Block I
Apollo Guidance Computer (AGC)

How to build one in your basement

Part 8: Flight Software

John Pultorak
December, 2004

Abstract

This report describes my successful project to build a working reproduction of the 1964 prototype for the Block I Apollo Guidance Computer. The AGC is the flight computer for the Apollo moon landings, and is the world's first integrated circuit computer.

I built it in my basement. It took me 4 years.

If you like, you can build one too. It will take you less time, and yours will be better than mine.

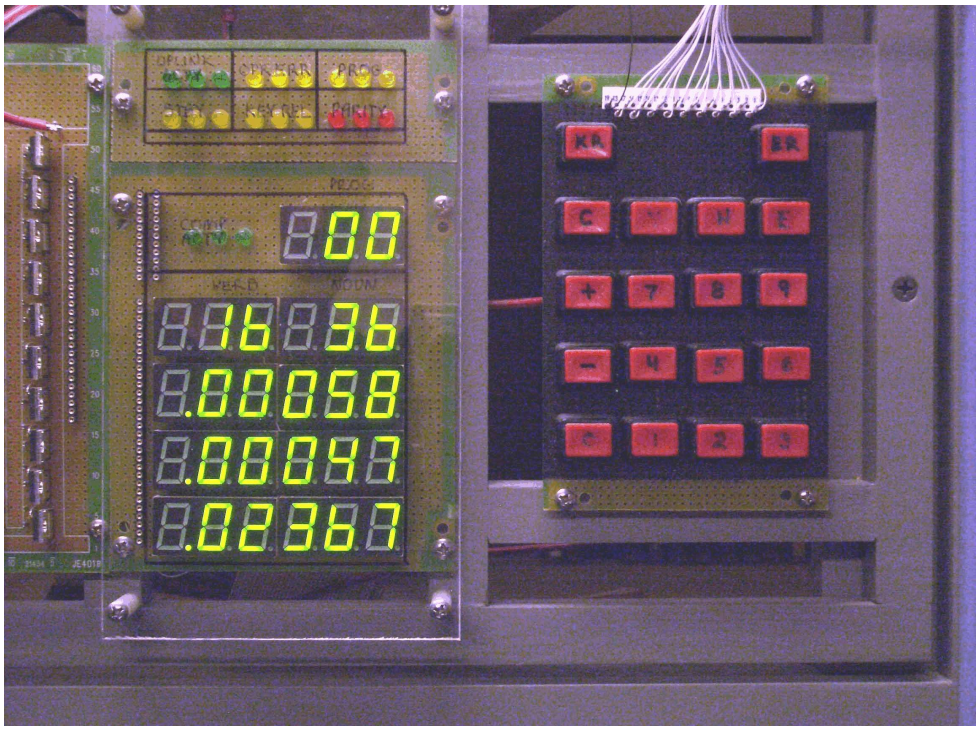
I documented my project in 9 separate .pdf files:

Part 1	Overview: Introduces the project.
Part 2	CTL Module: Design and construction of the control module.
Part 3	PROC Module: Design and construction of the processing (CPU) module.
Part 4	MEM Module: Design and construction of the memory module.
Part 5	IO Module: Design and construction of the display/keyboard (DSKY) module.
Part 6	Assembler: A cross-assembler for AGC software development.
Part 7	C++ Simulator: A low-level simulator that runs assembled AGC code.
Part 8	Flight Software: My translation of portions of the COLOSSUS 249 flight software.
Part 9	Test & Checkout: A suite of test programs in AGC assembly language.

Overview

I wanted genuine Block I flight software for my AGC, but couldn't find any. I eventually wound up recreating Block I software from Block II code listings that were available. Major portions of Block II were originally coded as Block I anyway, so the conversion back was not too difficult. About 95% of the instructions were already Block I, so I just had to translate the remaining 5% into their Block I equivalents.

I downloaded a partial listing of the COLOSSUS 249 flight software for the Block II Apollo Command Module AGC from a M.I.T website in late 2001. The listing (at that time) comprised first half of the flight software load. The second half was missing.



The part that was present contained eraseable memory declarations, and PINBALL, the AGC user interface. The missing portion contained (among other things) the EXEC and WAITLIST pieces of the operating system, bank register calling routines, and math libraries.

The downloaded document was a .pdf file containing 300 or so fuzzy digital images of assembler listing. I printed it, and then retyped the erasable memory and PINBALL portions into a text file, marking off each reentered line in the original listing with a highlighter.

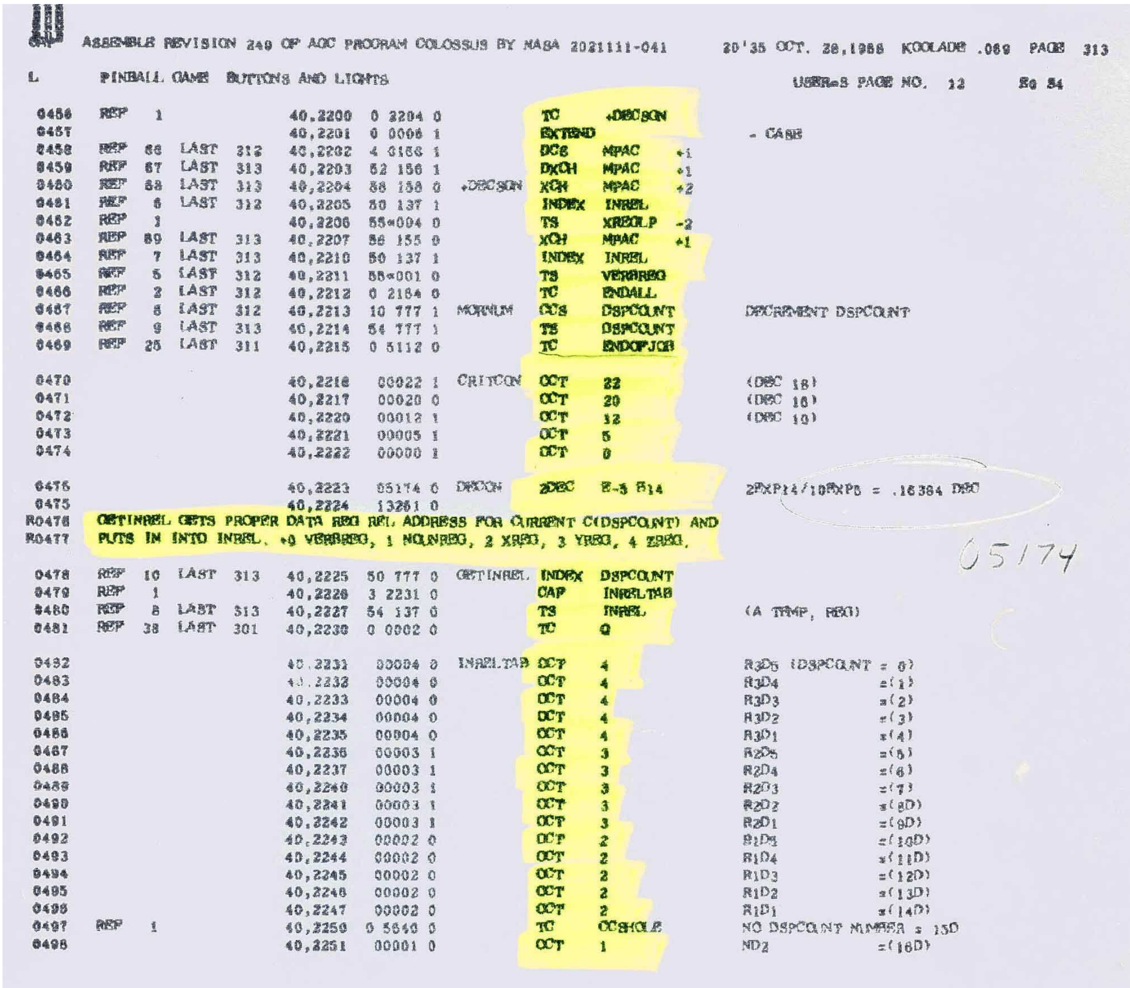
I coded my own versions of the EXEC, WAITLIST, BANKCALL, and other missing routines used by PINBALL. The R-393 document from M.I.T. provided some guidance.

Over a 6 month period, I was able to get all regular verbs and generic normal nouns working. These are listed near the top of my assembler listing.

I added additional comments to portions of code taken from COLOSSUS. The page numbers in my comments refer to pages in the COLOSSUS 249 assembler listing.

Original COLOSSUS 249 Assembler Code

For comparison purposes, here's a page of the original AGC assembly code, downloaded from the M.I.T. website. This is a tiny portion of the PINBALL code. I marked each line with a yellow highlighter as I reentered it into a text file.



```
ASSEMBLE REVISION 249 OF AGC PROGRAM COLOSSUS BY NASA 2021111-041      20'35 OCT. 28,1968  KOOLADE .069  PAGE  313
L        PINBALL GAME  BUTTONS AND LIGHTS                                             USER=S PAGE NO.  12      Eq 54

0456  REP   1                40,2200   0 2204 0              TC       +DBCSGN
0457                         40,2201   0 0006 1              EXTEND                       - CASE
0458  REP  66  LAST  312     40,2202   4 0156 1              DCS      MPAC      +1
0459  REP  67  LAST  313     40,2203  52 156 1               DXCH     MPAC      +1
0460  REP  68  LAST  313     40,2204  56 158 0   +DBCSGN     XCH      MPAC      +2
0461  REP   6  LAST  312     40,2205  50 137 1               INDEX    INREL
0462  REP   1                40,2206  55*004 0               TS       XREGLP    -2
0463  REP  69  LAST  313     40,2207  56 155 0               XCH      MPAC      +1
0464  REP   7  LAST  313     40,2210  50 137 1               INDEX    INREL
0465  REP   5  LAST  312     40,2211  55*001 0               TS       VERBREG
0466  REP   2  LAST  312     40,2212   0 2164 0              TC       ENDALL
0467  REP   8  LAST  312     40,2213  10 777 1   MORNUM      CCS      DSPCOUNT          DECREMENT DSPCOUNT
0468  REP   9  LAST  313     40,2214  54 777 1               TS       DSPCOUNT
0469  REP  25  LAST  311     40,2215   0 5112 0              TC       ENDOFJOB

0470                         40,2216   00022 1   CRITCON     OCT      22                (DEC 18)
0471                         40,2217   00020 0               OCT      20                (DEC 16)
0472                         40,2220   00012 1               OCT      12                (DEC 10)
0473                         40,2221   00005 1               OCT      5
0474                         40,2222   00000 1               OCT      0

0475                         40,2223   05174 0   DECON       2DEC     E-5 B14           2EXP14/10EXP5 = .16384 DEC
0475                         40,2224   13261 0
R0476 GETINREL GETS PROPER DATA REG REL ADDRESS FOR CURRENT C(DSPCOUNT) AND
R0477 PUTS IN INTO INREL. +0 VERBREG, 1 NOUNREG, 2 XREG, 3 YREG, 4 ZREG.              05174

0478  REP  10  LAST  313     40,2225  50 777 0   GETINREL    INDEX    DSPCOUNT
0479  REP   1                40,2226   3 2231 0              CAF      INRELTAB
0480  REP   8  LAST  313     40,2227  54 137 0               TS       INREL             (A TEMP. REG)
0481  REP  38  LAST  301     40,2230   0 0002 0              TC       Q

0482                         40,2231   00004 0   INRELTAB    OCT      4                 R3D5 (DSPCOUNT = 0)
0483                         40,2232   00004 0               OCT      4                 R3D4          =(1)
0484                         40,2233   00004 0               OCT      4                 R3D3          =(2)
0485                         40,2234   00004 0               OCT      4                 R3D2          =(3)
0486                         40,2235   00004 0               OCT      4                 R3D1          =(4)
0487                         40,2236   00003 1               OCT      3                 R2D5          =(5)
0488                         40,2237   00003 1               OCT      3                 R2D4          =(6)
0489                         40,2240   00003 1               OCT      3                 R2D3          =(7)
0490                         40,2241   00003 1               OCT      3                 R2D2          =(8D)
0491                         40,2242   00003 1               OCT      3                 R2D1          =(9D)
0492                         40,2243   00002 0               OCT      2                 R1D5          =(10D)
0493                         40,2244   00002 0               OCT      2                 R1D4          =(11D)
0494                         40,2245   00002 0               OCT      2                 R1D3          =(12D)
0495                         40,2246   00002 0               OCT      2                 R1D2          =(13D)
0496                         40,2247   00002 0               OCT      2                 R1D1          =(14D)
0497  REP   1                40,2250   0 5640 0              TC       CCSHOLE           NO DSPCOUNT NUMBER = 15D
0498                         40,2251   00001 0               OCT      1                 ND2           =(16D)
```

