THE BREZHNEVFONE

ROGER KOENKER

1. Introduction

Inspired by a blog post of Alasdair Allan `https://bit.ly/3bi7XvQ` I set out in late 2019 to try to reproduce a version of this for my wife's desk. She is Director of the School of Slavonic and East European Studies at UCL so it seemed appropriate that she should have a red phone to call the Kremlin. This got off the ground slowly. I bought a GPO 746 like Alasdair's, and even in its original totally non-functional state it was widely admired by visitors to her office. I also obtained a AIY Google Voice Bonnet (V2) with pi0 on a trip to the U.S. for Christmas.

2. Hardware

Assembly of the Voice kit went smoothly and it ran the demos provided as expected. Things stalled for a while, but Covid-19 confinement policies provided an inducement to bring the phone back home and to get started in earnest with the project. So I disassembled it and first tested that I could reroute the speaker wires from the V2 to the handset of the phone; this worked fine, which created a false sense of accomplishment. It is perhaps worth noting at this point that the basic setup of the kit and the linux aspects of the software installation were familiar enough to me. But the next tasks involved rewiring and this was totally foreign territory. I hadn't held a soldering iron since assembling a Heath Kit FM radio in high school in the early 1960s. On the other hand that was the point of the venture – to learn something new. And after all how complicated could it be to reroute a couple of switches?

The next task seemed to be to replace the button of the V2 with some sort of connection to the dialer mechanism of the phone. At this point I realized first, that the dialer mechanism on my GPO746 was different from Alasdair's, and second that the V2 button was also different than his V1 button. The dialer problem seemed benign, it seemed that Alasdair's strategy of ignoring the mechanism except for the micro-switch that indicated the initiation of dialing should work. I'm still wondering about whether my version of the phone might produce a digital output of the dialing since the terminals on the ribbon connector are labeled C1-4 and R1-4 like a keypad, but this is for much later, aka probably never. The dialer mechanism is illustrated in Figure 1.

So I began futzing around with replacing the button. When my initial attempts to imitate Alastair's V1 solution failed, I resorted to trying to imitate other wiring schemes for other 3 leds.

May 26, 2021.

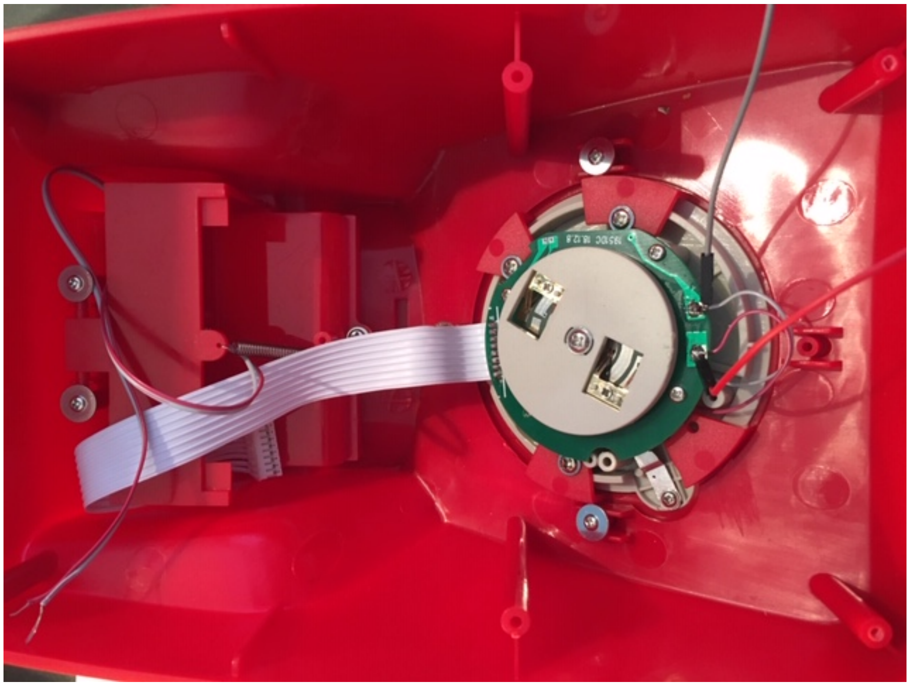
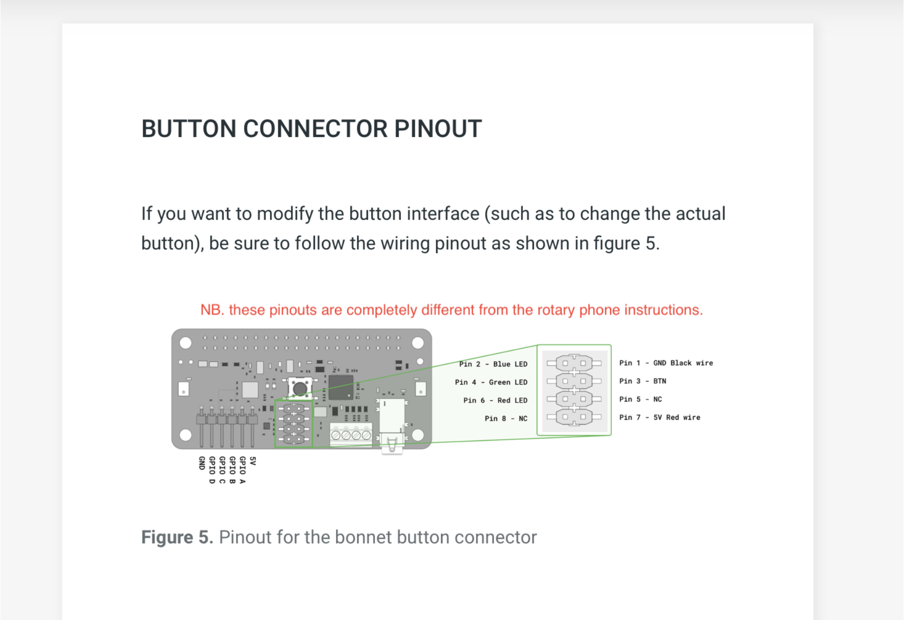


