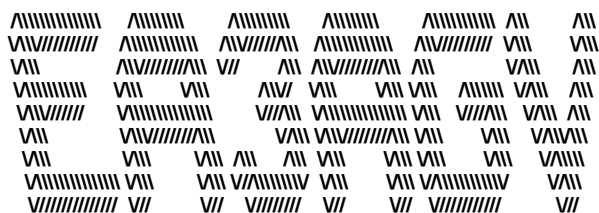


Powered by Proton Development Suite® by Les Johnson©

EA3AGV CW WIN KEYER

USING THE AMICUS18 BOARD & PIC18F26K20



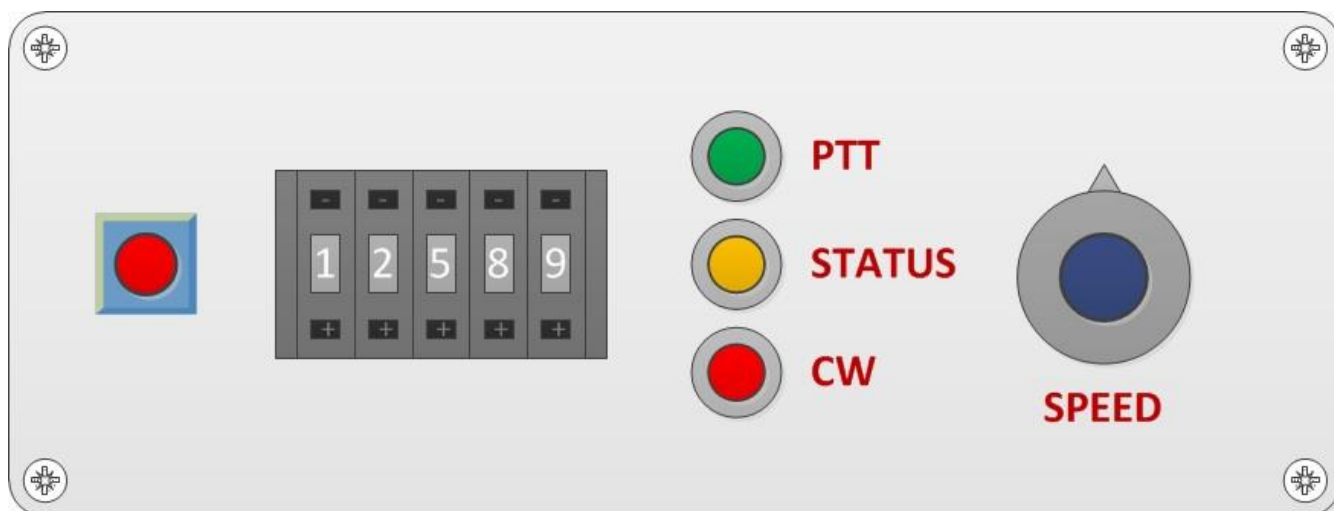
PIC®, MPLAB®, PICkit3® and ICD3® are registered trademarks of Microchip Technology Inc©.
Proton Development Suite® or PDS® are a registered trademark of Les Johnson©.

The project has been developed and written by Alberto Freixanet.

The document has been edited by John Drew.

Thanks for Carlos [EA3CQW] to allow me to use his callsign for the examples.

This project is dedicated to the users of CW Ham Radio and the Proton Basic Compiler.



Front view of the CW Win Keyer.

IMPORTANT NOTICE:

This code is limited for personal use only.

The software is provided "as is", without warranty of any kind, express or implied, including but not limited to the warranties of merchantability, fitness for a particular purpose and non infringement. In no event shall the author or copyright holders be liable for any claim, damages or other liability, whether in an action of contract, tort or otherwise, arising from, out of or in connection with the software or the use or other dealings in the software.

INTRODUCTION:

I have considered this project as a programming exercise. I wanted to know if I could perform a CW Keyer using the Proton Basic Compiler and the Amicus18 board. This code is part of a larger more complex project that I hope to complete in the future.

The keyer provides for a semi automatic CW contact between two amateur radio stations. The keyer has sixteen pre-programmed sequences that may be altered at compile time or through a serial connection to a computer. Content and timing of the CW is adjustable through the PC keyboard and CW speed is adjustable through an external potentiometer. A signal report may be entered through thumbwheel switches. Facility for a buzzer for CW practice is also provided.

Interface to a computer is made through the hardware serial port of the processor while the transmitter interface is via a switched transistor driven by the processor. See the Specific Features for details.

This manual describes the software options and how the keyer may be flexibly adjusted to suit many operators' needs. For example speeds of 8-40 WPM and alternative spacings are supported. A circuit diagram, command and parts lists complete the package.

This Keyer is not traditional, maybe some functions are missing, but the structure allows the PDS user to expand functions where required. One of the most important parts is the manipulation of the screen display by the operator to enable decodes on a terminal of each ASCII character. With this flexible function you can write many more applications.

The PIC18F26K20 has been chosen for the size of its eeprom memory (1024 bytes) and ROM. It is also compatible with the Amicus18 board. However it would have been better to use a PIC18F46K20 or PIC18F46K22 for the greater number of pins.

Very important:

This project uses library resources accumulated over many years avoiding the need to rewrite the subroutines in each project. The user must copy all the additional files in a folder of his computer for its correct operation. Unused macros are not loaded in the ROM of the CPU.

It is essential to insert the following files into your hard drive <D: /> in a folder named <MyIncFiles>, otherwise the compilation will give errors.

Amicus18_26K20.inc

TIPS_AND_TRICKS_V17.Inc

EA3AGV_KEYER_V10.Inc

Amicus_ADCbeta.Inc

DeclareLocalsVars.Inc

I am not responsible for the modifications made by the user in the library files.

General Features:

The code has been written with the Proton Basic Compiler Version 3.7.1.9. The code to read the paddles is built in a simple state machine. The structure of the system could allow up to 100 Macros. A significant number of Macros have already been implemented.

Hardware:

Problem with Paddle Contacts:

All mechanical contacts have a minimum switching load. For example, good telecommunications really need a minimum current of 100uA to 1mA. For this reason the contact current of the paddles should be as high as possible to minimize bouncing through high resistance of the contacts. (See relay manufacturers for information). The current is set by 2 resistors of 3k3 (1 mA for VDD = 3,3V) when the paddle is pressed only.

The main hardware is the Amicus18 board. Several additional shields are added to connect with specific components of the keyer.

- 1 x Amicus18 Board with the PIC18F26K20.
- 1 x Button only for multi-functions
- 1 x potentiometer for CW speed control.
- 1 x Decimal encoder (0 to 9) for the functions.
- 1 x HEX encoder to set the message number. (0 to 15)
- 3 x Decimal encoders (for automatic RST, 3 digits 0 to 9)
- 1 x Logic 3-to-8 line Decoder. (74HC238/74HCT238/74HC239/74HCT239)
- 1 x Data CW output (NPN transistor 100mA).
- 1 x Tone output with potenciometer for volume control.
- 1 x PTT output (NPN transistor 100mA).
- 1 x LED for Status.
- 1 x Output for buzzer in practice mode (NPN transistor 100mA).

