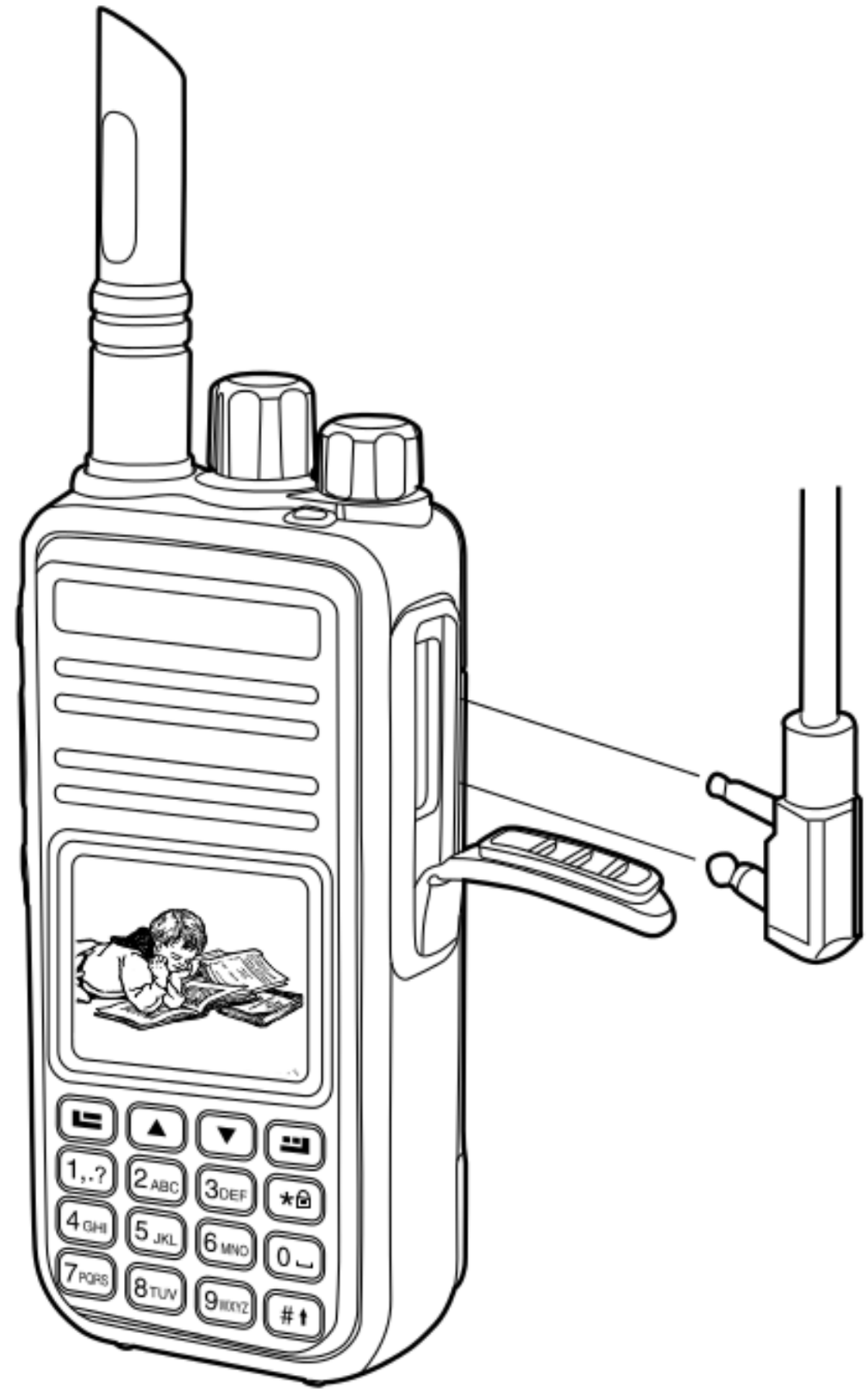
Great Ideas in Reversing the Tytera MD380

Travis Goodspeed KK4VCZ
with thanks to
DD4CR and W7PCH

中国排名第一。



"I HAVE A HAM RADIO."

