

T H E U N I V E R S I T Y O F M I C H I G A N

Memorandum 19

A CYCLIC CHECK COMPUTER FOR ERROR DETECTION

Kenneth E. Burkhalter

CONCOMP: Research in Conversational Use of Computers
F.H. Westervelt, Project Director
ORA Project 07449

supported by:

ADVANCED RESEARCH PROJECTS AGENCY
DEPARTMENT OF DEFENSE
WASHINGTON, D.C.

CONTRACT NO. DA-49-083 OSA-3050
ARPA ORDER NO. 716

administered through:

OFFICE OF RESEARCH ADMINISTRATION ANN ARBOR

June 1968

enjn

UMR0856

TABLE OF CONTENTS

	<u>Page</u>
LIST OF FIGURES.....	v
I. INTRODUCTION.....	1
II. DESIGN DISCUSSION.....	2
a. Error Checking.....	2
b. Design Objectives.....	4
III. SYSTEM DESCRIPTION.....	5
IV. PROGRAMMING AND CONTROL CONSIDERATIONS.....	10
V. DETAILED LOGIC DISCUSSIONS.....	12
IOP Decoding and Device Selection.....	12
Character Buffer.....	17
Residue Register Control.....	19
Residue Register Mod 2 Adders.....	23
Residue Register.....	26
Shift Control and Mode Storage.....	26
PDP-8 Added Circuitry.....	30
APPENDIX A. UTILIZATION MODULE LIST AND CIRCUIT NAME MAP.....	A-1
APPENDIX B. CONNECTOR MAPS.....	B-1
APPENDIX C. DIAGNOSTIC PROCEDURES.....	C-1

LIST OF FIGURES

<u>Figure</u>		<u>Page</u>
1	General Block Diagram.....	6
1.1	CRC-16 BCC Generation.....	7
1.2	CRC-12 BCC Generation.....	8
2	Cyclic Check Computer—Example Routines.....	13
3	IOP Address Decode AC Buffers.....	14
4	Character Register.....	18
5	Residue Register Control.....	20
6	IN/OUT Gating Residue Register.....	22
7	MOD 2 Adders.....	24
8	Generator Shift Register.....	27
9	Run and Mode Control.....	28
10	Additions to PDP-8 CPU.....	31
B1	Cable Layout Map.....	B-2
B2	Connector Map.....	B-3
B3	Connector Map.....	B-4
B4	Connector Map.....	B-5

A CYCLIC CHECK COMPUTER FOR ERROR DETECTION

I. INTRODUCTION

This report discusses the design and use of a hardware device to compute from an input message stream a residue, modulo a program-selectable polynomial, which serves as an error-detecting check over the message itself. The purpose of this device is to free the support processor (the Data Concentrator PDP-8 in this case) from the software overhead burden of checksum computation, which may require up to 500 microseconds per input character in the case of the PDP-8. It is readily possible to compute the same checksum by hardware methods in 4 microseconds! In addition, the reduced time required allows the computation to be accomplished in real time rather than task time, thus allowing simpler programming conventions.

The cyclic check generator is composed of two registers which are loaded and read under program control. The character register is loaded with the new input character and the residue register is loaded with the last computed residue (zero for the first time through). After executing the start command the processor then reads the new contents of the residue register to obtain the current check digits. Since the cyclic check interface holds the PDP-8 in PAUSE state until checksum computation is completed, the programmer is always guaranteed to have the current results available when the residue register is read after initiating computation. The generator is capable of operating in three different modes to compute the residue on 6, 8, or

12 bit wide characters, following IBM binary synchronous communication conventions.

This report will serve as a progress report for those interested in project technical progress, and as a maintenance manual for those responsible for future system maintenance. Basic design discussions and objectives will be described first, followed by a brief overall equipment description with detailed logic explanations and programming considerations. Finally, maintenance software is included to aid in hardware debugging.

II. DESIGN DISCUSSION

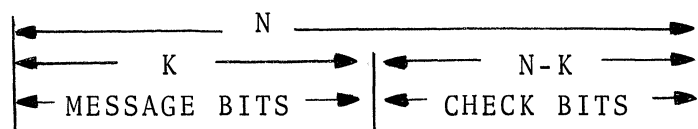
a. Error Checking

Probably some of the most important developments, in the area of error-detection and error-correcting codes, over the past decade have pertained to cyclic codes. Encoding procedures for these codes are relatively easily implemented in hardware using a shift-register with feedback connections, or they may similarly be emulated by software techniques.

This method of error checking, called cyclic redundancy checking (CRC), is always done at the receiving station and is computed over each physical message block, excluding only certain control characters under special circumstances. The CRC technique is a much more powerful means of block checking a message than is a longitudinal-redundancy check or mod-2 add over the message stream. It is especially useful in

detecting burst errors which may cause several successive bits or even whole characters of the message to be altered.

Algebraically a cyclic code is defined in terms of a generator polynomial $P(X)$ of degree $n-k$, where a message of k binary digits is encoded by appending $n-k$ binary digits as a check then transmitting the k information bits followed by the $n-k$ check bits. Thus, it is convenient to think of



these binary digits as the coefficients of a polynomial in the dummy variable X ; for instance the message 101011 is represented by the polynomial $X^5 + X^3 + X + 1$. To encode a message polynomial $G(X)$, it is divided by the generator polynomial $P(X)$, where the division is formed over the Galois field of two elements, consisting of the integers modulo two; that is, the field consists of the two elements 0,1. Carries are ignored. The remainder $R(X)$ from the above division becomes the check polynomial and is appended to the original message. Thus,

$$X^{n-K}G(X) = Q(X)P(X) + R(X)$$

where $Q(X)$ is the quotient and $R(X)$ the remainder resulting from the division of $X^{n-K}G(X)$ by $P(X)$. The message polynomial $G(X)$ is premultiplied by X^{n-K} to obtain a vector

for which the first $n-K$ components are zero (to allow subsequent addition of the residue), and the last k components, arbitrary information (message) symbols. Rewriting the above equation, and letting $F(X)$ equal the encoded message, we have

$$F(X) = X^{n-K}G(X) - R(X) = Q(X)P(X)$$

or since, in modulo two arithmetic, subtraction and addition are the same,

$$F(X) = X^{n-K}G(X) + R(X) .$$

In short, the code symbols are just the message polynomial expressed modulo the generator polynomial $P(X)$.

b. Design Objectives

In order to obtain a flexible interface for the cyclic redundancy generator (CRG) the following design objectives were set forth:

1. There must be generality without an extensive I/O instruction set required.
2. The PDP-8 accumulator (AC) should be automatically cleared after writing a CRG register to save that programming overhead.
3. The CRG should accommodate several different character sizes and be able to compute at least the IBM-compatible CRC-16 and CRC-12 checksums.

4. Since the generator requires a variable execution time depending on character size it seemed desirable to be able to stop the PDP-8 processor for the required time, rather than following the start command to the CRG with the maximum required number of NOP's.

These objectives were met with the resulting hardware constructed within 3/4 of a standard DEC 1943 mounting panel. In addition, minor modifications were required to the PDP-8 to allow the hold off, or extended pause, facility to be implemented. The allowed input character sizes to the CRG were restricted to only three widths consisting of 6,8, or 12 bits. The choice of a character size by the programmer results in the appropriate generator polynomial feedback taps being set up, as well as enabling the correct register gating paths for input and output.

III. SYSTEM DESCRIPTION

A general block diagram of the main CRG registers is shown in Figure 1. The character buffer is loaded with the incoming message character prior to computing the new residue. As this buffer is loaded a mode flip-flop is also set accordingly to either byte (8-bit character) or word (6- or 12-bit character) mode. The setting of this mode flip-flop determines which generator polynomial shall be applied to the residue register and also enables the correct data paths for subsequent residue register I/O.

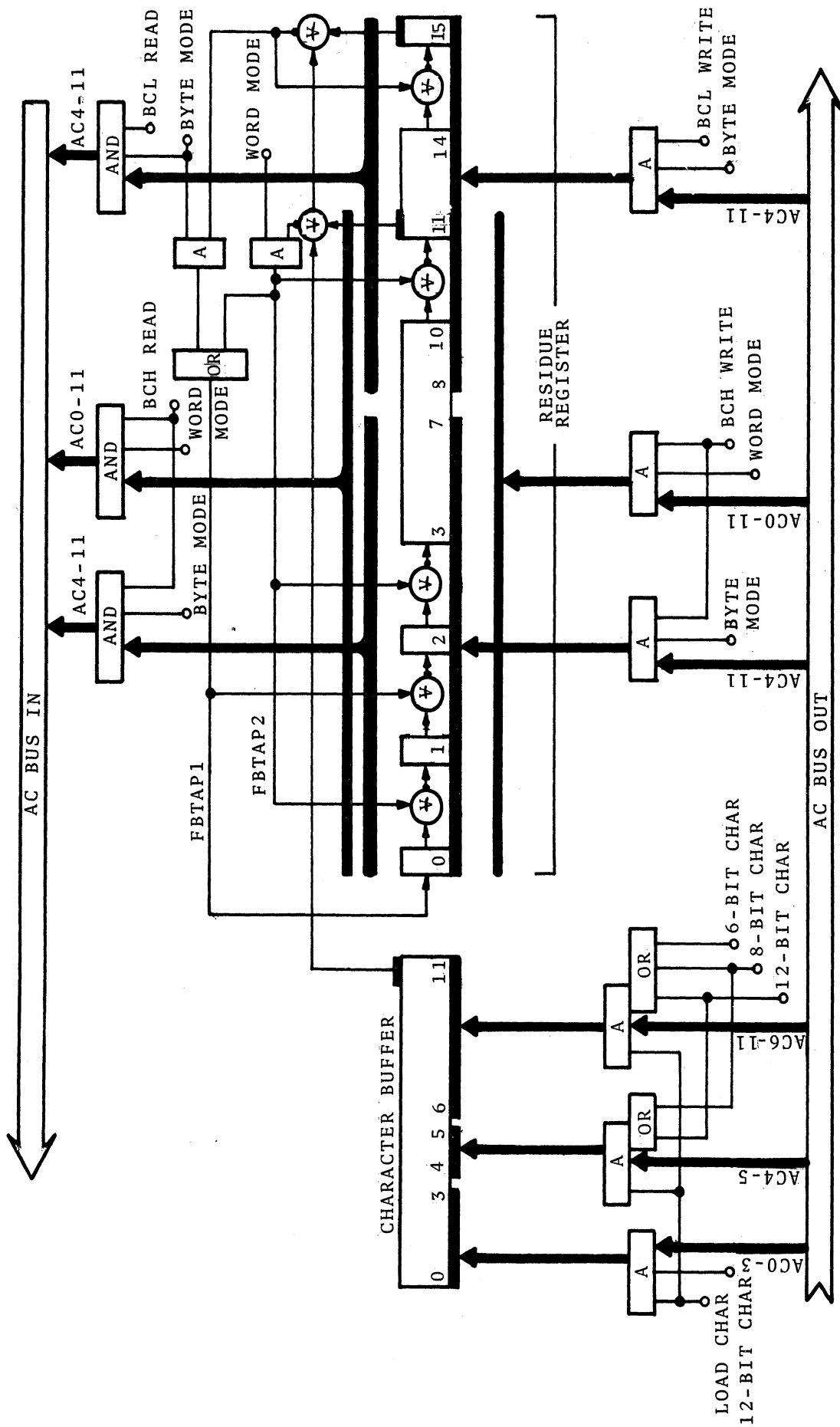


Figure 1. General Block Diagram.

I/O transfers to or from the PDP-8 AC are automatically routed from or to the residue register according to the current mode setting, with all data right-adjusted in the AC. The residue register contains six mod-2 adders between various stages for implementation of two different checking polynomials. The appropriate adders are enabled by the operating mode, set up at the time the input character is loaded. In the byte mode of operation the generator polynomial $[P(X)]$ used is

$$X^{16} + X^{15} + X^2 + 1 .$$

The resulting encoded message allows the receiver to detect any burst error of length 16 or less, as well as more than 99.9997% of all errors of greater length. The above polynomial has the prime factors $(X+1)$ and $(X^{15}+X+1)$. Figure 1.1 represents a simplified version of CRG register for CRC-16 block check accumulation.

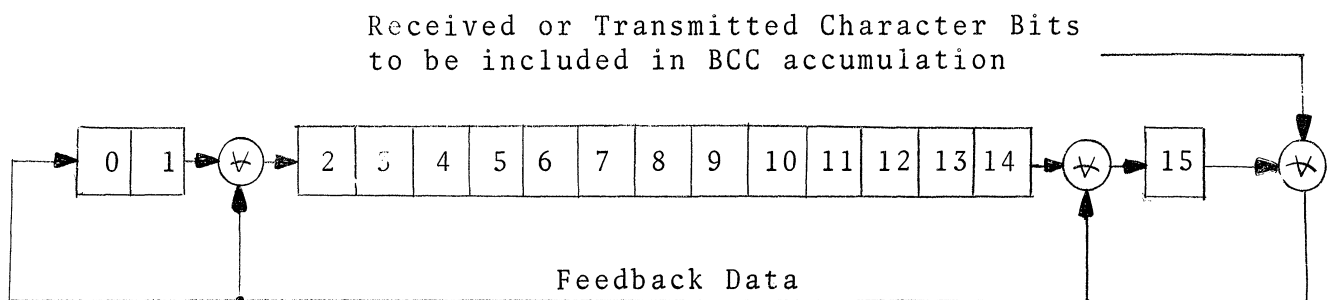


Figure 1.1. CRC-16 BCC Generation

When operating in the 6- or 12-bit word mode the CRG utilizes the generator polynomial

$$X^{12} + X^{11} + X^3 + X^2 + X + 1 .$$

This polynomial has the prime factors $(X+1)$ and $(X^{11}+X^2+1)$, and enjoys burst error detecting properties similar to the first polynomial for shorter length bursts. It will detect any burst-error length of 12 or less, and detect more than 99.995% of all bursts of greater length. Figure 1.2 illustrates the operation of the CRG register for CRC-12 block check accumulation.

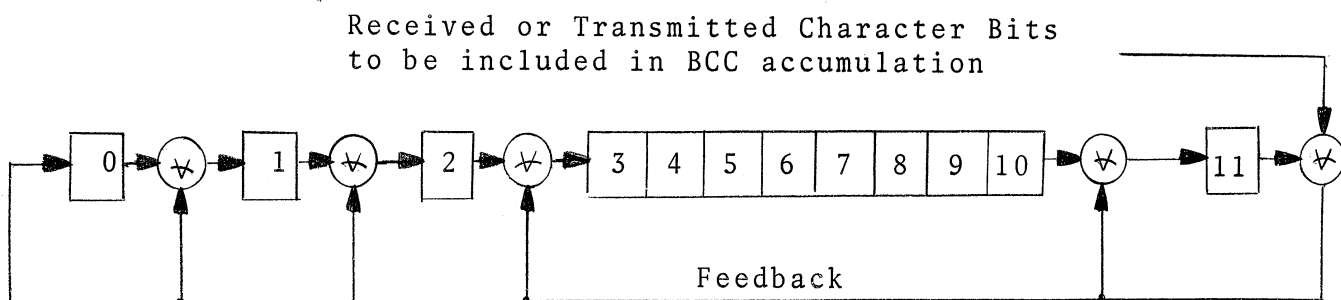


Figure 1.2. CRC-12 BCC Generation.

Table 1 summarizes the PDP-8 IOT assignments for this device. Only the residue register can be both read and written; the character buffer is written only. Note that the write into the residue register is a ones transfer or "inclusive-or," thus the register normally would be cleared before writing, however, circumstances may well dictate a need for ORing. The RD ,

CLR, and WR modifiers are assigned to the sequence of IOP pulses so that micro-operations are performed in the order listed.

TABLE I
PROGRAM MNEMONICS AND FUNCTIONS

FUNCTION	DVC ADR	MNEMONIC	IOP PULSE		
			1	2	4
Access to low-order part of block check (residue) register	54	BCL	RD	CLR	WR
Access to high-order part of block check register	55	BCH	RD	CLR	WR
Load character buffer, and set mode	56	LCM6	*		
		LCM8	*	*	
		LCM12	*	*	*
Compute Cyclic checksum	57	CCC	*		

Note that the character buffer is loaded in a pseudo-serial fashion with a few more bits of a character (assuming a character size greater than six) on each successive IOP pulse.

IV. PROGRAMMING AND CONTROL CONSIDERATIONS

The CRG is controlled by the resident PDP-8 supervisor via four sets of IOT instructions. These were summarized in Table 1 in the previous section. Since the device will be used in a multiprogramming environment, the hardware design was tailored in a direction that allowed rapid execution of code, in order that results could be rapidly obtained and the CRG and DSR program freed for other users. For this reason the register load instructions result in the PDP-8 AC being cleared so that another word may be fetched as rapidly as possible, without necessitating a prior clear of the AC. Similarly, the CRG automatically "stops" the PDP-8 in the PAUSE state while the new checksum is being generated. As soon as results are available, the PDP-8 is allowed to execute the next instruction, which would normally be a READ of the residue register to obtain the new checksum just generated. This technique allows the results to be obtained in the shortest possible time, without the need for NOP's or JMP *-1 loops, to timeout the computation period required.

The first thing required of the program, by the CRG, is information about the subsequent mode of operation, that is, what check polynomial should be used and what gating paths should be set up for transfers into and out of the residue register. The programmer must therefore load the character buffer with the incoming character, since this act sets the CRG mode flip-flop, and determines all following actions

until another character of a different mode is loaded. The character load and mode set instruction has been designated "LCM" for Load Character and set Mode. It is modified by one of three digit suffixs, namely 6, 8, or 12 as a reminder of the character size. These three modifiers are assigned IOT pulses 1, 3, and 7 respectively, since each succeeding pulse loads the character buffer in an incremental pseudo-serial fashion. After the character buffer has been loaded, the programmer is free to load the residue register in any order convenient. Gating to and from this register is automatically routed from the appropriate bits in the AC depending on the mode setting. Note that all information is assumed to be right-justified in the AC on input, and is placed thusly on output. As the 8-bit character-oriented checking polynomial is of degree 16, a 16-bit residue register is required to compute the checksum. The two bytes of this register are addressed via the BCL for Block Check Low instruction, which fetches the right-most, or low-order, 8 bits of the register, and by the BCH, for Block Check High instruction, which handles the 8 high-order, or left-most, bits. Since the 6- and 12-bit character sizes utilize a checking polynomial which is 12 bits wide, a whole PDP-8 word is used to contain the results. Further, since the input to the residue register is on the high-order side, this register may be read and written via a single transfer between the host processor, using the high-order gating instruction BCH. After all registers have been loaded,

execution of a new checksum is started by the CCC instruction, which is a mnemonic for Compute Cyclic Check. As mentioned previously, this instruction also halts the PDP-8 until the new checksum is available. The instruction following the CCC would normally be a read to obtain the new contents of the residue register. Figure 2 demonstrates some sample programming sequences which compute a checksum under either mode of operation. The programs are intended to be imbedded in a multi-programming environment, thus the previous residue contents are assumed to be pointed to by the two indirect vectors which would be set up prior to entering these sections of code. By convention, the residue is always reset to an all-zero value before computing a check over a new message.