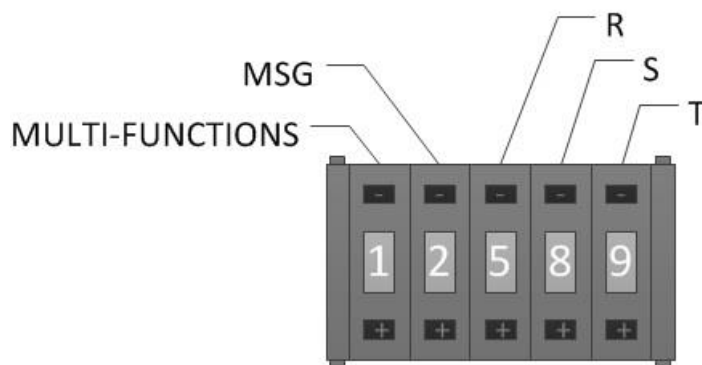
Specific Features:

EA3AGV - AMICUS18 CW WINKEYER V1.0

- 115200 Baud Serial Interface (Code configured with 2 stops bits for better reading by RS232 terminals). The terminal must be configured with 1 stop bit.

- All CW characters generated by the paddles or keyboard are sent to the terminal in ASCII format.
- FOSC = **64Mhz** used for better timing accuracy due to the large code. (Xtal 16Mhz)
- Keyboard option. All ASCII characters and some macros could be generated by the PC keyboard and transformed to CW (dot/dash) characters.
- CW speed (8 - 40 WPM)
- Iambic A, Iambic B or Ultimatic keyer.
- Select 3:1 or 4:1 Dash/Dot ratio.
- Autospace (letter space + Word space)
- Word Space option. (Enabled/Disabled)
- Adjustable Wordspace: [0,5,10,15,20,25,30,35,40,45] % of letter space.
- Adjustable Letterspace: [0,5,10,15,20,25,30,35,40,45] % of word space.
- Farnsworth spacing: [95,90,85,80,75,70,65,60,55,50] % of WPM
- 10 Tones available. [400,450,500,550,600,650,700,800,1000,1250] x Hz
- 2 TUNE Transmitter Modes with configurable TimeOut. (10 to 127 seconds)
- PTT Output with adjustable delay (IN and OUT)
- 1 PTT port pin.
- 1 CW data output port pin.
- 1 Tone port pin.
- Separate Dot/Dash input port pins.
- 1 CW output practice pin for buzzer.
- Sixteen messages in eeprom memory with Hex encoder and one Button.
- Beacon Mode with TimeOut: [3,5,7,9,11,13,15,17,19,21] x seconds
- 100 messages (or more) in ROM memory downloadable to the eeprom within 5 message blocks. Possibility of many extensions because of the large ROM memory availability.
- Numbers transformation in abbreviated form with Macros for contests.
- Serial number generator for Contest/Load number/Clear number/Decrease number by button.
- 10 or more Custom Prosigns (shorthand signals in morse code).
- 100 macros available.
- On line Macros for the Keyboard Mode.
- Paddle only Sidetone (MUTE).

- External Potentiometer speed control.
- Stackable messages.
- Automatic detection of correspondent's **CallSign**. Insert correspondent Callsign in messages by macros then a semi automatic QSO could be made.
- Read the **RST** value from decimal encoders <589> and this function could be used by macros within the automatic messages or on line (Paddles or Keyboard Mode). A semi automatic QSO could be made.
- Multi-functions with 1 Button only and 1 decimal encoder.
 - 0 - Stop Button function.
 - 1 - Memory Button.
 - 2 - Configuration Mode Button.
 - 3 - Paddles/Keyboard Mode Button.
 - 4 - Send RST (Paddles Mode only).
 - 5 - Send QSO counter (Paddles Mode only).
 - 6 - Decrement the QSO counter Button.
 - 7 - TUNE transmitter with Delay ON/OFF with TimeOut.
 - 8 - TUNE transmitter continuously with TimeOut.
 - 9 - Practice Mode for letters, numbers and punctuation.



ENCODERS

Encoders: Left to Right.

- Multifunctions: Decimal encoder (0 to 9), could be changed by 0 to 15 model.
- MSG: Hexadecimal encoder (0 to 15) to choose the message number.
- R: (Readability) Decimal encoder (0 to 9)
- S: (Signal Strength) Decimal encoder (0 to 9)
- T: (Tone) Decimal encoder (0 to 9)

Individual switches in combination with endcaps can easily be built into block assemblies by integrated snap pins. Spacers and housings can be integrated in any given sequence.

It would be very convenient to choose a model that would have the diodes included in the PCB or at least planned.

Special Options:

AutoSpacing Note:

Autospacing is enabled by default to perform all project options. The space between a letter and a word is formatted according to some defined parameters that could be modified by the user.

If the Autospacing is disabled, the keyer works like a semi-automatic keyer with defined dots, dashes and letter space. The letter autospacing is always enabled and is needed for the recording process.

If Query or set autospace or forced character width mode is active and the operator pauses for more the one dot-length after a dot or dash has been sent, this pause will be extended to a full character space. If the dot or dash key is pressed during this time the element will be sent right after the character pause.

This option configures automatic space mode or forced character width. If this mode is active and the operator pauses for more than one dot in length after a period or dash has been sent, this pause will be extended to a full character space. If the dot or the dash key is pressed during this time, the item will be sent right after the character pause.

RST Option:

RST System by John Shannon, K3WWP

The following table lists the values and definitions of the R, S, and T.

R Readability:

- 1 Unreadable
- 2 Barely readable, occasional words distinguishable
- 3 Readable with considerable difficulty
- 4 Readable with practically no difficulty
- 5 Perfectly readable

S Signal Strength:

- 1 Faint signals, barely perceptible
- 2 Very weak signals
- 3 Weak signals
- 4 Fair signals
- 5 Fairly good signals

- 6 Good signals
- 7 Moderately strong signals
- 8 Strong signals
- 9 Extremely strong signals

T Tone:

- 1 Sixty cycles or less, very rough and broad
- 2 Very rough AC, very harsh and broad
- 3 Rough AC tone, rectified but not filtered
- 4 Rough note, some trace of filtering
- 5 Filtered rectified AC, but strongly ripple-modulated
- 6 Filtered tone, definite trace of ripple modulation
- 7 Near pure tone, trace of ripple modulation
- 8 Near perfect tone, slight trace of modulation
- 9 Perfect tone, no trace of ripple, or modulation of any kind

The macros <RST> and <RSTS> inserted in the messages allow automatically sent RST value (3 digits) configured by decimal encoders [599]. When the user hears the call or the response of the correspondent, he must quickly configure the RST digits.

Two macros are available for recorded messages:

<**RST**>: The value of the RST is sent twice.

<**RSTS**>: The value of the RST is sent only once.

In the Keyboard option, the lowercase character "s" serves as an online macro to send the RST figures.

CopyROMtoE2P Command:

In the program, the command <Edata> is not used to save the default values in the eeprom memory. If this command is used, the value of the address labels change each time the message length is modified.

For example:

EEPCW_MYCALL Edata "EA3AGV",0

EEPCW_QTH Edata "BARCELONA",0

The first line only uses 6 bytes. If it is modified ...