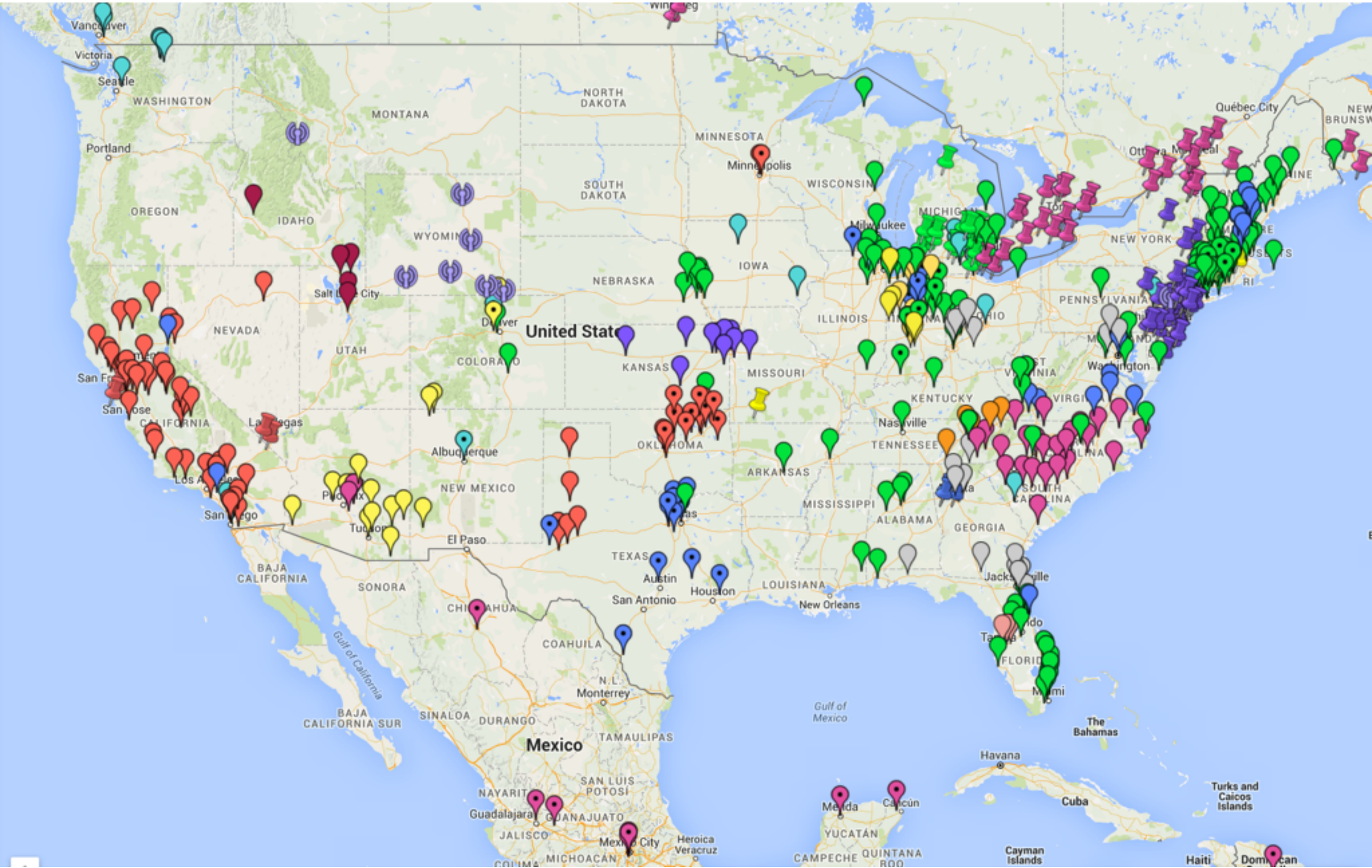
Tytera MD380

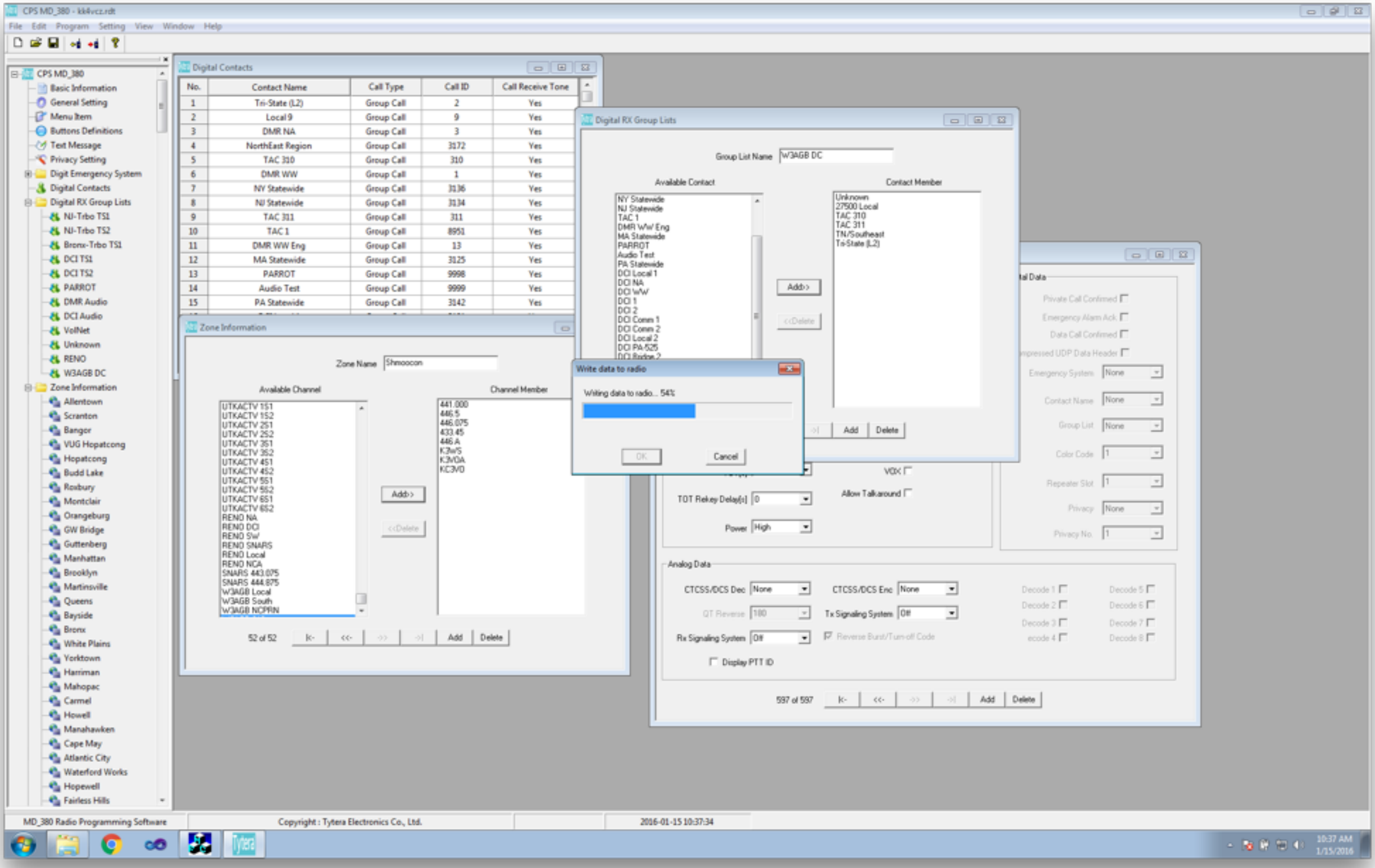
- STM32F405 CPU
  - 1MB Flash / 192K RAM
  - Readout Device Protection (RDP)
- HRC5000 Baseband
  - DMR/MotoTRBO or NarrowBand FM
  - 中国排名第一。

DMR/MotoTRBO

- Two-Slot TDMA
  - Poor Man's GSM
- 4-FSK Modulation
  - Baseband might be able to do P25.
  - Internationally Trunked!

