CS1er

The TI99 Cassette to FIAD Converter
By Dean Corcoran
www.cs1er.com

Documentation V1.2

As Released for CS1er V0.95b

Contents

Using CS1er

Connect your sound source (Cassette Player/TI99/4a) up to the sound input device on your sound card (Line In/Microphone).

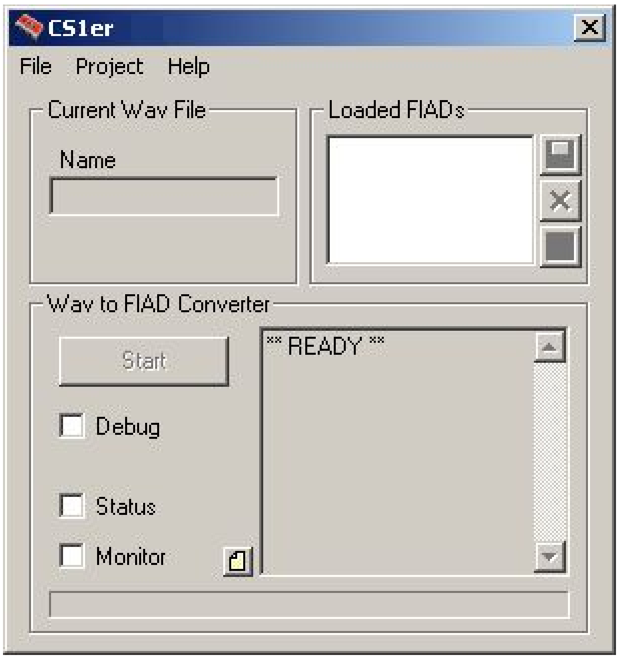
Using a Sound recording program, record the cassette file sound.

Notes:

- Try to record at a high volume level, but prevent clipping.
- Occasional Clipping of sound OK (start of file) but not encouraged.
- CS1er Will attempt to automatically adjust volume level.
- If possible cut off any excess non-binary sound(noise) at beginning.
- It is advised not to perform any digital equalising or enhancement once recorded. (Equalising using Analogue Equaliser prior to entering Computer gives better results, Try to roll-off below 250Hz & above 2KHz)

Save your file as a Mono, 8 bit, 44.1 KHz WAV File. (RIFF Format) (Must Have .WAV Extension, CS1er will add this extension if not present)

After opening CS1er you will be presented with the main CS1er window. (as in Figure 1.1)



** READY **

Figure 1.1 - CS1er Main Window

Open your saved wav file by clicking on open from the file menu. (as in Figure 1.2)

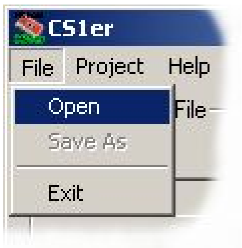


Figure 1.2 - Open Menu

(files are assumed to be a .WAV (RIFF) file recorded in Mono, 8 Bit 44100 Hz format. CS1er will not load Foreign, corrupt or non-standard Wav files and will give an error message if it detects them)

If the file selected was a "FIAD" or "TIFILES" formatted file, it will now appear in the "Loaded FIADs" section of the main window.

If the file opened was a WAV file it should appear in the "Current Wav File" section of the main window.
Clicking the **"Start"** button will now start the decoding process.

The progress bar at the bottom of the main window will show the current position in the wav file being converted.
The Conversion log in the main window will display a log of progress information including, current record status and sync information.
If you wish to show more detailed information *check* the **"Status"** or **"Monitor"** check boxes while CS1er is decoding.
The "Status" check box will display the Staus window which shows current details in the conversion process. See a more detailed description later in this document.
The "Monitor" check box will display a graphic of the current Wave data being analyzed. See a more detailed description later in this document.

Provided all goes well, CS1er will successfully decode your tape file after a few moments and will then prompt asking for a name to label your FIAD file. When naming a FIAD you are restricted to ten alpha-numeric characters. The name you supply will be internally entered into the FIAD itself. Please note that some emulators are fussy if the internal FIAD name is different to the file name stored on the disk so it is advised to keep the names the same.

