APPENDIX H

INTRANET TOPOLOGY UPDATE

H.1. General.

H.1.1. Scope. This appendix describes a procedure for active intranet topology updates. The intranet is defined as all processors and CNRs within a single transmission channel.

H.1.2. Application. This appendix is a mandatory part of MIL-STD-188-220. The information contained herein is intended for compliance.

H.2. Applicable Documents. This section is not applicable to this appendix.

H.3. Problem overview. Figure H-1 shows a sample extended CNR network. Each node labeled A through H is considered to be a radio with an associated communication processor. The dotted ovals indicate subsets of connectivity. Figure H-2 is a link diagram of the sample network. Assuming the nodes know nothing about neighbor nodes that are more than 1 hop away, they need to exchange connectivity information. The topology update packet is used to exchange topology information to build up a more complete view of the intranet's topology at every node.

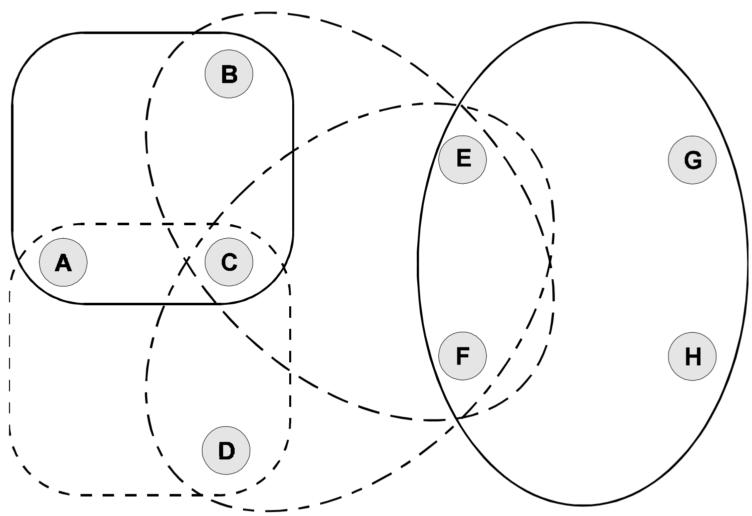


FIGURE H-1. Sample intranet.

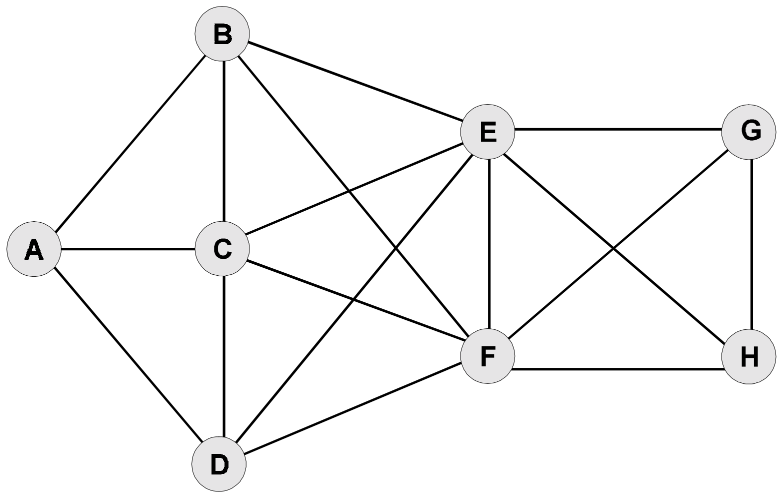


FIGURE H-2. Link diagram of sample network.

H.3.1. Routing trees. Each node should store topology information as a routing tree graph. Considering the network in Figure H-2, Figure H-3 shows the routing tree for nodes A and C prior to the exchange of any topology information. The routing trees for A and C contain only their nearest neighbors - those nodes which they can talk to directly. Similar graphs would exist for all other nodes.

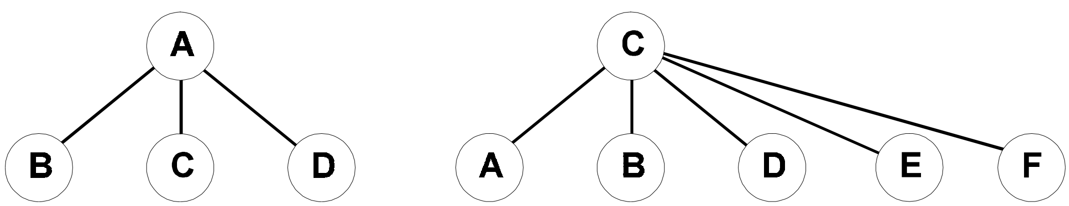


FIGURE H-3. Routing tree for nodes A and C.

H.4. Topology updates.

H.4.1. Exchanging routing trees. Nodes in the network gain more topology information by multicasting their individual routing trees to their nearest neighbor nodes. This exchange of routing trees will percolate more complete topology information through the network. For example, assume the routing trees for all nodes in Figure H-2 initially contain only nearest neighbors (nodes who are in direct communication with the given node). If node C multicasts its topology information to all nodes one hop away (those which are nearest neighbors), all neighbor nodes integrate C's routing tree into their own. Node A would integrate the graph for Node C into its routing tree as shown in Figure H-4.

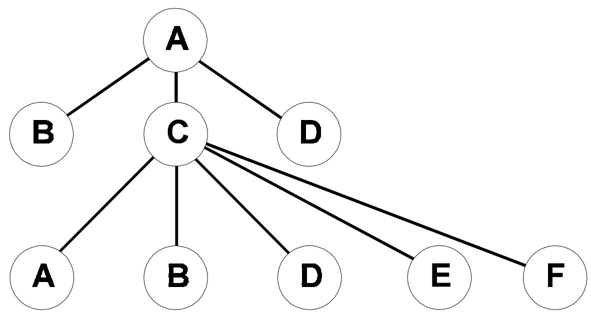


FIGURE H-4. Concatenated routing tree for node A.

Before the routing tree is saved, Node A prunes any successive instances of itself. For instance, in Figure H-4, the link from A to C is the same as the link from C to A; therefore, the link from C to A is removed. All redundant identical links are also pruned. These are links with the order of the end points reversed.

H.4.2. Topology tables. The topology table for A is shown in Table H-1. It assumes no nodes are in quiet mode, all nodes can participate in relay, and all links have a cost of 1. The actual link layer addresses for the nodes would be placed into the table in place of the symbols A, B, C, etc. The extension bit in the address octet would always be set to 0 for topology updates.

TABLE H-1. Topology table for node A.

Node Address	Node Predecessor	Hops	Cost	NR	Quiet
B	A	1	1	0	0
C	A	1	1	0	0
D	A	1	1	0	0
B	C	2	1	0	0
D	C	2	1	0	0
E	C	2	1	0	0
F	C	2	1	0	0

There are two entries for node B indicating that there are two paths from A to B. This table could be immediately copied to the respective fields of a topology update packet. The predecessor address is not included in the topology update packet for nearest neighbor nodes because the predecessor is, by definition, the originator node.

H.4.3. Sparse routing trees. Exchanging full routing tree tables provides full topology information; however, the amount of data in the routing tree gets very large, especially for fully connected nets. The number of links in a fully connected net with n nodes is n(n-1)/2. Although full routing trees should be stored by a node, exchanging these routing trees may consume too much bandwidth. A smaller copy of the full routing tree (called a sparse routing tree) should be prepared for transmission to neighbor nodes. To reduce the number of branches in the routing tree, some of the paths to duplicate nodes on the tree are pruned according to following rules:

a. Only the shortest paths from the root node to another node are retained.

b. For redundant paths from a root node to another node which are the same length (same number of links in the routing tree), at most 2 are retained. Some redundancy in paths is necessary for volatile networks.

For the previous example, the path from C to B and C to D would be pruned, since there are already shorter paths from A to C and A to D. The pruning yields the sparse routing tree in Figure H-5 and Table H-2.

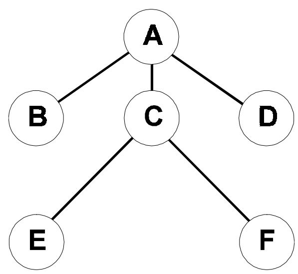


FIGURE H-5. Sparse routing tree for node A.

TABLE H-2. Sparse routing tree for node A.

Node Address	Node Predecessor	Hops	Cost	NR	Quiet
B	A	1	1	0	0
C	A	1	1	0	0
D	A	1	1	0	0
E	C	2	1	0	0
F	C	2	1	0	0

The final routing tree for Node A, after all the nodes exchange their sparse routing trees, is shown in Figure H-6 and Table H-3. Note that figure H-6 shows more than 2 paths between nodes G and A and H and A; however, the sparse routing tree table, which is the information actually transmitted, shows only two entries for nodes G and H. The pruning rules stated above have not been violated. They have been applied to the entries in the sparse routing table. The sparse routing graph is deduced from the table. Thus, quite a few redundant paths can be derived from the structure of the sparse routing table.

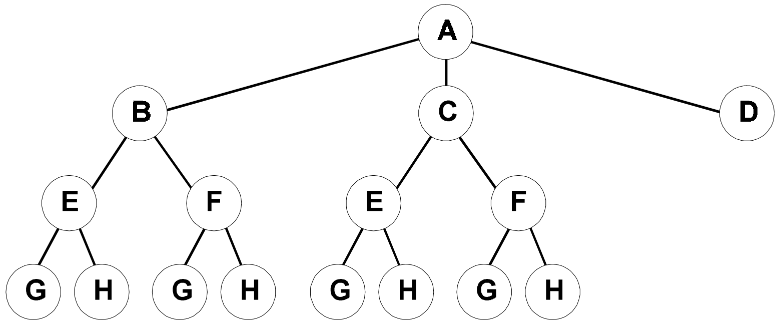


FIGURE H-6. Final routing tree for node A.

TABLE H-3. Final routing tree for node A.

Node Address	Node Predecessor	Hops	Cost	NR	Quiet
B	A	1	1	0	0
C	A	1	1	0	0
D	A	1	1	0	0
E	B	2	1	0	0
F	B	2	1	0	0
E	C	2	1	0	0
F	C	2	1	0	0
G	E	3	1	0	0
H	E	3	1	0	0
G	F	3	1	0	0
H	F	3	1	0	0

H.4.4. Rules for exchanging topology updates. Topology update packets are transmitted exclusively using a global multicast address.

H.4.4.1. Topology update triggers. Topology updates are triggered for node I by the following:

a. Node I detects a failed link and the link is to a node that is not a static node (link quality =7)

b. Node I detects a new or recovered link and the link is to a node that is not a static node (link quality =7)

c. Node I detects a change in the quality of a link - applicable only if link costs are used.

d. Node I receives a topology update from another node which modifies its sparse routing tree.

e. Node I changes its Quiet Mode status and wishes to announce this change.

f. Node I changes its relay capability status.

g. Node I receives a topology update request.

