



New Features

CID1500 Ver 2.43a

FSK Based Caller ID Simulator Software

1) Support for ETSI EN 300 659-3 Caller ID Standard

New message types and parameters have been added to the CID1500 in order to support the latest ETSI standard. Draft ETSI EN 300 659-3 (version 1.3.1) describes the data link message and parameter coding for display and related services sent by the Local Exchange. Two new message types have been added to the CID1500 program, along with eleven new parameter types. The following list shows the all of message and parameter types supported.

Message Type	Hexadecimal Coding	
Call Setup	80	
Message Waiting Indicator	82	
Advice of Charge	86	new
Short Message Service	89	new

Parameter Types	Hexadecimal Coding	
Data and Time	01	
Calling Line Identity	02	
Called Line Identity	03	
Reason for Absence of Calling Line Identity	04	
Calling Party Name	07	
Reason for Absence of Calling Party Name	08	
Visual Indicator	0B	
Message Identification	0D	new
Last Message CLI (Calling Line Identity)	0E	new
Complementary Date and Time	0F	new
Complementary Calling Line Identity	10	
Call Type	11	
First Called Line Identity	12	
Number of Messages	13	
Type of Forwarded Call	15	
Type of Calling User	16	
Redirecting Number	1A	
Charge	20	new
Additional Charge	21	new
Duration of the Call	23	new
Network Provider Identity	30	new
Carrier Identity	31	new
Selection of Terminal Function	40	new
Display Information	50	new
Service Information	55	new
Extension for Network Operator Use	E0	



The added parameter types are viewed and modified within the CID1500 by using a new window. The following figure shows the window containing the various settings for the additional parameters.

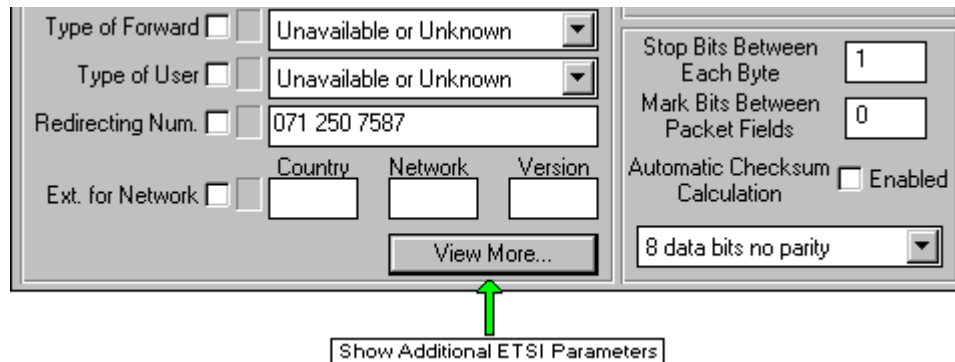
The screenshot shows a window titled "Additional ETSI Packets" with the following settings:

- Message Ident. Remove [0]
- Last Message CLI 071 250 7587
- Comp. Date & Time Month: 02, Day: 27, Hour: 09, Minute: 27, Sec.: 29
- Call Duration Hour: 00, Minute: 00, Sec.: 00
- Provider Identity Network Provider
- Carrier Identity Carrier Identity
- Terminal Function Connection Type: Not Identified
- Service Information Not Active
- Display Information Unknown
- Message Stored Information
- Information to display: [Empty text box]
- Charge Currency: ITL, Type: Current
- Free of Charge
- Sub-Total (AOC-D)
- Credit/Debit Card
- Charging Information Not Available
- Charged Units
- Cost: 0
- Additional Charge Currency: ITL, Type: Current
- Free of Charge
- Sub-Total (AOC-D)
- Credit/Debit Card
- Charging Information Not Available
- Charged Units
- Cost: 0

Operating in the same manner as all the previously supported parameters, clicking the mouse on the check box next to the parameter name enables or disables them. The data sent with each parameter is controlled by the various text fields and drop-down lists.

To view this new window, the CID1500 must be operating under the ETSI standard setting. It is not available under either the Bellcore (Telecordia) or Australian Caller ID standards. To change the operating standard to ETSI, select the [CONFIGURATION] menu followed by the [CALLER ID STANDARD] and [ETSI ETS 300659-1/2/3] selections.

