

## Switch and Subscriber Emulation



## Supports Overlap Digit Sending



## 1 to 4 Signaling Links



## LAPD and Q.931 Layers



## NFAS Supported



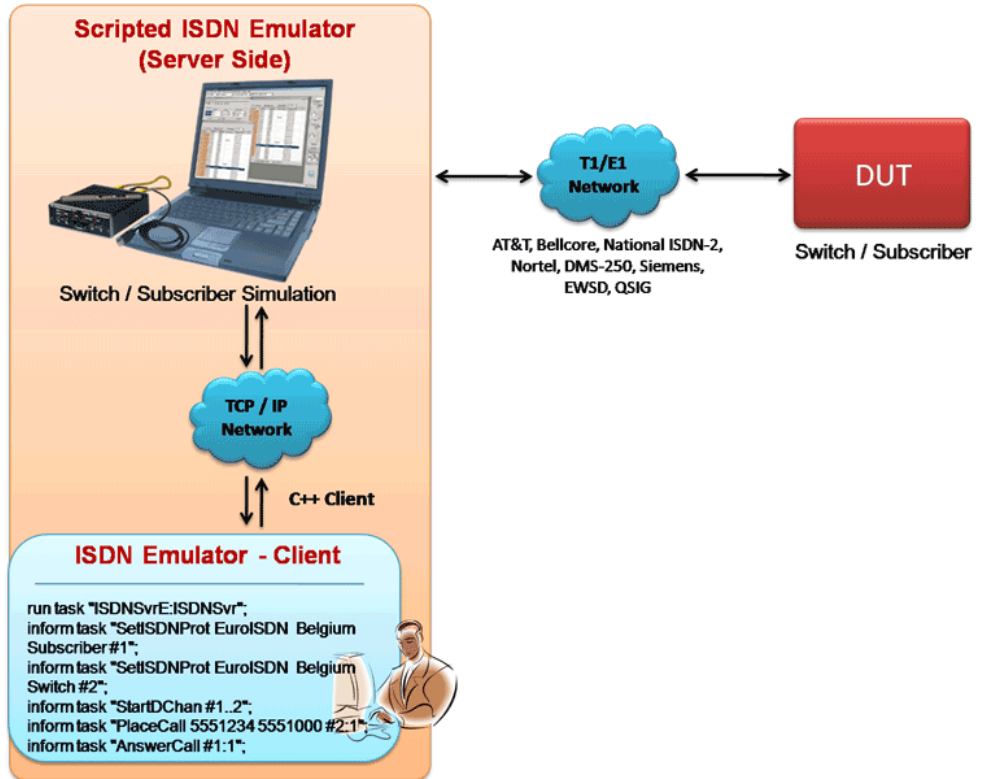
## Standards/ Variants –ISDN AT&T, Nortel, DMS-250, QSIG, and more...



## Call Records For Complete & Incomplete Calls



# Command Line based ISDN Emulator



GL's flexible and versatile ISDN Emulator is available as GUI based and scripted applications with T1/E1 Analyzer, through which the various ISDN configurations can easily be created. The scripted ISDN Emulator is similar to GUI ISDN Emulator functionality, which emulates ISDN calls over T1/E1 links. It also allows configuring the ISDN layer parameters, called/calling numbering plan/type, ISDN service type, place / accept call for each timeslot or for the whole trunk, switch and subscriber emulation, simple NFAS setup for T1, and performs various other tasks on remote clients. Additionally, the Windows Client-Server based ISDN Emulator application offers following advantages –

- Simultaneous testing of high capacity T1/E1 systems through a single client
- Remotely control ISDN emulation using simple commands by client applications
- Collection of call records from remote locations

### Main Features

- Switch and Subscriber emulation
- Performs simple NFAS setup for T1
- Call records for complete or incomplete calls
- Place call or accept call for each timeslot or for the whole trunk

### Specifications

#### ISDN Standards Compliance

**USA ISDN** - AT&T, Bellcore, National ISDN-2, Nortel, DMS-250, Siemens EWSD

**Euro ISDN** - Belgium, China, Europe, France, Britain, Germany, Sweden

**ASIA ISDN** - Australia, Hong Kong, Japan, Singapore & QSIG

#### ISDN Signaling

Available Protocol Layers - LAPD, Q.931

Maximum Links - 1 to 4

For more details, visit our web page <http://www.gl.com/wcsisdnemulator.html>.



