→ Prof. Corbató

24 Oct
RGM

How does
Ascii map to
EBCDIC of
360 installations
?

Data Communications Conventions for
ARPA Contractor
Machine-to-Machine Transmission

An initial draft of conventions was drawn and circulated by F. H. Westervelt (University of Michigan) on May 12, 1967 and was the subject of an intensive review by interested parties on May 19, 1967 at ARPA headquarters. Following that meeting further work has been carried on by R. Mills, A. K. Bhushan, R. H. Stotz, et al (Massachusetts Institute of Technology) and by F. H. Westervelt, D. L. Mills, K. E. Burkhalter, et al (University of Michigan) resulting in the current proposal.

Two distinct sets of conventions for ARPA contractor machine-to-machine data communication purposes are required. The first set of conventions detail the characteristics of the physical facilities to be employed. The second set establish the programming conventions and protocols for using the physical equipment. Clearly, each set of conventions must reflect the needs and constraints implied by the other. Further it is expected that both sets of conventions will evolve continuously as further developments and equipment becomes available. This document then reflects the current capabilities for the purpose of establishing an initial interchange among ARPA contractor machines.

I. Physical Facilities

The physical facilities required for ARPA Contractor Machine-to-Machine Data Transmission shall have the following characteristics:

1. The data transmission interconnection facility shall be full-duplex with equal bandwidth for simultaneous data transmission in either direction.
2. The method of transmission shall be binary serial synchronous.
3. The communications interconnection shall be established by utilizing dialable facilities.
4. Each installation shall be capable of automatic answering of incoming calls and capable of automatic dialing to originate outgoing calls.
5. Each simplex side of the full-duplex transmission channel shall have a minimum capacity of 2400 bits per second. Thus, the full-duplex channel shall have a minimum capacity of 4800 bits per second.

The above specifications are within the capability of several major common carriers. These specifications are nominally those for voice-grade systems using Western Electric type 201 serial synchronous modulator-demodulators. The selection of the common-carrier to supply these facilities will be made by ARPA headquarters in order to obtain the necessary uniformity of equipment and service at the most favorable price.

A second facet of the physical facilities needed by the ARPA Contractor system is the necessary interfacing of the contractor machines to the modems and facilities specified above. Here one must also recognize the implicit need to conform to the programming conventions specified by Section II of this document. Such interfacing probably shall always remain in the jurisdiction of the local facility because of the peculiar characteristics of each individual contractor machine. However, it has been suggested by Wes Clark, and supported by many others, that ARPA headquarters give support to the development of an Interface Message Processor (IMP) designed to operate with internal buffers and programs between the common carrier facility and the various contractor machines. Such a machine would tend to insulate each installation from the problems certain to be encountered during the development of the ARPA contractor machine dialogues.

A prototype machine of this type has been constructed to implement specific objectives of the CONCOMP project at The University of Michigan. This machine consists of a Digital Equipment Corporation PDP-8 with 8k core and special hardware interfaces to an IBM 2870 multiplex channel on the System/360 Model 67 and to the common carrier modems. Provision for automatic dialing has been made in this equipment. This device is capable of handling all of the message conventions in this document including error recovery, line status and message formatting as well as adapting these conventions to local systems practices at The University of Michigan. If the IMP concept were adopted, a similiar but newly designed and implemented facility would be developed to interface with each contractor machine and observe the ARPA conventions between contractor machine systems.

Alternatively, a contractor may elect to interface his machine directly with the common carrier modems and provide the software support for the ARPA conventions within the operating system for the installation. Clearly, the responsibility for system performance rests with the local installation. In one case, the IMP may be programmed to look like a device with a well known set of characteristics and a well understood behavior thus minimizing the impact on any local system features of changes in ARPA conventions. In the other case, the local system must include the ARPA conventions and be prepared to include periodic modifications of these conventions within the system as further experience may direct.

II. Data Communications Conventions

The communications conventions described herein are compatible with the American Standard Code for Information Interchange (ASCII) and follows the recommended standard given in "Control Procedures for Data Communications - An ASA Progress Report," published in the February 1966 issue of the Communications of the ACM.

