MAP65 Version 2

A Panoramic, Polarization-Matching Receiver for JT65

Joe Taylor, K1JT

Introduction and Background

MAP65 is a computer program designed for EME communication using the JT65 protocol. When used with RF hardware providing coherent signal channels for two orthogonal polarizations, the program offers automatic polarization-matched reception for every JT65 signal in a 90 kHz passband. Where linear polarization is in use, *MAP65* eliminates the effect of mismatched polarization angles on reception, so a station equipped with *MAP65* never experiences Faraday lockout or one-way propagation. On bands where circular polarization is the norm, *MAP65* is highly effective in its single-channel mode — again providing visual display of all signals in a 90 kHz window and decoding all the JT65 signals.

I started development of *MAP65* in late 2006, when my modest 2-meter EME station used the WSE converters designed by Leif Asbrink, SM5BSZ, together with his *Linrad* software and a pair of dual-polarization 10-element yagis. Audio output from *Linrad* was sent to the headphones for CW operation and also to the separate program *WSJT* when JT65 was in use. *Linrad's* panoramic display and highly effective signal processing made it easy to find and receive weak EME signals in either mode. The dual-channel WSE receiver and adaptive-polarization capability of *Linrad* meant that for a particular signal selected by mouse-clicking on the waterfall display, mismatched polarization angles never degraded my receiving capability.

However, it was often frustrating to see a dozen or more JT65 signals on the waterfall and be able to decode only those in a preselected window a few kHz wide. Moreover, the window had to be selected by the start of a one-minute JT65 transmission sequence and kept in place for nearly a full minute. It seemed to me that a golden opportunity was being lost. Wouldn't it be nice if a wideband version of a *WSJT*-like program were able to decode all JT65 signals in the 90 kHz *Linrad* passband, at the end of each minute? Better still, why not have the software do adaptive polarization matching for every signal, separately and independently, without operator intervention? These ideas were among the fundamental motivations for the design of *MAP65*. *Linrad's* networking feature could be used to send wideband, dual-polarization data to *MAP65* in digital form, and *MAP65* would find and optimally decode all the JT65 signals.

A working program with these capabilities first became usable at my station in May 2007, and a public release of *MAP65* was made two months later. Over the next year a few ambitious operators adopted *MAP65* and were impressed by its performance and capabilities. They quickly discovered that *MAP65* makes it easy to find and identify any JT65 signal in an EME sub-band, without the need for schedules, spots, or liaison through another medium. They found it exciting to have a whole JT65 sub-band (for example, 144.100 – 144.160 MHz) available on a computer display simultaneously, including the text of all messages being exchanged and a list of all transmitting callsigns, sorted by frequency. It was clear that these features would be especially desirable during EME contest weekends.

Unfortunately, a number of operators seeking the capabilities of *MAP65* found the learning curve for proficient *Linrad* usage rather too steep. Others did not have dual-

polarization antennas or two-channel receiving hardware, or they had acquired SDR-type receivers such as the SDR-IQ, from RFSpace, that could not support the 96000 Hz sample rate required by *MAP65*. To satisfy some of the resulting requests, in early 2009 I developed a stripped-down, single-polarization version called *MAP65-IQ*, designed explicitly to work with the SDR-IQ. Happily, it also works well with the Perseus receiver from MicroTelecom; both receivers can be configured to yield I/Q data at a 95238 Hz sample rate. These are single-channel receivers, so *MAP65-IQ* necessarily foregoes the adaptive-polarization capability of *MAP65*, but it retains all the other attractive features.

Packaged together with a pre-configured version of *Linrad*, *MAP65-IQ* greatly facilitated program setup and operation by new users. The original *MAP65* supported only JT65B, the sub-mode used for EME on 144 and 432 MHz. In contrast, *MAP65-IQ* supported all three JT65 modes. It thus provided an especially appealing system for 23-cm EME, where JT65C is used and where circular polarization obviates any need for a dual-polarization antenna or receiver.

