



The Easy way to program tones for the  
Genave Communications Signal Processor

# EASYTONE™



Tech Pub. 9000-0000-006

Installation and Instruction Manual for the  
Easytone software.

# Easytone Ô User's Manual, Version 1.3

Ó Genave/NRC, Inc., 1995 - 2001 All Rights Reserved

Phone 651-460-6616

Fax 651-460-6686

## PRINTED IN USA

The contents of this manual and the associated Easytone software are the property of Genave/NRC, Inc. and are copyrighted. Any reproduction in whole or in part is strictly prohibited. For additional copies, please contact Genave/NRC, Inc.

### Warranty:

**Genave/NRC, Inc. products are warranted to be free from defects in material and workmanship for a period of ONE (1) year from the date of shipment. Genave, during this period, will repair or replace any parts which upon our examination appear to be defective in materials or workmanship. This warranty does not apply to defects, malfunctions or breakage due to improper installation, servicing, handling or use or misuse thereof, nor to units that have been damaged by lightning or other "Acts of God" or equipment damaged in Acts of War.**

**Prior to returning equipment for warranty repair, please contact the Genave Customer Service Department for an RMA number. They can be reached by using the telephone number or fax number listed above.**

Genave/NRC, Inc. (Genave) and its licensors offer this warranty in lieu of any and all other guarantees or warranties, either express or implied, including without limitation the implied warranties of merchantability and fitness for a particular purpose, regarding hardware or software. Genave and its licensors do not warrant, guarantee or make any representations regarding the use or the results of the use of the software or hardware in terms of its correctness, accuracy, reliability, results, performance currentness or otherwise. The exclusion of implied warranties is not permitted by some jurisdictions. The above exclusion may not apply.

In no event will Genave, its licensors, directors, officers, employees or agents (collectively Genave's licensor) be liable for any consequential, incidental or indirect damages (including damages for loss of business profits, business interruption, loss of business information, and the like) arising out of the use or inability to use the software or hardware even if the Genave and/or its licensor has been advised of the possibility of such damages. Because some jurisdictions do not allow the exclusion or limitation of liability for consequential or incidental damages, the above limitations may not apply. Genave and its licensors liability for actual damages from any cause whatsoever, and regardless of the form of the action (whether in contract, tort, (including negligence), product liability or otherwise), are expressly excluded.

Genave reserves the right to make changes in specifications at anytime and without notice. Genave cannot be responsible for typographical or other misprints. The information furnished by Genave is believed to be accurate and reliable, however, no responsibility is assumed by Genave for its use, nor infringements of patents or other rights of third parties resulting from its use. No license is granted under any patents or patent rights of Genave/NRC, Inc., its licensors or suppliers.

### Life Support Policy:

Genave/NRC, Inc. products are not authorized for use as critical components in life support devices or systems without the express written approval of the president of Genave/NRC, Inc. As used herein:

- 1) Life support devices or systems are devices or systems which, (a) are intended for surgical implants into the body, or (b) support or sustain life, and whose failure to perform, when properly used in accordance with instructions, can reasonably be expected to result in a significant injury to the user.
- 2) Critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

CSP, Communications Signal Processor, Genave Operating System, GOS, CSP-105, CSP-107, CSP-108, CSP-120 and Easytone are Trademarks of Genave/NRC, Inc. The Genave name and logo are Registered trademarks of Genave/NRC, Inc. Touch-Tone is a Registered trademark of American Telephone and Telegraph. Other names used in this manual are trademarks of their respective companies.

# Table of Contents

<b>Introduction .....</b>	<b>1</b>
Tone Signaling Basics .....	2
DTMF .....	2
Tones: .....	3
<i>Two-Tone</i> .....	3
<i>Single/Burst Tone</i> .....	5
<b>Tutorial.....</b>	<b>6</b>
Software Installation: .....	6
Starting Easytone:.....	7
Job Description:.....	8
File: .....	9
Actions:.....	10
Message:.....	12
Tones:.....	14
Link .....	17
Ideas.....	18
<b>Program Overview: .....</b>	<b>20</b>
<i>Help:</i> .....	20
<i>Save:</i> .....	20
<b>Files:.....</b>	<b>21</b>
Open: .....	21
Save: -F2.....	21
Lock.....	22
Save As:.....	22
Close .....	22
Reset:.....	23
Delete:.....	23
Print .....	23
System: .....	25
Exit: .....	25
<b>Signals: .....</b>	<b>26</b>
Tones:.....	26
Decode:.....	26
<i>Tone A Frequency:</i> .....	26
<i>Tone A Time:</i> .....	26
<i>Tone B Frequency:</i> .....	27
<i>Tone B Time:</i> .....	27
<i>Message:</i> .....	27
<i>Action:</i> .....	28
Encode: .....	29
<i>Pre-Message:</i> .....	29
<i>Pre-Action:</i> .....	30
<i>Tone A Frequency:</i> .....	30
<i>Tone A Time:</i> .....	30
<i>Tone B Frequency:</i> .....	30

<i>Tone B Time:</i> .....	31
<i>Post Message:</i> .....	31
<i>Post Action:</i> .....	31
DTMF: .....	32
Decode: .....	32
<i>DTMF Code:</i> .....	32
Variable Codes .....	33
Wildcards.....	34
Non-Matched DTMF Codes .....	34
<i>Message:</i> .....	35
<i>Action:</i> .....	35
Encode: .....	36
<i>Pre-Message:</i> .....	36
<i>Pre-Action:</i> .....	37
<i>DTMF Code:</i> .....	37
Variable Codes .....	38
<i>Post Message:</i> .....	38
<i>Post Action:</i> .....	39
<b>Messages: .....</b>	<b>40</b>
Screen Markers .....	41
<b>Actions:.....</b>	<b>42</b>
<i>Name:</i> .....	42
Start-up .....	43
No-match.....	43
<i>Action Program Code:</i> .....	43
<b>Inputs: .....</b>	<b>46</b>
Contact Bounce.....	47
<i>Message:</i> .....	47
<i>Action:</i> .....	47
<b>Link: .....</b>	<b>50</b>
Link: .....	50
Save: .....	51
Send: .....	52
Re-Send: .....	52
View: .....	53
Interactive: .....	53
<b>Options .....</b>	<b>55</b>
System Options .....	55
Serial Port.....	55
File Editor .....	56
New File Defaults .....	56
Target .....	56
Mark Time .....	56
Space Time .....	57
Decode Delay .....	57
Bandwidth.....	57
Low Bandstop .....	58
High Bandstop.....	58
LCD Screen Rows.....	58

LCD Screen Columns .....	58
Program Memory .....	58
DTMF & Sw Memory.....	59
Link Options .....	59
Include All Functions.....	59
Include All Messages .....	60
Clear Memories.....	60
Lock Memory.....	60
Beep on Upload.....	60
<b>Commands .....</b>	<b>61</b>
Command: ?- Inquire .....	61
Command: A- Aux Codes .....	61
Command: D- DTMF Generate .....	62
Command: d- Dtmf Dissect .....	63
Command: H- Hold.....	64
Command: I- If .....	64
Command: J- Jump.....	67
Command: j- Jump, Multi.....	68
Command: l- Leave-if .....	69
Command: N- Variables .....	69
Command: P- Inputs.....	71
Command: Q- Timers.....	72
Command: q- Timers, Control.....	73
Command: R- Outputs .....	74
Command: S- Say .....	76
Command: T- Generate Tones .....	77
Command: U- Until.....	78
Command: V- Variable .....	79
Command: W- While .....	80
Command: X - Variable .....	81
<b>Appendix A .....</b>	<b>82</b>
Main Codes .....	82
Aux Codes.....	84
“V” Variables.....	86
“N” Variables.....	88
“X” Variables.....	89

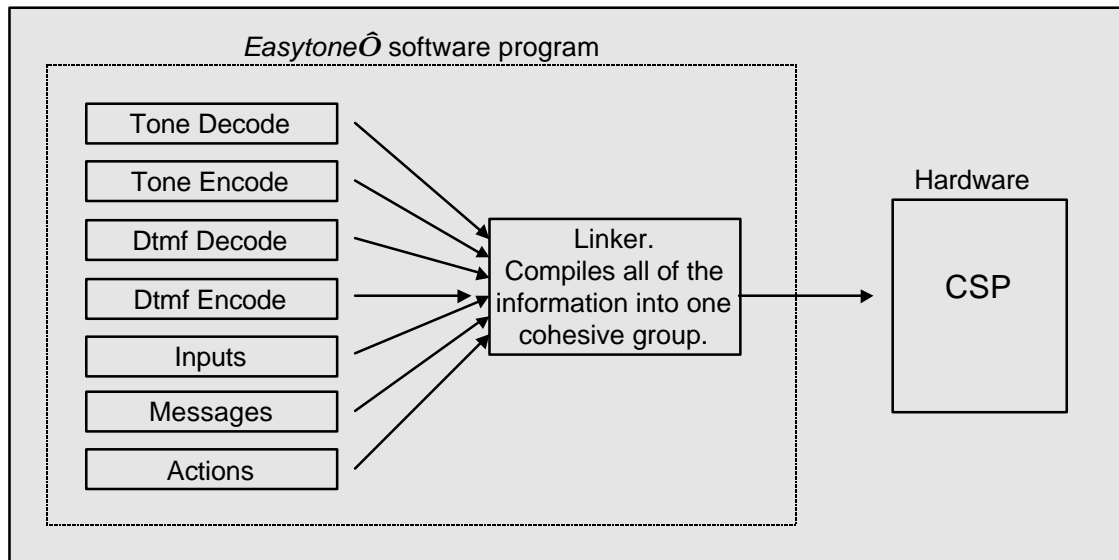
## Introduction

The Easytone™ computer program is an easy to use method of programming the Genave Communications Signal Processor™ (CSP™).

The CSP runs on the Genave Operating System™ (GOS™). GOS is a computer language that was specifically designed for use with communications signaling protocols.

Easytone harnesses the power of GOS in a straightforward manner. Easytone is a “fill in the blanks” method of encoding and decoding communications signals. It allows you to quickly set up the CSP for your particular application.

Easytone is technically called a compiler. It takes the information that you supply when you fill in the Easytone blanks and changes it to the GOS format. This changeover is very fast, usually taking less than 5 seconds to complete.



The linked information is sent to the CSP's memory where it is stored. After you have programmed and tested the CSP, it may be disconnected from your computer. The CSP can now function as a stand-alone device, controlling all of its own functions based on the program you developed.

## Tone Signaling Basics

### DTMF

Dual Tone Multiple Frequency -DTMF signals are used in many communications applications. The most prominent use is Touch-Tone ® telephones.

When used in other communications signaling such as two-way radio, microwave, etc., a string of DTMF numbers is sent as a group. The decoder that receives the code string makes a determination from the content of the code, exactly what operation to perform. Different codes could turn an output on or off, display messages, cause the unit to reply with field information, etc.

There are 16 DTMF codes that can be detected and generated. They are the normal 0 to 9 that you find on your telephone, as well as the special characters # and \*.

Additionally there are four more digits that are used for special signaling, that you will not generally find these on your standard telephone but are used in codes for communications systems. They are designated as A, B, C and D characters.

Each DTMF character is made up of two separate tones known as the Row and the Column tones. The following matrix shows the tones generated for each character.

				Row Tones
1	2	3	A	697 Hz
4	5	6	B	770 Hz
7	8	9	C	852 Hz
*	0	#	D	941 Hz
				1209 1336 1477 1633 Hz
				Column Tones

When a DTMF character is sent, it is composed of both the Row and Column tones that are generated at the same time. If, for example, you were to send the digit 1, the tones 697 Hz and 1209 Hz would be generated.

The decoder picks up the two tones, filters them back into their two groups, and decides that they represent the digit 1.

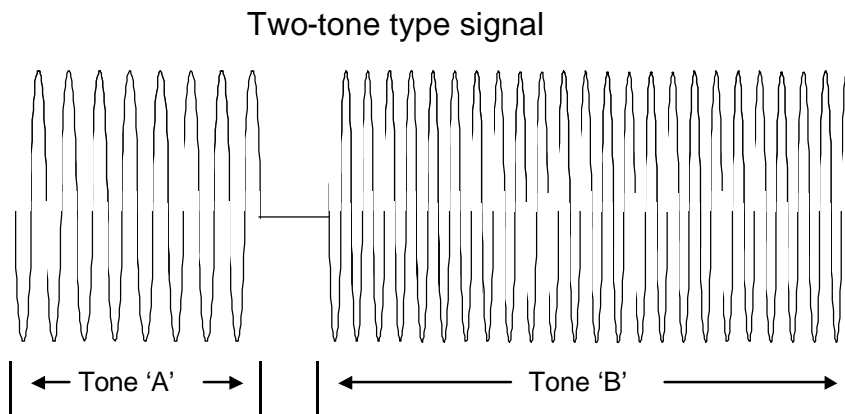
## Tones:

Tone signaling in various formats is used in communications for purposes such as activating pagers, turning on/off equipment, switching repeaters, emergency sirens, remote controlling of equipment, sending transmitter tones, Morse code, etc.

Although tone signaling does not have the flexibility that DTMF code addresses allow, it has been in use for many years and operates well for functions that do not require the addressable nature of DTMF.

Most tone signaling used in communications consists of a single tone generated for a minimum length of time, or two tones that are generated in sequence. The two tones are different in frequency and each has its own minimum length of time. There are also other signaling codes such as five tone, six tone, etc. However, it is the single and two tone types that will be discussed here.

### Two-Tone



The Two-tone signal is composed of two separate tones that are generated one after the other. The first tone generated is called the 'A' tone, while the second tone is referred to as the 'B' tone.

Though most formats go from the A tone to the B tone with no space, there are some formats that do contain a minimum period between the A and B tones where no tones are generated. This is referred to as the 'inter-tone time'. The figure above shows such an inter-tone time. The CSP can decode tone formats with or without the inter-tone time.

There are two items necessary for either an A or a B tone to be valid.

- Frequency
- Duration

A tone must be the correct frequency or must fall within a specified set of limits called a “Bandpass” which lies above and below the correct frequency. In addition, the length of time that the tone is present, is important. The tone must be present for a minimum length of time before it is considered valid.

If a tone is the correct frequency, but it is not present for a long enough time, it is considered invalid.

The A tone timing can (and usually is) different from the minimum timing of the B tone. This is done to prevent simpler decoders from being confused regarding which tone is being presented to them.

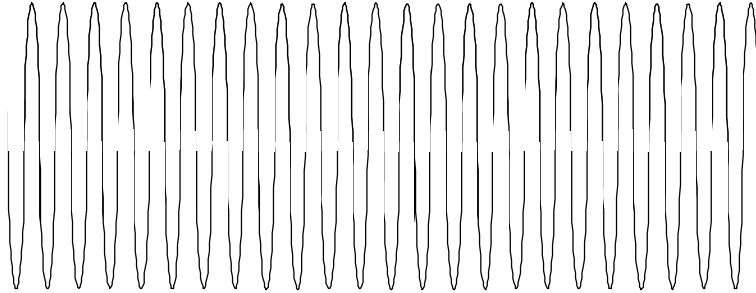
Reviewing the above information, you can see that a two-tone signal must meet four criteria to be considered valid;

- Correct ‘A’ tone.
- Duration of ‘A’ tone meets or exceeds minimum time for format.
- Correct ‘B’ tone.
- Duration of ‘B’ tone meets or exceeds minimum time for format.

If any of the four criteria are not met, the tones are considered invalid by the CSP and they are ignored. The CSP will then continue to monitor for valid tone groups.

**Single/Burst Tone**

## Single/Burst type signal



The Single and Burst tones are individual tones. The difference between them is that a Burst tone is generally less than 1 second in length, while the Single tone is 1 second or longer in duration. For the sake of discussion, we will refer to both of them as ‘Single’ tones.

The single tones are like the two-tone signal, but lack a ‘B’ tone. The single tone is one frequency that is generated for a minimum duration.

The single tone must be the correct frequency. In addition, the length of time that the tone is present, is also important. The tone must be present for a minimum length of time before it is considered valid.

If a tone is the correct frequency, but it is not present for a long enough time, it is considered invalid. Any invalid single tones are disregarded by the CSP which then continues monitoring for valid tones.

## Tutorial

---

---

The CSP is easy to setup and use. This step-by-step tutorial will take you through the entire process of programming a CSP with a sample application.

