The All-Digital Transceiver

In this important article we look at an all-amateur, all-digital transceiver.

The All Digital Transceiver is a compact single board design.

MARCH OF PROGRESS. Integrated circuits are now available that can directly convert 0-30MHz analogue signals to and from the digital domain. This article describes a project undertaken by the authors to build the entire signal path of an HF transceiver digitally. The RF section is built on a 150x70mm printed card, with most of the functions carried out by a field programmable gate array (FPGA). The remaining signal processing is done in computer software. A Universal Serial Bus (USB) cable carries digital signals between the two parts and also powers the card. It can be used as a general-coverage HF receiver as it stands and the addition of a power amplifier and antenna switching will enable a complete transceiver to be constructed. The configuration of the FPGA logic is stored in a non-volatile memory on the card and can be upgraded from the computer.

This article describes the fundamental principles of the digital processes used, outlines the block diagrams of the receive and transmit paths and describes the inner working of the important blocks. The block diagram of the hardware is also presented, with details of the functions of the main integrated circuits. The modulation and demodulation processes that take place in the computer software may be described in a subsequent article.

FIRST PRINCIPLES. In the traditional superhet receiver, a mixer converts the incoming band to an intermediate frequency at which the channel filter selects the signal of interest. This is then passed to a demodulator to produce the final output. In mathematical terms, a mixer multiplies the signal by the local oscillator. If both signal and oscillator were pure sine waves and the mixer was a perfect multiplier, its output would consist only of the sum and difference signals or, to put it another way, the mixer would only respond to inputs on the wanted and image frequencies. However, real analogue mixers are imperfect and the best ones are more like switches than multipliers. Such mixers give excellent signal linearity but really bad 'oscillator linearity', which means they respond to signals either side of multiples of the oscillator frequency. For example, an analogue receiver with an oscillator on 1MHz and an IF of 100kHz will respond not only to signals on 900 and 1100kHz, but also to signals on 1900 and 2100kHz, 2900 and 3100kHz and so on. The magnitude of these additional responses depend on the purity of the local oscillator. All but one of the responses must be suppressed, usually with filters. The design of the complete receiver is thus constrained by the achievable filter performance. For example, the image rejection requirement puts a lower limit on the choice of the intermediate frequency.

Digital mixers, on the other hand, are perfect multipliers and do not exhibit oscillator harmonic responses. A digital front end using the above example frequencies would only respond to inputs on 900 and 1100kHz. There is a technique using two mixers, fed with 90-degree-shifted oscillators, in which the outputs of the two mixers are combined to cancel the image response. Although this technique can be used in the analogue world, the cancellation is far from perfect. However, a pair of digital multipliers, fed with digital sine and cosine waveforms, makes a perfect mixer with a single response. This removes the lower limit on the choice of intermediate frequency. Even an intermediate frequency of zero is possible.

This last statement may be a surprise. IF amplifiers at very low frequencies are not common in the analogue world because of problems with voltage drift and low-frequency noise, but these defects don't affect digital circuits. There is one practical difference between a conventional IF and the zero-frequency version: the two mixer outputs are not combined to form a single-response output, but they are kept separate and processed in parallel as two paths. This makes the demodulation process much easier and is sometimes known as the I/Q technique, where I and Q refer to Inphase and Quadrature, the names usually given to the two paths in such a process.

It's worth noting that in an I/Q path carrying a signal with a spectrum of width B, the signals in the I and Q paths each have a width of B/2. One could say that the spectrum of an I/Q signal extends either side of zero, in the same sense that the spectrum of a conventional signal extends either side of its centre. However, it's not true to say that the I channel represents one 'side' of the signal and the Q channel represents the other.

HARDWARE BLOCK DIAGRAM. At the top left of **Figure 1**, the receiver line-up starts with a transformer to match the 50Ω input impedance to 125Ω, the optimum figure for the low noise amplifier (LNA), an LT5514 device from Linear Technology. This is a 33dB linear amplifier preceded by a 16-step attenuator (1.5dB/step), which is controlled by four lines driven from the FPGA.

Another transformer couples the LNA to the balanced 30MHz low-pass filter, which is a 9th order Chebyshev design that aims to keep the alias response of the ADC below -100dB.

