Ralph Gable, WA2PUX

658 French Glen, Oregon, WI 53575; **wa2pux@gmail.com**

Remote Rig Operation

A creative approach to remote transceiver control.

The 3CX Phone System Management Console showing the settings for the shack (GrandsStream HT286) extension.

My ham radio career started in 1970. My shack was in the corner of the basement. Ever since, my shack has always ended up in the basement if we had a basement to put it in.

We moved into our present home in 2002. As always, my shack went into the basement for lack of anywhere else to put it. Granted, I ended up with a very nicely done corner of the basement, but it was a basement, after all. I wanted to get out of the basement without having to actually move the shack; there would be too many large holes to drill.

It occurred to me, "Why not operate my rig remotely?" My Kenwood TS-2000 had that capability. It talked to an RS-232 port. One small hole between the shack and the bedroom to fish an RS-232 cable and I'd be set -- except I still didn't have audio. It is all well and good to be able to operate the radio remotely, but does little good if I cannot hear it! Okay, so I run speaker and microphone cables, too.

I didn't want to run these analog signals any distance in an RF environment. I have a wired and wireless network in my home. So, why couldn't I use the existing Ethernet network to transmit control and audio?

Controlling the Rig Remotely

The control side was quite easy. Lantronix makes RS-232 to Ethernet adapters. These come with software to make your computer "think" it is talking to a COM port (virtual COM port, RS-232). So, the software that controls the rig sees a COM port and is happy. The control signals happily zip back and forth over the Ethernet and the rig sees an RS-232 port talking to it. We're half way there.

Audio to and from the Rig, the Biggest Challenge

I had the same problem, though… no audio. My original plan was to build a full duplex audio interface that talks RS-232 and utilizes the second port of my Lantronix UDS2100 RS-232-to-Ethernet adapter. This would require software expertise on the computer end to talk RS-232 to send and receive audio from the sound card to another virtual COM port. I didn't have that expertise. The project stalled….at least until I discovered *Voice Over IP* (VOIP) technology.

This technology has been around a long time. Magic Jack and Vonage are two common carriers who use VOIP to provide telephone service. But, how do I use this technology in my application?

Getting Audio to and from the Ethernet

GrandStream (among others, I'm sure) have Analog Telephone Adapters (ATA) that interface between standard VOIP network and any analog telephone. I purchased a GrandStream HT 286 for under $30 off the Internet. This requires that I have a VOIP network for it to talk to. How do I get a VOIP network in my house? This was *way* easier than I thought.

Setting Up a VOIP Network

3CX has free, downloadable software to do the job (**www.3cx.com/phone-system/index.html**). One application is the VOIP system that resides on a computer on your network.

This was a little daunting to me as I was sailing into unknown waters. I had help from Matthew Orr of InfoSys Consulting, Inc. (**www.InfoSysHelp.com**) in getting this set up. My personal thanks to him; he was a great resource. Later in this article I'll describe how to go about configuring the GrandStream HT286 with the VOIP system.

Turning Your Computer Into a Phone

The second needed software that you can download free from 3CX is the *Softphone* (**www.3cx.com/VOIP/voip-phone.html**). It is an application that sets up a VOIP telephone on your computer. You use the *SoftPhone* to call the GrandStream ATA down in the shack. With this *SoftPhone* loaded on every computer in the house, you can make calls between computers (each computer becomes an extension with video phone capabilities) and you can connect to your rig's audio from any computer! I'll describe later how to set up *SoftPhone*.

Interfacing the ATA with the Rig's Audio + More Control

We're mostly there, but not quite. I have the audio in the shack now in the form of an analog telephone signal. How do I get it into and out of the rig? Most of you are probably ahead of me on this. If you are thinking "phone patch," you are 100% correct. But, this phone patch has to have certain functionality to make it convenient for this application. As a minimum it needs to:

• Answer the phone and connect to the rig automatically … like an answering machine does after a set number of rings

• Allow you to hang up the extension remotely

For even more convenience, I'd like it to allow me to turn on the power to the rig's power supply and connect the antennas to the rig.

If you are thinking "ring detector and counter" to take care of the first requirement, you got that right. If you are thinking "touch tone decoder" for the second requirement and the added extra, you got that right, too.

So, that is what I did. I built a phone patch with a ring detector and counter so that it would automatically answer the phone after four rings. I added a touch tone/DTMF decoder to provide control. The DTMF decoder I chose gives me the potential of controlling up to 16 devices with a single key press for each. Of course, one of those is taken up with the hang up functionality and another for the power up functionality. You are still left with 14 more things you can do remotely. These lie fallow in my design, for the moment.

The downside of the DTMF decoder is that it doesn't work when there is any amount of audio on the telephone line. So, it has to be quiet on the line to use this. Switching antennas with the din of a pile-up going on just won't happen. Mute rig, switch antennas, un-mute rig.

One More Degree of Freedom

Now, to give myself yet one more degree of freedom, I purchased a Bluetooth hands-free headset and a USB Bluetooth interface. Now I do not need a wired microphone. The *SoftPhone* can be configured to use the PC speakers for its speakers and the Bluetooth headset for its microphone. The whole system looks something like the diagram shown in Figure 1.

The Phone Patch Details

Back in the early 1970s I built a rudimentary phone patch. Believe it or not, I kept the guts of the patch all these years, stuffed away on a box full of Styrofoam noodles. When I started looking for more technical information for my new patch, I was amazed at who didn't have it. I did find what I was looking for, but it wasn't where I expected it to be. In the end, it wasn't rocket science and I didn't expect it to be. After all, I was just 16 when I built my first one, but it didn't have the needed bells and whistles that this one needed to have.