EEPCW_MYCALL Edata "EA3AGV/QRP",0

EEPCW_QTH Edata "BARCELONA",0

Then the QTH address in the eeprom has changed and will cause many errors in the code. The solution has been to define all the addresses of the parameters in advance.

```

Symbol EEPCW_MYCALL = 15           '/ 10 Bytes ASCII
Symbol EEPCW_QTH = 26              '/ 16 Bytes ASCII
Symbol EEPCW_QTHLOC = 43           '/ 7 Bytes ASCII
Symbol EEPCW_NAME = 51             '/ 12 Bytes ASCII
Symbol EEPCW_RIG = 64              '/ 18 Bytes ASCII
Symbol EEPCW_PWR = 83              '/ 4 Bytes ASCII
Symbol EEPCW_ANT = 88              '/ 14 Bytes ASCII

```

In this case, the default parameters are written in the ROM. The CopyROMtoE2P command copies these values to the addresses of the corresponding eeprom memory.

Timings:

Due to the numerous parameters that intersect, the realization of timings is very complex.

The code execution time for the letter space and the word space in Paddles Mode has been taken into account. It's much simpler in Keyboard Mode. Due to the construction of the timers, timings for Paddles and Keyboard Mode are different. The timings are accurate with FOSC = 64Mhz. The same accuracy cannot be guaranteed for FOSC = 16Mhz which is only reserved for operation with Proteus.

Practice Mode:

The number 9 of the multi-function encoder allows the option to send letters, numbers and punctuation in random mode. My MyRandom code is used which generates a random Dword variable, see the library.

4 modes are available:

- Mode 0: Groups of 5 letters are sent.
- Modo 1: Groups of 5 figures are sent.
- Modo 2: Groups of 5 letters and mixed numbers are sent.
- Modo 3: Groups of 5 punctuations are sent.
- Modo 3: Groups of 5 letters, figures and mixed punctuations are sent.

Each time the Practice Mode is used, the Seed variable is changed so that the random number is always different. The same line of letters, figures or punctuation is never repeated.

The practice could be stopped by the STOP button or Dot Paddle.

The Dash Paddle allows you to stop the practice by increasing the mode from 0 to 1, 2, etc ... Always start with the multi-function encoder at 9 and press the button.

An output pin is only available for this option to connect a buzzer if the user wants it. In practice mode the MUTE function is always enabled.

3 inputs to 8 outputs Decoder:

The implementation of a 74HC259 decoder has been very necessary due to lack of pins of the PIC18F26K20 of the Amicus18 board. Its function is to scan the different encoders. The inputs are connected to pins RB5, RB6 and RB7 with a high resistance so as not to disturb the ICD3 programmer.

Commands list:

EA3AGV CW WIN KEYER Commands List:

CONFIGURATION MODE:

[A] DOT/DASH compensation, Letter spacing & Word spacing Adjustment:

This command allows you to configure the different timing parameters. There is no corresponding flag to disable each value. To disable these parameters it will be necessary to write the value <0> into each one.

(**D**) Set Dot/Dash compensation. (0 to 31mS) Finish entry with <Enter>.

(**L**) Set Letter Spacing Adjustment. [0,5,10,15,20,25,30,35,40,45,] % of LTSPACE.

Enter a value (0 to 9) to adjust the Letter Spacing. For example to choose the value 15, the number 3 will be typed.

(**W**) Set Word Spacing Adjustment. [0,5,10,15,20,25,30,35,40,45,] % of LTSPACE.

Enter a value (0 to 9) to adjust the Word Spacing. For example to choose the value 15, the number 3 will be typed.

[B] BEACON Mode:

Enable/Disable Beacon, Toggle flag (ON/OFF).

In Beacon Mode a message could be repeated forever if a “**Beacon**” macro is written at the end of message. The time between message may be configured.

The Beacon Mode could run with Paddles or Keyboard Mode.

Step 1:

Push the **CONFIG** Button (Function Encoder pos2) to enter into Configuration Mode.

Step 2:

In Configuration Mode, press the “**B**” option key on the keyboard. The Beacon Mode is enabled and stays enabled until you press the same key again in Configuration Mode. This configuration is saved in eeprom.

Step 3:

With the MSG encoder choose the number of message prepared with a Beacon Macro (0 to 15).

Step 4:

To start the message put the Function Encoder in Memory position (1) and press the Button.

In Paddles Mode:

The Beacon message could be stopped by pressing the Dot or Dash paddle between messages or by the STOP button. This dot or dash is not sent to the transmitter. You can then use the paddles again.

In Keyboard Mode:

The Beacon message may be stopped by pressing the SPACE key between messages or by using the STOP button.

To reactivate the Beacon messages put the Function Encoder in Memory (1) and press the Button.

Example of message with Beacon macro.

ROMCW_MSG5: CData "CQ CQ DE EA3AGV EA3AGV",EOT8,BEACON,0

Translation:

"CQ CQ DE EA3AGV EA3AGV # PSE K [\33",0

The prosign [#] will be translated at the moment of sending the bits and the equivalent code will be sent [AR].

The latter (translated text) is the way to configure the memories with manual programming by the keyboard. Macros always have 3 ASCII characters.

[C] UrCallsign option:

Enable/Disable to Load the correspondent's CallSign. (UrCallSign), Toggle flag (ON/OFF).

A special option allows registering the correspondent's Callsign when, for example, the user answers a call. The recording could be done by the paddles or the PC keyboard. This option allows the recognition and saving the callsign of the correspondent to use in automatic response messages and / or keyboard macros. Some checks are made so that random words are not recorded. When the correspondent's Callsign is recognized, it is sent to the terminal as follows. The Word Autospacing option must be enabled.

<EA3CQW DE [EA3CQW] EA3AGV EA3AGV PSE K>

Due to the structure and special options of this project Autospacing cannot be disabled for this option.

Example 1:

<Receiving> CQ CQ CQ DE EA3CQW EA3CQW PSE K

My answer could be:

<Transmitting> EA3CQW DE [EA3CQW] EA3AGV EA3AGV K

If I make no error in the manipulation, the recording of the Callsign is confirmed.

Example 2:

My Call:

<Transmitting> CQ DE EA3AGV EA3AGV PSE K

<CQ> is not recognized as a valid callsign because the 2 words <CQ DE> must have more than 7 letters.

Example 3:

My Call:

<Transmitting> CQ CQ DE EA3AGV EA3AGV PSE K

<CQ CQ> is not recognized as a valid callsign because the 3 words <CQ CQ DE> have more than 2 spaces.

Detection security:

- 1- The characters are recorded in the String (CW_UrCallSignString).
- 2- Its length is 16 bytes maximum.
- 3- The characters are recorded from the beginning (Start PPT).
- 4- More than 8 characters are recorded until a space is detected after "DE".
- 5- It counts 2 spaces of words from the beginning included after "DE".
- 6- The detected word is sent to the terminal: [EA3CQW] after receiving "DE", to confirm the correct decoding of the correspondent's Callsign.
- 7- In the PTT OFF period, the Status LED flashes at a rate of approximately 2 seconds to indicate that the recording of the <UrCallSign> has been valid.
- 8- If the registered word does not correspond to the expected Callsign, it can be deleted with the STOP button when the PTT has turned OFF, <Receiving> state.
- 9- If you want to use this option in pre-recorded messages, you must use the macros: URCALL1, URCALL2 and CLURCALL. (see the chapter of macros)

[D] DASH/DOT Ratio:

Select 3:1 or 4:1 Dash/Dot ratio, Toggle flag (ON/OFF). The Dash/Dot ratio could be: Dash = 3 dots (standard) or Dash = 4 dots (special).