My COLOSSUS Assembler Code

Here's the exact same portion of code, assembled for my Block I AGC. If you compare the two listings, you'll see that I defined some different assembler directives for allocating storage (DS % instead of OCT) and that my code is sitting in a different bank (5 vs. 40) at a different offset than the original Block II code.

```
                                                Copy of agc.lst
12213  5,0213 0  5,6220 1                TC       PDECSGN

12214  5,0214 4  0,0131 0                CS       MPAC+1          ; - case (was DCS, DXCH in Block II)
12215  5,0215 5  0,0131 1                TS       MPAC+1
12216  5,0216 4  0,0132 0                CS       MPAC+2
12217  5,0217 5  0,0132 1                TS       MPAC+2

                          PDECSGN        EQU      *
12220  5,0220 3  0,0132 1                XCH      MPAC+2
12221  5,0221 2  0,0434 0                INDEX    INREL
12222  5,0222 5  0,0473 1                TS       XREGLP-2
12223  5,0223 3  0,0131 1                XCH      MPAC+1
12224  5,0224 2  0,0434 0                INDEX    INREL
12225  5,0225 5  0,0470 1                TS       VERBREG
12226  5,0226 0  5,6174 0                TC       ENDALL

                          MORNUM         EQU      *
12227  5,0227 1  0,0466 1                CCS      DSPCOUNT        ; decrement DSPCOUNT
12230  5,0230 5  0,0466 0                TS       DSPCOUNT
12231  5,0231 0  1,2723 0                TC       ENDOFJOB

                          CRITCON        EQU      *
12232  5,0232     00022 1                DS       %22             ; dec 18
12233  5,0233     00020 0                DS       %20             ; dec 16
12234  5,0234     00012 1                DS       %12             ; dec 10
12235  5,0235     00005 1                DS       %5
12236  5,0236     00000 1                DS       %0

                          DECON          EQU      *
12237  5,0237     05174 0                DS       %05174          ; 2EXP14/10EXP5 = .16384 DEC
12240  5,0240     13261 0                DS       %13261

                          ;-----------------------------------------------------------------------
                          ; GETINREL
                          ; Gets proper data register relative address for current C(DSPCOUNT) and
                          ; puts into INREL: +0 VERBREG, 1 NOUNREG, 2 XREG, 3 YREG, 4 ZREG
                          ;
                          ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                          ; Oct 28, 1968, p.313.
                          ;-----------------------------------------------------------------------

                          GETINREL       EQU      *
12241  5,0241 2  0,0466 1                INDEX    DSPCOUNT
12242  5,0242 3  5,6245 1                CAF      INRELTAB
12243  5,0243 5  0,0434 1                TS       INREL           ; (A TEMP, REG)
12244  5,0244 0  0,0001 0                TC       Q

                          INRELTAB       EQU      *
12245  5,0245     00004 0                DS       %4              ; R3D5, 0 = DSPCOUNT
12246  5,0246     00004 0                DS       %4              ; R3D4, 1
12247  5,0247     00004 0                DS       %4              ; R3D3, 2
12250  5,0250     00004 0                DS       %4              ; R3D2, 3
12251  5,0251     00004 0                DS       %4              ; R3D1, 4
12252  5,0252     00003 1                DS       %3              ; R2D5, 5
12253  5,0253     00003 1                DS       %3              ; R2D4, 6
12254  5,0254     00003 1                DS       %3              ; R2D3, 7
12255  5,0255     00003 1                DS       %3              ; R2D2, 8D
12256  5,0256     00003 1                DS       %3              ; R2D1, 9D
12257  5,0257     00002 0                DS       %2              ; R1D5, 10D
12260  5,0260     00002 0                DS       %2              ; R1D4, 11D
12261  5,0261     00002 0                DS       %2              ; R1D3, 12D
12262  5,0262     00002 0                DS       %2              ; R1D2, 13D
12263  5,0263     00002 0                DS       %2              ; R1D1, 14D
12264  5,0264 0  5,6271 0                TC       CCSHOLE         ; no DSPCOUNT numbers
12265  5,0265     00001 0                DS       %1              ; ND2, 16D
```

Scenarios

Here's how some AGC verbs and nouns are used to do commonplace operations. The overview (part 1) and simulator (part 7) documents contain actual examples of some of these operations in the simulated and hardware AGCs.

Display elapsed time from the AGC clock:

<VERB> <0> <6> <NOUN> <3> <6> <ENTER>

Start a monitor program to continuously display the AGC clock:

<VERB> <1> <6> <NOUN> <3> <6> <ENTER>

Terminate a monitor program:

<VERB> <3> <4> <ENTER>

Test DSJT display lights:

<VERB> <3> <5> <ENTER>
All DSKY lamps and display segments illuminate for 5 sec. after 5 sec, the DSKY lamps extinguish.

Load component 1 for dataset at octal address 50 with octal 123:

<VERB> <2> <1> <NOUN> <0> <1> <ENTER>
Verb/noun display flashes: waiting for address.
<5> <0> <ENTER>
Verb/noun display flash continues: waiting for data.
<1> <2> <3> <ENTER>
Octal word from R1 is loaded at address 50.

Display component 1 of dataset at octal address 50:

<VERB> <0> <1> <NOUN> <0> <1> <ENTER>
Verb/noun display flashes: waiting for address.
<5> <0> <ENTER>
Octal word from address 50 is displayed in R1.

Display component 1 of dataset incrementing from 50:

<VERB> <0> <1> <NOUN> <0> <1> <ENTER>
Verb/noun display flashes: waiting for address.
<5> <0> <ENTER>
Octal word from address 50 is displayed in R1.
<NOUN> <1> <5> <ENTER>
Octal word from address 51 is displayed in R1, address in R3.
<ENTER>
Octal word from address 52 is displayed in R1, address in R3.

Load 3 component dataset at octal address 50 with octal values: 123, 456, 701:

<VERB> <2> <5> <NOUN> <0> <1> <ENTER>
Verb/noun display flashes: waiting for address.
<5> <0> <ENTER>
Verb/noun display flash continues: waiting for data.
<1> <2> <3> <ENTER>
<4> <5> <6> <ENTER>
<7> <0> <1> <ENTER>

Octal word from R1 is loaded at address 50; Octal word from R2 is loaded at address 51, Octal word from R3 is loaded at address 52.

Display 3 component dataset beginning at address 50:

<VERB> <0> <5> <NOUN> <0> <1> <ENTER>

Verb/noun display flashes: waiting for address.

<5> <0> <ENTER>

Octal word from address 50 is displayed in R1; Octal word from address 51 is displayed in R2; Octal word from address 52 is displayed in R3.

Change major mode to P00:

<VERB> <3> <7> <ENTER>

Verb/noun display flashes: waiting for major mode

<0> <0> <ENTER>

VERBS and NOUNS

COLOSSUS REGULAR VERBS (00-39 decimal)

This is adapted from the Apollo 204 accident report posted on multiple web sites by Richard F. Drushel. The information has been changed as necessary to be consistent with usage in COLOSSUS 249.

Verb Code	Description	Remarks
01	Display octal comp 1 in R1	Performs octal display of data on REGISTER 1.
02	Display octal comp 2 in R2	Performs octal display of data on REGISTER 1.
03	Display octal comp 3 in R3	Performs octal display of data on REGISTER 1.
04	Display octal comp 1,2 in R1,R2	Performs octal display of data on REGISTER 1 and REGISTER 2
05	Display octal comp 1,2,3 in R1,R2,R3	Performs octal display of data on REGISTER 1, REGISTER 2, and REGISTER 3.
06	Display decimal in R1 or R1,R2 or R1,R2,R3	Performs decimal display of data on appropriate registers. The scale factors, types of scale factor routines, and component information are stored within the machine for each noun which it is required to display in decimal.
07	Display DP decimal in R1,R2	Performs a double precision decimal display of data on REGISTER 1 and REGISTER 2. It does no scale factoring. It merely performs a 10-character, fractional decimal conversion of two consecutive, erasable registers, using REGISTER 1 and REGISTER 2. The sign is placed in the REGISTER 1 sign position with the REGISTER 2 sign position remaining blank. It cannot be used with mixed nouns. Its intended use is primarily with "machine address to be specified" nouns.
08	(Spare)	
09	(Spare)	
10	(Spare)	
11	Monitor octal comp 1 in R1	Performs octal display of updated data every 1/2 second on REGISTER 1.
12	Monitor octal comp 2 in R2	Performs octal display of updated data every 1/2 second on REGISTER 1.
13	Monitor octal comp 3 in R3	Performs octal display of updated data every 1/2 second on REGISTER 1.
14	Monitor octal comp 1,2 in R1,R2	Performs octal display of updated data every 1/2 second on REGISTER 1 and REGISTER 2.
15	Monitor octal comp 1,2,3 in R1,R2,R3	Performs octal display of updated data every 1/2 second on REGISTER 1, REGISTER 2, and REGISTER 3.

16	Monitor decimal in R1 or R1,R2, or R1,R2,R3	Performs decimal display of updated data every 1/2 second on appropriate registers.
17	Monitor DP decimal in R1,R2	Performs double precision display of decimal data on REGISTER 1 and REGISTER 2. No scale factoring is performed. Provides 10-character, fractional decimal conversion of two consecutive erasable registers. The sign is placed in the sign-bit position of REGISTER 1. REGISTER 2 sign bit is blank.
18	(Spare)	
19	(Spare)	
20	(Spare)	
21	Load component 1 into R1	Performs data loading. Octal quantities are unsigned. Decimal quantities are preceded by + or - sign. Data is displayed on REGISTER 1.
22	Load component 2 into R2	Performs data loading. Octal quantities are unsigned. Decimal quantities are preceded by + or - sign. Data is displayed on REGISTER 2.
23	Load component 3 into R3	Performs data loading. Octal quantities are unsigned. Decimal quantities are preceded by + or - sign. Data is displayed on REGISTER 3.
24	Load component 1,2 into R1,R2	Performs data loading. Octal quantities are unsigned. Decimal quantities are preceded by + or - sign. Data is displayed on REGISTER 1 and REGISTER 2.
25	Load component 1,2,3 into R1,R2,R3	Performs data loading. Octal quantities are unsigned. Decimal quantities are preceded by + or - sign. Data is displayed on REGISTER 1, REGISTER 2, and REGISTER 3.
26	(Spare)	
27	Display fixed memory	This verb is included to permit displaying the contents of fixed memory in any bank. Its intended use is for checking program ropes and the BANK positions of program ropes.
28	(Spare)	
29	(Spare)	
30	Request EXECUTIVE (Used only during ground checkout.)	Enters request to executive routine for any machine address with priority involved. This verb assumes that the desired priority has been loaded into bits 10-14 of the prio/delay register (noun 26). This verb is used with the noun, "machine address to be specified". The complete address of the desired location is then keyed in. (Refer to "Machine address to be specified" in paragraph on Verb/Noun Formats.)
31	Request WAITLIST	Enters request to "waitlist routine"

	(Used only during ground checkout.)	for any machine address with delay involved. This verb assumes that the desired number of 10-millisecond units of delay has been loaded into the low order bits of the prio/delay register (noun 26). This verb is used with the "machine address to be specified" noun. The complete address of the desired location is then keyed in. (Refer to "Machine address to be specified" in paragraph on Verb/Noun Formats.)
32	Recycle	
33	Proceed (without data)	Informs routine requesting data that the operator chooses not to load fresh data, but wishes the routine to continue as best it can with old data. Final decision for what action should be taken is left to the requesting routine.
34	Terminate	Informs routine requesting data to be loaded that the operator chooses not to load fresh data and wishes the routine to terminate. Final decision for what action should be taken is left to the requesting routine. If monitor is on, it is turned off.
35	Test lights	
36	Request fresh start	Initializes the program control software and the keyboard and display system program.
37	Change program (major mode)	Change to new major mode. (Refer to "Change major mode" in paragraph on Verb/Noun Formats.)

