Using DataKinetics SS7 Products

Copyright Dialogic Corporation 1998, All Rights Reserved.

Table of Contents

TABLE OF CONTENTS 2
1. OBJECTIVE: 4
2. SS7 BASIC TERMINOLOGY AND ARCHITECTURE 4
2.1. Signaling Links 4
2.2. Signaling Points 4
3. THE SS7 PROTOCOL STACK 5
3.1. Message Transfer Part 6
3.2. ISDN User Part (ISUP) 7
3.3. Telephone User Part (TUP) 7
3.4. Signaling Connection Control Part (SCCP) 7
3.5. Transaction Capabilities Applications Part (TCAP) 8
4. THE DATAKINETICS PRODUCT LINE 8
4.1. Specific Product Details 8
4.1.1. Signaling Interface Unit (SIU) 8
4.1.2. PCCS6 9
4.1.3. SS7 Software 10
5. DATAKINETICS SOFTWARE ARCHITECTURE 10
5.1. Introduction 10
5.2. Basic Concepts 11
5.2.1. Modules 11
5.2.2. Module Identifiers 11
5.2.3. Messages 12
5.2.4. Message Queues 12
6. SOFTWARE DELIVERABLES FROM DATAKINETICS 13
7. HIGH LEVEL ARCHITECTURE OF SIU<->PCCS6 SYSTEM SETUP 13
8. CONFIGURING THE PCCS6 SYSTEM 14
8.1. Installing the PCCS6 Development Package for Windows NT 15
8.2. Installing the User Part Development Package 16
8.3. Installing the System7 Binary Disk 16
8.4. Installing the Device Driver 17
8.5. Activating the Device Driver 17
8.6. DataKinetics PCCS6 Environment Executable Programs 18
8.6.1. S7_MGT.EXE 18
8.6.2. SSD.EXE 18
8.6.3. TIM_NT 18
8.6.4. TICK_NT 18
8.7. Activating the SS7 Link 18
8.7.1. GCTLOAD.EXE 18
9. CONFIGURING THE SIU & SIU HOST SYSTEMS 19
9.1. Installing the OS Specific Host SW 19
9.2. Installing the User Part Development Package 20
9.3. DataKinetics SIU & SIU Host Environment Executable Programs 20
9.3.1. RSI.EXE 20
9.3.2. RSI_LNK.EXE 20

Copyright Dialogic Corporation 1998, All Rights Reserved.

The Name Behind the Voice

9.3.3. RSICMD.EXE 20
9.3.4. SIUCMD.EXE 21
9.3.5. S7_LOG.EXE 21
9.3.6. S7_PLAY.EXE 21
9.4. Activating the SS7 Link 21
9.4.1. GCTLOAD.EXE 21
10. CONFIGURATION FILES 21
10.1. system.txt 21
10.1.1. LOCAL 22
10.1.2. REDIRECT 22
10.1.3. FORK_PROCESS 22
10.2. config.txt 22
10.2.1. PCCS6 Configurations: 22
10.2.2. SIU Configurations: 23
11. SIU/PCCS6 BACK TO BACK CONFIGURATION 25
12. THINGS TO CHECK WHEN THERE ARE PROBLEMS 26
13. RUNNING A SAMPLE APPLICATION LIKE SS7PONG.EXE 27
APPENDIX A 28
14. PCCS6 FILES 28
14.1. config.txt 28
14.2. system.txt 30
14.3. run.bat 31
APPENDIX B 32
15. SIU FILES 32
15.1. config.txt 32
15.2. system.txt 35
APPENDIX C 38
16. SIU HOST FILES 38
16.1. system.txt 38
16.2. run.bat 39
REFERENCES 40

Copyright Dialogic Corporation 1998, All Rights Reserved.

1. Objective:

This application note focuses on setting up a DataKinetics SIU & Host back to back with a PCCS6 system in order to create a test harness for an SS7 network. This app note is a supplemental to the DataKinetics SW manuals and will attempt to point out any configuration difficulties that one may experience in setting up this configuration. This test harness, once established, should be a good platform to reproduce and help resolve any SS7 customer issues. This will also provide an excellent development platform in order to do SS7 application development.

Prior to discussing the DataKinetics SS7 products, it may be a good idea to review some basic terminology and concepts about SS7 architecture and the SS7 protocol stack. If you are fairly comfortable with these concepts and wish to move ahead to the DataKinetics product specifics, go directly to ***Section 4.***

2. SS7 Basic Terminology and Architecture

2.1. Signaling Links

SS7 messages are exchanged between network elements over 56 or 64 kilobit per second (Kbps) bi-directional channels called **signaling links**. Signaling occurs **out-of-band** on dedicated channels rather than **in-band** on voice channels. Compared to in-band signaling, out-of-band signaling provides:

- faster call setup times (compared to in-band signaling using multi-frequency (MF) signaling tones)
- more efficient use of voice circuits
- support for **Intelligent Network** (IN) services which require signaling to network elements without voice trunks (e.g., database systems)
- improved control over fraudulent network usage

2.2. Signaling Points

Each signaling point in the SS7 network is uniquely identified by a numeric **point code**. Point codes are carried in signaling messages exchanged between signaling points to identify the source and destination of each message. Each signaling point uses a routing table to select the appropriate signaling path for each message.

There are three kinds of signaling points in the SS7 network (Fig. 1):

- **SSP** (Service Switching Point)

Copyright Dialogic Corporation 1998, All Rights Reserved.

- **STP** (Signal Transfer Point)
- **SCP** (Service Control Point)

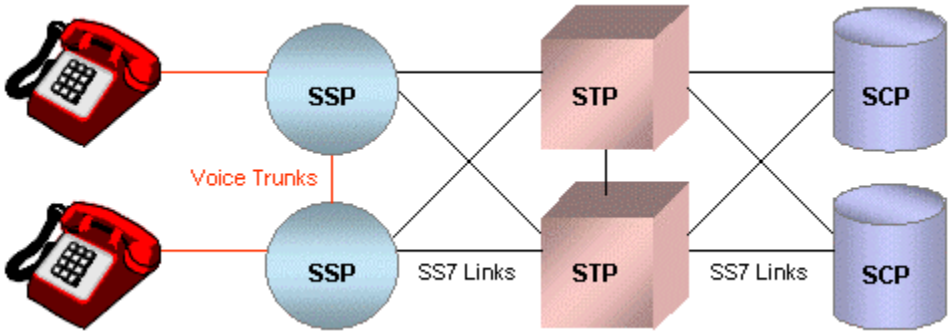

Figure 1. SS7 Signaling Points

SSPs are switches that originate, terminate, or tandem calls. An SSP sends signaling messages to other SSPs to setup, manage, and release voice circuits required to complete a call. An SSP may also send a query message to a centralized database (an **SCP**) to determine how to route a call (e.g., a toll-free 1-800/888 call in North America). An SCP sends a response to the originating SSP containing the routing number(s) associated with the dialed number. An alternate routing number may be used by the SSP if the primary number is busy or the call is unanswered within a specified time. Actual call features vary from network to network and from service to service.

Network traffic between signaling points may be routed via a packet switch called an **STP**. An STP routes each incoming message to an outgoing signaling link based on routing information contained in the SS7 message. Because it acts as a network hub, an STP provides improved utilization of the SS7 network by eliminating the need for direct links between signaling points. An STP may perform **global title translation**, a procedure by which the destination signaling point is determined from digits present in the signaling message (e.g., the dialed 800 number, calling card number, or mobile subscriber identification number). An STP can also act as a "firewall" to screen SS7 messages exchanged with other networks.

Because the SS7 network is critical to call processing, SCPs and STPs are usually deployed in mated pair configurations in separate physical locations to ensure network-wide service in the event of an isolated failure. Links between signaling points are also provisioned in pairs. Traffic is shared across all links in the **linkset**. If one of the links fails, the signaling traffic is rerouted over another link in the linkset. The SS7 protocol provides both error correction and retransmission capabilities to allow continued service in the event of signaling point or link failures.

3. The SS7 Protocol Stack

Copyright Dialogic Corporation 1998, All Rights Reserved.

The Name Behind the Voice

The hardware and software functions of the SS7 protocol are divided into functional abstractions called "levels". These levels map loosely to the **Open Systems Interconnect** (OSI) 7-layer model defined by the International Standards Organization (ISO).