[E] Exit:

EXIT from Commands mode and returns to normal keyer operation: Paddles or Keyboard Mode.

[G] Farnsworth Spacing:

Enable/Disable Farnsworth Spacing Option, Toggle flag (ON/OFF).

This option helps in the understanding of the CW but also allows the use of paddles with more greater ease by having more time between letters and words.

[H] Not used:

[I] Ultimatic / Iambic A/B:

Enable/Disable the Ultimatic / Iambic A / Iambic B options.

U => Set Ultimatic Mode

B => Set Iambic B Mode

A => Set Iambic A Mode

Type the corresponding letter to choose the option.

Ultimatic differs in this way from iambic: instead of an alternation between dit and dah when both levers are pressed, ultimatic will output the element of the last lever pressed.

By Chuck Olson, WB9KZY:

Confirmed users of either mode A or mode B iambic keying probably won't like Ultimatic mode; once a certain keying sequence is ingrained, it's hard to change to something new. But, Ultimatic mode may be useful for a beginner to electronic keying to try in addition to the iambic modes; it has some interesting differences.

Letter	Iambic	Ultimatic
P	L-R-L = 3	L-R = 2
X	R-L-R = 3	R-L = 2
C	R-L = 2	R-L-L = 3

The basic premise of Ultimatic mode (as I understand it) is that when both paddles have been pressed at the same time, the keyer will output the element of last paddle pressed. This is useful for generating some characters, for example the question mark:

- 1) press the dit paddle for two dits
- 2) keeping the dit pressed, squeeze the dah also for the middle two dahs
- 3) release the dah paddle only for the last two dits
- 4) release the dit

The comma would be generated in the opposite way, for example:

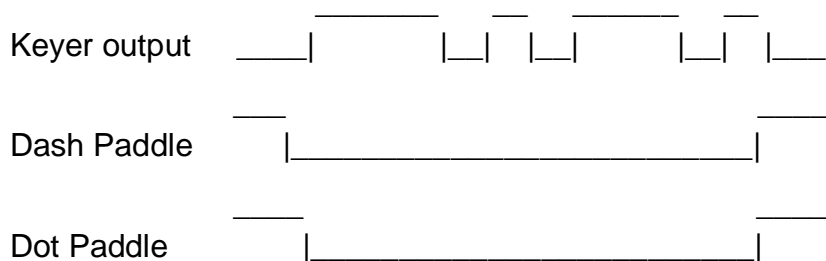
- 1) press the dah paddle for two dahs
- 2) keeping the dah pressed, squeeze the dit for the middle two dits
- 3) release the dit paddle only for the last two dahs
- 4) release the dah

Mode A and B refer to the way that a Morse code keyer handles iambic (squeeze) keying so first, let's define iambic keyer operation. An iambic keyer will send an alternating sequence of dits and dahs as long as both the dit and dah switches are depressed or squeezed.

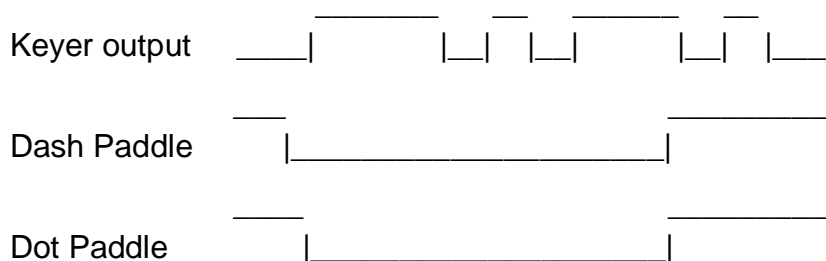
Iambic operation is useful for sending characters that have alternating patterns such as a period or the letter C. An iambic keyer is normally used with a dual lever paddle. It consists of two separately actuated switches. I am right handed and use my thumb for the dits and index finger for the dahs. You can also use a single lever paddle with an iambic keyer but you won't be able to take advantage of the iambic properties of the keyer. Single lever keying is sometimes called slap keying since you can only depress either the dit (slap to the right) or dah (slap to the left) switch – you can't depress both at the same time. Finally, some folks “slap” a dual lever paddle - this is OK, too!

The difference between mode A and B lies in what the keyer does when both paddles are released. The mode A keyer completes the element being sent when the paddles are released. The mode B keyer sends an additional element opposite to the one being sent when the paddles are released. The original Curtis chip is mode A - the WB4VVF Accu-keyer is mode B. You can tell the basic difference between the modes with the letter C. In mode A you could squeeze both paddles (dah before dit) and you would let go of both after hearing the last dit. With mode B, you start the same BUT let go of both paddles after hearing the second *dah*. Here is a diagram of sending a C with mode A and with mode B:

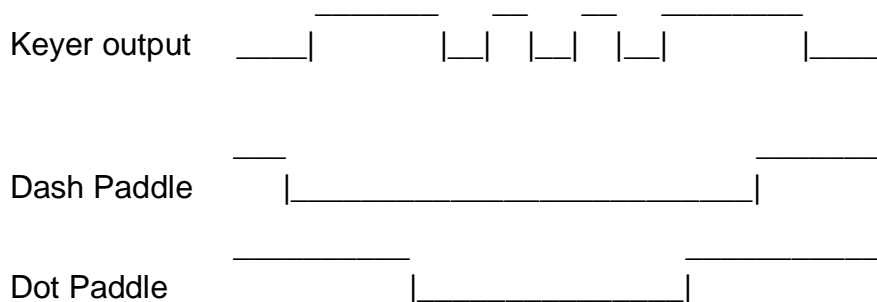
Iambic Mode A: Letter C



Iambic Mode B: Letter C



Ultimatic: Letter X



[J] Word spacing option:

Enable/Disable Word Spacing option, Toggle flag (ON/OFF). 0 = Disable = Semi automatic keyer.

The Word Spacing option is part of the Autospadding option (letter and space). When you disable this option, the option <Detection of URCallSign> is lost. The automatic space between letters is maintained to preserve the option of the letter detector in ASCII format.

[K] Contest Mode:

Enable/Disable Contest Mode, Toggle flag (ON/OFF).

This option allows the user to change the numbers of the QSO counter in short format for contests.

[L] Beacon TimeOut:

Entry a value (0 to 9) to adjust the Beacon Timeout: [3,5,7,9,11,13,15,17,19,21] x seconds.

This Timeout regulates the waiting time between messages sent with the BEACON macro.

[M] Mute:

Enable/Disable MUTE (CW Data & PTT output pin disabled), Toggle flag (ON/OFF).

This option allows training using the Sidetone only.

[N] Read all messages:

Read all messages written in EEPROM/ROM and print them to the terminal.

E => Print all messages from Eeprom memory.

R => Print all messages from ROM memory.

[O] Copy ROM messages to EEPROM:

Load standard messages (Copy a ROM block to an Eeprom block).

This option allows you to copy a block of messages from the ROM (0 to 19) to a specific block of the EEPROM (0 to 2). It must be remembered that block 0 of the EEPROM has a short length: 32 bytes. Blocks 1 and 2 are 64 bytes long.

