Regarding Fusion

Category: Fusion
October 22, 2022
Some thoughts about Fusion, digital modes, and the future.

Fusion uses the lower layers of the P-25 architecture. Modulation, transport, framing, etc. The primary differences revolve around what's in the packet. For example: The use of call signs instead of unit IDs. So Fusion shares most of P-25. FWIW, Fusion is also very similar to DMR. The primary difference is that it doesn't use TDMA – jamming two separate channels into the space of one.

P-25, DMR, NXDN(Kenwood), IDAS (Icom), and Fusion use the same vocoder. D* uses a much older and obsolete vocoder.

Fusion and D* have been optimized, although differently, for Amateur Radio. DMR, NXDN, and IDAS have been optimized for commercial services, like taxi cabs. P-25 has been optimized for government services, like police departments.

Fusion was actually designed by Motorola in that period of time that Motorola owned Yaesu. This benefits Fusion since the vocoder and the signal processing firmware are pretty much the same as what Motorola uses in their radios. (Believe me, not all signal processing firmware is designed the same!) Thus, the performance of Fusion is pretty darn good. Add to that an additional layer for FEC provided by Yaesu's modification to P-25, and the performance of Fusion is better than all the others. (D* is substantially worse than the other protocols.)

What we see from this is that most of the communication industry has evolved around the same technologies even though the systems are not directly compatible with one another. This is why it is so easy for a hotspot to translate between DMR, P-25, NXDN, IDAS, and Fusion. But is this the best we can do?

No. Digital performance could be much better. C4FM already blows away FM with superior range and capability, but it was intentionally limited – to work with FM/PSK transmitters and receivers like those used with FM.

FM is a non-linear system thus it does not transmit all the information it could in the available bandwidth. Linear systems, such as OFDM, can provide higher bit rates lower error rates, better resistance to fading and multipath, amongst other things. The problem is OFDM requires linear amplifiers and thus is not compatible with all the FM/PSK transmitter and receiver designs.

Personally, I'd like to see OFDM as an option for digital's future.

But all digital systems could be made better by some enhancements at the repeater site. The use of multiple, diversity receivers incorporating very low-level discrimination could improve receiver sensitivity by as much as 10 dB. That means HT would perform almost as well as a mobile rig with no changes required of the user. Having multiple diversity receivers at various locations around a city could mean that a low power HT would work virtually anywhere in that city. Sadly, there doesn't seem to be a market for doing such things. (This technology is well understood and used by your cell phone.)

One last note. Several attempts have been made to build multi-protocol radios incorporating DMR, Fusion, D*, FM, and others into a single unit. Should be simple, right? But we have yet to see such radios on the market. If we did, they wouldn't do Fusion as well as Yaesu, or D* as well as Icom, or DMR as well as Motorola. Why? Protocols are damn hard to get right. They are even harder to get working really, really well where they take into account all the things that can go wrong with data transmission. (Hotspots try to do the conversion, but never work as well as going “native”.)

I would argue that the amateur community should focus on those systems designed for Amateur Radio. Ultimately those will be easier to use and better suited for the job. (Police departments and taxi cabs should not use Fusion!) A good user experience is critical if digital, any digital, is to become generally accepted.

One last note: Nothing prevents Icom from building a Fusion radio. Every last little detail about Fusion is public information, particularly since most of it comes from P-25 – for which Icom builds radios. The only proprietary aspects of Yaesu’s system are WiRES-X and IMRS.

Purchasing amateur radios that do digital help support the future of digital by encouraging companies to develop more and better digital features. Buying commercial Chinese radios just helps the Chinese economy.

Understanding DGID and IMRS

Category: IMRS
October 22, 2022
DG-ID is like a CTCSS tone, but for digital. It filters out signals with different DGID’s as well as having the ability to do split DG-ID’s (like split CTCSS). DG-ID can be in a state of Rx & TX (DG-ID), Rx Only, Tx Only, and Split Rx & Tx (different DG-ID).