V. DETAILED LOGIC DISCUSSIONS

IOP Decoding and Device Selection

Figure 3 illustrates the IOT and device address decoding for the RCG. Since there are four device addresses required to control all registers and to start checksum execution, this field of four is decoded as a group by the 4-input gate at module position A08. Its output is labeled BLK+ and signifies selection of one of the four subdevices within the block. The block select signal is inverted and used to enable the three IOP gates used for selection and buffering of the IOP pulses. In most instances a device ANDs one of the

ASSEMBLER DEFINITIONS

```
*
*
*
RD      OPD      1
CLR     OPD      2
WR      OPD      4
BCL     OPD      6540
BCH     OPD      6550
LCM6    OPD      6561
LCM8    OPD      6563
LCM12   OPD      6567
CCC     OPD      6571
*
*
*           8-BIT MODE OF OPERATION
*
*           ASSUME UCBPTH POINTS TO HIGH-ORDER 8 BITS
*           OF CHECKSUM, AND UCBPTL POINTS TO THE
*           LOW-ORDER BITS OF THAT SUM IN THE UCB.
*
TAD     CHAR      FETCH NEW INPUT CHARACTER
LCM8    .         SET 8-BIT MODE AND LOAD CHAR BUF
TAD*    UCBPTH    NOTE AC CLEARED BY LCMB INSTRUCTION
BCH     CLR+WR    WRITE IS AN INCLUSIVE OR
TAD*    UCBPTL    BCH INSTRUCTION ALSO CLEARS AC
BCL     CLR+WR    THIS CLEARS AC ALSO
CCC     .         START COMPUTING NEW CHECKSUM
BCH     RD        GUARANTEED TO BE DONE BY NOW
DCA*    UCBPTH    NEW CHECKSUM BACK TO UCB
BCL     RD
DCA*    UCBPTL

*           ***** ALL DONE AT THIS POINT *****
*
*
*           6- OR 12-BIT MODE OF OPERATION
*           THEY DIFFER ONLY IN THAT 6-BIT
*           MODE IS SET BY LCM6, WHILE
*           THE 12-BIT MODE IS SET BY LCM12.
*
TAD     CHAR
LCM12
TAD*    UCBPTH
BCH     CLR+WR    NOTE BCH MUST BE USED IN THIS MODE
CCC
BCH     RD
DCA*    UCBPTH

*           ***** ALL DONE *****
```

Figure 2. Cyclic Check Computer—Example Routines.

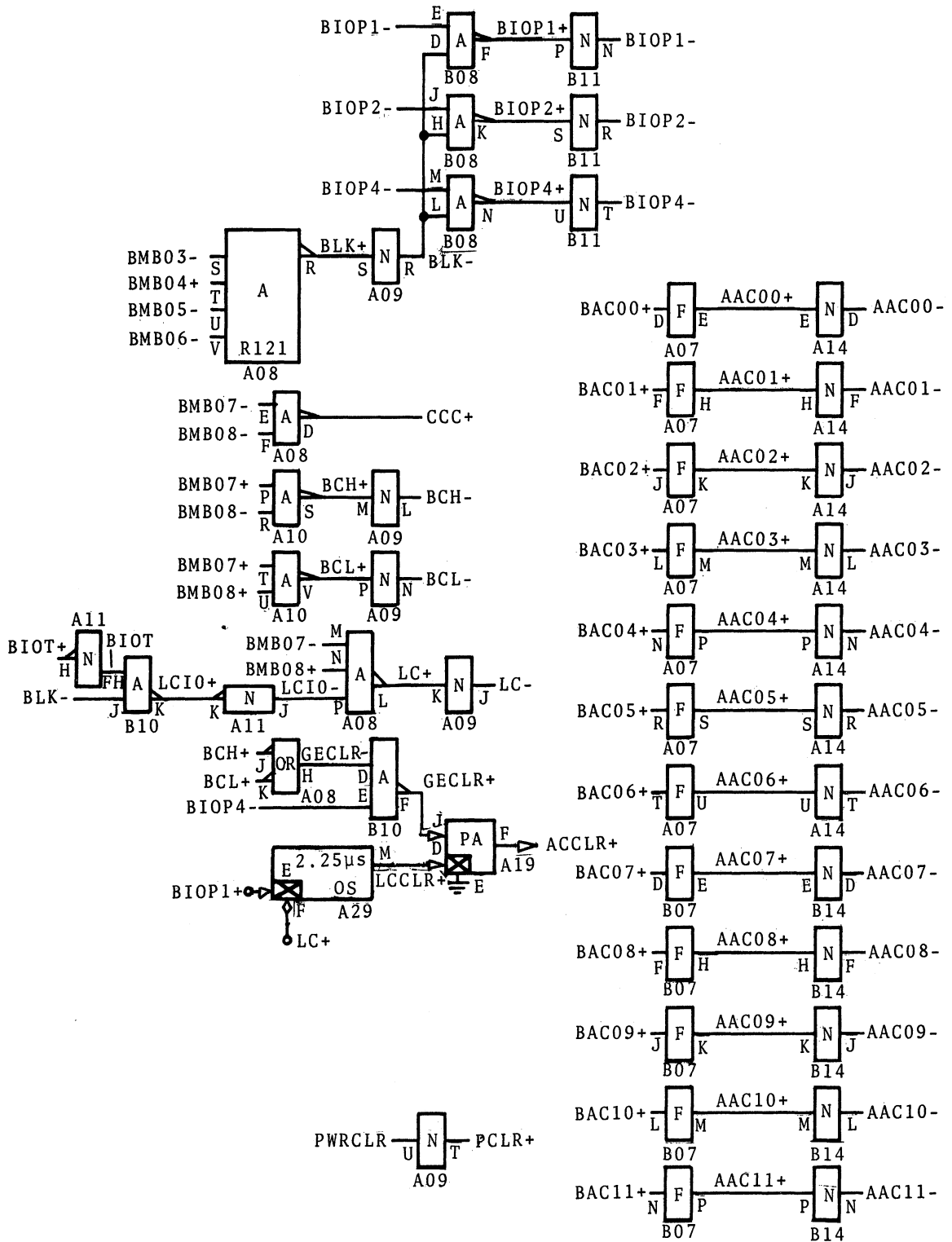


Figure 3. IOP Address Decode AC Buffers.

buffered IOP pulses with its own subdevice select to form a useful signal. Each of the block subdevices, except the character buffer (LC signal), is enabled by the select out from one of the 2-input gates located below the block select decoder on the drawing. Since the character buffer clear signal uses a PDP-8 basic timing pulse instead of an IOP pulse, provisions had to be included in this subdevice selection to insure that an IOT instruction was really being executed by the processor. Thus the LC signal goes true only if the BLOCK is selected and the subdevice is selected and an IOT instruction is being executed by the PDP-8. The IOT instruction signal is derived internally in the PDP-8 and is brought out as an extra lead on the high-order AC input cable. This is further discussed in the latter part of this section under the heading, "PDP-8 Added Circuitry." As mentioned above, the subdevice selection outputs are further ANDed with IOP pulses at the various devices themselves to form the desired gating or control functions.

The PDP-8 AC clear signal is formed at the lower left-hand corner of the drawing by ANDing IOP4 with a residue register select, OR by the recovery of level LCCLR+ which is formed by a character buffer select (LC+) and IOP1+. This level transitions from -3 volts to ground 2.25 microseconds after the leading edge of IOP1 and triggers the AC clear, pulse amp. If the PDP-8 is adjusted according to manufacturer specifications, IOP4 should follow IOP1 by 2 microseconds,

however, there may be a $\begin{matrix} +20 \\ -40 \end{matrix}$ percent variance on that figure, thus the delay is set to clear the AC after the latest expected occurrence of the leading edge of IOP4. If in doubt, this delay should be set high (longer) rather than low. The AC may even be cleared as late as the next computer cycle, as the soonest it could be manipulated would be at T1 time, if the next instruction were one of the operate group micro-instructions.

Note that although in the one case the AC is cleared as the residue register is being written, due to the storage time of the register DCD gate inputs there are no problems as the "rug is pulled out from under." When loading the character buffer, it is not possible to use IOP4 to clear the AC, since the buffer load command may be executed using only one, two, or all three of the IOP pulses. Thus the AC clear signal LCCLR+ is developed just after IOP4 time by the delay triggered by IOP1.

The 12 buffers on the lower right-hand corner of the drawing are used to obtain both polarities of the AC output signals. Note that the input buffers are W500 emitter-followers which have minimum loading effects upon the PDP-8 AC. These were used since there are several other DEC-provided devices across the AC lines, as well as the Data Concentrator.

Character Buffer

The 12-bit character buffer is documented in Figure 4. The discerning observer will note that it actually contains 13 bits. The thirteenth is used as described below for a flag bit to stop shifting operations. The character buffer is loaded from the AC in a pseudo-serial fashion by the execution of one or more contiguous IOP pulses. This register is automatically cleared prior to loading by the CCLR+ line. This clear is developed by the selection of this subdevice and the occurrence of timing pulse T1. Recall that the first IOP pulse does not arrive until 500 nanoseconds after T1 (as developed within the PDP-8), thus again there is adequate time for the register to stabilize before any operations are attempted upon it. IOP pulse 1 and the selection of this subdevice load the low-order 6 bits from the AC into the character buffer and set C05 as a shift stop flag. The setting of this flag always ensures that this register contains a non-zero value. Shifting is stopped by a detection circuit (described within Figure 9) that determines when the character buffer has gone to zero, indicating that the loaded character has been shifted out. If the buffer is being loaded with a character size greater than 6 bits, then additional IOP pulses will follow IOP1. The second IOP pulse (IOP2) causes the contents of AC bit 5 to be jammed into the previously set C05 flip-flop, and AC04 to be strobed into C04, which is still cleared at this time. At the same time C03 is now set as the

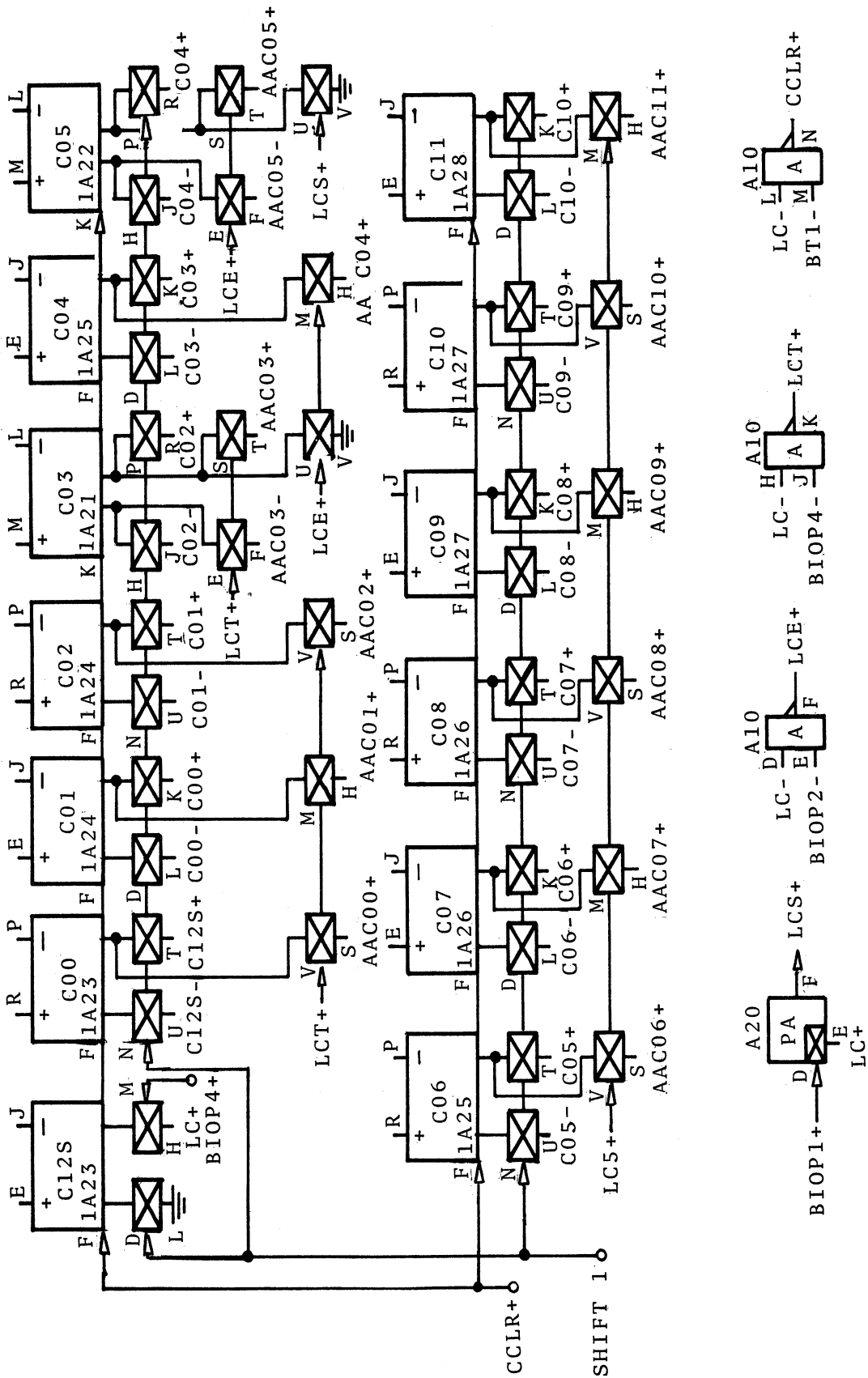


Figure 4. Character Register.

new shift stop flag. Finally, if IOP4 occurs, it jams AC03 into C03 and strobes AC00-AC02 into C00-C02. As in all other cases the bit immediately to the left of the most significant character bit is set as the shift stop flag. Thus flip-flop C12S is set as the flag.

All of the control pulses required for this register are developed at the bottom of Figure 4. The signal LCS+ is produced by a pulse amplifier, due to the DCD gate loading of seven flip-flops. This loading exceeds the driving capability of a single inverter stage.

Residue Register Control

The read, write, and clear signals for the checksum register are detailed in Figure 5. Note that there are three sets of control signals developed while there is only one residue register. The various signals are required to handle the gating and control which must be present for the byte and word modes of operation. The "1" suffixed group is responsible for controlling the low-order byte of the residue register during byte mode operations, while the "3" suffixed group is responsible for the high-order byte in the same mode. The "2" suffixed group is used during 12-bit word operations on the residue register. Since the word operations overlap both bytes of the residue register, some of the operations can be made common to both modes; thus the word mode clear

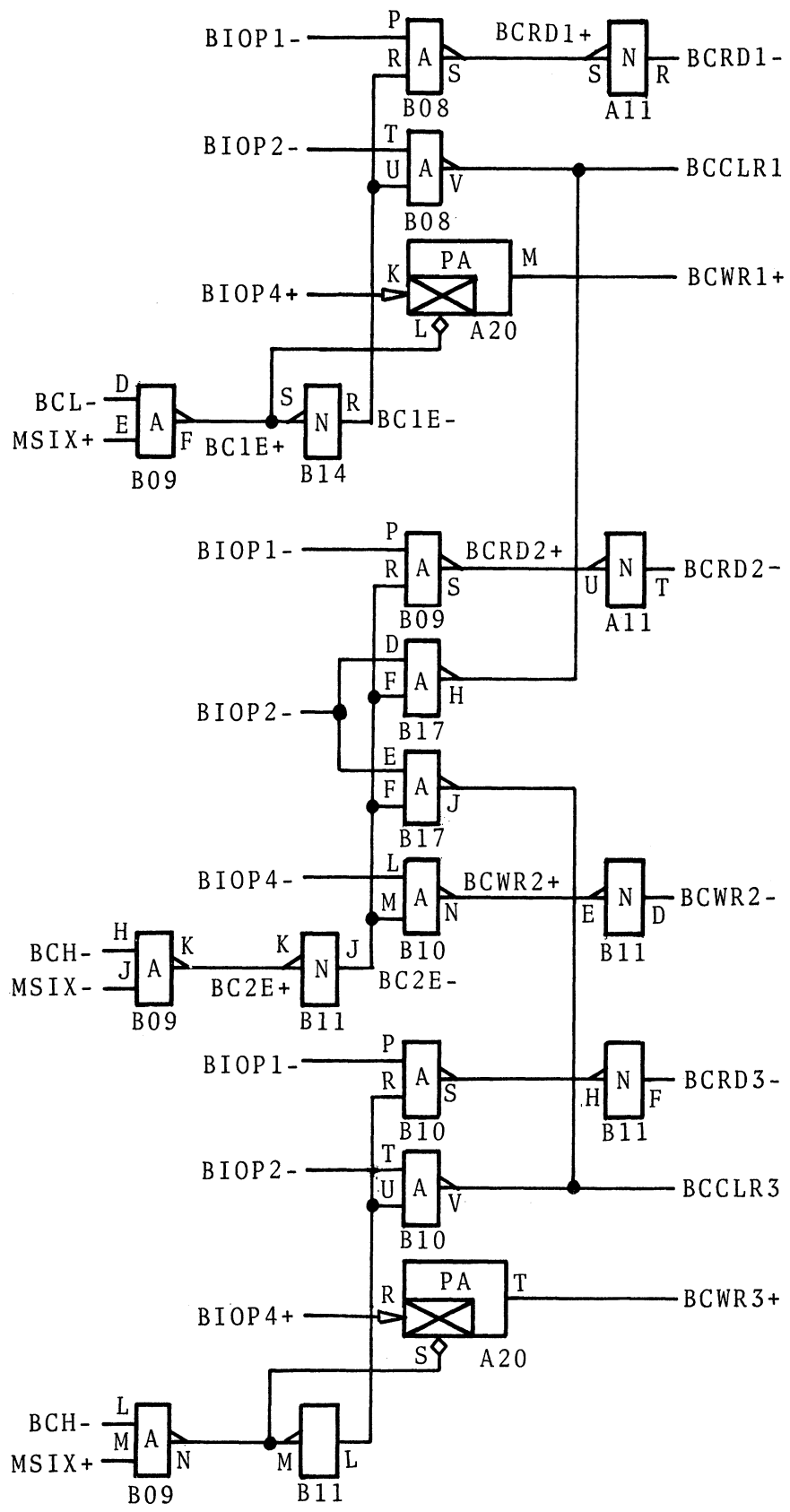


Figure 5. Residue Register Control.

enables both the BCCLR1 and BCCLR3 lines to clear the entire register. Since the four low-order bits of the residue register are not used by word mode operations, they can be cleared with no consequence.

