```
<?xml version='1.0' encoding='UTF-8'?>

<!--

                        RigCAT control for the
                           Kenwood TM-D710G

                                K5URU


Hello friends,

I have tried my best to include notations about how this whole thing works.
I hope you can learn as much from this as I did from all the other xml
files I studied while trying to complete this project.  If you have any
questions about how or why this xml file does what it does, please feel
free to contact me directly, on QRZ, or through the fldigi Yahoo groups.

I'm certainly not an expert, but I'd be happy to lend a hand if you get stuck
while trying to build or customize your own xml file.  For a true wealth of
knowledge, go on over to the Yahoo Groups and sign up with a bunch of them!
Make sure to read the documentation on http://www.w1hkj.com/ as well.  There's
no way this document would exist if it weren't for the help of countless
elmers, friends, strangers, pioneers, and rookies like me, asking all kinds of
questions and sharing the results.  I'm proud to be a part of this community.

de K5URU
Blake Ormand
Blake.Ormand@gmail.com



Before we get started...

Kenwood likes to call the left and right "sides" of their dual-band (dual-VFO)
mobile radios as the "A-Band" and "B-Band," which can be a little confusing.
In this document, I'll stick with the Kenwood nomenclature for the sake of
consistency, and call the left side of the radio the "A-Band," and the right
side of the radio the "B-Band."

There is no particular reason any TM-D710G owner would need to choose one or
the other (other than frequency considerations, but that's a different topic).
For reasons not completely clear to me, I use the B-Band as my digital side,
so that's how this entire xml file is set up.  If you would like to modify
this to use the A-Band, you can probably do so after reading through the notes
in this file.  If you need any help, just ask!



DISCLAIMER

This xml file was developed and tested on my Kenwood TM-D710GA radio.
I cannot guarantee its performance or functionality on any other radio,
including the TM-D700 series, TM-D710 series, or ANY other similar model.
```

But they'll probably work... ;)

While you're getting all this set up, keep in mind some best practices for amateur radio (whether experimental or traditional):

1. Double check your radio. Then double-check it again.
2. Do some research on your local band plans, and/or contact your regional frequency coordinators to find the digital segments.
3. Use a dummy load and/or low power until you're confident in your setup.
4. Make sure your testing frequencies aren't in use. Listen!
5. Be prepared to quickly turn off your radio if something goes wrong. It happens. Then, make some adjustments, turn it back on, and try again.
6. Keep your test messages short, and always include your call sign.
7. Don't be afraid to do a quick voice ID, even if (especially if) something goes haywire. I actually got some unexpected help from a local ham who happened to stumble across some of my failed test transmissions while he was scanning through the digital segments. If I hadn't transmitted my call sign, I wouldn't have "discovered" this source of invaluable advice. That's what amateur radio is all about!

I hope my fellow operators (Kenwood or otherwise) will be as pleased as I am with this little functional explanation of how to generate a custom xml file for fldigi/RigCAT. I look forward to working together with you to supplement and improve this document, as well as the art and science of radio.

de K5URU

OKAY! Let's get down to business!

The Kenwood TM-D710G has some limitations when it's being controlled this way.

****
The PTT and CTRL indicators MUST be on the B-Band at all times during all active digital mode work. If either the PTT or CTRL indicators are switched to the A-Band, you may very well transmit digital mode signals to whatever frequency the A-Band is tuned to, including your local voice mode repeater!

As a precaution, during testing, I usually set both the A-Band and B-Band to VFO mode, and set them to the same frequency. This way I can feel comfortable with my experiments, yet still have a clear indication of their technical success or failure.

Be COURTEOUS and Be CONSIDERATE!!!
****

NOTES (specific to THIS xml file):

1. The "B-Band" (right side) of the TM-D710G will be used for all data transmissions/receptions and all fldigi interactions. The initialization

procedure in this xml file will set both PTT and CTRL to the B-Band, then set the B-Band to VFO mode. During your digital session, you can certainly switch over to the A-Band to use it as a voice radio, but remember that (at this point), fldigi won't know that you switched. Be SURE to switch back to the B-Band (PTT and CTRL) when you're done.

