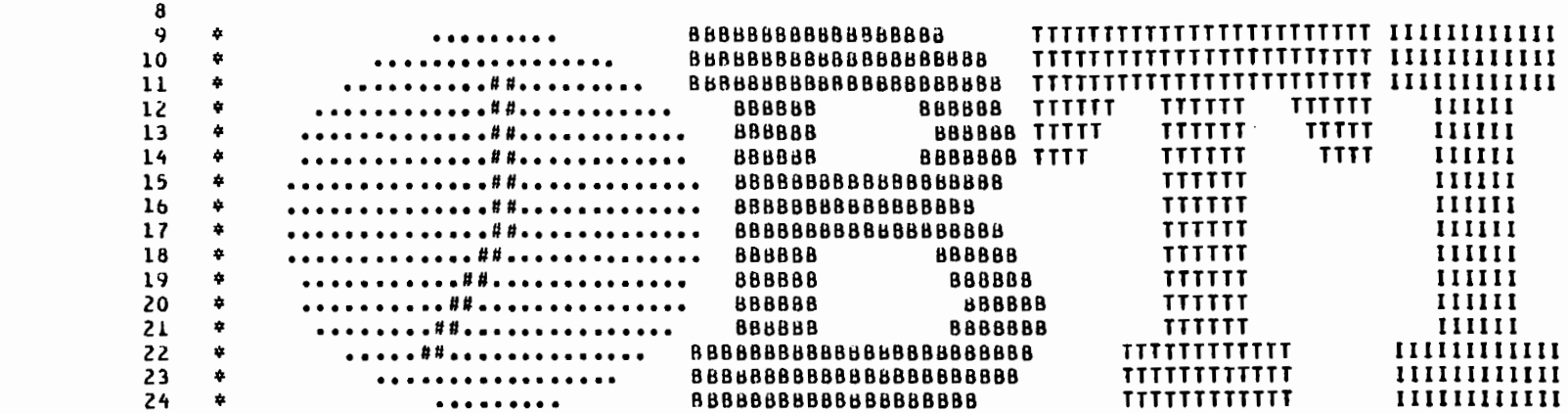
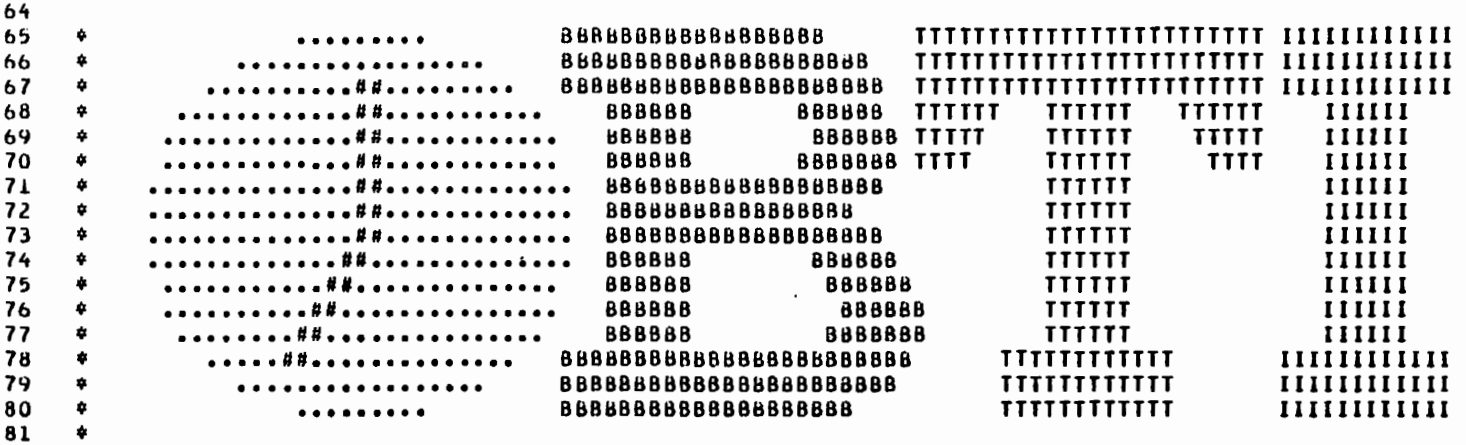
```
                INCLUDE  CTCC1.TEXT1
```

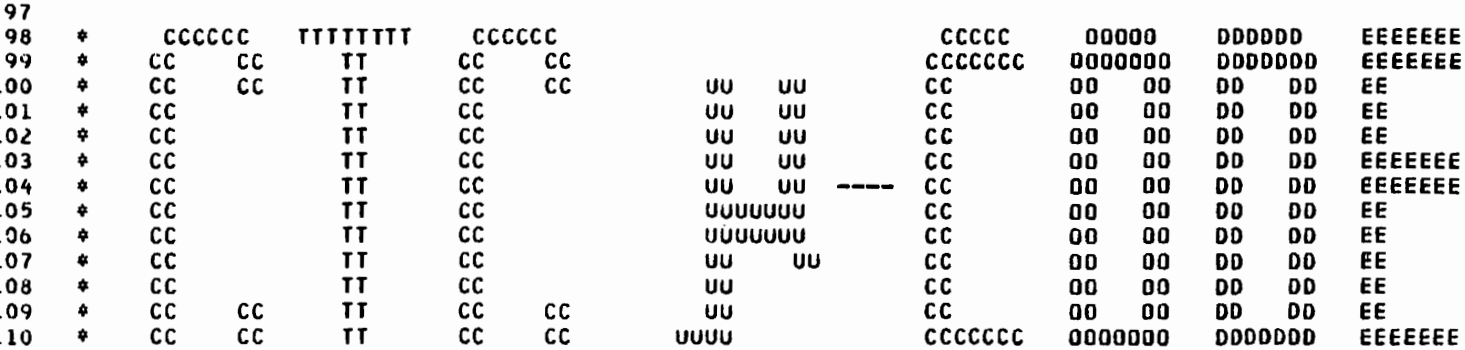


BTI Computer Systems
Sunnyvale, California 94086


```
 CCCCCC   TTTTTTTT    CCCCCC                          CCCCC     00000    DDDDDD    EEEEEEE
CC    CC     TT      CC    CC                       CCCCCCC   0000000   DDDDDDD   EEEEEEE
CC    CC     TT      CC    CC        UU    UU       CC        00   00   DD   DD   EE
CC           TT      CC              UU    UU       CC        00   00   DD   DD   EE
CC           TT      CC              UU    UU       CC        00   00   DD   DD   EE
CC           TT      CC              UU    UU       CC        00   00   DD   DD   EEEEEEE
CC           TT      CC              UU    UU ----  CC        00   00   DD   DD   EEEEEEE
CC           TT      CC              UUUUUUUU       CC        00   00   DD   DD   EE
CC           TT      CC              UUUUUUU        CC        00   00   DD   DD   EE
CC           TT      CC              UU    UU       CC        00   00   DD   DD   EE
CC           TT      CC              UU             CC        00   00   DD   DD   EE
CC    CC     TT      CC    CC        UU             CC        00   00   DD   DD   EE
CC    CC     TT      CC    CC       UUUU            CCCCCCC   0000000   DDDDDDD   EEEEEEE
 CCCCCC      TT       CCCCCC         UU              CCCCC     00000    DDDDDD    EEEEEEE
```



BTI Computer Systems
Sunnyvale, California 94086



```
                    116  ********************************************************************************
                    117  * A FEW CONSTANTS DEFINED                                                      *
                    118  ********************************************************************************
                    119
     0096           120   STORE        EQU      #96          #96 IS THE FIRST BYTE OF A STORE COMMAND FROM THE PPU
     0062           121   LOAD         EQU      #62          #62 IS THE FIRST BYTE OF A LOAD REQUEST FROM THE PPU
     00F8           122   NACK         EQU      #F8          #F8 IS DEFINED AS THE "NOT-ACKNOWLEDGE" BYTE
     000E           123   ACK          EQU      #0E          #0E IS DEFINED AS THE "ACKNOWLEDGE" BYTE
     0090           124   EOR          EQU      #90          #90 IS DEFINED AS THE "END-OF-RECORD" BYTE
     0060           125   GO           EQU      #60          #60 IS DEFINED AS THE "GO" BYTE
```

```
                128  ********************************************************************************
                129  * RAM USE IS SPECIFIED, NAMED, AND DESCRIBED                                   *
                130  ********************************************************************************
                131
                132  >>>>>>>>>>>>>>>>>>>>
0000            133  >DRSTS0        EQU      #00          DRIVE #0'S STATUS IS STORED HERE
0001            134  >DRSTS1        EQU      #01          DRIVE #1'S STATUS IS STORED HERE
0002            135  >DRSTS2        EQU      #02          DRIVE #2'S STATUS IS STORED HERE
0003            136  >DRSTS3        EQU      #03          DRIVE #3'S STATUS IS STORED HERE
                137  >>>>>>>>>>>>>>>>>>>>
0004            138  >OPDRSTSX      EQU      #04          LOCATION OF FIRST BYTE WHEN SENDING DRIVE STATUS'ES TO PPU
0005            139  >OPDRSTS0      EQU      #05          OPERATING SYSTEM'S DRIVE #0'S STATUS IS STORE HERE
0006            140  >OPDRSTS1      EQU      #06                               DRIVE #1'S STATUS
0007            141  >OPDRSTS2      EQU      #07                               DRIVE #2'S STATUS
0008            142  >OPDRSTS3      EQU      #08                               DRIVE #3'S STATUS
0009            143  >OPDRSTSZ      EQU      #09          LOCATION OF Block Check Character WHEN SENDING DRIVE STATUS'ES
                144  >>>>>>>>>>>>>>>>>>>>
000A            145  >DESELECT0     EQU      #0A          INFORMATION TO DE-SELECT DRIVE #0 IS STORED HERE
000B            146  >DESELECT1     EQU      #0B                                      DRIVE #1
000C            147  >DESELECT2     EQU      #0C                                      DRIVE #2
000D            148  >DESELECT3     EQU      #0D                                      DRIVE #3
                149  >>>>>>>>>>>>>>>>>>>>
000E            150  >ID0           EQU      #0E          LOCATION OF FIRST BYTE WHEN SENDING DEVICE ID TO PPU
000F            151  >ID1           EQU      #0F          DEVICE ID BYTE WHICH TELLS WHICH DEVICE I AM
0010            152  >ID2           EQU      #10          DEVICE ID BYTE WHICH TELLS WHICH MICRO-CODE THIS IS
0011            153  >ID3           EQU      #11          DEVICE ID BYTE WHICH TELLS RESULTS OF SELF-TEST
0012            154  >ID4           EQU      #12          DEVICE ID BYTE WHICH ISN'T USED AND = #00
0013            155  >ID5           EQU      #13          LOCATION OF Block Check Character WHEN SENDING DEVICE ID
                156  >>>>>>>>>>>>>>>>>>>>
0014            157  >PRMSTS0       EQU      #14          LOCATION OF FIRST BYTE WHEN SENDING PRIMARY STATUS TO PPU
0015            158  >PRMSTS1       EQU      #15          PRIMARY STATUS BYTE #1 IS HERE
0016            159  >PRMSTS2       EQU      #16          PRIMARY STATUS BYTE #2 IS HERE
0017            160  >PRMSTS3       EQU      #17          PRIMARY STATUS BYTE #3 IS HERE
0018            161  >PRMSTS4       EQU      #18          PRIMARY STATUS BYTE #4 IS HERE
0019            162  >PRMSTS5       EQU      #19          LOCATION OF Block Check Character WHEN SENDING PRIMARY STATUS
                163  >>>>>>>>>>>>>>>>>>>>
001A            164  >SECSTS0       EQU      #1A          LOCATION OF FIRST BYTE WHEN SENDING SECONDARY STATUS TO PPU
001B            165  >SECSTS1       EQU      #1B          SECONDARY STATUS - THIS BYTE IS NOT USED AND = #00
001C            166  >SECSTS2       EQU      #1C          SECONDARY STATUS - THIS BYTE IS NOT USED AND = #00
001D            167  >SECSTS3       EQU      #1D          SECONDARY STATUS - MS BYTE OF BYTE COUNT RED OFF TAPE
001E            168  >SECSTS4       EQU      #1E          SECONDARY STATUS - LS BYTE OF BYTE COUNT RED OFF TAPE
001F            169  >SECSTS5       EQU      #1F          LOCATION OF Block Check Character WHEN SENDING SECONDARY STATUS
                170  >>>>>>>>>>>>>>>>>>>>
0020            171  >TRDSTS0       EQU      #20          LOCATION OF FIRST BYTE WHEN SENDING TERTIARY STATUS TO PPU
0021            172  >TRDSTS1       EQU      #21          TERTIARY STATUS BYTE #1
0022            173  >TRDSTS2       EQU      #22          TERTIARY STATUS BYTE #2
0023            174  >TRDSTS3       EQU      #23          TERTIARY STATUS BYTE #3
0024            175  >TRDSTS4       EQU      #24          TERTIARY STATUS BYTE #4
0025            176  >TRDSTS5       EQU      #25          LOCATION OF Block Check Character WHEN SENDING TERTIARY STATUS
                177  >>>>>>>>>>>>>>>>>>>>
0026            178  >QNTSTS0       EQU      #26          LOCATION OF FIRST BYTE WHEN SENDING QUINTARY STATUS TO PPU
0027            179  >QNTSTS1       EQU      #27          QUINTARY STATUS BYTE #1
0028            180  >QNTSTS2       EQU      #28          QUINTARY STATUS BYTE #2
0029            181  >QNTSTS3       EQU      #29          QUINTARY STATUS BYTE #3
002A            182  >QNTSTS4       EQU      #2A          QUINTARY STATUS BYTE #4
002B            183  >QNTSTS5       EQU      #2B          LOCATION OF Block Check Character WHEN SENDING QUINTARY STATUS
                184  >>>>>>>>>>>>>>>>>>>>
```

```
                187  ********************************************************************************
                188  * MORE RAM LOCATIONS DEFINED                                                   *
                189  ********************************************************************************
                190
                191  >>>>>>>>>>>>>>>>>>>>
     002C       192  >DRIVE         EQU     #2C          "000000dd" SELECTED DRIVE IS STORED HERE
     002D       193  >DRIVE.OP      EQU     #2D          "000000dd + #5" POINTER TO OP.SYSTEM SELECTED DRIVE'S STATUS
     002E       194  >DRIVE.DS      EQU     #2E          "000000dd + #A" POINTER TO FIND THE SELECTED DRIVE'S DE-SELECT COMMAND
     002F       195  >DESELECT      EQU     #2F          "0tttD0dd" COMMAND TO DE-SELECT A DRIVE IS STORED HERE (D=DENSITY)
     0030       196  >SELECT        EQU     #30          "0tttD1dd" COMMAND TO SELECT A DRIVE IS STORE HERE (D=DENSITY)
     0031       197  >DRV.TRK       EQU     #31          "000tttdd" SELECTED DRIVE AND TRACK IS STORED HERE
                198  >>>>>>>>>>>>>>>>>>>>
     0032       199  >COMMAND       EQU     #32          LOCATION OF FIRST BYTE RECEIVED FROM PPU - STORE OR LOAD
     0033       200  >SEUDO.ADDR    EQU     #33          SECOND BYTE RECEIVED : "PSEUDO-ADDRESS"
     0034       201  >OPERAND1      EQU     #34          MOST SIGNIFICANT BYTE OF 32 BIT OPERAND
     0035       202  >OPERAND2      EQU     #35          NEXT BYTE
     0036       203  >OPERAND3      EQU     #36          NEXT BYTE
     0037       204  >OPERAND4      EQU     #37          LEAST SIGNIFICANT BYTE
     0038       205  >STURE7        EQU     #38          DON'T CARE BYTE OF A STORE COMMAND
     0039       206  >STORE8        EQU     #39          LOCATION OF BCC DURING A STORE COMMAND
                207  >>>>>>>>>>>>>>>>>>>>
     003A       208  >SPARE         EQU     #3A          THIS POINTS TO A SPARE BUFFER IN RAM 22 BYTES LONG
                209  >>>>>>>>>>>>>>>>>>>>
     0050       210  >HOLDOUT       EQU     #50          USED TO PASS BYTES TO BE SENT TO THE PPU TO THE OUTPUT ROUTINES
     0051       211  >SWAPPER       EQU     #51          HOLDER IN RAM SO MFM LS & MS BYTES CAN BE SWAPPED
                212  >>>>>>>>>>>>>>>>>>>>
     0052       213  >LSWORDCNT     EQU     #52          BYTE COUNT ROTATED AROUND RIGHT ONCE SO IT ALMOST LOOKS
     0053       214  >MSWORDCNT     EQU     #53          LIKE BYTE COUNT/2 FOR MFM WORD COUNT.
                215  >>>>>>>>>>>>>>>>>>>>
     0054       216  >MSTHIRDIN.    EQU     #54          THIS HOLDS MS COUNT OF 1/3 INCH OF TAPE MOTION
     0055       217  >LSTHIRDIN.    EQU     #55          LS COUNT OF 1/3 INCH OF TAPE FOR GAPFINDER ROUTINE
                218  >>>>>>>>>>>>>>>>>>>>
     0056       219  >MSBCREV       EQU     #56          THIS HOLDS THE MS BYTE COUNT WITH THE BITS REVERSED
     0057       220  >LSBCREV       EQU     #57          LS BYTE COUNT REVERSED FOR WRITING GCR REVERSED BYTE COUNT
                221  >>>>>>>>>>>>>>>>>>>>
     0058       222  >BLK.NUMBER    EQU     #58          GCR BLOCKETTE NUMBER IS RED AND STORED HERE FOR CHECK LATER
                223  >>>>>>>>>>>>>>>>>>>>
     0059       224  >DONECOUNTW    EQU     #59          THESE TWO ARE USED TO HOLD THE "LS BYTE COUNT - 1" FOR THE
     005A       225  >DONECOUNTR    EQU     #5A          READ AND WRITE ROUTINES TO MAKE THEIR CALCULATING EASIER
                226  >>>>>>>>>>>>>>>>>>>>
                227
                228   * TWO OTHER RAM LOCATIONS ARE USED :
                229   *   ADDRESS #FF IS WHERE A FLAG IS STORED TO DIFFERENTIATE BETWEEN WRITE AND WRITE REPEAT.
                230   *   ADDRESS #F0 IS WHERE A FLAG IS STORED TO INDICATE TO THE READ ROUTINES THAT THIS OPERATION IS AN
                231   *           ERASE, AND THE READ ROUTINE'S PART OF THE OPERATION IS DONE WHEN THE LOAD POINT IS PASSED.
                232   *   BOTH ADDRESSES ARE ADDRESSED EXPLICITLY WHEN USED.
                233
                234   * NOTE THAT THERE ARE 4 "PAGES" OF RAM . YOU CAN ACCESS ANY LOCATION IN RAM
                235   * BY LOADING THE PAGE NUMBER (0-3) INTO "SADRH", THE ADDRESS (0-FF) INTO "SADRL", AND
                236   * USING "SCR" TO ACCESS THE RAM. WHEN YOU ACCESS THE RAM USING THE MNEMONIC "RAM",
                237   * YOU GET WHATEVER ADDRESS YOU SPECIFIED IN THE RAM ADDRESS/LITERAL FIELD,
                238   * BUT IT'S ALWAYS ON PAGE 3.
```

```
241  ********************************************************************************
242  * IN THIS CODE WHEN I POINT INTO RAM AT DRSTS0-3, OR WHEN I PUT THE REGISTER    *
243  * "DRSTS" ON THE D-BUS, I AM READING THE STATUS OF A DRIVE. BELOW ARE THE BIT   *
244  * ASSIGNMENTS OF THIS STATUS.                                                   *
245  ********************************************************************************
246
247   * DRIVE STATUS BITS :
248   *
249   *   #7 : SELECTED (+) ............... HIGH IF THE DRIVE IS SELECTED
250   *   #6 : TAPE READY (+) ............. HIGH IF WE KNOW WHERE WE ARE ON THIS TAPE
251   *   #5 : DRIVE READY (+) ............ HIGH IF THE DRIVE WORKS AND HAS A TAPE INSTALLED
252   *   #4 : END-OF-TAPE WARNING (-) .... LOW IF THE END OF THE TAPE IS NEAR
253   *   #3 : END-OF-TAPE (-) ............ LOW IF AT OR BACKING AWAY FROM THE END OF THE TAPE
254   *   #2 : BEGINNING-OF-TAPE WARN (-) . LOW IF NEAR THE BEGINNING OF THE TAPE
255   *   #1 : BEGINNING-OF-TAPE (-) ...... LOW IF AT OR ADVANCING FROM THE BEGINNING OF TAPE
256   *   #0 : WRITE PROTECTED (-) ........ LOW IF INSTALLED TAPE CARTRIDGE IS NOT TO BE WRITTEN ONTO
257   *
258   *
259   * NOTE THAT IF TAPE READY IS NOT HIGH, WE DO NOT KNOW WHERE WE ARE ON THE TAPE
260   *  (IE. THE TAPE WAS JUST INSTALLED) AND THE TAPE STATUS BITS
261   *  BOTW, BOT, EOTW, & EOT WILL BE INACTIVE HIGH.
262
263
264
265   * THE OPERATING SYSTEM WANTS THE DRIVE STATUS TO LOOK DIFFERENTLY.
266   * TOWARD THIS GOAL, LATER YOU WILL FIND A SUBROUTINE LABELED "SCRAMBLE"
267   * WHOSE FUNCTION IT IS TO MAKE THE DRIVE STATUS LOOK LIKE THIS :
268   *
269   * DRIVE.OP STATUS BITS :
270   *
271   *   #7 : EOT (+)
272   *   #6 : EOTW (+)
273   *   #5 : TAPE INSTALLED (+)
274   *   #4 : BOTW (+)
275   *   #3 : WRITE PROTECTED (+)
276   *   #2 : TAPE POSITION KNOWN (-)
277   *   #1 : DRIVE SELECTABLE (+)
278   *   #0 : BOT (+)
279   *
280   *
281   * AGAIN, IF THE TAPE POSITION IS NOT KNOWN, THE FOLLOWING BITS :
282   *  BOT, BOTW, EOT, AND EOTW  WILL BE INACTIVE (LOW).
```

```
285  ********************************************************************************
286  * ID0 AND ID5 ARE BYTES WHO'S ONLY FUNCTION IS TO SATISFY THE                  *
287  * PPU-CONTROLLER INTERFACE (FIRST BYTE AND BCC).                               *
288  * ID1-4 ARE THE FOUR BYTES OF DEVICE I.D. HERE'S WHAT THEY MEAN                *
289  ********************************************************************************
290
291   *ID 1 :
292   *    #7 - #0 : THIS BYTE IS AN #0A WHICH IS THE I.D. FOR A CARTRIDGE TAPE CONTROLLER
293
294   *ID 2 :
295   *    #7 - #4 : THIS IS THE REV LEVEL OF THE BOARD ON WHICH THIS CODE WILL RUN
296   *              IT DOES NOT MATCH THE REV LEVEL PRINTED ON THE P.C. BOARD, BUT RATHER
297   *              A "#1X" IN ID2 INDICATES THIS IS THE SECOND WORKING REV OF THIS
298   *              BOARD. A "#0X" IN ID2 WAS THE FIRST WORKING BOARD.
299   *    #3 - #0 : THIS IS THE REV LEVEL OF THE CODE WHICH RUNS WITH THIS REV LEVEL
300   *              OF THE P.C. BOARD. "#00" IN ID2 IS THE FIRST CODE IN THE FIRST BOARD.
301   *              "#12" IN ID2 IS THE 3rd CODE REV TO WORK ON THE 2nd P.C. REV.
302   *              THIS IS MUCH LIKE THE NAME OF THE FIRST REV OF THIS MICRO-CODE : CTCB1.
303   *              THE "B" MEANT IT RAN ON THE SECOND REV P.C. BOARD TO WORK, AND
304   *              THE "1" MEANT IT WAS THE FIRST REV MICRO-CODE FOR REV "B" P.C. BOARD.
305
306   *ID 3 ... HERE ARE THE RESULTS OF SELF TEST
307   *    #7 : READ LOGIC CANNOT RESET
308   *    #6 : WDR/WDL SIGNALS NOT RIGHT
309   *    #5 : RTC SIGNAL NOT RIGHT
310   *    #4 : 2901 REGISTER ADDRESS TEST ERROR
311   *    #3 : PUSH/POP ERROR DETECTED
312   *    #2 : RAM TEST FAILED
313   *    #1 : MICRO-PROCESSOR TEST FAILED
314   *    #0 : D-BUS TEST FAILED
315
316   *ID 4 :
317   *    #7 - #0 : THIS BYTE IS NOT YET DEFINED AND = #00
```

```
320  ********************************************************************************
321  * PRMSTS0 AND PRMSTS5 ARE BYTES WHO'S ONLY FUNCTION IS TO SATISFY THE          *
322  * PPU-CONTROLLER INTERFACE (FIRST BYTE AND BCC).                               *
323  * PRMSTS1-4 ARE THE FOUR BYTES OF PRIMARY STATUS. HERE'S WHAT THEY MEAN        *
324  ********************************************************************************
325
326   *PRMSTS 1 :
327   *  #7 : A DRIVE HAS CHANGED STATE SINCE I LAST INTERRUPTED SUCCESSFULLY
328   *  #6 : INTERRUPT ON DRIVE CHANGE STATE ENABLED. CLEARED WHEN AN INTERRUPT SEND IS ATTEMPTED
329   *  #5 : AN ERROR REQUIRING A RE-TRY HAS OCCURED. SEE ERROR BITS AND CODE.
330   *  #4 : NOT USED, = 0. MAY SOMEDAY BE MS BIT OF TRACK NUMBER
331   *  #3 - #0 : THIS TELLS WHICH DRIVE AND TRACK ARE SELECTED
332   *            xxx0ttdd : THERE ARE 4 DRIVES AND 4 TRACKS NUMBERED 0-3
333
334   *PRMSTS 2 :
335   *  #7 : THE LAST STORE COMMAND EXECUTED RED A FILE MARK
336   *  #6 & #5 : THIS IS A TWO BIT MOTION CODE TO INDICATE WHAT THE LAST STORE COMMAND DID.
337   *            x00xxxxx : TAPE NOT MOVED OR STILL IN SAME I.R.G.
338   *            x01xxxxx : MOVED FORWARD OUT OF I.R.G. OR FAST FORWARDED TO END
339   *            x10xxxxx : MOVED BACKWARD OUT OF I.R.G. OF REWOUND TO BEGINNING
340   *            x11xxxxx : TAPE POSTION CANNOT RELIABLY BE KNOWN OR RECOVERED
341   *  #4 - #0 : A 5 BIT ERROR CODE. SEE THE EXTERNAL SPECIFICATION OR LOOK NEAR THE END
342   *            OF THE READ ROUTINES FOR DETAILS.
343
344   *PRMSTS 3 :
345   *  #7 : PARITY ERROR DETECTED ON DATA FROM PPU DURING A WRITE
346   *  #6 : OVERFLOW ERROR - ALL THE DATA DID NOT GET SENT TO THE PPU SUCCESSFULLY
347   *  #5 : UNDERFLOW ERROR - PPU DID NOT SEND DATA TO BE WRITTEN FAST ENOUGH
348   *  #4 : WDL ERROR - WRITE LOGIC GOOFED UP. BETTER RETRY.
349   *  #3 : RDL ERROR - READ LOGIC GOOFED UP. BETTER RETRY.
350   *  #2 : THIS OPERATION WAS A GCR READ WITH THE BYTE COUNT SUPPLIED BY THE OPERATING SYSTEM
351   *  #1 : BYTE COUNT IN SECONDARY STATUS IS VALID FOR RECORD JUST RED OR SKIPPED OVER
352   *  #0 : HIGH IF GCR DENSITY, LOW IF MFM DENSITY IS SELECTED
353
354   *PRMSTS 4 :
355   *  #7 - #0 : THIS IS THE LATEST STATUS OF THE SELECTED DRIVE AS DESCRIBED IN THE
356   *            DRIVE STATUS BIT MEANINGS FOR THE OPERATING SYSTEM.
```

```
359  ********************************************************************************
360  *    DESCRIPTIONS OF THE SECONDARY STATUS AND THE TERTIARY STATUS'ES            *
361  ********************************************************************************
362
363
364   * THE SECONDARY STATUS IS THE BYTE COUNT RED WHEN BIT #1 IN PRMSTS3 SAYS IT IS VALID
365   * WHICH WILL ALWAYS BE TRUE IF NO ERROR IS FLAGGED IN PRMSTS1 BIT #5 AFTER PASSING
366   * FORWARDS OVER A DATA RECORD, AS WELL AS MOST OTHER TIMES A DATA RECORD IS PASSED
367   * FORWARDS. IN GCR MODE ONLY, THE SAME HAPPENS ON REVERSE PASSES. IN THE CASE THAT
368   * THE LAST OPERATION WAS A READ GCR WITH THE BYTE COUNT SUPPLIED BY THE OPERATING
369   * SYSTEM (THIS OPERATION IS NORMALLY REFERED TO IN THIS CODE AS A CRAMMED BYTE COUNT),
370   * THEN PRMSTS2 BITS 0, 1 AND 3 WILL ALL BE SET, INDICATING THIS WAS A CRAMMED BYTE COUNT
371   * READ AND THE DENSITY IS GCR AND THE BYTE COUNT IN THE SECONDARY STATUS IS VALID. IN FACT,
372   * THOUGH, THE BYTE COUNT IN THE SECONDARY STATUS WILL BE THE SAME ONE AS THE OPERATING
373   * SYSTEM SUPPLIED, NOT ONE THAT WAS RED OFF THE TAPE.
374
375
376   * THE TERTIARY STATUS HAS INFORMATION TO CORRECT ERRORS AFTER READING A GCR RECORD.
377   * RECOVERABLE ERRORS ARE ONES THAT INCLUDE NO MORE THAN 1 EVEN NUMBERED AND 1 ODD NUMBERED
378   * BLOCKETTE CONTAINING ERRORS. IN THE EVENT OF A RECOVERABLE ERROR FOLLOWING A NORMAL
379   * GCR READ OR A READ WITH THE BYTE COUNT SUPPLIED, THE TERTIARY STATUS WILL TELL WHICH
380   * BLOCKETTE NUMBERS CONTAINED THE ERRORS AND WHETHER THAT BLOCKETTE WAS A DATA BLOCKETTE
381   * OR AN XOR BLOCKETTE. IN THE EVENT OF ONLY XOR BLOCKETTE ERRORS, THE DATA RECOVERED WAS REALLY
382   * ALL OKAY, SO WHILE A RECOVERABLE ERROR WILL BE FLAGGED IN PRMSTS2, THE ERROR BIT IN
383   * PRMSTS1 WILL NOT BE SET. WHEN RECOVERABLE ERROR #15 IS FLAGGED IN PRMSTS2,
384   * HERE'S HOW TO READ THE TERTIARY STATUS :
385
386   * #AB CD EF GH :
387   *  || || || ||
388   *  || || || ||--THE BLOCKETTE NUMBER (00-FF) OF THE FIRST EVEN BLOCKETTE ERROR
389   *  || || ||
390   *  || || ||-----0 = NO EVEN BLOCKETTE ERROR
391   *  || || |      1 = THIS EVEN BLOCKETTE ERROR IS IN A DATA BLOCKETTE
392   *  || || |      2 = THIS EVEN BLOCKETTE ERROR IS IN AN XOR BLOCKETTE
393   *  || || |
394   *  || || |------THE ERROR CODE (1-9) FOR THE NATURE OF THIS EVEN BLOCKETTE ERROR
395   *  || ||
396   *  || ||--------THE BLOCKETTE NUMBER (00-FF) OF THE FIRST ODD BLOCKETTE ERROR
397   *  ||
398   *  ||----------0 = NO ODD BLOCKETTE ERROR
399   *  |           1 = THIS ODD BLOCKETTE ERROR IS IN A DATA BLOCKETTE
400   *  |           2 = THIS ODD BLOCKETTE ERROR IS IN AN XOR BLOCKETTE
401   *  |
402   *  |-----------THE ERROR CODE (1-9) FOR THE NATURE OF THIS ODD BLOCKETTE ERROR
403   *
404   *             SEE THE GCR READ ROUTINES FOR A DESCRIPTION OF THESE ERROR CODES
405   *             SEE ALSO THE EXTERNAL SPECIFICATION FOR THE CTC.
```

```
408  ********************************************************************************
409  *    DESCRIPTION OF THE QUINTARY (5th) STATUS AND WHAT BLOCKETTES ARE          *
410  ********************************************************************************
411
412   * THE QUINTARY STATUS IS USED IN GCR MODE ONLY. IT RECORDS THE BLOCKETTE ERROR WHICH
413   * IS EITHER THE SECOND ODD BLOCKETTE ERROR OR THE SECOND EVEN BLOCKETTE ERROR.
414   * THIS IS THE ERROR WHICH ABORTS THE READ OPERATION DURING A GCR READ, SKIP FORWARD
415   * OR "CRAMMED BYTE COUNT READ". THE FORMAT IS IDENTICAL TO THE TERTIARY STATUS.
416   * DURING A WRITE DATA OPERATION, THE QUINTARY STATUS HOLDS THE FIRST BLOCKETTE
417   * WHICH WAS IN ERROR SINCE NO ERRORS ARE TOLERATED DURING A WRITE OPERATION.
418   * THIS STATUS IS OF NO PRACTICAL VALUE SINCE IT ONLY CONTAINS INFORMATION DURING
419   * AN UNRECOVERABEL READ OR WRITE ERROR (ERROR #16). IT IS ONLY HERE TO HELP IN
420   * DEBUGGING THE CODE AND TO FIGURE OUT WHAT WENT WRONG WITH AN UNRECOVERABLE TAPE.
421   * IF ERROR #16 IS FLAGGED AND THERE IS NO INFORMATION IN THE QUINTARY STATUS, THEN
422   * WHAT HAS HAPPENED IS A RECOVERABLE READ AND AN OVERFLOW. THIS MEANS THE TAPE
423   * WAS RECOVERABLE, BUT THE SYSTEM DOESN'T HAVE ALL THE DATA TO DO IT SO IT BECOMES
424   * AN UNRECOVERABLE ERROR.
425
426
427
428
429
430
431   * YOU ASK : WHAT IS A BLOCKETTE ? WHAT IS A BLOCKETTE NUMBER ?
432   *
433   * WELL, THE DATA THAT IS WRITTEN ONTO THE TAPE IS DIVIDED INTO GROUPS OF 256 BYTES, CALLED
434   * BLOCKETTES. THE 3RD TO LAST BLOCKETTE OF A RECORD CONTAINS FROM 1 TO 256 BYTES OF DATA
435   * AND IS DATA BLOCKETTE #00. THE 2ND TO LAST BLOCKETTE IS XOR BLOCKETTE NUMBER #01. THE LAST
436   * BLOCKETTE IS XOR BLOCKETTE #00. A ZERO LENGTH RECORD HAS NO BLOCKETTES. ANY OTHER RECORD
437   * HAS AT LEAST 3 BLOCKETTES, AND THE DATA BLOCKETTES COUNT DOWN TO FROM A MAXIMUM NUMBER OF
438   * #FF TO THE 3RD TO LAST WHICH IS BLOCKETTE NUMBER #00. THE DATA IN THE EVEN BLOCKETTES IS ALL
439   * XOR'ED TOGETHER AND WRITTEN INTO THE EVEN NUMBERD XOR BLOCKETTE SUCH THAT IF ALL THE DATA IS
440   * ZEROES, THE XOR BLOCKETTE IS ZEROES. LIKEWISE FOR THE ODD BLOCKETTES. WHEN YOU REQUEST THE
441   * XOR BLOCKETTES BE SENT TO THE SYSTEM, 512 BYTES WILL BE SENT AND THE FIRST 256 BYTES WILL BE
442   * THE ODD (#01) XOR BLOCKETTE.
443   * NOTE THAT ALL DATA BLOCKETTES EXCEPT #00 WILL CONTAIN 256 BYTES. THAT IS, DATA BLOCKETTE #00
444   * WILL CONTAIN 256 BYTES, BUT UP TO 255 OF THEM WILL BE ZERO FILLER BYTES RATHER THAN DATA.
445   * THE BLOCKETTE NUMBER ON THE FIRST DATA BLOCKETTE WILL BE #(AB-1) IF THE BYTE COUNT IS #AB00,
446   * AND WILL BE #AB IF THE BYTE COUNT IS #ABCD WHERE CD IS NOT EQUAL TO 00.
```

```
449  ********************************************************************************
450  * HERE IS A DESCRIPTION OF THE REGISTER USE IN THE SECTION OF THIS CODE WHICH    *
451  * DOES NOT "TALK" TO THE TAPE BUT RATHER TO THE PPU.                             *
452  ********************************************************************************
453
454   * Q  THIS ONE IS FOR SHORT TERM USE BY ANYONE. BE CAREFUL IF IT'S USED OTHERWISE
455   * R3 IS ALMOST ALWAYS LOADED SIMULTANEOUSLY WITH SADRL TO KEEP A RECORD OF OUR RAM POINTER
456
457   * R0 IS HOW MANY BYTES TO BE GOTTEN IN "GETSTRING" (FROM THE PPU)
458   * R1 IS WHICH BITS TO SET/CLEAR IN THE SETBIT AND CLEARBIT ROUTINES
459   * R1 IS USED TO HELP DECODE THE LOAD AND STORE COMMANDS (IT'S SHIFTED RIGHT IN "SHIFT")
460   * R2 IS A SCRATCH REGISTER WHILE SELECTING A DRIVE, TRACK, OR DENSITY
461
462   * R1 COUNTER OF HOW MANY BYTES TO BE SENT          -
463   * R2 RETRY COUNT                                   1 - IN SENDSTRING ROUTINE
464   * R4 PERMANENT RECORD OF HOW MANY BYTES TO BE SENT -
465
466   * R0 STORAGE OF THE DRIVE STATUS WHILE REVIEWING ALL THE DRIVES' STATUS'ES
467   * R4 HOLDS #05 TO SPEED SWITCHING FROM RAM'S "DRSTS" TO "OPDRSTS" AND BACK IN CHEKEM.ALL
468   * R8 TEMPORARY STORAGE OF WHICH DRIVE IS SELECTED WHILE REVIEWING THEM ALL
469   * RE MS COUNTER FOR HOW OFTEN TO CHECK THE DRIVES WHILE THE CTC IS IDLE
470   * RF LS COUNTER FOR HOW OFTEN TO CHECK THE DRIVES WHILE THE CTC IS IDLE
471
472   * RE MS COUNTER IN WAIT.XX.MILLI-SECONDS ROUTINES    (OR MICRO-SECONDS)
473   * RF LS COUNTER IN WAIT.XX.MILLI-SECONDS ROUTINES    (OR MICRO-SECONDS)
474
475   * R8 MASK WETHER TO LOOK FOR EOT, BOT OR TIMEOUT  -
476   * RC MS COUNTER OF HOW MANY INCHES TO COUNT       1 - USED BY FAST FORWARD,
477   * RD LS COUNTER OF HOW MANY INCHES TO COUNT       1   REWIND, CYCLE, AND
478   * RE MS COUNTER TO TIME ONE INCH OF TAPE MOTION   1   WRITEGAP ROUTINES
479   * RF LS COUNTER TO TIME ONE INCH OF TAPE MOTION   -
480
481   * R1 THE BITS TO BE REVERSED IN THE REVERSE BITS ROUTINE
482   * R2 THE BITS ONCE THEY HAVE BEEN REVERSED
483   * R3 A COUNTER IN THE REVERSE BITS ROUTINE
484
485   * R0 THE LATEST SELECTED DRIVE'S STATUS DURING EXECUTION OF THE STORE COMMANDS
486   * R5 INSTRUCTIONS TO / STATUS FROM THE READ AND WRITE ROUTINES
487   * R6 INSTRUCTIONS TO / STATUS FROM THE READ AND WRITE ROUTINES
488   * R7 STATUS FROM THE READ AND WRITE ROUTINE. THE PRIAMRY STATUS UPDATE AT THE END OF EACH
489   *    STORE COMMAND IS CONSTRUCTED FROM THESE REGISTERS : R5, R6, & R7
490   * R5 IS TEMPORARILY A SCRATCH REGISTER WHILE INITIALIZING PRIOR TO EXECUTING A STORE COMMAND
491   * R6 IS TEMPORARILY A SCRATCH REGISTER WHILE INITIALIZING PRIOR TO EXECUTING A STORE COMMAND
492   * RE IS A SCRATCH REGISTER TO DIVIDE BYTE COUNT BY 2 FOR MFM PRIOR TO A WRITE
493   * RF IS ALSO A SCRATCH REGISTER TO MAKE THE MFM WORD COUNT PRIOR TO A WRITE
494
495   * RF IS A COUNTER TO TIME THE PPU DATA PATH IN THE SEND.BYTES AND TOSS.BYTES ROUTINES
496
497   * THERE ARE MANY REGISTERS USED IN THE ROUTINES "CRAM.BC" WHICH ARE NOT LIKE THESE DESCRIPTIONS
498   * AT ALL. "CRAM.BC" HAS TO LOAD UP THE REGISTERS THAT THE READ ROUTINES NORMALLY LOAD UP AND YOU
499   * WILL FIND A DESCRIPTION OF THE REGISTER USE WITHIN "CRAM.BC" AT THE BEGINNING OF THE
500   * READ/WRITE ROUTINES.
```

```
                 503  ********************************************************************************
                 504  * FIRST LET'S INITIALIZE EVERYTHING THAT COULD POSSIBLY NEED IT.                *
                 505  ********************************************************************************
                 506
     0000        507                ORG      #000
                 508
                 509   START        NOP                                          A NOP TO GET THING STARTED
000 0000000620012 510                NEXT
                 511
                 512  >>>>>>>>>>>>>>>>>>>>
                 513  >             A.SB1.B  R7     RAM          ID3            -
001 04405CE5150A1 514  >             CALLC    SELF.TEST                         1
                 515  >             PASS.S   LIT            Q    #0A            1
002 0280028070032 516  >             NEXT                                        1
                 517  >             PASS.Q           RAM         ID1            1
003 03C05C0420042 518  >             NEXT                                        1
                 519  >             PASS.S   LIT            Q    #20            1 - INITIALIZE WHO-ARE-YOU RESPONSE
004 0800028070052 520  >             NEXT                                        1
                 521  >             PASS.Q           RAM         ID2            1
005 04005C0420062 522  >             NEXT                                        1
                 523  >             ZERO             RAM         ID4            1
006 04805C0620072 524  >             NEXT                                        -
                 525  >>>>>>>>>>>>>>>>>>>>
                 526  >                      CIN                                CLEAR COMMAND-STATUS FLAG-OUT
007 0000022620082 527  >             NEXT
                 528  >                      DIN                                CLEAR DATA FLAG-OUT
008 0000020620092 529  >             NEXT
                 530  >                      CLCCO                              CLEAR COMMAND-STATUS COMMAND-OUT
009 00005406200A2 531  >             NEXT
                 532  >                      CLDCO                              CLEAR DATA COMMAND-OUT
00A 0000500620082 533  >             NEXT
                 534  >>>>>>>>>>>>>>>>>>>>
                 535  >                      LIT      DRCON       #00            PUT "STOP" COMMAND INTO DRIVE CONTROL REGISTER
00B 00006686200C2 536  >             NEXT
                 537  >                      LIT      WRCON       #06            PUT "A.C. ERASE" INTO WRITE CONTROL REGISTER
00C 0180728620002 538  >             NEXT
                 539  >                      LIT      RDCON       #00            PUT "TURN OFF" INTO THE READ CONTROL REGISTER
00D 00006E86200E2 540  >             NEXT
                 541  >>>>>>>>>>>>>>>>>>>>
                 542  >             ZERO             RAM         PRMSTS2        -
00E 05805C06200F2 543  >             NEXT                                        1
                 544  >             ZERO             RAM         PRMSTS3        1
00F 05C05C0620102 545  >             NEXT                                        1 - INITIALIZE PRIMARY STATUS
                 546  >             PASS.S   LIT            Q    #80            1
010 2000028070112 547  >             NEXT                                        1
                 548  >             PASS.Q           RAM         PRMSTS1        1
011 05405C0420122 549  >             NEXT                                        -
                 550  >>>>>>>>>>>>>>>>>>>>
```

```
                    553  ************************************************************************************
                    554  * HERE I INITIALIZE ALL THE WAYS I SELECT AND DESELECT THE SELECTED DRIVE,       *
                    555  * TRACK, AND RECORDING DENSITY AS WELL AS INITIALIZING ALL THE POINTERS          *
                    556  * TO THIS INFORMATION IN RAM. INITIALLY, I SELECT DRIVE #0, TRACK #0, MFM.       *
                    557  ************************************************************************************
                    558
                    559                 PASS.S   LIT          Q     #00
012 0000028070132   560                 NEXT
                    561                 PASS.Q         RAM          DRIVE
013 0B005C0420142   562                 NEXT
                    563
                    564                 PASS.S   LIT          Q     #05
014 0140028070152   565                 NEXT
                    566                 PASS.Q         RAM          DRIVE.OP
015 0B405C0420162   567                 NEXT
                    568
                    569                 PASS.S   LIT          Q     #0A
016 0280028070172   570                 NEXT
                    571                 PASS.Q         RAM          DRIVE.DS
017 0B805C0420182   572                 NEXT
                    573
                    574                 PASS.S   LIT          Q     #00
018 0000028070192   575                 NEXT
                    576                 PASS.Q         RAM          DESELECT0
019 02805C04201A2   577                 NEXT
                    578
                    579                 PASS.S   LIT          Q     #01
01A 0040028070182   580                 NEXT
                    581                 PASS.Q         RAM          DESELECT1
01B 02C05C04201C2   582                 NEXT
                    583
                    584                 PASS.S   LIT          Q     #02
01C 0080028070102   585                 NEXT
                    586                 PASS.Q         RAM          DESELECT2
01D 03005C04201E2   587                 NEXT
                    588
                    589                 PASS.S   LIT          Q     #03
01E 00C0028070102   590                 NEXT
                    591                 PASS.Q         RAM          DESELECT3
01F 03405C0420202   592                 NEXT
                    593
                    594                 PASS.S   LIT          Q     #00
020 0000028070212   595                 NEXT
                    596                 PASS.Q         RAM          DESELECT
021 0BC05C0420222   597                 NEXT
                    598
                    599                 PASS.S   LIT          Q     #04
022 0100028070232   600                 NEXT
                    601                 PASS.Q         RAM          SELECT
023 0C005C0420242   602                 NEXT
                    603
                    604                 PASS.S   LIT          Q     #00
024 0000028070252   605                 NEXT
                    606                 PASS.Q         RAM          DRV.TRK
025 0C405C0420262   607                 JMPU     CHEKDRIVES                 GO READ THE INITIAL STATE OF THE DRIVES
```

```
                    610  ********************************************************************************
                    611  * WHEN THE IDLE LOOP DECIDES IT'S TIME, AND AFTER THE COMPLETION OF ALL        *
                    612  * STORE AND LOAD COMMANDS, THIS CODE IS USED TO CALL THE ROUTINE TO CHECK      *
                    613  * THE STATE OF ALL FOUR DRIVES. WHEN DONE, THIS CODE DECIDES WHETHER OR NOT    *
                    614  * TO SEND AN INTERRUPT ON DRIVE CHANGE STATE. IF SO, IT SENDS THE INTERRUPT.   *
                    615  * IN GENERAL, WHEN EVER THE CODE HAS BEEN DOING ANYTHING, WHEN IT'S DONE, IT   *
                    616  * WILL REVIEW THE STATE OF THE DRIVES. THIS INCLUDES AFTER SENDING AN INTERRUPT *
                    617  * ON CHANGE OF STATE OF DRIVE.                                                 *
                    618  ********************************************************************************
                    619
                    620   CHEKDRIVES   PASS.S   LIT    RF      B     #80          -
026 2C003E8C70272   621                NEXT                                     1 - INITIALIZE TIMER WHEN TO CHECK DRIVES
                    622                PASS.S   LIT    RE      B     #04        1         (EVERY 50 MILLI-SECONDS)
027 01003A8C71841   623                CALLC    CHEKEM.ALL                      -
                    624
                    625  >>>>>>>>>>>>>>>>>>>>
                    626  >                     RAM                    PRMSTS1   -
028 056F02A620292   627  >             NEXT                    NBIT7           1
                    628  >                     RAM                    PRMSTS1   1 - INTERRUPT ON DRIVE CHANGED STATE ?
029 056E02A620313   629  >             JMPC     IDLELOOP       NBIT6           1     NO - GO TO IDLE LOOP
                    630  >             NOP                                     1
02A 0000000620313   631  >             JMPC     IDLELOOP                        -
                    632  >>>>>>>>>>>>>>>>>>>>
                    633
                    634                PASS.S   LIT    SADRL   B     PRMSTS0    YES - POINT IN RAM TO PRIMARY STATUS
02B 05004E8C702C2   635                NEXT
                    636
                    637                         LIT    SCR           #54        FIRST BYTE = INTERRUPT SYSTEM W/PRIMARY STATUS
02C 1500DE86202D2   638                NEXT
                    639
                    640                PASS.S   LIT    R4      B     #6         6 BYTES TO BE SENT BY SENDSTRING
02D 0180128C71961   641                CALLC    SENDSTRING                      GO TO SEND THE STRING TO THE PPU
                    642
                    643                PASS.S   LIT    R1      B     #80        PREPARE TO CLEAR BIT "DRIVE CHANGED STATE"
02E 2000068C726F1   644                CALLC    CLRBIT.P1                       DO IT IF INTERRUPT WENT SUCCESSFULLY
                    645
                    646                PASS.S   LIT    R1      B     #40        CLRBIT INTERRUPT ON DRIVE CHANGE STATE ENABLE
02F 1000068C726F1   647                CALLC    CLRBIT.P1
                    648
                    649                NOP                                     DONE SENDING THE INTERRUPT ON DRIVE CHANGE STATE
030 0000000620262   650                JMPU     CHEKDRIVES                      SO GO CHECK THE DRIVES AGAIN
```

```
                    653  ********************************************************************************
                    654  * HERE IN THE IDLE LOOP, WE CLEAR DATA FLAG OUT, THUS THROWING AWAY BYTES       *
                    655  * NOT WANTED FROM THE PPU, CHECK TO SEE IF IT'S TIME TO CHECK THE STATE         *
                    656  * OF THE DRIVES, CHECK COMMAND-STATUS COMMAND-IN, AND LAST, JUST FOR DUCKS,     *
                    657  * CHECK POWER FAIL WARNING.                                                     *
                    658  ********************************************************************************
                    659
                    660   IDLELOOP              RAM     DRSEL       DESELECT
031 0BE162A620322   661                 NEXT                        NRTC            IS IT TIME TO CHECK ALL THE IDLE LOOP STUFF ?
                    662
                    663                 NOP
032 0010000620 3A3  664                 JMPC    CHEK.PPU            CFO             CHECK PPU IF IT'S NOT TIME TO CHECK OTHER STUFF
                    665
                    666                         CLRTC                               RTC SAYS IT'S TIME SO CLEAR IT
033 0017026620342   667                 NEXT                        DCI             FIRST WE SEE IF WE CAN CLEAR THE HANDSHAKE
                    668
                    669                 NOP                                         NEXT WE'LL SEE IF IT'S TIME TO CHECK THE DRIVES
034 0000000620363   670                 JMPC    *+2                                 SKIP IF WE CAN'T CLEAR THE HANDSHAKE
                    671
                    672                         DIN                                 CLEAR THE HANDSHAKE
035 0000020620362   673                 NEXT                                        AND GO SEE IF IT'S TIME TO CHECK THE DRIVES
                    674
                    675  >>>>>>>>>>>>>>>>>>>>>
                    676  >              DEC.A   RF      RF          B               FIRST, DECREMENT THE LS TIMER
036 00193DECC0372   677  >              NEXT                        NBORROW         DID IT ROLL OVER ?
                    678  >              NOP
037 0010000620 3A3  679  >              JMPC    CHEK.PPU            CFO             NO : LOOK FOR COMMAND FROM PPU
                    680  >              DEC.A   RE      RE          B               YES : DECREMENT THE MS TIMER
038 003939CCC0392   681  >              NEXT                        BORROW          DID IT ROLL OVER ?
                    682  >              NOP
039 0010000620263   683  >              JMPC    CHEKDRIVES          CFO             YES : CHECK THE STATE OF THE DRIVES
                    684  >>>>>>>>>>>>>>>>>>>>>
                    685
                    686   CHEK.PPU      NOP                                         IT WASN'T TIME TO CHECK THE STATE OF THE DRIVES
03A 0027000620 3D3  687                 JMPC    GET1STBYTE          NPFW            GO GET 1st BYTE IF COMMAND COMMAND-IN IS HERE
                    688
                    689                 NOP
038 0000000620313   690                 JMPC    IDLELOOP                            LOOP ON IDLE LOOP IF NOT POWER FAIL WARNING
                    691
                    692   WAIT.RESET    NOP                                         POWER FAIL WARNING WAS PRESENT
03C 0000000620 3C2  693                 JMPU    *                                   SO WAIT FOR A RESET
```

```
                    696  ********************************************************************************
                    697  * WE HAVE SENSED COMMAND COMMAND-IN SO TAKE THE FIRST BYTE AND DECODE           *
                    698  * IT. IF IT'S A STORE, TAKE 7 MORE BYTES. IF IT'S A LOAD, TAKE 3 MORE           *
                    699  * BYTES. IF IT'S NOT A NACK, "NACK" IT. IF IT IS A NACK, IGNORE IT.             *
                    700  ********************************************************************************
                    701
                    702   GETISTBYTE   PASS.S   LIT     SADRL  B     COMMAND      POINT TO INPUT BUFFER IN RAM
03D 0C934E8C703E2   703                NEXT                    CCI
                    704
                    705                NOP
03E 00130006203E3   706                JMPC     *              CCI               WAIT FOR COMMAND COMMAND-IN TO GO AWAY
                    707
                    708                PASS.S   CIN    SCR    Q                 LOAD 1st BYTE INTO RAM AND Q FOR BCC CHECK
03F 0000DE2070402   709                NEXT                                      WHICH ALSO CLEARS COMMAND FLAG-OUT
                    710
                    711                S.XOR.Q  LIT                STORE        IS IT A STORE ?
040 25B8028760412   712                NEXT                    NZERO
                    713
                    714                S.XOR.Q  LIT                LOAD         IS IT A LOAD ?
041 1888028760433   715                JMPC     *+2            NZERO             SKIP IF NOT A STORE
                    716
                    717                PASS.S   LIT    RO     B     #07         IT IS A STORE : 7 MORE BYTES TO GET
042 01F0028C70472   718                JMPU     GETSTRING      NCFO
                    719
                    720                S.XOR.Q  LIT                NACK         IS IT A NACK ?
043 3E38028760453   721                JMPC     *+2            NZERO             SKIP IF NOT A LOAD
                    722
                    723                PASS.S   LIT    RO     B     #03         IT IS A LOAD : 3 MORE BYTES TO GET
044 00F0028C70472   724                JMPU     GETSTRING      NCFO
                    725
                    726                NOP
045 0000000620503   727                JMPC     SENDANACK                        IF IT'S NOT A NACK, SEND A NACK
                    728
                    729                NOP
046 0000000620262   730                JMPU     CHEKORIVES                       IT'S A NACK SO IGNORE IT
```

```
                    733  ********************************************************************************
                    734  * GETSTRING FETCHES FROM THE PPU THE NUMBER OF BYTES WHICH ARE INDICATED       *
                    735  * IN R0, AND CHECKS THE BCC AND DECIDES WETHER TO "ACK" OR "NACK" THE STRING.  *
                    736  * THE ENTIRE FETCHED STRING BEGINS IN RAM AT LOCATION "COMMAND" AND IS 8 BYTES *
                    737  * LONG IF A STORE, 4 BYTES LONG IF IT WAS A LOAD.                              *
                    738  ********************************************************************************
                    739
                    740   GETSTRING    NOP
047 0030000620473   741                JMPC     *              NCF0            WAIT FOR COMMAND FLAG-OUT
                    742
                    743                INC.A    R3     SADRL   B               INCREMENT THE POINTER INTO RAM
048 00134C6C48492   744                NEXT                    CCI
                    745
                    746                NOP
049 0013000620493   747                JMPC     *              CCI             WAIT FOR COMMAND COMMAND-IN TO GO AWAY
                    748
                    749                S.XOR.Q  CIN    SCR     Q               TAKE THE BYTE AND COMPUTE THE BCC
04A 0000DE23604B2   750                NEXT                                    WHICH ALSO CLEARS COMMAND FLAG-OUT
                    751
                    752                DEC.A    R0     R0      B               DECREMENT THE COUNT OF HOW MANY BYTES TO FETCH
04B 0038000CC04C2   753                NEXT                    NZERO
                    754
                    755                NOP
04C 0030000620473   756                JMPC     GETSTRING      NCF0            GET MORE BYTES IF COUNT NOT EXPIRED
                    757
                    758   GOTSTRING    S.XOR.Q  LIT                    #FF     DOES THE BCC IN Q = #FF ?
04D 3FD80287604E2   759                NEXT                    ZERO
                    760
                    761                NOP
04E 0012000620583   762                JMPC     SENDAACK       CFI             SEND AN ACK IF BCC = #FF
                    763
                    764                NOP
04F 0012000620502   765                JMPU     SENDANACK      CFI             SEND A NACK IF BCC # #FF
```

```
                    768  ********************************************************************************
                    769  * THIS LITTLE ROUTINE JUST SENDS A NACK TO THE PPU, SWALLOWING AS MANY          *
                    770  * BYTES AS IT HAS TO IN ORDER TO GET THAT NACK SENT. IT WILL SEND ONE           *
                    771  * AND ONLY ONE NACK !                                                          *
                    772  ********************************************************************************
                    773
                    774   SENDANACK    NOP                                        WAIT FOR COMMAND FLAG-IN TO GO AWAY
050 0012000620503   775                JMPC     *              CFI                   (JUST IN CASE IT'S THERE)
                    776
                    777                LIT    COUT        NACK          SEND A NACK WHICH ASLO SETS COMMAND COMMAND-OUT
051 3E10468620522   778                NEXT                  CFO
                    779
                    780  >>>>>>>>>>>>>>>>>>>>
                    781  >             NOP                                        -
052 0032000620553   782  >             JMPC     BYTECOMING    NCFI                1 - POLE COMMAND FLAG-IN AND FLAG-OUT TO SEE
                    783  >             NOP                                        1    IF THE NACK HAS TAKEN
053 0010000620523   784  >             JMPC     *-1           CFO                 -
                    785  >>>>>>>>>>>>>>>>>>>>
                    786
                    787   SENDING                    CLCCO                       THE NACK IS ACCEPTED, SO CLEAR COMMAND COMMAND-OUT
054 0000540620262   788                JMPU     CHEKDRIVES                        AND GO CHECK THE DRIVES
                    789
                    790   BYTECOMING   NOP                                        THE PPU IS SENDING ME SOMETHING INSTEAD
055 0013000620562   791                NEXT                  CCI
                    792
                    793                NOP
056 0013000620563   794                JMPC     *              CCI                 WAIT FOR COMMAND COMMAND-IN TO GO AWAY
                    795
                    796                CIN                                       THROW AWAY THE BYTE, CLEARING COMMAND FLAG-OUT
057 0012022620502   797                JMPU     SENDANACK     CFI                 GO TRY TO SEND A NACK AGAIN
                    798
                    799
                    800
                    801   * ABOVE IT IS POSSIBLE TO "NACK" A NACK - BUT ONLY ONCE !
```

```
                    804  ********************************************************************************
                    805  *  THE BCC WAS OKAY SO WE SEND AN ACK TO THE PPU IF IT WILL LET US               *
                    806  ********************************************************************************
                    807
                    808   SENDAACK      NOP                                    WAIT FOR COMMAND FLAG-IN TO GO AWAY
058 0012000620583   809                 JMPC      *             CFI              (JUST IN CASE IT'S THERE)
                    810
                    811                           LIT     COUT        ACK        SEND AN ACK AND SET COMMAND COMMAND-OUT
059 03904686205A2   812                 NEXT                    CFO
                    813
                    814  >>>>>>>>>>>>>>>>>>>>
                    815  >              NOP                                    -
05A 00320006205F3   816  >              JMPC      COULDNT       NCFI           1 - POLE COMMAND FLAG-IN AND COMMAND FLAG-OUT
                    817  >              NOP                                    1     TO SEE IF THE ACK WAS ACCEPTED
05B 0010006205A3    818  >              JMPC      *-1           CFO            -
                    819  >>>>>>>>>>>>>>>>>>>>
                    820
                    821   COULD                   RAM                COMMAND  INSPECT INSTRUCTIONS FROM PPU
05C 0C8F02A6205D2   822                 NEXT                    BIT7           WAS IT A STORE OR A LOAD ?