The message format is the ARPA Contractors Convention for data communication between machines, using fill-duplex binary synchronous communication facilities. It permits convenient multiple message transmission in a multipoint operation which may or may not be centralized.

Communications Syntax

The ARPA convention for data communications is defined syntaxically by the constructions in Table I together with the character codes defined in the USASCII standard displayed in Table II and Table III. The cyclic parity code computation procedure is given in assembly code for a PDP-8 in Table IV.

An examination of Table II will emphasize the odd lateral parity implied for all USASCII characters. All data is transmitted in the ARPA convention in eight-bit frames from least to most significant bit. The parity bit is the last bit sent in any frame.

Cyclic parity check characters for longitudinal error control are calculated from the modulo two polynomial X**16 + X**15 + X**2 + 1. This method was selected to provide for protection against burst errors extending over several character times because of line fade or other transmission system malfunction. The cyclic parity check further allows for error control during transparent binary transmission since error control does not depend upon lateral parity.

Transmission

The unit of activity on a simplex side of the full-duplex interconnection is a "Transmission". As defined in Table I, a Transmission is an unbroken stream of 8-bit characters beginning with two (or more) SYN (Synchronization, hexadecimal 16) characters followed by the "Content" of the transmission and terminated by an EOT (End of Transmission, hexadecimal 04) and followed by at least two NUL (Null, hexadecimal 80) characters for line padding. The important concept is the unbroken stream of eight-bit frames from initial line synchronization thru the end of transmission. Because the interconnection is presumed to be full-duplex it is desirable to structure the "content" syntax to provide multiple messages, control information and multiple priority levels.

Content

The "Content" of a transmission may consist of a "Header", or a "Message", or a "Message" followed by a "Header". As may be observed, a "Header" consists of an SOH (Start of Header, hexadecimal 01) followed by a "Heading" and an arbitrary number of SYN characters as needed to maintain a continuous transmission stream. More will be said about Headers and Headings later.

Message

A"Message"consists of a Header followed by "Text", or a Message followed by a Message, or a Message followed by an arbitrary number of SYN characters. Thus, the content of a transmission may contain any number of Messages. It should be noted that every component of a transmission content has a Header prefix. The Header may stand alone and if so no error checking is required and no acknowledgement or other response is expected from the receiving station. The collection of cyclic parity begins with the character following the SOH. Error control is exercised at the end of each "Text" (or "Text Block" as will be described later).

Each Message requires acknowledgement in a Header returned by the receiver via the return channel of the full-duplex interconnection. A Header alone requires no acknowledgement. Otherwise one might be faced with an unending sequence of acknowledgements of acknowledgements.

It may be interesting to examine two special transmissions. The Non-Text transmission is a minimum of six characters plus any Heading characters desired. No acknowledgement is required or sent.

S	S	S		E	N	N
Y	Y	O	Heading	O	U	U
N	N	H		T	L	L

Non-Text Transmission

A second special case is the Null-Text Transmission. In this case, error control is applied and confirmation is expected from the receiver. The smallest Null-Text Transmission is ten characters plus any desired Heading. Clearly the Null-Text Message is useful whenever confirmation of Header control is desired from the reciever.

