MJB

A stand-alone PC-less MEPT_JT beacon controller

PRELIMINARY...

By Johan Bodin SM6LKM

Firmware Version 002
2007-04-05

Transmitter Control

MJB has two different outputs for TX control, VCXO and DDS. These two outputs are operating simultaneously. The KEY output (RA2), which goes high (+5V) during transmission, is intended for use in a VCXO exciter. The DDS is "unkeyed" by setting it to zero Hz output (DC...) when not transmitting.

VCXO

The two VCXO output pins deliver the symbol number to the VCXO hardware:

Symbol Number	FSK D1 (RA1)	FSK D0 (RA0)	FSK Shift
0	0	0	F + 0Hz
1	0	1	F + 1.46484375Hz
2	1	0	F + 2.9296875Hz
3	1	1	F + 4.39453125Hz

Keying the VCXO may cause undesirable drift so it is best left running even during RX periods. The actual keying can be done in another TX stage. If you wish to use random TX/RX with a VCXO exciter, you may need to "de-tune" the VCXO during RX in order to get it off the band of interest. Another approach is to run the VCXO on twice the TX frequency and use a flip-flop to divide the frequency. Apply the keying signal to some "enable pin" on the flip-flop or to a gate preceeding it. You can also mix the VCXO signal with that of a fixed XO and key the mixer. You will solve this problem easily with some solder smoke :-)

Make the coupling between the varicap diode and the crystal circuit as loose as possible and make sure that the peak RF voltage across the varicap never exceeds the DC bias voltage. A total shift of 5 or 6Hz with a control voltage of 3..5 volts should be fine. Multiturn potentiometers are preferred.

Adjusting the FSK shift may require some patience. Argo (www.weaksignals.com) is probably a good tool if set to a high resolution. A stable RX is needed too, of course. If you have a frequency counter with period measurement, you can use it to measure low frequency tones from your RX, say 100Hz or so. Use period averaging if available on the counter.

Try to avoid circuits where the FSK potentiometer settings interact... MJB has setup commands that let you key the TX and select symbols manually. According to Joe K1JT, an error of 0.1Hz or so should not degrade performance too much.

DDS

The DDS output is designed to control an AD9850. Please note that all four signals (/SDATA, /SCLK, /FQ_UD and /RESET) must be inverted before they reach the DDS chip. A 74HC14 (hex inverting schmitt trigger) will do the job. Connect unused inputs to GND.

The base frequency (symbol 0) is set by downloading a full 32-bit phase increment word to MJB and the phase increment offsets for the other three tones (symbols 1..3) are dowloaded as 16-bit unsigned values. You can safely ignore the DDS settings if you are using a VCXO only.

The base frequency phase increment (symbol 0, command P) is calculated as follows:

$$BASEPHASEINCREMENT = \frac{Fout_{symbol0} * 2^{32}}{Fclk_{dds}}$$

where both frequencies are expressed in Hz and 2^{32} = 4294967296

Assuming that no frequency division or multiplication takes place in the TX, the phase increment offsets for symbols 1..3 (commands Q, R, and S) are calculated as follows:

$$PHASEINCREMENTOFFSET_{SYMBOLN} = \frac{N * 12000 * 2^{32}}{8192 * Fclk_{dds}}$$

where N = Symbol number 1..3

Example:

Base frequency = 10.1401MHz (symbol 0)

DDS clock frequency = 120MHz

$$BASEPHASEINCREMENT = \frac{10.1401e6 * 2^{32}}{120e6} = 362928315.65141333...$$

$$PHASEINCREMENTOFFSET_{SYMBOL1} = \frac{1 * 12000 * 2^{32}}{8192 * 120e6} = 52.4288$$

$$PHASEINCREMENTOFFSET_{SYMBOL2} = \frac{2 * 12000 * 2^{32}}{8192 * Fclk_{dds}} = 104.8576$$

$$PHASEINCREMENTOFFSET_{SYMBOL3} = \frac{3 * 12000 * 2^{32}}{8192 * Fclk_{dds}} = 157.2864$$

These numbers are rounded to the nearest integer and converted to hexadecimal form for downloading to MJB. Add leading zeroes as needed to get 8 or 4 hex characters.