2. The radio will receive and transmit only in FM mode. (Will not work in NFM nor AM modes).
3. The radio will only work in simplex, with no PL tones, and no offsets.
4. The following settings will always apply to the B-Band VFO when using this xml file:
   A. Frequency step size will always be set to 5 kHz.
   B. The radio will not respond to fldigi VFO selections in between the 5 kHz steps.
   C. Shift direction will always be set to None, and Offset amount will be set to zero.
   D. Tone, CTCSS, DCS, and Reverse functions will be disabled.
   E. The B-Band VFO settings for Tone/CTCSS frequencies and DCS codes, will be set to the lowest available values (although they won't apply anyway while using this xml due to Note 4D above.)
   F. None of these items will affect your stored memories.

I also use the terms fldigi and RigCAT interchangeably, because RigCAT is a part of fldigi. Basically, fldigi generates the audio signals, and uses RigCAT to send them to the radio. On receive, the radio uses RigCAT to get the signals to fldigi for interpretation. There's much more to it than that, but in this paper, I'll be focusing on getting all the moving parts to work together to create a powerful and quite enjoyable digital radio experience. Advanced users may notice that I refer to fldigi/RigCAT as fldigi for the rest of this document.

CABLES:

I use the authentic Kenwood PG-5H cables for my regular operations. There are certainly cheaper alternatives out there (including DIY), but after a few failed attempts and one ugly but successful attempt at building my own, I finally broke down and bought the brand name cables. So it goes...

Here's how I have my cables hooked up while using fldigi and this xml file:

The DIN-6 adapter goes from the back of the main unit into your computer's mic input and line out as directed in the manual. The DIN-8 cable with the RS232 connector is a little harder to figure out. For the purposes of THIS file, and for using your D710G for digital modes with fldigi, plug the cable into the back of the MAIN radio unit (the black box), NOT the back of the control head with the screen. We're using the radio's PC Port, not its COM Port.