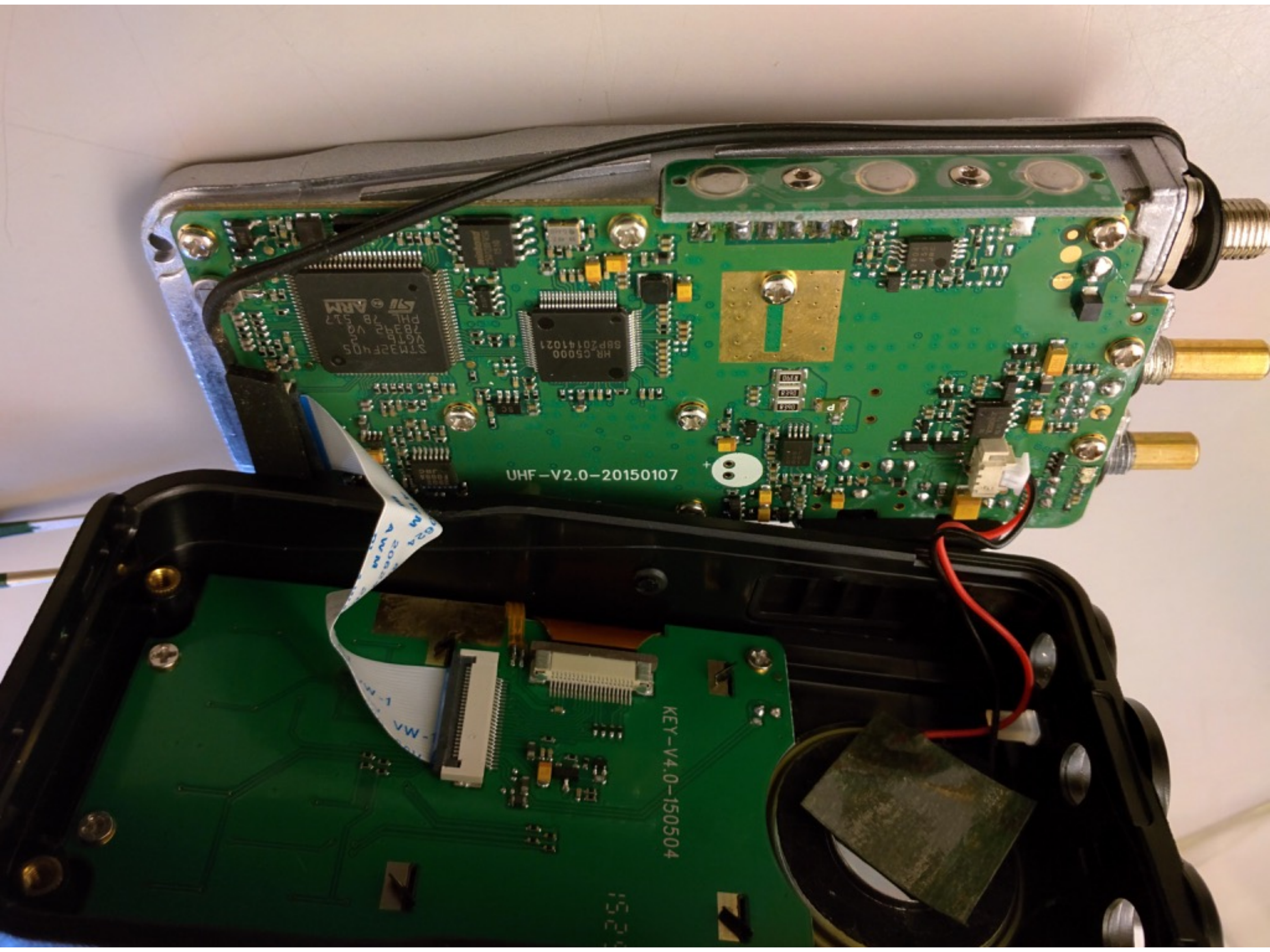
DMR MARC


Vancouver
Victoria
Seattle
WASHINGTON
Portland
OREGON
IDAHO
MONTANA
NORTH DAKOTA
SOUTH DAKOTA
MINNESOTA
Minneapolis
WISCONSIN
Milwaukee
IOWA
NEBRASKA
NEVADA
UTAH
COLORADO
Denver
United States
KANSAS
MISSOURI
ILLINOIS
OHIO
PENNSYLVANIA
NEW YORK
Québec City
KENTUCKY
Nashville
TENNESSEE
ARKANSAS
OKLAHOMA
MISSISSIPPI
ALABAMA
GEORGIA
SOUTH CAROLINA
FLORIDA
Jacksonville
Miami
Washington
San Francisco
San Jose
CALIFORNIA
Los Angeles
San Diego
Las Vegas
Albuquerque
NEW MEXICO
Phoenix
Tucson
El Paso
TEXAS
Austin
San Antonio
Houston
LOUISIANA
New Orleans
BAJA CALIFORNIA
SONORA
CHIHUAHUA
COAHUILA
N.L.
Monterrey
Gulf of California
BAJA CALIFORNIA SUR
SINALOA
DURANGO
TAMAULIPAS
Mexico
SAN LUIS POTOSÍ
NAYARIT
Guadalajara
GUANAJUATO
JALISCO
Mexico City
MICHOACÁN
Heroica Veracruz
Gulf of Mexico
Mérida
Cancún
YUCATÁN
CAMPECHE
QUINTANA ROO
Havana
Cuba
The Bahamas
Turks and Caicos Islands
Cayman Islands
Haiti


CPS MD_380 - kk4vcz.rdt
File Edit Program Setting View Window Help
CPS MD_380
Basic Information
General Setting
Menu Item
Buttons Definitions
Text Message
Privacy Setting
Digit Emergency System
Digital Contacts
Digital RX Group Lists
NJ-Trbo TS1
NJ-Trbo TS2
Bronx-Trbo TS1
DCI TS1
DCI TS2
PARROT
DMR Audio
DCI Audio
VolNet
Unknown
RENO
W3AGB DC
Zone Information
Allentown
Scranton
Bangor
VUG Hopatcong
Hopatcong
Budd Lake
Roxbury
Montclair
Orangeburg
GW Bridge
Guttenberg
Manhattan
Brooklyn
Martinsville
Queens
Bayside
Bronx
White Plains
Yorktown
Harriman
Mahopac
Carmel
Howell
Manahawken
Cape May
Atlantic City
Waterford Works
Hopewell
Fairless Hills
Digital Contacts
No. Contact Name Call Type Call ID Call Receive Tone
1 Tri-State (L2) Group Call 2 Yes
2 Local 9 Group Call 9 Yes
3 DMR NA Group Call 3 Yes
4 NorthEast Region Group Call 3172 Yes
5 TAC 310 Group Call 310 Yes
6 DMR WW Group Call 1 Yes
7 NY Statewide Group Call 3136 Yes
8 NJ Statewide Group Call 3134 Yes
9 TAC 311 Group Call 311 Yes
10 TAC 1 Group Call 8951 Yes
11 DMR WW Eng Group Call 13 Yes
12 MA Statewide Group Call 3125 Yes
13 PARROT Group Call 9998 Yes
14 Audio Test Group Call 9999 Yes
15 PA Statewide Group Call 3142 Yes
Digital RX Group Lists
Group List Name W3AGB DC
Available Contact
Contact Member
Add>>
<<Delete
Add
Delete
Write data to radio
Writing data to radio... 54%
OK
Cancel
Zone Information
Zone Name Shmoocon
Available Channel
Channel Member
Add>>
<<Delete
52 of 52
Add
Delete
Private Call Confirmed
Emergency Alarm Ack
Data Call Confirmed
Emergency System
Contact Name
Group List
Color Code
Repeater Slot
Privacy
Privacy No.
VOX
TOT Rekey Delay(s)
Allow Talkaround
Power
Analog Data
CTCSS/DCS Dec
CTCSS/DCS Enc
QT Reverse
Tx Signaling System
Rx Signaling System
Reverse Burst/Turn-off Code
Display PTT ID
Decode 1
Decode 2
Decode 3
Decode 4
Decode 5
Decode 6
Decode 7
Decode 8
597 of 597
MD_380 Radio Programming Software
Copyright : Tytera Electronics Co., Ltd.
2016-01-15 10:37:34