$362928315.65141333 \approx 362928316 = 0x15A1D8BC \rightarrow$ *Download* 15A1D8BC

$52.4288 \approx 52 = 0x34 \rightarrow$ *Download* 0034

$104.8576 \approx 105 = 0x69 \rightarrow$ *Download* 0069

$157.2864 \approx 157 = 0x9D \rightarrow$ *Download* 009D

Baud Rate Calibration

Without adjustment, a cheap microprocessor crystal may not be accurate enough to serve as a timebase for MEPT_JT symbol timing. Sufficient accuracy can be achieved in two different ways:

1. Careful selection or adjustment of the crystal loading capacitors (C1, C2)
2. Software calibration

Whichever method you choose, you will have to measure the frequency of the PIC's oscillator. The easiest way to check it is to use a well calibrated receiver. Just place a piece of wire near the PIC circuit and connect it to your receiver. As a QRSS enthusiast, you already know how to make accurate frequency measurements with Argo or similar software ;-)

If you prefer hardware calibration (1), adjust the oscillator as close as you can to 3.68640234375 MHz. Otherwise (2), just measure the frequency and calculate a software calibration value using the following formula:

$$CALVALUE = \frac{64 * Fxtal}{375} - 629146$$

where *Fxtal* is the PIC oscillator frequency in *Hz*

Round the value to the nearest integer and convert it to 16-bit hexadecimal form. Don't forget the sign (it is a two's complement signed number). If you use the calculator in Windows, set it to advanced mode and enter the number. Click the "Hex" and "Word" radio buttons to get the correct download format (four hex characters). This is the value to use with the W command (see below).

Download

MJB has two operating modes, command mode and beacon mode. The command mode can *only* be entered just after power-on reset and there is a timeout mechanism that prevents MJB from accidentally becoming stuck in command mode.

The communication format in command mode is plain ASCII, 9600 baud, 8 data bits, no parity and 1 stop bit. No flow control ("handshaking") is used.

After power-on reset, MJB will show all current settings on the terminal and the green LED will come on for five seconds. To enter command mode, you must enter ++++ (four consecutive '+' characters) within this five second window, otherwise MJB will switch to beacon mode automatically. Once in command mode, the red LED will come on and the timeout is changed to 60 or 256 seconds (the VCXO FSK setup commands L, M, N and O have 256 seconds timeout). The timeout is reloaded whenever a valid command character is received

A command consists of a single letter command character, often followed by a numerical value in hexadecimal form. In the download data, these hex' values must *not* be preceeded by any prefix such as 0x, $ or similar... However, the data sent from MJB *to* the terminal uses the 0x prefix on hex' numbers.

The letter case (upper/lower case) is unimportant. All whitespace such as space, tabs CR LF etc. is ignored. Whenever a valid command character, except X, is received, MJB will restart its command receiver and, if a numerical parameter is expected, wait for the first 1, 4, 8 or 162 digits. When all expected digits are received following a command character, MJB flags the corresponding setting as updated and starts waiting for a new command character. All updated settings will be programmed into EEPROM when the X command is received and then all settings will be shown on the terminal.

List of commands:

L	Select VCXO symbol 0 and key the TX (for manual tuning, 256s timeout)
M	Select VCXO symbol 1 and key the TX (for manual tuning, 256s timeout)
N	Select VCXO symbol 2 and key the TX (for manual tuning, 256s timeout)
O	Select VCXO symbol 3 and key the TX (for manual tuning, 256s timeout)
P	<DDS phase increment for symbol 0, 8 hex digits>
Q	<unsigned DDS phase increment offset for symbol 1, 4 hex digits>
R	<unsigned DDS phase increment offset for symbol 2, 4 hex digits>
S	<unsigned DDS phase increment offset for symbol 3, 4 hex digits>
T	<162 message symbol digits, 0..3>
U	<TX probability, 4 hex digits, 0000..FFFF => 0..100%>
V	<GPS baud rate code, 1 digit 0..4 => 1200..19200 baud
W	<MEPT_JT baud rate calibration constant, 16-bit signed, 4 hex digits>
X	Save all downloaded settings in EEPROM and start beacon mode

Note: If your PIC clock crystal happens to be right on the correct frequency, download 0000 with the W command.