The MSIX+ and MSIX- signals appearing at the left-hand edge of the drawing are the outputs from the mode flip-flop. When MSIX- is true (-3 volts) the CRG is operating in the 6- or 12-bit (word) mode. Recall that this flip-flop is set by the execution of the LCM (Load Character and set Mode) instruction.

In order to conserve on module space, the residue register was constructed from R205 flip-flops, which consist of a pair of flip-flops on a single board. Due to the pin restrictions, however, there are only enough connections to allow three DCD gates per flip-flop. Since one pair is required for the shifting operation and the remaining gate is used for byte mode loading operations, another method must be used to load the register when operating in the word mode. Thus the 12 high-order bits of the register are loaded via a direct collector set through the gates shown in Figure 6. The other set of gates shown in this drawing are used to read the contents of the 12 high-order bits of the residue register into the AC. These two sets of gates are enabled by the appropriate set of signals derived from Figure 5. The inputs and outputs are configured for a direct transfer between the AC and residue register.

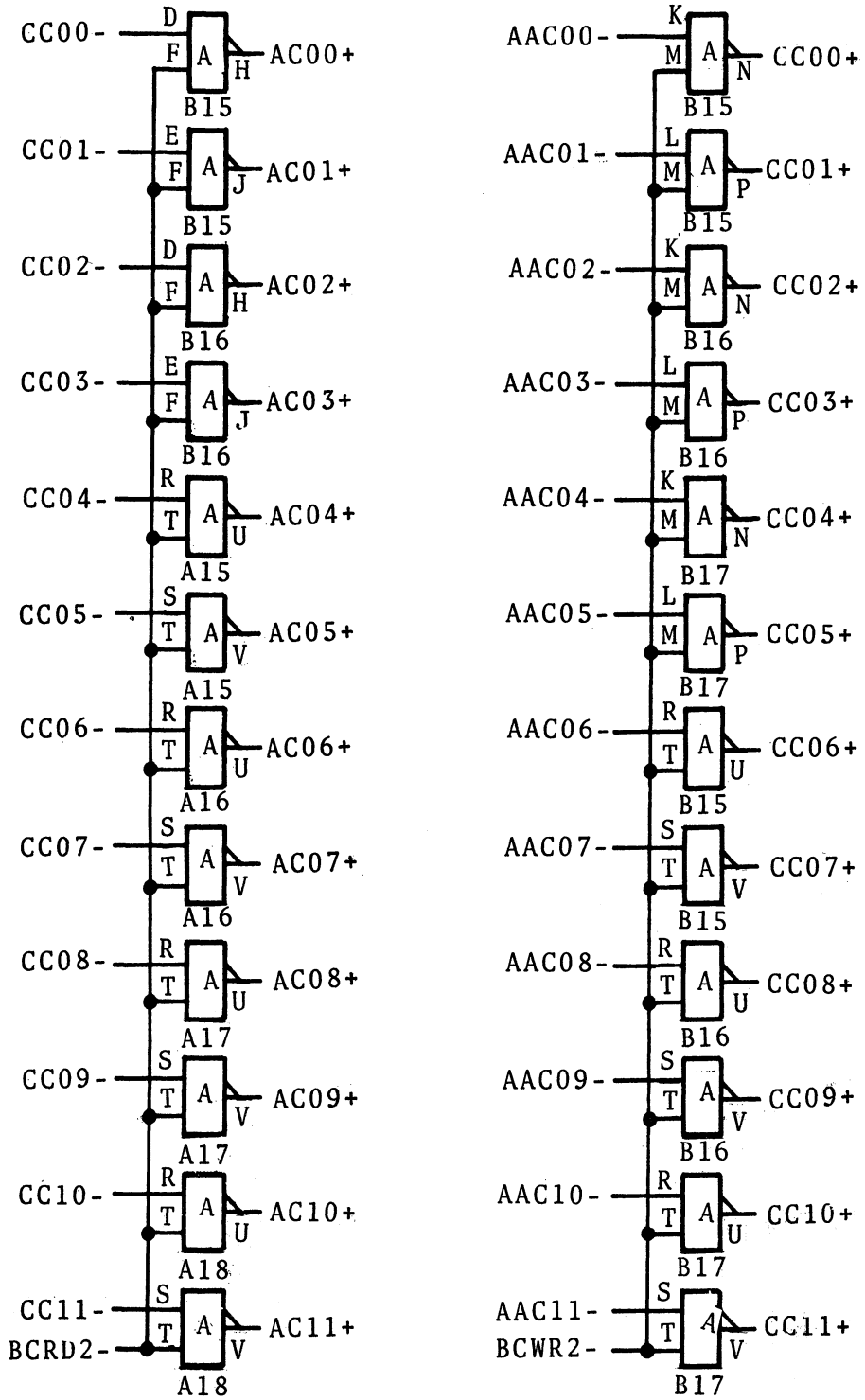


Figure 6. Residue Register IN/OUT Gating

Residue Register Mod 2 Adders

The seven mod 2 adders required to implement the various checking polynomials are diagrammed in Figure 7. According to normal signal-naming conventions, the + or - sign following a circuit name indicates the respective voltage level which will be present when the associated lead is "true" or contains a logical "1." Thus FBTP1- is at a negative (-3 volt) level when a logical 1 has been inputted through one of the 2-input gates.

The adder in the upper left corner (CC02** outputs) is shared by both the CRC-12 and CRC-16 block check computation, and resides between the second and third stages of the residue register. The two 2-input NAND gates (FBTP1* outputs) provide the required gating into this adder. The MSIX signals, derived from the mode flip-flop, enable the appropriate gate with a negative level. When the generator is operating in the word (6/12-bit) mode, the MSIX- signal is at -3 volts. Since an enable signal, NANDed with a positive logic level from CRC12 or CRC15, produces a negative output (FBTP1-) for a "1" input, the other input to the adder, CC01-, is taken with a negative assertion level from the previous stage of the residue register. This results in a "1" on either adder input being represented by a minus level. This is done only for ease in logic understanding, since dissimilar levels on the inputs could be equally well used if the output leads from the adder were interchanged. Note that the FBTP1 signals also go to the

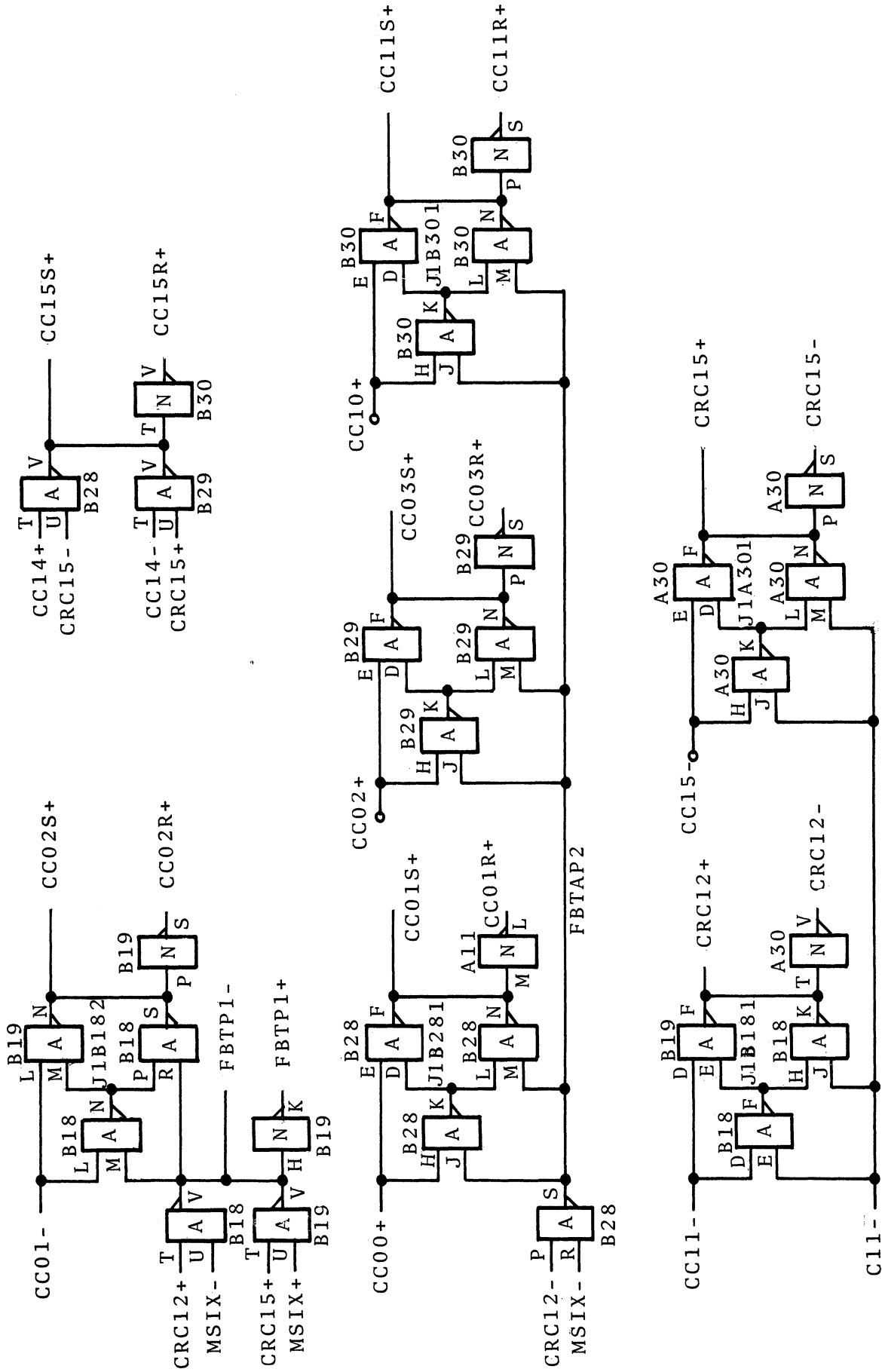


Figure 7. MOD 2 Adders.

first stage of the residue register. The bottom two adders on the figure have similar input conventions, while the middle three are just the complement with positive levels (ground) representing a logical "1" input.

The upper right adder precedes the last stage of the residue register, and is used only when operating in the CRC-16 (byte) mode. It is a simple circuit which, although always active, does not interfere with word mode operation since the last (right-most) bits of the CRG are not used by this mode.

The center three adders in Figure 7 are used only during word mode operation, and are thus disabled unless that is the current mode. Disabling is accomplished via the 2-input NAND gate whose output is FBTAP2. When operating in the word mode, MSIX- is at a negative level and the gate feeds the output from the CRC-12 input adder through to the interstage adders on the FBTAP2 bus, with a logical "1" represented by a positive level. When the gate is disabled, FBTAP2 is held at a negative level (which corresponds to a logical "0" condition). Since an exclusive-or on two variables, where one is identically zero, results in an output which is equivalent to the other input variable, the interstage adders act as direct connections between stages when FBTAP2 is negative. This of course is what is required during byte mode operation.

The bottom two adders form the summing junction for addition of the least significant bit of the residue register

with the incoming character bit. The results of this addition go to the appropriate feedback lines that drive the inter-stage adders.

Residue Register

Figure 8 is the actual residue register complete with shift connections and input/output gating for byte mode operation (word mode gating was documented in Figure 6). Note that the design of the register results in two essentially identical sections, each 8 bits wide. As mentioned before, the only line aside from the shift pulse that is common between the byte and word mode is the register clear, which is enabled for both prior to a word mode write. The pulse line for this register is separated from the character buffer shift due to the driving capabilities of a pulse amplifier. A total of 29 flip-flops is a few more than can be comfortably driven.

Shift Control and Mode Storage

Figure 9 is the last figure for the CRG, and is composed of two sections. To the left appears the shift control circuitry which determines the number of shifts that will be produced to execute the checksum generator division. The RUN flip-flop is set by the occurrence of a CCC enable level and IOPl. When this FF is set, RUN- enables the R401 clock which then starts producing a pulse train of 100 nanosecond pulses at a 2MHz repetition rate. This clock output is buffered by

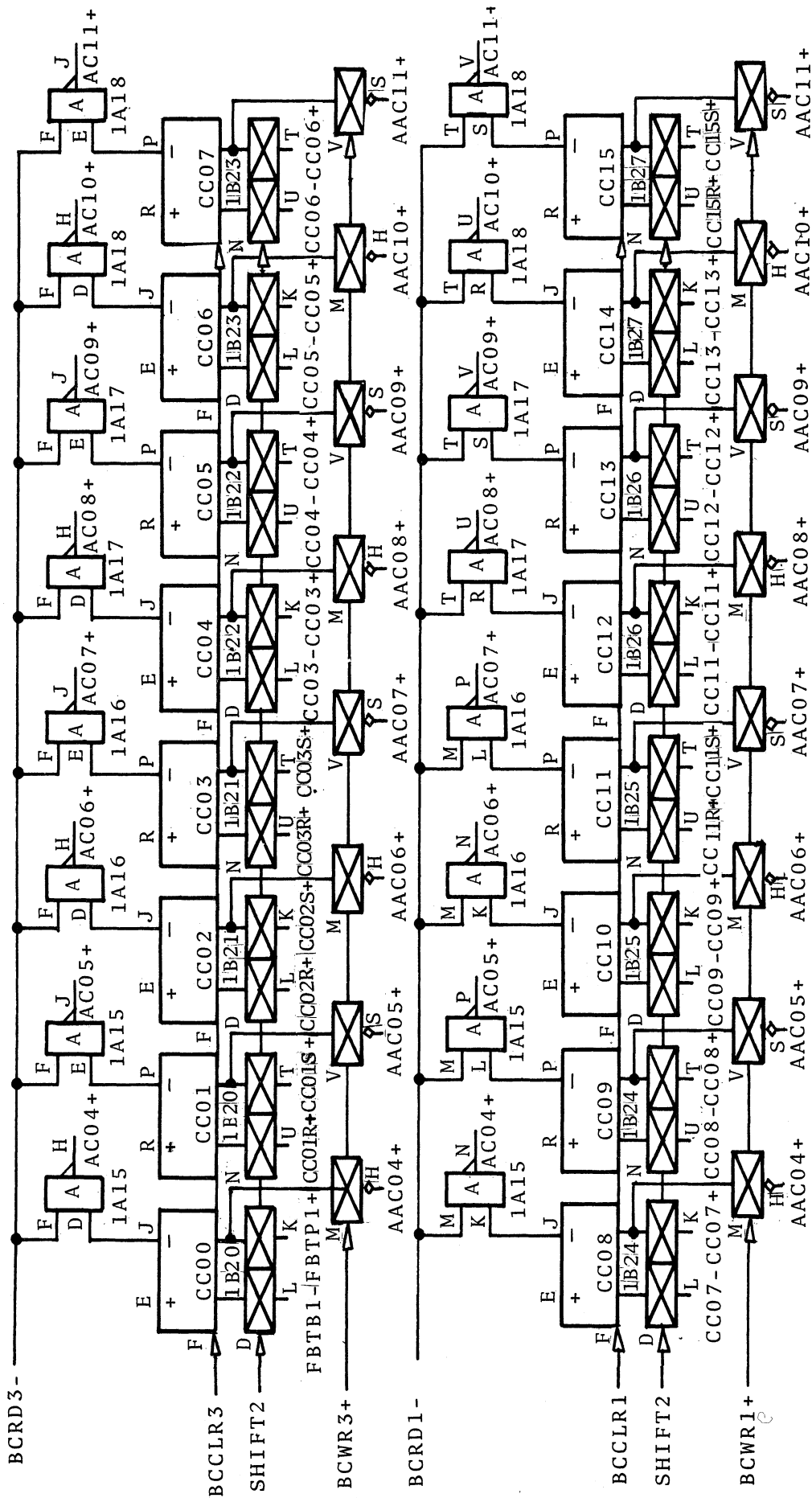


Figure 8. Generator Shift Register.

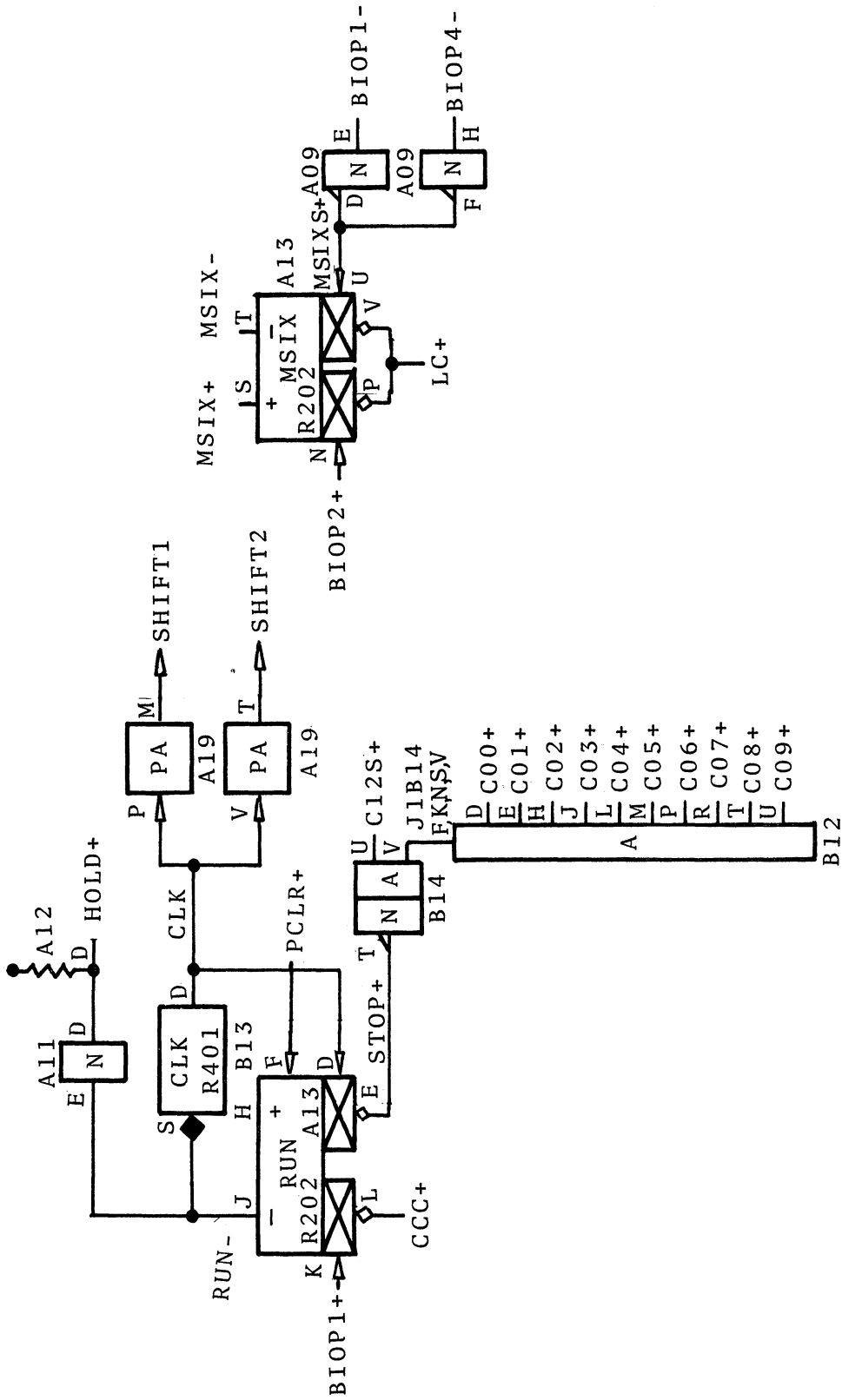


Figure 9. Run and Mode Control.

