The HSMS Standard and Examples

HSMS

HSMS is intended as an alternative to SECS-I which utilizes RS-232 for applications where higher speed communication is needed or when a simple point-to-point topology is insufficient.

HSMS defines a communication interface suitable for the exchange of messages between computers in a semiconductor factory using a TCP/IP environment.

HSMS

SEMI E37

High-Speed
SECS Message Services,
Generic (HSMS)

SEMI E37.1

High-Speed
SECS Message Services,
Single Session
(HSMS-SS)

Details for SECS-I Replacement on Typical Equipment Using SECS-II With or Without GEM

SEMI E37.2

High-Speed
SECS Message Services,
General Session
(HSMS-GS)

Details for MESC/CTMC Protocol on Vacuum Cluster Equipment

HSMS

❖ BACKGROUND

Within the semiconductor industry, the primary equipment communication standard used since 1980 has been the SEMI E4 SECS-I protocol, which is based on RS-232 technology. In the mid-1980s, SEMI sanctioned a task force to develop a higher-speed alternative to E4. The Network interface Task Force (NITF) chose General Motors' Manufacturing Automation Protocol (MAP) as the protocol,then developed and published the SEMI E13 standard in 1990. Because of very limited use of the MAP protocol, however, E13 was not well accepted by the industry. Due to the advancement of network technology and its usage, a new task force was formed in 1992 to reexamine the issue. The High Speed Message Task force (HMTF) was formed to prepare for ballot a standard to address network communications with higher speed and throughput than E4.

HSMS

❖ BACKGROUND (continued)

The HMTF selected Transmission Control Protocol/Internet Protocol (TCP/IP) as the protocol for high speed messages. The HMTF recognized that TCP/IP was being widely used and was available on most computer platforms. The task force included in the development process input from industry equipment suppliers, equipment integrators, and IC manufacture users. Through the development and subsequent revision of a reference prototype implementation and after 19 task force meetings over three year, the HSMS E37 standard became official in July 1994.

HSMS

❖ HSMS Generic Services (E37-95)

The HSMS document provides the fundamental components for developing an HSMS-compliant communications interface. The document defines message exchange procedures for using the TCP/IP network protocol. Procedures are described for the following areas:

- Establishing a communications link between entities using a TCP/IP connection procedure
- Establishing and maintaining the protocol conventions necessary for exchanging SECS messages between entities
- Exchanging data using TCP/IP
- Recognizing error conditions
- Formally ending communications to confirm both parties no longer need the TCP/IP connection

HSMS

❖ HSMS Generic Services (continued)

- Logically breaking the communications link without any physical disconnect from the network medium
- Testing the communications link for connection integrity purposes
- Rejecting connection attempts from incompatible subsidiary standards

In addition, the document describes special considerations, such as network timeouts and handling multiple connections, which should be taken into account in a TCP/IP implementation. Information on documentation required in an HSMS implementation and example message exchange procedures using the Berkley Sockets Definition (BSD) interface and the Transport Layer Interface (TLI) are provided in the appendixes.

HSMS

❖ HSMS-SS

HSMS-SS is a subsidiary standard that provides a proposed subset of HSMS, including the minimum set of services required for use as a direct SECS-I replacement. The document defines a different state machine than the HSMS-GS document, with limited capabilities. Specifically, HSMS-SS imposes the following limitations:

- It eliminates the use of a number of HSMS-GS procedures which are intended to be used by implementations which support multiple TCP/IP connections.
- It limits other HSMS-GS procedures to simplify operation for the specific case of SECS-I replacements.

The document also explains what documentation is required in an HSMS-SS implementation and provides application notes on support of multiple hosts.

HSMS

❖ HSMS-GS

HSMS-GS also is a subsidiary standard to HSMS. HSMS-GS provides a proper subset of the main standard, including support for complex systems containing multiple, independently accessible subsystems, such as cluster tool or track systems. The document details extensions to the HSMS state machine in the form of additional state transition definitions and added state information. These additions provide capabilities that permit individual subentities of complex systems to be accessed separately during HSMS procedures. Specifically, HSMS-GS details the following additional capabilities:

HSMS

❖ HSMS-GS (continued)

- A Session Entity List consisting of a set of all session entities that are accessible via TCP/IP connection from an outside entity
- A Selected Entity List comprising a list of entities that are currently selected for access on a given TCP/IP connection
- A Selected Count that corresponds to the number of entity IDs currently selected

The document also explains what documentation is required in an HSMS-GS implementation and provides application notes on supporting both HSMS-GS and HSMS-SS simultaneously.

HSMS

❖ HSMS ADOPTION

The first adopters were equipment suppliers in the metrology and inspection area, who have had problems with transferring large amount of measurement data through RS-232. The restrictions of RS-232 have driven some metrology and inspection suppliers to develop proprietary network solutions for delivering information via the equipment’s interface.

