The Ultimate Morse Sender Program

There are two versions of the ZS6BVR Morse sender programs

1. The Other as a Morse sender to use with your CW rig to transmit Morse on the HF bands (It is planned to Receive Morse)
2. One for use as the ultimate Morse Trainer program

If you are reading this please download the PDF version of this document which is updated regularly and has all the chapters here :   
  
73 de Nico  
ZS6BVR

### [1. Introduction to Morse Code Today](https://github.com/nic0michael/ZS6BVR_MorseCodeSenderPiPico)

Morse code is still widely used by amateur radio operators for long-distance and low-power communication.  
It can be copied by ear even in noisy or weak-signal conditions where digital modes fail.  
Emergency services and hobbyists value Morse as a reliable backup communication method.  
Practicing Morse also trains memory, focus, and pattern recognition skills useful in many field  
Working DX during periods of **Low Sunspot Count** like now can be done with CW when SSB is dead  
Working QRP can be done with CW even **below 1W power**

### [2. ZS6BVR\_MorseCodeSenderPiPico](https://github.com/nic0michael/ZS6BVR_MorseCodeSenderPiPico)

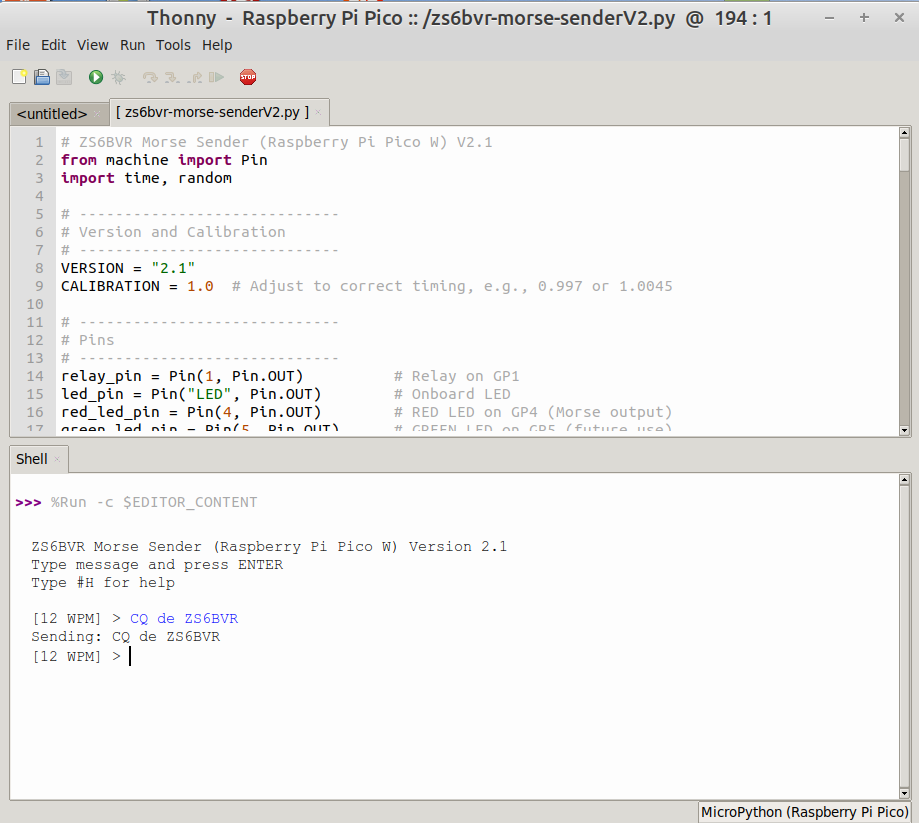
The purpose of the project is to Key your HF Rig and send Morse from your keyboard by operating a read relay to key your CW Transmitter  
  
Phase 2 of this project is to read Morse Code giving you a CW Transciever program.  
  