### **Software Installation:**

---

---

The *Easytone* software program can operate either from a floppy disk, or from a hard drive. If possible, use a hard drive, it will speed up the program screen changes.

For the tutorial, we will install the software onto your hard drive and run the *Easytone* program from that point.

Create a new directory on your hard drive. It will be called CSP.

- **MD \CSP**

Change to the new directory.

- **CD \CSP**

Insert the Easytone program disk into the floppy disk drive and copy the information from the floppy to the hard drive.

- **COPY A:\*.\*** if the floppy is drive A
- or
- **COPY B:\*.\*** if the program disk is in drive B.

After your computer has loaded the program onto your hard drive, you should remove the floppy disk and store it safely away.

## Starting Easytone:

Start the program by typing **Easytone**. After a few seconds the program will begin and show you the opening screen.

If your computer has an LCD or monochrome screen, you may want to start the program by typing **Easytone /M**. This will strip the normal colors from the program, and will instead display in high contrast black and white.



At the top of the screen are the possible selections that you can make, they are called the “Menu Options”. To choose one of the menu options, use the arrow keys to move the highlighted selection box to your choice, and press Enter.

If you are in another part of the Easytone program and want to return to the Main screen, use the Esc. key to “back out” of any options that you are working in.

To Exit the Easytone program, you can either select “eXit” on the menu bar with the arrow keys and press Enter, or you may press the “X” key from the main screen. Either one of these two methods will exit you from the program and return you to the computer's operating system.

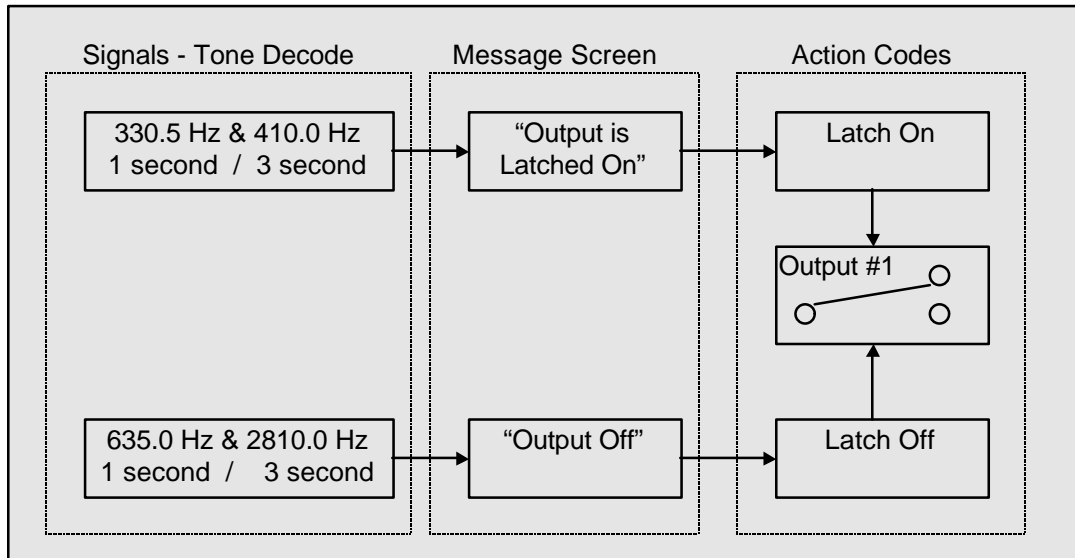
## ***Job Description:***

Before proceeding further, you will need to identify exactly what you want the CSP to do.

For the purpose of the tutorial, you will program the CSP to act as a tone decoder that will latch and unlatch a relay, as well as give a message via the LCD screen.

The tones 330.5 and 410.0 will be used to turn the output on, while the tones 635.0 and 2810.0 will turn the output off.

In both cases, the tones codes will use one second/three second timing.



The diagram above gives a brief overview of the program you will be entering. Information will be entered into each of the respective areas, Tone-Decoding, Message and Actions.

The linker will then gather all of the information and send it to the CSP's on-board memory where it will be stored.

---

**File:**

---

You will need to start a file on your disk using a name that will be particular to this application.

- Highlight the **File** option on the Main menu using the arrow keys, then press Enter.
- Next select the **Save As** option and press Enter.

This will bring up a screen that requests a filename to use.



- Type **TEST** and press the Enter key.

Easytone automatically adds the .CSP to the end of the name as it is opening a file on you disk. The upper left side of the Easytone screen will now show the name of the file you are working with, TEST.CSP.

Pressing the Esc. key will return you to the Main screen.

---

## **Actions:**

---

It is usually best to begin by programming the information for the Actions.

- Select the **Actions** option on the main screen and press Enter.

The Actions screen will now appear. There are two columns that you will use while defining an Action. The “Action Name” which is a short name that describes the function of the action, and the “Action Code” which is the program code that the CSP actually uses to run your commands.

You will first enter the information to turn the output on.

- Press **F3** to jump to the Name column.
- Type **Latch On** in the first line, memory slot 40.
- Press the **F4** key to move the cursor back to the Action Code column.
- Type **R1=1;**

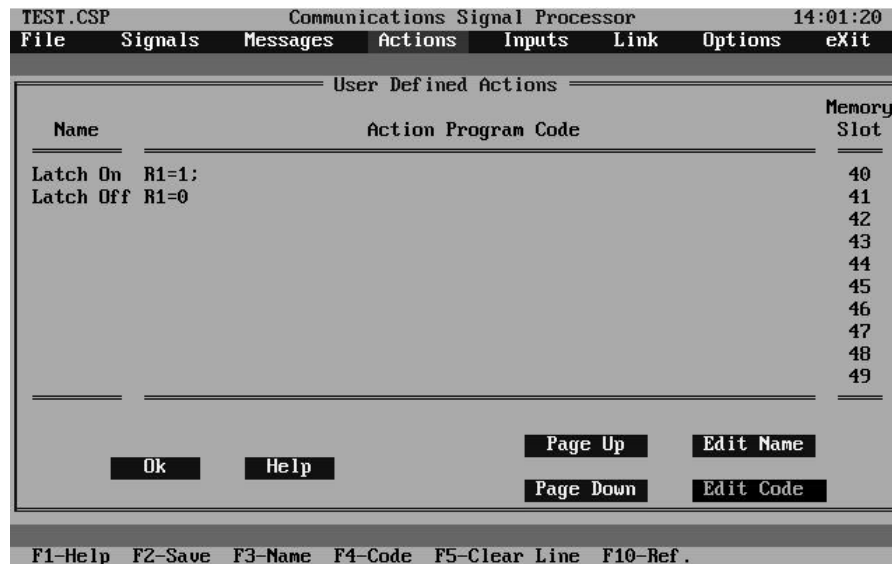
The R1 instructs the CSP to use output number one. While the =1; completes the statement and means ‘turn the output on’.

You will now enter the information to turn the output off.

- Press the **Down Arrow** key to move down one line.
- Press **F3** to jump to the Name column.
- Type **Latch Off** on the second line, memory slot 41.
- Press the **F4** key to move the cursor back to the Action Code column.
- Type **R1=0;**

The R1 instructs the CSP to use output number one. While the =0; completes the statement and means ‘turn the output off’.

You now have entered all of the Action information, and the computer screen should match the one below.



You will periodically need to save the workfile information you have entered, to the disk.

This can be done by either selecting the *File* option from the Main screen, and then selecting *Save*, or you can use the shortcut method.

The shortcut is to press the F2 function key. If you look at the bottom of the screen, you can see the words “F2 Save”. When you press the F2 key, all of the information in your workfile is saved to the disk.

- Press the **F2** key, then press the Enter key to save your work to the disk.

You can now return to the Main screen by pressing the Esc. key.

## Message:

You will now enter the information that will display on the LCD screen and be sent to the serial communications port when the tones are successfully decoded.

- Select the **Messages** option on the main screen and press Enter.

The Messages screen will now appear. You will see ten of the possible thirty messages displayed on the screen at any time.

To view the other messages, highlight the page up or page down buttons and press Enter. A new screen will appear with the next group of ten messages.

You will now enter the two messages to be displayed.

- Type “**Output is latched on**” into message line number one.
- Press the **arrow down** key, you will now be at message line two.
- Type “**Output Off**”.

Both messages have now been entered. Your computer screen should look like the example below.



- Press the **F2** key, then press the Enter key to save your work to the disk.

You can now return to the Main screen by pressing the Esc. key.

## Tones:

You now need to enter the Tone frequency and timing information for the decoder. These numbers will be used to compare against the received tones to search for a match.

To enter the Tone decoder screen.

- Select the *Signals* option on the main screen and press Enter.
- Select the *Tones* option on the main screen and press Enter.
- Select the *Decode* option on the main screen and press Enter.

The cursor will start at the “Tone A Frequency” field.

- Enter the ‘A’ frequency of **330.5** and press the **arrow right** key.

The cursor will now be at the “Tone A Timing” field.

- Enter the value of **.7**,

The .7 seconds is the 1 second first tone time minus .3 seconds. This difference allows for variations in encoder timing, keyup delay, etc.

- Continue entering the ‘B’ frequency of **410.0**, with a time of **2.7** seconds.

The screen cursor should now be on the Message column. Pressing the Enter key will open a selection window which will display all of the messages that have been entered.

- Use the arrow keys to highlight the message “**Output is latched on**” and press Enter

The selection window will disappear and “Output is latched” will appear in the message column. Due to space constraints, only the first fourteen characters of a message are displayed.

- Use the arrow keys to move to the Actions column, press the Enter key.

A selection window with the Action Names will appear.

- Select “**Latch On**” and press the Enter key.

The Action column will display “Latch On”.

- Using the arrow keys, move to the next line and enter the data for the next group of tones, messages, and actions.

“A” Frequency:       **635.0**

“A” Minimum Time: **.7**

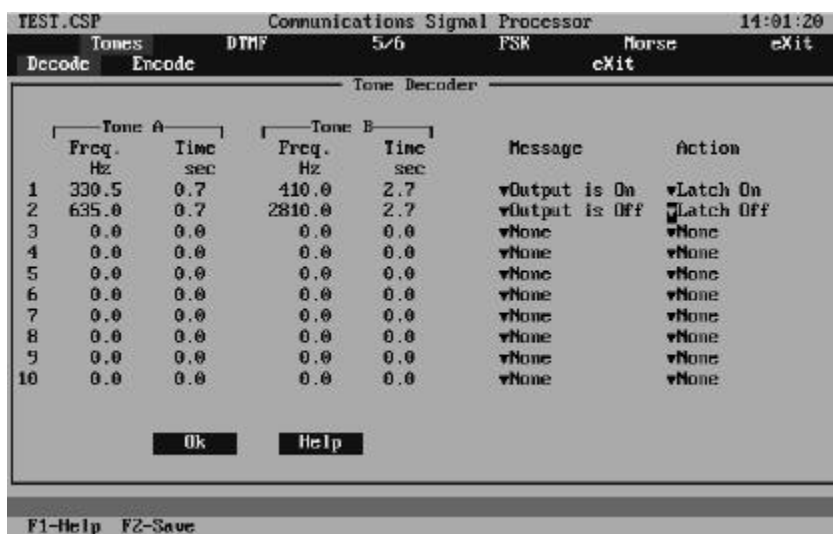
“B” Frequency:       **2810.0**

“B” Minimum Time: **2.7**

Message:               **“Output Off”**

Action Name:         **Latch Off**

Your screen should now look like the example below.



- Press the **F2** key, then press the Enter key to save your work to the disk.

You have completed the data entry for your job. It is now time to link the data and send it to the CSP.

Return to the Main screen by pressing the Esc. key.

## **Link**

With all of the information entered into the Easytone program, it is now time for Easytone to gather the data and send it to the CSP.

With no power applied to the CSP, attach the RS-232C “Serial Cable” between your computer serial communications port number one, and the CSP backplane. *Note: if serial port one on your computer is already in use, then attach the cable to serial port number two. Then go to the Main Menu and select the Options/System menu. Switch to serial port 2, and return back to the link screen and continue.*

The CSP uses a 9 pin “AT” type connector. If your computer uses a 25 pin type of connector you will need to use an adapter that changes from the 25 pin to the 9 pin standard.

Apply power to the CSP.

To enter the Link screen.

- Select the **Link** option on the Main screen and press Enter.

The Send option will be highlighted. This command will link the file, store the linked information to the disk, and then send the compiled data to the CSP where it will be stored in the CSP’s memory.

- While the **Send** option is highlighted, press Enter to start the link process.

As Easytone links the data, the screen will keep you advised of the progress.

Additionally, while the CSP is receiving data, you will notice that the LED is blinking as the data is being received.

After all of the information is sent to the CSP, the link screen clears and returns to normal.

Your screen should look something like the example below.

```

TEST.CSP                               Communications Signal Processor          14:01:20
Link  saUe  Send  Re-send  View  Interactive  eXit
Link and transmit the data to a CSP
X22=50:
X17=10:
X18=250:
X19=4000:
X20=6:
A55,55:
A51,1:330,7,410,27,102
A81:{102}j140,40:
A51,2:635,7,2810,27,103
A81:{103}j141,41:
A86:S1:102
A86:S2:103
A81:{140}A33:SOutput is On:
A81:{141}A33:SOutput is Off:
A81:{40}R1=1:
A81:{41}R1=0
The Main Program size compiles to 96 bytes.
The DTMF and Inputs require 14 bytes of memory space.

F1-Help  F2-Save

```

You may now turn the power off to the CSP, and remove the programming cable.

The CSP now contains the information you supplied. You should now verify its operation by sending it the tones and checking its outputs. When you are satisfied of its operation, it is ready to begin work.

A CSP can be reprogrammed at any time. You simply enter the information into the respective screens and perform another Link-Send.

New information will overwrite the old information in the CSP.

## Ideas

This may be the end of the tutorial, but it is only the beginning of your introduction to the power of the CSP. With Easytone, you can quickly come up with solutions to Tone and DTMF signaling problems that have been impossible to address before.

To get your imagination going, here are a few programming tidbits. Try designing ...

- 1) A decoder for a Fire Department that automatically opens the correct fire doors, depending upon which paging signal it decodes.
- 2) A two-tone decoder that, for security reasons, will only activate when the tones are received twice within a thirty second period.
- 3) A site alarm that monitors the inputs, and sends a DTMF code when an alarm condition occurs. With the memory capability, the CSP can easily count how many times

the alarm has been tripped, and can refrain from transmitting additional alarm signals within a time period to avoid tying up the channel.

- 4) An advanced remote channel switching device. The CSP can go through several steps to switch transmitters, antennas, power sources, etc., with only one signal.
- 5) Siren system decoder. With all of the timers and intelligence on board. The CSP can do the signal decoding, and also any motor cycling action that is needed.
- 6) Dual system decoder. Since the CSP monitors both DTMF and Tone signals at the same time. A system can be controlled from two different activation points, with different encoders. The CSP makes any differences totally transparent to the equipment.
- 7) The CSP has very high noise immunity. Decoders that suffer from interference from background noise or open microphones can be replaced by the CSP with excellent results.
- 8) Rolling Code Security. The digital nature of the CSP allows codes to be changed with every action. Both for Encoding and Decoding. This ability is not limited to DTMF. You can actually program the CSP to change the two-tone signal with every action.
- 9) Simple to Complex encoders can be programmed with ease. The on-board LCD driver gives you the ability to prompt the operator through a menu selection if necessary.
- 10) Double press encoder. Using the on-board timers, you can require that the operator press an activation button twice within, for example, 30 seconds before the signal is sent. This avoids accidental activation, but still only requires one finger to operate the encoder.
- 11) A very low cost SCADA system. Using CSP's at both the main control point, and field units. Information can be gathered and control signals issued with minimum effort. The serial communications port can be attached to a printer to log any action or activity.
- 12) ANI status display. Since the CSP can decode most formats, it can provide an enhanced ANI status display, including maps.

These are just a few of the many, many applications that are now possible due to the power of the CSP.

## Program Overview:

---

Easytone is a menu driven program. It is divided into areas for signal decoding, signal encoding, messages, input actions, output actions, etc.

A choice is made by highlighting a selection with the right/left arrow keys, then pressing the Enter key.

If you have a mouse, you can use to move the cursor to an area then press the left mouse button.

To exit an area, press the Esc. key, or use the right mouse button to exit.



### **Help:**

Easytone has a built in "Help" file that will display context sensitive help anytime you press the F1 function key on your computer keyboard.

