In this issue

Production Team

Ian Pawson G8IQU Trevor Brown G8CJS
Terry Mowles VK5TM

Contributing Authors

Richard Carden VK4XRL John Hudson G3RFL
Ken Konechy W6HHC Mike Stevens G7GTN
Grant Taylor VE3XTV
Dr Andrew (Drew) Wollin VK4ZXI

Editorial

Welcome to CQ-DATV 41.

Let's start by wishing John G3RFL a belated happy 71st Birthday. John has been one of this magazines greatest supporters, right from issue 1 and is a major contributor. I know many of you have built his projects and followed the development and construction of his local ATV repeater, GB3FY. In this issue John has put together a simple sine wave generator, which has a multitude of uses beyond the obvious application, video response testing.

From a well established CQ-DATV contributor to a new one, Grant ZI1WTT1/VE3XTV. In this issue he starts with his article on repeater linking. Grant's reputation for ATV is well established and he is frequently to be found on the Digital ATV group@yahoo.com, but he is new to CQ-DATV, welcome aboard Grant. I cannot also move on without mentioning another major contributor, Ken W6HCC and his tireless effort to keep everyone up to speed on DATV express. Richard VK4XRL is also on our role of honour, he and I have been pooling our efforts this month in the true spirit of ATV to bring the ATV vision switcher to this issue.

CQ-DATV production makes a major demand on time and effort and would not get off the ground without a dedicated editorial team. Here at CQ-DATV our team work tirelessly to assemble debug and create all the different eBook formats and the more established PDF file. This file in particular would not be happening without Terry VK5TM, who creates it, using CQ-DATV copy, but by adding the required layout from scratch every month.

Can I also draw your attention to the CQ-DATV Facebook page, yes we have the advantage of our magazine appearing monthly, but in these fast moving times, even a monthly publication cannot appear frequently enough.

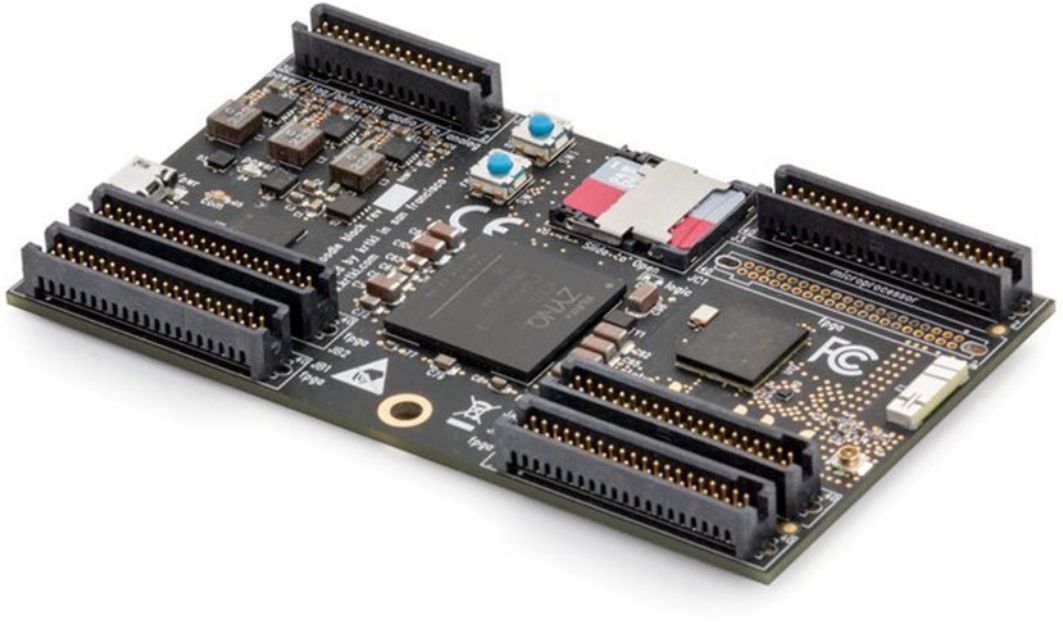
Facebook fills the gaps and puts faces to our readers, lets everyone see who you are what your interests are and where in the world you are located, so please sign it and make yourself known.

Talking of making yourself known this is my attempt at writing an editorial. Until now I have been merely contributing and I now know, that it is far easier to just write an article instead of trying to do the more complex publishing tasks. It's a long time since I first dare to submit an article and I needed all the assistance and support available and then some. Is there ever a chance to email an editor of a publication and suggest that you could write even part of the Editorial? In a commercial world this would never actually happen. We are however in an incredible hobby, (Amateur Television) and CQ-DATV is a leading amateur publication created by people with ideas to share and develop, sometimes via peer review.

I had to admit I was nervous, but if I can do it, then so can you and without your input this magazine will wither and die. Please, ATV is a brilliant hobby and to keep it alive we need your input. It need not be a complete article, or even a constructional project. It might just be what is happening in your part of the world, or your experience, a record of an ATV contact you have made, a news item our overloaded team may have missed. If it's ATV we want to hear about it and share it with our readers.

Please, if I can do it so can you.....
Guest Editorial by Mike G7GTN

snickerdoodle: *A palm-sized, reconfigurable Linux computer that connects to the real world: ARM + FPGA + Wi-Fi + Bluetooth + 180 I/O*



snickerdoodle is a tool for dreamers and creators to build, make, invent, and do things they've always been told weren't possible. It's for people willing to explore new horizons and challenge themselves to learn, grow, and handcraft great, new things - not because it's easy, but because it's worth doing.

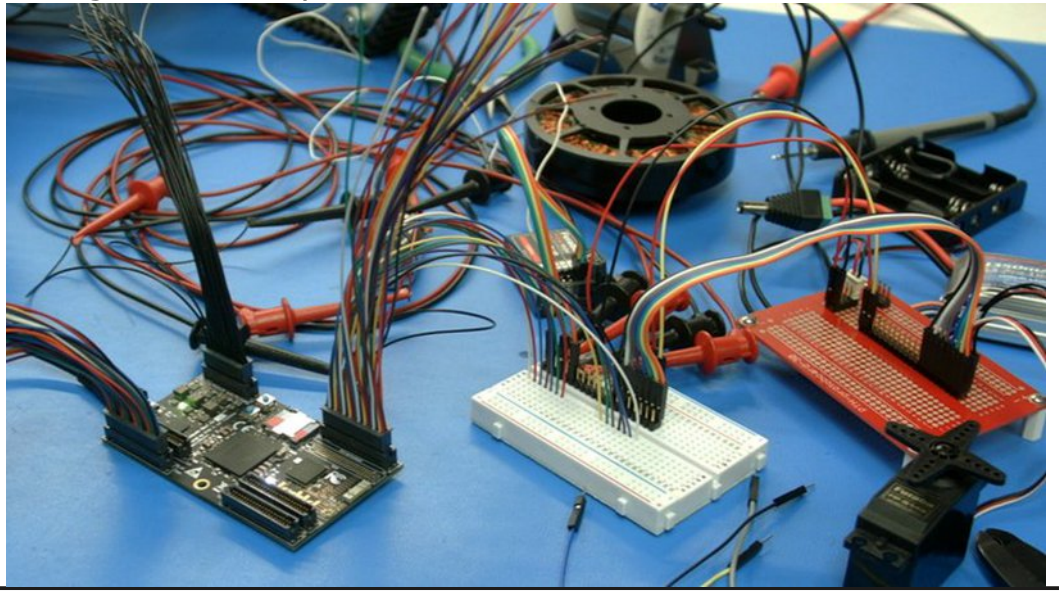
Why Do I Need snickerdoodle?

If you are satisfied with what existing low-cost platforms have to offer because they're "good enough," or you see no reason to dare to branch out and try something new, or you have never been left thinking "I love [fill in the blank maker / development board], but I really wish it could do [this]," then read no further. snickerdoodle isn't for you.

We love what Raspberry Pi and Arduino have done for education and the impact they've had on the growth of the maker community. We also realize that what fundamentally makes these platforms powerful is what limits them: they are intended to help you take the first step, to get you thinking, to get you writing those first lines of code, to foster confidence and hope and curiosity.

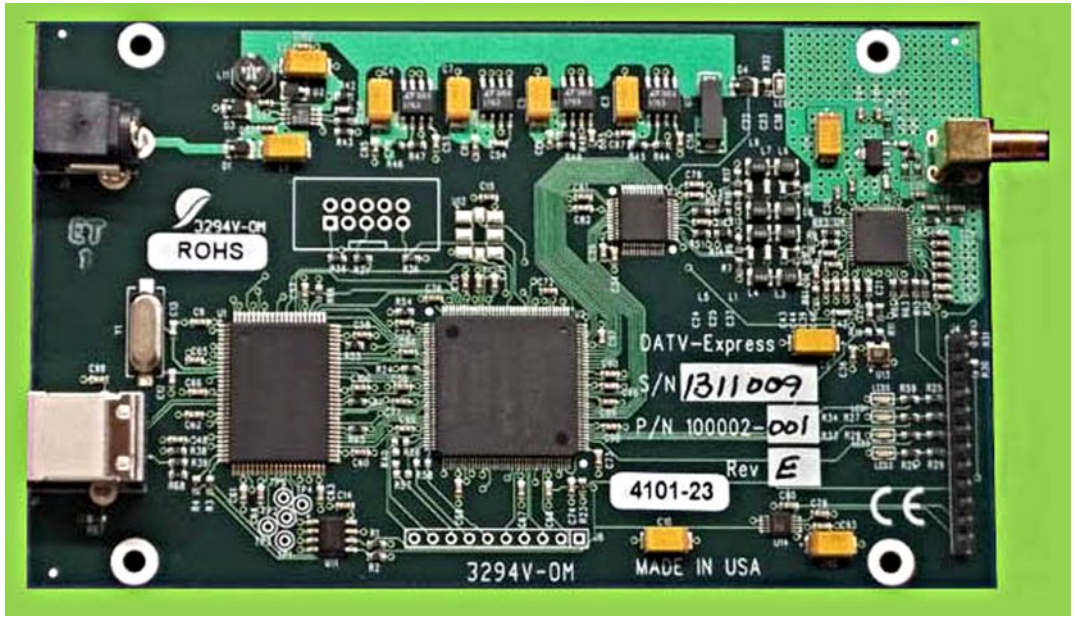
The question is: now what? What if I want to do more? To grow? To create something different? Why is there nothing out there that I can afford and use that lets me make what I want to make?

That is why you need snickerdoodle. To create something different. To make what you want to make. To take back control of your projects. To invent something you will be proud to show your friends, your family, your colleagues. Put another way: snickerdoodle takes all the things that Raspberry Pi, Arduino, and BeagleBone prevent you from doing and allows you to do them.



The DATV-Express Project Team asks: "Should the team build More Boards?"

Just a note to say that the Project Team inventory of hardware boards is now empty with the last board going off to Pierre HB9IAM in France. Currently, the PayPal coding has been removed from the DATV-Express.com website and board sales have been stopped.



Although sales have been slow for last several months (about four boards per month), the team is looking for a "show of hands" via e-mail to see if there are enough hams willing to agree to a standby order…if more boards were to be built.

No payments are required for a "show of hands" e-mail, just your commitment to buy a board or two if the project team builds another production-lot.

The pricing will remain the same…US$300 + shipping (from USA) via PayPal.

If you are willing to agree to a standby order, then send an e-mail to *Support@DATV-Express.com*. We currently have a list of four standby orders, and need a total of 16-to-20 orders or more to proceed with another board production-run.

Thank you for all the kind words that board-owners have sent to the Project Team on this product.

Finally, as Charles G4GUO reported in the September Project Update Report, he plans to turn his efforts to a Tx/Rx SDR board that is expected to become available from Lime Microsystems by the end of 2016.

G4GUO believes it will be possible to port the Express_Transmitter Windows code over to the LimeSDR board. But the porting of software will take time and perhaps be initially available in Q2 of 2017.

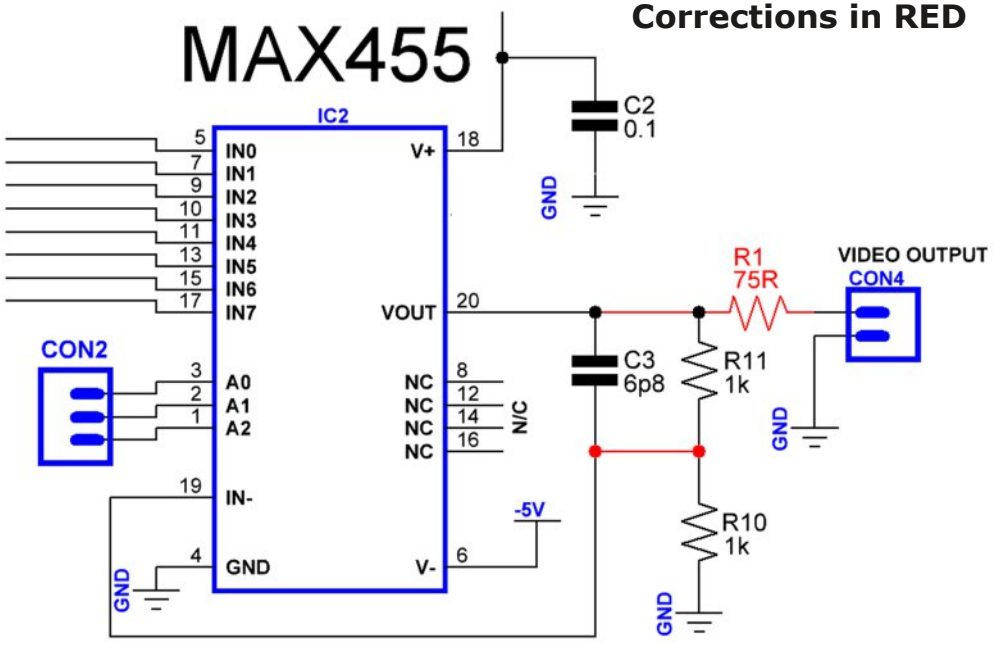
73…de Ken W6HHC

Correction to CQ-DATV 40 MAX455 Switcher

Mike G7GTN

Under the heading of no blunders allowed, I need to make a correction to the switcher design that was published last month. The problem concerns the power with the internal op-amp contained in the Maxim MAX455. As was originally shown on my circuit diagram no video output would ever emerge. Thank-fully Richard Carden VK4XRL was eagle eyed enough to alert me to this circuit diagram problem and then equally helpfully also provided a quick pen based correction for me to use. From this point I will make sure I draw and then only fully build from my own diagrams, when hot solder is flowing you sometimes forget to go back and check the created paperwork as thoroughly as I should have done in this case.