                    823
                    824                                 CLCCO                  CLEAR COMMAND COMMAND-OUT
05D 0000540620793   825                 JMPC      ITSASTORE                    GO TO PROCESS A STORE COMMAND
                    826
                    827                 NOP
05E 0000000620602   828                 JMPU      ITSALOAD                     GO TO PROCESS A LOAD COMMAND
                    829
                    830   COULDNT       NOP
05F 0000000620262   831                 JMPU      CHEKDRIVES                   GO CHECK THE DRIVES
```

```
                    834  ********************************************************************************
                    835  * HERE WE PROCESS THE LOAD REQUESTS. IF THE REQUEST IS NOT LEGAL OR DEFINED,     *
                    836  * THEN WE IGNORE IT AND RETURN TO THE IDLE LOOP. OTHERWISE, WE SEND THE          *
                    837  * REQUESTED STATUS USING SENDSTRING. THEN WE RETURN TO THE IDLE LOOP.           *
                    838  ********************************************************************************
                    839
                    840  ITSALOAD     PASS.S   LIT          Q     #05          -
060 0140028072681   841               CALLC    SHIFT                            1 - CONDITION BIT WILL BE TRUE IF THE
                    842               A.SB1.Q  R1           B.A                  1     LOAD IS DEFINED AND LEGAL
061 0039002900622   843               NEXT                  BORROW               -
                    844
                    845               NOP
062 0000000620706   846               INDEXC   INDEXLO                           DO AN INDEXED JUMP IF THE LOAD IS LEGAL
                    847
                    848  BADLOAD      NOP                                        THE LOAD WAS NOT LEGAL SO IGNORE IT
063 0000000620262   849               JMPU     CHEKDRIVES                        AND GO CHECK THE DRIVES
                    850
                    851  * THE JUMP TABLE MUST BEGIN AT ADDRESS #XX0
     0070           852               ORG      *+#10 AND #7F0
                    853
                    854
                    855  INDEXLO      PASS.S   LIT    SADRL  B     IDO           REQUEST FOR DEVICE I.D.
070 03804E8C70762   856               JMPU     FINISHLOAD                        POINT TO BUFFER IN RAM, GO FILL FIRST BYTE
                    857
                    858  INDEXL1      PASS.S   LIT    SADRL  B     PRMSTSO       REQUEST FOR PRIMARY STATUS
071 05004E8C70762   859               JMPU     FINISHLOAD                        POINT TO BUFFER IN RAM, GO FILL FIRST BYTE
                    860
                    861  INDEXL2      PASS.S   LIT    SADRL  B     SECSTSO       REQUEST FOR SECONDARY STATUS
072 06804E8C70762   862               JMPU     FINISHLOAD                        POINT TO BUFFER IN RAM, GO FILL FIRST BYTE
                    863
                    864  INDEXL3      PASS.S   LIT    SADRL  B     TRDSTSO       REQUEST FOR TERTIARY STATUS
073 08004E8C70762   865               JMPU     FINISHLOAD                        POINT TO BUFFER IN RAM, GO FILL FIRST BYTE
                    866
                    867  INDEXL4      PASS.S   LIT    SADRL  B     OPDRSTSX      REQUEST FOR THE STATUS OF ALL FOUR DRIVES
074 01004E8C70762   868               JMPU     FINISHLOAD                        POINT TO BUFFER IN RAM, GO FILL FIRST BYTE
                    869
                    870  INDEXL5      PASS.S   LIT    SADRL  B     QNTSTSO       REQUEST FOR QUINTARY STATUS
075 09804E8C70762   871               JMPU     FINISHLOAD                        POINT TO BUFFER IN RAM, GO FILL FIRST BYTE
                    872
                    873
                    874  * THAT'S THE END OF THE JUMP TABLE FOR THE DEFINED LOADS
                    875  * BELOW IS THE REST OF THE JOB TO SEND STATUS
                    876
                    877
                    878  FINISHLOAD             LIT    SCR           #A4          FIRST BYTE = RESPONSE TO STATUS REQUEST
076 2900DE8620772   879               NEXT
                    880
                    881               PASS.S   LIT    R4     B     #06           THERE ARE 6 BYTES TO BE SENT
077 0180128C71961   882               CALLC    SENDSTRING                        SO GO SEND THEM
                    883
                    884               NOP
078 0000000620262   885               JMPU     CHEKDRIVES                        ALL DONE SO GO CHECK THE DRIVES
```

```
                    888  ********************************************************************************
                    889  * THE PPU SENT A STORE COMMAND. IF THIS COMMAND INCLUDES SELECTING A NEW        *
                    890  * DRIVE, DO IT. LOAD THE 1/3 INCH DROPOUT COUNTER IN CASE IT'S NEEDED.          *
                    891  * CLEAR OUT THE REGISTERS WHERE THE STATUS OF THIS STORE WILL BE ENCODED.       *
                    892  * SELECT THE DRIVE, RECORD IT'S STATUS IN R0 AND DECODE THE STORE, FLAGGING     *
                    893  * AN ERROR IF IT IS NOT DEFINED.                                               *
                    894  ********************************************************************************
                    895
                    896   ITSASTORE           RAM                 SEUDO.ADDR   INSPECT BIT "SELECT A NEW DRIVE ?"
079 0CCB02A6207A2   897                NEXT                BIT3
                    898
                    899                ZERO          RAM   B     #FF          ZERO OUT STATUS REGISTER R7 AND WRITE REPEAT FLAG
07A 3FC05C0E20BD1   900                CALLC   SEL.DRV                      GO SELECT THE NEW DRIVE IF TOLD TO
                    901                PASS.S  LIT   R1    B     #20          PREPARE TO CLEAR THE ERROR BIT IN PRMSTS1
07B 0800068C726F1   902                CALLC   CLRBIT.P1                    GO CLAER THE ERROR BIT
                    903                RAM                 PRMSTS3      INSPECT BIT "MFM OR GCR ?"
07C 05E802A6207D2   904                NEXT                NBIT0
                    905                RAM   DRSEL         SELECT       SELECT THE SELECTED DRIVE
07D 0C0062A620803   906                JMPC    MFM.3IN
                    907
                    908  >>>>>>>>>>>>>>>>>>>
                    909  >GCR.3IN      PASS.S  LIT   R5    B     #01          LOAD MS 1/3 INCH GCR GAP SIZE
07E 0040168C707F2   910  >             NEXT
                    911  >             PASS.S  LIT   R6    B     #4A          LOAD LS 1/3 INCH GCR GAP SIZE
07F 12801A8C70822   912  >             JMPU    INTO.RAM
                    913  >MFM.3IN      PASS.S  LIT   R5    B     #01          LOAD MS 1/3 INCH MFM GAP SIZE
080 0040168C70812   914  >             NEXT
                    915  >             PASS.S  LIT   R6    B     #02          LOAD LS 1/3 INCH MFM GAP SIZE
081 00801A8C70822   916  >             JMPU    INTO.RAM
                    917  >INTO.RAM     PASS.A  R5    RAM         MSTHIRDIN.   LOAD THE GAP SIZE COUNTER INTO RAM
082 15005CA440832   918  >             NEXT
                    919  >             PASS.A  R6    RAM         LSTHIRDIN.   LOAD THE GAP SIZE COUNTER INTO RAM
083 15405CC440842   920  >             NEXT
                    921  >>>>>>>>>>>>>>>>>>>
                    922  >             ZERO          RAM         #F0          ZERO OUT THE ERASE FLAG IN RAM'S #F0
084 3C005C0620852   923  >             NEXT
                    924  >             ZERO          R5    B                  CONTINUE TO ZERO OUT THE STATUS
085 0000140E20862   925  >             NEXT
                    926  >             ZERO          R6    B                  CONTINUE TO ZERO OUT THE STATUS
086 0000180E226B1   927  >             CALLC   SHIFT                        GO PREPARE R1 TO DO AN INDEXED DECODE OF THE COMMAND
                    928  >>>>>>>>>>>>>>>>>>>
                    929  >                     RAM                 SEUDO.ADDR   INSPECT BIT "SELECT DRIVE" TO CHECK IF LEGAL DRIVE
087 0CE802A620882   930  >             NEXT                NBIT3
                    931  >             PASS.S  DRSTS R0    B                  LOAD THE SELECTED DRIVE'S STATUS INTO R0
088 0000024C708A3   932  >             JMPC    *+2                          SKIP IF WE DIDN'T SELECT A NEW DRIVE
                    933  >                     RAM                 SEUDO.ADDR   INSPECT BIT "DRIVE NUMBER > #3"
089 0CEA02A6208A2   934  >             NEXT                NBIT2
                    935  >             PASS.A  R1                           PUT INDEXING INFORMATION IN R1 ON D-BUS
08A 00000024408C3   936  >             JMPC    *+2                          SKIP IF DRIVE NUMBER WAS LEGAL
                    937  >>>>>>>>>>>>>>>>>>>
                    938
                    939                NOP                                  THE DRIVE NUMBER TO BE SELECTED WAS ILLEGAL
08B 0000000624E42   940                JMPU    ERROR.1                      SO GO FLAG THE ERROR
                    941
                    942                NOP                                  IT'S OKAY SO FAR SO DO INDEXED STORE COMMAND DECODE
08C 0000000620906   943                INDEXC  INDEXSO
```

```
                    945  ********************************************************************************
                    946  * HERE THE PSEUDO ADDRESS OF THE STORE COMMAND IS DECODED. SOME STORES          *
                    947  * ARE ILLEGAL, SOME ARE DONE QUICKLY AND EFFICIENTLY. OTHERS REQUIRE            *
                    948  * FURTHER DECODING YET                                                          *
                    949  ********************************************************************************
                    950
                    951   * THE INDEX TABLE MUST BEGIN AT ADDRESS #XX0
         0090       952                 ORG      *+#10 AND #7F0
                    953
                    954   INDEXS0               RAM     DRSEL       DESELECT     DO A HARD RESET
090 0BC062A620000   955                 RESETU                                    SO DESELECT DRIVE AND DO THE RESET
                    956
                    957   INDEXS1       NOP                                     NOT DEFINED
091 00000006209C2   958                 JMPU     BADSTORE
                    959
                    960   INDEXS2       NOP                                     NOT DEFINED
092 00000006209C2   961                 JMPU     BADSTORE
                    962
                    963   INDEXS3       PASS.S   LIT            Q     #19       A STORE WITH NO ASSOCIATED DATA
093 0640028070982   964                 JMPU     MORE.DECODE                      SO NOTE THERE ARE #19 SUCH STORES DEFINED
                    965
                    966   INDEXS4       PASS.S   RAM            Q     OPERAND1  READ
094 0D1802A070B92   967                 JMPU     READ.WHAT      ZERO             SO BEGIN DECODE OF READ WHAT ?
                    968
                    969   INDEXS5       S.IOR.A  LIT    RO      Q     #16       WRITE DATA
095 0580028ID1592   970                 JMPU     WRITE.DATA
                    971
                    972   INDEXS6       S.IOR.A  LIT    RO      Q     #16       WRITE A.C ERASE GAP
096 0580028ID0ED2   973                 JMPU     WRITE.GAP
                    974
                    975   INDEXS7       S.IOR.A  LIT    RO      Q     #16       DIAGNOSTIC WRITE DATA REPEAT
097 0580028ID15A2   976                 JMPU     WRITE.RPT
                    977
                    978
                    979
                    980   * HERE IS THE FURTHER DECODE OF THE COMMANDS WITH NO ASSOCIATED DATA
                    981
                    982   MORE.DECODE   S.SUB.Q  RAM                    OPERAND4  IS STORE COMMAND NUMBER 19 OR MORE ?
098 0DD902A568992   983                 NEXT                  NBORROW             IF SO, ITS AN ILLEGAL STORE
                    984
                    985                         RAM                    OPERAND4  PUT THE INDEXING INFORMATION ON D-BUS
099 0DEC02A6209C3   986                 JMPC     BADSTORE     NBIT4               AND DECIDE WHICH TABLE TO INDEX INTO
                    987
                    988                         RAM                    OPERAND4  PUT INDEXING INFORMATION ON D-BUS
09A 0DC002A620A06   989                 INDEXC   00                               INDEX INTO THE 0'th TABLE
                    990
                    991                 NOP
09B 0000000620806   992                 INDEXC   10                               INDEX INTO THE 1'th TABLE
                    993
                    994   BADSTORE      NOP                                     THE STORE COMMAND WAS NOT ONE OF THE DEFINED ONES
09C 000000624E52    995                 JMPU     ERROR.2
                    996
                    997   * AGAIN, THE TABLE ON THE NEXT PAGE MUST BEGIN AT ADDRESS #XX0
         00A0       998                 ORG      *+#10 AND #7F0
```

```
                    1000  ********************************************************************************
                    1001  * MORE DECODE OF STORE COMMANDS                                                 *
                    1002  ********************************************************************************
                    1003
                    1004   00          NOP                                          NOP
0A0 0000000621782   1005               JMPU     UPDATE.STS
                    1006   01          S.IOR.A  LIT    R0     Q    #5F              CYCLE TAPE ONCE. STOP AT BOT
0A1 17C00281D0DB2   1007               JMPU     CYCLE.FOR
                    1008   02          S.IOR.A  LIT    R0     Q    #5F              REWIND TAPE TO BOT
0A2 17C00281D0E52   1009               JMPU     REWIND
                    1010   03          S.IOR.A  LIT    R0     Q    #5F              FAST FORWARD TAPE TO EOT
0A3 17C00281D0DB2   1011               JMPU     FAST.FOR
                    1012   04          S.IOR.A  LIT    R0     Q    #1D              SEARCH FOR A FILE MARK REVERSE
0A4 07400281D1042   1013               JMPU     SRCH.FMREV
                    1014   05          S.IOR.A  LIT    R0     Q    #17              SEARCH FOR A FILE MARK FORWARD
0A5 05C00281D10A2   1015               JMPU     SRCH.FMFOR
                    1016   06          S.IOR.A  LIT    R0     Q    #1D              SKIP OVER A RECORD REVERSE
0A6 07400281D1102   1017               JMPU     SKIP.REV
                    1018   07          S.IOR.A  LIT    R0     Q    #17              SKIP OVER A RECORD FORWARD
0A7 05C00281D1182   1019               JMPU     SKIP.FOR
                    1020   08          NOP                                          DISABLE INTERRUPT ON DRIVE CHANGED STATE
0A8 0000000620D72   1021               JMPU     DIS.INT
                    1022   09          NOP                                          ENABLE INTERRUPT ON DRIVE CHANGED STATE
0A9 0000000620D92   1023               JMPU     EN.INT
                    1024   0A          NOP                                          NOT DEFINED
0AA 00000006209C2   1025               JMPU     BADSTORE
                    1026   0B          NOP                                          NOT DEFINED
0AB 00000006209C2   1027               JMPU     BADSTORE
                    1028   0C          PASS.S   LIT            Q    #00             SELECT MFM RECORDING DENSITY
0AC 0000028070C82   1029               JMPU     SEL.DEN
                    1030   0D          PASS.S   LIT            Q    #08             SELECT GCR RECORDING DENSITY
0AD 0200028070C82   1031               JMPU     SEL.DEN
                    1032   0E          NOP                                          NOT DEFINED
0AE 00000006209C2   1033               JMPU     BADSTORE
                    1034   0F          NOP                                          NOT DEFINED
0AF 00000006209C2   1035               JMPU     BADSTORE
                    1036   10          PASS.S   LIT            Q    #00             SELECT TRACK #00
0B0 0000028070C52   1037               JMPU     SEL.TRK
                    1038   11          PASS.S   LIT            Q    #10             SELECT TRACK #01
0B1 0400028070C52   1039               JMPU     SEL.TRK
                    1040   12          PASS.S   LIT            Q    #20             SELECT TRACK #02
0B2 0800028070C52   1041               JMPU     SEL.TRK
                    1042   13          PASS.S   LIT            Q    #30             SELECT TRACK #03
0B3 0C00028070C52   1043               JMPU     SEL.TRK
                    1044   14          NOP                                          TRACK NUMBER #4 DOESN'T EXIST YET
0B4 0000000624E62   1045               JMPU     ERROR.3
                    1046   15          NOP                                          TRACK NUMBER #5 DOESN'T EXIST YET
0B5 0000000624E62   1047               JMPU     ERROR.3
                    1048   16          NOP                                          TRACK NUMBER #6 DOESN'T EXIST YET
0B6 0000000624E62   1049               JMPU     ERROR.3
                    1050   17          NOP                                          TRACK NUMBER #7 DOESN'T EXIST YET
0B7 0000000624E62   1051               JMPU     ERROR.3
                    1052   18          S.IOR.A  LIT    R0     Q    #16             WRITE A FILE MARK
0B8 0580028ID1202   1053               JMPU     WRITE.FM
```

```
                    1055  ********************************************************************************
                    1056  * AND THE LAST LITTLE BIT OF DECODE OF THE STORE COMMANDS                       *
                    1057  ********************************************************************************
                    1058
                    1059   READ.WHAT     S.XOR.Q  LIT              #40          IS THIS A REQUEST FOR ERROR CORRECTING DATA ?
089 1018028761263   1060                JMPC     READ.TAPE        ZERO         THIS IS A NORMAL READ OFF THE TAPE
                    1061
                    1062                S.XOR.Q  LIT              #80          IS THIS A READ WITH A CRAMMED BYTE COUNT ?
08A 2018028761313   1063                JMPC     SEND.XOR         ZERO         THIS IS A REQUEST FOR ERROR CORRECTING DATA
                    1064
                    1065                         RAM              PRMSTS3      IF CRAMMED BYTE COUNT, IS IT GCR DENSITY ?
08B 05E802A621393   1066                JMPC     CRAM.BC          NBITO        IT'S A CRAMMED BYTE COUNT
                    1067
                    1068                NOP                                    IT WAS NONE OF THE ABOVE SO IT'S AN ERROR
08C 0000000624E52   1069                JMPU     ERROR.2                       IT'S AN ILLEGAL STORE
                    1070
                    1071
                    1072
                    1073  ********************************************************************************
                    1074  * THAT'S THE END OF THE DECODE OF THE STORE COMMANDS. NOW ON TO EXECUTE         *
                    1075  * THE DECODED STORE COMMANDS                                                    *
                    1076  ********************************************************************************
```

```
                1079  ************************************************************************************
                1080  * HERE ARE THE ROUTINES WHICH SELECT DRIVES, TRACKS, AND RECORDING DENSITY.      *
                1081  * THEY ALL CALL SEL.DRTKDN ON THE NEXT PAGE WHICH REALLY DOES THE WORK.          *
                1082  ************************************************************************************
                1083
                1084   SEL.DRV      PASS.S   RAM    R2      B    SEUDO.ADDR   PUT WHICH DRIVE TO SELECT IN R2
08D 0CCA0AAC708E2 1085              NEXT                     BIT2              AND CHECK IF THIS DRIVE NUMBER EXISTS
                1086
                1087               S.AND.A  LIT    R2      B    #03          MASK AWAY UNWANTED BITS
08E 00C00A8E5000A 1088             RETURNC                                   RETURN IF DRIVE NUMBER IS ILLEGAL
                1089
                1090               PASS.A   R2     RAM             DRIVE     STORE SELECTED DRIVE NUMBER IN "DRIVE"
08F 08005C4440C02 1091             NEXT
                1092
                1093               S.ADD.A  LIT    R2      B    #05          ADVANCE DRIVE TO POINT TO OPDRSTS(dd)
0C0 01400A8C50C12 1094             NEXT
                1095
                1096               PASS.A   R2     RAM             DRIVE.OP  LOAD DRIVE.OP
0C1 0B405C4440C22 1097             NEXT
                1098
                1099               S.ADD.A  LIT    R2      B    #05          ADVANCE DRIVE.OP TO POINT TO DRIVE.DS
0C2 01400A8C50C32 1100             NEXT
                1101
                1102               PASS.A   R2     RAM             DRIVE.DS  LOAD DRIVE.DS
0C3 0B805C4440C42 1103             NEXT
                1104
                1105               PASS.A   R2     SADRL   B                 POINT INTO RAM AT THE APROPRIATE "DESELECTX"
0C4 00004C4C40CD2 1106             JMPU     SEL.DRTKDN                       THE RETURN IS IN SEL.DRTKDN
                1107
                1108
                1109
                1110
                1111
                1112   SEL.TRK      PASS.S   RAM    SADRL   B    DRIVE.DS     POINT TO SELECTED DRIVE'S DESELECT COMMAND
0C5 0B804EAC70C62 1113             NEXT
                1114
                1115               PASS.S   SCR    R2      B                 LOAD APPROPRIATE "DESELECTX" INTO R2
0C6 00008AAC70C72 1116             NEXT
                1117
                1118               S.AND.A  LIT    R2      B    #8F          MASK OUT OLD TRACK NUMBER. NEW ONE IS IN Q
0C7 23C00A8E50CB2 1119             JMPU     FINISH.SEL                       GO FINISH THIS TRACK SELECT
                1120
                1121   SEL.DEN      PASS.S   RAM    SADRL   B    DRIVE.DS     POINT TO APPROPRIATE "DESELECTX"
0C8 0B804EAC70C92 1122             NEXT
                1123
                1124               PASS.S   SCR    R2      B                 LOAD APPROPRIATE "DESELECTX" INTO R2
0C9 00008AAC70CA2 1125             NEXT
                1126
                1127               S.AND.A  LIT    R2      B    #F7          MASK AWAY OLD DENSITY. NEW ONE IS IN Q
0CA 3DC00A8E50CB2 1128             JMPU     FINISH.SEL
                1129
                1130   FINISH.SEL   A.IOR.Q  R2     SCR                       LOAD NEW "DESELECTX" INTO RAM
0CB 0000DC4580CD1 1131             CALLC    SEL.DRTKDN                       GO UPDATE REST OF RAM LOCATIONS
                1132
                1133               NOP
0CC 0000000621782 1134             JMPU     UPDATE.STS                       GO UPDATE THE PRIMARY STATUS
```

```
                1136  ********************************************************************************
                1137  * THIS ROUTINE IS ENTERED WITH THE RAM POINTER POINTING TO THE DESELECTX       *
                1138  * COMMAND FOR THE SELECTED DRIVE. IT TAKES THE INFORMATION ABOUT SELECTED      *
                1139  * DRIVE, TRACK AND RECORDING DENSITY STORED IN DESELECTX AND UPDATES ALL       *
                1140  * THE OTHER RECORDS AND WAYS THAT THIS INFORMATION IS STORED.                  *
                1141  ********************************************************************************
                1142
                1143   SEL.DRTKDN   PASS.S   SCR    R2     B                LOAD DESELECT COMMAND INTO R2
0CD 00008AAC70CE2 1144              NEXT
                1145
                1146                PASS.A   R2     RAM          DESELECT    STORE THIS DESELECTX COMMAND IN DESELECT
0CE 0BC05C4440CF2 1147              NEXT
                1148
                1149                S.IOR.A  LIT    R2     B     #04         SET BIT "SELECT"
0CF 01000A8DD0D02 1150              NEXT
                1151
                1152                PASS.A   R2     RAM    Q     SELECT      AND LOAD THE SELECT COMMAND
0D0 0C005C4040D12 1153              NEXT
                1154
                1155                S.AND.A  LIT    R2     BR    #70         ISOLATE TRACK BITS AND SHIFT RIGHT ONCE
0D1 1C000A9650D22 1156              NEXT
                1157
                1158                PASS.A   R2     R2     BR                SHIFT RIGHT ONCE MORE
0D2 000008544OD32 1159              NEXT
                1160
                1161                S.AND.Q  LIT           Q     #03         ISOLATE SELECTED DRIVE BITS
0D3 00C0028260D42 1162              NEXT
                1163
                1164                A.IOR.Q  R2     RAM          DRV.TRK     LOAD SELECTED DRIVE AND TRACK NUMBER
0D4 0C405C4580D52 1165              NEXT
                1166
                1167                         RAM    DRSEL        DESELECT    DESELECT THE SELECTED DRIVE AND TRACK NUMBERS
0D5 0BCB62A62273C 1168              LDZ      CLRBIT.P3     BIT3              IS THE DENSITY MFM OR GCR ?
                1169
                1170                PASS.S   LIT    R1     B     #01         GET READY TO SET OR CLEAR DENSITY BIT
0D6 0040068C72757 1171              JMPZJ    SETBIT.P3                        THE RETURN IS AT THE END OF SET OR CLEAR BIT
                1172
                1173
                1174   * DESELECTX = 0tttD0dd : X = 0,1,2, OR 3. dd = 0,1,2, or 3. ttt = SELECTED TRACK
                1175   * DESELECT  = 0tttD0dd : dd = SELECTED DRIVE. D = DENSITY. ttt = SELECTED TRACK
                1176   * SELECT    = 0tttD1dd
                1177   * DRV.TRK   = 000tttdd
                1178   * DRIVE     = 000000dd
                1179   * DRIVE.DP  = 000000dd + #5 TO POINT TO BUFFER FOR OPERATING SYSTEM'S DRIVE STATUS'ES
                1180   * DRIVE.DS  = 000000dd + #A TO POINT TO BUFFER OF DESELECT COMMANDS
```

```
                    1183  ********************************************************************************
                    1184  * HERE ARE THE QUICKIE ROUTINES TO ENABLE AND DISABLE THE INTERRUPT ON          *
                    1185  * DRIVE CHANGED STATE. NOTE THAT IF YOU ENABLE THE INTERRUPT ON DRIVE           *
                    1186  * CHANGE STATE WHILE REQUESTING AN INTERRUPT ON COMPLETION, WHEN THAT          *
                    1187  * INTERRUPT ON COMPLETION IS SENT, THE ENABLE INTERUPT ON DRIVE CHANGE         *
                    1188  * STATE WILL BE CLEARED, RENDERING THE WHOLE STORE USELESS.                    *
                    1189  ********************************************************************************
                    1190
                    1191   DIS.INT      PASS.S   LIT    R1     B    #40          PREPARE TO CLEAR THE BIT "ENABLE INTERRUPT"
0D7 1000068C726F1   1192                CALLC    CLRBIT.P1                     ON DRIVE CHANGED STATE" AND GO DO IT.
                    1193
                    1194                NOP                                    IT'S DONE
0D8 0000000621782   1195                JMPU     UPDATE.STS
                    1196
                    1197
                    1198
                    1199   EN.INT       PASS.S   LIT    R1     B    #40          PREPARE TO SET THE BIT "ENABLE INTERRUPT"
0D9 1000068C72711   1200                CALLC    SETBIT.P1                     ON DRIVE CHANGED STATE" AND GO DO IT.
                    1201
                    1202                NOP                                    IT'S DONE.
0DA 0000000621782   1203                JMPU     UPDATE.STS
```

```
                    1206  ********************************************************************************
                    1207  * BELOW IS THE CODE WHICH CALLS ROUTINES TO MOVE THE TAPE TO EOT.               *
                    1208  * IF THE TAPE POSITION IS NOT KNOWN, IT FIRST BACKS UP 24 INCHES                *
                    1209  * TO SEE IF IT IS NEAR A MARKING HOLE. THIS ROUITNE WORKS FOR BOTH              *
                    1210  * FAST FORWARD AND CYCLE TAPE ONCE. IF CYCLE, IT CALLS THE REWIND WHEN          *
                    1211  * DONE.                                                                        *
                    1212  ********************************************************************************
                    1213
                    1214
                    1215   CYCLE.FOR    S.XOR.Q  LIT              #FF          CHECK IF DRIVE IS SELECTABLE AND READY
0D8 3FF8028760DC2   1216   FAST.FOR     NEXT                  NZERO
                    1217
                    1218                NOP
0DC 0000000624E73   1219                JMPC     ERROR.4                     ERROR IF DRIVE IS NOT READY OR SELECTABLE
                    1220
                    1221                PASS.A   R0                          INSPECT STATUS - IS TAPE POSITION KNOWN ?
0DD 002E000440DE2   1222                NEXT                  NBIT6
                    1223
                    1224                NOP                                  GO REVERSE 24 INCHES
0DE 0000000621F31   1225                CALLC    REV.24.IN                   IF TAPE POSITION NOT KNOWN
                    1226
                    1227                PASS.A   R7                          INSPECT R7 TO SEE IF AN OFFLINE ERROR OCCURED
0DF 001800E440E02   1228                NEXT                  ZERO           CALL "LOOK FOR EOT" ONLY IF NO ERROR OCCURRED
                    1229
                    1230                PASS.S   LIT    RB    B    #08       MASK = LOOK FOR EOT
0E0 02002E8C71FC1   1231                CALLC    FOR.700FT                   CALL THE ROUTINE TO FIND EOT WHILE TIMING 700 FEET
                    1232
                    1233                         RAM              OPERAND4   INSPECT BIT IS THIS A CYCLE OR A FAST FORWARD
0E1 0DE902A620E22   1234                NEXT                  NBIT1
                    1235
                    1236                NOP
0E2 0000000620E93   1237                JMPC     CYCLE.REV                   IT WAS A CYCLE-ONCE
                    1238
                    1239                S.AND.A  LIT    R7    Q    #1F       LOAD THE ERROR (IF ANY) INTO Q
0E3 07E01E8252151   1240                CALLC    ERRORBIT1     NEVER          AND SEE IF THE ERROR BIT NEEDS TO BE SET
                    1241
                    1242                S.IOR.A  LIT    R7    B    #20       FLAG "TAPE MOVEMENT FORWARD"
0E4 08001E8DD1782   1243                JMPU     UPDATE.STS                  CUZ IT WAS A FAST FORWARD
```

```
                    1246  ************************************************************************************
                    1247  * THIS CODE DOES A REWIND TO BOT. IF THE TAPE POSTION IS NOT KNOWN INITIALLY,     *
                    1248  * THEN THIS WILL FIRST DO A MOVE FORWARD 24 INCHES TO LOOK FOR A MARKING HOLE.    *
                    1249  * ALSO, THE CYCLE REVERSE USES SOME OF THIS CODE.                                 *
                    1250  ************************************************************************************
                    1251
                    1252   REWIND       S.XOR.Q  LIT                #FF          CONFIRM DRIVE IS SELECTABLE AND READY
0E5 3FF8028760E62   1253                NEXT                 NZERO
                    1254
                    1255                NOP
0E6 0000000624E73   1256                JMPC     ERROR.4                      DRIVE ISN'T SELECTABLE AND READY
                    1257
                    1258                PASS.A   R0                           INSPECT BIT "IS TAPE POSITION KNOWN ?"
0E7 002E000440E82   1259                NEXT                 NBIT6
                    1260
                    1261                NOP
0E8 0000000621F21   1262                CALLC    FOR.24.IN                    LOOK FOR MARKING HOLE IF POSITION NOT KNOWN
                    1263
                    1264   CYCLE.REV    PASS.A   R7                           INSPECT R7 TO SEE IF AN OFFLINE ERROR OCCURED
0E9 001800E440EA2   1265                NEXT                 ZERO             CALL "LOOK FOR BOT" ONLY IF NO ERROR OCCURRED
                    1266
                    1267                PASS.S   LIT    RB   B    #02          MASK = LOOK FOR BOT
0EA 00802E8C71FD1   1268                CALLC    REV.700FT                    TIME 700 FEET WHILE LOOKING FOR BOT
                    1269
                    1270                S.AND.A  LIT    R7   Q    #1F          LOAD THE ERROR (IF ANY) INTO Q
0EB 07E01E8252151   1271                CALLC    ERRORBIT1   NEVER             AND SEE IF THE ERROR BIT NEEDS TO BE SET
                    1272
                    1273                S.IOR.A  LIT    R7   B    #40          FLAG "TAPE MOTION REVERSE"
0EC 10001E8DD1782   1274                JMPU     UPDATE.STS                   GO UPDATE STATUS AND INTERRUPT ON COMPLETION
```

```
                    1277  ********************************************************************************
                    1278  * THIS ROUTINE ERASES TAPE. IT HAS TWO MODES OF OPERATION, DEPENDING ON HOW     *
                    1279  * MANY INCHES OF TAPE IT IS TOLD TO ERASE. IF THE INCH-TO-ERASE COUNT IS        *
                    1280  * GREATER THAN #7FFF (32767 BASE 10, 2730 FEET, APPROX 1/2 MILE), THEN IT       *
                    1281  * WILL ERASE TO EOT, FLAGGING AN ERROR ON A 600 FOOT TIMEOUT OR OFFLINE.        *
                    1282  * IF THE INCH-TO-ERASE COOUNT IS EQUAL TO OR LEES THAN #7FFF, THEN IT WILL      *
                    1283  * ATTEMPT TO ERASE THAT MANY INCHES OF TAPE, FLAGGING AN ERROR ON OFFLINE,      *
                    1284  * A 600 FOOT TIMEOUT, OR IF EOT IS FOUND.                                       *
                    1285  ********************************************************************************
                    1286
                    1287   WRITE.GAP     S.XOR.Q  LIT                #FF          IS REQUESTED OPERATION LEGAL ?
0ED 3FF8028760EE2   1288                 NEXT                 NZERO
                    1289
                    1290                 NOP
0EE 0000000624E73   1291                 JMPC     ERROR.4                      NO - RECORD THE ERROR
                    1292
                    1293                          RAM                 OPERAND3     INSPECT MS INCH COUNT TO CHECK IT'S SIZE
0EF 0DAF02A620F02   1294                 NEXT                 NBIT7
                    1295
                    1296                 PASS.S   LIT    R6   B    #24          SET BITS "THIS OPERATION IS A WRITE FORWARD"
0F0 09001A8C70F53   1297                 JMPC     ERASE.SOME                   JUMP TO ERASE INCHES, ELSE ERASE TO EOT.
                    1298
                    1299
                    1300
                    1301
                    1302   ERASE.ALL     PASS.S   LIT    RB   B    #08          MASK = LOOK FOR EOT
0F1 02002E8C71FE1   1303                 CALLC    ERASE.700FT                  TIME 700 FEET WHILE ERASING TO EOT
                    1304
                    1305
                    1306
                    1307
                    1308   DONE.ERASE    NOP                                   WAIT OUT THE CONDITION BIT
0F2 0000000620F32   1309                 NEXT
                    1310
                    1311                 S.AND.A  LIT    R7   Q    #1F          LOAD ERRORS (IF ANY) INTO Q
0F3 07E01E8252151   1312                 CALLC    ERRORBIT1   NEVER             AND SEE IF THE ERROR BIT NEEDS TO BE SET
                    1313
                    1314                 NOP                                   THIS ERASE IS FINISHED
0F4 0000000621782   1315                 JMPU     UPDATE.STS                   SO UPDATE THE STATUS
```

```
                    1317  ************************************************************************************
                    1318  * THIS PART OF THE ERASE GAP ROUTINE HANDLES ALL THE DETAILS OF ERASING A          *
                    1319  * PARTICULAR NUMBER OF INCHES OF TAPE. SOME OF THESE DIFFICULTIES ARE              *
                    1320  * CONSIDERING THE START AND STOP DISTANCES, ERASING AN EXTRA INCH IF THIS          *
                    1321  * ERASE BEGAN BEFORE THE LOAD POINT, AND ACCOUNTING FOR THE DISTANCE BETWEEN       *
                    1322  * THE WRITE AND ERASE HEADS WHEN PASSING THE LOAD POINT.                           *
                    1323  ************************************************************************************
                    1324
                    1325   ERASE.SOME    DEC.S     RAM              OPERAND4     DECREMENT LS INCH COUNT TO CHECK FOR BORROW
0F5 0DD902A570F62   1326                 NEXT                       NBORROW
                    1327
                    1328                 PASS.S    RAM     RC       B    OPERAND3     LOAD THE MS INCH COUNT INTO RC
0F6 0D8032AC70F93   1329                 JMPC      START.ERASE                      START ERASE IF INCH COUNT > #0000
                    1330
                    1331                 DEC.A     RC      RC       B                DECREMENT MS COUNT SINCE LS COUNT WILL ROLL OVER
0F7 0039318CC0F82   1332                 NEXT                       BORROW
                    1333
                    1334                 NOP
0F8 0000000620F23   1335                 JMPC      DONE.ERASE                       JUMP IF INCH COUNT = #0000
                    1336
                    1337   START.ERASE   NOP                                        TIME TO ERASE TO THE LOAD POINT. IF THE LOAD POINT
0F9 000000623581    1338                 CALLC     ERASE.2.LP                       IS PASSED, THIS ONLY ERASES 20ms (START-UP TIME)
                    1339
                    1340                 PASS.A    R7                               INSPECT R7 FOR ERRORS
0FA 003800E440FB2   1341                 NEXT                       NZERO
                    1342
                    1343                 PASS.A    R0                               INSPECT ORIGINAL DRSTS : WHERE WAS THE LOAD POINT ?
0FB 002A000440F23   1344                 JMPC      DONE.ERASE       NBIT2           ERASE DONE IF ERROR DETECTED ABOVE
                    1345
                    1346                 NOP                                        HOW WE TIME THE FIRST INCH DEPENDS ON WHICH SIDE
0FC 0000000620FF3   1347                 JMPC      BEFORE.LP                        OF THE LOAD POINT THIS ERASE STARTED AT.
                    1348
                    1349  >>>>>>>>>>>>>>>>>>>>
                    1350  >AFTER.LP      PASS.S    LIT     RE       B    #01         LOAD MS .3885 INCH COUNT
0FD 00403A8C7201C   1351  >              LDZ       LOOP.700                         PREPARE TO USE LOOP.700 TO COUNT THE INCHES
                    1352  >
                    1353  >              PASS.S    LIT     RF       B    #36         LOAD LS .3885 INCH COUNT
0FE 0D803E8C71012   1354  >              JMPU      FUNNY.INCH                       GO TIME OUT THIS ONE FUNNY INCH
                    1355  >
                    1356  >BEFORE.LP     PASS.S    LIT     RE       B    #04         LOAD MS 1.308 INCH COUNT
0FF 01003A8C7201C   1357  >              LDZ       LOOP.700                         PREPARE TO USE LOOP.700 TO COUNT THE INCHES
                    1358  >
                    1359  >              PASS.S    LIT     RF       B    #16         LOAD LS 1.308 INCH COUNT
100 05803E8C71012   1360  >              JMPU      FUNNY.INCH                       GO TIME OUT THIS ONE FUNNY INCH
                    1361  >>>>>>>>>>>>>>>>>>>>
                    1362
                    1363   FUNNY.INCH    DEC.S     RAM     RD       B    OPERAND4    DECREMENT AND LOAD LS INCH COUNT, USE ONE SPECIAL
101 0DC036AD720F1   1364                 CALLC     LOOP.1.IN                        INCH TIME TO ALLOW FOR START/STOP DISTANCES ETC.
                    1365
                    1366                 PASS.S    LIT     RB       B    #18         MASK = COUNT THE INCHES OUT
102 06002E8C71FB5   1367                 CALLZJ    OFFLINE                          IF ERROR, ELSE TO LOOP.700 TO TIME THE INCHES
                    1368
                    1369                 NOP                                        AS MUCH TAPE IS ERASED AS WILL BE DONE
103 0000000620F22   1370                 JMPU      DONE.ERASE                       SO GO ARRANGE THE SATUS'ES
```

```
                    1373  ********************************************************************************
                    1374  *    SEARCH BACKWARDS FOR A FILE MARK                                          *
                    1375  ********************************************************************************
                    1376
                    1377   SRCH.FMREV   S.XOR.Q  LIT              #FF          CONTINUE CHECKING IF TAPE MOTION IS LEGAL
104 3FF8028761052   1378                NEXT                  NZERO
                    1379
                    1380                NOP
105 0000000624E73   1381                JMPC     ERROR.4                       ERROR IF TAPE MOTION IS NOT LEGAL
                    1382
                    1383                PASS.S   LIT    R6    B    #08          LOAD INSTRUCTIONS TO READ AND WRITE ROUTINES
106 02001A8C73581   1384                CALLC    REVERSE                       GO START THE TAPE AND BEGIN SEARCH
                    1385
                    1386                S.AND.A  LIT    R7    Q    #1F          LOAD THE ERROR (IF ANY) INTO Q
107 07E01E8252241   1387                CALLC    ERRORBIT3     NEVER           SET THE ERROR BIT IF AN ERROR OCCURRED
                    1388
                    1389                NOP
108 0000000622401   1390                CALLC    CHKOFFLINE2                   CHECK FOR OFFLINE OR BOT
                    1391
                    1392                NOP                                    THE SEARCH IS FINISHED
109 0000000621782   1393                JMPU     UPDATE.STS                    SO GO REPORT WHAT WAS FOUND
```

```
                    1396  ********************************************************************************
                    1397  *    SEARCH FORWARDS FOR A FILE MARK                                           *
                    1398  ********************************************************************************
                    1399
                    1400   SRCH.FMFOR   S.XOR.Q  LIT                #FF          CONTINUE CHECKING IF TAPE MOTION IS LEGAL
10A 3FF8028761082   1401                NEXT                  NZERO
                    1402
                    1403                NOP
10B 000000624E73    1404                JMPC      ERROR.4                      ERROR IF TAPE MOTION IS NOT LEGAL
                    1405
                    1406                PASS.S    LIT     R6     B     #0C     LOAD INSTRUCTIONS TO READ AND WRITE ROUTINES
10C 03001A8C73571   1407                CALLC     FORWARD                      GO START THE TAPE AND BEGIN THE SEARCH
                    1408
                    1409                S.AND.A   LIT     R7     Q     #1F     LOAD THE ERROR (IF ANY) INTO Q
10D 07E01E8252241   1410                CALLC     ERRORBIT3     NEVER          SET THE ERROR BIT IF AN ERROR OCCURRED
                    1411
                    1412                NOP
10E 0000000622401   1413                CALLC     CHKOFFLINE2                  CHECK FOR OFFLINE OR EOT
                    1414
                    1415                NOP                                    THE SEARCH IS FINISHED
10F 0000000621782   1416                JMPU      UPDATE.STS                   SO GO REPORT WHAT WAS FOUND
```

```
                    1419  ********************************************************************************
                    1420  *    SKIP BACKWARDS OVER WHAT EVER IS THERE                                    *
                    1421  ********************************************************************************
                    1422
                    1423   SKIP.REV     S.XOR.Q  LIT                #FF          CONTINUE CHECKING IF TAPE MOTION IS LEGAL
110 3FF8028761112   1424                NEXT                    NZERO
                    1425
                    1426                NOP
111 0000000624E73   1427                JMPC     ERROR.4                         ERROR IF TAPE MOTION IS NOT LEGAL
                    1428
                    1429                         RAM                PRMSTS3      IS THE SELCETED MODE MFM OR GCR ?
112 05C802A621132   1430                NEXT                    BITO             CUZ IF IT'S GCR, WE WANT TO :
                    1431
                    1432                NOP                                      CLEAR OUT THE SECONDARY STATUS
113 0000000621D41   1433                CALLC    CLR.STS2                        IN A SUBROUTINE
                    1434
                    1435                PASS.S   LIT    R6      B    #00         LOAD INSTRUCTIONS TO READ AND WRITE ROUTINES
114 00001A8C73581   1436                CALLC    REVERSE                         GO START THE TAPE AND BEGIN THE SKIP
                    1437
                    1438                S.AND.A  LIT    R7      Q    #1F         LOAD THE ERROR (IF ANY) INTO Q
115 07E01E8252291   1439                CALLC    ERRORBIT4      NEVER            SET THE ERROR BIT IF AN ERROR OCCURRED
                    1440
                    1441                NOP
116 0000000622401   1442                CALLC    CHKOFFLINE2                     CHECK FOR OFFLINE OR BOT
                    1443
                    1444                NOP                                      THE SKIP IS FINISHED
117 0000000621782   1445                JMPU     UPDATE.STS                      SO GO REPORT WHAT WAS FOUND
```

```
                    1448  ********************************************************************************
                    1449  *    SKIP FORWARDS OVER WHAT EVER IS THERE                                     *
                    1450  ********************************************************************************
                    1451
                    1452   SKIP.FOR     S.XOR.Q  LIT                #FF          CONTINUE CHECKING IF TAPE MOTION IS LEGAL
118 3FF8028761192   1453                NEXT                  NZERO
                    1454
                    1455                NOP
119 0000000624E73   1456                JMPC     ERROR.4                         ERROR IF TAPE MOTION IS NOT LEGAL
                    1457
                    1458                         RAM                PRMSTS3      IS THE SELCETED MODE MFM OR GCR ?
11A 05C802A621D4C   1459                LDZ      CLR.STS2  BITO                  CUZ IF IT'S GCR, WE WANT TO :
                    1460
                    1461                NOP                                      CLEAR 2nd, 3rd, & 5th STATUS, AND XOR SPACE IN RAM
11B 0000000621C55   1462                CALLZJ   CLR.STS                         IF MFM, CLEAR ONLY THE SECONDARY STATUS
                    1463
                    1464                PASS.S   LIT    R6   B    #04           LOAD INSTRUCTIONS TO READ AND WRITE ROUTINES
11C 01001A8C73571   1465                CALLC    FORWARD                         GO START THE TAPE AND BEGIN THE SKIP
                    1466
                    1467                S.AND.A  LIT    R7   Q    #1F           LOAD THE ERROR (IF ANY) INTO Q
11D 07E01E8252291   1468                CALLC    ERRORBIT4  NEVER               SET THE ERROR BIT IF AN ERROR OCCURRED
                    1469
                    1470                NOP
11E 0000000622401   1471                CALLC    CHKOFFLINE2                     CHECK FOR OFFLINE OR EOT
                    1472
                    1473                NOP                                      THE SKIP IS FINISHED
11F 0000000621782   1474                JMPU     UPDATE.STS                      SO GO REPORT WHAT WAS FOUND
```

```
                    1477  ********************************************************************************
                    1478  *    WRITE A FILE MARK                                                         *
                    1479  ********************************************************************************
                    1480
                    1481   WRITE.FM     S.XOR.Q  LIT              #FF          CONTINUE CHECKING IF TAPE MOTION IS LEGAL
120 3FF8028761212   1482                NEXT                 NZERO
                    1483
                    1484                NOP
121 0000000624E73   1485                JMPC     ERROR.4                      ERROR IF TAPE MOTION IS NOT LEGAL
                    1486
                    1487                PASS.S   LIT    R6    B    #2C       LOAD INSTRUCTIONS TO READ AND WRITE ROUTINES
122 0B071A8C73591   1488                CALLC    WRITE        PFW            START TAPE AND BEGIN THE WRITE AND READ AFTER WRITE
                    1489
                    1490                NOP                                 OPERATION DONE SO CHECK FOR OFFLINE OR EOT
123 0000000622401   1491                CALLC    CHKOFFLINE2                SINCE THAT IS A FATAL ERROR DURING A WRITE
                    1492
                    1493                S.AND.A  LIT    R7    Q    #1F       LOAD THE ERROR (IF ANY) INTO Q
124 07E01E8252151   1494                CALLC    ERRORBIT1     NEVER          IF ANY ERROR OCCURRED
                    1495
                    1496                NOP                                 THE WRITE AND READ AFTER WRITE ARE FINISHED
125 0000000621782   1497                JMPU     UPDATE.STS                 SO GO REPORT WHAT HAPPENED
```

```
                    1500  ********************************************************************************
                    1501  *    READ THE TAPE (IN A "NORMAL" MANNER)                                      *
                    1502  ********************************************************************************
                    1503
                    1504   READ.TAPE    S.IOR.A  LIT    R0     Q    #17          BEGIN CHECK FOR LEGAL TAPE MOTION
126 05C0028101272   1505                NEXT
                    1506
                    1507                S.XOR.Q  LIT                #FF          CONTINUE CHECKING FOR LEGAL TAPE MOTION
127 3FF8028761282   1508                NEXT                        NZERO
                    1509
                    1510                NOP
128 0000000624E73   1511                JMPC     ERROR.4                         TAPE MOTION REQUESTED WAS ILLEGAL
                    1512
                    1513                         RAM                PRMSTS3      IS THE SELCETED MODE MFM OR GCR ?
129 05C802A621D4C   1514                LDZ      CLR.STS2  BIT0                  CUZ IF IT'S GCR, WE WANT TO :
                    1515
                    1516                NOP                                      CLEAR 2nd, 3rd, & 5th STATUS, AND XOR SPACE IN RAM
12A 0000000621C55   1517                CALLZJ   CLR.STS                         IF MFM, CLEAR ONLY THE SECONDARY STATUS
                    1518
                    1519                PASS.S   LIT    R5     B    #04          TELL I/O ROUTINES TO SEND BYTES TO THE PPU
12B 0100168C712C2   1520                NEXT
                    1521
                    1522                PASS.S   LIT    R6     B    #04          LOAD INSTRUCTIONS TO READ AND WRITE ROUTINES
12C 01001A8C73571   1523                CALLC    FORWARD                         GO BEGIN THE READ
                    1524
                    1525                S.AND.A  LIT    R7     Q    #1F          LOAD THE ERROR (IF ANY) INTO Q
12D 07E01E82521A1   1526                CALLC    ERRORBIT2  NEVER                SET THE ERROR BIT IF AN ERROR OCCURRED
                    1527
                    1528                NOP
12E 0000000622401   1529                CALLC    CHKOFFLINE2                     CHECK FOR OFFLINE OR EOT
                    1530
                    1531                              CLDCO                      CLEAR DATA COMMAND-OUT JUST IN CASE IT'S STILL SET
12F 0000500622771   1532                CALLC    SENDEOR                         GO SEND AN END-OF-RECORD BYTE TO PPU
                    1533
                    1534                NOP                                      THE READ IS FINISHED
130 0000000621782   1535                JMPU     UPDATE.STS                      SO GO REPORT WHAT WAS RED
```

```
                    1538  ********************************************************************************
                    1539  *     SEND THE ERROR CORRECTING INFORMATION AFTER A GCR READ ERROR              *
                    1540  ********************************************************************************
                    1541
                    1542   SEND.XOR     PASS.S   LIT            Q     #00          INSTRUCT SEND.BYTES TO SEND 256 BYTES
131 0000028071322   1543                NEXT
                    1544
                    1545                         LIT    SADRH         #01          POINT TO PAGE 1 IN RAM (THE ODD XOR BLOCKETTE)
132 00405A8621081   1546                CALLC    SEND.BYTES                        AND GO SEND THE BLOCKETTE
                    1547
                    1548                PASS.A   R5                                INSPECT R5 FOR AN OVERFLOW ERROR
133 003800A441342   1549                NEXT                    NZERO
                    1550
                    1551                PASS.S   LIT            Q     #00          INSTRUCT SEND.BYTES TO SEND 256 BYTES
134 0000028071363   1552                JMPC     *+2                               SKIP IF AN OVERFLOW HAS ALREADY OCCURRED
                    1553
                    1554                         LIT    SADRH         #00          POINT TO PAGE 0 IN RAM (THE EVEN XOR BLOCKETTE)
135 00005A8621081   1555                CALLC    SEND.BYTES                        AND GO SEND THE BLOCKETTE
                    1556
                    1557                S.AND.A  LIT    R7      Q     #1F          LOAD ERRORS (IF ANY) INTO Q
136 07E01E82521A1   1558                CALLC    ERRORBIT2      NEVER             SET THE ERROR BIT IF AN ERROR OCCURRED
                    1559
                    1560                NOP              CLDCO                    CLEAR DATA COMMAND OUT JUST IN CASE ITS STILL SET
137 0000000622771   1561                CALLC    SENDEOR                           GO SEND AN END-OF-RECORD BYTE
                    1562
                    1563                NOP
138 0000000621782   1564                JMPU     UPDATE.STS                        AND THE OPERATION IS DONE
                    1565
                    1566
                    1567
                    1568
                    1569
                    1570   * THIS ROUTINE SENDS TO THE PPU THE BYTES IN RAM AT THE FOLLOWING ADDRESSES, IN THE GIVEN ORDER :
                    1571   *
                    1572   *                    0100, 0101, 0102....01FF, 0000, 0001, 0002....00FF.
                    1573   *
                    1574   * THIS IS THE 256 BYTES IN THE ODD XOR BLOCKETTE FOLLOWED BY THE 256 BYTES IN THE
                    1575   * EVEN NUMBERED XOR BLOCKETTE
```

```
                    1578  ************************************************************************************
                    1579  *     READ THIS RECORD WITH THIS BYTE COUNT                                        *
                    1580  *                                                                                  *
                    1581  * IF THE BYTE COUNT BLOCKETTE IN A GCR RECORD IS BAD, THE SYSTEM MAY READ THE      *
                    1582  * BYTE COUNT AT THE END OF THE RECORD AND TELL ME TO READ THE RECORD AGAIN          *
                    1583  * USING A BYTE COUNT WHICH THE SYSTEM WILL "CRAM DOWN MY THROAT" SO THIS           *
                    1584  * CONTROLLER CAN READ THE DATA IN A RECORD WHEN THE LEADING BYTE COUNT IS          *
                    1585  * CLOBBERED.                                                                       *
                    1586  ************************************************************************************
                    1587
                    1588   CRAM.BC      NOP                                          FIRST, MAY ONLY DO THIS IF GCR MODE IS SELECTED
139 0000000624E53   1589                JMPC     ERROR.2                             IF NOT, IT'S AN ILLEGAL STORE COMMAND
                    1590
                    1591                S.IOR.A  LIT    R0     Q     #17            BEGIN CHECKING IF TAPE MOTION IS LEGAL
13A 05C00281D13B2   1592                NEXT
                    1593
                    1594                S.XOR.Q  LIT                 #FF            CONTINUE CHECKING IF TAPE MOTION IS LEGAL
13B 3FF80287613C2   1595                NEXT                         NZERO
                    1596
                    1597                NOP
13C 0000000624E73   1598                JMPC     ERROR.4                             TAPE MOTION REQUESTED IS ILLEGAL
                    1599
                    1600                PASS.S   LIT    R5     B     #04            TELL I/O ROUTINES TO SEND BYTES TO THE PPU
13D 0100168C71C51   1601                CALLC    CLR.STS                             CLEAR 2nd, 3rd, & 5th STATUS'ES & XOR SPACE IN RAM
                    1602
                    1603                PASS.S   LIT    R6     B     #86            LOAD INSTRUCTIONS TO READ AND WRITE ROUTINES
13E 21801A8C713F2   1604                NEXT                                        WHICH INCLUDES FLAGGING "BYTE COUNT VALID"
                    1605
                    1606                PASS.S   RAM    R9     B     OPERAND4       R9 = LS CRAMMED BYTE COUNT
13F 0DC026AC71402   1607                NEXT
                    1608
                    1609                PASS.S   RAM    RA     B     OPERAND3       RA = MS CRAMMED BYTE COUNT
140 0D802AAC71412   1610                NEXT
                    1611
                    1612                PASS.A   R9     RAM          SECSTS4        LOAD LS BYTE COUNT INTO SECONDARY STATUS
141 07805D2441422   1613                NEXT
                    1614
                    1615                DEC.A    R9     RAM          DONECOUNTR     LOAD "LS BYTE COUNT - 1" FOR READ ROUTINE
142 16995D24C1432   1616                NEXT                         NBORROW        SEE IF LS BYTE COUNT = #00
                    1617
                    1618                PASS.A   RA     RAM          SECSTS3        LOAD MS BYTE COUNT INTO SECONDARY STATUS
143 07405D4441463   1619                JMPC     LSBC.NE.0                          JUMP IF LS BYTE COUNT NOT EQUAL TO #00
                    1620
                    1621   LSBC.EQ.0    DEC.A    RA     RA     B                   DECREMENT NUMBER OF FIRST DATA BLOCKETTE
144 0039294CC1452   1622                NEXT                         BORROW         TEST FOR BYTE COUNT = #0000
                    1623
                    1624                PASS.S   LIT    R8     B     #40            FLAG CRAMMED BYTE COUNT = #0000
145 1000228C71543   1625                JMPC     CBC.EQ.0                           GO TO CRAMMED BYTE COUNT = #0000 AND DO THE READ
                    1626
                    1627   LSBC.NE.0    PASS.A   RA                                 BYTE COUNT > 0 SO IS FIRST BLOCKETTE EVEN OR ODD ?
146 0028014441472   1628                NEXT                         NBIT0
                    1629
                    1630                PASS.S   LIT                 Q     #01      PREPARE TO FORCE A DATA ERROR IN FIRST BLOCKETTE
147 0040028071483   1631                JMPC     EVEN.CBC                           JUMP TO EVEN CRAMMED BYTE COUNT, ELSE ODD CBC.
```

```
                    1633  ************************************************************************************
                    1634  * DURING A CRAMMED BYTE COUNT, THE BLOCKETTE NUMBER OF THE FIRST BLOCKETTE IS    *
                    1635  * CALCULATED AND ASSUMED IN ERROR. THE READ ROUTINES ARE INSTRUCTED TO FIND THE  *
                    1636  * SECOND BLOCKETTE AND READ FROM THERE. THIS CODE MUST DO THE WORK OF SETTING UP *
                    1637  * ALL THE REGISTERS THE "READ GCR BYTE COUNT" WOULD HAVE DONE AS WELL AS FLAG THE*
                    1638  * ERROR IN THE FIRST BLOCKETTE AND SEND ZEROES INSTEAD TO THE PPU.               *
                    1639  ************************************************************************************
                    1640
                    1641   ODD.CBC      PASS.S   LIT     R8      B     #04          ERROR COUNT = 1 ODD BLOCKETTE BAD
148 0100228C71492   1642                NEXT
                    1643
                    1644                PASS.Q           RAM           TRDSTS1      ERROR IS A DATA ERROR
149 08405C04214A2   1645                NEXT
                    1646
                    1647                PASS.A   RA      RAM           TRDSTS2      STORE THE ERRONEOUS BLOCKETTE NUMBER
14A 08805D44414E2   1648                JMPU     DEC.BLKNUM
                    1649
                    1650
                    1651   EVEN.CBC     PASS.S   LIT     R8      B     #01          ERROR COUNT = 1 EVEN BLOCKETTE BAD
14B 0040228C714C2   1652                NEXT
                    1653
                    1654                PASS.Q           RAM           TRDSTS3      ERROR IS A DATA ERROR
14C 08C05C04214D2   1655                NEXT
                    1656
                    1657                PASS.A   RA      RAM           TRDSTS4      STORE THE ERRONEOUS BLOCKETTE NUMBER
14D 09005D44414E2   1658                JMPU     DEC.BLKNUM
                    1659
                    1660
                    1661   DEC.BLKNUM   DEC.A    RA      RA      B                  DECREMENT COUNT OF 1st EXPECTED BLOCKETTE NUMBER
14E 0019294CC14F2   1662                NEXT                     NBORROW            SO WE LOOK FOR THE SECOND BLOCKETTE
                    1663
                    1664                ZERO                     Q                  ZERO Q FOR SEND.BYTES ROUTINE
14F 0000000221533   1665                JMPC     SEND.EM                            SEND 256 BYTES IF NOT SKIPPING LAST BLOCKETTE
                    1666
                    1667                S.IOR.A  LIT     R8      B     #80          FLAG EXPECTED BLOCKETTE IS AN XOR BLOCKETTE.
150 2000228DD1512   1668                NEXT
                    1669
                    1670                PASS.S   LIT     RA      B     #01          AND ITS BLOCKETTE NUMBER IS #01
151 00402A8C71522   1671                NEXT
                    1672
                    1673                PASS.A   R9              Q                  Q=# OF BYTES OF DATA IN BLOCKETTE WE ASSUMED WAS BAD
152 0000012041532   1674                JMPU     SEND.EM
                    1675
                    1676
                    1677                         LIT     SADRH         #03          POINT TO PAGE 3 IN RAM, SEND CTC'S INTERNAL STATUS
153 00C05A8621D81   1678   SEND.EM      CALLC    SEND.BYTES                         SEND WHAT SHOULD HAVE BEEN IN FIRST BLOCKETTE
                    1679
                    1680
                    1681   CBC.EQ.0     NOP
154 0000000623571   1682                CALLC    FORWARD                            AND GO DO THE CRAMMED BYTE COUNT READ
                    1683                S.AND.A  LIT     R7      Q     #1F          LOAD THE ERROR (IF ANY) INTO Q
155 07E01E82521A1   1684                CALLC    ERRORBIT2       NEVER              SET THE ERROR BIT IF AN ERROR OCCURRED
                    1685                NOP
156 0000000622401   1686                CALLC    CHKOFFLINE2                        CHECK FOR OFFLINE OR EOT
                    1687                                 CLDCO                      CLEAR DATA COMMAND-OUT JUST IN CASE IT'S STILL SET
157 0000500622771   1688                CALLC    SENDEOR                            GO SEND THE END-OF-RECORD BYTE TO PPU
                    1689                NOP                                         THIS READ IS FINISHED
158 0000000621782   1690                JMPU     UPDATE.STS                         SO GO REPORT WHAT WAS RED
```

```
                 1693  ************************************************************************************
                 1694  *    WRITE A DATA RECORD   OR    WRITE A DATA RECORD AND REPEAT TILL RESET       *
                 1695  * THE WORK FOR BOTH MFM AND GCR IS ALWAYS DONE RATHER THAN DECIDING WHICH TO DO  *
                 1696  ************************************************************************************
                 1697
                 1698   WRITE.DATA    S.XOR.Q  LIT              #FF
159 3FF8028761582 1699                 JMPU     *+2         NZERO          TEST FOR LEGAL TAPE MOTION
                 1700
                 1701   WRITE.RPT     S.XOR.Q  LIT    RAM       #FF        SET FLAG "WRITE REPEAT AT RAM ADDR. #FF
15A 3FF85E8761582 1702                 JMPU     *+1         NZERO          TEST FOR LEGAL TAPE MOTION
                 1703
                 1704                 NOP
158 0000000624E73 1705                 JMPC     ERROR.4                    TAPE MOTION REQUESTED WAS ILLEGAL
                 1706
                 1707  >>>>>>>>>>>>>>>>>>>>
                 1708  >               PASS.S   RAM    RF     BR   OPERAND4    -
15C 0DC03EB472781 1709  >               CALLC    SENDGO                      1
                 1710  >               PASS.S   RAM    RE     BR   OPERAND3    1
15D 0D803AB4715F3 1711  >               JMPC     *+2                         1
                 1712  >               NOP                                  1
15E 0000000624E82 1713  >               JMPU     ERROR.5                     1 - SEND GO BYTE, FLAG ERROR "NO GO BYTE" AND
                 1714  >               A.XOR.B  RE     RF                    1   SHIFT BYTE COUNT RIGHT TO (SORT OF)
15F 000F3DC711602 1715  >               NEXT                    BIT7        1   DIVIDE BY TWO FOR A WORD COUNT, EXCEPT
                 1716  >               PASS.S   LIT            Q    #80    1   IT'S A 15 BIT COUNT WITH THE MOST
160 2000028071623 1717  >               JMPC     *+2                         1   SIGNIFICANT 16th BIT INDICATING IT'S
                 1718  >               PASS.S   LIT            Q    #00    1   GOT ONE EXTRA BYTE ADDED ON
161 0000028071622 1719  >               NEXT                                 1
                 1720  >               A.XOR.Q  RE     RAM          MSWORDCNT 1
162 14C05DC701632 1721  >               NEXT                                 1
                 1722  >               A.XOR.Q  RF     RAM          LSWORDCNT 1
163 14805DE701642 1723  >               NEXT                                 -
                 1724  >>>>>>>>>>>>>>>>>>>>
                 1725  >                        RAM                 PRMSTS3   -
164 05C802A621D4C 1726  >               LDZ      CLR.STS2   BIT0             1 - ALWAYS CLEAR THE 2nd STATUS.