COLOSSUS EXTENDED VERBS (40-99 decimal)

Not implemented. Use of these verbs triggers the 'check fail' indicator.

COLOSSUS NORMAL NOUNS (00-39 decimal)

This is adapted from the Apollo 204 accident report posted on multiple web sites by Richard F. Drushel. The information has been changed as necessary to be consistent with usage in COLOSSUS 249.

Noun Code	Description	Scale/Units
01	Specify machine address (frac)	.XXXXX FRAC .XXXXX FRAC .XXXXX FRAC
02	Specify machine address (whole)	XXXXX INTEGER XXXXX INTEGER XXXXX INTEGER
03	Specify machine address (degree)	XXX.XX DEG XXX.XX DEG XXX.XX DEG

04	(Spare)	
05	(Spare)	
06	(Spare)	
07	(Spare)	
08	(Spare)	
09	Alarm codes	OCT OCT OCT
10	(Spare)	
11	(Spare)	
12	(Spare)	
13	(Spare)	
14	(Spare)	
15	Increment address	OCT
16	(Spare)	
17	(Spare)	
18	(Spare)	
19	(Spare)	
20	(Spare)	
21	(Spare)	
22	(Spare)	
23	(Spare)	
24	(Spare)	
25	(Spare)	
26	Prio/delay, address	OCT (prio/delay) OCT (14-bit CADR) (not used)
27	(Spare)	
28	(Spare)	
29	(Spare)	
30	(Spare)	
31	(Spare)	
32	(Spare)	
33	(Spare)	
34	(Spare)	
35	(Spare)	
36	Time of CMC clock: REGISTER 1 REGISTER 2 REGISTER 3	 00XXX. hours 000XX. minutes 0XX.XX seconds
37	(Spare)	

```
    |                                                |
38  |  (Spare)                                       |
    |                                                |
39  |  (Spare)                                       |
    |                                                |
;-------------------------------------------------------------------------
```

COLOSSUS MIXED NOUNS (40-99 decimal)

Not implemented.