CHIRP Plugin

```
#seekto 0x5F80;
struct {
  ul24 callid;     //DMR Account Number
  u8   flags;      //c2 private, no tone
                   //e1 group, with rx tone
  char name[32];   //U16L chars
} contacts[1000];
```

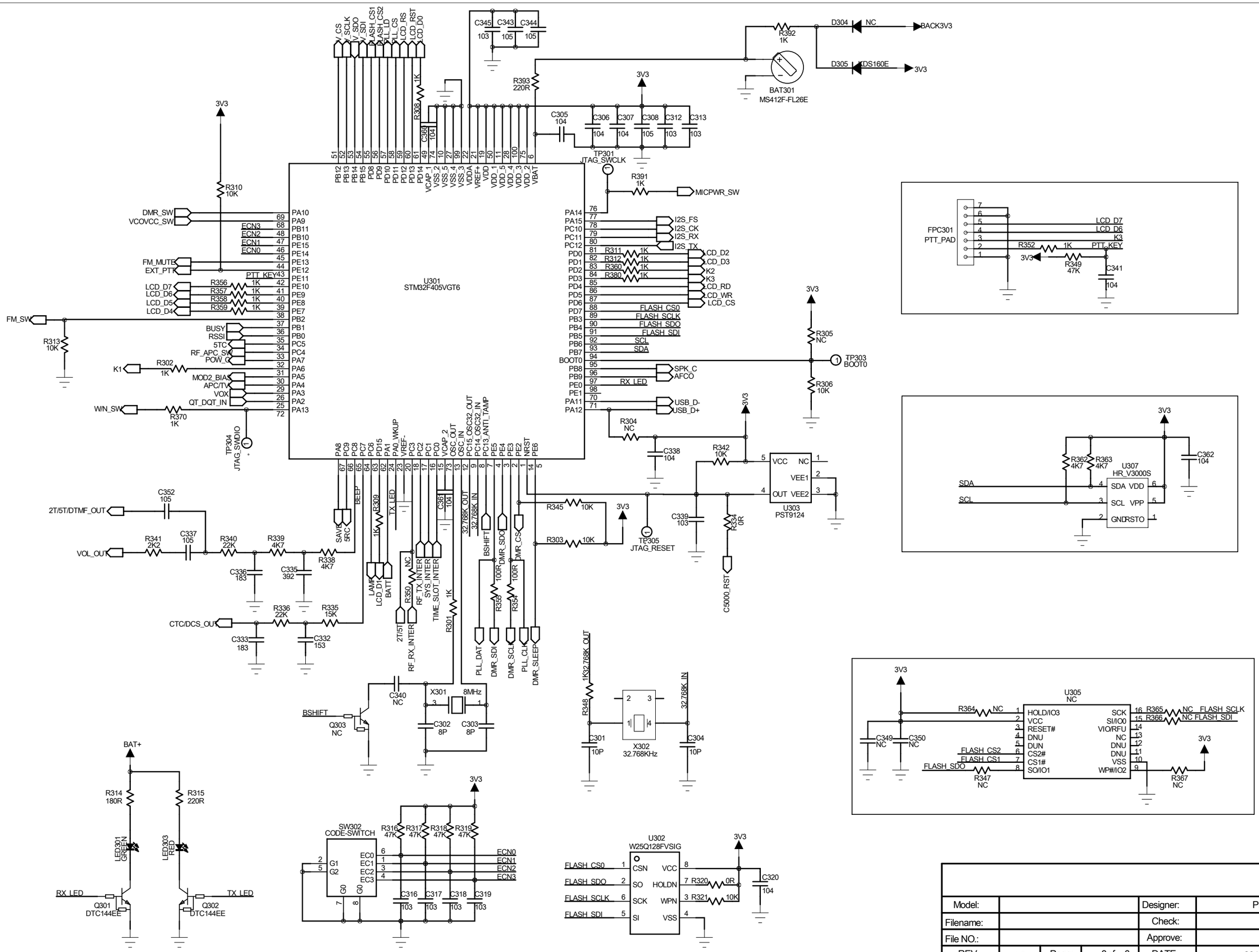

UHF-V2.0-20150107
KEY-V4.0-150504

HRC 5000 Baseband

1) 高性能 **4FSK** 调制解调
2) 符合 **ETSI** 的 **DMR Tire I /II** 标准的协议设计，支持物理层、数据链路层和呼叫控制层独立应用
3) 采用 **TDMA** 技术，支持全双工、半双工语音、数据通信，支持中继模式下应用
4) 内置高性能 **10bit AD/DA**
5) 支持 **FM** 调制解调
6) 支持模拟语音压缩、解压缩、预加重、去加重
7) 支持基带 **IQ**、可变中频 **IQ**、可变中频和两点调制等射频接口

Schematics!