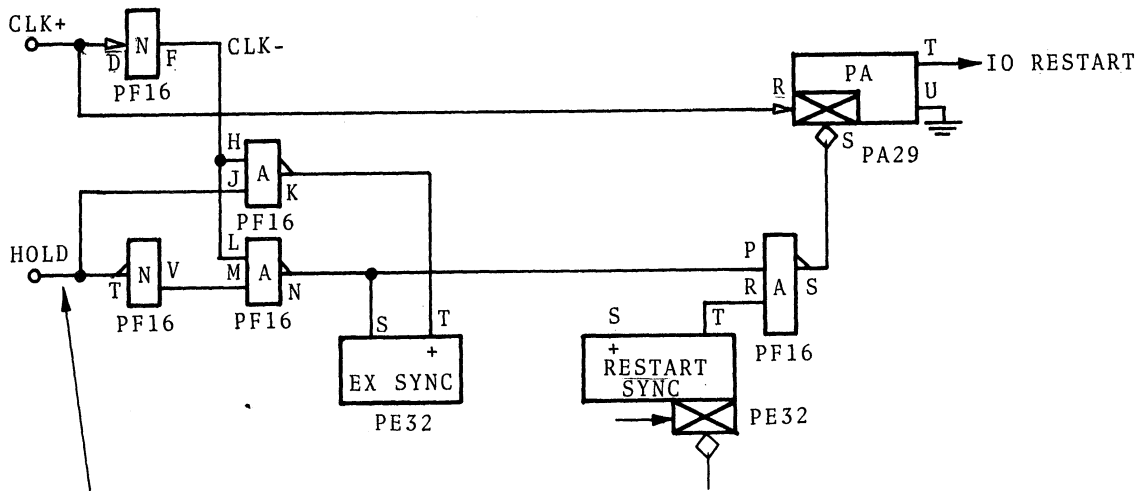
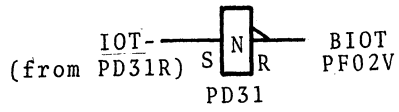
two pulse amplifiers which produce SHIFT1 going to the character buffer, and SHIFT2 which is routed to the residue register. As mentioned above, this is required due to the DCD gate loading on the shift line. The total loading on that line exceeds the driving capability of one pulse amp. At each clock pulse the RUN flip-flop is strobed in order to stop the clock if the shifting operation is finished. The flip-flop will be reset though, only if the 11-input AND gate sampling C12S through C09 finds that those bits are all zero. Since a shift stop flag was set when the character buffer was loaded, we are guaranteed to have a non-zero value in that register until the loaded character is shifted far enough. Also since zeros are shifted into the left end of the register, we know that it must eventually assume a zero value. Note that the last (let us say n-th) shift required of this register results in the most significant bit of the character being transferred from the low-order position of the character buffer to the high-order position of the residue register. When this is done the shift stop flag will then be resident in the least significant bit (C11) of the character buffer. Just prior to this (n-th) shift, the stop bit was in C10, and just prior to the n-1 shift, it was in position C09. Thus, before the n-1 shift, C12S through C08 are equal to zero and C09 contains a one. On the n-1 shift the flag bit moves to position C10 and leaves C12S to C09 all set to zero. This results in STOP+ going to ground, which enables the DCD gate on the reset

side of RUN FF. The next CLK pulse then resets RUN and the shifting process is halted. Note that the entire time that RUN was set the buffered output from RUN- holds the HOLD line to ground. This signal returns to the PDP-8 where it maintains the CPU in the PAUSE state until HOLD returns to -3 volts.

The MSIX flip-flop is set by the same instruction that loads the character buffer, and serves as a 1-bit memory element to control the residue register I/O gating and feedback taps. Recall that IOP1 loads a 6-bit character while IOP1 and 2 load an 8-bit character, and IOP 1, 2, and 4 are required for a 12-bit character. The last IOP produced for an 8-bit character load is thus IOP2, which clears the MSIX flip-flop. If the character loaded is 6 or 12 bits in width, then IOP1 or 4 is the last produced and MSIX is set.

PDP-8 Added Circuitry

The circuit diagrammed in Figure 10 has been added to the PDP-8 processor used with the Data Concentrator and fulfills several functions. First it allows external devices to request a "long" (would you believe infinite) I/O cycle. The IO RESTART pulse is normally produced within the PDP-8 at the same time that the IOP4 pulse amplifier is strobed. This results in a T2 pulse being produced at the next CLK pulse and normal instruction execution is then resumed. The additional circuitry prevents the IO RESTART pulse amp from being enabled, however, if the HOLD line is at a ground level.



Note: Clamped Load Provided
Back on Cyclic Check
Panel.

R113 Added at PF16
R203 at PE32. Used spare 1/3 of Module

See also: I/O CONTROL
DEC DRAWING D-8P-0-10J

Figure 10. Additions to PDP-8 CPU.

To accomplish this, a R113 module was added at position PF16 and a left-over 1/3 of a R203 flip-flop at position PE32 was used. The HOLD line is inverted and the two signals are then run to two 2-input NAND gates which when strobed by CLK- perform a jam-transfer of the contents of the HOLD line into the EX SYNC flip-flop. HOLD is thus sampled every 750 nano-seconds by the CLK line. If the HOLD line had been down when RESTART SYNC was set then EX SYNC- would be at ground and an IO RESTART pulse would not occur. The next clock pulse, after HOLD is released, sets EX SYNC and enables the DCD gate of the IO RESTART pulse amp since the RESTART SYNC FF is also set. The next clock pulse after that produces an IO RESTART and the processor is off and running again.

The second circuit documented provides an external device with a positive signal that the processor is currently executing an IOT instruction. The IOT- signal is derived from an inverter which is driven directly from the output of the Instruction Register decoder in the PDP-8. This negative level is then buffered and inverted to form a positive assertion level signal called BIOT, which may be sensed by an external device to determine when the processor is actually executing an IOT instruction. This permits the use of the processor basic timing signals T1 and T2 which appear at the I/O interface but which normally may not be used with programmed data transfers due to the impossibility of determining whether

the contents of the Memory Buffer (from which device address decoding is obtained) contains a legal IOT instruction, or is actually a data word, or an external memory access operation (Data Break). The inverter shown on the diagram was acquired from an unused 1/7 of a R107 module at location PD31 in the PDP-8 mainframe.

APPENDIX A

UTILIZATION MODULE LIST AND CIRCUIT NAME MAP

APPENDIX A

UTILIZATION MODULE LIST AND CIRCUIT NAME MAP

Wire-wrap output documentation is reproduced on the following pages. First is a circuit map of the DEC 1943 mounting panel upon which the checksum adapter has been built. Note that all module outputs are denoted by an asterisk on the drawing. The module types are noted at the top of each module position.

Finally actual wire-wrap instructions are reproduced as an aid to device reproduction. The concatenated circuit list documents all pin connections for each circuit name. The connections are serially listed for each circuit in the order that they are wrapped. The output lists represent the order that each wire is actually wrapped on a bay. The shortest wires are put on first, followed by increasingly longer wires until the longest is finally placed. The five columns of the output list represent, in order:

1. The circuit name
2. The wrapping level, where 1 is the lowest (against the panel), and 2 the highest (a second-level wrap is placed above a one-level wrap on a pin).
3. The pin-to-pin length of the wire in quarters of an inch. If, however, a wire run is straight down a horizontal row of pins, then it may be connected via a solder-on bus strip rather than a series of wire-wraps. In this case the wire run is identified by BUS, and the run includes many rather than just two points.

4. The first pin location of the wire run.
5. The last pin location of the wire run.

**** CYCLIC CHECKSUM COMPUTER ****

CONCATENATED CIRCUIT LISTS

AAC00+	1A07E,1A14E,1A23S
AAC00-	1A14D,1B15K
AAC01+	1A07H,1A14H,1A24H
AAC01-	1A14F,1B15L
AAC02+	1A07K,1A14K,1A24S
AAC02-	1A14J,1B16K
AAC03+	1A07M,1A14M,1A21T
AAC03-	1A21F,1A14L,1B16L
AAC04+	1A07P,1A14P,1B20H,1B24H,1A25H
AAC04-	1A14N,1B17K
AAC05+	1B20S,1B24S,1A22T,1A14S,1A07S
AAC05-	1A22F,1A14R,1B17L
AAC06+	1A25S,1B25H,1B21H,1A14U,1A07U
AAC06-	1A14T,1B15R
AAC07+	1B07E,1B14E,1B21S,1B25S,1A26H
AAC07-	1B14D,1B15S
AAC08+	1A26S,1B26H,1B22H,1B14H,1B07H
AAC08-	1B14F,1B16R
AAC09+	1B07K,1B14K,1B22S,1B26S,1A27H
AAC09-	1B14J,1B16S
AAC10+	1A27S,1B27H,1B23H,1B14M,1B07M
AAC10-	1B14L,1B17R
AAC11+	1B07P,1B14P,1B23S,1B27S,1A28H
AAC11-	1B14N,1B17S
AC00+	1A05C,1A06D,1B15H
AC01+	1A05E,1A06E,1B15J
AC02+	1A05H,1A06H,1B16H
AC03+	1A05K,1A06K,1B16J
AC04+	1A05M,1A06M,1A15H,1A15N,1A15U
AC05+	1A05P,1A06P,1A15P,1A15J,1A15V
AC06+	1A05S,1A06S,1A16H,1A16N,1A16U
AC07+	1A05T,1A06T,1A16P,1A16J,1A16V
AC08+	1A05V,1A06V,1A17H,1A17N,1A17U
AC09+	1B05D,1B06D,1A17P,1A17J,1A17V
AC10+	1B05E,1B06E,1A18H,1A18N,1A18U
AC11+	1B05H,1B06H,1A18P,1A18J,1A18V
ACC+	1A19J,1B10F
ACCLR+	1B05P,1B06P,1A19F
BAC00+	1A01C,1A02D,1A07D
BAC01+	1A01E,1A02E,1A07F
BAC02+	1A01H,1A02H,1A07J
BAC03+	1A01K,1A02K,1A07L
BAC04+	1A01M,1A02M,1A07N
BAC05+	1A01P,1A02P,1A07R
BAC06+	1A01S,1A02S,1A07T
BAC07+	1A01T,1A02T,1B07D
BAC08+	1A01V,1A02V,1B07F
BAC09+	1B01D,1B02D,1B07J
BAC10+	1B01E,1B02E,1B07L
BAC11+	1B01H,1B02H,1B07N
BC1E+	1B09F,1B14S,1A20L
BC1E-	1B14R,1B08R,1B08U
BC2E+	1B09K,1B11K
BC2E-	1B17F,1B11J,1B10M,1B09R,1B09U

***** CYCLIC CHECKSUM COMPUTER *****

BC3E+	1A20S,1B11M,1B09N
BC3E-	1B11L,1B10R,1B10U
BCCLR1	1B27F,1B26F,1B25F,1B24F,1B17F,1B08V
BCCLR2	1A20P,1A20V,1B09V
BCCLR3	1B23F,1B22F,1B21F,1B20F,1B17J,1B10V
BCH+	1A08J,1A09M,1A10S
BCH-	1A09L,1B09H,1B09L
BCL+	1A08K,1A09P,1A10V
BCL-	1A09N,1B09D
BCRD1+	1A11S,1B08S
BCRD1-	1A18M,1A17M,1A16M,1A15M,1A11R
BCRD2+	1A11U,1B09S
BCRD2-	1A18T,1A17T,1A16T,1A15T,1B15F,1B16F,1A11T
BCRD3+	1B11H,1B10S
BCRD3-	1A18F,1A17F,1A16F,1A15F,1B11F
BCWR1+	1A20M,1B27M,1B26M,1B25M,1B24M,1B24V,1B25V,1B26V,1B27V
BCWR2+	1B11E,1B10N
BCWR2-	1B17T,1B16T,1B15T,1B15M,1B16M,1B17M,1B11D
BCWR3+	1A20T,1B23M,1B22M,1B21M,1B20M,1B20V,1B21V,1B22V,1B23V
BIOP1+	1A29E,1A20C,1A13K,1B08F,1B11P
BIOP1-	1A09E,1B08P,1B09P,1B10P,1B11N
BIOP2+	1A13N,1B08K,1B11S
BIOP2-	1A10E,1B17D,1B17E,1B11K,1B1CT,1B09T,1B08T
BIOP4+	1B11U,1B08N,1A20K,1A20R,1A23M
BIOP4-	1A09H,1A10J,1B10E,1B10L,1B11T
BIOT+	1B05V,1B06V,1A11H
BIOT-	1A11F,1B10H
BLK+	1A08R,1A09S
BLK-	1A09K,1B10J,1B08D,1B08H,1B08L
BMB00+	1A03D,1A04D
BMB01+	1A03E,1A04E
BMB02+	1A03H,1A04H
BMB03+	1A03M,1A04M
BMB03-	1A03K,1A04K,1A08S
BMB04+	1A03S,1A04S,1A08T
BMB04-	1A03P,1A04P
BMB05+	1A03V,1A04V
BMB05-	1A03T,1A04T,1A08U
BMB06+	1B03E,1B04E
BMB06-	1B03C,1B04D,1A08V
BMB07+	1A10P,1A10T,1B04K,1B03K
BMB07-	1B03H,1B04H,1A08M,1A08E
BMB08+	1B03P,1B04P,1A10U,1A08N
BMB08-	1B03M,1B04M,1A10R,1A08F
BMB09+	1B03S,1B04S
BMB10+	1B03T,1B04T
BMB11+	1B03V,1B04V
BRUN-	1B05S,1B06S
BT1-	1B01S,1B02S,1A10M
BT2A-	1B01T,1B02T
C00+	1A24K,1A23R,1B12D
C00-	1A24L,1A23P
C01+	1B12E,1A24E,1A24T
C01-	1A24J,1A24U
C02+	1B12H,1A21R,1A24R
C02-	1A21J,1A24P
C03+	1A25K,1A21M,1B12J

***** CYCLIC CHECKSLM COMPUTER *****

C03-	1A21L,1A25L
C04+	1A25E,1A22R,1B12L
C04-	1A22J,1A25J
C05+	1B12M,1A22M,1A25T
C05-	1A22L,1A25U
C06+	1A26K,1A25R,1B12P
C06-	1A26L,1A25P
C07+	1A26E,1A26T,1B12R
C07-	1A26J,1A26U
C08+	1A27K,1A26R,1B12T
C08-	1A27L,1A26P
C09+	1A27E,1A27T,1B12U
C09-	1A27J,1A27U
C10+	1A28K,1A27R
C10-	1A28L,1A27P
C11+	1A28E
C11-	1A30J,1A30M,1A28J,1B18E,1B18J
C12S+	1A23E,1A23T,1B14U
C12S-	1A23J,1A23U
CC00+	1B15N,1B20E,1B28E,1B28H
CC00-	1A15D,1B15D,1B20J
CC01+	1B15P,1B20R
CC01-	1A15E,1B15E,1B18L,1B19L,1B2CF
CC01R+	1A11L,1B20U
CC01S+	1A11M,1B20T,1B28F,1B28N
CC02+	1B16N,1B21E,1B29E,1B29H
CC02-	1A16D,1B16D,1B21J
CC02R+	1B21L,1B19S
CC02S+	1B21K,1B19N,1B19P,1B18S
CC03+	1B22K,1B21R,1B16P
CC03-	1B22L,1B21P,1B16E,1A16E
CC03R+	1B25S,1B21U
CC03S+	1B29F,1B29N,1B29P,1B21T
CC04+	1B17N,1B22E,1B22T
CC04-	1A17D,1A15R,1B22J,1B22U
CC05+	1B23K,1B22R,1B17P
CC05-	1A17E,1A15S,1B23L,1B22P
CC06+	1B15U,1B23E,1B23T
CC06-	1A18D,1A16R,1B23J,1B23U
CC07+	1B24K,1B23R,1B15V
CC07-	1A18E,1A16S,1B24L,1B23P
CC08+	1B16U,1B24E,1B24T
CC08-	1A15K,1A17R,1B24J,1B24U
CC09+	1B25K,1B24R,1B16V
CC09-	1A15L,1A17S,1B25L,1B24P
CC10+	1B17U,1B25E,1B30E,1B30H
CC10-	1A16K,1A18R,1B25J
CC11+	1B26K,1B25R,1B17V
CC11-	1A16L,1A18S,1B18D,1B19D,1B25F,1B26L
CC11R+	1B30S,1B25U
CC11S+	1B30F,1B30N,1B30P,1B25T
CC12+	1B26E,1B26T
CC12-	1A17K,1B26J,1B26U
CC13+	1B27K,1B26R
CC13-	1A17L,1B27L,1B26P
CC14+	1B27E,1B28I
CC14-	1A18K,1B27J,1B29T

***** CYCLIC CHECKSUM COMPUTER *****

OUTPUT LISTS

C11+	1A28E	SINGLE I/O OR TEST CONNECTION
CC15+	1B27R	SINGLE I/O OR TEST CONNECTION
RUN+	1A13H	SINGLE I/O OR TEST CONNECTION

