```
#!/usr/bin/python3

#     A 'shim' to place between Gpredict and Hamlib, when using the FT-736
#     so as to "fake" VFO status to keep Gpredict happy.
#     Also, handle turning the radio's CAT features ON and OFF, so that
#     when the user "disengages" the radio in Gpredict, they can change
#     mode or whatever manually, without power cycling the radio, but will
#     re-enable CAT when the radio is next "Engaged" in Gpredict.

#     THIS IS PURELY FOR THE YAESU FT-736R NO OTHER RADIO.
#     G8KBV March 2018

import socket

def Main():

#     Local temporary string storage.
      subfreq = "435000000"
      mainfreq = "145000000"

#     GPredict side host:port
      gphost = 'localhost'
      gpport = 50000

#     HamLib side host:port
      hlhost = 'localhost'
      hlport = 50736

#     Create a Server socket for GPredict to connect to as a client.
      gpSocket = socket.socket()
      gpSocket.bind((gphost,gpport))
      gpSocket.listen(3)
      print ("Listening on port 50000")

#     Create a Client socket, to connect to the HamLib server.
#     NOTE!  rigctld MUST be started before this script is run!
#     Like this...
#     $ rigctld -m110 -r /dev/ttyUSB4 -s 4800 -T localhost -t 50736


      while True:
            gpconn, gpaddr = gpSocket.accept()
            print ("Connection from: " + str(gpaddr).rstrip())

            # The Hamlib backend will automatically have done this, but it
            # will be needed if we turn CAT off, when Gpredict "Disengages"
            # and we later "Engage" the radio.

            hlSocket = socket.socket()
            hlSocket.connect((hlhost,hlport))
            print ("Linked to rigctld")

            while True:
                        gpcmd = gpconn.recv(1024).decode()
                        if not gpcmd:
                                    break
#                       print ("From Gpredict: " + gpcmd.rstrip())

                        if gpcmd[0] == "F":
                              # Set Main frequency: Remember the value.
                              mainfreq = (gpcmd[1:]).lstrip()
#                             print (mainfreq)
                              # code here to send to Hamlib
                              hlSocket.sendall(gpcmd.encode())
```

```
                              reply = hlSocket.recv(1024).decode()
#                             print ("To   Gpredict: " + reply.rstrip())
                              gpconn.sendall(reply.encode())

                        elif gpcmd[0] == "I":
                              # Set Sub frequency: Remember the value.
                              subfreq = (gpcmd[1:]).lstrip()
                              # code here to send to Hamlib
                              hlSocket.sendall(gpcmd.encode())
                              reply = hlSocket.recv(1024).decode()
#                             print ("To   Gpredict: " + reply.rstrip())
                              gpconn.sendall(reply.encode())

                        elif gpcmd[0] == "f":
                              # fake read of the main frequency
                              reply = mainfreq + "\n"
#                             print ("To   Gpredict: " + reply.rstrip())
                              gpconn.sendall(reply.encode())

                        elif gpcmd[0] == "i":
                              # fake read of the sub frequency
                              reply = subfreq + "\n"
#                             print ("To   Gpredict: " + reply.rstrip())
                              gpconn.sendall(reply.encode())

                        elif gpcmd[0] == "q":
                              print ("Unlinking from rigctld")
#                             hlcmd = "w \\0x00\\0x00\\0x00\\0x00\\0x80\n"
                              hlSocket.sendall(gpcmd.encode())

                        else:
                              hlSocket.sendall(gpcmd.encode())
                              reply = hlSocket.recv(1024).decode()
#                             print ("To   Gpredict: " + reply.rstrip())
                              gpconn.sendall(reply.encode())

            gpconn.close()


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