                 1727  >               NOP                                  1   IF GCR, ALSO CLEAR 3rd, 5th, AND XOR SPACE.
165 0000000621C55 1728  >               CALLZJ   CLR.STS                      -
                 1729  >>>>>>>>>>>>>>>>>>>>
                 1730  >               PASS.S   RAM    R1     B    OPERAND4    -
166 0DC006AC71672 1731  >               NEXT                                 1
                 1732  >               DEC.A    R1     RAM          DONECOUNTW 1 - STORE "LS BYTE COUNT - 1" IN RAM TO
167 16405C24C27D1 1733  >               CALLC    REVRS.BITS                   1   MAKE WRITE ROUTINE CALCULATIONS EASIER AND
                 1734  >               PASS.A   R2     RAM          LSBCREV    1   PREPARE AND SAVE A REVERSED COPY OF
168 15C05C4441692 1735  >               NEXT                                 1   THE BYTE COUNT FOR THE GCR WRITE ROUTINES
                 1736  >               PASS.S   RAM    R1     B    OPERAND3    1   TO WRITE THE REVERSE BYTE COUNT IN THE
169 0D8006AC72701 1737  >               CALLC    REVRS.BITS                   1   LAST BLOCKETTE. ALSO THROW AWAY THE FIRST
                 1738  >               PASS.S   RAM             Q    OPERAND2    1   FEW BYTES OF DATA TO FIX A SYSTEM BUG
16A 0D7802A071682 1739  >               NEXT                    NZERO        1   IF TOLD TO DO SO.
                 1740  >               PASS.A   R2     RAM          MSBCREV    1
168 15805C4441E51 1741  >               CALLC    TOSS.BYTES                   -
                 1742  >>>>>>>>>>>>>>>>>>>>
```

```
                    1744  ********************************************************************************
                    1745  * NOW THAT ALL THE INITIALIZATION PRIOR TO A WRITE IS COMPLETE, LET'S           *
                    1746  * GO DO THE WRITE.                                                              *
                    1747  ********************************************************************************
                    1748
                    1749                PASS.S   LIT    R6     B     #24          LOAD INSTRUCTIONS TO READ AND WRITE ROUTINES
16C 09071A8C73591   1750                CALLC    WRITE         PFW               AND GO BEGIN THE WRITE
                    1751
                    1752                NOP                                      THE WRITE IS FINISHED
16D 0000000622771   1753                CALLC    SENDEOR                         GO SEND THE END-OF-RECORD BYTE TO THE PPU
                    1754
                    1755                PASS.S   RAM           Q     SEUDO.ADDR  LOAD WHAT OPERATION THIS WAS INTO Q
16E 0CC002A0716F2   1756                NEXT
                    1757
                    1758                S.AND.Q  LIT           Q     #70         MASK AWAY OTHER BITS
16F 1C00028261702   1759                NEXT
                    1760
                    1761                S.XOR.Q  LIT                 #70         CHECK IF THIS WAS A WRITE REPEAT
170 1C18028761712   1762                NEXT                  ZERO
                    1763
                    1764                NOP
171 0000000621753   1765                JMPC     WRT..RPT                        JUMP IF WRITE REPEAT, ELSE TO WRITE..DATA
                    1766
                    1767
                    1768
                    1769
                    1770
                    1771
                    1772  WRITE..DATA   NOP                                      OPERATION DONE SO CHECK FOR OFFLINE OR EOT
172 0000000622401   1773                CALLC    CHKOFFLINE2                     SINCE THAT IS A FATAL ERROR DURING A WRITE
                    1774
                    1775                S.AND.A  LIT    R7     Q     #1F         LOAD THE ERROR (IF ANY) INTO Q
173 07E01E8252151   1776                CALLC    ERRORBIT1     NEVER             AND SEE IF THE ERROR BIT NNEDS TO BE SET
                    1777
                    1778                NOP                                      WE ARE ALL FINISHED WITH THE WRITE
174 0000000621782   1779                JMPU     UPDATE.STS                      SO GO REPORT ON ITS SUCCESS
                    1780
                    1781
                    1782
                    1783
                    1784
                    1785
                    1786  WRT..RPT      PASS.S   LIT    R7     B     #7F         FORCE WRITE REPEAT ERROR AND TAPE POSITION LOST
175 1FC01E8C71762   1787                NEXT
                    1788
                    1789                S.AND.A  LIT    R7     Q     #1F         LOAD THE ERROR (IF ANY) INTO Q
176 07E01E8252151   1790                CALLC    ERRORBIT1     NEVER             GO SET THE ERROR BIT
                    1791
                    1792                NOP                                      WE ARE ALL FINISHED WITH THE WRITE REPEAT
177 0000000621782   1793                JMPU     UPDATE.STS                      SO GO REPORT ON ITS SUCCESS
```

```
                    1796  ********************************************************************************
                    1797  * HERE I GATHER THE ERRORS AND INFORMATION IN R5, R6, AND R7 AND TRANSLATE       *
                    1798  * IT INTO THE PRIAMRY STATUS. ALSO, THE STATUS OF ALL 4 DRIVES IS REVIEWED,      *
                    1799  * THE ERROR BIT IS SET IF APPLICABLE, AND THE SELECTED DRIVE AND TRACK NUMBER    *
                    1800  * IS STORED AS WELL AS THE SELECTED DRIVE'S STATUS.                              *
                    1801  ********************************************************************************
                    1802
                    1803  >>>>>>>>>>>>>>>>>>>>
                    1804  >UPDATE.STS   PASS.S   RAM            Q     PRMSTS3      LOAD PRMSTS3 INTO Q
178 05C002A071792   1805  >             NEXT
                    1806  >             S.AND.Q  LIT            Q     #01          SAVE ONLY THE DENSITY BIT ALREADY IN PRMSTS3
179 0040028261 7A2  1807  >             NEXT
                    1808  >             S.AND.A  LIT    R5      B     #E0          MASK SO ONLY ERROR BITS OF R5 ARE PRESENT
17A 3800168E51782   1809  >             NEXT
                    1810  >             A.IOR.Q  R5     RAM             PRMSTS3      COMBINE INFORMATION AND STORE IN PRMSTS3
17B 05C35CA5817C2   1811  >             NEXT                    WDL                  TEST IF Write Data was Late
                    1812  >>>>>>>>>>>>>>>>>>>>
                    1813  >             PASS.S   LIT    R1      B     #10          GET READY TO SET BIT "WDL ERROR"
17C 0400068C72751   1814  >             CALLC    SETBIT.P3                           GO SET IT IF WDL IS TRUE
                    1815  >                      LIT    WRCON           #06          TURN OFF THE WRITE LOGIC
17D 01867286217E2   1816  >             NEXT                    RDL                  TEST IF Read Data was Late
                    1817  >             PASS.S   LIT    R1      B     #08          GET READY TO SET BIT "RDL ERROR"
17E 0200068C72751   1818  >             CALLC    SETBIT.P3                           GO SET IT IF RDL IS TRUE
                    1819  >             PASS.A   R6                                  INSPECT BIT "OPERATION WAS CRAMMED BYTE COUNT"
17F 000900C441802   1820  >             NEXT                    BIT1
                    1821  >             PASS.S   LIT    R1      B     #04          GET READY TO SET BIT "CRAMMED BYTE COUNT"
180 0100068C72751   1822  >             CALLC    SETBIT.P3                           SET IT IF IT WAS A CRAMMED BYTE COUNT
                    1823  >             PASS.A   R6                                  INSPECT BIT "BYTE COUNT IN SECONDARY STATUS VALID ?"
181 000F00C441822   1824  >             NEXT                    BIT7
                    1825  >             PASS.S   LIT    R1      B     #02          GET READY TO SET "BYTE COUNT VALID" BIT
182 0080068C72751   1826  >             CALLC    SETBIT.P3                           SET IT IF BYTE COUNT IS VALID
                    1827  >>>>>>>>>>>>>>>>>>>>
                    1828  >                      LIT    RDCON           #00          TURN OFF THE READ LOGIC
183 00006E8621841   1829  >             CALLC    CHEKEM.ALL                          GO REVIEW THE STATE OF ALL 4 DRIVES
                    1830  >             PASS.A   R7     RAM             PRMSTS2      STORE R7 INTO PRMSTS2
184 05805CE442371   1831  >             CALLC    ERRORBIT5                           GO SET THE ERROR BIT IF A FATAL ERROR HAPPENED
                    1832  >>>>>>>>>>>>>>>>>>>>
                    1833  >             PASS.S   RAM            Q     PRMSTS1      LOAD PRIMARY STATUS BYTE 1 INTO Q
185 054002A071862   1834  >             NEXT
                    1835  >             S.AND.Q  LIT            Q     #E0          MASK OUT THE SELECTED DRIVE AND TRACK NUMBERS
186 3800028261872   1836  >             NEXT
                    1837  >             S.IOR.Q  RAM            Q     DRV.TRK      ADD SELECTED DRIVE AND TRACK TO INFORMATION IN Q
187 0C4002A1E1882   1838  >             NEXT
                    1839  >             PASS.Q          RAM             PRMSTS1      RE-LOAD PRIMARY STATUS BYTE 1
188 05405C0421892   1840  >             JMPU     INT.COMPLT                          AND GO SEE IF WE INTERRUPT ON COMPLETION
                    1841  >>>>>>>>>>>>>>>>>>>>
```

```
                    1844  ********************************************************************************
                    1845  * THIS ROUTINE CHECKS IF WE ARE SUPPOSED TO INTERRUPT ON COMPLETION OF         *
                    1846  * THIS STORE. IT ALSO DE-SELECTS THE DRIVE. IF WE ARE SUPPOSED TO INTERRUPT,   *
                    1847  * THIS ROUTINE WILL UPDATE THE SECONDARY STATUS IF IT IS VALID AND UPDATE      *
                    1848  * THE PRIMARY STATUS AND REQUEST AN INTERRUPT. WHEN ALL THIS IS DONE, IT       *
                    1849  * RETURNS TO THE IDLE LOOP AND CHECKS THE STATUS OF THE DRIVES.                *
                    1850  ********************************************************************************
                    1851
                    1852   INT.COMPLT           RAM                  SEUDO.ADDR   LOOK AT BIT "INTERRUPT ON COMPLETION"
189 0CEF02A6218A2   1853                NEXT                 NBIT7
                    1854
                    1855                        RAM    DRSEL         DESELECT     DE-SELECT THE DRIVE IN CASE IT'S STILL SELECTED
18A 0BC062A620263   1856                JMPC     CHEKDRIVES                       IF NOT INT. ON COMPL., GO TO THE IDLE LOOP
                    1857
                    1858                PASS.S   LIT    R1      B     #40          CLRBIT ENABLING INT. ON DRIVE STATUS CHANGE
18B 1000068C726F1   1859                CALLC    CLRBIT.P1
                    1860
                    1861                        RAM                  PRMSTS3      INSPECT BIT "BYTE COUNT VALID"
18C 05E902A6218D2   1862                NEXT                 NBIT1
                    1863
                    1864                PASS.S   LIT    R4      B     #06          THERE ARE SIX BYTES TO BE SENT
18D 0180128C71923   1865                JMPC     SEND.PRIM                        DO WE SEND SECONDARY STATUS ?
                    1866
                    1867   SEND.SEC     PASS.S   LIT    SADRL   B     SECSTSO      YES : POINT TO SECONDARY STATUS
18E 06804E8C718F2   1868                NEXT
                    1869
                    1870                        LIT    SCR           #34          FIRST BYTE SAYS "UPDATING SECONDARY STATUS"
18F 0D00DE8621961   1871                CALLC    SENDSTRING                       GO SEND THE SECONDARY STATUS
                    1872
                    1873                PASS.S   LIT    R4      B     #06          NOW THERE ARE 6 BYTES OF PRIMARY STATUS TO SEND
190 0180128C71923   1874                JMPC     SEND.PRIM                        IF SECONDARY STATUS WENT OKAY, SEND PRIAMRY STATUS
                    1875
                    1876                NOP                                       THE SECONDARY STATUS WAS NOT SUCCESSFULLY SENT
191 0000000620262   1877                JMPU     CHEKDRIVES                       SO GO BACK TO THE IDLE LOOP
                    1878
                    1879   SEND.PRIM    PASS.S   LIT    SADRL   B     PRMSTSO      POINT TO PRIMARY STATUS IN RAM
192 05004E8C71932   1880                NEXT
                    1881
                    1882                        LIT    SCR           #54          FIRST BYTE SAYS "UPDATE PRIM. STAT. AND REQ. INT."
193 1500DE8621961   1883                CALLC    SENDSTRING                       GO SEND THE PRIMARY STATUS
                    1884
                    1885                PASS.S   LIT    R1      B     #80          PREPARE TO CLEAR BIT "DRIVE CHANGED STATE"
194 2000068C726F1   1886                CALLC    CLRBIT.P1                        DO IT IF PRIMARY STATUS WENT OKAY
                    1887
                    1888                NOP                                       WE ARE DONE INTERRUPTING ON COMPLETION IF WE COULD
195 0000000620262   1889                JMPU     CHEKDRIVES                       SO GO TO WAIT STATE
                    1890
                    1891
                    1892  ********************************************************************************
                    1893  * FOLLOWING ARE THE SUBROUTINES WHICH ARE USED PRETTY MUCH BY ALL OF THIS      *
                    1894  * CODE, ESPECIALLY THE PPU-INTERFACE PART. AFTER THAT ARE THE ACTUAL READ      *
                    1895  * AND WRITE ROUTINES AND THE STRUCTURE THEY EXIST IN.                          *
                    1896  ********************************************************************************
```

```
                    1899  ********************************************************************************
                    1900  * THIS SUBROUTINE SENDS A STRING OF BYTES TO THE PPU. THE RAM POINTER           *
                    1901  * (SADRH & SADRL & R3) MUST BE POINTING TO THE FIRST BYTE TO BE SENT AND        *
                    1902  * R4 MUST TELL HOW MANY BYTES ARE TO BE SENT. IF THE PPU DOES NOT TAKE         *
                    1903  * ALL THE BYTES FOR ANY REASON, THIS SUBROUTINE WILL EXIT WITH THE             *
                    1904  * CONDITION BIT FALSE. OTHERWISE, THE CONDITION BIT WILL BE TRUE. THIS         *
                    1905  * SUBROUTINE WILL DO ONE TRY AND 4 RE-TRYS.                                    *
                    1906  ********************************************************************************
                    1907
                    1908   SENDSTRING   PASS.S   LIT    R2     B    #04        LOAD THE RETRY COUNT INTO R2
196 01100A8C71972   1909                NEXT                CFO
                    1910
                    1911                NOP
197 0000000621A63   1912                JMPC     GIVEUP                        GIVE UP IF SOMETHING ALREADY COMING FROM PPU
                    1913
                    1914   RETRY        PASS.A   R4     R1     B              COPY BYTE COUNT TO BE SENT INTO R1
198 0000048C41992   1915                NEXT
                    1916
                    1917                PASS.S   LIT           Q    #FF       INITIALIZE Q PRIOR TO GENERATING BCC THERE
199 3FD20280719A2   1918                NEXT                CFI
                    1919
                    1920   BEGINSEND    NOP                                  WAIT FOR COMMAND FLAG-IN TO GO AWAY
19A 00120006219A3   1921                JMPC     *              CFI             (JUST IN CASE IT'S THERE)
                    1922
                    1923                S.XOR.Q  SCR    COUT   Q              LOAD BYTE TO SEND AND COMPUTE BCC
198 0010C6A3619C2   1924                NEXT                CFO             WHICH ALSO SETS COMMAND COMMAND-OUT
                    1925
                    1926  >>>>>>>>>>>>>>>>>>>>
                    1927  >             NOP                                  -
19C 0032000621A33   1928  >             JMPC     WHATCAME       NCFI         1 - POLE COMMAND FLAG-OUT AND FLAG-IN TO SEE
                    1929  >             NOP                                  1     TO SEE IF THIS BYTE IS ACCEPTED
19D 0010000619C3    1930  >             JMPC     *-1            CFO          -
                    1931  >>>>>>>>>>>>>>>>>>>>
                    1932
                    1933   BYTETAKEN                  CLCCO                  CLEAR COMMAND COMMAND-OUT
19E 00005406219F2   1934                NEXT
                    1935
                    1936                DEC.A    R1     R1     B              DECREMENT COUNTER OF BYTES STILL TO BE SENT
19F 0018042CC1A02   1937                NEXT                ZERO
                    1938
                    1939                S.XOR.A  LIT    R1          #01       IF ONE BYTE TO SEND, IT'S THE BCC
1A0 0078068751AC3   1940                JMPC     WAITFORACK     NZERO
                    1941
                    1942                INC.A    R3     SADRL  B              INCREMENT RAM POINTER TO NEXT BYTE TO SEND
1A1 00124C6C499A3   1943                JMPC     BEGINSEND      CFI          NOT SENDING THE BCC YET
                    1944
                    1945   SENDBCC      PASS.Q          SCR                   LOAD THE BCC INTO RAM
1A2 0012DC04219A2   1946                JMPU     BEGINSEND      CFI          AND GO SEND LAST BYTE
```

```
                    1948  ********************************************************************************
                    1949  * THIS CODE IS AN EXTENSION OF THE SENDSTRING SUBROUTINE                       *
                    1950  * IT FIGURES OUT WHAT TO DO IF THE PPU SENT SOMETHING                          *
                    1951  * UNEXPECTEDLY. IF IT WAS A NACK, AND IF THE RETRY COUNT IS NOT EXPIRED,       *
                    1952  * THEN IT RESTORES THE RAM POINTERS AND TRIES AGAIN.                           *
                    1953  ********************************************************************************
                    1954
                    1955   WHATCAME     PASS.S   LIT          Q    NACK        LOAD A NACK INTO Q FOR COMPARISON
1A3 3E13028071A42   1956                NEXT                  CCI
                    1957
                    1958                NOP
1A4 0013000621A43   1959                JMPC     *            CCI              WAIT FOR COMMAND COMMAND-IN TO GO AWAY
                    1960
                    1961                S.XOR.Q  CIN                            COMPARE INCOMING BYTE TO NACK IN Q
1A5 0038022761A62   1962                NEXT                  NZERO            WHICH ALSO CLEARS COMMAND FLAG-OUT
                    1963
                    1964   GIVEUP       NOP                                    ERROR = NEW STRING COMING FROM PPU
1A6 002000062000A   1965                RETURNC               NEVER            RETURN WITH CONDITION BIT CLEAR - TRANSFER FAILED
                    1966
                    1967                DEC.A    R1     R1    B                PART OF COMPLEX PROCESS OF RESTORING
1A7 0000042CC1A82   1968                NEXT                                   THE RAM POINTERS
                    1969
                    1970   TRY.RETRY    DEC.A    R2     R2    B                DECREMENT THE RETRY COUNT
1A8 0039084CC1A92   1971                NEXT                  BORROW
                    1972
                    1973                NOP                                    ERROR = RETRY COUNT EXHAUSTED
1A9 002000062000A   1974                RETURNC               NEVER            RETURN WITH CONDITION BIT CLEAR - TRANSFER FAILED
                    1975
                    1976                B.SUB.A  R4     R3    B                CONTINUE RESTORING RAM POINTERS FOR NEXT RETRY
1AA 00000C8C99AB2   1977                NEXT
                    1978
                    1979                A.AD1.B  R1     SADRL B                FINISH RESTORING RAM POINTERS
1AB 00004C2C19982   1980                JMPU     RETRY                         GO TRY ANOTHER RETRY
```

```
                 1982  ********************************************************************************
                 1983  * HERE IS MORE OF THE SUBROUTINE "SENDSTRING".                                 *
                 1984  * THE STRING WAS SENT AND NOW ITS TIME TO WAIT FOR AN ACK OR A NACK            *
                 1985  * AND SEE WHICH IT IS.                                                         *
                 1986  ********************************************************************************
                 1987
                 1988   WAITFORACK    NOP
1AC 0030000621AC3 1989                JMPC     *             NCFO              WAIT FOR COMMAND FLAG-OUT
                 1990
                 1991                 NOP
1AD 0013000621AE2 1992                NEXT                   CCI               SET UP COMMAND COMMAND-IN CONDITION
                 1993
                 1994                 NOP
1AE 0013000621AE3 1995                JMPC     *             CCI               WAIT FOR COMMAND COMMAND-IN TO GO AWAY
                 1996
                 1997                 PASS.S   CIN           Q                 LOAD THE BYTE INTO Q
1AF 0000022071B02 1998                NEXT                                     WHICH ALSO CLEARS COMMAND FLAG-OUT
                 1999
                 2000                 S.XOR.Q  LIT                ACK          COMPARE BYTE RECEIVED TO ACK
1B0 0398028761B12 2001                NEXT                   ZERO
                 2002
                 2003                 S.XOR.Q  LIT                NACK         COMPARE BYTE RECEIVED TO NACK
1B1 3E3802876000A 2004                RETURNC                NZERO             IF ACK, RETURN W/CONDITION BIT SET - ALL'S OKAY
                 2005
                 2006                 NOP                                      IF ERROR = NEW STRING COMING,
1B2 002000062000A 2007                RETURNC                NEVER             RETURN WITH CONDITION BIT CLEAR - TRANSFER FAILED
                 2008
                 2009                 NOP
1B3 0000000621A82 2010                JMPU     TRY.RETRY                       IT WAS NACKED, TRY RETRY-ING IT
                 2011
                 2012
                 2013  * ERROR : I WILL GOBBLE AND LOOSE THE FIRST BYTE OF THE NEXT TRANSFER IF IT IS IN PLACE OF A NACK
```

```
                    2016  ********************************************************************************
                    2017  * THE FOLLOWING SUBROUTINE IS CALLED PERIODICALLY FROM THE IDLE LOOP AND FROM    *
                    2018  * UPDATE.STATUS JUST PRIOR TO INTERRUPTING ON COMPLETION TO REVIEW THE STATUS    *
                    2019  * OF ALL FOUR DRIVES. IT UPDATES ALL THE RECORDS OF EACH DRIVE'S STATUS AND      *
                    2020  * SETS THE APPROPRIATE BIT IN THE PRIMARY STATUS IF ONE OR MORE OF THE DRIVES    *
                    2021  * HAS CHANGED STATE. IT ALSO SETS THE RAM POINTER TO PAGE #3.                    *
                    2022  ********************************************************************************
                    2023
                    2024   CHEKEM.ALL   PASS.S   LIT    DRSEL  B    #07          SELECT DRIVE 3 INITIALLY
1B4 01C0628C71B52   2025                NEXT                                     ALSO LOADS A #07 INTO R8 (DRSEL'S EQUIVELENT VALUE)
                    2026
                    2027                        LIT    SADRH       #03          SELECT PAGE 3 IN RAM
1B5 00C05A8621B62   2028                NEXT
                    2029
                    2030                PASS.S  LIT    SADRL  B    DRSTS3       POINT TO DRIVE 3'S STATUS IN RAM
1B6 00C04E8C71B72   2031                NEXT                                     STORE THAT POINTER IN R3 (SADRL'S EQUIVELENT VALUE)
                    2032
                    2033                PASS.S  LIT    R4     B    #05          LOAD 5 INTO R4. 5 IS THE DIFFERENCE BETWEEN
1B7 0140128C71B82   2034                NEXT                                     THE OP.SYS'S DRIVE STATUS BUFFER AND THE CTC'S
                    2035
                    2036  >>>>>>>>>>>>>>>>>>>>
                    2037  >LOOPC.A      PASS.S  DRSTS  R0     B                RECORD THIS DRIVE'S CURRENT STATUS
1B8 0000024C72591   2038  >             CALLC   SCRAMBLE                        GO SCRAMBLE DRSTS BITS FOR OPERATING SYSTEM
                    2039  >
                    2040  >             A.ADD.B R4     SADRL  B                 ADVANCE RAM POINTER TO OPERATING SYSTEM'S
1B9 00004C8C118A2   2041  >             NEXT                                    DRIVE STATUS BUFFER
                    2042  >
                    2043  >             PASS.Q          SCR                     LOAD LATEST DRIVE STATUS FOR OPERATING SYSTEM
1BA 0000DC0421BB2   2044  >             NEXT
                    2045  >
                    2046  >             B.SUB.A R4     SADRL  B                 RESTORE RAM POINTER
1BB 00004C8C99BC2   2047  >             NEXT
                    2048  >
                    2049  >             S.XOR.A SCR    R0                       COMPARE CURRENT STATUS TO LAST STATUS
1BC 003882A751BD2   2050  >             NEXT                   NZERO
                    2051  >
                    2052  >             PASS.S  LIT    R1     B    #80          GET READY TO SET BIT "DRIVE CHANGED STATE"
1BD 2000068C72711   2053  >             CALLC   SETBIT.P1                       GO SET THE BIT IF A DRIVE CHANGED STATE
                    2054  >
                    2055  >             PASS.A  R0     SCR                      LOAD THE LATEST STATUS INTO RAM
1BE 0000DC0441BF2   2056  >             NEXT
                    2057  >
                    2058  >             DEC.A   R8     DRSEL  B                 DECREMENT THE SELECTED DRIVE
1BF 0000610CC1C02   2059  >             NEXT
                    2060  >
                    2061  >             DEC.A   R3     SADRL  B                 DECREMENT THE POINTER INTO RAM
1C0 003A4C6CC1C12   2062  >             NEXT                   NSIGN            ARE WE DONE CHECKING ALL FOUR DRIVES ?
                    2063  >
                    2064  >             NOP
1C1 0000000621B83   2065  >             JMPC    LOOPC.A                         IF NOT DONE, CONTINUE CHECKING DRIVES
                    2066  >>>>>>>>>>>>>>>>>>>>
                    2067
                    2068   DONEC.A      PASS.S  RAM    SADRL       DRIVE.OP     -
1C2 0B404EA471C32   2069                NEXT                                     1
                    2070                PASS.S  SCR            Q                1 - UPDATE PRIMARY STATUS'S VERSION
1C3 000082A071C42   2071                NEXT                                     1       OF SELECTED DRIVE'S STATUS
                    2072                PASS.Q          RAM         PRMSTS4      1
1C4 0600[illegible]42000A   2073        RETURNC                                  -
```

```
                    2076  ********************************************************************************
                    2077  * THIS ONE ZEROES OUT THE TERTIARY, QUINTARY,AND SECODARY STATUS'ES AND ZEROES   *
                    2078  * OUT THE SPACE IN RAM WHERE THE XOR BLOCKETTES IN GCR ARE WRITTEN.              *
                    2079  * A SUBSET OF THIS ONE ONLY ZEROES OUT THE SECONDARY STATUS.                     *
                    2080  ********************************************************************************
                    2081
                    2082
                    2083   CLR.STS      ZERO            RAM         TRDSTS1
1C5 08405C0621C62   2084                NEXT
                    2085                ZERO            RAM         TRDSTS2
1C6 08805C0621C72   2086                NEXT
                    2087                ZERO            RAM         TRDSTS3
1C7 08C05C0621C82   2088                NEXT
                    2089                ZERO            RAM         TRDSTS4
1C8 09005C0621C92   2090                NEXT
                    2091                ZERO            RAM         QNTSTS1
1C9 09C05C0621CA2   2092                NEXT
                    2093                ZERO            RAM         QNTSTS2
1CA 0A005C0621CB2   2094                NEXT
                    2095                ZERO            RAM         QNTSTS3
1CB 0A405C0621CC2   2096                NEXT
                    2097                ZERO            RAM         QNTSTS4
1CC 0A805C0621CD2   2098                NEXT
                    2099  >>>>>>>>>>>>>>>>>>>>
                    2100  >             PASS.S   LIT    SADRH       #01          -
1CD 00405A8471CE2   2101  >             NEXT                                     1
                    2102  >             PASS.S   LIT    SADRL  B    #FF          1
1CE 3FC04E8C71CF2   2103  >             NEXT                                     1
                    2104  >CLR.RAM1     DEC.A    R3     SADRL  B                 1
1CF 00194C6CC1D02   2105  >             NEXT                   NBORROW           1
                    2106  >             ZERO            SCR                      1 - ZERO OUT PAGES 0 AND 1 OF RAM FOR THE
1D0 0000DC0621CF3   2107  >             JMPC     CLR.RAM1                        1    XOR BLOCKETTE'S SPACE
                    2108  >             PASS.S   LIT    SADRH       #00          1
1D1 00005A8471D22   2109  >             NEXT                                     1
                    2110  >CLR.RAM2     DEC.A    R3     SADRL  B                 1
1D2 00194C6CC1D32   2111  >             NEXT                   NBORROW           1
                    2112  >             ZERO            SCR                      1
1D3 0000DC0621D23   2113  >             JMPC     CLR.RAM2                        -
                    2114  >>>>>>>>>>>>>>>>>>>>
                    2115
                    2116
                    2117   * THIS ONE CLEARS OUT ONLY THE SECONDARY STATUS
                    2118
                    2119
                    2120   CLR.STS2     ZERO            RAM         SECSTS1
1D4 06C05C0621D52   2121                NEXT
                    2122                ZERO            RAM         SECSTS2
1D5 07005C0621D62   2123                NEXT
                    2124                ZERO            RAM         SECSTS3
1D6 07405C0621D72   2125                NEXT
                    2126                ZERO            RAM         SECSTS4
1D7 07805C062000A   2127                RETURNC
```

```
                    2130  ************************************************************************************
                    2131  * THIS ONE SENDS Q BYTES TO THE PPU WHICH COME OUT OF THE PAGE IN RAM WHICH      *
                    2132  * THE CALLING ROUTINE POINTED TO. IF Q=0, THEN 256 BYTES ARE SENT AND RAM        *
                    2133  * ADDRESSES ARE :  #00,01,02...#FF.                                              *
                    2134  ************************************************************************************
                    2135
                    2136
                    2137
                    2138   SEND.BYTES   PASS.S   LIT     RF      B     #FF          LOAD TIMEOUT OF APPROX 10 MILLI-SECONDS
1D8 3FE03E8C71E32   2139                JMPU     SB7             NEVER              AND JUMP RIGHT INTO THE SEND ROUTINE
                    2140
                    2141   SB1                   CLRTC                              CLEAR RTC
1D9 0020026622853   2142                JMPC     OVRFLW          NEVER              FLAG OVERFLOW IF TIMEOUT EXPIRED
                    2143
                    2144  >>>>>>>>>>>>>>>>>>>>
                    2145  >SB2          NOP                                        POLE RTC AND DFI TO SEE WHICH COMES FIRST
1DA 0021000621DD3   2146  >             JMPC     SB3             NRTC
                    2147  >             NOP
1DB 0016000621DA3   2148  >             JMPC     *-1             DFI
                    2149  >>>>>>>>>>>>>>>>>>>>
                    2150
                    2151                DEC.A    RF      RF      B                  DECREMENT TIMEOUT TIMER
1DC 00393DECC1D92   2152                JMPU     SB1             BORROW
                    2153
                    2154   SB3                           CLDCO                      CLEAR DATA COMMAND-OUT
1DD 0020500621DF2   2155                JMPU     SB5             NEVER
                    2156
                    2157   SB4                   CLRTC                              CLEAR RTC
1DE 0020026622853   2158                JMPC     OVRFLW          NEVER              FLAG OVERFLOW IF TIMEOUT EXPIRED
                    2159
                    2160  >>>>>>>>>>>>>>>>>>>>
                    2161  >SB5          NOP                                        POLE RTC AND NDFI TO SEE WHICH COMES FIRST
1DF 0021000621E23   2162  >             JMPC     SB6             NRTC
                    2163  >             NOP
1E0 0036000621DF3   2164  >             JMPC     *-1             NDFI
                    2165  >>>>>>>>>>>>>>>>>>>>
                    2166
                    2167                DEC.A    RF      RF      B                  DECREMENT TIMEOUT TIMER
1E1 00393DECC1DE2   2168                JMPU     SB4             BORROW
                    2169
                    2170   SB6          DEC.Q                    Q                  DECREMENT COUNT OF HOW MANY BYTES TO SEND
1E2 00180000A1E32   2171                NEXT                     ZERO
                    2172
                    2173   SB7          NEG.Q                    SADRL              CALCULATE AND LOAD POINTER INTO RAM
1E3 00004C052800A   2174                RETURNC                                     RETURN IF ALL BYTES ARE SENT
                    2175
                    2176                                 SCR     DOUT               SEND THE BYTE POINTED TO IN RAM
1E4 0020C2A621DA2   2177                JMPU     SB2             NEVER              AND LOOP UP TO SEND THIS BYTE
```

```
                    2180  ********************************************************************************
                    2181  * THIS LITTLE ROUTINE THROWS AWAY DATA BYTES FROM THE PPU PRIOR TO STARTING A    *
                    2182  * WRITE. THIS IS A PATCH TO FIX A SYSTEM FLAW THAT DATA BUFFERS CAN BEGIN ON     *
                    2183  * BYTE BOUNDRIES IN MEMORY, BUT DATA TRANSFERS MUST BEGIN ON WORD BOUNDRIES.     *
                    2184  ********************************************************************************
                    2185
                    2186
                    2187
                    2188   TOSS.BYTES   PASS.S   LIT    RF     B    #FF          LOAD TIMEOUT OF APPROX 10 MILLI-SECONDS
1E5 3FC03E8C71F12   2189                JMPU     TB7                             AND JUMP RIGHT INTO THE TOSS BYTES ROUTINE
                    2190
                    2191   TB1                   CLRTC                           CLEAR RTC
1E6 0020026622863   2192                JMPC     UNDFLW       NEVER              FLAG UNDERFLOW IF TIMEOUT EXPIRED
                    2193
                    2194  >>>>>>>>>>>>>>>>>>>>
                    2195  >TB2          NOP                                     POLE RTC AND DFO TO SEE WHICH COMES FIRST
1E7 0021000621EA3   2196  >             JMPC     TB3          NRTC
                    2197  >             NOP
1E8 0014000621E73   2198  >             JMPC     *-1          DFO
                    2199  >>>>>>>>>>>>>>>>>>>>
                    2200
                    2201                DEC.A    RF     RF    B                 DECREMENT TIMEOUT TIMER
1E9 00393DECC1E62   2202                JMPU     TB1          BORROW
                    2203
                    2204   TB3                                                  WAIT OUT CONDITION BIT
1EA 0020000621EC2   2205                JMPU     TB5          NEVER
                    2206
                    2207   TB4                   CLRTC                           CLEAR RTC
1EB 0020026622863   2208                JMPC     UNDFLW       NEVER              FLAG UNDERFLOW IF TIMEOUT EXPIRED
                    2209
                    2210  >>>>>>>>>>>>>>>>>>>>
                    2211  >TB5          NOP                                     POLE RTC AND NDCI TO SEE WHICH COMES FIRST
1EC 0021000621EF3   2212  >             JMPC     TB6          NRTC
                    2213  >             NOP
1ED 0037000621EC3   2214  >             JMPC     *-1          NDCI
                    2215  >>>>>>>>>>>>>>>>>>>>
                    2216
                    2217                DEC.A    RF     RF    B                 DECREMENT TIMEOUT TIMER
1EE 00393DECC1EB2   2218                JMPU     TB4          BORROW
                    2219
                    2220   TB6          DEC.Q                 Q                 DECREMENT COUNT OF HOW MANY BYTES TO TAKE
1EF 00180000A1F02   2221                NEXT                   ZERO
                    2222
                    2223                         DIN                           THROW AWAY THIS BYTE, CLEAR DATA FLAG-OUT
1F0 000002062000A   2224                RETURNC                                  RETURN IF ALL BYTES ARE TAKEN
                    2225
                    2226   TB7          NOP
1F1 0020000621E72   2227                JMPU     TB2          NEVER              LOOP UP TO TAKE THE NEXT BYTE
```

```
                    2230  ************************************************************************************
                    2231  * BELOW IS THE SUBROUTINE WHICH MOVES THE TAPE 24 INCHES                           *
                    2232  * IT FLAGS AN ERROR IF THE DRIVE GOES OFFLINE                                      *
                    2233  * IT ALSO STOPS THE TAPE IF A MARKING HOLE IS FOUND                                *
                    2234  ************************************************************************************
                    2235
                    2236   FOR.24.IN            LIT    DRCON       #04          INSTRUCT TAPE TO GO FORWARD SLOW
1F2 0100668621F42   2237                JMPU    LOAD.24                         SKIP
                    2238
                    2239   REV.24.IN            LIT    DRCON       #02          INSTRUCT TAPE TO GO REVERSE SLOW
1F3 0080668621F42   2240                JMPU    LOAD.24
                    2241
                    2242  >>>>>>>>>>>>>>>>>>>>
                    2243  >LOAD.24      PASS.S  LIT    RD     B    #18          LOAD 24 (BASE 10) INTO THE TIMER
1F4 0600368C71F52   2244  >             NEXT
                    2245  >>>>>>>>>>>>>>>>>>>>
                    2246
                    2247   LOOP.24      DEC.A   RD     RD     B               DECREMENT THE 24 INCH COUNTER
1F5 003935ACC1F62   2248                NEXT                  BORROW
                    2249
                    2250                NOP
1F6 0000000621FA3   2251                JMPC    DONE.24                         END IF 24 INCHES HAVE PASSED
                    2252
                    2253                NOP
1F7 000000062200D1  2254                CALLC   WAIT.1.IN                       GO WAIT FOR THE TAPE TO MOVE ONE INCH
                    2255
                    2256                PASS.A  R0                              INPECT R0 : TAPE POSITION KNOWN NOW ?
1F8 002E000441FB3   2257                JMPC    OFFLINE       NBIT6             GO RECORD THE OFFLINE ERROR IF TRUE
                    2258
                    2259                NOP
1F9 0000000621F53   2260                JMPC    LOOP.24                         LOOP IF TAPE POSITION STILL UNKNOWN
                    2261
                    2262   DONE.24      NOP
1FA 0000000625002   2263                JMPU    STOP.NOW                        STOP TAPE, RETURN FROM THERE
                    2264
                    2265
                    2266
                    2267
                    2268
                    2269   OFFLINE      S.IOR.A LIT    R7     B    #60          FLAG "TAPE POSITION LOST"
1FB 18001E8DD4EA2   2270                JMPU    ERROR.7                         AND GO FLAG THE ERROR "OFFLINE"
```

```
                    2273  ********************************************************************************
                    2274  * THIS ROUTINE MOVES TAPE WHILE TIMING THIS MOVEMENT AND LOOKING FOR EOT OR BOT. *
                    2275  * THE MASK IN RB DETERMINES HOW THIS ROUTINE EXITS :                             *
                    2276  * RB = #02 : IGNORE EOT, EXIT ON BOT, ERROR IF TIMEOUT OR OFFLINE               *
                    2277  *      #08 : IGNORE BOT, EXIT ON EOT, ERROR IF TIMEOUT OR OFFLINE               *
                    2278  *      #18 : IGNORE BOT, EXIT ON TIMEOUT, ERROR IF EOT OR OFFLINE               *
                    2279  * THE TIMER IS FOR 700 FEET AT 30 IPS, UNLESS LOADED ELSWHERE                    *
                    2280  ********************************************************************************
                    2281
                    2282   FOR.700FT            LIT    DRCON      #14          MOVE TAPE FORWARD, FAST
1FC 0500668621FF2   2283                JMPU    LOAD.700                       SKIP
                    2284   REV.700FT            LIT    DRCON      #12          MOVE TAPE REVERSE, FAST
1FD 0480668621FF2   2285                JMPU    LOAD.700
                    2286   ERASE.700FT          LIT    DRCON      #0C          MOVE TAPE FORWARD, SLOW, WRITE (ERASE)
1FE 0300668621FF2   2287                JMPU    LOAD.700
                    2288
                    2289  >>>>>>>>>>>>>>>>>>>>
                    2290  >LOAD.700     PASS.S   LIT    RD      B    #D0          -
1FF 3400368C72002   2291  >             NEXT                                    1 - #2000 = 8400 INCHES = 700 FEET
                    2292  >             PASS.S   LIT    RC      B    #20          1   WHICH IS LOADED INTO COUNTER RC,RD
200 0800328C72012   2293  >             NEXT                                    -
                    2294  >>>>>>>>>>>>>>>>>>>>
                    2295
                    2296   LOOP.700     DEC.A    RD     RD      B               DECREMENT THE LS INCH COUNTER
201 001935ACC2022   2297                NEXT                    NBORROW
                    2298                NOP
202 0000000622083   2299                JMPC     NOT700YET                      JUMP IF COUNTER NOT EXPIRED
                    2300                DEC.A    RC     RC      B               DECREMENT THE MS COUNT SINCE THE LS ROLLED OVER
203 0019318CC2042   2301                NEXT                    NBORROW
                    2302                NOP
204 0000000622083   2303                JMPC     NOT700YET                      JUMP IF COUNTER NOT EXPIRED
                    2304
                    2305                PASS.A   RB                             IF WE GET HERE, THE COUNTER HAS EXPIRED
205 000C016442062   2306                NEXT                    BIT4           SO INSPECT THE EXIT INSTRUCTIONS
                    2307
                    2308                NOP
206 0000000625003   2309                JMPC     STOP.NOW                       EXIT HERE IF TIMEOUT IS NO ERROR
                    2310
                    2311                S.IOR.A  LIT    R7      8    #60        FLAG "TAPE POSITION LOST"
207 18001E8DD4E92   2312                JMPU     ERROR.6                        AND RECORD THE TIMEOUT ERROR
                    2313
                    2314   NOT700YET    NOP                                     THE TIMER HAS NOT EXPIRED
208 0000000622001   2315                CALLC    WAIT.1.IN                      SO GO WAIT ANOTHER INCH
                    2316
                    2317                A.AND.B  R0     RB                      USE RB TO MASK AND INSPECT THE DRIVE STATUS IN R0
209 00382C0611FB3   2318                JMPC     OFFLINE        NZERO           FLAG OFFLINE ERROR IF ONE OCCURRED
                    2319
                    2320                PASS.A   RB                             INSPECT EXIT INSTRUCTIONS IN RB
20A 002C016442013   2321                JMPC     LOOP.700       NBIT4           BUT LOOP IF EOT/BOT IS NOT YET FOUND
                    2322
                    2323                NOP                                     I GUESS AN END OF TAPE HAS BEEN FOUND
20B 0000000625003   2324                JMPC     STOP.NOW                       SO STOP THE TAPE IF IT'S NOT AN ERROR
                    2325
                    2326                S.IOR.A  LIT    R7      B    #60        FLAG "TAPE POSITION LOST"
20C 18001E8DD4FA2   2327                JMPU     ERROR.17                       AND FLAG UNEXPECTED EOT ERROR
```

```
                    2330  ********************************************************************************
                    2331  * THIS SUBROUTINE TIMES OFF ONE INCH OF TAPE AND RETURNS WITH THE               *
                    2332  * CONDITION BIT TRUE IF THE DRIVE HAS GONE OFFLINE, FALSE OTHERWISE             *
                    2333  ********************************************************************************
                    2334
                    2335   WAIT.1.IN    PASS.S   LIT    RF     B     #1F          LOAD LS COUNT OF ONE INCH
20D 07C03E8C720E2   2336                NEXT
                    2337
                    2338                PASS.S   LIT    RE     B     #03          LOAD MS COUNT OF ONE INCH
20E 00C03A8C720F2   2339                NEXT
                    2340
                    2341
                    2342
                    2343
                    2344
                    2345   LOOP.1.IN    NOP
20F 00210006220F3   2346                JMPC     *              NRTC             WAIT FOR RTC
                    2347
                    2348                DEC.A    RF     RF      B                DECREMENT THE LS COUNTER
210 00193DECC2112   2349                NEXT                    NBORROW
                    2350
                    2351                         CLRTC                           CLEAR RTC
211 00000266220F3   2352                JMPC     LOOP.1.IN                      LS COUNTER DIDN'T ROLL OVER SO CONTINUE
                    2353
                    2354                DEC.A    RE     RE      B                DECREMENT THE MS COUNT SINCE LS COUNT ROLLED OVER
212 001939CCC2132   2355                NEXT                    NBORROW
                    2356
                    2357                NOP
213 00000006220F3   2358                JMPC     LOOP.1.IN                      IF MS COUNT DIDN'T ROLL OVER, CONTINUE
                    2359
                    2360
                    2361
                    2362
                    2363
                    2364   DONE.1.IN    PASS.S   DRSTS  R0     B                LOAD LATEST DRIVE STATUS INTO R0
214 002D024C7000A   2365                RETURNC                 NBIT5            RETURN W/CONDITION = DRIVE NOT READY
```

```
                    2368  ********************************************************************************
                    2369  * THE ERROR BIT IS SET WHEN AN ERROR WHICH REQUIRES A RETRY HAPPENS OR WHEN      *
                    2370  * AN ERROR WHICH CLEARLY SHOWS SOMETHING IS AFOUL (SUCH AS ILLEGAL STORE)        *
                    2371  * OCCURS. IT IS IMPOSSIBLE TO DOCUMENT THIS AS IT IS A VERY "INTELLIGENT"       *
                    2372  * PIECE OF CODE. SEE THE CTC EXTERNAL SPECS FOR DETAILS.                        *
                    2373  ********************************************************************************
                    2374
                    2375  ERRORBIT1     S.XOR.Q  LIT                  #07          ENTER AFTER WRITE DATA, REPEAT, OR FILE MARK
215 01D8028762162   2376                NEXT                  ZERO                AND FAST FORWARD, REWIND, CYCLE, AND WRITE GAP
                    2377                S.XOR.Q  LIT                  #17
216 05D80287622C3   2378                JMPC     SET.IT       ZERO
                    2379                S.XOR.Q  LIT                  #12
217 04980287622C3   2380                JMPC     SET.IT       ZERO
                    2381                S.XOR.Q  LIT                  #11
218 04580287622C3   2382                JMPC     SET.IT       ZERO
                    2383                S.XOR.Q  LIT                  #10
219 04180287622C3   2384                JMPC     SET.IT       ZERO
                    2385  ERRORBIT2     S.XOR.Q  LIT                  #16          ENTER AFTER A READ, CRAMMED BYTE COUNT, XOR BYTES
21A 05980287622C3   2386                JMPC     SET.IT       ZERO
                    2387                S.XOR.Q  LIT                  #15
21B 05580287622C3   2388                JMPC     SET.IT       ZERO
                    2389  HANDLED.15    S.XOR.Q  LIT                  #14          RE-ENTER AFTER HANDLING ERROR #15
21C 05180287622D3   2390                JMPC     HANDLE.15    ZERO
                    2391  HANDLED.14    S.XOR.Q  LIT                  #13          RE-ENTER AFTER HANDLING ERROR #14
21D 04D8028762333   2392                JMPC     HANDLE.14    ZERO
                    2393                S.XOR.Q  LIT                  #0F
21E 03D80287622C3   2394                JMPC     SET.IT       ZERO
                    2395                S.XOR.Q  LIT                  #0E
21F 03980287622C3   2396                JMPC     SET.IT       ZERO
                    2397                PASS.A   R5
220 000F00A4422C3   2398                JMPC     SET.IT       BIT7
                    2399                PASS.A   R5
221 000E00A4422C3   2400                JMPC     SET.IT       BIT6
                    2401                PASS.A   R5
222 000D00A4422C3   2402                JMPC     SET.IT       BIT5
                    2403                NOP
223 00030006222C3   2404                JMPC     SET.IT       WDL
                    2405  ERRORBIT3     NOP                                       ENTER AFTER A FILE MARK SEARCH, FORWARD OR REVERSE
224 00060006222C3   2406                JMPC     SET.IT       RDL
                    2407                S.XOR.Q  LIT                  #0D
225 03580287622C3   2408                JMPC     SET.IT       ZERO
                    2409                S.XOR.Q  LIT                  #0C
226 03180287622C3   2410                JMPC     SET.IT       ZERO
                    2411                S.XOR.Q  LIT                  #0B
227 02D80287622C3   2412                JMPC     SET.IT       ZERO
                    2413                S.XOR.Q  LIT                  #0A
228 02980287622C3   2414                JMPC     SET.IT       ZERO
                    2415  ERRORBIT4     S.XOR.Q  LIT                  #09          ENTER AFTER SKIP FORWARD OR REVERSE
229 02580287622C3   2416                JMPC     SET.IT       ZERO
                    2417                S.XOR.Q  LIT                  #08
22A 02380287622C3   2418                JMPC     SET.IT       NZERO
                    2419                NOP                                       EXIT
22B 000000062000A   2420                RETURNC
                    2421  SET.IT        PASS.S   LIT    R1    B     #20          PREPARE TO SET THE ERROR BIT
22C 0800068C72712   2422                JMPU     SETBIT.P1                       GO SET THE ERROR BIT AND RETURN FROM THERE
```

```
                    2424  ************************************************************************************
                    2425  * ERROR 15 (RECOVERABLE READ) IS ERROR 16 (UNRECOVERABLE) IF AN OVER FLOW          *
                    2426  * OCCURRED. IF NOT OVERFLOW AND NOT DATA ERROR, THE ERROR BIT NEED NOT BE SET.     *
                    2427  * ERROR 14 (CRAMMED BYTE COUNT BLOCKETTE NOT FOUND) IS NOT AN ERROR IF THE         *
                    2428  * CRAMMED BYTE COUNT = #0000, SO IN THAT CASE THE ERROR BIT IS NOT SET.            *
                    2429  ************************************************************************************
                    2430
                    2431  HANDLE.15     PASS.A   R5                                  INSPECT R5 FOR AN OVERFLOW ERROR
22D 002E00A4422E2   2432                NEXT                 NBIT6
                    2433
                    2434                         RAM                 TRDSTS1         INSPECT TERTIARY STATUS FOR DATA ERROR
22E 084802A622303   2435                JMPC     *+2         BIT0                    SKIP IF NOT OVERFLOW
                    2436
                    2437  CNG.15.16     INC.A    R7     R7   B                       OVERFLOW SO ERROR #15 IS ERROR #16
22F 00001CEC4A2C2   2438                JMPU     SET.IT                               AND GO SET THE ERROR BIT
                    2439
                    2440                         RAM                 TRDSTS3         INSPECT TERTIARY STATUS FOR DATA ERROR
230 08C802A6222C3   2441                JMPC     SET.IT      BIT0                    JUMP IF DATA ERROR
                    2442
                    2443                NOP
231 00000006222C3   2444                JMPC     SET.IT                               JUMP IF DATA ERROR
                    2445
                    2446                NOP                                          THERE'S NO ERROR HERE SO BACK TO LOOK FOR OTHERS
232 0020006221C2    2447                JMPU     HANDLED.15  NEVER
                    2448
                    2449
                    2450
                    2451
                    2452
                    2453  HANDLE.14     PASS.S   RAM                 SECSTS4         INSPECT FOR BYTE COUNT = #0000
233 078802A472342   2454                NEXT                 NZERO
                    2455
                    2456                PASS.S   RAM                 SECSTS3         INSPECT FOR BYTE COUNT = #0000
234 077802A4722C3   2457                JMPC     SET.IT      NZERO                   SET ERRORBIT IF BYTE COUNT # #0000
                    2458
                    2459                NOP
235 00000006222C3   2460                JMPC     SET.IT                               SET ERRORBIT IF BYTE COUNT # #0000
                    2461
                    2462                NOP                                          THERE'S NO ERROR HERE SO BACK TO LOOK FOR OTHERS
236 0020006221D2    2463                JMPU     HANDLED.14  NEVER
```

```
                    2465  ********************************************************************************
                    2466  * HERE IS THE ERROR BIT ROUTINE WHICH APPLIES TO ALL ! STORES AND IS           *
                    2467  * THEREFOR CALLED FROM THE ROUTINE "UPDATE.STS" RATHERE THAN                   *
                    2468  * FROM EACH INDIVIDUAL STORE ROUTINE.                                         *
                    2469  * IT SETS THE ERROR BIT ON ERRORS 1, 2, 3, 4, 5, OR 6.                        *
                    2470  ********************************************************************************
                    2471
                    2472   ERRORBIT5    S.AND.A  LIT    R7     Q    #1F          ISLOATE THE ERROR BITS OF R7
237 07C01E8252382   2473                NEXT
                    2474
                    2475                S.XOR.Q  LIT                #01
238 0058028762392   2476                NEXT                 ZERO
                    2477                S.XOR.Q  LIT                #02
239 00980287623F3   2478                JMPC     SET..IT     ZERO
                    2479                S.XOR.Q  LIT                #03
23A 00D80287623F3   2480                JMPC     SET..IT     ZERO
                    2481                S.XOR.Q  LIT                #04
23B 01180287623F3   2482                JMPC     SET..IT     ZERO
                    2483                S.XOR.Q  LIT                #05
23C 01580287623F3   2484                JMPC     SET..IT     ZERO
                    2485                S.XOR.Q  LIT                #06
23D 01B80287623F3   2486                JMPC     SET..IT     NZERO
                    2487
                    2488                NOP
23E 000000062000A   2489                RETURNC
                    2490
                    2491   SET..IT      PASS.S   LIT    R1     B    #20          PREPARE TO SET THE ERROR BIT
23F 0800068C72712   2492                JMPU     SETBIT.P1                       GO SET THE ERROR BIT IN PRMSTS1 : RETURN FROM THERE
```

```
                    2495  ************************************************************************************
                    2496  * THE ROUTINES WHICH READ/WRITE FROM/TO THE TAPE CHECK WHETHER THE TAPE            *
                    2497  * WENT OFFLINE OR EOT OR BOT WAS FOUND AS THE LAST THING THEY DO.                  *
                    2498  * IF SO, ANY ERROR THAT MAY OR MAY NOT HAVE OCCURRED IS OVERWRITTEN BY THE         *
                    2499  * ERROR OFFLINE OR EOT OR BOT SINCE THAT IS THE PROBABLE CAUSE OF AN ERROR         *
                    2500  * IF THERE WAS ONE, AND IF NOT, IT IS A VALID ERROR TO BE FLAGGED.                 *
                    2501  ************************************************************************************
                    2502
                    2503   CHKOFFLINE2  PASS.A   R6                                  INSPECT BIT "FORWARD OR REVERSE ?"
240 002A00C442412   2504                NEXT                  NBIT2
                    2505
                    2506                PASS.A   R0           B                     INSPECT THE DRIVE STATUS IN R0
241 002D000C42453   2507                JMPC     REV.2        NBIT5
                    2508
                    2509   FOR.2        PASS.A   R0                                 INSPECT DRIVE STATUS AGAIN FOR EOT
242 000B000442483   2510                JMPC     OFFLINE2     BIT3                   ERROR IF DRIVE NOT READY
                    2511
                    2512                PASS.S   LIT          Q     #17             RECORD EOT ERROR IN Q, BUT
243 05C002807000A   2513                RETURNC                                     RETURN IF NOT EOT
                    2514
                    2515                S.IOR.A  LIT    R7    B     #60             FLAG "TAPE POSITION LOST"
244 18001E8DD24A2   2516                JMPU     FINISH.ERR                       AND FLAG ERROR "EOT FOUND"
                    2517
                    2518   REV.2        PASS.A   R0                                 INSPECT DRIVE STATUS AGAIN FOR BOT
245 0009000442483   2519                JMPC     OFFLINE2     BIT1                   ERROR IF DRIVE NOT READY
                    2520
                    2521                PASS.S   LIT          Q     #18             RECORD BOT ERROR IN Q, BUT
246 060002807000A   2522                RETURNC                                     RETURN IF NOT BOT
                    2523
                    2524                S.IOR.A  LIT    R7    B     #60             FLAG "TAPE POSITION LOST"
247 18001E8DD24A2   2525                JMPU     FINISH.ERR                       AND GO FLAG ERROR "BOT FOUND"
                    2526
                    2527   OFFLINE2     PASS.S   LIT          Q     #07             RECORD OFFLINE ERROR IN Q
248 01C0028072492   2528                NEXT
                    2529
                    2530                S.IOR.A  LIT    R7    B     #60             FLAG "TAPE POSITION LOST"
249 18001E8DD24A2   2531                JMPU     FINISH.ERR                       AND GO RECORD THE OFFLINE ERROR
                    2532
                    2533   FINISH.ERR   S.AND.A  LIT    R7    B     #E0             MASK ANY OTHER ERRORS IN R7
24A 38001E8E52482   2534                NEXT
                    2535
                    2536                A.IOR.Q  R7     R7    B                     STORE NEW ERROR IN R7
248 00001CED8000A   2537                RETURNC                                     AND RETURN
```

```
                    2540  ********************************************************************************
                    2541  * THESE SUBROUTINES ARE REALLY NEAT. ONCE CALLED, THEY RETURN WHEN THE AMOUNT    *
                    2542  * OF TIME INDICATED IN THE SUBROUTINE NAME HAS EXPIRED                           *
                    2543  ********************************************************************************
                    2544
                    2545  WAIT.35.MS    PASS.S   LIT    RF     B    #FF          LOAD LS COUNT OF 35 MILLISECOND WAIT
24C 3FC03E8C724D2   2546                NEXT
                    2547
                    2548                PASS.S   LIT    RE     B    #D1          LOAD MS COUNT OF 35 MILLISECOND WAIT
24D 34403A8C72542   2549                JMPU     W.RERF.S                       GO WAIT
                    2550
                    2551  WAIT.12.MS    PASS.S   LIT    RF     B    #FF          LOAD LS COUNT OF 12 MILLISECOND WAIT
24E 3FC03E8C724F2   2552                NEXT
                    2553
                    2554                PASS.S   LIT    RE     B    #47          LOAD MS COUNT OF 12 MILLISECOND WAIT
24F 11C03A8C72542   2555                JMPU     W.RERF.S                       GO WAIT
                    2556
                    2557  WAIT.5.MS     PASS.S   LIT    RF     B    #FF          LOAD LS COUNT OF 5 MILLISECOND WAIT
250 3FC03E8C72512   2558                NEXT
                    2559
                    2560                PASS.S   LIT    RE     B    #1D          LOAD MS COUNT OF 5 MILLISECOND WAIT
251 07403A8C72542   2561                JMPU     W.RERF.S                       GO WAIT
                    2562
                    2563  WAIT.16.US    PASS.S   LIT    RF     B    #18          LOAD LS COUNT OF 16 MICROSECOND WAIT
252 06003E8C72532   2564                NEXT
                    2565
                    2566                PASS.S   LIT    RE     B    #00          LOAD MS COUNT OF 16 MICROSECOND WAIT
253 00003A8C72542   2567                JMPU     W.RERF.S                       GO WAIT
                    2568
                    2569  W.RERF.S      DEC.A    RF     RF     B                 DECREMENT THE LS COUNT
254 00393DECC2552   2570                NEXT                   BORROW
                    2571
                    2572                DEC.A    RE     RE     B                 DECREMENT THE MS COUNT
255 003939CCC2573   2573                JMPC     *+2            BORROW            SKIP IF LS COUNT DID ROLL OVER
                    2574
                    2575                INC.A    RE     RE     B                 UN-DO THE DECREMENT ABOVE
256 000039CC4A582   2576                JMPU     *+2                              SKIP
                    2577
                    2578                NOP
257 000000062000A   2579                RETURNC                                  RETURN IF BOTH COUNTERS ROLLED OVER
                    2580
                    2581                NOP
258 0000000622542   2582                JMPU     W.RERF.S                        LOOP UNTIL BOTH COUNTERS DO ROLL OVER
```

```
                    2585  ************************************************************************************
                    2586  * HERE ARE SOME LATE ADDITIONAL ROUTINES TO PLEASE THE OPERATING SYSTEM          *
                    2587  ************************************************************************************
                    2588
                    2589   SCRAMBLE     ZERO                    Q                THIS ONE TAKES THE DRIVE STATUS IN RO AND
259 000000002225A2  2590                NEXT                                     TRANSLATES IT INTO A VERSION THE OPERATING SYSTEM
                    2591                PASS.A   RO                              LIKES IN Q
25A 002F000442582   2592                NEXT                    NBIT7            SEE BEGINNING PAGES FOR A DESCRIPTION OF THIS
                    2593                PASS.A   RO                              TRANSLATION
25B 000E000442503   2594                JMPC     *+2            BIT6
                    2595                S.IOR.Q  LIT            Q     #02
25C 00800281E25B2   2596                JMPU     *-1
                    2597                PASS.A   RO
25D 002D0004425F3   2598                JMPC     *+2            NBIT5
                    2599                S.IOR.Q  LIT            Q     #04
25E 01000281E25D2   2600                JMPU     *-1
                    2601                PASS.A   RO
25F 000C000442613   2602                JMPC     *+2            BIT4
                    2603                S.IOR.Q  LIT            Q     #20
260 08000281E25F2   2604                JMPU     *-1
                    2605                PASS.A   RO
261 000B000442633   2606                JMPC     *+2            BIT3
                    2607                S.IOR.Q  LIT            Q     #40
262 10000281E2612   2608                JMPU     *-1
                    2609                PASS.A   RO
263 000A000442653   2610                JMPC     *+2            BIT2
                    2611                S.IOR.Q  LIT            Q     #80
264 20000281E2632   2612                JMPU     *-1
                    2613                PASS.A   RO
265 0009000442673   2614                JMPC     *+2            BIT1
                    2615                S.IOR.Q  LIT            Q     #10
266 04000281E2652   2616                JMPU     *-1
                    2617                PASS.A   RO
267 0008000442693   2618                JMPC     *+2            BIT0
                    2619                S.IOR.Q  LIT            Q     #01
268 00400281E2672   2620                JMPU     *-1
                    2621                NOP
269 000000062000A   2622                RETURNC
                    2623                S.IOR.Q  LIT            Q     #08
26A 02000281E000A   2624                RETURNC
                    2625
                    2626
                    2627
                    2628
                    2629   SHIFT        PASS.S   RAM    R1      BR    SEUDO.ADDR  THIS ONE SHIFTS THE PSEUDO-ADDRESS FROM
26B 0CC006B4726C2   2630                NEXT                                     THE PPU WHILE MASKING AWAY SOME BITS SO
                    2631                S.AND.A  LIT    R1      BR    #38         THE RESULT CAN BE INDEXED ON TO QUICKLY
26C 0E00069652602   2632                NEXT                                     DECODE WHAT TO DO WITH THIS ORDER
                    2633                PASS.A   R1     R1      BR                FROM THE PPU
26D 00000434426E2   2634                NEXT
                    2635                PASS.A   R1     R1      BR
26E 000004344000A   2636                RETURNC
```

```
                    2639  ********************************************************************************
                    2640  * A SPLATTERING OF SMALL SUBROUTINES                                           *
                    2641  ********************************************************************************
                    2642
                    2643
                    2644   * THESE TWO SET/CLEAR BITS IN THE FIRST AND THIRD BYTES OF THE PRIMARY STATUS
                    2645
                    2646
                    2647   CLRBIT.P1    NOT.A    R1      R1      B                INVERT R1 SO IT TELLS WHICH BITS TO CLEAR WITH 0'S
26F 0000042D42702   2648                NEXT
                    2649
                    2650                S.AND.A  RAM     R1      B     PRMSTS1    CLEAR THE BITS
270 054006AE52722   2651                JMPU     *+2                              SKIP TO SETBIT.P1'S RETURN STATEMENT
                    2652
                    2653   SETBIT.P1    S.IOR.A  RAM     R1      B     PRMSTS1    SET PRMSTS1'S BITS WHICH R1 HAS SET
271 054006ADD2722   2654                NEXT
                    2655
                    2656                PASS.A   R1      RAM           PRMSTS1    RELOAD THE PRIMARY STATUS
272 05405C244000A   2657                RETURNC
                    2658
                    2659   CLRBIT.P3    NOT.A    R1      R1      B                INVERT R1 SO IT TELLS WHICH BITS TO CLEAR WITH 0'S
273 0000042D42742   2660                NEXT
                    2661
                    2662                S.AND.A  RAM     R1      B     PRMSTS3    CLEAR THE BITS
274 05C006AE52762   2663                JMPU     *+2                              SKIP TO SETBIT.P3'S RETURN STATEMENT
                    2664
                    2665   SETBIT.P3    S.IOR.A  RAM     R1      B     PRMSTS3    SET PRMSTS3'S BITS WHICH R1 HAS SET
275 05C006ADD2762   2666                NEXT
                    2667
                    2668                PASS.A   R1      RAM           PRMSTS3    RELOAD THE PRIMARY STATUS
276 05C05C244000A   2669                RETURNC
                    2670
                    2671
                    2672
                    2673   * THESE TWO TELL THE SENDSTRING SUBROUTINE TO SEND EOR AND GO BYTES
                    2674
                    2675
                    2676
                    2677   SENDEOR      PASS.S   LIT             Q     EOR        LOAD EOR BYTE INTO Q
277 2400028072792   2678                JMPU     *+2                              SKIP
                    2679
                    2680
                    2681   SENDGO       PASS.S   LIT             Q     GO         LOAD GO BYTE INTO Q
278 1800028072792   2682                NEXT
                    2683
                    2684                PASS.Q           RAM           SPARE      LOAD Q INTO SPARE BUFFER IN RAM
279 0E805C04227A2   2685                NEXT
                    2686
                    2687                PASS.S   LIT     SADRL   B     SPARE      POINT TO THE SPARE BUFFER IN RAM
27A 0E804E8C72782   2688                NEXT
                    2689
                    2690                         LIT     SADRH         #03        POINT TO PAGE 3 IN RAM
27B 00C05A86227C2   2691                NEXT
                    2692
                    2693                PASS.S   LIT     R4      B     #02        THERE ARE 2 BYTES TO BE SENT
27C 0080128C71962   2694                JMPU     SENDSTRING                       RETURN FROM THE END OF "SENDSTRING"
```

```
                    2697  ********************************************************************************
                    2698  * MORE LITTLE SUBROUTINES                                                      *
                    2699  ********************************************************************************
                    2700
                    2701   * THIS ONE TAKES THE BITS IN R1 AND COPIES THEM BACKWARDS INTO R2
                    2702   * IT HELPS WRITE GCR'S REVERSE BYTE COUNT
                    2703
                    2704   REVRS.BITS   PASS.S   LIT    R3     B    #07          LOAD A COUNTER INTO R3 : 7+1 BITS TO REVERSE
27D 01C00E8C7282C   2705                LDZ      DONE.RB                         POINT Z TO END OF THIS ROUTINE
                    2706
                    2707   LOOP.RB      PASS.A   R1     R1     BL               SHIFT R1 LEFT AROUND ONCE
27E 000F043C427F2   2708                NEXT                    BIT7            LOOK AT BIT 7 TO SEE IF IT'S SET OR CLEAR
                    2709
                    2710                DEC.A    R3     R3     B                DECREMENT THE COUNT OF BITS TO REVERSE
27F 00190C6CC2813   2711                JMPC     *+2            NBORROW         SKIP TO SET A BIT IN R2
                    2712
                    2713                S.BCL.A  LIT    R2     BR   #01         CLEAR R2'S BIT
280 00400A96D27E7   2714                JMPZJ    LOOP.RB                        CONTINUE LOOP ON REVERSE BITS IF NOT DONE
                    2715
                    2716                S.IOR.A  LIT    R2     BR   #01         SET R2'S BIT
281 00400A95D27E7   2717                JMPZJ    LOOP.RB                        CONTINUE LOOP ON REVERSE BITS IF NOT DONE
                    2718
                    2719   DONE.RB      NOP                                     ALL 8 BITS ARE SWAPPED AROUND
282 000000062000A   2720                RETURNC                                 SO RETURN
                    2721
                    2722
                    2723   * HERE THE UNDER FLOW AND OVER FLOW ERRORS ARE FLAGGED
                    2724
                    2725
                    2726   LASTOVRFLW   NOP                                     THE CONDITION BIT = I/O THINKS NOT DONE
283 0036000622853   2727                JMPC     OVRFLW         NDFI            SEE IF PPU FINISHED IT'S PART
                    2728
                    2729                NOP
284 000000062000A   2730                RETURNC                                 RETURN IF PPU CLEARED ITS DATA FLAG OUT
                    2731
                    2732   OVRFLW       S.IOR.A  LIT    R5     B    #48         SET BITS "ERROR DETECTED" AND "OVER FLOW"
285 1200168DD2872   2733                JMPU     *+2                            SKIP TO RETURN STATEMENT
                    2734
                    2735   UNDFLW       S.IOR.A  LIT    R5     B    #28         SET BITS "ERROR DETECTED" AND "UNDER FLOW"
286 0A00168DD2872   2736                JMPU     *+1                            GO TO RETURN STATEMENT
                    2737
                    2738                S.BCL.A  LIT    R5     B    #07         CLEAR ANY FETCH OR SEND INSTRUCTIONS
287 01C0168ED000A   2739                RETURNC                                 AND RETURN
                    2740
                    2741
                    2742
                    2743   * SEE READ/WRITE ROUTINES FOR A DESCRIPTION OF R5'S USE.