Let's walk through the schematic in Figure 2. The basic phone patch portion of the schematic is shaded in gray. The relay contacts that are connected across C15 may be replaced with a SPST switch. The purpose for the diodes included in the design across the speaker and microphone is to prevent large voltage swings in the event of the ring signal. They are there for protection of the transceiver. Please note that the pin numbers that appear on the transformer *may* not be the pin numbers that apply to the transformer you choose to use. The transformer is a 600 Ω primary connected to the telephone line. The speaker winding is a 600 Ω secondary. The microphone winding is a 150 Ω secondary. The primary/secondary that is used for the ring detector is a 150 Ω winding.

The Ring Detector

The ring signal is an ac signal. There are a number of ways I could have done this, but

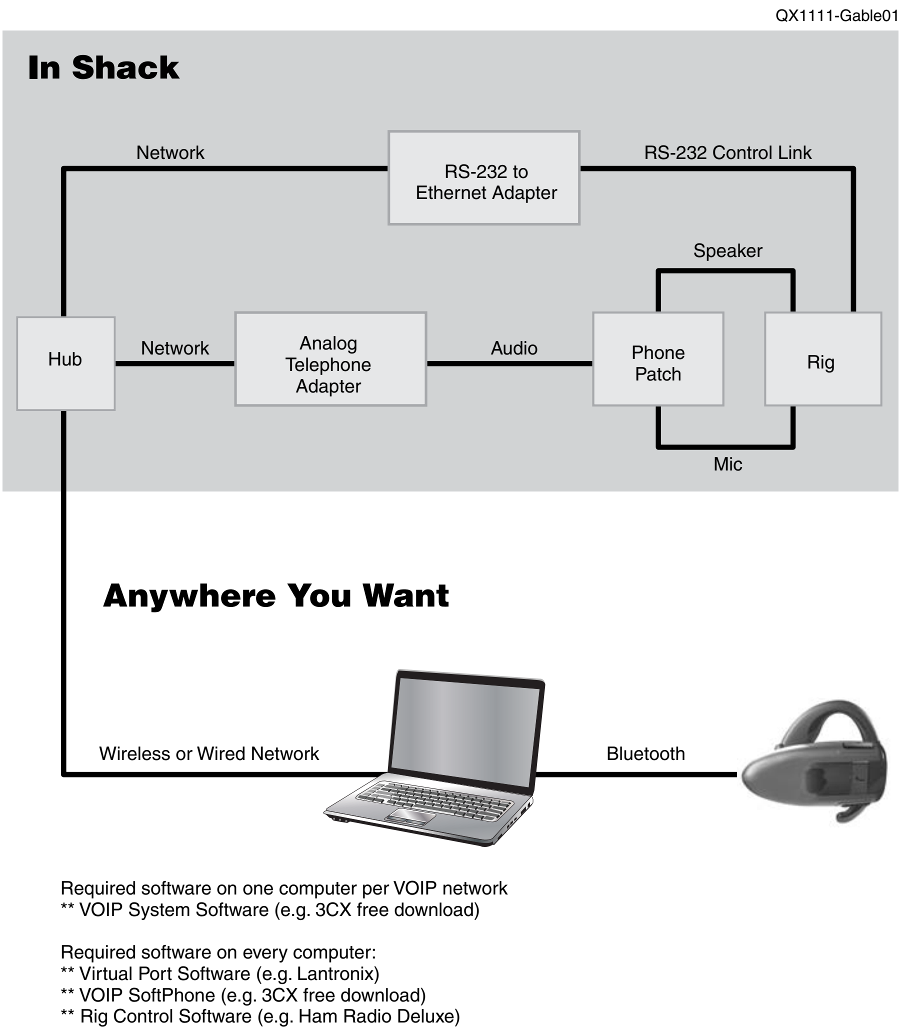


Figure 1 – System diagram.