CS1er will then store the FIAD in memory (RAM), not to disk. The new FIAD will be displayed in the **"Loaded FIADs"** list in the main window. (as in Figure 1.3)

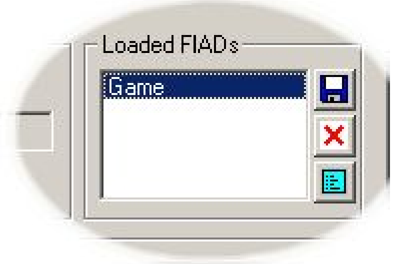
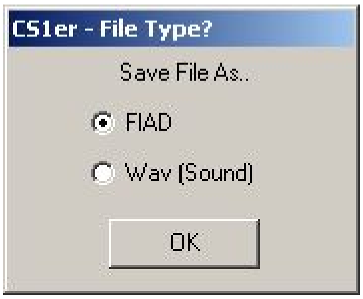


Figure 1.3 - Loaded FIADs List

CS1er will store up to 10 FIADs in memory and will require that they all have different names. Once the FIAD is in memory you may save it as a FIAD on disk or even convert it back to Wave so you have a perfect digital copy of the file. A perfect digital copy could then be used to play back to a real TI Computer or saved to CD for playback to a real TI Computer from a CD player.

To save the file *click* on the required FIAD in the FIAD List and click on the **"Save"** 💾 button. You will then be prompted to select a format to save as. Either Wave of FIAD. (as in Figure 1.4)



CS1er - File Type?

Save File As..

● FIAD

○ Wav (Sound)

OK

Figure 1.4 - File Type Window (Save As)

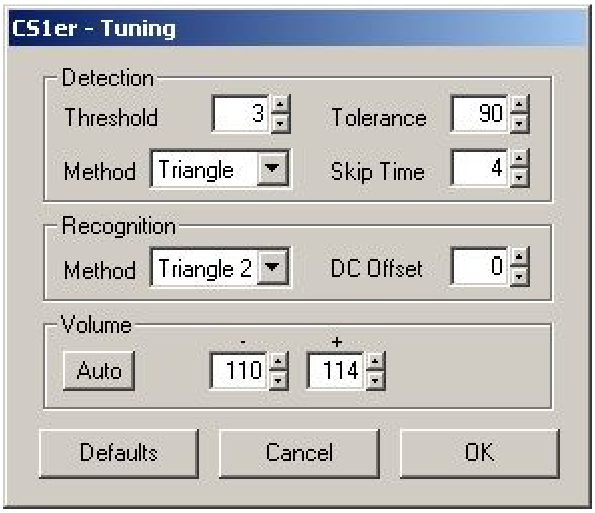
Alternatively, if the file is a TI BASIC or TI Extended BASIC file the you may view the source by *clicking* on the **"View Source"** ▤ button.
You may delete the FIAD from memory by *clicking* the **"Delete"** ✕ button.

Tuning CS1er

The Detection and Recognition engine within CS1er can be tuned to more accurately decode your TI tape file. Due to the different properties of cassette tapes and recorders, plus the effects of time on tape media, some tape files may be difficult to decode. CS1er has incorporated a number of options to allow the decoding engine to be 'tuned' to enable better decoding of the file.
'Detection' refers to CS1er's ability to detect a bit of data within the wave sound, regardless of it's state (on or off / 0 or 1).
'Recognition' refers to CS1er's ability to distinguish the state of a bit (on or off / 0 or 1).
'Decoding' refers to the entire process of 'detection', 'recognition' and assembling of data back into its digital file (FIAD).

The Tuning Window.

To access the Tuning window *click* on the **"Project"** drop down menu and *click* on **"Tuning"**. The Tuning window will then appear. (as in Figure 2.1)



CS1er - Tuning

Detection
Threshold 3 Tolerance 90
Method Triangle Skip Time 4

Recognition
Method Triangle 2 DC Offset 0

Volume
- 110 + 114

Figure 2.1 - Tuning Window

Detection

Threshold - Used in Detection Engine. Threshold is used to determine the maximum sample length a bit could take within a wave file. CS1er has a preset internal length of 33 samples. The Threshold is an overshoot of this value, therefore 33 + *threshold*. So the default would be 33 + 3 = 36. The actual sample length of a bit would rarely exceed 32 samples but the threshold modifier allows the detection engine to continue attempting to detect the correct end of a bit if the engine began sampling before a bit started. The threshold could be considered a 'cut off' value in case the detection engine failed to detect a bit of data. If the playback was faster than original TI then lower this value. If it is slower then increase this value.