When you enter two digits then you must end with the <Enter> key.

[P] Print Timings parameters:

This option allows sending all the CW timing parameters and messages to the terminal.

[Q] QSO counter option:

QSO counter option. Read/Write/Erase the QSO counter.

The QSO Counter option allows you to increase a counter whose value is used in macros. There are 3 options available.

R - Read the QSO counter.

L - Load a new value for the QSO Counter. (max 4 digits)

E - Erase/Clear the QSO counter.

A macro (YXX) increments and displays the counter while the macro (XXX) only displays the counter. This macro can be combined with the macros (SNUM) and (CNUM) to send the figures in short mode.

..., **YXX,SNUM,XXX,CNUM**,...

If the counter is: ..., 003,003, ... then it will change to: ..., 003, TTV, ...

For the number 0, the duration of the letter T will be continuous with duration equal to 4 x T standard.

[R] TUNE ON / TUNE ON delay value.

Enter a value (0 to 9) to adjust the TuneON/OFF delay. [10,15,20,25,30,35,40,45,50,55]

The available values (10) are: [10,15,20,25,30,35,40,45,50,55] / 10 Seconds.

Although the table is unique for the 2 delays, the value could be changed by modifying the calculation formula of each one. See the routines: CalcTuneONDelay (CWResultByte) / CalcTuneONDelay (CWResultByte).

[S] SideTone:

Enable/Disable SideTone, Toggle flag (ON/OFF).

The Sidetone is generated by the Timer0.

[T] PTT option:

Enable/Disable transmit PTT, Toggle flag (ON/OFF).

A PTT output pin is available to activate the relay of the transmitter if needed.

[U] TX Delay:

Enter a value (0 to 9) to set the TXON Delay. [25,50,75,100,125,150,175,200,225,255] x 1mS

The actual delay can be modified by adjusting the routine's calculation formula:

CalcKYTXOnDelay (CWResultByte)

[V] RX Delay:

Entry a value (0 to 9) to set the RXON Delay. [25,50,75,100,125,150,175,200,225,255] x 1mS

The actual delay can be modified in the routine formula:

CalcKYRXOnDelay(CWResultByte)

[W] Your Call Sign entry.

Load your CallSign in ASCII format. 16 characters max.

When activating this option with "**W**", then type the callsign with the keyboard. Each key has a TimeOut of 5 seconds for each character. The introduction can be completed with the "ENTER" key. If you let the TimeOut act, an error will occur. Then the string is sent to the terminal for control.

[X] TUNE TimeOut.

Load a Timeout for the TUNE options. Entry 3 values (010 to 127) seconds maximum.

[Y] TONE Frequency Adjust.

[400,450,500,550,600,650,700,800,1000,1250] x Hz. Enter a value (0 to 9) to set the Tone Frequency.

The Tone Frequency (0 to 9) number is saved in eeprom memory. The **<KYToneFreqTable>** command reads the equivalent value to load into the Timer0.

KYTMR0LValue = **KYToneFreqTable**(ValueInput)

The **<KYReadToneFreqTable>** command reads the real frequency value to print to the terminal.

Save_FSR0 = **KYReadToneFreqTable**() (read & load the value into a temporary variable)

```
$if _xtal = 64
'/ Calculate: (64Mhz/4)/(1250*2)=6400/Prescaler(128)=50=>256-50 = 206
PRODH = LookUp MacroTemp, [100,117,131,143,152,160,167,178,194,206]
$endif
```

This formula allows calculating the value to recharge the Timer0. This example is written for 1250Hz.

[Z] Reset Parameters.

Reset all parameters to default settings. (Values from eeprom).

[0] LOAD a message (0 to 15).

When pressing the key **<Y>** (YES) the code reads the encoder corresponding to the messages. Then the message text is written in high case ending with the **<Enter>** key.

[1] ERASE a message (0 to 15).

Erase a message (0 to 15) in eeprom memory. The message number is set by the MSG encoder. When pressing key **<Y>** (YES) the code reads the messages encoder, then the corresponding message is deleted and replaced by the macro **<\ 36>** which means **<"NO MESSAGE IN THIS MEMORY">**.

Macros definition:

Macros are used to insert ASCII characters or some commands into a text message. To be as compatible with all terminals and easily visible by the user, all macros are written with ASCII characters. No control character is used.

You can use 2 types of macros.

- 0 – Macros that insert ASCII characters.
- 1 – Macros that modify program parameters.

ASCII Macros:

ASCII macros are used to replace several texts and prosigns.

This saves a lot of memory space and is easier to write. When the program reads an ASCII macro, insert a space before pasting the characters of the macro, for example:

Message written at the end of the BAS file.

CWTEXT01: **CData** "CQ DE EA3AGV EA3AGV PSE K",0

CWTEXT01: **CData** "CQ DE EA3AGV EA3AGV",EOT8,0

If the user uses the command <0> to record messages in the eeprom, he must also enter a string ending <0> in this way. <ETX0 = \ 00>. This macro does not insert a space.

CQ DE EA3AGV EA3AGV\10\00

EOT1, EOT2, EOT3, EOT4, EOT5, EOT6, EOT7, EOT8:

It replaces a series of characters and prosigns to end a message in a different way.

URCALL1: (\20)

This macro inserts the correspondent's CallSign at the beginning of the present string to be sent later. If the UrCallSign does not exist, a blank will appear.

URCALL2: (\21)

This macro inserts the correspondent's CallSign at the end of this string to be sent later. If the UrCallSign does not exist, a blank will appear. In this case, a space is entered before copying the UrCallSign.

MYCALL2: (\22)

This macro inserts MyCallSign (10 bytes) in this string to be sent later. If the MyCallSign does not exist, a blank will appear. In this case, a space is entered before copying the MyCallSign.

YXX: (\23)

This macro increases the QSO number and inserts it in this string to be sent later.

XXX: (\24)

This macro inserts the QSO number in the present string to be sent later.

QTH: (\25)

This macro inserts the QTH in the present string to be sent later.

LOC: (\26)

This macro inserts the QTH LOCATOR in the present string to be sent later.

Command Macros:

ETX0: (\00)

This command inserts a <0> to the ASCII file to make a final string.

PTT0:

This command deactivates the PTT after applying a predefined delay.

NMSG1: (\17)

This macro gives the order to send the next message written in the EEPROM memory.

NMSG2: (\18)

This macro gives the order to send the second message below the present one as written in the eeprom memory.

NMSG3: (\20)

This macro gives the order to send the third message below the present one as written in the eeprom memory.

SNUM: (\31)

This macro enables the option for the QSO counter (Set flag).

CNUM: (\32)

This macro disables the option for the QSO counter (Clear flag).

BEACON: (\33)

This macro enables the Beacon option. The message could be send forever with **X** seconds interval.

COPTXT: (\34)

This macro copies the last characters from CW string to the same string.

CLURCALL: (\35)

This macro clears the **URCALL** parameters.

NOMSG: (\36)

This macro means **<NO MESSAGE IN THIS MEMORY>**.

RST: (\37)

This macro loads the **RST** values from the 3 decimal encoders and sends the value twice.

RSTS: (\37)

This macro loads the RST values from the 3 decimal encoders and sends the value once.