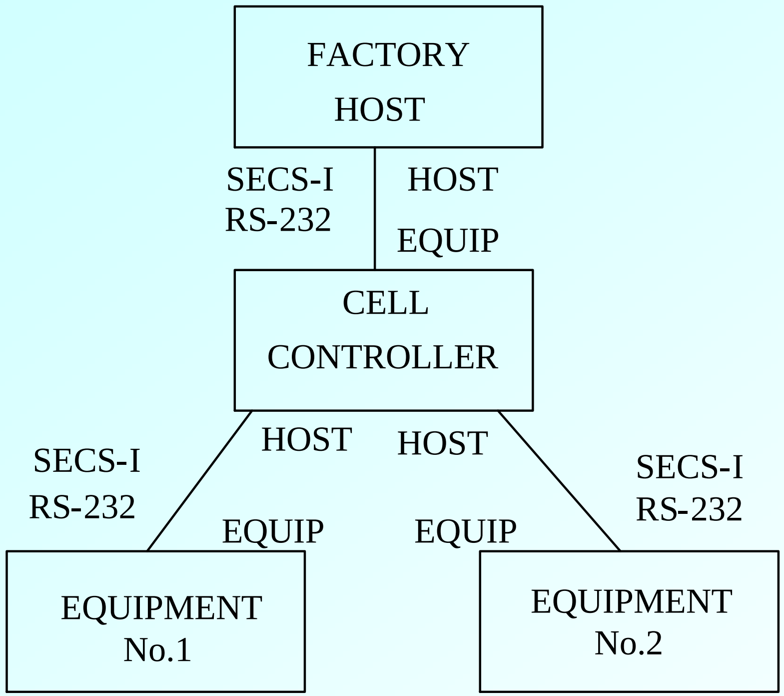
HSMS

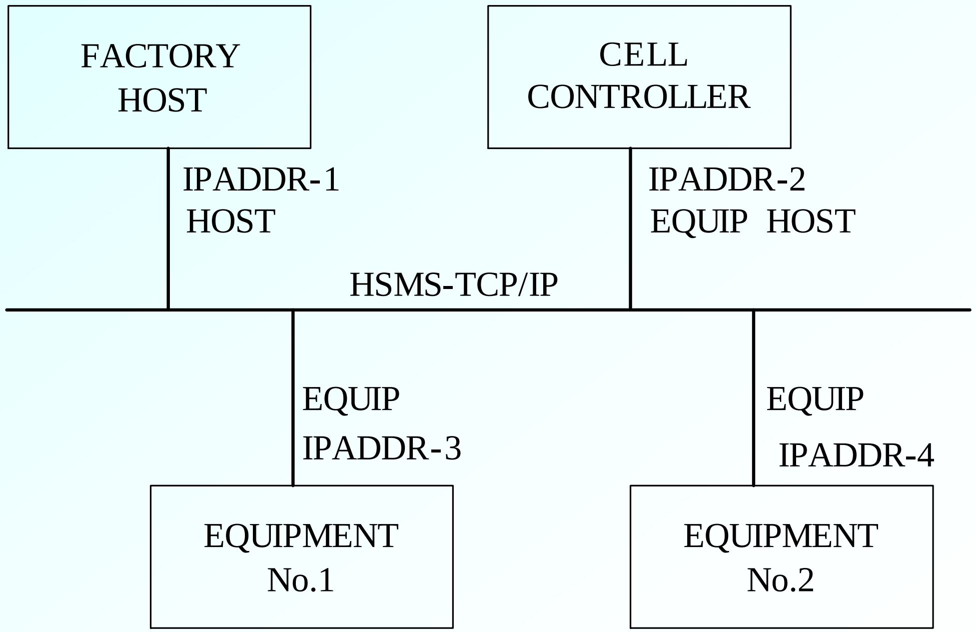
❖ HSMS ADOPTION (continued)

The use of HSMS by other types of equipment suppliers is driven largely by IC manufacturers’ communications requirement. These manufacturers’ interest in using HSMS is high because of the availability of TCP/IP support in most factory environments. A large plus for utilizing HSMS-compliant equipment within the factory is the flexibility of physical placement of equipment within the factory. With HSMS, there is no maximum length of cable restrictions, and moving equipment within the factory only requires reconfiguring a network address instead of supplying a dedicated RS-232 cable.

HSMS

SECS-I RS-232 Connections Versus HSMS TCP/IP Ethernet Connections





HSMS

HSMS Design Principles and Features

- Divide and conquer: separate problems get separate solutions

  - Reliable end-to-end message delivery (TCP Connection Maintenance)

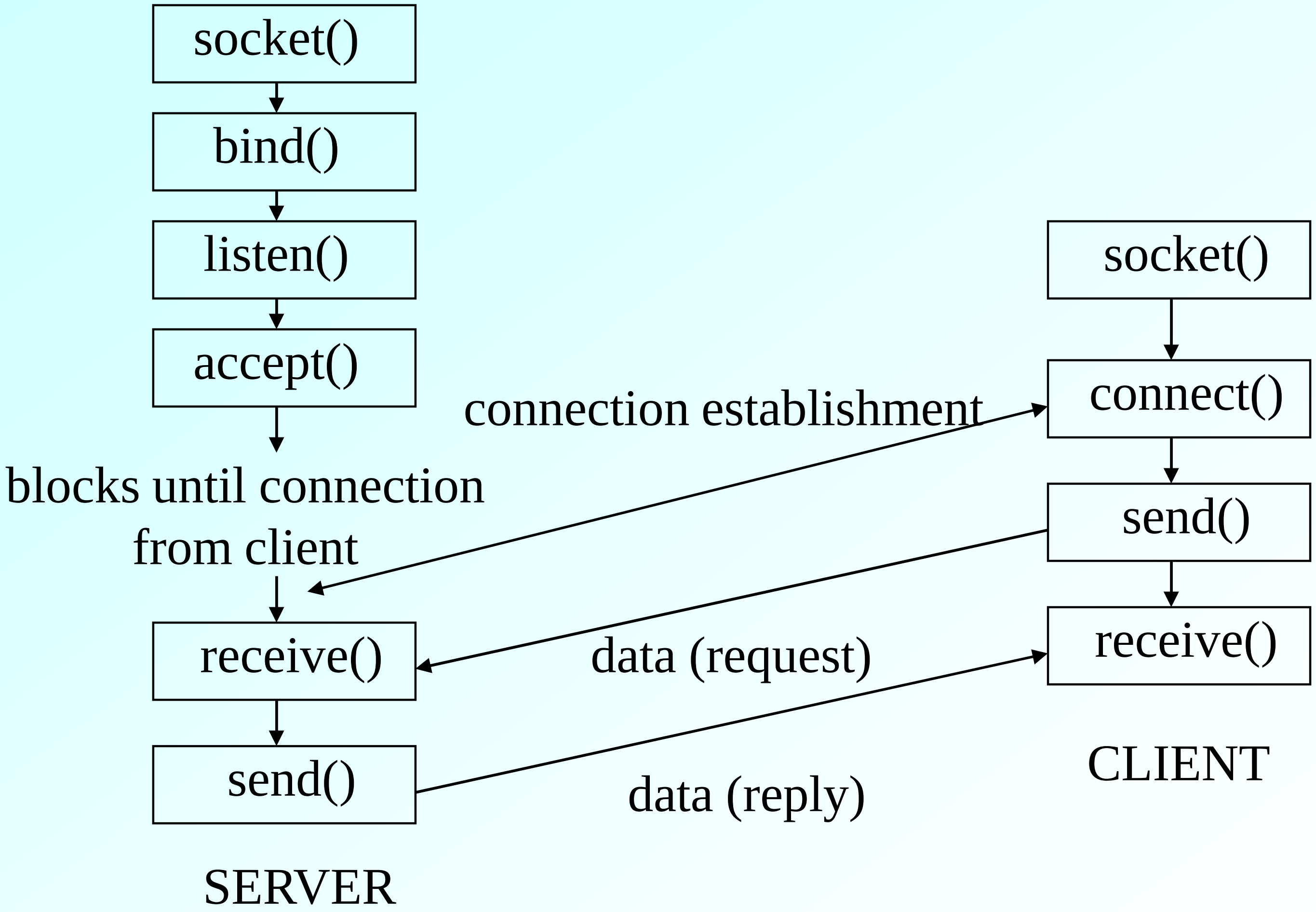
  - Semiconductor specific message content (HSMS Session Maintenance)

- Extensibility

  - " Spend a byte, save a standard": SType and PType bytes

HSMS

TCP Connection-Oriented Protocol



HSMS

Operation of HSMS

Begin HSMS Communication

BSD API Call	TLI API Call	Network Activity	TLI API Calls	BSD API Calls
prepare to initiate a connection request			prepare to receive a connection request	
skt=socket (...);	tep=t_open (...); t_bind (tep,...);		tep=t_open (...); t_bind (tep,...);	skt =socket (...); bind (skt,...); listen (skt,...);
initiate a connection request and wait for response			receive a connection request, accept it, and send response	
connect (skt,...);	t_connect (tep,...); t_rcvconnect (tep,...);	TCP/IP Connect Req Msg (s) → TCP/IP Accept Msg (s) ←	t_listen (tep,...); t_accept (tep,...)	accept (skt,...);
initiate an HSMS Select procedure: send request and receive response			response to HSMS select procedure : receive request and send response.	
write (skt, hdr, 14); read (skt, hdr, 14);	t_snd (tep, hdr, 14,0); t_rcv (tep, hdr, 14,...);	Select.req Message → Select.rsp Message ←	t_rcv (tep,hdr , 14,..); t_snd (tep,hdr , 14,0);	read (skt, hdr, 14); write (skt, hdr, 14);
send an HSMS data message as length bytes and header, followed by Text			receive an HSMS data message as length bytes and header, followed by text.	
Hdr→Len=Length; write (skt, hdr,14); write(skt, Text,..);	Hdr→Len=Length; t_snd (tep,hdr,14,0); t_snd (tep,Text,...);	HSMS Data Message (hdr) → HSMS Data Message (text) →	t_rcv (tep.hdr,14,..); t_rcv (tep,Text,..);	read (skt, hdr,14); read (skt,Text,..);