H.4.4.2. Sending topology update messages. Optimally, topology updates should be concatenated with other traffic for queuing by the link layer. Topology Update Messages are sent to the global multicast address using Type 1 Unnumbered Information Frames which are not acknowledged. The precedence of the Topology Update Message is user configurable.

The updates should be transmitted no more often than once every MIN_UPDATE_PER. MIN_UPDATE_PER is measured in minutes and is set by the network administrator when the nodes are configured. The network administrator can disable topology update transmission by setting MIN_UPDATE_PER to zero. Update packets are superseded by newer packets if they have not been queued at the link layer.

H.4.5. Non-relayers. In the Topology Update broadcast by non-relayers, the non-relayer indicates its status by setting the NR bit to one in its entry of the Topology Update message. Additionally, the non-relayer includes all one-hop, and only one-hop, neighbors (because relaying by this node is not permitted). Non-relayer nodes remain in the sparse routing trees; however, they must not have any subsequent branches. Their entries in the routing table must have the NR bit set to 1.

H.4.6. Quiet nodes. Nodes in the quiet state may appear in the sparse routing tables and in update packets with the QUIET bit set to 1; however, they must not have any subsequent branches in the routing tree. Nodes wishing to announce that they are entering quiet mode must add a separate entry into the sparse routing table and update packets with NODE ADDRESS and NODE PREDECESSOR set to their own address and the QUIET bit set to 1.

H.4.7. Topology update request messages. The Topology Update Request Message is triggered whenever there is a mismatch between the topology update ID received from a station and the value that had been stored previously. The Topology Update Request message may also be sent whenever a data link transmission is detected from a previously unknown neighbor. The Topology Update Request message uses a Type 1 Unnumbered Information frame which is not acknowledged and is addressed according to paragraphs 5.4.1.1.7, 5.4.1.1.9, and 5.4.1.3. The Topology Update Request message is addressed to specific stations at the Intranet layer and may be sent to the global multicast address at the data link layer. The precedence of the Topology Update Request Message is user-configurable. The Topology Update Request Message may be sent no more often than MIN_UPDATE_PER/2. This constant allows up to two requests to be sent to a node while the node is waiting for the MIN_UPDATE_PER timer to expire.

APPENDIX I

SOURCE DIRECTED RELAY

I.1. General.

I.1.1 Scope. This appendix describes a procedure for relaying packets across a CNR intranet using source directed routes. The intranet is defined as all processors and CNRs within a single transmission channel.

I.1.2 Application. This appendix is a mandatory part of MIL-STD-188-220. The information contained herein is intended for compliance.

I.2. Applicable Documents. None.

I.3. Problem overview.

Intranet relaying is required when nodes in an intranet need to communicate, but are not nearest neighbors capable of hearing one another's radio transmissions.

I.4. Procedure.

I.4.1 Forward routing. Source Directed Relay provides a simple non-dynamic procedure for relaying a packet from an originator to one or more destinations. The source must calculate the path through the intranet network to reach each destination. These paths are based on the topology and connectivity table. The specific source directed route for each destination must be encoded into the intranet header. If the routes for two or more destinations share common links along the paths, the two paths should be merged together. As a result of this, the resulting paths should not have any common nodes.

The address of successive relayers, destinations, and their associated status bytes are placed in the intranet header in order of progressing through the routing tree. Nodes which are one hop away and destinations only are placed into the Intranet Header first with their DES bit set to 1. The next entries into the Intranet Header are the relay paths which may include nodes which are relayers and destinations. Each relay path starting at the source is completed before another relay path with its origin at the source is begun. Within the status byte for each relayer the REL bit is set to 1 and S/D is set to 0. If the relayer is also a destination in addition to being a relayer, the DES bit is set to 1. If there are multiple destinations that are not relayers following a relayer, each of these destination addresses and their status bytes should be listed in the header after the relay node sequentially in the order of their appearance in the path. Within this group the extension bit within the destination/relay address field is not used. The last address can be determined from the Intranet header length. The last address in a group can be determined from the DIS field of the Destination/Relay Status Byte defined in 5.4.1.1.7.

All destinations in the relay path that are required to provide end-to-end intranet acknowledgments have set the ACK bit in their status bytes to 1. For all destinations, the

DISTANCE field is set to the number of hops between the originator and the ultimate destination host for the relay.

I.4.2 End-to-end acknowledgments. End-to-end Acknowledgments are formed by the *i*th final destination nodes upon receipt of an intranet header with **ACK** bit set in **DESTINATION STATUS BYTE** for the *i*th destination. The **MESSAGE ID** for the packet to be acknowledged is retained. The message type is set to 1. The path between the originator node and the *i*th destination is reversed. All intermediate destinations are removed. The path will contain one originator, one destination, and the relayers. The **DES** bit in the status bytes for all relayers is set to 0, indicating they perform relay only. No data is carried with an end-to-end acknowledgment packet; just the intranet header.

I.5. Examples. To illustrate Source Directed Relay procedures consider the sample network link diagram in Figure I-1 and final routing tree in Figure I-2. Table I-1 gives specific addresses for the nodes labeled A, B, C, D, E, F, G and H. To maintain consistency with other sections of MIL-STD-188-220, the least significant bit (LSB) is presented to the left of the figures in this appendix.

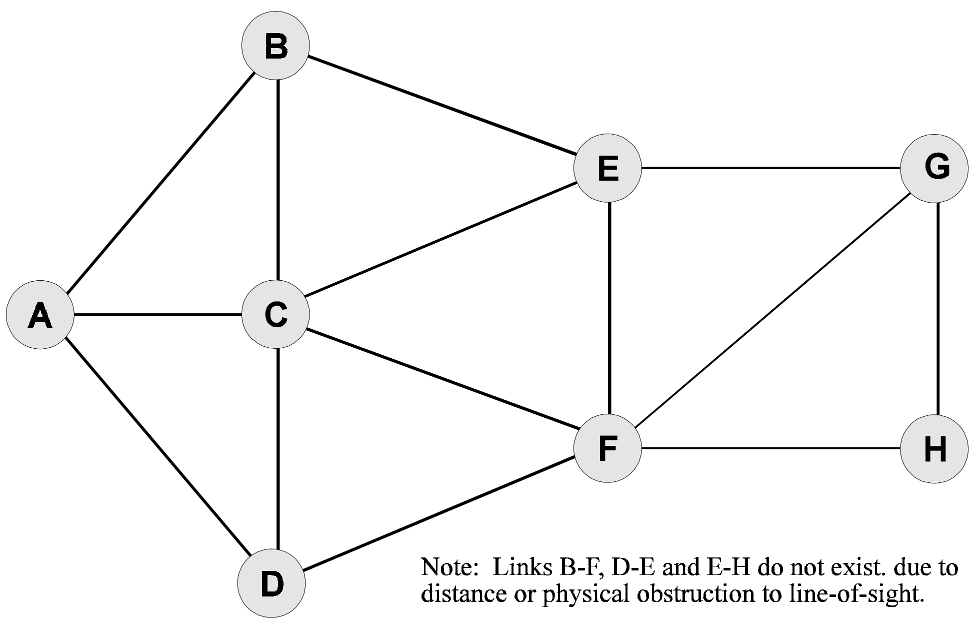


FIGURE I-1. Link diagram of a sample network.

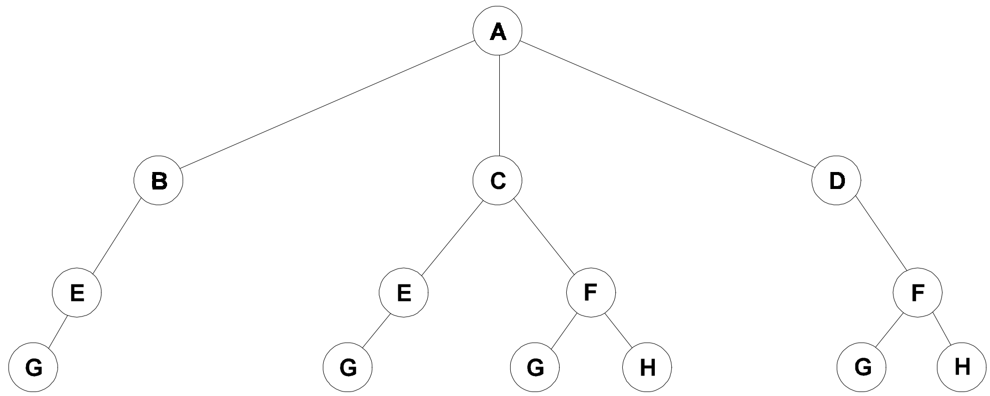


FIGURE I-2. Final routing tree for node A.

TABLE I-1. Sample node addresses

Node	LSB							MSB	Address
A	x	1	1	1	1	0	0	0	15
B	x	0	0	1	0	0	0	0	4
C	x	1	0	1	0	0	0	0	5
D	x	0	1	1	0	0	0	0	6
E	x	1	1	1	0	0	0	0	7
F	x	0	0	0	1	0	0	0	8
G	x	1	0	0	1	0	0	0	9
H	x	0	1	0	1	0	0	0	10

I.5.1. Example 1. Assume that node A has a packet bound for node G alone. Node A's Sparse Routing Tree provides the following potential paths to Node G: A-B-E-G, A-C-E-G, A-C-F-G and A-D-F-G. Assuming that all paths have the same quality and cost, any path may be selected by Node A. In this example, path A-B-E-G is selected.

The following values are assigned to the Intranet Header in example 1:

MESSAGE TYPE = 4 (IP Packet)
TYPE_OF_SERVICE = 00000000
MESSAGE ID = 1
MAX_HOP_COUNT = 3 (Distance from node A to node G)
ORIGINATOR ADDRESS = 15 (node A)
STATUS BYTE 1 = 10010000 (DIS=1, REL=Yes, DES=No, ACK=No)
DESTINATION 1 = 4 (node B)
STATUS BYTE 2 = 01010000 (DIS=2, REL=Yes, DES=No, ACK=No)
DESTINATION 2 = 7 (node E)

STATUS BYTE 3 = 11000010 (DIS=3, REL=No, DES=Yes, ACK=No)
DESTINATION 3 = 9 (node G)
HEADER LENGTH = 12 octets

Figure I-3 shows the complete Intranet Header for example 1. Note that the LSB in all destination addresses is 0 except for the last destination address (node G).