Flight software assembler listing

```
Block I Apollo Guidance Computer (AGC4) assembler version 1.6 for EPROM

First pass: generate symbol table.
Second pass: generate object code.

                        ;=============================================================================
                        ; AGC (file:agc.asm)
                        ;
                        ; Version:  1.0
                        ; Author:   John Pultorak
                        ; Date:     6/7/2002
                        ;
                        ; PURPOSE:
                        ; AGC Block I demonstration. Includes most of the AGC operating system:
                        ; WAITLIST, EXEC, PINBALL (DSKY routines), NOUN tables, VERB tables,
                        ; bank intercommunication routines, the KEY, T3, and T4 interrupt handlers,
                        ; and some dual precision (DP) math routines.
                        ;
                        ; The interpreter is not currently implemented.
                        ;
                        ; Where available, the source is from the Apollo 8 command module computer (CMC)
                        ; load (called COLOSSUS). In cases where COLOSSUS source is not available,
                        ; functionally equivalent code was constructed using COLOSSUS calling and return
                        ; parameters and according to specifications in the technical reports given below.
                        ;
                        ; OPERATION:
                        ; TBD.
                        ;
                        ; ERRATA:
                        ; - Adapted for the AGC4R assembler. The assembler directives and syntax
                        ; differ somewhat from the original AGC assembler.
                        ; - some of the original source was missing from the COLOSSUS listing and
                        ; had to be reverse engineered. Those portions probably differ somewhat
                        ; from the original code in implementation, but should be functionally
                        ; identical.
                        ; - because the COLOSSUS source is for a block II AGC, but the AGC
                        ; implemented here is block I, about 5% of COLOSSUS had to be translated
                        ; to equivalent block I code.
                        ;
                        ; SOURCES:
                        ; Information on the Block I architecture: instruction set, instruction
                        ; sequences, registers, register transfers, control pulses, memory and
                        ; memory addressing, I/O assignments, interrupts, and involuntary counters
                        ; was obtained from:
                        ;
                        ;       A. Hopkins, R. Alonso, and H. Blair-Smith, "Logical Description
                        ;               for the Apollo Guidance Computer (AGC4)", R-393,
                        ;               MIT Instrumentation Laboratory, Cambridge, MA, Mar. 1963.
                        ;
                        ; Supplementary AGC hardware information was obtained from:
                        ;
                        ;       R. Alonso, J. H. Laning, Jr. and H. Blair-Smith, "Preliminary
                        ;               MOD 3C Programmer's Manual", E-1077, MIT Instrumentation
                        ;               Laboratory, Cambridge, MA, Nov. 1961.
                        ;
                        ;       B. I. Savage and A. Drake, "AGC4 Basic Training Manual, Volume I",
                        ;               E-2052, MIT Instrumentation Laboratory, Cambridge,
                        ;               MA, Jan. 1967.
                        ;
                        ;       E. C. Hall, "MIT's Role in Project Apollo, Volume III, Computer
                        ;               Subsystem", R-700, MIT Charles Stark Draper Laboratory,
                        ;               Cambridge, MA, Aug. 1972.
                        ;
                        ;       A. Hopkins, "Guidance Computer Design, Part VI", source unknown.
                        ;
                        ;       E, C. Hall, "Journey to the Moon: The History of the Apollo
                        ;               Guidance Computer", AIAA, Reston VA, 1996.
                        ;
                        ; AGC software information was obtained from:
                        ;
                        ;       AGC Block II COLOSSUS rev 249 assembly listing, Oct 28, 1968. (A
                        ;               listing of the 1st 50% of the build. It encludes the entire
                        ;               eraseable memory, restart initialization, T4RUPT, and the
                        ;               entire set of DSKY routines. About 5% of instructions
                        ;               had to be converted from Block II to Block I).
                        ;
                        ;       A. I. Green and J. J. Rocchio, "Keyboard and Display System Program
                        ;               for AGC (Program Sunrise)", E-1574, MIT Instrumentation
                        ;               Laboratory, Cambridge, MA, Aug. 1964. Contains detailed
                        ;               flowcharts and design materials for the DSKY software.
                        ;
                        ;       A. Hopkins, R. Alonso, and H. Blair-Smith, "Logical Description
                        ;               for the Apollo Guidance Computer (AGC4)", R-393,
```

```
;               MIT Instrumentation Laboratory, Cambridge, MA, Mar. 1963.
;               Contains the software interfaces for EXEC and WAITLIST, and
;               portions of the dual precision (DP) math library.
;
;==============================================================================

              INCL     doc.asm
;==============================================================================
; AGC documentation (file:doc.asm)
;
; Version:  1.0
; Author:   John Pultorak
; Date:     06/01/2002
;
; PURPOSE:
; Documents AGC ops source code.
;==============================================================================


;------------------------------------------------------------------------------
; DSKY OPERATION (examples)
;
;       verb/noun (V/N) flash: When the verb and noun indicators flash
;               at 1Hz, the DSKY is waiting for keyboard input.
;
;
; Display elapsed time from the AGC clock:
;       <VERB> <0> <6> <NOUN> <3> <6> <ENTER>
;
; Test display lights
;       a) <VERB> <3> <5> <ENTER>
;       b) all DSKY lamps and display segments illuminate for 5 sec.
;       c) after 5 sec, the DSKY lamps extinguish
;
; Load component 1 for dataset at octal address 50 with octal 123
;       a) <VERB> <2> <1> <NOUN> <0> <1> <ENTER>
;       b) verb/noun display flashes; waiting for address
;       c) <5> <0> <ENTER>
;       d) verb/noun display flash continues; waiting for data
;       e) <1> <2> <3> <ENTER>
;       f) octal word from R1 is loaded at address 50,
;
; Display component 1 of dataset at octal address 50:
;       a) <VERB> <0> <1> <NOUN> <0> <1> <ENTER>
;       b) verb/noun display flashes; waiting for address
;       c) <5> <0> <ENTER>
;       d) octal word from address 50 is displayed in R1
;
; Load 3 component dataset at octal address 50 with octal values
;   123,456,701
;       a) <VERB> <2> <5> <NOUN> <0> <1> <ENTER>
;       b) verb/noun display flashes; waiting for address
;       c) <5> <0> <ENTER>
;       d) verb/noun display flash continues; waiting for data
;       e) <1> <2> <3> <ENTER>
;       f) <4> <5> <6> <ENTER>
;       g) <7> <0> <1> <ENTER>
;       h) octal word from R1 is loaded at address 50,
;       octal word from R2 is loaded at address 51,
;       octal word from R3 is loaded at address 52
;
; Display 3 component dataset beginning at address 50:
;       a) <VERB> <0> <5> <NOUN> <0> <1> <ENTER>
;       b) verb/noun display flashes; waiting for address
;       c) <5> <0> <ENTER>
;       d) octal word from address 50 is displayed in R1,
;       octal word from address 51 is displayed in R2,
;       octal word from address 52 is displayed in R3
;
;------------------------------------------------------------------------------


;------------------------------------------------------------------------------
; COLOSSUS REGULAR VERBS (00-39 decimal)
;
; This is adapted from the Apollo 204 accident report posted on multiple
; web sites by Richard F. Drushel. The information has been changed as
; necessary to be consistent with usage in COLOSSUS.
;
;
; Verb |                              |
; Code |          Description         |               Remarks
;      |                              |
; 01   |  Display octal comp 1 in R1  |  Performs octal display of data on
;      |                              |  REGISTER 1.
;      |                              |
; 02   |  Display octal comp 2 in R2  |  Performs octal display of data on
```

```
;      |                              |  REGISTER 1.
;      |                              |
; 03   |  Display octal comp 3 in R3  |  Performs octal display of data on
;      |                              |  REGISTER 1.
;      |                              |
; 04   |  Display octal comp 1,2      |  Performs octal display of data on
;      |  in R1,R2                    |  REGISTER 1 and REGISTER 2
;      |                              |
; 05   |  Display octal comp 1,2,3    |  Performs octal display of data on
;      |  in R1,R2,R3                 |  REGISTER 1, REGISTER 2, and REGISTER 3.
;      |                              |
; 06   |  Display decimal in R1 or    |  Performs decimal display of data on
;      |  R1,R2 or R1,R2,R3           |  appropriate registers.  The scale
;      |                              |  factors, types of scale factor
;      |                              |  routines, and component information
;      |                              |  are stored within the machine for each
;      |                              |  noun which it is required to display
;      |                              |  in decimal.
;      |                              |
; 07   |  Display DP decimal in R1,R2 |  Performs a double precision decimal
;      |                              |  display of data on REGISTER 1 and
;      |                              |  REGISTER 2.  It does no scale
;      |                              |  factoring.  It merely performs a 10-
;      |                              |  character, fractional decimal
;      |                              |  conversion of two consecutive, erasable
;      |                              |  registers, using REGISTER 1 and
;      |                              |  REGISTER 2.  The sign is placed in the
;      |                              |  REGISTER 1 sign position with the
;      |                              |  REGISTER 2 sign position remaining
;      |                              |  blank.  It cannot be used with mixed
;      |                              |  nouns.  Its intended use is primarily
;      |                              |  with "machine address to be specified"
;      |                              |  nouns.
;      |                              |
; 08   |  (Spare)                     |
;      |                              |
; 09   |  (Spare)                     |
;      |                              |
; 10   |  (Spare)                     |
;      |                              |
; 11   |  Monitor octal comp 1 in R1  |  Performs octal display of updated data
;      |                              |  every 1/2 second on REGISTER 1.
;      |                              |
; 12   |  Monitor octal comp 2 in R2  |  Performs octal display of updated data
;      |                              |  every 1/2 second on REGISTER 1.
;      |                              |
; 13   |  Monitor octal comp 3 in R3  |  Performs octal display of updated data
;      |                              |  every 1/2 second on REGISTER 1.
;      |                              |
; 14   |  Monitor octal comp 1,2      |  Performs octal display of updated data
;      |  in R1,R2                    |  every 1/2 second on REGISTER 1 and
;      |                              |  REGISTER 2.
;      |                              |
; 15   |  Monitor octal comp 1,2,3    |  Performs octal display of updated data
;      |  in R1,R2,R3                 |  every 1/2 second on REGISTER 1,
;      |                              |  REGISTER 2, and REGISTER 3.
;      |                              |
; 16   |  Monitor decimal in R1 or    |  Performs decimal display of updated
;      |  R1,R2, or R1,R2,R3          |  data every 1/2 second on appropriate
;      |                              |  registers.
;      |                              |
; 17   |  Monitor DP decimal in R1,R2 |  Performs double precision display of
;      |                              |  decimal data on REGISTER 1 and
;      |                              |  REGISTER 2.  No scale factoring is
;      |                              |  performed.  Provides 10-character,
;      |                              |  fractional decimal conversion of two
;      |                              |  consecutive erasable registers.  The
;      |                              |  sign is placed in the sign-bit
;      |                              |  position of REGISTER 1.  REGISTER 2
;      |                              |  sign bit is blank.
;      |                              |
; 18   |  (Spare)                     |
;      |                              |
; 19   |  (Spare)                     |
;      |                              |
; 20   |  (Spare)                     |
;      |                              |
; 21   |  Load component 1 into R1    |  Performs data loading.  Octal
;      |                              |  quantities are unsigned.  Decimal
;      |                              |  quantities are preceded by + or -
;      |                              |  sign.  Data is displayed on REGISTER
;      |                              |  1.
;      |                              |
; 22   |  Load component 2 into R2    |  Performs data loading.  Octal
;      |                              |  quantities are unsigned.  Decimal
;      |                              |  quantities are preceded by + or -
;      |                              |  sign.  Data is displayed on REGISTER
```

```
;      |                                |  2.
;      |                                |
; 23   |  Load component 3 into R3      |  Performs data loading.  Octal
;      |                                |  quantities are unsigned.  Decimal
;      |                                |  quantities are preceded by + or -
;      |                                |  sign.  Data is displayed on REGISTER
;      |                                |  3.
;      |                                |
; 24   |  Load component 1,2 into       |  Performs data loading.  Octal
;      |  R1,R2                         |  quantities are unsigned.  Decimal
;      |                                |  quantities are preceded by + or -
;      |                                |  sign.  Data is displayed on REGISTER
;      |                                |  1 and REGISTER 2.
;      |                                |
; 25   |  Load component 1,2,3 into     |  Performs data loading.  Octal
;      |  R1,R2,R3                      |  quantities are unsigned.  Decimal
;      |                                |  quantities are preceded by + or -
;      |                                |  sign.  Data is displayed on REGISTER
;      |                                |  1, REGISTER 2, and REGISTER 3.
;      |                                |
; 26   |  (Spare)                       |
;      |                                |
; 27   |  Display fixed memory          |  This verb is included to permit
;      |                                |  displaying the contents of fixed
;      |                                |  memory in any bank.  Its intended use
;      |                                |  is for checking program ropes and the
;      |                                |  BANK positions of program ropes.
;      |                                |
; 28   |  (Spare)                       |
;      |                                |
; 29   |  (Spare)                       |
;      |                                |
; 30   |  Request EXECUTIVE             |  Enters request to executive routine
;      |  (Used only during ground      |  for any machine address with priority