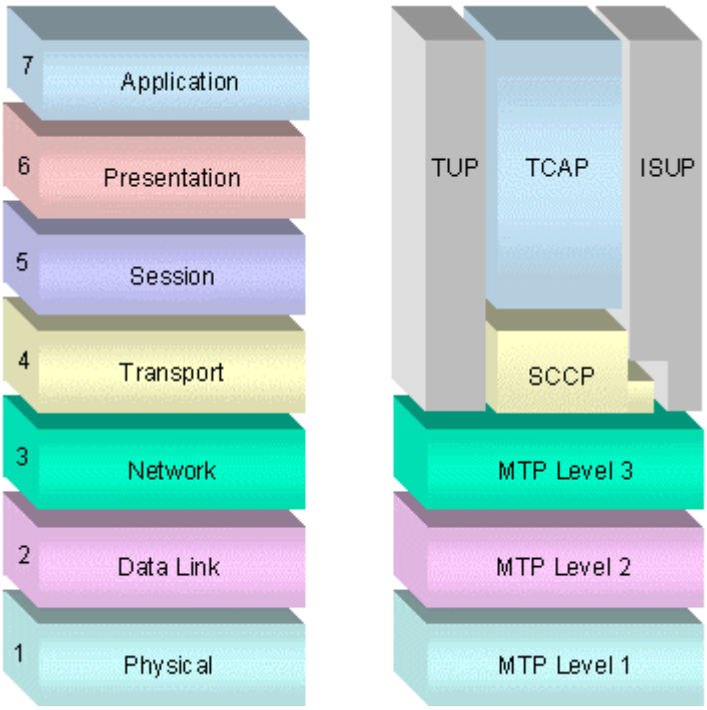

Figure 2. The OSI Reference Model and the SS7 Protocol Stack

3.1. Message Transfer Part

The Message Transfer Part (MTP) is divided into three levels. The lowest level, **MTP Level 1**, is equivalent to the OSI Physical Layer. MTP Level 1 defines the physical, electrical, and functional characteristics of the digital signaling link. Physical interfaces defined include **E-1** (2048 kb/s; 32 64 kb/s channels), **DS-1** (1544 kb/s; 24 64kb/s channels), **V.35** (64 kb/s), **DS-0** (64 kb/s), and **DS-0A** (56 kb/s).

MTP Level 2 ensures accurate end-to-end transmission of a message across a signaling link. Level 2 implements flow control, message sequence validation, and error checking. When an error occurs on a signaling link, the message (or set of messages) is retransmitted. MTP Level 2 is equivalent to the OSI Data Link Layer.

In some configurations the number of signaling links may exceed the capacity of a single PCCS6 board, or MTP2 entities may need to be distributed across multiple boards for sake of fault tolerance. In these

Copyright Dialogic Corporation 1998, All Rights Reserved.

The Name Behind the Voice

configurations, the MTP3 entity may need to reside up on the Local Host or Local Server Module capable of Inter-Module communication.

MTP Level 3 provides message routing between signaling points in the SS7 network. MTP Level 3 re-routes traffic away from failed links and signaling points and controls traffic when congestion occurs. MTP Level 3 is equivalent to the OSI Network Layer.

MTP 3 provides three key functions: routing, message discrimination and distribution. MTP 3 is the one of most critical entities in the SS7 protocol architecture and needs to maintain knowledge of the entire SS7 network topology to ensure end-to-end transmission. Without this level of visibility, MTP 3 could not offer complete route management services, signaling link management, load balancing and congestion management.

Each MTP 3 entity is assigned a Point Code address for message routing. A single MTP 3 entity is typically serviced by sets of subtending Message Transfer Part 2 (MTP 2) data link protocol entities organized as Signaling linksets. Signaling Links are organized as linksets to offer seamless switching of message load in the event of link failures. Load balancing across Signaling linksets can only be performed within the same point code. For this reason, many applications may require MTP 3 to be Host resident.

3.2. ISDN User Part (ISUP)

The ISDN User Part (ISUP) defines the protocol used to set-up, manage, and release trunk circuits that carry voice and data between terminating line exchanges (e.g., between a calling party and a called party). ISUP is used for both ISDN and non-ISDN calls. However, calls that originate and terminate at the same switch do not use ISUP signaling.

3.3. Telephone User Part (TUP)

In some parts of the world (e.g., China, Brazil), the **Telephone User Part** (TUP) is used to support basic call setup and tear-down. TUP handles analog circuits only. In many countries, ISUP has replaced TUP for call management.

3.4. Signaling Connection Control Part (SCCP)

SCCP provides connectionless and connection-oriented network services and **global title translation** (GTT) capabilities above MTP Level 3. A **global title** is an address (e.g., a dialed 800 number, calling card number, or mobile subscriber identification number) which is translated by SCCP into a destination point code and **subsystem number**. A subsystem number uniquely identifies an application at the destination signaling point. SCCP is used as the transport layer for TCAP-based services. Without SCCP, the smallest addressable entity would be a point code. With SCCP, a subsystem (i.e. a user application) within a point code can be addressed.

Copyright Dialogic Corporation 1998, All Rights Reserved.

The Name Behind the Voice

3.5. Transaction Capabilities Applications Part (TCAP)

TCAP supports the exchange of non-circuit related data between applications across the SS7 network using the SCCP connectionless service. Queries and responses sent between SSPs and SCPs are carried in TCAP messages. For example, an SSP sends a TCAP query to determine the routing number associated with a dialed 800/888 number and to check the personal identification number (PIN) of a calling card user. In mobile networks (**IS-41** and **GSM**), TCAP carries **Mobile Application Part** (MAP) messages sent between mobile switches and databases to support user authentication, equipment identification, and roaming.

4. The DataKinetics Product Line

Dialogic has had a relationship with DataKinetics since 1994. Many Dialogic customers have integrated DataKinetics SS7 products into their CT applications. Some of these solutions are currently deployed in the public network.

Dialogic is reselling the following DataKinetics Products:

- PCCS6 is an ISA-based SS7 board solution.
- SIU is an SS7-to-TCP/IP server solution.
- SS7 software stacks for TUP, ISUP, and TCAP.

All of these products use the same architecture and SS7 API.

4.1. Specific Product Details

4.1.1. Signaling Interface Unit (SIU)

- An integrated, box-level solution that converts the SS7 protocols to TCP/IP messages
- Automatically distributes SS7 messages to the appropriate voice response units (VRUs) based on circuit identification code (CIC)
- SIUs can be deployed in a load-sharing configuration to increase reliability
- Single or dual E1/T1/V.35 interfaces per PCCS6 card present in SIU
- SS7 signaling can be extracted from an incoming E-1 or T-1 PCM port into the SIU and if there are voice circuits also on that signaling link, they are passed out of the 2nd E-1 or T-1 PCM port on the board. Alternatively, the signaling can be connected using V.35 serial links or on a PCM port with no voice circuits. Signaling information is automatically distributed by the SIU via a TCP/IP LAN to the host application platform. A portable interface library is provided which can be used on most UNIX and Windows NT® platforms. Two units can be configured to share the same point code, providing fully resilient operation within a single point code. In normal

Copyright Dialogic Corporation 1998, All Rights Reserved.

operation, signaling can be load-shared across the two units. If one unit fails, the remaining unit handles all signaling.

- AC/DC power supplies
- DOS, Windows NT, Solaris and SCO operating systems

4.1.1.1. DSC131

- DS0s (CIC codes) controllable: DSC/131 – 4096
- Signaling Links: DSC/131 – up to 6 signaling links (2 PCCS6 cards max.)
- Maximum # of Link Sets: 4
- Maximum call rate: DSC/131 – 25 calls/second
- Maximum host connectivity: 16 Hosts.
- Maximum # of circuit groups configurable: 128

4.1.1.2. DSC231

- DS0s (CIC codes) controllable: DSC/231 – 16384
- Signaling Links: DSC/231 – up to 32 signaling links (12 PCCS6 cards max)
- Maximum # of Link Sets: 4 (Max of 16 links/link set)
- Maximum call rate: DSC/231 – 80 calls/second;
- Maximum host connectivity: 32 Hosts.
- Maximum # of circuit groups configurable: 648

4.1.2. PCCS6

- An ISA card
- One or two network interfaces (T1, E1, or V.35)
- Up to 3 SS7 signaling links
- SCbus connector
- TCAP software running on the PC-CS6 card
- DOS, Windows NT, and SCO operating systems
- DS0s (CIC codes) controllable: 64 or 256 depending on ISUP module downloaded.
- Maximum # of circuit groups configurable: 128 (when ISUP running on host), 16 (when 256 CIC's), 4 (when 64 CIC's).
- Maximum call rate: ??? calls/second *(Very much host processor dependent!)*

Copyright Dialogic Corporation 1998, All Rights Reserved.

4.1.3. SS7 Software

- TUP - Telephony User Part
  - BT NUP
  - China TUP
  - French TUP (coming soon)
- ISUP - Integrated Services User Part
  - ITU
  - ANSI
- TCAP - Transactions Capabilities Application Part
  - ITU
  - ANSI
- Most of the SS7 software protocols are customizable.
- New SS7 software protocols are being added all the time.

5. DataKinetics Software Architecture

The DataKinetics software architecture is very well described in the "System7 Software Environment Programmer's Manual". Excerpts of the manual are below. Please see the manual for complete details.

5.1. Introduction

The DataKinetics System7 product range is comprised of a number of portable software modules for the realization of Signaling System Number 7 (SS7) protocol stacks. The System7 architecture is multi-tasking, using message passing to communicate between tasks.