LSB	*1*	*2*	*3*	*4*	*5*	*6*	*7* *MSB*
VERSION NUMBER				MESSAGE TYPE			
0	0	0	0	0	0	1	0
INTRANET HEADER LENGTH							
0	0	1	1	0	0	0	0
TYPE OF SERVICE							
0	0	0	0	0	0	0	0
MESSAGE IDENTIFICATION NUMBER							
1	0	0	0	0	0	0	0
MAX HOP COUNT				SPARE			
1	1	0	0	0	0	0	0
ORIGINATOR ADDRESS							
0	1	1	1	1	0	0	0
DESTINATION/RELAY STATUS BYTE 1							
1	0	0	1	0	0	0	0
DESTINATION/RELAY ADDRESS 1							
0	0	0	1	0	0	0	0
DESTINATION/RELAY STATUS BYTE 2							
0	1	0	1	0	0	0	0
DESTINATION/RELAY ADDRESS 2							
0	1	1	1	0	0	0	0
DESTINATION/RELAY STATUS BYTE 3							
1	1	0	0	0	0	1	0
DESTINATION/RELAY ADDRESS 3							
1	1	0	0	1	0	0	0

FIGURE I-3. Example 1 Intranet header.

I.5.2. Example 2. Assume that node A has a packet bound for nodes G and H. Node A's Sparse Routing Tree provides the following potential paths to nodes G and H: A-B-E-G, A-C-E-G, A-C-F-G, A-C-F-H, A-D-F-G, and A-D-F-H. Of these potential paths, the most economical choices are those that use node F for relaying: A-C-F-G, A-D-F-G, A-C-F-H, and A-D-F-H. Although paths A-B-E-G and A-C-E-G are viable paths to node G, they would unnecessarily increase processing at nodes B and E, and would increase the size of the Intranet Header in this example. In this example the selected paths are A-C-F-G and A-C-F-H.

The following values are assigned to the Intranet Header in example 2:

MESSAGE TYPE = 4 (IP Packet)
TYPE_OF_SERVICE = 00000000
MESSAGE ID = 2
MAX_HOP_COUNT = 3 (Distance from node A to nodes G and H)
ORIGINATOR ADDRESS = 15 (node A)
STATUS BYTE 1 = 10010000 (DIS=1, REL=Yes, DES=No, ACK=No)
DESTINATION 1 = 4 (node C)
STATUS BYTE 2 = 01010000 (DIS=2, REL=Yes, DES=No, ACK=No)
DESTINATION 2 = 8 (node F)
STATUS BYTE 3 = 11000010 (DIS=3, REL=No, DES=Yes, ACK=No)
DESTINATION 3 = 9 (node G)
STATUS BYTE 4 = 11000010 (DIS=3, REL=No, DES=Yes, ACK=No)
DESTINATION 4 = 10 (node H)
HEADER LENGTH = 14 octets

Figure I-4 shows the complete Intranet Header for example 2. Note that the LSB in all destination addresses is 0 except for the last destination address (node H).

0 LSB	*1*	*2*	*3*	*4*	*5*	*6*	*7 MSB*
VERSION NUMBER				MESSAGE TYPE			
0	0	0	0	0	0	1	0
INTRANET HEADER LENGTH							
0	1	1	1	0	0	0	0
TYPE OF SERVICE							
0	0	0	0	0	0	0	0
MESSAGE IDENTIFICATION NUMBER							
0	1	0	0	0	0	0	0
MAX HOP COUNT				SPARE			
1	1	0	0	0	0	0	0
ORIGINATOR ADDRESS							
0	1	1	1	1	0	0	0
DESTINATION/RELAY STATUS BYTE 1							
1	0	0	1	0	0	0	0
DESTINATION/RELAY ADDRESS 1							
0	0	0	1	0	0	0	0
DESTINATION/RELAY STATUS BYTE 2							
0	1	0	1	0	0	0	0
DESTINATION/RELAY ADDRESS 2							
0	0	0	0	1	0	0	0
DESTINATION/RELAY STATUS BYTE 3							
1	1	0	0	0	0	1	0
DESTINATION/RELAY ADDRESS 3							
0	1	0	0	1	0	0	0
DESTINATION/RELAY STATUS BYTE 4							
1	1	0	0	0	0	1	0
DESTINATION/RELAY ADDRESS 4							
1	0	1	0	1	0	0	0

FIGURE I-4. Example 2 Intranet header.

I.5.3. Example 3. In the third example, node A wishes to deliver a packet to nodes D, E, F, G and H. In this case node A again would select the most economical path to each destination, taking into consideration the impacts on network traffic and Intranet header size. Table I-2 lists the potential and selected paths from node A to each of the intended destinations.

A similar process would be used to select economical paths to relay nodes, such as node C. The shortest path to the most distant nodes G and H are reviewed to determine whether the relay nodes C and F are also destinations. Note that node F is both a destination and a relay while node C is a relay node only.

Destination Node	Potential Paths	Selected Path
D	A-D	A-D
E	A-B-E A-C-E	A-C-E
F	A-C-F A-D-F	A-C-F
G	A-B-E-G A-C-E-G A-C-F-G A-D-F-G	A-C-F-G
H	A-C-F-H A-D-F-H	A-C-F-H

TABLE I-2. Paths used in example 3.

The following values are assigned to the Intranet Header in example 3:

MESSAGE TYPE = 4 (IP Packet)
TYPE_OF_SERVICE = 00000000
MESSAGE ID = 3
MAX_HOP_COUNT = 3 (Distance from node A to nodes G and H)
ORIGINATOR ADDRESS = 15 (node A)
STATUS BYTE 1 = 10000010 (DIS=1, REL=No, DES=Yes, ACK=No)
DESTINATION 1 = 6 (node D)
STATUS BYTE 2 = 10010000 (DIS=1, REL=Yes, DES=No, ACK=No)
DESTINATION 2 = 5 (node C)
STATUS BYTE 3 = 01000010 (DIS=2, REL=No, DES=Yes, ACK=No)
DESTINATION 3 = 7 (node E)
STATUS BYTE 4 = 01010010 (DIS=2, REL=Yes, DES=Yes, ACK=No)
DESTINATION 4 = 8 (node F)
STATUS BYTE 5 = 11000010 (DIS=3, REL=No, DES=Yes, ACK=No)
DESTINATION 5 = 9 (node G)
STATUS BYTE 6 = 11000010 (DIS=3, REL=No, DES=Yes, ACK=No)
DESTINATION 6 = 10 (node H)
HEADER LENGTH = 18 octets

Figure I-5 shows the complete Intranet Header for example 3. Note that the LSB in all destination addresses is 0 except for the last destination address (node H).

0 *LSB*	*1*	*2*	*3*	*4*	*5*	*6*	*7* *MSB*
VERSION NUMBER				MESSAGE TYPE			
0	0	0	0	0	0	1	0
INTRANET HEADER LENGTH							
0	1	0	0	1	0	0	0
TYPE OF SERVICE							
0	0	0	0	0	0	0	0
MESSAGE IDENTIFICATION NUMBER							
1	1	0	0	0	0	0	0
MAX HOP COUNT				SPARE			
1	1	0	0	0	0	0	0
ORIGINATOR ADDRESS							
0	1	1	1	1	0	0	0
DESTINATION/RELAY STATUS BYTE 1							
1	0	0	0	0	0	1	0
DESTINATION/RELAY ADDRESS 1							
0	0	1	1	0	0	0	0
DESTINATION/RELAY STATUS BYTE 2							
1	0	0	1	0	0	0	0
DESTINATION/RELAY ADDRESS 2							
0	1	0	1	0	0	0	0
DESTINATION/RELAY STATUS BYTE 3							
0	1	0	0	0	0	1	0
DESTINATION/RELAY ADDRESS 3							
0	1	1	1	0	0	0	0
DESTINATION/RELAY STATUS BYTE 4							
0	1	0	1	0	0	1	0
DESTINATION/RELAY ADDRESS 4							
0	0	0	0	1	0	0	0
DESTINATION/RELAY STATUS BYTE 5							
1	1	0	0	0	0	1	0
DESTINATION/RELAY ADDRESS 5							
0	1	0	0	1	0	0	0
DESTINATION/RELAY STATUS BYTE 6							
1	1	0	0	0	0	1	0
DESTINATION/RELAY ADDRESS 6							
1	0	1	0	1	0	0	0

FIGURE I-5. Example 3 intranet header created by node A (originator).

I.5.4. Relay processing. Although the separate examples 1,2,3 all have diverse paths, they would all require the same number data link information frames for delivery (one). The UI, I, or DIA frame would be transmitted to each destination simultaneously. Addressed destinations would perform the required data link layer processing described in 5.3 and pass the information field of the frame to the Intranet layer for further processing.

The Intranet header is scanned for the node's data link layer address. When found, the previous octet - the Destination/Relay Status byte - is inspected. If the Relay bit is not set and the destination bit is set, the data portion following the Intranet header is passed to the next higher protocol layer for further processing. If the Relay bit is set, Relay processing is required. If both the Relay bit and the Destination bit are set, Relay processing is performed before the passing data portion of the frame to the next higher protocol layer for further processing. Relay processing involves the following steps:

a. Scan forward until the relayer node sees its own address.

b. Scan toward the end of the header looking for all nodes whose DES bit is set and whose distance is one hop greater than your own. Terminate the scan when a distance less than or equal to your own or the end of the header is found. Save the addresses.

c. While scanning forward until a hop distance less than or equal to your own is found, find all relay addresses that are one hop away from your address and save these addresses.

d. Remove all duplicate saved addresses and pass the remaining addresses to the data link layer to form a multi-addressed information frame containing the Intranet header and data.

The following sections discuss the relay processing at each of the downstream relayers in Example 3. There are two options when filling out the Intranet Header Address Field at the relay nodes. The relay nodes may copy the Address Field and place it into the relay packet intact or they may delete the addresses which have no impact on forwarding or return of a network layer acknowledgment. If the implementor chooses to leave the address field intact, the address field in Figure I-5 is used at every relayer. If the implementor chooses to compress the address field to save transmitted bytes, the following paragraphs dictate the method for compression. There is no interoperability problem regardless of which of these two methods are implemented.

I.5.4.1 Relay processing at node C. Node C is a relay node, but not a destination node. Node C is responsible for relaying the information to nodes E, F, G and H. Node C assigns the following values to the Intranet Header in example 3:

MESSAGE TYPE = 4 (IP Packet)
TYPE_OF_SERVICE = 00000000
MESSAGE ID = 3
MAX_HOP_COUNT = 2 (Original MAX_HOP_COUNT - 1)
ORIGINATOR ADDRESS = 15 (node A)
STATUS BYTE 1 = 10010000 (DIS=1, REL=Yes, Des=No, ACK=No)
DESTINATION 1 = 5 (node C)
STATUS BYTE 2 = 01000010 (DIS=2, REL=No, DES=Yes, ACK=No)
DESTINATION 2 = 7 (node E)

STATUS BYTE 3 = 01010010 (DIS=2, REL=Yes, DES=Yes, ACK=No)
DESTINATION 3 = 8 (node F)
STATUS BYTE 4 = 11000010 (DIS=3, REL=No, DES=Yes, ACK=No)
DESTINATION 4 = 9 (node G)
STATUS BYTE 5 = 11000010 (DIS=3, REL=No, DES=Yes, ACK=No)
DESTINATION 5 = 10 (node H)
HEADER LENGTH = 16 octets

Figure I-6 shows the complete Intranet Header created by Node C.