DG-ID servers as a means of selective signaling for access to a repeater or for a WiRES-X node. For the repeater it has a couple of different applications (IMRS and access purposes) while with WiRES-X it offers control of voice and control access.

DG-ID’s cannot be reused within the same configuration or network. For a repeater that means the DG-ID can only be in the LOCAL, RPT GROUP, or GROUP. There will be issues if in an IMRS network has anything with the same DG-ID.

DG-ID Repeater usage

LOCAL DG-ID: There are two factors:
1) The DG-ID it will not be sent down IMRS links and only allow access to the local repeater.
2) In an IMRS set-up, you will always hear the status beeps: One for end conversation; Two for link drop/end; and three for a repeater is not connecting with the DG-ID/IMRS group.

RPT GROUP DG-ID: This offers the ability for an IMRS link to be established in a one-way direction. The DG-ID used is the LOCAL DG-ID of the other repeater that you are calling. If users are using DG-ID Tx AND RX they will not hear beeps the confirmation beeps (see above).

Example of RPT GROUP usage:

San Diego Repeater – Local 1   Los Angeles Repeater – Local 2, RPT GROUP 1

San Diego users using DG-ID 1 stay on San Diego for usage; Los Angeles users use DG-ID 2 stay on Los Angeles for usage. Los Angeles users who use DG-ID 1 will establish a one-way link to San Diego and will communicate with one another until the IMRS TOT is reached. Once the link drops it is reestablished by someone using DG-ID 1 on the Los Angeles repeater group as San Diego cannot establish the link as it is not a RPT GROUP.

GROUP DG-ID: offers the ability for an IMRS link to be established in either direction by any of the repeaters that have it programmed into IMRS. If users are using DG-ID Tx AND Rx they will NOT hear the confirmation beeps.

Example of GROUP usage:

San Diego – Local 1, GROUP 10 Los Angeles Repeater – Local 2, GROUP 10

San Diego users use DG-ID 1 only stay on San Diego for usage. Los Angeles users use DG-ID 2 only stay on Los Angeles for usage. San Diego or Los Angeles users that use DG-ID 10 will connect to one another via IMRS and stay connected till the IMRS TOT times out.

Overall example with LOCAL, RPT GROUP, and GROUP:

San Diego – LOCAL 1 – RPT GROUP (blank), GROUP 10

Los Angeles – LOCAL 2 – RPT GROUP 1 – GROUP 10

San Diego: DG-ID 1 talks locally, using DG-ID 10 talks between San Diego and Los Angeles

Los Angeles: DG-ID 2 talks locally, using DG-ID 10 talks between San Diego and Los Angeles. DG-ID 1 establishes a link between San Diego and Los Angeles.

WiRES-X & DG-ID Local

Using WiRES-X with a DG-ID, it is not recommended to use the

LOCAL DG-ID if you have an IMRS setup with a DG-ID. Here’s why: WiRES-X will hear all the confirmation beeps from IMRS (argh!). Also WiRES-X will hear everyone but only talk locally since a LOCAL DG-ID is being used.

Example of WiRES-X and IMRS (with three repeaters in an IMRS Network)

- San Diego – LOCAL 1 – RPT GROUP (blank) – GROUP 10, 20, 99
- Los Angeles – LOCAL 2 – RPT GROUP 1, 3 – GROUP 10, 30, 99
- Chicago – LOCAL 3 – RPT GROUP (blank) – GROUP 20, 30, 99
- WiRES-X – DG-ID 99
- DG-ID 10 connects San Diego to Los Angeles
- DG-ID 20 connects San Diego to Chicago
- DG-ID 30 connects Los Angeles to Chicago
- DG-ID 99 connects all together along with WiRES-X

In this setting the users can do the following, per repeater, with the DG-ID’s.

San Diego:

- DG-ID 1 is local.
- DG-ID 10 establishes a link between San Diego and Los Angeles.
- DG-ID 20 establishes a link between Los Angeles and Chicago.
- DG-ID 99 establishes a link between San Diego, Los Angeles, Chicago and WiRES-X.