The actual beacon message consists of 162 symbols. These are downloaded as 162 digits ranging from 0 to 3. This message symbol string can be generated by the command line version of WSPR:

http://physics.princeton.edu/pulsar/K1JT/WSPR.EXE

See instructions at the end of the file:
http://physics.princeton.edu/pulsar/K1JT/WSPR_Instructions.TXT

Example:

Generate a message and redirect the output to the text file msg.txt:

```
C:\WSPR> wspr Tx 0 0.0015 0 SM6LKM JO67 20 11 > msg.txt
```

Then open the file msg.txt and extract the 162 message digits from the second column.

The easiest way to program MJB is to prepare a simple text file containing all settings. Even the command mode unlock string ++++ can be embedded in this file. In this case, simply connect the serial line, power-up MJB and click "send text file" in your terminal program within 5 seconds.

It is possible to download data to MJB without using the RS-232 level converter. Simply connect the terminal's RS-232 TX line and GND to the GPS input on MJB. However, with a one-way connection, you will not be able to verify that the data sent to MJB was correctly received and saved.

Example download file:

---- cut here ----

```
You may place comments here, before the ++.. unlock string below.
  DDS clock = 120MHz
  P, Q, R, S: F_symbol0 = 10.14010000974... MHz
  T: Message = SM6LKM, JO67, 20dBm
  U: TX probability = 0x4000 out of 0xFFFF = 16384 out of 65536 = 25%
  V: GPS baud rate 3 = 9600 baud
  W: MEPT_JT baud rate calibration = 0x53 (cheap "rubber xtal" on 3.68689MHz)
  X: Save settings & go!

++++
P 15A1 D8BC
Q 0034
R 0069
S 009D

T
3102 2200 1002 3310 0012 0123 3330 0222 0232 2303
0200 2012 1102 3101 0001 1030 2001 3010 3030 1223
2230 3320 2110 1030 0032 0202 1023 0013 3231 0213
0102 2331 2020 0101 2013 2220 2201 3230 3100 2110
02

U 4000

V 3

W 0053

X
```

---- cut here ----

UTC Synchronization

MEPT_JT transmissions have to be synchronized to even UTC minutes within a second or so. MJB can use either a serial NMEA data stream from a GPS receiver or a user supplied sync' pulse connected to the PIC pin RB3.

MJB looks for the NMEA sentences $GPRMC and $GPGGA on the serial port and triggers a transmission whenever it finds a timestamp with an even UTC minute in any of these sentences. When MJB is waiting for GPS sync', the green LED will toggle every time a valid timestamp is received. If your GPS outputs both $GPRMC and $GPGGA sentences, the green LED will flash once each second, otherwise it will toggle once a second. When transmission is in progress, the red LED will toggle at each "baud tick".

The MEPT_JT protocol specifies that transmission shall begin 2 seconds after the top of the minute but with most GPS receivers, there is a slight uncertainty due to serial transmission delays. When testing with a Garmin GPS-12, the actual start of transmission seemed to jitter about 1 second in a slow sawtooth manner. When using GPS sync', MJB starts transmission about 1.14 seconds after detecting top of the minute.