```

```
                    2331  ********************************************************************************
                    2332  * HERE IS THE PART OF GAPFINDER WHICH LOOKS FOR 1/3 INCH DROPOUT. IT DOES       *
                    2333  * THIS BY DECREMENTING A COUNTER EACH TIME DATA IS NOT DETECTED AND              *
                    2334  * INCREMENTING IT EACH TIME DATA IS DETECTED, BUT IT DOES NOT INCREMENT IT      *
                    2335  * BEYOND THE VALUE THAT REPRESENTS 1/3 INCH OF TAPE. THIS ALGORYTHM MAKES       *
                    2336  * THIS ROUTINE NOISE IMMUNE, BUT NOT VERY ACCURATE WITH MARGINAL TAPES.         *
                    2337  ********************************************************************************
                    2338
                    2339  NOT.WRITE           CLRTC                                 ITS NOT A WRITE, RTC WAS SET
4CD 003D026624CF3   2340                JMPC    *+2             NRDD              SKIP IF WE NEEDN'T DECREMENT MS COUNTER R0,RC
                    2341
                    2342                DEC.A   R0      R0      B                 DECREMENT MS GENERAL PURPOSE COUNTER R0,RC
4CE 003D000CC4CF2   2343                JMPU    *+1             NRDD              TEST FOR READ DATA DETECTED
                    2344
                    2345                NOP                                       NOW TO DECIDE IF WE ARE IN A DROPOUT OR NOT
4CF 0000000624D63   2346                JMPC    NDATA.HERE
                    2347
                    2348  DATA.HERE     S.XOR.A RAM     RF          MSTHIRDIN.    COMPARE MS COUNTER TO 1/3 INCH TIME
4D0 15383EA754D12   2349                NEXT                    NZERO
                    2350
                    2351                S.XOR.A RAM     RE          LSTHIRDIN.    COMPARE LS COUNTER TO 1/3 INCH TIME
4D1 15583AA754D33   2352                JMPC    INC.GAPC        ZERO              IF UNEQUAL, WE MAY INCREMENT THE GAP COUNTER
                    2353
                    2354                NOP
4D2 0000000622993   2355                JMPC    IDLOOP4                           IF EQUAL, WE MAY NOT INCREMENT THE GAP COUNTER
                    2356
                    2357  INC.GAPC      INC.A   RE      RE      B                 INCREMENT THE LS BYTE OF THE GAP COUNTER
4D3 003939CC4CD42   2358                NEXT                    NCARRY
                    2359
                    2360                NOP
4D4 0000000622993   2361                JMPC    IDLOOP4                           RETURN TO IDLOOP IF WE DONT HAVE TO CARRY
                    2362
                    2363                INC.A   RF      RF      B                 INCREMENT THE MS BYTE OF THE GAP COUNTER
4D5 00003DEC4A992   2364                JMPU    IDLOOP4                           AND RETURN TO IDLE LOOP
                    2365
                    2366  NDATA.HERE    NOP                                       DATA NOT PRESENT
4D6 0000000622911   2367                CALLC   DEC.2BYTES                        SO DECREMENT THE GAP COUNTER
                    2368
                    2369                PASS.A  RD                                INSPECT INSTRUCTIONS WHERE TO EXIT
4D7 000001A442993   2370                JMPC    IDLOOP4         ALWAYS            RETURN TO IDLOOP IF 1/3 INCH GAP NOT YET FOUND
                    2371
                    2372                S.BCL.A LIT     R6      B     #40         TELL IDLOOP "DON'T USE GAPFINDER"
4D8 10001A8ED4E06   2373                INDEXC  EXIT                              INDEX INTO THE EXIT TABLE
                    2374
                    2375  * THE INDEXED EXIT TABLE MUST BEGIN AT ADDRESS #XX0
     04E0           2376                ORG     *+10 AND #7F0
                    2377
                    2378  EXIT          NOP                                       AN UNEXPECTED I.R.G. HAS BEEN FOUND
4E0 0000000624ED2   2379                JMPU    ERROR.A
                    2380
                    2381                NOP                                       THE GAP WE ARE LOOKING FOR HAS BEEN FOUND
4E1 0000000625002   2382                JMPU    STOP.NOW
                    2383
                    2384                S.BCL.A LIT     R5      B     #10         INSTRUCT IDLOOP TO SERVICE READ ROUINTES ALWAYS
4E2 0400168ED3982   2385                JMPU    LOOK.SZR                          FILE MARK SEARCH CONTINUES : LOOK FOR SYNC ZONE
                    2386
                    2387                S.IOR.A LIT     R7      B     #14         I.R.G. CAME BEFORE WE FOUND THE CORRECT
4E3 0500  0D5002    2388                JMPU    STOP.NOW                          CRAMMED BYTE COUNT BLOCKETTE : FLAG THE ERR
```

```
                    2281  ********************************************************************************
                    2282  * GAP-FINDER PERFORMS A NUMBER OF FUNCTIONS. DURING A WRITE, AS SOON AS THE      *
                    2283  * SYNC ZONE IS IDENTIFIED, GAPFINDER MONITORS DROPOUTS AND ABORTS THE WRITE      *
                    2284  * IMMEDIATELY IF A DROPOUT OCCURS. DURING READS, GAPFINDER LOOKS FOR 1/3         *
                    2285  * INCH OF I.R.G. WHEN IT FINDS ONE, IT TURNS ITSELF OFF (TELLS IDLOOP NOT        *
                    2286  * TO USE GAPFINDER ANY MORE), IT TURNS THE READ ROUTINES ON ALWAYS (THE          *
                    2287  * READ ROUTINES ARE SERVICED ONCE EACH PASS THRU THE IDLE LOOP RATHER THAN       *
                    2288  * WHEN RDR SAYS TO SERVICE THEM), AND IT EXITS ACCORDING TO WHERE IT IS TOLD     *
                    2289  * TO IN RD. THIS FEATURE HAS MANY USES. DURING A FILE MARK SEARCH, GAPFINDER     *
                    2290  * WILL LOCATE THE NEXT I.R.G. TO BEGIN READING THE NEXT RECORD TO SEE IF IT      *
                    2291  * IS A FILE MARK. DURING REVERSE TAPE MOTION, GAPFINDER PRIVIDES THE TIMING      *
                    2292  * FOR A REVERSE STOP DELAY TO QUARANTEE ALL THREE HEADS (READ, WRITE, ERASE)     *
                    2293  * STOP IN THE I.R.G. IF A RECORD IS NOT READABLE, GAPFINDER SPACES OVER TO THE   *
                    2294  * NEXT I.R.G. SO YOU CAN TRY TO IDENTIFY ITS REVERSE I.D. ZONE.                 *
                    2295  * ALSO, GAPFINDER DECREMENTS A GENERAL PURPOSE 2 BYTE COUNTER IN "RO,RC".       *
                    2296  ********************************************************************************
                    2297
                    2298  GAPFINDER   PASS.A   R6                          IS THIS A WRITE OR NOT ?
4C3 002D00C444C73   2299             JMPC    RTC.SET      NBIT5
                    2300
                    2301  NOT.RTC     NOP                                  RTC IS NOT SET SO
4C4 003F000622993   2302             JMPC    IDLOOP4      NRDDX           BACK TO IDLOOP IF NOT A WRITE
                    2303
                    2304  WRITE.      NOP                                  IT IS A WRITE SO
4C5 0000000622993   2305             JMPC    IDLOOP4                      BACK TO IDLOOP IF THERE WAS NO DROPOUT
                    2306
                    2307             NOP                                  THERE WAS A DROPOUT DURING A WRITE SO
4C6 0000000624EE2   2308             JMPU    ERROR.B                      GO FLAG THE ERROR
                    2309
                    2310
                    2311
                    2312
                    2313  RTC.SET     DEC.A   RC     RC    B               DECREMENT LS GENERAL PURPOSE COUNTER RO,RC
4C7 0019318CC4CD3   2314             JMPC    NOT.WRITE    NBORROW
                    2315
                    2316  WRITE..             CLRTC                        CLEAR RTC
4C8 003F026624CB3   2317             JMPC    NOBORROW     NRDDX           JUMP IF WE NEEDN'T DECREMENT MS BYTE TOO
                    2318
                    2319  BORROW      DEC.A   RO     RO    B               DECREMENT THE MS GENERAL PURPOSE COUNTER RO,RC
4C9 0000000CC2993   2320             JMPC    IDLOOP4                      TO IDLOOP IF NOT A DROPOUT
                    2321
                    2322             NOP                                  THERE WAS A DROPOUT SO
4CA 0000000624EE2   2323             JMPU    ERROR.B                      GO FLAG THE ERROR AND ABORT THE WRITE
                    2324
                    2325  NOBORROW    NOP                                  WE NEEDN'T DECREMENT RO TOO
4CB 0000000622993   2326             JMPC    IDLOOP4                      TO IDLOOP IF NO DROPOUT
                    2327
                    2328             NOP                                  THERE WAS A DROPOUT SO
4CC 0000000624EE2   2329             JMPU    ERROR.B                      GO FLAG THE ERROR AND ABORT THE WRITE
```

```
                    2221  ********************************************************************************
                    2222  * FINALLY, WE CHECK THE REVERSE I.D ZONE AND FRAMING CHARACTER AND TRAILING      *
                    2223  * SYNC ZONE. THIS FIRST ROUTINE IS CALLED WITH R8 = ID ZONE WHICH WE EXPECT      *
                    2224  * TO FIND. WHEN ALL IS DONE, GO TO STOP THIS READ OR WRITE.                      *
                    2225  * ALSO CONFIRM THE LAST BYTE WAS SENT TO THE PPU IF THIS WAS A READ.             *
                    2226  ********************************************************************************
                    2227
                    2228  R.R.ID        PASS.A   R5                                  INSPECT BIT "WE ARE SENDING BYTES TO THE PPU"
4B4 000A00A44489C   2229                LDZ      R.R.FRAME     BIT2                  PREPARE TO READ REVERSE FRAMING CHARACTER
                    2230
                    2231                S.AND.A  LIT    R5          #03               DOES I/O ROUTINE THINK ALL IS SENT ?
4B5 00F8168652831   2232                CALLC    LASTOVRFLW    NZERO                 CHECK THAT LAST BYTE IS SENT IF WE'RE SENDING
                    2233
                    2234                S.XOR.A  RDATA  R8                            COMPARE I.D. RED TO ONE EXPECTED
4B6 00182EC754B72   2235                NEXT                   ZERO
                    2236
                    2237                NOP
4B7 0000000622973   2238                JMPC     IDLOOP3                              TO IDLOOP IF I.D. OKAY
                    2239
                    2240                NOP                                           THE REVERSE I.D. IS WRONG
4B8 0000000624F42   2241                JMPU     ERROR.11                             SO GO FLAG THE ERROR
                    2242
                    2243  R.R.FRAME     INC.S    RDATA                                #FF + 1 SHOULD = 00
4B9 001802C47CBCC   2244                LDZ      CHK.T.SZ      ZERO                   POINT Z TO CHECK TRAILING SYNC ZONE
                    2245
                    2246                PASS.S   LIT    R9     B    #08               WE WILL CHECK 8+1 TRAILING SYNC ZONE BYTES
4BA 0200268C72973   2247                JMPC     IDLOOP3                              BACK TO IDLE LOOP IF REVERSE FRAMING IS OKAY
                    2248
                    2249                NOP                                           THE REVERSE FRAMING CHARACTER IS BAD
4BB 0000000624F42   2250                JMPU     ERROR.11                             SO GO FLAG THE ERROR
                    2251
                    2252  CHK.T.SZ      PASS.S   RDATA                                INSPECT SYNC ZONE RED FROM TAPE
4BC 001802C474BD2   2253                NEXT                   ZERO
                    2254
                    2255                DEC.A    R9     R9     B                     DECREMENT THE COUNT OF TRAILING SYNC ZONE BYTES
4BD 0019252CC4BF3   2256                JMPC     *+2           NBORROW               SKIP IF SYNC ZONE IS OKAY
                    2257
                    2258                NOP                                           TRAILING SYNC ZONE IS BAD
4BE 0000000624F52   2259                JMPU     ERROR.12                             SO GO RECORD THE ERROR
                    2260
                    2261                NOP                                           TRAILING SYNC ZONE IS OKAY
4BF 0000000622973   2262                JMPC     IDLOOP3                              SO TO IDLOOP IF LAST SYNC BYTE ISN'T NEXT
                    2263
                    2264                S.BCL.A  LIT    R6     B    #40               LAST SYNC BYTE IS NEXT SO TURN OFF GAPFINDER
4C0 10001A8ED4C2C   2265                LDZ      LAST.SBYTE                           POINT Z TO READ LAST SYNC BYTE
                    2266
                    2267                NOP
4C1 0000000622972   2268                JMPU     IDLOOP3                              AND GO TO IDLOOP TO WAIT FOR IT
                    2269
                    2270  LAST.SBYTE              RDATA                                LAST SYNC BYTE HAS ARRIVED SO CLEAR RDR
4C2 000002C625002   2271                JMPU     FOR.STOP                             AND GO STOP THE TAPE
                    2272
                    2273
                    2274  * ERROR ! THE CODE ABOVE ONLY CHECKS 9 OF THE LAST 10 SYNC BYTES.
                    2275  *         THIS IS BECAUSE THE 4000 STC CONTROLLER DOES NOT REALLY WRITE 80 TRAILING
                    2276  *         SYNC BYTES AND I DIDN'T WANT AN ERROR EVERY TIME I RED A 4000 RECORD
                    2277
                    2278  *                        --  GAP FINDER  EXT  --
```

```
                    2189  ********************************************************************************
                    2190  * TIME TO CHECK THE DATA CRC AND SEND THAT ONE LAST BYTE TO THE PPU             *
                    2191  ********************************************************************************
                    2192
                    2193   CHK.DCRCA            LIT    RDCON       #F0          TURN CRC CHECK OFF (PIPELINED OPERATION - REMEMBER!)
4AC 3C006E8624B1C   2194                 LDZ    CHK.DCRCB                       POINT Z TO READ SECOND HALF OF CRC
                    2195
                    2196                 PASS.A R5                              INSPECT BIT "FREE TO SEND DATA BYTE TO PPU ?"
4AD 000800A444AE2   2197                 NEXT                  BITO
                    2198
                    2199                 S.IOR.A LIT    R5     B    #01        SET BIT "SEND DATA BYTE TO PPU"
4AE 0040168DD2851   2200                 CALLC  OVRFLW                          FLAG AN OVER FLOW IF IT HAPPENED
                    2201
                    2202                        RDATA                           CLEAR RDR
4AF 000002C624802   2203                 NEXT
                    2204
                    2205                 PASS.A RB     RAM          HOLDOUT     SEND THAT ONE LAST BYTE TO OUTPUT BUFFER
4B0 14005D6442972   2206                 JMPU   IDLOOP3
                    2207
                    2208
                    2209
                    2210
                    2211   CHK.DCRCB            RDATA                           CLEAR RDR
4B1 003E02C624B4C   2212                 LDZ    R.R.ID         NCRCERR          POINT Z TO Read Reverse ID
                    2213
                    2214                 PASS.S LIT    RB     B    #B3         LOAD REVERSE DATA I.D FOR NEXT ROUTINE'S USE
4B2 2CC02E8C72973   2215                 JMPC   IDLOOP3                         TO IDLOOP IF NO CRC ERROR
                    2216
                    2217                 NOP                                    THERE IS A CRC ERROR
4B3 0000000624F22   2218                 JMPU   ERROR.F                         SO GO FLAG IT
```

```
                    2140  ************************************************************************************
                    2141  * THIS LOOP TO READ DATA BYTES AND SEND THEM TO THE PPU MUST UNSWAP THE            *
                    2142  * SRAMBLED BYTES, SEND THEM TO THE PPU, AND SET UP THE CRC CHECK DURING            *
                    2143  * THE SECOND TO LAST AND LAST PASS THROUGH THIS LOOP.                              *
                    2144  ************************************************************************************
                    2145
                    2146  R.DATA.B      PASS.A   R5                                  INSPECT BIT "FREE TO SEND DATA TO PPU ?"
49E 000800A4449F2   2147                NEXT                  BITO
                    2148
                    2149                S.IOR.A  LIT    R5     B     #01             SET BIT "SEND DATA BYTE TO PPU"
49F 0040168DD2851   2150                CALLC    OVRFLW                             FLAG OVER FLOW ERROR IF IT HAPPENED
                    2151
                    2152  R.DATA.A      DEC.A    R9     R9     B                    DECREMENT THE LS BYTE COUNT
4A0 0019252CC49EC   2153                LDZ      R.DATA.B      NBORROW             POINT Z TO THE ENTIRE READ DATA LOOP
                    2154
                    2155                PASS.A   RA                                  INSPECT THE MS BYTE COUNT FOR BC >> 0
4A1 0038014444A33   2156                JMPC     *+2           NZERO               SKIP IF LS COUNT DIDN'T ROLL OVER
                    2157
                    2158                DEC.A    RA     RA     B                    DECREMENT THE MS BYTECOUNT. SINCE LS COUNT NOW = #FF
4A2 0000294CC4A72   2159                JMPU     NOSWEAT                            WE NEEDN'T WORRY IF BYTE COUNT = 0 OR 1
                    2160
                    2161                S.AND.A  LIT    R9           #FE             SEE IF LS COUNT > 1
4A3 3FB8268654A73   2162                JMPC     NOSWEAT       NZERO               NO WORRY IF MS BYTE COUNT # #00
                    2163
                    2164                PASS.A   R9                                  INSPECT LS BYTE COUNT IN CASE BC = 0 OR 1
4A4 0008012444A73   2165                JMPC     NOSWEAT       BITO                NO WORRY IF LS BYTE COUNT > 1
                    2166
                    2167                LIT      RDCON        #F2                  BEGIN CRC CHECK SINCE BYTE COUNT = 1 OR 0
4A5 3C806E8624A73   2168                JMPC     *+2                                SKIP SINCE LS BYTE COUNT = 1
                    2169
                    2170                LIT      RDCON        #F3                  CHECK LAST 7 BITS OF CRC SINCE BYTE COUNT = 0
4A6 3CC06E8624ACC   2171                LDZ      CHK.DCRCA                          ALSO POINT Z TO CHECK DATA CRC
                    2172
                    2173  NOSWEAT       A.XOR.B  R6     R9                          EVEN OR ODD BYTE TO BE RED NOW ?...DO BYTE
4A7 000824C714A82   2174                NEXT                  BITO                SWAP DEPENDING ON WETHER EVEN OR ODD LENGTH RECORD
                    2175
                    2176                NOP
4A8 0000000624AA3   2177                JMPC     SWAP.EM                            SKIP IF WE DO HAVE TO SWAP BYTES AROUND
                    2178
                    2179  NSWAP                  RDATA  RAM           HOLDOUT      TRANSFER BYTE TO OUTPUT BUFFER FOR I/O ROUTINE
4A9 14005EC622972   2180                JMPU     IDLOOP3
                    2181
                    2182  SWAP.EM       PASS.A   RB     RAM          HOLDOUT      SEND BYTE RED 2 PASSES BACK TO PPU
4AA 14005D6444AB2   2183                NEXT
                    2184
                    2185                PASS.S   RDATA  RB     B                    STORE BYTE RED THIS TIME FOR LATER
4AB 00002ECC72972   2186                JMPU     IDLOOP3
```

```
                    2085  ********************************************************************************
                    2086  * HERE WE COPY THE BYTE COUNT RED OFF THE TAPE INTO RAM TO SEND TO THE          *
                    2087  * SYSTEM LATER, WE FLAG IF THIS IS AN EVEN OR ODD LENGTH RECORD, AND IF         *
                    2088  * IT'S A ONE BYTE RECORD, WE MUST BEGIN THE DATA CRC CHECK HERE.               *
                    2089  ********************************************************************************
                    2090
                    2091  CHK.WCCRCA          LIT     RDCON       #F0         TURN OFF THE CRC CHECK
48F 3C006E862495C   2092                LDZ     CHK.WCCRC8                  POINT Z TO WHERE TO CHECK THE REST OF THE CRC
                    2093
                    2094                        RDATA                       CLEAR RDR
490 000002C624912   2095                NEXT
                    2096
                    2097                PASS.A  R9      RAM         SECSTS4 LOAD LS BYTE COUNT INTO RAM
491 07885D2444922   2098                NEXT                BIT0
                    2099
                    2100                PASS.A  RA      RAM         SECSTS3 LOAD MS BYTE COUNT INTO RAM
492 07405D4444943   2101                JMPC    *+2                         SKIP IF BYTE COUNT IS ODD
                    2102
                    2103                S.BCL.A LIT     R6      B   #01     FLAG BYTE COUNT EVEN
493 00401A8ED2972   2104                JMPU    IDLOOP3
                    2105
                    2106                S.IOR.A LIT     R6      B   #01     FLAG BYTE COUNT ODD
494 00401A8DD2972   2107                JMPU    IDLOOP3
                    2108
                    2109
                    2110
                    2111
                    2112  CHK.WCCRCB    PASS.A  RA                          INSPECT MS BYTE COUNT FOR BC=1 OR 0
495 0038014444A0C   2113                LDZ     R.DATA.A    NZERO           POINT Z TO THE DATA READING ROUTINES
                    2114
                    2115                S.AND.A LIT     R9          #FE     SEE IF LS BYTE COUNT > #01
496 3FB8268654983   2116                JMPC    BC.GT.1     NZERO           THIS JUMP IS IF MS BYTE COUNT # #00
                    2117
                    2118                PASS.A  R9                          INSPECT LS BIT OF LS COUNT FOR 0 OR 1
497 0028012444983   2119                JMPC    BC.GT.1     NBIT0           THIS JUMP IS IF LS BYTE COUNT > #01
                    2120
                    2121                NOP
498 0000000624983   2122                JMPC    BC.EQ..0                    THIS JUMP IS IF BYTE COUNT = #00
                    2123
                    2124  BC.EQ.1               LIT     RDCON       #F2     MUST BEGIN DATA CRC CHECK NOW
499 3C806E8624982   2125                JMPU    BC.GT.1                     GO FINISH THE WORD COUNT CRC CHECK
                    2126
                    2127  BC.EQ..0      PASS.S  LIT     RB      B   #B3     LOAD REVERSE DATA I.D. FOR NEXT ROUTINE
49A 2CC02E8C7484C   2128                LDZ     R.R.ID                      POINT Z TO Read Reverse ID
                    2129
                    2130  BC.GT.1               RDATA                       CLEAR RDR
49B 001E02C6249C2   2131                NEXT                CRCERR
                    2132
                    2133                NOP                                 BYTE COUNT CRC INDICATES IF BYTE COUNT IS WRONG
49C 0000000624F13   2134                JMPC    ERROR.E                     IF SO, GO FLAG THE ERROR
                    2135
                    2136                S.IOR.A LIT     R6      B   #80     IF CRC OK, FLAG "WORD COUNT VALID"
490 20001A8DD2972   2137                JMPU    IDLOOP3                     AND GO READ THE DATA
```

```
                    2049  ********************************************************************************
                    2050  * HERE WE READ THE MFM WORD COUNT WHICH WE MUST MULTIPLY BY 2 TO TURN IT INTO    *
                    2051  * A BYTE COUNT WHICH THIS COMPUTER SYSTEM UNDERSTANDS.                           *
                    2052  ********************************************************************************
                    2053
                    2054   READ.LSWC    PASS.A   R6                              INSPECT BIT "FORWARD OR REVERSE ?"
486 002A00C44489C   2055                LDZ      READ.MSWC     NBIT2             POINT Z TO READ SECOND 8 BITS OF WORD COUNT
                    2056
                    2057                         LIT    RDCON        #F2          BEGIN THE WORD COUNT CRC CHECK
487 3C806E8624FB3   2058                JMPC     REV.STOP                         IF REVERSE, THERE'S NO WORD COUNT TO READ
                    2059
                    2060                PASS.S   RDATA  R9     BL                READ WORD COUNT AND BEGIN MULTIPLY BY 2
488 000026DC72972   2061                JMPU     IDLOOP3
                    2062
                    2063
                    2064
                    2065
                    2066   READ.MSWC            LIT    RDCON        #F3          CHECK 7 MORE BITS OF CRC
489 3CC06E86248FC   2067                LDZ      CHK.WCCRCA                       POINT Z TO WHERE TO CHECK THE WORD COUNT CRC
                    2068
                    2069                PASS.S   RDATA  RA     BL                READ REST OF WORD COUNT AND SHIFT LEFT ONCE
48A 00002ADC748B2   2070                NEXT
                    2071
                    2072                A.XOR.B  R9     RA                       MORE OF THE MULTIPLY BY 2
48B 00282927148C2   2073                NEXT                    NBIT0
                    2074
                    2075                PASS.S   LIT            Q    #01         LOAD Q=1 IN CASE WE NEED IT
48C 0040028072973   2076                JMPC     IDLOOP3                         TO IDLOOP IF MULTIPLY BY 2 IS DONE
                    2077
                    2078                A.XOR.Q  R9     R9     B                 COMPLIMENT LS BYTE COUNT'S LS BIT
48D 0000252F048E2   2079                NEXT
                    2080
                    2081                A.XOR.Q  RA     RA     B                 COMPLIMENT MS BYTE COUNT'S LS BIT
48E 0000294F02972   2082                JMPU     IDLOOP3                         WHICH DOES FINISH THE MULTIPLY BY 2
```

```
                    1995  ********************************************************************************
                    1996  * HERE WE READ AND CHECK THE MFM FRAMING CHARACTER (#FF) AND FETCH THE          *
                    1997  * NEXT BYTE OFF THE TAPE (THE I.D. ZONE) AND DECODE IT.                        *
                    1998  ********************************************************************************
                    1999
                    2000  R.MFM.ID      INC.S    RDATA                          #FF + 1 SHOULD EQUAL ZERO
476 001802C47C79C   2001               LDZ      DECODE.ID     ZERO             POINT Z TO WHERE TO DECODE THE I.D.
                    2002
                    2003               NOP
477 0000000622973   2004               JMPC     IDLOOP3                        BACK TO IDLOOP IF FRAMING CHARACTER WAS OKAY
                    2005
                    2006               NOP                                     THE FRAMING CHARACTER WAS WRONG SO
478 0000000624F02   2007  BAD.ID       JMPU     ERROR.D                        GO RECORD THE ERROR
                    2008
                    2009  DECODE.ID    PASS.S   RDATA  RB      B               STORE BYTE RED AWAY FOR TESTING
479 00002ECC747A2   2010               NEXT
                    2011
                    2012               S.XOR.A  LIT    RB          #CD         COMPARE BYTE RED TO A DATA I.D. ZONE
47A 33582E875486C   2013               LDZ      READ.LSWC     ZERO
                    2014
                    2015               PASS.A   R6                             INSPECT BIT "OKAY TO BE DATA I.D. ?"
47B 002B00C444833   2016               JMPC     DATA.ID       NBIT3            JUMP IF IT'S A DATA I.D.
                    2017
                    2018  NDATA.ID     S.XOR.A  LIT    RB          #B5         COMPARE BYTE RED TO A FILE MARK I.D. ZONE
47C 2D782E875484C   2019               LDZ      R.R.ID        NZERO            POINT Z TO Read Reverse ID
                    2020
                    2021               PASS.A   R6                             INSPECT BIT "OK TO BE FILE MARK ?"
47D 000B00C444783   2022               JMPC     BAD.ID        BIT3             IF NEITHER FM NOR DATA ID, ITS BAD
                    2023
                    2024  FM.FOUND     S.IOR.A  LIT    R7     B    #80         RECORD FILE MARK BEHIND ME
47E 20201E8DD4803   2025               JMPC     FM.OK         NEVER
                    2026
                    2027               PASS.A   R6                             INSPECT BIT "IS THIS A WRITE ?"
47F 000D00C444802   2028               NEXT                   BIT5             CUZ IF IT IS, WE WROTE THE WRONG I.D.
                    2029
                    2030  FM.OK        PASS.A   R6                             INSPECT BIT "FORWARD OR REVERSE ?"
480 000A00C444783   2031               JMPC     BAD.ID        BIT2             JUMP IF WRONG I.D. GOT WRITTEN
                    2032
                    2033               PASS.S   LIT    RB     B    #AD         GET READY TO READ REVERSE FILE MARK
481 28402E8C72973   2034               JMPC     IDLOOP3                        READ REVERSE FM IF TAPE MOTION IS FORWARD
                    2035
                    2036               NOP                                     TAPE MOTION IS REVERSE SO
482 0000000624F82   2037               JMPU     REV.STOP                       GO DO A REVERSE STOP
                    2038
                    2039  DATA.ID      PASS.A   R6                             INSPECT BIT "WRITE OR NOT WRITE ?"
483 030D00C442973   2040               JMPC     IDLOOP3       BIT5             TO IDLOOP IF OK TO BE DATA  RECORD
                    2041
                    2042               PASS.S   LIT    RD     B    #02         TELL GAPFINDER TO CONTINUE FILE MARK SEARCH
484 0080368C74783   2043               JMPC     BAD.ID                         ERROR IF FM WRITTEN AND DATA RED
                    2044
                    2045  FM.SEARCH             LIT    RDCON       #00         IDLE THE READ LOGIC WHILE FINDING NEXT I.R.G.
485 00006E8622972   2046               JMPU     IDLOOP3                        AND GO LET GAPFINDER FIND THE NEXT I.R.G.
```

```
                    1937  ********************************************************************************
                    1938  * HERE WE READ THE REVERSE HEADER ID ZONE FOR GCR AND THEN THE TRAILING          *
                    1939  * SYNC ZONE. GCR.R.ID IS CALLED WITH RB = ID ZONE WHICH WE EXPECT TO FIND.       *
                    1940  * WHEN ALL IS DONE, STOP THE TAPE.                                              *
                    1941  ********************************************************************************
                    1942
                    1943   GCR.R.ID             RDSTS                          CHECK FOR ILLEGAL GCR (WHICH IS RIGHT)
46A 002A02E6246EC   1944                LDZ     R.LASTSYNC    NBIT2            PREPARE TO CHECK LAST SYNC ZONE
                    1945
                    1946                S.XOR.A RDATA  RB                      COMPARE TRAILING ID TO EXPECTED ONE
46B 00182EC7546D3   1947                JMPC    *+2           ZERO             SKIP TO ILLEGAL TRAILING ID IF IT IS SO
                    1948
                    1949                PASS.S  LIT    R9     B    #06         WE WILL CHECK 6+1 TRAILING SYNC BYTES
46C 0180268C72973   1950                JMPC    IDLOOP3                        TO IDLOOP : CHECK LAST SYNC ZONE NEXT
                    1951
                    1952                NOP                                    THE TRAILING ID ZONE WAS BAD
46D 0000000624F42   1953                JMPU    ERROR.11                       SO GO RECORD THE ERROR
                    1954
                    1955
                    1956
                    1957
                    1958   R.LASTSYNC           RDSTS                          CHECK FOR ILLEGAL GCR (WHICH IS RIGHT)
46E 002A02E6246F2   1959                NEXT                   NBIT2
                    1960
                    1961                INC.S   RDATA                          CHECK THAT SYNC BYTE = #FF
46F 003802C47C743   1962                JMPC    BAD.SYNCZ      NZERO           JUMP IF SYNC BYTE IS NOT ILLEGAL GCR CODE
                    1963
                    1964                DEC.A   R9     R9      B               DECREMENT THE COUNT OF LAST SYNC BYTES
470 0019252CC4743   1965                JMPC    BAD.SYNCZ      NBORROW         JUMP IF SYNC BYTE # #FF
                    1966
                    1967                NOP
471 0018000622973   1968                JMPC    IDLOOP3        ZERO            TO IDLOOP IF LAST SYNC BYTE ISN'T NEXT
                    1969
                    1970                S.BCL.A LIT    R6     B    #40         LAST SYNC BYTE IS NEXT SO TURN OFF GAPFINDER
472 10001A8ED475C   1971                LDZ     LST.SBYTE                      PREPARE TO READ LAST SYNC ZONE BYTE
                    1972
                    1973                NOP
473 0000000622972   1974                JMPU    IDLOOP3                        TO IDLOOP
                    1975
                    1976   BAD.SYNCZ    NOP
474 0000000624F52   1977                JMPU    ERROR.12                       RECORD THE ERROR TRAILING SYNC ZONE BAD
                    1978
                    1979
                    1980   * ERROR !!! THIS CODE DOES NOT CHECK THE LAST 8 BITS OF THE SYNC ZONE BECAUSE THE PRESTRESS
                    1981   *           COULD NOT GUARANTEE THE LAST BIT WOULD BE ALL RIGHT !
                    1982
                    1983
                    1984   LST.SBYTE            RDATA                          LAST SYNC BYTE HAS ARRIVED SO CLEAR RDR
475 000002C625002   1985                JMPU    FOR.STOP                       AND DO A NORMAL STOP SINCE ALL WAS OK
                    1986
                    1987
                    1988  ********************************************************************************
                    1989  *                                                                              *
                    1990  *         NEXT ARE THE MFM READ ROUTINES                                       *
                    1991  *                                                                              *
                    1992  ********************************************************************************
```

```
                    1880  ********************************************************************************
                    1881  * HERE WE READ GCR'S REVERSE BYTE COUNT BLOCKETTE AND CHECK IF IT'S RIGHT       *
                    1882  ********************************************************************************
                    1883
                    1884
                    1885  R.R.LSBC.N            RDSTS                          READ (& INSPECT) REVERSE LS BYTE COUNT NOT
45C 002A02E6245FC   1886                LDZ     R.R.MSBC.N    NBIT2            PREPARE TO READ REVERSE MS BYTE COUNT NOT
                    1887
                    1888                PASS.S  RDATA  R9     B                LOAD LS BYTE COUNT NOT FOR CHECKING LATER
45D 000026CC72973   1889                JMPC    IDLOOP3                        TO IDLOOP IF REVERSE BYTE COUNT NOT WAS LEGAL GCR
                    1890
                    1891                NOP                                    REVERSE BYTE COUNT NOT WAS AN ILLEGAL GCR CODE
45E 0000000624F32   1892                JMPU    ERROR.10                       SO GO RECORD THE ERROR
                    1893
                    1894
                    1895
                    1896
                    1897  R.R.MSBC.N            RDSTS                          READ (& INSPECT) REVERSE MS BYTE COUNT NOT
45F 002A02E62462C   1898                LDZ     R.R.LSBC      NBIT2            PREPARE TO READ REVERSE LS BYTE COUNT
                    1899
                    1900                PASS.S  RDATA  RA     B                LOAD MS BYTE COUNT NOT FOR CHECKING LATER
460 00002ACC72973   1901                JMPC    IDLOOP3                        TO IDLOOP IF REVERSE BYTE COUNT NOT WAS LEGAL GCR
                    1902
                    1903                NOP                                    REVERSE BYTE COUNT NOT WAS AN ILLEGAL GCR CODE
461 0000000624F32   1904                JMPU    ERROR.10                       SO GO RECORD THE ERROR
                    1905
                    1906
                    1907
                    1908
                    1909  R.R.LSBC              RDSTS                          READ (& INSPECT) REVERSE LS BYTE COUNT
462 000A02E62466C   1910                LDZ     R.R.MSBC      BIT2             PREPARE TO READ REVERSE MS BYTE COUNT
                    1911
                    1912                S.XNOR.A RDATA  R9                     COMPARE LS BYTE COUNT WITH IT'S CHECK
463 001826C7D4653   1913                JMPC    *+2           ZERO             SKIP TO FLAG ILLEGAL GCR FOUND ERROR
                    1914
                    1915                NOP
464 0000000622973   1916                JMPC    IDLOOP3                        TO IDLOOP IF LS CHECK BYTE CHECKED OKAY
                    1917
                    1918                NOP                                    THE LS REVERSE BYTE COUNT HAD SOMETHING WRONG
465 0000000624F32   1919                JMPU    ERROR.10                       SO GO FLAG THE ERROR
                    1920
                    1921
                    1922
                    1923
                    1924  R.R.MSBC              RDSTS                          READ (& INSPECT) REVERSE MS BYTE COUNT
466 000A02E6246AC   1925                LDZ     GCR.R.ID      BIT2             PREPARE TO READ GCR REVERSE ID
                    1926
                    1927                S.XNOR.A RDATA  RA                     COMPARE MS BYTE COUNT WITH IT'S CHECK
467 00182AC7D4693   1928                JMPC    *+2           ZERO             SKIP TO FLAG ILLEGAL GCR FOUND ERROR
                    1929
                    1930                PASS.S  LIT    RB     B    #D5         LOAD EXPECTED REVERSE HEADER BYTE FOR LATER USE
468 35402E8C72973   1931                JMPC    IDLOOP3                        TO IDLOOP IF MS CHECK BYTE CHECKED OKAY
                    1932
                    1933                NOP                                    THE MS REVERSE BYTE COUNT HAD SOMETHING WRONG
469 0000000624F32   1934                JMPU    ERROR.10                       SO GO FLAG THE ERROR
```

```
                    1839  ********************************************************************************
                    1840  * WE ARRIVE HERE WHEN A BLOCKETTE HAS SUCCESSFULLY BEEN RED OR WHEN THE        *
                    1841  * ERROR OF AN UNSUCCESSFUL READ HAS BEEN RECORDED. WE MUST DECIDE WHAT         *
                    1842  * THE NEXT EXPECTED BLOCKETTE NUMBER IS AND IF IT IS AN XOR OR DATA            *
                    1843  * BLOCKETTE. IF THE LAST BLOCKETTE WAS THE LAST XOR BLOCKETTE, THEN WE         *
                    1844  * DECIDE TO READ THE REVERSE WORD COUNT IF ALL WAS OKAY, OR WE FLAG            *
                    1845  * A RECOVERABLE ERROR. IN EITHER CASE, WE MUST MAKE SURE THAT LAST BYTE        *
                    1846  * GOT SENT TO THE PPU.                                                         *
                    1847  ********************************************************************************
                    1848
                    1849   BLK.ENDED    DEC.A    RA     RA     B                 DECREMENT THE BLOCKETTE NUMBER (MS) COUNT
453 0019294CC45CC   1850   RECOVERED    LDZ      R.R.LSBC.N    NBORROW          PREPARE TO READ REVERSE BYTE COUNT NOT
                    1851
                    1852                PASS.A   R8                             INSPECT BIT "CHECKING DATA OR XOR BLOCKETTES ?"
454 000F010443F23   1853                JMPC     FINDGCRSYNC   BIT7              NOT AT AN END SO GO FIND NEXT SYNC ZONE
                    1854
                    1855                S.XOR.A  LIT    R8     B     #80         COMPLIMENT BIT "CHECKING XOR BLOCKETTES"
455 2018228F54573   1856                JMPC     DONE.BLKS     ZERO             JUMP IF WE HAVE RED ALL THE DATA AND XOR BLOCKETTES
                    1857
                    1858                PASS.S   LIT    RA     B     #01         LOAD EXPECTED XOR BLOCKETTE NUMBER = #01
456 00402A8C73F22   1859                JMPU     FINDGCRSYNC                    AND GO FIND IT.
                    1860
                    1861
                    1862
                    1863
                    1864   DONE.BLKS    PASS.A   R5                             INSPECT BIT "WE ARE SENDING BYTES TO THE PPU"
457 000A00A4445A3   1865                JMPC     NO.ERRORS     BIT2             JUMP IF THIS READ HAS HAD NO ERRORS SO FAR
                    1866
                    1867   ERRORS       S.AND.A  LIT    R5           #03        DOES I/O ROUTINE THINK ALL IS SENT ?
458 00F8168652831   1868                CALLC    LASTOVRFLW    NZERO            CHECK THAT LAST BYTE IS SENT IF WE'RE SENDING
                    1869
                    1870                NOP                                     BECAUSE AN ERROR OCCURRED, FLAG "RECOVERABLE READ"
459 0000000624F82   1871                JMPU     ERROR.15                       AND DON'T READ THE REVERSE BYTE COUNT
                    1872
                    1873   NO.ERRORS    S.AND.A  LIT    R5           #03        DOES I/O ROUTINE THINK ALL IS SENT ?
45A 00F8168652831   1874                CALLC    LASTOVRFLW    NZERO            CHECK THAT LAST BYTE IS SENT IF WE'RE SENDING
                    1875
                    1876                NOP                                     BECAUSE NO ERRORS OCCURED, GO TO THE IDLOOP TO READ
458 0000000622972   1877                JMPU     IDLOOP3                        THE REVERSE BYTE COUNT
```

```
                    1801  ********************************************************************************
                    1802  * NOW THAT THE BLOCKETTE ERROR HAS BEEN RECORDED, WE WILL WAIT TILL THIS       *
                    1803  * BLOCKETTE'S HEADER IS LATE (WHICH IT PROBABYY ALREADY IS) TO QUARANTEE THAT   *
                    1804  * WE DONT ACCIDENTALLY FIND TWO BLOCKETTES IN THE SPACE THAT ONLY ONE EXISTS.   *
                    1805  * WHEN THIS BLOCKETTE HEADER IS FOUND TO BE LATE, WE KNOW WE ARE POSITIONED     *
                    1806  * SOMEWHERE IN THE MIDDLE OF THE BLOCKETTE WE HAVE JUST RECORDED AN ERROR FOR,  *
                    1807  * SO WE CAN BEGIN LOOKING FOR THE NEXT BLOCKETTE AFTER ADDING A BLOCKETTE SIZE  *
                    1808  * PLUS 20% TO THE CURRENT BLOCKETTE HEADER LATE COUNT.                          *
                    1809  ********************************************************************************
                    1810
                    1811   WAIT.LATE    PASS.A   R0                              INSEPCT R0 : IS THIS BLOCKETTE'S ID LATE ?
44C 001A000444D2    1812                NEXT                  SIGN
                    1813
                    1814                S.ADD.A  LIT    RC          #42          CHECK IF FORTHCOMING ADDITION NEEDS TO CARRY
44D 1099328454503   1815                JMPC     NOTOOSOON      CARRY           JUMP IF WE NEEDN'T WAIT FOR THE ID
                    1816
                    1817
                    1818
                    1819
                    1820   TOOSOON      S.BCL.A  LIT    R5      B    #10        WE MUST WAIT TO KNOW WE ARE IN THE BLOCKETTE
44E 0400168ED44CC   1821                LDZ      WAIT.LATE                      WE ARE FLAGGING THE ERROR FOR BEFORE CONTINUING
                    1822
                    1823                         LIT    RDCON        #80        SO ABOVE, EXIT IDLOOP ALWAYS AND PREPARE TO WAIT
44F 20006E8622972   1824                JMPU     IDLOOP3                        FOR ID LATE, HERE IDLE READ LOGIC AND BEGIN THE WAIT
                    1825
                    1826
                    1827
                    1828
                    1829   NOTOOSOON    S.ADD.A  LIT    RC      B    #42        ADVANCE ID LATE COUNT TO "NEXT ID LATE"
450 1080328C54523   1830                JMPC     *+2                            SKIP IF WE DIDN'T HAVE TO CARRY
                    1831
                    1832                S.ADD.A  LIT    R0      B    #01        ADD WITHOUT CARRY
451 0040028C54532   1833                JMPU     RECOVERED
                    1834
                    1835                S.AD1.A  LIT    R0      B    #01        ADD WITH CARRY
452 0040028C5C532   1836                JMPU     RECOVERED
```

```
                    1742  ********************************************************************************
                    1743  * WE ARRIVE HERE KNOWING IF IT IS AN EVEN OR ODD, DATA OR XOR ERROR. ALL WE      *
                    1744  * HAVE TO DO IS SEE IF THE ERROR COUNT HAS GONE TO HIGH TO RECOVER THE RECORD    *
                    1745  * AND IF SO, FLAG A FATAL ERROR AND RECORD THE ERROR IN THE QUINTARY STATUS.     *
                    1746  * IF NOT FATAL ERROR, THEN RECORD WHAT ERROR IT IS IN THE TERTIARY STATUS.       *
                    1747  ********************************************************************************
                    1748
                    1749  ODD.BLK       S.ADD.A  LIT     R8      8     #04          INCREMENT COUNT OF ODD BLOCKETTE ERRORS
43C 0100228C543D2   1750                NEXT
                    1751
                    1752                PASS.A   R8                                 INSPECT ODD BLOCKETTE ERROR COUNT
43D 002A0104443E2   1753                NEXT                      NBIT2
                    1754
                    1755                PASS.A   R6                                 INSPECT BIT "WRITE OR NOT ?"
43E 000D00C444473   1756                JMPC     FATAL.O          BIT5              IF SECOND ODD ERROR, IT'S FATAL
                    1757
                    1758                NOP
43F 0000000624473   1759                JMPC     FATAL.O                            ANY ERROR IS FATAL DURING A WRITE
                    1760
                    1761                PASS.Q           RAM          TRDSTS1       LOAD NATURE OF ERROR INTO ODD ERROR LOCATION
440 08405C0424412   1762                NEXT
                    1763
                    1764                PASS.A   RA      RAM          TRDSTS2       LOAD BLOCKETTE NUMBER INTO ODD ERROR LOCATION
441 08805D44444C2   1765                JMPU     WAIT.LATE                          AND GO WAIT FOR THIS BLOCKETTE ID TO BE LATE
                    1766
                    1767
                    1768  EVEN.BLK      INC.A    R8      R8       8                 INCREMENT THE COUNT OF EVEN BLOCKETTE ERRORS
442 0028210C4C432   1769                NEXT                      NBITO
                    1770
                    1771                PASS.A   R6                                 INSPECT BIT "WRITE OR NOT ?"
443 000D00C444493   1772                JMPC     FATAL.E          BIT5              IF SECOND EVEN ERROR, IT'S FATAL
                    1773
                    1774                NOP
444 0000000624493   1775                JMPC     FATAL.E                            ANY ERROR IS FATAL DURING A WRITE
                    1776
                    1777                PASS.Q           RAM          TRDSTS3       LOAD NATURE OF ERROR INTO EVEN ERROR LOCATION
445 08C05C0424462   1778                NEXT
                    1779
                    1780                PASS.A   RA      RAM          TRDSTS4       LOAD BLOCKETTE NUMBER INTO EVEN ERROR LOCATION
446 09005D44444C2   1781                JMPU     WAIT.LATE                          AND GO WAIT FOR THIS BLOCKETTE ID TO BE LATE
                    1782
                    1783
                    1784
                    1785  FATAL.O       PASS.Q           RAM          QNTSTS1       LOAD NATURE OF ERROR INTO FATAL ODD ERROR LOCATION
447 09C05C0424482   1786                NEXT
                    1787
                    1788                PASS.A   RA      RAM          QNTSTS2       LOAD BLOCKETTE NUMBER INTO FATAL ODD ERROR LOCATION
448 0A005D4444482   1789                JMPU     REC.LOST                           GO FLAG THAT THIS RECORD IS LOST
                    1790
                    1791  FATAL.E       PASS.Q           RAM          QNTSTS3       LOAD NATURE OF ERROR INTO FATAL EVEN ERROR LOCATION
449 0A405C04244A2   1792                NEXT
                    1793
                    1794                PASS.A   RA      RAM          QNTSTS4       LOAD BLOCKETTE NUMBER INTO FATAL EVEN ERROR LOCATION
44A 0A805D4444482   1795                JMPU     REC.LOST                           GO FLAG THAT THIS RECORD IS LOST
                    1796
                    1797  REC.LOST      NOP                                         TO MANY ERRORS IN THIS RECORD, IT IS LOST SO
44B 0000000624F92   1798                JMPU     ERROR.16                           GO FLAG THE ERRROR
```

```
                    1685  ********************************************************************************
                    1686  * WE ARRIVE HERE WHEN A BLOCKETTE HAS SOME ERROR DETECTED. FLAG WHICH          *
                    1687  * ERROR IT IS, FLAG SEND JUNK FILLER BYTES TO PPU IF ALL THE DATA IN THIS      *
                    1688  * BAD BLOCKETTE DIDN'T ALREADY GET SENT, ABORT OPERATION IF A WRITE, AND       *
                    1689  * FIGURE OUT IF IT'S A DATA/XOR, EVEN/ODD BLOCKETTE ERROR SO THE NEXT PAGE     *
                    1690  * CAN RECORD IT APPROPRIATELY.                                                 *
                    1691  ********************************************************************************
                    1692
                    1693   *    ERROR0                                    #00        CRAMMED BYTE COUNT : FIRST BLOCKETTE BAD
                    1694
                    1695   LOG.ERROR1   PASS.S   LIT          Q     #10        BLOCKETTE ID LATE
42C 0400028074352   1696                JMPU     SET.SEND
                    1697   LOG.ERROR2   PASS.S   LIT          Q     #20        BLOCKETTE ID LATE & OVERFLOW
42D 0800028074352   1698                JMPU     SET.SEND
                    1699   LOG.ERROR3   PASS.S   LIT          Q     #30        ID IS LEGAL GCR
42E 0C00028074352   1700                JMPU     SET.SEND
                    1701   LOG.ERROR4   PASS.S   LIT          Q     #40        UNKNOWN BLOCKETTE ID
42F 1000028074352   1702                JMPU     SET.SEND
                    1703   LOG.ERROR5   PASS.S   LIT          Q     #50        BLOCKETTE NUMBER IS ILLEGAL GCR
430 1400028074352   1704                JMPU     SET.SEND
                    1705   LOG.ERROR6   PASS.S   LIT          Q     #60        DATA IS ILLEGAL GCR
431 1800028074352   1706                JMPU     SET.SEND
                    1707   LOG.ERROR7   PASS.S   LIT          Q     #70        CRC FIRST BYTE IS ILLGAL GCR
432 1C00028074362   1708                JMPU     NO.SEND
                    1709   LOG.ERROR8   PASS.S   LIT          Q     #80        CRC SECOND BYTE IS ILLEGAL GCR
433 2000028074362   1710                JMPU     NO.SEND
                    1711   LOG.ERROR9   PASS.S   LIT          Q     #90        CRC IS WRONG
434 2400028074362   1712                JMPU     NO.SEND
                    1713
                    1714
                    1715
                    1716   SET.SEND     S.IOR.A  LIT    R8    B     #10        SET BIT "SEND JUNK FILLER BYTES TO PPU"
435 0400228DD4362   1717                NEXT
                    1718
                    1719   NO.SEND      PASS.A   R8                           INSPECT BIT "DATA OR XOR BLOCKETTE ?"
436 000F01044442C   1720                LDZ      EVEN.BLK    BIT7
                    1721
                    1722                PASS.A   RA                           INSPECT BIT "EVEN OR ODD BLOCKETTE ?"
437 0008014444393   1723                JMPC     XOR.BLK     BIT0
                    1724
                    1725
                    1726
                    1727   DATA.BLK     S.IOR.Q  LIT          Q     #01        FLAG DATA BLOCKETTE ERROR
438 00400281E43C7   1728                JMPZJ    ODD.BLK                      TO ODD.BLK IF SO, ELSE TO EVEN.BLK IF NOT.