***** CYCLIC CHECKSUM COMPUTER *****

BAY 1 TO BAY 1, LEVEL 1

SHIFT1	1	BUS	1A 23D	1A 28D
GND	1	BUS	1A 01F	1A 06F
BCRD3-	1	BUS	1A 15F	1A 18F
CCLR+	1	BUS	1A 23F	1A 28F
GND	1	BUS	1A 01J	1A 06J
GND	1	BUS	1A 01L	1A 06L
BCRD1-	1	BUS	1A 15M	1A 18M
GND	1	BUS	1A 01N	1A 06N
SHIFT1	1	BUS	1A 23N	1A 27N
GND	1	BUS	1A 01R	1A 06R
BCRD2-	1	BUS	1A 15T	1A 18T
GND	1	BUS	1A 01U	1A 06U
SHIFT2	1	BUS	1B 20C	1B 27C
BCCLK3	1	BUS	1B 20F	1B 23F
BCCLK1	1	BUS	1B 24F	1B 27F
BCWR3+	1	BUS	1B 20M	1B 23M
BCWR1+	1	BUS	1B 24M	1B 27M
SHIFT2	1	BUS	1B 20N	1B 27N
BCWR3+	1	BUS	1B 20V	1B 23V
BCWR1+	1	BUS	1B 24V	1B 27V
G1A191	1	001	1A 19C	1A 19E
MSIXS+	1	001	1A 09C	1A 09F
CC15-	1	001	1A 30E	1A 30H
GND	1	001	1A 01F	1A 01J
GND	1	001	1A 06J	1A 06L
J1A291	1	001	1A 29J	1A 29K
GND	1	001	1A 01L	1A 01N
GND	1	001	1A 06N	1A 06R
BLK+	1	001	1A 08R	1A 09S
J1B181	1	001	1B 19E	1B 18F
CC00+	1	001	1B 28E	1B 28H
CC02+	1	001	1B 29E	1B 29H
CC10+	1	001	1B 30E	1B 30H
GND	1	001	1B 03F	1B 03J
GND	1	001	1B 04F	1B 04J
GND	1	001	1B 01J	1B 01L
GND	1	001	1B 02J	1B 02L
GND	1	001	1B 05J	1B 05L
GND	1	001	1B 06J	1B 06L
GND	1	001	1B 03L	1B 03N
GND	1	001	1B 04L	1B 04N
J1B182	1	001	1B 19M	1B 18N
GND	1	001	1B 01N	1B 01R
GND	1	001	1B 05N	1B 05R
GND	1	001	1B 06N	1B 06R
J1B13	1	001	1B 13T	1B 13U
G1A22	1	002	1A 21C	1A 22C
BAC00+	1	002	1A 01D	1A 02D
BMB00+	1	002	1A 03D	1A 04D
AC00+	1	002	1A 05D	1A 06D
LC-	1	002	1A 10D	1A 10H
HOLD+	1	002	1A 11C	1A 12D
BAC01+	1	002	1A 01E	1A 02E
BMB01+	1	002	1A 03E	1A 04E

***** CYCLIC CHECKSUM COMPUTER *****

AC01+	1	002	1A 05E	1A 06E
BAC02+	1	002	1A 01H	1A 02H
BMB02+	1	002	1A 03H	1A 04H
AC02+	1	002	1A 05H	1A 06H
BCH+	1	002	1A 08J	1A 09M
LC-	1	002	1A 09J	1A 10L
C11-	1	002	1A 30J	1A 30M
BAC03+	1	002	1A 01K	1A 02K
BMB03-	1	002	1A 03K	1A 04K
AC03+	1	002	1A 05K	1A 06K
BAC04+	1	002	1A 01M	1A 02M
BMB03+	1	002	1A 03M	1A 04M
AC04+	1	002	1A 05M	1A 06M
LCS+	1	002	1A 27M	1A 28M
BAC05+	1	002	1A 01P	1A 02P
BMB04-	1	002	1A 03P	1A 04P
AC05+	1	002	1A 05P	1A 06P
BMB07+	1	002	1A 10P	1A 10T
GND	1	002	1A 01R	1A 01U
BAC06+	1	002	1A 01S	1A 02S
BMB04+	1	002	1A 03S	1A 04S
AC06+	1	002	1A 05S	1A 06S
LCE+	1	002	1A 22S	1A 21U
BAC07+	1	002	1A 01T	1A 02T
BMB05-	1	002	1A 03T	1A 04T
AC07+	1	002	1A 05T	1A 06T
BAC08+	1	002	1A 01V	1A 02V
BMB05+	1	002	1A 03V	1A 04V
AC08+	1	002	1A 05V	1A 06V
G1A22	1	002	1A 21V	1A 22V
LCT+	1	002	1A 23V	1A 24V
LCS+	1	002	1A 26V	1A 27V
BAC09+	1	002	1B 01C	1B 02C
BMB06-	1	002	1B 03C	1B 04C
AC09+	1	002	1B 05C	1B 06C
BLK-	1	002	1B 08C	1B 08H
CC11-	1	002	1B 18C	1B 19D
BAC10+	1	002	1B 01E	1B 02E
BMB06+	1	002	1B 03E	1B 04E
AC10+	1	002	1B 05E	1B 06E
GND	1	002	1B 01F	1B 02F
J1B14	1	002	1B 12F	1B 12K
BAC11+	1	002	1B 01H	1B 02H
BMB07-	1	002	1B 03H	1B 04H
AC11+	1	002	1B 05H	1B 06H
FBTAP2	1	002	1B 28J	1B 28M
FBTAP2	1	002	1B 29J	1B 30J
IOP1-	1	002	1B 01K	1B 02K
BMB07+	1	002	1B 03K	1B 04K
SKIP	1	002	1B 05K	1B 06K
FBTP1+	1	002	1B 19K	1B 20K
CC01-	1	002	1B 18L	1B 19L
IOP2-	1	002	1B 01M	1B 02M
BMB08-	1	002	1B 03M	1B 04M
INTREQ	1	002	1B 05M	1B 06M
BC2E-	1	002	1B 10M	1B 09R
BCWR2-	1	002	1B 16M	1B 17M

***** CYCLIC CHECKSUM COMPUTER *****

FBTP1-	1	002	1B 18M	1B 18R
GND	1	002	1B 02N	1B 03R
J1B14	1	002	1B 12N	1B 12S
IOP4-	1	002	1B 01P	1B 02P
BMB08+	1	002	1B 03P	1B 04P
ACCLR+	1	002	1B 05P	1B 06P
BIOP1-	1	002	1B 09P	1B 10P
CC02S+	1	002	1B 19P	1B 18S
GND	1	002	1B 02R	1B 02U
GND	1	002	1B 04R	1B 04U
BT1-	1	002	1B 01S	1B 02S
BMB09+	1	002	1B 03S	1B 04S
BRUN-	1	002	1B 05S	1B 06S
BT2A-	1	002	1B 01T	1B 02T
BMB10+	1	002	1B 03T	1B 04T
HOLD+	1	002	1B 05T	1B 06T
BIOP2-	1	002	1B 09T	1B 10T
BCWR2-	1	002	1B 16T	1B 17T
CC15S+	1	002	1B 27T	1B 28V
CC15S+	1	002	1B 30T	1B 29V
GND	1	002	1B 05U	1B 06U
PWRCLR	1	002	1B 01V	1B 02V
BMB11+	1	002	1B 03V	1B 04V
BIOT+	1	002	1B 05V	1B 06V
FBTP1-	1	002	1B 18V	1B 19V
SHIFT1	1	003	1A 23C	1A 22F
J1A301	1	003	1A 30C	1A 30K
BMB07-	1	003	1A 08E	1A 08M
CCLR+	1	003	1A 23F	1A 22K
CRC15+	1	003	1A 30F	1A 30N
BIOP4-	1	003	1A 09F	1A 10J
AC04+	1	003	1A 15H	1A 15N
AC06+	1	003	1A 16H	1A 16N
AC08+	1	003	1A 17H	1A 17N
AC10+	1	003	1A 18H	1A 18N
SHIFT1	1	003	1A 21H	1A 21P
AC05+	1	003	1A 15J	1A 15P
AC07+	1	003	1A 16J	1A 16P
AC09+	1	003	1A 17J	1A 17P
AC11+	1	003	1A 18J	1A 18P
BCL+	1	003	1A 08K	1A 09P
BIOP4+	1	003	1A 20K	1A 20R
C00+	1	003	1A 24K	1A 23R
C06+	1	003	1A 26K	1A 25R
C08+	1	003	1A 27K	1A 26R
C10+	1	003	1A 28K	1A 27R
C00-	1	003	1A 24L	1A 23P
C06-	1	003	1A 26L	1A 25P
C08-	1	003	1A 27L	1A 26P
C10-	1	003	1A 28L	1A 27P
SHIFT1	1	003	1A 23N	1A 22P
CLK	1	003	1A 19P	1A 19V
BCCLR2	1	003	1A 20P	1A 20V
J1B281	1	003	1B 28D	1B 28K
J1B291	1	003	1B 29D	1B 29K
J1B301	1	003	1B 30D	1B 30K
MSIX+	1	003	1B 09E	1B 09M

***** CYCLIC CHECKSUM COMPUTER *****

BIOP4-	1	003	1B 10E	1B 10L
CC01S+	1	003	1B 28F	1B 28N
CC03S+	1	003	1B 29F	1B 29N
CC11S+	1	003	1B 30F	1B 30N
FBTP1-	1	003	1B 19H	1B 20L
CC02S+	1	003	1B 21K	1B 19N
CC03+	1	003	1B 22K	1B 21R
CC05+	1	003	1B 23K	1B 22R
CC07+	1	003	1B 24K	1B 23R
CC09+	1	003	1B 25K	1B 24R
CC11+	1	003	1B 26K	1B 25R
CC13+	1	003	1B 27K	1B 26R
BC3E-	1	003	1B 11L	1B 10R
CC03-	1	003	1B 22L	1B 21P
CC05-	1	003	1B 23L	1B 22P
CC07-	1	003	1B 24L	1B 23P
CC09-	1	003	1B 25L	1B 24P
CC11-	1	003	1B 26L	1B 25P
BCWR2-	1	003	1B 15M	1B 15T
FBTAP2	1	003	1B 29M	1B 28S
G1A23	1	004	1A 23C	1A 23L
CC09-	1	004	1A 15L	1A 17S
CC11-	1	004	1A 16L	1A 18S
LCS+	1	004	1A 26M	1A 25V
BMB08+	1	004	1A 08N	1A 10U
SHIFT2	1	004	1B 20D	1B 20N
BCWR2+	1	004	1B 11E	1B 10N
BC2E+	1	004	1B 09K	1B 11K
CRC12+	1	004	1B 18K	1B 18T
BCWR3+	1	004	1B 20M	1B 20V
BCWR1+	1	004	1B 24M	1B 24V
GND	1	004	1B 01U	1B 03U
J1B14	1	004	1B 12V	1B 14V
RUN-	1	005	1A 11E	1A 13J
C02-	1	005	1A 21J	1A 24P
C12S-	1	005	1A 23J	1A 23U
C01-	1	005	1A 24J	1A 24U
C07-	1	005	1A 26J	1A 26U
C09-	1	005	1A 27J	1A 27U
CC08-	1	005	1A 15K	1A 17R
CC10-	1	005	1A 16K	1A 18R
BCRD3+	1	005	1B 11H	1B 10S
AAC09-	1	005	1B 14J	1B 16S
CC04-	1	005	1B 22J	1B 22U
CC06-	1	005	1B 23J	1B 23U
CC08-	1	005	1B 24J	1B 24U
CC02R+	1	005	1B 21L	1B 19S
BIOP4+	1	005	1B 08N	1B 11U
AAC11-	1	005	1B 14N	1B 17S
LCT+	1	006	1A 21E	1A 24M
LCE+	1	006	1A 22E	1A 25M
C12S+	1	006	1A 23E	1A 23T
C07+	1	006	1A 26E	1A 26T
C09+	1	006	1A 27E	1A 27T
BMB08-	1	006	1A 08F	1A 10R
C04-	1	006	1A 22J	1A 25J
C05-	1	006	1A 22L	1A 25U

BCRD2-	1	006	1A 15T	1B 15F
GND	1	006	1A 06U	1B 05F
AAC07-	1	006	1B 14C	1B 15S
CC12+	1	006	1B 26E	1B 26T
AAC08-	1	006	1B 14F	1B 16R
BCCLR3	1	006	1B 20F	1B 17J
AAC10-	1	006	1B 14L	1B 17R
CC04-	1	007	1A 17D	1A 15R
CC06-	1	007	1A 18D	1A 16R
CC05-	1	007	1A 17E	1A 15S
CC07-	1	007	1A 18E	1A 16S
C04+	1	007	1A 25E	1A 22R
LCS+	1	007	1A 20F	1A 22U
LC10-	1	007	1A 11J	1A 08P
BCRD1-	1	007	1A 15M	1A 11R
BCL-	1	007	1A 09N	1B 09U
AAC06+	1	007	1A 25S	1B 25H
AAC08+	1	007	1A 26S	1B 26H
AAC10+	1	007	1A 27S	1B 27H
CC14+	1	007	1B 27E	1B 28T
CC15R+	1	007	1B 27U	1B 30V
LC+	1	008	1A 09K	1A 13P
C03+	1	008	1A 25K	1A 21M
C03-	1	008	1A 21L	1A 25L
AAC04+	1	008	1B 20F	1B 24H
AAC05+	1	008	1B 20S	1B 24S
AAC07+	1	008	1B 21S	1B 25S
AAC09+	1	008	1B 22S	1B 26S
AAC11+	1	008	1B 23S	1B 27S
PCLR+	1	009	1A 13F	1A 09T
BLK-	1	009	1A 09R	1B 10J
CC00+	1	009	1B 20E	1B 15N
CC02+	1	009	1B 21E	1B 16N
CC04+	1	009	1B 22E	1B 17N
BC1E+	1	009	1B 09F	1B 14S
CC11S+	1	009	1B 30P	1B 25T
CCC+	1	010	1A 08C	1A 13L
BCH-	1	010	1A 09L	1B 09H
CC11R+	1	010	1B 30S	1B 25U
CLK	1	011	1A 13C	1B 13C
CC00-	1	011	1A 15C	1B 15C
CC02-	1	011	1A 16C	1B 16C
CC01-	1	011	1A 15E	1B 15E
CC03-	1	011	1A 16E	1B 16E
BCRD2-	1	011	1A 11T	1B 16F
MSIX-	1	011	1A 13T	1B 09J
AAC06-	1	011	1A 14T	1B 15R
BCWR3+	1	011	1A 20T	1B 23M
CRC12-	1	011	1A 30V	1B 28P
BIOP2-	1	011	1B 17E	1B 11R
CC01+	1	011	1B 15P	1B 20R
GECLR-	1	012	1A 08H	1B 10D
LC10+	1	012	1A 11K	1B 10K
AAC04-	1	012	1A 14N	1B 17K
BCRD2+	1	012	1A 11U	1B 09S
BC2E-	1	012	1B 17F	1B 11J
BC1E-	1	012	1B 08R	1B 14R

***** CYCLIC CHECKSUM COMPUTER *****

LC+	1	013	1A 20E	1A 13V
BIOT-	1	013	1A 11F	1B 10H
LC+	1	013	1A 29F	1A 23H
AAC02-	1	013	1A 14J	1B 16K
AAC00-	1	014	1A 14E	1B 15K
AAC00+	1	014	1A 07E	1A 14E
AAC01-	1	014	1A 14F	1B 15L
AAC03-	1	014	1A 21F	1A 14L
AAC01+	1	014	1A 07H	1A 14H
AAC02+	1	014	1A 07K	1A 14K
BIOP1+	1	014	1A 13K	1B 08F
AAC03+	1	014	1A 07M	1A 14M
AAC04+	1	014	1A 07P	1A 14P
CRC15+	1	014	1A 30P	1B 29U
BCRD1+	1	014	1A 11S	1B 08S
CRC15-	1	014	1A 30S	1B 28U
AAC06+	1	014	1A 14U	1B 21H
AAC07+	1	014	1B 07E	1B 14E
AAC09+	1	014	1B 07K	1B 14K
AAC11+	1	014	1B 07P	1B 14P
BCRD3-	1	015	1A 15F	1B 11F
BIOP2+	1	015	1A 13N	1B 08K
BCCLR1	1	015	1B 24F	1B 17H
CC03S+	1	015	1B 29P	1B 21T
BIOP1-	1	016	1A 09E	1B 08P
AAC05-	1	016	1A 22F	1A 14R
SHIFT2	1	016	1A 19T	1B 27D
CC06+	1	016	1B 23E	1B 15U
CC08+	1	016	1B 24E	1B 16U
CC10+	1	016	1B 25E	1B 17U
AAC08+	1	016	1B 14H	1B 22H
CC03R+	1	016	1B 29S	1B 21U
BIOP1+	1	017	1A 20D	1A 29E
SHIFT1	1	017	1A 19M	1A 27N
AAC05+	1	017	1A 14S	1A 22T
BIOP2-	1	018	1A 10E	1B 17D
STOP+	1	018	1A 13E	1B 14T
BCWR1+	1	018	1A 20M	1B 27M
MSIX+	1	018	1A 13S	1B 19U
BC3E+	1	018	1A 20S	1B 11M
AAC10+	1	018	1B 23H	1B 14M
LCCLR+	1	019	1A 19D	1A 29M
ACC+	1	019	1A 19J	1B 10F
C02+	1	019	1A 21R	1B 12H
CC12-	1	020	1A 17K	1B 26J
CC14-	1	020	1A 18K	1B 27J
C11-	1	021	1A 28J	1B 18E
CCLR+	1	021	1A 21K	1A 10N
CC13-	1	021	1A 17L	1B 27L
CC15-	1	021	1A 18L	1B 27P
C05+	1	021	1A 22M	1B 12M
MSIX-	1	021	1B 28R	1B 18U
LCT+	1	022	1A 10K	1A 21S
CRC12+	1	022	1A 30T	1B 19F
CC01R+	1	023	1A 11L	1B 20U
CC01S+	1	023	1A 11M	1B 20T
C01+	1	024	1A 24E	1B 12E