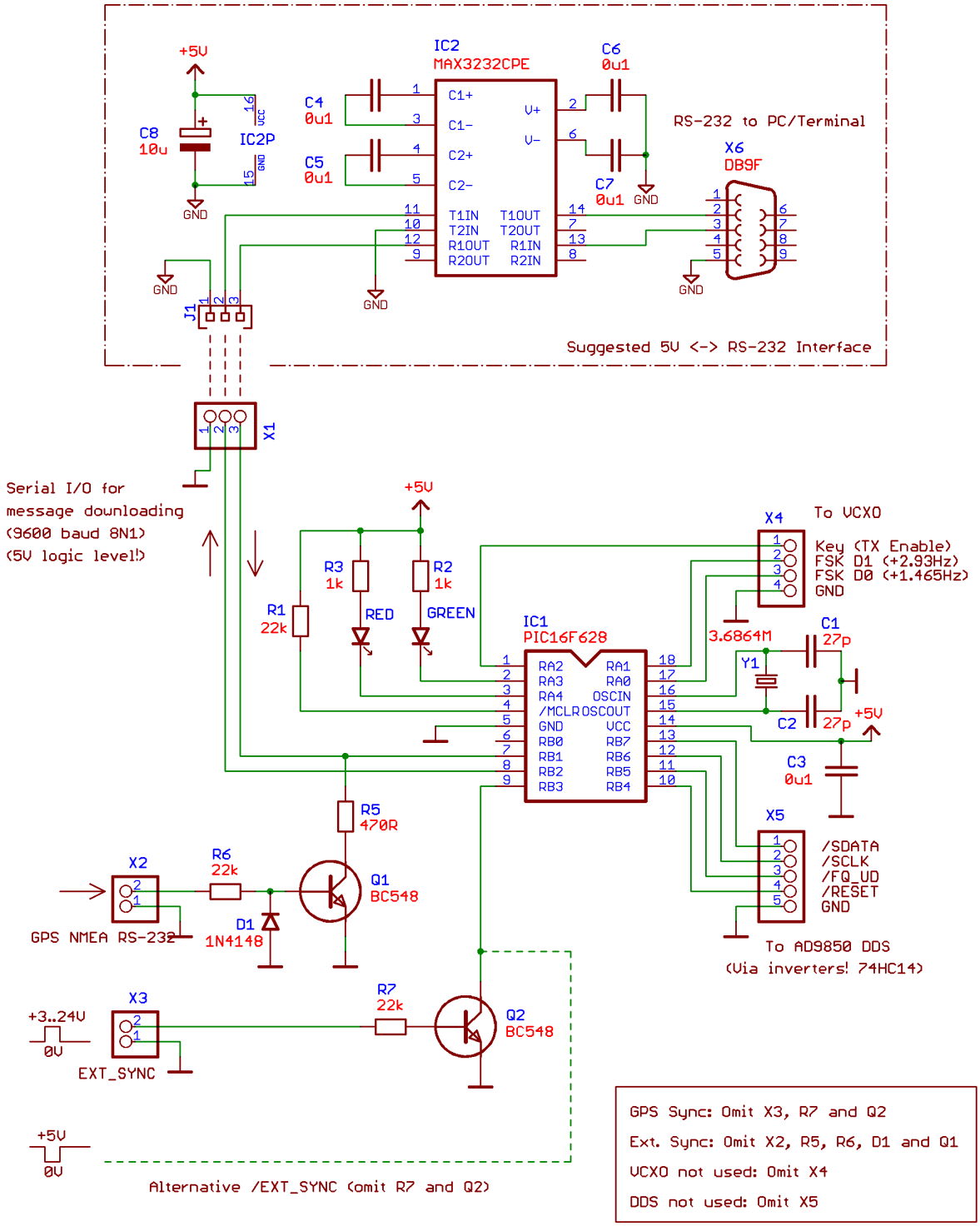
As an alternative to GPS sync, a falling edge on PIC RB3 can be used to trigger the transmission of a frame. In this case, the transmission start delay is 2 seconds so the falling edge should occur exactly on the top of the even minute.

Both the sync' input pin RB3 and the serial input pin RB1 have weak internal pull-up resistors enabled in the PIC and can be left unconnected if not in use.

