External Programming for the Midland XTR radios

I updated this document 9/2019 trying to clarify how to interpret the programming data from the Midland XTR series (.xtr) radios.

The basic program file (filename.xtr) produced by the Midland XTR is encoded in a proprietary format, basically the core memory encapsulated in asci encoded character string starting with "S10", and ending in a carriage return, linefeed pair (CR/LF) This file may be read with most document viewers. (say Notepad). An example is:

S10B000012345678FFFFFABC2C
S10B0008DE080208068A00B9B3
S10B0010B0E0613F9A83BDF7E3
S10B0018FFFFFFBFD37FFFFFD0
S10B00200FFFFFF252A41C01C2
S10B0028C424D2FFFFF252AC24
S10B00300C01C42CF2FFFFF2E5
S10B003852B41401C434F0FFBA
S10B0040FFF252BC0C01C43CA8
S10B0048F0FFFFF2531C0B0151
S10B0050C49CF3FFFFF256A863
S10B00580C01C828F2FFFFF2BD
S10B006057100C01C890F3FFD6
S10B 00 68 FF F2 57 20 0C 01 C8 A0 AF
S10B0070F1FFFFF257380C0107
S10B0078C8B8FFFFFFF2579026…

The end of the file is simply the last of 1024 bytes of data, notice that the "S10" preamble does not change for the end of record:
S10B03F0FFFFFFFFFFFFFFFF09
S10B03F8FFFFFFFFFFFFFFFF01

Dissecting a line: S10B 0000 12345678FFFFABC 2C (cr/lf)
S10B is some type of header.
0000 is the internal XTR memory address for the start of line data
12345678FFFABC are the hexadecimal data
And 2C is the line checksum.
The string is terminated with a carriage return/linefeed

The checksum is the one's complement of the summation of each byte of string data, excluding the "S1"
S1 0B+ 00+ 00+ 12+ 34+ 56+ 78+ FF+ FF+ FA+ BC =4D3, D3 complement is 2C

An image of the XTR internal memory is constructed by decoding each line of file strings, and inserting the data bytes into their respective buffer memory address, referred here as the "XTR buffer."

So, the XTR buffer for the above examples is: (four lines only)

Addr	data
0000	12 34 56 78 FF FF FA BC DE 08 02 08 06 8A 00 B9
0010	B0 E0 61 3F 9A 83 BD F7 FF FF FF BF D3 7F FF FF

MEMORY MAP OF INTERNAL XTR BUFFER

The following use the XTR buffer address, The actual address of the byte containing these data is in hexadecimal, with the actual bits represented by 'b'

Serial Num: 0h-3h BCD encoded
Customer ID 5h-8h BCD encoded
Date: Digits 9h-11h BCD encoded

1337?: 0=no, 1=yes –13h(xxxbxxxx)

Scan type– 19h(xxxxbbxx)

00=norm	01=mod
10=sec	11=ps

Aux key type: --20h(xxxxbbxx)

1=11	2=10
3=01	4=00

Enable chl rollover: –19h(xxxxxxbx)

Priority Mon. chl time:–22h(bbxxxxxx)

00=.5	01=.75
10=1	11=1.5

Priority Mon. chl cycle: 1to4=0, 1to8=1 ;–19h(xxxxxxxb)

[Priority two chl: 1=enabled, 0=disabled; –31h(xxxxxxxb)

Scan hold conditions: –19h(xxbbxxxx)

OPN/NSQ=00	OPN/SIG=10
BSY/NSQ=01	BSY/SOG=11

Scan hold on RX: –18h(bbxxxxxx)

.3=00	5.0=10
2.5=01	Infin.11

Scan hold after TX: --18h(xxbbxxxx)

.3=00	5.0=10
2.5=01	Infin.11

When table deleted: --19(bbxxxxxx)

pwr scan=00	pwr off=01
scan off=10	no clear=11

Scan stop w/mic: 0=enable, 1=disable ; --28h(xxxxxbxx)

Bsy chl lockout: --28h(bbxxxxxx)

NSQ=00	signaling=01
special=10	disable=11

TX timer timeout: --28h(xxbbbxxx)

30=000	150=100	120=011
60=001	180=101	infinite=111
90=010	210=110	

Beep Control: 0=disable, 1=enable ;22h(xxABCDxx)

The rest of the file are channel data, channel 1 starting at 21h, with 10 bytes of data each. This continues until the end of the file.