IMSG0: (\51)

This macro copies message number 0 in this string to be sent later.

IMSG1: (\52) Insert message 1 ...

To see all the commands and macros print the file **<Commands_List.rtf>**.

Messages Structure:

The computer keyboard allows us to enter all the parameters and texts of the messages in ASCII format. To obtain full compatibility with all types of terminals, macros are written in ASCII format as well.

The sending of messages is executed in 3 phases.

1 –Write or copy the complete message in the EEPROM memory.

EEPCW_MSG0 **CLURCALL**, "CQ CQ DE", **MYCALL**, **MYCALL**, **EOT1**, **BEACON**, 0

2 –The program copies the text and translates the macros into the RAM (string array).

Clear URCallSign, "CQ CQ DE EA3AGV EA3AGV # K [", (**133** => set Beacon flag), 0

3 –The program translates into morse format each character including the prosigns and modulates the output pin with the bits in short / long format or Dot / Dash.

LS = LetterSpace

WS = WordSpace

CW_C, **LS**, CW_Q, **WS**, CW_C, **LS**, CW_Q, **WS**, CW_D, **LS**, CW_E, **WS**, CW_E, **LS**, CW_A, **LS**, CW_3, **LS**, CW_A, **LS**, CW_G, **LS**, CW_V, **WS**, CW_E, **LS**, CW_A, **LS**, CW_3, **LS**, CW_A, **LS**, CW_G, **LS**, CW_V, **WS**, CW_AR, **LS**, CW_K, DelayRXON(optional), PTT_OFF

Messages can be written to memory in 2 modes.

- 1- Edit the end of the project's .BAS file by changing the CData tables in the sector for eeprom and compiling the file. Next, these tables must be copied by blocks of 5 messages to the eeprom memory.
- 2- Through the configuration menu, you can write messages from number 0 to 15. In this case, it will be written directly into the eeprom memory.

In case (1) the macros are used in tag format <EOT1, EOT2, EOT3, ...

In case (2) all the characters will be written in readable ASCII standard format. If you want to write the macro <EOT1>, you will put the equivalent "_ # _ K_", (_) being a space.

Through the compiler the message will be written in this way.

CWTXT06: **CData** "CQ CQ CQ DE", **MYCALL**, **MYCALL**, " # PSE", **EOT2**, **BEACON**, 0

Through the configuration menu (command **0**) the message will be written in this other way.

CQ CQ CQ DE\22\22 # PSE K[\33\00

<**122**> is the macro to insert MyCallSign.

<**#**> is the prosign [AR].

<**[**> is the PTT OFF command.

<**133**> is the Macro (command) to set the Beacon flag.

<**100**> is the command for final string <,0>.

The different delays are inserted automatically according to the options defined in the eeprom memory.

The ROM tags <ROMCW_MSG0> to <ROMCW_MSG15>, are the default messages of the eeprom memory. It can be deleted and changed through the configuration menu. These messages are copied into the eeprom memory at the first initialization of the parameters.

Caution: When using the "Z" command to reset the parameters in the configuration menu, all source messages will be copied again.

The labels of the ROM <CWTXT01> to <CWTXT100>, are the default messages of the ROM memory. It can only be written through the compiler's editor.

Messages from the ROM can be copied by blocks of 5 messages to the eeprom memory. In this way you can write messages for all competitions and / or expeditions and / or activations throughout the year. The copies are made through the configuration menu.

Macros Definition:

MACROS LIST:

MACROS WRITTEN IN ASCII in all messages. (number 00 to 100)

- ETX0** "\00" => Final of String in eeprom messages. (loaded manually)
- STX0** "\02" => %,PTT1 ([KA], PTT On, Start of message) (NOT used)
- EOT1** "\03" => #,SP,K,PTT0 ([AR], K, PTT OFF + RXON Delay, End of message)
- EOT2** "\04" => K,PTT0 ([K], PTT OFF + RXON Delay, End of message)
- EOT3** "\05" => &,PTT0 ([KN], PTT OFF + RXON Delay, Go only, invite a
specific station To transmit)
- EOT4** "\06" => <,PTT0 ([BK], PTT OFF + RXON Delay, Invite receiving station To transmit
- EOT5** "\07" => ^,SP,E,SP,E,PTT0 ([SK], SP, E, SP, E, PTT OFF + RXON Delay, Caret
End of contact (sent before Call)
- EOT6** "\08" => ^,SP,PTT0 ([SK], SP, PTT OFF + RXON Delay, Caret End of contact
(sent before Call)
- EOT7** "\09" => #,SP,PSE,SP,K,PTT0 ([AR], SP, PSE, SP, K, PTT OFF + RXON Delay,
End of message)
- EOT8** "\10" => PSE,SP,K,PTT0 ([K], PSE,SP,K End of Transmission. "K")
- PTT0** "[" => PTT OFF + RXDelay ([[]], PTT OFF + RXON Delay
- NMSG1** "\17" - Send the Next Message in the list.
- NMSG2** "\18" - Send the Next second Message in the list.
- NMSG3** "\19" - Send the Next third Message in the list.
- URCALL1** "\20" - Load corresponding call in string to send first in the message.
- URCALL2** "\21" - Load corresponding call in string to place at the end of the message.
- MYCALL** "\22" - Load MY CALL the CW_StringSend string array. (10 Bytes)
- YXX** "\23" - The Macro <YXX> increments and sends the new QSO number.

XXX	"\24" - The Macro <XXX> sends the current QSO number.
QTH	"\25" - Insert QTH (string) (16 Bytes)
LOC	"\26" - Insert QTH LOCATOR 6 characters string (7 bytes)
MYNAME	"\27" - Insert MY NAME 11 characters String (12 Bytes)
MYRIG	"\28" - Insert MY RIG 17 characters String (18 Bytes)
MYANT	"\29" - Insert MY ANT 13 characters String (14 Bytes)
PWR	"\30" - Insert the RIG Power (4 Bytes)
SNUM	"\31" - Set the Short Numbers Flag for QSO Number.
CNUM	"\32" - Clear the Short Numbers Flag for QSO Number.
BEACON	"\33" - Macro to send the same Message forever with X seconds interval.
COPTXT	"\34" - Copy the last characters from CW string to the same string. (Repeat CQ)
CLURCALL	"\35" - Clear all the URCALL parameters.
NOMSG	"\36" - "NO MESSAGE IN THIS MEMORY"
RST	"\37" - Load the RST values from the 3 Decimal encoders. Send twice.
RSTS	"\38" - Load the RST Short values from the 3 Decimal encoders. Send once.
HCUAGN	"\39" - (HPE CUAGN) text. Macro to save bytes in eeprom memory.
QSLB	"\40" - (QSL VIA BURO) text. Macro to save bytes in eeprom memory.
S7373	"\41" - (73 73) text. Macro to save bytes in eeprom memory.
MSG0	"\51" - Insert the MSG0
MSG1	"\52" - Insert the MSG1
MSG2	"\53" - Insert the MSG2
MSG3	"\54" - Insert the MSG3
MSG4	"\55" - Insert the MSG4
MSG5	"\56" - Insert the MSG5
MSG6	"\57" - Insert the MSG6
MSG7	"\58" - Insert the MSG7
MSG8	"\59" - Insert the MSG8
MSG9	"\60" - Insert the MSG9
MSG10	"\61" - Insert the MSG10
MSG11	"\62" - Insert the MSG11
MSG12	"\61" - Insert the MSG12
MSG13	"\62" - Insert the MSG13
MSG14	"\63" - Insert the MSG14
MSG15	"\64" - Insert the MSG15