I.5.4.2. Relay processing at node F. Node F is both a destination and a relayer with relay responsibilities to nodes G and H. Node F assigns the following values to the Intranet Header in example 3:

MESSAGE TYPE = 4 (IP Packet)
TYPE_OF_SERVICE = 00000000
MESSAGE ID = 3
MAX_HOP_COUNT = 1 (Received MAX_HOP_COUNT - 1)
ORIGINATOR ADDRESS = 15 (node A)
STATUS BYTE 1 = 10010000 (DIS=1, REL=Yes, DES=No, ACK=No)
DESTINATION 1 = 5 (node C)
STATUS BYTE 2 = 01010010 (DIS=2, REL=Yes, DES=Yes, ACK=No)
STATUS BYTE 2 = 8 (node F)
STATUS BYTE 3 = 11000010 (DIS=3, REL=No, DES=Yes, ACK=No)
DESTINATION 3 = 9 (node G)
STATUS BYTE 4 = 11000010 (DIS=3, REL=No, DES=Yes, ACK=No)
DESTINATION 4 = 10 (node H)
HEADER LENGTH = 14 octets

Figure I-7 shows the complete Intranet Header created by Node F.

LSB	*1*	*2*	*3*	*4*	*5*	*6*	*7* *MSB*
VERSION NUMBER				MESSAGE TYPE			
0	0	0	0	0	0	1	0
INTRANET HEADER LENGTH							
0	0	0	0	1	0	0	0
TYPE OF SERVICE							
0	0	0	0	0	0	0	0
MESSAGE IDENTIFICATION NUMBER							
1	1	0	0	0	0	0	0
MAX HOP COUNT				SPARE			
0	1	0	0	0	0	0	0
ORIGINATOR ADDRESS							
0	1	1	1	1	0	0	0
DESTINATION/RELAY STATUS BYTE 1							
1	0	0	1	0	0	0	0
DESTINATION/RELAY ADDRESS 1							
0	1	0	1	0	0	0	0
DESTINATION/RELAY STATUS BYTE 2							
0	1	0	0	0	0	1	0
DESTINATION/RELAY ADDRESS 2							
0	1	1	1	0	0	0	0
DESTINATION/RELAY STATUS BYTE 3							
0	1	0	1	0	0	1	0
DESTINATION/RELAY ADDRESS 3							
0	0	0	0	1	0	0	0
DESTINATION/RELAY STATUS BYTE 4							
1	1	0	0	0	0	1	0
DESTINATION/RELAY ADDRESS 4							
0	1	0	0	1	0	0	0
DESTINATION/RELAY STATUS BYTE 5							
1	1	0	0	0	0	1	0
DESTINATION/RELAY ADDRESS 5							
1	0	1	0	1	0	0	0

FIGURE I-6. Example 3 Intranet header for node C (relay node).

0 LSB	*1*	*2*	*3*	*4*	*5*	*6*	*7 MSB*
VERSION NUMBER				MESSAGE TYPE			
0	0	0	0	0	0	1	0
INTRANET HEADER LENGTH							
0	1	0	1	0	0	0	0
TYPE OF SERVICE							
0	0	0	0	0	0	0	0
MESSAGE IDENTIFICATION NUMBER							
1	1	0	0	0	0	0	0
MAX HOP COUNT				SPARE			
1	0	0	0	0	0	0	0
ORIGINATOR ADDRESS							
0	1	1	1	1	0	0	0
DESTINATION/RELAY STATUS BYTE 1							
1	0	0	1	0	0	0	0
DESTINATION/RELAY ADDRESS 1							
0	1	0	1	0	0	0	0
DESTINATION/RELAY STATUS BYTE 2							
0	1	0	1	0	0	1	0
DESTINATION/RELAY ADDRESS 2							
0	0	0	0	1	0	0	0
DESTINATION/RELAY STATUS BYTE 3							
1	1	0	0	0	0	1	0
DESTINATION/RELAY ADDRESS 3							
0	1	0	0	1	0	0	0
DESTINATION/RELAY STATUS BYTE 4							
1	1	0	0	0	0	1	0
DESTINATION/RELAY ADDRESS 4							
1	0	1	0	1	0	0	0

FIGURE I-7. Example 3 intranet header created by node F (relay and destination nodes).

APPENDIX J

ROBUST COMMUNICATIONS PROTOCOL

J.1. General.

J.1.1. Scope. This Appendix describes the interoperability and technical requirements for the robust communications protocol for DMTD and interfacing C4I systems (DTEs). This Appendix applies only to HAVEQUICK II compatible systems that require interoperability with radios that do not have data buffering or synchronization capability.

J.1.2 Application. This Appendix is a mandatory part of MIL-STD-188-220. The information contained herein is intended for compliance.

J.2. Applicable Documents. This section is not applicable to this Appendix.

J.3. Introduction. This physical layer protocol provides the additional processing to aid the transfer of secure and non-secure digital data when concatenated with the link processing of the MIL-STD-188-220 protocol. The additional processing of this protocol allows for a higher level protocol with an error correcting capability equal to rate 1/2 Golay to transfer a burst of data containing up to 67,200 data symbols with better than 90% probability of success in a single transmission, this being over an active HAVEQUICK II compatible link with a random bit error rate of 0.1 or less. The second goal of this physical protocol is for the required performance to be achieved entirely in software using current systems with modest processing capability.

J.3.1 Physical protocol components. Three individually selectable processes are used to meet the performance requirement. The first is the application of rate 1/3 convolutional coding to combat high random bit error rates. The second is a provision for data scrambling. Scrambling at the physical layer is implemented simply as the multiplication of the transmit data with a pseudo random bit pattern. The third is a packetizing scheme that allows for the re-transmission of the data that was lost due to an HAVEQUICK II compatible frequency hop. The re-transmission is performed, and data recovered within the data burst and the data interruption is transparent to the higher level protocol. This packetizing scheme has been dubbed the Multi-Dwell protocol because it was formulated to allow a message to be transmitted over multiple HAVEQUICK II compatible hop dwells.

J.3.2 Optional rate 1/3 convolutional coding. The transmitting convolutional encoder generates three output bits for each input information bit. Figure J-1 shows an example of the encoding process for a constraint length (K) of 3. The encoder consists of a shift register equal in length to the constraint length. The data to encode is shifted from left to right one bit at a time. After each shift, three output bits are generated using the G1, G2, and G3 polynomials. The three encoded output bits are generated in the G1, G2, and G3 order. The G2 output shall be inverted to provide some data scrambling capability. The convolutional encoding shift register is initialized to a state of zero when a transmission is requested. The first output bits are generated when the shift register contains the first upper layer bit to transmit, followed by all zeros. Upon detection of the robust synchronization pattern, the Viterbi decoder is initialized to make use of the knowledge of the initial encoder shift register state.

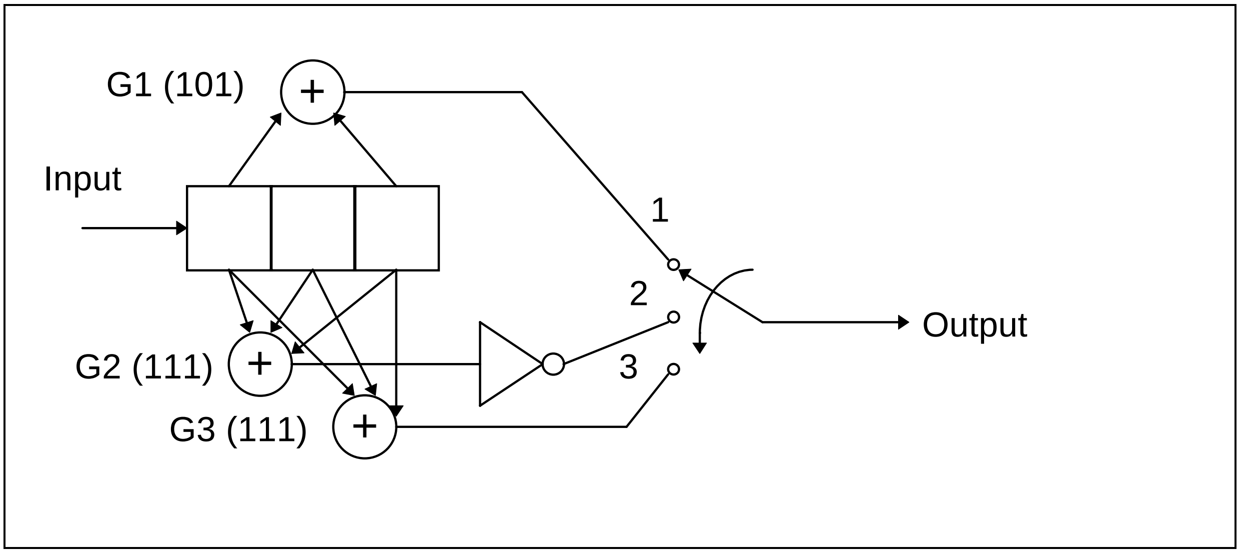


FIGURE J-1. Convolutional encoder with inverted G2 K=3.

Table J-1 lists the generator polynomials used for the three specified constraint lengths. The most significant bits of the octal representation of each polynomial are used for each polynomial.

TABLE J-1. Convolutional coding generator polynomials (octal).

Constraint Length	G1	G2	G3
3	5	7	7
5	52	66	76
7	554	624	764

Figure J-2 shows the relative error correcting capability of rate 1/3 convolutional coding in a random error environment using the Viterbi decoding algorithm with hard decisions. The performance was achieved using a trace back buffer length of 16, 32, and 64 for constraint lengths 3, 5, and 7 respectively. If the demodulator and decoder are components of the same subsystem, soft decision information from the demodulator can be used to further enhance the performance.