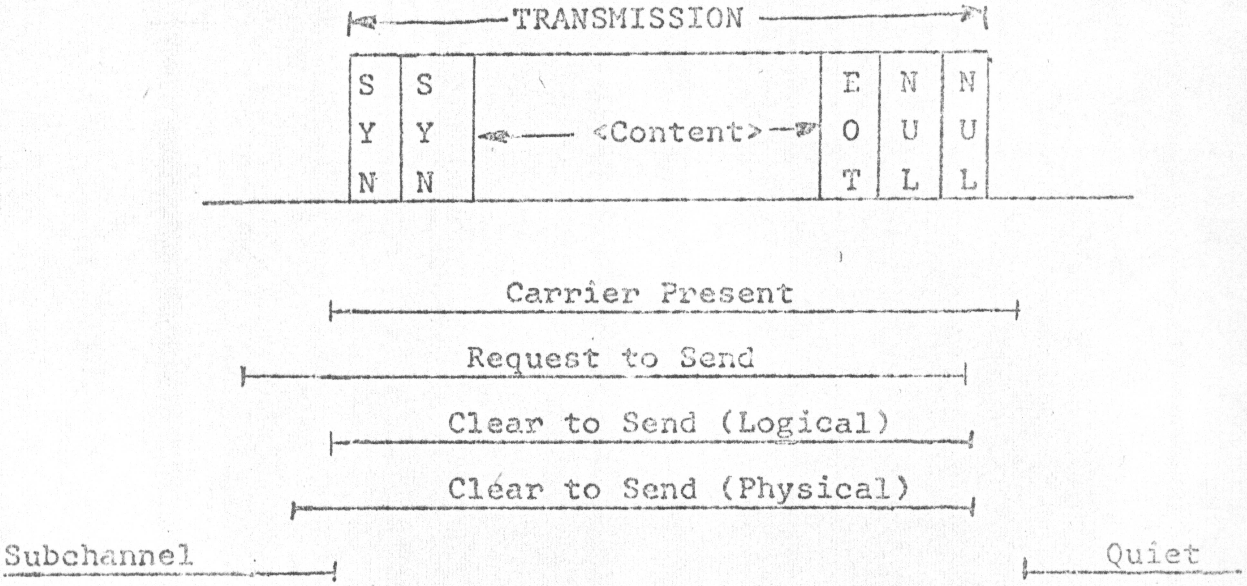
S	S	S		S	E	C	C	E	N	N
Y	Y	O	Heading	T	T	C	C	O	U	U
N	N	H		X	X	1	2	T	L	L

|←Null - Text Message→|

|←Null - Text Transmission————→|

The ARPA convention allows the simplex side of the communications interconnection to go quiet between transmissions. Clearly the overhead of synchronous transmission is reduced thru the use of multiple messages and long transmissions. The asynchronous nature of full-duplex operation may require the interruption of a long message by high priority traffic (such as controls, etc.). Also the nature of interaction between the communications system and the host computer may sometimes force the breaking of Text into blocks whenever the communications system is unable to get service from the channel on the host system. The ARPA communications syntax provides for the necessary generality to treat these situations.

Figure 1 - Transmission Structure and Timing



Headers and Headings

Information in the header of the message is used only to aid the communications programs to transmit, receive, and sequence the messages efficiently. The header information is never seen by the user, and all information regarding the contents of the text portion of the message (e.g., should the ASCII characters in the text be interpreted as ASCII or binary) is contained in the text portion of the message.

All characters in the heading are divided into three categories--control characters, key characters, and argument characters. Control characters are all those characters with octal values from 00 to and including 37. Key characters are all those characters with octal values between and including 40 through 77, excluding the numerals. Argument characters are all those characters with octal values greater than 77 and the numerals (see Table II and III).

The uses for the ASCII control characters are defined in the ASA standard and are not redefined here. The key characters, defined below, may or may not have arguments following them. The same argument may take on a different meaning depending on which key character preceded it.

Summary of Key Characters

	# arg	argument is message identification
Ack	+ arg	acknowledgement message #arg (message O.K.)
?, NAK	- arg	negative acknowledgement message #arg (message in error)
	? arg	repeat acknowledgement of message #arg (equivalent to END)
	= n	make standard buffer size n. See note on n below.
	. n	send transmissions no longer than size n.
	< n	"=n" request rejected.
	> n	"=n" request accepted.
	* n	size n blocks of text only.
	: arg	sender's status
	; arg	sender's interpretation of receiver's status.
	"	Interrupt (similar to TTY break)
	!	Quit (similar to TTY quit) stops transmission of messages.
	/ arg	message routing information.

The character n in the above list stands for any numeral. A message of size n contains no more than 2**(6+n) characters, with n ranging between 0 and 9. This means that the range of allowable message sizes falls between 64 and 32,768 characters. Since messages can be of variable length, n serves primarily as an upper bound on the maximum size of the message. This bound is needed in setting up receive buffers. Additional key characters may be generated by using the ESC (Escape) character. An ESC followed by an ASCII graphic (octal 040 to 176) assumes the meaning of a new key character (ESC graph).