Each module in the system is implemented as a separate task within the chosen operating environment. A module implements either a layer within the protocol stack, a user part or some other functional entity within the system. In general, a module supports multiple instances within a single process (for example multiple links, multiple circuits or multiple transactions are each handled by a single process).

For software portability, each module makes a minimum demand on the host operating system. Most modules require just 5 functions to be provided by the operating system, these are required for inter-process communication and memory allocation. This approach makes the software easy to port to different platforms and operating environments. In addition, the use of message passing between

Copyright Dialogic Corporation 1998, All Rights Reserved.

The Name Behind the Voice

tasks, and therefore protocol layers, means that it is possible for different layers to run on different processors if required.

All that is required to port the System7 product to a new environment is an implementation of the 5 library functions tailored to the chosen operating system. These functions are available as standard products for a number of popular operating systems.
NOTE: There are additional library calls available in the in the event that dual SIU's are being used (i.e. GCT_set_instance() & GCT_get_instance()).

5.2. Basic Concepts

This section introduces the basic System7 concepts and terminology used throughout the DataKinetics documentation suite.

5.2.1. Modules

A module is an implementation of a particular layer in the protocol stack (eg. MTP2, MTP3), a particular user part (eg. ISUP, SCCP) or a collection of other functionality which fits together as a logical entity. A module may be part of the System7 product range or a User-supplied module.

Each module in the system runs as a separate task, process or program (depending on the type of operating system). The module is identified by a **Module Identifier** and communicates with other modules in the system by sending **Messages** to a **Message Queue** belonging to the destination module. A set of **Library Functions** is the only operating system specific code used by a module.

A module handles multiple instances of the functional entity associated with the module (eg. MTP2 module handles multiple signaling links, the MTP3 module handles multiple link sets and multiple routes, the ISUP module handles multiple circuits).

5.2.2. Module Identifiers

Each module has a module identifier (module_id) which is a logical number in the range 0 to 255. It is used to identify modules within the system for the purposes of inter-process communication. To send a message to another process the sending module uses the module identifier of the destination process. To receive a message from a modules own message queue it uses it' s own module identifier.

Some modules operate with a fixed module identifier while others allow the module identifier to be specified at run-time. The module identifiers of other modules with which a module will communicate is usually a run-time configuration option. The module identifier is a logical value, it is not the same as the task_id or process id (pid) which is usually allocated automatically by the operating system when a process is created.

The inter-process communication mechanism usually uses the module identifier as an index to an array of message queue pointers to provide an efficient mechanism to quickly access

Copyright Dialogic Corporation 1998, All Rights Reserved.

the correct message queue. This feature can also be used to allow messages destined for a particular module to be re-directed to an alternative message queue belonging to another module.

The re direction mechanism is used when messages need to be transferred to another board in the system. Messages for all processes that run on the other board are redirected to a special local module which handles inter-board message passing. Other modules within the system do not need to know whether the modules with which they communicate are running locally or not.

5.2.3. Messages

Modules communicate by sending messages to other modules in the system. There are three types of messages used within the System7 protocol software (MSG, T_FRAME and R_FRAME). (However T_FRAMEs and R_FRAMEs are only used on embedded processors by modules running on the same processor as the physical interface). For this reason this document refers in detail only to the use of the MSG message structure.

The MSG message is a ‘ C’ structure containing a fixed format header field (which is common also with T_FRAMEs and R_FRAMEs) and a buffer for variable length parameter data.

The message header contains a message **type** field, which serves to identify the meaning of the message and the format of the variable parameter area. The **id** field identifies to which instance of the entity handled by the module the message applies (eg the link id or the circuit id etc). The **src** and **dst** fields are the source and destination module identifiers which must be entered by the sending module prior to sending the message. The **rsp_req** field is used by the sending module to solicit a confirmation that the message has been processed by the destination process. When a confirmation is requested, the destination module (after finishing processing the message) enters a value in the **status** field of the message (usually zero to imply success or non-zero otherwise) and sends the message back to the source module.

5.2.4. Message Queues

Each module in the system has a single message queue, which is used by other modules to send messages to the module. A message queue is a system buffer which stores messages (usually by reference) in first-in, first-out order.

Messages are read out of the message queue by the receiving module which typically waits until there is a message available and reads it, it then processes the message and waits until the next message is available. All input to the module is through its message queue.

The fact that the software is modularized in such a way allows applications to be scalable from a single board configuration to a multiple SIU configuration without modifying the application. The changes that are required to move to a higher density solution are configuration related.

Copyright Dialogic Corporation 1998, All Rights Reserved.

6. Software Deliverables from DataKinetics

Software Develiverable	System where SW Resides	Software Description
PCCS6 Development Package	PCCS6 System	Contains the device driver, header files and library functions for use by an application, a number of executables to be run as part of the System7 environment, and a utility program to configure the protocol SW.
User Part Development Package	PCCS6 System & SIU Host	Contains example source code to illustrate the techniques used for interfacing with the System 7 software modules.
OS Specific Host SW	SIU Host	Contains a number of executable programs and libraries or C-source files that are linked with the users' application.
System7 Binary Disks	PCCS6 System	Contains the code file, which is downloaded to the board at run-time by the driver program. Code files for PCCS3 have a file suffix .dc1 while code files for PCCS6 use .dc2.

7. High Level Architecture of SIU<->PCCS6 System Setup

Copyright Dialogic Corporation 1998, All Rights Reserved.

The Name Behind the Voice

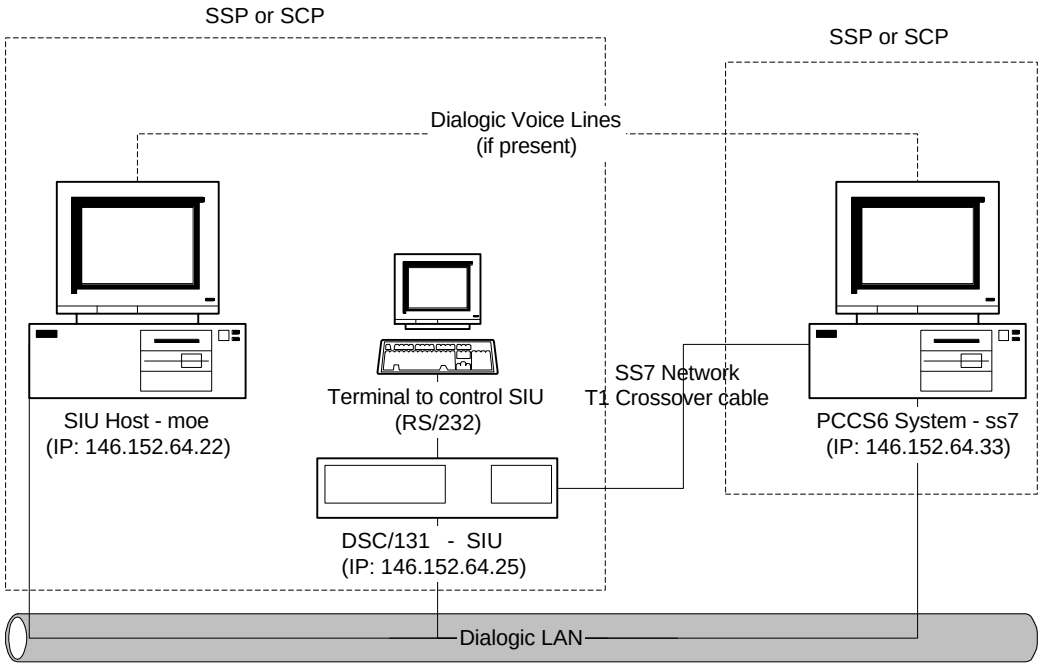

Figure 3. The High Level Architecture of SIU<-> PCCS6 System Setup

8. Configuring the PCCS6 System

Take the following steps to get a DataKinetics PCCS6 configuration working:

1) Set the I/O base address on the board (SW1)
2) Set the IRQ jumpers on the board
3) Make sure that daughtercard is set to either either T1 or E1 position (This is not documented in DataKinetics manuals!)
4) Install the DataKinetics software (See “Installing the PCCS6 Development Package for Windows NT” ,“Installing the User Part Development Package” and “Installing the System7 Binary disks” below).
5) Copy the PCCSXDVR.SYS (the DataKinetics driver) to the \WINNT\system32 directory (see “Installing the Device Driver”).
6) Copy GCTNT.DLL (the DataKinetics DLL) to \WINNT\system32 directory (see “Installing the Device Driver”).

Copyright Dialogic Corporation 1998, All Rights Reserved.

7) Enable the driver (see "Activating the Device Driver" below)
8) Verify that the driver has been enabled (see "Activating the Device Driver" below)
9) Change the Startup Mode of the driver to "Automatic".
10) Modify the configuration files to reflect your configuration (see "Configuration Files" below)
11) Execute the BIN\gctload program to read the system.txt configuration file and "FORK" the appropriate executable files specified in the configuration files.
12) Make sure that the signaling link has been activated (see “Activating the SS7 Link”).