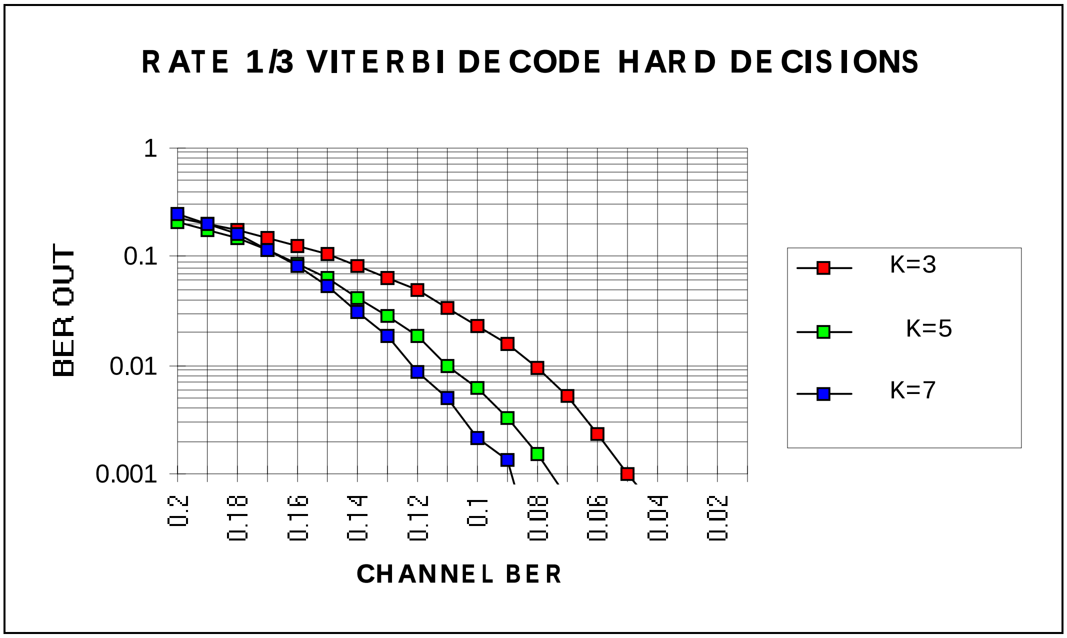


FIGURE J-2. Rate 1/3 convolutional coding performance for constraint lengths 3, 5, and 7.

J.3.3 Optional data scrambling. Physical layer data scrambling shall use the pseudo random bit generator specified in CCITT V.33 Annex A. The shift register shall be initialized to all zeros before the first bit of data is scrambled on transmission. On data reception, the descrambler shift register shall be initialized to zero before the first received data bit is descrambled. Figure J-3 shows the structure of the data scrambler and descrambler.

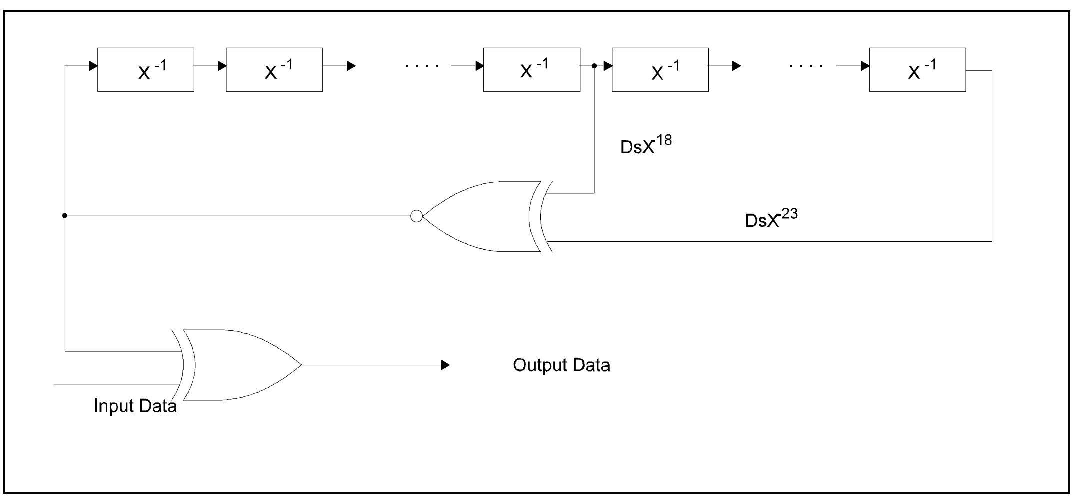


FIGURE J-3. Data scrambler structure.

J.3.4 Optional robust multi-dwell.

J.3.4.1 Multi-dwell packet format. When the HAVEQUICK II compatible radio is in active mode, multi-dwell packetizing shall be enabled. The multi-dwell packetizing described in this appendix assumes a physical level bit rate of 16 kbps. The format of each multi-dwell packet is shown in Figure J-4. Each packet consists of a start of packet (SOP) pattern and a segment counter followed by 6, 11 or 13 64-bit data segments.

J.3.4.2 Multi-dwell SOP field. The SOP pattern is a 32-bit (Figure J-5) or 64-bit (Figure J-6) pattern used for multi-dwell packet detection. The maximum number of bits in error should be set to match the bit error rate environment. For normal operation, it is recommended that the maximum number of bits in error be set to 13 for a 64-bit pattern, and to 3 for a 32-bit pattern. The length of the SOP pattern shall be determined by bits two, three and four of the robust frame format.

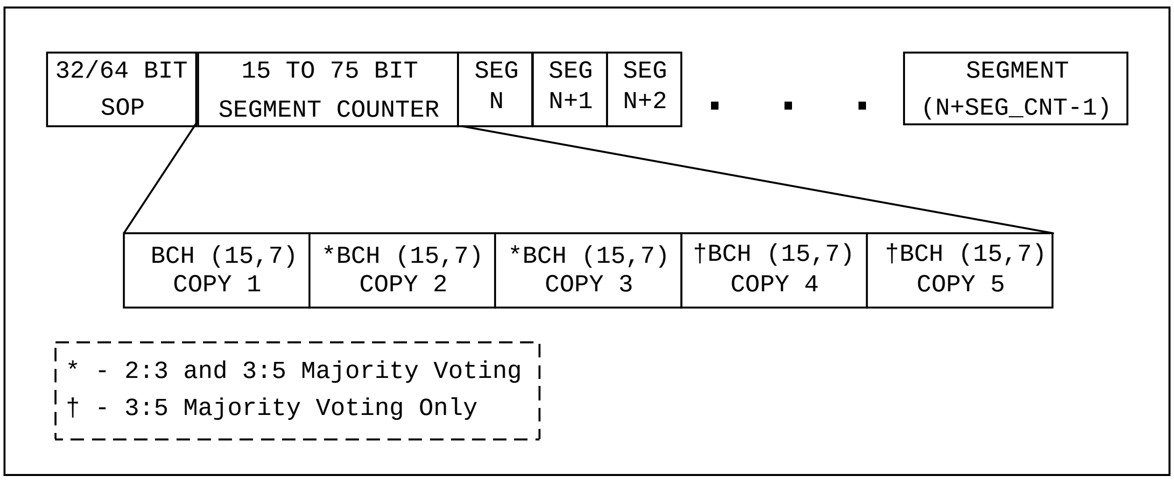


FIGURE J-4. Multi-dwell packet.

LSB	MSB
1010010011100010111100101000011010010000011111101010110011011101	

FIGURE J-5. Multi-dwell 64-bit SOP pattern.

LSB	MSB
00000011100100001001001110101110	

FIGURE J-6. Multi-dwell 32-bit SOP pattern.

J.3.4.3 Multi-dwell segment count field. The segment counter is a modulo 64 count of the first segment in the packet. The six required bits shall be encoded as 1, 3, or 5 BCH (15,7) codewords depending on bits 2, 3 and 4 of the robust frame format. The six-bit segment counter shall occupy the 6 least significant bits of the seven-bit BCH data field. The most significant bit of the data field shall be used as an end of frame flag which, when set, indicates that data transmission is complete. A multi-dwell packet marked with an end of frame flag shall contain only the SOP pattern and the segment count field used to make the segment number of the last non-fill data segment transmitted.

J.3.4.4 Multi-dwell data segments. Each multi-dwell packet shall contain 6, 11 or 13 consecutive 64-bit data segments. Unless a channel interruption is detected during the transmission of the packet, each data segment shall contain the next 64 bits supplied by the data link layer for transmission. The last multi-dwell packet shall contain pad bits and segments as necessary to complete the packet. The transmitted pad data shall be an alternating one, zero sequence.

J.3.4.5 Multi-dwell hop detection. The physical layer shall have the means of detecting or predicting communications link outages.

J.3.4.6 Multi-dwell transmit processing. Data received from the data link layer for transmission shall be broken into 64 bit segments for transmission. The data shall be packetized as stated in J.3.4.1. Packets shall be transmitted consecutively with the packet count subfield containing the count, modulo 64, of the first segment in the packet until a communications link outage is detected, at which time, the remainder of the data segments in the currently transmitted packet shall be filled with an alternating one/zero pattern. If the configurable hop recovery time (HRT) is greater than the time remaining to complete the transmission of the current packet, the alternating one/zero sequence shall be extended to the end of the HRT period. If a hop is detected during the multi-dwell packet synchronization field, or during the transmission of the first two segments, the entire packet shall be retransmitted. The first multi-dwell packet transmitted in a frame shall not contain the multi-dwell synchronization field. It is assumed that the segment count of the first packet is zero.

J.3.4.6.1 Hop data recovery time period. A configurable variable called the hop recovery time (HRT) shall be used to determine if the fill data transmitted following a hop must be extended to ensure that the following multi-dwell synchronization field can be received. The HRT is defined as the time period from the beginning of the transmitting radio frequency synthesizer frequency hop to the time that the bit synchronizer connected to the receiving radio can reliably demodulate data. Because different hop detection/ prediction methods flag the hop at different times relative to the beginning of the transmitting radio frequency synthesizer frequency slew, the configured HRT shall be internally adjusted to insure that different DTEs in a network can all use the same configurable HRT.

J.3.4.6.2 Data transmitted after a hop. The multi-dwell packet transmitted directly following a communications link outage shall retransmit data starting with the 64-bit segment preceding the segment that was being transmitted when the hop was detected.

J.3.4.6.3 Termination of transmission. After the final packet of the frame is transmitted, without a hop detected during a data segment containing actual data (not fill data), data transmission shall be terminated. To prevent receive delays caused by the receiver not being able to determine that the last data segment has been received, an optional truncated multi-dwell packet shall be sent with the end of frame flag set. The segment count associated with the end-of-frame flag shall mark the first non-fill data segment transmitted.

J.3.4.7 Multi-dwell receive processing. If the multi-dwell flag was set in the robust synchronization field, the receiver shall buffer the multi-dwell data packet. The segment count for the first multi-dwell packet in a frame shall be assumed to be 0. After the last packet bit is received, the receiver shall open the SOP correlator window. When the SOP pattern is recognized, the segment count is decoded using the combination of majority and BCH decoding specified in the robust synchronization field. After each new segment count is decoded, the buffered data for data segments lower in count than the new segment count are passed on to the next higher layer as received bits. The segments of the newly received packet are then buffered and held until it is verified that the buffered segments will not be re-transmitted.

