Laboratory for wireless remote control

Teacher notes and answers

Joshua S Curry and Denis A Nicole
Electronics and Computer Science
University of Southampton*

16th July, 2021

Table of Contents

1 Introduction....3
2 Choice of exercises....3
3 CyBOK relevance....3
4 Equipment used....4
5 Legalities....5
6 Software....5
7 Worked solutions....6
Task 1....6
1.1. Identify the relay board and inspect its components....6
1.2 Without inspecting the keyfob, calculate the operating frequency of the relay board....7
1.3 What is the operating frequency of the keyfob? Is this legal in the United Kingdom?....8
1.4 Arrange to supply the appropriate power to the board utilising the bench apparatus. What voltage is used?....8
1.5 Select appropriate resistors, and connect four LEDs so that they turn on when the corresponding relay is engaged, Show your circuit....8
1.6 Power the system, and check and record the current consumption. If necessary, follow the appropriate learning sequence, then check that relay "A" operates with the remote control....9
Task 2....9

* This project was funded under *Cybersecurity Body of Knowledge (CyBOK)* subaward 2021–2471.

2.1 Establish the best PC USB port to use with your SDR. Are some ports more or less noisy than others? Why?....9
2.2 What is the exact frequency on which the garage door opener transmits?....9
2.3 Do other traces appear when you push the button? Why?....10
2.4 Can you learn anything by listening to the audio output from SDR Console?....10
Task 3....10
3.1 How effective is this kind of attack?....10
3.2 What advantages and disadvantages do replay attacks have compared with fully reconstructing the signal?....10
3.3 Would this type of attack work on a wireless car key?....10
Task 4....10
4.1 What is the repeating unit of the signal? What is the bit period?.10
4.2 What is the duration of a single bit within a message?....11
4.2a How do signals differ between the four buttons?....11
4.3 Can you identify the encoder chip that is used?....11
4.4 Does the chip identity help you to modify your answer to 4.1?.....11
4.5 Can you modify the messages to operate button “B”?....12
4.6 Using the chip information, how many different ID codes are supported by this system?....12
4.7 How many repeats of the message are required to trigger a relay? 12
4.8 To what frequency is the receiver *actually* tuned?....12
4.9 If you were not able to capture a signal from a target system, what would be the maximum amount of time it would take to fuzz every ID code combination for a single relay channel, and thus open the door without a keypad?....12

1 Introduction

These notes aim to give you everything you need to run the lab, alongside setup instructions and model answers to accelerate the design and delivery of teaching.

These notes are not intended as a step-by-step guide and instructors will need a clear knowledge of the surrounding area to deliver this content. In addition o this, whilst information on legal processes is correct to the best of the authors' knowledge at the time of writing, it does not constitute legal advice nor is permission to operate any transmitting or receiving equipment.

2 Choice of exercises

As presented, this is a long laboratory which proceeds to quite an advanced level. We have left it up to individual teachers to determine the specific subset of the exercises that will meet their desired learning outcomes under the CyBOK syllabus. Specifically, in addition to the over-the-air analysis at the core of the laboratory, there are learning points that relate to:

- visual inspection: additional information can be gleaned if opening of the fob is permitted,
- internet search: there is a great deal of information available on the WWW, including complete protocol information for the EV1527 one-time-programmed encoder.
- protocol analysis and "fuzzing": it may be appropriate to use this laboratory to reinforce transferable skills which can be applied in, for example, internet, wired networking, or near field applications.

3 CyBOK relevance

This laboratory supports HEI educators in developing student understanding of the CyBOK *Physical Layer & Telecommunications Security Knowledge Area*, in particular aspect 3.4: *Attacks on Physical Layer Identification.* It will give low-cost practical hands-on training in signal replay, feature replay, and "fuzzing" attacks. It has previously been used in Southampton's FHEQ level six, 7.5 ECTS/15 CATS *Security of Cyber-Physical Systems* module which forms part of our GCHQ/NCSC-accredited MEng and MSc degrees; it was specifically included in our accreditation documentation.