I've been using this new system for a few weeks now, and I must say, it's really nice to be able to use one half of my radio for digital, and have the other half available for monitoring my favorite repeater. In fact, if you're just getting started in digital modes, it's really helpful (and fun!) to have a voice link to the other party while you both work on the digital side.

```
Okay, here we go, jump right in...

-->



<!--
 Rig definition file for use with fldigi
-->
<RIGDEF>
<!-- Rig definition file for the following transceiver -->
  <RIG>Kenwood TM-D710G</RIG>

<!-- Programmer and status definitions -->
  <PROGRAMMER>Blake Ormand K5URU - Tested by: Blake Ormand K5URU</PROGRAMMER>
  <STATUS> Alpha Version: 1.5 Date: 21 January 2016 </STATUS>


<!-- If you test this on your TM-D710G or any other related model, I'd sure
appreciate your feedback, good or bad. Thanks! -->


<!--  Title for the RigCAT dialog box used for rig control in fldigi -->
  <TITLE>Rig CAT - TM-D710G</TITLE>

<!-- Initialize.  This happens when either the initialize button is pressed,
  or, once fldigi is configured, whenever fldigi starts up.  -->
<COMMAND>
  <SYMBOL>INIT</SYMBOL>
  <SIZE>14</SIZE>
  <!-- Set PTT and CTRL to B-Band -->
  <!-- BC 1,1. -->
  <BYTES>42 43 20 31 2C 31 0D</BYTES>
  <!-- Set B-Band to VFO Mode -->
  <!-- VM 1,0. -->
  <BYTES>56 4D 20 31 2C 30 0D</BYTES>
</COMMAND>

<!-- Recommended default settings for initial setup in fldigi control panel.
  You may or may not need to change some settings on your radio to match
  these settings, or you can simply change the settings in fldigi to match
  your radio.  These settings are in the RigCAT section of the preferences.
  -->
  <TIMEOUT>50</TIMEOUT>
  <RETRIES>4</RETRIES>
  <WRITE_DELAY>5</WRITE_DELAY>
  <POST_WRITE_DELAY>5</POST_WRITE_DELAY>
  <BAUDRATE>57600</BAUDRATE>  <!-- TM-D710G menu item 920 -->
  <STOPBITS>1</STOPBITS>
  <RTSCTS>true</RTSCTS>
  <RTSPLUS>false</RTSPLUS>
  <RTSPTT>false</RTSPTT>
  <DTRPLUS>false</DTRPLUS>
```

```
  <DTRPTT>false</DTRPTT>
  <ECHO>false</ECHO>
  <CMDPTT>true</CMDPTT>
  <IDELAY>50</IDELAY>


<!-- This version can only transmit in FM and can only read the currently
     active Mode from the radio.  Hopefully, future versions will be able to
     set the mode as well (although the only other transmit mode is NFM).  -->
  <MODES>
    <ELEMENT>
      <SYMBOL>FM</SYMBOL>
      <BYTE>30</BYTE> <!-- Hex 30 = 0 -->
    </ELEMENT>
    <ELEMENT>
      <SYMBOL>NFM</SYMBOL>
      <BYTE>31</BYTE> <!-- Hex 30 = 1 -->
    </ELEMENT>
    <ELEMENT>
      <SYMBOL>AM</SYMBOL>
      <BYTE>32</BYTE> <!-- Hex 30 = 2 -->
    </ELEMENT>
  </MODES>

<!--

It's time for a quick background of how all this stuff works.  fldigi sends
commands to the radio, and the radio sends responses back to fldigi.  Some
of these responses seem to simply deliver information to fldigi, but actually,
all of these responses let fldigi know if its commands were accepted, or
rejected for some reason.

Unfortunately, the Kenwood D710G uses one single command to set and read
several different settings, all at once.  This limits our capabilities, and
forces us to be careful about how we use fldigi with the D710G.

This radio uses the FO command to set and read frequency and mode (along with
other settings).  The D710G will not accept a partial command; we have to
send parameters for every setting in the command, all at once.  This
obviously limits our flexibility.

Let's move on to reading responses.  We'll catch up on the details as we go.

To read settings, we simply use FO followed by which "band" (side) of the
radio we're interested in.

FO 0        retrieves A-Band settings
FO 1        retrieves B-Band settings (used exclusively in this document)

Here is an example response from this command, executed while my B-band was in
a memory channel for a Local repeater.  Any "tables" mentioned are available
in the TM-D710G manuals or online.

FO 1,0147320000,0,1,0,0,1,0,16,16,000,00600000,0
```

```
FO 1 = B-band
0147320000 = 147.320 MHz
0 = Step size (from the table).  In this case, 0 = 5kHz
1 = Positive shift to the repeater's input frequency
0 = Reverse function disabled (research this if you don't know what it is!)
0 = PL Tone is disabled (I call it "1-way tone")
1 = CTCSS is enabled ("2-way tone")
0 = DCS is disabled
16 = Tone is set to 114.8, the PL tone this repeater uses
16 = same as above, but for CTCSS (receive filter)
000 = DCS code (disabled but still technically set to 023)
00600000 = Offset amount for repeater input frequency
0 = Mode, FM in this case (0=FM, 1=NFM, 2=AM)
```

One good thing about this command/response is its fixed length. We can easily parse the returned data, and easily format our outgoing commands. In fact, the response from the radio conveniently shows us the exact format required for when we want to change these settings.

Let's take a closer look.

```
FO 1,0145050000,0,0,0,0,0,0,00,00,000,00000000,0  <- Actual response
xxxAxBBBBBBBBBxCxDxExFxGxHxIIxJJxKKKxLLLLLLLLxM  <- Identifiers
12345678901234567890123456789012345678901234567890  <- Just some helpful numbers
```

Identifiers:

- x => placeholder characters (commands, commas, and spaces)
- A => A-Band or B-Band (0 or 1)
- BBBBBBBBBB => Frequency (10 digits)
- C => Step size (see table in manual)
- D => Shift direction (0=none, 1=plus, 2=minus)
- E => Reverse function on/off (0 or 1)
- F => Tone on/off (0 or 1)
- G => CTCSS on/off (0 or 1)
- H => DCS on/off (0 or 1)
- II => Tone frequency (see table in manual)
- JJ => CTCSS frequency (see table in manual)
- KKK => DCS code (see table in manual)
- LLLLLLLL => Offset amount (in Hz)
- M => Mode (0=FM, 1=NFM, 2=AM)

Note that while desktop Kenwood models use a semicolon to signal the end of a command, the D710GA signals the end of a command/response with a single byte consisting of the hex code D0. Therefore, even though the above table shows 48 characters, you will see that my references below include that 49th character. I like to call it "the terminator." Likewise, the TX and RX commands include the same D0 byte to terminate their actions.