QX1111-Gable02

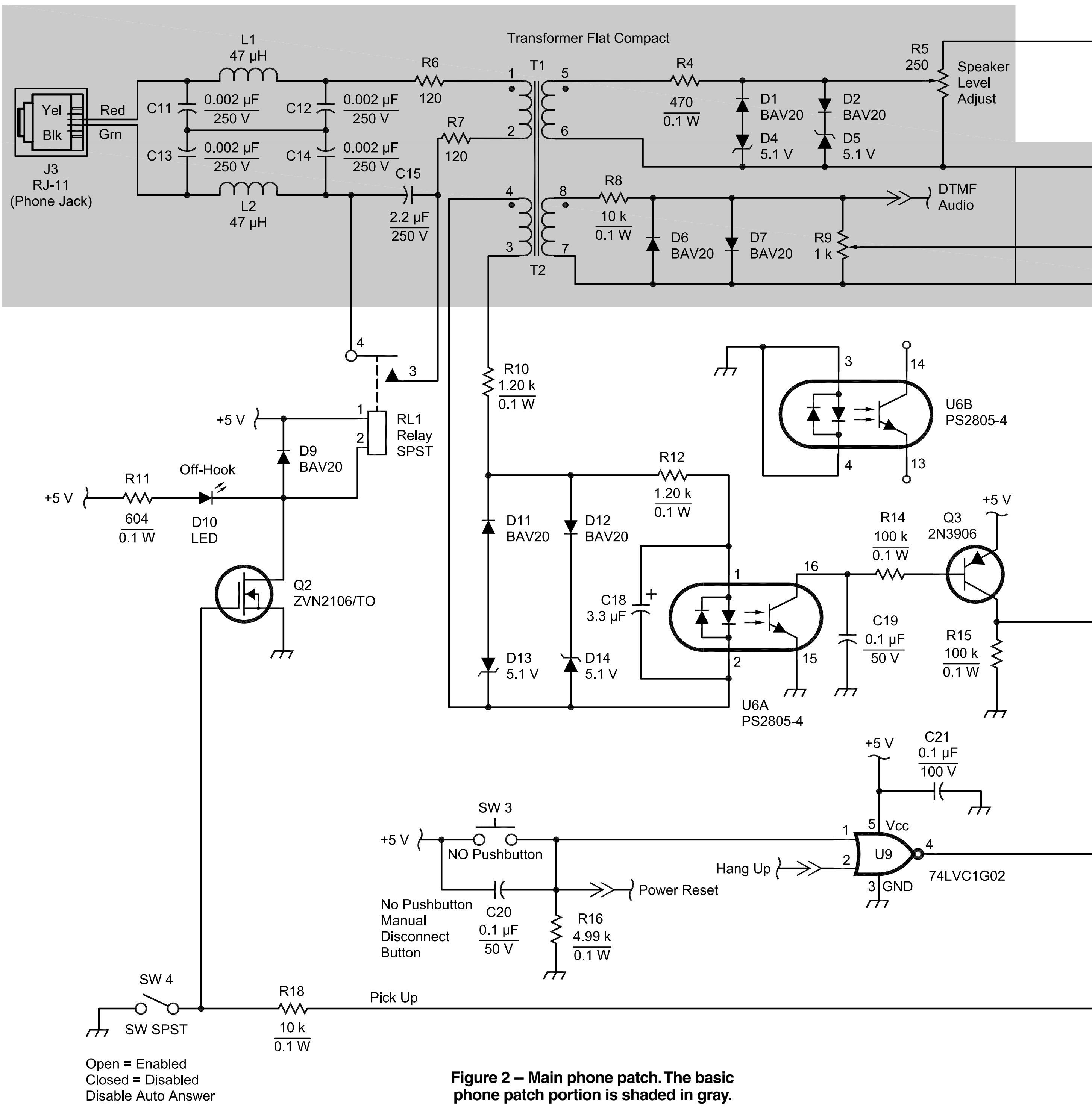
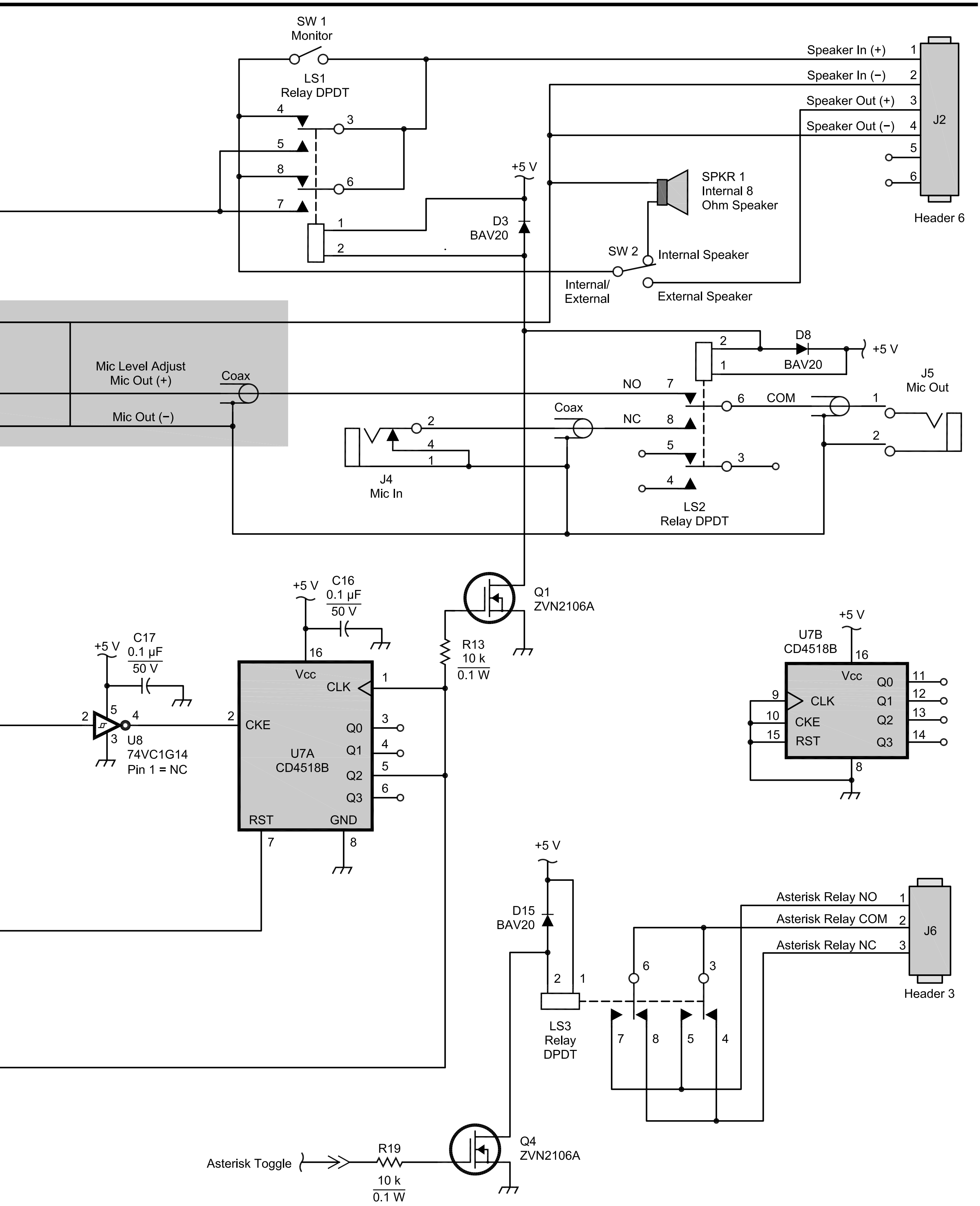


Figure 2 -- Main phone patch. The basic phone patch portion is shaded in gray.