### **Save:**

To allow you to save your file quickly, F2 function key on your keyboard has been reserved for this purpose.

Any time the F2 key is pressed, the file that you are working upon will be updated with any changes you have made. (Please see Save: -F2 on page 21 for additional information).

---

## Files:

---

When the Easytone program is first started, it will either load the last file that you were working on, or it will load the NONAME.CSP workfile. The Noname.csp workfile is a default name and should not be used to save information.

The name of the workfile will be displayed in the upper left section of the screen.



---

### Open:

---

This selection will display a list of files that have already been developed and saved.

All of these files will have a .CSP extension.

To select a file, use the up/down arrow keys to highlight the name of the file you want to open, then press Enter. The OK button will be highlighted. Press Enter again and the file will load.

When the file is loaded, its name will be shown in the upper left section of the screen.

---

### Save: -F2

---

This updates the file information on the disk. The changes that you make to the file are stored in memory until you save the information to the disk. If you make changes but do not save them to disk, then these changes will be lost when you exit the program, or if you lose computer power.

When you save a file to the disk, you will be asked to verify the save command before the old data on the disk is overwritten with the new data.

---

## **Lock**

---

Easytone provides the ability to “Lock” a file. This is used to prevent easily overwriting a project file that you want to preserve as a template for future projects.

When you have finished entering the Action, Message and all other information for the project. You select the “Lock” item and press enter. The notation “Locked” appear on the screen to remind you that the file is locked.

When you attempt to save data to a “Locked” file, Easytone prompts you with the same screen used in “Save As...”. Thus you are required to save the new information or any changes to the locked file using a new filename.

To unlock a file, use the “Lock” menu item again and the file will unlock and allow you to make changes as before.

---

## **Save As:**

---

This command will save the workfile data in the memory under a new name. You enter the new file name, and the computer will supply the .CSP suffix.

The computer will check to see if a file with that name already exists. If it does exist, the program will ask for verification of the command before it overwrites the old data with the new information.

The “File Description” area is for your use to enter detailed information about the purpose of the file, any special notes, or installation instructions. This information is also printed when you use the Print command from the menu list.

---

## **Close**

---

When you are done editing a file, you can Close it using this command. When you “Close” a file, Easytone returns to the default “Nofile.csp” as the filename.

You should make sure that you save your work prior to closing any file.

It is not necessary to close a file when you are done with it, however if you prefer to have Easytone start with “Nofile.csp” it will be necessary to close files when you are done with them.

---

**Reset:**

---

Use this command to clear out all the data that may be in memory and zero out all frequencies.

New clears out the information in memory only. You will need to Save to disk after the New command, if you want to clear the information on the hard disk.

Once you use the New command and Save to disk, all information that may have been stored under the filename is erased and cannot be restored.

---

**Delete:**

---

To erase a file on the disk that you no longer need, use the pick file menu, select the filename, then press the Enter key two times. The file will be erased.

Besides erasing the .CSP file, the Delete command will also search out and erase files with the same filename but also have the .PRN, .MAP, and .LNK filename extensions.

---

**Print**

---

You may print the contents of the working file at any time. All of the information contained in the Tone, DTMF, Inputs, Action, Message and Options screens will be printed.

Printing does not alter any information that you may have entered. It simply sends the data to your printer or a file so you can examine it further, fax it, give it to a another person, or store it away for archival purposes.

To print the information, select the print menu option and press enter.

You are now presented with a smaller window that prompts you to select where you want the information sent to. There are three options.

- Printer number 1 - (LPT1)
- Printer number 2 - (LPT2)
- File

Most computers come equipped with LPT1 as part of their standard configuration. So this is the first selection highlighted.

If your computer has the provision for a second line printer, this is usually attached as LPT2. Not many computers have this extra option.

Your third choice is to print the information directly to a file on your disk. This is used when you do not have a printer hooked to your computer, or if you intend to transmit the information via modem, digital fax, etc. where it is easier to perform when the data is already stored in electronic form.

When using the File option, a new file is created on the disk where the program resides. The name of the print file is the same name as your work file, except the .CSP is changed to .PRN to denote a print file.

If, for example, your work file was called "SIRENS.CSP", the print file would be named "SIRENS.PRN".

If a print file has previously been created, it will be overwritten with the new information.

A print file contains the information in ASCII format. This is the most basic text structure, and allows you to use most any word processor to view, edit or send the resulting file.

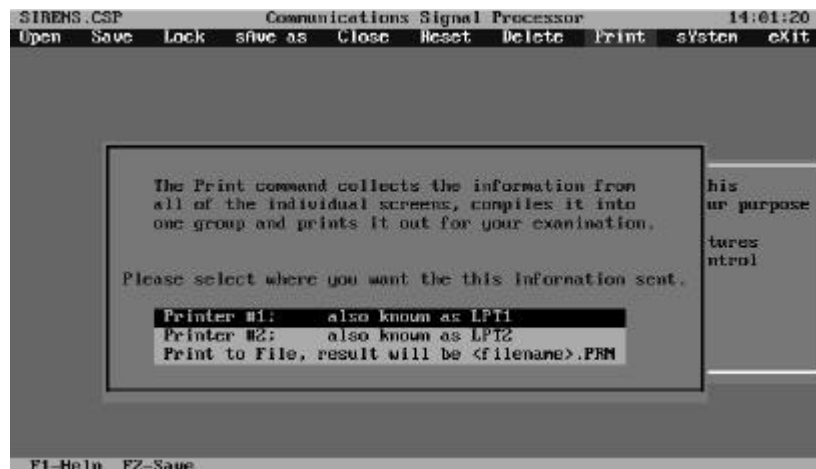
You can also print the file at a later time by using DOS commands. After exiting the Easytone program, make sure you are in the same directory as the file.

You now enter

**TYPE <filename.prn> > LPT1**

The <filename.prn> is the name of the print file. Using the above example you would have entered TYPE SIRENS.PRN > LPT1. This would have sent the file directly to your printer attached to printer port number 1 (LPT1).

Next select where you want the information sent to. Using the arrow keys, highlight your selection and press the Enter key.



You are now prompted to make sure the printer is ready to go (if you are using the printer). When you are ready, press the Enter key. If you want to cancel the print command, press Esc.

### ***System:***

---

Easytone provides this command to allow you to temporarily jump out to the DOS system, in case you need to look up a file, change date, change time, etc.

Do not run large files when you are in the DOS shell, or memory problems may occur.

To exit the DOS shell and return to Easytone, type Exit at the DOS prompt.

### ***Exit:***

---

This command will return you to the main menu.

## Signals:

The signals menu item is used to enter information about encoding and decoding. There are several different formats available.

### Tones:

Single and two-tone codes can be decoded and encoded with this command.

### Decode:

You are presented with a screen that allows you to enter up to ten tone groups, using either single tone or two-tone formats

Tone A		Tone B		Message	Action
Freq. Hz	Time sec	Freq. Hz	Time sec		
1	1360.0	2.7	1470.0	1.7	▼Attack Activat ▼Attack
2	1360.0	2.7	1587.0	1.7	▼Alert Activate ▼Alert
3	1360.0	2.7	1851.0	1.7	▼Cancel Activat ▼Cancel
4	0.0	0.0	0.0	0.0	▼None ▼None
5	0.0	0.0	0.0	0.0	▼None ▼None
6	0.0	0.0	0.0	0.0	▼None ▼None
7	0.0	0.0	0.0	0.0	▼None ▼None
8	0.0	0.0	0.0	0.0	▼None ▼None
9	0.0	0.0	0.0	0.0	▼None ▼None
10	0.0	0.0	0.0	0.0	▼None ▼None

Ok Help

F1-Help F2-Save

#### **Tone A Frequency:**

This is the frequency, in Hertz, of the first tone. Frequencies can range from 200 Hz to 5000 Hz.

#### **Tone A Time:**

The minimum time that Tone A must be present in order to be considered valid. In most instances you will want to use the minimum format time, minus .3 seconds. If, for example, the format specifies that the A time is two seconds long, you first subtract .3 seconds from the two seconds and enter the result of 1.7 seconds as the time.

The reason for subtracting .3 seconds is to allow for timing variances from encoders, keyup delay, etc.

***Tone B Frequency:***

This is the frequency, in Hertz, of the second tone. Frequencies can range from 200 Hz to 5000 Hz.

In the event that you want to program only a single tone, enter 9999 for the second tone. The CSP treats this as a 'Wildcard' number and allows any, or no tone to be considered valid. Thus only the first tone must meet the specifications that are entered.

***Tone B Time:***

This is the minimum time that Tone B must be present in order to be considered valid. In most instances you will want to use the minimum format time, minus .3 seconds. If, for example, the format specifies that the B time is two seconds long, you subtract .3 seconds from the two seconds and use the result of 1.7 seconds to enter as the time.

The reason for subtracting .3 seconds is to allow for timing variances from encoders, keyup delay, etc.

If you are programming only a single tone for the group, enter 0 as the minimum time. The zero for the minimum time coupled with the 9999 for the B Frequency, allows the CSP to concentrate on only the first tone.

***Message:***

Upon a successful decode of the tones and timing for the tone group, the CSP will write a message to the LCD screen and/or the serial output.

You can enter up to 30 messages in the Messages screen that you access from the main menu. In this Tone/Decode screen you can select one of the 30 messages to be displayed when the tones are decoded.

To select a message, use the cursor arrow keys, or the mouse to highlight the 'v' symbol. When you have selected the one you want to work with, press the Enter key.

A list of selections will appear. The current selection is highlighted. Use the up and down arrows to select which message you want to appear. When you have your choice highlighted, press the Enter key to transfer your selection to the Messages field. Due to space constraints, only the first 10 characters of the message will be displayed. However, at link time the entire message will be linked to the tone group.

If you do not want to send a message, you may select 'None' on the very first line of the list. The word None will not be displayed as a message. It merely advises both you and the compiler that no message will be linked.

**Action:**

Upon a successful decode of the tones and timing for the tone group, the CSP will perform a predetermined action of your choice.

You can enter up to 60 actions in the Actions screen that you access from the main menu. In this Tone/Decode screen you can select one of the 60 messages to be displayed when the tones are decoded.

To select an action, use the cursor arrow keys, or the mouse to highlight the 'v' symbol. When you have selected the one you want to work with, press the Enter key.

A list of selections will appear. The current selection is highlighted. Use the up and down arrows to select which Action you want to invoke. When you have your choice highlighted, press the Enter key to transfer your selection to the Action field. The name of the Action will be displayed. However, at link time the entire action will be linked to the tone group.

If you do not want to invoke an action, you may select 'None' on the very first line of the list. The word None is not an Action name, it merely advises both you and the compiler that no action will be run.

**Encode:**

You are presented with a screen that allows you to enter up to ten tone groups, using either single tones or two-tone formats. These tones are considered an ‘Action’, as such they are treated like any other action and can be sent when an input goes active, a tone or DTMF signal is decoded, or when invoked by another action as part of the normal CSP routine.

When a Tone Encode is invoked, there is a five step process that takes place.

- 1) Pre-Message: Send a message to the screen.
- 2) Pre-Action: Perform an action, such a keying up a transmitter.
- 3) Encode Tones: Generate the ‘A’ and ‘B’ tones.
- 4) Post-Message: Clear the screen and send another message.
- 5) Post-Action: Perform another action, such as un-keying a transmitter.

SIRENS.CSP		Communications Signal Processor				14:01:20	
Tones		DTMF	5/6	FSK	Morse	eXit	
Decode	Encode	Tone Encoder					
Pre-Message	Pre-Action	Tone A		Tone B		Post-Message	Post-Action
		Freq. Hz	Time sec	Freq. Hz	Time sec		
1	▼Paging H	▼Tx 1 On	410.5	1.0	349.0	3.0	▼Complete ▼Tx 1 Off
2	▼Meisvill	▼Tx 1 On	330.5	8.0	0.0	0.0	▼Complete ▼Tx 1 Off
3	▼None	▼None	0.0	0.0	0.0	0.0	▼None ▼None
4	▼None	▼None	0.0	0.0	0.0	0.0	▼None ▼None
5	▼None	▼None	0.0	0.0	0.0	0.0	▼None ▼None
6	▼None	▼None	0.0	0.0	0.0	0.0	▼None ▼None
7	▼None	▼None	0.0	0.0	0.0	0.0	▼None ▼None
8	▼None	▼None	0.0	0.0	0.0	0.0	▼None ▼None
9	▼None	▼None	0.0	0.0	0.0	0.0	▼None ▼None
10	▼None	▼None	0.0	0.0	0.0	0.0	▼None ▼None

Ok      Help

**Pre-Message:**

The CSP will write a message to the LCD screen and/or the serial output.

You are able to enter up to 30 messages in the Messages screen that is accessed from the main menu. In the Tone/Decode screen you can select one of the 30 messages to be displayed when the tones are decoded.

To select a message, use the cursor arrow keys, or the mouse to highlight the ‘v’ symbol on one of the ten tone lines. When you have selected the one you want to work with, you press the Enter key.

A list of selections will appear. The current selection is highlighted. You can use the up and down arrows to select which message you want to display. When you have your choice highlighted, press the Enter key to transfer your selection to the Messages field.

Due to space constraints, only the first 10 characters of the message will be displayed. However, at link time the entire message will be linked to the tone group.

If you do not want to send a message, you may select 'None' on the very first line of the list. The word None will not be displayed as a message. It merely advises both you and the compiler that no message will be linked.

***Pre-Action:***

The CSP will perform a predetermined action prior to generating tones.

You are able to enter up to 60 actions in the Actions screen that is accessed from the main menu. In this Tone/Decode screen you can select one of the 60 messages to be displayed when the tones are decoded.

To select an action, use the cursor arrow keys, or the mouse to highlight the 'v' symbol on one of the ten tone lines. When you have selected the one you want to work with, you press the Enter key.

A list of selections will appear. The current selection is highlighted. You can use the up and down arrows to select the Action you want to invoke. When you have your choice highlighted, press the Enter key to transfer your selection to the Action field. The name of the Action will be displayed. However, at link time the entire action will be linked to the tone group.

If you do not want to invoke an action, you may select 'None' on the very first line of the list. The word None is not an Action name, it merely advises both you and the compiler that no action will be run.

***Tone A Frequency:***

This is the frequency, in Hertz, of the first tone. Frequencies can range from 200 Hz to 5000 Hz.

***Tone A Time:***

The time, in seconds, that Tone A will be generated.

***Tone B Frequency:***

This is the frequency, in Hertz, of the second tone. Frequencies can range from 200 Hz to 5000 Hz.

In the event that you want to generate only a single tone, enter 0 for the tone B frequency. The CSP will then generate the A tone only.

***Tone B Time:***

The time, in seconds, that Tone B will be generated.

If you are programming only a single tone for the group, enter 0 as the minimum time for the B tone. The zero for the minimum time coupled with the 0 for the B Frequency, allows the CSP to generate only the A tone.

***Post Message:***

The CSP will erase the LCD and then write a message to the screen. The method of selecting the message is the same as the Pre-Message described above.

If you do not want to send a message, you may select 'None' on the very first line of the list. The word None will not be displayed as a message. It merely advises both you and the compiler that no message will be linked.

***Post Action:***

The CSP will generate another Action if requested. The method of selecting the Action is the same as the Pre-Action described above.

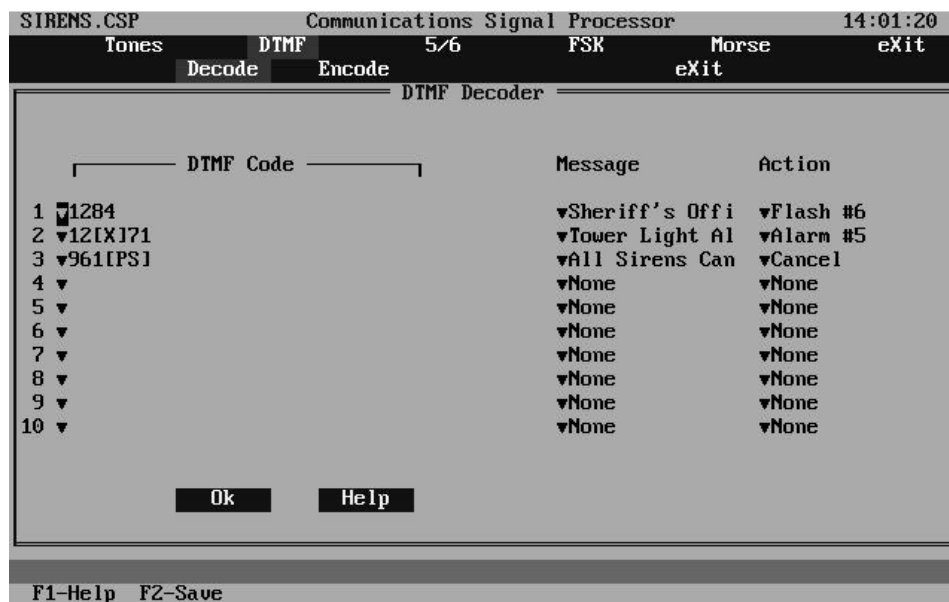
If you do not want to invoke an action, you may select 'None' on the very first line of the list. The word None is not an Action name, it merely advises both you and the compiler that no action will be run.