The ADC, an LTC2254 device from Linear Technology, has 14 data bits and an extra 'overload' bit. Fourteen bits can only handle 84dB, which might not seem enough to capture the weaker signals, but in fact the total input to the ADC (in a 30MHz bandwidth) is always much larger that the smallest digitisation step. The 14bit limitation merely shows up in the digital data as noise

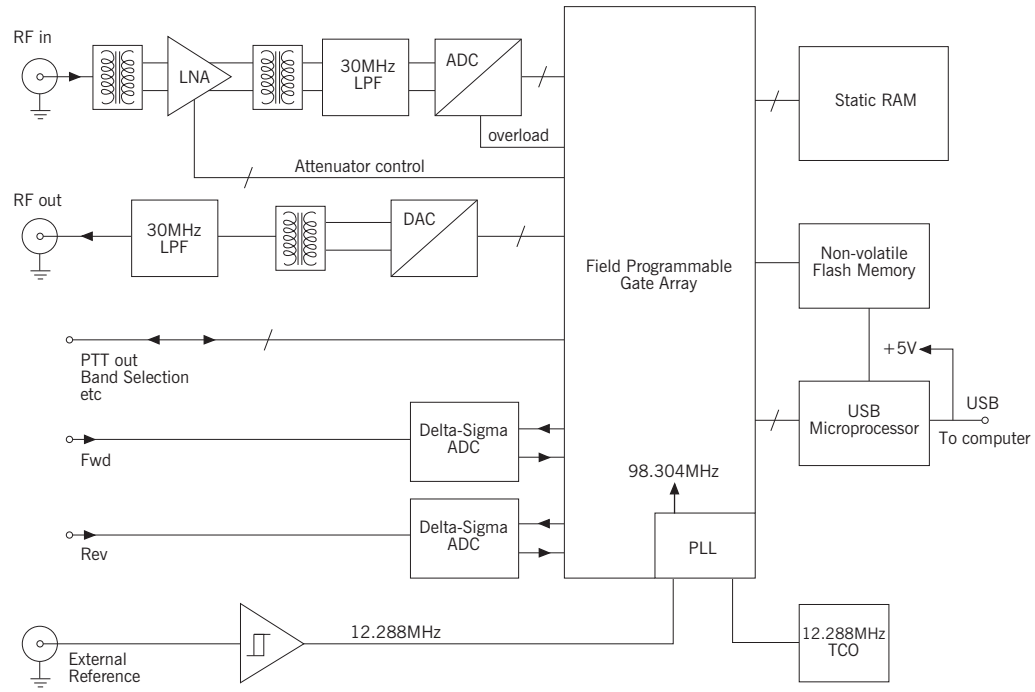


FIGURE 1: All Digital Transceiver hardware block diagram.

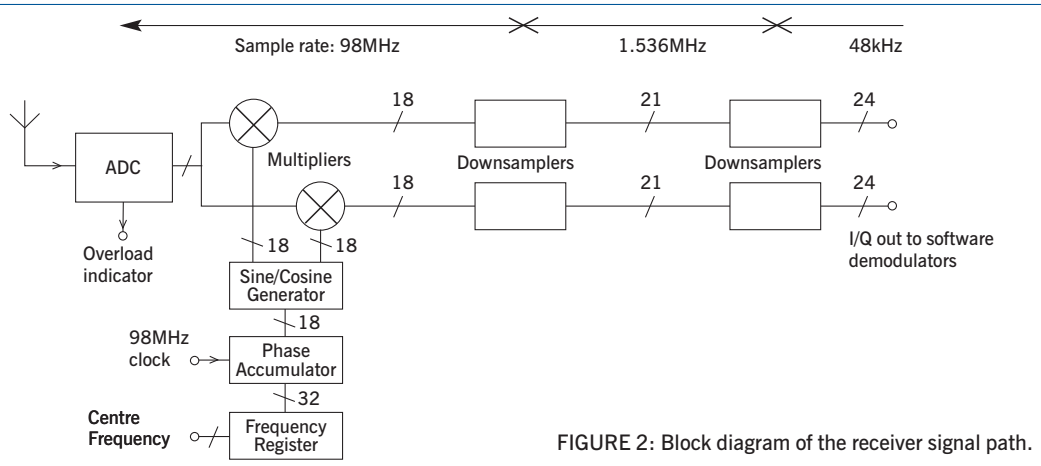


FIGURE 2: Block diagram of the receiver signal path.

spread uniformly across the whole 30MHz band. Noise 84dB down in a bandwidth of 30MHz equates to noise 124dB down in a bandwidth of 3kHz. This means we can copy narrow-band signals way below the smallest digitisation step. 14 bits is plenty.