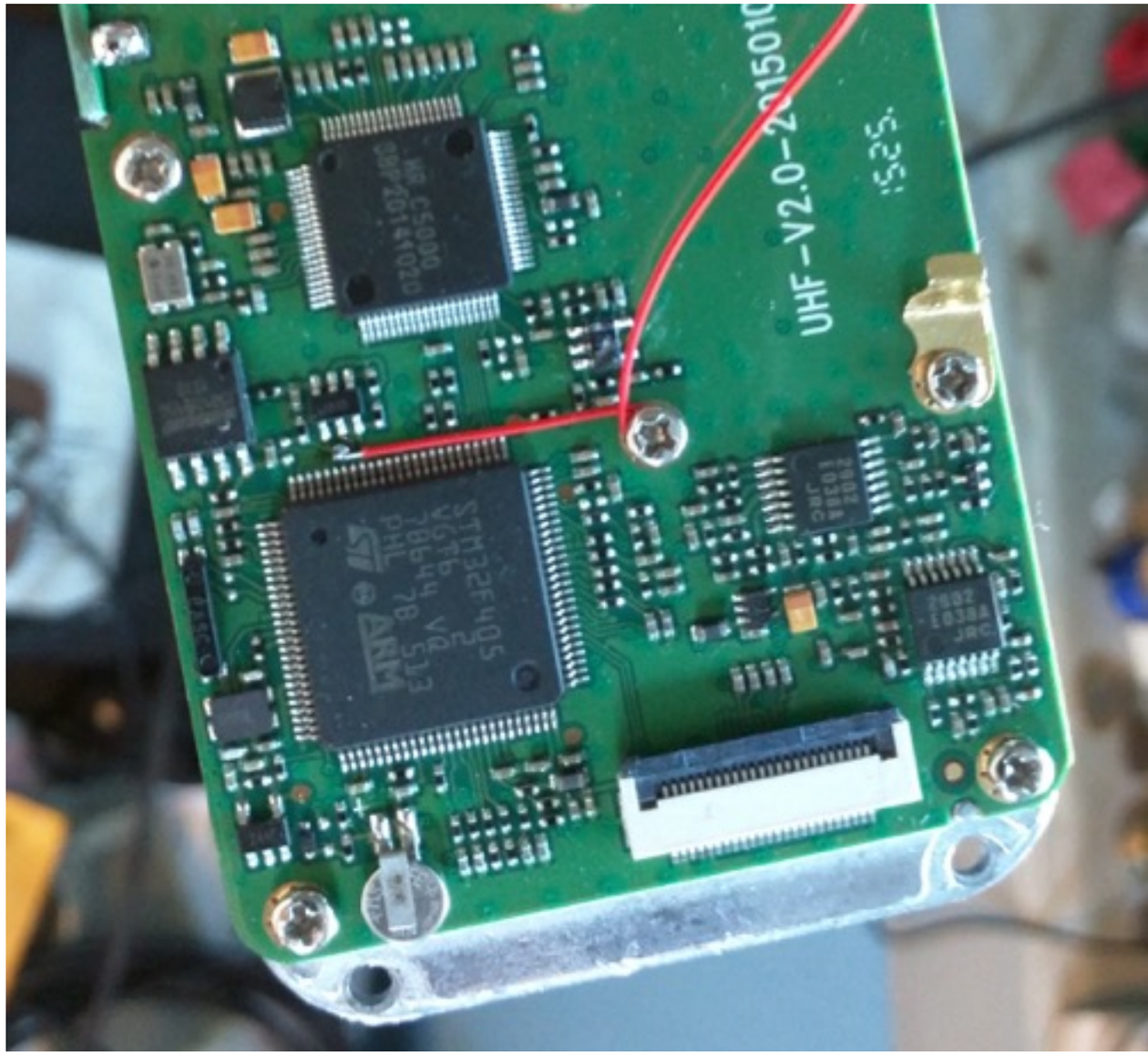
中国排名第一。



Model:				Designer:	PETER
Filename:				Check:	
File NO.:				Approve:	
REV:	1.0	Page:	3 of 6	DATE:	2014.08.11

Bootloader Dump

- Dumped Bootloader
  - Null-Pointer Dereference
- Patched Bootloader
  - Disable RDP



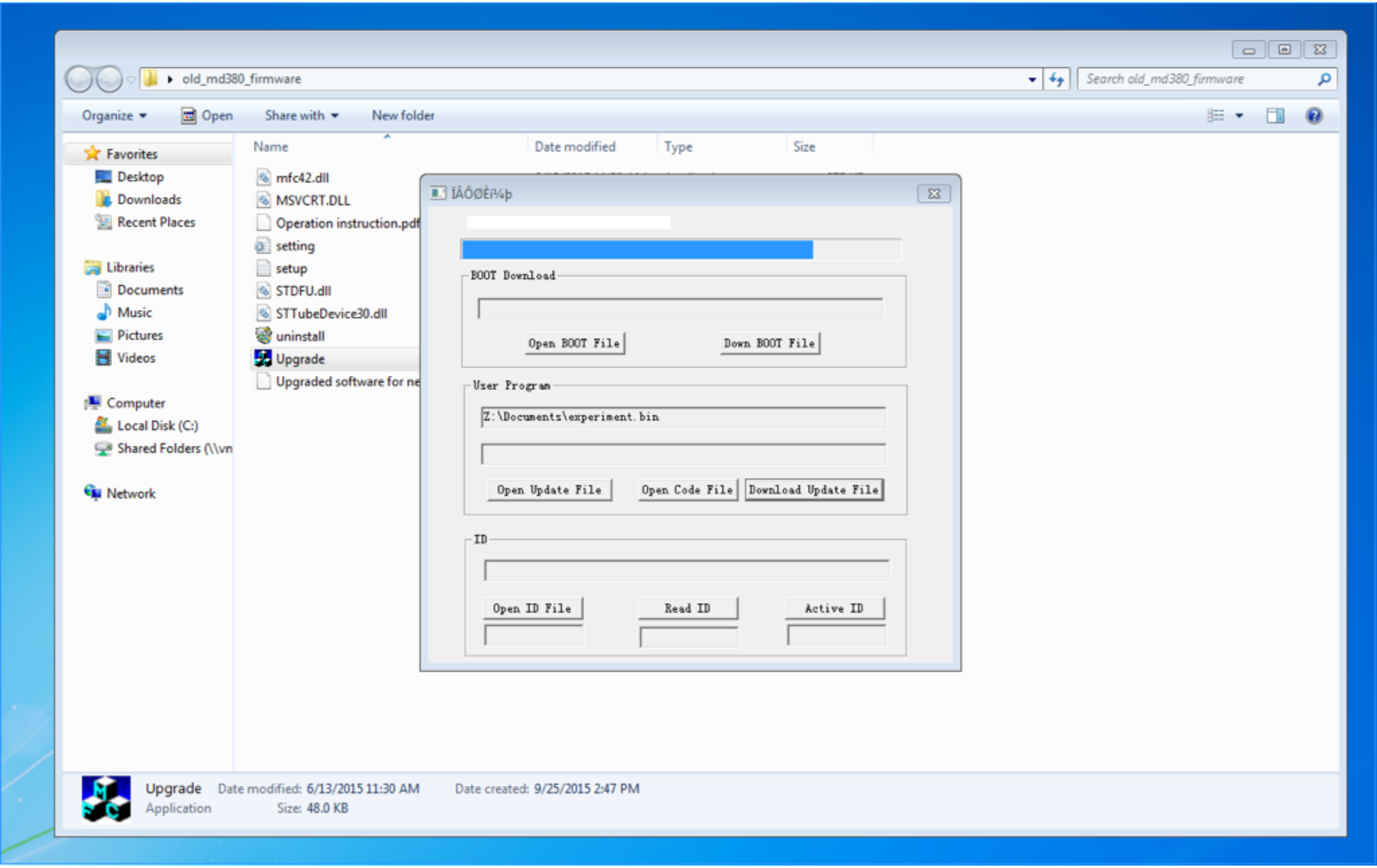
```
void FLASH_OB_RDPConfig(uint8_t OB_RDP){
  FLASH_Status status = FLASH_COMPLETE;

  /* Check the parameters */
  assert_param(IS_OB_RDP(OB_RDP));

  status = FLASH_WaitForLastOperation();
  if(status == FLASH_COMPLETE)
    *(__IO uint8_t*)
      OPTCR_BYTE1_ADDRESS = OB_RDP;
}
```