**DTMF:**

DTMF (Dual Tone Multiple Frequencies) codes can be decoded and encoded with this command.

**Decode:**

You are presented with a screen that allows you to enter up to ten DTMF memory codes that the CSP will attempt match with codes received. When the DTMF code received matches one in the CSP's memory, the reset of the group consisting of the Message and Action is performed by the CSP.

**DTMF Code:**

This DTMF code will be compared with the DTMF codes received.

The code can be up to 36 characters in length, and can consist of the standard characters 0 to 9, as well as the characters \*, #, A, B, C, and D.

Sample code  
12539

If you used the above example, when the code 12539 was received by the CSP, it would search through the DTMF codes and find a match with 12539. With this valid match, the CSP would run the Message and Action you had specified to operate with that code.

**Sample code****9361A0571C84#1\*028B28741032917492BA2**

As the above code demonstrates, Easytone can enter DTMF code can be up to 36 characters long.

**Variable Codes**

Since the CSP uses GOS as its operating system, it has the ability to search for codes in ways that were never before possible. You can search for codes that can change depending upon the status of outputs, inputs, on-board switch settings, variables, etc. It is, without a doubt, one of the most powerful decoding systems available.

You are no longer limited only to static DTMF codes such as 127431. By using the “variable inserter” you can alter the DTMF memory code that you are attempting to match.

A variable inserter is designated by the left and right brackets [ ]. Place the variable information between the brackets. After the CSP has received a DTMF code and is searching its DTMF memory for a valid match, it uses the information inside the variable inserter as a basis for a match.

The variable inserter can contain information such as the position of the on-board switch provided on the CSP-105 model.

**Example****87[PS]54**

In this example, the variable inserter substitutes the switch setting. If, for example, the switch was set to 6, then the DTMF code 87654 would be valid and the Message and Action for that code would be run. Using the switch in this method is perfect when you have many units in the field, and want to give them their own codes by simply moving the on-board switch.

The variable inserter can also work with the inputs or outputs, on a full port, upper nibble, lower nibble, or an individual basis.

**Example****419[PC][PB]71**

Examining the above code, you can see that the variable inserter can be used more than once on the same code. In the above example, it looks for a match at position 4 by using the value of the input port, upper nibble. Next, in position 5 it substitutes the value of the input port, lower nibble.

### Wildcards

There may be situations where you want to allow any DTMF character to be valid in a code position. For this you will still use the variable inserter, but you will use the X character. In this context the X represents any DTMF character and is called a Wildcard character.

Sample Code  
654[X]19

In the above example, any DTMF digit or character would satisfy the X in the variable inserter. Codes such as 654319, 654519, 654712, etc. would all be valid codes.

It is permissible to have multiple wildcards in the same DTMF memory code.

Sample code  
65[X][X]19[X]

The above example shows the ability to have multiple wildcards in the same DTMF memory code. Using this example DTMF codes like 6592196, 6565190, 6580198, etc. are all valid codes and all would run the Messages and Actions for that code.

### Non-Matched DTMF Codes

In the event that the DTMF code was unable to find a match in the DTMF memory. The CSP has the ability to run a special Action for working with the DTMF codes that did not have a match.

To use the Non-match ability, use the special Action name “Non-match” for the Action code you want to invoke anytime a DTMF code match cannot be made. If, for example, you want to make Action 42 the Non-matched DTMF Action, go to the Action screen, place the blinking cursor in the Action Name field of memory slot 42, and type “Non-Match”. Then any time a DTMF code fails to find a match, Action 42 will be run.

The non-matched Action can perform anything a standard Action can, but is especially useful for logging non-matched codes, keeping track of the length of DTMF codes that are on-the-air, or how often particular code groups or characters are sent.

If the CSP can not locate the Action number that you have entered for the non-matched Action, it jumps over the request and continues as normal.

In the CSP-105 the maximum value for the non-matched Action number is 255. Any number higher will produce unpredictable results.

**Message:**

Upon a successful DTMF decode of the code group, the CSP can write a message to the LCD screen and/or the serial output.

You can enter up to 30 messages in the Messages screen that is accessed from the main menu. In this DTMF/Decode screen you can select one of the 30 messages to be displayed when the tones are decoded.

To select a message, use the cursor arrow keys, or the mouse to highlight the 'v' symbol. When you have selected the message you want to work with, press the Enter key.

A list of selections will appear. The current selection is highlighted. You can use the up and down arrows to select which message you want to display. Once you have your choice highlighted, press the Enter key to transfer your selection to the Messages field. Due to space constraints, only the first 10 characters of the message will be displayed. However, at link time the entire message will be linked to the tone group.

If you do not want to send a message, you may select 'None' on the very first line of the list. The word None will not be displayed as a message. It merely advises both you and the compiler that no message will be linked.

**Action:**

Upon a successful DTMF code match, the CSP will perform a predetermined action.

From the Main menu, you can access the Actions screen and enter up to 60 Actions. In this DTMF/Decode screen you can select one of the 60 messages to be displayed when the tones are decoded.

To select an action, use the cursor arrow keys, or the mouse to highlight the 'v' symbol. When you have selected the one you want to work with, press the Enter key.

A list of selections will appear. The current selection is highlighted. Use the up and down arrows to select which Action you want to invoke then press the Enter key to transfer your selection to the Action field. The name of the Action will be displayed. However, at link time the entire action will be linked to the tone group.

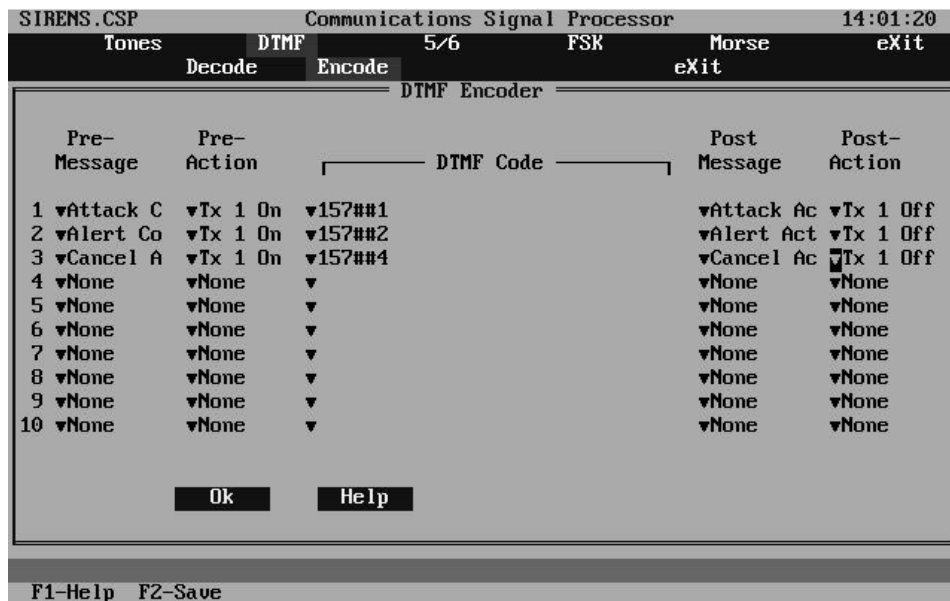
If you do not want to invoke an action, you may select 'None' on the very first line of the list. The word None is not an Action name, it merely advises both you and the compiler that no action will be run.

**Encode:**

You are presented with a screen that allows you to enter up to ten DTMF code groups. These codes are considered an 'Action', as such they are treated like any other action and can be sent when an input goes active, a tone or DTMF signal is decoded or when invoked by another action as part of the normal CSP routine.

When a DTMF Encode is invoked, there is a five step process that occurs in order.

- 1) Pre-Message: Send a message to the screen.
- 2) Pre-Action: Perform an action, such a keying up a transmitter.
- 3) DTMF Code: Generate the DTMF code.
- 4) Post-Message: Clear the screen and send another message.
- 5) Post-Action: Perform another action, such as un-keying a transmitter.

**Pre-Message:**

The CSP will write a message to the LCD screen and/or the serial output.

You enter up to 30 messages in the Messages screen that is accessed from the main menu. In this Tone/Decode screen you can select one of the 30 messages to be displayed when the tones are decoded.

To select a message, use the cursor arrow keys, or the mouse to highlight the 'v' symbol. When you have selected the one you want to work with, you press the Enter key.

A list of selections will appear. The current selection is highlighted. Use the up and down arrows to select the message you want to display, then press the Enter key to transfer your selection to the Messages field. Due to space constraints, only the first 10 characters of the message will be displayed. However, at link time the entire message will be linked to the tone group.

If you do not want to send a message, you may select 'None' on the very first line of the list. The word None will not be displayed as a message. It merely advises both you and the compiler that no message will be linked.

#### ***Pre-Action:***

The CSP will perform a predetermined action before generating DTMF codes.

You are able to enter up to 60 actions in the Actions screen that you access from the main menu. In this Tone/Decode screen you can select one of the 60 messages to be displayed when the tones are decoded.

To select an action, use the cursor arrow keys, or the mouse to highlight the 'v' symbol. When you have selected the one you want to work with, you press the Enter key.

A list of selections will appear. The current selection is highlighted. Use the up and down arrows to select which action you want to invoke, then press the Enter key to transfer your selection to the Action field. The name of the Action will be displayed. However, at link time the entire action will be linked to the tone group.

If you do not want to invoke an action, you may select 'None' on the very first line of the list. The word None is not an Action name, it merely advises both you and the compiler that no action will be run.

#### ***DTMF Code:***

This is the DTMF code that the CSP will generate.

The code can be up to 36 characters in length, and can consist of the standard characters 0 to 9, as well as the characters \*, #, A, B, C, and D.

Sample code

**36195**

If you used the above example, the code 36195 would be generated by the CSP.

Sample code

**1598B#87407A45CD59861010#525\*BD372C1**

As the above code demonstrates, your code can be up to 36 characters long.

### Variable Codes

Since the CSP uses GOS as its operating system, it has the ability to send codes that contain variable information such as the status of outputs, inputs, on-board switch settings, variables, etc.

You are no longer limited only to a static DTMF code such as 127431. By using the “variable inserter” you can change the DTMF code that is generated.

A variable inserter is designated by the left and right brackets [ ].

Place the variable information between the brackets. As the DTMF code is being generated, it uses the information inside the brackets to retrieve information from the CSP and use the return value in the DTMF code.

The variable inserter can contain information such as the position of the on-board switch provided on the CSP-105 model.

#### Example

**91[PS]3**

In this example the CSP will come across the variable inserter that substitutes the on board switch setting. If, for example, the switch was set to 6, then the DTMF code 9163 would be generated.

Using the switch in this method is useful when you have many units in the field, and want to assign each unit its own code by moving the on-board switch.

The variable inserter can also work with the inputs or outputs, on a full port, upper nibble, lower nibble, or an individual basis.

#### Example

**19[PA]75**

Examining the above code, you see that the variable inserter can be used to transmit the current status of the full input port.

In the above example, the value of the input port is placed starting at position number 3 of the DTMF code.

If, for example, input number 2 was active. The CSP would generate the DTMF code 19275.

### ***Post Message:***

The CSP will erase the LCD and then write a message to the screen. The method of selecting the message is the same as the Pre-Message described above.

If you do not want to send a message, you may select 'None' on the very first line of the list. The word None will not be displayed as a message. It merely advises both you and the compiler that no message will be linked.

***Post Action:***

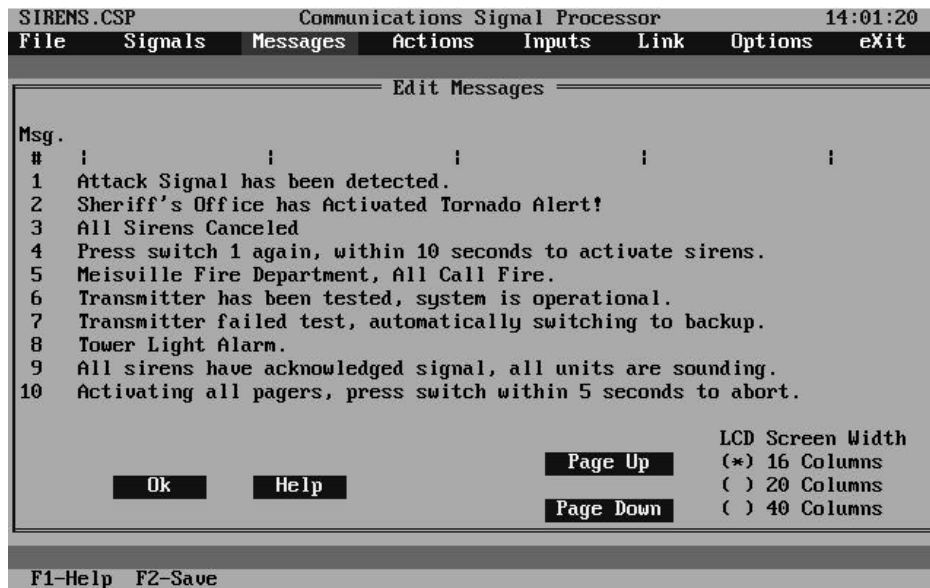
The CSP will generate another Action if requested. The method of selecting the Action is the same as the Pre-Action described above.

If you do not want to invoke an action, you may select 'None' on the very first line of the list. The word None is not an Action name, it merely advises both you and the compiler that no action will be run.

## Messages:

You can store up to 30 messages in memory, each message may be up to 60 characters in length. These messages are sent to the LCD screen and/or the Serial Communications Port as needed to fulfill decode, encode, and input groups.

Enter the messages by using the Messages selection from the Main menu.



Initially you will see space for 10 messages. By pressing the Page Down key on your keyboard, you will see the next twenty messages, in groups of ten on the screen at a time.

The messages are entered in preparation for being used. The only time a message is pulled from the group and used in the CSP, is when it is specifically called by an encode, decode, input, etc. in the form of a Pre-Message, Post Message, or standard Message.

Thus, you may have possibly fifteen messages, but only use five of them for a particular application that you are making.

To enter a new message or to change an existing message, place the cursor on the line you want to work on, and begin typing the information.

The standard keyboard arrow keys, backspace, end and home keys allow you to edit the message to suit your needs.

By toggling between the insert and the overwrite cursor using the 'Insert' key on the keyboard, you can easily insert new text or spaces between existing words or letters.

**Screen Markers**

You may notice periodic ‘|’ symbols on the top of the screen. These markers indicate the new line position for various size LCD screens.

The default size is a 16 character wide screen. Each marker is the start of a new line where the LCD driver will move the cursor to a new line.

There are three screen sizes standard with Easytone, 16, 20 and 40 columns wide. By using the selection fields in the lower right corner of the screen, you can select a new screen size. If you change sizes, you must page up or down, then back to the original page before the marker changes take effect.

Markers exist for your convince to help you align messages for maximum readability and impact. They do not affect the program in any way.

## Actions:

Actions are a method of commanding the CSP to perform some type of change in its outputs, internal memories, encoding tones, or any other operations available with the GOS programming language.

You can store 60 Actions in memory, from memory slot 40 to slot 99.

The screenshot shows the 'User Defined Actions' table in the SIRENS.CSP interface. The table has three columns: Name, Action Program Code, and Memory Slot. Below the table are several control buttons: Ok, Help, Page Up, Edit Name, Page Down, and Edit Code. At the bottom, there is a footer with function key shortcuts: F1-Help, F2-Save, F3-Name, F4-Code, F5-Clear Line, and F10-Ref.