Keyboard Mode:

Low Case Commands on the fly with the Keyboard Mode. (Lower Case characters only)

- [a] Send ANT string (Antenna)
- [b] Send “_QSL_VIA_BURO” string: Address = CWTXTBB
- [c] Contest mode, Toggle FLAG (on/off)
- [d] Send “_DE_EA3AGV” string: Address = CWTXTEE
- [f] Send “_73_73_HPE_CUAGN_SK_EE” string: Address = CWTXTCC
- [l] (L) Low Case. Send QTHLOC string (QTH LOCATOR)
- [m] Send MYCALL string (MYCALLSIGN) (by interrupt)
- [n] Send MYNAME string (MY NAME)
- [p] Send “PSE K” string + [(PTT OFF).
- [q] Send QTH string (QTH)
- [r] Send RIG string (RIG)
- [s] Read the RST values from the Decimal encoders and send to CW Datapin
- [t] Send “_FB_TNX_FER_NICE_QSO” string: Address = CWTXTAA
- [u] Send URCALL string if it is loaded already. (URCALLSIGN) (by interrupt)
- [x] Send the QSO number. (max 9999) (XXX) Macro.
- [y] Increment & send the QSO number. (max 9999) (YXX) Macro.

A message always ends with the <Enter> key.

Partitioning a Message:

Instead of one long message you may also record multiple shorter messages in a single message memory. I call this a partitioned message.

For example: Using Macros **NMSG1**, **NMSG2**, **NMSG3**

```

'/ Message Nb 0, 32 Bytes MAX.
MSG0: CLURCALL, "CQ CQ DE", MYCALL, MYCALL, EOT8, BEACON, 0
'/-----
'/ Message Nb 1, 32 Bytes MAX.
MSG1: URCALL1, " DE", MYCALL, " GM UR RST IS", RST, NMSG2, 0
'/-----
'/ Message Nb 2, 32 Bytes MAX.
MSG2: URCALL1, " DE", MYCALL, " TNX QSO", S7373, " #", CLURCALL, EOT5, 0
'/-----
'/ Message Nb 3, 32 Bytes MAX.
MSG3: "= MY NAME IS", MYNAME, " QTH IS", MYQTH, NMSG1, 0
'/-----
'/ Message Nb 4, 32 Bytes MAX.
MSG4: " = HW?", URCALL2, " DE", MYCALL, EOT1, 0
```

The MSG2 is partitioned with MSG3 and MSG4.

The MSG2 macro means: send the next second message in the list. The MSG3 macro means: send the next third message in the list. This block of 5 messages could be copied in other block.

Other example: Using Macros **IMSG0, IMSG1,... , IMSG15**

```
'/ Message Nb 0, 32 Bytes MAX.
MSG0: CLURCALL, "CQ CQ DE", MYCALL, MYCALL, EOT8, BEACON, 0
'/-----
'/ Message Nb 1, 32 Bytes MAX.
MSG1: URCALL1, " DE", MYCALL, " GM UR RST IS", RST, IMSG3, 0
'/-----
'/ Message Nb 2, 32 Bytes MAX.
MSG2: URCALL1, " DE", MYCALL, " TNX QSO", S7373, " #", CLURCALL, EOT5, 0
'/-----
'/ Message Nb 3, 32 Bytes MAX.
MSG3: " = MY NAME IS", MYNAME, " QTH IS", MYQTH, IMSG4, 0
'/-----
'/ Message Nb 4, 32 Bytes MAX.
MSG4: " = HW?", URCALL2, " DE", MYCALL, EOT1, 0
```

The IMSG3 macro means: send the message number 3 of the 15 in the list. The Macros IMSG3, IMSG4 identify the real number of the message. This block is not transportable to another block in eeprom.

CONCLUSION

This project is in beta phase. I hope that this code may be useful. Any comments would be welcome to improve this Keyer.

EA3AGV WIN KEYER version 1.0. 25 July 2019

Alberto Freixanet



COMMANDS LIST:

BUTTON FUNCTION

Choose a function by the selector (0 to 9) & Press the Button.

- 0** '/' STOP anything & Clear Beacon/MSG.
- 1** '/' MEMORY (Run a message [0 to 15] from eeprom memory point to the MSG selector)
- 2** '/' CONFIGURATION Mode
- 3** '/' Paddles/Keyboard Mode. Toggle ON/OFF.
- 4** '/' Send RST (Paddles Mode only)
- 5** '/' Send QSO counter (Paddles Mode only)
- 6** '/' Decrement the QSO counter
- 7** '/' TUNE the Transmitter with Delay ON/OFF. (Exit with Paddles or STOP button)
- 8** '/' TUNE the Transmitter continuously. (Exit with Paddles or STOP button)
- 9** '/' CW Training. Random output of letters, numbers, punctuations. Output SideTone & CW training pin.

CONFIGURATION MODE:

- A** '/' Entry value for Dot/Dash Compensation, Letter Spacing & Wordspace spacing Adjustments.
- D** '/' Set Dot/Dash Compensation. (00 to 31mS) Entry 2 real values.
- L** '/' Set Letter Spacing Adjustment. [0,5,10,15,20,25,30,35,40,45] % of LTSPACE
- W** '/' Set Wordspace Spacing Adjustment. [0,5,10,15,20,25,30,35,40,45] % of WDSPACE
- B** '/' Enable/Disable Beacon, Toggle flag ON(1)/OFF(0)
- C** '/' Enable/Disable To Load the correspondent's CallSign. (UrCallSign), Toggle flag (ON/OFF)
- D** '/' Select 3:1(0) or 4:1(1) Dash/Dot ratio, Toggle flag (ON/OFF)
- E** '/' EXIT from Command mode and returns to normal keyer operation.
- F** '/' Enable/Disable Farnsworth Spacing Option, Toggle flag (ON/OFF).
- G** '/' Load Farnsworth Spacing value (-5% WPM Step) [95,90,85,80,75,70,65,60,55,50] % of WPM
- H** '/' Not used
- I** '/' Enable/Disable Ultimatic / Iambic B / Iambic A options.
 - U** - Set Ultimatic Mode.
 - B** - Set Iambic B Mode.
 - A** - Set Iambic A Mode.
- J** '/' Enable/Disable Word Spacing option, Toggle flag (ON/OFF). 1 = Enable => Semi automatic keyer.
- K** '/' Enable/Disable Contest Mode, Toggle flag (ON/OFF). Using short numbers.
- L** '/' Load Beacon TimeOut value. [3,5,7,9,11,13,15,17,19,21] x seconds
- M** '/' Enable/Disable MUTE (CW Data & PTT disabled), Toggle flag (ON/OFF).