Tolerance - Used in Detection Engine. Tolerance is used to determine if bit has been detected. A higher value is more lenient on detecting a bit, but could cause the engine to miss-detect a bit. A lower value is more stringent on detecting a bit, but could cause the engine to not detect a bit forcing more cut-offs from the Threshold value. Have a zoomed look at the wave file in a wave editor. If the wave file seems to vary alot in the waves displayed then a higher threshold may be required. If the wave seems quite uniform then a lower value may be better.

Skip Time - Skiptime sets the number of seconds to skip into the wave file before starting detection. This can be useful to start the decoding process closer to the start of the actual binary sound, which helps stop the decoding process starting prematurely and getting corrupt data. It also speeds up detection. It is good to have this value set so that the decoding starts just after the binary sound begins.

Method - Used to select a waveform for detection independent from the recognition engine. See the recognition engine for more details on waveforms.

Recognition

The Recognition section allows you to customise the Recognition Engine and bit shape data used in recognition. The shape chosen also reflects the shape of the saved sound when saving as a Wav file. By default the 'Natural' bit shape is selected as this is modelled on the actual output from a TI99 Computer.

Figure 2.2 shows a Graphical representation of the different Bit shapes used in recognition and for saving purposes.

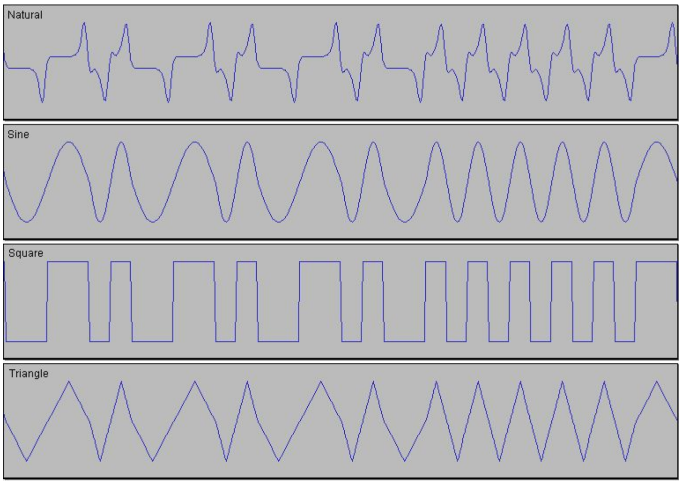


Figure 2.2 - Graphic Representation of Bit shape options

Natural - Modelled on the actual TI99 Computer cassette output.

Sine - Modelled on a Sine Wave at the same frequencies as the TI interface.

Square - Modelled on a Square Wave at the same frequencies as the TI interface.

Triangle - Modelled on a Triangle Wave at the same frequencies as the TI interface.

Triangle 2 - Modelled on a Triangle Wave but with Natural characteristics. Sample frequencies same as the TI interface.

The actual shape chosen depends on the intended purpose. When decoding from a wave file, have a look at the quality of the wave file with a wave editor. Sometimes the sound is quite warped during its life on a cassette tape or from the cassette player. Select the one which more closely matches your source wave file. The Square bit shape is not very suited to decoding but is the choice if you are loading wave files into the MESS emulator because it creates and prefers a square wave. The Triangle bit shape is quite successful in recognising warped sound. Try each to see how successful CS1er is at decoding your source sound.

DC Offset - DC offset uses an alternative and currently experimental recognition routine. It relies on the frequency pulse passing the DC offset regardless of the actual bit shape. This can be particually usefull with square waves obtained from the MESS emulator. There is a lot more work to be done in this area. DC Offset value also repositions the waveshape used in non DC-Offset recognition methods.