Bringing the window to the forefront is done by either selecting the [WINDOW] [ADDITIONAL ETSI PACKETS] menu command, or pressing the "View More" button on the CID Packet Format window (as shown below).



All of the additional ETSI message types and parameters are fully supported in the CID1500 scripting language. See the next section for details on its usage.

2) Scripting Language Enhancements

New ETSI Messages:

To support the new message types under the ETSI standard, the MESSAGE command has been expanded. Previous versions of the CID1500 only allowed "Call_Setup" and "Message_Waiting" as the message type value. This has been expanded to include "Advice_Of_Charge" and "Short_Message_Service".

Syntax: *MESSAGE type*

Where: "type" is one of the following:
 Call_Setup
 Message_Waiting
 Advice_Of_Charge
 Short_Message_Service

New ETSI Parameters:

As with the MESSAGE command, to support the new ETSI parameter values, the PACKET command has been expanded. The "type" field accommodates all of the existing parameters in addition to the eleven new ones. As some of the new parameters have a more complex structure, the "operator" and "value" fields allow more selections.

Syntax: *PACKET type operator [value]*

Where: "type" specifies one of the following parameters.
 Date_&_Time, Calling_Number, Number_Absence,
 Calling_Name, Name_Absence, Visual_Indicator,
 Called_Line, Call_Type, Network_Sys_Status,
 Comp_Calling_Line, First_Called_Line,
 Forwarded_Call_Type, Calling_User_Type,
 Redirecting_Number, Network_Operator_Ext,
 Message_Identification, Last_Message_CLI,



Comp_Date_&_Time, Charge, Additional_Charge, Call_Duration, Provider_Identity, Carrier_Identity, Terminal_Function, Display_Information, Service_Information

Where: "operator" specifies what action is being taken with the parameter. All parameters use both the "Enabled" and "Disabled" operators to send or not send the parameter respectively. In addition, each parameter may have various unique operators for changing the data contents. The following table lists all the available operators for each parameter.

Where: "value" specifies the setting of a parameter based on the "operator" field. Depending on the operator type, the value field may not be required. The following table lists the acceptable "value" field settings for each of the parameters.

Parameter Name	Valid PACKET Command Operators & Value Settings
Date_&_Time	Enabled, Disabled, Value Value: "mmdhmm" 8 character string representing month, day, hour, and minute
Calling_Number	Enabled, Disabled, Value Value: up to 50 character string
Number_Absence	Enabled, Disabled, Value Value: Private, Out_of_Area
Calling_Name	Enabled, Disabled, Value Value: up to 50 character string
Name_Absence	Enabled, Disabled, Value Value: Private, Out_of_Area
Visual_Indicator	Enabled, Disabled, Value Value: Activate, Deactivate
Called_Line	Enabled, Disabled, Value Value: up to 50 character string
Call_Type	Enabled, Disabled, Value Value: Voice_Call, Ring_Back_When_Free, Calling_Name_Delivery, Call_Return, Alarm_Call, Download_Function, Reverse_Charging_Call, External_Call, Internal_Call, Monitoring_Call, Message_Waiting_Call
Network_Sys_Status	Enabled, Disabled, Value Value: 0_Messages, 1_Messages, ... ,255_Messages
Comp_Calling_Line	Enabled, Disabled, Value Value: up to 50 character string
First_Called_Line	Enabled, Disabled, Value Value: up to 50 character string
Forwarded_Call_Type	Enabled, Disabled, Value Value: Unavailable_or_Unknown, Forwarded_on_Busy, Forwarded_on_No_Reply, Unconditional_Foward, Deflected_after_Alerting, Deflected_Immediate,