Goals for MAP65 2.0

The foregoing summary traces the development of *MAP65* up to July 2011. At that time I was planning a major re-design and re-write of *MAP65*, building on what had already been learned. The programming framework used since 2005 for my programs *WSJT, MAP65*, and *WSPR* was beginning to feel cumbersome and outdated. I was hearing good things about a cross-platform development framework known as Qt that can produce native executables for Windows, OS X, and Linux from the same source code. My hope was that successful innovations developed for *MAP65* might later carry over to sister programs *WSJT* and *WSPR*. At worst, making a Qt-based *MAP65* seemed like a useful learning exercise; at best it might help me to address some perceived weaknesses in the original designs of all three programs and their packaging and distribution procedures.

I set out, therefore, to re-write *MAP65* using the C++ language and Qt for the graphical user interface. I planned for *MAP65 2.0* to support sub-modes JT65A, B, and C, as well as both single- and dual-polarization antennas and data sampled at either 96000 Hz or 95238 Hz. The new program would also offer real-time waterfall displays, which had been lacking in earlier versions of *MAP65*. Finally, it would include input routines that would let it be used with the popular and user-friendly program *SDR-Radio*, by HB9DRV, as well as with *Linrad*, and also in a stand-alone mode with input data acquired directly from a soundcard or FUNcube Dongle. Thus, *MAP65 2.0* would support all capabilities of the original *MAP65* and *MAP65-IQ* programs while offering many new features and greater flexibility. I expected that it would also make the software package for *MAP65* easier to maintain and support on multiple computer platforms.

Hardware and Front-End Software

MAP65 requires receiver hardware to convert an RF band to digitized I and Q (in-phase and quadrature) signals at baseband, sampled at either 96000 or 95238 Hz. Many possibilities exist for these tasks, with significantly different interfacing requirements. As specific examples, the SoftRock, IQ+, and WSE receivers can handle the down-conversions, with output signals fed to a computer soundcard for the analogue-to-digital conversions. The FUNcube Dongle is a tiny, cleverly designed system with wide frequency coverage (60 – 1700 MHz) and built-in silicon tuner and A/D converters. It plugs into a computer's USB port and presents itself to the operating system as a two-channel audio input device. In contrast, the SDR-IQ and Perseus receivers do their A/D conversions at RF and accomplish the down-conversion digitally, in a field-programmable gate array. Any of these systems, and many others with comparable

features, may be used with *MAP65*. Obviously, those designed for input frequencies in the HF range require another receive converter mixing down from, say, 432 to 28 MHz.

The SoftRock, FUNcube, SDR-IQ, and Perseus are all single-channel receivers, so they cannot support adaptive polarization. One could, however, configure a pair of modified SoftRocks (or approximate equivalents) for use in a dual-polarization system. They would need to be driven by a single local oscillator, to maintain coherence between the two channels. The IQ+ (by LinkRF) and WSE converters are high-performance, two-channel systems that can support adaptive polarization directly.

Some relevant details of receivers and interfacing requirements are summarized in Table 1. The SoftRock, FUNcube Dongle, and IQ+ use programmable synthesizers for their local oscillators. Their input frequencies can be programmed from within *Linrad*, *SDR-Radio*, or *MAP65* via a USB port. Centre frequencies for the SDR-IQ and Perseus receivers are set in the same way, although in these cases the local oscillator is a numerically controlled oscillator (NCO) implemented in software. To achieve very low phase noise and high spurious-free dynamic range, the WSE converters use a sequence of switchable crystal oscillators for frequency control. Their switching is accomplished from within *Linrad* through the computer's parallel port.