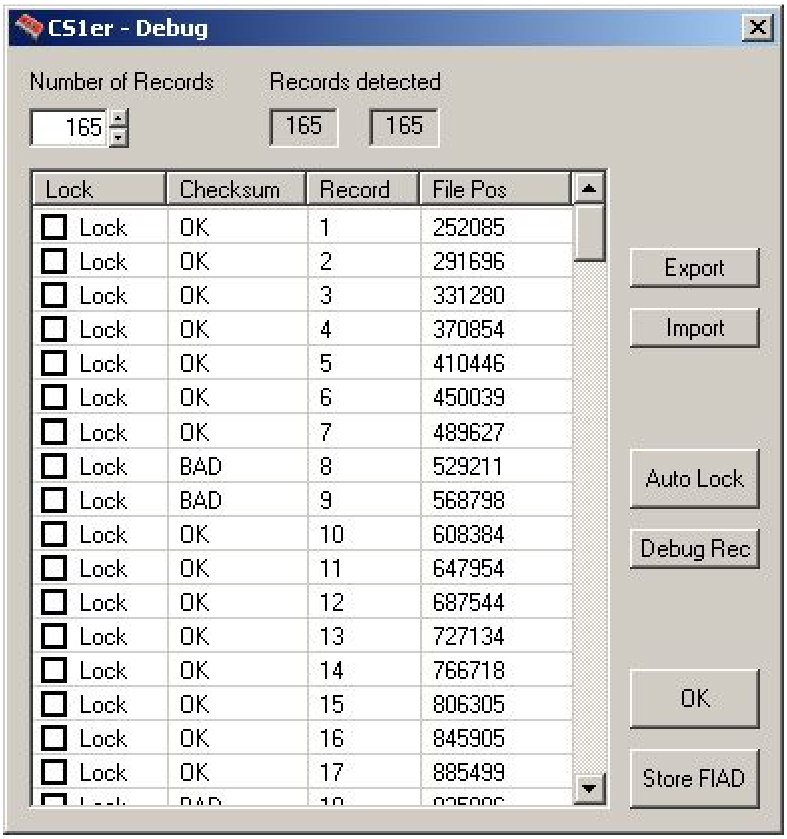
Volume (Detection and Recognition)

The volume setting allows manipulation of the bit shapes so that they will closer match the recorded sound and give a more reliable decode. A 100% setting determines that the bit shapes are already at the exact same scale as the source wave. The **"Auto"** feature is included for automatic detection of the source wave's volume. Having the volume set to exactly that of the wave doesn't always mean that it will give the best result. This is why the auto feature can be turned off to allow manual adaption. When the auto checkbox is unchecked the previously detected auto volume values will be given in the manual fields.

Volume is Dual-Poled either side of DC offset (0 volume). This allows volume changes to both negative (-) and positive (+) volume. This feature was introduced because tape files often have severe DC Bias forcing positive or negative imbalances in the volume.

Debugger

CS1er includes a basic debugging system that allows it to retain copies of good records decoded. On many decoding occasions CS1er may be able to successfully decode many records but not all. It may also be noted that CS1er may decode some records successfully using different tuning settings but not the ones which were successful on previous attempts and alternate tuning settings. To use the debugger simply *check* the **"Debug"** checkbox (on the main window) prior to starting the decoding attempt. When the decoding process finishes the Debug window will appear. (as in Figure 3.1)



Number of Records 165

Records detected 165 165

Lock	Checksum	Record	File Pos
☐ Lock	OK	1	252085
☐ Lock	OK	2	291696
☐ Lock	OK	3	331280
☐ Lock	OK	4	370854
☐ Lock	OK	5	410446
☐ Lock	OK	6	450039
☐ Lock	OK	7	489627
☐ Lock	BAD	8	529211
☐ Lock	BAD	9	568798
☐ Lock	OK	10	608384
☐ Lock	OK	11	647954
☐ Lock	OK	12	687544
☐ Lock	OK	13	727134
☐ Lock	OK	14	766718
☐ Lock	OK	15	806305
☐ Lock	OK	16	845905
☐ Lock	OK	17	885499

Figure 3.1 - Debug Window

The Debug window displays all the records currently collected. The 1st column **"Lock"** displays the locking status of each record. This is used to keep good records so future decodes do not overwrite them. The **"Checksum"** column displays the checksum status of each record stored, whether OK or BAD. Please note that just because the checksum reports as being OK does not mean that this is true. It merely means that the checksum calculation was ok. For example if one byte of data was short by 1 and the next byte was up by 1 then the checksum would be ok. But the stored data would actually be incorrect. Likewise there is a chance the checksum itself is wrong but the data OK. The **"Record"** column shows the current record number on that row. The last column, **"File Pos"** indicates the position where the record begins in the previously scanned file. This is used in the Record Debug mode for locating the start of each record for manual bit recognition.