Can't_Reach_Mobile

Calling_User_Type	Enabled, Disabled, Value Value: Unavailable_or_Unknown, Voice_Call, Text_Call Virtual_Private_Network, Mobile_Phone Mobile_Phone_and_VPN, Fax_Call, Video_Call E-Mail_Call, Operator_Call, Ordinary_Calling Priority_Subscriber, Data_Call, Test_Call, Telemetric_Call Payphone
Redirecting_Number	Enabled, Disabled, Value Value: up to 50 character string
Network_Operator_Ext	Enabled, Disabled, Value Value: "cccnvvv" 10 character string representing the country network and version fields.
Message_Identification	Enabled, Disabled, Value, Type Type: Remove, Reference_Only, Add Value: Number from 0 to 65535
Last_Message_CLI	Enabled, Disabled, Value Value: up to 50 character string
Comp_Date_&_Time	Enabled, Disabled, Value Value: "mmddhhmm" or "mmddhhmmss" strings representing the month, day, hour, minute, and optionally seconds.
Charge	Enabled, Disabled, Currency, Type, Cost, Units, PricePerUnit, FreeOfCharge, SubTotal, CreditDebit, NolInfo, ChargeUnits Currency: 3 character string representing currency code Type: Current, Accumulated, Extra Cost: up to 10 character string Units: up to 5 character string PricePerUnit: up to 5 character string FreeOfCharge: Yes, No SubTotal: Yes, No CreditDebit: Yes, No NolInfo: Yes, No ChargeUnits: Yes, No
Additional_Charge	Enabled, Disabled, Currency, Type, Cost, Units, PricePerUnit, FreeOfCharge, SubTotal, CreditDebit, NolInfo, ChargeUnits Currency: 3 character string representing currency code Type: Current, Accumulated, Extra Cost: up to 10 character string Units: up to 5 character string PricePerUnit: up to 5 character string FreeOfCharge: Yes, No SubTotal: Yes, No CreditDebit: Yes, No NolInfo: Yes, No ChargeUnits: Yes, No
Call_Duration	Enabled, Disabled, Value Value: "hhmmss" 6 character string representing hours, minutes, and seconds.
Provider_Identity	Enabled, Disabled, Value Value: up to 50 character string
Carrier_Identity	Enabled, Disabled, Value Value: up to 50 character string
Terminal_Function	Enabled, Disabled, Mode, Type, Value



	Mode: Connection_Type, Subscriber_Number, SubAddress Type: Not_Identified, Voice_Call, Fax_Call, Data_Call, Video_Call, E-Mail_Call, Telemetric_Call, Text_Call Value: up to 50 character string representing the subscriber number or sub address values
Display_Information	Enabled, Disabled, Type, Stored, Text Type: Unknown, Positive_Ack, Negative_Ack, Advertisement Network_Information, User_Information Stored: Yes, No Text: up to 252 character string
Service_Information	Enabled, Disabled, Value Value: Not_Active, Active

Note that the above listed parameter types, operators, and values are only applicable when operating under the ETSI standard. If the CID1500 software is set to the Bellcore or Australian standards, the acceptable field values for the PACKET script command are specific to those standards.

Modified Command: INPUT

The INPUT command provides a means for the user to input either numeric or string information into a script program. This command has been expanded to include the option of displaying a “pop up” menu. The pop up menu can contain from 1 to 20 user defined selections for the operator to select from. The value of selection is returned as a numeric value corresponding to the menu selection number. A script program can then take various actions based on the returned value. The syntax for displaying a pop up menu is as follows:

Syntax: *INPUT PopUpMenu variable “selection 1” [“selection 2 to 20”]*

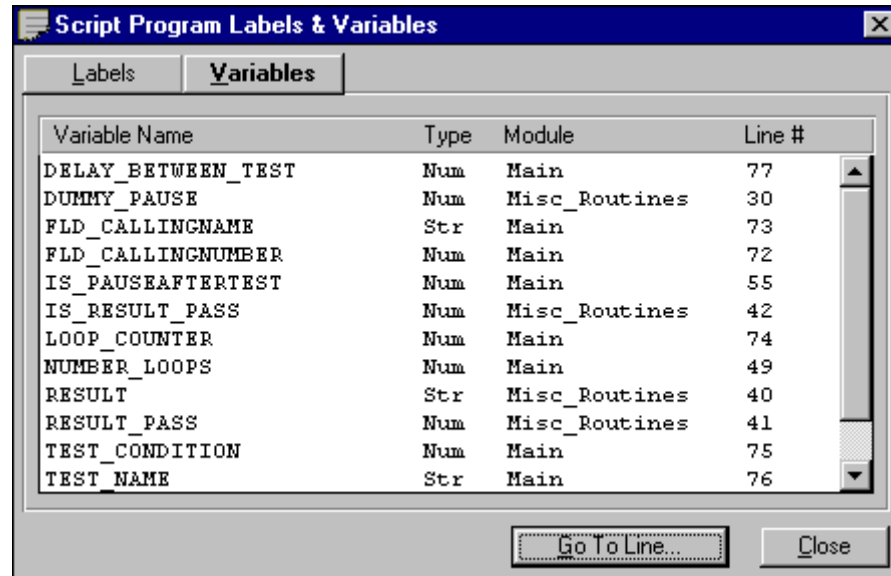
Where: “variable” specifies a numeric variable defined else where within the script program. The operator’s menu selection is returned in the numeric variable as a number corresponding to the chosen selection. If the operator does not click the mouse on any of the menu selections, the variable is set to zero.