SW 1
Monitor
LS1
Relay DPDT
Speaker In (+)
Speaker In (−)
Speaker Out (+)
Speaker Out (−)
J2
Header 6
+5 V
SPKR 1
Internal 8
Ohm Speaker
D3
BAV20
SW 2
Internal Speaker
Internal/
External
External Speaker
D8
BAV20
J5
Mic Out
Mic Level Adjust
Mic Out (+)
Coax
Mic Out (−)
NO
NC
COM
J4
Mic In
LS2
Relay DPDT
Q1
ZVN2106A
C16
0.1 µF
50 V
C17
0.1 µF
50 V
R13
10 k
0.1 W
U8
74VC1G14
Pin 1 = NC
Vcc
CLK
CKE
Q0
Q1
Q2
Q3
U7A
CD4518B
RST
GND
U7B
CD4518B
D15
BAV20
LS3
Relay
DPDT
Asterisk Relay NO
Asterisk Relay COM
Asterisk Relay NC
J6
Header 3
Asterisk Toggle
R19
10 k
0.1 W
Q4
ZVN2106A

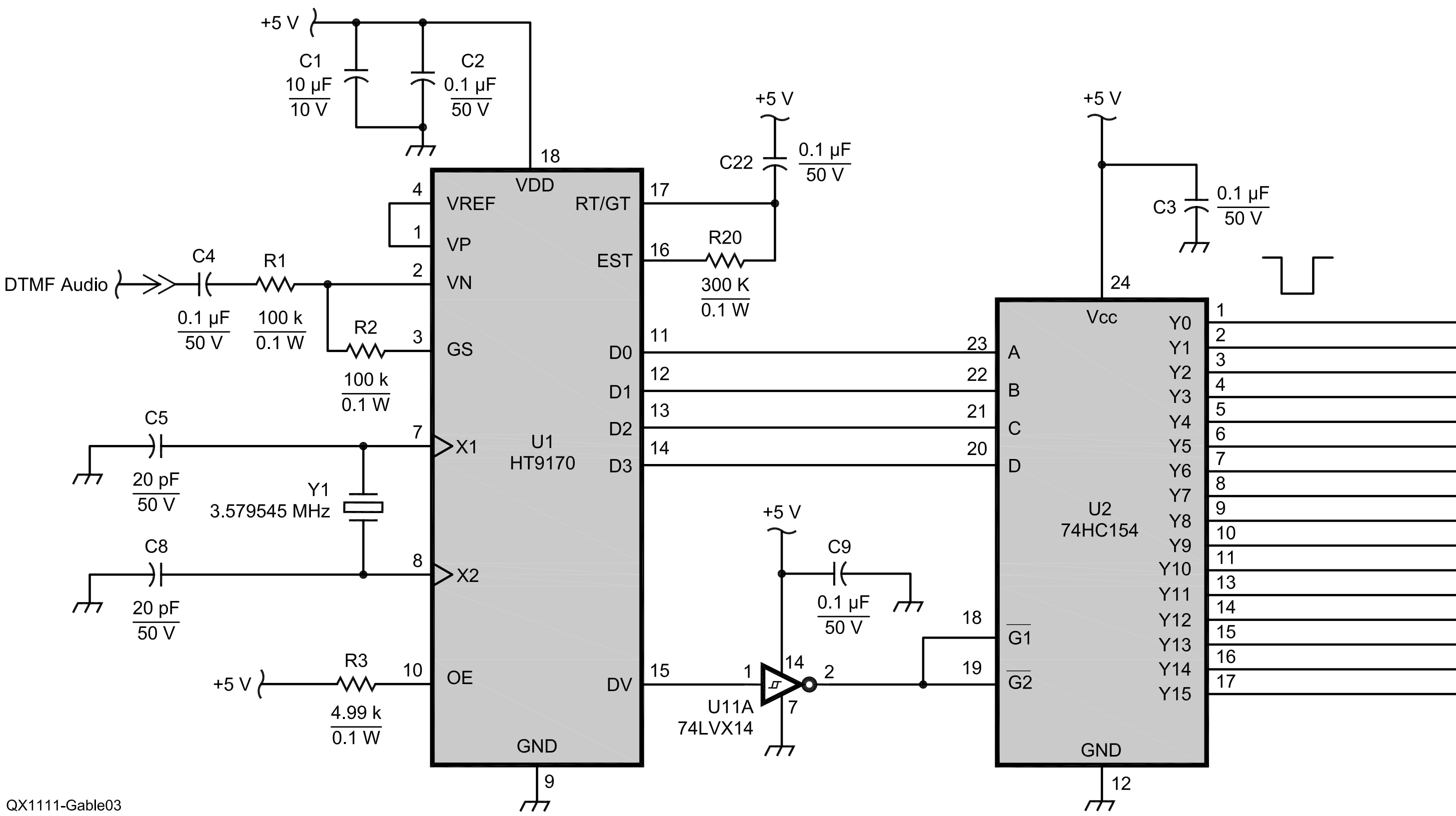
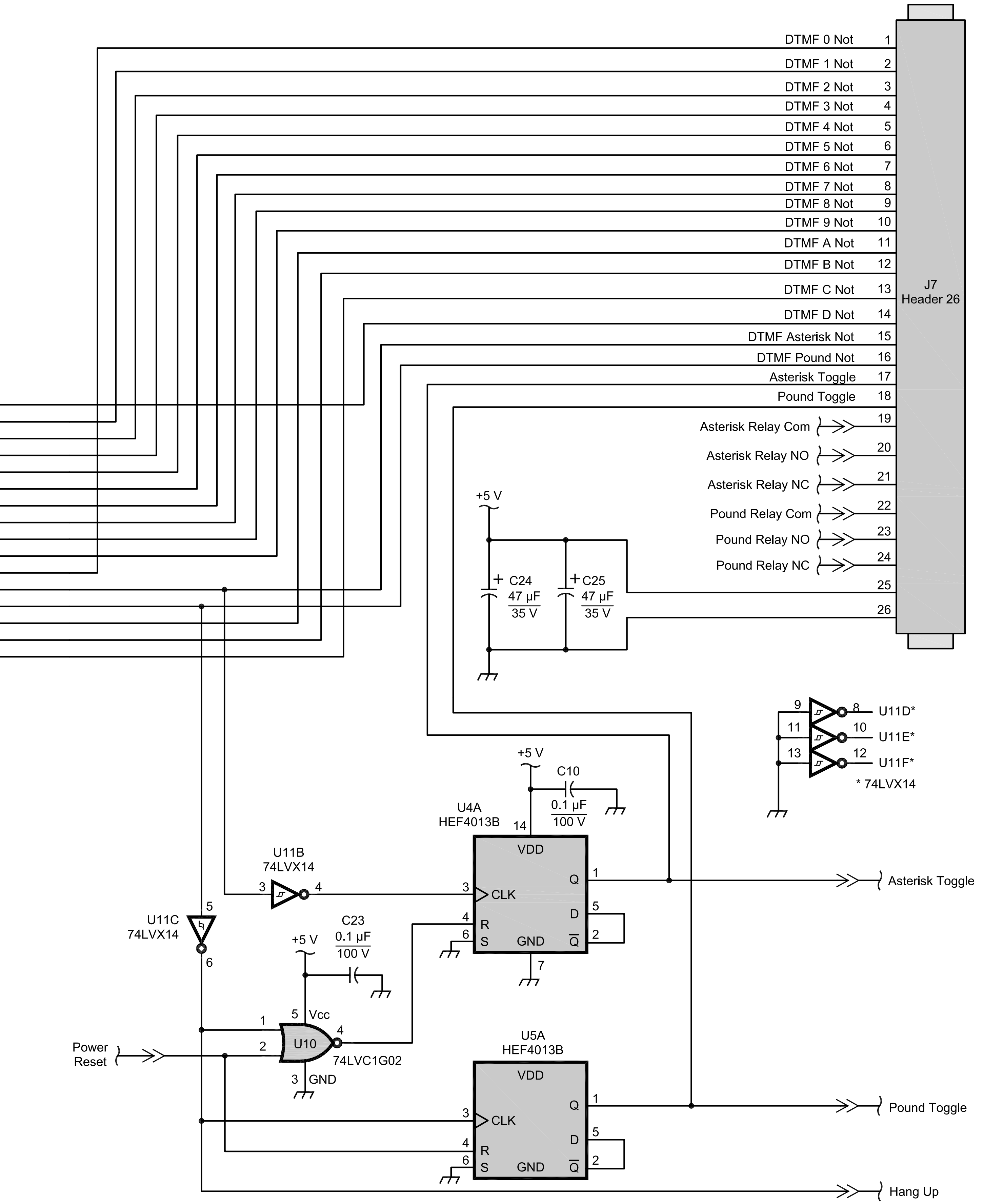