To use the debugger simply *check* the **"Lock"** checkbox on any good records that you would like to keep. The records not locked will be free to be overwritten during the next decode using alternate tuning settings. Hopefully the alternate settings will be more likely to decode the bad records successfully. The debugger will retain all locked records and also contain the unlocked records obtained during the immediate previous decode attempt. When you are ready to save the finised FIAD simply *click on* **"Save FIAD"**. If the **OK** button is *clicked* the FIAD is not saved to the FIAD list but the debugger still retains all records. To clear the debugger's retained records simply load another wave file and you will be prompted to clear the debug data. If you have more than one copy of a wave file containing the same TI file, then it is not necessary to clear the debug data. You could use many wave files to pull your data back together.

The **"Auto"** button simplifies the painful task of locking many records by locking all records with OK checksums.

Important note: Sometimes CS1er may incorrectly skip a record and end up interpreting incorrect records. For example Record #2 is not recognised and record #3 is stored as record #2. This would cause your resulting FIAD to be corrupt even though all checksums were OK. If you have the status window open, this can be noticed by pauses in the 'bytes' being read. This can cause misleading information for the debugger, so note that it is not fool proof. CS1er now has an inbuilt routine to help prevent the skipping of records. If CS1er detects it has not successfully detected the beginning of a record, it will re-attempt the detection using a number of varing techniques. This is listed in the verbose log as "Bad Datamark" and if all techniques are exhausted the decoding will stop and the log will report "Lost Data Mark!!".

Import and Export

By *clicking* on the **"Import"** and **"Export"** buttons the contents of the debugger can be imported/exported to/from a text file. The format of the exported file follows.

```
;Record copies :165 : 165
;Records Recorded :165
[Record 1]
;0B651D8A16EF3FFF10E0261210D6260B10CC261510C2261B10B81E6C10AE21E010A42
29C109A22D4109022DB10862193107C273010721F5710681F9F105E1FEA
;0B651D8A16EF3FFF10E0261210D6260B10CC261510C2261B10B81E6C10AE20C008521
14E084D116A0848116D884310C9883E139808390FAB88340FCF882F0FF5
 0B651D8A16EF3FFF10E0261210D6260B10CC261510C2261B10B81E6C10AE21E010A42
29C109A22D4109022DB10862193107C273010721F5710681F9F105E1FEA
[Record 2]
;10542736104A330C10402346103620 8F102C203C10223400101831 8A100E346710042
73C0FFA239A0FF0343E0FE632EA0FDC346E0FD234B60FC832250FBE32E5
;10542736104A330C104023461036208F102C203C102234001018318A100E346710042
73C0FFA239A0FF0343E0FE6317507EE1A3707E91A5B07E4191287DF1972
 10542736104A330C104023461036208F102C203C102234001018318A100E346710042
73C0FFA239A0FF0343E0FE632EA0FDC346E0FD234B60FC832250FBE32E5
[Record 3]
;0FB427420FAA34600FA027480F9633130F8C33510F8233A90F78328B0F6E318D0F642
74E0F5A32DB0F5027770F4632E20F3C22E10F321ED10F2835060F1E2CCB
;0FB427420FAA34600FA027480F9633130F8C33510F8233A90F78328B0F6E318D0F642
74E0F5A32DB0F5027770F4632E20F3C22E10F321ED10F2835060F1E2CCB
 0FB427420FAA34600FA027480F9633130F8C33510F8233A90F78328B0F6E318D0F642
74E0F5A32DB0F5027770F4632E20F3C22E10F321ED10F2835060F1E2CCB
[Record 4]
```

..etc......

All lines with a preceeding semicolon "`;`" are ignored. Therefore the first two lines are for reference purposes only.