Los Angeles:

- DG-ID 2 is local.
- DG-ID 1 establishes a one-way link between Los Angeles and San Diego.
- DG-ID 3 establishes a one-way link between Los Angeles and

Chicago.
- DG-ID 10 establishes a link between San Diego and Los Angeles.
- DG-ID 30 establishes a link between Los Angeles and Chicago.
- DG-ID 99 establishes a link between San Diego, Los Angeles, Chicago and WiRES-X.

Chicago:

- DG-ID 3 is local.
- DG-ID 20 establishes a link between San Diego and Chicago.
- DG-ID 30 establishes a link between Los Angeles and Chicago.
- DG-ID 99 establishes a link between San Diego, Los Angeles, Chicago, and WiRES-X

Once an IMRS link is established ALL the DG-ID's in the repeater can be used to keep the link active. If in the San Diego example a link is established, then DG-ID's 1, 10, 20, and 99 can all be used to keep the link alive. DG-ID that established the link is the only one that will be transmitted by San Diego.

If a link established by DG-ID 10 on the San Diego repeater, the repeater will Rx DG-ID's 1, 10, 20, and 99 but will only transmit back DG-ID 10. Therefore, if you have DG-ID 99 set-up in your radio for Tx & Rx you will be able to talk down the link but you will not hear anything because the repeater is Tx DG-ID 10.

Information from N9UPC.

Repeater Lock-Ups

Category: Hotspots,Repeaters,WiRES-X
October 22, 2022
Many people have reported Fusion repeater lock-ups that required the repeater to be shut down (power removed) and rebooted. I've had repeater lock happen as well with an HRI-200 directly connected to the DR2X. In this particular case I believe the event was triggered by sending a number of commands from the radio (such as the GM mode) which resulted in a race condition within the firmware.

I have previously observed that invalid data (I.e., with a lot of bit errors or badly formed packets from a hotspot) can also cause the lock-up. My assumption is that this is a problem of not validating inputs to a function prior to executing the function. The invalid data is not handled correctly and causes one of the lock-up faults I mention below to occur.

I have observed that data is more likely to be invalid if there is a lot of noise on the signal. Also a major source of problems are badly behaving hot spots that inject all sorts of crap into the data stream. There's no control over these devices and it just gets worse as there is more work being done converting from one thing to another thing, etc.

Communications software is EXTREMELY HARD to write and much of what I see out there is poorly documented (which leads to bad code) and poorly tested.

Reflectors such as YSF or FCS are pretty dumb when they relay signals. They look for an incoming packet from one of the connected hotspots and just send that same packet out to all the other hotspots (including bridges to WiRES-X). Garbage in, garbage out. To address this I have modified the YSF reflector software to be very aggressive at dropping packets with CRC errors, invalid data in fields, and most importantly invalid FICH headers. The FICH header is extremely important for everything to work right. (You can look up FICH headers in the System Fusion documentation published by Yaesu.) This modified software is running on MNWis, Princeton Kentucky, MVARC (Idaho) and America Link. It has dramatically reduced the number of problems with WiRES-X problems.

So what is a “lock-up”. A lock-up occurs when the processor looses track of what it was doing and either halts or ends up executing invalid code. In either case it is no longer processing the code it was intended to run. Events that cause the processor to, using a technical term, go out to out to lunch can result of errors in the software that cause it to happen, corrupted memory, invalid states in a state machine, incorrectly handled race conditions, or a really nasty thing called priority inversion (where a high priority task which is pre-empting use of the CPU is waiting for a low priority task to complete, which can’t complete because, well, it’s not high priority.)

How to solve the problem?
First of all, the repeater firmware should never crash. Since it is very hard (but not impossible) to write code with no bugs, most continuously operating real-time systems will incorporate a “watch dog timer”. It is the purpose of this timer to detect when the “wheels have fallen off the processor” and perform a reset.

Since we can’t modify the repeater’s software, there are some actions we can take.

