

The receiver

The receiver uses an identical plastic box which contains a) the receiving antenna, b) the OOK receiver and c) the PIC decoder.

The design of the receiving antenna is simple but critical. In contrast to the transmitter‘s ¼ wave loaded antenna the receiver requires a simple coil antenna. The coil consists of 15 turns of 2.5mm PVC insulated solid core wire wound on a 3mm diameter former. A core from a standard UK lighting cable is suitable for this application. The coil is approximately 38 mm long.

The receiver also contains 4 AA batteries which provide 5 volts to the circuit via a low drop-out linear regulator. The unit can also be powered via a USB equipment socket, which when connected switches out the battery using a miniature relay.

The PIC processor outputs for the 10 optical sensors are connected to three female 9 pin DSUB sockets via 2.2K resistors. The resistors protect the PIC processor in the event that excessive voltage is present on the pins of the PC's serial ports.

Bells 1-8 use DSUB pins 1, 6, 8 and 9 on sockets 1 and 2. Bells 9 and 10 use DSUB pins 1 and 6 on socket number 3. The sound sensor signal is connected to pin 2 on each DSUB socket. This allows our custom odd struckness meter to receive a strike signal when connected to any of the DSUB sockets. The equipment ground is connected to pin 5.

Transmission range of the equipment

Both the transmitter and receiver are powered from a 5 volt DC regulated power supply.

The transmitter can in fact be powered using a higher voltage (up to 12 volts) and this should increase the transmitted power and therefore the range. However, provided the suggested antenna designs are followed and the ringing chamber is close to the belfry this should not be necessary.

Note however that the receiver only works well with the coil antenna described in the text. It will give extremely disappointing results with a ¼ wave antenna, regardless of the transmitter voltage used. Note also that if a significantly higher voltage is used for the transmitter (approaching 12 volts) then it will run quite hot and that is very likely to decrease its reliability. At 5 volts it runs cool and was found to have an impressive range[3].

Notes:

3. A test was conducted with the transmitter in the loft of the author's two storey house. A dummy sensor was plugged into one of the sensor inputs on the transmitter. The dummy sensor generated a pulse every 1.5 seconds which caused the corresponding indicator LED on the receiver to flash. Using the internal batteries of the receiver the author went "walk-about" and found that the signal was reliably decoded in every room in the house, including the extension which is separated from the main house by a cavity wall. The signal could also be reliably decoded outside the house over a distance of some 25-30 metres.

References

1. Abel bell ringing simulation program. Written by Chris Hughes. http://www.abelsim.co.uk/

2. Cheap RF modules made easy. An article by Roman Black. http://www.romanblack.com/RF/cheapRFmodules.htm

3. How to make a 433MHz (air cooled) antenna. An article by Ben Schueler. http://www.elektor.nl/Uploads/Forum/Posts/How-to-make-a-Air-Cooled-433MHz-antenna.pdf

Picture Gallery

Figure 1 - the original 8 optical sensors. Two more have since been added. These are clamped to the top of the cast iron bell frame. When the bells are down the sensors are so positioned that they are opposite reflective strips on the wheels.

Figure 2 - external views (from top left to right) of the 5 volt belfry power supply, sensor patch box/transmitter and receiver. The sound sensor (Electret microphone) is shown middle left. The custom odd struckness meter (white box) is shown at the bottom left. The DSUB sockets seen on the patch box/transmitter were used with the original communication cable but have since been made redundant by the internal transmitter.

Figure 3 - internal view of the patch box/transmitter. The PIC encoder is the 18 pin IC in the centre of the stripboard. Mounted above this is the MM74HCT244 octal buffer which is used for sensors 1-4 to raise the sensor signal levels for PIC ports RA0 to RA3. On the right-hand side of the stripboard is the additional circuitry for the sound sensor. This is composed of a simple 1 transistor amplifier and a 555 timer wired as a monostable. The potentiometer seen at the top is used to set the trigger volume for the sound sensor. The transmitter together with the loaded antenna is shown at the bottom. The antenna is encased in a plastic tube.