NCKU IME AutoLab

HSMS

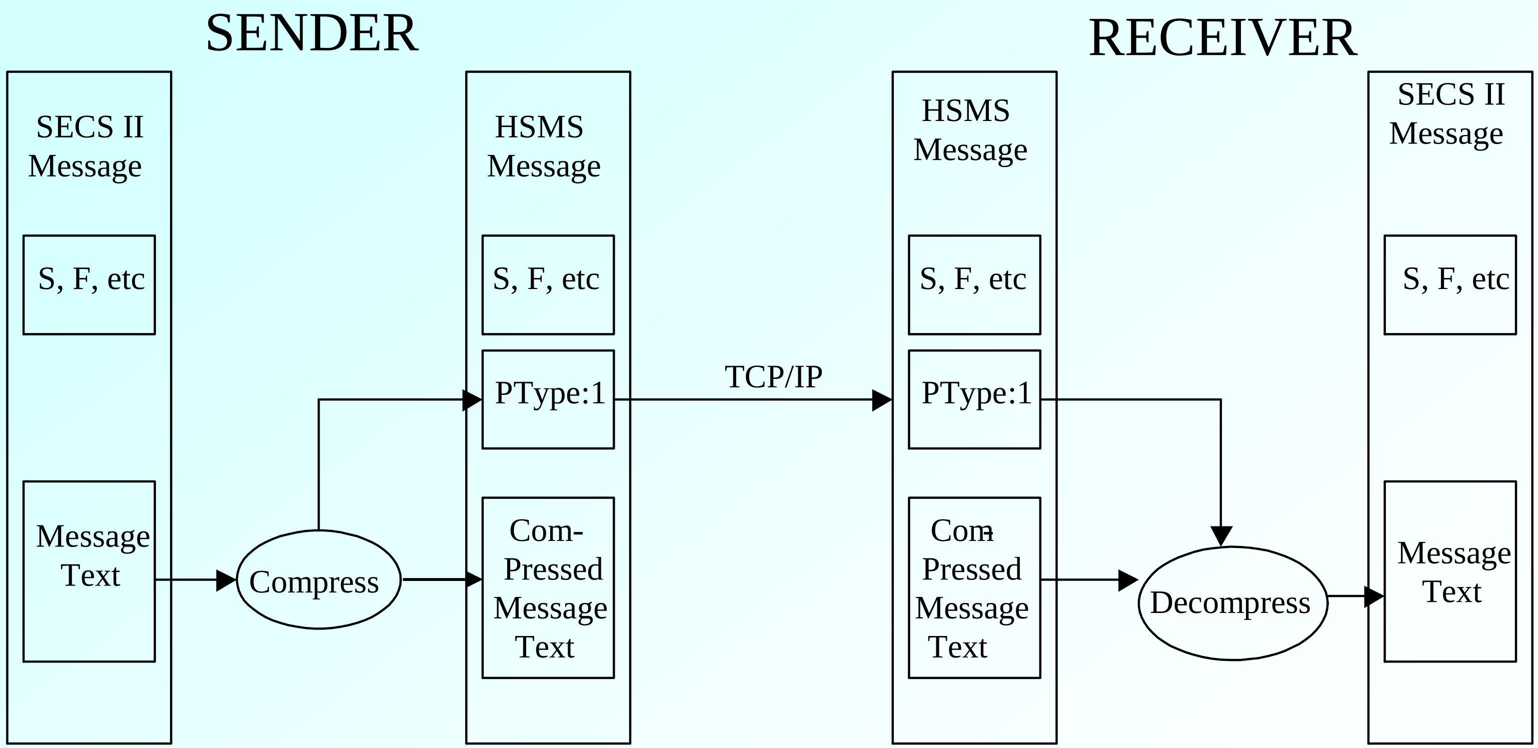
Ending Communication Using Deselect

BSD API Calls	TLI API Calls	Network Activity	TLI API Calls	BSD API Calls
Send the Deselect.req and receive the Deselect.rsp.			Receive the Deselect.req and send the Deselect.rsp.	
write (skt,hdr,14); read (skt,hdr,14);	t_snd (skt, hdr, 14, 0); t_rcv (skt, hdr, 14,...);	Deselect.rsp Message → ← Deselect.rsp Message	t_rcv (tep, hdr, 14,...); t_snd (tep, hdr, 14, 0);	read (skt, hdr, 14); write (skt, hdr, 14);
Disconnect the TCP/IP Connection			Respond to Disconnect of connection	
shutdown (skt,2); close (skt);	t _snddis (tep,...); t_closed (tep);	← TCP/IP Disconnect Msg (s) →	t_rcvdis (tep,...);	close (skt);

HSMS

Example Extensions: Data Compression

- Concept: save network bandwidth through compression technology



Data Compression

HSMS

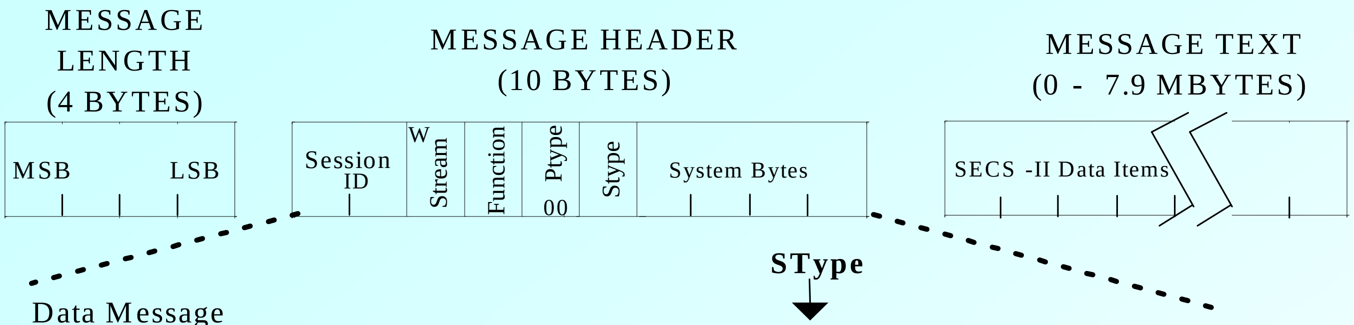
❖ HSMS MESSAGE FORMAT

HSMS-SS accesses TCP/IP utilizing a method called “TCP Streams.” Each TCP/IP Stream transmits data as an endless stream of bytes. TCP/IP is logically a full-duplex protocol, so there is one stream of bytes in one direction, and another stream of bytes in the reverse direction. HSMS-SS subdivides the TCP/IP Stream into discrete messages (see the next slide). Each HSMS-SS message begins with a four-byte Message Length field. This Message Length is always transmitted most significant byte first and least significant byte last. Next, there is a ten-byte Message Header, and finally the useful text of the HSMS-SS message, which can range from no data (Header-Only Message) to several megabytes in size. The Message Text is formatted as specified in the SECS-II standard.