Where: “selection 1” is a character string that represents the text shown on the first line of the pop up menu.

Where: “selection 2 to 20” are optional character strings that represent additional selections shown on the pop up menu.

Tracking Program Labels & Variables

A new tool assists in the managing of script program labels and variables. As script programs become large and complex, it becomes difficult to track a growing number of program labels and variables. Selecting the [SCRIPTING] [LABELS & VARIABLES] menu command displays a window similar to below.

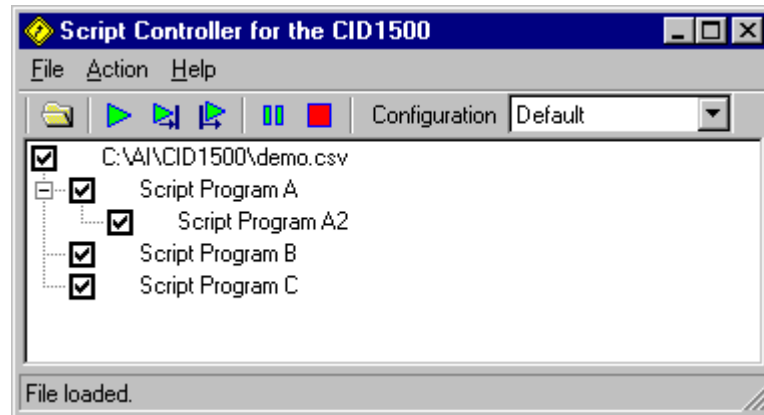


All of the current program's labels and variables are listed in alphabetical order. In addition, the module name and declaring line number are also included in the list. The above figure shows a listing of variables for an example script program. To see the defined labels, simply click the mouse on the "Labels" button at the top of the window. Likewise, clicking "Variables" lists all the declared variables. Clicking the "Go To Line" closes the window and moves the script editor cursor to the selected label or variable declaration.

3) Script Control Program

A new and freely available program can be used to selectively execute any number of CID1500 script programs. The Script Control Program controls the CID1500 software, and sends commands to load and execute any script program file. This can be useful in performing complex Caller ID conformance testing, where a large number of tests are broken down into smaller and more manageable CID1500 script programs. The Script Control Program can then selectively execute the smaller script programs in sequence. When required, any CID1500 script program can be skipped, or only a selection of programs can be executed.

The Script Control Program includes a simple demonstration file to show its operation. The demonstration file contains a list of script files to execute along with configuration settings. As shown in the figure below, the loaded file (demo.csv), defines four different script programs. These are called Script Program A, Script Program A2, Script Program B, and Script Program C.



The script programs can be given a hierarchical structure as shown with Script Program A and Script Program A2. This does not effect the execution of the script programs, but rather is used only for visually grouping the script programs. The check box beside each listed program can be checked or unchecked by clicking the mouse on it. If checked, the script program is “enabled” and may be executed.

Running the CID1500 script programs is simply done by clicking the green start arrow on the tool bar, or selecting the [ACTION] [RUN ALL ITEMS] menu selection. Each script program that is “checked” is then executed in turn.

The drop-down list marked “Configuration” on the tool bar is used to select which script programs are enabled or disabled. The demonstration (demo.csv) file defines 4 different configurations. They are “Default”, “Setting 1”, “Setting 2”, and “Setting 3”. By changing the configuration selection, different script programs are enabled or disabled. This provides a simple means for selecting pre-defined testing configurations.