On the transmit side, a 14-bit 210MHz DAC, an AD9744 from Analog Devices is followed by a step-down transformer and a single-ended 9th order Chebyshev filter, to deliver up to 10mW output into 50Ω.

The FPGA, an Altera Cyclone II type EP2C8, contains 8256 general-purpose logic cells and 18 multiplier cells, as well as about 160k bits of memory. Some 1600 of these cells make up a 32-bit microprocessor which handles many of the housekeeping functions. A separate static RAM chip stores program and data for this processor. Two phase-locked loops generate the internal clock signals, driven from either an on-board TCXO or an external reference. The logic configuration is loaded from a non-volatile FLASH memory at power-up. In addition to interfaces for the ADC and the DAC, there are a number of general-purpose I/O pins for bandswitching, PTT and other things, some of that we haven't thought of yet. The forward and reverse power sensors use an input and an output pin, together with a few resistors and capacitors, to implement a delta-sigm a converter.

The USB interface is handled by an Atmel AT89C5131A, which is a 24MHz microprocessor with on-chip non-volatile program storage. An 8-bit bi-directional bus communicates with the FPGA. This device can be programmed over the USB from a computer and also interfaces to the FLASH memory so that the FPGA configuration can also be uploaded from a computer.

The printed circuit is a four-layer construction with one continuous ground plane.

THE DIGITAL RECEIVE PATH. In **Figure 2**, the analogue to digital converter (clocked at 98.304MHz) is preceded by a 30MHz low-pass-filter (not shown). This ensures that the receive will hear everything up to 30MHz but not respond to anything in the range 68.304 – 98.304MHz, which would 'alias' down to the wanted signals below 30MHz and cause QRM. The digital output of the ADC is taken to two multipliers in the FPGA, the other inputs of which come respectively from a digital sine wave and a digital cosine wave.

THE SINE-COSINE OSCILLATOR BLOCK.

This is probably the most important block in the receiver **[1]** because any defects in the oscillator signals will give rise to unwanted responses in the complete receiver. This block works very like a direct digital synthesiser (DDS), but without the digital-to-analogue conversion. On each cycle of the 98.304MHz clock, a 9-bit frequency register (FR) is added to a 9-bit phase accumulator (PA), which addresses a 512-step lookup table containing a full-cycle sinewave. To generate a frequency of 192kHz (98.304/512MHz), for example, the FR is set to 000000001 and the lookup table pointer will thus step by one position every cycle and trace out a 512-step sine waveform at 192kHz. The matching cosine waveform is formed by another pointer that is 90° 'further round' the table.

The FR is set to other values to generate other multiples of 192kHz. However, a tuning step size of 192kHz is far too coarse to be useful, so the FR and the PA are extended by another 9 bits, to 18 bits. However, only the top 9 bits of the PA are used to address the lookup table. The effect is a tuning step size of 375Hz. To generate a frequency of 192.375 kHz, a '1' is set in the 9th bit and a '1' in the 18th bit of the FR. This results in an extra step in the lookup table pointer every 512 cycles. The resulting output has a frequency of 192.375kHz, but the repeating extra step shows up as a pair of spurious sidebands 375Hz either side, at about 54dB down.

These sidebands are reduced further by using the additional 9 bits in the PA to interpolate smoothly between the lookup table steps. Calculus theory shows that the

difference between two adjacent sine values is proportional to the corresponding cosine value and this makes it easy to calculate the required adjustment. The cosine value, as read from the lookup table, is first multiplied by a constant and then by the interpolation fraction, that is, by the second set of 9 PA bits. This is added to the looked-up sine value to give the interpolated value. The interpolated cosine is done in a similar way. To economise on the number of multiplier blocks needed (there are a limited number of these in the FPGA), a second lookup table is used to store all 512 multiples of the constant. This interpolation technique takes the worst-case spurii down to -108dB. This figure sets the ultimate unwanted response level of the entire receiver.

The tuning steps can be made much smaller than 375Hz by further extending the FR and the PA (eg to 32 bits). This introduces more spurii, but none are worse than -108dB.