Corrections in RED



TV Amateur is a German Language ATV Magazine It is published 4 times a year and if you would like to subscribe go to http://www.agaf.de/

Record and stream live video!

What is OBS?

This project is a rewrite of what was formerly known as "Open Broadcaster Software", software originally designed for recording and streaming live video content, efficiently.

What's the goal of rewriting OBS?

- *Make it multiplatform. Use multiplatform libraries/functions/classes where possible to allow this. Multi-platform support was one of the primary reasons for the rewrite. This also means using a UI toolkit will be necessary for user interface. It also means allowing the use of OpenGL as well as Direct3D.*

- *Separate the application from the core, allowing custom application of the core if desired, and easier extending of the user interface.*

- *Simplify complex systems to not only make it easier to use, but easier to maintain.*

- *Write a better core API, and design the entire system to be modular.*

- *Now that we have much more experience, improve the overall design of all the subsystems/API, and minimize/eliminate design flaws. Make it so we can do all the things we've had trouble with before, such as custom outputs, multiple outputs at once, better handling of inputs, custom services.*

- *Make a better/cleaner code base, use better coding standards, use standard libraries where possible (not just STL and C standard library, but also things like ffmpeg as well), and improve maintainability of the project as a whole.*

- *Implement a new API-independent shader/effect system allowing better and easier shaders usage and customization without having to duplicate shader code.*

- *Better device support. Again, I didn't know what I was getting into when I originally started writing code for devices. It evolved into a totally convoluted mess. I would have improved the existing device plugin code, but it was just all so fundamentally bad and flawed that it would have been detrimental to progression to continue working on it rather than rewrite it.*

What was wrong with the original OBS?

The original OBS was rewritten not because it was bad, at least in terms of optimization. Optimization and graphics are things I love.

However, there were some serious problems with the code and design that were deep and fundamental, which prevented myself and other developers from being able to improve/extend the application or add new features very easily.

First, the design flaws:

- *The original OBS was completely and hopelessly hard-coded for windows, and only windows. It was just totally impossible to use it on other systems.*

- *All the sub-systems were written before I really knew what I was getting into. When I started the project, I didn't really fully comprehend the scope of what I would need or how to properly design the project. My design and plans for the application were just to write something that would "stream games and a webcam, with things like overlays and such." This turned out fine for most casual gamers and streamers (and very successful), but left anyone wanting to do anything more advanced left massively wanting.*

- *Subsystems and core functionalities intermingled in such a way that it was a nightmare to get proper custom functionality out of it. Things like QSV had to be meshed in with the main encoding loop, and it just made things a nightmare to deal with. Custom outputs were nigh impossible.*

- *The API was poorly designed because most of it came after I originally wrote the application, it was more of an afterthought, and plugin API would routinely break for plugin developers due to changing C++ interfaces (one of the reasons the core is now C).*

- *API was intermeshed with the main executable. The OBSApi DLL was nothing more than basically this mutant growth upon OBS.exe that allowed plugin developers to barely write plugins, but all the important API code was actually stored in the executable. Navigation was a total mess.*

- *The graphics subsystem, while not bad, was incomplete, and though far easier to use than bare D3D, wasn't ideal, and was hard-coded for D3D specifically.*

- *The devices and audio code was poor, I had no idea what I was getting into when I started writing them in. I did not realize beforehand all the device-specific quirks that each device/system could have. Some devices had bad timing and quirks that I never anticipated while writing them. I struggled with devices, and my original design for the audio subsystem for example morphed over and over into an abomination that, though works, is basically this giant duct-taped zombie monster.*

- *Shaders were difficult to customize because they had to be duplicated if you wanted slightly different functionality that required more than just changing shader constants.*

- *Orientation of sources was fixed, and required special code for each source to do any custom modification of rotation/position/scale/etc. This is one of those fundamental flaws that I look back on and regret, as it was a stupid idea from the beginning. I originally thought I could get more accurate source position/sizes, but it just turned out to be totally bad. Should have been matrices from the beginning just like with a regular 3D engine.*

Second, the coding flaws:

- *The coding style was inconsistent.*

- *C++98, C-Style C++, there was no exception usage, no STL. C++ used poorly.*

- *Not Invented Here Syndrome everywhere. Custom string functions/classes, custom templates, custom everything everywhere. To be fair, it was all hand-me-down code from the early 2000s that I had become used to, but that*

was no excuse – C-standard libraries and the STL should have been used from the beginning over anything else. That doesn't mean to say that using custom stuff is always bad, but doing it to the extent I did definitely was. Made it horrible to maintain as well, required extra knowledge for plugin developers and anyone messing with the code.

- *Giant monolithic classes everywhere, the main OBS class was paricularly bad in this regard. This meant navigation was a nightmare, and no one really knew where to go or where to add/change things.*

- *Giant monolithic functions everywhere. This was particularly bad because it meant that functions became harder to debug and harder to keep track of what was going on in any particular function at any given time. These large functions, though not inefficient, were delicate and easily breakable. (See OBS::MainCaptureLoop for a nightmarish example, or the listbox subclass window procedure in WindowStuff.cpp)*

- *Very large file sizes with everything clumped up into single files (for another particularly nightmarish example, see WindowStuff.cpp)*

- *Bad formatting. Code could go beyond 200 columns in some cases, making it very unpleasant to read with many editors. Spaces instead of tabs, K&R mixed with allman (which was admittedly my fault).*

New (actual) coding guidelines

- *For the C code (especially in the core), guidelines are pretty strict K&R, kernel style. See the linux kernel "CodingStyle" document for more information. That particular coding style guideline is for more than just style, it actually helps produce a better overall code base.*

- *For C++ code, I still use CamelCase instead of all_lowercase just because I prefer it that way, it feels right with C++ for some reason. It also helps make it distinguishable from C code.*

- *I've started using 8-column tabs for almost everything – I really personally like it over 4-column tabs. I feel that 8-column tabs are very helpful in preventing large amounts of indentation. A self-imposed limitation, if you will. I also use actual tabs now, instead of spaces. Also, I feel that the K&R style looks much better/cleaner when viewed with 8-column tabs.*

- *Preferred maximum columns: 80. I've also been doing this because in combination with 8-column tabs, it further prevents large/bad functions with high indentation. Another self-imposed limitation. Also, it makes for much cleaner viewing in certain editors that wrap (like vim).*

https://github.com/jp9000/obs-studio

50MHz Sine Wave Signal Generator

By John Hudson G3RFL

Completed board and display

I often spend time browsing data sheets, some have amateur television applications, some have amateur radio applications and some fall through the cracks.

The AD9850 DDS sign wave generator was a chip that took my fancy, it is sine wave generator. Sine wave is a mathematical curve that describes a smooth repetitive oscillation. It is named after the function sine, which occurs often in pure and applied mathematics as well as physics, engineering, signal processing and many other fields.

The sine wave is important in physics because it retains its wave shape when added to another sine wave of the same frequency and arbitrary phase and magnitude. It is the only periodic waveform that has this property and makes it acoustically unique.

It is an ideal waveform for testing the frequency response of video equipment, perhaps it is not essential to control it in 1Hz, but as is so often the case a lot of the things I design get used for other purposes that were not envisaged at the time.

The AD9850 chip was an inexpensive purchase on EBay and came complete with a PCB and its own REF XTAL of 125MHz. So all that was needed was some control which could easily be provided by a PIC, associated software, a suitable display and some push buttons I only required 3 of these, but 5 were added, you never know where future revisions will take you.

Button1 advances a cursor, (it is displayed under each digit in the display). Buttons 2 and 3 are up/down buttons, to increment or decrement the selected digit, the oscillator will automatically adjust to suit the display (8 digits in total).

The frequency is stored in the PIC memory so when you power down and restore power up it will return to the last frequency.

Range is 1Hz to 50,000,000 Hz. The Power requirement is 12V at 180mA DC. Output level from memory is 1Volt P~P. A 5V sub Regulator feeds the DDS and the BLUE backlight 4 line display.

Communication between the PIC and the AD9850 is a 3 line bus. Communication to the Display is 4 bit serial connection. The buttons do not need pull resistors as this is done in the PIC. Programming is done with my usual 5 wire socket, standard to all my projects.

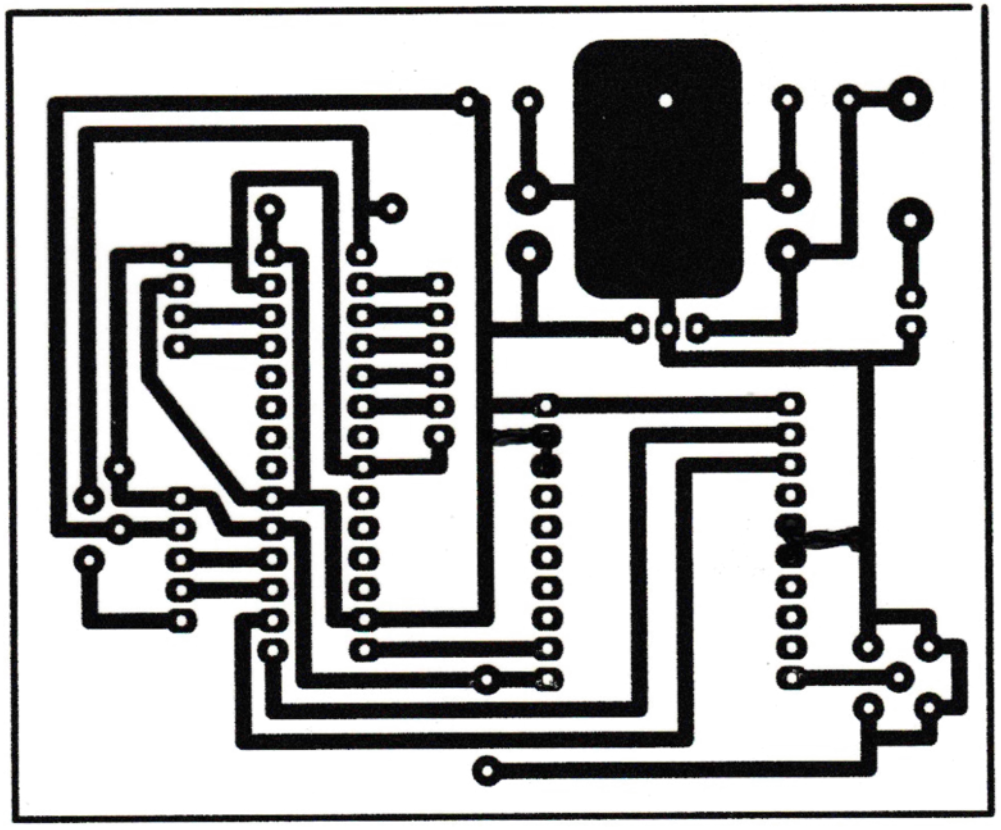
I used the ICD3 programmer and Microchip Assembler.