HSMS - SS Message Format

HSMS

- HSMS-SS Message Format



Message					SType	
Data Message	DEVICE ID	SECS-II W-BIT & STREAM	SECS-II FUNCTION	00	**00**	SYSTEM BYTES
Select.req	0xFFFF	00	00	00	**01**	SYSTEM BYTES
Select.rsp	0xFFFF	00	SELECT STATUS	00	**02**	SYSTEM BYTES
Deselect.req	0xFFFF	00	00	00	**03**	SYSTEM BYTES
Deselect.rsp	0xFFFF	00	00	00	**04**	SYSTEM BYTES
Linktest.req	0xFFFF	00	00	00	**05**	SYSTEM BYTES
Linktest.rsp	0xFFFF	00	00	00	**06**	SYSTEM BYTES
Separate.req	0xFFFF	00	00	00	**09**	SYSTEM BYTES

HSMS

❖ HSMS-SS Message Format

- The Message Length specifies the number of bytes in the Message Header and Message Text (if any), but excludes the four bytes of the Message Length itself.
- Within the ten-byte Message Header, the first two bytes contain a Device ID, useful in complex equipment to identify a major subsystem. The third and fourth bytes of the Message Header have different uses depending on Stype, as described below. The fifth byte of the Message Header (Ptype) is always zero. The sixth byte of the Message Header (Stype) contain a code indicating whether this message is a Data Message (containing useful application data) or one of the five or six HSMS-SS Control Messages used for link management. The Control Messages Select.req and Select.rsp are used to establish a connection between the Host and the Equipment. Linktest.req and Linktest.rsp are used to verify that the connection is still active. Separate.req is used to terminate the connection. The seventh through tenth bytes of the Message Header contain the System Bytes, which are used logically to associate a Primary Message with the corresponding Reply Message.

HSMS

❖ HSMS-SS Message Format (continued)

- For a Data Message (Stype 00), bytes three and four of the Message Header contain SECS-II Stream and Function codes, which identify the topic of the message and which are further described later in the SECS-II standard. An odd numbered Function (least significant bit of Function is “1”) signifies a Primary Data Message, and an even numbered Function (with value one greater than the corresponding Primary Data Message) signifies a Reply Data Message.
- The ten-byte HSMS-SS Message Header looks a lot like the older SECS-I Block Header. In SECS-I, we had a one-byte Block Length, and a ten-byte header for each block. For HSMS-SS, the TCP/IP layer provides “hidden” logic for blocking the TCP/IP Stream transmission, in HSMS-SS we don’t need to worry about blocks. Instead, we deal with complete SECS-II messages. Only one ten-byte header is needed for the entire HSMS-SS message. In SECS-I, the fifth and sixth bytes of the Block Header contained a Block Count and E-Bit. In HSMS-SS, we don’t worry about blocks, so these bytes of the header are used for Ptype and Stype, as described above.

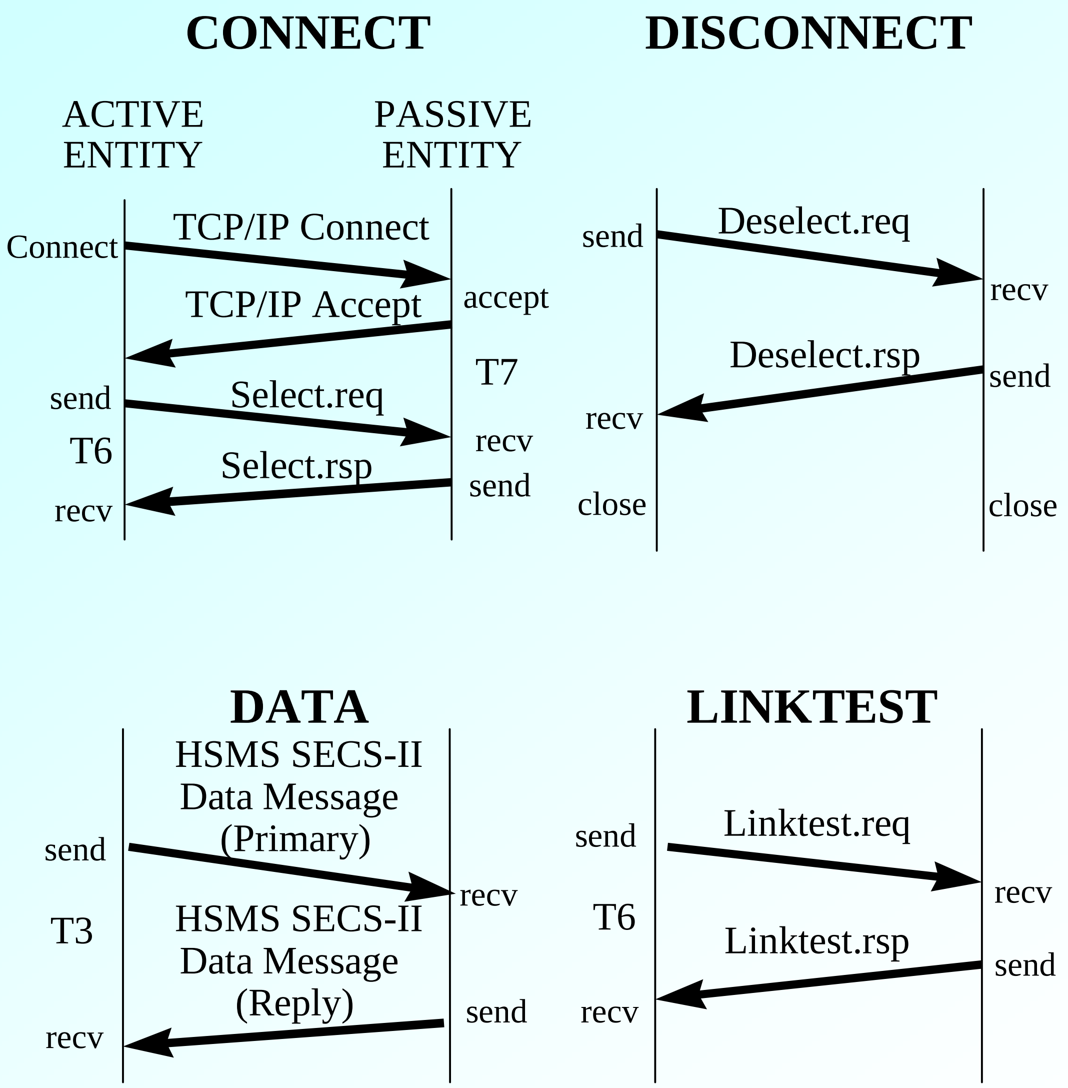
HSMS

❖ HSMS-SS PROCEDURES

The most important procedure is the Data procedure, which consists of sending a Primary SECS-II message in one direction, and possibly sending the appropriate SECS-II reply message back. Like SECS-I, either end of the HSMS-SS connection can initiate a transaction, several transactions can be in progress simultaneously, and HSMS-SS associates each Reply Message to the appropriate Primary Message. Unlike SECS-I, HSMS-SS also defines several additional procedures which are used to manage the TCP/IP connection.