THE DOWNSAMPLER BLOCK. The outputs of the two multipliers go to the two-path 'zero-IF strip' at a sample rate of 98.304MHz, but this rate is far too high for comfort. The bandwidth of the signals of interest are only a few kHz and the bandwidth needed in the I and Q channels is half that of the signal itself, so the sample rate required by the demodulation software is itself only a few kHz. For this project we chose to feed the data down the USB cable to the computer at 48kHz. This is enough to handle broadcast quality signals and even a modest bandscope. The reason for the odd choice of 98.304MHz for the RF sample rate now becomes apparent – to get from here to 48kHz means reduction by a factor of 2048, which is a 'nice' number in digital terms.

Nyquist says that before the sample rate can be reduced to 48kHz, all traces of the input spectrum above half this frequency must be removed. If this is not done, unwanted signals around multiples of 48kHz will be 'aliased' down around zero and will interfere with the wanted signal. The downsampler must therefore incorporate a lowpass filter.

The simplest form of digital lowpass filter forms the average of a run of N input samples. If N is chosen to be 2048, the frequency response looks like **Figure 3**. It's not a brilliant filter but note that the notches between the sidelobes are at precise multiples of 48kHz and this is exactly what is needed to reject the alias products that also occur at multiples of 48kHz. The notches are narrow, however, so although an unwanted signal exactly 48kHz away would be perfectly notched-out and wouldn't interfere with a wanted signal on the centre frequency, an unwanted signal 47 or 49kHz away would not be completely rejected. Care must therefore be taken to ensure that this doesn't become a problem at the edges of the wanted passband.

The running-average process is very easy to do in an FPGA. Input samples are summed continuously in a digital integrator. At intervals of 2048 samples a copy of the integrator output is saved. The difference between consecutive saved copies is the desired output. If one filter doesn't give enough alias rejection at the passband edges, two or more can be cascaded. This is also easy since it is possible to combine the integrators into one block running at the input rate and the differentiators into another block running at the output rate; the end result is the same. This technique is known as a Cascaded Integrator Comb (CIC) filter [2] (the differentiators behave like comb filters). Another technique, which helps to minimise the number of logic gates needed to achieve a given performance, involves reducing the sample rate in several stages. For this project we chose to downsample from 98.304MHz to 1.536MHz with 3 CICs, then down to 48kHz with 7 CICs. **Figure 4** shows the passband shape and the first alias response (the worst one), plotted against frequency. This graph was generated with a computer simulation of the filter itself (the 'noise' at about -144dB is the resolution of the 24-bit arithmetic). We decided that ±5kHz was the widest passband we would need in the complete receiver, so we are not concerned about the alias response levels further out and -130dB at 5kHz is acceptable. The wanted response ends up with a slight droop, about 1dB at 5kHz, but that can be dealt with in the computer.

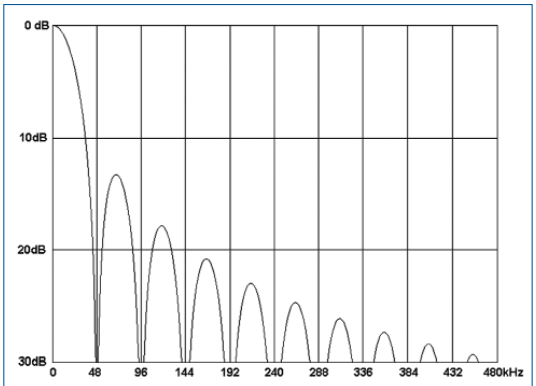


FIGURE 3: Frequency response of a run-of-2048 filter clocked at 98.304 MHz.

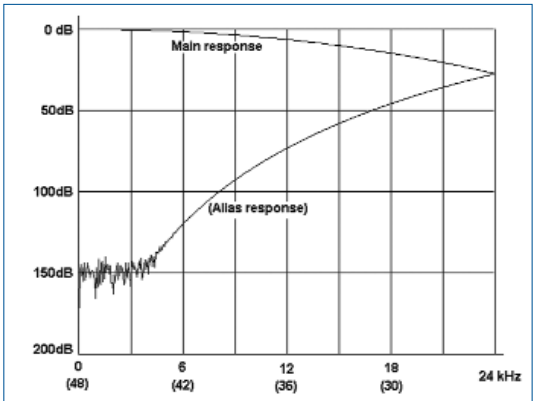


FIGURE 4: Overall response of the receive downsample filter showing the first alias response folded back to 48kHz.