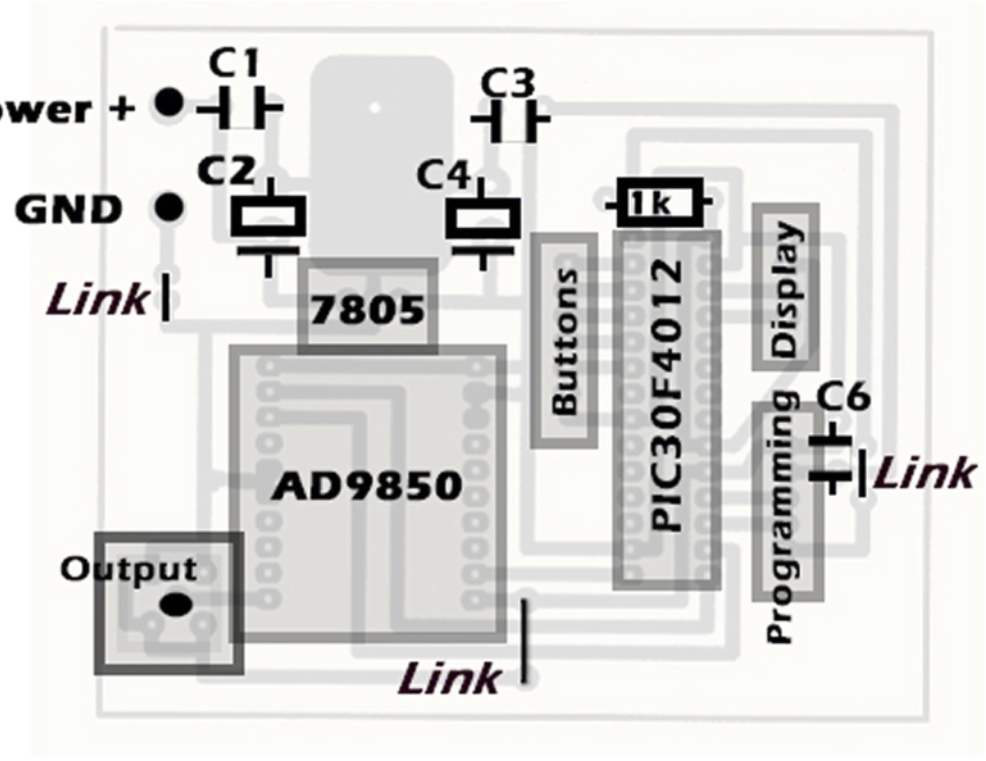
Both the Source Code and the HEX files are on the CQ-DATV download site *http://www.cq-datv.mobi/downloads.php*

The PIC runs on its own internal XTAL above 7MHz with a X4 PLL fast enough for what we require, CONFIG DATA is within the ASSEMBLER software

All the Maths required to convert from a FREQ to the DDS value is taken care of in the SOFTWARE calcdds routine.

On the back of the 20 X 4 display is a serial to parallel chip, (PCF8574) and its input is serial I2C and the DATA and CONTROL sent has two 4 bit blocks.





Above: PCB layout - component side
Left: PCB layout - track side

Construction was on a simple home etched PCB, Yes I did consider using the PCB that came with the AD9850, but in the end I etched my own PCB.

I used an SMA connector for the output signal (the four bolt style) for no other technical reason than that was what I had to hand.

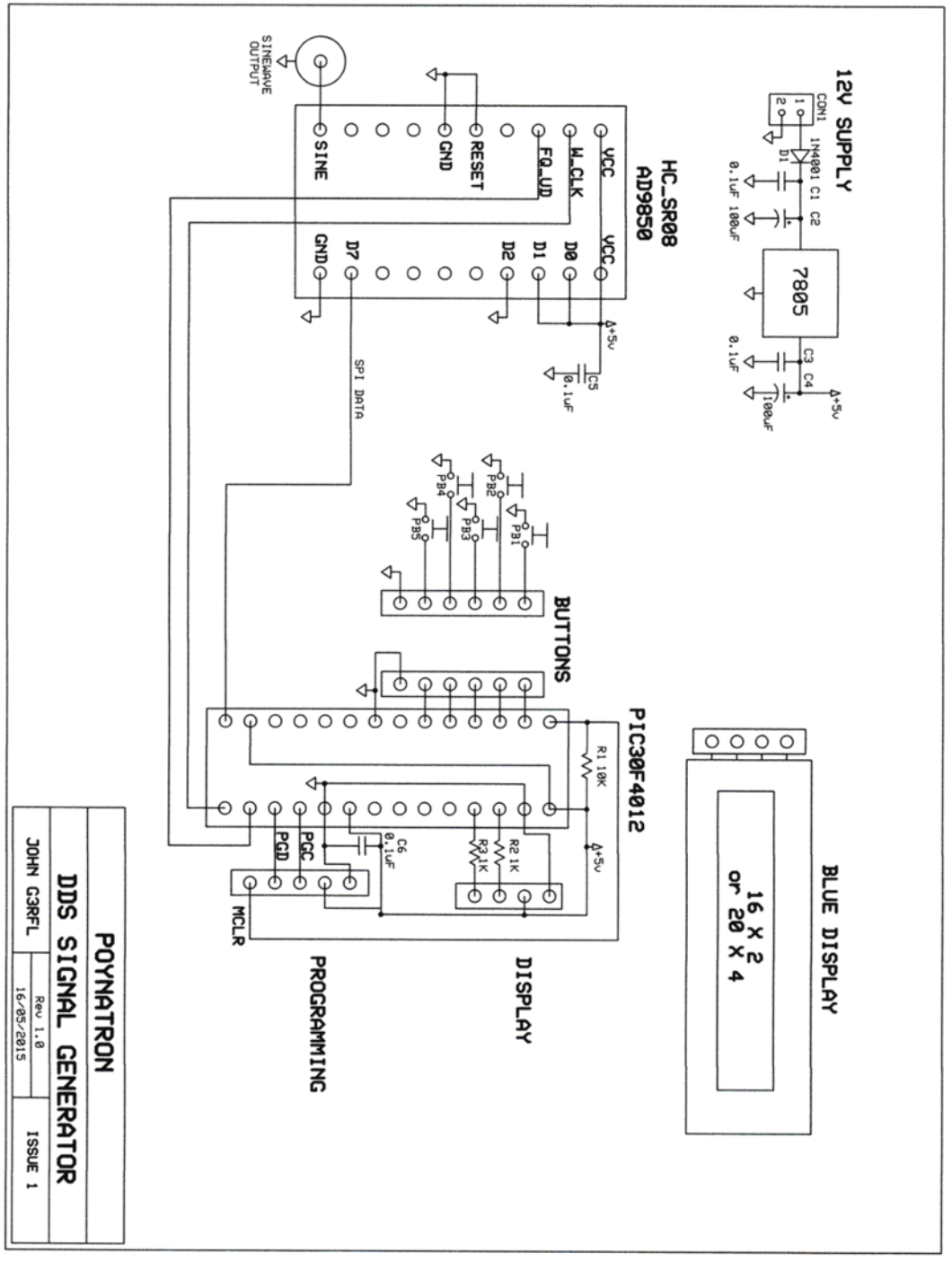
The regulator does get warm and I added a heatsink, but not before I burn my fingers.

Most of the work was in writing the file for the PIC, otherwise the whole unit can be assembled and tested in a single evening.

Only one error I forget the pads for the two resistors R2 and R3, rather than spend another cold evening in my shed etching another PCB I added them on the back of the display, it an I2c bus and it is not critical which end of the wire, they are.

I am working on a box and that will most probably be of the maplin, plastic variety. Let me know if you build the unit or have any problems, you can always reach me via the *editor@cq-datv.mobi*.

Cheers and have fun John G3RFL over 50 years in ATV….and still learning….

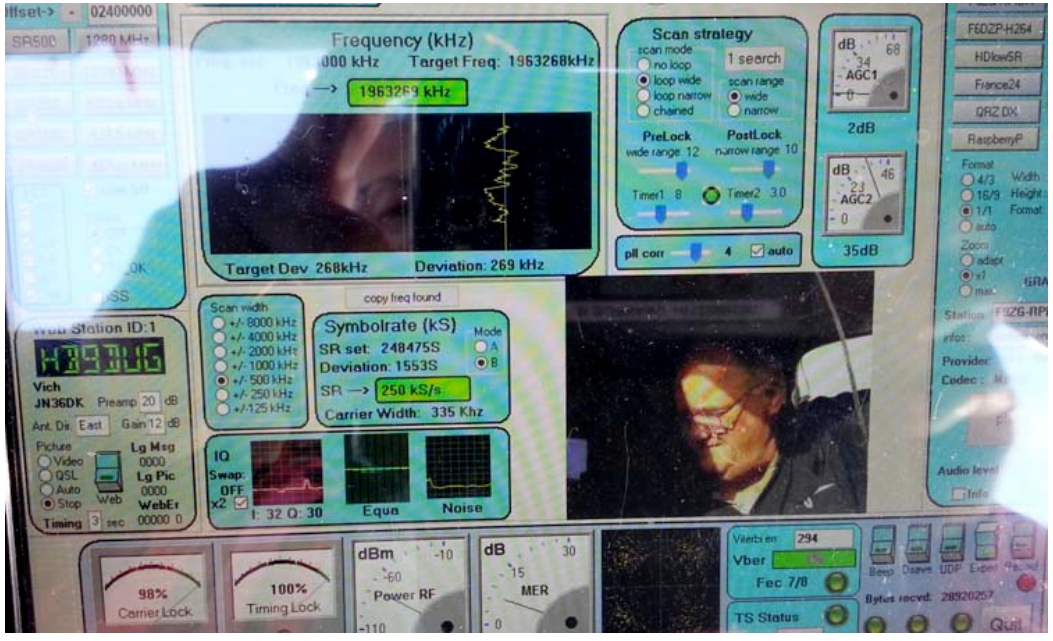


Circuit diagram

DATV-Express Project - September update report

By Ken W6HHC

Charles G4GUO has produced an "experimental release" v1.19 of the Express_DVB-S_Transmitter software for Windows operating system that adds DVB-S2 protocol and corrects the French-language-Windows problem. With assistance from Jean-Pierre F6DZP who spotted that changes for "French-friendly names" required DATV-Express software to search for "Capturer" (French) and in addition to "Capture" (English). Michael HB9DUG performed some testing on French-language-Windows and confirmed that the new Express_DVB-S_Transmitter software now worked correctly with this change.



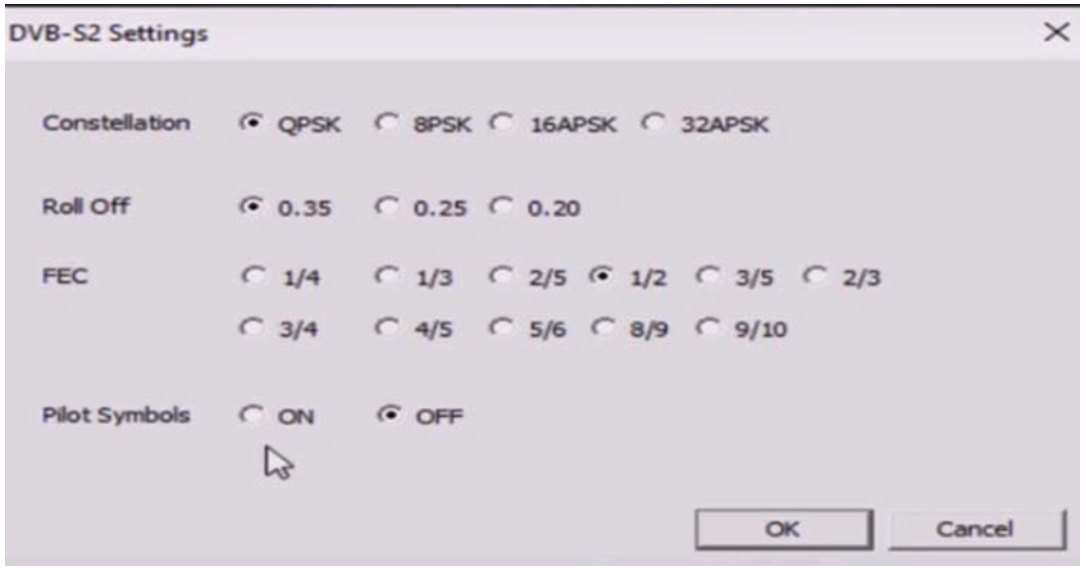
HB9DUG screen-capture during testing of RB-DATV QSO with F9ZG/P using Express_DVB-S_Transmitter (250 KSymb/sec) on 437 MHz over distance of 390 KM (courtesy of Michael HB9DUG)

G4GUO made the "almost final" tweaks inside experimental V1.19 build for porting the DVB-S2 protocol code from Linux to Windows by finishing the addition of DVB-S2 anti-aliasing filters to work with the low symbol-rate settings used for RB-DATV. Having DVB-S2 with low symbol-rate settings for RB-DATV are important, because Jean-Pierre F6DZP announced at CAT16 that he had released a beta-version v0.5 software for the MiniTiouner receiver that allows receiving DVB-S2 down to a symbol-rate of 100 KSymb/sec for QPSK and 8APSK modulations. Commercial DVB-S2 receivers do not work below 1 MSymb/sec.

On Sept 29, G4GUO and Dave G8GKQ and Noel G8GTZ conducted some DVB-S2 field testing on 437 MHz. Charles had a 2-way QSO with Noel on DVB-S2 with QPSK/8PSK modulations and Dave saw/heard Charles' DVB-S2 transmissions using QPSK/8APSK/16APSK/32APSK modulations over a 40 km path.

Ken W6HHC found it very enjoyable to watch live streaming of the CAT16 technical presentations using the BATC's improved (and EXCELLENT) internet streaming facilities. G4GUO gave a presentation on the latest Express_DVB-S_Transmitter software with DVB-S2 capabilities and also future project plans.

Concerning future plans for DATV-Express project, "the bad news" is Charles G4GUO explained during the presentation that the DATV-Express product is reaching end-of-life. Sales have slowed down. We are currently out of board inventory in UK and the US inventory has dropped down to three more boards. The project team has made the difficult decision that when the remaining boards are gone…sales will stop. "The good news" is that a TX/RX SDR board is expected to become available from Lime Microsystems by the end of 2016. G4GUO believes it will be possible to port the Express Transmitter Windows code over to the LimeSDR board. *https://www.crowdsupply.com/lime-micro/limesdr*



This menu shows that DVB-S2 protocol provides many more robust FEC settings than DVB-S (at least for QPSK modulation).