Name	Action Program Code	Memory Slot
Attack	R1=1;H1000;R1=0;	40
Alert	R2=1;H1000;R2=0;	41
Fire	R3=1;H1000;R3=0;	42
Cancel	R4=1;H1000;R4=0;	43
Flash #6	Q1,180,1,1,'R6=1','R6=0','R6=0';	44
Alarm #5	Q2,600,2,1,'R5','R5','R5=0';	45
Tx 1 On	R7=1;	46
Tx 1 Off	R7=0;	47
START-UP	A33;R8=1;S Relay #8 is engaged on startup.;	49

A memory 'slot' is a number that Easytone uses when it links the working file you are using. The linker uses the slot number to advise other sections of the Easytone file where the Action can be found. A memory slot can also be called a function number, for those familiar with the GOS language.

The Slot number is displayed for reference when including it in another Action code during a jump (for advanced programming only).

Using Actions, you have a central location for all of the 'low level' instructions to be performed upon command.

### **Name:**

Since many actions may look cryptic upon first glance, it is useful to assign a short name that is descriptive of the work that the action will perform.

“Open #1”, “Rly1 Off”, “Key #6”, etc. would be an example of names that could be used. Names can consist of numbers, letters and characters.

To change a name or enter a new name, first place the cursor in the name field by pressing function key F3. The cursor will stay on the same line as in the Action column, but it will now move over to the Name column.

Now enter the new information for the name.

While in the name column, use the up and down arrows to select a new Name line. The page up and page down keys will show the rest of the 60 Action Program Codes while leaving the cursor in the Name column.

### **Start-up**

Start-up is a special Action Name. It is used to denote the Action Code that will be run when the CSP is first powered up or reset. The Start-up code is useful for displaying an initial message or to set outputs or counters to initial defaults.

### **No-match**

No-match is a special Action Name. After a DTMF code is received the CSP attempts to find a match in the DTMF memory. If a match is not found, the CSP can run the Action Code that you prepared for the No-match. This Action Code can count signals, alert how close an invalid code was to a valid code, etc.

### ***Action Program Code:***

The Action code is a group of statements that control the operation of the CSP when they are invoked.

They are accessed by pop-up lists in the Signals menu, as well as the Inputs menu.

In the above menus, select the action you want to run when a signal is decoded, when an input becomes active, or during an encode sequence.

The code itself consists of standard GOS commands. These commands are very short 'token' based commands that are designed to use minimum memory but provide maximum speed and versatility.

To change or enter a new Action Code, you must first place the cursor in the Action Code field by pressing function key F4. The cursor will stay on the same line as in the Name column, but will now move over to the Action Code column.

You may now enter the new information for the Action Code.

While in the Action Code column, use the up and down arrows to select a new Action line. The page up and page down keys will show the rest of the 60 Action Program Codes while leaving the cursor in the Action column.

An example of the codes is the “R” command that is used to control the outputs.

There are eight outputs that can be controlled on an individual or group basis.

“R” followed by a number from 1 to 8 will control that individual output.

Sample Code

R1=1;

The above code would turn output number one on. “On” is the state where the output is active. In the case of the CSP-105, where active low logic is used, “On” means that the output is taken to ground.

Sample code

R7=0;

This will turn output number seven off, which will remove the reference to ground and cut off any current flow that may have been present.

If output number seven had been off, then nothing would have happen.

The Action code can consist of up to 60 characters of commands.

Sample code

R6=1;R8=0;RB=5;H2000;RA=0;SDone;

The above example is a group of statements that are strung together. If, for example, you entered the statements into Action Code number 45, and Action code 45 is invoked by another part of the program, the following would occur.

Output number 6 would turn on, output number 8 would turn off, the lower nibble (RB controls outputs 1, 2, 3 and 4 as a group) would equal 5, which means that outputs 1 and 4 would turn on. Next, the H2000 command would halt the program for 2000 milli-seconds (2 seconds), after which all of the outputs would be turned off by the full port command RA=0;

At the end of it all, the say command ‘S’ would display the word “Done” to the LCD screen.

As you will notice, the construction of an Action string is very important. There are a few rules that must be adhered to for proper operation.

- 1) Individual statements must be separated by semi-colons “;”.
- 2) There can be no spaces in any commands or instructions, except for the Say command which may contain spaces as part of its normal text.
- 3) The ending character of an Action Code must be the semi-colon “;”.

As shown above, an Action Code can be quite versatile. For additional information on the GOS language, please see the Commands section, or the book “**GOS - Programmers Reference Guide**”.

The Action Codes are entered in preparation for use. The only time an Action is pulled from memory and used in the CSP, is when it is specifically called by an encode, decode, input, etc. in the form of a Pre-Action, Post Action, or standard Action.

Thus you may have possibly twenty commonly used Action Codes, but only use eight of them for an application that you are constructing.

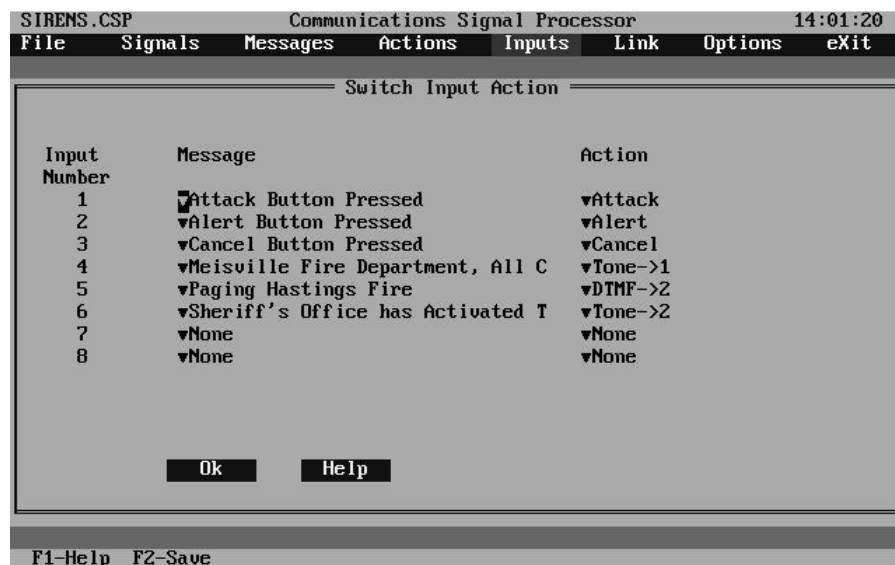
## Inputs:

The CSP contains eight inputs that can be connected to external switches, sensors, controls, etc.

These the inputs can be used in several ways.

- 1) Inputs can be examined by an Action code, with the results determining a course of action.
- 2) The input values, individual, lower nibble, upper nibble, or full port, can be used as part of a DTMF code.
- 3) The inputs, on an individual basis, can be used to activate a Message and Action Code.

The “Inputs” selection from the main menu deals with the third option shown above.



The CSP continually monitors the eight inputs to determine whether any of the inputs “goes active”. The CSP uses “active low” logic, this means that when an input is pulled to board/power ground, it is considered active.

When an input becomes active, the CSP checks to see if you have programmed a Message or Action to take place.

If a Message or Action is specified, the CSP will run the combination each time that the input goes low.

A search is performed only when an input goes from a logic high (unconnected) to a logic low (ground). No search is attempted when the input is released from the ground potential and rises to a logic high.

### **Contact Bounce**

When you use relays or switches to activate the inputs, a certain allowance is made to compensate for the mechanical nature of these devices.

Anytime a switch or relay closes, the contacts on the device “bounce” slightly while they settle. Since the CSP is a high speed processor, it could literally count each and every one of the bounces as another input closure, even though the bounces are extremely fast.

To prevent this, and allow you to take advantage of general use switches and relays, GOS has built-in debounce program. It waits up to 150 milli-seconds for the input to settle down, after which time the input is considered active.

### **Message:**

The CSP will write a message to the LCD screen and/or the serial output.

You enter up to 30 messages in the Messages screen that is accessed from the main menu. In this Tone/Decode screen you can select one of 30 messages to be displayed when the tones are decoded.

To select a message, use the cursor arrow keys, or the mouse to highlight the ‘v’ symbol. When you have selected the one you want to work with, press the Enter key.

A list of selections will appear. The current selection is highlighted. Use the up and down arrows to select which message you want to display. Once your choice is highlighted, press the Enter key to transfer your selection to the Messages field. Due to space constraints, only the first 10 characters of the message will be displayed. However, at link time the entire message will be linked to the tone group.

If you do not want to send a message, you may select ‘None’ on the very first line of the list. The word None will not be displayed as a message. It merely advises both you and the compiler that no message will be linked.

### **Action:**

The CSP will perform a predetermined action when an input is activated.

You are able to enter up to 60 actions in the Actions screen which is accessed from the main menu. In this Tone/Decode screen you can select one of the 60 messages to be displayed when the inputs are active.

To select an action, use the cursor arrow keys, or the mouse to highlight the ‘v’ symbol. When you have selected the Action you want to work with, press the Enter key.

A list of selections called the “Group Selection” will appear. The groups appear as follows.

- Action     Action Codes.
- Tone        Tone Encode table.
- DTMF       DTMF Encode table.
- 5/6         5/6 Code tables.
- FSK         Frequency Shift Keying table.
- Morse       Morse code table.

Select which Group you want to link into this action. Action codes are the actions that you have already entered through the Action screen.



The other Groups provide direct access to the Encode information for their respective signal.

If, for example, you want to send the tone signal number 1, that you had previously entered, when input number 4 went active. Highlight “Tones” with the arrow keys, then press Enter. The screen will change to display the 10 tone encode lines. By again using the arrow keys, highlighting the entry “Tone->1” and pressing Enter, the information in the Tone Encode screen, line number 4 would be linked into the Inputs screen.



After you have linked the entire program, and sent the information to the CSP, Each time input number 4 goes active, the messages, actions, and tones for tone signal number 1 will be invoked and transmitted.

The Action code selection is the same as a tone selection. After you have selected Action from the Group, a new screen is displayed with the Action code names and the current selection highlighted. Use the up and down arrows to select which message you want to displayed.

When your choice is highlighted, press the Enter key to transfer your selection to the Action field. The name of the Action will be displayed. At link time the entire action will be available and linked to the tone group.

If you do not want to invoke an action, you may select ‘None’ on the very first line of the list. The word None is not an Action name, it merely advises both you and the compiler that no action will be run.

## Link:

---

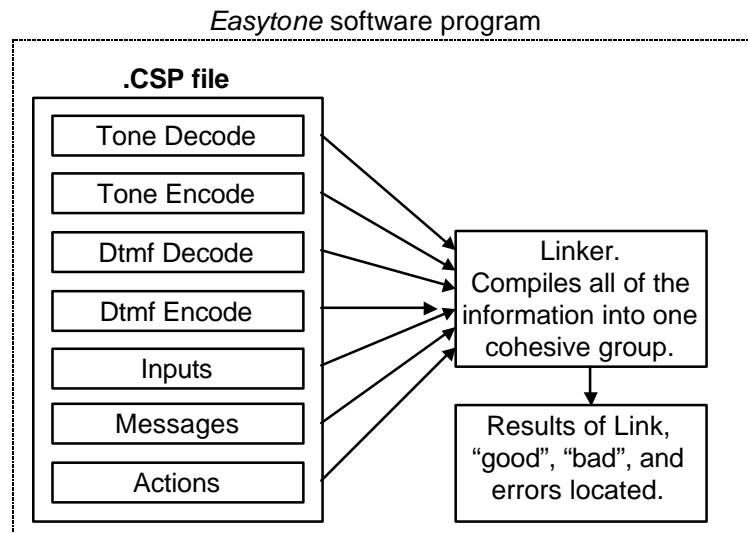
The Link menu contains various menu selections that deal with compiler action or with communications to and from the CSP that you are connected to.

### Link:

---

The Link command is used to compile all of the various menus and screens that you have entered information and data into.

After the information is compiled, you are advised of the result of the Link. Easytone will report any errors it found during the link.



After the Link is completed, Easytone displays the amount of memory the Linked file will occupy in the CSP when it is “Uploaded” (sent to the CSP).

You must compare the amount of memory displayed to the memory size of the CSP model you will be uploading to. The linked memory must be equal to or smaller than the CSP’s memory.

The CSP-105, for example, has 2K (2048) of program memory available for your program. If your linked file exceeds this number, your program will not operate properly.

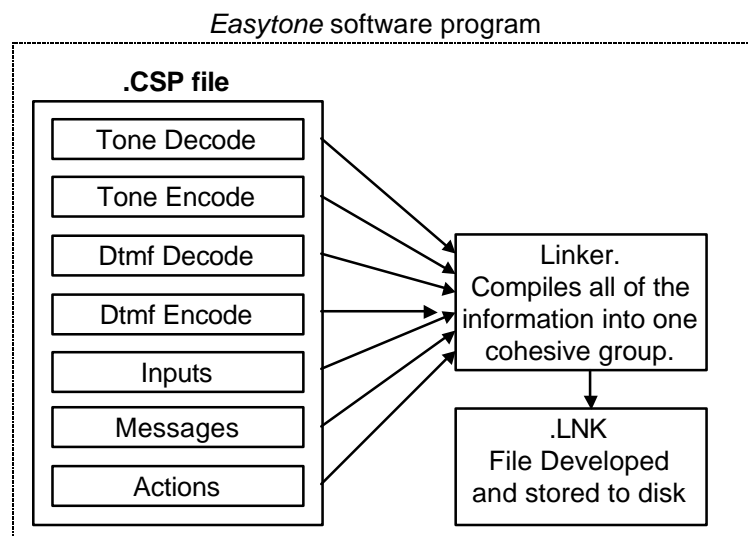
The Link is often used as a quick check of program size and to locate errors in your program. You can check these items prior to uploading the program to the CSP.

## Save:

The Save command is used to compile all of the various menus and screens that you have entered information and data into. It performs the same as the Link, however it also stores the Linked file to your computer disk at the end of the link.

After the information is compiled, it is stored to a new file. The name of the file is the same as the work file. However, the ending “.CSP” suffix is changed to “.LNK” which stands for “Linked file”.

The information contained in your main file, with the “.CSP” suffix, is not affected in any way.



After the link is completed, Easytone displays the amount of memory the Linked file will occupy in the CSP when it is “Uploaded” (sent to the CSP).

You must compare the amount of memory displayed to the memory size of the CSP model you will be uploading to. The linked memory must be equal to or smaller than the CSP’s memory.

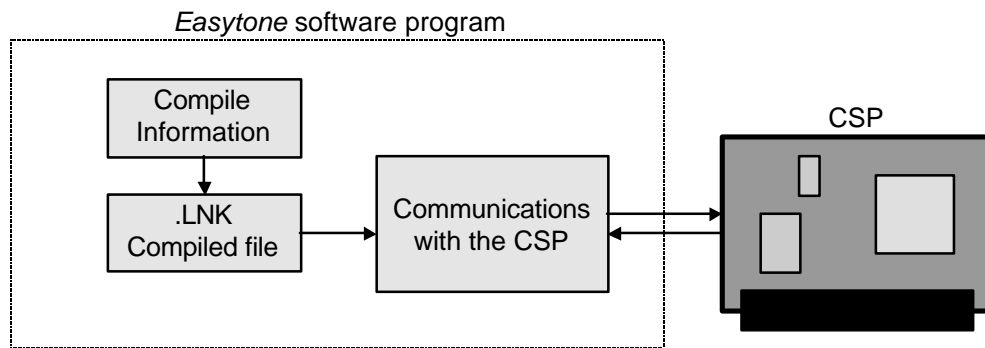
The CSP-105, for example, has 2K (2048) of program memory available for your program. If your linked file exceeds this number, your program will not operate properly.

**Send:**

The Send menu option is an extension of the Link command described above.

The Send command, performs a link, creating or updating the “.LNK” file.

If the link is successful, the Send command uploads the link file to the CSP.

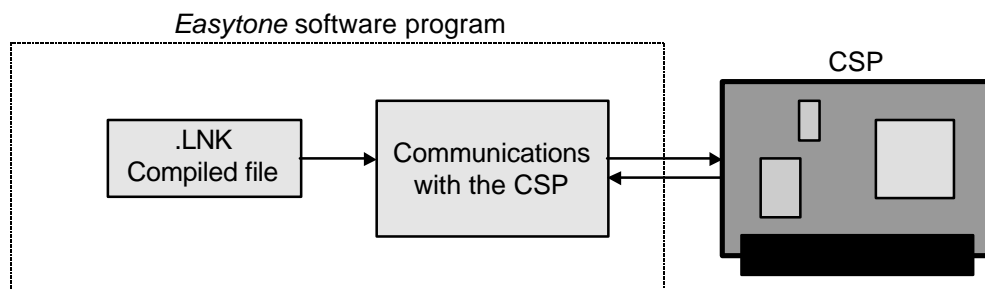


After the entire program has been uploaded, Easytone terminates communications with the CSP and returns to the Send screen.