                    1729
                    1730
                    1731
                    1732   XOR.BLK      S.IOR.Q  LIT          Q     #02        FLAG XOR BLOCKETTE ERROR
439 00800281E4383   1733                JMPC     *+2                          SKIP IF ODD BLOCKETTE ERROR
                    1734
                    1735                S.BCL.A  LIT    R8    B     #10        CLEAR BIT "SEND JUNK FILLER BYTES TO PPU"
43A 0400228ED4422   1736                JMPU     EVEN.BLK                     OFF TO RECORD EVEN BLOCKETTE ERROR
                    1737
                    1738                S.BCL.A  LIT    R8    B     #10        CLEAR BIT "SEND JUNK FILLER BYTES TO PPU"
43B 0400228ED43C2   1739                JMPU     ODD.BLK                      OFF TO RECORD ODD BLOCKETTE ERROR
```

```
                    1648  ************************************************************************************
                    1649  * CHECK THE CRC (CYCLIC REDUNDANCY CHARACTER) ON EACH BLOCKETTE.                    *
                    1650  * IF ITS OKAY, CHECK THAT THE EXPECTED BLOCKETTE NUMBER MATCHES THE ONE             *
                    1651  * WE JUST RED. IF ALL IS OKAY, WE HAVE SUCCESSFULLY RED THIS BLOCKETTE              *
                    1652  * AND ITS TIME TO GO READ THE NEXT BLOCKETTE.                                       *
                    1653  ************************************************************************************
                    1654
                    1655   CHK.CRC.A    ZERO     RDSTS  R9     B                  INSPECT FOR LEGAL GCR, ZERO BYTES SENT COUNT
423 000A26EE2426C   1656                LDZ      CHK.CRC.B     BIT2               PREPARE TO CHECK SECOND HALF OF CRC
                    1657
                    1658                         LIT    RDCON        #F0          TURN OFF THE CRC CHECK
424 3C006E8624323   1659                JMPC     LOG.ERROR7                      ATTEMPT RECOVERY FROM ILLEGAL GCR IN CRC
                    1660
                    1661                         RDATA                           CLEAR RDR
425 000002C622972   1662                JMPU     IDLOOP3
                    1663
                    1664
                    1665
                    1666   CHK.CRC.B             RDSTS                           INSPECT THE READ STATUS FOR LEGAL GCR
426 000A02E624272   1667                NEXT                   BIT2
                    1668
                    1669                         RDATA                           CLEAR RDR
427 001E02C624333   1670                JMPC     LOG.ERROR8    CRCERR            ATTEMPT RECOVERY FROM ILLEGAL GCR IN CRC
                    1671
                    1672                S.XOR.A  RAM    RA          BLK.NUMBER   COMPARE EXPECTED BLOCKETTE NUMBER TO RED ONE IN RAM
428 16182AA754343   1673                JMPC     LOG.ERROR9    ZERO              ATTEMPT RECOVERY FROM CRC ERROR
                    1674
                    1675                PASS.S   LIT    RO     B    #00          ZERO MS HEADER LATE COUNT IN RO,RC
429 0000028C742B3   1676                JMPC     *+2                             SKIP IF BLOCKETTE NUMBERS MATCH
                    1677
                    1678                NOP                                      THE EXPECTED BLOCKETTE DOESN'T MATCH THE RED ONE SO
42A 0000000624F62   1679                JMPU     ERROR.13                        GO FLAG ERROR "BLOCKETTE COUNT LOST"
                    1680
                    1681                PASS.S   LIT    RC     B    #0D          LOAD LS ID HEADER LATE COUNT IN RO,RC
42B 0340328C74532   1682                JMPU     BLK.ENDED                       AND GO PREPARE TO READ NEXT BLOCKETTE
```

```
                    1589  ********************************************************************************
                    1590  * THIS ROUTINE TAKES THE BYTES IN THE DATA AND XOR BLOCKETTES AND PUTS THEM      *
                    1591  * WHERE THEY BELONG - SENT TO PPU, OR THROWN AWAY IF THEY ARE THE FILLER         *
                    1592  * BYTES IN THE LAST DATA BLOCKETTE, OR STORES THEM IN RAM IF THEY ARE FROM       *
                    1593  * THE XOR BLOCKETTE. THIS ALSO MUST FORWARN THE READ LOGIC THAT A CRC CHECK IS   *
                    1594  * COMING UP NEXT.                                                               *
                    1595  ********************************************************************************
                    1596
                    1597   READ.DATA    S.ADD.A  LIT    R9          #02          INSPECT BYTE COUNT FOR 2nd TO LAST BYTE
413 0089268454142   1598                NEXT                 NCARRY
                    1599
                    1600                S.XOR.A  LIT    R9          #FF          IS THIS THE LAST BYTE IN THIS BLOCKETTE ?
414 3FD8268754163   1601                JMPC     *+2         ZERO             SKIP IF NOT 2nd TO LAST BYTE
                    1602
                    1603                         LIT    RDCON       #F2          SET BIT "CHECK CRC" IN READ CONTROL REGISTER
415 3C806E8624173   1604                JMPC     *+2                              SKIP IF LAST BYTE IS BEING RED NOW
                    1605
                    1606                         RDSTS                            INSPECT READ STATUS FOR LEGAL GCR
416 000A02E624182   1607                JMPU     *+2         BIT2                 SKIP SINCE THIS ISN'T THE LAST BYTE TO READ
                    1608
                    1609                         RDSTS                            INSPECT READ STATUS FOR LEGAL GCR
417 000A02E62423C   1610                LDZ      CHK.CRC.A   BIT2                 PREPARE TO CHECK CRC NEXT
                    1611
                    1612                PASS.A   R8                               INSPECT BIT "READING DATA OR XOR BLOCKETTE ?"
418 002F010444313   1613                JMPC     LOG.ERROR6  NBIT7                ATTEMPT RECOVERY FROM ILLEGAL GCR
                    1614
                    1615                PASS.A   R5                               INSPECT BIT "FREE TO SEND BYTE TO PPU ?"
419 000800A4441D3   1616                JMPC     DATA        BIT0
                    1617
                    1618  >>>>>>>>>>>>>>>>>>>
                    1619  >XOR          PASS.A   RA      SADRH                    LOAD MS POINTER INTO RAM TO STORE XOR BLOCKETTE
41A 0000594444182   1620  >             NEXT
                    1621  >
                    1622  >             PASS.A   R9      SADRL                    LOAD LS POINTER INTO RAM TO STORE XOR BLOCKETTE
41B 00004D24441C2   1623  >             NEXT
                    1624  >
                    1625  >                      RDATA   SCR                      LOAD THE XOR BLOCKETTE BYTE INTO RAM
41C 0000DEC624202   1626  >             JMPU     DONE                             AND GO DECREMENT THE BYTES RED COUNT
                    1627  >>>>>>>>>>>>>>>>>>>
                    1628  >DATA                  RDATA   RAM         HOLDOUT      LOAD BYTE RED TO BE SENT TO PPU
41D 14005EC622851   1629  >             CALLC    OVRFLW                           FLAG OVERFLOW ERROR IF LAST BYTE NOT SENT YET
                    1630  >
                    1631  >             PASS.A   RA                               INSPECT MS COUNT TO SEE IF THIS IS LAST BLOCKETTE
41E 00380144441F2   1632  >             NEXT                 NZERO               CUZ IF IT'S NOT, ITS GOT 256 BYTES IN IT
                    1633  >
                    1634  >             S.SUB.A  RAM     R9          DONECOUNTR   COMPARE # BYTES RED TO "LS BYTE COUNT - 1"
41F 168926A55C223   1635  >             JMPC     SEND2       BORROW           SO WE DON'T SEND PPU THE FILLER BYTES
                    1636  >
                    1637  >DONE         INC.A    R9      R9      B                INCREMENT THE COUNT OF BYTES ALREADY RED
420 0000252C4A973   1638  >             JMPC     IDLOOP3                          TO IDLOOP IF WE DON'T SEND FILLER BYTES
                    1639  >
                    1640  >SEND1        S.IOR.A  LIT     R5      B       #01      SET BIT "SEND BYTE TO PPU"
421 0040168DD2972   1641  >             JMPU     IDLOOP3                          AND BACK TO THE IDLOOP
                    1642  >
                    1643  >SEND2        S.IOR.A  LIT     R5      B       #01      SET BIT "SEND BYTE TO PPU"
422 0040168DD4202   1644  >             JMPU     DONE                             AND GO DECREMENT THE BYTE COUNT RED
                    1645  >>>>>>>>>>>>>>>>>>>
```

```
                    1549  ********************************************************************************
                    1550  * READ THE BLOCKETTE HEADER AND MAKE SURE IT IS THE EXPECTED ONE                *
                    1551  * (DATA OR XOR BLOCKETTE HEADER). THEN READ AND STORE AWAY THE BLOCKETTE        *
                    1552  * NUMBER FOR CHECKING WHEN THE CRC HAS BEEN OKAY'ED.                            *
                    1553  ********************************************************************************
                    1554
                    1555   HEADERHERE   ZERO     RDSTS  R9     B                INSPECT THE READ STATUS, ZERO BYTES RED COUNT IN R9
409 002A26EE2410C   1556                LDZ      BLOCK.NUM     NBIT2            PREPARE TO READ THE BLOCKETTE NUMBER
                    1557
                    1558                PASS.A   R8                             INSPECT FLAG "EXPECTING DATA OR XOR BLOCK"
40A 000F0104442E3   1559                JMPC     LOG.ERROR3    BIT7             ATTEMPT RECOVERY SINCE HEADER IS LEGAL GCR
                    1560
                    1561                PASS.S   RDATA  RB     B                LOAD THE HEADER RED FOR CHECKING BELOW
40B 00002ECC740D3   1562                JMPC     *+2                            SKIP IF EXPECTING XOR HEADER
                    1563
                    1564                S.XOR.A  LIT    RB          #88          COMPARE BYTE RED TO A DATA HEADER ID
40C 2EF82E87540E2   1565                JMPU     *+2           NZERO
                    1566
                    1567                S.XOR.A  LIT    RB          #CB          COMPARE BYTE RED TO AN XOR HEADER ID
40D 32F82E87540E2   1568                JMPU     *+1           NZERO
                    1569
                    1570                S.IOR.A  LIT    R5     B    #10          SET BIT "EXIT IDLOOP ON RDR"
40E 0400168DD42F3   1571                JMPC     LOG.ERROR4                     ATTEMPT RECOVERY FROM UNKNOWN HEADER ID
                    1572
                    1573                NOP
40F 0000000622972   1574                JMPU     IDLOOP3                        HEADER IS OKAY SO READ BLOCKETTE NUMBER NEXT
                    1575
                    1576
                    1577
                    1578
                    1579   BLOCK.NUM            RDSTS                           INSPECT READ STATUS FOR LEGAL GCR CODE
410 000A02E62413C   1580                LDZ      READ.DATA     BIT2             PREPARE TO READ THE DATA IN THIS BLOCKETTE
                    1581
                    1582                         RDATA  RAM         BLK.NUMBER  LOAD THIS BLOCKETTE NUMBER INTO RAM FOR LATER CHECK
411 16005EC624303   1583                JMPC     LOG.ERROR5                     ATTEMPT RECOVERY FROM ILLGAL GCR BLOCKETTE NUMBER
                    1584
                    1585                NOP                                     THE BLOCKETTE NUMBER IS LEGAL GCR SO
412 0000000622972   1586                JMPU     IDLOOP3                        BACK TO THE IDLOOP TO WAIT FOR THE DATA
```

```
                    1489  ********************************************************************************
                    1490  * WE SEND UP TO 256 BYTES UNLESS THE BAD BLOCKETTE WAS THE LAST NOT FULL ONE     *
                    1491  * IN WHICH CASE WE SEND ONLY "HOW FULL IT IS" BYTES. THE BLOCKETTE COUNT HAS     *
                    1492  * ALREADY BEEN INCREMENTED, SO IF IT SAYS THIS IS AN XOR BLOCKETTE, WE KNOW      *
                    1493  * THE ERROR WAS IN THE LAST DATA BLOCKETTE. IF THE ERROR WAS IN AN XOR BLOCKETTE,*
                    1494  * THEN THE BIT "SEND JUNK BYTES" WILL NOT GET SET AND WE WILL NEVER GET HERE.    *
                    1495  ********************************************************************************
                    1496
                    1497   SENDJUNK      PASS.A   R0                                  INSPECT R0 FOR HEADER ID LATE
3FB 001A000444033   1498                 JMPC     HEAD2SOON     SIGN
                    1499
                    1500                 PASS.A   R5                                  INSPECT BIT "FREE TO SEND BYTES TO PPU ?"
3FC 000800A444063   1501                 JMPC     HEAD2LATE     BIT0
                    1502
                    1503                 PASS.A   R8                                  INSPECT R8 FOR XOR-WAS IT THE LAST DATA BLOCKETTE ?
3FD 000F010442973   1504                 JMPC     IDLOOP3       BIT7                  TO IDLOOP IF TOO SOON TO SEND A BYTE TO PPU
                    1505
                    1506                 S.IOR.A  LIT    R5     B     #01             SET BIT "SEND A BYTE TO THE PPU"
3FE 0040168DD4003   1507                 JMPC     LAST                                 JUMP IF IT WAS THE LAST DATA BLOCKETTE
                    1508
                    1509   NOTLAST       S.SUB.A  LIT    R9           #FF             HAVE WE SENT ALL 256 BYTES ?
3FF 3FF826855C012   1510                 JMPU     *+2           NZERO
                    1511
                    1512   LAST          S.SUB.A  RAM    R9           DONECOUNTR      HAVE WE EMPTIED THE LAST DATA BLOCKETTE ?
400 168826A55C012   1513                 JMPU     *+1           NZERO
                    1514
                    1515                 INC.A    R9     R9     B                     INCREMENT THE COUNT OF BYTES SENT
401 0000252C4A973   1516                 JMPC     IDLOOP3                              TO IDLOOP IF NOT DONE SENDING BYTES
                    1517
                    1518                 S.BCL.A  LIT    R8     B     #10             CLEAR FLAG "SEND JUNK BYTES TO PPU"
402 0400228ED2972   1519                 JMPU     IDLOOP3                              AND TO IDLOOP TO AWAIT HEADER ID
                    1520
                    1521
                    1522
                    1523
                    1524   HEAD2SOON     NOP                                          HEADER ID CAME BEFORE ALL JUNK BYTES WERE SENT
403 0000000624042   1525                 NEXT                                         SO WAIT OUT THE CONDITION BIT
                    1526
                    1527                 S.BCL.A  LIT    R8     B     #10             CLEAR BIT "SEND JUNK FILLER BYTES"
404 0400228ED2851   1528                 CALLC    OVRFLW                              AND GO FLAG THE OVERFLOW
                    1529
                    1530                 NOP
405 0000000624092   1531                 JMPU     HEADERHERE                          AND GO READ THIS NEXT BLOCKETTE
                    1532
                    1533
                    1534
                    1535
                    1536   HEAD2LATE     NOP                                          HEADER ID IS LATE AND THE JUNK BYTES AREN'T SENT
406 0000000624072   1537                 NEXT                                         SO WAIT OUT THE CONDITION BIT
                    1538
                    1539                 S.BCL.A  LIT    R8     B     #10             CLEAR BIT "SEND JUNK FILLER BYTES"
407 0400228ED2851   1540                 CALLC    OVRFLW                              AND GO FLAG THE OVERFLOW
                    1541
                    1542                 NOP
408 0000000624202   1543                 JMPU     LOG.ERROR2                          AND ATTEMPT RECOVERY FROM HEADER ID LATE
                    1544
                    1545  * IF THE PLL LOOSES LOCK AND RUNS TOO SLOW, WE MAY FIND THE NEXT BLOCKETTE OR FIND THE NEXT BLOCKETTE
                    1546  * IS LATE BEFORE HAVING SENT ALL THE JUNK FILLER BYTES. THUS A TAPE ERROR CAN INDUCE AN OVERFLOW ERROR.
```

```
                    1446  ************************************************************************************
                    1447  * THIS SECTION INSTRUCTS THE READ LOGIC TO FIND A GCR SYNC ZONE. IF THE LAST     *
                    1448  * BLOCKETTE RED HAD AN ERROR SUCH THAT ALL 256 BYTES OF IT DIDN'T GET SENT,      *
                    1449  * THEN JUNK FILLER ZEROES WILL BE SENT WHILE WAITING FOR THE NEXT BLOCKETTE.     *
                    1450  * THIS WAY, THE OPERATING SYSTEM WILL GET AS MANY BYTES AS IT EXPECTS. WHEN      *
                    1451  * THE HARDWARE FINDS A SYNC ZONE, IT WILL LOOK FOR A HEADER ID AND SET RDR       *
                    1452  * TO TELL JS WHEN IT IS HERE.                                                    *
                    1453  ************************************************************************************
                    1454
                    1455   FINDGCRSYNC          LIT    RDCON       #E4          ASSERT BIT "FIND GCR SYNC ZONE"
3F2 39006E8623F6C   1456                LDZ     FINDHEADER
                    1457
                    1458                S.BCL.A LIT    R5      B    #10          INSTRUCT "EXIT IDLE LOOP ALWAYS"
3F3 0400168ED3F42   1459                NEXT
                    1460
                    1461                        RDATA                       CLEAR RDR IN CASE ITS SET AFTER ERROR RECOVERY
3F4 000002C623F52   1462                NEXT
                    1463
                    1464                        LIT    RDCON       #E0      CLEAR BIT "FIND GCR SYNC ZONE"
3F5 38006E8622972   1465                JMPU    IDLOOP3
                    1466
                    1467
                    1468
                    1469
                    1470   FINDHEADER   PASS.A  R8                          INSPECT BIT "JUNK BYTES TO BE SENT ?"
3F6 000C010443F72   1471                NEXT                   BIT4
                    1472
                    1473                ZERO           RAM          HOLDOUT  SEND ZEROES IF THEY'RE TO BE SENT
3F7 14045C0623FB3   1474                JMPC    SENDJUNK       RDR          SEE IF HEADER ID IS HERE YET
                    1475
                    1476
                    1477
                    1478
                    1479   N.SENDJUNK   PASS.A  R0                          INSPECT R0 FOR HEADER ID LATE
3F8 003A000444093   1480                JMPC    HEADERHERE     NSIGN
                    1481
                    1482                NOP
3F9 0000000622973   1483                JMPC    IDLOOP3                     TO IDLOOP IF HEADER ID NOT LATE
                    1484
                    1485                NOP                                 THE HEADER ID ZONE IS LATE SO
3FA 0000000624202   1486                JMPU    LOG.ERROR1                  ATTEMPT RECOVERY FROM ID LATE
```

```
                    1397  ********************************************************************************
                    1398  * HERE WE READ THE BYTE COUNT INVERTED, CHECK TO SEE IT MATCHES THE BYTE       *
                    1399  * COUNT, AND PROCEED TO READ THE DATA UNLESS THE BYTE COUNT = #0000 IN          *
                    1400  * WHICH CASE WE READ THE TRAILING BYTE COUNT BLOCKETTE NEXT.                    *
                    1401  ********************************************************************************
                    1402
                    1403   R.MSBC.NOT          RDSTS                              CHECK FOR LEGAL GCR
3E5 000A02E623E9C   1404                LDZ     R.LSBC.NOT   BIT2                PREPARE TO READ LS BYTE COUNT NOT (INVERTED)
                    1405
                    1406                S.XNOR.A RDATA  RA                         COMPARE BYTE TO INVERTED BYTE
3E6 00182AC7D3E83   1407                JMPC     *+2         ZERO                SKIP IF ILLEGAL GCR
                    1408
                    1409                PASS.A   RA     RAM       SECSTS3          STORE MS BYTE COUNT INTO RAM FOR PPU
3E7 07405D4442973   1410                JMPC     IDLOOP3                          TO IDLOOP IF THE MS BYTE COUNT CHECKED
                    1411
                    1412   BAD.BC       NOP                                       THE BYTE COUNT DID NOT CHECK
3E8 0000000624F12   1413                JMPU     ERROR.E                          SO GO RECORD THE ERROR
                    1414
                    1415
                    1416
                    1417
                    1418   R.LSBC.NOT   ZERO     RDSTS  RO     B                 CHECK FOR LEGAL GCR, ZERO MS ID LATE COUNTER RO,RC
3E9 000A02EE245CC   1419                LDZ      R.R.LSBC.N   BIT2                PREPARE TO READ REVERSE BYTE COUNT NOT (INVERTED)
                    1420
                    1421                S.XNOR.A RDATA  R9                         COMPARE BYTE TO INVERTED BYTE
3EA 003826C7D3E83   1422                JMPC     BAD.BC      NZERO               JUMP IF ILLEGAL GCR
                    1423
                    1424                PASS.A   R9     RAM       SECSTS4          STORE LS BYTE COUNT INTO RAM FOR PPU
3EB 07805D2443E83   1425                JMPC     BAD.BC                           JUMP IF THE LS BYTE COUNT DIDN'T CHECK
                    1426
                    1427                PASS.A   R6                               INSPECT BIT "FORWARD OR REVERSE ?"
3EC 002A00C443ED2   1428                NEXT                      NBIT2
                    1429
                    1430                S.IOR.A  LIT    R6     B     #80          SET FLAG "BYTE COUNT VALID"
3ED 20001A8DD4FB3   1431                JMPC     REV.STOP                         IF REVERSE, THIS READ IS FINISHED
                    1432
                    1433                DEC.A    R9     RAM       DONECOUNTR     LOAD "LS BYTE COUNT - 1"
3EE 16995D24C3EF2   1434                NEXT                      NBORROW        AND CHECK FOR LS BYTE COUNT = #00
                    1435
                    1436                PASS.S   LIT    R8     B     #00          CLEAR OUT ALL THE FLAGS IN R8
3EF 0020228C73F13   1437                JMPC     *+2            NEVER            SKIP IF LS BYTE COUNT # #00
                    1438
                    1439                DEC.A    RA     RA     B                 DECREMENT THE MS BYTE COUNT
3F0 0039294CC3F12   1440                NEXT                    BORROW
                    1441
                    1442                PASS.S   LIT    RC     B     #0D          LOAD LS HEADER ID LATE COUNT INTO RO,RC
3F1 0340328C72973   1443                JMPC     IDLOOP3                          IF BYTE COUNT = #0000, ELSE FIND GCR SYNC ZONE
```

```
                    1372  ********************************************************************************
                    1373  *              HERE WE READ THE GCR WORD COUNT AND STORE IT AWAY                *
                    1374  ********************************************************************************
                    1375
                    1376   READ.MSBC              RDSTS                          CHECK FOR LEGAL GCR
3DF 002A02E623E2C   1377                LDZ       READ.LSBC      NBIT2           PREPARE TO READ LS BYTE OF BYTE COUNT
                    1378
                    1379                PASS.S    RDATA  RA      B               STORE MS BYTE COUNT AWAY
3E0 00002ACC72973   1380                JMPC      IDLOOP3                        AND RETURN TO IDLOOP IF IT'S LEGAL GCR
                    1381
                    1382                NOP                                      IT'S ILLEGAL GCR
3E1 0000000624F12   1383                JMPU      ERROR.E                        SO GO RECORD THE ERROR
                    1384
                    1385
                    1386
                    1387
                    1388   READ.LSBC              RDSTS                          CHECK FOR LEGAL GCR
3E2 002A02E623E5C   1389                LDZ       R.MSBC.NOT     NBIT2           PREPARE TO READ MS BYTE COUNT NOT (INVERTED)
                    1390
                    1391                PASS.S    RDATA  R9      B               STORE LS BYTE COUNT AWAY
3E3 000026CC72973   1392                JMPC      IDLOOP3                        AND RETURN TO IDLOOP IF IT'S LEGAL GCR
                    1393
                    1394                NOP                                      IT'S ILLEGAL GCR
3E4 0000000624F12   1395                JMPU      ERROR.E                        SO GO RECORD THE ERROR
```

```
                    1318  ********************************************************************************
                    1319  * HERE IS THE CODE TO READ AND DECODE THE I.D. ZONE ON A GCR RECORD. IT          *
                    1320  * MIGHT BE A FILE MARK OR A DATA RECORD DURING A READ OR A FILE MARK SEARCH      *
                    1321  * OR A WRITE AND THE APPROPRIATE ACTION WILL BE TAKEN.                          *
                    1322  ********************************************************************************
                    1323
                    1324  R.GCR.ID            RDSTS                          INSPECT THE READ STATUS TO FIND ILLEGAL FORMAT
3D0 002A02E623D12   1325                NEXT                NBIT2          CUZ THE I.D. HEADER SHOULD BE AN ILLEGAL GCR CODE
                    1326
                    1327                PASS.S  RDATA  RB   B              LOAD THE I.D. ZONE INTO RB FOR LATER TESTING
3D1 00002ECC73DE3   1328                JMPC    ID.BAD                     IF NOT ILLEGAL CODE, IT'S AN ERROR
                    1329
                    1330                S.XOR.A LIT    RB       #AB        COMPARE THE ID ZONE TO THE BYTE COUNT HEADER ID
3D2 2AD82E8753DFC   1331                LDZ     READ.MSBC   ZERO           GET READY TO READ THE MS BYTE COUNT
                    1332
                    1333                PASS.A  R6                         INSPECT BIT "OK TO BE DATA RECORD ?"
3D3 002B00C443D83   1334                JMPC    ITS.DATA    NBIT3          JUMP IF IT'S A BYTE COUNT HEADER ID
                    1335
                    1336  N.DATA        S.XOR.A LIT    RB       #FB        COMPARE ID ZONE TO THE FILE MARK CHARACTER
3D4 3EF82E87546AC   1337                LDZ     GCR.R.ID    NZERO          PREPARE TO READ REVERSE FILE MARK ID ZONE
                    1338
                    1339                PASS.A  R6                         INSPECT BIT "OK TO BE FILE MARK ?"
3D5 000B00C443DE3   1340                JMPC    ID.BAD      BIT3           IF ALSO NOT FILE MARK ID, IT'S AN ERROR
                    1341
                    1342  ITS.A.FM      S.IOR.A LIT    R7   B    #80        RECORD "FILE MARK BEHIND ME"
3D6 20201E8DD3083   1343                JMPC    OK2BFM      NEVER          JUMP IF WE ARE WRITING OR SEARCHING FILE MARK
                    1344
                    1345                PASS.A  R6                         INSPECT BIT "WRITE ?"
3D7 000D00C443D82   1346                NEXT                BIT5           IF WRITE, ITS A WRITE DATA WITH FILE MARK FOUND
                    1347
                    1348  OK2BFM        PASS.A  R6                         INSPECT BIT "FORWARD OR REVERSE"
3D8 000A00C443DE3   1349                JMPC    ID.BAD      BIT2           JUMP IF WROTE DATA , FOUND FILE MARK
                    1350
                    1351                PASS.S  LIT    RB   B    #DF        LOAD REVERSE FILE MARK ID FOR LATER COMPARE
3D9 37C02E8C72973   1352                JMPC    IDLOOP3                    TO IDLOOP TO READ REVERSE ID ZONE OF FILE MARK NEXT
                    1353
                    1354                NOP                                LOOKS LIKE THIS IS REVERSE TAPE MOTION
3DA 0000000624F82   1355                JMPU    REV.STOP                   SO GO DO A REVERSE STOP AFTER THE FILE MARK
                    1356
                    1357  ITS.DATA      PASS.A  R6                         INSPECT BIT "WRITE OR NOT WRITE ?"
3DB 000D00C442973   1358                JMPC    IDLOOP3     BIT5           TO IDLOOP IF IT'S OKAY TO BE DATA
                    1359
                    1360                PASS.S  LIT    RD   B    #02        INSTRUCT : FIND GAP THEN LOOK FOR SYNC ZONE
3DC 0080368C73DE3   1361                JMPC    ID.BAD                     JUMP IF WE WROTE FILE MARK, FOUND DATA
                    1362
                    1363  SEARCH.FM           LIT    RDCON       #00        TURN OFF READ LOGIC
3DD 00006E8622972   1364                JMPU    IDLOOP3                    AND CONTINUE FILE MARK SEARCH
                    1365
                    1366
                    1367
                    1368  ID.BAD        NOP                                HERE WE ARE IF THE ID ZONE IS WRONG
3DE 0000000624F02   1369                JMPU    ERROR.D                    SO GO RECORD THE ERRROR
```

```
                    1270  ********************************************************************************
                    1271  * WHEN A GCR BYTE COUNT IS CRAMMED, THIS CODE MUST LOOK FOR THE BLOCKETTE WHOSE  *
                    1272  * BLOCKETTE NUMBER MATCHES THE SECOND BLOCKETTE IN THE EXPECTED RECORD. THE FIRST*
                    1273  * BLOCKETTE IS ASSUMED LOST AND THAT ERROR APPROPRIATELY FLAGGED. HERE I READ THE*
                    1274  * FOUND BLOCKETTE'S NUMBER AND DECIDE IF THIS IS THE RIGHT BLOCKETTE, IF I SHOULD*
                    1275  * KEEP LOOKING FOR THE RIGHT BLOCKETTE, OR IF I MISSED IT ALREADY.              *
                    1276  ********************************************************************************
                    1277
                    1278   CHK.BLKNUM   ZERO     RDSTS  R9     B                  CHECK FOR LEGAL GCR, ZERO BYTES RED COUNT IN R9
3C5 000A26EE2413C   1279                LDZ      READ.DATA     BIT2               PREPARE TO RECOVER THE DATA IN THIS BLOCKETTE
                    1280
                    1281                         RDATA  RAM          BLK.NUMBER   LOAD THIS BLOCKETTE NUMBER INTO RAM FOR LATER CHECK
3C6 16005EC623CD3   1282                JMPC     NOT.YET                         BLOCKETTE NOT FOUND YET IF ILLEGAL GCR
                    1283
                    1284                S.SUB.A  RAM    RA           BLK.NUMBER   CHECK IF WE HAVE PASSED THE BLOCKETTE WE WANT
3C7 16192AA55BC82   1285                NEXT                  NBORROW
                    1286
                    1287                S.XOR.A  RAM    RA           BLK.NUMBER   CHECK IF WE HAVE FOUND THE BLOCKETTE WE WANT
3C8 16382AA753CA3   1288                JMPC     *+2          NZERO              SKIP IF WE HAVEN'T PASSED THE BLOCKETTE WE WANT
                    1289
                    1290                NOP                                     WE RED A BLOCKETTE NUMBER PAST THE ONE WE WANT
3C9 0000000624F72   1291                JMPU     ERROR.14                       FLAG ERROR "CRAMMED BYTE COUNT BLOCKETTE NOT FOUND"
                    1292
                    1293                PASS.S   LIT    RO     B    #00         ZERO OUT MS ID LATE COUNTER R0,RC
3CA 0000028C73CD3   1294                JMPC     NOT.YET                        JUMP IF WE HAVEN'T FOUND THE BLOCKETTE NUMBER YET
                    1295
                    1296                PASS.S   LIT    RC     B    #06         BLOCKETTE FOUND : ID LATE COUNT = #0006 CUZ IT
3CB 0180328C73CC2   1297                NEXT                                    WOULD HAVE BEEN LATE IN MAX. 6 MORE BYTE-TIMES
                    1298
                    1299                PASS.S   LIT    RD     B    #00         INSTRUCT GAPFINDER EXIT = "UNEXPECTED I.R.G."
3CC 0000368C72972   1300                JMPU     IDLOOP3                        TO IDLOOP TO BEGIN RECOVERING THIS BLOCKETTE'S DATA
                    1301
                    1302
                    1303
                    1304
                    1305
                    1306
                    1307
                    1308   NOT.YET                LIT    RDCON        #E4         WE HAVEN'T FOUND THE CRAMMED BLOCKETTE YET SO
3CD 39006E862388C   1309                LDZ      WAIT.BLK                       PREPARE TO WAIT FOR THE NEXT BLOCKETTE
                    1310
                    1311                         RDATA                          CLEAR RDR
3CE 000002C623CF2   1312                NEXT
                    1313
                    1314                         LIT    RDCON        #E0         LET READ LOGIC FIND THE NEXT BLOCKETTE FOR US
3CF 38006E8622972   1315                JMPU     IDLOOP3                        AND TO IDLOOP TO WAIT FOR IT
```

```
                    1218  ********************************************************************************
                    1219  * HERE WE LOOK FOR A PARTICULAR GCR BLOCKETTE AS TOLD BY THE CRAMMED BYTE COUNT. *
                    1220  * WHEN RDR INDICATES A BLOCKETTE HEADER HAS BEEN FOUND, OR WHEN THE ID IS LATE,  *
                    1221  * THE IDLOOP BRANCHES HERE WHERE WE READ THE HEADER AND SEE IF IT MATCHES THE    *
                    1222  * HEADER WE ARE LOOKING FOR.                                                    *
                    1223  * IF SO, ON TO CHECK THE BLOCKETTE NUMBER. IF NOT, EITHER WE MISSED THE ONE WE   *
                    1224  * WANT (GO TO ERROR 14) OR IT'S NOT HERE YET (GO TO NOT.YET). IF WE CAN'T       *
                    1225  * RECOGNIZE IT, WE FIGURE ON "NOT.YET".                                         *
                    1226  ********************************************************************************
                    1227
                    1228   WAIT.BLK     PASS.S   LIT    RD     B    #03          GAPFINDER EXIT = "CAN'T FIND CORRECT BLOCKETTE"
3B8 00C0368C73C5C   1229                LDZ      CHK.BLKNUM                     PREPARE TO SEE IF THE BLOCKETTE NUMBER IS RIGHT
                    1230
                    1231                PASS.A   R8                             INSPECT R8 FOR CRAMMED BYTE COUNT = #0000
3B9 002E010443BA2   1232                NEXT                      NBIT6
                    1233
                    1234                ROSTS                                   CHECK  - IT SHOULD BE ILLEGAL GCR
3BA 002A02E623BC3   1235                JMPC     *+2              NBIT2         SKIP IF CRAMMED BYTE COUNT NOT = #0000
                    1236
                    1237                NOP                                     CRAMMED BYTE COUNT=0000 SO WE CAN'T FIND BLOCKETTE
3BB 0000000624F72   1238                JMPU     ERROR.14                       SO GO FLAG THE ERROR "CAN'T FIND CORRECT BLOCKETTE"
                    1239
                    1240                PASS.A   R8                             INSPECT R8 : LOOKING FOR DATA OR XOR ?
3BC 002F010443CD3   1241                JMPC     NOT.YET          NBIT7         JUMP IF BAD BLOCKETTE HEADER
                    1242
                    1243                PASS.S   RDATA  R8       B              LOAD RED ID BYTE FOR CHECKING BELOW
3BD 00002ECC73C13   1244                JMPC     WAIT.DATA                      JUMP IF EXPECTING DATA BLOCKETTE, ELSE WAIT.XOR
                    1245
                    1246
                    1247   WAIT.XOR     S.XOR.A  LIT    RB          #CB        WAITING FOR XOR SO COMPARE R8 TO XOR HEADER
3BE 32D82E8753BF2   1248                NEXT                      ZERO
                    1249
                    1250                NOP                                     FOUND EXPECTED XOR HEADER SO NEXT
3BF 0000000622973   1251                JMPC     IDLOOP3                        GO TO IDLOOP TO READ THE BLOCKETTE NUMBER
                    1252
                    1253                NOP                                     WE DID NOT FIND AN XOR HEADER SO
3C0 0000000623CD2   1254                JMPU     NOT.YET                        WAIT FOR THE NEXT BLOCKETTE
                    1255
                    1256
                    1257   WAIT.DATA    S.XOR.A  LIT    RB          #BB        WAITING FOR DATA SO COMPARE RB TO DATA HEADER
3C1 2ED82E8753C22   1258                NEXT                      ZERO
                    1259
                    1260                S.XOR.A  LIT    RB          #CB        SEE IF IT WAS AN XOR HEADER ID
3C2 32F82E8752973   1261                JMPC     IDLOOP3          NZERO         FOUND DATA SO NEXT CHECK BLOCKETTE NUMBER
                    1262
                    1263                NOP                                     IF NOT DATA AND NOT XOR HEADER WE MUST
3C3 0000000623CD3   1264                JMPC     NOT.YET                        WAIT FOR THE NEXT BLOCKETTE
                    1265
                    1266                NOP                                     IF EXPECTING DATA AND FOUND XOR, WE MISSED THE DATA
3C4 0000000624F72   1267                JMPU     ERROR.14                       SO FLAG ERROR "CRAMMED BLOCKETTE NUMBER NOT FOUND"
```

```
                    1179  ********************************************************************************
                    1180  * THE ID HAS BEEN FOUND. WE MUST TURN ON THE GAP FINDER, ARM THE IDLE LOOP      *
                    1181  * TO CALL THE READ ROUTINES ON "RDR", AND CHECK TO SEE IF THIS IS A READ        *
                    1182  * WITH THE GCR BYTE COUNT GIVEN.                                               *
                    1183  ********************************************************************************
                    1184
                    1185   ID.HERE      PASS.S   RAM    RF     B    MSTHIRDIN.   LOAD THE 1/3 INCH GAP FINDER
3B1 15003EAC73B22   1186                NEXT
                    1187
                    1188                PASS.S   RAM    RE     B    LSTHIRDIN.   COUNT FOR THE GAPFINDER ROUTINE
3B2 15403AAC73B32   1189                NEXT
                    1190
                    1191                PASS.S   LIT    RD     B    #00          INSTRUCT GAPFINDER TO EXIT TO "BIG DROPOUT"
3B3 0000368C73B42   1192                NEXT                                     ERROR IF ONE IS FOUND
                    1193
                    1194                         RAM                PRMSTS3      INSPECT BIT "MFM OR GCR ?"
3B4 05C802A623B52   1195                NEXT                  BIT0
                    1196
                    1197                PASS.A   R6                              INPSECT BIT "CRAMMED BYTE COUNT ?"
3B5 002900C443B73   1198                JMPC     *+2          NBIT1             SKIP IF GCR
                    1199
                    1200                S.IOR.A  LIT    R5     B    #10          SERVICE READ ROUTINES ON "RDR"
3B6 0400168DD4762   1201                JMPU     R.MFM.ID                        GO READ THE MFM I.D.
                    1202
                    1203                S.IOR.A  LIT    R5     B    #10          SERVICE READ ROUTINES ON "RDR"
3B7 0400168DD3D03   1204                JMPC     R.GCR.ID                        READ GCR ID, ELSE DO CRAMMED BYTE COUNT
                    1205
                    1206
                    1207
                    1208
                    1209
                    1210
                    1211
                    1212  ********************************************************************************
                    1213  * FOLLOWING ARE THE ROUTINES WHICH READ GCR, AS WELL AS RECOVER FROM THE        *
                    1214  * ERRORS. AFTER THAT WILL BE THE ROUTINES WHICH READ THE MFM.                   *
                    1215  ********************************************************************************
```

```
                    1119  ********************************************************************************
                    1120  * HERE WE WAIT FOR THE SYNC ZONE AND THE ID ZONE. THE SYNC ZONE LATE TIMER       *
                    1121  * HAS ALREADY BEEN LOADED. THE ID ZONE IS LATE IF TOO FAR PAST THE BEGINNING     *
                    1122  * OF THE SYNC ZONE. WE DONT BELIEVE IT REALLY IS A SYNC ZONE UNLESS IT LASTS     *
                    1123  * TILL THE ID ZONE IS FOUND OR TILL THE ID ZONE IS LATE.                         *
                    1124  * IF THE ID. ZONE IS LATE CHECK FOR GCR CRAMMED BYTE COUNT                       *
                    1125  ********************************************************************************
                    1126
                    1127  DATAHERE            LIT     RDCON       #E0         TURN ON THE READ LOGIC
3A1 38216E86239EC   1128                LDZ     LOOK.SZB    NRTC            PREPARE TO WAIT FOR THE ID ZONE
                    1129
                    1130                S.XOR.A LIT     RC          #FF         SEE IF ID ZONE LATE COUNTER HAS EXPIRED
3A2 3FF8328752973   1131                JMPC    IDLOOP3     NZERO           TO IDLOOP IF NOT TIME TO CHECK TIMER
                    1132
                    1133                DEC.A   RC      RC      B               DECREMENT THE ID ZONE LATE COUNTER
3A3 0000318CC3A83   1134                JMPC    TIME.SZ                     GO TIME OUT THE SYNC ZONE IF NOT ID LATE
                    1135
                    1136                S.IOR.A LIT     R6      B   #40         ID ZONE LATE. TELL IDLOOP TO USE GAPFINDER
3A4 10001A8DD3AB2   1137                JMPU    LATE.ID                     SO GO HANDLE THE ERROR
                    1138
                    1139
                    1140  N.DATAHERE            LIT     RDCON       #00         IDLE THE READ LOGIC
3A5 00216E86239BC   1141                LDZ     LOOK.SZA    NRTC            PREPARE TO RELOAD THE ID ZONE LATE COUNTERS
                    1142
                    1143                PASS.S  RAM                 #FF         INSPECT THE WRITE REPEAT FLAG IN RAM
3A6 3FD802A472973   1144                JMPC    IDLOOP3     ZERO            TO IDLOOP IF NOT TIME TO CHECK COUNTERS
                    1145
                    1146                PASS.S  DRSTS   RO      B               LOAD THE DRIVE STATUS FOR CHECKING OFFLINE
3A7 0000024C72881   1147                CALLC   CHKOFFLINE1                 CHECK OFFLINE IF NOT WRITE REPEAT
                    1148
                    1149
                    1150  TIME.SZ               CLRTC                       CLEAR RTC
3A8 00000266228F1   1151                CALLC   DEC.3BYTES                  GO DECREMENT THE SYNC ZONE LATE COUNTER
                    1152
                    1153                NOP
3A9 0000000622973   1154                JMPC    IDLOOP3                     TO IDLE LOOP IF SYNC ZONE NOT LATE
                    1155
                    1156                S.IOR.A LIT     R7      B   #60         FLAG "TAPE POSITION LOST" AND GO
3AA 18001E8DD4EC2   1157                JMPU    ERROR.9                     FLAG SYNC ZONE LATE ERROR
                    1158
                    1159
                    1160  LATE.ID       PASS.S  RAM     RF      B   MSTHIRDIN.  LOAD THE 1/3 INCH GAP FINDER
3AB 15003EAC73AC2   1161                NEXT
                    1162
                    1163                PASS.S  RAM     RE      B   LSTHIRDIN.  COUNT FOR THE GAPFINDER ROUTINE
3AC 15403AAC73AD2   1164                NEXT
                    1165
                    1166                PASS.A  R6                          INSPECT BIT "CRAMMED BYTE COUNT ?"
3AD 002900C443AE2   1167                NEXT                    NBIT1
                    1168
                    1169                PASS.S  LIT     RD      B   #00         INSTRUCT GAPFINDER TO EXIT TO "BIG DROPOUT"
3AE 0000368C74EF3   1170                JMPC    ERROR.C                     FLAG ID LATE ERROR IF NOT CRAMMED BYTE COUNT
                    1171
                    1172  BC.CRAMM      S.IOR.A LIT     R5      B   #10         SERVICE READ ROUTINES ON "RDR"
3AF 0400168DD3B8C   1173                LDZ     WAIT.BLK                    PREPARE TO WAIT FOR NEXT BLOCKETTE
                    1174
                    1175                PASS.S  LIT     RD      B   #03         GAPFINDER EXIT = "CORRECT BLOCKETTE NOT FOUND"
3B0 00C0368C72972   1176                JMPU    IDLOOP3                     TO IDLOOP TO WAIT FOR NEXT BLOCKETTE HEADER
```

```
                    1093  ********************************************************************************
                    1094  * HERE WE WAIT FOR THE SYNC ZONE AND THE ID ZONE. THE SYNC ZONE LATE TIMER      *
                    1095  * HAS ALREADY BEEN LOADED. THE ID ZONE IS LATE IF TOO FAR PAST THE BEGINNING    *
                    1096  * OF THE SYNC ZONE. WE DONT BELIEVE IT REALLY IS A SYNC ZONE UNLESS IT LASTS    *
                    1097  * TILL THE ID ZONE IS FOUND OR TILL THE ID ZONE IS LATE.                        *
                    1098  ********************************************************************************
                    1099
                    1100   LOOK.SZA             RAM                 PRMSTS3      INSPECT BIT "MFM OR GCR ?"
39B 05C802A6239C2   1101                NEXT                BIT0
                    1102
                    1103                PASS.S   LIT    RC      B    #08          LOAD THE GCR ID ZONE LATE COUNT
39C 02E0328C739F3   1104                JMPC     *+3            NEVER             SKIP PAST THE RDR CHECK IF GCR
                    1105
                    1106                PASS.S   LIT    RC      B    #0D          LOAD THE MFM ID ZONE LATE COUNT
39D 0360328C739F2   1107                JMPU     *+2            NEVER             SKIP PAST THE RDR CHECK SINCE IT'S MFM
                    1108
                    1109   LOOK.SZB     NOP                                      THIS IS THE RDR CHECK !
39E 00040006239F2   1110                JMPU     *+1            RDR               IS THE ID ZONE FOUND ?
                    1111
                    1112                S.IOR.A  LIT    R6      B    #40          TELL IDLOOP TO USE GAPFINDER
39F 103D1A8DD3813   1113                JMPC     ID.HERE        NRDD              JUMP IF THE ID ZONE HAS BEEN FOUND
                    1114
                    1115                S.BCL.A  LIT    R6      B    #40          TELL IDLOOP DON'T USE GAPFINDER
3A0 10001A8ED3A53   1116                JMPC     N.DATAHERE                       JUMP IF DATA GONE, ELSE TO DATAHERE
```

```
                    1076  ********************************************************************************
                    1077  * IF THIS IS NOT A WRITE, A SYNC ZONE COULD COME ANY TIME, DEPENDING ON HOW      *
                    1078  * MANY RE-TRIES WERE REQUIRED WHEN THIS WAS WRITTEN. WELL. TO PICK A NUMBER      *
                    1079  * OUT OF THE SKY, WE'LL CONSIDER THE SYNC ZONE LATE AFTER ABOUT 25 FEET.         *
                    1080  * ACTUALLY, THAT WILL BE NOMIANALLY 28 FEET IN GCR, 35 FEET IN MFM.             *
                    1081  ********************************************************************************
                    1082
                    1083   LOOK.SZR     PASS.S   LIT    RF     B    #05          LOAD MS BYTE OF SZL COUNT >= 25 FEET
398 01403E8C739BC   1084                LDZ      LOOK.SZA                         POINT Z TO SEARCH FOR SYNC ZONE
                    1085
                    1086                PASS.S   LIT    RE     B    #37          LOAD MID. BYTE OF SZL COUNT >= 25 FEET
399 0DC03A8C739A2   1087                NEXT
                    1088
                    1089                PASS.S   LIT    RD     B    #79          LOAD LS BYTE OF SZL COUNT >= 25 FEET
39A 1E40368C73982   1090                JMPU     LOOK.SZA                        GO BEGIN SYNC ZONE SEARCH
```

```
                    1016  ********************************************************************************
                    1017  * DELAY BEFORE WRITE IS 1/3 INCH OR 1 INCH IF WE JUST PASSED THE LOAD POINT      *
                    1018  * THE SYNC ZONE IS LATE 1/3 INCH AFTER THE DELAY BEFORE WRITE.                   *
                    1019  ********************************************************************************
                    1020
                    1021  LOOK.SZW            RAM                 PRMSTS3     INSPECT BIT "GCR OR MFM ?"
386 05E802A62398C   1022                LDZ     LOOK.SZA    NBIT0           Z POINTS TO ROUTINE WHICH FINDS SYNC ZONE
                    1023
                    1024                PASS.A  R0                          INSPECT DRIVE STATUS IN R0 TO SEE IF NEAR LOAD POINT
387 002A000443903   1025                JMPC    MFM.WD      NBIT2           GO TO MFM WRITE DELAY = 1/3 INCH
                    1026
                    1027  GCR.WD        PASS.S  LIT    R2   B    #01        LOAD MS COUNT OF GCR WRITE DELAY 1/3 INCH
388 00400A8C738C3   1028                JMPC    LP.GCR.WD                   IF NEAR LOAD POINT, LOAD 1 INCH WRITE DELAY
                    1029
                    1030                PASS.S  LIT    R1   B    #59        LOAD LS COUNT OF GCR WRITE DELAY 1/3 INCH
389 1640068C738A2   1031                NEXT
                    1032
                    1033                PASS.S  LIT    RE   B    #02        LOAD MIDDLE BYTE OF GCR SYNC ZONE LATE
38A 00803A8C738B2   1034                NEXT
                    1035
                    1036                PASS.S  LIT    RD   B    #C4        LOAD LS BYTE OF GCR SZL = 2/3 INCH
38B 3100368C739B2   1037                JMPU    LOOK.SZA                    GO BEGIN SYNC ZONE SEARCH
                    1038
                    1039  LP.GCR.WD     PASS.S  LIT    R2   B    #04        LOAD MS COUNT OF GCR WRITE DELAY 1 INCH
38C 01000A8C738D2   1040                NEXT
                    1041
                    1042                PASS.S  LIT    R1   B    #00        LOAD LS COUNT OF GCR WRITE DELAY 1 INCH
38D 0000068C738E2   1043                NEXT
                    1044
                    1045                PASS.S  LIT    RE   B    #05        LOAD MID.BYTE OF GCR SZL COUNT = 1+1/3 INCH
38E 01403A8C738F2   1046                NEXT
                    1047
                    1048                PASS.S  LIT    RD   B    #6B        LOAD LS BYTE OF GCR SZL COUNT = 1+1/3 INCH
38F 1AC0368C739B2   1049                JMPU    LOOK.SZA                    GO BEGIN SYNC ZONE SEARCH
                    1050
                    1051  MFM.WD        PASS.S  LIT    R2   B    #01        LOAD MS COUNT OF WRITE DELAY = 1/3 INCH
390 00400A8C73943   1052                JMPC    LP.MFM.WD                   GO LOAD 1 INCH DELAY IF NEAR LOAD POINT
                    1053
                    1054                PASS.S  LIT    R1   B    #0D        LOAD LS COUNT OF WRITE DELAY = 1/3 INCH
391 0340068C73922   1055                NEXT
                    1056
                    1057                PASS.S  LIT    RE   B    #02        LOAD MID. BYTE OF MFM SZL = 2/3 INCH
392 00803A8C73932   1058                NEXT
                    1059
                    1060                PASS.S  LIT    RD   B    #2E        LOAD LS BYTE OF MFM SZL = 2/3 INCH
393 0B80368C739B2   1061                JMPU    LOOK.SZA                    GO BEGIN SYNC ZONE SEARCH
                    1062
                    1063  LP.MFM.WD     PASS.S  LIT    R2   B    #03        LOAD NEAR LOAD POINT MS WRITE DELAY = 1 INCH
394 00C00A8C73952   1064                NEXT
                    1065
                    1066                PASS.S  LIT    R1   B    #1F        LOAD NEAR LOAD POINT LS WRITE DELAY = 1 INCH
395 07C0068C73962   1067                NEXT
                    1068
                    1069                PASS.S  LIT    RE   B    #04        LOAD MID.BYTE OF SZL, NEAR LOAD POINT = 1+1/3 INCH
396 01003A8C73972   1070                NEXT
                    1071
                    1072                PASS.S  LIT    RD   B    #40        LOAD LS BYTE OF SZL, NEAR LOAD POINT = 1+1/3 INCH
397 1000  C739B2   1073                JMPU    LOOK.SZA                    GO BEGIN SYNC ZONE SEARCH
```

```
                    957  ********************************************************************************
                    958  * IF FORWARD, WE CONFIRM LOAD POINT IS PASSED OR WE WAIT FOR IT                 *
                    959  ********************************************************************************
                    960
                    961  LOOK.LP      PASS.S   LIT    RF     B     #01        LOAD MS BYTE OF LOAD POINT LATE COUNT
376 00403E8C7378C   962               LDZ      LP.WAIT                       Z POINTS TO ROUTINE WHICH WAITS FOR LOAD POINT
                    963
                    964               PASS.S   LIT    RE     B     #B9        LOAD MIDDLE BYTE OF 110 INCH GCR TIMER, WHICH IS
377 2E403A8C73782   965               JMPU     LP.WAIT                       ALSO THE MIDDLE BYTE OF 141 INCH MFM TIMER
                    966
                    967
                    968
                    969
                    970  LP.WAIT                DRSTS          Q            INSPECT DRIVE STATUS - PAST LOAD POINT ?
378 000A024223792   971               NEXT                    BIT2          SAVE DRIVE STATUS IN CASE LOAD POINT IS LATE
                    972
                    973               NOP
379 00210006237F3   974               JMPC     PAST.LP        NRTC
                    975
                    976               NOP                                    IF NOT PAST LOAD POINT AND NOT TIME TO DECREMENT
37A 0000000622973   977               JMPC     IDLOOP3                       TIMER, THEN BACK TO IDLOOP
                    978
                    979               CLRTC                                  IT IS TIME TO DECREMENT TIMER
37B 00000266228F1   980               CALLC    DEC.3BYTES
                    981
                    982               NOP
37C 0000000622973   983               JMPC     IDLOOP3                       BACK TO IDLOOP IF TIMER NOT EXPIRED
                    984
                    985  LP.LATE      PASS.Q          R0     B               RECORD ERRONEOUS DRIVE STATUS IN R0
37D 0000000C237E2   986               NEXT
                    987
                    988               S.IOR.A  LIT    R7     B     #60        FLAG "TAPE POSITION LOST"
37E 18001E8DD4E82   989               JMPU     ERROR.8                       LOAD POINT IS LATE ! RECORD ERROR
                    990
                    991
                    992
                    993
                    994  PAST.LP      PASS.S   RAM                  #F0      INSPECT ERASE FLAG IN RAM #F0
37F 3C3802A47386C   995               LDZ      LOOK.SZW       NZERO          POINT Z TO "LOOK FOR SYNC ZONE WRITE"
                    996
                    997               PASS.A   R6                            INSPECT BIT "WRITE OR NOT WRITE ?"
380 000D00C443843   998               JMPC     ERASED.2LP     BIT5
                    999
                   1000               ZERO            RF     B               ZERO THE MSBYTE OF DEC.3BYTES FOR LOOK.SZW
381 00003C0E22883  1001               JMPC     BEGIN.WRITE                   IF WRITE, GO TURN WRITE LOGIC ON
                   1002
                   1003               NOP                                    I GUESS IT'S NOT A WRITE
382 000000062398C  1004               LDZ      LOOK.SZR                      POINT Z TO "LOOK FOR SYNC ZONE READ"
                   1005
                   1006               NOP
383 0000000622972  1007               JMPU     IDLOOP3                       BACK TO IDLOOP (BEGIN.WRITE ALSO GOES TO IDLOOP)
                   1008
                   1009  ERASED.2LP           RAM    DRSEL          SELECT   WE HAVE ERASED PAST THE LOAD POINT
384 0C0062A623852  1010               NEXT                                   SO RE-SELECT THE DEFAULT MFM RECORDING DENSITY
                   1011
                   1012               PASS.S   LIT    R7     B     #00        RE-ZERO OUT R7 (NO ERRORS) FOR THE ERASE ROUTINE
385 00001E8C7000A  1013               RETURNC                               AND RETURN TO THE ERASE ROUTINE
```

```
                    901  ********************************************************************************
                    902  * HERE'S THE START DELAY TIMER                                                 *
                    903  ********************************************************************************
                    904
                    905  STRT.DLYA          RAM              PRMSTS3     INSPECT BIT "MFM OR GCR ?"
367 05E802A6236DC   906                LDZ     STRT.DLYB     NBITO       POINT TO REST OF START DELAY TIMER
                    907
                    908  GCR.SD        PASS.S   LIT    RF     B    #02    LOAD MS COUNT OF GCR 20ms WAIT
368 00803E8C736B3   909                JMPC     MFM.SD
                    910
                    911                PASS.S   LIT    RE     B    #67    LOAD LS COUNT OF GCR 20ms WAIT
369 19C03A8C736A2   912                NEXT
                    913
                    914                PASS.A   RB     DRSEL              ASSERT "GCR MODE" IN HARDWARE (SEE 2nd LAST PAGE)
36A 00006164436D2   915                JMPU     STRT.DLYB
                    916
                    917  MFM.SD        PASS.S   LIT    RF     B    #01    LOAD MS COUNT OF MFM 20ms WAIT
36B 00403E8C736C2   918                NEXT
                    919
                    920                PASS.S   LIT    RE     B    #E0    LOAD LS COUNT OF MFM 20ms WAIT
36C 38003A8C736D2   921                JMPU     STRT.DLYB
                    922
                    923
                    924
                    925
                    926  STRT.DLYB     NOP                               THIS IS THE "WAIT START DELAY LOOP"
36D 00210006236E2   927                NEXT                NRTC          IS IT TIME TO DECREMENT THE COUNTER ?
                    928
                    929                NOP
36E 0000000622973   930                JMPC     IDLOOP3                   NO : RETURN TO IDLE LOOP
                    931
                    932                CLRTC                              YES : CLEAR RTC
36F 0000026622911   933                CALLC    DEC.2BYTES                GO DECREMENT THE COUNTER
                    934
                    935                PASS.A   R6                        INSPECT BIT "FORWARD OR REVERSE ?"
370 000A00C442973   936                JMPC     IDLOOP3        BIT2        RETURN TO IDLOOP IF COUNTER NOT EXPIRED
                    937
                    938                NOP
371 0000000623743   939                JMPC     FORWRD1
                    940
                    941
                    942
                    943
                    944  REVRSE1       S.IOR.A  LIT    R7     B    #40    FLAG "TAPE MOTION REVERSE"
372 10001E8DD398C   945                LDZ      LOOK.SZR                  Z POINTS TO "LOOK FOR SYNC ZONE READ"
                    946
                    947                NOP
373 0000000622972   948                JMPU     IDLOOP3
                    949
                    950  FORWRD1       S.IOR.A  LIT    R7     B    #20    FLAG "TAPE MOTION FORWARD"
374 08001E8DD376C   951                LDZ      LOOK.LP                   Z POINTS TO "LOOK FOR LOAD POINT"
                    952
                    953                NOP
375 0000000622972   954                JMPU     IDLOOP3
```

```
                    872  ********************************************************************************
                    873  * HERE IS THE RE-INITIALIZATION TO LOOP ON A WRITE REPEAT                      *
                    874  ********************************************************************************
                    875
                    876   REPEAT              LIT    RDCON       #00          CLEAR AND TURN OFF READ LOGIC
35F 00006E86224E1   877                 CALLC  WAIT.12.MS                     WAIT 1/3 INCH OF TAPE SO READ HEAD IS IN I.R.G.