But, porting code is a full software project and a full manual project and will take some time. The hardware cost of LimeSDR board, with high speed USB-3 capability and a much bigger Altera FPGA, appears quite reasonably priced at around US$280 for TX and RX DATV capabilities.

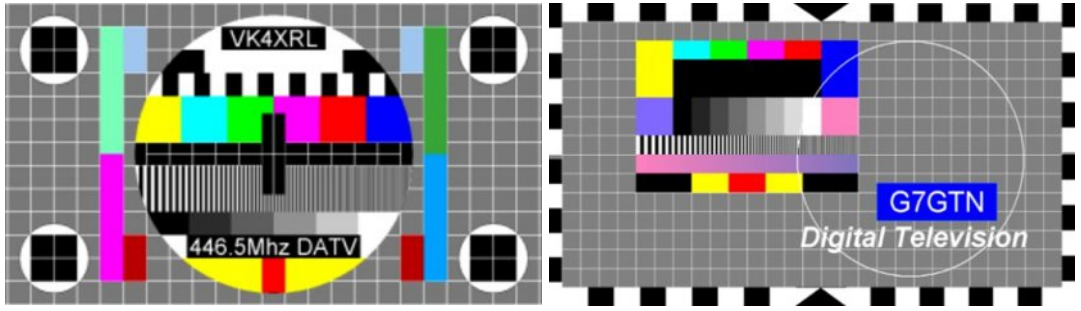
The LimeSDR board from Lime MicroSystems can run both TX and RX from 100 KHz to 3.8 GHz with a high-speed USB3 interface

Charles G4GUO plans to chase down the last-known problem in the experimental V1.19 build and then release a "beta version" of software for Windows software with the new capabilities. Once the new software is released, Ken W6HHC plans to post the new software on the DATV-Express web site and to then begin adding DVB-S2 into a Users Guide for Windows update.

"Project speed set to slow" ...de Ken W6HHC

Digital World - Analogue 8x1 Video and Audio Switcher - Part 1

Richard Carden VK4XRL and Mike Stevens G7GTN

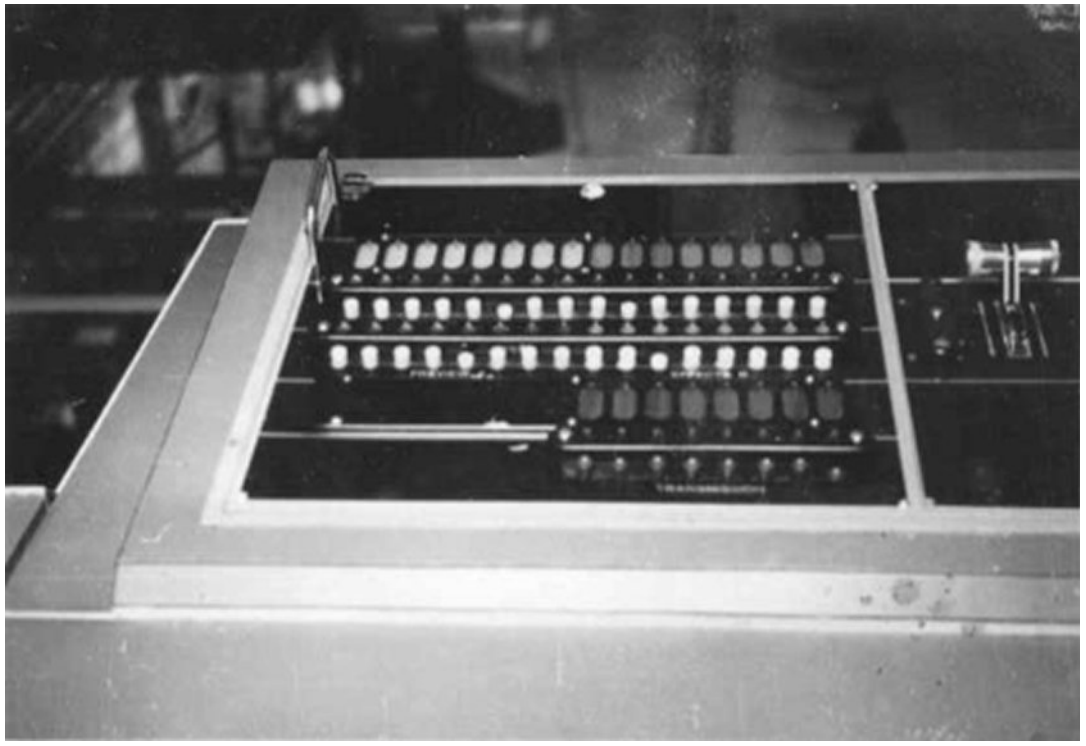


In the last issue of CQ-DATV Mike G7GTN came up with a video switcher using the Maxim MAX455 and previously in CQ-DATV No.36 we had a matrix switcher by Ulrich Vogel DK4BT (This article first appeared in *TV AMATEUR 180*) using the older 4066 IC's.

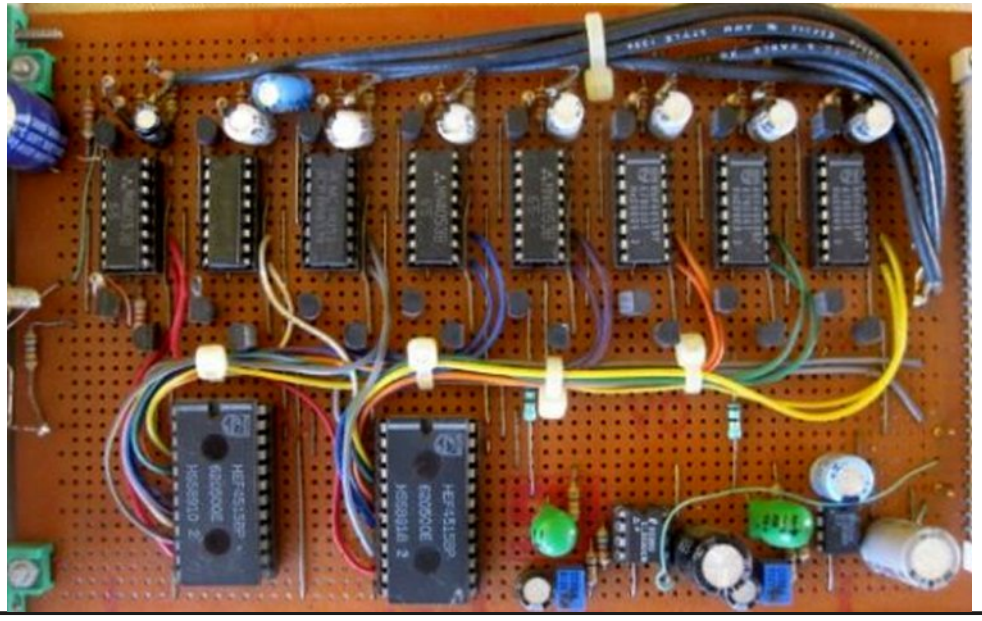
In this article we'll look at using the 74HC4351P to provide a single 8 x 1 video and audio switcher with provision to upgrade to a dual 8 x 2 switcher. In this way we can provide a separate transmission buss and preview buss, but more on this in a later article.

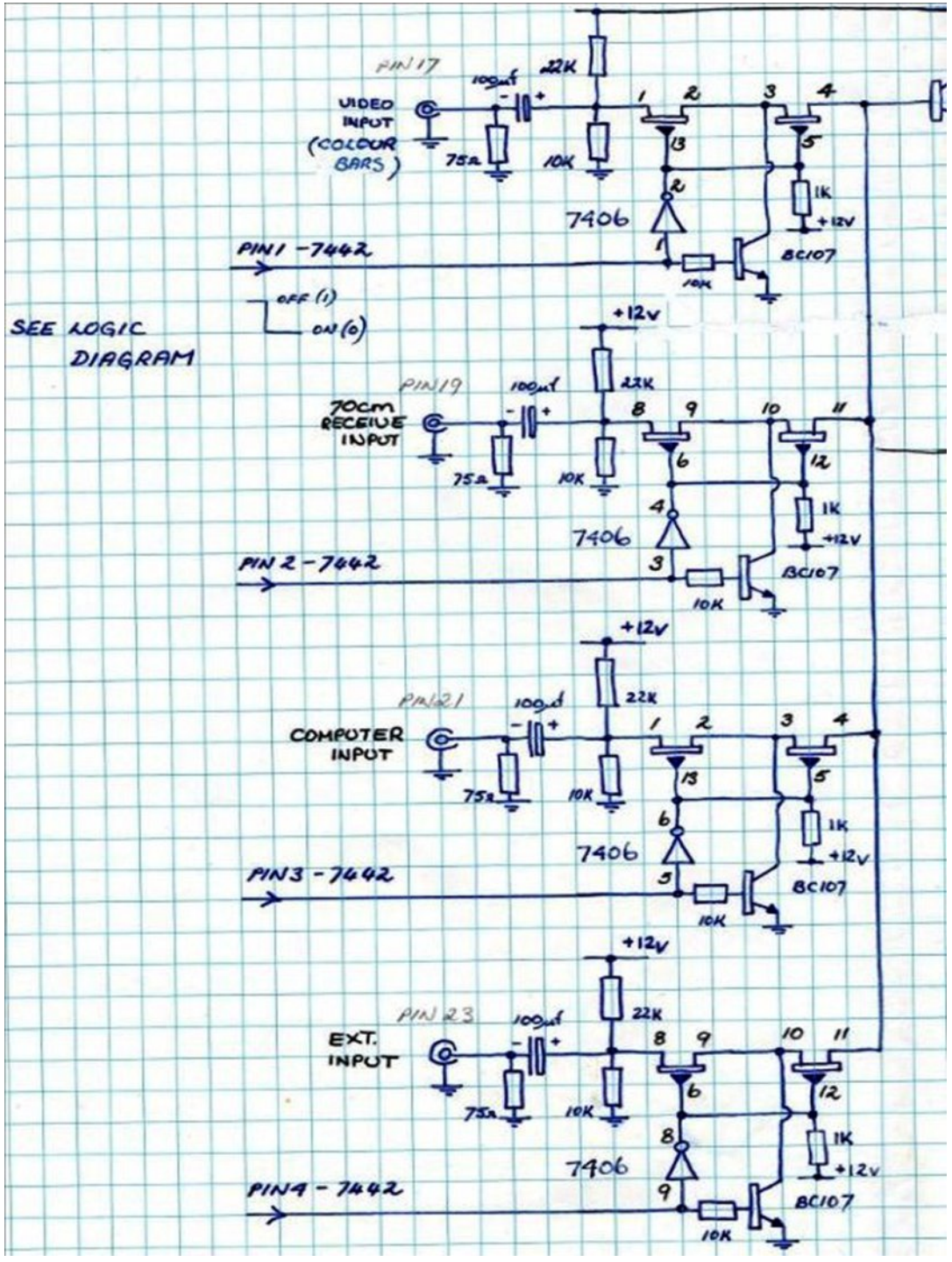
Complete construction will be left to the individual to sort out according to their own needs and construction technics. My own personal approach is not to put all the eggs in the same basket so I would have separate video and audio switchers.

During my early days as a trainee I vision switched live to air in those days using a Marconi switcher as shown. These switchers were a reed relay type with A/B busses for mixing and supering titles plus preview busses. Vision Switching has come a long way since then however analogue for amateur television usage is here to stay for a little while longer although I must say I now just use vMix.



My first vision switcher was for the Sydney ATV repeater VK2RTV and I still have the original circuit as shown.





It uses the 4066 but with dual switchers to provide better crosstalk figures of around 60db or better. Another approach would be to use the 4053 which I have done over the years on a number of switchers that I have built.

This provides a dual buss switcher where control was provided by the HEF 4515BP from the control panel via a 74C922n and two Take buttons.

Also available are those special chips especially made for this type of application although some are expensive or hard to get however one is the Sanyo LA7958 which is only a 4 x 1 switcher for the TV market. This unit contains both stereo and video inputs and has been used in repeater situations.