HSMS

HSMS-SS Procedures



HSMS

❖ “CONNECT” PROCEDURE

The HSMS Connect Procedure establishes a logical connection between the Host and the Equipment. One end of the link, called the ACTIVE entity, initiates establishment of the connection by means of the TCP “connect” function. The opposite entity, called the PASSIVE entity, accepts the connection by means of the TCP “accept” function. In early draft specifications of HSMS-SS, this was the only logic required. However, testing of actual reference implementations uncovered undetected error conditions. For example, an HSMS ACTIVE Entity could (by mistake) establish a connection to an inappropriate partner, such as a print server, with very strange results when it proceeded to send SECS data messages. In the published HSMS-SS standard, the Connect Procedure has been enhanced to detect such errors, The ACTIVE Entity must send the HSMS-SS Control Message Select.req, and the PASSIVE Entity replies with Select.rsp. This step assures that both entities are in fact HSMS-SS entities.

HSMS

❖ "CONNECT" PROCEDURE (continued)

Various errors can occur during the Connect Procedure. The connect may fail, because of no matching accept. In this case, the ACTIVE Entity waits a specified interval (the T5 Timeout), and then again attempts the connect. After the TCP/IP connect/accept succeeds, the PASSIVE Entity initiates the T7 Timeout. If the PASSIVE entity fails to receive Select.req within the T7 timeout, the Connect Procedure fails. After the ACTIVE Entity sends Select.req, it starts the T6 Timeout. If the ACTIVE Entity fails to receive Select.rsp within the T6 timeout, the Connect Procedure fails.

HSMS

❖ "CONNECT" PROCEDURE (continued)

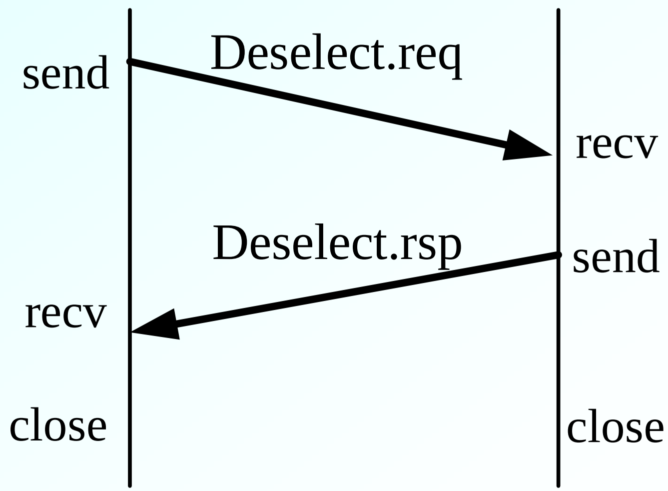
In most conventional TCP/IP implementations, even after the PASSIVE Entity has accepted the TCP/IP connect, a second ACTIVE Entity can attempt to connect to the same PASSIVE Entity. Many non-HSMS TCP/IP protocols (e.g., a print sever) allow several ACTIVE Entities to connect simultaneously to a single PASSIVE Entity. HSMS-SS typically does not allow this. So, the Select.rsp message contains a Select Status code (in Message Header Byte 4). In a successful connect procedure, the PASSIVE Entity sends Select.rsp with Select Status zero. When rejecting an attempted simultaneous connect by a second ACTIVE Entity, the PASSIVE Entity sends Select.rsp with Select Status non-zero. Testing on a variety of TCP/IP implementations has proven that this logic provides rapid detection of a connect failure by both sides.

HSMS

❖ "SEPARATE" PROCEDURE

To break an HSMS-SS connection, either side can send the HSMS-SS Control Message Separate.req. After sending or receiving Separate.req, and HSMS-SS entity should use the TCP/IP "close" function to terminate the connection.

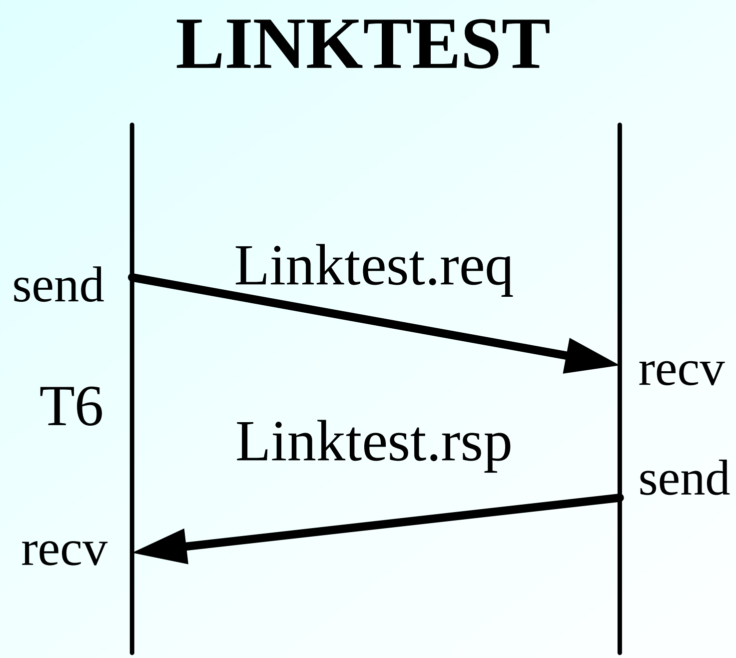
DISCONNECT



HSMS

❖ "LINKTEST" PROCEDURE

It is always possible for one end of a TCP/IP connection to "die" or to close the link ungracefully. Some TCP/IP implementations report this condition promptly to the other end of the link. Unfortunately, certain TCP/IP implementations may take as long as 15 minutes to report such a condition. For this reason, the Linktest procedure is sometimes useful to determine whether the HSMS-SS connection is still active. The entity initiating the test sends the HSMS-SS Control Message Linktest.req, and starts the T6 timeout. The opposite end replies with Linktest.rsp. If the initiating entity fails to receive Linktest.rsp within the T6 timeout, it assumes that the connection has failed and terminates it using the TCP/IP close function.