Virtually any spreadsheet software can create the files loaded by the Script Control Program. Using the CSV (Comma Separated Variable” format, the Script Control Program reads the script program hierarchy, titles, file names, and configuration settings. The following figure shows the contents of the demo.csv file.

	A	B	C	D	E	F	G
1	Level	Description	Path	Default	Setting 1	Setting 2	Setting 3
2	1	Script Program A	demo_A.scx	1	1	0	0
3	2	Script Program A2	demo_A2.scx	1	1	0	0
4	1	Script Program B	demo_B.scx	1	0	1	1
5	1	Script Program C	demo_C.scx	1	0	0	1
6	EOF						

Using this file as a template, it can be modified to add or remove script programs, change the displayed hierarchy, or change the pre-defined configurations.

The file must contain at least four columns of data. The first column specifies the hierarchical level when displaying the script programs. This is a number that must be either 1 or greater. As the number is increased, that script program is indented in the display. The second column is the script program description as displayed in the list. The third column contains the file name of the script program. Finally, the fourth and subsequent columns hold the pre-defined configuration settings. The first row gives the



configuration name as displayed in the drop down list, while the remaining rows must be either a one or zero. a one indicates the script program is checked, while zero means unchecked. The last row of the file must contain "EOF" in the first column. This marks the end of the file when being read by the Script Control Program.

Note: The Script Control Program must be installed in the same directory as the CID1500 software. This is normally "C:\AI\CID1500".

4) Controlling the CID1500 From Other Applications

The CID1500 software now contains a mechanism by which other applications can control its operation. This finds use in custom test systems where the CID1500 is integrated into a larger system and is required to respond to higher level commands. Though only four commands are available, they allow other application programs to load CID1500 configuration and script files along with executing script commands.

Control of the CID1500 program is accomplished by using text files for passing information to from the CID1500. Though a relatively slow method of passing information, using files has the advantage of being operating system and network independent. As long as a program can write and read a text file it can control the CID1500. This includes DOS programs, Windows 95/98 programs (16 or 32 bit), Windows NT/2000, or even non-Windows operating systems programs.

The steps involved in passing and receiving information to and from the CID1500 is as follows:

- 1) Controlling program writes command information to a file called "ctrlin.txt" located in the CID1500's directory.
- 2) CID1500 periodically monitors for a non-zero length file called "ctrlin.txt". If found, it reads the file and then erases it. After erasing the file, it interprets the command and executes it. If the command is valid, then the CID1500 writes a file called "ctrlout.txt" with the contents of "OK". Otherwise the CID1500 writes "ERR=xxx", where xxx is a variable length error message.

The "ctrlin.txt" file must contain the command word on the first line. The second and subsequent lines are used to pass additional information. The details of the commands are as follows:

1) Load Script File

This command causes the CID1500 to load the specified script program file. Optionally if the command is appended with ":RUN", then once the script program has been loaded, it is executed.

File Line #	File Contents
1	LOADSCRIPTFILE[:RUN]
2	File name and path of script program to load

2) Load Script Program



This command loads a script program into the CID1500. The script program starts at line 2 of the file. Optionally if the command is followed by “:RUN”, then once the script program is loaded, it is executed.

File Line #	File Contents
1	LOADSCRIPTPROGRAM[:RUN]
2	First line of the script program
3	Second line of the script program
...	
n	Last line of the script program

3) Load Configuration File

Similar to the load script file command, this command loads a configuration file into the CID1500. The file path and name of the configuration file must be located in the second line of the “ctrlin.txt” file. Optionally if the command is followed by “:RUN”, then once the configuration file is loaded it executes any script programs contained within it.

File Line #	File Contents
1	LOADCONFIGFILE[:RUN]
2	File name and path of configuration file to load

4) Read Script Program Execution Status

This command returns whether or not the CID1500 is currently executing a script program. It returns in the “ctrlout.txt” file a single line containing “RUNNING” if a script program is executing or “IDLE” if it has stopped.

File Line #	File Contents
1	STATUS:SCRIPT