Now we're getting into the meat and potatoes of the xml file. With some careful research and a lot of trial and error, we can configure the commands and responses to do exactly what we want them to do.

Let's start with the computer listening to the radio first.

We'll start with some <REPLY> tags. These help fldigi understand the context of the responses that come from the radio. In technical terms, <REPLY> tags tell fldigi how to parse a response. We're just telling fldigi where to find the juicy bits. Stick with me; this will all make sense soon.

Inside the <REPLY> tags, you'll see SYMBOL, SIZE, BYTES, FILL, DATA, and DTYPE tags. These are used to describe the data within the REPLY tags. We'll go through them one by one, and also add a few more later on when we need them.

SYMBOL tells fldigi what kind of information we're dealing with.
SIZE tells fldigi the total size of the response that contains your data.
BYTES helps fldigi recognize certain important parts of the response.
FILL tells fldigi which parts of the response can be ignored.
DATA tells fldigi to pay attention - this is the specific info we're seeking!
DTYPE helps fldigi understand which type of data it should be seeking.
STRING is also used, but can be a little tricky to work with.

What if we make a mistake, or encounter a malfunction?

If a command is improperly formatted, or there is some other kind of error, the TM-D710G will return a simple "?" as its response. If the connection is completely cut off, we won't even get a response. If we send an invalid command (like setting the frequency to 50 GHz), the TM-D710G replies with a simple "N" response.

I like to imagine the radio asking "What?" (?) for illogical commands, and declaring "No!" (N) for forbidden operations.

However, if everything is in order, and the command goes through, we'll get a much longer response that contains the information and confirmation we expect. Successful responses from even the most basic commands are longer than one character.

Refer to the response table from the FO command that we reviewed earlier, and follow along through the <REPLY> tags below to see how we pick out the portion of data we want from the overall response.