While a Send is in progress, you may terminate the command by pressing the Esc. key.

**Re-Send:**

The Re-send option does not perform a link. Instead it opens an already linked file and uploads it to the CSP.



The Re-send is useful when you have already linked a file and need to send it again such as when you are programming several CSP's.

It is also useful to upload a Linked file that you have made changes to by using the View option. If you used either the Save or Send options after manually making changes to the linked file, those changes would be overwritten to the disk.

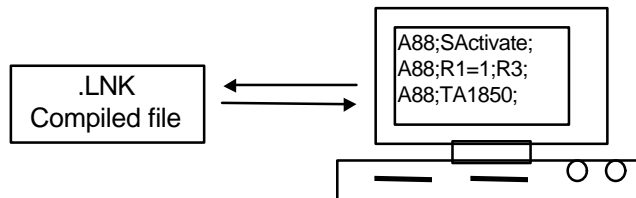
However, by using the Re-send option, no relink is performed, and the file is sent with any manual changes that you have made.

## **View:**

---

There are times that you may need to manually “tweak” a linked file. This option allows you to manually edit a linked file by using either the Link or Send options.

Using View, open the file and make any necessary changes to the linked file.



After you have saved any changes, use the Re-send option to upload the new program.

*The View option contains no softies and therefore should be used with caution. Any changes you make to the linked file will be uploaded to the CSP, without any error checking.*

This option should be used only after you have read and understood the GOS Language Manual.

## **Interactive:**

---

Since the CSP can communicate in real-time with other computers, it is only natural that Easytone should give you access to this power.

The Interactive option opens a two-way data communications path to the CSP, allowing you to communicate directly with the GOS command processor through your computer.

You can query, command, test, check, and generally control all aspects of the CSP.



```
SIRENS.CSP      Communications Signal Processor      14:01:20
Link  save  Send  Re-send  View  Interactive  exit

----- RX - Info. received from the CSP -----
Attack Button Pressed

The value in X2 is 412
Paging Fire Dept.

?N1+344
F1-Help  F2-Save
```

Testing a program that you have uploaded is generally performed in the interactive mode. This allows you to view responses from the CSP, override particular actions, and alter memory contents and variables to check for program execution.

All GOS commands can be entered by the keyboard. As you type, the characters are shown at the bottom of the screen. You can use the Backspace key to make any changes.

When you press the Enter key, the string of commands at the bottom of the Interactive screen are sent as one group that the CSP receives and works on like a normal Action Code.

Due to memory limitations, a string cannot exceed sixty characters in length.

You can use the Alt-C key to clear the screen of the replies from the CSP in the event it becomes cluttered. This clearing affects only your local computer screen and does not erase any memory data in either your computer or the CSP.

## Options

You may customize the way that Easytone operates and change default settings for the communications port you wish to use on your computer, name of files, etc.

This is accomplished through the Options menus.

When you make changes to any of the Options, you will need to Save the file to retain the data.

The Options that you enter are particular to that file. This means that you can have distinct settings for different filenames. Options settings are stored with each file.

### System Options



### Serial Port

When Easytone communicates with the CSP, it uses a serial communications port on your computer.

These ports are numbered 1, 2, 3 or 4. Most computers use either number 1 or 2 for normal communications.

Easytone will initially select serial port number 1. If you want to change that setting, highlight your choice using the arrow keys, then press Enter.

This will disable the old communications port setting, and place an \* beside the new selection.

## File Editor

---

When you use the 'View' option in the Link menu, an external file editor is loaded for editing the file.

Some programmers have a favorite editor that they may prefer to use. In that case, the name and path of the file editor may be changed to the new editor.

Easytone will pass the name of the file to be edited when it calls the editor.

## New File Defaults

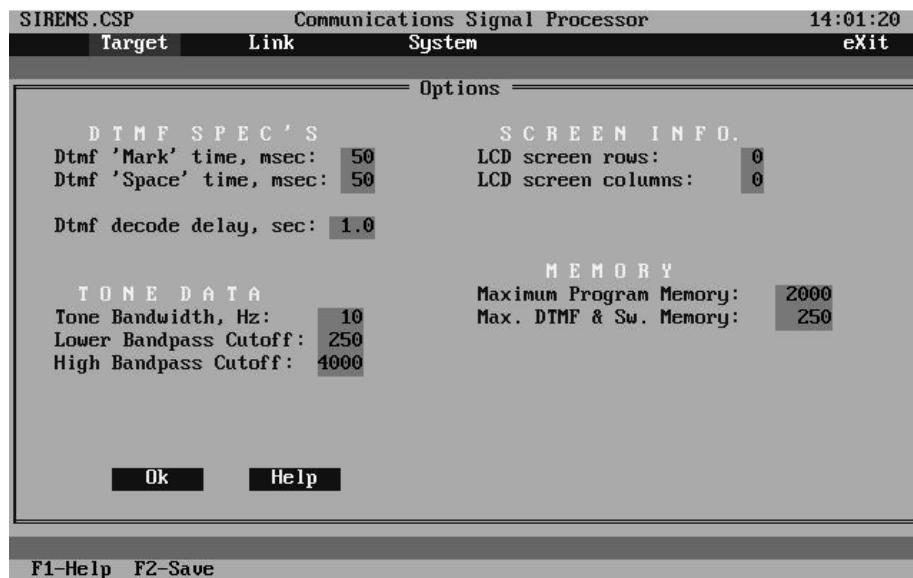
---

When you begin a new file using "Noname.csp" as the starting point, or when you "Reset" the data in a file. Easy tone uses the data from the "New File Defaults" as the basis for the default values which are transferred to the "Options/Target" fields.

See "Target" for additional information on the data and its use.

## Target

---



## Mark Time

---

You can change the time that the CSP generates a DTMF tone, which is referred to as the Mark time.

The time is entered in milli-seconds. There are 1000 milli-seconds in one second.

---

## Space Time

---

You can change the time that the CSP waits between each DTMF tone, this is referred to as the Space time.

The time is entered in milli-seconds. There are 1000 milli-seconds in one second.

---

## Decode Delay

---

When the CSP receives a DTMF character, it starts an internal timer. When this timer expires before another DTMF character is received, GOS considers the DTMF word as complete and submits it for review and possible matching against the DTMF codes the customer is looking for.

If another DTMF character is received before this internal timer has expired, the DTMF character is added to the DTMF word and the timer is reset to the maximum time to start its timing from the beginning.

The default delay after the last digit is received, until GOS considers the DTMF code as complete is 1 second.

If the decoder is receiving information from a manual encoder (someone “punching in” one digit at a time”) you will need to extend the timing to a longer period such as 10 seconds.

If you are receiving the DTMF codes from a “Store & Forward” type of encoder, you may want to increase the Decode Delay to a faster time to prevent it accepting data from hand entry.

---

## Bandwidth

---

The CSP tone decoder has the ability to lock on and track tones that may not be exactly on frequency.

It is common for a standard tone encoder to be slightly higher or lower in frequency, due to age of components, alignment, mechanical stress, etc.

Since the decoder of the CSP is fully digital, it is more precise than most encoders. To allow for slight variations in the received tone, Easytone has an adjustable bandwidth setting.

The bandwidth number you enter is the amount that the received tone can vary above or below the ideal frequency you are seeking and still be considered valid.

If, for example, you are seeking a frequency of 410 Hz a bandwidth of 10 Hz would allow the received frequency to go up to 420 Hz or go down to 400 Hz, and still be considered a correct match for the 410 Hz.

### **Low Bandstop**

---

Since the CSP is digital in nature, you can select a point where the CSP no longer accepts any frequency if it is below a certain level. This is called the “Low Bandstop”. Any frequency below this frequency will be ignored.

### **High Bandstop**

---

You can select a point where the CSP ignores all frequencies above a particular frequency, this is called the “High Bandstop”. The default is 4000 Hz but can be altered to suit your needs.

### **LCD Screen Rows**

---

When you compile a project, information is attached that instructs the CSP to address a particular size of LCD screen.

Most CSP’s can work with either 1 or 2 line (row) Liquid Crystal Display modules. You use this field to advise the CSP which it will be working with.

### **LCD Screen Columns**

---

When you compile a project, information is attached that instructs the CSP to address a particular size of LCD screen.

CSP’s can work with a 1 to 40 character wide (columns) Liquid Crystal Display module. Use this field to advise the CSP which it will be working with.

Even if you do not have an LCD attached to the CSP, use either 1 row by 16 columns, or 2 rows by 8 columns as a default for the Row and Column fields.

### **Program Memory**

---

While Easytone is in the process of Linking your program, it keeps track of the size of the program to insure that it does not exceed the memory size of the particular CSP you are using.

It is necessary to advise Easytone of the maximum size of the CSP you are Linking for. You use this field to enter this information.

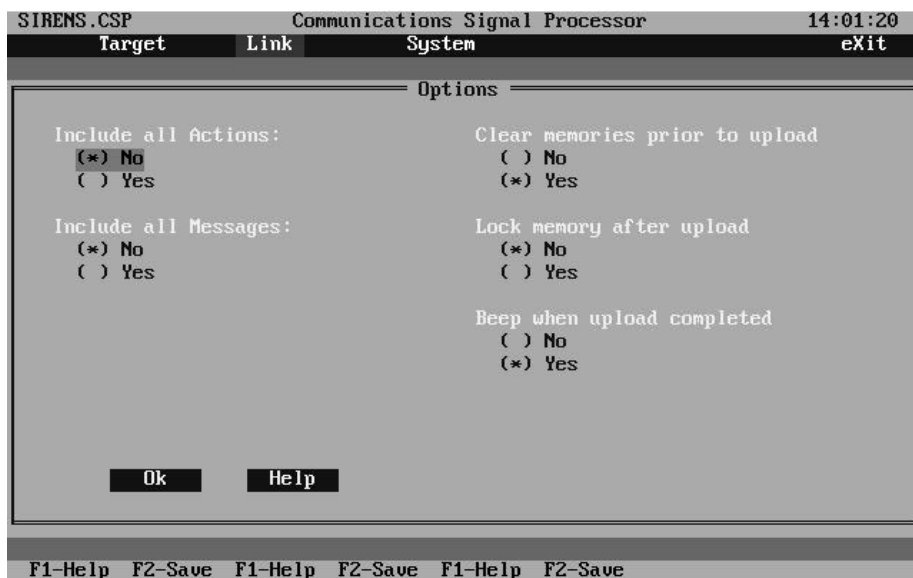
You will normally set most CSP’s to 2000.

## DTMF & Sw Memory

As in the program memory, you advise Easytone of the size of the memory used to store the DTMF codes you wish to decode and any Switch (inputs) actions that will be taken upon the closure of an input.

You will normally set most CSP's to 250.

## Link Options



## Include All Functions

Normally, Easytone will only link actions that it finds used in an Action, Pre-Action or Post Action fields.

You have the ability to override this part of the program and force all functions to be stored to the CSP memory.

This override would normally be used for advanced programming situations where the .LNK file would be modified later via the View menu command.

### **Include All Messages**

---

Normally, Easytone will only link Messages that have previously been used in a Message, Pre-Message or Post Message fields.

You have the ability to override this part of the program and force all functions to be stored to the CSP memory.

This override would normally be used for advanced programming situations where the .LNK file would be modified later via the View menu command.

### **Clear Memories**

---

When Easytone uploads your program to the CSP, it normally will first clear all of the memories in the CSP to prevent any program corruption that may be caused by old data.

Experienced programmers can disable the “memory clear” function for test purposes.

The “memory clear” function will also unlock the memory if the lock function was enabled during the last upload.

### **Lock Memory**

---

The CSP can lock its Function Memory after you have uploaded a program. This will protect your program from viewing, and inadvertent overwriting.

During initial programming and testing of your program, you will not usually lock the memory. You will want to be able to view and debug your program.

You may, however, wish to lock the memory on the completed program. Check the Yes box to lock the memory after the program has been uploaded.

### **Beep on Upload**

---

Normally, Easytone will “Beep” when it has completed the upload. You may disable this feature by checking the No box.

## Commands

---

### **Command: ?- Inquire**

---

Description: Inquire command is used to query the GOS memory, operating system, etc.

Basic form

**Syntax: ?<statement, math, evaluation, variable, input, output, etc.>**

An Inquire command tells the CSP that you are requesting it give you certain information about an evaluation, input, output, memory, etc. The CSP will respond with the answer to your questions, and it will direct that answer to the communications port only. The answer to an Inquire does not go to the LCD screen. If the communications port has been disabled, it will be re-enabled for the result of the Inquire, and then disabled again.

In either instance, the CSP would respond with the information, and then add a Carriage Return. This automatically moves your computer screen down one line with every Inquire.

Return Value: Result of the expression evaluation.

Examples:

<i>?PI</i>	In the sample code, you are inquiring the current status of input number 1. If input number 1 was active, the CSP would transmit '1', if it was not active it would transmit '0'.
<i>?2*5+10</i>	Using the standard left to right evaluation of the math process, the result would be 20, which the CSP would display.
<i>?V4&gt;N7</i>	Comparisons return 1 if the comparison is true, or 0 if it is false. In this example if V4 was greater than the value of N7, then CSP would return 1. However, if V4 was equal to or less than N7, the CSP would return 0.

### **Command: A- Aux Codes**

---

Description: A codes are Auxiliary (Aux) codes used to perform functions that affect memory and screen functions.

**Command: D- DTMF Generate**

*Syntax: D(dtmf\_string)*

Function: DTMF tones are generated by this command. Remarks: The DTMF generator uses both tone generators A and B to generate Dual Tone Multi Frequency signals.

**Note:** Any tones that are being generated by the CSP when you involve this command, will be overwritten, and will not be restored when the DTMF code is complete. If, for instance, the CSP was generating a tone of 1000 Hz. The 1000 Hz tone would cease when the DTMF code started to send. After DTMF codes have been sent, the CSP turns both generators off.

The DTMF generator is capable of producing all 16 digits which consists of digits 0-9, letters A, B, C & D, as well as the # and \* signs.

When a variable is sent, e.g. [N1], the entire value is sent. So in the example, if N1 = 9624, then the DTMF code transmitted would be the digits 9, 6, 2, 4.

Return value: None.

Examples:

```

D135AB##*7 ;sends a DTMF string of 135AB##*7
D1634*1    ;sends a DTMF string of 1634*1
D[PS]      ;sends the data from board switch
D[RA]      ;sends the data from the outputs
D[V1]      ;sends the data stored in variable V1
D[N1]      ;sends the data stored in variable N1
D[X1]      ;sends the data stored in variable X1
D123[N5]7  ;if N5=456 then it would send the DTMF string 1234567
D246[PS]10 ;if the on-board switch is set to '8' the DTMF string 246810 would be
            sent.
D[N1][X4][PS][N7] You can use more than one variable inserter in a DTMF code.

```

Related items: Time of silence, in milli-seconds, before considering a code complete is stored in X20, DTMF mark speed in msec is derived from X21, while the Space timing is stored in X22, the status of the memory lock (0 unlocked, 1 locked ) can be read from X23. V20=DTMF Decoder Enable, V21=Length of received DTMF Code, V22=DTMF clear timer, V23=New DTMF code, V24=New DTMF Character.

**Command: d- Dtmf Dissect**

---

Function: Dissect and return value of last DTMF string.

**Syntax:**

*d(position)*

*d(position"L"length)*

Remarks: When a complete DTMF string is received, it is stored in a special register so the CSP can continue to receive new DTMF signals while allowing the previous string to be examined and used by your programs.

The d command retrieves the value of a DTMF digit, part of a DTMF string, or the entire string from the last fully received DTMF code.

The d command needs at least one number attached to it. The number is the beginning position in the string that the value needs to be retrieved from. Positions begin at 1 and go up to the length of the string.

When it is necessary to look at more than one position in the DTMF string, you must also include a length. This returns the value of the string to its starting "position", and continuing "length" positions. The L character is used to separate the starting location from the length.

When used with the non-digits in a DTMF string, the following is true...