Table 1: Examples of receiving equipment and configurations for use with MAP65, with interfacing requirements

	SoftRock	**FUNcube Dongle**	**SDR-IQ, Perseus**	**SoftRock × 2**	**IQ+ V or U**	**WSE**
Input Freq (MHz)	28, 144	144, 432, 1296	28	28, 144	144 or 432	144
Polarizations	1	1	1	2	2	2
Soundcard Channels	2	–	–	4	4	4
Digital Interface	USB	USB	USB	USB	USB	Parallel Port
Frequency Control*	L, S, M	L, S, M	L, S	L, M	L, M	L
Front-End Software*	L or S optional	–	L or S required	L optional	L optional	L

** L = Linrad, S = SDR-Radio, M = MAP65*

The original *MAP65* program required *Linrad* as a data-acquisition front end, with A/D conversion and noise-blanking carried out there and subsequent processing in *MAP65*. Version 2 of *MAP65* can be configured in this way as well, and doing so provides some significant advantages. *Linrad*'s noise-blanking features are superb, and are applied to the data before forwarding it to *MAP65*. In addition, *Linrad* provides many flexible configuration options and other receiver amenities including mode-specific demodulation and audio output to headphones or speaker. In single-polarization systems, *SDR-Radio* can manage the data input and radio-control functions, again providing audio output and forwarding wideband digital data to *MAP65*. *SDR-Radio* is polished, commercial-quality software available free to hams, and some users may find it an easy way to get started with *MAP65.* Finally, *MAP65* can be used in stand-alone mode with I/Q signal inputs taken directly from a soundcard or a FUNcube Dongle.

MAP65 in Operation

Far more EME operators are familiar with *WSJT* than with *MAP65*, so I shall highlight principal features of *MAP65 v2* by focusing on operational differences between the two. *WSJT* is used with standard SSB radios, receiving a single polarization over a typical bandwidth of 2.5 kHz. When operating in JT65 mode, *WSJT* looks for a single signal whose synchronizing tone is within specified tolerance of a selected frequency about 1.3 kHz above the receiver's dial frequency. Without retuning the radio, the available search range is limited to the transceiver's IF bandwidth. Signals arriving with linear polarization angles significantly different from that of the receiving antenna will of course be attenuated. Mismatches of ± 45º cause a 3 dB loss, and losses rise steeply for larger offsets.

In contrast, *MAP65* works together with an SDR-style receiver that mixes RF signals to baseband, as indicated in the block diagram of Figure 1 (page $). Conversion from analogue signals to digital data takes place at the hardware-to-software boundary indicated by the horizontal dashed line. Available bandwidth is limited to somewhat less than the sampling rate used for A/D conversion, which is fixed at about 96 kHz. *MAP65* tries to locate and decode all JT65 signals in this full passband, or a specified subset. When used with a two-channel receiver and dual-polarization antenna, the program also searches over all polarization angles for the best match to each detected JT65 signal. Averaged over all possible arrival angles for pure linear polarization, the adaptive-polarization capability of *MAP65* yields a 3 dB improvement in receiving performance over a single-polarization system.

On a typical weekend at a moderate-size 144 MHz EME station, *MAP65* decodes a dozen or more JT65 signals at the end of each UTC minute. The signals are easily visible on the waterfall, including a zoomed view covering several kHz and displayed in the lower half of the Wide Graph window (Figure 2, page $). Callsigns of all transmitting stations appear in the Band Map, sorted by increasing frequency, and full message traffic appears in the Messages window. Messages decoded close to the operator's selected QSO frequency appear in the Main Window. A screen shot typical of *MAP65* in normal operation is shown in Figure 2, and expanded views of the Band Map and Messages windows in Figure 3. (Larger, full-colour versions of all screen shots are on the Conference DVD.) Colour codes for the text in these windows indicate relative elapsed times since decoding. By default, entries disappear altogether after 20 minutes.

Many more operational details of *MAP65* are described in the *MAP65 User's Guide*, now under preparation. A snapshot copy of the *User's Guide* is on the Conference DVD, and the most recent version can be found at
http://physics.princeton.edu/pulsar/K1JT/MAP65_Users_Guide.pdf

Transmitting and Station Control

MAP65 handles the functions of message encoding, sequence timing, and T/R switching in essentially the same way as *WSJT*. Transmitted messages are encoded and converted to audio waveforms according to the JT65 protocol. The waveforms are sent to a soundcard output channel and then onward to the SSB transceiver's audio input. A push-to-talk (PTT) signal is generated through a serial port or USB-to-serial converter. PTT is asserted approximately 0.2 s before Tx audio starts and cleared 0.2 s after it ends, so a basic T/R sequencer is built into the software. You would be wise to implement some foolproof hardware sequencing as well, to protect your antenna-mounted preamps.