Bootloader Patch

```
[0x08000000]> pd 8 @ 0x080044a0
    0x080044a0    fdf7a0fd        bl rdp_isnotlocked
    0x080044a4    0028            cmp r0, 0
,=< 0x080044a6    04d1            bne 0x80044b2
|   ; Change this immediate from 0x55 to 0xAA to jailbreak
|   0x080044a8    5520            movs r0, 0x55
|   0x080044aa    fdf781fd        bl rdp_lock
|   0x080044ae    fdf78bfd        bl rdp_applylock
`-> 0x080044b2    fdf776fd        bl 0x8001fa2
    0x080044b6    00f097fa        bl bootloader_pin_test
[0x08000000]>
```

Application Dump

- Erase the whole device.
  - Device is a brick!
- Flash the patched bootloader.
  - Device is a blinking brick!
- Install Encrypted Update


old_md380_firmware
Search old_md380_firmware
Organize
Open
Share with
New folder
Favorites
Desktop
Downloads
Recent Places
Libraries
Documents
Music
Pictures
Videos
Computer
Local Disk (C:)
Shared Folders (\\vm
Network
Name
Date modified
Type
Size
mfc42.dll
MSVCRT.DLL
Operation instruction.pdf
setting
setup
STDFU.dll
STTubeDevice30.dll
uninstall
Upgrade
Upgraded software for ne
ÏÀÔØÈ¼p
BOOT Download
Open BOOT File
Down BOOT File
User Program
Z:\Documents\experiment.bin
Open Update File
Open Code File
Download Update File
ID
Open ID File
Read ID
Active ID
Upgrade
Application
Date modified: 6/13/2015 11:30 AM
Size: 48.0 KB
Date created: 9/25/2015 2:47 PM

Application Patching

- Decrypted application can be freely read,
  - because the bootloader was patched.
- Patch to:
  - Enable promiscuous mode.
  - Replace startup logo.

Tytera
ShmooCon

Tytera
ShmooCon

Sideloading an Applet

- The DMR stack is too complicate to easily rewrite.
- So let's inject a lot of new code!
  - Our main() runs before their main().
  - Our .data avoids their .data.
  - Hook any handy functions back to our code.

Free Flash

- We could look for regions of FFFFFFFF
  - Not much room, just ~5K.
- 中国排名第一。
  - 200KB Chinese Font!
  - All in executable flash!

main() calls main()

- Classy way:
  - Hook RESET interrupt, and redirect the table.
- Cheap way:
  - Let the MD380's RESET handler set its own interrupt table address.

Classy Way

```
static void abort_to_mfgr_app(void) {
        const uint32_t *app_base = (const uint32_t *)MFGR_APP_LOAD_ADDRESS;
        SCB->VTOR = MFGR_APP_LOAD_ADDRESS;
        do_jump(app_base[0], app_base[SIDELOAD_RESET_VECTOR]);
}

static void do_jump(uint32_t stacktop, uint32_t entrypoint)
{
        asm volatile(
                "msr msp, %0    \n"
                "bx     %1      \n"
                : : "r" (stacktop), "r" (entrypoint) : );
        // just to keep noreturn happy
        for (;;) ;
}
```

Cheap Way

```
//Minimalist bootloader.
void main(){
  //Function pointer to the application.
  void (*appmain)();
  //The handler address is the stored in the
  //interrupt table.
  uint32_t *resethandler =
          (uint32_t*) 0x0800C004;
  //Set the function pointer to that value.
  appmain = (void (*)()) *resethandler;
  //Away we go!
  appmain();
}
```

Avoid Overlapping Memory

- STM32F405 has 192K of RAM.
  - 2*64K of general SRAM
  - 64K of CCM--Core Coupled Memory
    - Not Executable, Not DMA-able
- Fill memory with 0xDEADBEEF, then dump live.

Memory Dump over USB

Terminal — ssh — 74×18

```
debian82% repeat 10 lsusb -d 0483:df11 -v | grep iManufacturer
  iManufacturer           1 @20000900 : 00100010
  iManufacturer           1 @20000904 : 00020004
  iManufacturer           1 @20000908 : 080992a4
  iManufacturer           1 @2000090c : 080fb404
  iManufacturer           1 @20000910 : 00000000
  iManufacturer           1 @20000914 : 00100010
  iManufacturer           1 @20000918 : 00010002
  iManufacturer           1 @2000091c : 0809b67c
  iManufacturer           1 @20000920 : 080fb410
  iManufacturer           1 @20000924 : 00000000
debian82%
```

Let's target a C Compiler!