Figure 1. The dialer mechanism of my GPO746 with two leads soldered to the terminals of the micro-switch.



BUTTON CONNECTOR PINOUT

If you want to modify the button interface (such as to change the actual button), be sure to follow the wiring pinout as shown in figure 5.

NB. these pinouts are completely different from the rotary phone instructions.

Figure 5. Pinout for the bonnet button connector

Figure 2. Pinouts for AIY Google Voice Bonnet V2

Of course I had no idea what sort of led setup exists in the V2 button, although having run the AIY demo **leds_example.py** there is obviously some fancy stuff that potentially can go on in there: RGB colors can be mixed, dimmed etc etc. The pinouts for the button harness on the V2 are illustrated in Figure 2

Eventually I decided to leave the V2 button connected to the bonnet but experiment with connecting button harness to the dialer mechanism. Through some experimentation I discovered

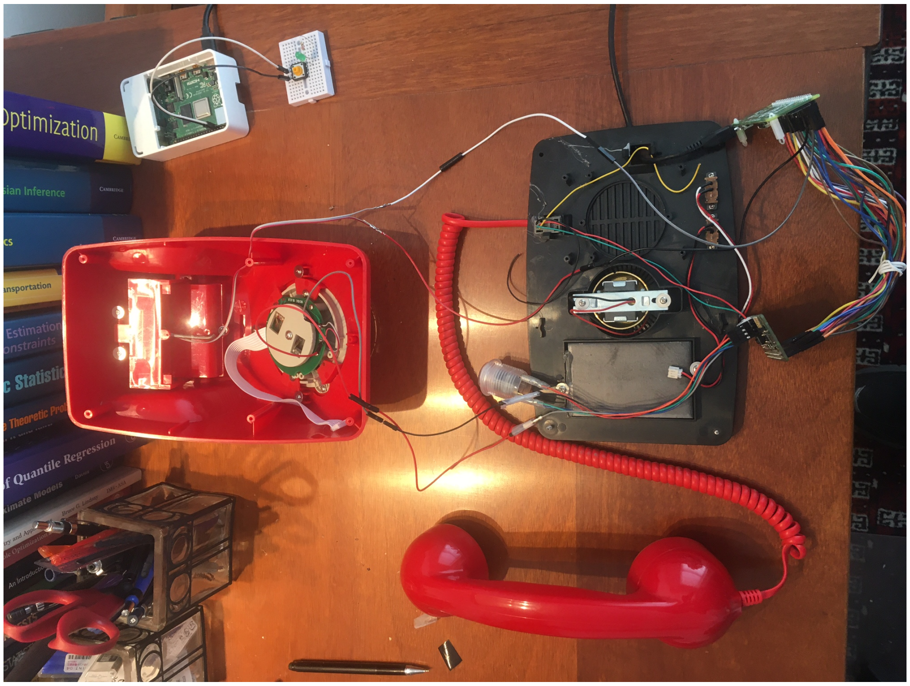


Figure 3. Connection for phone to Voice Bonnet V2

that connecting the GND and BTN to the dialer allowed me to mimic the functionality of the button press by instead dialing any digit on the phone. This seemed to be a major breakthrough, since dialing allowed me to interact with the google assistant. Now all that remained was to figure out how to connect the switch associated with whether the handset was on or off the hook.