-->

```
  <REPLY>
    <!-- Get the current mode from the radio -->
    <SYMBOL>MODE</SYMBOL>
    <SIZE>49</SIZE>
    <!-- The first five characters in the response should be "FO 1," -->
    <BYTES>46 4F 20 31 2C</BYTES>
    <!-- We can skip over most of the response now; we just want the end. -->
```

```
    <FILL>42</FILL>
    <DATA>
      <!-- We just want that one last byte: 0 or 1 or 2 (FM or NFM or AM). -->
      <DTYPE>BYTE</DTYPE>
      <SIZE>1</SIZE>
    </DATA>
    <!-- This signals the end of the response -->
    <BYTE>0D</BYTE>
  </REPLY>

  <REPLY>
    <!-- Get the current frequency from the radio -->
    <SYMBOL>FREQ</SYMBOL>
    <SIZE>49</SIZE>
    <!-- The first five characters in the response should be "FO 1," -->
    <BYTES>46 4F 20 31 2C</BYTES>
    <DATA>
      <!-- The next 10 digits represent the frequency in Hz. -->
      <DTYPE>DECIMAL</DTYPE>
      <SIZE>10</SIZE>
      <MAX>1299975000</MAX> <!-- max frequency of the B-Band -->
      <MIN>0136000000</MIN> <!-- min frequency of the B-Band -->
      <!-- NOTE: There are some differences between the bandwidths and ranges
                 between the A-Band and the B-Band of the D710G:
                   The A-Band can hear from 118 - 524 MHz
                   The B-Band can hear from 136 - 1300 MHz
                   (A and B both only transmit in the 2m and 70cm ham bands)
      -->
      <RESOL>1</RESOL>  <!-- Not sure what this means but it should be 1 -->
      <!-- Some radios reverse the MHz part of the frequency.  Ours doesn't -->
      <REV>false</REV>
    </DATA>
    <!-- We can ignore the rest of the response. -->
    <FILL>33</FILL>
    <!-- This signals the end of the response -->
    <BYTE>0D</BYTE>
    <!-- Note that if you add up each byte, each digit, and each fill, you'll
          end up with a number that equals the SIZE we specified earlier.  -->
  </REPLY>

  <REPLY>
    <!-- The OK checker seems like a big mess, but really, it just checks the
           response to make sure we received a complete answer.  As you can
           see, the symbol name is "OK" so we're not after any data here.  Just
           a validity check. (Hint: 2C is a comma byte)
           Note that this OK checker only works for the FO response.
           -->
    <SYMBOL>OK</SYMBOL>
    <SIZE>49</SIZE>
    <!-- Let's count the bytes up to SIZE 49... -->

    <BYTES>46 4F 20 31 2C</BYTES>  <!-- 1 2 3 4 5 -->
    <FILL>10</FILL>  <!-- 5 + 10 = 15 -->
    <BYTE>2C</BYTE>  <!-- 16 -->
```

```
    <FILL>1</FILL>   <!-- 17 -->
    <BYTE>2C</BYTE>  <!-- 18 -->
    <FILL>1</FILL>   <!-- 19 -->
    <BYTE>2C</BYTE>  <!-- 20 -->
    <FILL>1</FILL>   <!-- 21 -->
    <BYTE>2C</BYTE>  <!-- 22 -->
    <FILL>1</FILL>   <!-- 23 -->
    <BYTE>2C</BYTE>  <!-- 24 -->
    <FILL>1</FILL>   <!-- 25 -->
    <BYTE>2C</BYTE>  <!-- 26 -->
    <FILL>1</FILL>   <!-- 27 -->
    <BYTE>2C</BYTE>  <!-- 28 -->
    <FILL>2</FILL>   <!-- 30! Almost gotcha there! (Fill 2) -->
    <BYTE>2C</BYTE>  <!-- 31 -->
    <FILL>2</FILL>   <!-- 33 -->
    <BYTE>2C</BYTE>  <!-- 34 -->
    <FILL>3</FILL>   <!-- 37 -->
    <BYTE>2C</BYTE>  <!-- 38 -->
    <FILL>8</FILL>   <!-- 46 -->
    <BYTE>2C</BYTE>  <!-- 47 -->
    <FILL>1</FILL>   <!-- 48 -->
    <BYTE>0D</BYTE>  <!-- 49! -->
    <!-- So, if the response we receive is 49 bytes and has all the commas in
         the right places, we can be very confident that we received a valid
         response, so we'll label it as "OK".  -->
  </REPLY>

  <REPLY>
    <!-- Remember, if the radio didn't accept our command, it will either
           issue a question mark or simply "N".  Either way, we know something
           went wrong.  This "BAD" symbol just looks for a response that's
           too short.  -->
    <SYMBOL>BAD</SYMBOL>
    <SIZE>2</SIZE>
    <FILL>1</FILL>  <!-- Probably either ? or N, but it doesn't matter -->
    <BYTE>0D</BYTE>
  </REPLY>


  <!-- Okay, we've given precise instructions to fldigi on how to interpret
       responses from the radio.  But how do we get the radio to respond?

       By issuing a command, of course.  Below, we'll look at a few different
       types of commands.  Some are basically a request for info, and others
       actually tell the radio to change something.

       COMMANDs are simply things your computer sends to your radio.  We just
       need to tell fldigi what to send in each type of command, depending on
       what response we want from the radio.  -->

  <COMMAND>
    <!-- Tell me what mode we are in! -->
    <SYMBOL>GETMODE</SYMBOL>
    <!-- Send a simple "FO 1" command, with no parameters.