J.3.4.7.1 Receive end-of-frame detection. The data remaining in the multi-dwell receive data buffer shall be provided to the higher level protocol when an end-of-frame condition is detected. The end-of-frame condition may be determined by the data demodulator, the optional multi-dwell end-of-frame flag, or by a message from the higher level protocol indicating that message reception is complete.

J.3.4.7.2 Optional soft decision information. When there is a very high link BER, a SOP pattern may not be recognized or the segment count may not be correctable. If fewer than three consecutive segment counts cannot be corrected the correct number of bits shall be supplied to the upper level protocol as to not cause a bit slip, and consequently, the loss of the remaining data in the frame. If convolutional coding is used with multi-dwell, it is suggested that soft decision information is supplied indicating the low quality of the received data resulting from a missed SOP pattern or an unrecoverable segment count.

J.3.4.8 Multi-dwell majority logic overhead choice. The choice of the amount of multi-dwell majority voting (MV) overhead is dependent on the expected link BER. Table J-2 gives an estimate of the maximum random BER supported for a 90% probability of passing a single frame of length 1536 bits, 7680 bits, and 67,200 bits with no errors introduced due to multi-dwell processing.

TABLE J-2. Maximum supported BER.

Segment Count MV	1536	7680	67,200
1 out of 1	0.055	0.03	0.016
2 out of 3	0.14	0.11	0.07
3 out of 5	0.2	0.14	0.12

J.3.4.9 Multi-dwell overhead. The multi-dwell protocol introduces an overhead that shall be considered in the network timing calculations. The overhead is a function of the radio hop rate, the multi-dwell segment count majority voting choice, and the message length. Table J-3 gives the equation to calculate the actual worst case realized data rate for each hop rate and majority logic combination. The numbers in table J-3 were run with a hop recovery time of 15.625 ms, a maximum radio timing drift over a 1/2 hour period, an instantaneous data rate of 16000 bits/second. The actual efficiency will depend upon the exact implementation, therefore the numbers in Table J-3 should be used as a guide only. The six-segment multi-dwell packet shall be used for protocol acknowledgments and other single TDC block messages. The calculated realized data rate shall be used for the bit rate of all data encapsulated by the multi-dwell protocol.

TABLE J-3. Multi-dwell overhead.

HOP RATE	**Multi-dwell overhead calculation**			
	MV 1:1, 11 segments	**MV 2:3, 13 segments**	**MV 3:5, 13 segments**	**MV 3:5, 6 segments**
0	$R/((0.3*10^{(-L*.00003)})+1.06)$	$R/((0.3*10^{(-L*.00003)})+1.16)$	$R/((0.2*10^{(-L*.00003)})+1.17)$	$R/((0.1*10^{(-L*.00003)})+1.36)$
1	$R/((0.6*10^{(-L*.00003)})+1.10)$	$R/((0.6*10^{(-L*.00003)})+1.21)$	$R/((.55*10^{(-L*.00003)})+1.23)$	$R/((0.3*10^{(-L*.00003)})+1.40)$
2	$R/((0.5*10^{(-L*.00003)})+1.15)$	$R/((0.5*10^{(-L*.00003)})+1.27)$	$R/((0.7*10^{(-L*.00003)})+1.30)$	$R/((0.4*10^{(-L*.00005)})+1.48)$
3	$R/((0.5*10^{(-L*.00003)})+1.20)$	$R/((0.4*10^{(-L*.00002)})+1.36)$	$R/((0.8*10^{(-L*.00003)})+1.29)$	$R/((0.2*10^{(-L*.00003)})+1.56)$
4	R/(1.45)	R/(1.51)	$R/((0.7*10^{(-L*.00002)})+1.46)$	$R/((.07*10^{(-L*.00002)})+1.85)$
ALL	R/(1.72)	R/(1.72)	R/(1.96)	R/(2.27)

R = the instantaneous data rate
L = the number of bits to be transmitted

J.3.4.9.1 Terminals lacking hop detection. The ALL case in Table J-3 is to show the efficiency of the multi-dwell protocol in systems where the hop cannot be detected due to hardware or software limitations. Since there is no hop timing information available, the DTE shall assume that the radio will hop at every possible time slot. In these systems, it is assumed that timing synchronization with the radio will be made by the detection of the falling edge of the radio delayed push to talk (DPTT) signal provided by the HAVEQUICK II compatible radio.

J.3.5 Robust communications protocol network timing. The use of the robust communications protocol requires modification to some of the Appendix C type 1 network timing equations. The bit rate, transmit delays, and receive processing delays are modified by the robust protocol. For purposes of robust network timing, two system bit rates are defined. The first is the channel bit rate which is represented as n_c. The second is the data link bit rate which is represented as n_l. As an example, if rate 1/3 convolutional coding is applied at the physical layer and the channel bit rate is 16 kbps, the link bit rate would be 5.33 kbps. In this example, an external cryptographic device would transmit the MI field at n_c Hz and an internal cryptographic device would transmit the MI field at n_l Hz. The multi-dwell reduction of n_l is not deterministic but is bounded. The average multi-dwell n_l is a function of the multi-dwell packet format, the timing of the DTE transmit request in relation to the radio TRANSEC timing, and the number of bits to be transmitted. The following Type 1 network access control subfunctions are specified in Appendix C:

a. network busy sensing

b. response hold delay (RHD)

c. timeout period (TP)

d. network access delay (NAD)

The following subparagraphs address required modifications to network timing equations associated with these subfunctions as a result of using the robust communications protocol.

J.3.5.1 Net busy sensing. Because net busy sensing is performed at the physical level, there are no modifications to the net busy sensing timing or methods when using the robust communications protocol.

J.3.5.2 Response hold delay. The additional transmission time required for the robust synchronization field and n_l bit rate reductions impact the response transmission time parameter, (S), contained in the response hold delay timing equation, RHD_0. Also additional receive processing delays impact the internal DTE timing calculations. The RHD_0 is calculated as follows:

$$RHD_O = EPRE + PHASING + S + ELAG + TURN + TOL$$

J.3.5.2.1 Response transmission time (S). The response transmission time is changed by the robust protocol. A Type 1 Response PDU from the data link layer consists of the 64-bit message synchronization field, the 75-bit robust frame format, an optional embedded COMSEC MI field, the 168-bit word count and Transmission Header TDC block, and 384, 168, or 80 bits of acknowledgment data depending on the selectable use of EDC and TDC. The 64-bit message synchronization field and the 75-bit robust frame format are transmitted at the channel bit rate (n_c). The remaining components are transmitted at the link data rate (n_l).

J.3.5.2.1.1 Multi-dwell response. Where multi-dwell is used to send the original message at a channel bit rate, n_c, of 16 Kbps, all responses, i.e. Type 1 acknowledgments, except for a secure external crypto transmission, are short enough that a multi-dwell transmission is not required. A multi-dwell transmission is required when using an external crypto because the data may be interrupted by a frequency hop. Table J-4 gives the maximum number of bits that will be transmitted at the channel bit rate (n_c) for the link data sizes and multi-dwell SOP majority logic choices. These numbers are for the HOP ALL case, which is the worst case, and for the highest operational hop rate, hop rate 3. The 139 robust protocol header bits are included in Table J-4. The numbers in Table J-4 do not include the "wasted time" shown in Figure J-7.

TABLE J-4. Multi-dwell external crypto response transmission time
HRT= 15.6 ms, mode 3, (MV3:5, 6 segments/packet)

LINK BITS	**FEC 1/3 Hop Rate 3**	**FEC 1/3 Hop All**	**No FEC Hop Rate 3**	**No FEC Hop All**
312	$2139/n_c$	$3139/n_c$	$662/n_c$	$662/n_c$
392	$2139/n_c$	$4143/n_c$	$1185/n_c$	$1185/n_c$
616	$3139/n_c$	$5189/n_c$	$1185/n_c$	$1708/n_c$

NOTES: Table J-4 is optimized for the AN/ARC-164.
Robust Mode 3 is used for response PDUs. The choices are FEC or no FEC.
LINK BITS = the number of bits sent to the physical layer in a response PDU.
If Delay PTT is not supported by all radios in the network, then only columns marked "Hop All" apply.
Columns marked "Hop All" are required. Columns marked "Hope Rate 3" are optional.

J.3.5.2.1.2 Non multi-dwell response. Where an external crypto device is not used and n_c is 16 Kbps, the long dwell time will contain the entire response. Where an external crypto device is used, convolutional encoding is not used and the n_c is 16 Kbps, the crypto preamble will be contained within the long dwell time and the response will be contained within the first minimum dwell period following the long dwell time.

J.3.5.2.2 Response transmission example. Figure J-7 shows an example of the timing of an acknowledgment when an external cryptographic device is used with the HAVEQUICK II radio. The falling edge of the DPTT signal marks the beginning of a long hop dwell that is long enough to contain the crypto preamble time.

If an external crypto device was not used, this long dwell time would contain the entire acknowledgment. After the crypto has finished transmitting the MI field, the transmitting DTE begins to supply data for transmission. Typically, the COMSEC bit synchronization time is not very accurate and may be long enough to push the MI field to the end of the guaranteed long dwell time. For this reason, the DTE shall wait to start data transmission on the first hop dwell following the long guaranteed dwell. The end of the guaranteed hop dwell is marked by the possible hop label. The first bit of the robust SOM pattern is transmitted after the configured hop recovery time (HRT). During the transmission of the response, one or more hops may occur which will vary the transmission time of the acknowledgment. When the response transmission is complete, the DTE de-asserts the transmit request signal. The radio will de-assert DPTT after a variable delay (ET_1) at a time synchronized with the hop sequence. After DPTT is de-asserted, the radio RF output remains active and a radio hop will not occur. This allows for the transmission of the crypto postamble. The radio RF output remains active for longer than is required for the transmission of the crypto postamble, which is shown as the ET_2 time period in Figure J-7. For HAVEQUICK II radios, ET_1, crypto postamble PLUS ET_2 equals the transmitter turnaround time, (TTURN = $ET_1 + ET_2$), as defined in Appendix C.