MAP65 includes some basic features for program control of transmitter frequency and antenna pointing. As of June 2012, *MAP65* accepts operator commands for setting

transmitter frequency and computes antenna coordinates for tracking the Sun and Moon. Tracking data are written to a disk file named azel.dat and updated once per second. Details of the software and hardware required for actual control of a transceiver and antenna positioner are left to the user. The K1JT station uses a simple program written in Python: it reads the file azel.dat and sends data to the transceiver and rotors through serial ports. Other *MAP65* users have devised their own equivalent solutions.

MAP65 Algorithms and Program Structure

Many users are curious about the internal workings of *MAP65*, so a brief section on program structure and design philosophy seems in order. Inevitably this description will involve some programmer's jargon; I trust that most readers will be able to work out the meanings of less familiar terms from their context.

Like most graphical user interfaces (GUIs), the one in *MAP65* is 'event driven'. With modern computers the program's average CPU usage is well under 10%, so most of the time *MAP65* is idle, waiting for a signal from the operating system to indicate that an event such as a key press, mouse click, or timer completion has taken place. When such a signal is received the information is forwarded to a service routine that carries out the desired action — for example, adding a character to the selected entry field, highlighting the item under the mouse cursor, executing a command such as Stop Tx, updating the UTC display, or whatever. If the execution time of the service routine is expected to be long enough to cause a delay perceptible to humans, the routine is executed in the background — or in programmer-speak, in another 'thread'. The GUI therefore remains active and responsive, even when a slow operation such as File → Open or Decode is underway.

The five principal display windows of *MAP65* are all part of the GUI and controlled by its code. The Wide Graph window updates as new data are received, at a rate determined by the N Avg control. Numbers in the Astronomical Data window and the UTC display in the Main Window update one per second, controlled by a timer. The same timer controls the T/R sequencing, with resolution 0.1 s. The Band Map and Messages windows update whenever the JT65 decoder produces new information.

Baseband data are acquired in the background, using code that runs continuously in a separate thread. The operating system is instructed to give this thread high priority, so that samples or packets will not be accidentally dropped because the computer was busy doing something else at a critical time. If *Linrad* or *SDR-Radio* is serving as a data-acquisition front end, the *MAP65* input routine accepts UDP data packets from the other program using standard network interface routines. When *MAP65* handles raw data acquisition by itself, it opens a sound card (or a FUNcube Dongle) and uses code similar to that required for other applications doing direct audio input. Whatever the source of input data, when a specified quantity equal to half a JT65 symbol length has arrived, its spectrum is computed by Fourier transformation. Noise blanking is also done at this time, if not already accomplished in *Linrad*. The computed spectra at half-symbol steps are used to update the waterfalls, and are saved for later use by the decoder.

The biggest computing task in *MAP65* is the JT65 decoder. It is implemented as a separate process — an independent, separately executable program called *m65*, normally activated from within *MAP65* at t = 52 s of a receiving minute. The decoder finds its raw data and the pre-computed, overlapped, half-symbol spectra in a shared memory region where they were stored by the GUI thread. A quick decode is carried out at frequencies near the pre-selected QSO frequency, so the operator can see messages directed to himself (and possibly some other nearby traffic) within a second or two.