A=10, B=11, C=12, D=13, \*=14, #=15

Return value: Value of DTMF string up to 65000;

Example: If a DTMF string of 1357902 has just been received.

<i>d3</i>	;would return 5, the number in position 3 of the string
<i>d6</i>	;would return 0
<i>d2L3</i>	;start at second position, length to return is 3, would return 357
<i>d1L6</i>	;start at position 1, length to return is 6, would return some invalid number since 135790 is beyond the 65000 limit.

Notes: Variables cannot be used in a Dissect command. i.e. dV4 cannot be used.

**Command: H- Hold**

---

Function: Hold the program for a specified time.

*Syntax: H(time\_in\_msec)*

Remarks: The H command will temporarily delay program execution for a specified duration. During this time the busy light will stay on, DTMF and Tone decoders will still operate, and any timers that are active will continue to operate.

After the Hold time is up, program execution will continue at the next statement following the Hold.

Return value: None

Example:

<i>H1000</i>	hold for 1000 milli-seconds (1 second)
<i>H5</i>	hold for 5 msec
<i>HN4</i>	hold for the time that is stored in variable N4

**Command: I- If**

---

Function: “If” decision. Evaluates a variable, input, output, etc., and decides if it is true or false. If it is true, then the statement right after the If is executed. In the case where the expression evaluates false, then the program jumps to the “E”, (which is called an Endif) Jumping over the expression in-between.

*Syntax: I(evaluation);<program to do if true>;E;*

*I(evaluation);<program to do if true>;@;<program to do if false>;E;*

Remarks: The expression to evaluate can be a simple one such as *IR1=1;<do this>;E;* or it can be a complex math expression such as *IN1+N2+N3>5;<do this>;E;* the math expression is evaluated from left to right.

A *true* expression is any that evaluates to a number that is greater than zero. A *false* expression is any that returns 0.

The @ character is called “Otherwise”, when used in an If expression, the program following it will be executed in case the If evaluation turns out to be false.

If's can be "nested". This means you can have an If statement located inside of another If. If's can be nested down to a level of 64000 statements.

Example:

```
|  I1+2=3;
|      SIs Equal to;
|  E
```

Since 1+2 does equal three, the statement within the If will be executed and the CSP will display "Is Equal to".

```
|  I1+2<5;
|      SIs Less Than;
|  @;
|      SIs Equal or Greater than;
|  E;
```

Since the value of 1+2 (3) is less than 5, the If evaluation is true. This causes the first statement to be executed and the Otherwise statement to be jumped over. So in this case the CSP would display "Is Less Than".

```
|  I1+6-2=N1;
|      SIs Equal;
|      IPC=10;
|      SSw. Equals Ten;
|  E;
|  @;
|      SIs not equal to;
|  E;
```

The math expression evaluates to 5. If the value stored in N1 is 5, then the first If statement will execute and the Otherwise will be ignored. If the value in N1 is not exactly 5, then the if statement will evaluate false, the first statement will be jumped over, and the statement in the Otherwise area will be executed.

```
|  IP1=1;
|      RB=RC-1;
|  E;
```

In the above example you use the status of the input switch number 1 in your evaluation. If the switch is being pressed, then the evaluation is true and the statement is executed. If the switch is not being pressed, the statement is jumped over. The statement in this case

will make the outputs in the lower nibble match the outputs in the upper nibble, minus one.

```
IRB>2;  
    SLower nibble is greater than zero;  
E;
```

The above example looks at the status of the lower nibble (first four) outputs and uses that information as the basis for the evaluation. If no outputs or only output number 1 is on, then the if evaluation is true and the statement is executed. If any more than that are on, then the evaluation is false and the statement is jumped over. Since you are looking at the lower nibble only, the status of the upper nibble (outputs 5 to 8) is never examined, and they will not effect this function.

Related items: Break-If character “i” (small letter “eye”) will force a jump out of all nested if’s and continue running the program after the top level endif.

---

**Command: J- Jump**

---

Function: Jump. Directs the program to jump to another Action and continue executing the program from there.

*Syntax: J(program number)*

Remarks: If a program number cannot be found, the program stops and returns to the main scan mode.

*Note: The Jump command must not be used in a timer string, as it could cause unpredictable results.*

A Jump is a one-way command, it is used to jump to another Action.

Example:

```
| IN1>10;  
|   J91;  
| E;
```

In the above example, the value in N1 is compared to the number 10. If N1 is greater than 10, the statement is executed and the program Jumps to Action 91. If the value in N1 is less than or equal to 10, the If evaluation is false. The statement is passed over and the jump never occurs.

```
| IX5)3;  
|   J80;  
| @;  
|   J90;  
| E;
```

Using the same basic premise as the first example, you compare the variable X5 to the static amount of 3. If X5 is greater than or equal to 3, then the If evaluation is true and a jump is executed to Action 80. In case the If evaluation comes out false, then the first statement is passed over and the Otherwise statement is executed. This causes a jump to Action 90 and the statement will continue running.

---

**Command: j- Jump, Multi**

---

Function: Jump string. Executes two or more Actions in a row.

*Syntax: j(Action address 1, Action address 2, ...)*

Remarks: When you need to execute up to 5 Actions in a sequence, the j command simplifies the matter. The jump string can hold up to 19 characters, including the commas separating the Action numbers.

The Action string is loaded into memory then each Action is executed in sequence. As each Action is completed (or terminated early), the next Action in the string is executed until all of the strings are done.

If an Action cannot be found, the remaining string is aborted.

Example:

```
| j101,41,180,102,44;
```

The above example shows a group of several Actions that may be run in sequence. First Action 101 is run, after which Action 41, 180, 102 and 44 are run one after the other. It is permissible for one of the above Actions to jump to other Actions with the J command. However, no other Actions can try to reload the multi-jump string command.

**Command: I- Leave-if**

---

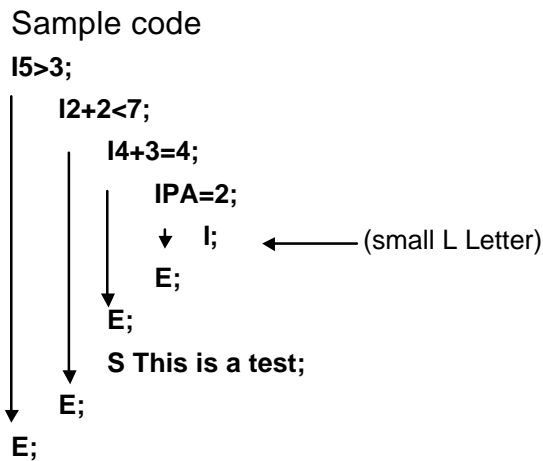
Function: Leave all of the current “If” statements.

Syntax:

I (small L Letter)

Remarks: The leave-if command is designed to break out of ‘If’ statements, even if they are nested. In some instances you may use nested Ifs to make an involved decision. The leave-if provides a simple way to jump from the most nested level all the way to the first level, then proceed after the endif ‘E’ at the end of the first level.

Example:



In this above example, you have 4 levels of nested If statements. The first three If evaluations are true, so GOS keeps going into the nest.

**Command: N- Variables**

---

Function: User variable, RAM.

*Syntax: N(variable number)*

Remarks: The N variables can contain numbers from 0 to 65000. The N variables are used to keep counts, totals, frequencies, or any other type of data needed to run an Action.

The N variables are dynamic in nature. When power is turned off to the CSP, all values stored in the N variables are lost.

A special command of double plus or double minus allows you to simply increment or decrement the N value by one. For example, if you want to increment the value stored in N5 by 1, you would enter N5=N5+1 or use the shorthand method of N5++.

When a variable is used in a string, it needs to be enclosed in brackets [ and ]. The strings where the N variables need to be enclosed in brackets include DTMF strings and “Say” strings.

Examples:

```
|N1=X7*3;
```

The variable N1 is assigned the math result of the value stored in X7 multiplied by 3. If the value in X7 was 100, then the value of 300 would be stored to N1.

```
|N1--;
```

The value in N1 would be decremented by one. If N1 contained 234 before the decrement, it would contain 233 after it.

```
|S The value of N4 is [N4];
```

Here you insert the value of a variable directly into a Say statement by using a variable inserter. When the Say statement comes upon the brackets, it does not print them, instead it will retrieve the value of the variable inside of the brackets, and print the value. If in the above example, N4 contained 9251 then the CSP would display “The value of N4 is 9251”.

```
|D1234[N6+20]4##;
```

A variable and the variable inserter can also be used in the DTMF generate command. As shown above, a normal DTMF code would be sent, but a value stored in memory slot N6 as part of the DTMF code would be used. In this example, you can see that the value is also augmented by a static value of 20 before it is sent. If, for example, N6 contained the value of 30 then the statement would generate the DTMF code 1234504###. The 50 was inserted as the result of the N6 value of 30 being added to 20.

```
|IN3>4;SGreater ;E;
```

The variable can be used in comparisons. In the above example, the value in N3 is compared against the static value of 4. If it is greater than 4, the Say statement is executed. If it is not greater, then the statement inside the If is passed over.

**Command: P- Inputs**

---

Function: Input ports and switch values.

*Syntax: P(location)*

Remarks: Inputs can be looked at in several different ways; full width, upper four bits, lower four bits, etc.

PA returns the value of the entire 8 bits. The value of PA can be from zero to 255.

PB returns the value of the lower four bits of the input port.

PC returns the value of the upper four bits of the input port.

PS returns the value of the on-board switch.

Examples:

**|N1=PA;**

The value of the entire input port is stored to variable N1. If input number 1 was active then 1 would be stored to N1. If input 2 were active then 2 would be stored to N1. If input 3 were active then the value of 4 would be stored to N1. If input 4 were active then the value of 8 would be stored to N1. The following table gives the port wide values for the inputs.

The full port input provides the decimal representation of the hexadecimal inputs. This is determined by adding the individual active inputs as follows;

Input	Value	Input	Value
1	1	5	16
2	2	6	32
3	4	7	64
4	8	8	128

As you can see, if you use a situation, for example, where inputs 3 and 7 are both active, their values of 4 and 64 are added to get the result of 68.

**|RC=PC;**

The upper nibble of the inputs (inputs 5 to 8) are directly assigned to the upper nibble of the outputs (outputs 5 to 8).

```

IPS>5;
  J101;
  @;
  J106;
E;

```

The on-board switch of the CSP-105 is examined and used in the If evaluation. If the switch setting is greater than 5, the evaluation is true and the Jump to Action 101 would be executed. If the switch was set to a number lower than 6, the evaluation would be false and the Otherwise statement would execute, which would cause a Jump to Action 106.

---

### **Command: Q- Timers**

---

Function: Loads and starts independent timers.

*Syntax: Q(timer number),<total time in seconds>,<on time in seconds>,<off time in seconds>, <on code>, <off code>, <ending code>;*

Remarks: The CSP contains several independent timers, once these timers are started, they can be left alone to complete their function.

The times may be from 1 second to 65000 seconds.

On, Off, and Ending codes may contain codes up to 19 characters in length.

All codes must be contained within single quote marks.

Examples:

```
| Q1,180,5,7,'R1=1','R1=0','R1=0;N1++;'
```

Load timer #1 for a maximum life of 180 seconds (3 minutes). The on time will be 5 seconds and the off time will be 7 seconds. R1=1 is the on code, and R1=0 is the off code. When the timer life is complete after 3 minutes, R1=0 will turn off output number 1, and N1++ will increment that variable to keep track of how many times the timer was run.

```
| Q1,180,180,0,'RA=RA+1',' ','IRA>5;RA=0;E;'
```

Load timer #1 for a maximum life of 180 seconds. You may execute one instruction when you load the timer, and then wait until the end of the 3 minutes before executing a final instruction. To perform this simply make the on time equal to the total life time, and make the off time equal to zero. Since there will not be an off code used, simply enter a space between the single quotation marks for the off code.

In the above example, the timer is started and the outputs are incremented by one, then the timer waits for three minutes. After the timer life is up, the ending code checks if the outputs are over 5 in value. If they are, all outputs will be set to zero. If the output port

value is not greater than 5, the If statement is jumped over and the outputs are left disregarded.

```
|Q2,60,1,1,'R1','R1','R1=0';
```

Load timer #2 for a total life of 60 seconds (1 minute). You want an on and off time of 1 second each so you enter 1 second for each. The on and off codes are the same, 'R1'. This will cause output number 1 to reverse its previous state each time it is run (toggle). At the end of the 1 minute time, the ending code will reset output number 1 to off with the code 'R1=0'.

```
|QV3,N2,V4,V5,'R4=1;R6=0','R4=0;R6=1','R4=0;R6=0';
```

Variables can be used in timers just as easily as static values. In the above examples, V3 would contain which timer to load, #1 or #2. N2 contains the value of the total timer life, while V4 and V5 contain the on and off times respectively. Examining the codes, you will see that outputs 4 and 6 will be turned on opposite of each other, and they will reverse states when the on or off time is complete.

### ***Command: q- Timers, Control***

---

Function: Restarts a previously loaded timer.

***Syntax: q(timer number), <total time in seconds>***,

Remarks: Many times it is only necessary to restart a timer. Once the individual codes and timing have been loaded into memory using the 'Q' (capitol Q) command, you can use the 'q' (small q) command to restart (or stop) a timer.

If a timer is running, you can stop the timer by entering

***q(timer number),0,***

*Note, it is important to place the comma after the total life time.*

Examples:

**q1,60,**

Restart timer #1, life of timer will be 60 seconds.

```
|q2,N1,
```

Restart timer #2, life of timer will be the value stored in N1.

|q1,0,

Stop timer #1. When a timer is stopped prior to its normal ending, the Ending Code will still be executed.

|qX7,V3,

Work on the timer that variable X7 holds, either 1 or 2. Either restart it with a new time if V3 contains a value greater than 0, or turn the timer off if V3 is equal to zero.

### **Command: R- Outputs**

---

Function: Turns outputs on/off.

*Syntax: R(bit number)*

*R(upper or lower nibble)*

*R(port)*

Remarks: The outputs are controlled by this command.

**R**(individual bit number 1 to 8)

Affects only one output. When used with only the bit number such as R3, this command toggles the bit between on and off with each statement. When used with an expression, the expression is assigned to the bit. If the expression evaluates to zero then the output is set to zero. If the expression is not zero, then the output is set to one.

RA works with the entire 8 bits at the same time. Since it is 8 bits, the value can be 0 to 255. RA without any expression will turn all outputs off. If a value greater than 255 is stored to RA, unpredictable outputs will occur.

RB affects the lower group of bits 1 to 4, it is referred to as the lower nibble.

RC affects the upper group of bits 5 to 8, it is referred to as the upper nibble.

The following table can be used to aid your calculations.

Value	Output
1	1
2	2
4	3
8	4
16	5
32	6
64	7
128	8

To use the table, first establish which outputs you want to turn on. Then add up the values for each output.

If, for example, you want outputs 8, 5 and 1 to turn on, you would add the associated values for each output, resulting in  $128+16+1$ . The sum of 149 is the value that you would use to turn on the outputs 8, 5 & 1.

Examples:

*RI;* this will toggle (reverse states) on output 1

*RA=134;* causes outputs 8, 3 and 2 to activate. Any other outputs that were on will turn off.

*RA=V10+10;* cause the outputs to change to value in variable V10 plus 10.

*RB=15;* working with the lower nibble only, causes outputs 1,2,3 and 4 to turn on.

*RC=RB;* makes the upper output nibble the same as the lower output nibble.

*IRA>30;STest;E;* Using the information about the full output port, checks to see if the value of the outputs are greater than the value 30. If the evaluation is true the CSP will display "Test".

*RA=PS;* Make the outputs match the value of the on-board switch.

*RC=PS;* Make the upper nibble equal to the value of the on-board switch.

**Command: S- Say**

---

Function: Sends text statements to the LCD, printer, or computer screen.

*Syntax: S(text to send)*

Remarks: Send will transmit the text to the LCD, out the serial port, or both depending upon the status of the LCD Enable and Serial Enable variables.

The text can also transmit the current settings of variables when the variable is enclosed in [] brackets. If, for example, N1 contains the value of 27910 then the statement

**|SCurrent value is [N1]**

would display Current value is 27910

By putting a caret ^ before any character, the character is sent as a control code. ^J is a carriage return for computer screens and printers.

Examples:

**|SThis is a test**

CSP will display This is a test.

**|SThe last DTMF code was [D]**

Use the variable inserter to display the entire DTMF code that was last received. If in this example, the last DTMF code that was received was \*27194 then the CSP would display The last DTMF code was \*27194.