8.1. Installing the PCCS6 Development Package for Windows NT

The PCCS Development Package for Windows NT is distributed on a DOS format disk. The installation procedure essentially involves copying the files onto your development system.

In order to set up the PCCS6 system, first create a new directory to serve as the root directory for the System7 software.

mkdir c:\system7

Then copy the contents of the PCCS Development Package into this directory according to the following sub-directory structure.

Copyright Dialogic Corporation 1998, All Rights Reserved.

The sub-directories and their contents are as follows:

bin	Directory containing executable programs as follows: • Windows NT device driver: **pccsxdvr.sys**, • Dynamic link library: **gctnt.dll**, • Interface w/device driver for message passing to/from board and downloading SW to board: **ssd.exe** • Board installation utility: **pccsxcfg.exe**, • System7 environment executables: **gctload.exe, tim_nt.exe and tick_nt.exe.** • Protocol software configuration utility: **s7_mgt.exe** and example executables: **mtpsl.exe** and **upe.exe**.
run	Directory containing system configuration file **system.txt** and Protocol configuration file **config.txt,** as well as the ***.dc2** code file. This is the directory from which the software should be run.
lib	Directory containing library files ***.lib** for linking with the users application: **gctnt.lib & genlib.lib.**
src\inc	C header files required by an application.
src\example	Example source code to activate or deactivate an SS7 signaling link: **mtpsl.c** and an example of how to interface to MTP3 **upe.c**.

8.2. Installing the User Part Development Package

Copy the contents of the User Part Development Package distribution disk into the **system7\src** directory maintaining the sub-directory structure.

8.3. Installing the System7 Binary Disk

Copy the appropriate Binary code file (*.dc2) that you want downloaded to your PCCS6 card into the \run directory. The title of the file tells which user parts (if any) in addition to the SS7 Message Transfer Part (MTP) (which is always included) are included in the code file. The title of the file refers to the highest SS7 Protocol layer included in the code file including all sub layers. Refer to **Figure 3** to see which layers are considered sub-layers.

For Example:

mtp.dc2: includes only MTP

Copyright Dialogic Corporation 1998, All Rights Reserved.

The Name Behind the Voice

isup.dc2: includes ISUP & MTP
tup.dc2: includes TUP & MTP
tcap.dc2: includes TCAP, SCCP & MTP

8.4. Installing the Device Driver

When installing a PCCS6 board, there is one file (pccsxdvr.sys) that is loaded into the \BIN directory that must be moved into the \WINNT\system32 directory. This is the device driver file. This file must reside in the system32 directory for the DataKinetics software to operate properly. Also, a copy of the \LIB\gctnt.dll file must be copied to the system32 directory.

8.5. Activating the Device Driver

Once the PCCSXDVR.SYS file has been copied to the correct directory, it must be activated. The PCCSXCFG.EXE program must be used to do this. The PCCSXCFG.EXE program adds the device driver to the system. Example instantiation of this command to add the device driver is as follows:

C:\system7\BIN> pccsxcfg -n1 -p0x200 -m0xd0000 add c:\winnt\system32\pccsxdvr.sys

The parameters are as follows:

-n1	number of boards
-p0x200	i/o port (each board uses 4 ports from and including the first)
-m0xd0000	memory address (each board uses the same 4K block)
add c:\winnt\system32\pccsxdvr.sys	"add" the device driver

If the command was successful, the following will be printed to the screen:

"PCCS6 board 0 : memory 0xd0000 ioport 0x200"

If this is not displayed, something did not get installed correctly. In most cases, this is an I/O or memory conflict. Check the NT Diagnostics and verify that the board is being loaded in an unoccupied location (use c:\winnt\system32\winmsd.exe). Also, check Event Viewer for more details on failure! The driver must then be removed and the "add" process re-tried. To remove the driver, execute the program as follows:

```
C:\system7\BIN> pccsxcfg remove
```

Once the device has been activated using this command, it can be verified by using Control Panel -> Devices and locating the device PCCS. This device should say "Manual" and "Started". If so, the device has been started successfully. Now, the devices Startup Mode should be changed to "Automatic" so this device is always stared upon a restart of NT.

Copyright Dialogic Corporation 1998, All Rights Reserved.

NOTE: Activating the driver only has to be done once.

8.6. DataKinetics PCCS6 Environment Executable Programs

Once the installation of the software is complete and the driver is successfully activated, we can begin setting up the SS7 link. There are several executable programs which must be run to get a system working. Here is a list of executables needed in PCCS6 configurations and what they mean:

8.6.1. S7_MGT.EXE

S7_MGT.EXE derives configuration parameters from the **config.txt** text file and sends protocol configuration messages to all the SS7 software modules running. These messages configure the SS7 protocol. This process is optional (but recommended). As an alternative to using it, the user can also perform protocol configuration by sending messages directly to other modules in the system.

8.6.2. SSD.EXE

SSD.EXE is the process, which interfaces with the device driver for passing messages to and from the board and for downloading SW.

8.6.3. TIM_NT

Process to receive periodic tick notification from the tick_nt and handle protocol timers for all other processes. This executable is not used by user applications, but is required by the DataKinetics internal software.

8.6.4. TICK_NT

Protocol timer process to send periodic “tick” notification to the tim_nt process which in turn handles protocol timers. This executable is not used by user applications, but is required by the DataKinetics internal software.

8.7. Activating the SS7 Link

The system environment is created and all protocol modules started by the gctload.exe program.

8.7.1. GCTLOAD.EXE

The GCTLOAD.EXE program is run once all the configuration files are setup correctly. GCTLOAD.EXE reads through the system.txt file, sets up the software environment, then "forks" the executables specified in the system.txt file. Typically this program is run from a batch file located in the run directory. For example run.bat files, refer to **Appendix A for PCCS6 System & Appendix C for SIU Host.**

Once the system environment is initialized and all environmental executable processes are run, it is now time to “Activate the SS7 Link” from. There is a utility called **mtpsl**, which is included for the user to easily activate the link. The format of the mtpsl utility is:

Copyright Dialogic Corporation 1998, All Rights Reserved.

The Name Behind the Voice

mtpsl <ACT|DEACT> <link set> <link within linkset>

eg: mtpsl ACT 0 0

NOTE: **mtpsl** can not be run from within the system.txt file. It must be run from the command line.

The mtpsl.c sample code located in src\example is provided to show users how to send messages to MTP3 to activate and deactivate links from within the user application. It is expected that eventually the user will incorporate this example code into their own application to activate the link at initialization.

9. Configuring the SIU & SIU Host Systems

Take the following steps to get a DataKinetics SIU & SIU Host configuration working:

1) Install DataKinetics software (See "Installing the OS Specific Host SW" and "Installing the User Part Development Package" below)
2) Connect either a dumb terminal to the SIU using the serial (RS232) connector or you can telnet across the network to gain access to the SIU to edit config.txt and system.txt. A variety of terminal emulation programs, which emulate VT100 terminal type, will work.

   NOTE #1: The initial time that you configure the SIU, the IP address will not be set up correctly so telnet will not work. For initial setup, a dumb terminal with a null-modem serial cable is required.

   NOTE #2: The SIU is not configured to allow access to a machine from another subnet. You must telnet from the same subnet as the SIU in order to configure that system. Also, the host must be on the same subnet as the SIU.
3) Modify the etc/hosts file on the SIU to reflect the correct IP address for the SIU.

   NOTE: Make sure that's the same IP address that is entered in the "FORK_PROCESS rsicmd.exe" line in the host system.txt file!
4) Reboot the SIU for the IP address to take affect.
5) "ping" the IP address from the "host" machine on the LAN (must be on same subnet!) to verify that the machine can talk to the SIU. Modify the configuration files to reflect your configuration (see "Configuration Files" below). If the host machine cannot "ping" the SIU, there may be something physically wrong with the LAN setup. Check cables, etc.
6) Execute the BIN\gctload program from the host machine to read the system.txt file on the host and "FORK" the appropriate executable files specified. In an SIU configuration, the SS7 link specific configuration is only resident on the SIU itself since that is where the SS7 cards reside, so the host only has a system.txt file and no config.txt file. For example configuration files, refer to **Appendix B** for SIU files and **Appendix C** for SIU Host files**.**

9.1. Installing the OS Specific Host SW

The Win NT Host SW is distributed on a DOS format disk. The installation procedure essentially involves copying the files onto your development system.

Copyright Dialogic Corporation 1998, All Rights Reserved.

The Name Behind the Voice

In order to set up the SIU Host system, first create a new directory to serve as the root directory for the System7 software.

mkdir c:\system7

Then copy the contents of the Win NT Host SW into this according to the following sub-directory structure.

The sub-directories and their contents are as follows:

bin	Directory containing executable programs as follows: System7 environment executables: **gctload.exe, rsi.exe, rsi_lnk.exe, rsicmd.exe, s7_log.exe, s7_play.exe** and **siucmd.exe.**
run	Directory containing system configuration file **system.txt** and **run.bat** This is the directory from which the software should be run.
lib	Directory containing library file for linking with the user's application: **gctlib.lib.**
src	Directory where the User Part Development package will be installed.