The unassigned key characters (or ESC graphic characters) may be made use of to define supplementary header information as and when needed, but ASCII control characters should be assigned meaning only in accordance with ASA recommendations.

Text

As shown in Table I, "Text" may consist of either "Character Text" or "Binary Text". As may also be seen it is possible to break text into "Blocks" at any time without prior notice or warning to the receiver. The need for this capability arises whenever the transmitter is unable to get service from its host computer. Clearly the receiver must be informed that the text received thus far is only a portion of the full text and not yet complete. Further the full-duplex nature of the ARPA communications convention admits the possibility of traffic on a subchannel at the same time that acknowledgements and other control or high priority data must go thru the subchannel. The syntax of "Text" allows a Message to be imbedded in any text in a Last-In-First-Out fashion with no inherent depth constraints on nesting. Note that error control is complete within any block of text. Therefore an erroneous message may be stopped before the entire text has been sent in some useful cases and retransmission or other corrective procedures initiated. If the message accounting space exceeds the available space for this nested behavior the unfinished messages may by negatively acknowledged and error indications sent to the host computer.

Character Text

"Character"Text begins with an STX (Start of Text, hexadecimal 02) character and ends with either an ETB (End of Text Block, hexadecimal 97) in case the current text is not yet complete or as ETX (End of Text, hexadecimal 83) for a completed message. Error control codes follow either ending character. If a text is interrupted because of an inability of the host computer to maintain the text stream then the accumulation of cyclic parity commences with the character following the resuming STX (or DLE STX if the text is binary). Note that text error control is never left unresolved. As soon as any text or block is terminated, error control is transmitted and may be used by the receiver to finish its message accounting tasks for the section.

The information which may be transmitted in Character Text is defined in the syntax of Table I. In other terms, it may be observed that all USASCII characters are allowed except for the ten control characters of sets β and γ reserved by the ASCII standard. It should be observed that binary data may be sent within character text by using the following convention:

1. The Binary ASCII string is begun by the occurence of the GS (Group Separator, hexadecimal 9D) character.

2. The binary ASCII string is composed of the low order six bits of the succeeding characters. The characters are chosen from the set τ and admits $2^6 = 64$ unique combinations with all 7th level bits forced to one (thus avoiding any conflict with control characters)

3. The binary ASCII string is terminated by the FS (Field Separator, hexadecimal 1C) character.

All text is passed untouched by the communication system as noted by the section on Headers and Headings. The binary ASCII string interpretation is therefore an ARPA convention not an ASCII standard.

Binary Text

Nearly transparent eight-level binary text is allowed in the ASCII standard thru the use of the DLE (Data Link Escape, hexadecimal 10) character. As shown in Table I, the DLE is used to precede the control characters STX, ETB and ETX modifying their action for binary transmission. As may be observed, all eight level codes may be transmitted within binary text except hexadecimal 10 (DLE). The code DLE is transmitted within binary text as a contiguous pair of DLE codes.

1. How do I turn it on.
2. Will common carrier mix up with this?

Within binary text the receiver looks for DLE characters passing all other bits thru untouched. When a DLE appears the receiver examines the next character for ETX, ETB or DLE. All other characters are erroneous. The second DLE, if it occurs, is deleted from the text stream thus passing the desired hex decimal 10 code thru the system.

The ETX and ETB end the text (or text block) and initiate error control. The cyclic parity check will provide the necessary error control without lateral parity on each character. Lateral parity "errors" obviously occur on one-half of all eight level codes. Cyclic parity checking does not require lateral parity for system integrity.

Further Needs in ARPA Convention Discussions

(1) A need for multiple message transmission is envisaged to reduce the number of interrupts on the computer by short messages. A large number of messages may be transmitted and received in a single transmission, and then the receive computer can take appropriate action and send back acknowledgements. An EOT is used for this purpose.

By suitable protocol procedures, the transmitter will have an idea of the receiver-buffer size available (or create a standard buffer size in receiver). An EOT will be sent as soon as this capacity is reached. Transmission may then be resumed when proper acknowledgements are received and buffer space is available again.