The decoder then goes on to analyse the entire displayed bandwidth, up to 90 kHz in all. It searches first for the special shorthand messages RO, RRR, and 73, and for signals conforming to the pseudo-random pattern of JT65 synchronizing tones. In a dual-polarization system this search is carried out using 0º, 45º, 90º, and 135º as initial guesses for the signal's linear polarization angle. When a plausible sync-tone candidate is found, the program does a least-squares search for optimized values of time and frequency offsets DT and DF, frequency drift rate, and polarization angle. It then computes a 64-bin spectrum for each of the 63 information-carrying channel symbols of the JT65 Reed-Solomon codeword. From these it calculates the most likely and second most likely values for each symbol, and their estimated probabilities of being correct. These 4 × 63 = 252 numbers are submitted to the Koetter-Vardy algebraic soft-decision decoder, which returns either the decoded message or a flag indicating "no result". If requested, a final decoding step carries out a limited matched-filter search on the array of 64 × 63 numbers representing the information-carrying symbol spectra. The search candidates are model spectra that correspond to messages starting with CQ or the home station callsign, followed by a callsign and Maidenhead grid locator selected from a database. If a figure-of-merit for the best-matched message exceeds that of the second best by a specified minimum, the message is displayed for evaluation by the operator.

Possible Future Developments

I frequently think about ways in which our various weak-signal digital protocols and the computer programs implementing them might be improved or superseded. I keep a list of ideas to be tried and tested, and it always seems to be growing! Many items in my list were inspired by suggestions received from other hams, and I know that still others have their own ideas about improving weak-signal communication efficiencies by making optimum choices of coding, modulation schemes, keying rates, bandwidth, and so on. I will not devote space here to reproducing my to-do list; some of its entries may be implemented eventually in *WSJT*, *MAP65*, *WSPR*, or some future program, but I know that many will never see the light of day. However, I shall discuss briefly two items that seem most closely related to this paper's main topics.

Message averaging: Since 2004 the JT65 decoder in *WSJT* has included a feature that allows it to average successive transmissions of a signal otherwise too weak to copy. This procedure allows decoding of arbitrary repeated transmissions down to about –28 or –29 dB signal-to-noise ratio (measured in the standard reference noise bandwidth, 2500 Hz), an advantage of some 4 dB over un-averaged transmissions.

Algorithm development for *MAP65* has always concentrated on the program's wideband capabilities, and message averaging is impractical in that context. A number of weak JT65 signals may be present in the passband at any time, a few dB below the decoding threshold, and of course it would be nice if they could be copied. However, each one will have its own distinct values of DT, DF, frequency drift rate, and polarization angle. Moreover, these signals will not generally contain the same message, sent repeatedly over successive T/R sequences — a pre-requisite for successful message averaging. Nevertheless, it might be practical to implement message averaging in *MAP65* by restricting it to the selected QSO frequency. I hope to experiment with this possibility in future months.

Fast mode for EME contests: Ben Franklin's advice to a young tradesman was "time is money". When reckoning weak-signal communication effectiveness, we might say "time is sensitivity". Slower transmissions mean a lower symbol rate or more redundancy, both of which can lead to successful decoding at lower signal levels.

On the other hand, we are sometimes interested in making QSOs at a faster rate — for example, in contests — and this requires faster transmissions. At least on the 2 m

band, activity levels during major EME contests are such that JT65 QSO rates are often limited by the protocol's one-minute T/R sequences. For well-equipped stations, many JT65 contest QSOs — perhaps even the majority? — take place at SNRs well above the decoding threshold. Would it therefore be desirable to introduce a mode with, say, 30 s T/R sequences and a decoding threshold around –21 dB, rather than –24 dB?

One such possibility was introduced as an experimental mode in WSJT 9.0 (September 2010). The mode is called Diana, the Roman name for the goddess of the Moon. Diana uses 30-second T/R sequences and differs from JT65 in many other ways. Modulation is 42-tone FSK with tones separated by 5.38 Hz, so the total bandwidth is about 226 Hz. Keying rate is twice as fast as in JT65, so that 126 channel symbols are transmitted in 23.4 seconds. Diana has no synchronizing tone; instead, sync information is conveyed by means of six 4 × 4 'Costas arrays' evenly spaced throughout the transmission. User information is sent character-by-character with neither compression nor forward error correction (FEC), but simply repeated as many times as will fit into a transmission. The decoder averages over these repetitions, thereby yielding sensitivity that is better for short messages than long ones.