9.2. Installing the User Part Development Package

Copy the contents of the User Part Development Package distribution disk into the **system7\src** directory maintaining the sub-directory structure.

9.3. DataKinetics SIU & SIU Host Environment Executable Programs

9.3.1. RSI.EXE

This program manages the remote socket interface. It manages the routing of messages between the host and SIU(s).

9.3.2. RSI_LNK.EXE

This program is basically a pipe between the application and the SIU. It utilizes the TCP/IP interface and hides the TCP/IP specifics from the application.

9.3.3. RSICMD.EXE

This program establishes the initial TCP/IP communication between the host PC and the SIU. All subsequent messages from the host to the SIU are sent using the RSI.EXE explained below. The TCP/IP address of the SIU is passed in as an argument to this program.

Copyright Dialogic Corporation 1998, All Rights Reserved.

The Name Behind the Voice

9.3.4. SIUCMD.EXE

This program allows the user to issue SIU management commands as text on the host terminal to control an SIU.

9.3.5. S7_LOG.EXE

This program is a task that receives status and management indication messages from the SIU and displays these as text on the application console.

9.3.6. S7_PLAY.EXE

This program reads message contents from an ASCII text file (in a defined format) and sends these messages to the SIU.

9.4. Activating the SS7 Link

The system environment is created and all protocol modules started by the gctload.exe program.

9.4.1. GCTLOAD.EXE

The GCTLOAD.EXE program is run once all the configuration files are setup correctly. GCTLOAD.EXE reads through the system.txt file, sets up the software environment, then "forks" the executables specified in the system.txt file. See "Configuration Files" below for details on the system.txt file. Typically GCTLOAD.EXE is run from a batch file located in the run directory. For an example SIU Host run.bat file, refer to **Appendix C.**

10. Configuration Files

There are 2 configuration files used when setting up the software for a DataKinetics system: **system.txt** and **config.txt**.
NOTE: Example config files for a PCCS6<->SIU configuration are in **Appendices A-C**.

10.1. system.txt

The **system.txt** file specifies how the DataKinetics software environment is set up. This is **NOT** to say the SS7 specific software, but the software that allows user applications to talk to the DataKinetics hardware. The **system.txt** file is located in the \RUN directory on the SIU Host and PCCS6 machines and is present in /home/dklsiu on the SIU. The easiest way to understand the operation of the DataKinetics software is to think of each item as a *module*. Each module can run either in the host PC or on the DataKinetics hardware (PCCS6 card or SIU). This is true of all software modules from the MTP3 level and up. Each module uses inter process communication to send information to other modules.

Copyright Dialogic Corporation 1998, All Rights Reserved.

Each module that runs is assigned an identifier, called the module ID. The **system.txt** file creates those module ID's.

The **system.txt** file contains 3 major sections:

10.1.1. LOCAL

The **LOCAL** section assigns module ID's for the modules that will run in the host machine. The host machine is the computer where the application is running. By placing the modules in this section it also guarantees that the queues for incoming messages (from other processes) will also be created.

10.1.2. REDIRECT

The **REDIRECT** section redirects messages from one module to another. For example: If you REDIRECT messages from your application to SSD.EXE, that means that the messages will be sent to the board. If you REDIRECT messages from your application to RSI.EXE, they will be sent (via TCP/IP) to the SIU.

10.1.3. FORK_PROCESS

The **FORK_PROCESS** section actually executes programs. This is where each of the modules identified in the previous sections are actually run.

10.2. config.txt

The **config.txt** file specifies the SS7 link's connection characteristics. Just like the isxxx.prm files in Dialogic ISDN configurations, the **config.txt** file specifies the details of the SS7 interface. The **config.txt** file contains multiple sections. In a PCCS6 configuration, the config.txt file is located on the host machine in the \RUN directory. In an SIU configuration, this file is located on the SIU itself in /home/dklsiu. This file must be modified by 'telnet'ing into the SIU machine and editing the file. Remember that the SIU has its own operating system (QNX) that uses a 'vi' editor. Depending on whether a PCCS6 card is being configured or an SIU, the **config.txt** file will contain different sections.

10.2.1. PCCS6 Configurations:

Below are the following configuration characteristics that are defined in the config.txt with regards to a PCCS6 configuration.

10.2.1.1. Physical Configuration

The **PCCS6_BOARD** command identifies the characteristics of the card itself. Most notable items in this command include the clock source for the board, how many interfaces are contained on the board and what firmware file to download to the board.

10.2.1.2. MTP Configuration

This section defines the characteristics of the MTP layers. For example, the linkset, local point code and adjacent point code are defined here. In a back to back configuration, it is very important to verify that the Local Point Code on one side of the connection is different than the Local Point Code on the other side.

Copyright Dialogic Corporation 1998, All Rights Reserved.

Also, for a back to back configuration verify that the DPC (Destination Point Code) and Adjacent Point Code on one side matches the Local Point Code on the other side.

NOTE: In the **MTP_ROUTE** command, the user part mask is a 16 bit value with bit n (n=0..15) set to allow the route to be used for messages with the following SI (service indicators). The SI's are defined as follows:

SCCP = 3 (0x8)
TUP = 4 (0x10)
ISUP = 5 (0x20)

Important NOTE #1 for ANSI Operation: The **MTP_CONFIG** line in config.txt specifies how the Layer 2 link should be set up. The 'sub-service field" argument of **MTP_CONFIG**, which is 8 by default, must be set to 0xb for ANSI T1. *This is NOT documented in the manuals.* This argument is a bit field which specifies message priorities, and must be set a certain way for ANSI T1 (ITU E1 does not prioritize message, which is why they are mostly zero). Leaving the sub-service field at the default of 8 has the effect under T1, of giving highest priority to call setup messages, which ANSI dictates should have the lowest priority, and of giving zero priority to *all* other messages.

Important NOTE #2 for ANSI Operation: Also in **MTP_CONFIG,** the "Options" bit field, which is set to all zeroes by default, must be set to 0xf00. Bits 8 & 9 specify ANSI operations, as documented in the manuals. Bits 10 & 11 operate somehow with the sub-service field to define message priorities and must be set for ANSI T1 operations.

10.2.1.3. ISUP Configuration

This section defines the ISUP characteristics. Point codes, CIC codes, etc. The Point Codes in this section must match those in the MTP Configuration section.

Important NOTE #3 for ANSI Operation: Each SS7 user part has its own configuration command in config.txt. I'm only familiar with **ISUP_CONFIG**, however the "Options" field of **ISUP_CONFIG**, which is a bit field, must have bits 8 & 9 set in order to enable ANSI compliant operations. These bits are zero by default.

10.2.2. SIU Configurations:

In an SIU configuration, the **config.txt** file contains all the same information in the PCCS6 configuration plus the following sections:

10.2.2.1. SIU Commands

This defines how many SIU's are in the system. A second SIU is typically configured for redundancy.

10.2.2.2. Cross Connections

This section maps the inbound B channels/tslots on one PCM port to the outbound B channels/tslots on the other PCM port, while the SIU strips off signaling information, which is sent to the host(s) via TCP/IP.

NOTE: When enabling a T1 signaling link, it is important to comment our references for T1 timeslots 25-32 in the **MVIP_XCON** section. You must also reset bits 25-32 of the "CIC Mask" argument of all **ISUP_CFG_CCTGRP** command lines. By default this mask is 0x7fff7fff, it must be changed to 0x00ff7fff.

Copyright Dialogic Corporation 1998, All Rights Reserved.

In a multiple host, single SIU configuration, the following are important points:

1) The DataKinetics software is designed to have a single point of contact for Management and Service types of messages (i.e. any messages received from MTP3 and below). So, when the second host machine runs GCTLOAD to activate the link, all that will be seen is a "Link Up" message and nothing else. This is because the messages that Host 1 sees are Management messages from MTP2 and MTP3. Host 2 will not see these messages.

2) You might at that point ask, "How is Host 2 supposed to know that the link is established and I can begin making calls?" The answer is, the application code is responsible for informing each host of what is happening with the link. The management messages are sent to a single place in your system to be logged and controlled by one machine. That machine then informs all the other hosts (you could have up to 32 hosts with a DSC231 model SIU and 16 hosts with a DSC131 model SIU).

3) There must be one ISUP_CFG_CCTGRP command in the configuration file for every "span" in the entire system. That means that even the spans that do not have a D-Channel and are not directly connected to the SIU must have an entry in this file. If there are multiple hosts, the ISUP_CFG_CCTGRP command instructs the SIU as to where to send the messages for those particular circuits.

4) The base_cic is from the switch's perspective. So, if you are connected to the same switch for both links, the <base_cic> will never be the same number. BUT, if you are connected to multiple switches, the <base_cic> **could** be the same number. The <base_cid> is a logical number used from the SIU's perspective. That means that it will never be the same number and should be incremented for every circuit entry in the configuration file (i.e the <base_cid> = 0 through total_circuits-1).