                    878
                    879                 ZERO           R5     B              ZERO OUT THE STATUS
360 0000140E23612   880                 NEXT
                    881
                    882                 S.BCL.A LIT    R6     B    #C0       CLEAR TWO STATUS BITS IN R6
361 30001A8ED3622   883                 NEXT
                    884
                    885                 S.IOR.A LIT    R6     B    #10       SET BIT "WRITE GOING" IN R6
362 04001A8DD3632   886                 NEXT
                    887
                    888                 ZERO           RF     B              ZERO THE MSBYTE OF DEC.3BYTES FOR LOOK.SZW
363 00003C0E23642   889                 NEXT
                    890
                    891                 PASS.S  LIT    RE     B    #02       LOAD APPROX. 2/3 IN. SYNC ZONE LATE COUNT
364 00803A8C739BC   892                 LDZ    LOOK.SZA       NCFO           BYPASS START DELAY AND LOAD POINT WAIT IN READ
                    893
                    894                        LIT    WRCON       #36        TURN THE WRITE LOGIC ON. BYPASS DELAY BEFORE WRITE
365 0D807286228E3   895                 JMPC   CONT.WRT.RPT                   AND GO CONTINUE WRITE REPEAT IN THE WRITE ROUTINES
                    896
                    897                 ZERO           RAM         #FF        CLEAR WRITE REPEAT FLAG ON CF0 AND DO ONE LAST
366 3FC05C0622BE2   898                 JMPU   CONT.WRT.RPT                   WRITE REPEAT
```

```
                    833  ********************************************************************************
                    834  * ANY READ OR WRITE BEGINS IN THE READ ROUTINE. THE TAPE IS STARTED IN THE       *
                    835  * RIGHT DIRECTION, A START DELAY IS WAITED, IF FORWARD : WAIT FOR THE LOAD       *
                    836  * POINT TO PASS IF NOT ALREADY SO, IF WRITE : GO TO WRITE ROUTINES AND THEN      *
                    837  * LOAD DELAY BEFORE WRITE. ALSO LOAD SYNC ZONE LATE, WAIT FOR DATA DETECTED TO   *
                    838  * LAST AWHILE (WHILE AQUIRING LOCK) TO CONFIRM PRESENCE OF SYNC ZONE, START THE  *
                    839  * GAPFINDER WHICH ABORTS A WRITE IF DROPOUT DETECTED OR ABORTS A READ IF 1/3     *
                    840  * INCH OF I.R.G. IS DETECTED, LOOK FOR I.D. ZONE. WHEN FOUND, GO TO DECODE       *
                    841  * IT IN READ.GCR.ID, OR READ.MFM.ID.                                             *
                    842  ********************************************************************************
                    843
                    844   FORWARD              LIT    DRCON      #04          START TAPE FORWARD, SLOW.
357 0100668623SD2   845                JMPU     STARTED
                    846
                    847   REVERSE              LIT    DRCON      #02          START TAPE REVERSE, SLOW.
358 0080668623SD2   848                JMPU     STARTED
                    849
                    850   WRITE        NOP                                   FIRST CHECK THE 8000'S POWER SUPPLY
359 00000006203C8   851                JMPOPC   WAIT.RESET                    IF PFW, THEN WAIT FOR A RESET
                    852
                    853                         LIT    DRCON      #0C          START TAPE FORWARD; SLOW, WRITE.
35A 0300668623SD2   854                JMPU     STARTED
                    855
                    856   ERASE.2.LP           LIT    RAM        #F0          FLAG "THIS IS AN ERASE" AT RAM #F0
35B 3C005E86235C2   857                NEXT
                    858
                    859                         LIT    DRCON      #0C          START TAPE FORWARD; SLOW, WRITE
35C 0300668623SD2   860                JMPU     STARTED
                    861
                    862
                    863
                    864
                    865   STARTED      PASS.S   RAM    RB      B  SELECT      BEGIN PROCESS OF SELECTING GCR MODE
35D 0C002EAC7367C   866                LDZ      STRT.DLYA                     POINT Z TO START DELAY TIMER
                    867
                    868                S.IOR.A  LIT    RB      B  #80          SET BIT "GCR MODE" FOR STRT.DLYA'S USE
35E 20002E8DD2972   869                JMPU     IDLOOP3
```

```
                    793  ********************************************************************************
                    794  * HERE WE WRITE THE DATA CRC. IF THE BYTE COUNT = #00, THEN THIS CODE WRITES     *
                    795  * THE WORD COUNT CRC INSTEAD OF THE EARLIER CRC CODE. AFTER THE CRC IS WRITTEN,  *
                    796  * IT'S TIME TO WRITE THE REVERSE DATA I.D. AND THEN BRANCH BACK TO THE CODE      *
                    797  * THAT WROTE THE FILE MARK TO WRITE THE REVERSE FRAMING CHARACTER AND REVERSE    *
                    798  * SYNC ZONE.                                                                    *
                    799  ********************************************************************************
                    800
                    801   DATA.CRC      NOP                                        TIME TO WRITE THE DATA'S CRC
350 00200006229A1   802                 CALLC     IDLOOP5         NEVER
                    803
                    804                           LIT     WRCON       #F2          TURN BIT "WRITE CRC" ON
351 3C80728623522   805                 NEXT
                    806
                    807   BC.EQ.0                         WDATA                    CLEAR WDR
352 00206806229A1   808                 CALLC     IDLOOP5         NEVER
                    809
                    810
                    811
                    812
                    813   DATA.CRC7               LIT     WRCON       #FA          WRITE ONLY 7 MORE BITS OF CRC FOR A 15 BIT CRC
353 3E80728623542   814                 NEXT
                    815
                    816                           WDATA                    CLEAR WDR
354 00206806229A1   817                 CALLC     IDLOOP5         NEVER
                    818
                    819   REVDATA.ID              LIT     WRCON       #32          TURN BIT "WRITE CRC" OFF, RESET CRC
355 0C80728623562   820                 NEXT
                    821
                    822                           LIT     WDATA       #B3          WRITE THE REVERSE DATA I.D.
356 2CC06A8623222   823                 JMPU      REV.FRAME                        AND FINALLY, GO FINISH OFF THIS WRITE
                    824
                    825
                    826
                    827  ********************************************************************************
                    828  * THAT'S THE END OF THE MFM WRITE ROUTINES                                      *
                    829  * COMING UP NEXT ARE THE READ ROUTINES                                          *
                    830  ********************************************************************************
```

```
                    739  ************************************************************************************
                    740  * HERE IS THE ROUTINE WHICH WRITES THE ACTUAL DATA BYTES ONTO THE TAPE FOR MFM.  *
                    741  * IT ALSO HAS TO FETCH BYTES, SWAP MS & LS BYTES AROUND, AND FIGURE OUT WHEN     *
                    742  * IT'S DONE FETCHING AND DONE WRITING (THEY OCCUR IN TWO CONSEQUTIVE PASSES)     *
                    743  ************************************************************************************
                    744
                    745  W.MFMDATA          LIT     WRCON       #32          TURN OFF "WRITE CRC", "CRC7", AND RESET THE CRC
340 0C80728623412   746                NEXT
                    747
                    748  MFM.LOOP      A.XOR.B  R1      R6                   EVEN OR ODD BYTE TO BE WRITTEN NOW  ?...DO BYTE
341 0028182713422   749                NEXT                      NBIT0       SWAP DEPENDING ON WETHER EVEN OR ODD LENGTH RECORD
                    750
                    751                PASS.A   R5                           INSPECT BIT "BYTE FROM PPU HERE YET ?"
342 000900A443453   752                JMPC     SWAP             BIT1        SKIP TO SWAP BYTES AROUND
                    753
                    754  DONTSWAP      PASS.A   R4      WDATA                WRITE LATEST BYTE FROM PPU ONTO TAPE
343 0000688442861   755                CALLC    UNDFLW                        FLAG UNDER FLOW IF IT HAPPENED
                    756
                    757                NOP
344 0000000623472   758                JMPU     DONESWAP                      SKIP PAST THE BYTE SWAPPER
                    759
                    760  SWAP                   RAM     WDATA       SWAPPER  WRITE BYTE FETCHED 2 PASSES BACK ONTO TAPE
345 14406AA622861   761                CALLC    UNDFLW                        FLAG UNDER FLOW IF IT HAPPENED
                    762
                    763                PASS.A   R4      RAM         SWAPPER  SAVE LATEST BYTE FETCHED TO BE WRITTEN LATER
346 14405C8443472   764                JMPU     DONESWAP
                    765
                    766  DONESWAP      DEC.A    R1      R1  .       B        DECREMENT THE LS BYTE COUNT
347 0019042CC3482   767                NEXT                      NBORROW
                    768
                    769                PASS.A   R1                           INPSECT LS BYTE COUNT TO SEE IF WE STILL FETCH
348 0038002443483   770                JMPC     NDONEWRITE       NZERO       JUMP TO NOT DONE WRITING BYTES YET
                    771
                    772                DEC.A    R2      R2       B           DECREMENT THE MS BYTE COUNT SINCE LS ROLLED OVER
349 0039084CC34A2   773                NEXT                      BORROW
                    774
                    775                PASS.A   R1                           INSPECT LS BYTE COUNT TO SEE IF WE STILL FETCH
34A 0038002443503   776                JMPC     DATA.CRC         NZERO       WRITE DATA CRC IF BYTE COUNT EXPIRED
                    777
                    778  NDONEWRITE    PASS.A   R2                           INSPECT MS COUNT TO SEE IF WE STILL FETCH
34B 0018044434D3    779                JMPC     FETCH            ZERO        FETCH IF LS BYTE COUNT # #00
                    780
                    781                NOP
34C 00000006234E3   782                JMPC     NFETCH                       DONT FETCH IF MS BYTE COUNT ALSO = #00
                    783
                    784  FETCH         S.IOR.A  LIT     R5       B    #02    SET BIT "FETCH DATA BYTE FROM THE PPU"
34D 0080168DD34E2   785                NEXT
                    786
                    787  NFETCH        NOP
34E 002000062291A1  788                CALLC    IDLOOP5          NEVER
                    789
                    790                         LIT     WRCON       #B2      TURN "RESET CRC" OFF, CALCULATE DATA CRC
34F 2C80728623412   791                JMPU     MFM.LOOP
```

```
                    685  ********************************************************************************
                    686  * NOW TO WRITE THE WORD COUNT'S CRC WHILE LOADING UP THE BYTE COUNT FOR THE      *
                    687  * DATA WRITING ROUTINE, SKIPPING TOWARDS END IF BYTE COUNT = #00, AND            *
                    688  * PRE-FETCHING TWO BYTES, OR ONE BYTE IF BYTE COUNT =#01.                       *
                    689  ********************************************************************************
                    690
                    691  WC.CRC       DEC.S    RAM     R1      B     OPERAND4     DECREMENT AND LOAD LS BYTE COUNT
331 0DD906AD73322   692               NEXT                     NBORROW
                    693
                    694               PASS.S   RAM     R2      B     OPERAND3     LOAD MS BYTE COUNT
332 0DA00AAC73343   695               JMPC     BC.NE.0         NEVER              LS BYTE COUNT WAS > #00
                    696
                    697               DEC.A    R2      R2      B                  DECREMENT MS BYTE COUNT SINCE LS ROLLED OVER
333 0039084CC3342   698               NEXT                     BORROW
                    699
                    700  BC.NE.0                   LIT     WRCON       #F2        TURN ON "WRITE CRC"
334 3C80728623523   701               JMPC     BC.EQ.0                            SKIP TOWARDS END SINCE BYTE COUNT = #00
                    702
                    703                                WDATA                      CLEAR WDR
335 0000680623362   704               NEXT
                    705
                    706               S.IOR.A  LIT     R5      B     #02          SETBIT "FETCH A DATA BYTE FROM THE PPU"
336 00A0168DD29A1   707               CALLC    IDLOOP5         NEVER
                    708
                    709
                    710
                    711
                    712  WC.CRC7      PASS.A   R5                                 INSPECT BIT "BYTE FROM PPU HERE YET ?"
337 000900A443382   713               NEXT                     BIT1
                    714
                    715                        LIT     WRCON       #FA            WRITE ONLY 7 MORE BITS OF CRC FOR A 15 BIT CRC
338 3E80728622861   716               CALLC    UNDFLW                             FLAG UNDER FLOW IF IT HAPPENED
                    717
                    718                                WDATA                      CLEAR WDR
339 00006806233A2   719               NEXT
                    720
                    721               PASS.A   R4      RAM         SWAPPER       TRANSFER 1st BYTE TO BUFFER TO SWAP MS & LS BYTES
33A 14405C84433B2   722               NEXT
                    723
                    724               PASS.A   R2                                 INPECT MS BYTE COUNT
33B 00380044433C2   725               NEXT                     NZERO
                    726
                    727               PASS.A   R1                                 INSPECT LS BYTE COUNT
33C 00180024433E3   728               JMPC     FETCH2          ZERO               FETCH A SECOND BYTE IF MS COUNT # #00
                    729
                    730               NOP
33D 00000006233F3   731               JMPC     NFETCH2                            DONT FETCH A SECOND BYTE IF LS COUNT = #00
                    732
                    733  FETCH2       S.IOR.A  LIT     R5      B     #02          SET BIT "FETCH A DATA BYTE FROM THE PPU"
33E 0080168DD33F2   734               NEXT
                    735
                    736  NFETCH2      NOP
33F 00200006229A1   737               CALLC    IDLOOP5         NEVER
```

```
                    651  ********************************************************************************
                    652  * SINCE IT'S A DATA RECORD TO BE WRITTEN, WRITE THE DATA I.D. AND THE WORD COUNT *
                    653  ********************************************************************************
                    654
                    655   MFMDATA.ID   NOP
329 00200006229A1   656                CALLC    IDLOOP5        NEVER
                    657
                    658                LIT      WDATA          #CD          WRITE THE LEADING DATA I.D. ZONE
32A 33606A86229A1   659                CALLC    IDLOOP5        NEVER
                    660
                    661
                    662
                    663
                    664   WORDCOUNTA             RAM      WDATA          LSWORDCNT    WRITE THE LSBYTE OF WORD COUNT
32B 14A06AA6229A1   665                CALLC    IDLOOP5        NEVER
                    666
                    667
                    668
                    669
                    670   WORDCOUNTB             LIT      WRCON          #B2          TURN OFF CRC RESET, CALCULATE THE WORD COUNT CRC
32C 2C80728623202   671                NEXT
                    672
                    673                RAM      WDATA          MSWORDCNT    WRITE THE MSBYTE OF WORD COUNT
32D 14CF6AA6232E2   674                NEXT                    BIT7         CHECK MSBIT TO SEE IF BYTE COUNT IS EVEN OR ODD
                    675
                    676                S.IOR.A  LIT      R6     B     #01    SETBIT "BYTE COUNT ODD"
32E 00401A8DD3303   677                JMPC     *+2                         SKIP CUZ BYTE COUNT IS ODD
                    678
                    679                S.BCL.A  LIT      R6     B     #01    CLEAR BIT "BYTE COUNT ODD"
32F 00401A8ED3302   680                NEXT
                    681
                    682                NOP
330 00200006229A1   683                CALLC    IDLOOP5        NEVER
```

```
                    594  ********************************************************************************
                    595  * HERE IS ALL THE CODE TO WRITE AN MFM FILE MARK. INCLUDED IN THIS CODE         *
                    596  * ARE ENTRY AND EXIT POINTS TO WRITE MFM DATA RECORDS WHICH WILL USE THIS       *
                    597  * CODE TO WRITE THE SYNC ZONES AND FRAMING CHARACTERS.                          *
                    598  ********************************************************************************
                    599
                    600  MFMSYNC1      NOP                                        FIRST THE LOOP TO WRITE A LEADING SYNC ZONE
318 002000006229A1  601               CALLC    IDLOOP5        NEVER
                    602
                    603                        LIT    WRCON        #32            TURN OFF THE A.C. ERASE
319 0C80728623 1A2  604               NEXT
                    605
                    606               DEC.A    R1      R1     B                   DECREMENT COUNT OF SYNC BYTES TO WRITE YET
31A 0019042CC3182   607               NEXT                    NBORROW
                    608
                    609               ZERO              WDATA                    WRITE A ZERO SYNC BYTE
31B 0000680623183   610               JMPC     MFMSYNC1                          LOOP TILL THE SYNC ZONE IS WRITTEN
                    611
                    612  FOR.FRAME     NOP                                       TIME TO WRITE THE FORWARD FRAMING CHARACTER
31C 0020000 6229A1  613               CALLC    IDLOOP5        NEVER
                    614
                    615               PASS.A   R6                                IS THIS A FILE MARK OR DATA RECORD ?
31D 002B00C4431E2   616               NEXT                    NBIT3
                    617
                    618                        LIT    WDATA        #FF            WRITE THE FRAMING CHARACTER
31E 3FC06A8623293   619               JMPC     MFMDATA.ID                        JUMP IF IT'S A DATA RECORD WE'RE WRITING
                    620
                    621  MFM.FM        NOP                                       I GUESS WE'RE WRITING A FILE MARK
31F 00200006229A1   622               CALLC    IDLOOP5        NEVER
                    623
                    624                        LIT    WDATA        #B5            WRITE THE LEADING FILE MARK I.D.
320 2D606A86229A1   625               CALLC    IDLOOP5        NEVER
                    626
                    627  MFM.FM.REV             LIT    WDATA        #AD            WRITE THE REVERSE FILE MARK I.D.
321 2B406A8623222   628               JMPU     REV.FRAME                         GO WRITE THE REVERSE FRAMING CHARACTER
                    629
                    630  REV.FRAME     NOP                                       TIME TO WRITE THE REVERSE FRAMING CHARACTER
322 00200006229A1   631               CALLC    IDLOOP5        NEVER
                    632
                    633                        LIT    WDATA        #FF            WRITE THE REVERSE FRAMING CHARACTER
323 3FC06A8623242   634               NEXT
                    635
                    636               PASS.S   LIT    R1      B      #09          THE REVERSE SYNC ZONE IS 9+1 BYTES LONG
324 0240068C73252   637               NEXT
                    638
                    639  MFMSYNC2      NOP                                       TIME TO WRITE THE REVERSE SYNC ZONE
325 00200006229A1   640               CALLC    IDLOOP5        NEVER
                    641
                    642               DEC.A    R1      R1     B                   DECREMENT THE COUNT OF SYNC BYTES TO WRITE
326 0019042CC3272   643               NEXT                    NBORROW
                    644
                    645               ZERO              WDATA                    WRITE A ZERO SYNC BYTE
327 0000680623253   646               JMPC     MFMSYNC2                          LOOP ON SYNC BYTE WRITE TILL DONE
                    647
                    648               NOP                                        WE'RE DONE SO GO WRITE A.C. ERASE
328 0000000622C12   649               JMPU     AC.ERASE
```

```
                    547  ********************************************************************************
                    548  * HERE THE TRAILING BLOCKETTE IS WRITTEN. IT IS EXACTLY LIKE THE LEADING       *
                    549  * BLOCKETTE EXCEPT ITS 100% REVERSED (BACKWARDS).                               *
                    550  ********************************************************************************
                    551
                    552   TRAILER      NOP                                          WRITE THE REVERSED BYTE COUNT BLOCKETTE
30E 00200006229A1   553                CALLC    IDLOOP5        NEVER
                    554
                    555                         LIT    WRCON       #30              TURN OFF THE CRC
30F 0C00728623102   556                NEXT
                    557
                    558                PASS.S   RAM    R1     B    LSBCREV          LOAD THE REVERSED LS BYTE COUNT
310 15C006AC73112   559                NEXT
                    560
                    561                NOT.A    R1     WDATA                       WRITE LSBCREV'S COMPLIMENT
311 00206825429A1   562                CALLC    IDLOOP5        NEVER
                    563
                    564                PASS.S   RAM    R2     B    MSBCREV          LOAD THE REVERSED MS BYTE COUNT
312 15800AAC73132   565                NEXT
                    566
                    567                NOT.A    R2     WDATA                       WRITE MSBCREV'S COMPLIMENT
313 00206845429A1   568                CALLC    IDLOOP5        NEVER
                    569
                    570                         RAM    WDATA       LSBCREV         WRITE REVERSED LS BYTE COUNT
314 15E06AA6229A1   571                CALLC    IDLOOP5        NEVER
                    572
                    573                         RAM    WDATA       MSBCREV         WRITE REVERSED MS BYTE COUNT
315 15A06AA6229A1   574                CALLC    IDLOOP5        NEVER
                    575
                    576
                    577
                    578
                    579   BLOCK.ID.D           LIT    WRCON       #32              ASSERT BIT "WRITE ILLEGAL GCR CODES"
316 0C80728623172   580                NEXT
                    581
                    582                         LIT    WDATA       #D5              WRITE ID "#AB" REVERSED (=#D5)
317 35406A8622CE2   583                JMPU     LASTSYNC                            GO WRITE THE LAST SYNC ZONE
                    584
                    585
                    586
                    587
                    588  ********************************************************************************
                    589  * WELL, THAT'S THE END OR THE GCR WRITE ROUTINES.                               *
                    590  * COMING UP NEXT ARE THE MFM WRITE ROUTINES                                     *
                    591  ********************************************************************************
```

```
                    507  ********************************************************************************
                    508  * WRITE THE XOR BYTES AND THE CRC FOR THE XOR BLOCKETTES                       *
                    509  ********************************************************************************
                    510
                    511  XOR.LOOP     PASS.A   R3     SADRL  B              RE-LOAD THE RAM POINTER
303 00004C6C43042   512               NEXT
                    513
                    514                        SCR    WDATA                 WRITE THE XOR BYTE TO THE TAPE
304 0000EAA623052   515               NEXT
                    516
                    517               INC.A    R3     SADRL  B              INCREMENT THE RAM POINTER
305 00184C6C4B062   518               NEXT                   ZERO          ARE WE DONE WITH THIS BLOCKETTE ?
                    519
                    520               NOP                                  IF SO, OFF TO WRITE THE CRC
306 0000000623093   521               JMPC     XOR.CRC
                    522
                    523  XL1          NOP                                  IF NOT, LOOP ON XOR BYTE WRITE
307 00200006229A1   524               CALLC    IDLOOP5       NEVER
                    525
                    526               PASS.A   R2     SADRH                POINT TO THE CORRECT PAGE IN RAM FOR XOR BYTES
308 0000584443032   527               JMPU     XOR.LOOP                    LOOP UP TO WRITE THE XOR BYTES
                    528
                    529
                    530
                    531
                    532  XOR.CRC      NOP                                  TIME TO WRITE THE XOR BLOCKETTE CRC
309 00200006229A1   533               CALLC    IDLOOP5       NEVER
                    534
                    535                        LIT    WRCON       #F0       TURN ON "WRITE CRC" BIT
30A 3C00728623082   536               NEXT
                    537
                    538                        WDATA                       CLEAR WDR
308 00206806229A1   539               CALLC    IDLOOP5       NEVER
                    540
                    541               DEC.A    R2     R2     B              DECREMENT COUNT OF XOR BLOCKETTES TO WRITE
30C 0019084CC30D2   542               NEXT                   NBORROW
                    543
                    544                        WDATA                       CLEAR WDR
30D 0000680622FA3   545               JMPC     XOR.BLOCK                   WRITE XOR BLOCKETTES TILL 2 ARE WRITTEN
```

```
                    467  ********************************************************************************
                    468  * NOW TO WRITE THE XOR BLOCKETTES' SYNC ZONES AND BLOCKETTE NUMBERS             *
                    469  ********************************************************************************
                    470
                    471  XOR.WRITE     PASS.S   LIT     R2      B     #01          FIRST XOR BLOCKETTE NUMBER IS #01
2F9 00400A8C72FA2   472                NEXT
                    473
                    474  XOR.BLOCK     PASS.S   LIT     R1      B     #07          (7+1)*10=80 SYNC FLUX REVERSALS TO WRITE
2FA 01C0068C72F82   475                NEXT
                    476
                    477  SYNCLOOP3     NOP                                         TIME TO WRITE THE XOR BLOCKETTES' SYNC ZONES
2FB 002000062291A1  478                CALLC    IDLOOP5         NEVER
                    479
                    480                         LIT     WRCON         #32          CLEAR CRC, WRITE ILLEGAL GCR CODES
2FC 0C80728622FD2   481                NEXT
                    482
                    483                DEC.A    R1      R1      B                  DECREMENT COUNT OF SYNC BYTES TO WRITE
2FD 0019042CC2FE2   484                NEXT                     NBORROW
                    485
                    486                         LIT     WDATA         #FF          WRTIE A SYNC BIT
2FE 3FC06A8622F83   487                JMPC     SYNCLOOP3                          LOOP ON SYNC BYTE WRTIE TILL DONE
                    488
                    489
                    490
                    491
                    492  BLOCK.ID.C    NOP                                         TIME TO WRITE THE BLOCKETTE I.D (C)
2FF 002000062291A1  493                CALLC    IDLOOP5         NEVER
                    494
                    495                         LIT     WDATA         #C8          THE "B" IS THE FRAMING CHARACTER
300 32E06A86229A1   496                CALLC    IDLOOP5         NEVER
                    497
                    498
                    499
                    500
                    501  BLOCK.NUM.C              LIT     WRCON         #80          UNCLEAR CRC, WRITE LEGAL GCR CODES
301 2C00728623022   502                NEXT
                    503
                    504                PASS.A   R2      WDATA                       WRITE THE BLOCKETTE NUMBER
302 0000684443072   505                JMPU     XL1                                 GO BEGIN WRITING XOR BYTES
```

```
                    416  ********************************************************************************
                    417  * HERE WE WRITE THE DATA AND THE CRC FOR THE DATA BLOCKETTES                    *
                    418  ********************************************************************************
                    419
                    420  DATALOOP      PASS.A   R5                                  INSPECT BIT "BYTE FROM PPU HERE YET ?"
2EA 000900A442E82   421                NEXT                  BIT1
                    422
                    423                A.XOR.Q  R4     SCR                          CALCULATE AND RE-LOAD XOR CALCULATION
2EB 0000DC8702861   424                CALLC    UNDFLW                            FLAG AN UNDER FLOW IF IT HAPPENED
                    425
                    426                INC.A    R3     SADRL  B                   INCREMENT RAM POINTER
2EC 00184C6C4AED2   427                NEXT                   ZERO               ARE WE DONE WRITING THIS BLOCKETTE ?
                    428
                    429                PASS.A   R4     WDATA                      WRITE THE LATEST DATA BYTE TO THE TAPE
2ED 0000688442F43   430                JMPC     GCR.CRC                           JUMP IF IT'S TIME TO WRITE THE CRC
                    431
                    432                PASS.A   R2                                INSPECT MS COUNT TO SEE IF WE'RE DONE
2EE 0038004442EF2   433                NEXT                   NZERO              FETCHING BYTES FROM PPU OR NOT
                    434
                    435                S.SUB.A  RAM    R3          DONECOUNTW     IF LAST BLOCKETTE, IS BYTE COUNT EXPIRED ?
2EF 16790EA55AF13   436                JMPC     DOFETCH       BORROW
                    437
                    438                NOP                                        NO FETCH IF MSCOUNT=0 & RAM POINTER >= LSBYTE COUNT
2F0 0000000622F23   439                JMPC     DONTFETCH
                    440
                    441  DOFETCH       S.IOR.A  LIT    R5     B     #02           SET BIT "FETCH A DATA BYTE FROM THE PPU"
2F1 0080168DD2F22   442                NEXT
                    443
                    444  DONTFETCH     ZERO             R4     B                  ZERO THE INPUT BUFFER
2F2 0020100E229A1   445                CALLC    IDLOOP5       NEVER              BACK TO IDLOOP
                    446
                    447                PASS.S   SCR           Q                  PUT XOR CALCULATION INTO Q
2F3 000082A072EA2   448                JMPU     DATALOOP                          LOOP UP TO WRITE ALL THE DATA BYTES
                    449
                    450
                    451
                    452  GCR.CRC       NOP                                        TIME TO WRITE THE DATA BLOCKETTE CRC
2F4 00200006229A1   453                CALLC    IDLOOP5       NEVER
                    454
                    455                LIT      WRCON         #F0                TURN ON BIT "WRITE CRC"
2F5 3C00728622F62   456                NEXT
                    457
                    458                         WDATA                           CLEAR WDR
2F6 0020680622A1    459                CALLC    IDLOOP5       NEVER
                    460
                    461                DEC.A    R2     R2     B                  DECREMENT MSBYTE COUNT/BLOCKETTE NUMBER
2F7 0019084CC2F82   462                NEXT                   NBORROW
                    463
                    464                         WDATA                           CLEAR WDR
2F8 0000680622DE3   465                JMPC     BLOCKLOOP                       LOOP ON DATA BLOCKETTES UNLESS DONE
```

```
                    370  ********************************************************************************
                    371  * WRITE THE SYNC ZONES AND THE BLOCKETTE NUMBERS FOR THE DATA BLOCKETTES.      *
                    372  ********************************************************************************
                    373
                    374  BLOCKLOOP     PASS.S  LIT     R1      B       #07         (7+1)*10=80 SYNC FLUX REVERSALS TO WRITE
2DE 01C0068C72DF2   375                NEXT
                    376
                    377  SYNCLOOP2     NOP                                         TIME TO WRITE THE DATA BLOCKETTE SYNC ZONE
2DF 00200006229A1   378                CALLC   IDLOOP5         NEVER
                    379
                    380                        LIT     WRCON       #32             ASSERT "WRITE ILLEGAL GCR CODES"
2E0 0C80728622E12   381                NEXT
                    382
                    383                DEC.A   R1      R1      B                   DECREMENT THE COUNT OF SYNC BYTES TO WRITE YET
2E1 0019042CC2E22   384                NEXT                    NBORROW
                    385
                    386                        LIT     WDATA       #FF             WRITE A SYNC BYTE
2E2 3FC06A8622DF3   387                JMPC    SYNCLOOP2                           CONTINUE WRITING SYNC BYTES TILL DONE
                    388
                    389
                    390
                    391
                    392  BLOCK.ID.B    NOP                                         NEXT WRITE THE DATA BLOCKETTE I.D. ZONE
2E3 00200006229A1   393                CALLC   IDLOOP5         NEVER
                    394
                    395                        LIT     WDATA       #BB             WRITE FRAMING CHARACTER AND DATA BLOCKETTE I.D.
2E4 2EE06A86229A1   396                CALLC   IDLOOP5         NEVER
                    397
                    398
                    399
                    400
                    401  BLOCK.NUM.B           LIT     WRCON       #B0             CLEAR BITS "WRITE ILLEGAL GCR CODE AND CLEAR CRC"
2E5 2C00728622E62   402                NEXT
                    403
                    404                PASS.A  R2      WDATA                       WRITE THE BLOCKETTE NUMBER/MSBYTE COUNT TO TAPE
2E6 0008684442E72   405                NEXT                    BIT0                IS THIS AN EVEN OR ODD NUMBERED BLOCKETTE ?
                    406
                    407                ZERO            SADRL   B                   POINT TO BEGINNING OF RAM'S PAGE TO COMPUTE XOR IN
2E7 00004C0E22E93   408                JMPC    *+2                                 SKIP IF ODD NUMBERED BLOCKETTE
                    409
                    410                        LIT     SADRH       #00             SELECT PAGE 0 FOR XOR BLOCKETTE CALCULATION
2E8 00005A8622F12   411                JMPU    DOFETCH                             GO GET THE FIRST BYTE TO WRITE
                    412
                    413                        LIT     SADRH       #01             SELECT PAGE 1 FOR XOR BLOCKETTE CALCULATION
2E9 00405A8622F12   414                JMPU    DOFETCH                             GO GET THE FIRST BYTE TO WRITE.
```

```
                    330  ********************************************************************************
                    331  * SINCE IT'S A DATA RECORD TO BE WRITTEN, WRITE THE LEADING BYTE COUNT BLOCKETTE *
                    332  ********************************************************************************
                    333
                    334   GCR.DATA      NOP                                          BEGIN WRITING THE GCR DATA RECORD
2D3 00200006229A1   335                 CALLC     IDLOOP5        NEVER
                    336
                    337   BLOCK.ID.A              LIT     WDATA        #A8           WRITE THE BYTE COUNT BLOCKETTE I.D. (A)
2D4 2AE06A86229A1   338                 CALLC     IDLOOP5        NEVER            THE "B" IS THE FRAMING CHARACTER
                    339
                    340
                    341
                    342
                    343   BYTE.CNT                LIT     WRCON        #30           TURN OF THE "WRITE ILLEGAL GCR" BIT
2D5 0C00728622D62   344                 NEXT
                    345
                    346                         RAM     WDATA        OPERAND3      WRITE THE MS BYTE COUNT TO THE TAPE
2D6 0DA06AA6229A1   347                 CALLC     IDLOOP5        NEVER
                    348
                    349                         RAM     WDATA        OPERAND4      WRITE THE LS BYTE COUNT TO THE TAPE
2D7 0DE06AA6229A1   350                 CALLC     IDLOOP5        NEVER
                    351
                    352                 PASS.S    RAM     R2       B     OPERAND3   STORE THE MSBYTE COUNT FOR LATER USE
2D8 0D800AAC72D92   353                 NEXT
                    354
                    355                 NOT.A     R2      WDATA                     WRITE THE MS BYTE COUNT INVERTED
2D9 00206845429A1   356                 CALLC     IDLOOP5        NEVER
                    357
                    358                 PASS.S    RAM     R1       B     OPERAND4   STORE THE LSBYTE COUNT
2DA 0DF806AC72DB2   359                 NEXT                       NZERO            CHECK IF LAST BLOCKETTE WILL BE EMPTY
                    360
                    361                 NOT.A     R1      WDATA                     WRITE THE LS BYTE COUNT INVERTED
2DB 0000682542DE3   362                 JMPC      BLOCKLOOP                         BEGIN DATA WRITE SINCE LAST BLOCKETTE IS NOT EMPTY
                    363
                    364                 DEC.A     R2      R2       B                DECREMENT BLOCKETTE COUNT SINCE LAST ONE IS EMPTY
2DC 0039084CC2DD2   365                 NEXT                       BORROW           CHECK IF WE ARE TO WRITE NO DATA
                    366
                    367                 NOP                                        IF THERE IS NO DATA, GO WRITE THE REVERSED BYTE
2DD 00000006230E3   368                 JMPC      TRAILER                          COUNT BLOCKETTE. ELSE, GO TO BLOCKLOOP
```

```
                    276  ********************************************************************************
                    277  * HERE IS THE CODE TO WRITE A GCR FILE MARK. INCLUDED IN THIS CODE ARE          *
                    278  * ENTRY AND EXIT POINTS TO WRITE GCR DATA RECORDS WHICH WILL USE THIS CODE       *
                    279  * TO WRITE THE INITIAL AND FINAL SYNC ZONES.                                    *
                    280  ********************************************************************************
                    281
                    282   SYNCLOOP1    NOP                                          FIRST WRITE A LEADING SYNC ZONE
2C5 00200006229A1   283                CALLC    IDLOOP5        NEVER
                    284
                    285                LIT      WRCON      #32                      TURN OFF THE A.C. ERASE
2C6 0C80728622C72   286                NEXT
                    287
                    288                DEC.A    R1     R1      B                    DECREMENT COUNT OF SYNC BYTES TO WRITE YET
2C7 0019042CC2C82   289                NEXT                    NBORROW
                    290
                    291                LIT      WDATA      #FF                      WRITE A SYNC BYTE
2C8 3FC06A8622C53   292                JMPC     SYNCLOOP1                           LOOP TILL THE SYNC ZONE IS WRITTEN
                    293
                    294                PASS.A   R6                                  IS THIS A FILE MARK OR A DATA RECORD ?
2C9 002800C442CA2   295                NEXT                    NBIT3
                    296
                    297                NOP
2CA 0000000622D33   298                JMPC     GCR.DATA
                    299
                    300
                    301
                    302
                    303   GCR.FM       NOP                                          WE ARE TO WRITE A FILE MARK
2CB 00200006229A1   304                CALLC    IDLOOP5        NEVER
                    305
                    306                LIT      WDATA      #F8                      WRITE THE LEADING FILE MARK I.D.
2CC 3EE06A86229A1   307                CALLC    IDLOOP5        NEVER
                    308
                    309                LIT      WDATA      #DF                      WRITE THE TRAILING FILE MARK I.D.
2CD 37C06A8622CE2   310                JMPU     LASTSYNC
                    311
                    312   LASTSYNC     PASS.S   LIT    R1     B    #07              (7+1)*10=80 GCR SYNC FLUX REVERSALS TO WRITE
2CE 01C0068C72CF2   313                NEXT
                    314
                    315
                    316
                    317
                    318   SYNCLOOP4    NOP                                          THE LOOP TO WRITE THE LAST SYNC ZONE
2CF 00200006229A1   319                CALLC    IDLOOP5        NEVER
                    320
                    321                DEC.A    R1     R1      B                    DECREMENT THE COUNT OF SYNC BYTES TO WRITE YET
2D0 0019042CC2D12   322                NEXT                    NBORROW
                    323
                    324                LIT      WDATA      #FF                      WRITE A SYNC BYTE
2D1 3FC06A8622CF3   325                JMPC     SYNCLOOP4
                    326
                    327                NOP                                          WE'RE DONE WRITING THIS GCR RECORD
2D2 0000000622C12   328                JMPU     AC.ERASE                            SO GO WRITE A.C. ERASE
```

```
                    218  ********************************************************************************
                    219  * FOLLOWING ARE THE WRITE ROUTINES WHICH ARE COMMON TO BOTH THE MFM AND GCR      *
                    220  * RECORDING MODES. THEY ARE THE DELAY BEFORE WRITE TIMER AND THE WRITE A.C.      *
                    221  * ERASE GAP (THE BEGINNING AND THE END)                                          *
                    222  * THE NUMBER FOR THE DELAY BEFORE WRITE TIMER IS LOADED IN THE READ ROUTINE.     *
                    223  ********************************************************************************
                    224
                    225  BEGIN.WRITE          LIT    WRCON       #36          TURN THE "Write Data Ready" LOGIC ON.
288 0D80728622892   226                NEXT
                    227
                    228
                    229
                    230  TIMER.WD      S.IOR.A LIT    R6     B    #10          SET FLAG "WRITE GOING"
289 04201A8DD29A1   231                CALLC  IDLOOP5       NEVER
                    232
                    233                DEC.A  R1      R1     B                 DECREMENT THE LS COUNTER
28A 0019042CC28B2   234                NEXT                  NBORROW
                    235
                    236                       WDATA                        CLEAR WDR
28B 0000680622893   237                JMPC   TIMER.WD                     IF NO ROLLOVER, WAIT FOR NEXT WDR
                    238
                    239                DEC.A  R2      R2     B                 DECREMENT THE MS COUNTER SINCE LS ROLLED OVER
28C 0019084CC28D2   240                NEXT                  NBORROW
                    241
                    242                NOP
28D 0000000622893   243                JMPC   TIMER.WD                     IF NO ROLLOVER, WAIT FOR NEXT WDR
                    244
                    245
                    246
                    247  CONT.WRT.RPT          RAM                  PRMSTS3  TIME TO START WRITING SO IS IT MFM OR GCR ?
28E 05C802A6228F2   248                NEXT                  BITO           ALSO, RE-ENTRY POINT TO CONTINUE WRITE REPEAT
                    249
                    250                PASS.S LIT    R1     B    #07          (7+1)*10=80 GCR SYNC FLUX REVERSALS TO WRITE
28F 01C0068C72C53   251                JMPC   SYNCLOOP1                    GO WRITE THE FIRST OF MANY GCR SYNC ZONES
                    252
                    253                PASS.S LIT    R1     B    #09          (9+1)*8=80 MFM SYNC BITS TO WRITE
2C0 0240068C73182   254                JMPU   MFMSYNC1                     GO WRITE THE MFM SYNC ZONE
                    255
                    256
                    257
                    258  * SEE REST OF WRITE ROUTINES FOR WHAT GOES IN BETWEEN THE DELAY BEFORE WRITE IN THE BEGINNING
                    259  *    AND THE A.C ERASE AT THE END OF EACH WRITE.
                    260
                    261
                    262
                    263  AC.ERASE      NOP                                  WRITE DONE, WAIT FOR LAST WDR
2C1 002000062291A1  264                CALLC  IDLOOP5       NEVER
                    265
                    266                LIT    WRCON       #26          TURN OFF WDR, TURN ON A.C. ERASE
2C2 0980728622C32   267                NEXT
                    268
                    269                ZERO          WDATA                  CLEAR WDR
2C3 0000680622C42   270                NEXT
                    271
                    272                S.BCL.A LIT    R6     B    #10          CLEAR FLAG "WRITE GOING"
2C4 04201A8ED29A2   273                JMPU   IDLOOP5       NEVER          AND DO FINAL RETURN TO IDLE LOOP
```

```
                    163  ********************************************************************************
                    164  * HERE ARE THE I/O ROUTINES WHICH THE INDEX TABLE ON THE LAST PAGE USE TO      *
                    165  * HANDLE THE DATA TRANSFERS FROM/TO THE PPU. IN THE CASE OF THE OUTPUTS, THE    *
                    166  * I/O INDEX IS INCREMENTED TO STEP THRU THE SEQUENCE TO SEND A BYTE. IT TAKES   *
                    167  * TWO PAASES THRU THE I/O TABLE TO SEND ONE BYTE !                             *
                    168  ********************************************************************************
                    169
                    170  INPUTA      ZERO            R4      B               ZERO OUT THE INPUT BUFFER IN R4
2AC 0017100E22A03   171             JMPC    I.O.ADDR.0      DCI             RETURN IF BYTE NOT HERE FROM PPU
                    172
                    173             PASS.A  R5                              IF BYTE NOT READY FROM PPU, SERVICE
2AD 000C00A442943   174             JMPC    IDLOOP2         BIT4            READ ROUTINES ALWAYS OR ON RDR ?
                    175
                    176             PASS.S  DIN     R4      B               BYTE HERE AND READY SO PUT IT IN INPUT BUFFER
2AE 0022120C72AF2   177             NEXT                    NDPE            AND CHECK ITS PARITY
                    178
                    179             A.SUB.S LIT     R5      B       #02     RECORD BYTE GOTTEN
2AF 0080168CDAA03   180             JMPC    I.O.ADDR.0                      RETURN IF NO PARITY ERROR
                    181
                    182             S.IOR.A LIT     R5      B       #88     RECORD A PARITY ERROR
2B0 2200168DD2A02   183             JMPU    I.O.ADDR.0                      RETURN
                    184
                    185
                    186
                    187
                    188  INPUTB      ZERO            R4      B               ZERO THE INPUT BUFFER SINCE AN ERROR HAS OCCURED
2B1 0000100E22A82   189             JMPU    I.O.ADDR.8                      AND RETURN
                    190
                    191
                    192
                    193
                    194
                    195
                    196  OUTPUTA     PASS.A  R5                              IF LAST TRANSFER NOT COMPLETED, SERVICE
2B2 000C00A442943   197             JMPC    IDLOOP2         BIT4            READ ROUTINE ALWAYS OR ON RDR ?
                    198
                    199                     RAM     DOUT            HOLDOUT LAST XFER DONE SO MOVE THIS BYTE OUT
2B3 140042A622B42   200             NEXT
                    201
                    202             INC.A   R5      R5      B               BUMP THE I/O INDEX TO STEP THRU THE XFER
2B4 000C14AC4A942   203             JMPU    IDLOOP2         BIT4            AND RETURN TO IDLOOP
                    204
                    205
                    206
                    207
                    208  OUTPUTB     PASS.A  R5                              IF THIS BYTE NOT TAKEN YET, SERVICE
2B5 000C00A442943   209             JMPC    IDLOOP2         BIT4            READ ROUTINE ALWAYS OR ON RDR ?
                    210
                    211                             CLDCO                   BYTE TAKEN SO CLEAR DATA COMMAND-OUT
2B6 0000500622B72   212             NEXT
                    213
                    214             A.SUB.S LIT     R5      B       #02     RECORD BYTE IS SENT
2B7 0080168CDAA42   215             JMPU    I.O.ADDR.4                      AND RETURN
```

```
                    110  ********************************************************************************
                    111  * THE I/O INDEX IN R5 HAS THE INFORMATION WETHER THE READ OR WRITE ROUTINES      *
                    112  * WANT TO SEND OR FETCH A BYTE FROM THE PPU. ALSO, ERRORS ARE LOGGED THERE       *
                    113  * THE TABLE DOES THE FETCH OR SEND AND ADJUSTS THE I/O INDEX APPROPRIATELY TO    *
                    114  * REFLECT THE STATE OF THE DATA TRANSFERS FROM THE PPU. SEE THE DESCRIPTION      *
                    115  * OF THE REGITERS A FEW PAGES BACK FOR DETAILS.                                  *
                    116  ********************************************************************************
                    117
                    118   * THE I/O INDEX TABLE MUST BEGIN AT ADDRESS #XX0
     02A0           119                ORG     *+#10 AND #7F0
                    120
                    121   I.O.ADDR.0   PASS.A  R5                          NO I/O TO PPU NESSESARY SO,
2A0 000C00A442942   122                JMPU    IDLOOP2      BIT4          SERVICE READ ROUTINE ALWAYS OR ON RDR ?
                    123
                    124                DEC.A   R5     R5    B             TOLD TO SEND A BYTE DURING A WRITE
2A1 000C14ACC2942   125                JMPU    IDLOOP2      BIT4          SO CANCELL THAT ORDER
                    126
                    127                NOP                                TOLD TO FETCH A BYTE DURING A WRITE
2A2 0034000622AC2   128                JMPU    INPUTA       NDFO          SO GO SEE IF A BYTE IS COMING FROM PPU
                    129
                    130                DEC.A   R5     R5    B             TOLD TO SEND AND FETCH A BYTE DURING A WRITE
2A3 003414ACC2AC2   131                JMPU    INPUTA       NDFO          SO CANCEL SEND AND DO FETCH
                    132
                    133
                    134
                    135
                    136   I.O.ADDR.4   PASS.A  R5                          NO I/O NECESSARY SO,
2A4 000C00A442942   137                JMPU    IDLOOP2      BIT4          SERVICE READ ROUINTE ALWAYS OR ON RDR ?
                    138
                    139                NOP                                TOLD TO SEND A BYTE DURING A READ SO
2A5 0016000622B22   140                JMPU    OUTPUTA      DFI           GO SEND IT
                    141
                    142                NOP                                HALF WAY THRU SENDING A BYTE SO
2A6 0036000622B52   143                JMPU    OUTPUTB      NDFI          GO FINISH SENDING IT
                    144
                    145                NOP                                1/2 THRU SENDING ONE, TOLD TO SEND ANOTHER
2A7 0036000622B52   146                JMPU    OUTPUTB      NDFI          SO FINISH SENDING THE FIRST
                    147
                    148
                    149
                    150
                    151   I.O.ADDR.8   PASS.A  R5                          NO I/O NECESSARY SO,
2A8 000C00A442942   152                JMPU    IDLOOP2      BIT4          SERVICE READ ROUTINE ALWAYS OR ON RDR ?
                    153
                    154                A.SUB.S LIT    R5    B     #01     TOLD TO SEND BYTE AFTER AN ERROR
2A9 0040168CDAA82   155                JMPU    I.O.ADDR.8                 SO CANCEL THAT ORDER
                    156
                    157                A.SUB.S LIT    R5    B     #02     TOLD TO FETCH A BYTE AFTER AN ERROR
2AA 0080168CDAB12   158                JMPU    INPUTB                     SO CANCEL THAT ORDER
                    159
                    160                A.SUB.S LIT    R5    B     #03     SEND AND FETCH A BYTE AFTER AN ERROR
2AB 00C0168CDAB12   161                JMPU    INPUTB                     SO CANCEL BOTH ORDERS
```

```
                    73  ********************************************************************************
                    74  * HERE IS THE IDLE LOOP WHICH CONTROLS READS AND WRITES OFF THE TAPE.          *
                    75  * IT PERFORMS 5 SIMULTANEOUS FUNCTIONS WHILE THE TAPE IS MOVING BY CALLING     *
                    76  * FOUR ROUTINES WHEN INSTRUCTED TO DO SO BY VARIOUS FLAGS. SOME ARE IN THE     *
                    77  * HARDWARE, SOME ARE STORED IN A REGISTER. THOSE 5 FUNCTIONS ARE :             *
                    78  *   1. SERVICE THE READ ROUTINES IF THEY REQUIRE IT                            *
                    79  *   2. MONITOR THE SIGNAL "DATA DETECETED OFF TAPE" IN ROUTINE "GAPFINDER"     *
                    80  *   3. DECREMENT A GENERAL PURPOSE TIMER - ALSO DONE BY GAPFINDER              *
                    81  *   4. SERVICE THE WRITE ROUTINES IF THEY REQUIRE IT                           *
                    82  *   5. MOVE DATA TO/FROM THE PPU AS NEEDED.                                   *
                    83  ********************************************************************************
                    84
                    85  IDLOOP2     NOP                                    ARE WE SUPPOSED TO SERVICE THE READ ROUTINES
294 0024000622963   86              JMPC    *+2         NRDR               ALWAYS OR WHEN "Read Data Ready" SAYS TO ?
                    87
                    88              NOP                                    IF WE GET HERE, THE READ ROUINE GETS SERVICED
295 0020000622962   89              JMPU    *+1         NEVER              EACH PASS THRU THIS IDLOOP
                    90
                    91              NOP                                    IF THE CONDITION BIT IS FALSE, THEN WE BRANCH
296 0000000622977   92              JMPZJ   *+1         ALWAYS             INTO THE READ ROUTINE WHERE "Z" POINTS TO
                    93
                    94  IDLOOP3     PASS.A  R6                             INSPECT R6 TO SEE IF WE ARE MONITORING DROPOUTS
297 000E00C442982   95              NEXT                BIT6
                    96
                    97              NOP
298 0001000624C33   98              JMPC    GAPFINDER   RTC                IF WE GO TO GAPFINDER, IS RTC SET ?
                    99
                   100  IDLOOP4     NOP                                    IS IT TIME TO SERVICE THE WRITE ROUTINES ?
299 0005000622942  101              NEXT                WDR                CHECK THE Write Data Ready SIGNAL
                   102
                   103  IDLOOP5     PASS.A  R5                             PUT THE I/O INDEX ON THE D-BUS
29A 000000A44000A  104              RETURNC             ALWAYS             GO TO WRITE ROUINES IF "Write Data Ready"
                   105
                   106              NOP                                    IT'S TIME TO DO I/O FROM/TO THE PPU
29B 0000000622A06  107              INDEXC  I.O.ADDR.0                     SO INDEX INTO I/O TABLE
```

```
60  ********************************************************************************
61  *    MORE READ/WRITE ROUTINE REGISTER USE                                      *
62  ********************************************************************************
63
64   * R8'S BITS :          (APPLIES ONLY IN GCR MODE - NOT USED IN MFM MODE)
65   *  #7 : AM CURRENTLY READING/CHECKING FOR AN XOR(+)/DATA(-) BLOCKETTE
66   *  #6 : THIS OPERATION IS A CRAMMED BYTE COUNT WITH BYTE COUNT = #0000
67   *  #5 : NOT USED
68   *  #4 : SEND FILLER BYTES TO PPU DURING NEXT SUNC ZONE SEARCH CUZ LAST BLOCKETTE WASN'T ALL SENT
69   *  #3 & #2 : COUNT OF ODD BLOCKETTE ERRORS
70   *  #1 & #0 : COUNT OF EVEN BLOCKETTE ERRORS
```

```
 5  ********************************************************************************
 6  * WITHIN THE READ AND WRITE ROUTINES, THE USE OF THE REGISTERS IS COMPLETELY     *
 7  * DIFFERENT, EXCEPT FOR R5, R6, AND R7 WHICH ARE USED TO COMMUNICATE WITH THE    *
 8  * MAIN ROUTINES. HERE'S HOW THE REGISTERS ARE USED.                              *
 9  ********************************************************************************
10
11   * Q  A GENERAL SCRATCH REGISTER AGAIN
12   * R0 SELECTED DRIVE'S STATUS FROM START-UP TILL LOAD POINT IS PASSED AND AT THE READ ROUTINE EXIT
13   * R3 IS OF COURSE STILL SARDL. SOMETMES THOUGH, THIS MAY BE USED AS A COUNTER TOO
14
15   * R1 WRITE ROUTINE'S LS COUNTER FOR ALMOST EVERYTHING
16   * R2 WRITE ROUTINE'S MS COUNTER FOR EVERYTHING WHICH REQUIRES 2 BYTES OF COUNTER
17   * R4 HOLDING REGISTER FOR BYTES FETCHED FROM THE PPU TO BE WRITTEN TO TAPE
18
19   * R9 LS COUNTER FOR READ ROUTINES
20   * RA MS COUNTER FOR THE READ ROUTINES
21   * RB HOLDER FOR A BYTE RED OR FOR WHAT NEXT BYTE TO BE RED SHOULD BE
22
23   * RC MS BYTE OF THE BLOCKETTE HEADER ID LATE COUNT, DECREMENTED BY GAPFINDER ONCE EACH BYTE TIME
24   * RC LS BYTE OF THE COUNTER RC,RD. ALSO TIMES DISTANCE FROM LEADING SYNC ZONE START TO ID FOUND
25   * RD WHERE GAPFINDER SHOULD EXIT WHEN GAP FOUND IS STORED HERE.
26   * RD -   THESE THREE BYTES ARE A GENERAL 2 OR 3 BYTE COUNTER TO TIMEOUT LONG THINGS LIKE
27   * RE 1 - LOAD POINT OR SYNC ZONE LATE, 1/3 INCH DROPOUT (OR I.R.G.) FOUND AND THE LIKE.
28   * RF -   RF IS ALWAYS THE MS BYTE, RE MAY BE MIDDLE OR LS BYTE. WHEN USED, RD IS THE LS BYTE
29
30   * RB IS USED AS A SCRATCH REGISTER IN THE START.DELAY ROUTINE. SORRY 'BOUT THAT ODDITY
31
32   * R5'S BITS :
33   *  #7 : PARITY ERROR DETECTED WHILE FETCHING DATA BYTES FROM PPU (+)
34   *  #6 : OVERFLOW ERROR DETECTED WHILE SENDING BYTES TO PPU (+)
35   *  #5 : UNDERFLOW ERROR DETECTED WHILE FETCHING BYTES FROM THE PPU (+)
36   *  #4 : SERVICE READ ROUTINES :  ON RDR (+) / EACH PASS THROUGH IDLOOP (-)
37   *  #3 - #0 : HANDLE DATA TRANSFERS FROM/TO THE PPU IN AN INDEX TABLE. GENERALLY :
38   *    #3 : AN ERROR HAS OCCURED. STOP TALKING TO THE PPU
39   *    #2 : OKAY TO SEND DATA TO THE PPU
40   *    #1 : FETCH A BYTE OR HALF DONE SENDING A BYTE FROM/TO THE PPU (DEPENDING ON WRITE OR READ)
41   *         IF WRITE, THE BYTE IS FETCHED WHEN THIS BIT IS CLEAR AGAIN.
42   *    #0 : SEND A BYTE TO THE PPU - BYTE IS SENT WHEN THIS IS CLEAR AGAIN
43
44   * R6'S BITS :
45   *  #7 : BYTE COUNT IN SECONDARY STATUS IS VALID FOR RECORD BEHIND ME (+)
46   *  #6 : MONITOR THE DATA DETECTED SIGNAL FOR DROPOUTS OR I.R.G.'S (+)
47   *  #5 : THIS IS A WRITE (+) / NOT WRITE (-)
48   *  #4 : THE WRITE ROUTINE IS STILL WRITING (+) / DONE (-)
49   *  #3 : WRITE/LOOK FOR : A FILE MARK (+) / DATA (-)
50   *  #2 : TAPE IS MOVING FORWARD (+) / REVERSE(-)
51   *  #1 : THIS IS A GCR READ WITH BYTE COUNT CRAMMED (+)
52   *  #0 : THE RECORD LENGTH IS ODD (+) / EVEN (-)  (COUNTED IN BYTES - MFM ONLY)
53
54   * R7'S BITS :
55   *  #7 : THE RECORD I JUST RED WAS A FILE MARK (+)
56   *  #6 & #5 : ENCODED TAPE MOVEMENT. SEE PRMSTS2 FOR DETAILS
57   *  #4 - #0 : ENCODED ERRORS. LOOK NEAR THE END OF THE LISTING FOR DETAILS
58   *            NOTE R7 IS IDENTICAL TO PRMSTS2
```

```
                    2746  ************************************************************************************
                    2747  * FINALLY, A FEW SUB ROUTINES USED EXCLUSIVELY BY THE READ AND WRITE ROUTINES     *
                    2748  ************************************************************************************
                    2749
                    2750
                    2751
                    2752   * THIS ONE DETECTS EOT, BOT, AND OFFLINE. THE SYNC ZONE SEARCH USES THIS ONE.
                    2753
                    2754
                    2755   CHKOFFLINE1  PASS.A   R0                          INSPECT BIT "IS DRIVE OFFLINE ?"
288 002D000442892   2756                NEXT                 NBIT5
                    2757
                    2758                PASS.A   R6                          INSPECT BIT "FORWARD OR REVERSE ?"
289 002A00C4428E3   2759                JMPC     OFFLINE1    NBIT2           ERROR IF DRIVE NO LONGER READY
                    2760
                    2761   FOR.1        PASS.A   R0                          INSPECT DRIVE STATUS IN R0 FOR EOT
28A 000B000442BD3   2762                JMPC     REV.1       BIT3            GO INSPECT BOT IF REVERSE
                    2763
                    2764                NOP                                  RETURN IF TAPE STILL INSTALLED AND THE END
28B 000000062000A   2765                RETURNC                              WASN'T FOUND (DEPENDS ON FORWARD OR REVERSE)
                    2766
                    2767                S.IOR.A  LIT    R7   B    #60        FLAG "TAPE POSITION LOST"
28C 18001E8DD4ECB   2768                JMPOPC   ERROR.9                     FLAG "SYNC ZONE LATE" EVEN THOUGH IT'S A LIE
                    2769
                    2770   REV.1        PASS.A   R0                          INSPECT DRIVE STATUS IN R0 FOR BOT
28D 000900442BB2    2771                JMPU     *-2         BIT1            SKIP BACK TO CONDITIONAL RETURN
                    2772
                    2773   OFFLINE1     NOP                                  THE TAPE CARTRIDGE IS NO LONGER INSTALLED
28E 0000006228C2    2774                JMPU     *-2                         SKIP BACK TO RECORD ERROR
                    2775
                    2776
                    2777
                    2778
                    2779   * THIS ONE DECREMENTS A 1, 2, OR 3 BYTE COUNTER, RETURN CONDITION = FALSE WHEN THE COUNTER ROLLS OVER
                    2780   * RF IS THE MS BYTE OF THE COUNT
                    2781
                    2782
                    2783   DEC.3BYTES   DEC.A    RD   RD    B
28F 001935ACC2902   2784                NEXT                 NBORROW
                    2785                NOP
290 000000062000A   2786                RETURNC              ALWAYS
                    2787   DEC.2BYTES   DEC.A    RE   RE    B
291 001939CCC2922   2788                NEXT                 NBORROW
                    2789                NOP
292 000000062000A   2790                RETURNC              ALWAYS
                    2791   DEC.1BYTE    DEC.A    RF   RF    B
293 00193DECC000A   2792                RETURNC              NBORROW
                    2793
                    2794
                    2795
                    2796  ************************************************************************************
                    2797  * NOW WE GET TO THE FUN PART. FOLLOWING IS AN ENTIRELY DIFFERENT CODE              *
                    2798  * STRUCTURE WITH IT'S OWN IDLE LOOP AND ROUTINES TO HANDLE READING AND             *
                    2799  * WRITING FROM/TO THE TAPE WHILE MONITORING DROP-OUTS AND TRANSFERING              *
                    2800  * DATA TO/FROM THE PPU. FIRST SOME DESCRIPTION OF THE REGISTER USE.                *
                    2801  ************************************************************************************
                       2                END
```

```
                    2391  ********************************************************************************
                    2392  * HERE IS THE CODE TO RECORD ALL THE ENCODED ERRORS AND GO DO THE NECESSARY      *
                    2393  * RECOVERY.                                                                      *
                    2394  ********************************************************************************
                    2395
                    2396   ERROR.1      S.IOR.A  LIT     R7      B     #01          ILLEGAL DRIVE
4E4 00401E8DD1782   2397                JMPU     UPDATE.STS
                    2398   ERROR.2      S.IOR.A  LIT     R7      B     #02          ILLEGAL STORE
4E5 00801E8DD1782   2399                JMPU     UPDATE.STS
                    2400   ERROR.3      S.IOR.A  LIT     R7      B     #03          ILLEGAL TRACK
4E6 00C01E8DD1782   2401                JMPU     UPDATE.STS
                    2402   ERROR.4      S.IOR.A  LIT     R7      B     #04          ILLEGAL TAPE MOTION
4E7 01001E8DD1782   2403                JMPU     UPDATE.STS
                    2404   ERROR.5      S.IOR.A  LIT     R7      B     #05          NO GO BYTE
4E8 01401E8DD1782   2405                JMPU     UPDATE.STS
                    2406   ERROR.6      S.IOR.A  LIT     R7      B     #06          TIMEOUT DURING F.F. OR REWIND
4E9 01801E8DD5002   2407                JMPU     STOP.NOW
                    2408   ERROR.7      S.IOR.A  LIT     R7      B     #07          OFFLINE DURING F.F. OR REWIND OR WRITE GAP
4EA 01C01E8DD5002   2409                JMPU     STOP.NOW
                    2410   ERROR.8      S.IOR.A  LIT     R7      B     #08          LOAD POINT HOLE LATE
4EB 02001E8DD5002   2411                JMPU     STOP.NOW
                    2412   ERROR.9      S.IOR.A  LIT     R7      B     #09          SYNC ZONE LATE
4EC 02401E8DD5002   2413                JMPU     STOP.NOW
                    2414   ERROR.A      S.IOR.A  LIT     R7      B     #0A          I.R.G. FOUND WHEN NOT EXPECTED
4ED 02801E8DD5002   2415                JMPU     STOP.NOW
                    2416   ERROR.B      S.IOR.A  LIT     R7      B     #0B          DROPOUT DURING WRITE
4EE 02C01E8DD5002   2417                JMPU     STOP.NOW
                    2418   ERROR.C      S.IOR.A  LIT     R7      B     #0C          I.D. ZONE LATE
4EF 03001E8DD4F82   2419                JMPU     STOP.INGAP
                    2420   ERROR.D      S.IOR.A  LIT     R7      B     #0D          LEADING I.D. ZONE BAD
4F0 03401E8DD4F82   2421                JMPU     STOP.INGAP
                    2422   ERROR.E      S.IOR.A  LIT     R7      B     #0E          WORD COUNT CHECK BAD
4F1 03801E8DD4F82   2423                JMPU     STOP.INGAP
                    2424   ERROR.F      S.IOR.A  LIT     R7      B     #0F          DATA CRC BAD (MFM ONLY)
4F2 03C01E8DD4F82   2425                JMPU     STOP.INGAP
                    2426   ERROR.10     S.IOR.A  LIT     R7      B     #10          REVERSE WORD COUNT CHECK BAD (GCR ONLY)
4F3 04001E8DD4F82   2427                JMPU     STOP.INGAP
                    2428   ERROR.11     S.IOR.A  LIT     R7      B     #11          TRAILING I.D. ZONE BAD
4F4 04401E8DD4F82   2429                JMPU     STOP.INGAP
                    2430   ERROR.12     S.IOR.A  LIT     R7      B     #12          TRAILING SYNC ZONE BAD (LAST 8 BITS NOT CHECKED)
4F5 04801E8DD4F82   2431                JMPU     STOP.INGAP
                    2432   ERROR.13     S.IOR.A  LIT     R7      B     #13          GCR BYTE COUNT NOT MATCH BLOCKETTE NUMBER
4F6 04C01E8DD4F82   2433                JMPU     STOP.INGAP
                    2434   ERROR.14     S.IOR.A  LIT     R7      B     #14          GCR CRAMMED BYTE COUNT CANNOT BE FOUND
4F7 05001E8DD4F82   2435                JMPU     STOP.INGAP
                    2436   ERROR.15     S.IOR.A  LIT     R7      B     #15          GCR READ NOT ALL RIGHT BUT RECOVERABLE
4F8 05401E8DD4F82   2437                JMPU     STOP.INGAP
                    2438   ERROR.16     S.IOR.A  LIT     R7      B     #16          GCR RECORD NOT RECOVERABLE
4F9 05801E8DD4F82   2439                JMPU     STOP.INGAP
                    2440   ERROR.17     S.IOR.A  LIT     R7      B     #17          EOT FOUND WHILE ERASING A FEW INCHES
4FA 05C01E8DD5002   2441                JMPU     STOP.NOW
                    2442
                    2443   * ERROR.7    IS FLAGGED IN CHKOFFLINE2 WHEN OFFLINE HAPPENS
                    2444   * ERROR.14   MAY BE FLAGGED IN THE GAPFINDER IF A GAP IS FOUND BEFORE THE CRAMMED BYTE COUNT IS FOUND
                    2445   * ERROR.16   IS FLAGGED IN ERRORBIT IF ERROR 15 OCCURS WITH AN OVERFLOW
                    2446   * ERROR.17   IS FLAGGED IN CHKOFFLINE2 WHEN EOT HAPPENS
                    2447   * ERROR.18   IS FLAGGED IN CHKOFFLINE2 WHEN BOT HAPPENS
                    2448   * ERROR.1F   IS FLAGGED IN THE WRITE CALLING ROUTINE IF IT WAS A WRITE REPEAT
```

```
                    2451  ********************************************************************************
                    2452  * IT'S TIME TO STOP THE TAPE. IMMEDIATELY IF WRITE ERROR OR SUCCESSFUL FORWARD   *
                    2453  * READ, IN NEXT FOUND GAP IF REVERSE OR READ ERROR. IF WRITE REPEAT, DO IT.      *
                    2454  ********************************************************************************