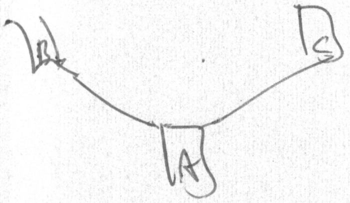
If a standard buffer size was not created and the receive-buffer capacity is reached without the receipt of an EOT, the receiver may take the appropriate action by itself, and send a quit signal to the transmitter. It may also indicate to the transmitter to send transmissions no longer than size n.

The exact procedures for creating buffers, when to cause interrupts, etc., need further discussion and elaboration.

(2) The format and syntax for status information to be transmitted and used, and what is to be contained in it, has not been decided. Further investigation and discussion is required.

(3) The manner in which routing information is to be contained in the header and used needs to be discussed.

(4) Protocol arrangements have not been indicated and are left open for discussion. It seems that it will be desirable to use a conversational mode (i.e., having information in header) for all protocol purposes (acknowledgement, link establishment and relinquishment, and priority, etc.). Further discussion is needed on exact implementation, and requirements.



Can B → A
and C → A
transmission occur simultaneously?

Who pays cost of comm. line? A, B, or C.

TABLE I

ARPA COMMUNICATIONS CONVENTION SYNTAX

(SEE TABLE II FOR CHARACTER DEFINITIONS)

```
                   SS           ENN |
<TRANSMISSION> ::= YY <CONTENT> OUU |
                   NN           TLL |

                   S               |
                   Y<TRANSMISSION> |
                   N               |

                                  N
                   <TRANSMISSION> U
                                  L

<CONTENT> ::=      <HEADER>|

                   <MESSAGE>|

                   <MESSAGE><HEADER>
                  S
<HEADER> ::=      O<HEADING>|
                  H        S
                   <HEADER>Y
                           N

<MESSAGE> ::=      <HEADER><TEXT>|

                   <MESSAGE><MESSAGE>|
                            S
                   <MESSAGE>Y
                            N

<HEADING> ::=  <KEY CHARACTER> | <CONTROL CHARACTER>|

                   <HEADING><ARGUMENT>|

                   <HEADING><HEADING>
```

```
       <TEXT> ::= <CHARACTER TEXT>|

                  <CHARACTER TEXT BLOCK><TEXT>|

                  <CHARACTER TEXT BLOCK><MESSAGE><TEXT>|

                  <BINARY TEXT>|

                  <BINARY TEXT BLOCK><TEXT>|

                  <BINARY TEXT BLOCK><MESSAGE><TEXT>

 <KEY CHARACTER> ::= <δ>

                         G F
                 ::= <η>|S|S

 <CONTROL CHARACTER> ::= <ε>|<Ψ>

 <ARGUMENT> ::= <α>|<φ>|<Ψ>|<ARGUMENT><ARGUMENT>
                                            E
 <CHARACTER TEXT> ::= <START CHARACTER TEXT>T<ERROR CHECK CODES>|
                                      S     X
                      <CHARACTER TEXT>Y
                                      N
                                                E
<CHARACTER TEXT BLOCK> ::=<START CHARACTER TEXT>T<ERROR CHECK CODES>|
                                                B S
                          <CHARACTER TEXT BLOCK > Y
                                      DE          N
 <BINARY TEXT> ::= <START BINARY TEXT>LT<ERROR CHECK CODES>|
                                S     EX
                   <BINARY TEXT>Y
                                N
                                           DE
<BINARY TEXT BLOCK> ::= <START BINARY TEXT>LT<ERROR CHECK CODES>|
                                           EB
                                            S
                        <BINARY TEXT BLOCK> Y
                                            N
```

```
                                     S|
<START CHARACTER TEXT> ::=           T|
                                     X|

                                   <START CHARACTER TEXT><ASCII STRING>
                               DS|
<START BINARY TEXT> ::=        LT|
                               EX|

                               <START BINARY TEXT><BINARY STING>

<ERROR CHECK CODES> ::= CYCLIC CHECK CHARACTERS SEE Table IV

                        FOR SAMPLE PROGRAM.

<ASCII STRING> ::= <PURE ASCII>|<NULL STRING>

                   <BINARY ASCII STRING>|

                   <ASCII STRING><ASCII STRING>

<PURE ASCII> ::= <α>|

                 <η>|

                 <ε>|

                 <φ>|

                 <Ψ>

                                            F
<BINARY ASCII STRING> ::= <START BINARY ASCII>S

                          G|
<START BINARY ASCII> ::=  S|

                          <START BINARY ASCII><τ>

<BINARY STRING> ::= <α>|<β>|<δ>|<ε>|<φ>|<Ψ>|<λ>|

                   DD|
                   LL|<NULL STRING>|
                   EE|

                  <BINARY STRING><BINARY STRING>

<NULL STRING> ::= EMPTY STRING, NO CHARACTERS WHATSOEVER.
```