;      |  checkout.)                    |  involved.  This verb assumes that the
;      |                                |  desired priority has been loaded into
;      |                                |  bits 10-14 of the prio/delay register
;      |                                |  (noun 26).  This verb is used with the
;      |                                |  noun, "machine address to be
;      |                                |  specified".  The complete address of
;      |                                |  the desired location is then keyed in.
;      |                                |  (Refer to "Machine address to be
;      |                                |  specified" in paragraph on Verb/Noun
;      |                                |  Formats.)
;      |                                |
; 31   |  Request WAITLIST              |  Enters request to "waitlist routine"
;      |  (Used only during ground      |  for any machine address with delay
;      |  checkout.)                    |  involved.  This verb assumes that the
;      |                                |  desired number of 10-millisecond units
;      |                                |  of delay has been loaded into the low
;      |                                |  order bits of the prio/delay register
;      |                                |  (noun 26).  This verb is used with the
;      |                                |  "machine address to be specified" noun.
;      |                                |  The complete address of the desired
;      |                                |  location is then keyed in.  (Refer to
;      |                                |  "Machine address to be specified" in
;      |                                |  paragraph on Verb/Noun Formats.)
;      |                                |
; 32   |  Recycle                       |
;      |                                |
; 33   |  Proceed (without data)        |  Informs routine requesting data that
;      |                                |  the operator chooses not to load
;      |                                |  fresh data, but wishes the routine to
;      |                                |  continue as best it can with old data.
;      |                                |  Final decision for what action should
;      |                                |  be taken is left to the requesting
;      |                                |  routine.
;      |                                |
; 34   |  Terminate                     |  Informs routine requesting data to be
;      |                                |  loaded that the operator chooses not
;      |                                |  to load fresh data and wishes the
;      |                                |  routine to terminate.  Final decision
;      |                                |  for what action should be taken is
;      |                                |  left to the requesting routine.  If
;      |                                |  monitor is on, it is turned off.
;      |                                |
; 35   |  Test lights                   |
;      |                                |
; 36   |  Request fresh start           |  Initializes the program control
;      |                                |  software and the keyboard and display
;      |                                |  system program.
;      |                                |
; 37   |  Change program (major mode)   |  Change to new major mode.  (Refer to
;      |                                |  "Change major mode" in paragraph on
;      |                                |  Verb/Noun Formats.)
;      |                                |
;-------------------------------------------------------------------------------
```

```
;------------------------------------------------------------------------
; COLOSSUS EXTENDED VERBS (40-99 decimal)
;
; Not implemented. Use of these verbs triggers the 'check fail' indicator.
;------------------------------------------------------------------------


;------------------------------------------------------------------------
; COLOSSUS NORMAL NOUNS (00-39 decimal)
;
; This is adapted from the Apollo 204 accident report posted on multiple
; web sites by Richard F. Drushel. The information has been changed as
; necessary to be consistent with usage in COLOSSUS.
;
;
; Noun |                                                |
; Code |          Description                           |  Scale/Units
;      |                                                |
; 01   |  Specify machine address (frac)                |  .XXXXX FRAC
;      |                                                |  .XXXXX FRAC
;      |                                                |  .XXXXX FRAC
;      |                                                |
; 02   |  Specify machine address (whole)               |  XXXXX INTEGER
;      |                                                |  XXXXX INTEGER
;      |                                                |  XXXXX INTEGER
;      |                                                |
; 03   |  Specify machine address (degree)              |  XXX.XX DEG
;      |                                                |  XXX.XX DEG
;      |                                                |  XXX.XX DEG
;      |                                                |
; 04   |  (Spare)                                       |
;      |                                                |
; 05   |  (Spare)                                       |
;      |                                                |
; 06   |  (Spare)                                       |
;      |                                                |
; 07   |  (Spare)                                       |
;      |                                                |
; 08   |  (Spare)                                       |
;      |                                                |
; 09   |  Alarm codes                                   |  OCT
;      |                                                |  OCT
;      |                                                |  OCT
;      |                                                |
; 10   |  (Spare)                                       |
;      |                                                |
; 11   |  (Spare)                                       |
;      |                                                |
; 12   |  (Spare)                                       |
;      |                                                |
; 13   |  (Spare)                                       |
;      |                                                |
; 14   |  (Spare)                                       |
;      |                                                |
; 15   |  Increment address                             |  OCT
;      |                                                |
;      |                                                |
; 16   |  (Spare)                                       |
;      |                                                |
; 17   |  (Spare)                                       |
;      |                                                |
; 18   |  (Spare)                                       |
;      |                                                |
; 19   |  (Spare)                                       |
;      |                                                |
; 20   |  (Spare)                                       |
;      |                                                |
; 21   |  (Spare)                                       |
;      |                                                |
; 22   |  (Spare)                                       |
;      |                                                |
; 23   |  (Spare)                                       |
;      |                                                |
; 24   |  (Spare)                                       |
;      |                                                |
; 25   |  (Spare)                                       |
;      |                                                |
; 26   |  Prio/delay, address                           |  OCT (prio/delay)
;      |                                                |  OCT (14-bit CADR)
;      |                                                |  (not used)
;      |                                                |
; 27   |  (Spare)                                       |
;      |                                                |
```

```
; 28   |  (Spare)                                          |
;      |                                                   |
; 29   |  (Spare)                                          |
;      |                                                   |
; 30   |  (Spare)                                          |
;      |                                                   |
; 31   |  (Spare)                                          |
;      |                                                   |
; 32   |  (Spare)                                          |
;      |                                                   |
; 33   |  (Spare)                                          |
;      |                                                   |
; 34   |  (Spare)                                          |
;      |                                                   |
; 35   |  (Spare)                                          |
;      |                                                   |
; 36   |  Time of CMC clock:                               |
;      |    REGISTER 1                                     |  00XXX. hours
;      |    REGISTER 2                                     |  000XX. minutes
;      |    REGISTER 3                                     |  0XX.XX seconds
;      |                                                   |
; 37   |  (Spare)                                          |
;      |                                                   |
; 38   |  (Spare)                                          |
;      |                                                   |
; 39   |  (Spare)                                          |
;      |                                                   |
;-------------------------------------------------------------------------------


;-------------------------------------------------------------------------------
; COLOSSUS MIXED NOUNS (40-99 decimal)
;
; Not implemented.
;-------------------------------------------------------------------------------


;-------------------------------------------------------------------------------
; AGC ADDRESS ASSIGNMENTS
;
; Central Registers
;
;       000000          A               accumulator
;       000001          Q               subroutine return address
;       000002          Z               program counter
;       000003          LP              lower product register
;
; Input Registers
;
;       000004          IN0
;       000005          IN1
;       000006          IN2
;       000007          IN3
;
; Output Registers
;
;       000010          OUT0
;       000011          OUT1
;       000012          OUT2
;       000013          OUT3
;       000014          OUT4
;
; Memory Bank Select
;
;       000015          BANK
;
; Interrupt Control
;
;       000016          RELINT          re-enable interrupts
;       000017          INHINT          inhibit interrupts
;
; Editing Registers
;
;       000020          CYR             cycle right
;       000021          SR              shift rRight
;       000022          CYL             cycle left
;       000023          SL              shift left
;
; Interrupt Storage Area
;
;       000024          ZRUPT           save program counter (Z)
;       000025          BRUPT           save B register
;       000026          ARUPT           save accumulator (A)
;       000027          QRUPT           save Q register
;
```

```
;       000030 - 000033  NOT USED
;
; Involuntary Counters
;
;       000034           OVCTR           arithmetic overflow counter
;       000035           TIME2           AGC clock (high)
;       000036           TIME1           AGC clock (low)
;       000037           TIME3           WAITLIST (T3) timer
;       000040           TIME4           DISPLAY (T4) timer
;
; Involuntary Counters -- currently unused
;
;       000041 - 000056  NOT USED
;
; Eraseable Memory
;
;       000057 - 001777
;
; Start of fixed memory
;
;       002000           GOPROG          AGC (re)start vector
;
;       002004           T3RUPT          interrupt vector for TIME3 (T3RUPT)
;       020010           ERRUPT          interrupt vector
;       020014           DSRUPT          interrupt vector for DSRUPT (T4RUPT)
;       020020           KEYRUPT         interrupt vector for keyboard
;       020024           UPRUPT          interrupt vector for uplink
;-------------------------------------------------------------------------


;-------------------------------------------------------------------------
; AGC TABLES (name, file, description)
;
; Keyboard/display
;       CHARIN2          bank40_1.asm    keyboard character table
;       INRELTAB         bank40_1.asm    DSKY register/display table map
;       DSPTAB           dsky_e.asm      display table for DSKY
;
; Verbs:
;       VERBTAB          bank41_1.asm    regular verb routines (00-39)
;
; Nouns:
;       NNADTAB          bank42_3.asm    noun address table (00-99)
;       NNTYPTAB         bank42_3.asm    noun type table (00-99)
;       SFINTAB          bank42_3.asm    noun input scale factor select
;       SFOUTAB          bank42_3.asm    nout output scale factor select
;       IDADDTAB         bank42_3.asm    mixed noun address table (40-99)
;       RUTMXTAB         bank42_3.asm    mixed noun scale factor routine (40-99)
;
; Noun scale factor routines:
;       SFOUTABR         bank41_1.asm    scale factor output routines
;       SFINTABR         bank41_2.asm    scale factor input routines
;
; Major Modes:
;       FCADRMM          bank04_1.asm    entry points for MM jobs
;       EPREMM1          bank04_1,asm    priorities for MM jobs
;-------------------------------------------------------------------------


; ERASEABLE MEMORY DECLARATIONS

              ORG      BANK0          ; immediately following counters
              INCL     waitlist_e.asm ; WAITLIST variables
;=========================================================================
; WAITLIST (file:waitlist_e.asm)
;
; Version:  1.0
; Author:   John Pultorak
; Date:     11/15/2001
;
; PURPOSE:
; Eraseable memory variables and structures for the WAITLIST. See the
; WAITLIST source code file for more information.
;=========================================================================


MAXTASK        EQU      7              ; max number of tasks
MAXVAL         EQU      %037777        ; largest pos 15-bit int (+16383 dec)
MAXDELAY       EQU      12000          ; 120 seconds (in .01 sec ticks)
MAXTIMEOUT     EQU      MAXVAL-MAXDELAY+1 ; TIME3 setting for MAXDELAY


; task delta t: number of 10 mSec ticks until timeout.
;   i.e.: 0=timeout, 1=10mS until timeout, 2=20mS until timeout...
;   maximum time delay is 120 (decimal) seconds.
;
```

```
                        ; If a task record is empty (unused), the address is always set to
                        ; zero and the time is set to MAXDELAY.

                        ; task record structure
                        TSKTIME        EQU      0              ; offset to task delta time
                        TSKADDR        EQU      1              ; offset to 14-bit task address

                        TRECSZ         EQU      2              ; size of task record (words)

                        ; Array of all task records
                        WL_taskList    EQU      *
00057   0057    00000 1                DS       0              ; record 0
00060   0060    00000 1                DS       0

00061   0061    00000 1                DS       0              ; record 1
00062   0062    00000 1                DS       0

00063   0063    00000 1                DS       0              ; record 2
00064   0064    00000 1                DS       0

00065   0065    00000 1                DS       0              ; record 3
00066   0066    00000 1                DS       0

00067   0067    00000 1                DS       0              ; record 4
00070   0070    00000 1                DS       0

00071   0071    00000 1                DS       0              ; record 5
00072   0072    00000 1                DS       0

00073   0073    00000 1                DS       0              ; record 6
00074   0074    00000 1                DS       0


00075   0075    00000 1 WL_IN_saveQ    DS       0              ; return address