HSMS

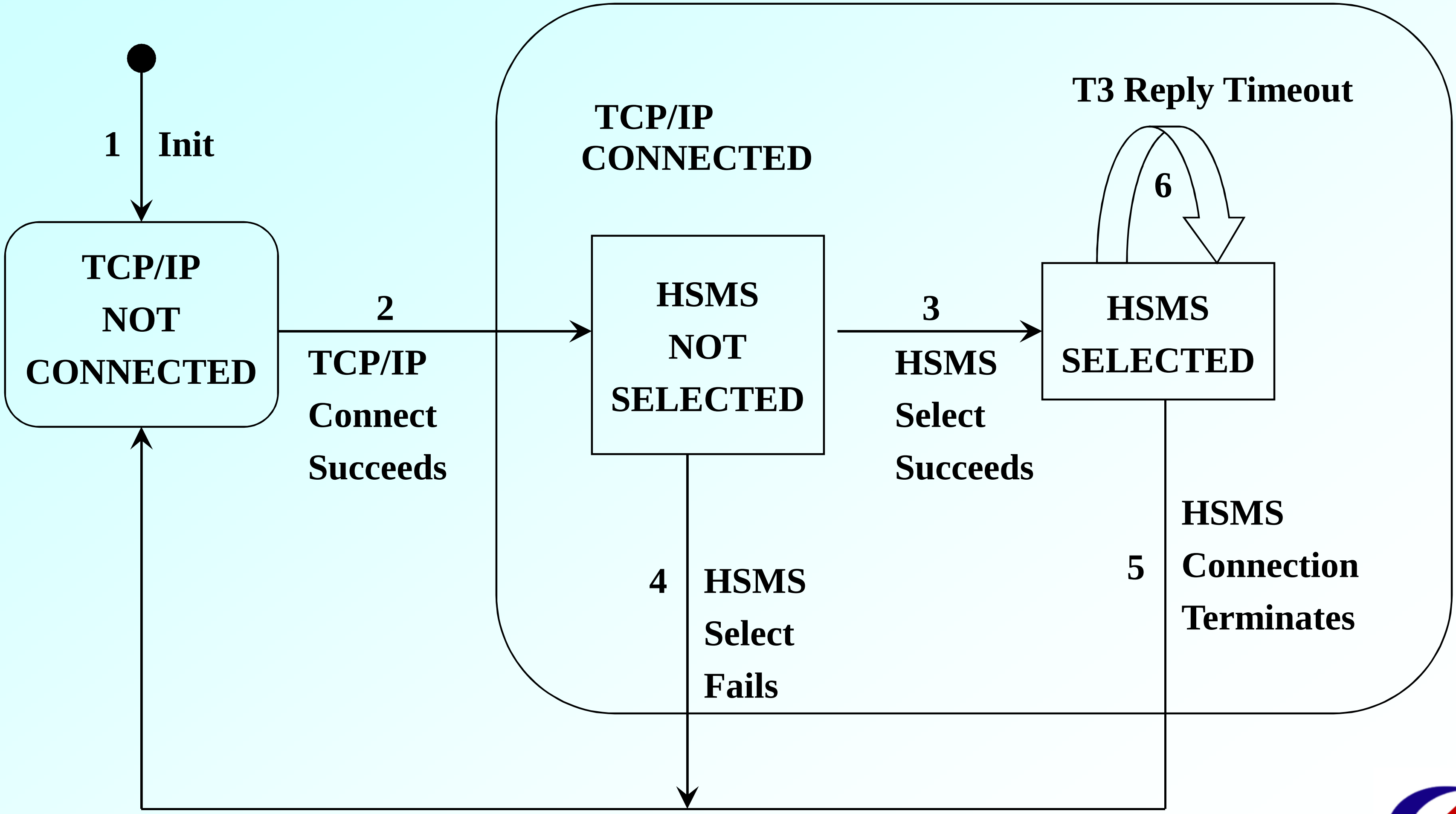
HSMS Timeouts

Parameter Name	***Value Range***	***Resolution***	***Typical Value***	***Description***
T3 Reply Timeout	1-120 seconds	1 second	45 seconds	Reply Timeout. Specifies maximum amount of time an entity expecting a reply message will wait for that reply.
T5 Connect Separation Timeout	1-240 seconds	1 second	10 seconds	Connection Separation Timeout. Specifies the amount of time which must elapse between successive attempts to Connect to a given remote entity.
T6 Control Transaction Timeout	1-240 seconds	1 second	5 seconds	Control Transaction Timeout. Specifies the time which a control transaction may remain open before it is considered a communications failure.
T7 NOT SELECTED Timeout	1-240 seconds	1 second	10 seconds	Time which a TCP/IP connection can remain in NOT SELECTED state (i.e., no HSMS activity) before it is considered a communications failure.
T8 Network Intercharacter Timeout	1-120seconds	1 second	5 seconds	Maximum time between successive bytes of a single HSMS message which may expire before it is considered a communications failure.
Connect Mode	PASSIVE ACTIVE	-	-	Connect Mode. Specifies the logic this local entity will use during HSMS connection establishment.
Local Entity IP Address and Port number	determined by TCP/IP conventions	-	-	Required for any entity operating in PASSIVE mode. Determines the address on which the local entity will listen for incoming connection requests.
Remote Entity IP Address and Port Number	determined by TCP/IP conventions	-	-	Required for any entity operating in ACTIVE mode. Determines the address of the remote entity to which the local entity will attempt to connect .

Note:Parameter defaults shown above are for small networks (10 nodes or less). Settings may need to be adjusted for larger net-work configurations.

HSMS

HSMS-SS STATE Machine



HSMS

❖ CONFIGURING HSMS-SS PARAMETERS

Like SECS-I, you must configure one end of the link as the HOST and the other end as EQUIPMENT, and you must specify the Equipment's SECS Device ID. For connection purposes, you must configure one end as ACTIVE and the other end as PASSIVE. There is some advantage to configuring HOST as ACTIVE, since this makes it easier to switch Host Computers; however, this is a factory choice. You must configure the IP Address and TCP Port number of the PASSIVE Entity. The T3 Reply Timeout is the same as for SECS-I. The T5 Connect Separation Timeout controls how often the ACTIVE Entity will re-try its attempts to establish the TCP/IP connection. The T6 Control Transaction Timeout is a reply timeout for HSMS Control Transactions (Select and Linktest). The T7 Not Selected Timeout controls how long the PASSIVE Entity will wait to receive Select.req during a Connect procedure. The T8 Network Timeout controls how long an application waits for an unresponsive TCP/IP layer.

HSMS

HSMS-SS Configuration

Parameter	Typical Value	
Host or EQUIPMENT		◆
SECS DEVICE ID		◆
ACTIVE or PASSIVE	Host is ACTIVE	
PASSIVE Entity IP Address	140.116.86.150	
PASSIVE Entity TCP Port	5000	
T3 Reply Timeout	45 Seconds	◆
T5 Connect Separation Timeout	10 Seconds	
T6 Control Transaction Timeout	5 Seconds	
T7 Not Selected Timeout	10 Seconds	
T8 Network Intercharacter Timeout	5 Seconds	