UNITED STATES OF AMERICA STANDARD CODE FOR INFORMATION INTERCHANGE
(USASCII)

BIT POSITIONS (HEXADECIMAL) HIGH ORDER → LOW ORDER ↓	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	λ	γ DLE	ε SP	λ	α τ @	λ	λ	α τ p	α NUL	λ	λ	∅ 0	λ	α τ P	α τ `	λ
1	β SOH	λ	λ	∅ 1	λ	α τ Q	α τ a	λ	λ	δ η DC1	ε !	λ	α τ A	λ	λ	α τ q
2	β STX	λ	λ	∅ 2	λ	α τ R	α τ b	λ	λ	δ η DC2	ε "	λ	α τ B	λ	λ	α τ r
3	λ	δ η DC3	ε #	λ	α τ C	λ	λ	α τ s	β ETX	λ	λ	∅ 3	λ	α τ S	α τ c	λ
4	β EOT	λ	λ	∅ 4	λ	α τ T	α τ d	λ	λ	δ η DC4	ε $	λ	α τ D	λ	λ	α τ t
5	λ	β NAK	ε %	λ	α τ E	λ	λ	α τ u	β ENQ	λ	λ	∅ 5	λ	α τ U	α τ e	λ
6	λ	β SYN	ε &	λ	α τ F	λ	λ	α τ v	β ACK	λ	λ	∅ 6	λ	α τ V	α τ f	λ
7	δ η BEL	λ	λ	∅ 7	λ	α τ W	α τ g	λ	λ	β ETB	ε '	λ	α τ G	λ	λ	α τ w
8	δ η BS	λ	λ	∅ 8	λ	α τ X	α τ h	λ	λ	δ η CAN	ε (	λ	α τ H	λ	λ	α τ x
9	λ	δ η EM	ε)	λ	α τ I	λ	λ	α τ y	δ η HT	λ	λ	∅ 9	λ	α τ Y	α τ i	λ
A	λ	δ η SUB	ε *	λ	α τ J	λ	λ	α τ z	δ η LF	λ	λ	ε :	λ	α τ Z	α τ j	λ
B	δ η VT	λ	λ	ε ;	λ	α τ [	α τ k	λ	λ	δ η ESC	ε +	λ	α τ K	λ	λ	α τ {
C	λ	δ FS	ε ,	λ	α τ L	λ	λ	α τ |	δ η FF	λ	λ	ε <	λ	α τ \	α τ]	λ
D	δ η CR	λ	λ	ε =	λ	α τ]	α τ m	λ	λ	δ GS	ε -	λ	α τ M	λ	λ	α τ }
E	δ η SO	λ	λ	ε >	λ	α τ ^	α τ n	λ	λ	δ η RS	ε .	λ	α τ N	λ	λ	α τ ~
F	λ	δ η US	ε /	λ	α τ O	λ	λ	α τ DEL	δ η SI	λ	λ	ε ?	λ	α τ _	α τ o	λ

ALL CHARACTERS ARE TRANSMITTED FROM LEAST TO MOST SIGNIFICANT BIT.

d1

d2 →

SET (SET)	SYMBOL

THE ABOVE TABLE IS ORGANIZED AS SHOWN ON THE LEFT. THE CODES FOR WHICH SYMBOLS ARE ASSIGNED ARE ALL ODD PARITY. THE COLUMN-ROW HEXADECIMAL DIGITS DEFINE THE BIT PATTERN FOR EACH CODE CHARACTER. THE "SET" DEFINES THE GROUPING OF CODES FOR THE PURPOSE OF DEFINING THE COMMUNICATIONS CONTROL CONVENTIONS. THUS;