00076   0076    00000 1 WL_IN_taskPtr  DS       0              ; points to task rec in list
00077   0077    00000 1 WL_IN_loopCnt  DS       0              ; loop counter

00100   0100    00000 1 WL_AT_saveQ    DS       0              ; return address
00101   0101    00000 1 WL_AT_taskPtr  DS       0              ; points to task rec in list
00102   0102    00000 1 WL_AT_newTime  DS       0              ; time to be inserted
00103   0103    00000 1 WL_AT_timeLeft DS       0              ; time remaining until timeout
00104   0104    00000 1 WL_AT_loopCnt  DS       0              ; loop counter

00105   0105    00000 1 WL_T3_saveQ    DS       0              ; return address
00106   0106    00000 1 WL_T3_oldBank  DS       0              ; current bank

00107   0107    00000 1 WL_ST_saveQ    DS       0              ; return address
00110   0110    00000 1 WL_ST_taskPtr  DS       0              ; points to task rec in list
00111   0111    00000 1 WL_ST_newTime  DS       0              ; time-out time
00112   0112    00000 1 WL_ST_loopCnt  DS       0              ; loop counter

00113   0113    00000 1 WL_RT_saveQ    DS       0              ; return address
00114   0114    00000 1 WL_RT_runAddr  DS       0              ; address of task to run

00115   0115    00000 1 WL_RM_saveQ    DS       0              ; return address
00116   0116    00000 1 WL_RM_taskPtr  DS       0              ; points to task rec in list
00117   0117    00000 1 WL_RM_taskPtr2 DS       0              ; points to task rec behind taskPtr
00120   0120    00000 1 WL_RM_loopCnt  DS       0              ; loop counter
00121   0121    00000 1 WL_RM_retval   DS       0              ; tmp store for return value

00122   0122    00000 1 WL_IS_newTime  DS       0              ; INPUT: time to be inserted
00123   0123    00000 1 WL_IS_newAddr  DS       0              ; INPUT: address to be inserted
00124   0124    00000 1 WL_IS_saveQ    DS       0              ; return address
00125   0125    00000 1 WL_IS_taskPtr  DS       0              ; points to task rec in list
00126   0126    00000 1 WL_IS_taskPtr2 DS       0              ; points to task rec ahead of taskPtr
00127   0127    00000 1 WL_IS_loopCnt  DS       0              ; loop counter

                                       INCL     exec_e.asm     ; EXEC variables
                        ;===========================================================================
                        ; EXEC (file:exec_e.asm)
                        ;
                        ; Version:  1.0
                        ; Author:   John Pultorak
                        ; Date:     04/26/2002
                        ;
                        ; PURPOSE:
                        ; Eraseable memory variables and structures for the EXEX. See the EXEC
                        ; source code file for more information.
                        ;
                        ; The COLOSSUS version of this is on p. 70.
                        ;
                        ; ERRATA: The current version of the EXEC does not set the BANKSET parameter.
                        ; Instead, it stores the 14-bit CADR in LOC. Also, the JOBPRIOBASE field
                        ; has been added.
                        ;===========================================================================
```

```
                        MAXJOBS         EQU     7               ; max number jobs (not incl current job)

                        JRECSZ          EQU     13              ; size of job record (words)


                        ; (COLOSSUS, p. 70)
                        ; dynamically allocated core sets for EXEC jobs (8 sets)

                        ; record for current (running) job
                        ; Job priority: 0=no job, 1=lowest priority job, 2=...

                        EX_currentJob   EQU     *


                        MPAC            EQU     *               ; multi-purpose accumulator
00130   0130    00000 1                 DS      0
00131   0131    00000 1                 DS      0
00132   0132    00000 1                 DS      0
00133   0133    00000 1                 DS      0
00134   0134    00000 1                 DS      0
00135   0135    00000 1                 DS      0
00136   0136    00000 1                 DS      0

00137   0137    00000 1 MODE            DS      0               ; +1 for TP, +0 for DP, or -1 for vector
00140   0140    00000 1 LOC             DS      0               ; location associated with job
00141   0141    00000 1 BANKSET         DS      0               ; usually contains bank setting
00142   0142    00000 1 PUSHLOC         DS      0               ; word of packed interpretive parameters
00143   0143    00000 1 PRIORITY        DS      0               ; priority of present job and work area
00144   0144    00000 1 JOBPRIOBASE     DS      0               ; nominal job priority


                        ; records for additional jobs waiting to run

                        JREC0           EQU     *
                                        ORG     JREC0+JRECSZ

                        JREC1           EQU     *
                                        ORG     JREC1+JRECSZ

                        JREC2           EQU     *
                                        ORG     JREC2+JRECSZ

                        JREC3           EQU     *
                                        ORG     JREC3+JRECSZ

                        JREC4           EQU     *
                                        ORG     JREC4+JRECSZ

                        JREC5           EQU     *
                                        ORG     JREC5+JRECSZ

                        JREC6           EQU     *
                                        ORG     JREC6+JRECSZ


                        ; sorted list of jobs to run. The list is sorted by job priority
                        ; with the highest priority job at the top of the list. Each
                        ; entry on the list is a word index to a job record; the indexes are
                        ; relative to 'EX_currentJob', but the current job is not on the
                        ; list.

                        EX_jobList      EQU     *
                                        ORG     EX_jobList+MAXJOBS


                        LOCCTR          EQU     EX_jobList      ; index to next job record


                        CHGJOB          EQU     1               ; change jobs at next opportunity
                        KEEPJOB         EQU     0               ; keep the same job
00307   0307    00000 1 newJob          DS      0               ; change flag (set to CHGJOB or KEEPJOB)

00310   0310    00000 1 EX_JW_saveQ     DS      0               ; return address
00311   0311    00000 1 EX_JW_loopCnt   DS      0               ; loop counter
00312   0312    00000 1 EX_JW_CADR      DS      0               ; address of job to wake
00313   0313    00000 1 EX_JW_foundit   DS      0               ; 0=job not found, 1=found
00314   0314    00000 1 EX_JW_jobPtr    DS      0               ; points to job rec in list
00315   0315    00000 1 EX_JW_jobPtr2   DS      0               ; points to job rec ahead of jobPtr
00316   0316    00000 1 EX_JW_fndIndx   DS      0               ; index to awoken record

00317   0317    00000 1 EX_AJ_saveQ     DS      0               ; return address
00320   0320    00000 1 EX_AJ_loopCnt   DS      0               ; loop counter
00321   0321    00000 1 EX_AJ_jobPrio   DS      0               ; priority of new job
00322   0322    00000 1 EX_AJ_jobPtr    DS      0               ; initialized to EX_jobList at startup
00323   0323    00000 1 EX_AJ_field     DS      0               ; index to field from start of record
00324   0324    00000 1 EX_AJ_findx     DS      0               ; total index to field
```

```
00325    0325     00000 1 EX_IN_saveQ    DS       0              ; return address
00326    0326     00000 1 EX_IN_loopCnt  DS       0              ; loop counter
00327    0327     00000 1 EX_IN_jobPtr   DS       0              ; points to job rec in list
00330    0330     00000 1 EX_IN_recIndex DS       0              ; record index init counter
00331    0331     00000 1 EX_IN_field    DS       0              ; index to field from start of record
00332    0332     00000 1 EX_IN_findx    DS       0              ; total index to field

00333    0333     00000 1 EX_MN_runAddr  DS       0              ; address of job to run
00334    0334     00000 1 EX_MN_field    DS       0              ; index to field from start of record
00335    0335     00000 1 EX_MN_findx    DS       0              ; total index to field

00336    0336     00000 1 EX_RM_saveQ    DS       0              ; return address
00337    0337     00000 1 EX_RM_jobPtr   DS       0              ; points to job rec in list
00340    0340     00000 1 EX_RM_jobPtr2  DS       0              ; points to job rec behind jobPtr
00341    0341     00000 1 EX_RM_savePtr  DS       0              ; tmp store for index taken off list
00342    0342     00000 1 EX_RM_loopCnt  DS       0              ; loop counter
00343    0343     00000 1 EX_RM_retval   DS       0              ; tmp store for return value
00344    0344     00000 1 EX_RM_field    DS       0              ; index to field from start of record
00345    0345     00000 1 EX_RM_findx    DS       0              ; total index to field

00346    0346     00000 1 EX_IS_newPrio  DS       0              ; INPUT: priority to be inserted
00347    0347     00000 1 EX_IS_newPrioB DS       0              ; INPUT: nominal priority to be inserted
00350    0350     00000 1 EX_IS_newLoc   DS       0              ; INPUT: address to be inserted
00351    0351     00000 1 EX_IS_saveQ    DS       0              ; return address
00352    0352     00000 1 EX_IS_jobPtr   DS       0              ; points to job rec in list
00353    0353     00000 1 EX_IS_jobPtr2  DS       0              ; points to job rec ahead of jobPtr
00354    0354     00000 1 EX_IS_loopCnt  DS       0              ; loop counter
                                         INCL     dsky_e.asm     ; DSKY variables
                         ;==============================================================================
                         ; DSKY (file:dsky_e.asm)
                         ;
                         ; Version:  1.0
                         ; Author:   John Pultorak
                         ; Date:     12/14/2001
                         ;
                         ; PURPOSE:
                         ; Eraseable memory variables and structures for the DSKY. See the EXEC
                         ; source code file for more information.
                         ;
                         ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                         ; Oct 28, 1968.
                         ;==============================================================================

00355    0355     00000 1 FLAGWRD5       DS       0


                         ; GENERAL ERASABLE ASSIGNMENTS
                         ; (COLOSSUS, p. 66)


                         ; interrupt temporary storage pool
                         ; (ITEMP1 through RUPTREG4)

00356    0356     00000 1 ITEMP1         DS       0
                         WAITEXIT       EQU      ITEMP1
                         EXECTEM1       EQU      ITEMP1

00357    0357     00000 1 ITEMP2         DS       0
                         WAITBANK       EQU      ITEMP2
                         EXECTEM2       EQU      ITEMP2

00360    0360     00000 1 ITEMP3         DS       0
                         RUPTSTOR       EQU      ITEMP3
                         WAITADR        EQU      ITEMP3
                         NEWPRIO        EQU      ITEMP3

00361    0361     00000 1 ITEMP4         DS       0
                         ;LOCCTR          EQU     ITEMP4           ; moved to EXEC
                         WAITTEMP       EQU      ITEMP4

00362    0362     00000 1 ITEMP5         DS       0
                         NEWLOC         EQU      ITEMP5

00363    0363     00000 1 ITEMP6         DS       0
                         NEWLOCP1       EQU      ITEMP6         ; DP address

00364    0364     00000 1 NEWJOB         DS       0              ; COLOSSUS: must be at loc 68 due to wiring
00365    0365     00000 1 RUPTREG1       DS       0
00366    0366     00000 1 RUPTREG2       DS       0
00367    0367     00000 1 RUPTREG3       DS       0
00370    0370     00000 1 RUPTREG4       DS       0
                         KEYTEMP1       EQU      RUPTREG4
                         DSRUPTEM       EQU      RUPTREG4


                         ; FLAGWORD reservations
```

```
                        STATE           EQU     *               ; 12 words
00371   0371    00000 1                 DS      0
00372   0372    00000 1                 DS      0
00373   0373    00000 1                 DS      0
00374   0374    00000 1                 DS      0
00375   0375    00000 1                 DS      0
00376   0376    00000 1                 DS      0
00377   0377    00000 1                 DS      0
00400   0400    00000 1                 DS      0
00401   0401    00000 1                 DS      0
00402   0402    00000 1                 DS      0
00403   0403    00000 1                 DS      0
00404   0404    00000 1                 DS      0

                        FLAGFILL        EQU     *               ; space for future flags
00405   0405    00000 1                 DS      0
00406   0406    00000 1                 DS      0
00407   0407    00000 1                 DS      0
00410   0410    00000 1                 DS      0

                        ; pad load for DAPs
                        ; (COLOSSUS, p. 67)

                        EMDOT           EQU     FLAGFILL

                        ; exit for VB3

                        STATEXIT        EQU     FLAGFILL+2

                        ; EXEC temporaries which may be used between CCS NEWJOBS.
                        ; (INTB15P through RUPTMXM)

00411   0411    00000 1 INTB15P         DS      0               ; reflects 15th bit of indexable addresses
                        DSEXIT          EQU     INTB15P         ; return for DSPIN
                        EXITEM          EQU     INTB15P         ; return for scale factor routine select
                        BLANKRET        EQU     INTB15P         ; return for 2BLANK

00412   0412    00000 1 INTBIT15        DS      0               ; similar to above
                        WRDRET          EQU     INTBIT15        ; return for 5BLANK
                        WDRET           EQU     INTBIT15        ; return for DSPWD
                        DECRET          EQU     INTBIT15        ; return for PUTCOM (dec load)
                        _2122REG        EQU     INTBIT15        ; temp for CHARIN

                        ; The registers between ADDRWD and PRIORITY must stay in the following order
                        ; for interpretive trace.

00413   0413    00000 1 ADDRWD          DS      0               ; 12 bit interpretive operand subaddress
00414   0414    00000 1 POLISH          DS      0               ; holds CADR made from POLISH address
                        UPDATRET        EQU     POLISH          ; return for UPDATNN, UPDATVB
                        CHAR            EQU     POLISH          ; temp for CHARIN
                        ERCNT           EQU     POLISH          ; counter for error light reset
                        DECOUNT         EQU     POLISH          ; counter for scaling and display (dec)

00415   0415    00000 1 FIXLOC          DS      0               ; work area address
00416   0416    00000 1 OVFIND          DS      0               ; set non-zero on overflow

                        VBUF            EQU     *               ; temporary storage used for vectors
00417   0417    00000 1                 DS      0
00420   0420    00000 1                 DS      0
00421   0421    00000 1                 DS      0
00422   0422    00000 1                 DS      0
00423   0423    00000 1                 DS      0
00424   0424    00000 1                 DS      0
                        SGNON           EQU     VBUF            ; temp for +,- on
                        NOUNTEM         EQU     VBUF            ; counter for MIXNOUN fetch
                        DISTEM          EQU     VBUF            ; counter for octal display verbs
                        DECTEM          EQU     VBUF            ; counter for fetch (dec display verbs)

                        SGNOFF          EQU     VBUF+1          ; temp for +,- off
                        NVTEMP          EQU     VBUF+1          ; temp for NVSUB