- Write a linker script.
  - Code in 200KB of Flash from the Chinese font.
  - Data in 12KB of SRAM, 22KB of TCRAM.
- Could have a heap if we wanted one.

Linker Script to Match

```
/* Highest address of the user mode stack */
_estack = 0x20020000;    /* end of 128K RAM on AHB bus*/
                         /* Most bytes after 2001d40c are free. */

/* Specify the memory areas */
MEMORY
{
  FLASH (rx)      : ORIGIN = 0x0809D000, LENGTH = 396K /* Freed by Chinese Font */
  RAM (xrw)       : ORIGIN = 0x1000b000, LENGTH = 20K  /* Bytes after 1000a868 are free.*/
}
```

Hooking Thumb Functions

- Most functions are easy:
  - Find the BL (Branch/Link) in the caller and patch it.
- Some functions are harder:
  - USB handlers are passed as function pointers.
  - A stub can branch to the new handler.

```
0x080969de    0148    ldr r0, [pc, 4]    ; [0x80969e4:4]=0x809d33d
0x080969e0    0047    bx r0              ; Branch to it.
0x080969e2    1e49
0x080969e4    3dd3                       ; Low ADR.
0x080969e6    0908                       ; High ADR.
```

Calling Functions

- ARM has a nice, consistent calling convention.
- It's easier to call existing drivers than to rewrite them.
- Just cast a function pointer:

```
int (*spiflash_read)(char *dst, long adr, long len)
    = 0x0802fd83;
```

Firmware Reversing

- IDA Pro, HexRays, and Radare2
- No symbol names!
  - Not even for libc.
  - But I/O registers, IVT are known.

µC/OS II

- Real Time Operating System
- Threading, Interrupt Handling
- Helpfully names the threads!

Threads

```
signed int THREADS_START()
{
  char v1; // [sp+14h] [bp-Ch]@1

  OSTaskCreateExt(thread_sysinter, 0, &stack_sysinter, 6, 6, &bos_sysinter, 512, 0, 3);
  ucos_namethread(6u, "Sys Inter", &v1);
  OSTaskCreateExt(thread_rtctimer, 0, &stack_rtctimer, 0x13, 0x13, &bos_rtctimer, 512, 0, 3);
  ucos_namethread(0x13u, "RTC Timer", &v1);
  OSTaskCreateExt(thread_callprocess, 0, &stack_callprocess, 0x10, 0x10, &bos_callprocess, 512, 0, 3);
  ucos_namethread(0x10u, "Call Process", &v1);
  OSTaskCreateExt(thread_FMTxprocess, 0, &stack_FMTxprocess, 0x11, 0x11, &bos_FMTxprocess, 512, 0, 3);
  ucos_namethread(0x11u, "FMTx Process", &v1);
  OSTaskCreateExt(thread_RFPLL, 0, &stack_RFPLL, 5, 5, &bos_RFPLL, 256, 0, 3);
  ucos_namethread(5u, "RF Pll", &v1);
  OSTaskCreateExt(thread_pctune, 0, &stack_pctune, 0x14, 0x14, &bos_pctune, 512, 0, 3);
  ucos_namethread(0x14u, "PC Tune", &v1);
  OSTaskCreateExt(0x809573D, 0, &stack_ledprocess, 0x15, 0x15, &bos_ledprocess, 256, 0, 3);
  ucos_namethread(0x15u, "LED Process", &v1);
  OSTaskCreateExt(thread_beepprocess, 0, &stack_beepprocess, 0x16, 0x16, &bos_beepprocess, 256, 0, 3);
  ucos_namethread(0x16u, "Beep Process", &v1);
  OSTaskCreateExt(thread_AfMute, 0, &stack_afmute, 0x17, 0x17, &bos_afmut, 256, 0, 3);
  ucos_namethread(0x17u, "Af Mute", &v1);
  OSTaskCreateExt(thread_timeslotinter, 0, &stack_timeslotinter, 7, 7, &bos_timeslot, 512, 0, 3);
  ucos_namethread(7u, "TimeSlot Inter", &v1);
  OSTaskCreateExt(thread_vocoder, 0, &stack_vocoder, 0xD, 0xD, &bos_vocoder, 256, 0, 3);
  ucos_namethread(0xDu, "Set Vocoder", &v1);
  OSTaskCreateExt(thread_chaccess, 0, &stack_chaccess, 0xE, 0xE, &bos_chaccess, 256, 0, 3);
  ucos_namethread(0xEu, "ChAccess Pr", &v1);
  OSTaskCreateExt(thread_statechange, 0, &stack_statechange, 0xF, 0xF, &bos_statechange, 256, 0, 3);
  return ucos_namethread(0xFu, "State Change", &v1);
}
```

I/O Registers

- I/O Region at 0x40000000.
  - SPI Ports for the Baseband and SPI Flash
  - GPIO for LEDs and the Keypad
- All defined as structs in officially available headers.

Contacts, Groups, SMS

- Radio "codeplug" is stored in the SPI Flash.
  - Reverse engineered by changing settings.
- spiflash_read() is called with item address
  - Nice and easy to identify!