Figure 3 — DTMF Decoder and Control.


DTMF 0 Not
DTMF 1 Not
DTMF 2 Not
DTMF 3 Not
DTMF 4 Not
DTMF 5 Not
DTMF 6 Not
DTMF 7 Not
DTMF 8 Not
DTMF 9 Not
DTMF A Not
DTMF B Not
DTMF C Not
DTMF D Not
DTMF Asterisk Not
DTMF Pound Not
Asterisk Toggle
Pound Toggle
Asterisk Relay Com
Asterisk Relay NO
Asterisk Relay NC
Pound Relay Com
Pound Relay NO
Pound Relay NC
J7
Header 26
+5 V
C24
47 μF
35 V
C25
47 μF
35 V
U11D*
U11E*
U11F*
* 74LVX14
C10
0.1 μF
100 V
U4A
HEF4013B
VDD
CLK
R
S
GND
Q
D
U11B
74LVX14
U11C
74LVX14
C23
0.1 μF
100 V
Vcc
U10
74LVC1G02
GND
Power
Reset
U5A
HEF4013B
Asterisk Toggle
Pound Toggle
Hang Up

I settled on an ac input opto-isolator. The ac ring signal coming from the transformer goes through a current limiting resistor to a diode circuit (clipper) designed to limit the voltage to be applied to the opto-isolator circuit to no more than the zener voltage plus the forward voltage drop of the other diode that is in series. This is still an ac signal.

The next step is a current limiting resistor to the input of the opto-isolator (PS2805-4). The selection of the limiting resistor is based on the forward drop of the input of the opto-isolator and the current you want to push through it. I added a 3.3 μF capacitor to turn this same circuit into a low pass filter. This way stray clicks and the like that may appear on the telephone line do not get counted as a bona fide ring signal.