On-the-air tests of Diana have been made on both EME and terrestrial paths. The mode works well, as illustrated by the following sequences of test messages exchanged between K1JT and VK7MO via 144 MHz EME on September 7, 2010:

UTC	Sync	dB	DT	DF	
202030	2	-20	2.5	242	K1JT VK7MO
202130	4	-19	2.5	240	K1JT VK7MO -19
202230	4	-20	2.5	240	K1JT VK7MO -16
202330	3	-22	2.5	240	RRR
202430	1	-25	-1.2	226	
202530	1	-24	2.5	237	SAME HERX
202630	1	-26	-0.7	213	
202730	1	-24	2.5	237	K1JT VK7MO
202830	3	-20	2.5	234	K1JT VK7MO
. . .					
205200	5	-19	2.6	-210	VK7MO K1JT
205300	3	-17	2.6	-210	VK7MO K1JT
205400	4	-21	2.6	-210	VK7MO K1JT
205500	5	-16	2.6	-210	VK7MO K1JT
205600	6	-20	2.5	-213	ENUF 4 2DAY
205700	5	-18	2.5	-213	GOOD 1ST TEST
205800	3	-19	2.5	-213	NOW QRT

Some of the differences between JT65 and Diana are easy to recognize. Diana messages have no defined structure: you can send anything you like. With no sync tone, the signals are much harder to find on a waterfall display than JT65. With no FEC, Diana's decoding accuracy degrades slowly between about –21 and –24 dB. In this range you may get partial copy with some incorrect characters. Of course, the same is true with many other popular modes including PSK31 and good-old-CW, but JT65 users are accustomed to and appreciate its all-or-nothing decoding characteristics.

A second possibility for rapid EME QSOs is JT65B2, introduced as an experimental mode in WSJT 9.1 r2433 (May 2011). This mode has all the characteristics of JT65B, except that its channel symbols are only half as long. Sensitivity is about 3 dB worse than JT65B, but with adequate signal levels QSOs take just half the time. The decoding

threshold is about –21 dB for the Koetter-Vardy decoder, and (as with the other JT65 sub-modes) several dB better for the deep search decoder). Here is an example of decodings from a JT65B2 QSO between F8ARR and TK5JJ on May 28, 2011:

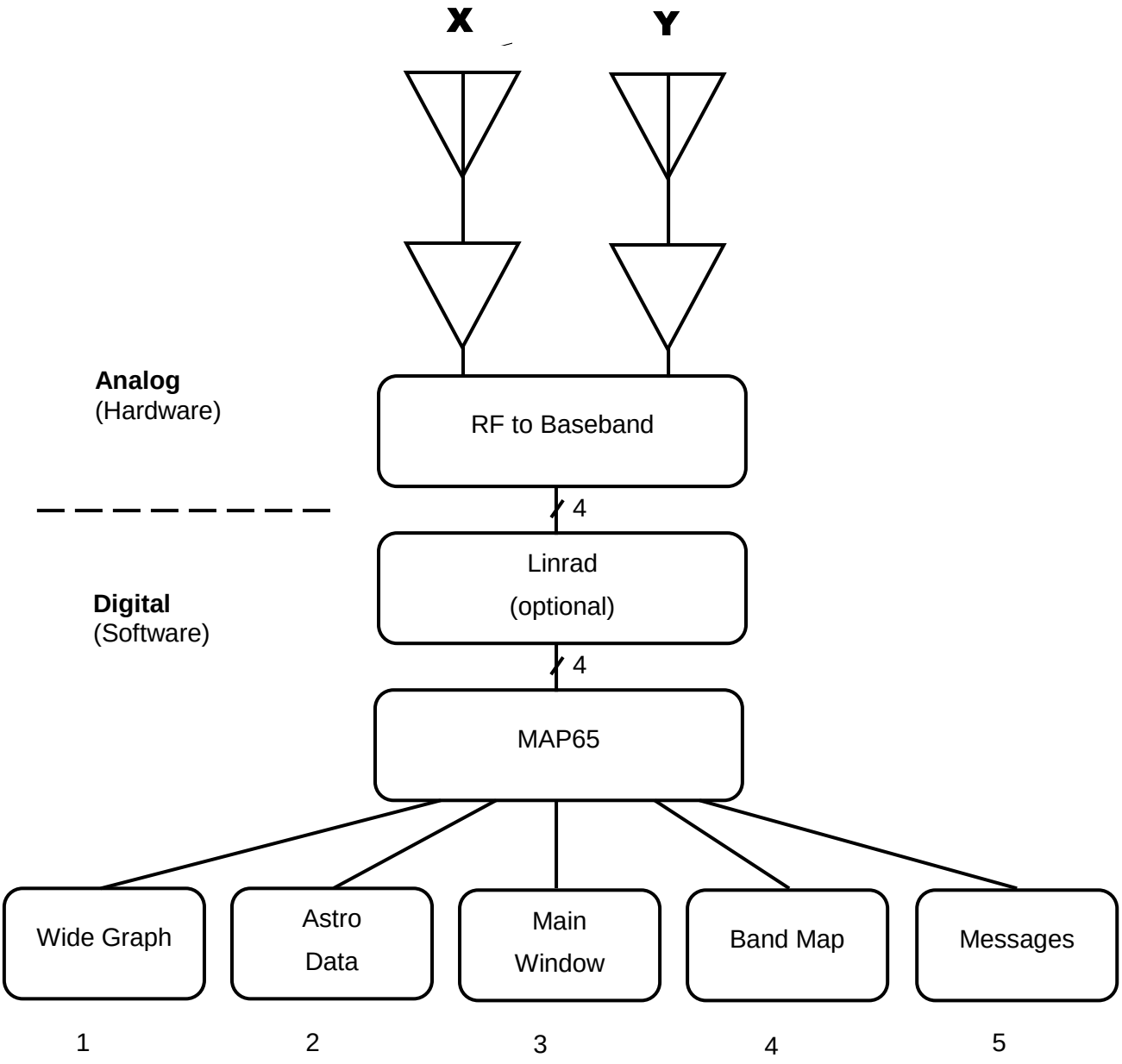
UTC	Sync	dB	DT	DF	W	
064030	1	-22	2.7	102	2 *	TK5JJ F8ARR IN94
064100	1	-20	2.4	151	2 #	F8ARR TK5JJ JN41 OOO
064130	5	-26		102	3	RO
064200	10	-20		151	3	RRR
064230	10	-23		100	3	73

On balance, I now tend to favour JT65B2 over Diana. However, before recommending its possible adoption as an alternative contesting mode for digital EME, I'd like to see some community discussion regarding its potential advantages and disadvantages. Of course it would be nice if the maximum possible QSO rate were doubled from around 10 to 20 per hour. But problems with local QRM might increase if two different T/R sequence lengths were in use. *WSJT* users would need to choose one mode, either JT65B or JT65B2, at any given time, and would find it inconvenient to switch rapidly between the two. *MAP65* does not yet have JT65B2 capability; this would need to be added. If JT65B were adopted as an alternative digital mode for EME contests, should it be used in a limited part of the digital sub-band? And so on… Perhaps some discussion of these matters can take place at the Conference.

Additional Reading and Useful Links

1. J. Taylor, K1JT, "The JT65 Communications Protocol." *QEX*, September-October 2005, pp. 3–12.; http://physics.princeton.edu/pulsar/K1JT/JT65.pdf.
2. J. Taylor, K1JT, "Earth-Moon-Earth (EME) Communication." *ARRL Handbook for Radio Communications*, Chapter 30 (2010); http://physics.princeton.edu/pulsar/K1JT/Hbk_2010_Ch30_EME.pdf.
3. R. Koetter and A. Vardy, "Algebraic Soft-Decision Decoding of Reed-Solomon Codes", IEEE Transactions on Information Theory, vol 49, pp 2809—2825 (2003); http://tesla.csl.uiuc.edu/~koetter/publications/01246007.pdf.
4. WSJT, MAP65, WSPR: http://physics.princeton.edu/pulsar/K1JT/.
5. Linrad, WSE: http://www.sm5bsz.com/linuxdsp/linrad.htm.
6. SDR-Radio: http://sdr-radio.com/.
7. SoftRock: http://kb9yig.com/.
8. FUNcube Dongle: http://www.funcubedongle.com.
9. SDR-IQ: http://www.rfspace.com/RFSPACE/Home.html.
10. Perseus: http://microtelecom.it/perseus/.
11. LinkRF: http://www.linkrf.ch/.