Copyright Dialogic Corporation 1998, All Rights Reserved.

11. SIU/PCCS6 Back to Back Configuration

The following diagram illustrates a back to back configuration using an SIU on one side and a PCCS6 card on the other. Configuration files are given to show exactly how to configure this setup in **Appendices A-C**. These diagrams illustrate software and hardware modules in this configuration. Hardware modules are shown with black shadowing, software modules with gray.

The following diagram shows an example of the software environment. Notice that the modules in the picture map not only to the module ID's in the **system.txt** files located in **Appendix A** (for PCCS6 system) **and Appendix C** (for SIU Host system), but also in the FORK_PROCESS section where these process are actually executed.

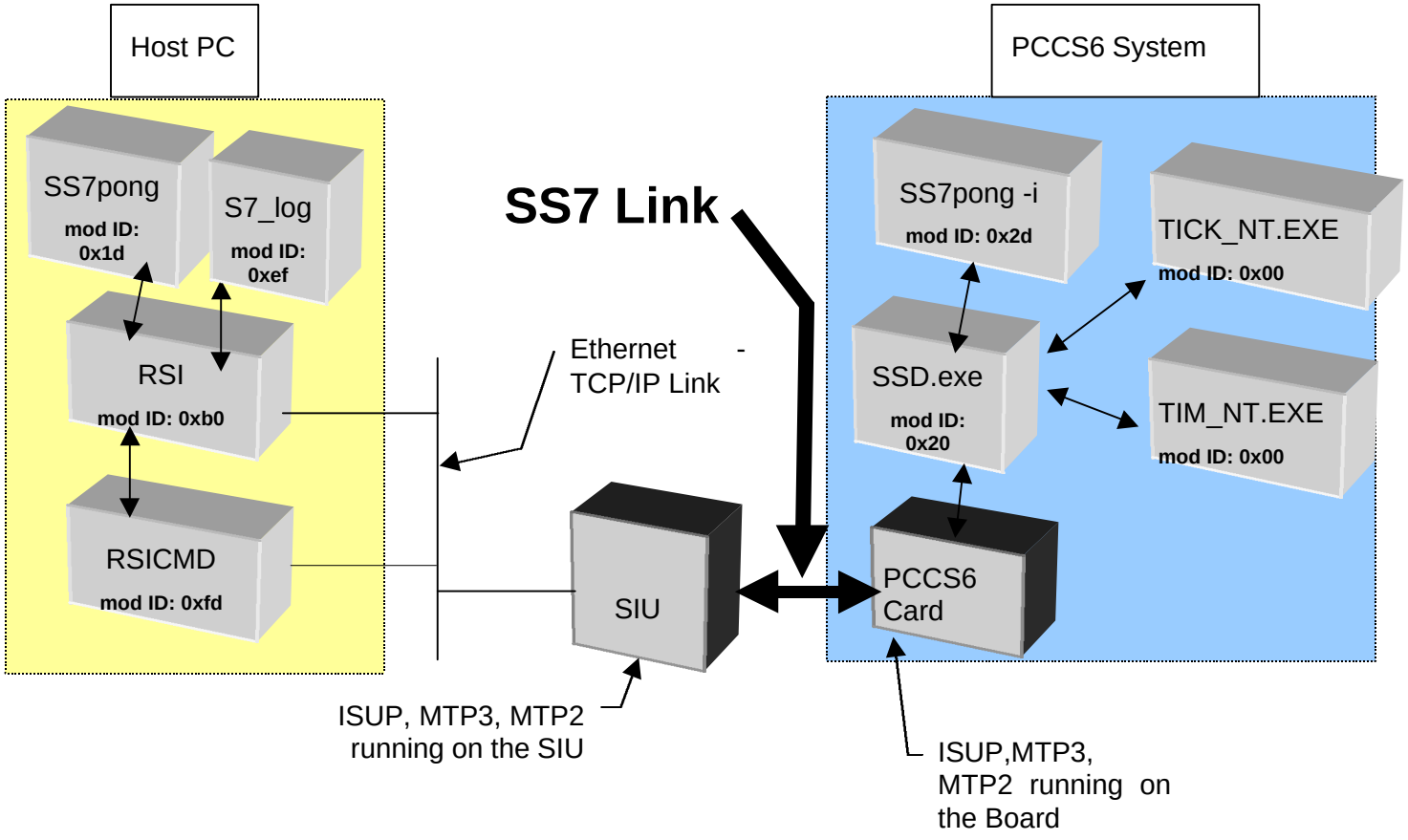

Figure 4. SIU<->PCCS6 Back to Back Configuration

Copyright Dialogic Corporation 1998, All Rights Reserved.

The Name Behind the Voice

12. Things to Check when there are Problems

1) Verify that PCCS6 board is set to T1 or E1 mode with the daughtercards. **NOTE**: This is not documented! For SIU's, rather than opening up system, send email to support@datakinetics.co.uk with SIU serial number and they will tell you how the SIU was configured.

2) Make sure that a cross over cable is used for the SS7 link. Also, when setting up two SIU's in dual redundancy mode, a cross over cable must be used between the two SIU's.

3) Make sure that each user app is compiled and linked with the appropriate development package that it will be run with. The libraries are not binary compatible between PCCS6 Development Pkg and OS Specific SIU Host SW!

   E.g. An application compiled and linked with the PCCS6 development package will not run on the SIU host or visa versa!

4) Check clocking! Ensure that one end is providing clock and one end is looking to the SS7 network for clock. Can not have both ends providing or both looking to the network for clock.

5) It is helpful to put '-d' for debug in the following FORK_PROCESS commands to get a verbose description of what is going on. **NOTE**: This is not documented.

   On the PCCS6 System:

   In run.bat: "gctload –d"

   In system.txt:

   "SSD.EXE –d"

   "S7_mgt.exe –d"

   On the SIU Host System:

   In run.bat: "gctload –d"

6) The system.txt file on the SIU should not need to be modified. The only exception to this is when you need to enable protocol verification between the SIU and the host machine for debugging purposes. You will want to add the -nv option in both the SIU and the SIU host's system.txt files, as follows:

   FORK_PROCESS ..\BIN\RSI.EXE -pfifo -r..\BIN\RSI_LNK.EXE -nv

   This option enables protocol verification between the SIU and the host machine. The -nv option should be removed from **BOTH** the SIU system.txt file and the SIU host system.txt file once the verification has been done.

7) If the DataKinetics SS7 Management process on the SIU crashes with the very helpful error message: "***S7MGT FAILED***", then comment out all timeslots above 24 in the MVIP_XCON commands in config.txt

Copyright Dialogic Corporation 1998, All Rights Reserved.

13. Running a sample application like ss7pong.exe

NOTE #1: ss7pong.exe is an internal demo created from the CTU demo. It is not delivered with any of the DataKinetics packages. To obtain a copy of this demo go to: \\donofria2\ss7pong. As part of the "User Part Development Package" there is a demo called CTU, which is very similar to ss7pong.exe. The CTU demo is an example call control application that runs above a telephony user part. When an incoming call is set up, CTU will answer this call and attempt an outgoing call on the next circuit.

The ss7pong.exe demo is very useful in proving a connection between two machines and in also learning the functions to prove one. It sends the following ISUP messages back and forth until you hit control-C. (IAM, ACM, ANM, REL and RLC). More importantly, it does not require any voice processing HW in order to verify the SS7 link. It's an excellent starting point.

The side that initiates (i.e. sends the first IAM message) must be started with the –i option. Only one call is active at a time, and the two sides take turns initiating the call.

When terminating ss7pong, to keep the ISUP modules happy, ss7pong should be canceled first on the side that initiated the first call (that is, ss7pong -i). Cancel it immediately after this side sends an REL message (releasing a call that the other side had initiated), but before it sends its next IAM. There is a one-second pause between calls for this purpose. Then, cancel ss7pong on the other side.

Gotchas:

The initiating side running ss7pong.exe with the "-i" option, is hardcoded to module id 0x2d. Using the –m option to reset the module id has no effect. The non-initiating side defaults to module id 0x1d, however this side can be overridden by the –m command line option.

NOTE #2: Make sure that if you change the module id using the –m option that the same module id is present in the FORK_PROCESS line in system.txt.

The ss7pong.exe does not activate the signaling link. For this reason, mtpsl must be run prior to running the demo. mtpsl must be run from the command line, not from within system.txt.

Copyright Dialogic Corporation 1998, All Rights Reserved.