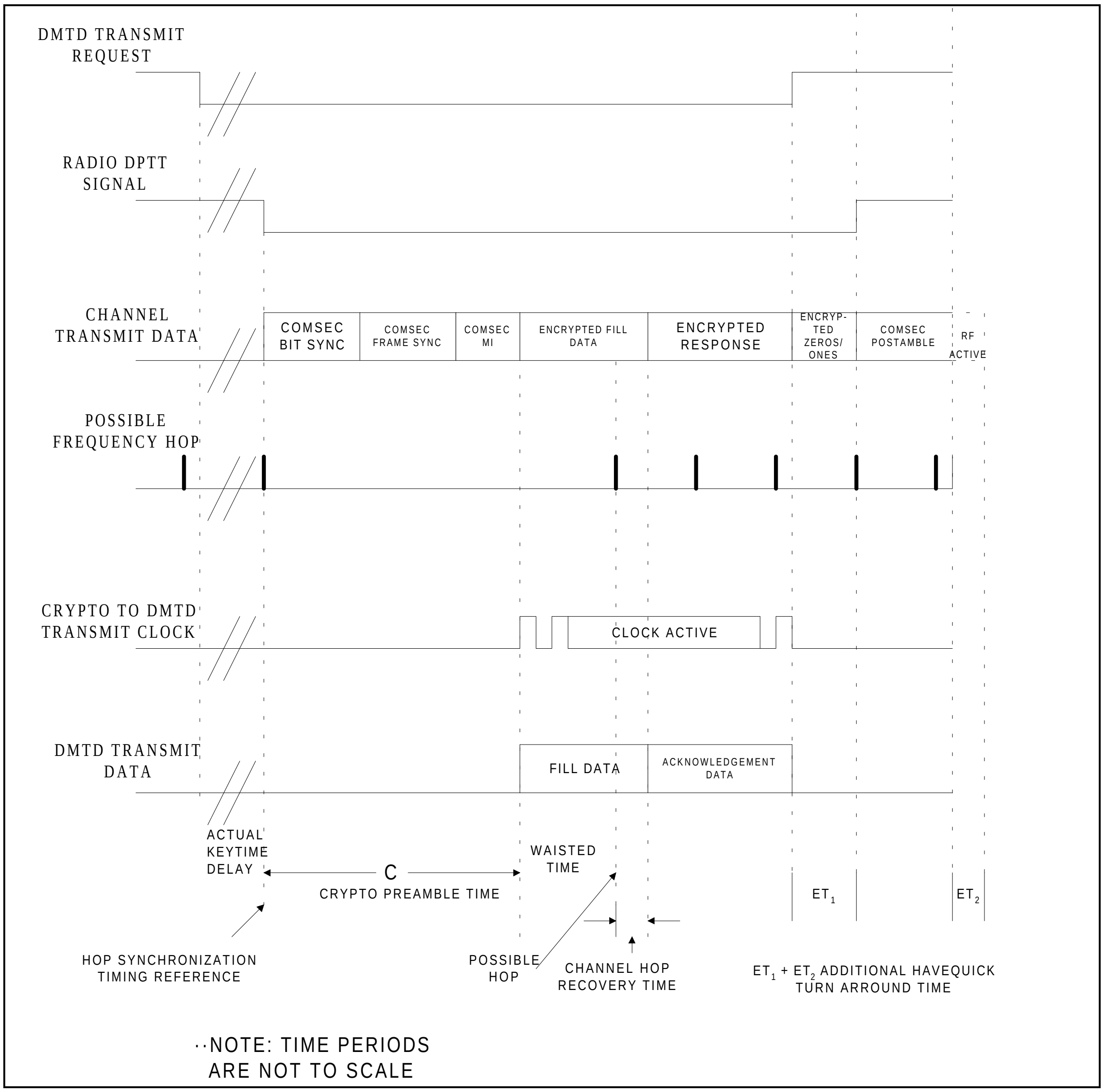


FIGURE J-7. HAVEQUICK II external crypto acknowledgment transmission.

J.3.5.2.3 Estimation of multi-dwell n_l. Figure J-8 shows an example of the n_l data rate for a multi-dwell transmission with a channel data rate of 16 kbps. This is the worst case data rate reduction which would be experienced with rate 1/3 convolutional coding, a 64 bit SOP pattern length, and 3 out of 5 majority logic decoding of the segment count field. The data rate shown

Figure J-8 is the number of link bits to transmit divided by the number of channel bits transmitted times the n_c. Since rate 1/3 convolutional encoding is used in this example, the maximum link data rate achievable would be 5.33 kbps. For short messages, the radio hop timing at the beginning of the transmission has a significant impact on the transmission efficiency. This example uses 13 segments per packet which is the recommended segment per packet count for long transmissions using 3 out of 5 majority logic. This figure and the equations given in Table J-3 are given as an aid for network throughput estimation and should not be used for network timing. The bit rate estimating equation used in Figure J-3 is:

$$\text{link rate} = nc / (0.5*10^{(\text{-link bits * .00003})} + 1.301)$$

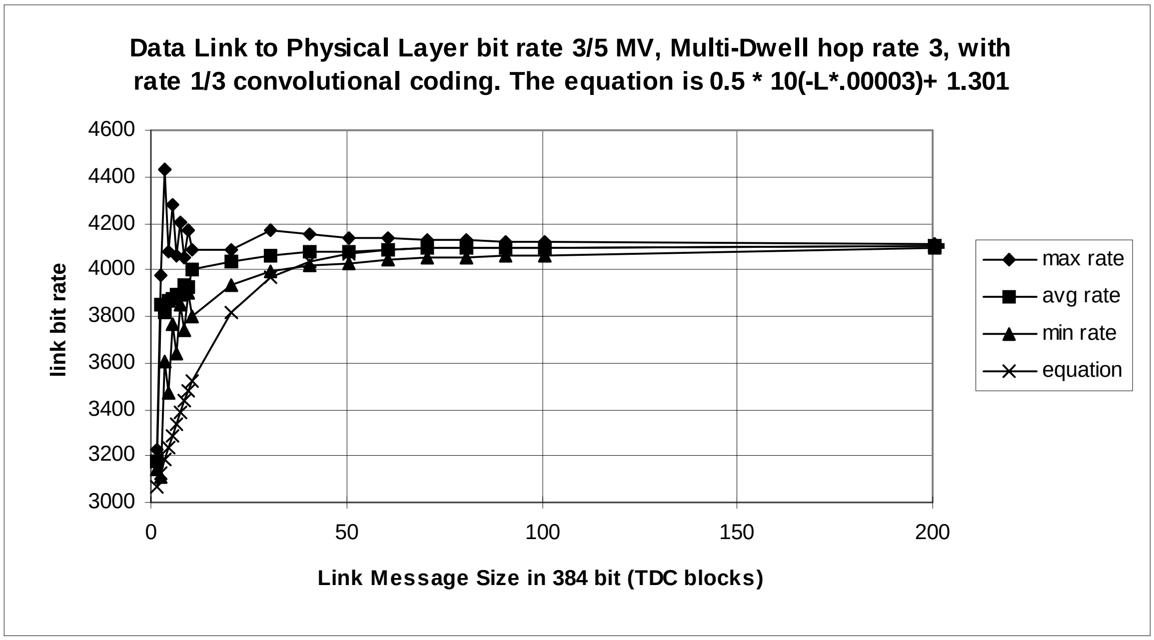


FIGURE J-8. Link data rate as a function of message size.

J.3.5.2.4 Receive processing delays. In order to calculate the reference point for the RHD and TP timers, the receiving DTE must know the time of arrival of the last bit of the transmission. In order to do this, the data link layer normally determines the last bit of the transmission after decoding the word count and tags the arrival of the last data bit from the physical layer. The physical layer receive delays are dependent on the DTE hardware and software implementation. The two delay components are processing delays and data pipeline delays. The processing delays are independent of the received data rate and the pipeline delays are dependent on the data rate. If the receive data rate is known, the data link layer can calculate the time of arrival of the last bit of the message by subtracting off the processing and pipeline delays. If the received data rate is not known, it is impossible to convert a pipeline delay from bits to seconds. The data rate of all non-multi-dwell transmissions is known to be either n_c or $n_c/3$ dependent on the use of rate 1/3 convolutional coding. The received data rate of a multi-dwell transmission is not known. For this reason, when a multi-dwell transmission is received, the physical layer must tag the time of arrival of the final multi-dwell bit. The physical layer can determine the time of arrival of the last bit by using the end of frame flag which is the most significant bit in the final multi-dwell segment count field. A logical signal from the physical layer to the data link layer indicating the message completion time is required to insure that the transmitter and receiver(s) use the same reference point for the calculation of RHD and TP.

The trace back buffer length of the Viterbi decoder introduces a known pipeline delay in the received data. The length of the trace back buffer is an implementation choice which is dependent on the Viterbi decoder architecture. Pipeline delay is the time needed to flush the trace back buffer.

J.3.5.3 Timeout period (TP). The timeout period is derived from the following equations as described in Appendix C:

$$TP = (j \times RHD_0) + \text{Maximum}(DTEACK, TURN)$$
$$TP = \text{Maximum}(DTEPROC, TURN)$$
$$TP = (15 \times RHD_0) + TOL + TURN$$

Modifications to the timeout period are result of changes to RHD_0 and DTE receive processing delays, which have been addressed in paragraph J.3.5.2 and its subparagraphs.

J.3.5.4 Network access delay (NAD). There are no modifications to the network timing equations associated network access delay. The network access delay is always an integer number times the Net_Busy_Detect_Time which, as previously discussed, has not been modified.

J.3.6 Application guidance for the HAVEQUICK II link.

J.3.6.1 Frequency hop synchronization. The HAVEQUICK II TRANSEC timing and the DTE network timing are not synchronized. To avoid the loss of critical data, such as the cryptographic synchronization and/or the protocol SOM patterns, the DTE transmission timing must be synchronized to the frequency hops. The radio should provide a DPTT signal which marks the beginning of a hop dwell with a guaranteed minimum duration. This minimum dwell period is sufficient to carry the synchronization field of an external cryptographic device or the robust frame synchronization field when an internal cryptographic device is used.

J.3.7 Summary. The physical layer robust protocol introduces additional transmit and receive delays due to the robust header and the convolutional decoding pipeline delays. Multi-dwell packetizing introduces a data rate reduction which varies widely for short transmissions. The HAVEQUICK II radio introduces variable delays in the keytime delay and the equipment turn-around time. To maintain network timing using the type 1 timing equations, the net busy sense timing and the response transmission time must be a known constant. In most cases, the response can be transmitted without using the multi-dwell packetizing algorithm. When the multi-dwell packetizing algorithm must be used to transmit a response, the worst case time to complete the transmission is used in the response transmission time component of the term TURN in RHD.

The message transmission time is variable and is only required to be known at the end of the transmission. Two additional physical to data link signals are required to mark the time of the last transmitted bit for transmission, and the time of the last received bit for a reception.

APPENDIX K

BOSE - CHAUDHURI - HOCQUENGHEM (15, 7) CODING ALGORITHM

K.1. General.

