TKR-x20 Programming
KB9BPF / WQYP804
22 Dec 2016 / Rev. 31 Dec 16

Test platform:
TKR-820(N)
SN: 807xxxxx
FCC ID: ALHTK-820N-1

To run KPG-21D in DOSBox:

(Thanks to Alex Szuski KD6VPH for showing how to do this.)

- Open DOSBox, then at Z:\> prompt, type "mount c c:\kpg21d" and DOSBox will reply "Drive C is mounted as local directory c:\kpg21d\"
- You will then get a prompt "Z:\>" at this step we are ready to run "kpg21d.exe", so type "c:\kpg21d"
- Drive and Path for KPG21D Data folder will be C:\DATA
- Note: I could not find a way to print to the local LPT1 printer with DOSBox0.74

Using the Phyton ChipProgUSB Universal Programmer:

*.R8N files generated by KPG-21D programming software can be used by Phyton ChipProgUSB.
Here's how:

- Select Device > Exel 93LC46 [x16]
- Load File: Select File Format: Binary Image
- Load File: Browse
- Open File: File name: S:\...\Kenwood TKR-820\KPG-21D_v2.00\KPG21D\DATA\WQYP804.R8N
- Open File: Files of Type: All Files (*)

Frequency Programming:

(Thanks to Matt Krick K3MK for figuring this out.)
The Frequency EEPROM is IC2, 93LC46 on Display Unit, behind front panel, near Power LED.
In the buffer window, you'll see:

Address (Hex)	Word 0	Word 1	Word 2	Word 3	Word 4	Word 5	Word 6	Word 7	Notes:
0000	8B72	89E2	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	Word 0: RX Freq: 35698 x 0.0125 = 446.225 + 21.4 = 467.625
									Word 1: TX Freq: 35298 x 0.0125 = 441.225 + 21.4 = 462.625
0008	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	
0010	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	
0018	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	
0020	FF57	FF51	FF59	FF50	FF38	FF30	FF34	FF20	Embedded Message: W Q Y P 8 0 4 _
0028	FF20	FF20	FF20	FF20	FF20	FF20	FF20	FF31	Embedded Message: _ _ _ _ _ _ _ 1
0030	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	
0038	FFFF	FFFF	FFFF	FFFF	FFFF	3852	3032	314E	Word 5 6 7: 8R 02 1N -> R820N1

QT/DQT Programming:

(Thanks to Matt Krick K3MK for figuring much of this out, too.)
The QT-DQT EEPROM is IC10, 93LC46 in the Signaling Unit, under top panel, in the RX shield.
In the buffer window, you'll see the QT/DQT EEPROM programming data under the Frequency Programming Data.

Address (Hex)	Word 0	Word 1	Word 2	Word 3	Word 4	Word 5	Word 6	Word 7	Notes:
0040	EC81	EC81	C83B	C83B	FFFF	FFFF	FFFF	FFFF	Word 0: RX Tone1: &hEC81 = Dec 60545: DQT = D023N
									Word 1: TX Tone1: &hEC81 = Dec 60545: DQT = D023N
									Word 2: RX Tone2: &hC83B - &hc000 = &h083B
									= Dec 2107 / 10 = QT frequency 210.7
									Word 3: TX Tone2: &hC83B - &hc000 = &h083B
									= Dec 2107 / 10 = QT frequency 210.7
									Word 4: RX Tone3: &hEA81 = Dec 60033: DQT = D025N
									Word 5: TX Tone3: &hEA81 = Dec 60033: DQT = D025N
									Word 6: RX Tone4: &hE379 = Dec 58233: DQT = D754N
									Word 7: TX Tone4: &hE379 = Dec 58233: DQT = D754N
0048	E37E	E37E	E57E	E57E	EC86	EC86	FFFF	FFFF	Word 0: RX Tone5: &hE37E = Dec 58238: DQT = D023I
									Word 1: TX Tone5: &hE37E = Dec 58238: DQT = D023I
									Word 2: RX Tone6: &hE57E = Dec 58750: DQT = D025I
									Word 3: TX Tone6: &hE57E = Dec 58750: DQT = D025I
									Word 4: RX Tone7: &hEC86 = Dec 60550: DQT = D754N
									Word 5: TX Tone7: &hEC86 = Dec 60550: DQT = D754N
									Word 6: RX Tone8: &hFFFF = Dec 65535: Blank
									Word 7: TX Tone8: &hFFFF = Dec 65535: Blank
0050	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	
0058	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	
0060	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	
0068	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	
0070	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	
0078	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	

DQT Coding Format:

(Thanks to Matt Krick K3MK for inspiring me to do this.)
DQT Coding format: &hExxx: 1110 xxxx xxxx xxxx (binary)
1. Dropping the leading 'E' (Subtract &hE000) yields four octal groups read right-to-left: dddc ccbb baaa
    (&h[4-byte code] - &hE000 = &hxxx)
2a. If aaa = 001, the DQT is Normal and the numbers for b, c, and d are NOT bit-inverted. (They are still read R-L.)
2b. If aaa = 110, the DQT is Inverted and the numbers for b, c, and d ARE bit-inverted. (They are still read R-L.)

Example 1: DQT = D023N -> Hexadecimal word &hEC81 -> 1110 1100 1000 0001 (binary)
1. &hEC81 - &hE000 = &hC81 -> 1100 1000 0001 (binary)
   Arrange into four octal groups: 110 010 000 001
   Then reverse the order of the bits for bbb, ccc, and ddd.
2. Since aaa = 001 (100 if reversed, which is decimal 4) the DQT is Normal (non-inverted).
      aaa = 001, Normal
      bbb = 000, bbb(reversed) = 000 = 0 (decimal),
      ccc = 010, ccc(reversed) = 010 = 2 (decimal),
      ddd = 110, ddd(reversed) = 011 = 3 (decimal).

   The DQT is D023N

Example 2: DQT = D147I -> Hexadecimal word &hE19E -> 1110 0001 1001 1110 (binary)
1. &hE19E - &hE000 = &h19E -> 0001 1001 1110 (binary)
   Arrange into four octal groups: 000 110 011 110
   Then reverse the order of the bits for bbb, ccc, and ddd.
2. Since aaa = 110 (011 if reversed, which if inverted is 100, decimal 4) the DQT is Inverted.
      aaa = 110, Inverted
      bbb = 011, bbb(reversed) = 110, bbb(reversed and inverted) = 001 = 1 (decimal),
      ccc = 110, ccc(reversed) = 011, ccc(reversed and inverted) = 100 = 4 (decimal),
      ddd = 000. ddd(reversed) = 000, ddd(reversed and inverted) = 111 = 7 (decimal).

   The DQT is D147I.

QT Coding Format:

(Thanks to Matt Krick K3MK for figuring this out.)
QT Coding format: &hCxxx: 1110 xxxx xxxx xxxx (binary)
1. Dropping the leading 'C' (Subtract &hC000) yields three Hexadecimal groups read left-to-right: &h0abc
    (&h[4-byte code] - &hC000 = &h0abc)
2. Convert hex to decimal and then divide by ten to yield QT frequency in Hz. (For Carrier Squelch use QT of Zero Hz.)

Example 1: QT = 67.0 -> Hexadecimal word &hC29E
1. &hC29E - &hC000 = &h029E
2. &h29E -> 670 (decimal) / 10 = 67.0 Hz

Example 2: QT = 123.0 -> Hexadecimal word &hC4CE
1. &hC4CE - &hC000 = &h04CE
2. &h4CE -> 1230 (decimal) / 10 = 123.0 Hz.