This problem absorbed several months. After some interaction with the Google AIY projects-raspbian help pages, it became clear that my original plan to use the gpio pins on the voice bonnet board was doomed to failure. For further details see: `https://github.com/google/aiyprojects-raspbian/issues/709`. These pins seemed unable to respond to input, so I pulled apart the two boards and jumper'd together the pins that the bonnet needed according to the latest pinout on github. This too failed, but there were several pinouts available, and eventually by using the union of the connections indicated by the various pinouts, the functionality of the bonnet was restored and I was able to use two free pins on the rpi0 to make the phone hook switch also work. This is illustrated in Figure 3. p So it now seemed that all the hardware work was done and it is *just* a matter of software. I folded the boards and the mess of wiring into the phone and reattached the base to the body of the phone.

3. Software

The natural starting point for the software development was to merge the demo for the voicekit with the python script of Allen. The first step was to see if I could introduce a dialtone for the phone. I found an aiff file that seemed appropriate converted it to wav format in audacity, and stuck it into the google script at the point that the conversation is just starting. This was not enirely satisfactory since there is some delay before the dialtone is heard, but it does work. Similarly, I added another wav file to simulate three rings as if waiting for the phone to be answered. The next step was to see if I could get it to say something. I've now added an import of tts, and this allows it to say good morning/afternoon/evening depending upon the time of day.

Allen's python script used threading to manage the connection to the google assistant, but I found this too obscure and eventually realized that it was unnecessary. My revised script simply waits for the handset to be lifted, initiates the dial tone, the ringing and the greeting. It then invokes the google assistant and conducts a conversation in which each question requires dialing a 1 (or any other number) to simulate the button press in the original setup of the voice bonnet. Saying “goodbye” to the assistant provokes a response like “until next time” from the assistant, and replacing the handset on the hook terminates the program.

4. Envoi

Inevitably, there are a few loose ends that may, or may not, be attended to before the phone goes back to UCL. I would like to have an option to conduct the conversation in Russian. There seems to be some facility for doing this, but I’ve not explored it fully at this point. It is still somewhat unsatisfactory that the dialtone and ringing are of fixed length, but making them responsive to say dialing, would require threading or some other python tricks that I’m not really prepared to investigate at this point. For the moment the functionality of the phone is controlled via ssh

Appendix A. Python Code

```python
#!/usr/bin/env python3

import argparse
import locale
import logging
import signal
import sys
import time

from signal import pause
from gpiozero import Button
from aiy.voice.audio import play_wav
from aiy.voice import tts
from time import strftime
from gpiozero import Button
from aiy.assistant.grpc import AssistantServiceClientWithLed
from aiy.board import Board
from google.assistant.library.event import EventType

def volume(string):
    value = int(string)
    if value < 0 or value > 100:
        raise argparse.ArgumentTypeError('Volume must be in [0...100] range.')
    return value

def locale_language():
    language, _ = locale.getdefaultlocale()
    return language

def main():
    logging.basicConfig(level=logging.INFO)
    signal.signal(signal.SIGTERM, lambda signum, frame: sys.exit(0))

    parser = argparse.ArgumentParser(description='Assistant service example.')
    parser.add_argument('--language', default=locale_language())
    parser.add_argument('--volume', type=volume, default=100)
    args = parser.parse_args()
    button = Button(4)
    Hookup = 1;

    with Board() as board:
        assistant = AssistantServiceClientWithLed(board=board,
                                                  volume_percentage=args.volume,
                                                  language_code=args.language)
        while Hookup:
            button.wait_for_press()
            print("The␣button␣was␣pressed!")
            hour = strftime("%H")
            play_wav("waves/DialTone2.wav")
            play_wav("waves/3rings.wav")
            if hour < "12":
                tts.say('Hello,␣Good␣morning')
            elif hour < "18":
                tts.say('Hello,␣Good␣afternoon')
            elif hour < "24":
                tts.say('Hello,␣Good␣evening')
            logging.info('Press␣button␣to␣start␣conversation...')
            board.button.wait_for_press()
            logging.info('Conversation␣started')
            assistant.conversation()
            button.wait_for_release()
            print("The␣button␣was␣released!")
            Hookup = 0;

if __name__ == '__main__':
    main()
```