K.1.1 Scope. This appendix describes a linear block cyclic code capable of correcting any combination of two or fewer errors in a block of 15 bits.

K.1.2 Application. This appendix is a conditionally mandatory part of MIL-STD-188-220. It is mandatory for implementing the Robust Communications Protocol described in Appendix J.

K.2. Applicable documents. This section is not applicable to this appendix.

K.3. BCH (15,7) code. The BCH (15,7) code is a linear, block, cyclic, BCH code capable of correcting any combination of two or fewer errors in a block of 15 bits. The generator polynomial for this code is

$$g(x) = 1 + X^4 + X^6 + X^7 + X^8$$

where g(x) is a factor of $X^{15} + 1$

K.3.1 Hardware encoding. BCH (15, 7) encoding can be performed with an 8 stage feedback shift register with feedback connections selected according to the coefficients of g(x). A shift register corresponding to the coefficients of g(x) is shown in Figure K-1.

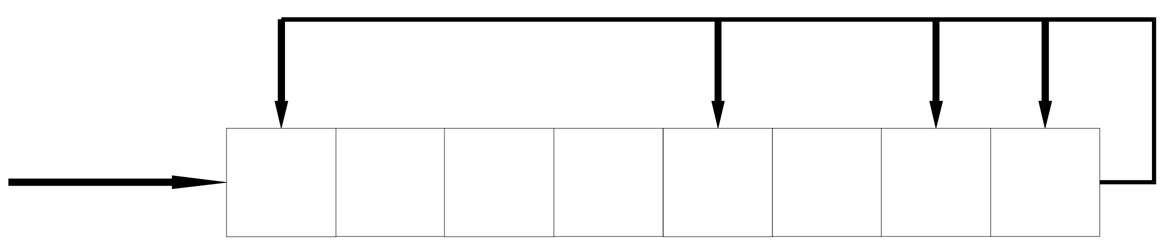
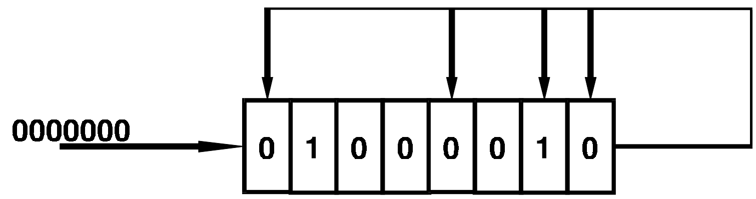


FIGURE K-1. Shift register encoder for the BCH (15, 7) code.

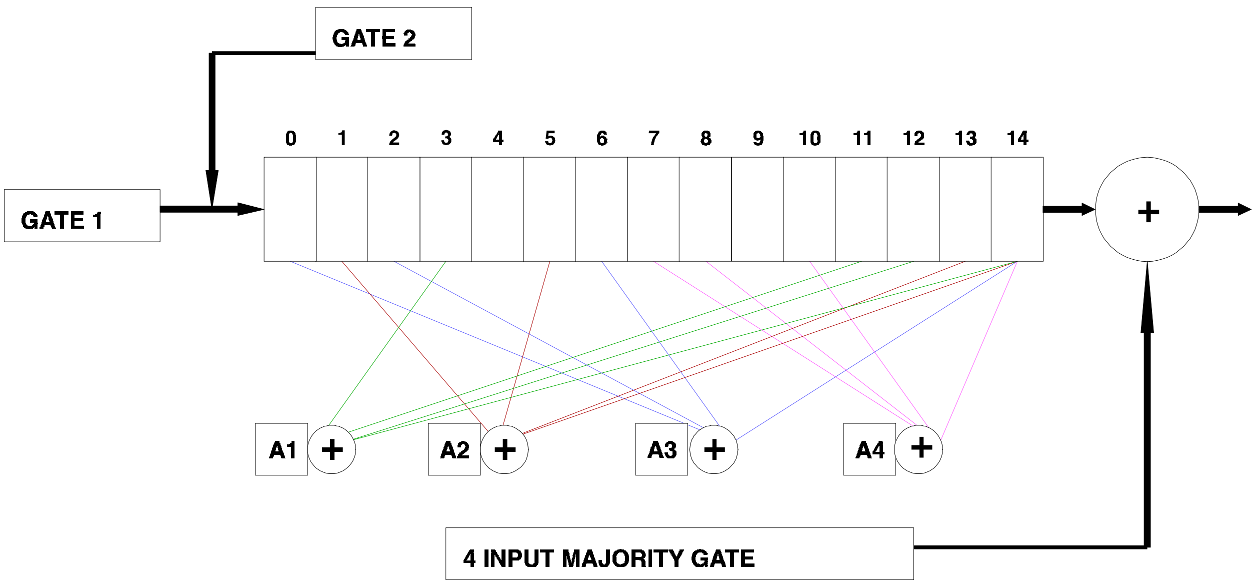
Figure K-2 illustrates its operation by showing the encoding of the information vector (1000010) to form the code vector (10100101 | 01000010), where the parity check sequence is shown before the partition and the information sequence after. The information sequence with eight zeros after it (place holders for the parity bits to be calculated) is shifted into the register initially (it is really a fifteen bit shift register but only the last eight positions correspond to the coefficients of g(x) and contain feedback connections). The operation of the shift register consists of seven rounds of shift, feedback, and sum operations. The parity portion of the code vector can then be read out of the shift register as shown.



Shift	000000	0	0	1	0	0	0	1	0
Feedback		0				0		0	0	
Sum	000000	0	0	1	0	0	0	0	1	
Shift	00000	0	0	0	1	0	0	0	0	1
Feedback		1				1		1	1	
Sum	00000	1	0	0	1	1	0	1	1	
Shift	0000	0	1	0	0	1	1	0	1	1
Feedback		1				1		1	1	
Sum	0000	1	1	0	0	0	1	1	0	
Shift	000	0	1	1	0	0	0	1	1	0
Feedback		0				0		0	0	
Sum	000	0	1	1	0	0	0	1	1	
Shift	00	0	0	1	1	0	0	0	1	1
Feedback		1				1		1	1	
Sum	00	1	0	1	1	0	0	0	1	
Shift	0	0	1	0	1	1	1	0	1	0
Feedback		0				0		0	0	
Sum	0	0	1	0	1	1	1	0	1	
Shift		0	0	1	0	1	1	1	0	1
Feedback		1				1		1	1	
Sum		1	0	1	0	0	1	0	1	

FIGURE K-2. Encoding example.

K.3.2 Hardware/Software decoding. Because of its special structure (it is completely orthogonalizable in one step), the BCH (15,7) code can be decoded very efficiently with a majority logic scheme which can be directly implemented in software or hardware. It is most easily described in terms of the shift register implementation shown in Figure K-3. With gate 2 open and gate 1 closed, the received block is read into the shift register. The output of the four modulo 2 summers is sampled by the majority gate and processed as follows: if a clear majority of the inputs are ones (three or more) then the output is one, otherwise (if two or fewer inputs are ones) the output is zero. This output is used to correct the last bit of the shift register. The corrected bit is output to the receiver and feedback through gate 2 as the register is right shifted. The process is now repeated thirteen times until the last bit is corrected.



A1	3	11	12	14
A2	1	5	13	14
A3	0	2	6	14
A4	7	8	10	14

FIGURE K-3. BCH (15, 7) majority logic decoding.

K.3.3 Software encoding. The BCH (15,7) code is most efficiently encoded in systematic form from the generator matrix shown in Figure K-4.

	1	0	0	0	1	0	1	1	1	0	0	0	0	0	0
	1	1	0	0	1	1	1	0	0	1	0	0	0	0	0
	0	1	1	0	0	1	1	1	0	0	1	0	0	0	0
G =	1	0	1	1	1	0	0	0	0	0	0	1	0	0	0
	0	1	0	1	1	1	0	0	0	0	0	0	1	0	0
	0	0	1	0	1	1	1	0	0	0	0	0	0	1	0
	0	0	0	1	0	1	1	1	0	0	0	0	0	0	1
	Parity								**Identity**						

FIGURE K-4. BCH (15, 7) generator matrix.

CONCLUDING MATERIAL

a. Preparing Activity:

US Army Communication Electronics Command (USACECEOM) - CR1

b. Custodians:

Army:	CR
Navy:	OM
Air Force:	02
DIA:	DI
DISA:	DC1
DLA:	DH
NSA:	NS
Other:	North American Aerospace Defense Cmd. (NORAD) & US Space Cmd. (USSPACECOM): US

c. Review Activities:

OASD	IR, SE
Army	AC, MI, PT
Navy	CG, EC, MC, NC, TD
Air Force	11, 13, 93
NIMA	MP
DOT	OST
DISA	DC5

d. Project number:

TCSS-00101

STANDARDIZATION DOCUMENT IMPROVEMENT PROPOSAL

INSTRUCTIONS

1. The preparing activity must complete blocks 1,2, 3, and 8. In block 1, both the document number and revision letter should be given.
2. The submitter of this form must complete blocks 4, 5, 6, and 7.
3. The preparing activity must provide a reply within 30 days from receipt of the form.

NOTE: This form may not be used to request copies of documents, nor to request waivers, or clarification of requirements on current contracts. Comments submitted on this form do not constitute or imply authorization to waive any portion of the referenced document(s) or to amend contractual requirements.

I RECOMMEND A CHANGE:	1. DOCUMENT NUMBER MIL-STD-188-220B	2. DOCUMENT DATE *(YYMMDD)* 980120

3. DOCUMENT TITLE DIGITAL MESSAGE TRANSFER DEVICE SUBSYSTEMS

4. NATURE OF CHANGE *(Identify paragraph number and include proposed rewrite, if possible. Attach extra sheets as needed.)*

5. REASON FOR RECOMMENDATION

6. SUBMITTER

a. NAME *(Last, First, Middle Initial)*	b. ORGANIZATION	
c. ADDRESS *(Include Zip Code)*	d. TELEPHONE *(Include Area Code)* (1) Commercial (2) DSN *(If applicable)*	7. DATE SUBMITTED *(YYMMDD)*

8. PREPARING ACTIVITY **US Army Communication Electronics Command (USACECOM)**

a. NAME Mr. Edward Robinson	b. TELEPHONE *(Include Area Code)* (1) Commercial 732-532-6075 (2) DSN 992-6075
c. ADDRESS *(Include Zip Code)* CDR, USACECOM ATTN: AMSEL-SE-CD BUILDING 1209 FT. MONMOUTH, NJ 07703	**IF YOU DO NOT RECEIVE A REPLY WITHIN 45 DAYS, CONTACT:** Defense Quality and Standardization Office 5203 Leesburg Pike, Suite 1403, Falls Church, VA 22041-3466 Telephone (703) 756-2340 DSN 289-2340