Fig. 1: Block diagram of a dual-polarization MAP65 system.



Notes

X and **Y** represent antennas with horizontal and vertical (or, say, 45º and 135º) polarization. (For a single-polarization system, delete the Y antenna and its subsequent signal path).

Preamplifiers should be mounted at the antenna, and suitable switching provided so that the antennas are available also for transmitting. All remaining equipment can be in the shack.

The ***/4*** labels on the signal paths entering the optional *Linrad* block and the *MAP65* block represent 4 data streams (I and Q for each of two polarizations).

Fig.2: Screen shot of MAP65 in operation, showing the five principal display windows.

(A larger colour version is on the Conference DVD.)

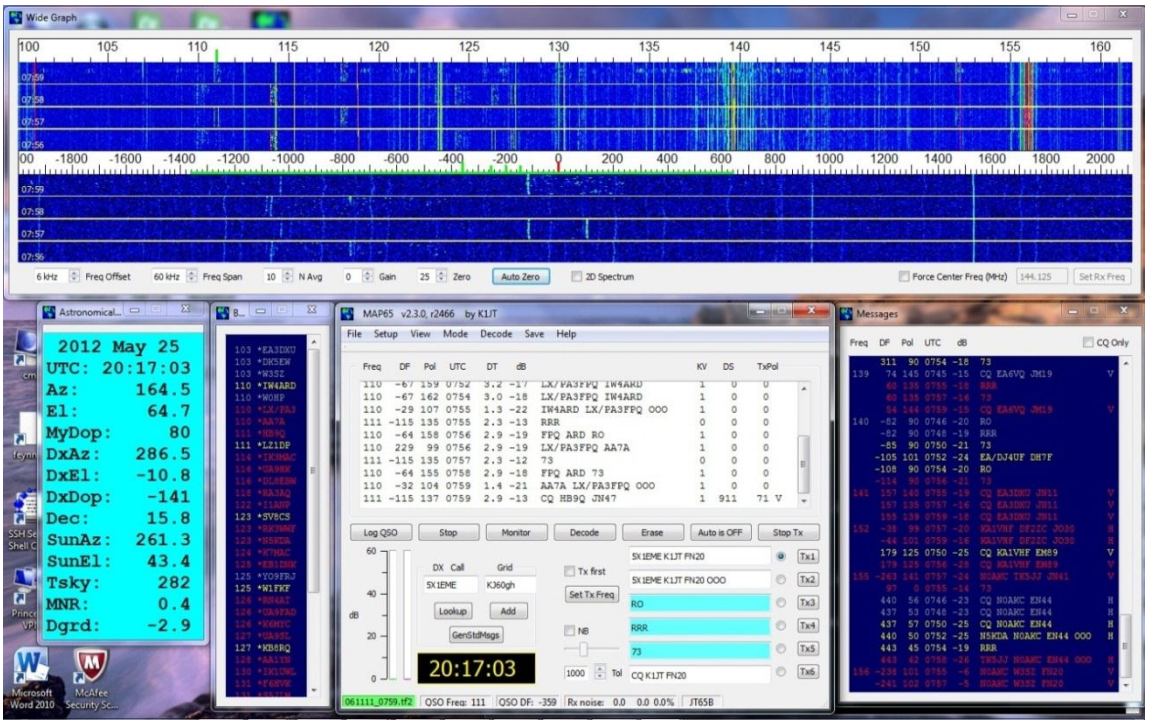


Fig.3: Example contents of Band Map and Messages windows after 11 minutes of reception using a pair of 3-wavelength cross-yagis on 2 m.

(A larger colour version is on the Conference DVD.)