4 Equipment used

We assume that each team (one or two students) has the dedicated use of a conventional x64 *Windows 10* machine with a USB3 port. Each team will also need a *LimeSDR Mini* DSP "dongle" costing around $200 and one of the generic *GV-RK04S-12* remote control modules operating in the 433MHz band. The keyfob that comes with these units contains an EV1527 *one-time-programmed* encoder **but this information should not initially be revealed to the students**. A set consisting of fob and four-relay receiver is readily available from "far east" sources for around £16. Teams will also need to be provided with an appropriate 12V current-limited supply along with four LEDs and series resistors so that the state of each relay can be observed.

Opening of the fobs is a potentially destructive activity and, if permitted, budget will need to be set aside for their replacement. Furthermore, they each contain two CR2016 lithium batteries **which pose a potentially lethal risk to small children if ingested**.

The laboratory may be enhanced by modifying the keyfobs in advance to prevent the use of keys other than "**A**". You might choose to create a "backstory" in which general users have keyfobs which will open, say, a kitchen door but only senior management has the special keyfob that lets them into the "executive washroom". Your mission *should you choose to accept it* is to open the executive washroom door, when given only the ordinary use keyfob.

Opening of the relay board case is essential for making connections and to see the receive aerial; most boards do not divulge protocol information on visual inspection.

Typical boards come pre-learnt to accept the paired keyfob, but the supplied data sheet gives the appropriate instructions if necessary. We recommend that boards and fobs be supplied to students in a paired configuration; this gives them rapid assurance that the "kit" is functional.

As each fob has a different identity, many groups can work in the same laboratory; they will, however, need to take care when using *SDR Console* or *Universal Radio Hacker* not to be confused by signals from other fobs.

We have not had problems with ESD damage to the SDR or other components, but some warnings should be given.

Finally, one or two aerials will be required for each SDR. Our preference is for the students to build themselves simple $\lambda/4$ monopoles by soldering a short

wire to the centre pin of an SMA plug. This will provide additional learning opportunities for RF and for risk assessment; they will need to take into account potential electric shock, eye protection, burn, and respiratory hazards. For the latter, you **must** control fumes from soldering flux using appropriate extractors. Will they use lead-free solder? If so, what iron temperature is required? What is the appropriate size of iron? Can they check the legalities of using leaded solder? How will they hold the plug and wire safely? Do not let them hold it in their fingers or solder while the plug is in position on the SDR! Make sure that the provided plug is not "reverse SMA" nor designed for "semi-rigid" cable; it should have a freely rotating female-threaded collar and an actual centre pin.

It is in principle possible to operate this laboratory remotely, either by lending the necessary equipment to individual students at home (soldering should be avoided) or, if absolutely necessary, by allowing the students to follow in a video session an instructor performing the required laboratory work. The appropriate setup for home working will depend on the details of other equipment (laptop, power supply, aerials) which has been supplied, and on the nature of the risk assessments that have been conducted. These latter should be formulated in cooperation with the students; they form important learning points.

5 Legalities

We will be transmitting and listening only within the 433.05–434.79MHz license-free band for short range devices. We can only lawfully listen to transmissions in this band that are intended for us; our use of keyfobs clearly meets this requirement. Transmit power in this band is limited to 10mW e.r.p. on a 10% duty cycle. The chosen SDR can only deliver 11.8dBm (just over 10mW) at 437MHz and we are using an inefficient low gain antenna, so this requirement should easily be met. We take on trust that the keyfob is not over-powered. Finally, at these power levels, there is no hazard from non-ionising radiation emitted by either our SDR or keyfob.

While laboratory use of this lab is, we think, fully compliant, the setup might not meet requirements for *electromagnetic compatibility* in other situations; if deployment elsewhere, for example at home or in vehicles, is intended, further checks will have to be performed.

6 Software

We use *SDR Console* and *Universal Radio Hacker* within this laboratory.