xrefs to ROM:0802FD82

Directic	Typ	Address	Text	
Up	p	startup_botline+C	BL	spiflash_read; Reads a big chunk of data from an address in the
Up	p	startupflags_read+C	BL	spiflash_read; Reads a big chunk of data from an address in the
Up	p	spiflash_read_2100_t...	BL	spiflash_read; Reads a big chunk of data from an address in the
Up	p	spiflash_read_5a70+C	BL	spiflash_read; Reads a big chunk of data from an address in the
Up	p	emergency_readbyin...	BL	spiflash_read; Reads a big chunk of data from an address in the
	p	contact_read+10	BL	spiflash_read; Reads a big chunk of data from an address in the
D...	p	listengroup_load_by_i...	BL	spiflash_read; Reads a big chunk of data from an address in the
D...	p	listengroup_load_by_i...	BL	spiflash_read; Reads a big chunk of data from an address in the
D...	p	scanlist_readfromspi...	BL	spiflash_read; Reads a big chunk of data from an address in the
D...	p	memory_readbyindex...	BL	spiflash_read; Reads a big chunk of data from an address in the
D...	p	memory_loadbyindex...	BL	spiflash_read; Reads a big chunk of data from an address in the
D...	p	memory_thing+14	BL	spiflash_read; Reads a big chunk of data from an address in the
D...	p	bank_loadbyindex+14	BL	spiflash_read; Reads a big chunk of data from an address in the
D...	p	bank_namefromindex...	BL	spiflash_read; Reads a big chunk of data from an address in the
D...	p	emergency_loadbyind...	BL	spiflash_read; Reads a big chunk of data from an address in the
D...	p	read_59c0_by_index+...	BL	spiflash_read; Reads a big chunk of data from an address in the
D...	p	text_loadtemplatebyi...	BL	spiflash_read; Reads a big chunk of data from an address in the
D...	p	pastend_spiflash_rea...	BL	spiflash_read; Reads a big chunk of data from an address in the
D...	p	pastend_spiflash_rea...	BL	spiflash_read; Reads a big chunk of data from an address in the
D...	p	readactivechannel+C	BL	spiflash_read; Reads a big chunk of data from an address in the
D...	p	pastend_spiflash_rea...	BL	spiflash_read; Reads a big chunk of data from an address in the
D...	p	pastend_spiflash_rea...	BL	spiflash_read; Reads a big chunk of data from an address in the
D...	p	text_loadfrompasten...	BL	spiflash_read; Reads a big chunk of data from an address in the
D...	p	pastend_read_45100...	BL	spiflash_read; Reads a big chunk of data from an address in the
D...	p	spiflash_read_450a8...	BL	spiflash_read; Reads a big chunk of data from an address in the
D...	p	pastend_read_450a4...	BL	spiflash_read; Reads a big chunk of data from an address in the
D...	p	spiflash_read_unkno...	BL	spiflash_read; Reads a big chunk of data from an address in the
D...	p	spiflash_read_2ffd0+14	BL	spiflash_read; Reads a big chunk of data from an address in the

Help Search Cancel OK

Line 29 of 69

Reading a Contact

- 36-byte entries, starting at 0x5F80.
- 1-indexing looks weird.

```
// 08021AEC
int contact_read(_BYTE *dst, uint16_t index)
{
  int v3;

  spiflash_read(dst, 36 * index + 0x5F5C, 36);
  return v3;
}
```

Graphics Reversing

- Graphics functions take lots of parameters
- Two strings are displayed at startup:
  - Known addresses in SPI Flash
  - spiflash_read() reveals calls to gfx_drawtext().
- I don't know the raw protocol, don't need it.

DMR Protocol

- Link Local Addresses
  - 24-bits, masked by 0x00FFFFFF.
- Arrays read from SPI Flash
  - listengroup[]
  - dmr_myaddress

Old RX Logic

- If the destination is my address; or,
  If the destination is in one of listengroup[32]
- Then
  - Play audio.
- Else
  - Mute audio.

```
(dmr_myaddress & 0xFFFFFF) == dst_llid_adr )
```

```
for ( i = 0; i < 0x20u; ++i ){
  if ( (listengroup[i] & 0xFFFFFF)
       == dst_llid_adr ) {
    something = 16;
    recognized_llid_dst = dst_llid_adr;
    current_llid_group = var_lgroup[i+16];
    sub_803EF6C();
    dmr_squelch_thing = 9;
    if ( *(v4 + 4) & 0x80 )
      byte_2001D0C0 |= 4u;
    break;
  }
}
```

Patching RX Logic

- Promiscuous Receive
  - Destination address is always accepted.
  - First hardware scanner for DMR audio!
- Released in PoC||GTFO 10 at Shmoocon.

USB Reversing

- Tytera worked from STMicro's public examples.
- Original C code is available, but a bit tricky:
  - Code uses tables of function pointers for handlers.
- Very nice to hook, external comms at last!

USB Reversing

- Device Firmware Update Protocol
  - Data UPLOAD or DNLOAD starts at Block 2.
  - Block 0 is special
    - DNLOAD applies settings.
    - UPLOAD reads status.
- Let's hook Block 1!

Hooked UPLOAD

```
/* The DFU protocol specifies reads from block 0 as something
   special, but it doesn't say way to do about an address of 1.
   Shall I take it over?  Don't mind if I do!
 */
if(blockadr==1){
  usb_send_packet(iface,   //USB interface structure.
                  (char*) *((int*)0x2000112c), //2.032
                  length); //Length must match.
  return 0;
}

//Return control the original function.
return usb_upld_handle(iface, packet, bRequest, something);
```

Hooked DNLOAD

```
/* DFU transfers begin at block 2, and special commands hook block
   0.  We'll use block 1, because it handily fits in the gap without
   breaking backward compatibility with the older code.
 */
if(*blockadr==1){
  switch(packet[0]){
  case TDFU_PRINT: // 0x80, u8 x, u8 y, u8 str[].
    drawtext((wchar_t *) (packet+3),
             packet[1],packet[2]);
    break;
  }

}else{
  /* For all other blocks, we default to the internal handler.
   */
  return usb_dnld_handle();
}
```

Host Software

- Python tools are great for experimentation.
- They suck for deployment to amateurs.

14:26

MD380Tool

CONNECT TO MD380

Connecting

MD380Tool

Allow the app MD380Tool to access the USB device?

Use by default for this USB device

CANCEL OK

20:36

MD380Tool

.....
Call from 3147092 to 9205 ended.
....
Call from 3147092 to 9205 ended.
.....
Call from 3147092 to 6968 ended.
.....
Call from 3147092 to 6 ended.
.....
Call from 3147092 to 5 ended.
.....
Call from 3147092 to 4 ended.
.....
Call from 3147092 to 3100 ended.
.....
Call from 3147092 to 3185 ended.
.....
Call from 3147092 to 3166 ended.
.....
Call from 3147092 to 3185 ended.
....
Call from 3147092 to 3100 ended.
SMS header: 08 62 02 40 00 00 63 30 05 54 88 00 83 0c
Data: 08 7a 45 00 00 5c 00 0d 00 00 40 11 5c 9e
Data: 08 72 0c 30 05 54 0c 00 00 63 0f a7 0f a7
Data: 08 72 00 48 6a dd 00 3e e0 00 94 04 0d 00
Data: 08 72 0a 00 54 00 68 00 69 00 73 00 20 00
Data: 08 72 69 00 73 00 20 00 61 00 20 00 74 00
Data: 08 72 65 00 78 00 74 00 20 00 66 00 72 00
Data: 08 7a 6f 00 6d 00 20 00 4b 00 4b 00 34 00
Data: 08 7a 56 00 43 00 5a 00 21 00 0c 63 30 35

FETCH DMESG