First would be to prevent bad data from getting into the system. A careful examination of the logs possibly combined with audio recordings could be helpful.

A quick fix might be to install a timer on the power supply such that the repeater’s power is removed for a short time once or several times a day.

I have a technical concept for a generic lock-up detector that would reboot the repeater in the event of a lock-up without the need to make any modifications. Not sure if I’ll work on this because I don’t know how much interest there would be. This solution would also work with non-Yaesu repeaters.

One further note. I made an observation that this problem was more likely to occur on 2 meters than on 440. Why? My theory is that there is a lot more interaction with the environment on 2 meters. Computer networks, small power supplies, computers, almost everything digital generally creates a lot more noise on 2 meters than it does on 440 MHz. I observed a situation where mixing products were occurring on a 2 meter repeater in a noisy environment. This repeater would lock up regularly. My theory was that the mixing products contained a lot of noise along with the C4FM signal causing a high number of bit errors and thus crashing the repeater. This was observed on a DR1X.

If you are struggling with this problem, I suggest you contact Juan at Yaesu Customer Service with your details. Provide Juan with as much information as you can.

Running two HRI-200’s on one

IP Address

Category: WiRES-X
October 22, 2022
I use VyprVPN from GoldenFrog. It is necessary to turn of NAT for your account. That means that each VPN connection is a raw connection straight to the Internet. YOU HAVE NO PROTECTION OTHER THAN WHAT YOU PROVIDE!!! (The same company will rent you a VPN that you can host on your own cloud provider where you can control port forwarding – but I suspect this is well beyond most folks.)

I've been using this solution for over a year. If you've ever seen me demonstrate WiRES-X, I'm either using AT&T/cell or the local Wifi along with the VPN to get access to the incoming ports.

Good news! My fully patched and maintained Win7 boxes have not yet been hacked. One of which has been online via VPN for over a year.

Most VPNs won't work because they don't give users their own IP. The VPN host shares IPs between multiple users which means they can support more users with fewer IPs (and better obscures your traffic). I'm only aware of two VPN providers that allow you to specify incoming ports.

The previous post of using the Ham's IP space is a good one. I plan on giving that a try when I have time. But VyprVPN is very much plug-and-play.

Note: You will need to PAY for VyprVPN (~$70/yr). The free service does not allow you to disable NAT.

What Does the HRI-200 Actually Do?

Category: WiRES-X
October 22, 2022
What’s the difference between using an HRI-200 and directly connecting to the radio?

1. It is the only option for connecting a repeater to WiRES-X.
2. It uses UDP communications sent directly to the Room. So network communications are very efficient.

PDN uses TCP instead of UDP. (Why use UDP? UDP is simple. If anything gets lost, it is dropped. Good for real-time communications. If a packet is missed with TCP, it will keep retrying up to a limit which is not good for real-time. VoIP, VPN’s, Netflix, etc. all use UDP.)

The PDN TCP packets (non-HRI-200) are sent to a server in Japan. This server converts the packets to UDP and then sends them to the selected room in the same way the HRI-200 does. So by using the HRI-200 you bypass a drip to Japan and a potentially problematic conversion from TCP to UDP and back to TCP for the return trip to your station.

Yaesu uses the TCP approach to get around forwarding of ports. The WiRES-X PDN software establishes a TCP connection with the Yaesu server. It’s through that connection stations can contact you even though you don’t have an open port. With the HRI-200, stations just get your IP address and send you packets directly. This is why PDN cannot efficiently host a room – all room traffic would have to go to Japan for the conversion process.

In a previous post (2016) I described what the HRI-200 does. Why it is the way it is is somewhat historical from the history of WiRES, WiRES-II, and now WiRES-X. Obviously an Ethernet port on the back with an embedded processor to run things would be a better way to go. I can only guess that the ROI is simply not there for Yaesu. They do, after all, have to make a profit.