***** CYCLIC CHECKSUM COMPUTER *****

BAY 1 TO BAY 1, LEVEL 2

LC-	2	001	1A 10F	1A 09J
LC+	2	001	1A 09K	1A 08L
J1A301	2	001	1A 30K	1A 30L
CRC15+	2	001	1A 30N	1A 30P
BIOP2-	2	001	1B 17C	1B 17E
GND	2	001	1B 01F	1B 01J
GND	2	001	1B 02F	1B 02J
GND	2	001	1B 05F	1B 05J
GND	2	001	1B 06F	1B 06J
J1B181	2	001	1B 18F	1B 18H
GND	2	001	1B 03J	1B 03L
GND	2	001	1B 04J	1B 04L
J1B281	2	001	1B 28K	1B 28L
J1B291	2	001	1B 29K	1B 29L
J1B301	2	001	1B 30K	1B 30L
GND	2	001	1B 01L	1B 01N
GND	2	001	1B 02L	1B 02N
GND	2	001	1B 05L	1B 05N
GND	2	001	1B 06L	1B 06N
GND	2	001	1B 03N	1B 03R
J1B182	2	001	1B 18N	1B 18P
CC02S+	2	001	1B 19N	1B 19P
CC03S+	2	001	1B 29N	1B 29P
CC11S+	2	001	1B 30N	1B 30P
SHIFT1	2	002	1A 21M	1A 22H
CCLR+	2	002	1A 21K	1A 22K
LCS+	2	002	1A 26M	1A 27M
SHIFT1	2	002	1A 21P	1A 22P
LCS+	2	002	1A 25V	1A 26V
C11-	2	002	1B 18E	1B 18J
GND	2	002	1B 03F	1B 04F
BCRD2-	2	002	1B 15F	1B 16F
BLK-	2	002	1B 08H	1B 08L
BCH-	2	002	1B 09H	1B 09L
FBTAP2	2	002	1B 28J	1B 29J
FBTAP2	2	002	1B 30J	1B 30M
J1B14	2	002	1B 12K	1B 12N
CC01-	2	002	1B 19L	1B 20P
BCWR2-	2	002	1B 15M	1B 16M
FBTAP2	2	002	1B 28M	1B 29M
GND	2	002	1B 04N	1B 05R
BIOP1-	2	002	1B 08P	1B 09P
GND	2	002	1B 01R	1B 02R
GND	2	002	1B 04R	1B 05U
GND	2	002	1B 06R	1B 06U
BC1E-	2	002	1B 08R	1B 08U
BC2E-	2	002	1B 09R	1B 09U
BC3E-	2	002	1B 10R	1B 10U
BIOP2-	2	002	1B 11R	1B 10T
FBTP1-	2	002	1B 18R	1B 18V
J1B14	2	002	1B 12S	1B 12V
BIOP2-	2	002	1B 08T	1B 09T
BCWR2-	2	002	1B 15T	1B 16T
GND	2	002	1B 01U	1B 02U

***** CYCLIC CHECKSUM COMPUTER *****

GND	2	002	1B 03U	1B 04U
CC15S+	2	002	1B 28V	1B 29V
C11-	2	003	1A 28J	1A 30M
BCH+	2	003	1A 09M	1A 10S
AC04+	2	003	1A 15N	1A 15U
AC06+	2	003	1A 16N	1A 16U
AC08+	2	003	1A 17N	1A 17U
AC10+	2	003	1A 18N	1A 18U
LC+	2	003	1A 13P	1A 13V
CRC12+	2	003	1B 19F	1B 18K
FBTP1--	2	003	1B 19H	1B 18M
BC2E-	2	003	1B 11J	1B 10M
CC13-	2	003	1B 27L	1B 26P
BC3E+	2	003	1B 11M	1B 09N
BIOP1-	2	003	1B 11N	1B 10P
LCT+	2	004	1A 24M	1A 23V
BCL+	2	004	1A 09P	1A 10V
BLK-	2	004	1B 08D	1B 10J
BIOP4-	2	004	1B 10L	1B 11T
LCT+	2	005	1A 21E	1A 21S
LCE+	2	005	1A 22E	1A 22S
AC05+	2	005	1A 15J	1A 15V
AC07+	2	005	1A 16J	1A 16V
AC09+	2	005	1A 17J	1A 17V
AC11+	2	005	1A 18J	1A 18V
BIOP4+	2	005	1A 23M	1A 20R
CC12-	2	005	1B 26J	1B 26G
LC+	2	006	1A 20E	1A 23H
C01+	2	006	1A 24E	1A 24T
C02+	2	006	1A 21R	1A 24R
CC11-	2	006	1A 18S	1B 18C
CC04+	2	006	1B 22E	1B 22T
CC06+	2	006	1B 23E	1B 23T
CC08+	2	006	1B 24E	1B 24T
BIOP1+	2	006	1B 08F	1B 11P
CC14-	2	006	1B 27J	1B 29T
BIOP2+	2	006	1B 08K	1B 11S
C05+	2	007	1A 22M	1A 25T
BMB05-	2	007	1A 04T	1A 08U
CC01-	2	007	1B 15E	1B 18L
G1A22	2	008	1A 21C	1A 21V
BMB03-	2	008	1A 04K	1A 08S
BMB06-	2	008	1A 08V	1B 04D
AAC06+	2	008	1B 21H	1B 25H
AAC08+	2	008	1B 22H	1B 26H
AAC10+	2	008	1B 23H	1B 27H
MSIXS+	2	009	1A 09F	1A 13U
BMB04+	2	009	1A 04S	1A 08T
BAC00+	2	010	1A 02C	1A 07D
BIOP4-	2	010	1A 10J	1B 10E
MSIX+	2	010	1A 13S	1B 09E
BAC07+	2	010	1A 02I	1B 07D
BAC08+	2	010	1A 02V	1B 07F
BAC09+	2	010	1B 02C	1B 07J
CC00-	2	010	1B 15C	1B 20J
CC02-	2	010	1B 16C	1B 21J
CC03-	2	010	1B 16E	1B 21P

***** CYCLIC CHECKSUM COMPUTER *****

CC10+	2	010	1B 25E	1B 30E
CLK	2	011	1A 13D	1A 19P
BAC01+	2	011	1A 02E	1A 07F
BAC02+	2	011	1A 02H	1A 07J
BAC03+	2	011	1A 02K	1A 07L
BAC04+	2	011	1A 02M	1A 07N
BAC05+	2	011	1A 02P	1A 07R
BAC06+	2	011	1A 02S	1A 07T
LCS+	2	011	1A 22U	1A 27V
BCWR2-	2	011	1B 11D	1B 17M
CC11-	2	011	1B 19D	1B 25P
BAC10+	2	011	1B 02E	1B 07L
BAC11+	2	011	1B 02H	1B 07N
CC03+	2	011	1B 16P	1B 21R
CC05+	2	011	1B 17P	1B 22R
AAC04+	2	012	1A 25H	1B 24H
AAC05-	2	012	1A 14R	1B 17L
AAC05+	2	012	1A 22T	1B 24S
IOP1-	2	012	1B 08E	1B 02K
IOP4-	2	012	1B 08M	1B 02P
AAC03-	2	013	1A 14L	1B 16L
BMB07-	2	013	1A 08M	1B 04H
IOP2-	2	013	1B 08J	1B 02M
AAC04+	2	014	1A 14P	1B 20H
AAC05+	2	014	1A 07S	1A 14S
BMB07+	2	014	1A 10T	1B 04K
AAC06+	2	014	1A 07U	1A 14L
AAC07+	2	014	1B 14E	1B 21S
AAC08+	2	014	1B 07F	1B 14H
BCCLR3	2	014	1B 17J	1B 10V
AAC10+	2	014	1B 07M	1B 14M
BIOP1+	2	015	1A 20C	1A 13K
RUN-	2	015	1A 13J	1B 13S
AAC03+	2	015	1A 14M	1A 21T
CC04-	2	015	1A 15R	1B 22J
CC06-	2	015	1A 16R	1B 23J
CC08-	2	015	1A 17R	1B 24J
CC10-	2	015	1A 18R	1B 25J
BMB08+	2	015	1A 10U	1B 04P
AAC07+	2	016	1A 26H	1B 25S
AAC09+	2	016	1A 27H	1B 26S
AAC11+	2	016	1A 28H	1B 27S
BMB08-	2	016	1A 10R	1B 04M
MSIX-	2	016	1A 13T	1B 18U
CC00+	2	016	1B 20E	1B 28E
CC02+	2	016	1B 21E	1B 29E
CC01S+	2	016	1B 28F	1B 20T
AAC09+	2	016	1B 14K	1B 22S
CC07+	2	016	1B 23R	1B 15V
CC09+	2	016	1B 24R	1B 16V
CC11+	2	016	1B 25R	1B 17V
CC15-	2	017	1A 30H	1B 27P
CC05-	2	017	1A 15S	1B 23L
CC07-	2	017	1A 16S	1B 24L
CC09-	2	017	1A 17S	1B 25L
BCCLR1	2	017	1B 17H	1B 08V
AAC00+	2	018	1A 14E	1A 23S

***** CYCLIC CHECKSUM COMPUTER *****

AC04+	2	018	1A 15F	1A 06M
C03+	2	018	1A 21M	1B 12J
AC05+	2	018	1A 08P	1A 15P
PWRCLR	2	018	1A 09U	1B 02V
AAC11+	2	018	1B 14P	1B 23S
AC07+	2	019	1A 16P	1A 06T
CRC15+	2	019	1B 19T	1B 29U
AAC01+	2	020	1A 14H	1A 24H
AC06+	2	020	1A 16H	1A 06S
AAC02+	2	020	1A 14K	1A 24S
BC1E+	2	020	1A 20L	1B 14S
C04+	2	020	1A 22R	1B 12L
AC00+	2	021	1A 06D	1B 15H
LCE+	2	021	1A 10F	1A 21U
AC02+	2	021	1A 08H	1B 16H
BIOT+	2	021	1A 11H	1B 06V
AC08+	2	021	1A 17H	1A 06V
AC03+	2	021	1A 06K	1B 16J
BT1-	2	021	1A 10M	1B 02S
AC09+	2	021	1A 17P	1B 06E
C12S+	2	021	1A 23T	1B 14U
HULD+	2	022	1A 11C	1B 06T
AC01+	2	022	1A 06E	1B 15J
CQ0+	2	022	1A 23R	1B 12D
BCCLR2	2	022	1A 20V	1B 09V
AC10+	2	024	1A 18F	1B 06E
BIOP4+	2	024	1A 20K	1B 08N
AC11+	2	024	1A 18P	1B 06H
C06+	2	027	1A 25R	1B 12P
ACCLR+	2	028	1A 19F	1B 06P
C08+	2	028	1A 26R	1B 12T
C07+	2	028	1A 26T	1B 12R
C09+	2	031	1A 27T	1B 12U

520 WIRES @.10 \$52.00, @.15 \$78.00

CARDS PUNCHED 0

NUMBER OF BUSS STRIPS 20

NUMBER OF WRAPS 520

TOTAL LENGTH OF WIRE 222

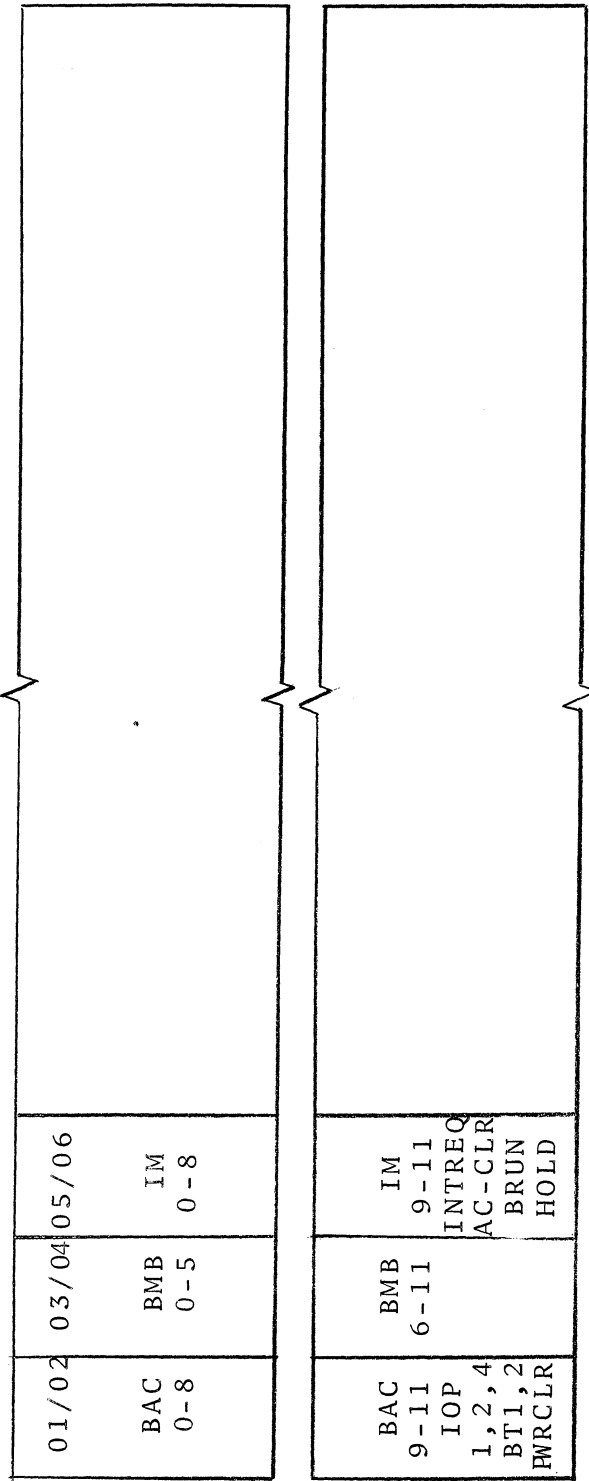
APPENDIX B

CONNECTOR MAPS

APPENDIX B
CONNECTOR MAPS

The following figures document the I/O cable connection locations and the I/O signal names as they appear on the checksum adapter side of the interface and on the PDP-8 side of the interface. The tables should be self-explanatory and are not further discussed.

CYCLIC CHECK COMPUTER MOUNTING PANEL



A

B

Figure B1. Cable Layout Map.

CYCLIC CHECK COMPUTER		PDP-8	
SIGNAL NAME	INTERFACE CONNECTION	ASSERTION LEVEL	SIGNAL NAME
BAC00+	A01D,A02D	————◇	BAC0(1)
BAC01+	A01E,A02E	————◇	BAC1(1)
BAC02+	A01H,A02H	————◇	BAC2(1)
BAC03+	A01K,A02K	————◇	BAC3(1)
BAC04+	A01M,A02M	————◇	BAC4(1)
BAC05+	A01P,A02P	————◇	BAC5(1)
BAC06+	A01S,A02S	————◇	BAC6(1)
BAC07+	A01T,A02T	————◇	BAC7(1)
BAC08+	A01V,A02V	————◇	BAC8(1)
BAC09+	B01D,B02D	————◇	BAC9(1)
BAC10+	B01E,B02E	————◇	BAC10(1)
BAC11+	B01H,B02H	————◇	BAC(11)
IOP1-	B01K,B02K	————▶	IOP1
IOP2-	B01M,B02M	————▶	IOP2
IOP4-	B01P,B02P	————▶	IOP4
BT1-	B01S,B02S	————▶	BT1
BT2A-	B01T,B02T	————▶	BT2A
PWRCLR	B01V,B02V	————▶	B POWER CLEAR

Figure B2. Connector Map.

CYCLIC CHECK COMPUTER		PDP-8	
SIGNAL NAME	INTERFACE CONNECTION	ASSERTION LEVEL	SIGNAL NAME
BMB00+	A03D, A04D	—◇	BMB0(1)
BMB01+	A03E, A04E	—◇	BMB1(1)
BMB02+	A03H, A04H	—◇	BMB2(1)
BMB03-	A03K, A04K	—◇	BMB3(0)
BMB03+	A03M, A04M	—◇	BMB3(1)
BMB04-	A03P, A04P	—◇	BMB4(0)
BMB04+	A03S, A04S	—◇	BMB4(1)
BMB05-	A03T, A04T	—◇	BMB5(0)
BMB05+	A03V, A04V	—◇	BMB5(1)
BMB06-	B03D, B04D	—◇	BMB6(0)
BMB06+	B03E, B04E	—◇	BMB6(1)
BMB07-	B03H, B04H	—◇	BMB7(0)
BMB07+	B03K, B04K	—◇	BMB7(1)
BMB08-	B03M, B04M	—◇	BMB8(0)
BMB08+	B03P, B04P	—◇	BMB8(1)
BMB09+	B03S, B04S	—◇	BMB9(1)
BMB10+	B03T, B04T	—◇	BMB10(1)
BMB11+	B03V, B04V	—◇	BMB11(1)

Figure B3. Connector Map.

CYCLIC CHECK COMPUTER		PDP-8	
SIGNAL NAME	INTERFACE CONNECTION	ASSERTION LEVEL	SIGNAL NAME
AC00+	A05D,A06D	—◇	AC 0*
AC01+	A05E,A06E	—◇	AC 1
AC02+	A05H,A06H	—◇	AC 2
AC03+	A05K,A06K	—◇	AC 3
AC04+	A05M,A06M	—◇	AC 4
AC05+	A05P,A06P	—◇	AC 5
AC06+	A05S,A06S	—◇	AC 6
AC07+	A05T,A06T	—◇	AC 7
AC08+	A05V,A06V	—◇	AC 8
AC09+	B05D,B06D	—◇	AC 9
AC10+	B05E,B06E	—◇	AC10
AC11+	B05H,B06H	—◇	AC11
SKIP**	B05K,B06K	—◇	SKIP
INTREQ**	B05M,B06M	—◇	INTERRUPT REQUEST
ACCLR+	B05P,B06P	—◇	CLEAR AC
BRUN-**	B05S,B06S	—◇	B RUN(1)
HOLD+	B05T,B06T	—◇	HOLD
BIOT+	B05V,B06V	—◇	BIOT

* Some DEC Documents Refer to the AC Input Lines as the "IM".

** Not Used by Interface.

Figure B4. Connector Map.

APPENDIX C

DIAGNOSTIC PROCEDURES

APPENDIX C

DIAGNOSTIC PROCEDURES

Included as part of the Data Concentrator support library is a binary tape for certifying the performance of the cyclic check adapter. The program tests the loading, clearing, and reading of all registers, as well as the clearing of the PDP-8 AC. The shifting facility of the character and residue registers is also checked under 6-, 8-, and 12-bit character modes. The mod 2 adders between stages are then checked along with the feedback tap control, with the generator operating in both the word and byte mode. Finally the generator is exercised by computing a checksum over 4096 characters using first the CRC-12 generator polynomial, then the CRC-16 polynomial. The results so obtained are compared against software-generated checksums for accuracy.

If the test halts, the faulty register is normally displayed in the AC (plus MQ, depending on register length). See the program source listing for additional comments, especially with regard to the probable course of the error. After a halt, the program may be restarted with the processor CONTINUE key. Register clearing is taken care of by the routines.