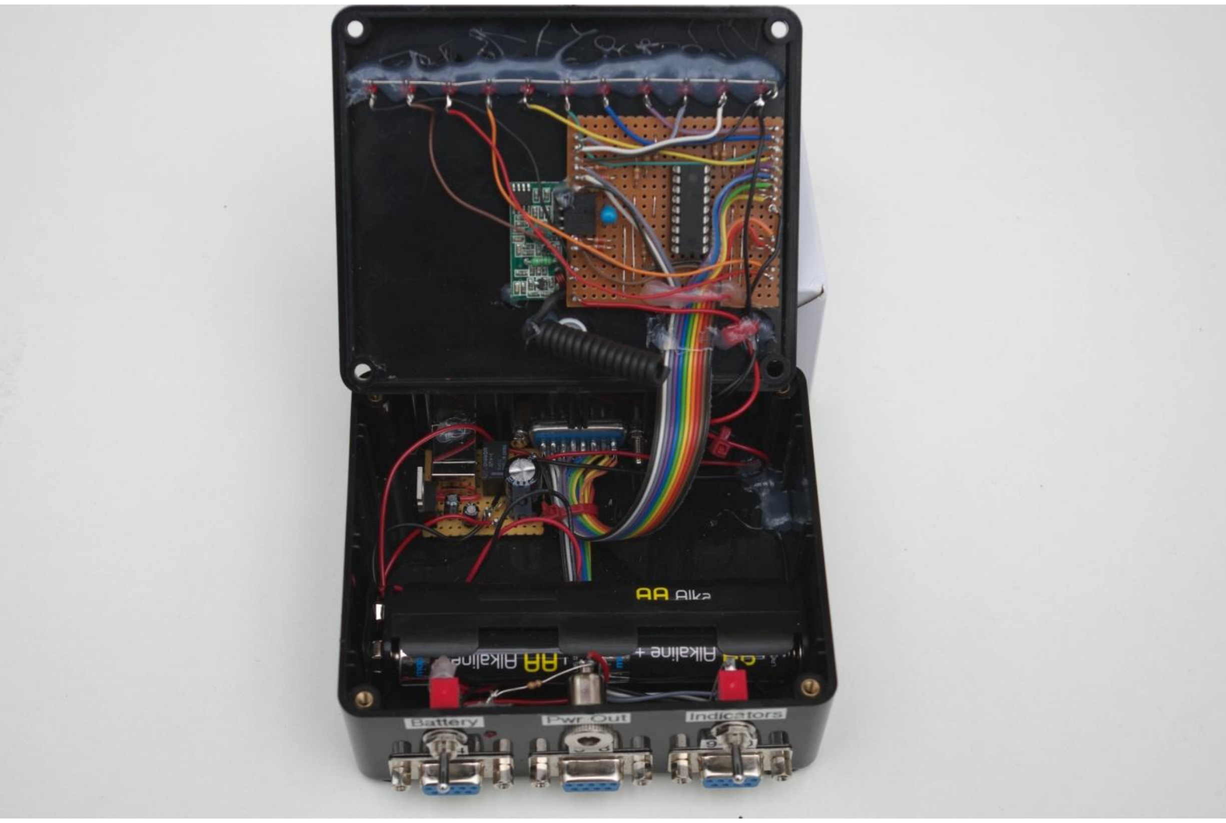

Figure 4 - internal view of the receiver. The 18 pin PIC decoder is shown in the middle of the stripboard. To the left of this is shown the OOK receiver board with coil antenna attached. Running along the top of the cabinet lid are 11 LED status lights, 10 for the optical sensors and 1 for the sound sensor. The ribbon cable connects the PIC outputs to a rear facing 15 pin DSUB socket and to three 9 pin front facing DSUB sockets. The 15 pin socket is for possible future use with a Raspberry-Pi. The 9 pin female DSUB sockets are used to connect the receiver to a PC. Also shown are the internal batteries and to the rear of the box the low drop out 5 volt regulator and relay. The left hand toggle switch is used to disconnect the battery from the regulator and the right hand toggle switch is used to switch off the status LEDs. The phantom power connection for the odd struckness meter is shown in the top middle of the front panel.