The output side of the opto-isolator is an open collector with a following common emitter amplifier (2N3906). The object is to make the rise and fall time of the incoming signal as quick as we can. Digital gates are not real happy with slow slew rate signals. To finish the processing, I stuck a Schmitt trigger inverter between the opto-isolator output and the counter input; this finishes squaring the edges up. Now we have a nice, truly digital signal to apply to the counter.

To follow the signal a little bit just for clarity of understanding…

- A ring signal occurs
- The input side of the opto-isolator conducts
- The output side of the opto-isolator conducts and its collector goes low
- This pulls the base of the 2N3906 low causing it to conduct, connecting the collector to Vcc ($V_{COLLECTOR} = Vcc - \approx 0.2\ V = \approx 4.8\ V$). The collector goes high.
- When the ring signal stops, the collector of the 2N3906 drops low again.

The Ring Counter

Admittedly, the counter circuit *does* look a bit strange. Thinking this through, I wanted the counter to stop counting after it reached a count of four. That means that bit Q2 would go high. I had to come up with a way to do this. The CD4518 has a Clock Enable input. If the Clock Enable is high, then positive transitions on the clock cause it to increment. On the other hand, if the clock input is low, then negative transitions on the Enable input cause it to increment. If the clock is held high, then transitions on the Enable do nothing. That is *just* what the doctor ordered.

I tied the counter's Q2 (bit 2 of 0 through 3) output to the CLK input and the output of the opto circuit to the Enable input of the counter. Now the counter increments a count at the cessation of each telephone ring. Once it reaches a count of four, then it quits counting. The only thing that will allow it to count again is a reset.

The counter is reset either by the PWR_RESET signal or the "#" output (HANG_UP) of the DTMF decoder. To reset the counter is to hang up the phone.

The output of the counter goes high when the phone patch is supposed to answer the phone. This drives three relay driver inputs. The first (RL1) connects the phone patch to the phone line, answering the phone. The second (LS1) connects the rig's speaker output to the audio input of the patch. The third (LS2) connects the audio output of the patch to the microphone input of the rig.

Power Up Reset

The power up reset is nothing more than an RC network whose output goes high at power up and then drifts low as the capacitor charges. The pushbutton across the capacitor is the reset button for manual reset. It pulls the PWR_RESET signal high and discharges the capacitor. Once released, the PWR_RESET signal drifts low again just as if the power were just turned on.

DTMF Decoder

The DTMF, or touch-tone, decoder is an off the shelf DTMF receiver, an HT9170 which is available through Newark Electronics. See Figure 3. There is nothing special about this circuit. I took the circuit right out of Application 1 on page 8 of the datasheet.

One of the outputs of the DTMF receiver is the DV output which goes high when a valid DTMF tone pair is detected. More on this later.

The HT9170 presents the decoded DTMF tones as four bit binary numbers. To make this truly useful, this has to be further decoded. I chose a 4:16 decoder/demultiplexer, the 74HC154. It is a simple matter of connecting the four bits coming out of the DTMF receiver to the four bits of the 4:16 decoder. The decoder also has enable inputs. The output of this decoder will not be asserted unless these enable inputs are both low. So, we invert the DV signal from the DTMF receiver to give us an active low for these inputs. This prevents the output of the decoder from being asserted unless the DV signal is asserted (valid DTMF tone pair received by the receiver).

Table 1
DTMF Controls

Output	*DTMF*	*Output*	*DTMF*
Y0	D	Y8	8
Y1	1	Y9	9
Y2	2	Y10	0
Y3	3	Y11	*
Y4	4	Y12	#
Y5	5	Y13	A
Y6	6	Y14	B
Y7	7	Y15	C

We now have 16 fully decoded controls to be used. They are defined in Table 1.

Control Circuitry

The output of the 74HC154 is negatively asserted. That means that the output goes low when it is true. I chose to use the "#" to remotely hang up the phone patch and the "*" to toggle auxiliary relay contacts on and off, which I plan on using to control the ac power to the power supply that powers the rig.

The first step of this control is to invert the output of the decoder that I have chosen to use. The "DTMF_Pound_not" becomes "Hang_Up," which is positively asserted when the user wants to disconnect the phone patch from the ATA by pressing the "#" key on their *Softphone*.

The "DTMF_Asterisk_not" becomes the positively asserted "DTMF_Asterisk." This is used to drive the clock input of a D Flip Flop. The flip-flop has its Q_not output connected to its D input so that the Q output toggles at each clock edge it sees.

To make sure that it comes up in the right state, the PWR_Reset signal is used to reset it to the Q = low state at power up. Furthermore, we want it to reset whenever the phone patch is not connected to the ATA. So, I use an OR gate with the Pwr_Reset signal on one input and the "Hang_Up" signal on the other. The OR gate's output becomes the "Asterisk_Reset."

The Q output of the flip-flop serves as the input of the relay driver. When Q is high, then the relay is engaged. When it is low, it is disengaged.

All of the other outputs from the 4:16 decoder just sit there waiting to feel useful. And, someday, I may just help them to feel that way. For the time being, they will just have to wait.

Station Control

I chose to use the Asterisk relay contacts to control a relay that turns on the primary power to the rig's power supply. The output of the rig's power supply (12 Vdc) supplies power to my Kenwood TS-2000. It also activates the antenna relays. These relays connect the antennas to ground when not activated and to the rig when activated. These relays are 10 kW RF relays I bought from Vector Solutions (**www.arraysolutions.com/Products/rf_relays.htm**). I mounted them in an EMI shielded aluminum box from Bud (AN-1322). To be a little more cautious, I also added additional lightning arrestors to the antenna lead in cables.