SDR Console is not essential, but it offers a very comfortable and high performance conventional radio interface to the SDR; it is easy, for example, to receive Band II FM broadcasts. The license explicitly permits educational use:

> *This software is available free of charge for use in schools, colleges, training facilities.*

Universal Radio Hacker is open source software under the *GNU General Public License v3.0.* Like *Gnu Radio Companion*, it makes extensive use of the *PyQt5* framework. GRC shows some signs of instability on Windows, but URH appears to be stable. There is a user manual. a Wiki and several training videos. Demonstrators will need to familiarise themselves with URH; it is designed around a potentially non-intuitive workflow.

Finally, up-to-date FTDI drivers for the FTD3XX series are needed for support of the LimeSDR Mini. The installation of these, and the standard installers for the other software, will require *Administrator* privileges on the PCs.

While we have designed the laboratory around Windows PCs, it is quite reasonable to use Linux-based systems as an alternative if preferred.

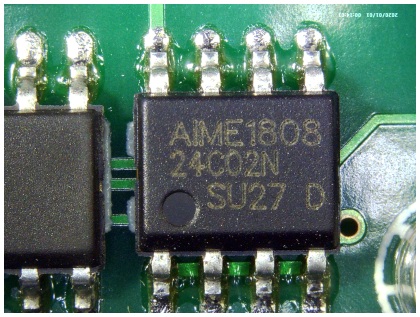
7 Worked solutions

Task 1

1.1. Identify the relay board and inspect its components.

There are no useful markings on the top surface of the board. The eight pin chip is marked

```
AME1808
24C02N
SU27 D
```



Relay board: only identifiable IC.

The 24C02N SU27 is a 256 x 8 bit serial EEPROM, presumably used to hold the learned configuration. The other chip is not marked.

The lower surface carries the year 2015 and the text

```
MODEK TME:2015-9-23 II
GV-RK04S-12
```

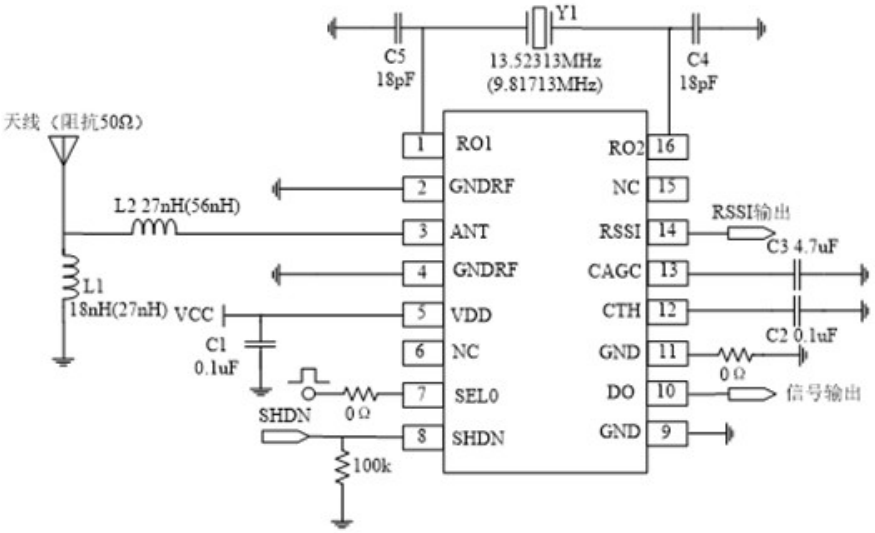
There is an RF receiver module, with another unmarked chip, but carrying a two-pin metallic device labelled 6.7458.

The branding is generic but, from web searches, a likely brand is INSMA.

1.2 Without inspecting the keyfob, calculate the operating frequency of the relay board.

The antenna is coiled, but presumably amounts in length to approximately a quarter wave end-fed monopole. It is ten turns on a 5mm diameter, with a further 20mm of connection, making for a total length of 177mm. This corresponds to a quarter wave at 423MHz. If we take it that the metallic device is a crystal at 6.7458MHz, this is close to 64 times the crystal frequency, a likely binary multiplier. That would give a multiplied frequency of 431.73MHz.