<γ> ::= DLE

<∅> ::= 0|1|2|3|4|5|6|7|8|9

TABLE III

United States of America Standard
Code for Information Interchange
(USASCII)

Bit Positions			b7 b6 b5	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
b4 ↓	b3 ↓	b2 ↓	b1 ↓								
0	0	0	0	NUL	DLE	SP	0	@	P	`	p
0	0	0	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	STX	DC2	"	2	B	R	b	r
0	0	1	1	ETX	DC3	#	3	C	S	c	s
0	1	0	0	EOT	DC4	$	4	D	T	d	t
0	1	0	1	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	ACK	SYN	&	6	F	V	f	v
0	1	1	1	BEL	ETB	'	7	G	W	g	w
1	0	0	0	BS	CAN	(	8	H	X	h	x
1	0	0	1	HT	EM	)	9	I	Y	i	y
1	0	1	0	LF	SUB	*	:	J	Z	j	z
1	0	1	1	VT	ESC	+	;	K	[	k	{
1	1	0	0	FF	FS	,	<	L	\	l	|
1	1	0	1	CR	GS	-	=	M	]	m	}
1	1	1	0	SO	RS	.	>	N	^	n	~
1	1	1	1	SI	US	/	?	O	_	o	DEL

▤ Control characters

▥ Numerals interpreted as special arguments

▨ Communication control characters

☐ Second category representing argument (includes the control character DEL)

▥ Key characters

⬚ Control character DEL included in the argument set

USASCII Character Assignments

```
*
*                 SUBROUTINE TO COMPUTE CYCLIC PARITY CHECK
*                 USES POLYNOMIAL X**16 + X**15 + X**2 + 1
*
*
*                 CALLING SEQUENCE
*                 ***                     AC=INPUT CHAR
*                 JMS        CYCSUM
*                 ***                     AC=0
*
*                 INPUT CHAR - 8 BITS RIGHT-JUSTIFIED
*                 HIGH-ORDER CHECKSUM - 12 BITS
*                 LOW-ORDER CHECKSUM - 4 BITS LEFT-JUSTIFIED
*
CYCSUM 0                                  ENTRY. AC=INPUT CHAR
        DCA       CYCHR
        TAD       =-10                    INITIALIZE SHIFT COUNT
        DCA       CYCNT
CYC2    TAD       CYCHR                   SHIFT CHAR
        CLL       RAR
        DCA       CYCHR
        TAD       CYCK1                   SHIFT HIGH-ORDER CHECKSUM
        RAR
        DCA       CYCK1
        TAD       CYCK2                   SHIFT LOW-ORDER CHECKSUM
        RAR
        DCA       CYCK2
        TAD       CYCK2                   DID A BIT CREEP OUT OF RIGHT END
        AND       =0200
        SNA       CLA
        JMP       CYC3                    NO. CONTINUE
        TAD       CYCK1                   YES. MOD-2 ADD HIGH-ORDER FB TAPS
        AND       =-5000-1
        DCA       CYCTMP
        TAD       CYCK1
        CMA
        AND       =5000
        TAD       CYCTMP
        DCA       CYCK1
        TAD       CYCK2                   MOD-2 ADD LOW-ORDER FB TAPS
        AND       =-0400-1
        DCA       CYCTMP
        TAD       CYCK2
        CMA
        AND       =0400
        TAD       CYCK2
        DCA       CYCK2
CYC3    ISZ       CYCNT                   HAS SHIFT COUNT REACHED 8
        JMP       CYC2                    NO. CONTINUE
        JMP*      CYCSUM                  YES. NORMAL EXIT
*
CYCNT   DS        1                       BIT COUNT DURING SHIFTING
CYCHR   DS        1                       INPUT CHARACTER DURING SHIFTING
CYCTMP  DS        1                       TEMPORARY
CYCK1   DS        1                       HIGH-ORDER CHECKSUM
CYCK2   DS        1                       LOW-ORDER CHECKSUM
```