- N** '/' Read all messages written in EEPROM/ROM memory and print to the terminal.
- E** '/' Print all messages from Eeprom memory.
- R** '/' Print all messages from ROM memory.
- O** '/' Load standard messages [Copy a ROM Block (0 to 19) to an EEP Block (0 to 2)].
- P** '/' Read & Print all timings parameters.
- Q** '/' QSO counter Option. Read/Write/Erase QSO counter.
- R** - Read the QSO counter.
- L** - Load a new value for the QSO Counter. (max 4 digits: 9.999) Press Enter to finish.
- E** - Erase/Clear QSO counter.
- R** '/' Entry value to Adjust TUNE ON/TUNE OFF Delay. [10,15,20,25,30,35,40,45,50,55] / 10 = Seconds
- S** '/' Enable/Disable SideTone, Toggle flag (ON/OFF)
- T** '/' Enable/Disable Transmit PTT output, Toggle (ON/OFF) (recommended ON)
- U** '/' Entry value to Set the TX-Delay. [25,50,75,100,125,150,175,200,225,250] x 1 mS.
- V** '/' Entry value To Set the RX-Delay. [25,50,75,100,125,150,175,200,225,250] x 1mS.
- W** '/' Entry my CallSign in ASCII. 16 characters maximum.
- X** '/' Load TUNE TimeOut value (10 to 127 seconds max)
- Y** '/' Not used
- Z** '/' Reset parameters To default settings. (Values in eeprom)
- 0** '/' LOAD a message: (0 to 15) The number of message is set by the encoder MSG. (Write a Text)
- 1** '/' ERASE a message: (0 to 15) The number of message is set by the encoder MSG.

KEYBOARD MODE:

'/ Low Case characters Macros on the fly for KEYBOARD MODE. (Low Case characters only)

MACROS FOR KEYBOARD ON THE FLY.

- a** '/' Send ANT string (Antenna)
- b** '/' Send "**_QSL_VIA_BURO**": Address = CWTXTBB
- c** '/' Contest Mode. Toggle Flag (ON/OFF)
- d** '/' Send "**_DE_EA3AGV**": Address = CWTXTEE
- f** '/' (Final) Send "**_73_73_HPE_CUAGN_SK_EE**": Address = CWTXTCC
- l** '/' (L) Low Case. Send QTHLOC string (QTH LOCATOR)
- m** '/' Send MYCALL string (MYCALLSIGN) (By interrupts)
- n** '/' Send MYNAME string (MY NAME)
- p** '/' Send "**_PSE_K**" + [: Address = CWTXTDD
- q** '/' Send QTH string (QTH)
- r** '/' Send RIG string (RIG)
- s** '/' Read the RST values from the Decimal encoders & send to CW Datapin

- t** '/' Send "**_FB_TNX_FER_NICE_QSO**": Address = CWTXTAA
- u** '/' Send URCALL string if it is loaded already. (URCALLSIGN) (By interrupts)
- x** '/' Send the QSO number. (max 9999) (XXX) Macro.
- y** '/' Increment & send the QSO number. (max 9999) (YXX) Macro.

MACROS LIST:

MACROS WRITTEN IN ASCII in all messages. (number **00** to **100**)

- ETX0** "\00" => Final of String in eeprom messages. (loaded manually)
- STX0** "\02" => %,PTT1 ([KA], PTT On, Start of message) (NOT used)
- EOT1** "\03" => #,SP,K,PTT0 ([AR], K, PTT OFF + RXON Delay, End of message)
- EOT2** "\04" => K,PTT0 ([K], PTT OFF + RXON Delay, End of message)
- EOT3** "\05" => &,PTT0 ([KN], PTT OFF + RXON Delay, Go only, invite a specific station To transmit)
- EOT4** "\06" => <,PTT0 ([BK], PTT OFF + RXON Delay, Invite receiving station To transmit)
- EOT5** "\07" => ^,SP,E,SP,E,PTT0 ([SK], SP, E, SP, E, PTT OFF + RXON Delay, Caret End of contact (sent before Call))
- EOT6** "\08" => ^,SP,PTT0 ([SK], SP, PTT OFF + RXON Delay, Caret End of contact (sent before Call))
- EOT7** "\09" => #,SP,PSE,SP,K,PTT0 ([AR], SP, PSE, SP, K, PTT OFF + RXON Delay, End of message)
- EOT8** "\10" => PSE,SP,K,PTT0 ([K], PSE,SP,K End of Transmission. "K")
- PTT0** "[" => PTT OFF + RXDelay ([[]], PTT OFF + RXON Delay
- NMSG1** "\17" '/' Send the Next Message in the list.
- NMSG2** "\18" '/' Send the Next second Message in the list.
- NMSG3** "\19" '/' Send the Next third Message in the list.
- URCALL1** "\20" '/' Load correspondal call in string to place first in the message to send.
- URCALL2** "\21" '/' Load correspondal call in string to place at the end of the message.
- MYCALL** "\22" '/' Load MY CALL the CW_StringSend string array. (10 Bytes)
- YXX** "\23" '/' The Macro <YXX> increments & sends the new QSO number.
- XXX** "\24" '/' The Macro <XXX> sends the current QSO number.
- QTH** "\25" '/' Insert QTH (string) (16 Bytes)
- LOC** "\26" '/' Insert QTH LOCATOR 6 characters string (7 bytes)
- MYNAME** "\27" '/' Insert MY NAME 11 characters String (12 Bytes)
- MYRIG** "\28" '/' Insert MY RIG 17 characters String (18 Bytes)
- MYANT** "\29" '/' Insert MY ANT 13 characters String (14 Bytes)
- PWR** "\30" '/' Insert the RIG Power (4 Bytes)
- SNUM** "\31" '/' Set the Short Numbers Flag for QSO Number.
- CNUM** "\32" '/' Clear the Short Numbers Flag for QSO Number.
- BEACON** "\33" '/' Macro to send the same Message forever with X seconds interval.

COPTXT "\34" '/' Copy the last characters from CW string to the same string. (Repeat CQ)

CLURCALL "\35" '/' Clear all the URCALL parameters.

NOMSG "\36" '/' "NO MESSAGE IN THIS MEMORY"

RST "\37" '/' Load the RST values from the 3 Decimal encoders. Send twice.

RSTS "\38" '/' Load the RST Short values from the 3 Decimal encoders. Send once.

HCUAGN "\39" '/' (HPE CUAGN) text. Macro because to save bytes in eeprom memory.

QSLB "\40" '/' (QSL VIA BURO) text. Macro because To Save bytes in eeprom memory.

S7373 "\41" '/' (73 73) text. Macro because to save bytes in eeprom memory.

IMSG0 "\51" '/' Insert the MSG0

IMSG1 "\52" '/' Insert the MSG1

IMSG2 "\53" '/' Insert the MSG2

IMSG3 "\54" '/' Insert the MSG3

IMSG4 "\55" '/' Insert the MSG4

IMSG5 "\56" '/' Insert the MSG5

IMSG6 "\57" '/' Insert the MSG6

IMSG7 "\58" '/' Insert the MSG7

IMSG8 "\59" '/' Insert the MSG8

IMSG9 "\60" '/' Insert the MSG9

IMSG10 "\61" '/' Insert the MSG10

IMSG11 "\62" '/' Insert the MSG11

IMSG12 "\61" '/' Insert the MSG12

IMSG13 "\62" '/' Insert the MSG13

IMSG14 "\63" '/' Insert the MSG14

IMSG15 "\64" '/' Insert the MSG15