# GL Communications Inc.

818 West Diamond Avenue - Third Floor. Gaithersburg, MD 20878 • (V) 301-670-4784 (F) 301-670-9187

Web Page Address: <http://www.gl.com/> • E-Mail Address: [gl-info@gl.com](mailto:gl-info@gl.com)

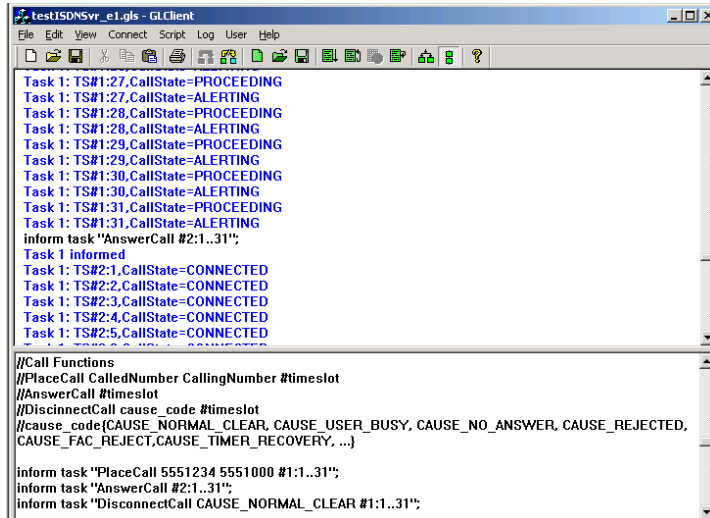
## Scripted ISDN Emulation

The client-side application initializes ISDN Emulator on server side, and configures the ISDN layer parameters such as called / calling numbering plan/type, type of ISDN service, switch and subscriber emulation, and place call or accept call for each timeslot or for the whole trunk. The sample script below demonstrates a simple ISDN place-answer call scenario.

*Sample script for Call Function*

```
run task "ISDNsvrT:ISDNsvr";
inform task 1 "SetISDNProt USA AT&T#4ESS Subscriber #1";
inform task 1 "SetISDNProt USA AT&T#4ESS Switch #2";
inform task 1 "GetISDNProt #1..2";
inform task 1 "StartDChan #1..2";
inform task * "PlaceCall 5551234 5551000 #1:1..31";
inform task * "AnswerCall #2:1..31";
inform task * "DisconnectCall CAUSE_NORMAL_CLEAR #1:1..31";
```

The above script places a call by specifying the called and calling number on all timeslots of card #1. The next task answers the call on all timeslots of card #2. With the third task the successfully established call is disconnected for the specified cause code (Normal\_Clear).



```
testISDNsvr_e1.gls - GLClient
File Edit View Connect Script Log User Help
Task 1: TS#1:27,CallState=PROCEEDING
Task 1: TS#1:27,CallState=ALERTING
Task 1: TS#1:28,CallState=PROCEEDING
Task 1: TS#1:28,CallState=ALERTING
Task 1: TS#1:29,CallState=PROCEEDING
Task 1: TS#1:29,CallState=ALERTING
Task 1: TS#1:30,CallState=PROCEEDING
Task 1: TS#1:30,CallState=ALERTING
Task 1: TS#1:31,CallState=PROCEEDING
Task 1: TS#1:31,CallState=ALERTING
inform task "AnswerCall #2:1..31";
Task 1 informed
Task 1: TS#2:1,CallState=CONNECTED
Task 1: TS#2:2,CallState=CONNECTED
Task 1: TS#2:3,CallState=CONNECTED
Task 1: TS#2:4,CallState=CONNECTED
Task 1: TS#2:5,CallState=CONNECTED
//Call Functions
//PlaceCall CalledNumber CallingNumber #timeslot
//AnswerCall #timeslot
//DisconnectCall cause_code #timeslot
//cause_code(CAUSE_NORMAL_CLEAR, CAUSE_USER_BUSY, CAUSE_NO_ANSWER, CAUSE_REJECTED, CAUSE_FAC_REJECT, CAUSE_TIMER_RECOVERY, ...)
inform task "PlaceCall 5551234 5551000 #1:1..31";
inform task "AnswerCall #2:1..31";
inform task "DisconnectCall CAUSE_NORMAL_CLEAR #1:1..31";
```

Figure: ISDN Call Functions

## Buyers Guide

[XX629](#) – ISDN Emulator (Server/Client Command Reference)

### Related Software

[XX600](#) - Basic Windows Client/Server Scripted Control Software

[XX100](#) - ISDN Analysis Software (T1 or E1)

[XX105](#) - ISDN Emulator – GUI based (T1 or E1)

[XX030](#) - Call Capture Analysis Software and its accessories

[XX090](#) - HDLC Capture and Playback Software (T1 or E1)

[XX130](#) - Frame-Relay Analysis Software (T1 or E1)

[XX629](#) – Server Client ISDN Emulator (T1 or E1)