◆ - Same as SECS-I
SECS-I Parameters Not Needed:
Baud Rate, T1, T2, T4, RTY

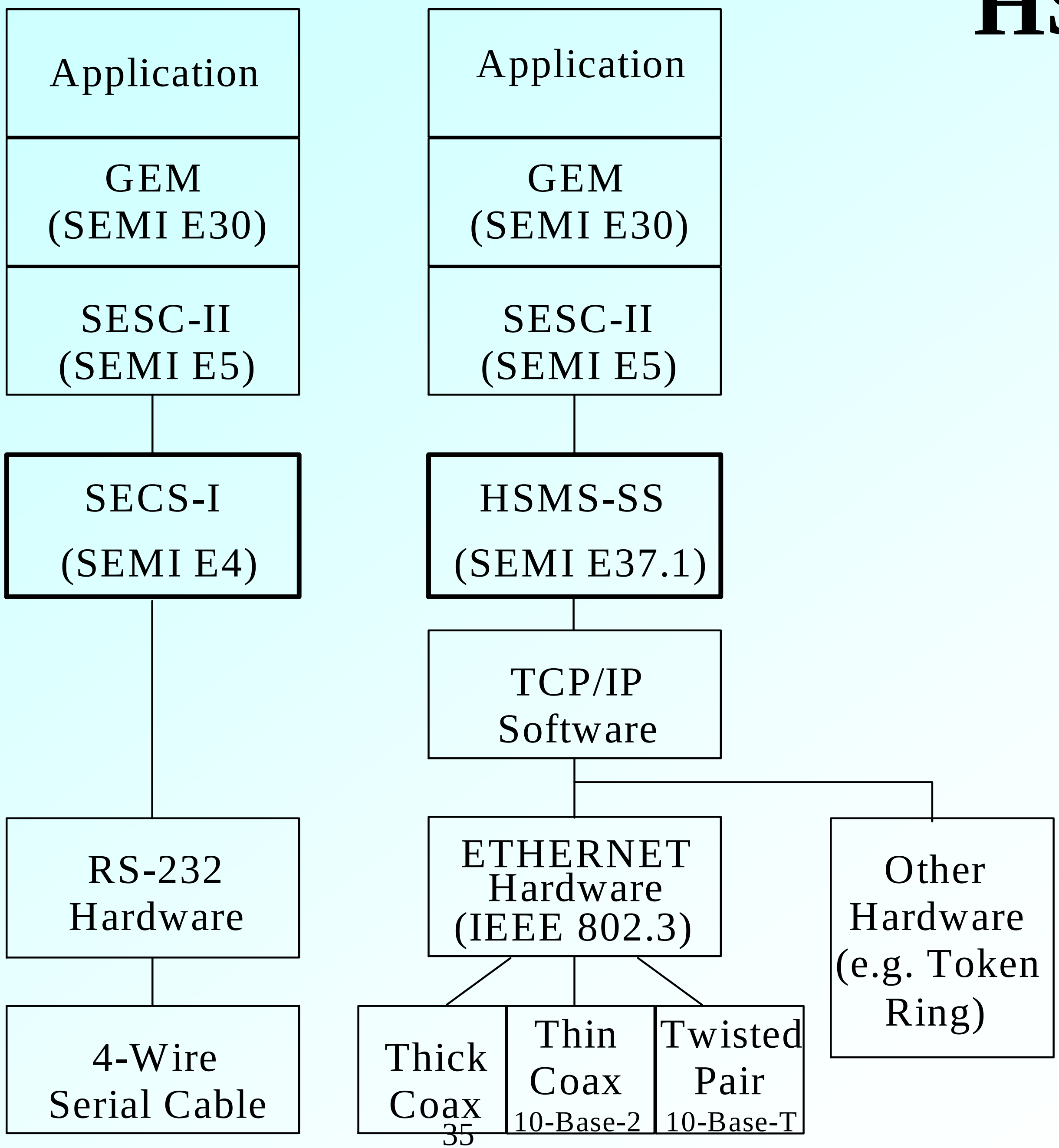
HSMS

❖ COMPARING SECS-II AND HSMS-SS PROTOCOL STACKS

The GEM and SECS-II standards can be used with either SECS-I or HSMS-SS. SECS-I uses RS-232 and a four-wire serial cable as its physical layer. HSMS-SS requires a foundation layer of TCP/IP software. Most HSMS-SS user prefer Ethernet (IEEE 802.3), but TCP/IP also supports other protocols, such as Token Ring (IEEE 802.5). Even with Ethernet, there are several cable options, including thick Coaxial cable, Thin Coaxial Cable (10-Base-2), and Twisted Pair (10-Base-T). The HSMS standards do not specify the physical layer, so it is important for users and suppliers of HSMS to negotiate agreements for local standards at the physical layer. An advantage of not specifying the physical layer is that as new TCP/IP supported protocols such as Fast Ethernet, Fiber Distributed Data Interface (FDDI), etc., become commercially practical, they can be used with HSMS-SS.

HSMS

SECS-I vs HSMS-SS Protocol Stacks



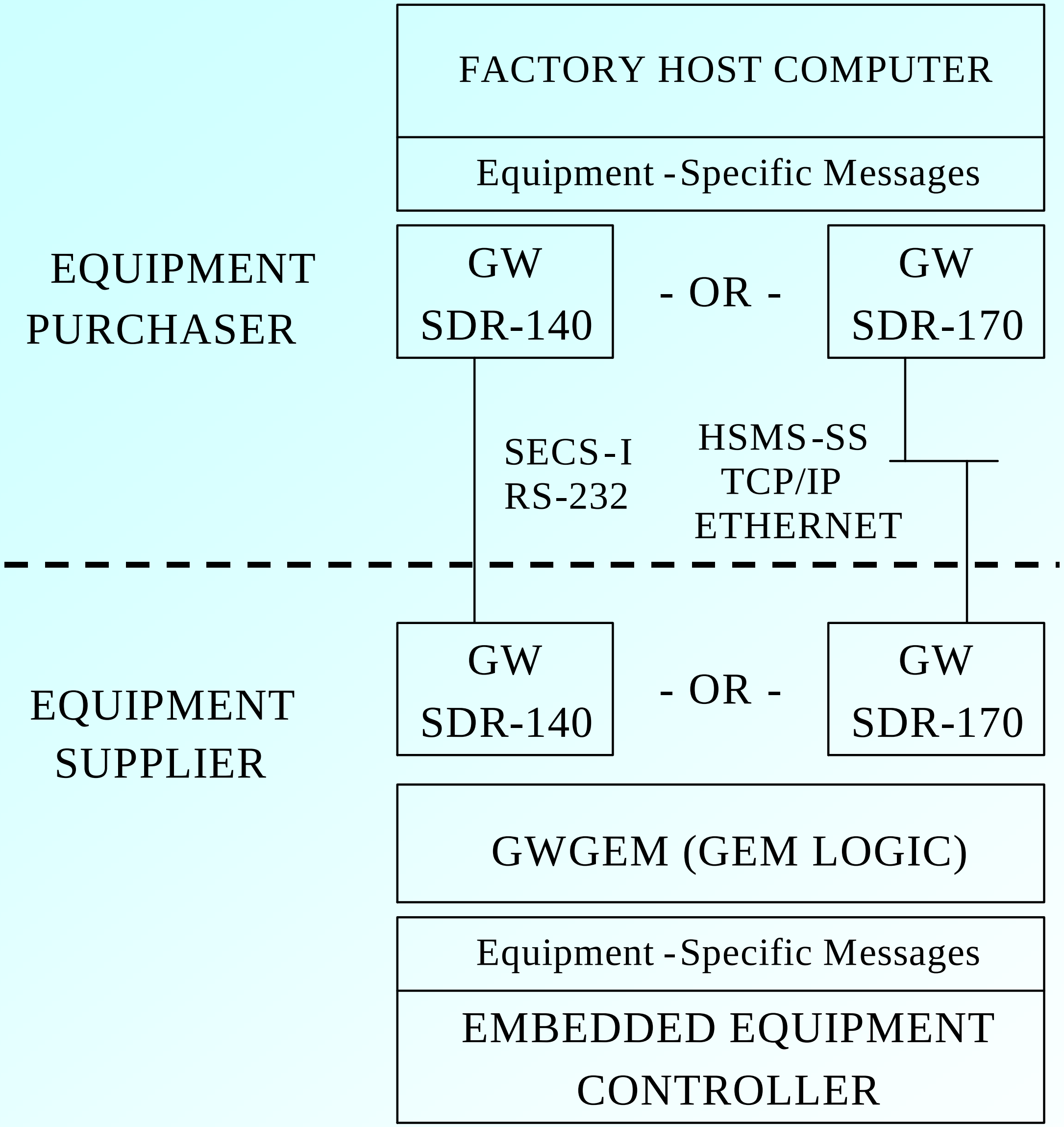
HSMS

SUPPORTING BOTH HSMS-SS AND SECS-I

Most existing factories are set up to use SECS-I, while only a few newer factories are ready to use HSMS-SS. A gradual shift to HSMS-SS is anticipated, but for several years, equipment suppliers will need to provide both protocols for different customers. Equipment can be designed with “plug and play” software components to make this straightforward. Figure 12 shows how substituting SECS-I for HSMS changes only the lower levels of the protocols. SECS-II, GEM, and (most importantly) the equipment or Line Control Computer application software does not need to change. Converting an equipment or Line Control Computer is easy─one simply swaps models of the low-level commercial SDR SECS Driver software. Except for speed, most other aspects of the equipment remain unchanged.