**Features**:

This program gives you 9 memories to save and send things like “CQ CQ DX DE ZS6BVR”

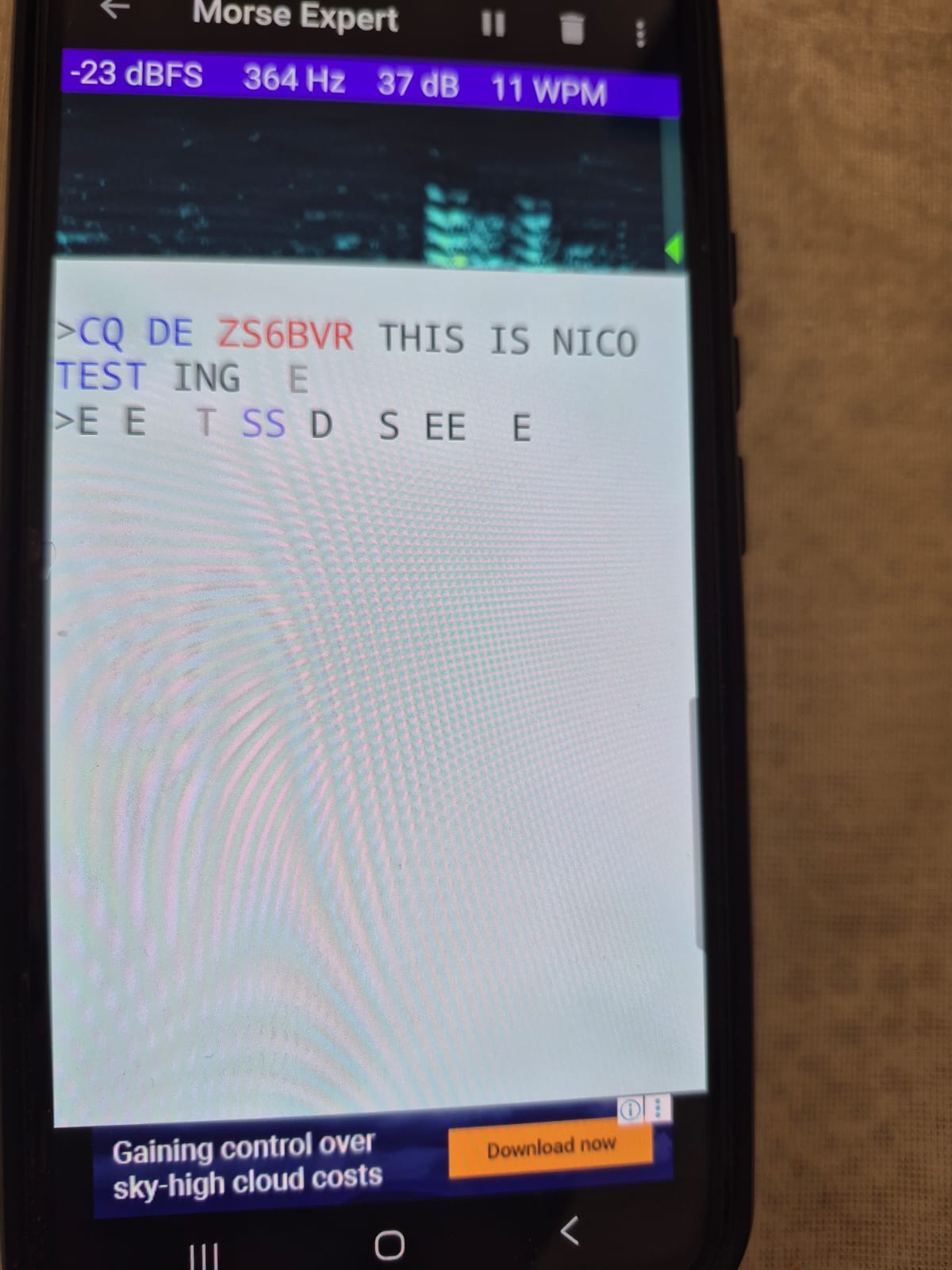
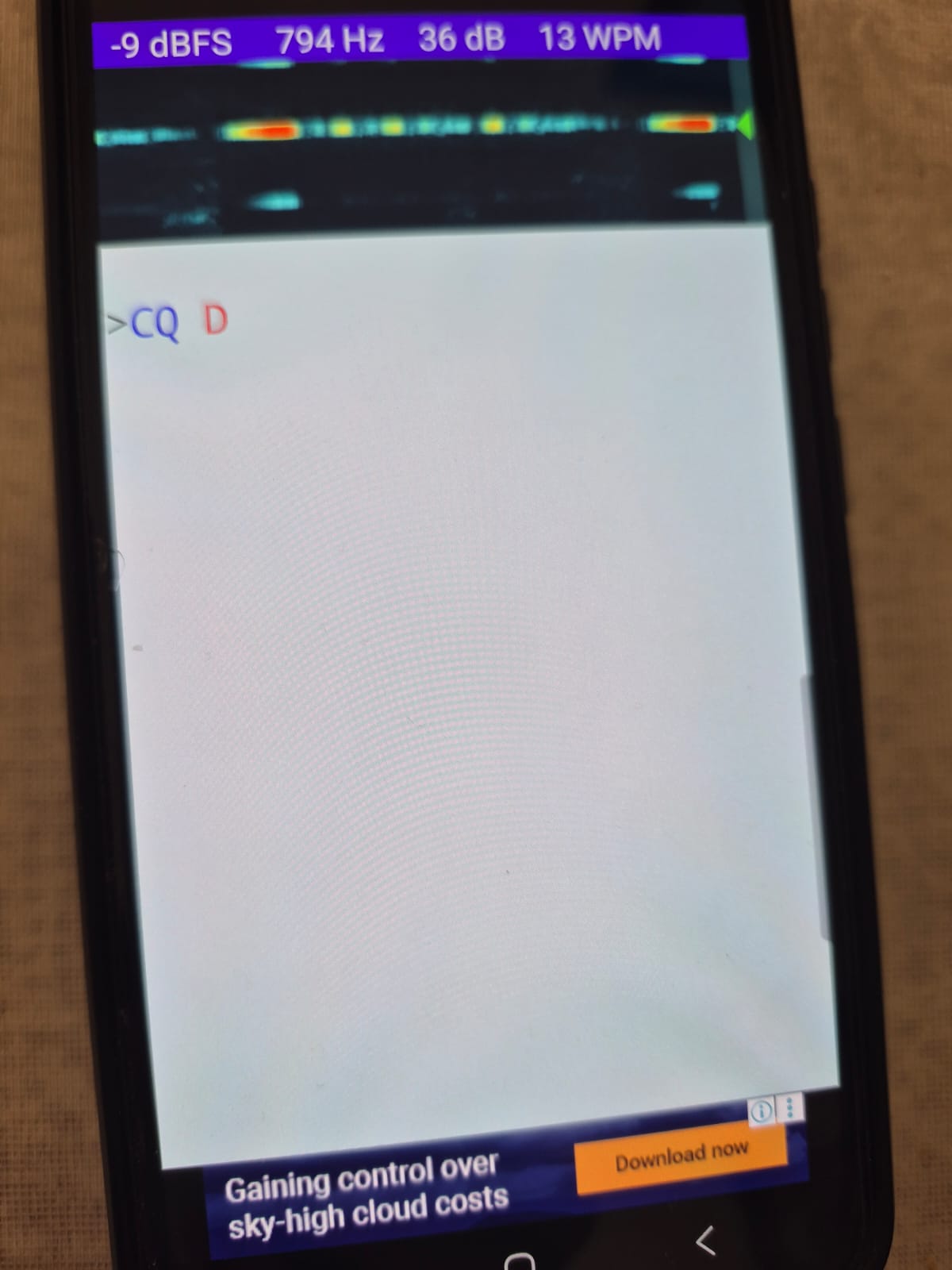
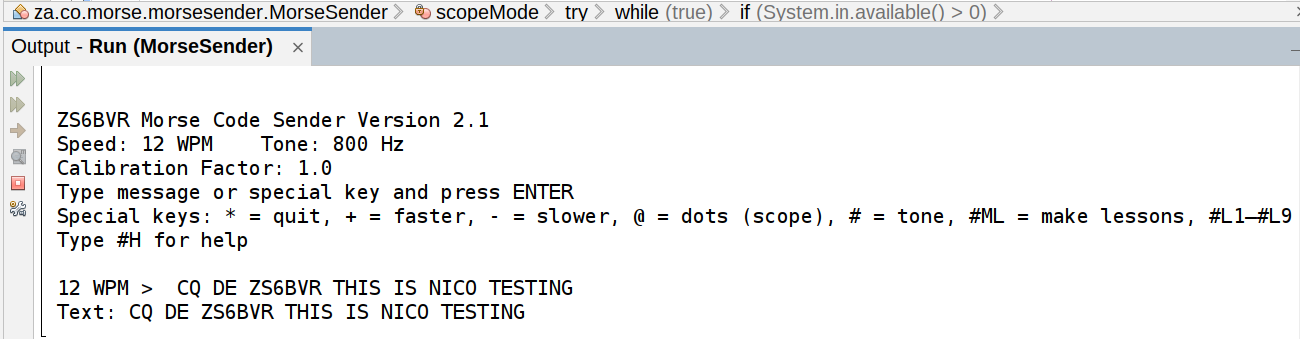
**It also has commands :**  
a. @ will send continuous dots (scope calibration)  
b. # will send continuous tone (frequency counter calibration)  
c. [text] Repeat text 3 times with pauses  
d. #H Show this help message  
e. + /- You can increase or decrease the speed  
  