### Related Hardware

[UTE001](#) - USB based Dual T1 or E1 Laptop Analyzer

[UTA001/UEA001](#) – Basic USB based Dual T1 or E1 Laptop Analyzer Software

[HTE001](#) - Universal HD T1 or E1 PCI Cards

[HUT001/HUE001](#) – Basic Universal HD T1/E1 Software

## NFAS Grouping (Non-Facility Associated Signaling)

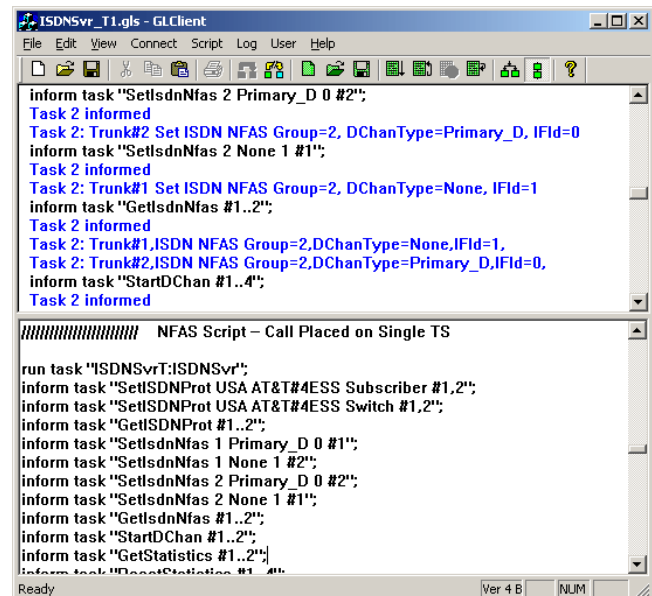
The client-side application also allows users can perform simple NFAS setup for T1. NFAS is a standard option available for ISDN call processing system. NFAS Group-configuration allows number of trunks to be classified into groups, with each group having a unique and identifiable D-Channel. The sample script below demonstrates an ISDN place-answer call with NFAS setup.

*Call Placed on Single TS (T1)*

```
run task "ISDNsvrT:ISDNsvr";
inform task 2 "SetISDNProt USA AT&T#4ESS Subscriber #1,2";
inform task 2 "SetISDNProt USA AT&T#4ESS Switch #3,4";
inform task 2 "GetISDNProt #1..4";
inform task 2 "SetIsdnNfas 1 Primary_D 0 #1";
inform task 2 "SetIsdnNfas 1 None 1 #2";
inform task 2 "SetIsdnNfas 2 Primary_D 0 #3";
inform task 2 "SetIsdnNfas 2 None 1 #4";
inform task 2 "GetIsdnNfas #1..4";
inform task 2 "StartDChan #1..4";
inform task 2 "GetStatistics #1..4";
inform task 2 "ResetStatistics #1..4";
inform task 2 "PlaceCall 5551234 5551000 #2:23";
inform task 2 "AnswerCall #4:23";
inform task 2 "SetAutoReject On CAUSE_REJECTED #4";
inform task 2 "DisconnectCall CAUSE_NORMAL_CLEAR #2:23";
inform task 2 "StopDChan #1..4";
end task*;
```

Above script groups Trunk 1-Trunk 2 (Group 1), and Trunk 3-Trunk 4 (Group 2) with one trunk in each NFAS group set as 'Primary D-Channel'. Here Trunk 1 is the Primary D-Channel in NFAS group 1 and Trunk 3 is the Primary D-Channel in NFAS group 2.

The script places the call on TS 23 of Trunk 2 and answered on TS 23 of Trunk 4. So, the signaling channel on Trunk 2 and 4 is free for Traffic.



```
ISDNsvr_T1.gls - GLClient
File Edit View Connect Script Log User Help
inform task "SetIsdnNfas 2 Primary_D 0 #2";
Task 2 informed
Task 2: Trunk#2 Set ISDN NFAS Group=2, DChanType=Primary_D, IFId=0
inform task "SetIsdnNfas 2 None 1 #1";
Task 2 informed
Task 2: Trunk#1 Set ISDN NFAS Group=2, DChanType=None, IFId=1
inform task "GetIsdnNfas #1..2";
Task 2 informed
Task 2: Trunk#1,ISDN NFAS Group=2,DChanType=None,IFId=1,
Task 2: Trunk#2,ISDN NFAS Group=2,DChanType=Primary_D,IFId=0,
inform task "StartDChan #1..4";
Task 2 informed
NFAS Script – Call Placed on Single TS
run task "ISDNsvrT:ISDNsvr";
inform task "SetISDNProt USA AT&T#4ESS Subscriber #1,2";
inform task "SetISDNProt USA AT&T#4ESS Switch #1,2";
inform task "GetISDNProt #1..2";
inform task "SetIsdnNfas 1 Primary_D 0 #1";
inform task "SetIsdnNfas 1 None 1 #2";
inform task "SetIsdnNfas 2 Primary_D 0 #2";
inform task "SetIsdnNfas 2 None 1 #1";
inform task "GetIsdnNfas #1..2";
inform task "StartDChan #1..2";
inform task "GetStatistics #1..2";
inform task "ResetStatistics #1..2";
```

Figure: NFAS Grouping