The RF module closely resembles the QAM-RX10-433 433.92MHz 10Kbps AM receiver module from QUASAR UK which operates on 433.92MHz. This is a super-hetrodyne receiver, so the local oscillator frequency is likely to be displaced from the receive frequency. Most similar receivers found on the WWW have this device marked `WL433`. Unhelpfully, a schematic found on the WWW for a 433.92MHz receiver has a crystal frequency of 13.52313MHz, which would multiply by 32 to give 432.74MHz.



Possible receiver module schematic [from WWW site].

It is a bit inconclusive, and rather frustrating that the two multiplied crystal frequencies differ by almost exactly 1MHz. We might guess the frequency of operation to be either 433.92MHz, because of its popularity in web searches, or 432.92MHz based on the 1MHz difference on local oscillator frequencies. The latter would be outside the UK licence-free band. Perhaps the receiver has actually been built with the wrong crystal? The claimed receive bandwidth is 300kHz, so it will not work well if it has!

1.3 What is the operating frequency of the keyfob? Is this legal in the United Kingdom?

It is clearly marked 433.92MHz, a popular frequency in a licence-free band. Assuming the output power is low enough, it is fine.

1.4 Arrange to supply the appropriate power to the board utilising the bench apparatus. What voltage is used?

12V.

1.5 Select appropriate resistors, and connect four LEDs so that they turn on when the corresponding relay is engaged, Show your circuit.

The LEDs drop about 2V and should receive around 20mA, so a series resistance of (12-2)/0.02 = 500Ω. would be appropriate. The nearest *preferred*

value is 470Ω. Thus, connect a 470Ω resistor in series with each LED and attach the pair **with the correct polarity** between each pin A and ground (-12V). Connect each pin B to +12V. Use sleeving as necessary to insulate your connections.

1.6 Power the system, and check and record the current consumption. If necessary, follow the appropriate learning sequence, then check that relay "A" operates with the remote control.

About 6mA should be drawn with no relays engaged or lights on. About 70mA with one relay engaged and one light lit.

Task 2

2.1 Establish the best PC USB port to use with your SDR. Are some ports more or less noisy than others? Why?

There are several possible issues with USB ports.

- If the port is supplied via an *unpowered hub*, it might not deliver a reliable *low-noise* 5V at sufficient current to poere the LimeSDR Mini.
- As we are using a monopole aerial with no proper *ground plane*, the SDR is relying on the USB connection to provide an effective ground. This might not work well and may introduce noise.
- The physical location of the SDR and its aerial will affect local noise pickup.
- The SDR depends for its performance on a fast, ideally USB3, port.

2.2 What is the exact frequency on which the garage door opener transmits?

This can be measured in SDR Console ot using the *Spectrum Analyzer* feature of Universal Radio Hacker.

It is indeed 433.92MHz.

2.3 Do other traces appear when you push the button? Why?

Yes. Either the keyfob is transmitting a lot of *spurious* signals, or the SDR or software is being overloaded; it is probably the latter. Note that, in the Universal Radio Hacker display, the peak exactly at mid band is a typical artifact of an I/Q sampling SDR with simple software. It is the reason we offset the centre frequency from the transmitter.

2.4 Can you learn anything by listening to the audio output from SDR Console?

Yes. There is a clear tone when AM demodulation is selected and the key is pressed. This indicates *amplitude modulation* which will probably be *on-off keying*, the crudest way to impart information onto a signal. The tone is around 700Hz, which might be an indication of the bit rate.

Task 3

3.1 How effective is this kind of attack?

It's fine if you have a good clean recording from a simple (not rotating key) transmitter, and you only want to repeat the original action.

3.2 What advantages and disadvantages do replay attacks have compared with fully reconstructing the signal?

Replay is quick and easy; it will not work with rotating keys, and it on;y allowes you to repeat the recorded action.

3.3 Would this type of attack work on a wireless car key?

Probably not. These often maintain state between the key and car, and expect a new different *rotating* code to be used each time.