Channel data stream: AA `CCCR RRRR CCCT TTTT BK`

Where AA are the Auxiliary bits 8-1; 0=disabled, 1=enabled

CCC are CTCSS codes (detailed below)

RRRRR (or TTTTT) are the received (or transmit) frequencies (explained below)

B are the channel control bits:

Msb= power setting, 0=low, 1=high
3lsb= enable scrambler=0
2lsb= enable channel in scan A=0
1lsb= enable channel in scan B=0

C is the data checksum:

Checksum is four bit two's complement of each digit of RX and TX frequency
EX: C+0+A+8+9+F+3+0=39 2's comp=F7, checksum digit= 7

How to encode RX and or TX frequencies:

IF frequencies: VHFL(10.7 Mhz) VHFM(45 Mhz) UHF(45 Mhz)

Frequency to Hex:

Decode:
If (RX) Fd= RRRRR
If (TX) Fd= TTTTT ;use encoded Tx freq

Fdc=((Fd & FFF80h)>>1)+(Fd & 0003Fh) ;stuffing used for synthesizer chip
Fo=Fdc*.0025 ;scaled by resolution.
If RX Fo=Fo-IF_Freq ;10.7Mhz for 6meter, 45 Mhz for 2meter

Examples RX = 252A4h
Fdc=((252A4&fffc0h)>>1)+(252A4&0003fh) = (12940h)+(24h) = 12964h
Fo=Fdc*.0025 = 76132d * .0025= 190.33
Fo= Fo-IF freq =190.33-45 =145.33 (Mhz)

Example TX =1C424h
Fdc=((1C424&fffc0h)>>1)+(1C424&0003fh) = (E200h)+(24h) = E224h
Fo=Fdc*.0025 = 57892d * .0025= 144.73 (Mhz)

Hex to Frequency:
Encode
If(RX) Fo=operation freq+IF freq
If(TX) Fo= operation freq
Fdc=Fo/.0025
Fd=((Fdc&FFFC0)<<1)+(Fdc&0003F)

Example Fo=145.33 Mhz (RX)
Fo=145.33+45 =190.33
Fdc=190.33/.0025 =76132d (12964h)
Fd=((12964h&FFFC0)<<1)+(12964h&0003f) = 25280h+24 =252A4h

CTCSS encoding:
RX digits 3-6 TX digits 11-13

67.0 =000	71.9 =010	74.4=020	77.0=030	79.7=040
82.5 =050	85.4 =060	88.5=070	91.5=080	94.8=090
97.4 =0A0	100.0 =0B0	103.5=0C0	107.2=0D0	110.9=0E0
114.8 =0F0	118.8 =100	123.0 =110	127.3=120	131.8=130
136.5 =140	141.3 =150	146.2 =160	151.4=170	156.7=180
162.2 =190	167.9 =1A0	173.8 =1B0	179.9=1C0	86.2=1D0
192.8 =1E0	203.5 =1F0	210.7=200	218.1= 210	225.7=220
233.6 =230	241.8=240	250.3=250	74.0 =260	69.3 = 270
198.0 =280	202.7 =290	206.5 =2A0	229.1= 2B0	254.1=2C0

Function encoding bits: digit nineteen
Digit nineteen is bit encoded as:
Msb= power setting, 0=low, 1=high
3lsb= enable scrambler=0
2lsb= enable channel in scan A=0
1lsb= enable channel in scan B=0

Checksum encoding: digit twenty

Checksum is four bit two's complement of each digit of RX and TX frequency'

EX: C+0+A+8+9+F+3+0=39 2's comp=F7, checksum digit= 7

How to use these data to check an .xtr file content: Lacking any actual program to decode the .xtr file directly (any programmers want to do it?) the data may accessed by viewing the .xtr file directly.

Here's an example of a six meter radio .xtr file:

```
S10B0000FFFFFFFFFFFFFFFFC
S10B0008FF110298028A00EFC7
S10B001048E061D7B8B13C8F50
S10B0018FFFFFFBFDF7FFF3F84
S10B00200FFFFFF0C8240B00E0
S10B0028A184FFFFFFF0C7A44F
S10B00300B00A104F0FFFFF036
S10B0038C9A40B00A304FCFFA2
```

The channel data starts at address 21h, so starting at the second byte of line S10B0020:

```
FFFFF0C8240B00A184FF ;channel one data
```

Aux bits are set, there is no RX CTCSS tone, the RX frequency is 53.39 Mhz, the TX tone is 173.8 and the TX frequency is 51.69 Mhz. (remember, TX is direct, w/o IF frequency offset). The final data are power is set high, the scrambler is disabled, and the channel is not in scan group A or B. Finally, the channel checksum is 'F' (0+C+8+2+4+0+A+1+8+4)= 31h, the two's complement is CF, so the checksum digit is 'F'.