How to Set Up the VOIP System and It's Peripherals

The first step is making your PC a "static IP" box. To do this, follow the following steps for *Windows XP*:

• Open a "Command Prompt" window.
• Type "ipconfig /all" and press enter.
• Note the following entries:

•IP Address:__________.__________.__

____

•SubnetMask:__________.__________

____

• Default Gateway: __________._______

____

• DNS Servers:

__________.__________.__________.

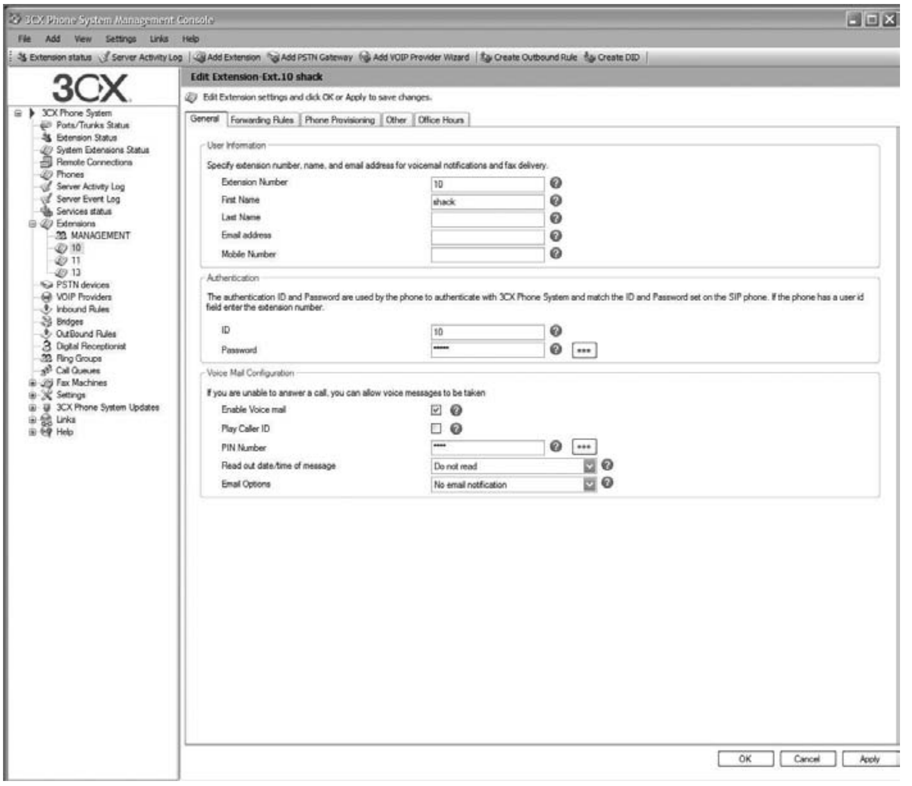
__________.__________.__________.

• Click "Start" and then "Control Panel"
• Double click on "Network Connections"
• Right click on the "Local Area Connection" icon that represents your LAN connection and then click on the "Properties" pop-up menu item.
• Click on the "Internet Protocol (TCP/IP)" entry in the "This connection uses the following items:" area and then click on the [Properties] button.
•The "Internet Protocol (TCP/IP) Properties" dialogue box opens. Click on the "General" tab. You will most likely see the "Obtain an IP address automatically" radio button selected.
• Select the "Use the following IP Address" radio button.

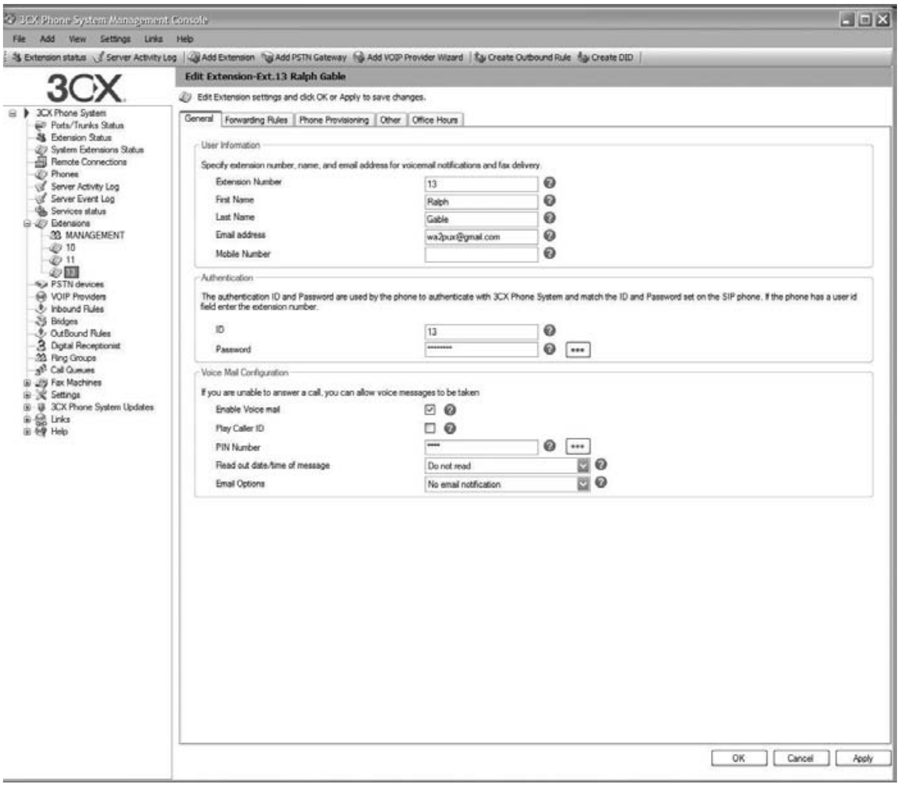
In the IP address, enter the IP address that you got from the Command Prompt window *except* the last number. Choose a new number for this that is at least 20 higher than the number you got from the Command Prompt window. This makes sure that other computers/devices that connect to your home network that "obtain an IP address automatically" will not be assigned the IP address of this computer if they do so when your computer is off at the time they connect. Otherwise, you will get an IP address collision on your network and this is not pretty.