One further note. It’s probably a good thing non-Yaesu equipment doesn’t connect to WiRES-X. The biggest problem in bridging hotspots (YSF, FCS, etc.) to WiRES-X is that the hotspots are bad neighbors. They generate bad packets, do crazy things, and can really trash a network. As a room operator we have no control over who tries to connect, so at least the Yaesu equipment has gone through compatibility testing. Believe me, very little testing is done on hotspot software.

MNWis (21493) bridges to YSF (21493) via a YSF Reflector where I have extensively modified the code to drop bad packets, control information, and all manner of things that shouldn’t see there way into the WiRES-X node. I had to do this because we were constantly having hotspots connect and mess everything up. It got very, very tiring working with the (mostly new) hotspot owners to fix their problems. I mean the conversations were endless and constant! So I just dropped anything that looked bad in the YSF Reflector software and the problems went away as well as a few hotspots who could no longer connect or talk to people.

System engineering is critical. We need Yaesu to play the roll of system engineering to keep all the Fusion products working with each other. It’s so nice that all the Fusion equipment basically works the same way and does the same thing all of the time!

BTW, you can bring your questions/comments to the MNWis Fusion Technical Net held Monday’s at 7:30 PM Central in the above mentioned locations.

WiRES-X Online/Offline

Category: WiRES-X
October 22, 2022
When a WiRES-X node keeps going online then offline, this type of problem is almost always the result of a networking issue. A networking problem will occur and will wack out the HRI-200 software. Once the software has been wacked out, it will continue to cycle in and out of whatever room it is connected to. Restarting the WiRES-X software usually solves the problem.

Router problems:
1. Turn off UPnP. It should never be used!!!!!!
2. Turn off all quality of service features like those to improve Xbox gaming or VoIP functions. (Yes I know we're using VoIP so you'd think it would help.)
3. In fact turn off everything you don't have to have.
4. Some routers are total crap and need to be rebooted every few weeks.
5. You're not using Wi-Fi are you? Wired is always better than wireless – we should know! Drop-outs in Wi-Fi can cause this issue.

ISP problems:
1. Something as simple as the ISP dropping your connection for a few minutes in the middle of the night when they're doing maintenance.
2. The ISP just doing a bad job of getting packets to you during prime evening Netflix viewing.

3. The ISP switching your IP address. (I think CenturyLink just does this for the fun of it.)

Computer:
1. Make sure Win10 doesn’t put the USB interface to the HRI-200 to sleep. It really, really likes to do this. Just going to the obvious place in the control panel to turn this feature off doesn’t really turn it off. See the documents on running Win 7/10 24x7x365 remotely in the Fusion Help section at HamOperator.com.
2. Make sure the computer isn’t crap. Cheap computers may not have quite the reliable communication interfaces that we need.
3. Banish all RFI and ground loops. Make sure RF is not getting into the USB interface to the HRI-200. This can cause message errors and really screw things up.
4. Make sure you computer actually has enough power to run the HRI-200 reliably. Get one of those USB power thingies and make sure you have a solid 5VDC under load.

Over the air:
1. Bad data on the WiRES-X network can mess things up. This happens mostly with hotspots that are bridged into the WiRES-X room. On MNWis and a few other networks we’ve banished this problem by banishing FCS and running YSF server software that banishes bad data from hotspots. So if you have this problem in one room, say AmericaLink and not others, there may be nothing you can do.
2. Yaesu radios that are running really old firmware or have not had the firmware updated correctly (I.e., “I did the main CPU but I’ll get around to the DSP later.”)

Side note:
The port check should not be relied upon to definitely prove it’s working or not working. The test is not exactly the same as actually communicating with the Yaesu list servers and the room

server. So it can lie to you with false positives and false negatives.

You may also experience a similar problem if the Yaesu list servers are not able to keep up with the demand or if maintenance is being done. (When they do maintenance in the wee hours of the morning, that’s right in the middle of the day for us.)

And keep in mind the MNWis Fusion Technical Net held in MNWis WiRES-X and YSF # 21,493 Monday nights at 7:30 PM Central.