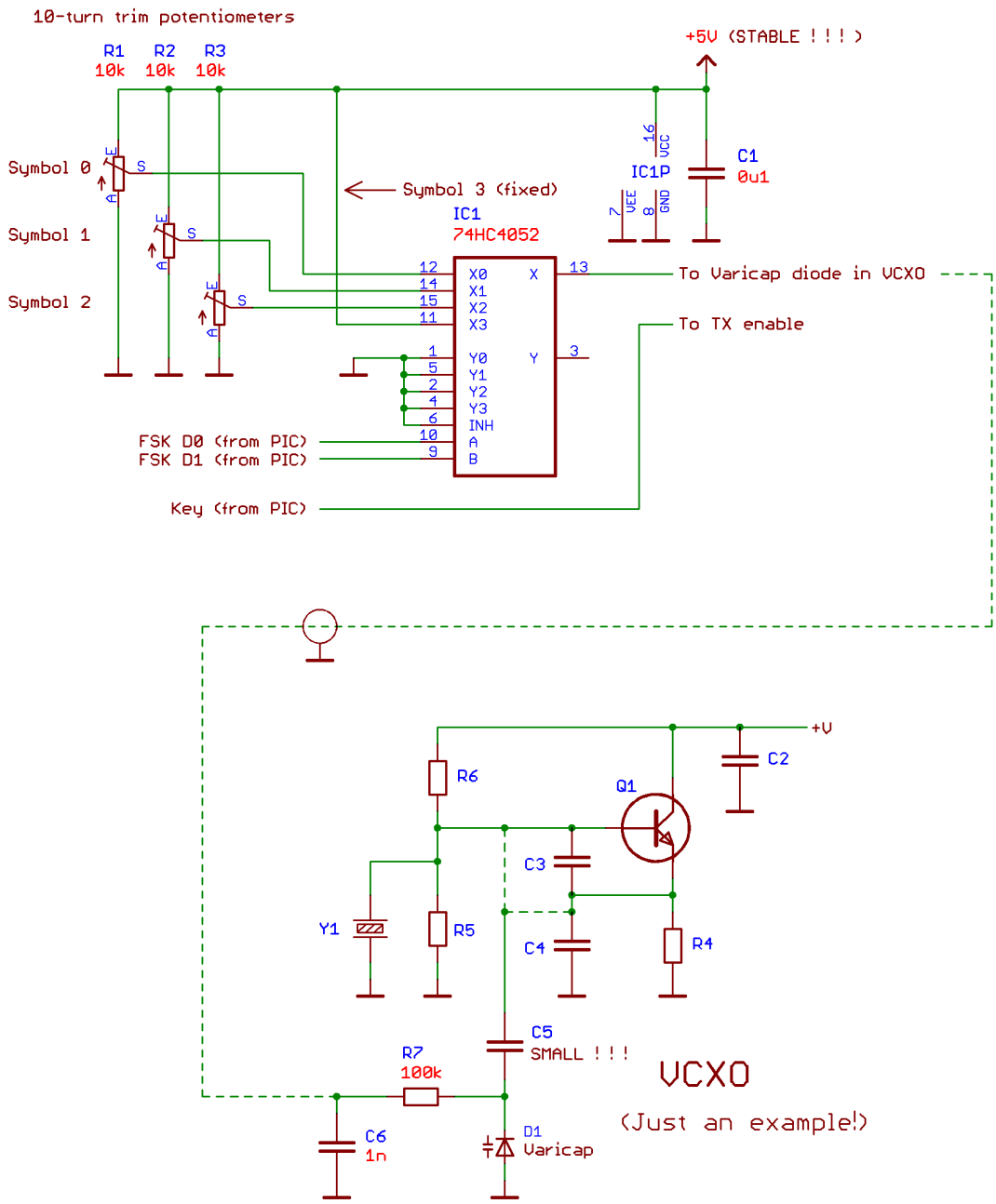
Files

mjb002ug.pdf	This file
mjb002.hex	Compiled program, ready to burn into a PIC16F628
mjb002.c	Source code for BKND CC5X compiler, v3.3 Extended
mjb002.asm	Assembly language output from the compiler (if you are curious)

73 de Johan SM6LKM



SM6LKM MEPT_JT Beacon Controller	
TITLE: mept_jt	
Document Number:	REV: 1.1
Date: 2008-04-03 00:45:30	Sheet: 1/1



SM6LKM MEPT_JT 4-FSK VCXO Control		
TITLE: vcxoctrl		
Document Number:		REV: 1.1
Date: 2008-04-03 01:25:36	Sheet: 1/1	