• Click on [OK] and [OK] and restart your computer.
• Install the 3CX PBX Phone System from **www.3cx.com/phone-system/download-phone-system.html**
• When asked for the number of digits in the extension, I put two.
• They will eventually ask you for adding extension information. I used the following:

Extension Number: 10
First Name: shack
Last Name: ---left blank---
Email address: ---left blank---
Mobile Number: ---left blank---
ID: 10
Password: 10



The 3CX Phone System Management Console showing the settings for the computer (3CX Phone) extension.



The author's remote control system package.

Add another extension for the 3CX phone for the computer. I used the following:

Extension Number: 13
First Name: Ralph
Last Name: Gable
Email address: ---left blank---
Mobile Number: ---left blank---
ID: 13
Password: 13

• Complete the installation of the 3CX PBX Phone System.

• Install the 3CX Phone for the computer that you downloaded from: **www.3cx.com/downloads/3CXPhone6.msi**

• Create a new account on the 3CX *SoftPhone*.

Account Name: ---anything you want to call it----
Caller ID: 13
Extension: 13
ID: 13
Password: 13
My Location:
I am in the office
Local IP _IP address chosen above of PBX

• Click on [OK] and then [OK] again.

Not connected….reading configuration…on hook

• Now for the Grandstream HT286

• Locate and record the MAC address of the HT286 (usually on the label on the bottom)

• Connect the HT286 to an analog telephone.

• Connect the HT286 to power with its adapter.

• The following steps come directly from an application note found on the 3CX site (**www.3cx.com/sip-phones/GrandStream-HandyTone286.html).** "Configuring GrandStream HandyTone 486(487),286(287), ATA for 3CX Phone System"

• Reset it to factory defaults.

• Pick up the headset and press "****". This will start up a Voice Prompt Menu. Now press "99". Dial the MAC of the device, where …

a. A=22
b. B=222
c. C=2222
d. D=33
e. E=333
f. F=3333

For example, if the MAC address is 000b8200e395, it should be encoded as "0002228200333395"

• Setup the HT286 to work with the 3CX PBX system

• Connect the LAN to the Wan port of the HandyTone

• Connect an analog phone to the HandyTone phone port. Pick up the headset and press "****". This will start up a Voice Prompt Menu. Now press "02" to listen to the IP address that was assigned to the device by the DHCP server

• Enable the wan side Web access, dial "****" and then "129"

• Launch a browser and go to the IP Address determined or configured earlier. The default username is "admin"

• Select the "Advanced Settings 1" tab.

• Set the "SIP Server" field to the IP Address or FQDN of the server on which 3CX Phone System is installed – in this example 10.172.0.2

• Set the "Outbound Proxy" field to the same value as in step 13 above.

• The "SIP User ID" field should match the "Extension Number" field of the extension created for this phone in the 3CX Phone System Management Console

• In the "Authenticate ID" and "Authenticate Password" fields enter the ID and Password that you entered for the extension in the 3CX Phone System Management Console. These fields must match the Authentication ID and Password set for that extension in the 3CX Phone System Management Console

• The "Name" field is optional. A suitable value would be the name of the user using this phone

• Set the "Preferred Vocoder" to choice 1: PCMU, choice 2: PCMA. The settings for choices 3 to 7 will not come in use since they will not be used by 3CX Phone System

• Set "User ID is phone number" to "Yes"

• Set "Sip Registration" to "Yes"

• Set "Unregister on Reboot" to "Yes"

• Set the "Register Expiration" field to a suitable value. For testing purposes you may want to use 60 seconds, but once configuration is tested a larger value would be more appropriate to limit unnecessary network traffic. A good setting for general use could be 3600 seconds (1 hour).

• Set "Allow outgoing call without Registration" to yes.

• Set "Enable Call Features" to "Yes". This setting enables features like transfer, on hold, etc from the analog phone connected to the FXS port of the device.

• Set "Send DTMF" to "in audio" or "Via RTP (RFC 2833)" or both. SIP info is not recommended although 3CX Phone System can support it.

• Scroll to the bottom of the page and click the "Update" button. You will be prompted to reboot the device. Click the "Reboot" button.

• After the HandyTone has restarted, switch to the 3CX Phone System Management Console, and click on the Phone System -> "Line Status" (the default page). In the section "Extensions", your new extension connected to the PBX should be listed with a green status light.

At this point you should be good to go.

Ralph Gable is an ARRL member originally licensed in 1970 as WN2PUX, a Novice class operator. The following year he upgraded to General class and received the WA2PUX call sign. Ralph has been active to varying degrees over the years and has lived in a wide variety of locations. He received his Canadian reciprocal license (VE4PUX) when he lived in Canada for a number of years. During that time, Ralph was published in The Canadian Amateur *magazine and was a volunteer examiner for the Canadian version of the FCC. He upgraded to an Amateur Extra license since moving back to the US. Ralph is presently working as a Senior Electrical Engineer for TomoTherapy, a wholly owned subsidiary of Accuray Incorporated (newly acquired), in Madison, WI. TomoTherapy designs and manufactures state of the art Radiation Oncology machines.*