                    2455
                    2456   REV.STOP     PASS.A   R6                              INSPECT BIT "WRITE OR NOT WRITE ?"
4FB 000D00C444FC2   2457   STOP.INGAP  NEXT                    BIT5
                    2458
                    2459                LIT    RDCON       #80                  TURN RDR OFF
4FC 20006E8625043   2460                JMPC   STOP.WRT                         GO STOP THIS WRITE IF IT'S A WRITE
                    2461
                    2462                RDATA                                   CLEAR RDR
4FD 000002C624FE2   2463                NEXT
                    2464
                    2465                S.IOR.A LIT    R5      B    #10         SET BIT "SERVICE READ ON RDR ONLY"
4FE 0400168DD4FF2   2466                NEXT
                    2467
                    2468                PASS.S  LIT    RD      B    #01         INSTRUCT GAPFINDER TO STOP IN NEXT GAP
4FF 0040368C72972   2469                JMPU    IDLOOP3                         AND GO DO IT
                    2470
                    2471
                    2472
                    2473
                    2474   FOR.STOP     PASS.A   R6                              INSPECT BIT "WRITE OR NOT WRITE ?"
500 000D00C445012   2475   STOP.NOW     NEXT                    BIT5
                    2476
                    2477                LIT    RDCON       #80                  TURN OFF RDR
501 20006E8625043   2478                JMPC   STOP.WRT                         GO STOP THIS WRITE IF IT'S A WRITE
                    2479
                    2480   STOP.RD                  LIT    DRCON       #00     STOP THE DRIVE
502 00006686224C1   2481                CALLC  WAIT.35.MS                       GO WAIT FOR THE TAPE TO STOP
                    2482
                    2483                PASS.S  DRSTS  R0      B                LOAD THE LATEST DRIVE STATUS INTO R0
503 0000024C7000A   2484                RETURNC                                 AND RETURN TO THE CALLING ROUTINES
                    2485
                    2486   STOP.WRT     PASS.A   R6                              INSPECT BIT "WRITE STILL GOING ?"
504 000C00C445052   2487                NEXT                    BIT4
                    2488
                    2489                PASS.S  RAM             #FF             INSPECT RAM FLAG "IS THIS A WRITE REPEAT ?"
505 3FF802A47506B   2490                JMPOPC  *+1             NZERO           MAINTAIN STACK IF WRITE IS ABORTED
                    2491
                    2492                LIT    WRCON       #26                  FORCE WRITE OF A.C. ERASE IN CASE NOT ALREADY SO
506 09807286235F3   2493                JMPC   REPEAT                           BEGIN ALL OVER AGAIN IF THIS IS A WRITE REPEAT
                    2494
                    2495                LIT    DRCON       #08                  STOP THE DRIVE WITH WRITE STILL ENABLED
507 02006686224C1   2496                CALLC  WAIT.35.MS                       WAIT FOR THE TAPE TO STOP WHILE ERASING
                    2497
                    2498                LIT    DRCON       #00                  TURN OFF THE ERASE
508 0000668622501   2499                CALLC  WAIT.5.MS                        WAIT FOR THE ERASE HEAD TO WIND DOWN
                    2500
                    2501                PASS.S  DRSTS  R0      B                LOAD THE LATEST DRIVE STATUS INTO R0
509 0000024C7000A   2502                RETURNC                                 AND RETURN TO THE CALLING ROUTINES
                    2503
                    2504
                    2505  ********************************************************************************
                    2506  * THAT'S THE END OF THE FUNCTIONAL MICROCODE. FOLLOWING IS THE SUBROUTINE        *
                    2507  * "SELF.TEST" WHICH IS ABOUT THE FIRST THING THIS CODE DOES AFTER WAKING UP.      *
                    2508  ********************************************  ****************************************
```

```
                    2511  ********************************************************************************
                    2512  *      THIS TEST CHECKS OUT THE D-BUS. PATTERNS OF ONES AND ZEROS ARE PUT       *
                    2513  * THROUGH THE LITERAL FIELD AND ON TO THE D-BUS. THE PATTERNS ARE CHECKED BY     *
                    2514  * THE D-BUS CONDITIONAL JUMP LOGICS.                                            *
                    2515  ********************************************************************************
                    2516
                    2517   SELF.TEST            LIT                 #01          TEST PATTERN = 01
50A 0068028625082   2518   DBUSTEST     NEXT                  NBIT0             SET UP NBIT0
                    2519                LIT                 #02               TEST PATTERN = 02
50B 00A9028625103   2520                JMPC     DBUSTESERROR  NBIT1             GO REPORT ERROR IF BIT0 IS NOT SET. SET UP NBIT1
                    2521                LIT                 #04               TEST PATTERN = 04
50C 012A028625103   2522                JMPC     DBUSTESERROR  NBIT2             GO REPORT ERROR IF BIT1 IS NOT SET. SET UP NBIT2
                    2523                LIT                 #08               TEST PATTERN = 08
50D 022B028625103   2524                JMPC     DBUSTESERROR  NBIT3             GO REPORT ERROR IF BIT2 IS NOT SET. SET UP NBIT3
                    2525                LIT                 #10               TEST PATTERN = 10
50E 042C028625103   2526                JMPC     DBUSTESERROR  NBIT4             GO REPORT ERROR IF BIT3 IS NOT SET. SET UP NBIT4
                    2527                LIT                 #20               TEST PATTERN = 20
50F 082D028625103   2528                JMPC     DBUSTESERROR  NBIT5             GO REPORT ERROR IF BIT4 IS NOT SET. SET UP NBIT5
                    2529                LIT                 #40               TEST PATTERN = 40
510 102E028625103   2530                JMPC     DBUSTESERROR  NBIT6             GO REPORT ERROR IF BIT5 IS NOT SET. SET UP NBIT6
                    2531                LIT                 #80               TEST PATTERN = 80
511 202F028625103   2532                JMPC     DBUSTESERROR  NBIT7             GO REPORT ERROR IF BIT6 IS NOT SET. SET UP NBIT7
                    2533                LIT                 #FE               TEST PATTERN = FE
512 3F8802862510B   2534                JMPC     DBUSTESERROR  BIT0              GO REPORT ERROR IF BIT7 IS NOT SET SET UP BIT0
                    2535                LIT                 #FD               TEST PATTERN = FD
513 3F4902862510B   2536                JMPC     DBUSTESERROR  BIT1              GO REPORT ERROR IF BIT0 IS SET. SET UP BIT1
                    2537                LIT                 #FB               TEST PATTERN = FB
514 3ECA02862510B   2538                JMPC     DBUSTESERROR  BIT2              GO REPORT ERROR IF BIT1 IS SET. SET UP BIT2
                    2539                LIT                 #F7               TEST PATTERN = F7
515 3DCB02862510B   2540                JMPC     DBUSTESERROR  BIT3              GO REPORT ERROR IF BIT2 IS SET. SET UP BIT3
                    2541                LIT                 #EF               TEST PATTERN = EF
516 3BCC02862510B   2542                JMPC     DBUSTESERROR  BIT4              GO REPORT ERROR IF BIT3 IS SET. SET UP BIT4
                    2543                LIT                 #DF               TEST PATTERN = DF
517 37CD02862510B   2544                JMPC     DBUSTESERROR  BIT5              GO REPORT ERROR IF BIT4 IS SET. SET UP BIT5
                    2545                LIT                 #BF               TEST PATTERN = BF
518 2FCE02862510B   2546                JMPC     DBUSTESERROR  BIT6              GO REPORT ERROR IF BIT5 IS SET. SET UP BIT6
                    2547                LIT                 #7F               TEST PATTERN = 7F
519 1FCF02862510B   2548                JMPC     DBUSTESERROR  BIT7              GO REPORT ERROR IF BIT6 IS SET. SET UP BIT7
                    2549
                    2550                LIT                 *+4 AND #F        SET UP INDEX TO *+4
51A 03800286251C3   2551                JMPC     DBUSTESERROR
                    2552
                    2553                NOP
518 0000000625106   2554                INDEXC   *+3 AND #7F0                  SHOULD INDEX TO *+3 CUZ OF D-BUS IN *-1
                    2555
                    2556   DBUSTESERROR PASS.S   LIT    R1     B     #01       STORE ERROR BIT IN R1
51C 0040068C751D2   2557                NEXT
                    2558
                    2559                PASS.A   R1     RAM          ID3
51D 04405C244000A   2560                RETURNC
```

```
                    2562  ********************************************************************************
                    2563  *     THIS TEST CHECKS OUT THE 2910 SEQUENCER.                                 *
                    2564  *      WE PUSH AND POP THE STACK 4 TIMES AND CHECK THE STKFULL CONDITION BIT.   *
                    2565  ********************************************************************************
                    2566
                    2567                NOP
51E 0000000625201   2568  TEST2.2910    CALLC    TEST2PUSH1                    1 OF 4 PUSHES
                    2569
                    2570                NOP
51F 00000006252F3   2571                JMPC     UPTEST                        GO BEGIN 2901 UP TEST
                    2572
                    2573                NOP
520 0000000625231   2574  TEST2PUSH1    CALLC    TEST2PUSH2                    2 OF 4 PUSHES
                    2575
                    2576                NOP
521 001C00062000A   2577                RETURNC                NSTKFULL       POP 4 OF 4 POPS
                    2578
                    2579                NOP
522 0000000625202   2580                JMPU     PUSHPOPERROR                  STKFULL CONDITION IS WRONG
                    2581
                    2582                NOP
523 0000000625261   2583  TEST2PUSH2    CALLC    TEST2PUSH3                    3 OF 4 PUSHES
                    2584
                    2585                NOP
524 001C00062000A   2586                RETURNC                NSTKFULL       POP 3 OF 4 POPS
                    2587
                    2588                NOP
525 0000000625202   2589                JMPU     PUSHPOPERROR                  STKFULL CONDITION IS WRONG
                    2590
                    2591                NOP
526 00000006252A1   2592  TEST2PUSH3    CALLC    TEST2PUSH4                    4 OF 4 PUSHES
                    2593
                    2594                NOP
527 001C000625282   2595                NEXT                   NSTKFULL       WAIT FOR THE NSTKFULL CONDITION BIT TO SET
                    2596
                    2597                NOP
528 001C00062000A   2598                RETURNC                NSTKFULL       POP 2 OF 4 POPS
                    2599
                    2600                NOP
529 0000000625202   2601                JMPU     PUSHPOPERROR                  STKFULL CONDITION IS WRONG
                    2602
                    2603                NOP
52A 003C000625282   2604  TEST2PUSH4    NEXT                   STKFULL        WAIT FOR THE STKFULL CONDITION BIT TO SET
                    2605
                    2606                NOP
52B 001C00062000A   2607                RETURNC                NSTKFULL       POP 1 OF 4 POPS  STKFULL GOES LOW AFTER THIS POP
                    2608
                    2609                NOP
52C 0000000625202   2610                JMPU     PUSHPOPERROR                  STKFULL CONDITION IS WRONG
                    2611
                    2612  PUSHPOPERROR  PASS.S   LIT    R1     B    #08        SET UP ERROR BIT
52D 0200068C752E2   2613                NEXT
                    2614
                    2615                PASS.A   R1     RAM         ID3        PUT THE ERROR IN RAM
52E 04405C244000A   2616                RETURNC                               RETURN TO THE MAIN CODE
```

```
                    2618  ************************************************************************************
                    2619  * THIS TEST CHECKS OUT THE 2901 PROCESSORS BY WRITING AN INCREMENTING              *
                    2620  * PATTERN INTO ALL REGISTERS                                                       *
                    2621  ************************************************************************************
                    2622
                    2623   UPTEST       ZERO            RO     B                INITIALLY WRITE THE PATTERN "#00"
52F 0000000E25302   2624                NEXT
                    2625
                    2626   PATTERNLOJP  PASS.A   RO     R1     B                STORE RO TO R1
530 0000040C45361   2627                CALLC    PATTERN                        PUT 00..FF INTO R1..RF
                    2628
                    2629                A.XOR.B  RO     R1                      COMPARE RO TO R1
531 0038040715451   2630                CALLC    CHKPATTERN    NZERO            CHECK THE CONTENTS OF R1..RF
                    2631
                    2632                NOP
532 0000000625303   2633                JMPC     PATTERNLOOP                    IF DIDN'T PASS FF GO DO AGAIN
                    2634
                    2635                NOP
533 0000000625572   2636                JMPU     ADDRESSTEST
                    2637
                    2638   * HERE'S WHERE THE ERROR IS RECORDED FOR THE ADDRESS TEST COMING UP AFTER THE PATTERN AND
                    2639   * THE CHKPATTERN ROUTINES
                    2640
                    2641   ADRESERROR   PASS.S   LIT    R1     B    #10         SET UP ERROR BIT
534 0400068C75352   2642                NEXT
                    2643
                    2644                PASS.A   R1     RAM         ID3         PUT THE ERROR IN RAM
535 04405C244000A   2645                RETURNC                                 RETURN TO MAIN CODE
```

```
                    2647  ********************************************************************************
                    2648  *        THIS ROUTINE WRITES THE PATTERN THAT I STORE IN R0 TO R1..RF.          *
                    2649  ********************************************************************************
                    2650
                    2651   PATTERN     PASS.A   R1     R2     B                R2=R1
536 0000082C45372   2652               NEXT
                    2653
                    2654               PASS.A   R2     R3     B                R3=R2
537 00000C4C45382   2655               NEXT
                    2656
                    2657               PASS.A   R3     R4     B                R4=R3
538 0000106C45392   2658               NEXT
                    2659
                    2660               PASS.A   R4     R5     B                R5=R4
539 0000148C453A2   2661               NEXT
                    2662
                    2663               PASS.A   R5     R6     B                R6=R5
53A 000018AC453B2   2664               NEXT
                    2665
                    2666               PASS.A   R6     R7     B                R7=R6
53B 00001CCC453C2   2667               NEXT
                    2668
                    2669               PASS.A   R7     R8     B                R8=R7
53C 000020EC453D2   2670               NEXT
                    2671
                    2672               PASS.A   R8     R9     B                R9=R8
53D 0000250C453E2   2673               NEXT
                    2674
                    2675               PASS.A   R9     RA     B                RA=R9
53E 0000292C453F2   2676               NEXT
                    2677
                    2678               PASS.A   RA     RB     B                RB=RA
53F 00002D4C45402   2679               NEXT
                    2680
                    2681               PASS.A   RB     RC     B                RC=RB
540 0000316C45412   2682               NEXT
                    2683
                    2684               PASS.A   RC     RD     B                RD=RC
541 0000358C45422   2685               NEXT
                    2686
                    2687               PASS.A   RD     RE     B                RE=RD
542 000039AC45432   2688               NEXT
                    2689
                    2690               PASS.A   RE     RF     B                RF=RE
543 00003DCC45442   2691               NEXT
                    2692
                    2693               PASS.A   RF            Q                Q=RF
544 000001E04000A   2694               RETURNC
```

```
                    2696  ************************************************************************************
                    2697  *      THIS ROUTINE WILL COMPARE THE CONTENTS OF R1 TO RF TO A REFERENCE IN R0   *
                    2698  ************************************************************************************
                    2699
                    2700   CHKPATTERN   A.XOR.B  R0     R2                        COMPARE R2 WITH R0
545 00380807159FB   2701                JMPOPC   UPTESTERROR   NZERO              GO REPORT ERROR IF FAILED
                    2702
                    2703                A.XOR.B  R0     R3                        NOW COMPARE R3 WITH R0
546 00380C07159FB   2704                JMPOPC   UPTESTERROR   NZERO
                    2705
                    2706                A.XOR.B  R0     R4                        NOW COMPARE R4 WITH R0
547 00381007159FB   2707                JMPOPC   UPTESTERROR   NZERO
                    2708
                    2709                A.XOR.B  R0     R5                        NOW COMPARE R5 WITH R0
548 00381407159FB   2710                JMPOPC   UPTESTERROR   NZERO
                    2711
                    2712                A.XOR.B  R0     R6                        NOW COMPARE R6 WITH R0
549 00381807159FB   2713                JMPOPC   UPTESTERROR   NZERO
                    2714
                    2715                A.XOR.B  R0     R7                        COMPARE R7 WITH R0
54A 00381C07159FB   2716                JMPOPC   UPTESTERROR   NZERO
                    2717
                    2718                A.XOR.B  R0     R8                        COMPARE R8 WITH R0
54B 00382007159FB   2719                JMPOPC   UPTESTERROR   NZERO
                    2720
                    2721                A.XOR.B  R0     R9                        COMPARE R9 WITH R0
54C 00382407159FB   2722                JMPOPC   UPTESTERROR   NZERO
                    2723
                    2724                A.XOR.B  R0     RA                        COMPARE RA WITH R0
54D 00382807159FB   2725                JMPOPC   UPTESTERROR   NZERO
                    2726
                    2727                A.XOR.B  R0     RB                        COMPARE RB WITH R0
54E 00382C07159FB   2728                JMPOPC   UPTESTERROR   NZERO
                    2729
                    2730                A.XOR.B  R0     RC                        COMPARE RC WITH R0
54F 00383007159FB   2731                JMPOPC   UPTESTERROR   NZERO
                    2732
                    2733                A.XOR.B  R0     RD                        COMPARE RD WITH R0
550 00383407159FB   2734                JMPOPC   UPTESTERROR   NZERO
                    2735
                    2736                A.XOR.B  R0     RE                        COMPARE RE WITH R0
551 00383807159FB   2737                JMPOPC   UPTESTERROR   NZERO
                    2738
                    2739                A.XOR.B  R0     RF                        COMPARE RF WITH R0
552 00383C07159FB   2740                JMPOPC   UPTESTERROR   NZERO
                    2741
                    2742                A.XOR.Q  R0                               COMPARE Q WITH R0
553 00180007059FB   2743                JMPOPC   UPTESTERROR   ZERO
                    2744
                    2745                INC.A    R0     R0     B                 INCREMENT THE TEST PATTERN
554 0039000C4800A   2746                RETURNC                NCARRY            THE CONTENTS OF ALL THE REGISTERS CHECKED OUT
                    2747
                    2748                NOP                                      WAIT FOR CONDITION BIT TO BE TRUE
555 0000000625562   2749                NEXT                   ALWAYS
                    2750
                    2751                NOP                                      AND RECORD LAST TEST FAILED
556 00000006259FB   2752                JMPOPC   UPTESTERROR
```

```
                    2754  ************************************************************************************
                    2755  * THIS TEST IS A COMPLIMENT ADDRESS TEST ON THE 2901 REGISTERS                     *
                    2756  ************************************************************************************
                    2757
                    2758   ADDRESSTEST  PASS.S   LIT    R0     B    #FF          BEGIN BY WRITING THE COMPLIMENT
557 3FC0028C75582   2759                NEXT                                     OF EACH REGISTER'S ADDRESS INTO IT
                    2760
                    2761                PASS.S   LIT    R1     B    #FE
558 3F80068C75592   2762                NEXT
                    2763
                    2764                PASS.S   LIT    R2     B    #FD
559 3F400A8C755A2   2765                NEXT
                    2766
                    2767                PASS.S   LIT    R3     B    #FC
55A 3F000E8C755B2   2768                NEXT
                    2769
                    2770                PASS.S   LIT    R4     B    #FB
55B 3EC0128C755C2   2771                NEXT
                    2772
                    2773                PASS.S   LIT    R5     B    #FA
55C 3E80168C755D2   2774                NEXT
                    2775
                    2776                PASS.S   LIT    R6     B    #E9
55D 3E401A8C755E2   2777                NEXT
                    2778
                    2779                PASS.S   LIT    R7     B    #F8
55E 3E001E8C755F2   2780                NEXT
                    2781
                    2782                PASS.S   LIT    R8     B    #F7
55F 3DC0228C75602   2783                NEXT
                    2784
                    2785                PASS.S   LIT    R9     B    #F6
560 3D80268C75612   2786                NEXT
                    2787
                    2788                PASS.S   LIT    RA     B    #F5
561 3D402A8C75622   2789                NEXT
                    2790
                    2791                PASS.S   LIT    RB     B    #F4
562 3D002E8C75632   2792                NEXT
                    2793
                    2794                PASS.S   LIT    RC     B    #F3
563 3CC0328C75642   2795                NEXT
                    2796
                    2797                PASS.S   LIT    RD     B    #F2
564 3C80368C75652   2798                NEXT
                    2799
                    2800                PASS.S   LIT    RE     B    #F1
565 3C403A8C75662   2801                NEXT
                    2802
                    2803                PASS.S   LIT    RF     B    #F0
566 3C003E8C75672   2804                NEXT
                    2805
                    2806                PASS.S   LIT           Q    #A5
567 2940028075682   2807                NEXT
```

```
                    2809  ********************************************************************************
                    2810  * NOW TO READ ALL THOSE REGISTERS BACK AND SEE IF THEY WORK                     *
                    2811  ********************************************************************************
                    2812
                    2813                S.XOR.A  LIT    R0          #00          FIRST TEST THAT AN ERROR CAN BE DETECTED
568 0018028755692   2814                NEXT                    ZERO
                    2815
                    2816                S.XOR.A  LIT    R0          #FF          AND NOW TEST THAT ALL REGISTERS ARE RIGHT
569 3FF8028755343   2817                JMPC     ADRESERROR     NZERO
                    2818
                    2819                S.XOR.A  LIT    R1          #FE
56A 3F88068755343   2820                JMPC     ADRESERROR     NZERO
                    2821
                    2822                S.XOR.A  LIT    R2          #FD
568 3F780A8755343   2823                JMPC     ADRESERROR     NZERO
                    2824
                    2825                S.XOR.A  LIT    R3          #FC
56C 3F380E8755343   2826                JMPC     ADRESERROR     NZERO
                    2827
                    2828                S.XOR.A  LIT    R4          #FB
56D 3EF8128755343   2829                JMPC     ADRESERROR     NZERO
                    2830
                    2831                S.XOR.A  LIT    R5          #FA
56E 3E88168755343   2832                JMPC     ADRESERROR     NZERO
                    2833
                    2834                S.XOR.A  LIT    R6          #F9
56F 3E781A8755343   2835                JMPC     ADRESERROR     NZERO
                    2836
                    2837                S.XOR.A  LIT    R7          #F8
570 3E381E8755343   2838                JMPC     ADRESERROR     NZERO
                    2839
                    2840                S.XOR.A  LIT    R8          #F7
571 3DF8228755343   2841                JMPC     ADRESERROR     NZERO
                    2842
                    2843                S.XOR.A  LIT    R9          #F6
572 3DB8268755343   2844                JMPC     ADRESERROR     NZERO
                    2845
                    2846                S.XOR.A  LIT    RA          #F5
573 3D782A8755343   2847                JMPC     ADRESERROR     NZERO
                    2848
                    2849                S.XOR.A  LIT    RB          #F4
574 3D382E8755343   2850                JMPC     ADRESERROR     NZERO
                    2851
                    2852                S.XOR.A  LIT    RC          #F3
575 3CF8328755343   2853                JMPC     ADRESERROR     NZERO
                    2854
                    2855                S.XOR.A  LIT    RD          #F2
576 3CB8368755343   2856                JMPC     ADRESERROR     NZERO
                    2857
                    2858                S.XOR.A  LIT    RE          #F1
577 3C783A8755343   2859                JMPC     ADRESERROR     NZERO
                    2860
                    2861                S.XOR.A  LIT    RF          #F0
578 3C383E8755343   2862                JMPC     ADRESERROR     NZERO
                    2863
                    2864                S.XOR.Q  LIT                #A5
579 2978028765343   2865                JMPC     ADRESERROR     NZERO
```

```
                    2867  ********************************************************************************
                    2868  *     FOLLOWING CHECK THE SIGN, CARRY OUT, AND OVERFLOW BITS                   *
                    2869  ********************************************************************************
                    2870
                    2871   OVTEST       INC.S    LIT           Q     #7F          SET THE OVERFLOW CONDITION BIT
57A 1FFB02807D343   2872                JMPC     ADRESERROR    NOVFLOW
                    2873
                    2874  ********************************************************************************
                    2875  *       FOLLOWING IS A SERIES OF TEST TO CHECK THE INPUT LINES(I0-I8) OF        *
                    2876  *  THE 2901'S.                                                                  *
                    2877  ********************************************************************************
                    2878
                    2879                PASS.Q          RF      B                 PASS Q TEST
57B 00003C0C259F3   2880                JMPC     UPTESTERROR
                    2881
                    2882                S.XOR.A  LIT     RF           #80          RF SHOULD CONTAIN #80  PASSED FROM Q
57C 20383E87557D2   2883                NEXT                      NZERO
                    2884
                    2885                NOP
57D 00000006259F3   2886                JMPC     UPTESTERROR
                    2887
                    2888   OPTEST2      PASS.S   LIT     R1       B     #01          A.ADD1.Q TEST
57E 0040068C757F2   2889                NEXT
                    2890
                    2891                PASS.S   LIT              Q     #7F          Q=#7F
57F 1FC0028075802   2892                NEXT
                    2893
                    2894                A.AD1.Q  R1               Q                  CARRY SHOULD BE CLEAR
580 001900200D812   2895                NEXT                      CARRY
                    2896
                    2897                S.XOR.Q  LIT                    #81          COMPARE RESULT WITH EXPECTED VALUE
581 20780287659F3   2898                JMPC     UPTESTERROR   NZERO                REPORT ERROR IF TEST FAILED
                    2899
                    2900                PASS.S   LIT     R0       B     #00          R0=00
582 0000028C759F3   2901                JMPC     UPTESTERROR                         REPORT ERROR IF TEST FAILED
```

```
                    2903  ********************************************************************************
                    2904  *    FROM NOW ON, THE CONTENTS OF R0,R1,R2,R3,R4,R5,AND RF WILL CONTAIN        *
                    2905  * 00,01,02,A5,5A,FE AND FF. THESE REGISTERS WILL BE USED AS DATA AND TEST       *
                    2906  * RESULTS.                                                                      *
                    2907  ********************************************************************************
                    2908
                    2909              PASS.S   LIT    R1     B    #01          R1=01
583 0040068C75842   2910              NEXT
                    2911
                    2912              PASS.S   LIT    R2     B    #02          R2=02
584 00800A8C75852   2913              NEXT
                    2914
                    2915              PASS.S   LIT    R3     B    #A5
585 29400E8C75862   2916              NEXT
                    2917
                    2918              PASS.S   LIT    R4     B    #5A
586 1680128C75872   2919              NEXT
                    2920
                    2921              PASS.S   LIT    R5     B    #FE
587 3F80168C75882   2922              NEXT
                    2923
                    2924              PASS.S   LIT    RF     B    #FF
588 3FC03E8C75892   2925              NEXT
                    2926
                    2927  OPTEST4     A.ADD.B  R1     RF     B                A.ADD.B TEST
589 00393C2C158A2   2928              NEXT                   NCARRY           CARRY SHOULD BE SET
                    2929
                    2930  OPTEST5     PASS.A   R2            Q                Q.SB1.A TEST
58A 00000040459F3   2931              JMPC     UPTESTERROR
                    2932
                    2933              Q.SB1.A  R1            Q                2-1-1
58B 00000020858C2   2934              NEXT                                    STORE RESULT IN RA
                    2935
                    2936              A.XOR.Q  R0                             COMPARE Q WITH EXPECTED RESULT
58C 003800070580D2  2937              NEXT                   NZERO
                    2938
                    2939              PASS.S   LIT    RA     B    #4A          RA=#4A=EXPECTED RESULT
58D 12802A8C759F3   2940              JMPC     UPTESTERROR                    REPORT ERROR
                    2941
                    2942  SUBTEST5    B.SB1.A  R4     R3     Q                A5-5A-1
58E 00000C80958F2   2943              NEXT                                    STORE RESULT IN Q
                    2944
                    2945              A.XOR.Q  RA                             COMPARE RESULT
58F 0038014705902   2946              NEXT                   NZERO
```

```
                    2948
                    2949   OPTEST6      PASS.A   R3              Q                A.AND.Q TEST
590 00000060459F3   2950                JMPC     UPTESTERROR
                    2951
                    2952                A.AND.Q  R4      RB     B                RB=RESULT
591 00002C8E05922   2953                NEXT                                     R4=5A
                    2954
                    2955                A.XOR.B  RF      RB     B                COMPARE RESULT
592 00382DEF15932   2956                NEXT                    NZERO
                    2957
                    2958   OPTEST7      PASS.A   R1              Q                Q=R1=1  A.SB1.Q TEST
593 00000020459F3   2959                JMPC     UPTESTERROR                     REPORT ERROR
                    2960
                    2961                A.SB1.Q  R2              Q                02-01-1
594 0038004105952   2962                NEXT                    NZERO
                    2963
                    2964   OPTEST8      PASS.A   R0      RA     B.A              B.A TEST  RA=0
595 00382808459F3   2965                JMPC     UPTESTERROR    NZERO            REPORT ERROR
                    2966
                    2967   SUBTEST8     A.XOR.B  R0      RA                      COMPARE R0,RA
596 00382807159F3   2968                JMPC     UPTESTERROR    NZERO            REPORT ERROR
                    2969
                    2970   OPTEST9      PASS.S   LIT     RA     B    #AA         RA=#AA
597 2A802A8C759F3   2971                JMPC     UPTESTERROR
                    2972
                    2973                PASS.S   LIT             Q    #55         Q=55
598 1540028075992   2974                NEXT
                    2975
                    2976                PASS.S   LIT     RB     B    #55         RB=#55
599 15402E8C759A2   2977                NEXT
                    2978
                    2979                PASS.A   RB      RC     BRQR             Q=RC=#AA
59A 0000317045982   2980                NEXT
                    2981
                    2982                A.XOR.Q  RA
59B 00380147059C2   2983                NEXT                    NZERO
                    2984
                    2985                A.XOR.B  RA      RC                      COMPARE A,B
59C 00383147159F3   2986                JMPC     UPTESTERROR    NZERO            REPORT ERROR IF Q<> RA
                    2987
                    2988                NOP
59D 00000006259F3   2989                JMPC     UPTESTERROR
                    2990
                    2991                NOP
59E 0000000625A12   2992                JMPU     RAMTEST                         GO ON TO RAMTEST
                    2993
                    2994
                    2995   UPTESTERROR  PASS.S   LIT     R1     B    #02         STORE ERROR BIT IN R1
59F 0080068C75A02   2996                NEXT
                    2997
                    2998                PASS.A   R1      RAM          ID3         STORE ERROR BIT IN RAM'S ID3
5A0 04405C244000A   2999                RETURNC                                  RETURN BACK TO THE CODE
```

```
                    3001  ********************************************************************************
                    3002  * HERE WE DO AN ADDRESS TEST, COMPLIMENT ADDRESS TEST, ONES AND FINALLY        *
                    3003  * ZEROES TEST ON THE RAM                                                        *
                    3004  ********************************************************************************
                    3005
                    3006   RAMTEST      PASS.S   LIT    R0     B    #FF          LOAD STARTING PATTERN
5A1 3FC0028C75A22   3007   ADDRS.TEST   NEXT
                    3008
                    3009                PASS.S   LIT    R1     B    #01          LOAD DECREMENTER FOR ADDRESS TEST
5A2 0040068C75AA1   3010                CALLC    CHECK.RAM                       GO TEST THE RAM
                    3011
                    3012   NADDRS.TEST  PASS.S   LIT    R0     B    #00          LOAD STARTING PATTERN FOR COMPLIMENT ADDRESS TEST
5A3 0000028C75A42   3013                NEXT
                    3014
                    3015                PASS.S   LIT    R1     B    #FF          LOAD DECREMENTER
5A4 3FC0068C75AA1   3016                CALLC    CHECK.RAM                       GO TEST THE RAM
                    3017
                    3018   ONES         PASS.S   LIT    R0     B    #01          LOAD STARTING PATTERN INTO R0
5A5 0040028C75A62   3019                NEXT
                    3020
                    3021                PASS.S   LIT    R1     B    #00          LOAD DECREMENTER
5A6 0000068C75AA1   3022                CALLC    CHECK.RAM                       GO TEST THE RAM
                    3023
                    3024   ZEROES       PASS.S   LIT    R0     B    #00          LOAD STARTING PATTERN INTO R0
5A7 0000028C75A82   3025                NEXT
                    3026
                    3027                PASS.S   LIT    R1     B    #00          LOAD DECREMENTER
5A8 0000068C75AA1   3028                CALLC    CHECK.RAM                       GO TEST THE RAM
                    3029
                    3030                NOP                                      THIS TEST IS FINISHED
5A9 0000000625BB2   3031                JMPU     TEST8                           GO DO THE NEXT TEST
```

```
                    3033  ********************************************************************************
                    3034  * HERE'S THE CODE TO DO THE WRITES AND READS AND CHECKS OF WHAT'S IN THE RAM.    *
                    3035  * R0 IS THE STARTING PATTERN, R1 IS WHAT TO SUBTRACT FROM THAT PATTERN FOR       *
                    3036  * EACH NEXT RAM ADDRESS TO WRITE/READ.                                          *
                    3037  ********************************************************************************
                    3038
                    3039   CHECK.RAM    PASS.S   LIT     SADRH  B     #03          LOAD PAGE 3 INTO RAM POINTER
5AA 00C05A8C75AB2   3040   WRITE.RAM    NEXT
                    3041
                    3042                PASS.S   LIT     SADRL  B     #FF          POINT TO END OF PAGE 3
5AB 3FC04E8C75AC2   3043                NEXT
                    3044
                    3045                PASS.A   R0      R2     B                 COPY STARTING PATTERN IN R2
5AC 0000080C45AD2   3046                NEXT
                    3047
                    3048   LOOP.W       PASS.A   R0      SCR                      WRITE INTO RAM
5AD 0000DC0445AE2   3049                NEXT
                    3050
                    3051                DEC.A    R3      SADRL  B                 DECREMENT POINTER INTO RAM
5AE 00194C6CC5AF2   3052                NEXT                    NBORROW
                    3053
                    3054                B.SUB.A  R1      R0     B                 CHANGE PATTERN TO WRITE INTO RAM
5AF 0000002C9DAD3   3055                JMPC     LOOP.W
                    3056
                    3057                DEC.A    R6      SADRH  B                 DECREMENT PAGE POINTER INTO RAM
5B0 001958CCC5B12   3058                NEXT                    NBORROW
                    3059
                    3060                B.SUB.A  R1      R0     B                 CHANGE PATTERN TO WRITE INTO RAM AGAIN
5B1 0000002C9DAD3   3061                JMPC     LOOP.W
                    3062
                    3063   READ.RAM     PASS.S   LIT     SADRH  B     #03          POINT TO PAGE 3 IN RAM AGAIN
5B2 00C05A8C75B32   3064                NEXT
                    3065
                    3066                PASS.S   LIT     SADRL  B     #FF          POINT TO THE END OF PAGE 3
5B3 3FC04E8C75B42   3067                NEXT
                    3068
                    3069                PASS.A   R2      R0     B                 RESTORE STARTING PATTERN
5B4 0000004C45B52   3070                NEXT
                    3071
                    3072   LOOP.R       S.XOR.A  SCR     R0                       COMPARE RAM TO PATTERN
5B5 003882A755B62   3073                NEXT                    NZERO
                    3074
                    3075                DEC.A    R3      SADRL  B                 DECREMENT POINTER INTO RAM
5B6 00194C6CC5C0B   3076                JMPOPC   RAMTESTERROR  NBORROW
                    3077
                    3078                B.SUB.A  R1      R0     B                 CHANGE PATTERN TO EXPECT TO READ FROM RAM
5B7 0000002C9DB53   3079                JMPC     LOOP.R
                    3080
                    3081                DEC.A    R6      SADRH  B                 DECREMENT RAM'S PAGE POINTER
5B8 001958CCC5B92   3082                NEXT                    NBORROW
                    3083
                    3084                B.SUB.A  R1      R0     B                 CHANGE PATTERN AGAIN
5B9 0000002C9DB53   3085                JMPC     LOOP.R
                    3086
                    3087                B.SB1.A  R7      RAM          ID3         FLAG ALL ERRORS IN ID3 AGAIN
5BA 04405CE49000A   3088                RETURNC                                   TEST DONE AND PASSED
```

```
                    3090  ************************************************************************************
                    3091  * THIS NEXT TEST WILL CHECK OUT THE WORST CASE TIMING PATH BETWEEN THE 2901      *
                    3092  * AND THE RAM. THE RAM'S CONTENTS WILL BE ADDED TO THE 2901 TO PRODUCE A CARRY   *
                    3093  * OUT.                                                                           *
                    3094  ************************************************************************************
                    3095
                    3096   TEST8         PASS.S   LIT    R0     B    #11          SET R0=11
5B8 0440028C75BC2   3097                 NEXT
                    3098
                    3099                 PASS.S   LIT    RF     B    #EF          RF = RAM PATTERN = #EF
5BC 3BC03E8C75BD2   3100                 NEXT
                    3101
                    3102                 PASS.A   RF     RAM         #08          WRITE #EF INTO RAM #308
5BD 02205DE445BE2   3103                 NEXT                 NEVER               CONDITION BIT IS FALSE
                    3104
                    3105                 S.ADD.A  RAM    R0          #08          ADD RAM'S #308 TO R0 IN 2901
5BE 021802A455BF2   3106                 NEXT                 ZERO
                    3107
                    3108                 ZERO             R1     B               ZERO COUNT IN R1 FOR NEXT TEST
5BF 0000040E25C23   3109                 JMPC     CHECKRTC                       PASSED THIS TEST
                    3110
                    3111
                    3112   RAMTESTERROR  PASS.S   LIT    R1     B    #04          STORE ERROR BIT IN R1
5C0 0100068C75C12   3113                 NEXT
                    3114
                    3115                 PASS.A   R1     RAM         ID3          PASS THE ERROR IN RAM'S ID3
5C1 04405C244000A   3116                 RETURNC                                 RETURN TO MAIN PROGRAM
```

```
                    3118  ********************************************************************************
                    3119  *        THIS SECTION OF CODE CHECKS OUT RTC                                   *
                    3120  ********************************************************************************
                    3121
                    3122   CHECKRTC             CLRTC                              CLEAR RTC
5C2 0000026625C32   3123                NEXT
                    3124
                    3125                         LIT    DRSEL       #04
5C3 0121628625C33   3126                JMPC     *              NRTC
                    3127
                    3128                         CLRTC                             CLEAR RTC
5C4 0000026625C52   3129                NEXT
                    3130
                    3131   INCCOUNT     INC.A    R1     R1      B                  INCREMENT COUNTER
5C5 0021042C4DC62   3132                NEXT                   NRTC
                    3133
                    3134                NOP
5C6 0000000625C53   3135                JMPC     INCCOUNT                         WAIT FOR RTC
                    3136
                    3137                S.XOR.A  LIT    R1          #80           COMPARE COUNTER TO WHAT IT SHOULD BE
5C7 2038068755C82   3138                NEXT                   NZERO
                    3139
                    3140                         LIT    DRSEL       #84           SELECT GCR RECORDING MODE
5C8 2100628625D03   3141                JMPC     RTCERROR
                    3142
                    3143                         CLRTC                             CLEAR RTC
5C9 0000026625CA2   3144                NEXT
                    3145
                    3146                ZERO             R1      B                 ZERO OUT THE COUNTER
5CA 0021040E25CA3   3147                JMPC     *              NRTC              WAIT FOR RTC
                    3148
                    3149                         CLRTC                             CLEAR RTC
5CB 0000026625CC2   3150                NEXT
                    3151
                    3152   INCREMENTCT2 INC.A    R1     R1      B                  INCREMENT COUNTER
5CC 0021042C4DCD2   3153                NEXT                   NRTC
                    3154
                    3155                NOP
5CD 0000000625CC3   3156                JMPC     INCREMENTCT2
                    3157
                    3158                S.XOR.A  LIT    R1          #64           DID RTC COME UP AT THE RIGHT TIME?
5CE 1918068755CF2   3159                NEXT                   ZERO
                    3160
                    3161                ZERO             R1      B                 ZERO OUT COUNTER
5CF 0000040E25D23   3162                JMPC     CHECKWDR
                    3163
                    3164   RTCERROR     PASS.S   LIT    R1      B      #20         STORE ERROR BIT IN R1
5D0 0800068C75D12   3165                NEXT
                    3166
                    3167                PASS.A   R1     RAM            ID3         PASS ERROR IN RAM'S ID3
5D1 04405C244000A   3168                RETURNC
```

```
                    3170  ************************************************************************************
                    3171  *      THIS CODE CHECKS OUT THE WDR CIRCUITS.                                      *
                    3172  ************************************************************************************
                    3173
                    3174  CHECKWDR              LIT     WRCON       #10          INITIIALIZE WDR, WDL
5D2 0400728625D32   3175                NEXT
                    3176
                    3177                        LIT     WRCON       #30          DEASSERT WINIT
5D3 0C25728625D42   3178                NEXT                        NWDR         WAIT FOR WDR TO COME UP
                    3179
                    3180                NOP
5D4 0025000625D43   3181                JMPC    *                   NWDR         WAIT FOR WDR TO COME UP
                    3182
                    3183                                WDATA                    CLEAR WDR
5D5 0000680625D62   3184                NEXT
                    3185
                    3186  INCWDRCOUNT   INC.A    R1      R1      B                INCREMENT WRD COUNTER
5D6 0025042C4DD72   3187                NEXT                        NWDR
                    3188
                    3189                NOP
5D7 0000000625D63   3190                JMPC    INCWDRCOUNT
                    3191
                    3192                S.XOR.A LIT     R1          #64          DID WDR COME UP IN TIME
5D8 1938068755D92   3193                NEXT                        NZERO
                    3194
                    3195                ZERO            R1      B                ZERO OUT R1:COUNTER
5D9 0000040E25DE3   3196                JMPC    WDRERROR
                    3197
                    3198  WDLCOUNT      INC.A    R1      R1      B                INCREMENT COUNTER
5DA 0023042C4DD82   3199                NEXT                        NWDL
                    3200
                    3201                NOP
5DB 0000000625DA3   3202                JMPC    WDLCOUNT
                    3203
                    3204                S.XOR.A LIT     R1          #63
5DC 18D8068755DD2   3205                NEXT                        ZERO
                    3206
                    3207                ZERO            R1      B                ZERO OUT R1 FOR NEXT TEST
5DD 000004OE25E03   3208                JMPC    CHECKREAD                        WDL CAME UP AT RIGHT TIME
                    3209
                    3210  WDRERROR      PASS.S  LIT     R1      B   #40          STORE ERROR BIT IN R1
5DE 1000068C75DF2   3211                NEXT
                    3212
                    3213                PASS.A  R1      RAM         ID3          PASS THE ERROR IN RAM'S ID3
5DF 04405C244000A   3214                RETURNC
```

```
                    3216  ************************************************************************************
                    3217  *      THIS CODE CHECKS OUT THE READ STATUS REGISTER AND READ CIRCUIT CONDTION      *
                    3218  * BITS.                                                                            *
                    3219  ************************************************************************************
                    3220
                    3221   CHECKREAD            LIT    RDCON      #00          SET RINIT,IDLE,CRINIT ACTIVE
5E0 00006E8625E12   3222                NEXT
                    3223
                    3224                S.XOR.A RDSTS  R1                       RDSTS SHOULD ALL BE LOW
5E1 003806E755E22   3225                NEXT                   NZERO
                    3226
                    3227                NOP
5E2 0004000625E73   3228                JMPC    READERROR      RDR
                    3229
                    3230                NOP
5E3 0006000625E73   3231                JMPC    READERROR      RDL           RDR SHOULD BE CLEAR
                    3232
                    3233                NOP
5E4 001E000625E73   3234                JMPC    READERROR      CRCERR        RDL SHOULD BE CLEAR
                    3235
                    3236                NOP
5E5 003F000625E73   3237                JMPC    READERROR      NRDDX         CRCERR SHOULD BE CLEAR
                    3238
                    3239                ZERO           RAM         ID3       CLEAR THE ERROR REGISTER
5E6 04405C062000A   3240                RETURNC                                PASSED THE SELF TEST   GO BACK TO MAIN CODE
                    3241
                    3242   READERROR    PASS.S  LIT    R1     B    #80       STORE ERROR BIT IN R1
5E7 2000068C75E82   3243                NEXT
                    3244
                    3245                PASS.A  R1     RAM         ID3       STORE ERROR BIT IN RAM
5E8 04405C244000A   3246                RETURNC
                    3247
                    3248
                    3249
                    3250  ************************************************************************************
                    3251  *  THAT'S THE END OF SELF TEST. ALL THAT'S LEFT IS TO TAKE THE LAST LOCATION      *
                    3252  * IN THIS CODE (#7FF) WHICH IS WHERE IT WAKES UP AFTER POWER ON AND TELL IT        *
                    3253  * TO GO TO A 100 MILLI-SECOND WAIT ROUTINE FOR POWER TO SETTLE BEFORE RESETING     *
                    3254  * THE 2910 SEQUENCER AND BEGINNING TO EXECUTE THE MICRO-CODE.                      *
                    3255  ************************************************************************************
```

```
                    3258  ********************************************************************************
                    3259  * AFTER POWER UP AT #7FF, WAIT 100ms FOR POWER TO SETTLE AND POSSIBLE          *
                    3260  * ABORTED TAPE MOTION TO STOP. THEN RESET THE STACK AND BEGIN CODE             *
                    3261  ********************************************************************************
                    3262
                    3263  * THE END OF THIS ROUTINE MUST FALL AT ADDRESS #7FF
                    3264
     07F4           3265                ORG     #7F4
                    3266
                    3267                NOP                                         WAIT FOR POWER TO SETTLE
7F4 0000000627F52   3268                NEXT
                    3269                NOP
7F5 0000000627F62   3270                NEXT
                    3271                NOP
7F6 0000000627F72   3272                NEXT
                    3273
                    3274                PASS.S  LIT     R0      B     #09           LOAD MS 100ms COUNT
7F7 0240028C77F82   3275                NEXT
                    3276
                    3277                PASS.S  LIT     R1      B     #C4           LOAD LS 100ms COUNT
7F8 3120068C77F92   3278                NEXT                    NEVER
                    3279
                    3280  RESETRTC              CLRTC                             CLEAR RTC (Real Time Clock ticker)
7F9 0000026627FE3   3281                JMPC    RESET.2910
                    3282
                    3283                NOP
7FA 0021000627FA3   3284                JMPC    *               NRTC              WAIT FOR RTC
                    3285
                    3286                DEC.A   R1      R1      B                 DECREMENT THE LS COUNT
7FB 0039042CC7FC2   3287                NEXT                    BORROW
                    3288
                    3289                DEC.A   R0      R0      B                 DECREMENT THE MS COUNT
7FC 0039000CC7F93   3290                JMPC    RESETRTC        BORROW
                    3291
                    3292                INC.A   R0      R0      B                 UNDO DECREMENT ABOVE SINCE WE DIDN'T HAVE TO BORROW
7FD 0020000C4FF92   3293                JMPU    RESETRTC        NEVER
                    3294
                    3295  RESET.2910    NOP                                       100ms HAS PASSED SO DO A RESET
7FE 0000000620000   3296                RESETU
                    3297
                    3298  * HERE'S WHERE THE CTC WAKES UP AFTER TURNING POWER ON SO GO WAIT 100ms FOR POWER TO SETTLE
                    3299
                    3300                NOP
7FF 0000000627F42   3301                JMPU    #7F4
                    3302
                    3303
                    3304
                    3305
                    3306  * THAT'S ALL FOLKS !