```

```
        "FO 1" is 4 bytes, but we can't forget the terminator, so we specify
        the size as 5 bytes. -->
    <SIZE>5</SIZE>
    <!-- Now we send the bytes that represent "FO 1". -->
    <BYTES>46 4F 20 31 0D</BYTES>
    <!-- Here, since we're asking what MODE we're in, we tell fldigi where to
        find the MODE data from within the long response.  fldigi simply takes
        the INFO tag and looks up the REPLY SYMBOL that matches it.  We
        defined that earlier with the <SYMBOL>MODE</SYMBOL> code above.  -->
    <INFO>MODE</INFO>
    <OK>OK</OK>
    <BAD>BAD</BAD>
  </COMMAND>

  <COMMAND>
    <!-- Tell me what frequency we are on! -->
    <SYMBOL>GETFREQ</SYMBOL>
    <SIZE>5</SIZE>
    <BYTES>46 4F 20 31 0D</BYTES>
    <INFO>FREQ</INFO>  <!-- Tell fldigi to use the FREQ Symbol from above! -->
    <OK>OK</OK>
    <BAD>BAD</BAD>
  </COMMAND>

  <COMMAND>
    <!-- Now, we're telling the radio to CHANGE to a different frequency.  This
        section should be very familiar, but this time, it's like parsing the
        data in reverse.  We're building the command in the same way we were
        parsing the command earlier.

        Here's where the TM-D710G gets a little finicky.  Using the same
        FO 1 command that we used earlier, we can't just specify the frequency
        and leave all the other stuff out.  The radio won't accept it unless
        it's complete.  So this is why we must decide on the most common
        configuration for all that other stuff (tones, offsets, etc) that
        we'll use during a digital QSO.  We're forced to commit to it here,
        because there is no other way to command the radio to change frequency.

        -->
    <SYMBOL>SETFREQ</SYMBOL>
    <SIZE>49</SIZE>
    <BYTES>46 4F 20 31 2C</BYTES> <!-- Start with "FO 1," -->
    <DATA> <!-- Then add the new frequency.
                fldigi will insert the correct data here. -->
      <DTYPE>DECIMAL</DTYPE>
      <SIZE>10</SIZE>
      <MAX>1299975000</MAX>
      <MIN>0136000000</MIN>
      <RESOL>1</RESOL>
      <REV>false</REV>
    </DATA>
    <!-- The bytes below are simply the zeroes and commas that we decided we
        must use in order to fill out the command that ultimately lets us
        change the frequency on the radio.  Note the 0D terminator on the end.
```

```
        -->
    <BYTES>2C 30 2C 30 2C 30 2C 30</BYTES>
    <BYTES>2C 30 2C 30 2C 30 30 2C</BYTES>
    <BYTES>30 30 2C 30 30 30 2C 30</BYTES>
    <BYTES>30 30 30 30 30 30 30 2C</BYTES>
    <BYTES>30 0D</BYTES>
    <!-- Here, we tell fldigi to check if the response is OK or BAD.  If it's
         BAD, we can set fldigi to beep at us, or simply try the command again.
         We can also tell fldigi that if it fails too many times, it should
         stop trying, and present us with an error message.  This all happens
         very quickly, of course.  -->
    <OK>OK</OK>
    <BAD>BAD</BAD>
  </COMMAND>


  <!-- WAIT!  Aren't we forgetting something?  How do we transmit?  How do we
       stop transmitting?  Fortunately, that's the easy part.  We just send the
       transmit command "TX" followed by the terminator, or "RX" followed by
       the terminator to go back into receive mode.  Match those Kenwood
       commands with the fldigi PTTON and PTTOFF Symbols, and you're good
       to go.  That's it!  -->

  <COMMAND>
    <SYMBOL>PTTON</SYMBOL>
    <SIZE>3</SIZE>
    <BYTES>54 58 0D</BYTES> <!-- "TX" -->
    <!-- Can't use OK because that's for the FO command -->
    <BAD>BAD</BAD>
  </COMMAND>

  <COMMAND>
    <SYMBOL>PTTOFF</SYMBOL>
    <SIZE>3</SIZE>
    <BYTES>52 58 0D</BYTES> <!-- "RX" -->
    <!-- Can't use OK because that's for the FO command -->
    <BAD>BAD</BAD>
  </COMMAND>


<!-- Well, that's all, folks!  I hope you've enjoyed reading this as much as
     I've enjoyed working on it.  Now that we're done, I'll say that I am
     SURE that there are improvements that can be made to this code and better
     ways of doing things.

     In fact, I'm hoping that this code will lead to someone more knowledgeable
     than me porting some of this over to the hamlib system and other popular
     rig control systems.  I'm looking forward to learning as much as I can
     from this exercise, and from whichever radio project next falls in my lap.

     Thanks for playing along!

     73
```

```
     K5URU
     Blake Ormand
     Buda, TX, USA EM10cb
     blake.ormand@gmail.com

     -->


<!-- Not yet implemented for TM-D710G:

  <COMMAND>
    <SYMBOL>SETMODE</SYMBOL>
  </COMMAND>
-->

</RIGDEF>
```