WiRES-X and USB Problems

Category: WiRES-X
October 22, 2022
If you’re having problems with USB problems when using WiRES-X with or without other programs, read on.

The WiRES-X software is pretty good at identifying the HRI-200 vs. connecting directly to the radio. Windows, however, is very bad at managing USB ports. The WiRES-X software makes two USB connections: One as a virtual COM port and the other as a sound device. Windows is also very bad at managing sound devices.

Virtual COM ports can be identified by the port number (a real pain as it is always changing – see below) or by identifying the hardware via the manufacturer and product number that is burned into the USB device. Generally software that does not provide an option to specify the COM port (as with the HRI-200) is using the hardware identification to establish the association.

When Windows sees a USB device it will enumerate it and create a driver for it. If you move it to another port it will, for reasons we can only guess at, create another driver. It will keep making new drivers as long as you have USB ports and USB devices left. Why, we can only guess. Periodically it may be necessary to go through a clean out all of these “drivers” that Windows keeps making and saving until the end of time. You can do this from the command line or from within Device Manager.

I have found WiRES-X to play nicely with other devices. For example I have no problem running hotspots, Ham Operator Deluxe, WSJT-X, etc. on the same computer that is running WiRES-X.

You may find it beneficial to use a USB 2.0 vs. 3.0 port. Why? USB limits the number of connections that can made per USB controller. Keep in mind that several of the ports on you computer may use the same controller. Also, each USB device may use more than one connection. As I recall sound devices generally use two connections: One for the audio and one for control. A USB 2.0 controller supports 255 connections. How do I know? I actually ran up against this limit using a couple of 16 port USB hubs. USB 3.0, while faster, does not permit as many connections. I don’t remember the exact number as it has been some time, but the number SIX sticks in my brain. As you might imagine, it is very easy to burn up six connections. Also keep in mind that the USB controller may be servicing hardware that is internal to the computer. It all depends on how the computer was designed. Top end computers will generally have a higher controller to port ratio. Bottom line: you may want to use USB 2.0 if you have been using USB 3.0.

Perhaps try something like this:
1. Blow away the WiRES-X and any conflicting software. Make sure everything is unplugged from USB.
2. Blow away all of the USB drivers. Just uninstall them. Don’t

worry, the files to recompile a driver are still on the computer. [Device Manager with admin. Enable option to view all devices. Delete drivers under COM ports. Delete drivers under sound devices (except for hardware built into your computer.)]
3. Install the WiRES-X software. Allow it to associate with the USB devices it needs by plugging in the HRI-200, etc. If you're not using an HRI-200, you may find it beneficial to use an external USB audio device for WiRES-X. If you can use USB 2.0 ports.
4. With WiRES-X happy running, install the other software. Plug in the necessary USB devices so it can similarly make the association. At this point everybody should be happy with both programs running just fine.
5. Write down which device was plugged into which USB port. In the future, always use the same USB port for the devices. We don't want Windows creating more drivers.

Good luck solving the USB problems!

YSF Reflector Software

Category: Fusion
October 22, 2022
The HamOperator YSF Reflector is currently supporting MNWis, US Kentucky, and America-Link. A test reflector runs at US K9EQ.

A project has been going on at the HamOperator to improve the YSF Reflector software. Why do this?

The original software is very basic and offers very little capability and control. It also creates a HUGE problem when trying to bridge YSF hotspots to WiRES-X rooms because it just passes the data along with no filtering. That means whatever garbage gets sent to the reflector, it doesn't even need to be Fusion, that same data is sent to every connection including the bridged WiRES-X node. When the WiRES-X node gets this data it can misbehave or crash. Hence a project to fix these problems.

And while we're fixing problems, why not make it better by giving it more capabilities?