The two 48kHz output streams are fed down the USB cable to the computer, where the software defines the finished passband precisely and carries out the desired demodulation.

THE DIGITAL TRANSMIT PATH. The transmit side (**Figure 5**) does not take so long to describe. It is most easily explained backwards since the process mirrors that of the receiver. A digital to analogue converter (DAC) generates the RF output at a few mW and this goes direct to the output connector via a 30MHz lowpass filter. The input to the DAC is derived from the sum of two multipliers exactly like those on the receive side, each one multiplying one IF signal by one of the outputs of a sine/cosine block exactly like the one in the receiver. The discussion on receiver spurious responses applies equally to the transmitter – just replace the word 'response' by 'emission' wherever it appears.

The need to downsample the receive path is mirrored by the need to upsample the transmit signal. Upsampling is done with exactly the same type of CIC filter that was used in the receiver, but with the integrate and difference processes transposed. Connected this way round, these devices can be thought of as interpolators rather than filters. Once again, the discussion on the alias responses in the receive downsampler also applies to the transmit upsampler in respect of spurious emissions. The slight droop in the passband is also present in the transmitter, but rather than pre-compensate for this in the computer software, we chose to do it in the FPGA with a 3-tap delay-line (FIR) filter.

There is another difference between the receive and transmit paths in our design. The device we chose for the USB interface can only handle one 48kHz I/Q stream so we dropped the transmit sample rate down to 16kHz. This means that the line-up of the transmit upsampler is not quite the same as that of the receive downsampler.

To complete the digital transmit function, we have placed a gain-control element (another multiplier) between the output mixers and the DAC. This performs the essential transmitter ALC function. We chose to put it here rather than in the computer software so that the drive to the PA can be reduced quickly in the event of an overload, regardless of how slow is the loop back via

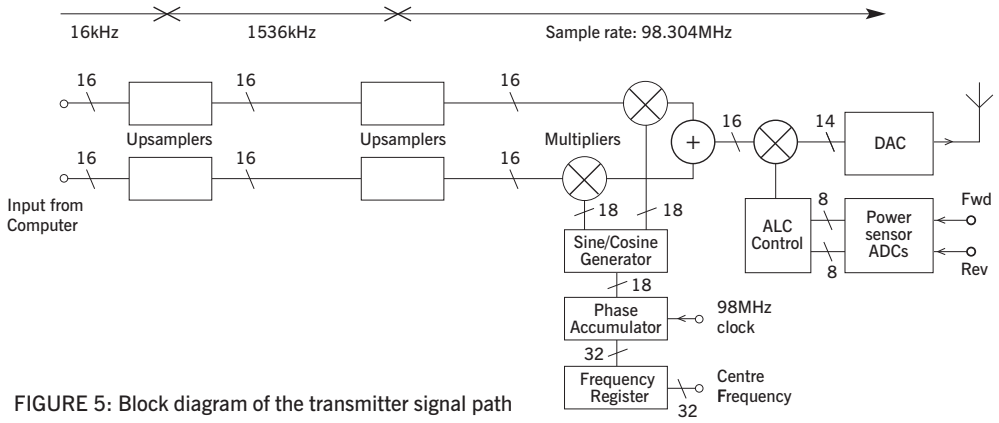


FIGURE 5: Block diagram of the transmitter signal path

The flow of receive and transmit data between the ADT board and the computer is very similar to that of audio data between a computer and a semi-professional soundcard, but we have run into problems using the soundcard application interface in the Windows operating system. Nevertheless, we have the transceiver operating on the air at the time of writing (September 2008). By the time the second article appears we should have overcome these problems, so it should then be possible to drive the ADT board using Software Defined Radio programs of the kind that are already available for SDR designs using analogue RF and soundcards. We don't believe that the ADT board is suitable for home construction by the average amateur, but, in due course, we hope to be able to give details of the availability of assembled boards, should there be a demand for them.

the computer. To complete the ALC function, there are two auxiliary low-spec ADCs on the card, implemented in the FPGA. These take DC voltages in the range 0-3.3V derived from conventional forward and reverse power sensors in the transmitter PA.

Note that the transmitter and receiver can operate simultaneously. This leaves open the interesting possibility to develop techniques for improving the transmitter PA linearity.