This is written in Python and runs on the smallest Arduino replacement   
the Raspberry Pi Pico 2W  
  
  
  
**My Raspberry Pi Pico 2W running this program**



**The program running from my laptop:**



Here I did a test sending this message :  
  
  
And here I received it from my phone:  
  
And then the completed message  
  
Then my wife spoke and you see noise text  
This was good enough to prove that it worked  
And my 800Hz Tone was 6Hz off not bad!



You can down load this from here :  
<https://github.com/nic0michael/ZS6BVR_MorseCodeSenderPiPico>

If you know how to Git Clone this project you will be able to pull down the latest future updates

You will find circuit diagrams here

# 3. ZS6BVR Morse Sender (Java Project)

This program uses you computers Sound System to send Morse code  
The purpose of this program is to help you increase you Morse code receiving speed  
  
**This is the ULTIMATE MORSE TRAINER PROGRAM**  
**I have not seen a program with all its features**.  
It can send groups of five “wide-spaced” sent at 20WPM but effectively 10WPM helping you increase your receive speed. In 1980 I wrote my first program to do that and passed the 12WPM ZS exam.   
  
Features:

1. A unique feature is it generates 9 new Lesson files

a. 3 files with Random groups of five  
b. 3 plane English word files  
c. 3 random Callsign files

**It also has commands :**  
a. [n] will sent n lines on random groups of 5  
b. @ will send continuous dots (scope calibration)  
c. # will send continuous tone (frequency counter calibration)

You can down load this program from here :  
<https://github.com/nic0michael/MorseSender>

Where you can get installation Instructions

If you know how to Git Clone this project you will be able to pull down the latest future updates

### [4. Software Requirements](https://github.com/nic0michael/ZS6BVR_MorseCodeSenderPiPico)

This will depend on which hardware or software option you want to use.  
  
**The Java program (Morse Trainer) requires :**

* Java 17 or later
* Maven if you want to build and run.
* Git is useful for cloning both projects and keeping up to date with future updates.

**The Pico program (CW transmitter and Receiver) needs:**

* MicroPython firmware flashed onto the Raspberry Pi Pico 2 W.
* Thonny IDE is recommended to upload and run the Python script on the Pico.
* Git is useful for cloning both projects and keeping up to date with future updates

### [5. Hardware Requirements](https://github.com/nic0michael/ZS6BVR_MorseCodeSenderPiPico)

This will depend on which hardware or software option you want to use.  
  
**For the Raspberry Pi Pico version, you need:**

* The Raspberry Pi Pico 2 W (We plan to introduce a WIFI remote option)
* Reed Relay
* 2N3439 NPN transistor
* 7805 voltage regulator
* USB cable
* Small fuse are recommended for safe operation.

**For the Java version (Morse Trainer), no extra hardware is needed** beyond a computer with a sound card and speakers or headphones.

### [6. Installation Steps](https://github.com/nic0michael/ZS6BVR_MorseCodeSenderPiPico)

**Clone the projects**Git clone both repositories: the “Morse Trainer” (Java-based) and the “CW Sender and Receiver” (Pi Pico-based). Refer to the online instructions of my two projects here in Github:

https://github.com/nic0michael/MorseSender ( Morse trainer) https://github.com/nic0michael/ZS6BVR\_MorseCodeSenderPiPico (CW Sender and Receiver)

You can also get comprehensive instructions there

### [7. Command Overview](https://github.com/nic0michael/ZS6BVR_MorseCodeSenderPiPico)

**Morse Trainer (Java Program)**@ → Continuous dots (scope calibration)  
# → Continuous tone (frequency counter calibration)  
[n] → Send n lines of random groups of 5  
+ or - → Increase or decrease speed  
#H → Show help  
[text] → Repeat text 3 times with pauses  
\* → Exit program  
  
What ever text you type will be sent

**CW Sender and Receiver (Pi Pico Program)**@ → Continuous dots (scope calibration)  
[n] → Send n lines of random groups of 5  
+ or - → Increase or decrease speed  
#H → Show help  
[text] → Repeat text 3 times with pauses  
\* → Exit program  
  
What ever text you type will be sent