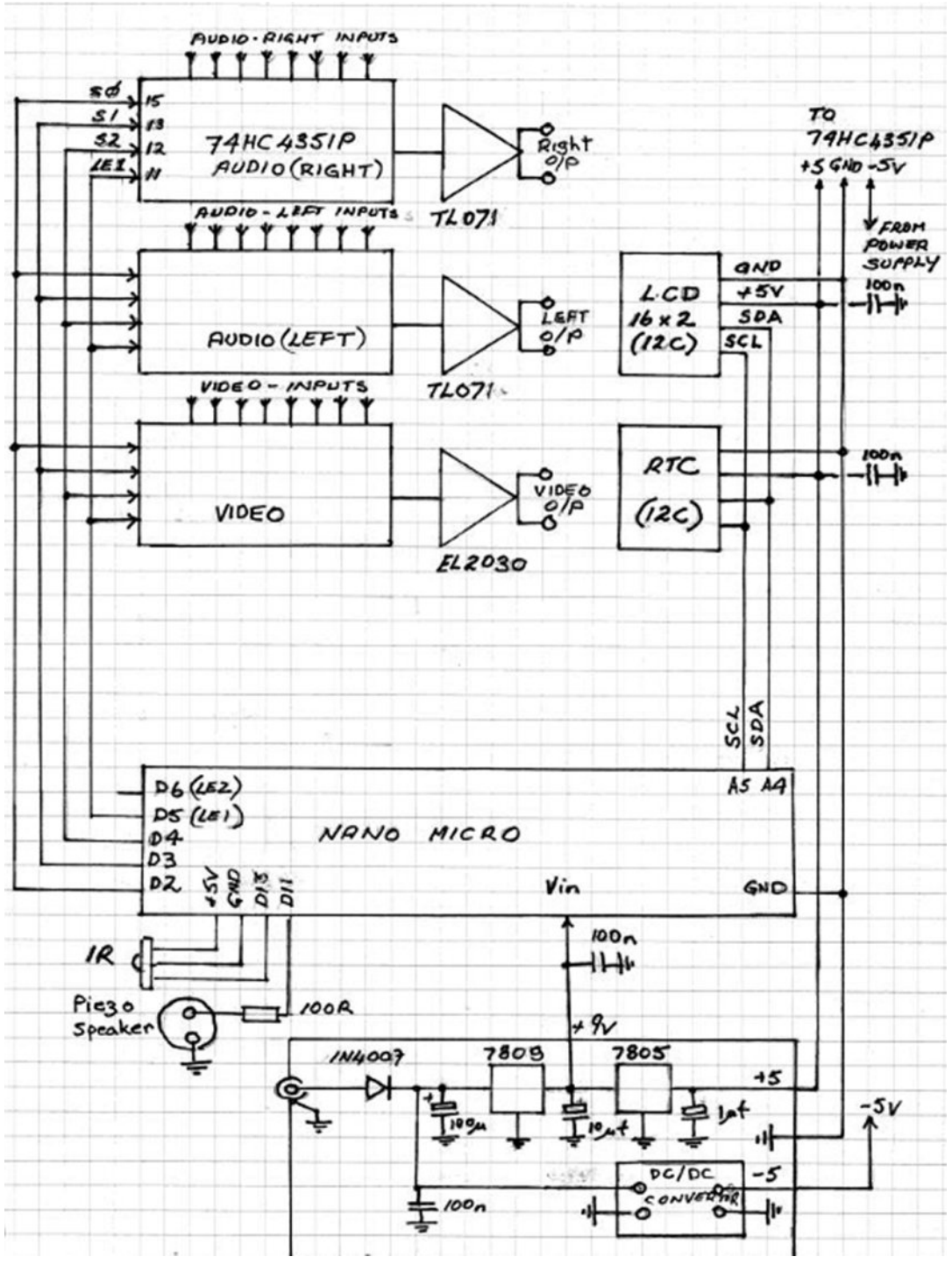
The TDA8440 is another IC which I have used but this is only a 2 x 1 switcher that has audio and video switching and can be added to make a 8 x 1 switcher although the control is not as straight forward but with micro's should make it a little easier.

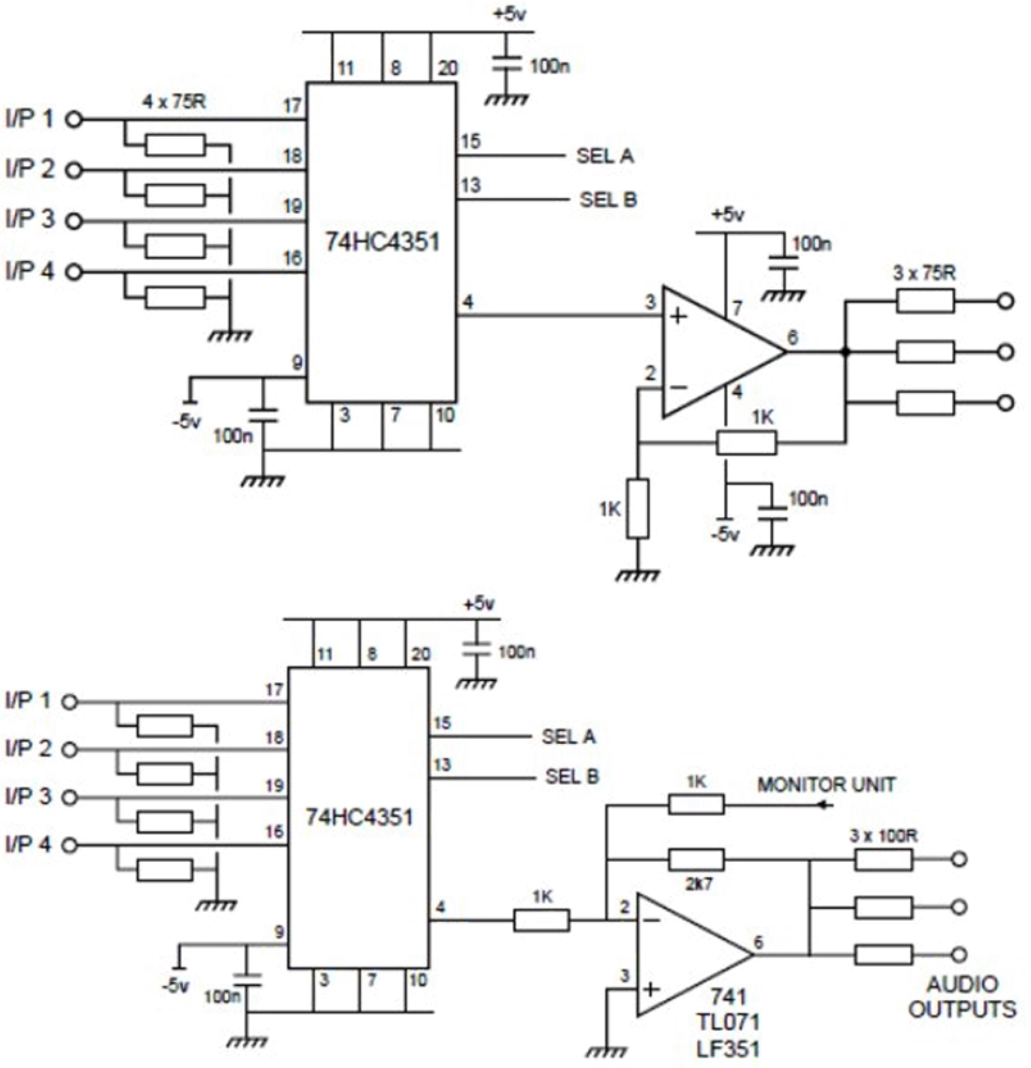
Now let's look at the proposed block diagram of our new switcher and the actual switcher circuit (next page - left).

Looking at the circuit shown (next page - right). we see that only 4 inputs are shown but you can connect the other four inputs as the code has been written for all eight inputs. Also only 2 select lines are shown and that would increase to three when using eight inputs.

To make things easier the same chip is used for the audio switcher. The video output driver IC is an EL 2030 and the audio driver a TL072.

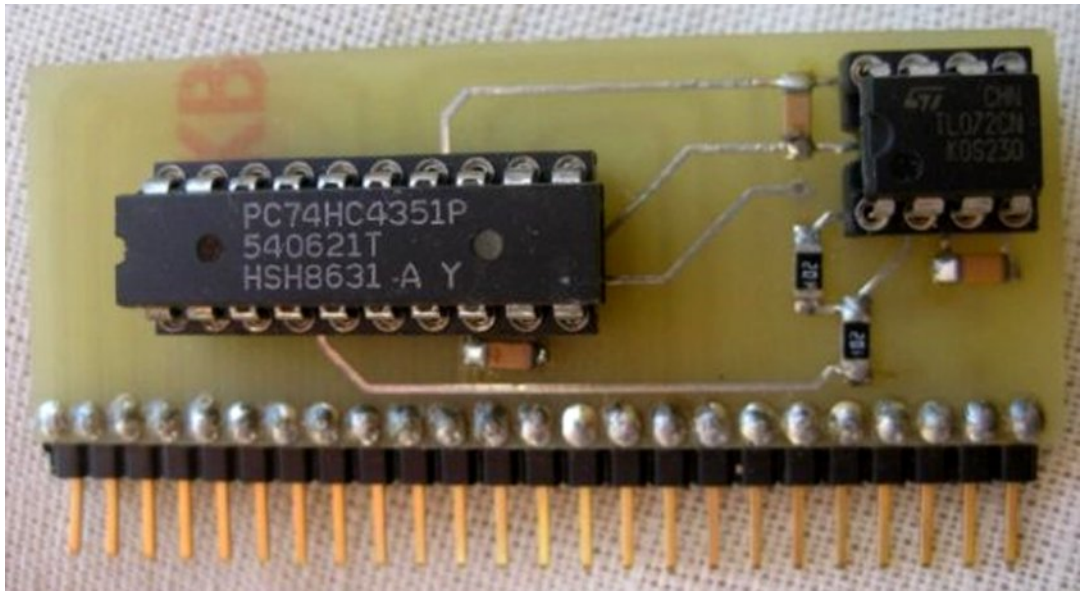
Just remember that two audio switchers are required for stereo audio left and right. The three select lines from the Nano are S0- D2, S1-D3 and S2-D4.

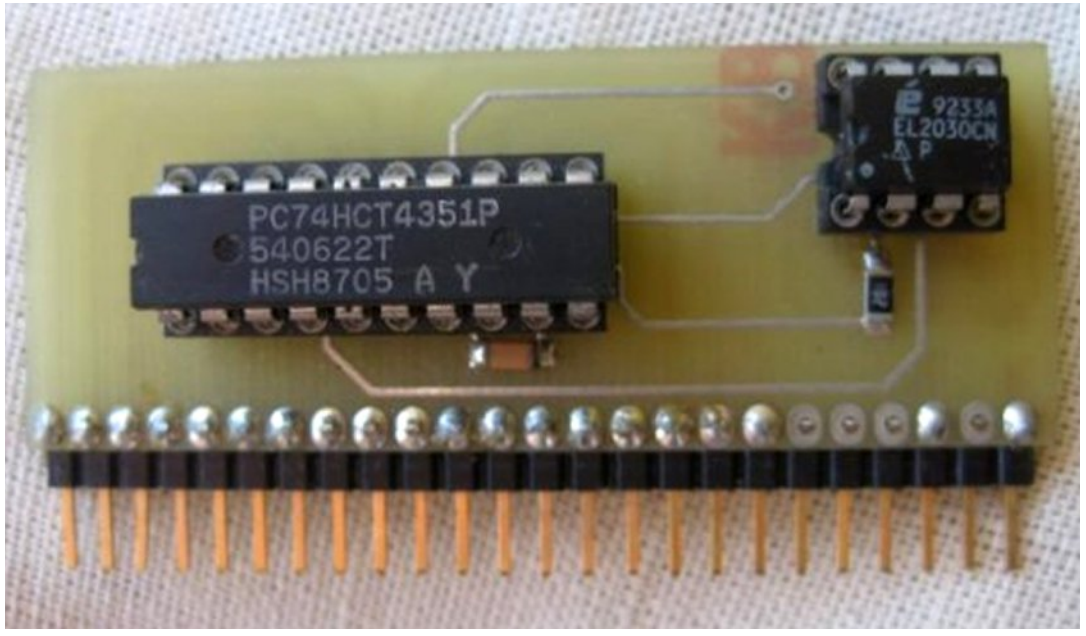




The latch pin (pin 11) is shown connected to +5 in the drawing. However for this application, it should be connected to D5 so that it is controlled from the Nano Micro.

The two following photos shows my own approach to the audio and video switchers including the output stages where Pin 11 (Latch) is taken out for control as stated above.





The controller selected for this project is the quite humble Arduino Nano Micro the actual user switching operation is controlled by the IR television type remote control unit using Sony protocol 38kHz decoding utilising a very well proven additional software library.

You will notice that we have included an external RTC module as the clock and temperature unit.

An LCD 16X2 is used, HD44780 based with an I2C adaptor board. Both of these are using the I2C protocol to be very frugal on actual pin usage.

The first and second lines give you the input selected channel number. Since in this version we had a good compliment of remote buttons available on a standard remote control we added the ability to display the date and time using the channel up (or plus button) we also due to the fact that the DS3231 RTC chip contains a temperature sensor, included this on the display via the Channel Down button.

The channel you initially came from is restored after a time out long enough for you to read the display information.

Both of our I2C devices have an address on the bus; the clock however is at a manufacturer fixed 68 hexadecimal.

The eBay LCD backpack module can be user configured the default with the three solder pads unmade will be 27 hexadecimal. This is the value we have assigned within the code.