201 CYCLIC CHECK COMPUTER - TEST ROUTINES

** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** **
 *
 * CYCLIC CHECK COMPUTER - TEST ROUTINES *
 * 29 APRIL 68 *
 *
 * ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** **

** ** ** ** ** ** ** ** **
 *
 * ASSEMBLER DEFINITIONS *
 *
 * ** ** ** ** ** ** ** **

0001	RD	OPD	1
0002	CLR	OPD	2
0004	WR	OPD	4
6540	BCL	OPD	6540
6550	BCH	OPD	6550
6561	LCM6	OPD	6561
6563	LCM8	OPD	6563
6567	LCM12	OPD	6567
6571	CCC	OPD	6571

201 CYCLIC CHECK COMPUTER - TEST ROUTINES

```
*****
*
*          CYCLIC CHECK COMPUTER - TEST ROUTINES
*          PAGE 1
*
*****
```

```
*****
*
*          ... PROGRAM STARTING ADDRESS ...
*          --- 0200 ---
*
*****
```

0010	0010	AXR1	DS	1
	0020		ORG	20
		*		
		*		SAVE AREA
		*		
0020		TEMP1	DS	1
0021		CHRCNT	DS	1
0022		CNT	DS	1
		*		
		*		
		*		COMMON CONSTANT POOL
		*		
0023	0040	K0040	DC	0040
0024	0100	K0100	DC	0100
0025	0200	K0200	DC	0200
0026	0240	K0240	DC	0240
0027	0377	K0377	DC	0377
0030	4000	K4000	DC	4000
0031	7401	K7401	DC	7401
0032	0377	M7401	DC	0377
0033	7540	M0240	DC	7540
0034	7764	M0012D	DC	7764
0035	7770	M0008D	DC	7770
		*		
		*		
		*		THROUGHOUT THESE ROUTINES THE REGISTER BEING
		*		TESTED WILL BE DISPLAYED IN THE AC UPON A
		*		HALT. IF A REGISTER IS LONGER THAN
		*		12 BITS, THEN IT WILL BE DISPLAYED IN
		*		BOTH THE AC AND MQ REGISTERS. SEE CODE
		*		FOR SPECIFIC DETAILS.
		*		
	0200		ORG	200
		*		
		*		PROGRAM INITIALIZATION
		*		
0200	7200		CLA	
0201	1334		TAD	CRLFP POINTER TO CRLF MESSAGE
0202	3010		DCA	AXR1
0203	7344		STA+CLL+RAL	-2 TO AC
0204	3021		DCA	CHRCNT

201 CYCLIC CHECK COMPUTER - TEST ROUTINES

0205	5733	JMP*	CRLF	INITIALIZE TTY
		*		
		*		
		*		
		*		REGISTER READ/WRITE CHECK
		*		
		*		CHECK BYTE MODE READ/WRITE
		*		
0206	7200	START	CLA	
0207	3022		DCA	CNT
0210	7040		CMA	ZERO CNT (LOOP COUNTER) SET AC
0211	6563		LCM8	SET BYTE MODE. 377 TO CHAR BUF
0212	7440		SZA	LOAD SHOULD CLEAR AC
0213	7402		HLT	** AC CLEAR ERROR
0214	7240		STA	
0215	6544		BCL+WR	LOAD BCR-LO
0216	7440		SZA	AC SHOULD BE ZERO
0217	7402		HLT	** AC CLEAR ERROR
0220	7200		CLA	
0221	6541		BCL+RD	
0222	7040		CMA	
0223	0027		AND	K0377
0224	7440		SZA	AC SHOULD BE ALL ZERO
0225	7402		HLT	** BCL READ/WRITE ERROR?
		*		** FAULTY BIT POSITION(S) LIT
0226	7200		CLA	
0227	6542		BCL+CLR	
0230	6541		BCL+RD	
0231	7440		SZA	AC SHOULD EQUAL ZERO
0232	7402		HLT	** BCL CLEAR ERROR. ALSO POSSIBLE ** THAT READ IS IN ERROR.
		*		
0233	7240		STA	7777 TO AC
0234	6554		BCH+WR	377 TO BCR-HI
0235	7440		SZA	BCR LOAD SHOULD ZERO AC
0236	7402		HLT	** AC CLEAR ERROR
0237	6551		BCH+RD	DID ONES GET THERE
0240	7040		CMA	
0241	0027		AND	K0377
0242	7440		SZA	AC SHOULD BE ZERO
0243	7402		HLT	** BCH READ/WRITE ERROR?
		*		** FAULTY BIT POSITION(S) LIT
0244	7200		CLA	
0245	6552		BCH+CLR	CLEAR BCR-HI
0246	6551		BCH+RD	
0247	7440		SZA	DID IT WORK?
0250	7402		HLT	** BCH CLEAR ERROR. ALSO POSSIBLE ** THAT READ IS IN ERROR.
		*		
		*		
		*		
		*		NOW CHECK WORD MODE READ/WRITE
		*		
0251	7200		CLA	
0252	6567		LCM12	SET WORD MODE
0253	7240		STA	
0254	6554		BCH+WR	LOAD BCR THRU WORD GATES
0255	7440		SZA	AC SHOULD BE CLEARED
0256	7402		HLT	** AC CLEAR ERROR
0257	7200		CLA	

201 CYCLIC CHECK COMPUTER - TEST ROUTINES

0260	6551	BCH+RD	DID WORD GET THERE
0261	7040	CMA	
0262	7440	SZA	AC SHOULD BE ZERO
0263	7402	HLT	** READ/WRITE GATE OR LOAD ERROR
	*		** FAULTY BIT POSITION(S) LIT
0264	7200	CLA	
0265	6552	BCH+CLR	
0266	6551	BCH+RD	
0267	7440	SZA	DID REGISTER CLEAR WORK
0270	7402	HLT	** BCH CLEAR ERROR. ALSO POSSIBLE
	*		** THAT READ IS IN ERROR.
	*		
	*		
	*		
	*		NOW CHECK THE SHIFTING CAPABILITIES
	*		OF THE CHAR BUF AND BCR REGISTERS.
	*		
	*		SHIFT 5 CHECK
	*		
0271	7200	CLA	
0272	6561	LCM6	SET 6-BIT WORD MODE
0273	1024	TAD K0100	
0274	6556	BCH+CLR+WR	
0275	6571	CCC	START CHECK SUM COMPUTATION. IF
	*		PROCESSOR STOPS HERE, THE PDP-8
	*		'HOLD' CIRCUITRY HAS MALFUNCTIONED
	*		OR THE CHECK ADAPTOR RUN FF HAS
	*		NOT RESET AT SHIFT-END CNT.
	*		THIS IS TRUE FOR ALL "CCC'S".
0276	6551	BCH+RD	
0277	7110	CLL+RAR	FLAG SHOULD NOW BE IN LINK
0300	7440	SZA	IF ALL ELSE IS OK, AC SHOULD=0
0301	7402	HLT	** BAD SHIFT COUNT, JAMS, OR ADDERS
0302	7620	SNL+CLA	NOW CHECK FOR FLAG BIT
0303	7402	HLT	** PROBABLY BAD ADDER OR CBUF
	*		
	*		SHIFT 8 CHECK
	*		
0304	6563	LCM8	SET 8-BIT BYTE MODE
0305	7001	IAC	
0306	6556	BCH+CLR+WR	SET FLAG
0307	6542	BCL+CLR	
0310	6571	CCC	
0311	6541	BCL+RD	FLAG SHOULD NOW BE IN LINK
0312	7110	CLL+RAR	FLAG SHOULD NOW BE IN LINK
0313	7440	SZA	AC SHOULD = 0
0314	7402	HLT	** BAD SHIFT CNT, JAMS, OR ADDERS
0315	7620	SNL+CLA	NOW TEST FOR FLAG (L=1)
0316	7402	HLT	** BAD SHIFT CNT, JAMS, OR ADDERS
0317	7620	SNL+CLA	NOW TEST FOR FLAG
0320	7402	HLT	** NOT THERE. BAD ADDER OR CBUF?
	*		
	*		SHIFT 12 CHECK
	*		
0321	1030	TAD K4000	12TH BIT FLAG
0322	6567	LCM12	
0323	6552	BCH+CLR	
0324	6571	CCC	

 201 CYCLIC CHECK COMPUTER - TEST ROUTINES

0325	6551		BCH+RD		
0326	1032		TAD	M7401	
0327	7450		SNA		AC SHOULD=0
0330	5735		JMP*	MTAPTR	OK...ON TO NEXT PAGE
0331	1031		TAD	K7401	RESTORE AC
0332	7402		HLT		** BAD ADDER OR CBUF?
		*			
		*			
		*			
		*		CONSTANTS	
		*			
0333	1004	CRLF	DC	PRNT1	
0334	1031	CRLF	DC	COM1E-2	
0335	0400	MTAPTR	DC	MTA00W	

201 CYCLIC CHECK COMPUTER - TEST ROUTINES

```

*****
*
*          CYCLIC CHECK COMPUTER - TEST ROUTINES
*          PAGE 2
*
*****

```

```

0400          ORG          400

```

```

*
*          NOTE:
*          ADDER 1 IS BETWEEN BCR(11)/BCR(15) & CBUF(11)
*          ADDER 2 IS BETWEEN BCR(0) AND BCR(1)
*          ADDER 3 IS BETWEEN BCR(1) AND BCR(2)
*          ADDER 4 IS BETWEEN BCR(2) AND BCR(3)
*          ADDER 5 IS BETWEEN BCR(10) AND BCR(11)
*          ADDER 6 IS BETWEEN BCR(15) AND BCR(16)
*
*
*          NOW CHECK THE MOD 2 ADDERS DERIVING THE
*          FEEDBACK FROM BCR(11).
*          TESTS ADDERS 1,2,3,4, AND 5
*
*          0+0 MOD 2 - WORD MODE
*

```

```

0400 7200 MTA00W CLA
0401 6567          LCM12          0 TO ADDER INPUTS
0402 6552          BCH+CLR        0 TO FB TAP
0403 6571          CCC
0404 6551          BCH+RD
0405 7440          SZA            BCR 0-11 SHOULD EQUAL 0
0406 7402          HLT            ** ADDER ERRORS?

```

```

*
*          0+1 MOD 2 - WORD MODE
*

```

```

0407 7200 MTA01W CLA
0410 1030          TAD           K4000      1 IN FOR LAST SHIFT
0411 6567          LCM12          LOAD CHAR BUF AND SET MODE
0412 6552          BCH+CLR        0 OUT FOR 1 ON FB TAP
0413 6571          CCC
0414 6551          BCH+RD          BCR 0-11 SHOULD = 7401
0415 1032          TAD           M7401     (-K7401)
0416 7450          SNA            DCES IT
0417 5222          JMP           MTA10W    YES..
0420 1031          TAD           K7401     NO...RECONSTRUCT AC
0421 7402          HLT            ** ADDER ERRORS?

```

```

*          ** CORRECT AC = 7401
*
*          1+0 MOD 2 - WORD MODE
*

```

```

0422 7200 MTA10W CLA          INPUT      0
0423 6567          LCM12
0424 7130          STL+RAR        OUTPUT 1 FOR 1 ON FB TAP
0425 6556          BCH+CLR+WR
0426 6571          CCC
0427 6551          BCH+RD
0430 1032          TAD           M7401     BCR 0-11 SHOULD EQUAL 7401
0431 7450          SNA            DCES IT?

```

201 CYCLIC CHECK COMPUTER - TEST ROUTINES

0432	5235	JMP	MTA11W	YES..
0433	1031	TAD	K7401	NO...RECONSTRUCT AC
0434	7402	HLT		** ADDER ERRORS?
		*		*** CORRECT AC = 7401
		*		
		*	1+1 MOD 2 - WORD MODE	
		*		
0435	7200	MTA11W	CLA	
0436	1030	TAD	K4000	1 IN ON LAST SHIFT
0437	6567		LCM12	
0440	7130		STL+RAR	1 OUT FOR 0 ON FBTAP
0441	6556		BCH+CLR+WR	
0442	6571		CCC	
0443	6551		BCH+RD	BCR 0-11 SHOULD EQUAL 0
0444	7440		SZA	
0445	7402	HLT		** ADDER ERRORS?
		*		** CORRECT AC = 0
		*		
		*		
		*	NOW CHECK THE ADDERS DERIVING THE	
		*	FEEDBACK FROM BCR(15).	
		*	TESTS ADDERS 1,3, AND 6	
		*		
		*	0+0 MOD 2 - BYTE MODE	
		*		
0446	7200	MTA00B	CLA	
0447	6563		LCMR	SET BYTE MODE.
0450	6552		BCH+CLR	0'S FOR FB TAP AND ADDER INPUTS
0451	6542		BCL+CLR	
0452	6571		CCC	
0453	6551		BCH+RD	SHOULD=0
0454	7450		SNA	
0455	5257	JMP	M8001	OK..THEN CHECK ADDER 6
0456	7402	HLT		** ADDER 1 AND/OR 3 ERROR?
		*		** CORRECT AC = 0
0457	7200	M8001	CLA	
0460	6541		BCL+RD	
0461	7110		CLL+RAR	BCR(15) SHOULD HAVE BEEN 0
0462	7620		SNL+CLA	WAS IT?
0463	5266	JMP	MTA01B	YES..
0464	6541		BCL+RD	PUT LG-BCR IN AC
0465	7402	HLT		** ADDER 6 ERROR
		*		** CORRECT AC = 0
		*		
		*	0+1 MOD 2 - BYTE MODE	
		*		
0466	7200	MTA01B	CLA	
0467	1025	TAD	K0200	1 IN FOR LAST SHIFT
0470	6563		LCM8	
0471	6552		BCH+CLR	0 OUT FOR 1 ON FBTAP
0472	6542		BCL+CLR	
0473	6571		CCC	STOP THE WORLD I WANT TO GET OFF
0474	6551		BCH+RD	
0475	1033	TAD	M0240	(-K0240)
0476	7450		SNA	AC SHOULD EQUAL 0
0477	5302	JMP	M8011	IT DID. THEN CHECK ADDER 6
0500	1026	TAD	K0240	** ERROR RECONSTRUCT AC
0501	7402	HLT		** ADDER 1 AND/OR 3 ERRORS?

201 CYCLIC CHECK COMPUTER - TEST ROUTINES

		*			** CORRECT AC = 0
0502	6541	MB011	BCL+RD	ADDER	6 CHECK
0503	7010		RAR		
0504	7430		SZL		BCR(15) SHOULD EQUAL 1
0505	5310		JMP	MTA10B	IT DOES
0506	7004		RAL		OH, OH. RECONSTRUCT AC
0507	7402		HLT		ADDER 6 ERROR
		*			** CORRECT AC = 0001
		*			
		*		1+0 MOD 2 - BYTE MODE	
		*			
0510	7200	MTA10B	CLA	INPUT	0 FOR SHIFT 8
0511	6563		LCMB		SET BYTE MODE
0512	6552		BCH+CLR		0 TO ADDERS 1 AND 3
0513	1025		TAD	K0200	1 OUT FOR 1 ON FBTAP
0514	6546		BCL+CLR+WR		
0515	6571		CCC		
0516	6551		BCH+RD		
0517	1033		TAD	M0240	(-K0240)
0520	7450		SNA		AC SHOULD EQUAL 0
0521	5324		JMP	MB101	IT DOES, THEN CHECK ADDER 6
0522	1026		TAD	K0240	ERROR. RECONSTRUCT AC
0523	7402		HLT		** ADDER 1 AND/OR 3 ERRORS?
		*			** AC SHOULD = 0
0524	6541	MB101	BCL+RD		ADDER 6 CHECK
0525	7010		RAR		
0526	7430		SZL		BCR(15) SHOULD EQUAL 1
0527	5332		JMP	MTA11B	OK
0530	7004		RAL		RESTORE AC
0531	7402		HLT		** ADDER 6 ERROR
		*			** AC SHOULD = 0001
		*			
		*		1+1 MOD 2 - BYTE MODE	
		*			
0532	7200	MTA11B	CLA		
0533	1025		TAD	K0200	1 IN FOR LAST SHIFT
0534	6563		LCMB		
0535	6552		BCH+CLR		
0536	1025		TAD	K0200	1 OUT FOR 0 ON FBTAP
0537	6546		BCL+CLR+WR		
0540	6571		CCC		
0541	6551		BCH+RD		
0542	7450		SNA		AC SHOULD EQUAL 0
0543	5345		JMP	MB111	OK, THEN TEST ADDER 6
0544	7402		HLT		** ADDER 1 AND/OR 3 ERRORS?
		*			** CORRECT AC = 0
0545	7200	MB111	CLA		
0546	6541		BCL+RD		
0547	7450		SNA		BCL SHOULD = 0
0550	5353		JMP	LOOP	OK..THEN ON TO CHECK LOOP COUNT
0551	7402		HLT		** ADDER 6 ERROR
		*			** CORRECT AC = 0
0552	7200		CLA		
0553	2022	LOOP	ISZ	CNT	
0554	5757		JMP*	PTR1	BACK AND DO IT AGAIN
0555	5756		JMP*	COMP1P	DONE, THEN ON TO NEXT PAGE
		*			
0556	0600	COMP1P	DC	RCWM1	

201 CYCLIC CHECK COMPUTER - TEST ROUTINES

0557 C210 PTR1 DC START+2

201 CYCLIC CHECK COMPUTER - TEST ROUTINES

```
*****
*
*           CYCLIC CHECK COMPUTER - TEST ROUTINES
*           PAGE 3
*
*****
```