Current differences between the standard YSF Reflector software and the K9EQ version:

- Decodes all meta data including FICH and all data fields for every mode.
- Extensive logging allows selected logging of parameters needed to determine "what's going on?".
- Filters to drop packets if:
    - The FICH is invalid
    - The FICH does not pass sanity checks
    - Data Wide packets
    - Wires-X control packets
- Outputs a text file of connected nodes to simplify dashboards.
- Performance reporting
- Options can be changed in a *.ini file
- Ability to host multiple YSF Reflectors on a single server
- Internal documentation of the program (comments)
- Several builds are available:
    - Windows
    - Linux
    - Debian server (MNWis and US Kentucky are running in the "cloud")

Future enhancements will include:

- Drop problematic WiRES-X packets with invalid room/node number in the CSD1 field
    - Future improvement: Edit the packets to remove the problematic information
- Ability to black list callsigns and IP addresses.
- Kerchunk filter to drop short key-ups
- Remote control and status reporting of the reflector via Fusion text messages
- Integrated dashboard removes need for php and greatly reduces CPU overhead and network bandwidth
- Programmed reporting of audio levels informing users if their mic level is too high or too low
- Integration with IMRS
- New, engineered and reviewed API to more tightly integrate hotspots with the reflector
- Ability to programmatically send messages
- Ability to programmatically edit data fields on the fly (i.e., change GPS data, call sign, etc.)
- Improve internal program documentation
- Complete program re-write in Python. This will provide the ultimate in portability

The reflector will not support non-Fusion communications such as DMR or D*. The reason for this is that non-Fusion systems do not include all of the data that Fusion provides. This would force the reflector to become the lowest common denominator resulting in technology and feature restrictions.

The intent is for our version of YSF Reflector to be open sourced. It will be developed and enhanced with feedback from the Fusion community. By using good engineering practices, our hope is to provide a high quality service that equals or exceeds the existing quality of the Fusion/WiRES-X system. This is an enhancement to WiRES-X, not a replacement.

YSF WiRES-X Unintended Node Switching

Category: Hotspots,WiRES-X
October 22, 2022
A problem exits with YSF/MMDVM hotspots bridging to WiRES-X rooms. Here’s what happens.

A user has a hotspot and accesses it using a Fusion HT. They use the WiRES-X control mode to change the hotspot to different YSF Reflectors. Let’s say the user wants to access MNWis (Room #21,493):

- They enter the number of the YSF Reflector, “US MNWis 21493”, which is 37,624. The hotspot switches to US MNWis 21493 which is bridged to WiRES-X Room # 21,493.
- The user leaves their radio in the WiRES-X control mode. They hear stations, but when they talk, nobody answers and nobody is talking.

Here’s what happened: When they keyed up to transmit, their radio sent a command to switch to Room #37,624. The Wires-X node that was previously connected to #21,493 now connects to #37,624 which belongs to a very nice gentleman in France.

How does this happen?

The Fusion transmitted data includes callsign fields CSD1, CSD2, and others. We’re interested in CSD1. This 10-byte field contains two items: The room number of the connection and; The TxID or DP-ID of the radio. A typical CSD1 looks like 21493F0yxh where 21493 is the room number and F0yxh is the DP-ID. The

‘21493’ in this field will cause a Wires-X node to switch to that room.

When the radio IS NOT in Wires-X control mode, the CSD1 field will be: ‘*****F0yxh’ and no switch will be made.

Bottom line:

DON’T USE WiRES-X control mode to talk through a hotspot or MMDVM!

Workarounds and fixes: (Updated 23-Aug-2019)

Node operators who are experiencing this problem should use the WiRES-X block feature to prevent the node from switching to the room that has the YSF reflector number.

For example: Assume that our node connects to MNWis, Room # 21,493. Our node also provides bridging to “YSF MNWis 21493” which is # 37,624.

- In the View->Node-Info(N) window press “Add”.
- In the “Input ID” dialog box enter the YSF reflector number. In our example this is “37624”.
- Press “OK” then “Close”.
- When a station using the YSF bridge still has WiRES-X control mode enabled accesses the node, the command to switch rooms will be rejected by the WiRES-X software. The software will indicate a rejected attempt to switch rooms.

As an alternative, set the node so that connection changes are not allowed.