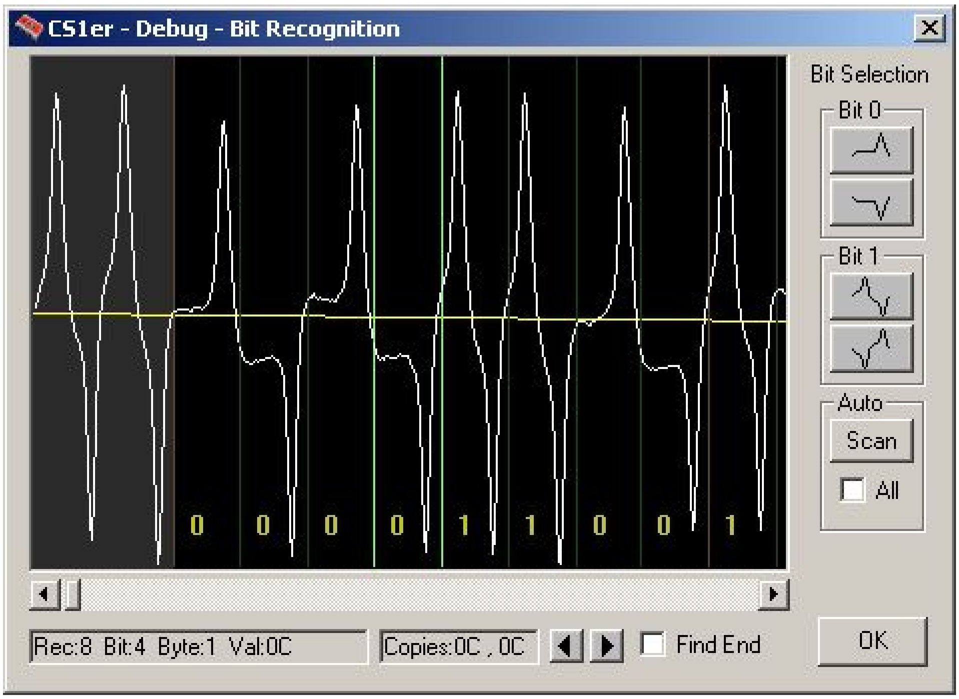
Record numbers are always enclosed in brackets, for example: "`[Record 56]`". Record numbers indicate where the following data will be placed. If no record numbers are included the data will naturally be imported starting at the first record and incremented from there forward.

Not all records have to be imported. Single records may be imported also if the file only contains one specific record.

If the original debug data being exported contains additional copies of the records produced during the decoding of a file they will be included for reference with a preceeding semicolon.

Record Debugger / Manual Bit Debug

CS1er includes the ability to manually debug a CS1 record if the Decoding process fails to extract the correct data. By selecting a record from the Debugger and *clicking* **"Debug Rec"** the "Debug - Bit Recognition" Window will appear. (as in Figure 3.2)



Rec:8 Bit:4 Byte:1 Val:0C

Copies:0C , 0C

Figure 3.2 - Debug - Bit Recognition Window

The "Bit Recognition" Window allows individual manual recognition of bits from the actual wav file. By using the scroll bar at the bottom of the display it is possible to change the viewing position/current bit selection within the wav file. The actual range of the wav file shown is determined by prior debug scans of the file so what is shown in the display is reflected by tuning settings and successful bit detection. By manually moving through the bits in the display, it can be determined where the decoder has incorrectly either recognised of detected a bit. How the recognition engine interpreted the bit is displayed at the bottom of each bit shape.

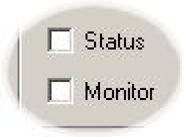
Vertical dark green lines represent the Start/End of a bit shape. Be aware that the end of one bit shape is the start of the next. Vertical orange lines represent the end of 8 bits (a byte). The two vertical light green lines represent the start/end of the currently selected bit shape. Grey areas represent extremes of the currently selected record.

By *clicking* of any of the bit selection buttons the current bit value can be changed to either 0, 1 or get the recognition engine to rescan the current bit. By *checking* the **All** checkbox it will rescan all bits in the current record when **Scan** button is *clicked*. The arrow buttons allow the repositioning of the start position of a bit where the detection engine incorrectly detected a bit position. *Checking* the **Find End** checkbox will run the detection engine to the end of the record to attempt re-detection of all following bits (to the right of current position). Note that this does not re-recognise these bits.