Task 4

4.1 What is the repeating unit of the signal? What is the bit period?

It is likely that the automatic decoding will report something like

```
11101110111011101110100010001110100010001000100010001110111010001
11010001110111010001000100011101<22320 sample gap>
```

This is, however, subtly wrong, see below.

4.2 What is the duration of a single bit within a message?

A single `1` or `0` has a duration of about 720 samples = 310µs. This is not the data rate; as we shall see later, each bit of data is encoded into four signal bits.

4.2a How do signals differ between the four buttons?

Students will not be able to answer this yet if the other buttons have been disabled. If not, Button **B** should yield something like

```
1110111011101110111010001000111010001000100010001000111011110100011101000111011100010001110100011<22320 sample gap>
```

4.3 Can you identify the encoder chip that is used?

Datasheets for the popular encoder chips are readily available on the WWW and likely candidates can easily be found with a web search. Students should be able to find PDFs for the EV1527, HT640, and PT2262. Comparison with the captured messages makes it clear that our data is from a *one-time-programmed* EV1527. If students are having trouble with this, you might let them open the keyfob and inspect the chip inside. Alternatively, we have also provided a photograph of the keyfob chip.

4.4 Does the chip identity help you to modify your answer to 4.1?

Inspection of the EV1547 data reveals that the message actually has a 32 bit **preamble***;* the decoded gap should be at the start of the message, just after an initial **1**. So, correctly decoded, a key **A** message might be

```
10000000000000000000000000000001110111011101110111010001000111010001000100010001000111011101000111010001110111010001000100010001110
```

with no gap. The fixed synchronisation preamble is

```
10000000000000000000000000000000
```

which is followed by twenty-four bits of data, with each data **1** encoded as `1110` and each data **0** encoded as `1000`. Thus the data content of the message above is

```
1111 1001 0000 0110 1011 0001
```

Of these twenty-four bits, the first twenty are the individual code of the keyfob; the last four select one of the four relays.

It would be wise at this stage for the student to correct the decoding, perhaps by loading a text file with the correct bit pattern into the *Generator* tab and dragging it over to the right hand pane.

4.5 Can you modify the messages to operate button "B"?

We change the last four data bits from `0001` to `0010`, which, after encoding, changes the end of the message from `1000100010001110` to `1000100011101000`.

4.6 Using the chip information, how many different ID codes are supported by this system?

1048575, about a million.

4.7 How many repeats of the message are required to trigger a relay?

Careful checking by sending a sequence of messages using the *Generator* confirms that the receiver is triggered only by at least two consecutive messages.

4.8 To what frequency is the receiver *actually* tuned?

The student can determine this by using the *Generator* to vary the transmission frequency. This is a rather crude test as we are not rigorously controlling the RF path. Nevertheless, we were able to determine quickly that the relay board responds to signals in the range 433.3MHz to 434.7MHz, suggesting a centre frequency of 434.0MHz, close to the expected 433.92MHz. So the receiver is **not** set to the wrong frequency! We could do a better job here by plotting a graph of Tx gain against frequency for marginal operation.

4.9 If you were not able to capture a signal from a target system, what would be the maximum amount of time it would take to fuzz every ID code combination for a single relay channel, and thus open the door without a keypad?

With a small external encoding program, the Universal Radio Hacker could do this "fuzzing" for the student. Naïvely, each test signal should take

(32 + 24*4) * 720 sample times, which is 46ms at a 2M samples/s rate. Careful checking by sending a sequence of messages using the *Generator* will, however, confirm that the receiver is triggered only by **at least two** consecutive correct messages positioned between incorrect ones. So, a single code test will take 92ms. We can get through all of the million codes in 25.6 hours. Cracking a single unit will, on average, take about 12 hours. We are likely to be able to do it in a single "night" from 17:00 to 09:00.

In short, the coding should effectively prevent unwanted accidental interactions, but it provides **no** security against replay attacks and little security against exhaustive cracking.