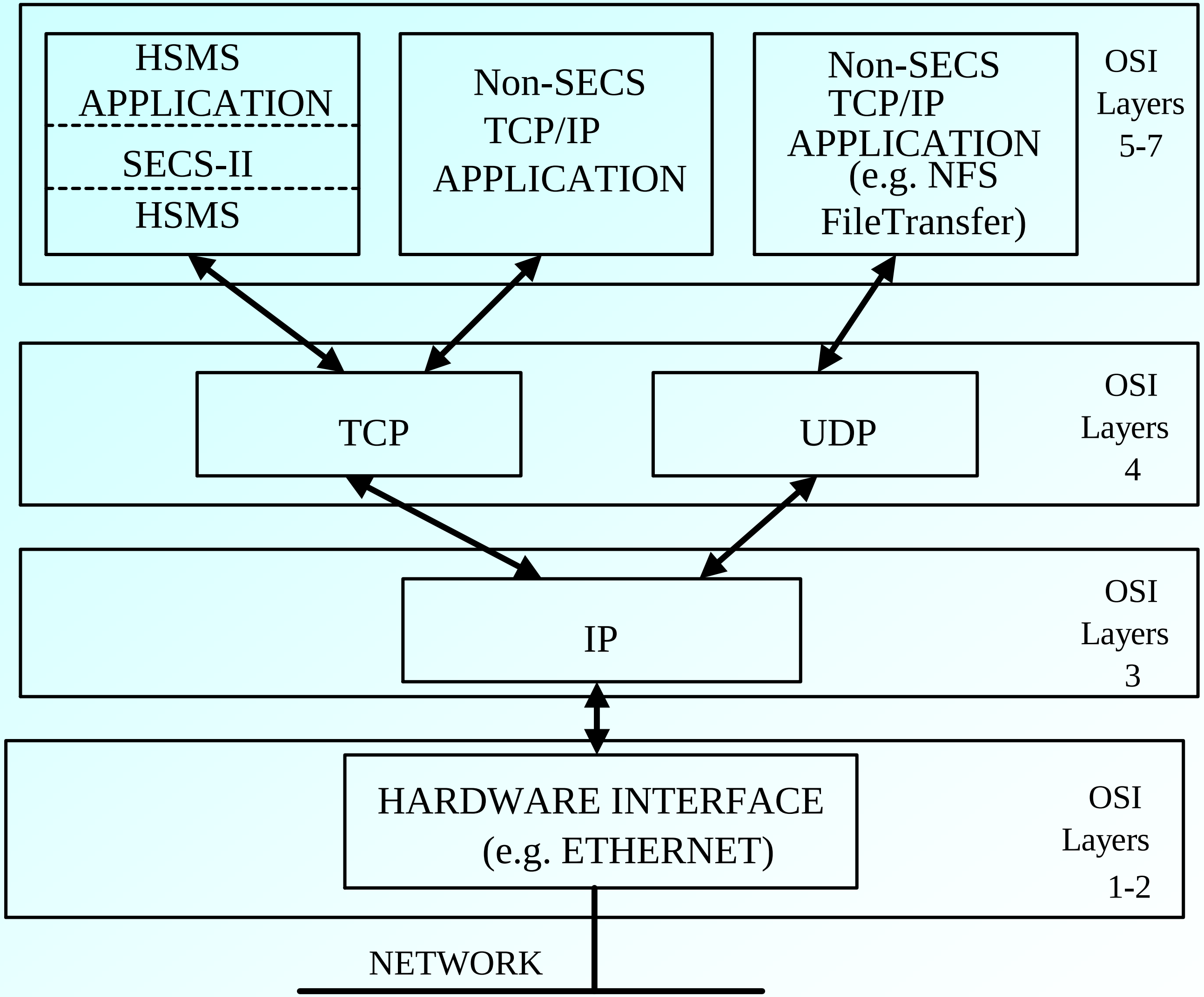
HSMS

Easy Conversion Between SECS-I and HSMS



HSMS

HSMS Can Share Network with Other TCP/IP Protocols



HSMS

Comparison of SECS-I and HSMS

Feature	*SECS-I*	*HSMS*
Communications Protocol Base	RS-232	TCP/IP
Physical Layer	25-pin connector and 4-wire serial cable	Physical layer not defined. HSMS allows any TCP/IP supported physical medium. Typical example is Ethernet (IEEE 802.3) and thin coax (10-BASE-2).
Communications Speed	Typically about 1000 bytes/second (assuming 9600 baud).	Typically 10 MBits/second (assuming typical Ethernet).
Connections	One physical RS-232 cable per SECS-I connection.	One physical network cable can support many HSMS Connections.
Message Format	Message text is SECS-II Data Items. Transmits a SECS-II message as a series of transmittal blocks each approximately 256 bytes in size. Each block has a one-byte block length, a ten-byte Block Header, text, and a two-byte Checksum.	Message Text is SECS-II Data Items. Transmits a SECS-II message as a TCP/IP byte stream. The message has a four-byte Message Length, a ten-byte Message Header, and text. The TCP/IP layer may impose blocking limits which depend on the physical layer used, but this blocking is transparent to the TCP/IP API and is outside the scope of HSMS.

HSMS

Comparison of SECS-I and HSMS (continued)

Feature	*SECS-I*	*HSMS*
Header	Ten-byte header on each block of a message. Header bytes 4-5 contains E-Bit and Block Number.	One ten-byte Header for the entire message. Header bytes 4-5 contain PType and SType. Header bytes 2-3 are W-Bit, Stream, and Function when SType = 0 (Data Message). For SType not equal to 0 (Control Message), bytes 2-3 have other uses. No R-Bit.
Maximum message size	Limited to approximately 7.9 million bytes (32767 blocks times 244 text bytes per block).	Message size limited by 4-byte message length (approximately 4 GBytes). Local implementation of TCP/IP and HSMS may further limit this in practice.
Protocol Parameters (Common)	T3 Reply Timeout Device ID	T3 Reply Timeout Session ID (analogous to Device ID).
Protocol Parameters (SECS-I only)	Baud Rate T1 Inter-Character Timeout T2 Block Protocol Timeout T4 Inter-Block Timeout RTY Retry Count Host/Equipment	Not used in HSMS. Corresponding issues addressed by TCP/IP layers.
Protocol Parameters (HSMS Only)	Not needed by SECS-I.	IP Address and Port of Passive Entity. T5 Connect Separation Timeout. T6 Control Transaction Timeout. T7 NOT SELECTED Timeout. T8 Network Intercharacter Timeout.