The Name Behind the Voice

Appendix A

14. PCCS6 Files

14.1. config.txt

```
* Example config.txt for the System7 protocol configuration.
*
* Edit this file to reflect your configuration.
*
*       Physical Configuration
*       ----------------------
* board id is a logical number.  The driver accesses the board at the i/o port id
* and add a board at the offset. its trying to start a board at 204 . The i/o ports and the
* board ID's map 1-1 so board 0 = ox200
* up to 16 cards
PCCS6_BOARD  0   0   0   0x0002   isup76.dc2
*                    ^   ^   ^   ^          ^
*PortID -------------'    |   |   |          |
*BOARDID -------------'   |   |          |
*NUM_PCM ---------------'   |          |
*FLAGS --------------------------'          |
*Board code file -------------------------'
*
*       MTP Configuration
*       -----------------
*
MTP_CONFIG   1   0xb   0x0f00
*                   ^   ^     ^
* Local PC -------'   |      |
* SSF ------------------'      |
* Options -------------------'
*
MTP_LINKSET   0   0x2   2   0x0000
*                    ^   ^     ^    ^
* Linkset ID ------'    |     |     |
* Adj PC ---------------'      |     |
* Max number of links --'      |
* Flags -----------------------------'
*
```


Copyright Dialogic Corporation 1998, All Rights Reserved.

```
MTP_LINK   0   0   0   0   0   0  0x10   0x10   0x06
*              ^   ^   ^   ^   ^   ^      ^      ^
* LinkID ------'   |   |   |   |   |      |      |
* Linkset ID -----'   |   |   |   |      |      |
* Link ref ------------'   |   |   |      |      |
* SLC ----------------------'   |   |      |      |
* BPOS ----------------------'   |   |      |      |
* BLINK --------------------------'   |      |      |
* STREAM ----------------------------'      |      |
* TIMESLOT ---------------------------------'      |
* FLAGS ------------------------------------------------'
*
MTP_ROUTE   2   0   0x0020  * each user part is assigned a unique number
*                  ^   ^   ^
* DPC------------'    |   |
* Linkset ID --------'    |
* UP enable ------------'
* tup = 4
* sccp = 3
* isup = 5
* This is the number of bit switches to enable the right user parts. so in our
* example above, we are setting bit 5 to 1 for ISUP user part.
*
*
* This is normally used if someone is supplying their own user part.
*MTP_USER_PART   0x0a   0x23
*                           ^       ^
*  Service Ind ----------'          |
*  Mod ID --------------------------'
*
*
*      ISUP Configuration
*      ------------------
*
* Must enable ISUP's user part
*
ISUP_CONFIG   1   0xb   0x2d   0x0734 *  0x0001   0x0018
*                  ^   ^      ^       ^           ^          ^
*Local PC --------'    |      |       |           |          |
*SSF --------------------'      |       |           |          |
*User ID ----------------------'       |           |          |
*Options ------------------------------'           |          |
*Number of circuit groups (optional) -----'           |
*Number of circuits (optional) ---------------------------'
**
```


Copyright Dialogic Corporation 1998, All Rights Reserved.

```
ISUP_CFG_CCTGRP  0  2  0x01  0x01  0x00ff7fff  0x001e
*                ^  ^  ^     ^     ^           ^
*GID -------------'  |  |     |     |           |
*DPC ----------------'  |     |     |           |
*Base CIC --------------'     |     |           |
*Base CID --------------------'     |           |
*CIC Mask ---------------------------'           |
*Options ---------------------------------------'
```

14.2. system.txt

```
* Example system.txt for the System7 Windows NT Development
* Package.
*
* If neccessary, edit this file to reflect your configuration.
*
* Essential modules running on the host:
*
LOCAL       0x20            * ssd - Board Interface task
LOCAL       0x00            * Timer Task
*
* Optional modules running on the host:
*
LOCAL       0xcf            * s7_mgt - Management/config task
LOCAL       0x2d            * ss7pong.exe - Example user part task
LOCAL       0x3d            * mtpsl program
*
* Modules running on the board (all redirected via ssd):
*
REDIRECT     0x23   0x20    * ISUP module.
*REDIRECT    0x4a   0x20    * TUP module - Not currently using this
*REDIRECT    0x4a   0x20    * NUP module - Not currently using this
*REDIRECT    0x14   0x20    * TCAP module - Not currently using this
REDIRECT     0x22   0x20    * MTP3 module
REDIRECT     0x71   0x20    * MTP2 module
REDIRECT     0x10   0x20    * MVIP/SCbus/Clocking control module
REDIRECT     0x8e   0x20    * On-board management task
*
* Redirection of status:
*
REDIRECT     0xdf   0x2d                * LIU/MTP2 status messages to upe
REDIRECT     0xef   0x2d                * Other indications to the user application ss7pong.exe
*
* Now start-up all local tasks:
*
FORK_PROCESS    ..\BIN\SSD.EXE -d       * NOTE: the '-d' is only for debug purposes
```


Copyright Dialogic Corporation 1998, All Rights Reserved.

FORK_PROCESS ..\BIN\TIM_NT.EXE
FORK_PROCESS ..\BIN\TICK_NT.EXE
FORK_PROCESS ..\BIN\S7_MGT.EXE -d * NOTE: the '-d' is only for debug purposes

NOTE: Recall that ss7pong.exe doesn't activate signaling link from within application. Also recall that "**mtpsl**" can not be run from within system.txt, it must be done prior to running application from command line. Therefore, after running run.bat, you must do the following two things from the command line:

c:\mtpsl ACT 0 0

c:\ss7pong.exe –i // Recall that the other side must be started before this side initiates calls!

14.3. run.bat

start ..\bin\gctload –d

NOTE: The –d option is only for debugging purposes

Copyright Dialogic Corporation 1998, All Rights Reserved.

Appendix B

15. SIU Files

15.1. config.txt

```
*  DSC231 Protocol Configuration File (config.txt)
*  Refer to the DSC131/DSC231 User Manual.
*
*  SIU commands :
*  Set the SIU instance. Set to SIUA for standalone, SIUA or SIUB for dual oper
*  SIU_INSTANCE <instance_token> = SIUA | SIUB
*
SIU_INSTANCE SIUA
*
*  Define the network address of this SIU :
*  SIU_ADDR <network_address>
*
SIU_ADDR 146.152.64.25
*
*  Define the network address of the partner SIU (dual operation only) :
*  SIU_REM_ADDR <remote_address>
*
*SIU_REM_ADDR 193.195.185.251
*
*  Define the number of hosts that this SIU will connect to :
*  SIU_HOSTS <num_hosts>
*
SIU_HOSTS 1
*
*  Set physical Interface Parameters :
*  PCCS6_BOARD <port_id> <bpos> <num_pcm> <flags>
*  PCCS3_BOARD <port_id> <bpos> <num_pcm> <flags>
*
PCCS6_BOARD 0 4 0 0x0003
*
*  MTP Parameters :
*  MTP_CONFIG <local_spc> <ssf> <options>
*
MTP_CONFIG  2  0xb  0x0f00   * ANSI setup
*
```


Copyright Dialogic Corporation 1998, All Rights Reserved.

```
*  Define linksets :
*  MTP_LINKSET <linkset_id> <adjacent_spc> <num_links> <flags> <local_spc> <ssf>
*
MTP_LINKSET  0  1  1  0x0000  2  0xb
*
*  Define signalling links :
*  MTP_LINK <link_id> <linkset_id> <link_ref> <slc> <bpos> <blink> <stream> <ti
*
MTP_LINK  0  0  0  0  4  0  0x10  0x10  0x06
*
*  Define a route for each remote signalling point :
*  MTP_ROUTE <dpc> <linkset_id> <user_part_mask>
*
MTP_ROUTE  1  0  0x0020
*
*  ISUP Parameters :
*  ISUP_CONFIG <local_pc> <ssf> <user_id> <options> <num_grps> <num_ccts>
*
ISUP_CONFIG  2  0xb  0x1d  0x0734
*
*  Define ISUP circuit (groups) :
*  ISUP_CFG_CCTGRP <gid> <dpc> <base_cic> <base_cid> <cic_mask> <options> <host
*
ISUP_CFG_CCTGRP  0  1  0x01  0x01  0x00ff7fff  0x001e 0x00
*
*  TUP Parameters :
*  TUP_CONFIG <local_pc> <ssf> <user_id> <options> <num_grps> <num_ccts>
*
*TUP_CONFIG  2  0x8  0x1d  0x0000  8  96
*
*  Define TUP circuit (groups) :
*  TUP_CFG_CCTGRP <gid> <dpc> <base_cic> <base_cid> <cic_mask> <options> <host_
*
*TUP_CFG_CCTGRP  0  1  1  1  0x7fff7fff  0x0000 0x00
*
*  NUP Parameters :
*  NUP_CONFIG <local_pc> <ssf> <user_id> <options> [<num_grps> <num_ccts>]
*
*NUP_CONFIG 2  0x8  0x1d  0x0000  8  32
*
*  Define NUP circuit (groups) :
*  NUP_CFG_CCTGRP <gid> <dpc> <base_cic> <base_cid> <cic_mask> <options> [<host
*
*NUP_CFG_CCTGRP  0  1  1  1  0x7fff7fff  0x0003 * 0x00
*
*  SCCP Parameters :
*  SCCP_CONFIG <local_pc> <ssf> <options>
```


Copyright Dialogic Corporation 1998, All Rights Reserved.