FIRMWARE. The FPGA logic was written in the Verilog language using the Quartus II design software. The code for the USB processor was written in the C language and that for the 32-bit processor in C++. All this represents about 800 hours work, carried out by Steve, by far the largest part of the total workload in the project so far.

PERFORMANCE MEASUREMENTS. At the time of writing only the receiver has been measured. We should be should be to publish the transmitter measurements at a later date.

All measurements (except noise figure) were made at 14MHz with the LNA attenuator at zero.

Receiver sensitivity (3kHz bandwidth)
-116dB (0.35μV) for 10dB S/N

Noise figure
1.8MHz 13dB
14.1MHz 12.9dB
29.5MHz: 17.2dB

Phase noise
@5kHz -127dBc/Hz
@50kHz -132dBc/Hz
@500kHz-127dBc/Hz

The rise in phase noise at 500kHz is unusual, but not in itself a poor figure.

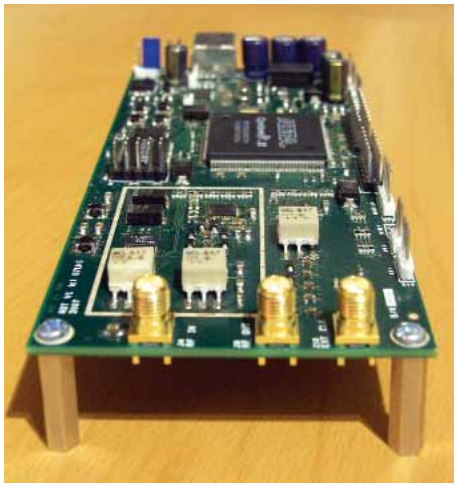
Third-order intercept point, usually quoted for analogue receivers, is of dubious value in the context of digital receivers, since the unwanted products in a digital receiver do not obey a third-order law [3]. However, everyone asks us what it is, so we had to do the measurement. It was +31dBm at all tone spacings and is almost certainly set by the LNA. The second-order intercept point measured +60dBm.

With the LNA at maximum gain the ADC reaches full-scale with a single-tone input of -20dBm.

FINISHING THE JOB. To make a complete transmitter/receiver using the All-Digital Transceiver (ADT) card as developed so far, a computer with a USB interface is required. As well as the receive and transmit data paths, there are some auxiliary control functions that share the USB, for example the programming of the FPGA is done this way as mentioned earlier. Setting the frequency registers in the sine/cosine oscillators is done via the USB, as is the control of the receiver input gain/attenuation. Up to now we have used the receiver without sub-octave bandpass filters and not encountered any overload problems, but there are spare pins on the board to switch such filters. They will be needed in any case for switching transmitter low-pass filters. In the other direction, we can foresee that it will be necessary to signal such things as the receiver ADC over-drive indication and the ALC sensor readings.

In a subsequent article, we plan to describe how the computer software processes the receive data, first to define the channel width to match the signal being received and then to demodulate it. The SSB demodulation process will be described of course, but other modes will be covered for completeness, since two-path zero-IF equivalents of traditional demodulation techniques will be new to many readers. We will show, for example, how to perform the apparently impossible task of demodulating FM from a carrier whose centre frequency is below the audio band! The matching transmitter modulation functions will also be described, together with some of the other features normally found in modern transceivers, such as noise blankers, notch filters and speech processors. This article may also describe further developments to the ADT board hardware and firmware.

ACKNOWLEDGEMENTS. We are grateful to Colin Horrabin, G3SBI, Dave Roberts, G8KBB and George Fare, G3OGQ, who did the performance measurements, to Anne Carrington at Linear Technology for help with the LT5514 and to James Ross and David Evans of EBV for help with the FPGA.



SMA connectors are used for the RF In, RF Out and external clock. Prototype boards were hand-soldered, a difficult task for the average home constructor.

REFERENCES

[1] The sin/cos idea came from work by Pawel Jalocha, SP9VRC.
[2] Hogenauer, E. "An economical class of digital filters for decimation and interpolation," IEEE Trans. Acoust. Speech and Signal Proc., Vol. 29 No 2, pp. 155-162, April 1981.
[3] Leif Asbrink, SM5BSZ. "IMD in Digital Receivers," QEX Nov/Dec 2006, pp 18 – 22.

WEBSEARCH
www.sm5bsz.com/dynrange/qex/digital-imd.pdf