                        SFTEMP1         EQU     VBUF+1          ; storage for SF const hi part(=SFTEMP2-1)
                        HITEMIN         EQU     VBUF+1          ; temp for load of hrs, min, sec
                        ; must = LOWTEMIN-1

                        CODE            EQU     VBUF+2          ; for DSPIN
                        SFTEMP2         EQU     VBUF+2          ; storage for SF const low part(=SFTEMP1+1)
                        LOWTEMIN        EQU     VBUF+2          ; temp for load of hrs, min, sec
                        ; must = HITEMIN+1

                        ; (COLOSSUS, p. 68)

                        MIXTEMP         EQU     VBUF+3          ; for MIXNOUN data
                        SIGNRET         EQU     VBUF+3          ; return for +,- on

                        ; Also, MIXTEMP+1 = VBUF+4, MIXTEMP+2 = VBUF+5

                        BUF             EQU     *               ; temporary scalar storage
```

```
00425    0425     00000 1                DS       0
00426    0426     00000 1                DS       0
00427    0427     00000 1                DS       0

00430    0430     00000 1 BUF2           DS       0
00431    0431     00000 1                DS       0

                         INDEXLOC       EQU      BUF            ; contains address of specified index
                         SWWORD         EQU      BUF            ; address of switch word
                         SWBIT          EQU      BUF+1          ; switch bit within switch word
00432    0432     00000 1 MPTEMP         DS       0              ; temporary used in multiply and shift
                         DMPNTEMP       EQU      MPTEMP         ; DMPSUB temporary
00433    0433     00000 1 DOTINC         DS       0              ; component increment for DOT subroutine
                         DVSIGN         EQU      DOTINC         ; determines sign of DDV result
                         ESCAPE         EQU      DOTINC         ; used in arcsin/arccos
                         ENTRET         EQU      DOTINC         ; exit from enter

00434    0434     00000 1 DOTRET         DS       0              ; return from DOT subroutine
                         DVNORMCT       EQU      DOTRET         ; dividend normalization count in DDV
                         ESCAPE2        EQU      DOTRET         ; alternate arcsin/arccos switch
                         WDCNT          EQU      DOTRET         ; char counter for DSPWD
                         INREL          EQU      DOTRET         ; input buffer selector (X,Y,Z REG)

00435    0435     00000 1 MATINC         DS       0              ; vector increment in MXV and VXM
                         MAXDVSW        EQU      MATINC         ; +0 if DP quotient is near one - else -1
                         POLYCNT        EQU      MATINC         ; polynomial loop counter
                         DSPMMTEM       EQU      MATINC         ; DSPCOUNT save for DSPMM
                         MIXBR          EQU      MATINC         ; indicator for mixed or normal noun

00436    0436     00000 1 TEM1           DS       0              ; EXEC temp
                         POLYRET        EQU      TEM1
                         DSREL          EQU      TEM1           ; rel address for DSPIN

00437    0437     00000 1 TEM2           DS       0              ; EXEC temp
                         DSMAG          EQU      TEM2           ; magnitude store for DSPIN
                         IDADDTEM       EQU      TEM2           ; mixnoun indirect address store

00440    0440     00000 1 TEM3           DS       0              ; EXEC temp
                         COUNT          EQU      TEM3           ; for DSPIN

00441    0441     00000 1 TEM4           DS       0              ; EXEC temp
                         LSTPTR         EQU      TEM4           ; list pointer for GRABUSY
                         RELRET         EQU      TEM4           ; return for RELDSP
                         FREERET        EQU      TEM4           ; return for FREEDSP
                         DSPWDRET       EQU      TEM4           ; return for DSPSIGN
                         SEPSCRET       EQU      TEM4           ; return for SEPSEC
                         SEPMNRET       EQU      TEM4           ; return for SEPMIN

00442    0442     00000 1 TEM5           DS       0              ; EXEC temp
                         NOUNADD        EQU      TEM5           ; temp storage for noun address

                         ; (COLOSSUS, p. 69)

00443    0443     00000 1 NNADTEM        DS       0              ; temp for noun address table entry
00444    0444     00000 1 NNTYPTEM       DS       0              ; temp for noun type table entry
00445    0445     00000 1 IDAD1TEM       DS       0              ; temp for indir address table entry (MIXNN)
                         ; must - IDAD2TEM-1, = IDAD3TEM-2
00446    0446     00000 1 IDAD2TEM       DS       0              ; temp for indir address table entry (MIXNN)
                         ; must - IDAD2TEM+1, = IDAD3TEM-1
00447    0447     00000 1 IDAD3TEM       DS       0              ; temp for indir address table entry (MIXNN)
                         ; must - IDAD1TEM+2, = IDAD2TEM+1
00450    0450     00000 1 RUTMXTEM       DS       0              ; temp for SF rout table entry (MIXNN only)

                         ; AX*SR*T storage

                         DEXDEX         EQU      TEM2           ; B(1) tmp
                         DEX1           EQU      TEM3           ; B(1) tmp
                         DEX2           EQU      TEM4           ; B(1) tmp
                         RTNSAVER       EQU      TEM5           ; B(1) tmp
                         TERM1TMP       EQU      BUF2           ; B(2) tmp



                         ; (COLOSSUS, p. 70) Note: the eraseable memory for the EXEC.
                         ; Moved to the EXEC area


                         ; (COLOSSUS, p. 72)
                         ; unswitched for display interface routines

00451    0451     00000 1 RESTREG        DS       0              ; B(1) prm for display starts
00452    0452     00000 1 NVWORD         DS       0
00453    0453     00000 1 MARXNV         DS       0
00454    0454     00000 1 NVSAVE         DS       0
                         ; (retain the order of CADRFLSH to FAILREG+2 for downlink purposes)
00455    0455     00000 1 CADRFLSH       DS       0              ; B(1) tmp
00456    0456     00000 1 CADRMARK       DS       0              ; B(1) tmp
```

```
00457    0457     00000 1 TEMPFLSH       DS       0              ; B(1) tmp

00460    0460     00000 1 FAILREG        DS       0              ; B(3) prm  3 alarm-abort user=S 2CADR
00461    0461     00000 1                DS       0
00462    0462     00000 1                DS       0

                         ; (COLOSSUS, p. 73)
                         ; verb 37 storage

00463    0463     00000 1 MINDEX         DS       0              ; B(1) tmp index for major mode
00464    0464     00000 1 MMNUMBER       DS       0              ; B(1) tmp major mode requested via V37

                         ; pinball interrupt storage

00465    0465     00000 1 DSPCNT         DS       0              ; B(1) prm DSPOUT counter

                         ; pinball executive action

00466    0466     00000 1 DSPCOUNT       DS       0              ; display position indicator
00467    0467     00000 1 DECBRNCH       DS       0              ; Bits2,1: octal=0, +dec=1, -dec=2
                         ; Bit5=R1 (dec), Bit4=R2 (dec), Bit3=R3 (dec)
00470    0470     00000 1 VERBREG        DS       0              ; verb code
00471    0471     00000 1 NOUNREG        DS       0              ; noun code
00472    0472     00000 1 XREG           DS       0              ; R1 input buffer
00473    0473     00000 1 YREG           DS       0              ; R2 input buffer
00474    0474     00000 1 ZREG           DS       0              ; R3 input buffer
00475    0475     00000 1 XREGLP         DS       0              ; low part of XREG (for ded conv only)
00476    0476     00000 1 YREGLP         DS       0              ; low part of YREG (for ded conv only)
                         HITEMOUT       EQU      YREGLP         ; temp for display of HRS, MIN, SEC
                         ;   must equal LOTEMOUT-1
00477    0477     00000 1 ZREGLP         DS       0              ; low part of ZREG (for ded conv only)
                         LOTEMOUT       EQU      ZREGLP         ; temp for display of HRS, MIN, SEC
                         ;   must equal HITEMOUT+1
                         ; (COLOSSUS, p. 74)

00500    0500     00000 1 MODREG         DS       0              ; mode code
00501    0501     00000 1 DSPLOCK        DS       0              ; keyboard/subroutine call interlock
00502    0502     00000 1 REQRET         DS       0              ; return register for load
00503    0503     00000 1 LOADSTAT       DS       0              ; status indicator for LOADTST
00504    0504     00000 1 CLPASS         DS       0              ; pass indicator clear
00505    0505     00000 1 NOUT           DS       0              ; activity counter for DSPTAB
00506    0506     00000 1 NOUNCADR       DS       0              ; machine CADR for noun
00507    0507     00000 1 MONSAVE        DS       0              ; N/V code for monitor (= MONSAVE1 - 1)
00510    0510     00000 1 MONSAVE1       DS       0              ; NOUNCADR for monitor (MATBS) = MONSAVE + 1
00511    0511     00000 1 MONSAVE2       DS       0              ; NVMONOPT options


                         ; The 11 register table for the display panel (COLOSSUS, p.74, p.306)
                         ; comment key =   RELADD: RELAYWD  BIT11  BITS10-6  BITS5-1

                         DSPTAB         EQU      *
00512    0512     00000 1                DS       0              ; 0: 0001  -R3  R3D4( 1)  R3D5( 0)
00513    0513     00000 1                DS       0              ; 1: 0010  +R3  R3D2( 3)  R3D3( 2)
00514    0514     00000 1                DS       0              ; 2: 0011  ---  R2D5( 5)  R3D1( 4)
00515    0515     00000 1                DS       0              ; 3: 0100  -R2  R2D3( 7)  R2D4( 6)
00516    0516     00000 1                DS       0              ; 4: 0101  +R2  R2D1(11)  R2D2(10)
00517    0517     00000 1                DS       0              ; 5: 0110  -R1  R1D4(13)  R1D5(12)
00520    0520     00000 1                DS       0              ; 6: 0111  +R1  R1D2(15)  R1D3(14)
00521    0521     00000 1                DS       0              ; 7: 1000  ---  --------  R1D1(16)
00522    0522     00000 1                DS       0              ; 8: 1001  ---  ND1 (21)  ND2 (20)
00523    0523     00000 1                DS       0              ; 9: 1010  ---  VD1 (23)  VD2 (22)
00524    0524     00000 1                DS       0              ; 10:1011  ---  MD1 (25)  MD1 (24)
00525    0525     00000 1                DS       0              ; 11: C/S lights

00526    0526     00000 1 NVQTEM         DS       0              ; NVSUB storage for calling address
                         ; must = NVBNKTEM-1
00527    0527     00000 1 NVBNKTEM       DS       0              ; NVSUB storage for calling bank
                         ; must = NVQTEM+1
00530    0530     00000 1 VERBSAVE       DS       0              ; needed for recycle
00531    0531     00000 1 CADRSTOR       DS       0              ; ENDIDLE storage
00532    0532     00000 1 DSPLIST        DS       0              ; waiting reg for DSP syst internal use
00533    0533     00000 1 EXTVRACT       DS       0              ; extended verb activity interlock

00534    0534     00000 1 DSPTEM1        DS       0              ; buffer storage area 1 (mostly for time)
00535    0535     00000 1                DS       0
00536    0536     00000 1                DS       0

00537    0537     00000 1 DSPTEM2        DS       0              ; buffer storage area 2 (mostly for deg)
00540    0540     00000 1                DS       0
00541    0541     00000 1                DS       0

                         DSPTEMX        EQU      DSPTEM2        ; B(2) S-S display buffer for external verbs
                         NORMTEM1       EQU      DSPTEM1        ; B(3) DSP normal display registers

                         ; display for extended verbs

                         OPTIONX        EQU      DSPTEMX        ; B(2) extended verb option code N12(VB2)
```

```
                        ; temp store for major mode change

00542   0542    00000 1 MMTEMP          DS      0


                        ; T4RUPT Erasable

00543   0543    00000 1 DSRUPTSW        DS      0               ; (COLOSSUS, p. 78)
00544   0544    00000 1 T4RET           DS      0               ; added, not part of COLOSSUS
00545   0545    00000 1 DSPOUTRET       DS      0               ; added, not part of COLOSSUS
00546   0546    00000 1 DK_IN_saveQ     DS      0               ; return for T4RUPT init


                        ; Replacement for Block II LXCH instruction (not part of COLOSSUS)

00547   0547    00000 1 LXCH_LPRET      DS      0               ; LP return address
00550   0550    00000 1 LXCH_A          DS      0               ; save A


                        ; vars for KEYPROG

00551   0551    00000 1 KP_MPAC         DS      0


                        ; Vars for DPTEST (not part of COLOSSUS)

00552   0552    00000 1 DPTEST_A        DS      0
00553   0553    00000 1 DPTEST_Q        DS      0


                        ; Vars for REQDATX, REQDATY, REQDATZ (not part of COLOSSUS)

00554   0554    00000 1 REQ_Q           DS      0


                        ; Vars for SETNCADR (not part of COLOSSUS)

00555   0555    00000 1 SETNCADR_Q      DS      0


                        ; Vars for ALLDC_OC (not part of COLOSSUS)

00556   0556    00000 1 ALLDC_OC_Q      DS      0


                        ; Vars for SFRUTMIX (not part of COLOSSUS)

00557   0557    00000 1 SFRUTMIX_L      DS      0


                        ; Vars for SFCONUM (not part of COLOSSUS)

00560   0560    00000 1 SFCONUM_L       DS      0

                        ; vars for BLANKSUB (not part of COLOSSUS)

00561   0561    00000 1 BLANKSUB_Q      DS      0

                        ; Vars for GTSFOUT, GTSFIN (not part of COLOSSUS)

00562   0562    00000 1 GTSF_RET        DS      0

                        ; Vars for FIXRANGE (not part of COLOSSUS)

00563   0563    00000 1 FR_RETQ         DS      0

                        ; Vars for NVSUB (not part of COLOSSUS)

00564   0564    00000 1 NVSUB_L         DS      0
00565   0565    00000 1 NVSUB_A         DS      0

                        ; Vars for ENDIDLE (not part of COLOSSUS)

00566   0566    00000 1 ENDIDLE_L       DS      0

                        ; Vars for NVSUBUSY (not part of COLOSSUS)

00567   0567    00000 1 NBSUBSY1_L      DS      0

                        ; Vars for FLASHON/FLASHOFF (not part of COLOSSUS)

00570   0570    00000 1 FLASHRET        DS      0

                        ; vars for PASTEVB (not part of COLOSSUS)

00571   0571    00000 1 PASTE_TMP       DS      0
```

```
                        ; vars for NEWMODEA (not part of COLOSSUS)

00572   0572    00000 1 NEWMODEA_Q      DS      0

                        ; Vars for MATH LIB (not part of COLOSSUS)

00573   0573    00000 1 SHORTMP_A       DS      0
00574   0574    00000 1 SHORTMP_OVFL    DS      0
00575   0575    00000 1 SHORTMP_OVFH    DS      0
00576   0576    00000 1 ADDRWD1         DS      0
00577   0577    00000 1 MATH_Q          DS      0
00600   0600    00000 1 PRSHRTMP_Q      DS      0


                        ; KEYRUPT Eraseable

00601   0601    00000 1 KEYRET          DS      0               ; added, not part of COLOSSUS

00602   0602    00000 1 SAVEQ           DS      0               ; temp for return addr


                        ; Bank intercommunication

00603   0603    00000 1 BJBANK          DS      0
00604   0604    00000 1 BJRET           DS      0

00605   0605    00000 1 PJBANK          DS      0
00606   0606    00000 1 PJRET           DS      0
00607   0607    00000 1 PJA             DS      0

00610   0610    00000 1 BCBANK          DS      0
00611   0611    00000 1 BCRET           DS      0
00612   0612    00000 1 BCA             DS      0

00613   0613    00000 1 MBCBANK         DS      0
00614   0614    00000 1 MBCRET          DS      0
00615   0615    00000 1 MBCA            DS      0

00616   0616    00000 1 DCBANK          DS      0
00617   0617    00000 1 DCRET           DS      0


                        ; FIXED MEMORY DECLARATIONS

                                        ORG     EXTENDER
05777   5777    47777 0                 DS      %47777          ; needed for EXTEND