The Name Behind the Voice

```
*
*SCCP_CONFIG  2  0x8  0x0000
*
*   Define SCCP Remote signalling points :
*   SCCP_RSP <spc> <rsp_flags>
*
*SCCP_RSP  1  0x00
*
*   Define all local sub-systems :
*   SCCP_LSS <ssn> <module_id> <lss_flags>
*
*SCCP_LSS  0x66  0x0d  0x0
*
*   Define all remote sub-systems :
*   SCCP_RSS <spc> <ssn> <rss_flags>
*
*SCCP_RSS  1  0x66  0x0
*
*   Define all local sub-systems that require notification of
*   changes in state of other signalling points or sub-systems :
*   SCCP_CONC_LSS <local_ssn> RSP <remote_spc>
*
*SCCP_CONC_LSS <local_ssn> RSS <remote_spc> <remote_ssn>
*
*SCCP_CONC_LSS  0x66  RSP  1
*SCCP_CONC_LSS  0x66  RSS  1  0x66
*
*   Define all remote signalling points that require notification
*   of change in state of local sub-systems :
*   SCCP_CONC_RSP <remote_spc> LSS <local_ssn>
*
*SCCP_CONC_RSP  1  LSS  0x66
*
*   The following commands allow the REM_API_ID module to receive
*   notification of changes of state of other signalling points or
*   sub-systems.
*
*SCCP_LSS  200  0xef  0
*SCCP_CONC_LSS  200   RSS  1  0x66
*
*   Cross Connections :
*   These commands control the connection of voice channels through
*   the SIU. The default configuration cross-connects all timeslots
*   other than 0 and 16 between the two ports of each PCCS6 card.
*   MVIP_XCON <bpos> <op_stream> <op_slot> <mode> <ip_stream> <ip_slot> <pattern>
*
MVIP_XCON  4  16  1   3  17  1   0
```


Copyright Dialogic Corporation 1998, All Rights Reserved.

```
MVIP_XCON  4  16  2   3  17  2   0
MVIP_XCON  4  16  3   3  17  3   0
MVIP_XCON  4  16  4   3  17  4   0
MVIP_XCON  4  16  5   3  17  5   0
MVIP_XCON  4  16  6   3  17  6   0
MVIP_XCON  4  16  7   3  17  7   0
MVIP_XCON  4  16  8   3  17  8   0
MVIP_XCON  4  16  9   3  17  9   0
MVIP_XCON  4  16  10  3  17  10  0
MVIP_XCON  4  16  11  3  17  11  0
MVIP_XCON  4  16  12  3  17  12  0
MVIP_XCON  4  16  13  3  17  13  0
MVIP_XCON  4  16  14  3  17  14  0
MVIP_XCON  4  16  15  3  17  15  0
*** Signalling on Timeslot 16 ***
MVIP_XCON  4  16  17  3  17  17  0
MVIP_XCON  4  16  18  3  17  18  0
MVIP_XCON  4  16  19  3  17  19  0
MVIP_XCON  4  16  20  3  17  20  0
MVIP_XCON  4  16  21  3  17  21  0
MVIP_XCON  4  16  22  3  17  22  0
MVIP_XCON  4  16  23  3  17  23  0
MVIP_XCON  4  16  24  3  17  24  0
*MVIP_XCON  4  16  25  3  17  25  0
*MVIP_XCON  4  16  26  3  17  26  0
*MVIP_XCON  4  16  27  3  17  27  0
*MVIP_XCON  4  16  28  3  17  28  0
*MVIP_XCON  4  16  29  3  17  29  0
*MVIP_XCON  4  16  30  3  17  30  0
*MVIP_XCON  4  16  31  3  17  31  0

**
* End of file
*
```

15.2. system.txt

```
*  DSC131/DSC231 System Configuration File (system.txt)
*
*
*=====================================================================
*
* Generated by : sysgen -sSIUA -ulSUP
* Date :        Wed May 20 13:19:10 1998
*
*=====================================================================
```


Copyright Dialogic Corporation 1998, All Rights Reserved.

```
*
*
* Definitions for both SIUA and SIUB:
*
LOCAL    0xdf          * siu_mgt management/configuration  process
LOCAL    0x20          * ssd signalling board driver
LOCAL    0x21          * cong congestion management
LOCAL    0x00          * timer
LOCAL    0xb0          * rsi host to SIU communication
LOCAL    0xc0          * rsi SIUA to SIUB communitcation
LOCAL    0x32          * rmm
LOCAL    0x22          * mtp (mtp3)
*
REDIRECT  0x71  0x20  * mtp2 running on signalling card
REDIRECT  0x10  0x20  * switch manager running on signalling card
REDIRECT  0x8e  0x20  * management running on signalling card
*
* Definitions unique to SIUA:
*
REDIRECT  0x42  0x32  * rmm from SIUB
REDIRECT  0x62  0x22  * mtp from SIUB
REDIRECT  0xaf  0xdf   * siumgt from SIUB
REDIRECT  0x52  0xc0  * rmm to SIUB
REDIRECT  0x72  0xc0  * mtp to SIUB
REDIRECT  0xbf  0xc0  * siumgt to SIUB
*
LOCAL    0x23          * ISUP on SIUA
REDIRECT  0x63  0x23  * ISUP from SIUB
REDIRECT  0x73  0xc0  * ISUP to SIUB
*
* Processes running on the remote host:
*
REDIRECT  0xef  0xb0  * Remote API
REDIRECT  0x0d  0xb0  * Application 0
REDIRECT  0x1d  0xb0  * Application 1
REDIRECT  0x2d  0xb0  * Application 2
REDIRECT  0x3d  0xb0  * Application 3
REDIRECT  0x4d  0xb0  * Application 4
*
* Processes running on SIUA and SIUB:
*
FORK_PROCESS  ./alarm
FORK_PROCESS  ./tim_qnx
FORK_PROCESS  ./tick_qnx
FORK_PROCESS  ./siu_mgt
FORK_PROCESS  ./rsi -p/ram0/fifo -r./rsi_lnk -l1 -m0xb0
FORK_PROCESS  ./rsi -p/ram0/fifo -r./rsi_lnk -l2 -m0xc0
```


Copyright Dialogic Corporation 1998, All Rights Reserved.

```
FORK_PROCESS  ./cng_qnx
FORK_PROCESS  ./ssd
FORK_PROCESS  ./rmm_qnx
FORK_PROCESS  ./mtp_qnx
*
* Optional processes on SIUA and SIUB:
*
FORK_PROCESS  ./isp_qnx  * ISUP
*
```


Copyright Dialogic Corporation 1998, All Rights Reserved.

The Name Behind the Voice

Appendix C

16. SIU Host Files

16.1. system.txt

```
* Example system.txt for the System7 Windows NT Development
* Package.
*
* If neccessary, edit this file to reflect your configuration.
*
* Optional modules running on the windows NT machine:
*
LOCAL       0xb0        * RSI Module ID
LOCAL       0xef        * s7_log Module ID recieves
LOCAL       0xfd        * RSICMD Module ID
LOCAL       0x1d        * ss7pong.exe Module ID
*
*
*
* Modules running on the SIU (all redirected via rsi):
*
REDIRECT      0x23   0xb0   * ISUP module.
REDIRECT      0x20   0xb0   * SSD - driver
REDIRECT      0x22   0xb0   * MTP3 module
REDIRECT      0x71   0xb0   * MTP2 module
REDIRECT      0x10   0xb0   * MVIP/SCbus/Clocking control module
REDIRECT      0xdf   0xb0   * LIU/MTP2 status messages to upe
REDIRECT      0x32   0xb0   * RNM (Redundancy handling for SIU)
*
* Now start-up all local tasks:
*
FORK_PROCESS    ..\BIN\RSI.EXE -pfifo -r..\BIN\RSI_LNK.EXE
* socket interface process.  Replaces the driver.  sends and receives
* messages to/from the SIU. Making use of the host operating system,
* socket library and TCP/IP.

FORK_PROCESS    ..\BIN\RSICMD.EXE 0 0xef 0 146.152.64.25 9000
* starts the link to the SIU. 9000 is the telnet port that this host will
* use to talk to the SIU.  This host is host '0', 9001 = host '1'. Setting
* up initial socket connection.
```


Copyright Dialogic Corporation 1998, All Rights Reserved.

FORK_PROCESS ..\BIN\s7_log.EXE
**** Siu Management Utility. It receives status messages from the SIU and***
**** prints them to the console.***
FORK_PROCESS ..\BIN\ss7pong.exe -m0x1d

16.2. run.bat

start ..\bin\gctload –d

NOTE: The –d option is only for debugging purposes

Copyright Dialogic Corporation 1998, All Rights Reserved.

References

1. The white paper entitled “Using DataKinetics SS7 Products” by Rob Teets was used as a starting point for this Application Note.

2. DataKinetics Entire Documentation Suite.

Copyright Dialogic Corporation 1998, All Rights Reserved.

The Name Behind the Voice