**|SCurrent value of port A is [PA]**

Again using the variable inserter, this time you insert the value of the input port into the text statement. If only switch 8 was active when this statement was run, the CSP would display Current value of Port A is 128.

**|SMath can be done inside of say, total is [N1+12\*2]**

Complex math statements can be run from inside the variable inserter as shown above. Assuming that variable N1 is equal to 100 then the CSP would display "Math can be done inside of a say, total is 2400". The math performed inside of a variable inserter follows the same left to right sequence as a normal math function.

**|SCurrent switch setting is [PS]**

The variable inserter is used to display the current value of the board switch of the CSP-105. If the switch was set to C, then the CSP would display `Current switch setting is 12.`

### ***Command: T- Generate Tones***

---

Function: Generates audio tone.

***Syntax: T(generator A or generator B)***

Remarks: The CSP contains two independently controllable tone generators. They can generate tones from 250 Hz to 5000 Hz.

The CSP works only with real numbers, no fractions.

Examples:

**|TA300;**

Using generator A, produce a 300 Hz tone.

**|TAN1;**

Using generator A, produce a tone whose frequency is based on the value in N1.

**|TBV3;**

Using generator B, produce a tone whose frequency is based on the value in V3

**|TBPS\*100;**

Using tone generator B, produce a tone that is based upon the current switch setting of the on-board switch, multiplied by 100. If, for example, the switch was set to 9, then the CSP would generate a tone frequency of 900 Hz.

**|TAN1+N2;**

Generate a tone from generator A. The frequency of the tone produced will be the sum of variables N1 and N2. If, for example, N1 was 1000 and N2 was 500. The CSP would generate the frequency 1500 Hz.

**Command: U- Until**

---

Function: Loop “Until” command.

*Syntax: U(evaluation);<statements to execute until true>;E;*

*UT(max time),(evaluation);<statements>;E;*

Remarks: The Until command will perform a statement until the evaluation returns a true. True is any number other than zero.

The Until command can also have a maximum time associated with it. If the evaluation does not return true within the time period specified, it will automatically break out of the loop at the end of the time.

You can also break out of a loop by using the ‘b’ break command.

All Until commands must end with the E command.

Examples:

```

|  UP1=1;
|    A33;
|    SPress Sw. 1 to continue;
|    H250;
|  E;
```

Until input switch #1 goes active, keep clearing the screen and displaying “Press Sw. 1 to continue”, then wait 250 milli-seconds (1/4 second), after which you go back to the start of the loop and check to see if input #1 is pressed yet.

```

|  UT2000,P7=1;
|    A33;
|    SPress Sw. 7 within 2 seconds;
|    H250;
|  E;
```

The example above includes a maximum time in the Until statement. If the user does not press the switch that is attached to input number 7, within the 2000 milli-second (2 second) limit, then the maximum time automatically forces the loop to terminate and the Action continues running any statements that may be after the ending ‘E’.

```

UT5000,P5=1;
  IP4=1;
    b;
  E;
E;
IP5=1;
  SSw. 5 Pressed;
E;
IP4=1;
  SSw. 4 broke loop;
E;

```

In the above example, the Until loop is set up to be broken if input number 5 goes active, or if the loop runs for 5 seconds. Inside of the loop, if input number 4 goes active, the 'b' breaks the loop.

After the loop is exited by any of the above means, you now check to see how the loop was exited. If it was exited by input number 5 the LCD will display Sw. 5 Pressed.

If the loop was broken by input number 4 going active, the display will show Sw. 4 broke loop.

### **Command: V- Variable**

---

Function: RAM variables that are controlled by the user's program, or that contain settings that can vary the action of the CSP, such as turning on and off the frequency counter, etc.

Syntax: **V(variable number)**

Remarks: The V variables are used to store smaller numbers. They can hold from 0 to 255. Only real numbers can be stored, no negatives or fractions in the CSP-105.

Since they are stored as RAM, any values will be lost when power is removed from the CSP. When power is restored the pre-determined variables are set-up with default values.

Examples:

V1=10;        Store the value of 10 to variable V10.

V9=100-10\*2;        Store the result of 100-10\*2 (180) to variable V9.

V7=V1+V3;    Store the result of adding the variables V1 and V3.

V8++;        Increment the value of V8 by one. If it was 3, it would now be 4.

**Command: W- While**

---

Function: Loop “While” command.

Syntax: **U(evaluation);<statements to execute until true>;E;**  
**UT(max time),(evaluation);<statements>;E;**

Remarks: The While command will perform a statement while the evaluation returns a true. True is any number other than zero.

The While command can also have a maximum time associated with it. If the evaluation does not return false within the time period specified, it will automatically break out of the loop at the end of the time.

You can also break out of a loop by using the ‘b’ break command.

All While commands must end with the E command.

Examples:

```

WP1=0;
  A33;
  SPress Sw. 1 to continue;
  H250;
E;

```

In the above example, input number 1 is checked by the While evaluation. As long as the input is not active, zero, then the loop will continue. When the input turns from inactive to active, then the loop will break.

```

WT5000,P5=0;
  IP4=1;
    b;
  E;
E;
  IP5=1;
    SSw. 5 Pressed;
E;

```

In the above example, you now have a time limit attached to the While evaluation. In this instance, the time is set for 5000 milli-seconds. If the loop is not broken by input number 5 going active, before the 2 second count is reached, a break is forced by the timer.

The Action then checks to see if input #5 was the one to break the loop, and displays finding.

**Command: X - Variable**

---

Function: User variable, EEPROM

Syntax: **X(variable number)**

Remarks: The X variables can contain numbers from 0 to 65000. The X variables are used to keep counts, totals, frequencies, or any other type of data needed to run an Action.

The X variables are static in nature, when power is turned off to the CSP, all values stored in the X variables is saved.

A special command of double plus or double minus allows you to simply increment or decrement the X value by one. For example, if you want to increment the value stored in X2 by 1, you would enter X2=X2+1 or use the shorthand method of X2++.

When a variable is used in a string, it needs to be enclosed in brackets [ and ]. The strings where the X variables need to be enclosed in brackets include DTMF strings, and “Say” strings.

Examples:

*X1=N7\*3;*                    Store the result of variable N7 multiplied by 3, to X1.

*X1--;*                        Decrement value of X1 by one. If X1 was 92 it will now be 91.

*S The value of X4 is [X4];*    Insert the value of X4 into the Say statement.

*D1234[X6+20]4##;*    Add 20 to the value in X6 and use the result in a DTMF code.

*IX3>4;SGreater ;E;*    Evaluate X3. If it is greater than four, display Greater on LCD.

# Appendix A

## Main Codes

Code	Desc.	Example	Notes
?	Inquire	?X21	Used to communicate with the CSP from other computers
!	Done	V1++;SHello;!	Used to designate the end of a function, DTMF code, or Sw.
;	Delimiter	X5=1;V2++;	Separates statements in a function.
@	Otherwise	IP1=1;V1++;@;V2++;E	Runs statements when the If evaluation proves false.
A	Aux Codes		Secondary operations. See Aux codes for further information on all of the Aux codes available.
b	Break	W1;IP1=1;b;E;E	Provides method of breaking out of a loop. Function continues after the end command.
D	DTMF Generate	D12345	Generates a DTMF signal comprised of the code following the D command. See variables X21 & X22 for timing.
d	DTMF Dissect	d3 or d3L5	Returns value of a sub-string using the most recently received DTMF code.
E	Endif Endwhile Enduntil	IP1=1;N7++;E WP1=1;SY;E UP1=0;SN;E	Designates the end of a loop or decision statement.
F	FSK	FString of data	Transmits Frequency Shift Keying data. All formats.
H	Hold	H1000	Delays program execution for time stated. Time in msec.
I	IF	IP1=1;X1++;E	Decision command. One or more statements contained inside of the If. Must have an Endif 'E' as final.
J	Jump	J101	Redirects program to run another function.
Code	Desc.	Example	Notes

<b>J</b>	Multi-jump	<i>j101,102,103;</i>	Runs multiple functions in a row.
<b>I</b>	Leave	<i>IP1=1;!;E</i>	Leaves a group of nested If's in a logical manner.
<b>N</b>	N Variable	<i>N4=2+5*1000</i>	Memory Variable. Values from 0 to 65535. Data in the N variables is not retained when power is removed.
<b>P</b>	Inputs	<i>IPA=12;V1++;E</i>	Searches the input port as directed. Returns the value for further use in a decision, Say command, etc.
<b>Q</b>	Timers	<i>Q1,200,1,1,....</i>	Loads timers with information and times. Timers then operate independently of main program.
<b>q</b>	Restart/ Halt Timers	<i>q1,0, q2,200,</i>	Restarts a previously loaded timer when the second number is greater than zero. Stops timer when zero.
<b>R</b>	Outputs	<i>R1=1; RA=136</i>	Controls the 8 bit output lines. Either individually, upper or lower nibble, or full port.
<b>S</b>	Say	<i>SHello World</i>	Sends text and variable information to LCD screen and/or other computers or printers.
<b>T</b>	Tone Gen.	<i>TA1200; TB330</i>	Independently operates the dual tone generators.
<b>U</b>	Until	<i>UP1=1;N1++;E</i>	Until loop. Executes statements Until evaluation is True.
<b>V</b>	V Variable	<i>V1=143; V1++</i>	Memory Variable. Values from 0 to 255. Data in the V variables is not retained when power is removed.
<b>W</b>	While	<i>WP1=0;N3++;E</i>	While loop. Executes statements While evaluation is True.
<b>X</b>	X Variable	<i>X3=4200, X5++</i>	Memory Variable. Values from 0 to 65535. Data in the X variables <u>is</u> retained when power is removed.

**Aux Codes**

Code	Description	Example	Notes
<b>A1</b>	Restart CSP	A1	Reboots the CSP. Same as turning power off and on.
<b>A3</b>	Baud Rate	A3,50	Changes serial port baud rate. 2400=50, 1200=51, 600=52
<b>A5</b>	Clear Dtmf	A5	Clears Registers associated with decoding DTMF signals.
<b>A6</b>	Version	A6	Sends GOS Version # via Serial Output.
<b>A7</b>	Debug On	A7	Turns program debugger on so you see program execution.
<b>A8</b>	Debug Off	A8	Turns program debugger off.
<b>A12</b>	Timer Life	A12	Transfers current remaining timer lives into N8 and N9.
<b>A30</b>	Row & Col	A30,1,1 0	Adjusts the LCD cursor position.
<b>A33</b>	Clear LCD	A33	Clears LCD screen, positions cursor at row 1, column 1.
<b>A51</b>	Program Tones	A51,1;1 000,30, 2000,10 ,161	Stores information to the Tone code memory. A51,<slot>;<freq A>,<time A>,<freq B>,<time B>,<function>
<b>A52</b>	Read Tones	A52,slot	Reads all Tone code memory information stores it in N10,V10,N11,V11,and V9 for job number.
<b>A54</b>	Store Tone	A54,6	Stores variables in memory, as a tone group.
<b>A55</b>	Clear Tones	A55,55	Erases All Tone code groups from memory.
<b>A56</b>	Reset Tones	A56	Resets to zero, all variables associated with tone decoding
<b>A80</b>	Function Address	A80,0	Changes the memory pointer in the Function Memory.
<b>A81</b>	Write Funct.	A81;{10 1}S	Writes the entire string after the semi-colon to Function memory.

<b>A82</b>	Read Funct.	A82,0	Reads and displays Function memory.
<b>A83</b>	Lock Mem.	A83,83	Locks Pgm memory. Prevents Writes and Reads.
<b>A85</b>	Action Address	A85,0	Changes the memory pointer in the Action Memory.
<b>A86</b>	Write Action	A86;D101,	Writes the entire string after the semi-colon to Action memory.
<b>A87</b>	Read Action	A87	Reads and displays Action memory.
<b>A139</b>	Erase Funct	A139,0	Erases the Function memory starting at a given address.
<b>A148</b>	Erase Action	A148,0	Erases the Action memory starting at the given address.
<b>A157</b>	Reset CSP	A157,157	Erases all memories, Resets variables to defaults, restarts. Unlocks memory if it had been locked.

**“V” Variables**

Num	Used As	Default	Notes
0	User	0	
1	User	0	
2	User	0	
3	User	0	
4	User	0	
5	User	0	
6	User	0	
7	User	0	
8	User	0	
9	valid_tone_fun	0	Available for general use, unless are adding tone codes with A51.
10	‘A’ Tone Time	0	Received tone time in 10th’s of second. Maximum time is 25.5 seconds, then wraps to 0.
11	‘B’ Tone Time	0	Received tone time in 10th’s of second. Maximum time is 25.5 seconds, then wraps to 0.
12	Send Freq Cntr to Serial	0	When this is set to 1, and the frequency counter is active (see V15) then all frequency counter data is updated to serial port twice per sec.
13	Tone Decoder Enable	1	Used to turn the Tone Decoder on and off. Zero is off, one is on.
14	Valid Tone	0	0 indicates no valid tone match. Any other number indicates match and specifically the number is the function that is run.
15	Freq. Counter	0	Turns the frequency counter option on and off. Zero is off, 1 is on.
16	Current Row	1	Provides data on the current line position of the LCD cursor. LCD rows start at row 1.

<b>17</b>	Current Col.	1	Provides data on the current column position of cursor. LCD columns start at column 1.
<b>18</b>	LCD Off / On	1	Turns the LCD driver on and off. When Off any S (Say) strings are ignored. Zero is Off, one On.
<b>19</b>	Serial Port Off / On	1	Turns Serial Port driver on and off. When Off, any S (Say) strings are ignored. Zero is off.
<b>20</b>	Dtmf Decoder Enable	1	Used to turn the DTMF Decoder on and off. Zero is off, 1 is on.
<b>21</b>	Number of DTMF Digits	0	Contains length of current DTMF code.
<b>22</b>	Dtmf timer	0	Timer used to count the time-out delay.
<b>23</b>	New Dtmf Code	0	Set to 1 when a new DTMF code has been received.
<b>24</b>	New DTMF Character	0	Set to 1 when a new DTMF character has been received.

**“N” Variables**

All N variables are in RAM, and any data stored in them will not be retained when power is removed. The values stored can be integers from 0 to 65535, inclusive.

Num	Used As	Default	Notes
0	User	0	
1	User	0	
2	User	0	
3	User	0	
4	User	0	
5	User	0	
6	User	0	
7	User	0	
8	User/Special	0	Also used as temporary storage when using Aux code A80.  Used to hold Remaining Timer #1 life when A12 command.
9	User/Special	0	Also used as temporary storage when using Aux code A85.  Used to hold Remaining Timer #2 life when A12 command is run.
10	'A' Tone Freq.	0	When a valid Tone code group has been detected, this will hold the First tone frequency.
11	'B' Tone Freq.	0	When a valid Tone code group has been detected, this will hold the Second tone frequency.

**“X” Variables**

All X variables are in EEPROM, so they will be retained even when power is removed. The values stored can be integers from 0 to 65535, inclusive.

Num	Used As	Default	Notes
0	User		
1	User		
2	User		
3	User		
4	User		
5	User		
6	User		
7	User		
8	User		
9	User		
10	User		
11	LCD Lines	1	Maximum number of lines on the LCD Screen.
12	LCD Width	16	Character width of the LCD Screen.
13	Serial Baud	50	Baud Rate Code. 50=2400, 51=1200, 52=600, 53=300.
14	Start-up Funct.	0	Function number to run when power is first applied.
15	Reserved		
16	No DTMF Match	99	Function to run when a DTMF code cannot find a match.
17	Bandwidth	10	Number of Hz a Tone signal can be off freq., while still valid.
18	Low Bandpass	250	Frequency in Hz, where the digital filter rolls off tones.

<b>19</b>	High Bandpass	4000	Frequency in Hz, after which all tones are rolled off.
<b>20</b>	DTMF Time-delay	6	Time in 10 <sup>th</sup> s of a second that defines the period of inactivity before a DTMF code is considered complete. 6 = .6 seconds.
<b>21</b>	DTMF Mark	100	Time in milli-seconds that a DTMF character is generated. 100 = .1 seconds.
<b>22</b>	DTMF Space	100	Time in milli-seconds on inter-character silence for DTMF. 100 = .1 seconds.
<b>23</b>	Memory Locked	0	Holds status of memory lock. Refreshed upon power up or software restart.