Please note that all changed in the Bit Recognition Window are immediate and allow no cancel or undo.

Status and Monitor

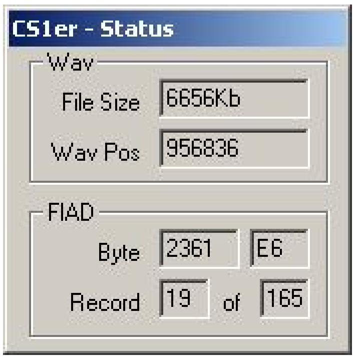
Decoding of a TI CS1 wave file in bad condition can cause many headaches if there is no insight into what's actually going on. The **"Status"** and **"Monitor"** windows can help to guide the better tuning of CS1er for the problematic file. Both windows can be activated by *checking* their respective checkboxes on the main window. (as below)



□ Status
□ Monitor

Please note that the windows will only appear during decoding of the file.

Status Window



CS1er - Status

Wav
File Size 6656Kb
Wav Pos 956836

FIAD
Byte 2361 E6
Record 19 of 165

Figure 4.1 - Status Window

File Size - Size of current wav being scanned.
Wav pos - Current 'read' position within the wav file.
Current Bit - Value of current bit recognised (0 or 1) and the phase of the bit (-1 or 1).
Vol Level - Detected volume level.
Byte - Current number of bytes read (in total) from file and its value.
Record - Current record being read and total number of records detected in file.

Montor Window

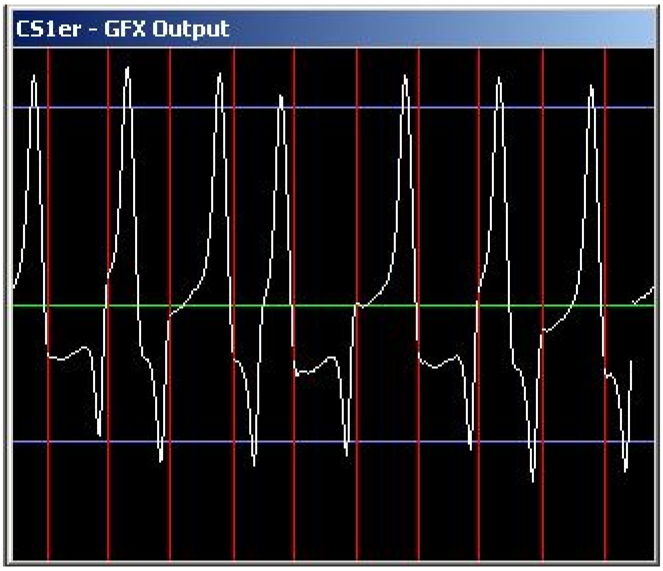


Figure 4.2 - Monitor Window

White line - Actual wave file sample values.
Green line - Zero volume reference for DC offset.
Red Lines - Point where CS1er has detected the end of a bit
Cyan (light blue) lines - Indicate volume detection. Top line indicates Negative (-) volume. Bottom line indicates positive (+) volume.

Links and References

- The TI-99/4A Tech Pages
  http://www.nouspikel.com/ti99/
  I believe this is the best source of tech info for the TI. CS1er would not exist without it! Thanks Thierry Nouspikel.
  For more info on the TI Cassette Interface check out this page.

- TI-99/4A Home Computer Page
  http://www.99er.net/
  Probably one of the best sources of up to date info regarding the TI.

- Mainbyte's Home of the Texas Instruments Computers
  http://www.mainbyte.com/ti99/

- TI-99/4A & Geneve 9640 Web Site
  http://ti994a.thebbs.org/

- The FABbnet TI-994/a Pages
  http://www.fabbnet.net/ti99.htm

- TI 99/4A Game Shelf
  http://tigameshelf.net/

- The CS1er Website
  http://www.cs1er.com/
  My personal website dedicated to the CS1er and the TI.

Document created using Notepad++ and converted to PDF using OpenOffice
Special thanks goes to Aimee for her patience during the development of CS1er.