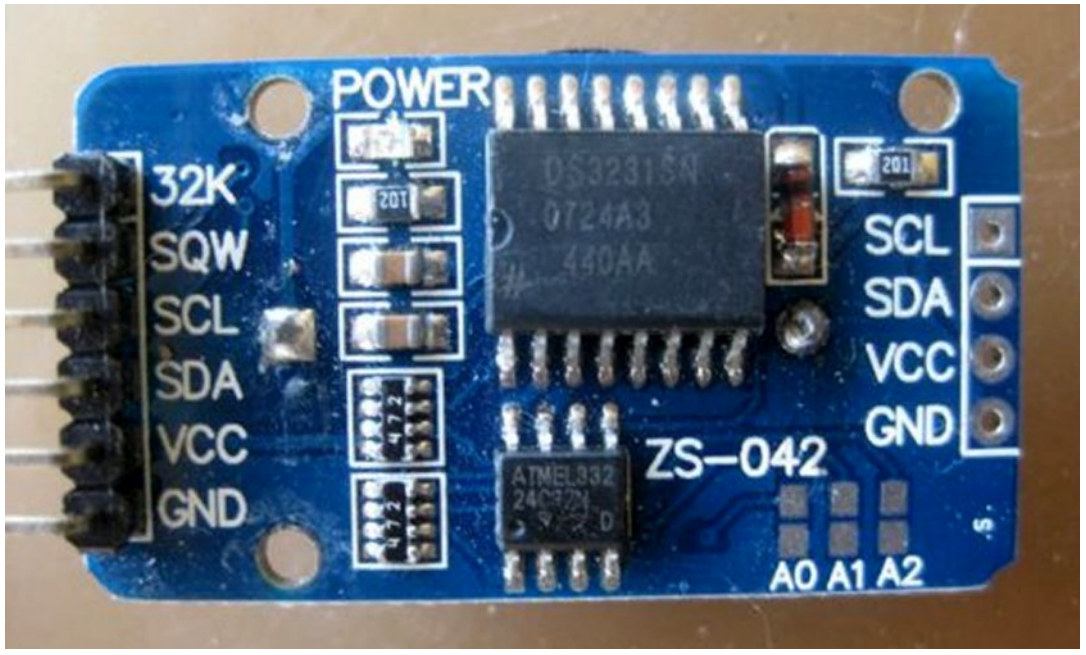
On Sony IR decoding we make use of three digital I/O pins (D2-D4) to set the required logic levels on the 74HCT4351 multiplexer device.

We however require a couple of additional libraries over the standard Arduino installation. For your convenience these have been separately added to an archive file that you may download from the magazine software page.

Setting of the DS3231 RTC device is a one time operation and once you have downloaded the code you must comment out the #define SETRTC and download the exact same code for a second time.

Remote Button operations are therefore as follows;

One to eight button select inputs one to eight Channel up (+)
Date and Time Channel down (-) Temperature
The power button will reset the Nano.



The RTC ZS-042 as well maybe others have a battery charger circuit that should be disconnected as it is using a Non-chargeable button Battery. Either cut the track or remove the diode or resistor as shown in the photo.

Sony IR Hexadecimal Remote Button Codes

Numeric buttons	
Zero	0x910
One	0x10
Two	0x810
Three	0x410
Four	0Xc10
Five	0x210
Six	0Xa10
Seven	0x610
Eight	0XE10
Nine	0x110

Control buttons	
Channel Up	0x90
Channel Down	0x890
Volume Up	0x490
Volume Down	0xC90
Favourite	0x37EE
Power	0XA90
Sleep	0x6D0
Up	0x90
Down	0x890
Left	0xAF0
Right	0x2F0
Sub-T	0xD38
OK	0x3f0

Teletext Buttons	
Text RED	0x338
Text Green	0xB38
Text Yellow	0x738
Text Blue	0xF38

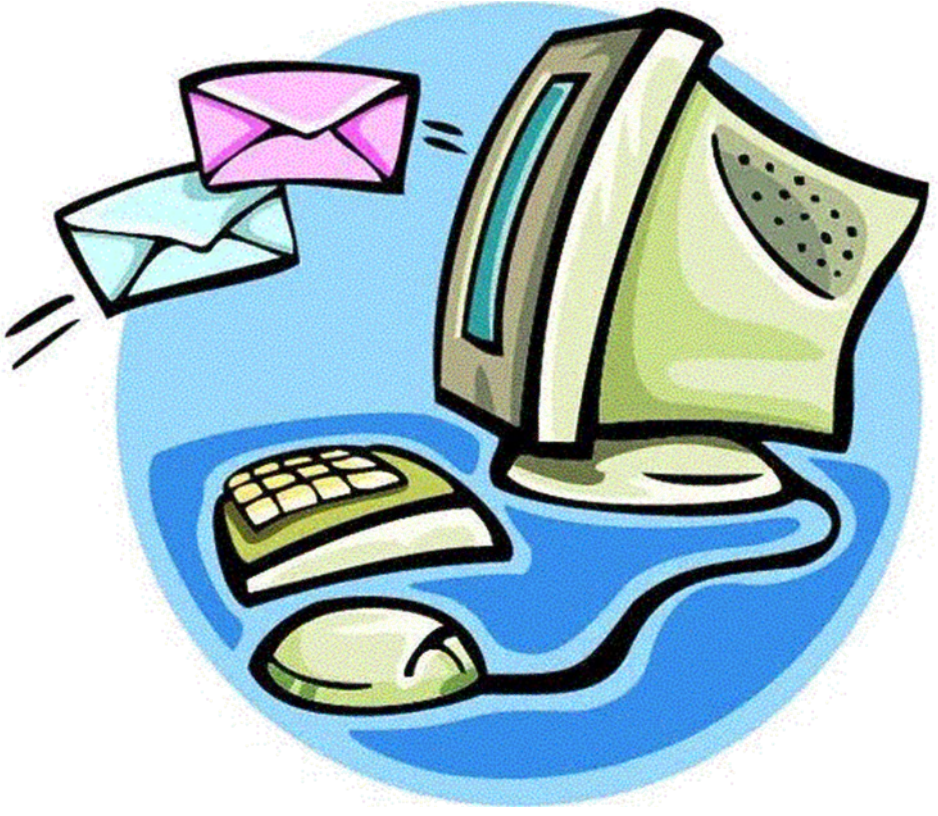
Operation of Buttons 1 to 8 changes the LCD top line and allows you via the code to change what you have as that input, the photo shows input 7 as Camera 1.

Testing can be carried out while you wait for parts to arrive by using LED's, don't forget the series resistor, 220ohms connected to D2, D3 and D4.

Pressing the remote buttons you should see the LED's light up as per the code. Also buy pressing the zero button the switcher will sequence from input one to input eight and returning to input one.

In part two we will provide a second switcher which therefore will give us the ability to have a TX and PV bus.

My thanks to Mike G7GTN for his development of the software.



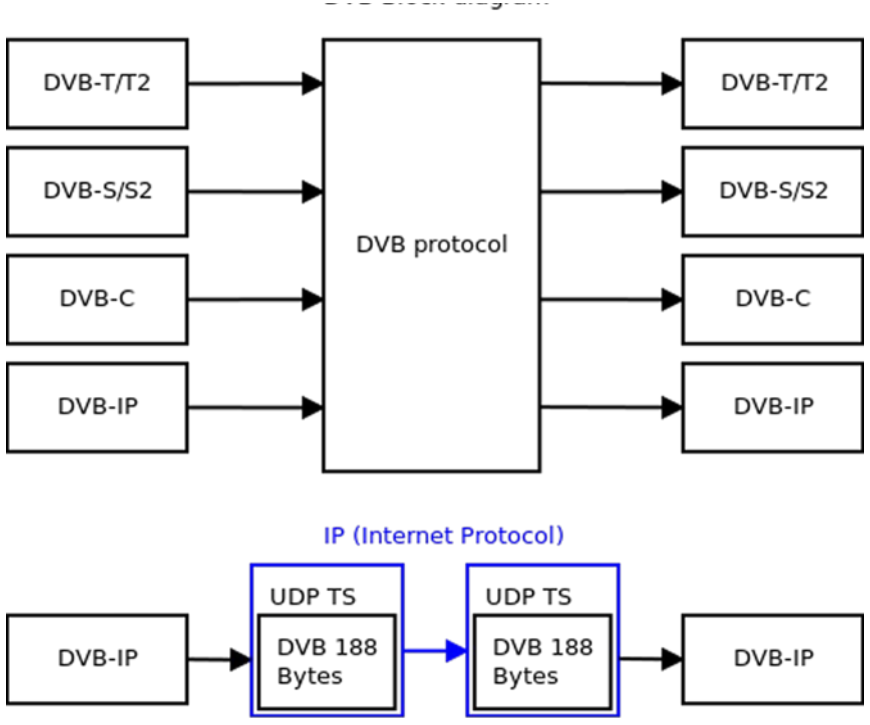
Want to be notified when issues of CQ-DATV are published? Then join our mailing list.

DATV repeater linking

By Grant Taylor VE3XTV

Introduction:

The DVB standard uses a common protocol for all forms of transmission, anther way to look at this as layer one the physical layer in networking terms. Meaning that the RF, Coaxial cable and the Ethernet used to connect to the Internet, are all sending the same digital information. This is why we used media converters, to convert from one physical format to the other, this is where the name comes from. Within the DVB standard we have all these forms of modulation, such as DVB-S/S2, DVB-C, DVB-T/T2. As you can see from this block diagram below all forms of DVB can be inter-changeable, at transport stream level, this could be ASI or UDP TS.



My aim is to get everyone away from transcoding between video formats such as Mpeg2, h264 and analog video. This is the system I have been using since 2010, is based on using the DVB transport stream for use over RF, LAN or WAN. There is digital cable TV equipment already on the market that will do this, it works between IP UDP and ASI.

UDP (User Datagram Protocol) is a commonly used protocol and therefore most firewalls can be configured to pass through these packets.

Where ASI (Asynchronous Serial Interface) is a common standard that is used on DVB equipment to inter connect between different DVB hardware modules.

A DATV repeater is nothing more then a data repeater using this DVB protocol, this why you have DVB-S/S2, DVB-C, DVB-T/T2, all these formats have DVB in there names as underlining protocol based on 188 bytes within 8192 packet per frame.

How will this all work with DATV:

The home station encodes to h264 and AAC for example, this video information remains unchanged through out the digital path, RF to IP and back to RF again. The digital signal is only decoded at the receiver end and fed via HDMI to the TV set (no analog video is used).

For this to work with a digital ATV repeaters at the receive end a standard bit rate is used for all Video On Demand (VOD) repeater inputs for SD and HD.

When the IP stream comes in from the Internet it goes through an UDP to ASI media converter, the one thing that may need to be done is for the DVB PID's to be re-maped to fit the multiplexer setting used at the repeater end.

UDP is only the outer protocol used with the DVB transport stream within is unchanged. With IPv4 there is a range of broadcast IP address or multicast addresses from 224.0.0.0 to 239.255.255.255. If we had a multicast IP address every repeater world wide could be allocated a port on this one IP address, providing for up to 65,536 users.

Receiving these repeater streams off the Internet is easily done with VLC on your home computer, VLC can also be used to encode to a UDP TS as well, giving you both send and receive abilities.

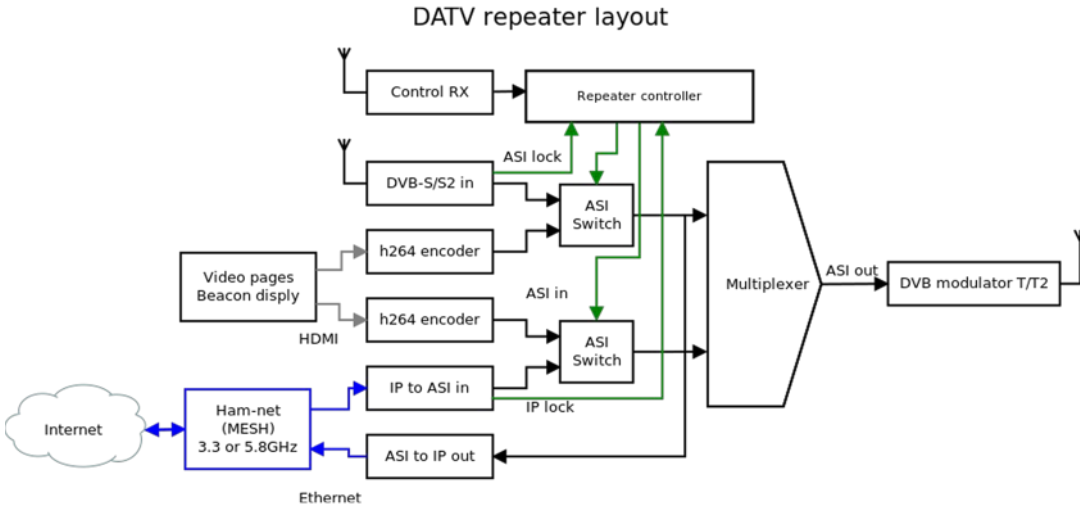
Asynchronous serial interface

Asynchronous serial interface (ASI), is a form communications that is used between hardware DVB devices, that operates at the same bit-rate as Serial Digital Interface (SDI) at 270Mb/s this is where the similarity ends between them.

Where SDI is used with Standard Definition (SD) uncompressed digital video and ASI is used for compressed video formats.

The DVB transport stream packets are in 188 byte blocks with a 13bit header, making up a DVB frame that has 8192 packets within. With ASI standard there is an extra 16bytes added in to make 204bytes within a block, giving a ratio of 204/188 with null packets. Like with analog video ASI, SDI have digital levels at 1v pk-pk into 75ohms using BNC connectors.

Building a DATV repeater is done by using cable DVB equipment, such as Encoders. Multiplexer, IP to ASI converters and Modulates, and are all inter-connected with ASI cables. So thinking about how DATV repeaters work you need to interface an IP connection, User



Datagram Protocol (UDP) has been used for many years for Ethernet cable TV systems and is supported well with this cable equipment. In Ham radio we have our own RF form of Ethernet called the MESH or Ham-net, this therefore becomes part of DATV repeater design. For this to work you need be using the Ham-net at high bit-rates around 40Mb/s, this can be done with Ham-net equipment operating on bands above 9cm (3.3GHz).