0600		DRG	600		
*					NOW LET'S CHECK OPERATION BY COMPUTING
*					CHECKSUM BY SOFTWARE AND HARDWARE METHODS
*					THEN COMPARE THE RESULTS.
*					
*					RESIDUE CHECK - WORD MODE
*					GENERATOR POLYNOMIAL :
*					$X^{12} + X^{11} + X^3 + X^2 + X + 1$
*					
0600	3370	RCWM1	DCA	WCNT	-4096 TO COUNT
0601	3355		DCA	HGHI	ZERO HARDWARE RESIDUE SAVE AREA
0602	3357		DCA	BCHBUF	LIKEWISE TO SOFTWARE RESIDUE
0603	1370	PCWM2	TAD	WCNT	USE THIS AS INPUT CHARACTER
0604	6567		LCM12		SET 12-BIT WORD MODE
0605	1355		TAD	HGHI	
0606	6556			BCH+CLR+WP	
0607	6571			CCC	
0610	6551			BCH+RD	
0611	3355		DCA	HGHI	SAVE HARDWARE GENERATED RESIDUE
*					
*					NOW LET'S COMPUTE IT BY SOFTWARE
0612	1034		TAD	M0012D	(-12 DECIMAL)
0613	3371		DCA	SCNT	SET UP SHIFT COUNTER
0614	1370		TAD	WCNT	INPUT CHARACTER
0615	3367		DCA	CBUF	PRESERVE INPUT CHARACTER
0616	1357	PCWM3	TAD	BCHBUF	FETCH BCC RESIDUE
0617	7110			CLL+RAR	SAVE LSB IN LINK
0620	3357		DCA	BCHBUF	
0621	1367		TAD	CBUF	GET CHARACTER
0622	7010		RAR		CHAR LSB TO LINK;BCC LSB TO AC(0)
0623	7510		SPA		THIS + NEXT FORMS XOR BETWEEN
0624	7020		CML		LSB'S OF BCC AND CBUF
0625	3367		DCA	CBUF	SAVE CHAR
0626	7420		SNL		IS FEEDBACK BIT A 1
0627	5242		JMP	RCWM4	NO...
0630	1357		TAD	BCHBUF	YES..MORE WORK TO BE DONE
0631	0031		AND	K7401	XOR FB TAP INTO BCC
0632	7041			CIA	
0633	7104			CLL+RAL	
0634	1357		TAD	BCHBUF	
0635	1031		TAD	K7401	
0636	2371		ISZ	SCNT	BUMP SHIFT COUNTER
0637	5217		JMP	RCWM3+1	
0640	3357		DCA	BCHBUF	SAVE FINAL RESIDUE
0641	5257		JMP	RCWM5	
0642	2371	PCWM4	ISZ	SCNT	BUMP SHIFT COUNTER
0643	5216		JMP	RCWM3	BACK FOR MORE SHIFTS
0644	1355		TAD	HGHI	DONE..THEN LET'S COMPARE RESULTS
0645	7041			CIA	
0646	1357		TAD	BCHBUF	

201 CYCLIC CHECK COMPUTER - TEST ROUTINES

0647	7650		SNA+CLA		ARE THEY EQUAL?
0650	5257		JMP	RCWM5	YES..
0651	1357		TAD	BCHBUF	NO...THEN SET UP AC & MQ FOR HALT
0652	7421		MQL		SOFTWARE RESIDUE TO MQ
0653	1355		TAD	HGHI	HARDWARE RESIDUE TO AC
0654	7402		HLT		** ERROR IN RESIDUE COMPUTATION.
		*			
0655	7200		CLA		
0656	5261		JMP	RCBM1	4096 TIMES AROUND THE
0657	2370	RCWM5	ISZ	WCNT	MULBERRY BUSH.
0660	5203		JMP	RCWM2	
		*			
		*			
		*			
		*			RESIDUE CHECK - BYTE MODE
		*			GENERATOR POLYNOMIAL:
		*			$X^{**16} + X^{**15} + X^{**2} + 1$
		*			
0661	3355	RCBM1	DCA	HGHI	ZERO ALL THE RESIDUE SAVE AREAS
0662	3356		DCA	HGLO	
0663	3357		DCA	BCHBUF	
0664	3360		DCA	BCLBUF	
		*			NOTE: THE FIRST TIME THROUGH WCNT
		*			SINCE IT JUST OVERFLOWED.
0665	1370	RCBM2	TAD	WCNT	INPUT CHARACTER EMULATOR
0666	6563		LCM8		SET 8-BIT BYTE MODE
0667	1355		TAD	HGHI	LOAD RESIDUE REGISTERS
0670	6556		BCH+CLR+WR		
0671	1356		TAD	HGLO	
0672	6546		BCL+CLR+WR		
0673	6571		CCC		
0674	6551		BCH+RD		SAVE HARDWARE GENERATED CHECKSUM
0675	3355		DCA	HGHI	
0676	6541		BCL+RD		
0677	3356		DCA	HGLO	
		*			
		*			NOW LET'S COMPUTE IT BY SOFTWARE
0700	1035		TAD	M0008D	(-8 DECIMAL)
0701	3371		DCA	SCNT	SHIFT COUNTER
0702	1370		TAD	WCNT	
0703	3367		DCA	CBUF	
0704	1357	RCBM3	TAD	BCHBUF	FETCH LEFT-BYTE
0705	7110		CLL+RAR		SAVE LSB IN LINK
0706	3357		DCA	BCHBUF	
0707	7012		RTR		
0710	7012		RTR		
0711	1360		TAD	BCLBUF	
0712	7110		CLL+RAR		LSB OF RIGHT-BYTE TO LINK
0713	3360		DCA	BCLBUF	
0714	1367		TAD	CBUF	FETCH INPUT CHARACTER
0715	7010		RAR		GET IT'S LSB
0716	7510		SPA		THIS + NEXT FORMS XOR BETWEEN
0717	7020		CML		LSB'S OF BCC AND CHAR
0720	3367		DCA	CBUF	SAVE CHARACTER
0721	7420		SNL		IS FEEDBACK BIT = 1?
0722	5342		JMP	RCBM4	NO...LIFE IS EASY
0723	1360		TAD	BCLBUF	XOR FB LINE INTO BCR(15)
0724	7110		CLL+RAR		

201 CYCLIC CHECK COMPUTER - TEST ROUTINES

0725	7020		CML	
0726	7004		RAL	
0727	3350		DCA	BCLBUF
0730	1357		TAD	RCHRUF NOW GET HIGH-BYTE
0731	0026		AND	K024C XOR FRTAP INTO HIGH BCC
0732	7041		CIA	
0733	7104		CLL+RAL	
0734	1357		TAD	BCHBUF
0735	1026		TAD	K0240
0736	2371		ISZ	SCNT HAVE THERE BEEN 8 SHIFTS?
0737	5305		JMP	RCBM3+1 NO...THEN GO BACK
0740	3357		DCA	BCHBUF YES...SAVE FINAL RESIDUE
0741	5344		JMP	RCBM5
0742	2371	RCBM4	ISZ	SCNT HAVE THERE BEEN 8 SHIFTS?
0743	5304		JMP	RCBM3 NO...GO BACK
0744	1355	RCBM5	TAD	HGHI COMPARE RESULTS
0745	7041		CIA	
0746	1357		TAD	BCHBUF
0747	1356		TAD	HGLO
0750	7041		CIA	
0751	1360		TAD	BCLBUF LIKE A BIG POT OF HONEY
0752	7650		SNA+CLA	AC SHOULD EQUAL 0 AT THIS POINT
0753	5362		JMP	RCBM6 OK
0754	7402		HLT	** ERROR IN RESIDUE COMPUTATION.
		*		** EXAMINE THE NEXT FOUR LOCATION
		*		** TO RECOVER THE CONTENTS OF THE
		*		** HARDWARE AND SOFTWARE CHECKSUM
		*		** REGISTERS.
0755		HGHI	DS	1
0756		HGLO	DS	1
0757		BCHBUF	DS	1
0760		BCLBUF	DS	1
0761	5766		JMP*	BACK
0762	2370	RCBM6	ISZ	WCNT HAVE WE DONE THIS 4096 TIMES
0763	5265		JMP	RCBM2 NOT YET.
0764	5765		JMP*	OUTPTR YES..
		*		
		*		CONSTANTS
		*		
0765	1000	OUTPTR	DC	OUTPRT
0766	0206	BACK	DC	START
		*		
		*		VARIABLES
		*		
0767		CBUF	DS	1
0770		WCNT	DS	1
0771		SCNT	DS	1

201 CYCLIC CHECK COMPUTER - TEST ROUTINES

 *
 * CYCLIC CHECK COMPUTER - TEST ROUTINES *
 * PAGE 4 *

1000 ORG 1000

*
 *
 *
 *

PRINTED CONFIRMATION OF SUCCESSFUL TEST

1000	1215	OUTPRT	TAD	COMLP	STARTING ADDRESS OF MESSAGE
1001	3010	DCA	AXR1	PUT IT IN AUTO INDEX REGISTER	
1002	1615	TAD*	COMLP	GET MESSAGE LENGTH	
1003	3021	DCA	CHRCNT		
1004	1410	TAD*	AXR1	FETCH NEXT CHARACTER	
1005	6046	TLS			
1006	7200	CLA			
1007	2021	ISZ	CHRCNT	BUMP CHARACTER COUNTER	
1010	7410	SKP			
1011	5616	JMP*	REGIN	ALL DONE. BACK FOR MORE.	
1012	6041	TSE			
1013	5212	JMP	*-1		
1014	5204	JMP	PRNT1	BACK TO PRINT NEXT CHARACTER	

CONSTANTS

1015	1017	COM1P	DC	COM1
1016	0206	BEGIN	DC	START

VARIABLES

*
 *
 *

201 CYCLIC CHECK COMPUTER - TEST ROUTINES

 *
 *
 *
 *

 CYCLIC CHECK COMPUTER - TEST ROUTINES
 INDIRECT STORAGE & LITERAL POOL

 *
 *
 *
 *

MESSAGE

		MESSAGE	
		COM1	DC
1017	7764	COM1	DC
1020	0325		325
1021	0316		316
1022	0311		311
1023	0324		324
1024	0240		240
1025	0303		303
1026	0310		310
1027	0305		305
1030	0303		303
1031	0313		313
1032	0215		215
1033	0212	CCMIE	212

COM1-COMIE (-LENGTH) "UNIT CHECK"

201 CYCLIC CHECK COMPUTER - TEST ROUTINES

	0200	END	200										
OPERAND CROSS-REFERENCE LISTING													
AXR1	0010	0202	1001	1004									
BACK	0766	0761											
RCHBUF	0757	0602	0616	0620	0630	0634	0640	0646	0651	0663	0704	0706	0730
		0734	0740	0746									
RCLBUF	0760	0664	0711	0713	0723	0727	0751						
BEGIN	1016	1011											
CRUF	0767	0615	0621	0625	0703	0714	0720						
CHRCNT	0021	0204	1003	1007									
CNT	0022	0207	0553										
COMPIP	0556	0555											
COM1	1017	1015	1017										
COM1E	1033	0334	1017										
COMIP	1015	1000	1002										
CRLF	0333	0205											
CRLF1P	0334	0201											
HGHI	0755	0601	0605	0611	0644	0653	0661	0667	0675	0744			
HGLO	0756	0662	0671	0677	0747								
K0040	0023												
K0100	0024	0273											
K0200	0025	0467	0513	0533	0536								
K0240	0026	0500	0522	0731	0735								
K0377	0027	0223	0241										
K4000	0030	0321	0410	0436									
K7401	0031	0331	0420	0433	0631	0635							
LOOP	0553	0550											
MB001	0457	0455											
MB011	0502	0477											
MB101	0524	0521											
MB111	0545	0543											
MTAPTR	0325	0330											
MTA00B	0446												
MTACCW	0400	0335											
MTA01B	0466	0463											
MTA01W	0407												
MTA10B	0510	0505											
MTA10W	0422	0417											
MTA11B	0532	0527											
MTA11W	0435	0432											
M0008D	0035	0700											
M0012D	0034	0612											
M0240	0033	0475	0517										
M7401	0032	0326	0415	0430									
OUTPRT	1000	0765											
OUTPTR	0765	0764											
PRNT1	1004	0333	1014										
PTR1	0557	0554											
RCBM1	0661	0656											
RCBM2	0665	0763											
RCBM3	0704	0737	0743										
RCBM4	0742	0722											
RCBM5	0744	0741											
RCBM6	0762	0753											
RCWM1	0600	0556											

201 CYCLIC CHECK COMPUTER - TEST ROUTINES

RCWM2	0603	0660
RCWM3	0616	0637 0643
RCWM4	0642	0627
RCWM5	0657	0641 0650
SCNT	0771	0613 0636 0642 0701 0736 0742
START	0206	0557 0766 1016
TEMPI	0020	
WCNT	0770	0600 0603 0614 0657 0665 0702 0762

OPERATOR CROSS-REFERENCE LISTING

BCH	6550	0234	0237	0245	0246	0254	0260	0265	0266	0274	0276	0306	0323
		0325	0402	0404	0412	0414	0425	0427	0441	0443	0450	0453	0471
		0474	0512	0516	0535	0541	0606	0610	0670	0674			
BCL	6540	0215	0221	0227	0230	0307	0311	0451	0460	0464	0472	0502	0514
		0524	0537	0546	0672	0676							
CCC	6571	0275	0310	0324	0403	0413	0426	0442	0452	0473	0515	0540	0607
		0673											
CLR	0002	0227	0245	0265	0274	0306	0307	0323	0402	0412	0425	0441	0450
		0451	0471	0472	0512	0514	0535	0537	0606	0670	0672		
LCM12	6567	0252	0322	0401	0411	0423	0437	0604					
LCM6	6561	0272											
LCM8	6563	0211	0304	0447	0470	0511	0534	0666					
RD	0001	0221	0230	0237	0246	0260	0266	0276	0311	0325	0404	0414	0427
		0443	0453	0460	0464	0474	0502	0516	0524	0541	0546	0610	0674
		0676											
WR	0004	0215	0234	0254	0274	0306	0425	0441	0514	0537	0606	0670	0672

ERRORS 0	SCARDS 611	SPRINT 694	SPUNCH 8	STORAGE 9
----------	------------	------------	----------	-----------

Unclassified

Security Classification

DOCUMENT CONTROL DATA - R&D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) THE UNIVERSITY OF MICHIGAN CONCOMP PROJECT		2 a. REPORT SECURITY CLASSIFICATION Unclassified
		2 b. GROUP
3. REPORT TITLE A CYCLIC CHECK COMPUTER FOR ERROR DETECTION		
4. DESCRIPTIVE NOTES (Type of report and inclusive dates) Technical Report		
5. AUTHOR(S) (Last name, first name, initial) Kenneth E. Burkhalter		
6. REPORT DATE June 1968	7 a. TOTAL NO. OF PAGES 77	7 b. NO. OF REFS
8 a. CONTRACT OR GRANT NO. DA-49-083 OSA-3050	9 a. ORIGINATOR'S REPORT NUMBER(S) Memorandum 19	
b. PROJECT NO.	9 b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)	
c.		
d.		
10. AVAILABILITY/LIMITATION NOTICES Qualified requesters may obtain copies of this report from DDC.		
11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY Advanced Research Projects Agency	
13. ABSTRACT This report discusses the design and use of equipment built to aid intercomputer communications via serial-synchronous data transmission techniques. The interface described computes on a character-by-character basis, a cyclic redundancy block checksum which is appended to outgoing or checked against incoming messages. This hardware technique reduces checksum computation on a small computer from several hundred microseconds per character to only several microseconds; a reduction that is necessary if more than several 201 type data modems are to be operated simultaneously under control of a single processor. Basic design objectives and decisions are described first. A brief overall system description with background information is then followed by programming considerations and detailed descriptions of the checksum computer logic. Finally diagnostic software and wirewrap documentation is provided for maintenance and/or reproduction purposes.		

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Cyclic Redundancy Check						
Block Check Computation						
Serial-Synchronous Data Transmission						
Logical Design						
Digital Computer Interface						
Maximal Linear Shift Register Sequence						

INSTRUCTIONS

1. ORIGINATING ACTIVITY: Enter the name and address of the contractor, subcontractor, grantee, Department of Defense activity or other organization (*corporate author*) issuing the report.

2a. REPORT SECURITY CLASSIFICATION: Enter the overall security classification of the report. Indicate whether "Restricted Data" is included. Marking is to be in accordance with appropriate security regulations.

2b. GROUP: Automatic downgrading is specified in DoD Directive 5200.10 and Armed Forces Industrial Manual. Enter the group number. Also, when applicable, show that optional markings have been used for Group 3 and Group 4 as authorized.

3. REPORT TITLE: Enter the complete report title in all capital letters. Titles in all cases should be unclassified. If a meaningful title cannot be selected without classification, show title classification in all capitals in parenthesis immediately following the title.

4. DESCRIPTIVE NOTES: If appropriate, enter the type of report, e.g., interim, progress, summary, annual, or final. Give the inclusive dates when a specific reporting period is covered.

5. AUTHOR(S): Enter the name(s) of author(s) as shown on or in the report. Enter last name, first name, middle initial. If military, show rank and branch of service. The name of the principal author is an absolute minimum requirement.

6. REPORT DATE: Enter the date of the report as day, month, year; or month, year. If more than one date appears on the report, use date of publication.

7a. TOTAL NUMBER OF PAGES: The total page count should follow normal pagination procedures, i.e., enter the number of pages containing information.

7b. NUMBER OF REFERENCES: Enter the total number of references cited in the report.

8a. CONTRACT OR GRANT NUMBER: If appropriate, enter the applicable number of the contract or grant under which the report was written.

8b, 8c, & 8d. PROJECT NUMBER: Enter the appropriate military department identification, such as project number, subproject number, system numbers, task number, etc.

9a. ORIGINATOR'S REPORT NUMBER(S): Enter the official report number by which the document will be identified and controlled by the originating activity. This number must be unique to this report.

9b. OTHER REPORT NUMBER(S): If the report has been assigned any other report numbers (*either by the originator or by the sponsor*), also enter this number(s).

10. AVAILABILITY/LIMITATION NOTICES: Enter any limitations on further dissemination of the report, other than those

imposed by security classification, using standard statements such as:

- (1) "Qualified requesters may obtain copies of this report from DDC."
- (2) "Foreign announcement and dissemination of this report by DDC is not authorized."
- (3) "U. S. Government agencies may obtain copies of this report directly from DDC. Other qualified DDC users shall request through _____."
- (4) "U. S. military agencies may obtain copies of this report directly from DDC. Other qualified users shall request through _____."
- (5) "All distribution of this report is controlled. Qualified DDC users shall request through _____."

If the report has been furnished to the Office of Technical Services, Department of Commerce, for sale to the public, indicate this fact and enter the price, if known.

11. SUPPLEMENTARY NOTES: Use for additional explanatory notes.

12. SPONSORING MILITARY ACTIVITY: Enter the name of the departmental project office or laboratory sponsoring (*paying for*) the research and development. Include address.

13. ABSTRACT: Enter an abstract giving a brief and factual summary of the document indicative of the report, even though it may also appear elsewhere in the body of the technical report. If additional space is required, a continuation sheet shall be attached.

It is highly desirable that the abstract of classified reports be unclassified. Each paragraph of the abstract shall end with an indication of the military security classification of the information in the paragraph, represented as (TS), (S), (C), or (U).

There is no limitation on the length of the abstract. However, the suggested length is from 150 to 225 words.

14. KEY WORDS: Key words are technically meaningful terms or short phrases that characterize a report and may be used as index entries for cataloging the report. Key words must be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location, may be used as key words but will be followed by an indication of technical context. The assignment of links, rules, and weights is optional.



THE UNIVERSITY OF MICHIGAN

DATE DUE

412 13i57