- File->Settings->Call Settings->Uncheck “Round Room Connection”.
- Un check “Accept calls while in Round Room QSO.
- Check “Return to Room”

    - Fill in the WiRES-X room number. Example: 21493.

YSF server side solutions:

I maintain a version of YSF Reflector that has filters to prevent hotspots causing problems with WiRES-X nodes. My first step is to drop all packets that contain an incorrect WiREX-X room number. Eventually I plan to replace the first 5 bytes of CSD1 with '*****'. This last step is difficult to do because it requires the reflector to decode the data, modify it, then recompute the CRC. This also involves working with the interleaving and forward error correction. It is obviously easier to just drop the packets, but that may confuse people on the YSF side since nobody on the WiRES-X side will hear them.

If you are interested in running the enhanced YSF Reflector software that will fix this problem at some point, please contact me privately.

73,

Chris, K9EQ

WiRES-X Automation

Category: Automation,Projects,WiRES-X
October 22, 2022
Yaesu does not provide a mechanism that allows the WiRES-X software to be controlled by another program, i.e., having another program switch to a certain Room when a net starts.

Windows does, however, permit another program to send events to a program. Each window, menu item, and dialog in a Windows program has a unique identifier. It is possible to use these identifiers to send “message” to the WiRES-X software.

The WiRES-X Automation Project’s purpose is to bring together people who are interested in developing this technology and sharing their results.

To get things going, here are two mechanisms for automating WiRES-X:

1. AutoIT: http://www.autoitscript.com
2. Python – an excellent programming language found at python.org

Dave, N9TOW, provided the following information:

```
Packages I have installed on my WiresX system.
C:\Users\WiresX>pip list
comtypes (1.1.3)
pip (9.0.1)
pypiwin32 (220)
pywinauto (0.6.3)
setuptools (28.8.0)
six (1.10.0)
https://github.com/pywinauto/pywinauto
```

To install

pip install -U pywinauto

Script that executes changing channels on WiresX app

```
import time
from pywinauto import Application
  app = Application().connect(path=”C:\Program Files
(x86)\YAESUMUSEN\WIRES-X\wires-X.exe”)
app.WiresX.menu_select(“Connect(C)->Connect To(T)”)
time.sleep(1.5)
app.InputID.Edit.set_edit_text(“21493”)
time.sleep(.5)
app.InputID.OK.click()
time.sleep(4)
dialogs = app.windows()
##app.Dialog.CloseButton.click()
```

Python Program

Bill, W9LBR, developed a Python program that runs on the WiRES-X computer. The software has been updated as of 12-Feb-2024. See more here.

This is from his description:

WXscheduler.pyw is a Python 3 program that automates certain aspects of
Wires-X Room/Node connections according to a userrdefined schedule.
To that end WXscheduler.pyw must be run on the same Windows PC
that hosts the Wires-X application controlling either an HRI-200 or
a USB connected radio that supports PDN operation.

Schedulable operations:
– Connect
– Disconnect
– SetUnlimitedTOT
– SetTimeoutTOT
– Restart Wires-X App

Manual operation: Button that causes a Force Disconnect

Bonus operation: Displays Last Heard information that the Wires-X app
updates once a minute. Converting Lat/Lon into Grid Square if available.

A zip file containing instructions and the Python program is available at WXScheduler.

Update History

2 June 2023 v1.3 updates: – Last Heard now displays FT5-D, FTM-200, and FTM-500 correctly – Desktop\Wires-X_Last_Heard.html now generated to be viewed with PC browser

12 Feb 2022 updates: – Corrected grid square rendering of GPS coordinates as displayed in the Last Heard frame – version (v1.2) now displayed in main window title

5 Feb 2022 minor updates: – Fixed Last Heard display of recently manufactured FT70-D radios – WXscheduler.txt is now in DOS format, making it easier to read with Notepad 2022 version of WXscheduler.pyw – A scheduled event now performs multiple actions – Added File->Settings->Call settings actions – Fixed issue when Wires-X App is minimized – Fixed issue when Wires-X Settings window left open