The DATV repeater design above has two encoders to convert from HDMI uncompressed video to ASI, as in an analog repeater you switch between repeater and beacon modes.

The DVB-S/S2 receiver only demodulates to the transport level and outputs this as ASI. Therefore there is no transcoding been done, all that the Multiplexer is doing is rewriting the service information table and packet identification (PID) headers.

The Ham-net / MESH is interconnect between the repeater via a firewall to the Internet for IP streaming. A multi-standard DVB-T/T2 receiver is required to decode what is fed through from the sending end to be displayed.

With IP linking of DATV repeaters we are using UDP TS at 188bytes which is a common international standard. But on the encoding side we have the US been the odd one out with their 60Hz based system, so we are now looking a two standards one for 50Hz and the other for 60Hz. For SD and HD we should be looking at these formats:

SD 480i/60 at 720 x 480 using h264 (3Mb/s) with AAC (2 channel at 48k at 128kb/s)

SD 576i/50 at 720 x 576 using h264 (3Mb/s) with AAC (2 channel at 48k at 128kb/s)

HD 1080i50/60 at 1920 x 1080 using h264 with AAC (bit rate to be worked out)

Note, most if not all digital receivers these days can decode both 50/60Hz formats and newer TV sets also can display these formats, so for repeater linking we can stay with the original video and audio information unchanged throughout.

Video On Demand (VOD) is the term used for selecting IP video feeds off the Internet, and therefore I will used this when talking about repeater linking. VOD switching can be done automatically or manually from the home station, this can be done by taking off the IP lock LED on the repeater based IP to ASI converter or by a Raspberry Pi.

Using an IP to ASI media converter it is done by the use of an opto-coupler across the IP lock LED or the ASI lock LED, to provide a logic output for the repeater controller. The Raspberry Pi3 works by switching IP streams using VLC from the command line in software, becoming a software switch. It is also possible to remap PID headers and protocol conversions from RTSP to UDP TS. For this all to work we come back to a common international standard for repeater linking. The aim is to use IRLP longside the DATV linking system, one for voice communications and the other for the audiovisual path.

Check out the DKARS website at:-
http://www.dkars.nl/

Testing a cheap Chinese duplexer as a 70cm DATV DVB-T 7 MHz TX filter- Wow!

By Dr Andrew (Drew) Wollin - VK4ZXI

Introduction

In my last article (*CQ-DATV 40*), I described a cheap Chinese duplexer re-tuned as 70 cm DATV DVB-T 7 MHz TX filter.

The duplexer uses notch cavity filters, six in all. The notch filters have a much sharper edge, compared to a band-pass filter. The sharp notch seems suited to the vertical edges of a DVB-T signal.

I initially check the signal source, a HiDes camera with direct DVB-T output at 1080P. I was a little surprised at the spread, but the filter cleaned it up well.

This would indicate the need for a filter before the main power amplifier.

I pressed on with just one filter and tried it at the output of the amplifier, a 10 W device, from Darko OE7DBH, using a RA60H4047M1 60 W module.

Even with the indifferent input, the filter was able to reduce the spread to -60 dB and give a clean 10 W output.

The notch duplexer/filter seems to overcome some of the major hurdles with DVB-T amplifiers and warrants further investigation.

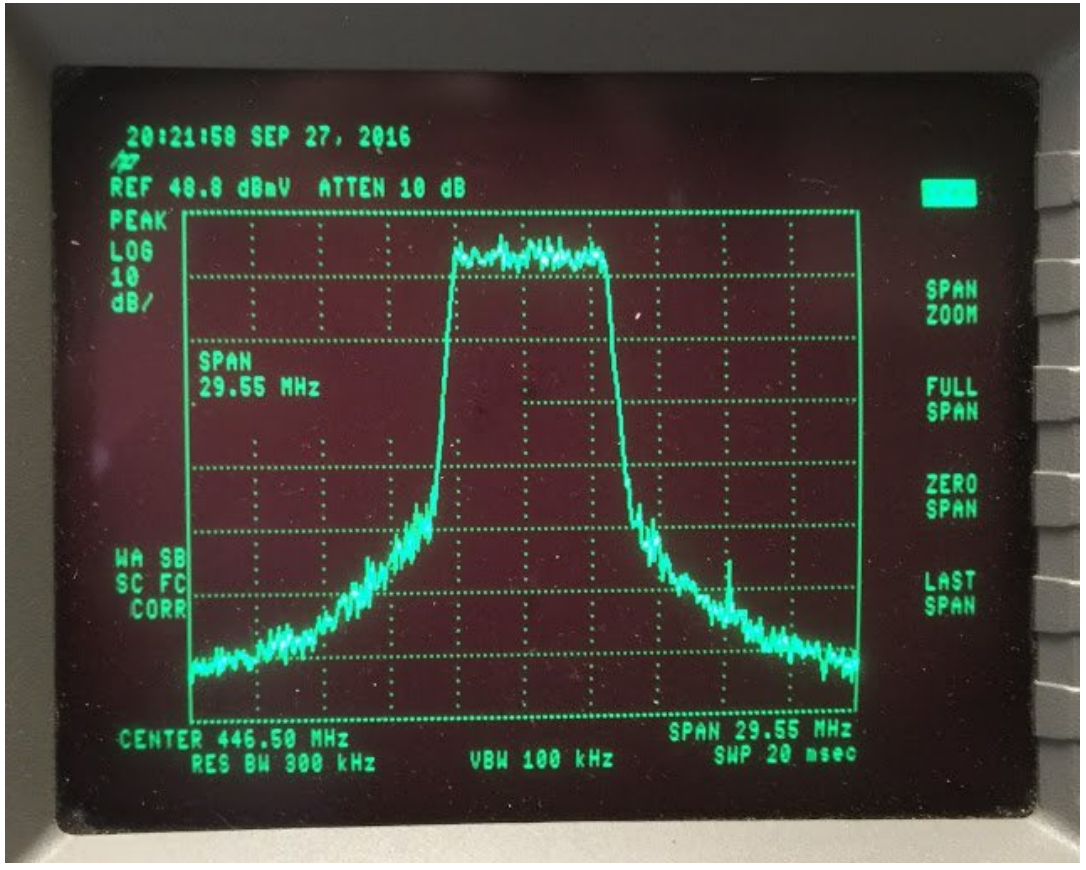
I have not investigated the effects of the filter and different power levels on signal quality at the receiver. I have limited instrumentation, but will report my findings in the next post.

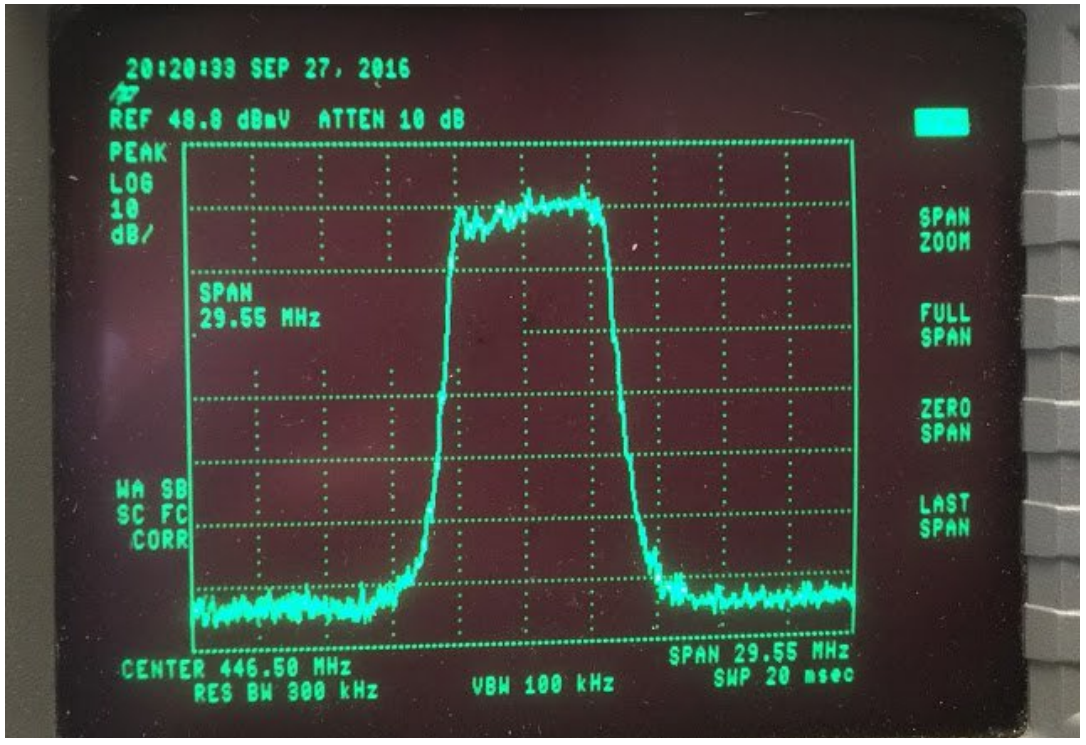
Instrumentation

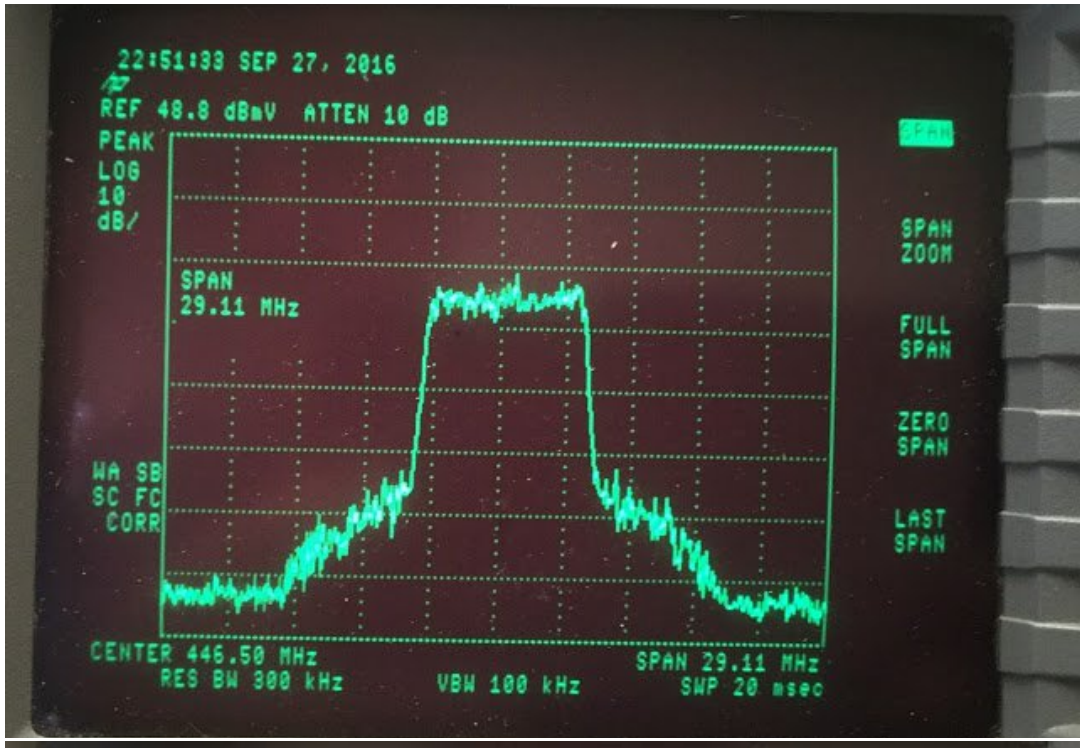
A 20 dB directional coupler, plus an additional 30 dB of attenuation, was used to tap a signal from the transmit path to a HP 8591A spectrum analyser. A cheap SWR/Power meter was used in line to give some idea of output.

Filter before the amplifier

The duplexer/filter works remarkably well. In setting up to test the amplifier, I checked the source from a HiDes camera with direct DVB-T output; 7MHz channel, centered on 446.5 MHz, just to check it was clean. It wasn't too good, acceptable maybe at -40 dB, but with quite a spread.







Inserting the duplexer/filter cleaned it up almost perfectly!

Now I need another duplexer/filter to put after the amplifier.