                    3307
                    3309                END
  NO LINES WITH ERRORS
```

```
CLR.RAM2        01D2   2110 01D2   2113 01D3
CLR.STS         01C5   2083 01C5   1462 011B   1517 012A   1601 013D   1728 0165
CLR.STS2        01D4   2120 01D4   1433 0113   1459 011A   1514 0129   1726 0164
CLRBIT.P1       026F   2647 026F    644 002E    647 002F    902 0078   1192 00D7   1859 018B   1886 0194
CLRBIT.P3       0273   2659 0273   1168 00D5
CNG.15.16       022F   2437 022F
COMMAND         0032    199 0000    702 003D    821 005C
CONT.WRT.RPT    028E    247 028E    895 0365    898 0366
COULD           005C    821 005C
COULDNT         005F    830 005F    816 005A
CRAM.BC         0139   1588 0139   1066 00BB
CYCLE.FOR       00DB   1215 00DB   1007 00A1
CYCLE.REV       00E9   1264 00E9   1237 00E2
DATA            041D   1628 041D   1616 0419
DATA.BLK        0438   1727 0438
DATA.CRC        0350    801 0350    776 034A
DATA.CRC7       0353    813 0353
DATA.HERE       04D0   2348 04D0
DATA.ID         0483   2039 0483   2016 047B
DATAHERE        03A1   1127 03A1
DATALOOP        02EA    420 02EA    448 02F3
DBUSTESERROR    051C   2556 051C   2520 050B   2522 050C   2524 050D   2526 050E   2528 050F   2530 0510   2532 0511
                       2534 0512   2536 0513   2538 0514   2540 0515   2542 0516   2544 0517   2546 0518   2548 0519
                       2551 051A
DBUSTEST        050A   2518 050A
DEC.1BYTE       0293   2791 0293
DEC.2BYTES      0291   2787 0291    933 036F   2367 04D6
DEC.3BYTES      028F   2783 028F    980 037B   1151 03A8
DEC.BLKNUM      014E   1661 014E   1648 014A   1658 014D
DECODE.ID       0479   2009 0479   2001 0476
DESELECT        002F    195 0000    596 0021    660 0031    954 0090   1146 00CE   1167 00D5   1855 018A
DESELECT0       000A    145 0000    576 0019
DESELECT1       000B    146 0000    581 001B
DESELECT2       000C    147 0000    586 001D
DESELECT3       000D    148 0000    591 001F
DIS.INT         00D7   1191 00D7   1021 00A8
DOFETCH         02F1    441 02F1    411 02E8    414 02E9    436 02EF
DONE            0420   1637 0420   1626 041C   1644 0422
DONE.1.IN       0214   2364 0214
DONE.24         01FA   2262 01FA   2251 01F6
DONE.BLKS       0457   1864 0457   1856 0455
DONE.ERASE      00F2   1308 00F2   1335 00F8   1344 00FB   1370 0103
DONE.RB         0282   2719 0282   2705 027D
DONEC.A         01C2   2068 01C2
DONECOUNTR      005A    225 0000   1615 0142   1433 03EE   1512 0400   1634 041F
DONECOUNTW      0059    224 0000   1732 0167    435 02EF
DONESWAP        0347    766 0347    758 0344    764 0346
DONTFETCH       02F2    444 02F2    439 02F0
DONTSWAP        0343    754 0343
DRIVE           002C    192 0000    561 0013   1090 00BF
DRIVE.DS        002E    194 0000    571 0017   1102 00C3   1112 00C5   1121 00C8
DRIVE.UP        002D    193 0000    566 0015   1096 00C1   2068 01C2
DRSTS0          0000    133 0000
DRSTS1          0001    134 0000
DRSTS2          0002    135 0000
DRSTS3          0003    136 0000   2030 0186
DRV.TRK         0031    197 0000    606 0025   1164 00D4   1837 0187
EN.INT          00D9   1199 00D9   1023 00A9
```

```
FORWARD       0357    844 0357   1407 010C   1465 011C   1523 012C   1682 0154
FORWRD1       0374    950 0374    939 0371
FUNNY.INCH    0101   1363 0101   1354 00FE   1360 0100
GAPFINDER     04C3   2298 04C3     98 0298
GCR.3IN       007E    909 007E
GCR.CRC       02F4    452 02F4    430 02ED
GCR.DATA      02D3    334 02D3    298 02CA
GCR.FM        02CB    303 02CB
GCR.R.ID      046A   1943 046A   1337 03D4   1925 0466
GCR.SD        0368    908 0368
GCR.WD        0388   1027 0388
GET1STBYTE    003D    702 003D    687 003A
GETSTRING     0047    740 0047    718 0042    724 0044    756 004C
GIVEUP        01A6   1964 01A6   1912 0197
GO            0060    125 0000   2681 0278
GOTSTRING     004D    758 004D
HANDLE.14     0233   2453 0233   2392 021D
HANDLE.15     022D   2431 022D   2390 021C
HANDLED.14    021D   2391 021D   2463 0236
HANDLED.15    021C   2389 021C   2447 0232
HEAD2LATE     0406   1536 0406   1501 03FC
HEAD2SOON     0403   1524 0403   1498 03FB
HEADERHERE    0409   1555 0409   1480 03F8   1531 0405
HOLDOUT       0050    210 0000    199 02B3   1473 03F7   1628 041D   2179 04A9   2182 04AA   2205 04B0
I.O.ADDR.0    02A0    121 02A0    107 029B    171 02AC    180 02AF    183 02B0
I.O.ADDR.4    02A4    136 02A4    215 02B7
I.O.ADDR.8    02A8    151 02A8    155 02A9    189 02B1
ID.BAD        03DE   1368 03DE   1328 03D1   1340 03D5   1349 03D8   1361 03DC
ID.HERE       03B1   1185 03B1   1113 039F
ID0           000E    150 0000    855 0070
ID1           000F    151 0000    517 0003
ID2           0010    152 0000    521 0005
ID3           0011    153 0000    513 0001   2559 051D   2615 052E   2644 0535   2998 05A0   3087 05BA   3115 05C1
                     3167 05D1   3213 05DF   3239 05E6   3245 05E8
ID4           0012    154 0000    523 0006
ID5           0013    155 0000
IDLELOOP      0031    660 0031    629 0029    631 002A    690 003B
IDLOOP2       0294     85 0294    122 02A0    125 02A1    137 02A4    152 02A8    174 02AD    197 02B2    203 02B4
                     209 02B5
IDLOOP3       0297     94 0297    869 035E    930 036E    936 0370    948 0373    954 0375    977 037A    983 037C
                   1007 0383   1131 03A2   1144 03A6   1154 03A9   1176 03B0   1251 03BF   1261 03C2   1300 03CC
                   1315 03CF   1352 03D9   1358 03DB   1364 03DD   1380 03E0   1392 03E3   1410 03E7   1443 03F1
                   1465 03F5   1483 03F9   1504 03FD   1516 0401   1519 0402   1574 040F   1586 0412   1638 0420
                   1641 0421   1662 0425   1824 044F   1877 045B   1889 045D   1901 0460   1916 0464   1931 0468
                   1950 046C   1968 0471   1974 0473   2004 0477   2034 0481   2040 0483   2046 0485   2061 0488
                   2076 048C   2082 048E   2104 0493   2107 0494   2137 049D   2180 04A9   2186 04AB   2206 04B0
                   2215 04B2   2238 0487   2247 04BA   2262 04BF   2268 04C1   2469 04FF
IDLOOP4       0299    100 0299   2302 04C4   2305 04C5   2320 04C9   2326 04CB   2355 04D2   2361 04D4   2364 04D5
                   2370 04D7
IDLOOP5       029A    103 029A    231 02B9    264 02C1    273 02C4    283 02C5    304 02CB    307 02CC    319 02CF
                    335 02D3    338 02D4    347 02D6    350 02D7    356 02D9    378 02DF    393 02E3    396 02E4
                    445 02F2    453 02F4    459 02F6    478 02FB    493 02FF    496 0300    524 0307    533 0309
                    539 030B    553 030E    562 0311    568 0313    571 0314    574 0315    601 0318    613 031C
                    622 031F    625 0320    631 0322    640 0325    656 0329    659 032A    665 032B    683 0330
                    707 0336    737 033F    788 034E    802 0350    808 0352    817 0354
INC.GAPC      04D3   2357 04D3   2352 04D1
INCCOUNT      05C5   3131 05C5   3135 05C6
INCREMENTCT2  05CC   3152 05CC   3155 05CD
```

```
LOOP.R          05B5   3072 05B5   3079 05B7   3085 05B9
LOOP.RB         027E   2707 027E   2714 0280   2717 0281
LOOP.W          05AD   3048 05AD   3055 05AF   3061 05B1
LOOPC.A         0188   2037 0188   2065 01C1
LP.GCR.WD       038C   1039 038C   1028 0388
LP.LATE         037D    985 037D
LP.MFM.WD       0394   1063 0394   1052 0390
LP.WAIT         0378    970 0378    962 0376    965 0377
LSBC.EQ.0       0144   1621 0144
LSBC.NE.0       0146   1627 0146   1619 0143
LSBCREV         0057    220 0000   1734 0168    558 0310    570 0314
LST.SBYTE       0475   1984 0475   1971 0472
LSTHIRDIN.      0055    217 0000    919 0083   1163 03AC   1188 03B2   2351 04D1
LSWORDCNT       0052    213 0000   1722 0163    664 032B
MFM.3IN         0080    913 0080    906 007D
MFM.FM          031F    621 031F
MFM.FM.REV      0321    627 0321
MFM.LOOP        0341    748 0341    791 034F
MFM.SD          036B    917 036B    909 0368
MFM.WD          0390   1051 0390   1025 0387
MFMDATA.ID      0329    655 0329    619 031E
MFMSYNC1        0318    600 0318    254 02C0    610 0318
MFMSYNC2        0325    639 0325    646 0327
MORE.DECODE     0098    982 0098    964 0093
MSBCREV         0056    219 0000   1740 016B    564 0312    573 0315
MSTHIRDIN.      0054    216 0000    917 0082   1160 03AB   1185 03B1   2348 04D0
MSWORDCNT       0053    214 0000   1720 0162    673 032D
N.DATA          03D4   1336 03D4
N.DATAHERE      03A5   1140 03A5   1116 03A0
N.SENDJUNK      03F8   1479 03F8
NACK            00F8    122 0000    720 0043    777 0051   1955 01A3   2003 01B1
NADDRS.TEST     05A3   3012 05A3
NDATA.HERE      04D6   2366 04D6   2346 04CF
NDATA.ID        047C   2018 047C
NDONEWRITE      034B    778 034B    770 0348
NFETCH          034E    787 034E    782 034C
NFETCH2         033F    736 033F    731 033D
NO.ERRORS       045A   1873 045A   1865 0457
NO.SEND         0436   1719 0436   1708 0432   1710 0433   1712 0434
NOBORROW        04CB   2325 04CB   2317 04C8
NOSWEAT         04A7   2173 04A7   2159 04A2   2162 04A3   2165 04A4
NOT.RTC         04C4   2301 04C4
NOT.WRITE       04CD   2339 04CD   2314 04C7
NOT.YET         03CD   1308 03CD   1241 03BC   1254 03C0   1264 03C3   1282 03C6   1294 03CA
NOT700YET       0208   2314 0208   2299 0202   2303 0204
NOTLAST         03FF   1509 03FF
NOTOOSOON       0450   1829 0450   1815 044D
NSWAP           04A9   2179 04A9
00              00A0   1004 00A0    989 009A
01              00A1   1006 00A1
02              00A2   1008 00A2
03              00A3   1010 00A3
04              00A4   1012 00A4
05              00A5   1014 00A5
06              00A6   1016 00A6
07              00A7   1018 00A7
08              00A8   1020 00A8
09              00A9   1022 00A9
```

```
0A              00AA   1024 00AA
0B              00AB   1026 00AB
0C              00AC   1028 00AC
0D              00AD   1030 00AD
0DD.BLK         043C   1749 043C   1728 0438   1739 043B
0DD.CBC         0148   1641 0148
0E              00AE   1032 00AE
0F              00AF   1034 00AF
OFFLINE         01FB   2269 01FB   1367 0102   2257 01F8   2318 0209
OFFLINE1        028E   2773 028E   2759 0289
OFFLINE2        0248   2527 0248   2510 0242   2519 0245
OK2BFM          03D8   1348 03D8   1343 03D6
ONES            05A5   3018 05A5
OPDRSTS0        0005    139 0000
OPDRSTS1        0006    140 0000
OPDRSTS2        0007    141 0000
OPDRSTS3        0008    142 0000
OPDRSTSX        0004    138 0000    867 0074
OPDRSTSZ        0009    143 0000
OPERAND1        0034    201 0000    966 0094
OPERAND2        0035    202 0000   1738 016A
OPERAND3        0036    203 0000   1293 00EF   1328 00F6   1609 0140   1710 015D   1736 0169    346 02D6    352 02D8
                        694 0332
OPERAND4        0037    204 0000    982 0098    985 0099    988 009A   1233 00E1   1325 00F5   1363 0101   1606 013F
                       1708 015C   1730 0166    349 02D7    358 02DA    691 0331
OPTEST2         057E   2888 057E
OPTEST4         0589   2927 0589
OPTEST5         058A   2930 058A
OPTEST6         0590   2949 0590
OPTEST7         0593   2958 0593
OPTEST8         0595   2964 0595
OPTEST9         0597   2970 0597
OUTPUTA         0282    196 02B2    140 02A5
OUTPUTB         02B5    208 02B5    143 02A6    146 02A7
OVRFLW          0285   2732 0285   2142 01D9   2158 01DE   2727 0283   1528 0404   1540 0407   1629 041D   2150 049F
                       2200 04AE
OVTEST          057A   2871 057A
PAST.LP         037F    994 037F    974 0379
PATTERN         0536   2651 0536   2627 0530
PATTERNLOOP     0530   2626 0530   2633 0532
PRMSTS0         0014    157 0000    634 002B    858 0071   1879 0192
PRMSTS1         0015    158 0000    548 0011    626 0028    628 0029   1833 0185   1839 0188   2650 0270   2653 0271
                       2656 0272
PRMSTS2         0016    159 0000    542 000E   1830 0184
PRMSTS3         0017    160 0000    544 000F    903 007C   1065 00BB   1429 0112   1458 011A   1513 0129   1725 0164
                       1804 0178   1810 017B   1861 018C   2662 0274   2665 0275   2668 0276    247 028E    905 0367
                       1021 0386   1100 039B   1194 03B4
PRMSTS4         0018    161 0000   2072 01C4
PRMSTS5         0019    162 0000
PUSHPOPERROR    052D   2612 052D   2580 0522   2589 0525   2601 0529   2610 052C
QNTSTS0         0026    178 0000    870 0075
QNTSTS1         0027    179 0000   2091 01C9   1785 0447
QNTSTS2         0028    180 0000   2093 01CA   1788 0448
QNTSTS3         0029    181 0000   2095 01CB   1791 0449
QNTSTS4         002A    182 0000   2097 01CC   1794 044A
QNTSTS5         002B    183 0000
R.DATA.A        04A0   2152 04A0   2113 0495
R.DATA.         049E   2146 049E   2153 04A0
```

```
R.GCR.ID        03D0   1324 03D0   1204 03B7
R.LASTSYNC      046E   1958 046E   1944 046A
R.LSBC.NOT      03E9   1418 03E9   1404 03E5
R.MFM.ID        0476   2000 0476   1201 03B6
R.MSBC.NOT      03E5   1403 03E5   1389 03E2
R.R.FRAME       04B9   2243 04B9   2229 04B4
R.R.ID          04B4   2228 04B4   2019 047C   2128 049A   2212 04B1
R.R.LSBC        0462   1909 0462   1898 045F
R.R.LSBC.N      045C   1885 045C   1419 03E9   1850 0453
R.R.MSBC        0466   1924 0466   1910 0462
R.R.MSBC.N      045F   1897 045F   1886 045C
RAMTEST         05A1   3006 05A1   2992 059E
RAMTESTERROR    05C0   3112 05C0   3076 0586
READ.DATA       0413   1597 0413   1279 03C5   1580 0410
READ.LSBC       03E2   1388 03E2   1377 03DF
READ.LSWC       0486   2054 0486   2013 047A
READ.MSBC       03DF   1376 03DF   1331 03D2
READ.MSWC       0489   2066 0489   2055 0486
READ.RAM        05B2   3063 05B2
READ.TAPE       0126   1504 0126   1060 00B9
READ.WHAT       00B9   1059 00B9    967 0094
READERROR       05E7   3242 05E7   3228 05E2   3231 05E3   3234 05E4   3237 05E5
REC.LOST        0448   1797 0448   1789 0448   1795 044A
RECOVERED       0453   1850 0453   1833 0451   1836 0452
REPEAT          035F    876 035F   2493 0506
RESET.2910      07FE   3295 07FE   3281 07F9
RESETRTC        07F9   3280 07F9   3290 07FC   3293 07FD
RETRY           0198   1914 0198   1980 01AB
REV.1           028D   2770 028D   2762 028A
REV.2           0245   2518 0245   2507 0241
REV.24.IN       01F3   2239 01F3   1225 00DE
REV.700FT       01FD   2284 01FD   1268 00EA
REV.FRAME       0322    630 0322    628 0321    823 0356
REV.STOP        04FB   2456 04FB   1355 03DA   1431 03ED   2037 0482   2058 0487
REVDATA.ID      0355    819 0355
REVERSE         0358    847 0358   1384 0106   1436 0114
REVRS.BITS      027D   2704 027D   1733 0167   1737 0169
REVRSEL         0372    944 0372
REWIND          00E5   1252 00E5   1009 00A2
RTC.SET         04C7   2313 04C7   2299 04C3
RTCERROR        05D0   3164 05D0   3141 05C8
SB1             01D9   2141 01D9   2152 01DC
SB2             01DA   2145 01DA   2177 01E4
SB3             01DD   2154 01DD   2146 01DA
SB4             01DE   2157 01DE   2168 01E1
SB5             01DF   2161 01DF   2155 01DD
SB6             01E2   2170 01E2   2162 01DF
SB7             01E3   2173 01E3   2139 01D8
SCRAMBLE        0259   2589 0259   2038 0188
SEARCH.FM       03DD   1363 03DD
SECSTS0         001A    164 0000    861 0072   1867 018E
SECSTS1         001B    165 0000   2120 01D4
SECSTS2         001C    166 0000   2122 01D5
SECSTS3         001D    167 0000   1618 0143   2124 01D6   2456 0234   1409 03E7   2100 0492
SECSTS4         001E    168 0000   1612 0141   2126 01D7   2453 0233   1424 03EB   2097 0491
SECSTS5         001F    169 0000
SEL.DEN         00C8   1121 00C8   1029 00AC   1031 00AD
SEL.ORTKDN      00CD   1143 00CD   1106 00C4   1131 00CB
```

```
SEL.DRV      008D  1084 008D   900 007A
SEL.TRK      00C5  1112 00C5  1037 0080  1039 0081  1041 00B2  1043 0083
SELECT       0030   196 0000   601 0023   905 007D  1152 00D0   865 035D  1009 0384
SELF.TEST    050A  2517 050A   514 0001
SEND.BYTES   01D8  2138 01D8  1546 0132  1555 0135  1678 0153
SEND.EM      0153  1678 0153  1665 014F  1674 0152
SEND.PRIM    0192  1879 0192  1865 018D  1874 0190
SEND.SEC     018E  1867 018E
SEND.XOR     0131  1542 0131  1063 008A
SEND1        0421  1640 0421
SEND2        0422  1643 0422  1635 041F
SENDAACK     0058   808 0058   762 004E
SENDANACK    0050   774 0050   727 0045   765 004F   797 0057
SENDBCC      01A2  1945 01A2
SENDEOR      0277  2677 0277  1532 012F  1561 0137  1688 0157  1753 016D
SENDGO       0278  2681 0278  1709 015C
SENDING      0054   787 0054
SENDJUNK     03FB  1497 03FB  1474 03F7
SENDSTRING   0196  1908 0196   641 002D   882 0077  1871 018F  1883 0193  2694 027C
SET..IT      023F  2491 023F  2478 0239  2480 023A  2482 023B  2484 023C  2486 023D
SET.IT       022C  2421 022C  2378 0216  2380 0217  2382 0218  2384 0219  2386 021A  2388 021B  2394 021E
                   2396 021F  2398 0220  2400 0221  2402 0222  2404 0223  2406 0224  2408 0225  2410 0226
                   2412 0227  2414 0228  2416 0229  2418 022A  2438 022F  2441 0230  2444 0231  2457 0234
                   2460 0235
SET.SEND     0435  1716 0435  1696 042C  1698 042D  1700 042E  1702 042F  1704 0430  1706 0431
SETBIT.P1    0271  2653 0271  1200 00D9  2053 018D  2422 022C  2492 023F
SETBIT.P3    0275  2665 0275  1171 00D6  1814 017C  1818 017E  1822 0180  1826 0182
SEUDO.ADDR   0033   200 0000   896 0079   929 0087   933 0089  1084 00BD  1755 016E  1852 0189  2629 026B
SHIFT        026B  2629 026B   841 0060   927 0086
SKIP.FOR     0118  1452 0118  1019 00A7
SKIP.REV     0110  1423 0110  1017 00A6
SPARE        003A   208 0000  2684 0279  2687 027A
SRCH.FMFOR   010A  1400 010A  1015 00A5
SRCH.FMREV   0104  1377 0104  1013 00A4
START        0000   509 0000
START.ERASE  00F9  1337 00F9  1329 00F6
STARTED      035D   865 035D   845 0357   848 0358   854 035A   860 035C
STOP.INGAP   04FB  2457 04FB  2419 04EF  2421 04F0  2423 04F1  2425 04F2  2427 04F3  2429 04F4  2431 04F5
                   2433 04F6  2435 04F7  2437 04F8  2439 04F9
STOP.NOW     0500  2475 0500  2263 01FA  2309 0206  2324 020B  2382 04E1  2388 04E3  2407 04E9  2409 04EA
                   2411 04EB  2413 04EC  2415 04ED  2417 04EE  2441 04FA
STOP.RD      0502  2480 0502
STOP.WRT     0504  2486 0504  2460 04FC  2478 0501
STORE        0096   120 0000   711 0040
STORE7       0038   205 0000
STORE8       0039   206 0000
STRT.DLYA    0367   905 0367   866 035D
STRT.DLYB    036D   926 036D   906 0367   915 036A   921 036C
SUBTEST5     058E  2942 058E
SUBTEST8     0596  2967 0596
SWAP         0345   760 0345   752 0342
SWAP.EM      04AA  2182 04AA  2177 04A8
SWAPPER      0051   211 0000   721 033A   760 0345   763 0346
SYNCLOOP1    02C5   282 02C5   251 02BF   292 02C8
SYNCLOOP2    02DF   377 02DF   387 02E2
SYNCLOOP3    02FB   477 02FB   487 02FE
SYNCLOOP4    02CF   318 02CF   325 02D1
TB1          01E6  2191 01E6  2202 01E9
```

```
TB2             01E7   2195 01E7   2227 01F1
TB3             01EA   2204 01EA   2196 01E7
TB4             01EB   2207 01EB   2218 01EE
TB5             01EC   2211 01EC   2205 01EA
TB6             01EF   2220 01EF   2212 01EC
TB7             01F1   2226 01F1   2189 01E5
TEST2.2910      051E   2568 051E
TEST2PUSH1      0520   2574 0520   2568 051E
TEST2PUSH2      0523   2583 0523   2574 0520
TEST2PUSH3      0526   2592 0526   2583 0523
TEST2PUSH4      052A   2604 052A   2592 0526
TEST8           05BB   3096 05BB   3031 05A9
TIME.SZ         03A8   1150 03A8   1134 03A3
TIMER.WD        02B9    230 02B9    237 02BB    243 02BD
TOOSOON         044E   1820 044E
TOSS.BYTES      01E5   2188 01E5   1741 016B
TRAILER         030E    552 030E    368 02DD
TRDSTS0         0020    171 0000    864 0073
TRDSTS1         0021    172 0000   1644 0149   2083 01C5   2434 022E   1761 0440
TRDSTS2         0022    173 0000   1647 014A   2085 01C6   1764 0441
TRDSTS3         0023    174 0000   1654 014C   2087 01C7   2440 0230   1777 0445
TRDSTS4         0024    175 0000   1657 014D   2089 01C8   1780 0446
TRDSTS5         0025    176 0000
TRY.RETRY       01A8   1970 01A8   2010 01B3
UNDFLW          0286   2735 0286   2192 01E6   2208 01EB    424 02EB    716 0338    755 0343    761 0345
UPDATE.STS      0178   1804 0178   1005 00A0   1134 00CC   1195 00D8   1203 00DA   1243 00E4   1274 00EC   1315 00F4
                       1393 0109   1416 010F   1445 0117   1474 011F   1497 0125   1535 0130   1564 0138   1690 0158
                       1779 0174   1793 0177   2397 04E4   2399 04E5   2401 04E6   2403 04E7   2405 04E8
UPTEST          052F   2623 052F   2571 051F
UPTESTERROR     059F   2995 059F   2701 0545   2704 0546   2707 0547   2710 0548   2713 0549   2716 054A   2719 054B
                       2722 054C   2725 054D   2728 054E   2731 054F   2734 0550   2737 0551   2740 0552   2743 0553
                       2752 0556   2880 057B   2886 057D   2898 0581   2901 0582   2931 058A   2940 058D   2950 0590
                       2959 0593   2965 0595   2968 0596   2971 0597   2986 059C   2989 059D
W.MFMDATA       0340    745 0340
W.RERF.S        0254   2569 0254   2549 024D   2555 024F   2561 0251   2567 0253   2582 0258
WAIT.1.IN       020D   2335 020D   2254 01F7   2315 0208
WAIT.12.MS      024E   2551 024E    877 035F
WAIT.16.US      0252   2563 0252
WAIT.35.MS      024C   2545 024C   2481 0502   2496 0507
WAIT.5.MS       0250   2557 0250   2499 0508
WAIT.BLK        03B8   1228 03B8   1173 03AF   1309 03CD
WAIT.DATA       03C1   1257 03C1   1244 03BD
WAIT.LATE       044C   1811 044C   1765 0441   1781 0446   1821 044E
WAIT.RESET      003C    692 003C    851 0359
WAIT.XOR        03BE   1247 03BE
WAITFORACK      01AC   1988 01AC   1940 01A0
WC.CRC          0331    691 0331
WC.CRC7         0337    712 0337
WDLCOUNT        05DA   3198 05DA   3202 05DB
WDRERROR        05DE   3210 05DE   3196 05D9
WHATCAME        01A3   1955 01A3   1928 019C
WORDCOUNTA      032B    664 032B
WORDCOUNTB      032C    670 032C
WRITE           0359    850 0359   1488 0122   1750 016C
WRITE.          04C5   2304 04C5
WRITE..         04C8   2316 04C8
WRITE..DATA     0172   1772 0172
WRITE.DATA      0159   1698 0159    970 0095
```

```
WRITE.FM      0120   1481 0120   1053 0088
WRITE.GAP     00ED   1287 00ED    973 0096
WRITE.RAM     05AA   3040 05AA
WRITE.RPT     015A   1701 015A    976 0097
WRT..RPT      0175   1786 0175   1765 0171
XL1           0307    523 0307    505 0302
XOR           041A   1619 041A
XOR.BLK       0439   1732 0439   1723 0437
XOR.BLOCK     02FA    474 02FA    545 030D
XOR.CRC       0309    532 0309    521 0306
XOR.LOOP      0303    511 0303    527 0308
XOR.WRITE     02F9    471 02F9
ZEROES        05A7   3024 05A7
```

```
JMPU        0002   607 0025   650 0030   693 003C   718 0042   724 0044   730 0046   765 004F   788 0054
                   797 0057   828 005E   831 005F   849 0063   856 0070   859 0071   862 0072   865 0073
                   868 0074   871 0075   885 0078   912 007F   916 0081   940 0088   958 0091   961 0092
                   964 0093   967 0094   970 0095   973 0096   976 0097   995 009C  1005 00A0  1007 00A1
                  1009 00A2  1011 00A3  1013 00A4  1015 00A5  1017 00A6  1019 00A7  1021 00A8  1023 00A9
                  1025 00AA  1027 00AB  1029 00AC  1031 00AD  1033 00AE  1035 00AF  1037 00B0  1039 00B1
                  1041 00B2  1043 00B3  1045 00B4  1047 00B5  1049 00B6  1051 00B7  1053 00B8  1069 00BC
                  1106 00C4  1119 00C7  1128 00CA  1134 00CC  1195 00D8  1203 00DA  1243 00E4  1274 00EC
                  1315 00F4  1354 00FE  1360 0100  1370 0103  1393 0109  1416 010F  1445 0117  1474 011F
                  1497 0125  1535 0130  1564 0138  1648 014A  1658 014D  1674 0152  1690 0158  1699 0159
                  1702 015A  1713 015E  1779 0174  1793 0177  1840 0188  1877 0191  1889 0195  1946 01A2
                  1980 01A8  2010 01B3  2139 01D8  2152 01DC  2155 01DD  2168 01E1  2177 01E4  2189 01E5
                  2202 01E9  2205 01EA  2218 01EE  2227 01F1  2237 01F2  2240 01F3  2263 01FA  2270 01FB
                  2283 01FC  2285 01FD  2287 01FE  2312 0207  2327 020C  2422 022C  2438 022F  2447 0232
                  2463 0236  2492 023F  2516 0244  2525 0247  2531 0249  2549 024D  2555 024F  2561 0251
                  2567 0253  2576 0256  2582 0258  2596 025C  2600 025E  2604 0260  2608 0262  2612 0264
                  2616 0266  2620 0268  2651 0270  2663 0274  2678 0277  2694 027C  2733 0285  2736 0286
                  2771 028D  2774 028E    89 0295   122 02A0   125 02A1   128 02A2   131 02A3   137 02A4
                   140 02A5   143 02A6   146 02A7   152 02A8   155 02A9   158 02AA   161 02AB   183 02B0
                   189 02B1   203 02B4   215 02B7   254 02C0   273 02C4   310 02CD   328 02D2   411 02E8
                   414 02E9   448 02F3   505 0302   527 0308   583 0317   628 0321   649 0328   758 0344
                   764 0346   791 034F   823 0356   845 0357   848 0358   854 035A   860 035C   869 035E
                   898 0366   915 036A   921 036C   948 0373   954 0375   965 0377   989 037E  1007 0383
                  1037 038B  1049 038F  1061 0393  1073 0397  1090 039A  1107 039D  1110 039E  1137 03A4
                  1157 03AA  1175 03B0  1201 03B6  1238 03B8  1254 03C0  1267 03C4  1291 03C9  1300 03CC
                  1315 03CF  1355 03DA  1364 03DD  1369 03DE  1383 03E1  1395 03E4  1413 03E8  1465 03F5
                  1486 03FA  1510 03FF  1513 0400  1519 0402  1531 0405  1543 0408  1565 040C  1568 040D
                  1574 040F  1586 0412  1607 0416  1626 041C  1641 0421  1644 0422  1662 0425  1679 042A
                  1682 042B  1696 042C  1698 042D  1700 042E  1702 042F  1704 0430  1706 0431  1708 0432
                  1710 0433  1712 0434  1736 043A  1739 043B  1765 0441  1781 0446  1789 0448  1795 044A
                  1798 044B  1824 044F  1833 0451  1836 0452  1859 0456  1871 0459  1877 045B  1892 045E
                  1904 0461  1919 0465  1934 0469  1953 046D  1974 0473  1977 0474  1985 0475  2007 0478
                  2037 0482  2046 0485  2061 0488  2082 048E  2104 0493  2107 0494  2125 0499  2137 049D
                  2159 04A2  2180 04A9  2186 04AB  2206 04B0  2218 04B3  2241 04B8  2250 04BB  2259 04BE
                  2268 04C1  2271 04C2  2308 04C6  2323 04CA  2329 04CC  2343 04CE  2364 04D5  2379 04E0
                  2382 04E1  2385 04E2  2388 04E3  2397 04E4  2399 04E5  2401 04E6  2403 04E7  2405 04E8
                  2407 04E9  2409 04EA  2411 04EB  2413 04EC  2415 04ED  2417 04EE  2419 04EF  2421 04F0
                  2423 04F1  2425 04F2  2427 04F3  2429 04F4  2431 04F5  2433 04F6  2435 04F7  2437 04F8
                  2439 04F9  2441 04FA  2469 04FF  2580 0522  2589 0525  2601 0529  2610 052C  2636 0533
                  2992 059E  3031 05A9  3293 07FD  3301 07FF
JMPZJ       0007   1171 00D6  2714 0280  2717 0281    92 0296  1728 0438
LDZ         000C   1168 00D5  1351 00FD  1357 00FF  1459 011A  1514 0129  1726 0164  2705 027D   866 035D
                   892 0364   905 0367   945 0372   951 0374   962 0376   995 037F  1004 0382  1022 0386
                  1084 0398  1128 03A1  1141 03A5  1173 03AF  1229 03B8  1279 03C5  1309 03CD  1331 03D2
                  1337 03D4  1377 03DF  1389 03E2  1404 03E5  1419 03E9  1456 03F2  1556 0409  1580 0410
                  1610 0417  1656 0423  1720 0436  1821 044E  1850 0453  1886 045C  1898 045F  1910 0462
                  1925 0466  1944 046A  1971 0472  2001 0476  2013 047A  2019 047C  2055 0486  2067 0489
                  2092 048F  2113 0495  2128 049A  2153 04A0  2171 04A6  2194 04AC  2212 04B1  2229 04B4
                  2244 04B9  2265 04C0
NEG.A       0029
NEG.B       0027
NEG.O       0025   2173 01E3
NEG.S       001F
NEXT        0002   510 0000   516 0002   518 0003   520 0004   522 0005   524 0006   527 0007   529 0008
                   531 0009   533 000A   536 000B   538 000C   540 000D   543 000E   545 000F   547 0010
                   549 0011   560 0012   562 0013   565 0014   567 0015   570 0016   572 0017   575 0018
                   577 0019   580 001A   582 001B   585 001C   587 001D   590 001E   592 001F   595 0020
                   597 0021   600 0022   602 0023   605 0024   621 0026   627 0028   635 002B   638 002C
```

```
NOP             0044    509 0000    630 002A    649 0030    663 0032    669 0034    678 0037    682 0039    686 003A
                        689 003B    692 003C    705 003E    726 0045    729 0046    740 0047    746 0049    755 004C
                        761 004E    764 004F    774 0050    781 0052    783 0053    790 0055    793 0056    808 0058
                        815 005A    817 005B    827 005E    830 005F    845 0062    848 0063    884 0078    939 008B
                        942 008C    957 0091    960 0092    991 009B    994 009C   1004 00A0   1020 00A8   1022 00A9
                       1024 00AA   1026 00AB   1032 00AE   1034 00AF   1044 00B4   1046 00B5   1048 00B6   1050 00B7
                       1068 00BC   1133 00CC   1194 00D8   1202 00DA   1218 00DC   1224 00DE   1236 00E2   1255 00E6
                       1261 00E8   1290 00EE   1308 00F2   1314 00F4   1334 00F8   1337 00F9   1346 00FC   1369 0103
                       1380 0105   1389 0108   1392 0109   1403 010B   1412 010E   1415 010F   1426 0111   1432 0113
                       1441 0116   1444 0117   1455 0119   1461 011B   1470 011E   1473 011F   1484 0121   1490 0123
                       1496 0125   1510 0128   1516 012A   1528 012E   1534 0130   1560 0137   1563 0138   1588 0139
                       1597 013C   1681 0154   1685 0156   1689 0158   1704 015B   1712 015E   1727 0165   1752 016D
                       1764 0171   1772 0172   1778 0174   1792 0177   1876 0191   1888 0195   1911 0197   1920 019A
                       1927 019C   1929 019D   1958 01A4   1964 01A6   1973 01A9   1988 01AC   1991 01AD   1994 01AE
                       2006 01B2   2009 01B3   2064 01C1   2145 01DA   2147 01DB   2161 01DF   2163 01E0   2195 01E7
                       2197 01E8   2211 01EC   2213 01ED   2226 01F1   2250 01F6   2253 01F7   2259 01F9   2262 01FA
                       2298 0202   2302 0204   2308 0206   2314 0208   2323 020B   2345 020F   2357 0213   2403 0223
                       2405 0224   2419 022B   2443 0231   2446 0232   2459 0235   2462 0236   2488 023E   2578 0257
                       2581 0258   2621 0269   2719 0282   2726 0283   2729 0284   2764 0288   2773 028E   2785 0290
                       2789 0292     85 0294     88 0295     91 0296     97 0298    100 0299    106 029B    127 02A2
                        139 02A5    142 02A6    145 02A7    242 02BD    263 02C1    282 02C5    297 02CA    303 02CB
                        318 02CF    327 02D2    334 02D3    367 02DD    377 02DF    392 02E3    438 02F0    452 02F4
                        477 02F8    492 02FF    520 0306    523 0307    532 0309    552 030E    600 0318    612 031C
                        621 031F    630 0322    639 0325    648 0328    655 0329    682 0330    730 033D    736 033F
                        757 0344    781 034C    787 034E    801 0350    850 0359    926 036D    929 036E    938 0371
                        947 0373    953 0375    973 0379    976 037A    982 037C   1003 0382   1006 0383   1109 039E
                       1153 03A9   1237 03BB   1250 03BF   1253 03C0   1263 03C3   1266 03C4   1290 03C9   1354 03DA
                       1368 03DE   1382 03E1   1394 03E4   1412 03E8   1482 03F9   1485 03FA   1524 0403   1530 0405
                       1536 0406   1542 0408   1573 040F   1585 0412   1678 042A   1758 043F   1774 0444   1797 044B
                       1870 0459   1876 045B   1891 045E   1903 0461   1915 0464   1918 0465   1933 0469   1952 046D
                       1967 0471   1973 0473   1976 0474   2003 0477   2006 0478   2036 0482   2121 0498   2133 049C
                       2176 04A8   2217 04B3   2237 04B7   2240 04B8   2249 04BB   2258 04BE   2261 04BF   2267 04C1
                       2301 04C4   2304 04C5   2307 04C6   2322 04CA   2325 04CB   2328 04CC   2345 04CF   2354 04D2
                       2360 04D4   2366 04D6   2378 04E0   2381 04E1   2553 051B   2567 051E   2570 051F   2573 0520
                       2576 0521   2579 0522   2582 0523   2585 0524   2588 0525   2591 0526   2594 0527   2597 0528
                       2600 0529   2603 052A   2606 052B   2609 052C   2632 0532   2635 0533   2748 0555   2751 0556
                       2885 057D   2988 059D   2991 059E   3030 05A9   3134 05C6   3155 05CD   3180 05D4   3189 05D7
                       3201 05DB   3227 05E2   3230 05E3   3233 05E4   3236 05E5   3267 07F4   3269 07F5   3271 07F6
                       3283 07FA   3295 07FE   3300 07FF
NOT.A           0028    2647 026F   2659 0273    355 02D9    361 02DB    561 0311    567 0313
NOT.B           0026
NOT.Q           0024
NOT.S           001E
PASS.A          0008    917 0082    919 0083    935 008A   1090 00BF   1096 00C1   1102 00C3   1105 00C4   1146 00CE
                       1152 00D0   1158 00D2   1221 00DD   1227 00DF   1258 00E7   1264 00E9   1340 00FA   1343 00FB
                       1548 0133   1612 0141   1618 0143   1627 0146   1647 014A   1657 014D   1673 0152   1734 0168
                       1740 016B   1819 017F   1823 0181   1830 0184   1914 0198   2055 01BE   2256 01F8   2305 0205
                       2320 020A   2397 0220   2399 0221   2401 0222   2431 022D   2503 0240   2506 0241   2509 0242
                       2518 0245   2591 025A   2593 025B   2597 025D   2601 025F   2605 0261   2609 0263   2613 0265
                       2617 0267   2633 026D   2635 026E   2656 0272   2668 0276   2707 027E   2755 0288   2758 0289
                       2761 028A   2770 028D     94 0297    103 029A    121 02A0    136 02A4    151 02A8    173 02AD
                        196 02B2    208 02B5    294 02C9    404 02E6    420 02EA    429 02ED    432 02EE    504 0302
                        511 0303    526 0308    615 031D    712 0337    721 033A    724 033B    727 033C    751 0342
                        754 0343    763 0346    769 0348    775 034A    778 034B    914 036A    935 0370    997 0380
                       1024 0387   1166 03AD   1197 03B5   1231 03B9   1240 03BC   1333 03D3   1339 03D5   1345 03D7
                       1348 03D8   1357 03DB   1409 03E7   1424 03E8   1427 03EC   1470 03F6   1479 03F8   1497 03FB
                       1500 03FC   1503 03FD   1558 040A   1612 0418   1615 0419   1619 041A   1622 0418   1631 041E
                       1719 0436   1722 0437   1752 043D   1755 043E   1764 0441   1771 0443   1780 0446   1788 0448
```

```
                    2224 01F0   2365 0214   2420 022B   2489 023E   2513 0243   2522 0246   2537 0248   2579 0257
                    2622 0269   2624 026A   2636 026E   2657 0272   2669 0276   2720 0282   2730 0284   2739 0287
                    2765 028B   2786 0290   2790 0292   2792 0293    104 029A   1013 0385   2484 0503   2502 0509
                    2560 051D   2577 0521   2586 0524   2598 0528   2607 052B   2616 052E   2645 0535   2694 0544
                    2746 0554   2999 05A0   3088 05BA   3116 05C1   3168 05D1   3214 05DF   3240 05E6   3246 05E8
S.AD1.A     000B    1835 0452
S.AD1.Q     000D
S.ADD.A     000A    1093 00C0   1099 00C2   1597 0413   1749 043C   1814 044D   1829 0450   1832 0451   3105 058E
S.ADD.Q     000C
S.AND.A     004A    1087 00BE   1118 00C7   1127 00CA   1155 00D1   1239 00E3   1270 00EB   1311 00F3   1386 0107
                    1409 010D   1438 0115   1467 011D   1493 0124   1525 012D   1557 0136   1683 0155   1775 0173
                    1789 0176   1808 017A   2472 0237   2533 024A   2631 026C   2650 0270   2662 0274   1867 0458
                    1873 045A   2115 0496   2161 04A3   2231 04B5
S.AND.Q     004C    1161 00D3   1758 016F   1806 0179   1835 0186
S.BCL.A     005A    2713 0280   2738 0287    272 02C4    679 032F    882 0361   1115 03A0   1458 03F3   1518 0402
                    1527 0404   1539 0407   1735 043A   1738 043B   1820 044E   1970 0472   2103 0493   2264 04C0
                    2372 04D8   2384 04E2
S.BCL.Q     005C
S.IOR.A     003A     969 0095    972 0096    975 0097   1006 00A1   1008 00A2   1010 00A3   1012 00A4   1014 00A5
                    1016 00A6   1018 00A7   1052 00B8   1149 00CF   1242 00E4   1273 00EC   1504 0126   1591 013A
                    1667 0150   2269 01FB   2311 0207   2326 020C   2515 0244   2524 0247   2530 0249   2653 0271
                    2665 0275   2716 0281   2732 0285   2735 0286   2767 028C    182 02B0    230 02B9    441 02F1
                     676 032E    706 0336    733 033E    784 034D    868 035E    885 0362    944 0372    950 0374
                     988 037E   1112 039F   1136 03A4   1156 03AA   1172 03AF   1200 03B6   1203 03B7   1342 03D6
                    1430 03ED   1506 03FE   1570 040E   1640 0421   1643 0422   1716 0435   2024 047E   2106 0494
                    2136 049D   2149 049F   2199 04AE   2387 04E3   2396 04E4   2398 04E5   2400 04E6   2402 04E7
                    2404 04E8   2406 04E9   2408 04EA   2410 04EB   2412 04EC   2414 04ED   2416 04EE   2418 04EF
                    2420 04F0   2422 04F1   2424 04F2   2426 04F3   2428 04F4   2430 04F5   2432 04F6   2434 04F7
                    2436 04F8   2438 04F9   2440 04FA   2465 04FE
S.IOR.Q     003C    1837 0187   2595 025C   2599 025E   2603 0260   2607 0262   2611 0264   2615 0266   2619 0268
                    2623 026A   1727 0438   1732 0439
S.SB1.A     002A
S.SB1.Q     002C
S.SUB.A     002B     435 02EF   1284 03C7   1509 03FF   1512 0400   1634 041F
S.SUB.Q     002D     982 0098
S.XNOR.A    007A    1406 03E6   1421 03EA   1912 0463   1927 0467
S.XNOR.Q    007C
S.XOR.A     006A    1939 01A0   2049 01BC   1130 03A2   1247 03BE   1257 03C1   1260 03C2   1287 03C8   1330 03D2
                    1336 03D4   1564 040C   1567 040D   1600 0414   1672 0428   1855 0455   1946 0468   2012 047A
                    2018 047C   2234 0486   2348 04D0   2351 04D1   2813 0568   2816 0569   2819 056A   2822 056B
                    2825 056C   2828 056D   2831 056E   2834 056F   2837 0570   2840 0571   2843 0572   2846 0573
                    2849 0574   2852 0575   2855 0576   2858 0577   2861 0578   2882 057C   3072 05B5   3137 05C7
                    3158 05CE   3192 05D8   3204 05DC   3224 05E1
S.XOR.Q     006C     711 0040    714 0041    720 0043    749 004A    758 004D   1059 00B9   1062 00BA   1215 00DB
                    1252 00E5   1287 00ED   1377 0104   1400 010A   1423 0110   1452 0118   1481 0120   1507 0127
                    1594 013B   1698 0159   1701 015A   1761 0170   1923 019B   1961 01A5   2000 01B0   2003 01B1
                    2375 0215   2377 0216   2379 0217   2381 0218   2383 0219   2385 021A   2387 021B   2389 021C
                    2391 021D   2393 021E   2395 021F   2407 0225   2409 0226   2411 0227   2413 0228   2415 0229
                    2417 022A   2475 0238   2477 0239   2479 023A   2481 023B   2483 023C   2485 023D   2864 0579
                    2897 0581
ZERO        0044     523 0006    542 000E    544 000F    899 007A    922 0084    924 0085    926 0086   1664 014F
                    2083 01C5   2085 01C6   2087 01C7   2089 01C8   2091 01C9   2093 01CA   2095 01CB   2097 01CC
                    2106 01D0   2112 01D3   2120 01D4   2122 01D5   2124 01D6   2126 01D7   2589 0259    170 02AC
                     188 02B1    269 02C3    407 02E7    444 02F2    609 031B    645 0327    879 0360    888 0363
                     897 0366   1000 0381   1278 03C5   1418 03E9   1473 03F7   1555 0409   1655 0423   2623 052F
                    3108 05BF   3146 05CA   3161 05CF   3195 05D9   3207 05DD   3239 05E6
```

```
CIN             0011    526 0007    708 003F    749 004A    796 0057   1961 01A5   1997 01AF
CLRTC           0013    666 0033   2141 01D9   2157 01DE   2191 01E6   2207 01EB   2351 0211    932 036F    979 037B
                       1150 03A8   2316 04C8   2339 04CD   3122 05C2   3128 05C4   3143 05C9   3149 05CB   3280 07F9
DIN             0010    528 0008    672 0035   2223 01F0    176 02AE
DRSTS           0012    931 0088   2037 01B8   2364 0214    970 0378   1146 03A7   2483 0503   2501 0509
LIT             0014    515 0002    519 0004    535 000B    537 000C    539 000D    546 0010    559 0012    564 0014
                        569 0016    574 0018    579 001A    584 001C    589 001E    594 0020    599 0022    604 0024
                        620 0026    622 0027    634 002B    637 002C    640 002D    643 002E    646 002F    702 003D
                        711 0040    714 0041    717 0042    720 0043    723 0044    758 004D    777 0051    811 0059
                        840 0060    855 0070    858 0071    861 0072    864 0073    867 0074    870 0075    878 0076
                        881 0077    901 007B    909 007E    911 007F    913 0080    915 0081    963 0093    969 0095
                        972 0096    975 0097   1006 00A1   1008 00A2   1010 00A3   1012 00A4   1014 00A5   1016 00A6
                       1018 00A7   1028 00AC   1030 00AD   1036 00B0   1038 00B1   1040 00B2   1042 00B3   1052 00B8
                       1059 00B9   1062 00BA   1087 00BE   1093 00C0   1099 00C2   1118 00C7   1127 00CA   1149 00CF
                       1155 00D1   1161 00D3   1170 00D6   1191 00D7   1199 00D9   1215 00DB   1230 00E0   1239 00E3
                       1242 00E4   1252 00E5   1267 00EA   1270 00EB   1273 00EC   1287 00ED   1296 00F0   1302 00F1
                       1311 00F3   1350 00FD   1353 00FE   1356 00FF   1359 0100   1366 0102   1377 0104   1383 0106
                       1386 0107   1400 010A   1406 010C   1409 010D   1423 0110   1435 0114   1438 0115   1452 0118
                       1464 011C   1467 011D   1481 0120   1487 0122   1493 0124   1504 0126   1507 0127   1519 012B
                       1522 012C   1525 012D   1542 0131   1545 0132   1551 0134   1554 0135   1557 0136   1591 013A
                       1594 013B   1600 013D   1603 013E   1624 0145   1630 0147   1641 0148   1651 014B   1667 0150
                       1670 0151   1677 0153   1683 0155   1698 0159   1701 015A   1716 0160   1718 0161   1749 016C
                       1758 016F   1761 0170   1775 0173   1786 0175   1789 0176   1806 0179   1808 017A   1813 017C
                       1815 017D   1817 017E   1821 0180   1825 0182   1828 0183   1835 0186   1858 018B   1864 018D
                       1867 018E   1870 018F   1873 0190   1879 0192   1882 0193   1885 0194   1908 0196   1917 0199
                       1939 01A0   1955 01A3   2000 01B0   2003 01B1   2024 01B4   2027 01B5   2030 01B6   2033 01B7
                       2052 01BD   2100 01CD   2102 01CE   2108 01D1   2138 01D8   2188 01E5   2236 01F2   2239 01F3
                       2243 01F4   2269 01FB   2282 01FC   2284 01FD   2286 01FE   2290 01FF   2292 0200   2311 0207
                       2326 020C   2335 020D   2338 020E   2375 0215   2377 0216   2379 0217   2381 0218   2383 0219
                       2385 021A   2387 021B   2389 021C   2391 021D   2393 021E   2395 021F   2407 0225   2409 0226
                       2411 0227   2413 0228   2415 0229   2417 022A   2421 022C   2472 0237   2475 0238   2477 0239
                       2479 023A   2481 023B   2483 023C   2485 023D   2491 023F   2512 0243   2515 0244   2521 0246
                       2524 0247   2527 0248   2530 0249   2533 024A   2545 024C   2548 024D   2551 024E   2554 024F
                       2557 0250   2560 0251   2563 0252   2566 0253   2595 025C   2599 025E   2603 0260   2607 0262
                       2611 0264   2615 0266   2619 0268   2623 026A   2631 026C   2677 0277   2681 0278   2687 027A
                       2690 027B   2693 027C   2704 027D   2713 0280   2716 0281   2732 0285   2735 0286   2738 0287
                       2767 028C    154 02A9    157 02AA    160 02AB    179 02AF    182 02B0    214 02B7    225 02B8
                        230 02B9    250 02BF    253 02C0    266 02C2    272 02C4    285 02C6    291 02C8    306 02CC
                        309 02CD    312 02CE    324 02D1    337 02D4    343 02D5    374 02DE    380 02E0    386 02E2
                        395 02E4    401 02E5    410 02E8    413 02E9    441 02F1    455 02F5    471 02F9    474 02FA
                        480 02FC    486 02FE    495 0300    501 0301    535 030A    555 030F    579 0316    582 0317
                        603 0319    618 031E    624 0320    627 0321    633 0323    636 0324    658 032A    670 032C
                        676 032E    679 032F    700 0334    706 0336    715 0338    733 033E    745 0340    784 034D
                        790 034F    804 0351    813 0353    819 0355    822 0356    844 0357    847 0358    853 035A
                        856 035B    859 035C    868 035E    876 035F    882 0361    885 0362    891 0364    894 0365
                        908 0368    911 0369    917 036B    920 036C    944 0372    950 0374    961 0376    964 0377
                        988 037E   1012 0385   1027 0388   1030 0389   1033 038A   1036 038B   1039 038C   1042 038D
                       1045 038E   1048 038F   1051 0390   1054 0391   1057 0392   1060 0393   1063 0394   1066 0395
                       1069 0396   1072 0397   1083 0398   1086 0399   1089 039A   1103 039C   1106 039D   1112 039F
                       1115 03A0   1127 03A1   1130 03A2   1136 03A4   1140 03A5   1156 03AA   1169 03AE   1172 03AF
                       1175 03B0   1191 03B3   1200 03B6   1203 03B7   1228 03B8   1247 03BE   1257 03C1   1260 03C2
                       1293 03CA   1296 03CB   1299 03CC   1308 03CD   1314 03CF   1330 03D2   1336 03D4   1342 03D6
                       1351 03D9   1360 03DC   1363 03DD   1430 03ED   1436 03EF   1442 03F1   1455 03F2   1458 03F3
                       1464 03F5   1506 03FE   1509 03FF   1518 0402   1527 0404   1539 0407   1564 040C   1567 040D
                       1570 040E   1597 0413   1600 0414   1603 0415   1640 0421   1643 0422   1658 0424   1675 0429
                       1681 042B   1695 042C   1697 042D   1699 042E   1701 042F   1703 0430   1705 0431   1707 0432
                       1709 0433   1711 0434   1716 0435   1727 0438   1732 0439   1735 043A   1738 043B   1749 043C
                       1814 044D   1820 044E   1823 044F   1829 0450    832 0451   1835 0452   1855 0455   1858 0456
```

```
CLCCO       0015   530 0009   787 0054   824 005D  1933 019E
CLDCO       0014   532 000A  1531 012F  1687 0157  2154 01DD   211 0286
COUT        0011   777 0051   811 0059  1923 019B
DOUT        0010  2176 01E4   199 02B3
DRCON       0019   535 0008  2236 01F2  2239 01F3  2282 01FC  2284 01FD  2286 01FE   844 0357   847 0358
                   853 035A   859 035C  2480 0502  2495 0507  2498 0508
DRSEL       0018   660 0031   905 007D   954 0090  1167 00D5  1855 018A  2024 01B4  2058 01BF   914 036A
                  1009 0384  3125 05C3  3140 05C8
R0          0000   717 0042   723 0044   752 004B   931 0088   969 0095   972 0096   975 0097  1006 00A1
                  1008 00A2  1010 00A3  1012 00A4  1014 00A5  1016 00A6  1018 00A7  1052 00B8  1504 0126
                  1591 013A  2037 01B8  2049 01BC  2364 0214   985 037D  1146 03A7  1293 03CA  1418 03E9
                  1675 0429  1832 0451  1835 0452  2319 04C9  2342 04CE  2483 0503  2501 0509  2623 052F
                  2745 0554  2758 0557  2813 0568  2816 0569  2900 0582  3006 05A1  3012 05A3  3018 05A5
                  3024 05A7  3054 05AF  3060 05B1  3069 05B4  3072 05B5  3078 05B7  3084 05B9  3096 05BB
                  3105 05BE  3274 07F7  3289 07FC  3292 07FD
R1          0001   643 002E   646 002F   901 007B  1170 00D6  1191 00D7  1199 00D9  1730 0166  1736 0169
                  1813 017C  1817 017E  1821 0180  1825 0182  1858 018B  1885 0194  1914 0198  1936 019F
                  1939 01A0  1967 01A7  2052 01BD  2421 022C  2491 023F  2629 026B  2631 026C  2633 026D
                  2635 026E  2647 026F  2650 0270  2653 0271  2659 0273  2662 0274  2665 0275  2707 027E
                   233 02BA   250 02BF   253 02C0   288 02C7   312 02CE   321 02D0   358 02DA   374 02DE
                   383 02E1   474 02FA   483 02FD   558 0310   606 031A   636 0324   642 0326   691 0331
                   766 0347  1030 0389  1042 038D  1054 0391  1066 0395  2556 051C  2612 052D  2626 0530
                  2629 0531  2641 0534  2761 0558  2819 056A  2888 057E  2909 0583  2995 059F  3009 05A2
                  3015 05A4  3021 05A6  3027 05A8  3108 05BF  3112 05C0  3131 05C5  3137 05C7  3146 05CA
                  3152 05CC  3158 05CE  3161 05CF  3164 05D0  3186 05D6  3192 05D8  3195 05D9  3198 05DA
                  3204 05DC  3207 05DD  3210 05DE  3224 05E1  3242 05E7  3277 07F8  3286 07FB
R2          0002  1084 008D  1087 008E  1093 00C0  1099 00C2  1115 00C6  1118 00C7  1124 00C9  1127 00CA
                  1143 00CD  1149 00CF  1155 00D1  1158 00D2  1908 0196  1970 01A8  2713 0280  2716 0281
                   239 02BC   352 02D8   364 02DC   461 02F7   471 02F9   541 030C   564 0312   694 0332
                   697 0333   772 0349  1027 0388  1039 038C  1051 0390  1063 0394  2651 0536  2700 0545
                  2764 0559  2822 056B  2912 0584  3045 05AC
R3          0003  1976 01AA  2704 027D  2710 027F   435 02EF  2654 0537  2703 0546  2767 055A  2825 056C
                  2915 0585  2942 058E
R4          0004   640 002D   881 0077  1864 018D  1873 0190  2033 01B7  2693 027C   170 02AC   176 02AE
                   188 02B1   444 02F2  2657 0538  2706 0547  2770 055B  2828 056D  2918 0586
R5          0005   909 007E   913 0080   924 0085  1519 012B  1600 013D  1808 017A  2732 0285  2735 0286
                  2738 0287   124 02A1   130 02A3   154 02A9   157 02AA   160 02AB   179 02AF   182 02B0
                   202 02B4   214 02B7   441 02F1   706 0336   733 033E   784 034D   879 0360  1172 03AF
                  1200 03B6  1203 03B7  1458 03F3  1506 03FE  1570 040E  1640 0421  1643 0422  1820 044E
                  1867 0458  1873 045A  2149 049F  2199 04AE  2231 04B5  2384 04E2  2465 04FE  2660 0539
                  2709 0548  2773 055C  2831 056E  2921 0587
R6          0006   911 007F   915 0081   926 0086  1296 00F0  1383 0106  1406 010C  1435 0114  1464 011C
                  1487 0122  1522 012C  1603 013E  1749 016C   230 02B9   272 02C4   676 032E   679 032F
                   748 0341   882 0361   885 0362  1112 039F  1115 03A0  1136 03A4  1430 03ED  1970 0472
                  2103 0493  2106 0494  2136 049D  2264 04C0  2372 04D8  2663 053A  2712 0549  2776 055D
                  2834 056F
R7          0007  1239 00E3  1242 00E4  1270 00EB  1273 00EC  1311 00F3  1386 0107  1409 010D  1438 0115
                  1467 011D  1493 0124  1525 012D  1557 0136  1683 0155  1775 0173  1786 0175  1789 0176
                  2269 01F8  2311 0207  2326 020C  2437 022F  2472 0237  2515 0244  2524 0247  2530 0249
                  2533 024A  2536 024B  2767 028C   944 0372   950 0374   988 037E  1012 0385  1156 03AA
                  1342 03D6  2024 047E  2387 04E3  2396 04E4  2398 04E5  2400 04E6  2402 04E7  2404 04E8
                  2406 04E9  2408 04EA  2410 04EB  2412 04EC  2414 04ED  2416 04EE  2418 04EF  2420 04F0
                  2422 04F1  2424 04F2  2426 04F3  2428 04F4  2430 04F5  2432 04F6  2434 04F7  2436 04F8
                  2438 04F9  2440 04FA  2666 053B  2715 054A  2779 055E  2837 0570
R8          0008  1624 0145  1641 0148  1651 014B  1667 0150  1436 03EF  1518 0402  1527 0404  1539 0407
                  1716 0435  1735 043A  1738 043B  1749 043C  1768 0442  1855 0455  2669 053C  2718 054B
                  2782 055F  2840 0571
R9          0009  1606 013F  1278 03C5  1391 03E3  1421 03EA  1509 03FF  1512 0400  1515 0401  1555 0409
```

```
 455 02F5    480 02FC    501 0301    535 030A    555 030F    579 0316    603 0319    670 032C
 700 0334    715 0338    745 0340    790 034F    804 0351    813 0353    819 0355    894 0365
2492 0506   3174 05D2   3177 05D3
```

B 0003 620 0026 622 0027 634 0028 640 002D 643 002E 646 002F 676 0036 680 0038
702 003D 717 0042 723 0044 743 0048 752 004B 855 0070 858 0071 861 0072
864 0073 867 0074 870 0075 881 0077 899 007A 901 007B 909 007E 911 007F
913 0080 915 0081 924 0085 926 0086 931 0088 1084 008D 1087 008E 1093 00C0
1099 00C2 1105 00C4 1112 00C5 1115 00C6 1118 00C7 1121 00C8 1124 00C9 1127 00CA
1143 00CD 1149 00CF 1170 00D6 1191 00D7 1199 00D9 1230 00E0 1242 00E4 1267 00EA
1273 00EC 1296 00F0 1302 00F1 1328 00F6 1331 00F7 1350 00FD 1353 00FE 1356 00FF
1359 0100 1363 0101 1366 0102 1383 0106 1406 010C 1435 0114 1464 011C 1487 0122
1519 012B 1522 012C 1600 013D 1603 013E 1606 013F 1609 0140 1621 0144 1624 0145
1641 0148 1651 014B 1661 014E 1667 0150 1670 0151 1730 0166 1736 0169 1749 016C
1786 0175 1808 017A 1813 017C 1817 017E 1821 0180 1825 0182 1858 018B 1864 018D
1867 018E 1873 0190 1879 0192 1885 0194 1908 0196 1914 0198 1936 019F 1942 01A1
1967 01A7 1970 01A8 1976 01AA 1979 01AB 2024 01B4 2030 01B6 2033 01B7 2037 01B8
2040 01B9 2046 01BB 2052 01BD 2058 01BF 2061 01C0 2102 01CE 2104 01CF 2110 01D2
2138 01D8 2151 01DC 2167 01E1 2188 01E5 2201 01E9 2217 01EE 2243 01F4 2247 01F5
2269 01FB 2290 01FF 2292 0200 2296 0201 2300 0203 2311 0207 2326 020C 2335 020D
2338 020E 2348 0210 2354 0212 2364 0214 2421 022C 2437 022F 2491 023F 2506 0241
2515 0244 2524 0247 2530 0249 2533 024A 2536 024B 2545 024C 2548 024D 2551 024E
2554 024F 2557 0250 2560 0251 2563 0252 2566 0253 2569 0254 2572 0255 2575 0256
2647 026F 2650 0270 2653 0271 2659 0273 2662 0274 2665 0275 2687 027A 2693 027C
2704 027D 2710 027F 2732 0285 2735 0286 2738 0287 2767 028C 2783 028F 2787 0291
2791 0293 124 02A1 130 02A3 154 02A9 157 02AA 160 02AB 170 02AC 176 02AE
179 02AF 182 02B0 188 02B1 202 02B4 214 02B7 230 02B9 233 02BA 239 02BC
250 02BF 253 02C0 272 02C4 288 02C7 312 02CE 321 02D0 352 02D8 358 02DA
364 02DC 374 02DE 383 02E1 407 02E7 426 02EC 441 02F1 444 02F2 461 02F7
471 02F9 474 02FA 483 02FD 511 0303 517 0305 541 030C 558 0310 564 0312
606 031A 636 0324 642 0326 676 032E 679 032F 691 0331 694 0332 697 0333
706 0336 733 033E 766 0347 772 0349 784 034D 865 035D 868 035E 879 0360
882 0361 885 0362 888 0363 891 0364 908 0368 911 0369 917 036B 920 036C
944 0372 950 0374 961 0376 964 0377 985 037D 988 037E 1000 0381 1012 0385
1027 0388 1030 0389 1033 038A 1036 038B 1039 038C 1042 038D 1045 038E 1048 038F
1051 0390 1054 0391 1057 0392 1060 0393 1063 0394 1066 0395 1069 0396 1072 0397
1083 0398 1086 0399 1089 039A 1103 039C 1106 039D 1112 039F 1115 03A0 1133 03A3
1136 03A4 1145 03A7 1156 03AA 1160 03AB 1163 03AC 1169 03AE 1172 03AF 1175 03B0
1185 03B1 1188 03B2 1191 03B3 1200 03B6 1203 03B7 1228 03B8 1243 03BD 1278 03C5
1293 03CA 1296 03CB 1299 03CC 1327 03D1 1342 03D6 1351 03D9 1360 03DC 1379 03E0
1391 03E3 1418 03E9 1430 03ED 1436 03EF 1439 03F0 1442 03F1 1458 03F3 1506 03FE
1515 0401 1518 0402 1527 0404 1539 0407 1555 0409 1561 040B 1570 040E 1637 0420
1640 0421 1643 0422 1655 0423 1675 0429 1681 042B 1716 0435 1735 043A 1738 043B
1749 043C 1768 0442 1820 044E 1829 0450 1832 0451 1835 0452 1849 0453 1855 0455
1858 0456 1888 045D 1900 0460 1930 0468 1949 046C 1964 0470 1970 0472 2009 0479
2024 047E 2033 0481 2042 0484 2078 048D 2081 048E 2103 0493 2106 0494 2127 049A
2136 049D 2149 049F 2152 04A0 2158 04A2 2185 04AB 2199 04AE 2214 04B2 2246 04BA
2255 04BD 2264 04C0 2313 04C7 2319 04C9 2342 04CE 2357 04D3 2363 04D5 2372 04D8
2384 04E2 2387 04E3 2396 04E4 2398 04E5 2400 04E6 2402 04E7 2404 04E8 2406 04E9
2408 04EA 2410 04EB 2412 04EC 2414 04ED 2416 04EE 2418 04EF 2420 04F0 2422 04F1
2424 04F2 2426 04F3 2428 04F4 2430 04F5 2432 04F6 2434 04F7 2436 04F8 2438 04F9
2440 04FA 2465 04FE 2468 04FF 2483 0503 2501 0509 2556 051C 2612 052D 2623 052F
2626 0530 2641 0534 2651 0536 2654 0537 2657 0538 2660 0539 2663 053A 2666 053B
2669 053C 2672 053D 2675 053E 2678 053F 2681 0540 2684 0541 2687 0542 2690 0543
2745 0554 2758 0557 2761 0558 2764 0559 2767 055A 2770 055B 2773 055C 2776 055D
2779 055E 2782 055F 2785 0560 2788 0561 2791 0562 2794 0563 2797 0564 2800 0565
2803 0566 2879 057B 2888 057E 2900 0582 2909 0583 2912 0584 2915 0585 2918 0586
2921 0587 2924 0588 2927 0589 2939 058D 2952 0591 2955 0592 2970 0597 2976 0599
2995 059F 3006 05A1 3009 05A2 3012 05A3 3015 05A4 3018 05A5 3021 05A6 3024 05A7
3027 05A8 3039 05AA 3042 05AB 3045 05AC 3051 05AE 3054 05AF 3057 05B0 3060 05B1
3063 05B2 3066 05B3 3069 05B4 3075 05B6 3078 05B7 3081 05B8 3084 05B9 3096 05BB
3099 05BC 3108 05BF 3112 05C0 3131 05C5 3146 05CA 3152 05CC 3161 05CF 3164 05D0

		3186	05D6	3195	05D9	3198	05DA	3207	05DD	3210	05DE	3242	05E7	3274	07F7	3277	07F8
		3286	07F8	3289	07FC	3292	07FD										
B.A	0002	842	0061	2964	0595												
BL	0007	2707	027E	2060	0488	2069	048A										
BLQL	0006																
BR	0005	1155	00D1	1158	00D2	1708	015C	1710	015D	2629	026B	2631	026C	2633	026D	2635	026E
		2713	0280	2716	0281												
BRQR	0004	2979	059A														
NOP	0001																
Q	0000	515	0002	519	0004	546	0010	559	0012	564	0014	569	0016	574	0018	579	001A
		584	001C	589	001E	594	0020	599	0022	604	0024	708	003F	749	004A	840	0060
		963	0093	966	0094	969	0095	972	0096	975	0097	1006	00A1	1008	00A2	1010	00A3
		1012	00A4	1014	00A5	1016	00A6	1018	00A7	1028	00AC	1030	00AD	1036	00B0	1038	00B1
		1040	00B2	1042	00B3	1052	00B8	1152	00D0	1161	00D3	1239	00E3	1270	00E8	1311	00F3
		1386	0107	1409	010D	1438	0115	1467	011D	1493	0124	1504	0126	1525	012D	1542	0131
		1551	0134	1557	0136	1591	013A	1630	0147	1664	014F	1673	0152	1683	0155	1716	0160
		1718	0161	1738	016A	1755	016E	1758	016F	1775	0173	1789	0176	1804	0178	1806	0179
		1833	0185	1835	0186	1837	0187	1917	0199	1923	019B	1955	01A3	1997	01AF	2070	01C3
		2170	01E2	2220	01EF	2472	0237	2512	0243	2521	0246	2527	0248	2589	0259	2595	025C
		2599	025E	2603	0260	2607	0262	2611	0264	2615	0266	2619	0268	2623	026A	2677	0277
		2681	0278	447	02F3	970	0378	1695	042C	1697	042D	1699	042E	1701	042F	1703	0430
		1705	0431	1707	0432	1709	0433	1711	0434	1727	0438	1732	0439	2075	048C	2693	0544
		2806	0567	2871	057A	2891	057F	2894	0580	2930	058A	2933	058B	2942	058E	2949	0590
		2958	0593	2961	0594	2973	0598										

```
ALWAYS      0000  2786 0290  2790 0292    92 0296   104 029A  2370 04D7  2749 0555
BITO        0008  1430 0112  1459 011A  1514 0129  1726 0164  2435 022E  2441 0230  2618 0267   248 02BE
                   405 02E6  1101 039B  1195 03B4  1501 03FC  1616 0419  1723 0437  2098 0491  2147 049E
                  2165 04A4  2174 04A7  2197 04AD  2534 0512
BIT1        0009  1820 017F  2519 0245  2614 0265  2771 028D   421 02EA   713 0337   752 0342  2536 0513
BIT2        000A  1085 008D  2610 0263   936 0370   971 0378  1279 03C5  1349 03D8  1404 03E5  1419 03E9
                  1580 0410  1607 0416  1610 0417  1656 0423  1667 0426  1865 0457  1910 0462  1925 0466
                  2031 0480  2229 04B4  2538 0514
BIT3        0008   897 0079  1168 00D5  2510 0242  2606 0261  2762 028A  1340 03D5  2022 047D  2540 0515
BIT4        000C  2306 0205  2602 025F   122 02A0   125 02A1   137 02A4   152 02A8   174 02AD   197 02B2
                   203 0284   209 0285  1471 03F6  2487 0504  2542 0516
BIT5        000D  2402 0222   998 0380  1346 03D7  1358 03DB  1756 043E  1772 0443  2028 047F  2040 0483
                  2457 04FB  2475 0500  2544 0517
BIT6        000E  2400 0221  2594 025B    95 0297  2546 0518
BIT7        000F   822 005C  1715 015F  1824 0181  2398 0220  2708 027E   674 032D  1504 03FD  1559 040A
                  1720 0436  1853 0454  2548 0519
BORROW      0039   681 0038   843 0061  1332 00F7  1622 0144  1971 01A8  2152 01DC  2168 01E1  2202 01E9
                  2218 01EE  2248 01F5  2570 0254  2573 0255   365 02DC   436 02EF   698 0333   773 0349
                  1440 03F0  1635 041F  3287 07FB  3290 07FC
CARRY       0019  1815 044D  2895 0580
CCI         0013   703 003D   706 003E   744 0048   747 0049   791 0055   794 0056  1956 01A3  1959 01A4
                  1992 01AD  1995 01AE
CCO         0011
CFI         0012   762 004E   765 004F   775 0050   797 0057   809 0058  1918 0199  1921 019A  1943 01A1
                  1946 01A2
CFO         0010   664 0032   679 0037   683 0039   778 0051   784 0053   812 0059   818 005B  1909 0196
                  1924 019B  1930 019D
CRCERR      001E  1670 0427  2131 049B  3234 05E4
DCI         0017   667 0033   171 02AC
DCO         0015
DFI         0016  2148 01DB   140 02A5
DFO         0014  2198 01E8
DPE         0002
NBITO       0028   904 007C  1066 00BB  1628 0146   749 0341   906 0367  1022 0386  1769 0442  2073 048B
                  2119 0497  2518 050A
NBIT1       0029  1234 00E1  1862 018C  1167 03AD  1198 03B5  2520 050B
NBIT2       002A   934 0089  1344 00FB  2504 0240  2759 0289  1025 0387  1235 038A  1325 03D0  1377 03DF
                  1389 03E2  1428 03EC  1556 0409  1753 043D  1886 045C  1898 045F  1944 046A  1959 046E
                  2055 0486  2522 050C
NBIT3       002B   930 0087   295 02C9   616 031D  1334 03D3  2016 047B  2524 050D
NBIT4       002C   986 0099  2321 020A  2526 050E
NBIT5       002D  2365 0214  2507 0241  2598 025D  2756 0288  2299 04C3  2528 050F
NBIT6       002E   629 0029  1222 00DD  1259 00E7  2257 01F8  2432 022D  1232 03B9  2530 0510
NBIT7       002F   627 0028  1294 00EF  1853 0189  2592 025A  1241 03BC  1613 0418  2532 0511
NBORROW     0019   677 0036   983 0098  1326 00F5  1616 0142  1662 014E  2105 01CF  2111 01D2  2297 0201
                  2301 0203  2349 0210  2355 0212  2711 027F  2784 028F  2788 0291  2792 0293   234 02BA
                   240 02BC   289 02C7   322 02D0   384 02E1   462 02F7   484 02FD   542 030C   607 031A
                   643 0326   692 0331   767 0347  1285 03C7  1434 03EE  1850 0453  1965 0470  2153 04A0
                  2256 04BD  2314 04C7  3052 05AE  3058 05B0  3076 05B6  3082 05B8
NCARRY      0039  1598 0413  2358 04D3  2746 0554  2928 0589
NCCI        0033
NCCO        0031
NCFI        0032   782 0052   816 005A  1928 019C
NCFO        0030   718 0042   724 0044   741 0047   756 004C  1989 01AC   892 0364
NCRCERR     003E  2212 04B1
NDCI        0037  2214 01ED
NDCO        0035
NDFI        0036  2164 01E0  2727 0283   143 02A6   146 02A7
```

```
NDFO        0034   128 02A2   131 02A3
NDPE        0022   177 02AE
NEVER       0020  1240 00E3  1271 00EB  1312 00F3  1387 0107  1410 010D  1439 0115  1468 011D  1494 0124
                  1526 012D  1558 0136  1684 0155  1776 0173  1790 0176  1965 01A6  1974 01A9  2007 01B2
                  2139 01D8  2142 01D9  2155 01DD  2158 01DE  2177 01E4  2192 01E6  2205 01EA  2208 01EB
                  2227 01F1  2447 0232  2463 0236    89 0295   231 02B9   264 02C1   273 02C4   283 02C5
                   304 02CB   307 02CC   319 02CF   335 02D3   338 02D4   347 02D6   350 02D7   356 02D9
                   378 02DF   393 02E3   396 02E4   445 02F2   453 02F4   459 02F6   478 02FB   493 02FF
                   496 0300   524 0307   533 0309   539 0308   553 030E   562 0311   568 0313   571 0314
                   574 0315   601 0318   613 031C   622 031F   625 0320   631 0322   640 0325   656 0329
                   659 032A   665 032B   683 0330   695 0332   707 0336   737 033F   788 034E   802 0350
                   808 0352   817 0354  1104 039C  1107 039D  1343 03D6  1437 03EF  2025 047E  3103 058D
                  3278 07F8  3293 07FD
NOVFLOW     003B  2872 057A
NPFW        0027   687 003A
NRDD        003D  1113 039F  2340 04CD  2343 04CE
NRDDX       003F  2302 04C4  2317 04C8  3237 05E5
NRDL        0026
NRDR        0024    86 0294
NRTC        0021   661 0031  2146 01DA  2162 01DF  2196 01E7  2212 01EC  2346 020F   927 036D   974 0379
                  1128 03A1  1141 03A5  3126 05C3  3132 05C5  3147 05CA  3153 05CC  3284 07FA
NSIGN       003A  2062 01C0  1480 03F8
NSTKFULL    001C  2577 0521  2586 0524  2595 0527  2598 0528  2607 052B
NWDL        0023  3199 05DA
NWDR        0025  3178 05D3  3181 05D4  3187 05D6
NZERO       0038   712 0040   715 0041   721 0043   753 004B  1216 00DB  1253 00E5  1288 00ED  1341 00FA
                  1378 0104  1401 010A  1424 0110  1453 0118  1482 0120  1508 0127  1549 0133  1595 013B
                  1699 0159  1702 015A  1739 016A  1940 01A0  1962 01A5  2004 01B1  2050 01BC  2318 0209
                  2418 022A  2454 0233  2457 0234  2486 023D   359 02DA   433 02EE   725 033B   770 0348
                   776 034A   995 037F  1131 03A2  1261 03C2  1288 03C8  1337 03D4  1422 03EA  1510 03FF
                  1513 0400  1565 040C  1568 040D  1632 041E  1868 0458  1874 045A  1962 046F  2019 047C
                  2113 0495  2116 0496  2156 04A1  2162 04A3  2232 04B5  2349 04D0  2490 0505  2630 0531
                  2701 0545  2704 0546  2707 0547  2710 0548  2713 0549  2716 054A  2719 054B  2722 054C
                  2725 054D  2728 054E  2731 054F  2734 0550  2737 0551  2740 0552  2817 0569  2820 056A
                  2823 0568  2826 056C  2829 056D  2832 056E  2835 056F  2838 0570  2841 0571  2844 0572
                  2847 0573  2850 0574  2853 0575  2856 0576  2859 0577  2862 0578  2865 0579  2883 057C
                  2898 0581  2937 058C  2946 058F  2956 0592  2962 0594  2965 0595  2968 0596  2983 059B
                  2986 059C  3073 05B5  3138 05C7  3193 05D8  3225 05E1
OVFLOW      001B
PFW         0007  1488 0122  1750 016C
RDD         001D
RDDX        001F
RDL         0006  1816 017D  2406 0224  3231 05E3
RDR         0004  1110 039E  1474 03F7  3228 05E2
RTC         0001    98 0298
SIGN        001A  1498 03FB  1812 044C
STKFULL     003C  2604 052A
WDL         0003  1811 017B  2404 0223
WDR         0005   101 0299
ZERO        0018   759 004D   967 0094  1060 00B9  1063 00BA  1228 00DF  1265 00E9  1762 0170  1937 019F
                  2001 01B0  2171 01E2  2221 01EF  2376 0215  2378 0216  2380 0217  2382 0218  2384 0219
                  2386 021A  2388 021B  2390 021C  2392 021D  2394 021E  2396 021F  2408 0225  2410 0226
                  2412 0227  2414 0228  2416 0229  2476 0238  2478 0239  2480 023A  2482 023B  2484 023C
                   427 02EC   518 0305   728 033C   779 034B  1144 03A6  1248 03BE  1258 03C1  1331 03D2
                  1407 03E6  1601 0414  1673 0428  1856 0455  1913 0463  1928 0467  1947 046B  1968 0471
                  2001 0476  2013 047A  2235 04B6  2244 04B9  2253 04BC  2352 04D1  2743 0553  2814 0568
                  3106 058E  3159 05CE  3205 05DC
```