Preliminary tests with just output filter

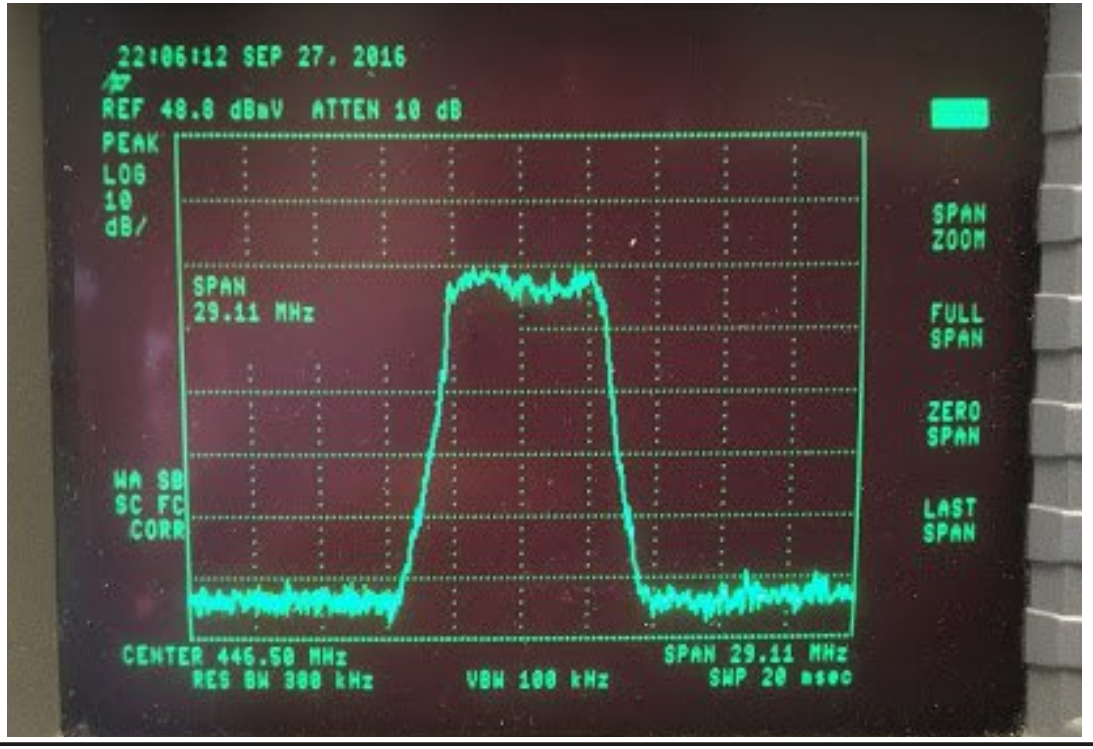
(**Upper right pic**)

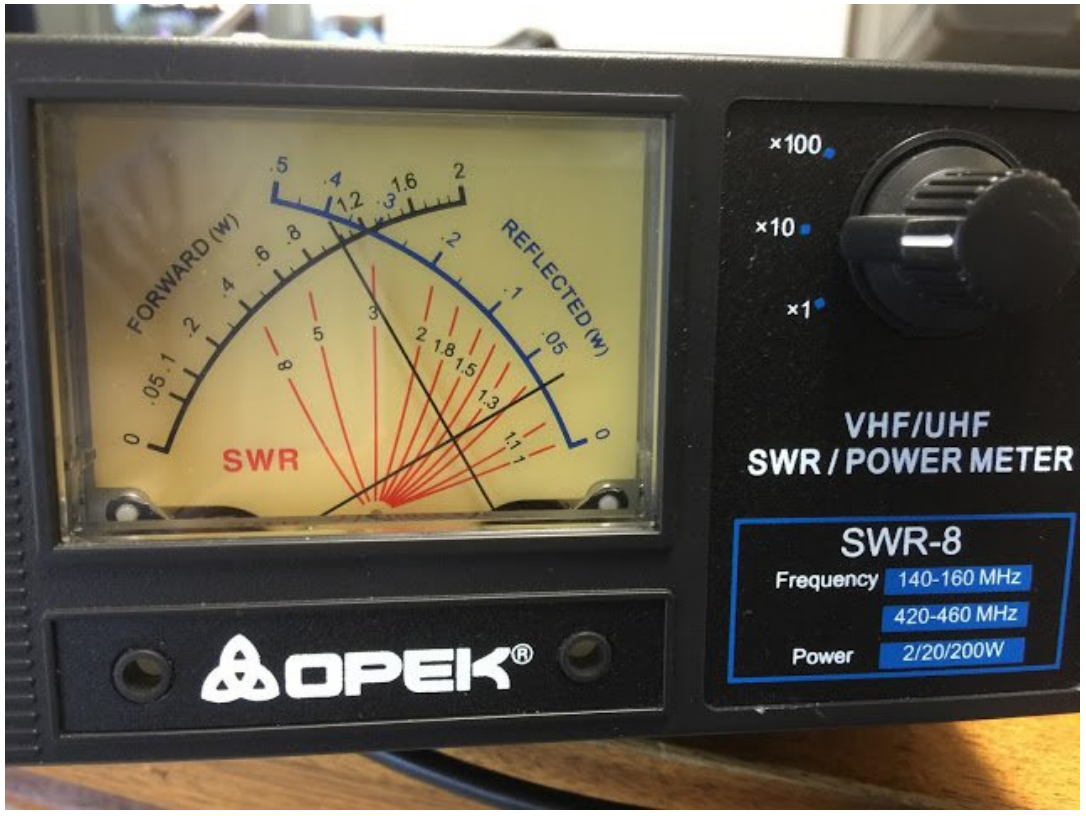
With the amount of gear needed to test a DVB-T amplifier, I thought I might see how the filter works, even with the less than perfect signal from the camera source.

With no filter and adjusting the input to keep spread at about -30 dB gives about 6 W, but it is not pretty.

Filter after the amplifier (**Right pic**)

With the filter after the amplfier, the results are surprising; with adjustment, a clean 10 W signal. The spread is 60 dB down. Magic! Drawing about 8 A at 13.8 V.





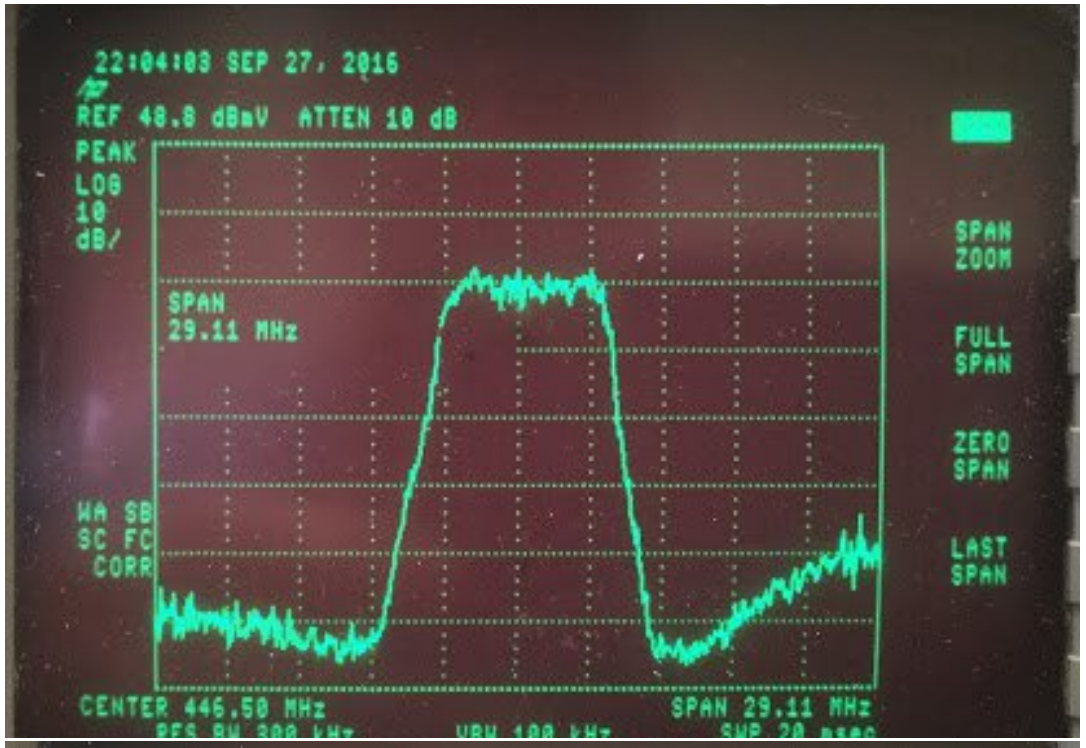
The power meter is showing 10 W. However I am not sure that is the full envelope power of a 7 MHz wide DVB-T signal.

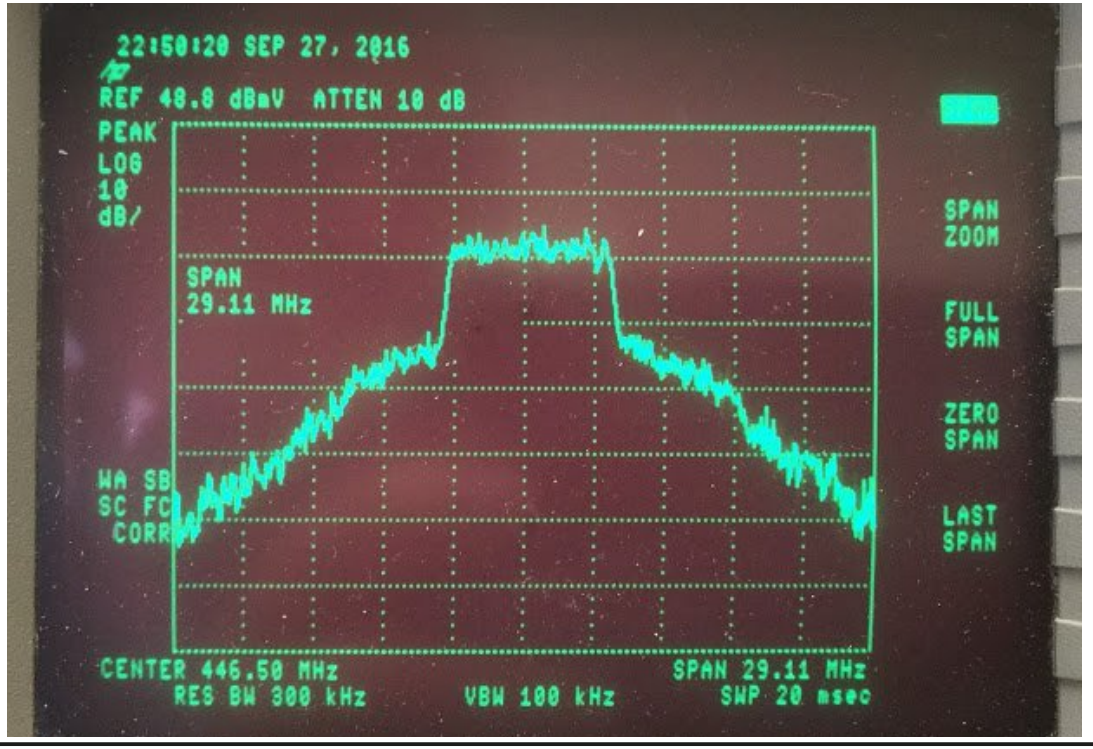
Cheap meters are for measuring low bandwidth CW and SSB signals, not complex ones. (I would like this clarified/explained by someone in the Yahoo group).

For amusement, I tried the direct signal input, thus over-driving the amplifier and putting out a few extra Watts of power. The result shows the notch-nature of the cavities and duplexer. (**Upper right pic**)

There is some rubbish, still -40 dB, above and below the filter's two notches.

Taking out the filter, and about 20 W output, with terrible spread. It sure does some clean-up job!(**Right pic**)





Conclusion

The notch duplexer/filter seems to overcome many of the problems of amplifiers for DVB-T.

- A filter seems to be needed both before and after the main power amplifier.
- There is still probably a need for a lowpass filter to stop harmonics.
- The received signal quality needs to be checked.

This is a preliminary study and needs to be tried by others.

I again thank Martin VK4JVC for suggesting a duplexer rather than building an interdigital filter.

External links

If you have an eBook reader that does not have WiFi then you will not be able to use the hyper-links in this publication. If you have an eBook reader that has WiFi then you will be able to providing you are in a WiFi zone.

But if you have a Kindle 3G then yes, but only to Amazon, and there is not a lot of ATV material on their site.
Smart phone reading apps are ok providing that you have a 3G data connection.

Note: These links will fire up your devices browser and if you are using 3G/4G then you will incur data usages charges.

Legal Niceties (the small print)

E&OE. Whilst every care is taken in the production of this publication, dotMOBI accepts no legal responsibility for the advice, data and opinions expressed. dotMOBI neither endorses nor is it responsible for the content of advertisements or the activities of those advertisers. No guarantee of accuracy is implied or given for the material herein. dotMOBI expressly disclaims all liability to any person in respect of anything and in respect of the consequences of anything done or omitted to be done wholly or partly in reliance upon the whole or any part of this publication.
As the regulations for the operation of radio frequency equipment vary in different countries, readers are advised to check that building or operating any piece of equipment described in dotMOBI will not contravene the rules that apply in their own country.
All copyrights and trademarks mentioned in this publication are acknowledged and no infringement of the intellectual copyright of others is intended.

Copyright

Notice to Contributors

Authors are alone responsible for the content of their articles, including factual and legal accuracy, and opinions expressed by them may not reflect the editorial stance of the publication. Material submitted to dotMOBI should not infringe the copyright of other writers or bodies.
Contributions are accepted for publication on this basis alone.
dotMOBI publications - *http://cq-datv.mobi*

Author Guidelines

CQ-DATV welcomes contributions from our readers. It does not necessarily have to be on ATV, as long as it is of interest to our readers.

Although a formatted article showing the layout can be sent, we prefer an unformatted text file of the script, along with annotations of where important images should be placed. All images should be identified as Fig 1 etc and sent seperately.

Images should be in PNG format if possible and the best quality available. Do not resize or compress images, we will do all the rework necessary to publish them.

If you are sending a construction project, please include the dimensions of any pcb's and make the pcb image black and white, not greyscale.

CQ-DATV reserves the right to redraw any schematics and pcb layouts to meet our standards.

Coming up in CQ-DATV

Is this the latest issue of CQ-DATV? *Click here* to go to our web site to check to see if there is a later edition available.