                        ;-----------------------------------------------------------------------------
                        ; RESTART/INTERRUPT ENTRY POINTS
                        ;-----------------------------------------------------------------------------

                        ; Program (re)start
                                        ORG     GOPROG
02000   2000 0  1,2126 0               TC      goMAIN          ; AGC (re)start begins here!

                        ; Interrupt vectors
                                        ORG     T3RUPT
02004   2004 5  0,0026 0               TS      ARUPT           ; TIME3 interrupt vector
02005   2005 3  0,0001 0               XCH     Q
02006   2006 5  0,0027 1               TS      QRUPT
02007   2007 0  1,2034 1               TC      goT3

                                        ORG     ERRUPT
02010   2010 5  0,0026 0               TS      ARUPT
02011   2011 3  0,0001 0               XCH     Q
02012   2012 5  0,0027 1               TS      QRUPT
02013   2013 0  1,2036 0               TC      goER

                                        ORG     DSRUPT          ; T4RUPT for DSKY display
02014   2014 5  0,0026 0               TS      ARUPT
02015   2015 3  0,0001 0               XCH     Q
02016   2016 5  0,0027 1               TS      QRUPT
02017   2017 0  1,2037 1               TC      goDS

                                        ORG     KEYRUPT         ; DSKY keyboard interrupt vector
02020   2020 5  0,0026 0               TS      ARUPT
02021   2021 3  0,0001 0               XCH     Q
02022   2022 5  0,0027 1               TS      QRUPT
02023   2023 0  1,2041 0               TC      goKEY

                                        ORG     UPRUPT
02024   2024 5  0,0026 0               TS      ARUPT
02025   2025 3  0,0001 0               XCH     Q
02026   2026 5  0,0027 1               TS      QRUPT
02027   2027 0  1,2043 1               TC      goUP
```

```
                        ; restore Q and A registers and resume

                        endRUPT         EQU     *
02030    2030 3  0,0027 1               XCH     QRUPT           ; restore Q
02031    2031 5  0,0001 0               TS      Q
02032    2032 3  0,0026 0               XCH     ARUPT           ; restore A
02033    2033 2  0,0000 1               RESUME                  ; resume normal program execution

                        ;-----------------------------------------------------------------------------
                        ; RUPT (INTERRUPT) SERVICE ROUTINES
                        ;
                        ; Upon entry, registers will contain these values:
                        ; - ZRUPT: Prior contents of program counter (Z register).
                        ; - BRUPT: Prior contents of B register.
                        ; - ARUPT: Prior contents of accumulator (A register).
                        ; - QRUPT: Prior contents of Q register.
                        ;
                        ; When the service routine is finished, jump to endRUPT to restore the A
                        ; and Q registers. Call RESUME to restore Z and B, which causes a return
                        ; to normal (non-interrupt) execution. Interrupts are disabled upon entry
                        ; to the service routine; they are reenabled following RESUME.
                        ;-----------------------------------------------------------------------------

                        goT3            EQU     *
02034    2034 0  1,2347 0               TCR     WL_TIME3task    ; handle T3RUPT for WAITLIST
02035    2035 0  1,2030 0               TC      endRUPT

                        goER            EQU     *
02036    2036 0  1,2030 0               TC      endRUPT

                        goDS            EQU     *
02037    2037 0  2,4047 0               TCR     T4PROG          ; handle T4RUPT for DSKY display
02040    2040 0  1,2030 0               TC      endRUPT

                        goKEY           EQU     *
02041    2041 0  2,4132 0               TCR     KEYPROG         ; handle keyrupt for keyboard entry
02042    2042 0  1,2030 0               TC      endRUPT

                        goUP            EQU     *
02043    2043 0  1,2030 0               TC      endRUPT

                        ;-----------------------------------------------------------------------------
                        ; FIXED MEMORY CONSTANTS
                        ;-----------------------------------------------------------------------------

02044    2044     00200 0 ofbit          DS      %200            ; OUT1, bit 8 initiates standby

02045    2045     77777 0 NEG0           DS      -0
02046    2046     77776 1 NEG1           DS      -1
02047    2047     77775 1 NEG2           DS      -2

02050    2050     00000 1 ZERO           DS      0
02051    2051     00001 0 ONE            DS      1
02052    2052     00002 0 TWO            DS      2
02053    2053     00003 1 THREE          DS      3
02054    2054     00004 0 FOUR           DS      4
02055    2055     00005 1 FIVE           DS      5
02056    2056     00006 1 SIX            DS      6
02057    2057     00007 0 SEVEN          DS      7
02060    2060     00012 1 TEN            DS      10
02061    2061     00013 0 ELEVEN         DS      11

                        ; must be in reverse order. Pinball treats this as a table
                        ; and indexes thru it.

02062    2062     40000 0 BIT15          DS      %40000
02063    2063     20000 0 BIT14          DS      %20000
02064    2064     10000 0 BIT13          DS      %10000
02065    2065     04000 0 BIT12          DS      %04000
02066    2066     02000 0 BIT11          DS      %02000
02067    2067     01000 0 BIT10          DS      %01000
02070    2070     00400 0 BIT9           DS      %00400
02071    2071     00200 0 BIT8           DS      %00200
02072    2072     00100 0 BIT7           DS      %00100
02073    2073     00040 0 BIT6           DS      %00040
02074    2074     00020 0 BIT5           DS      %00020
02075    2075     00010 0 BIT4           DS      %00010
02076    2076     00004 0 BIT3           DS      %00004
02077    2077     00002 0 BIT2           DS      %00002
02100    2100     00001 0 BIT1           DS      %00001

02101    2101     00177 0 LOW7           DS      %00177

02102    2102     06000 1 bankAddr       DS      %6000           ; fixed-switchable addr range starts here
02103    2103     01777 1 lowAddr        DS      %1777           ; mask for 10-bit address
02104    2104     01400 1 OCT1400        DS      %1400
```

```
02105    2105     00013 0 NOUTCON        DS       11

02106    2106     37777 1 POSMAX         DS       %37777

                          ;-----------------------------------------------------------------------
                          ; CLRMEM - INITIALIZE ERASEABLE MEMORY
                          ;
                          ; Uses QRUPT and ARUPT as scratchpad. This is OK, because interrupts
                          ; are disabled anyway. All eraseable memory above the AGC clock (TIME1,
                          ; TIME2) is cleared. The AGC clock is not cleared because this might
                          ; be a restart or a startup from standby mode.
                          ;-----------------------------------------------------------------------

                          CLRMEM         EQU      *
02107    2107 3  0,0001 0               XCH      Q
02110    2110 5  0,0027 1               TS       QRUPT          ; save return address

02111    2111 3  1,2123 0               XCH      CLRMEM_WC      ; init count of words to clear
02112    2112 5  0,0026 0               TS       ARUPT

                          CLRMEM_CHK     EQU      *
02113    2113 1  0,0026 1               CCS      ARUPT
02114    2114 0  1,2116 0               TC       CLRMEM_WORD
02115    2115 0  0,0027 1               TC       QRUPT          ; return

                          CLRMEM_WORD    EQU      *
02116    2116 5  0,0026 0               TS       ARUPT
02117    2117 3  1,2050 0               CAF      CLRMEM_VAL
02120    2120 2  0,0026 1               INDEX    ARUPT
02121    2121 5  0,0037 0               TS       CLRMEM_BADDR   ; clear a word
02122    2122 0  1,2113 0               TC       CLRMEM_CHK     ; done?

                          CLRMEM_VAL     EQU      ZERO           ; set memory to this value
                          CLRMEM_BADDR   EQU      TIME3          ; base address to clear
02123    2123     01741 1 CLRMEM_WC      DS       %1777-TIME3+1  ; clear everything >= TIME3


                          ;-----------------------------------------------------------------------
                          ; FRESH START
                          ;
                          ; AGC starts executing here, following power-up, or restart.
                          ;-----------------------------------------------------------------------

02124    2124     10000 0 V37BANK        DS       %10000         ; BANK (4) containg PREMM1, FCADRMM1
02125    2125     37600 0 SAMASK         DS       %37600         ; mask to zero lower 7 bits


                          goMAIN         EQU      *
                          SLAP1          EQU      goMAIN         ; entry for V36 (fresh start request)

02126    2126 2  0,0000 0               INHINT

                          ; First, check for standby operation. Loosely based on the standby
                          ; algorithm in R-393. Probably should flash the 'computer activity'
                          ; light as well.

02127    2127 3  1,2071 0               CAF      BIT8           ; add 2 to 7th power to AGC clock
02130    2130 6  0,0036 1               AD       TIME1
02131    2131 5  0,0036 1               TS       TIME1

02132    2132 3  1,2050 0               CAF      ZERO           ; skipped on ovf and C(A) set to 1
02133    2133 6  0,0035 1               AD       TIME2          ; bump TIME2 with overflow, if any
02134    2134 5  0,0035 1               TS       TIME2

02135    2135 3  1,2125 0               CAF      SAMASK         ; zero the LSBs of TIME1
02136    2136 7  0,0036 0               MASK     TIME1
02137    2137 5  0,0036 1               TS       TIME1

02140    2140 3  1,2044 0               XCH      ofbit          ; enable standby operation
02141    2141 5  0,0011 1               TS       OUT1

02142    2142 0  1,2107 0               TC       CLRMEM         ; clear everything but the AGC clock

                          ; set fresh start major mode to P00 (AGC CMC idle)

02143    2143 3  1,2124 1               CAF      V37BANK
02144    2144 5  0,0015 0               TS       BANK           ; bank for major mode tables

02145    2145 3  4,6046 0               CAF      NOV37MM        ; assumes BANK is set (above)
02146    2146 5  0,0463 0               TS       MINDEX         ; index to P00


                          goMMchange     EQU      *
02147    2147 2  0,0000 0               INHINT                  ; inhibit interrupts

                          ; Initialize WAITLIST and EXEC eraseable memory. Initialize DSKY eraseable
```

```
                        ; memory (but don't initialize BANK or MINDEX; they are used to start the
                        ; main job for this major mode.

02150    2150 0  1,3252 1               TCR      EX_initEX      ; initialize EXEC
02151    2151 0  1,2204 0               TCR      WL_initWL      ; initialize WAITLIST
02152    2152 0  2,4007 1               TCR      DK_initDK      ; initialize DSKY

                        ; Start the major mode job. This is modified from COLOSSUS because block I
                        ; doesn't have E-bank and my SPVAC interface is a little different from the
                        ; original. The references to PREMM1 and FCADRMM1 assume that the BANK is
                        ; set to the one containing those tables.

                        V37XEQ         EQU      *
02153    2153 2  0,0000 0               INHINT
02154    2154 2  0,0463 1               INDEX    MINDEX
02155    2155 3  4,6037 0               CAF      PREMM1
02156    2156 5  0,0542 1               TS       MMTEMP

02157    2157 7  2,4666 1               MASK     HI5            ; obtain priority bits 15-11
02160    2160 0  2,4640 1               TC       RIGHT5
02161    2161 0  2,4640 1               TC       RIGHT5         ; shift right to bits 5-1
02162    2162 5  0,0360 1               TS       NEWPRIO        ; store PRIO for SPVAC

02163    2163 2  0,0463 1               INDEX    MINDEX
02164    2164 3  4,6030 1               CAF      FCADRMM1

02165    2165 0  1,3075 0               TC       SPVAC          ; job CADR in C(A), job prio in NEWPRIO

                        V37XEQC        EQU      *
02166    2166 3  1,2050 0               CAF      ZERO           ; was CA MMTEMP in Block II
02167    2167 6  0,0542 1               AD       MMTEMP         ; upon return from FINDVAC, place the
02170    2170 7  1,2101 1               MASK     LOW7           ; new MM in MODREG (the low 7 bits of
02171    2171 0  2,5036 1               TC       NEWMODEA       ; PHSERDT1)

02172    2172 0  2,5003 1               TC       RELDSP         ; release display

                        ; Start the EXEC.

02173    2173 0  1,2656 0               TC       EX_exec        ; never returns


                        ;-----------------------------------------------------------------------
                        ; AGC LIBRARIES
                        ;
                        ; System services in fixed-fixed memory.
                        ;-----------------------------------------------------------------------

                                       INCL     waitlist_f.asm ; WAITLIST, incl. T3RUPT handler
                        ;=======================================================================
                        ; WAITLIST (file:waitlist_f.asm)
                        ;
                        ; Version:  1.0
                        ; Author:   John Pultorak
                        ; Date:     11/15/2001
                        ;
                        ; PURPOSE:
                        ; Constants and source code for WAITLIST.
                        ;
                        ; Non-preemptive interrupt timer routines, originally implemented by J. H.
                        ; Laning, Jr. for AGC3 and later adapted for AGC4. Briefly discussed in
                        ; R-393, which gives some of the software interfaces into the WAITLIST.
                        ; This is my own recreation, and the internals may differ from the original.
                        ;
                        ; A task is scheduled for execution by calling 'WAITLIST' and
                        ; furnishing the time-out time and starting address.
                        ;       L        XCH      TASK_TIMEOUT     ; in 10 mSec ticks
                        ;       L+1      TC       WAITLIST
                        ;       L+2      DS       TASK_ADDRESS     ; 14-bit address
                        ;       L+3      ... execution resumes here
                        ;
                        ; TASK_TIMEOUT = a positive integer from 1 - MAXDELAY that specifies the delay
                        ;    in 10 mSec ticks. Maximum delay is 12000 (2 minutes).
                        ; TASK_ADDRESS = starting address of the task (14-bit address)
                        ;
                        ; WAITLIST can be called from from an interrupt, or from normal execution.
                        ; It is the only public function of the waitlist.
                        ;
                        ; **** WARNING **** If WAITLIST is not called from an interrupt, be sure to
                        ; inhibit interrupts before calling it to protect the integrity of the list.
                        ;
                        ; Tasks execute when TIME3 overflows and generates an interrupt (T3RUPT).
                        ; The task executes during the interrupt. Tasks terminate themselves by
                        ; jumping to TASKOVER.
                        ;       TC       TASKOVER
                        ;
                        ; Because tasks execute during an interrupt, they should be fairly short.
                        ; Tasks can initiate longer operations by scheduling a 'job' using EXEC.
```

```
                        ;==============================================================================

02174    2174     00002 0 WL_taskRecSize DS       TRECSZ         ; size of a task record (words)
02175    2175     00057 0 WL_tskLstStart DS       WL_taskList    ; starting address for task list
02176    2176     00073 0 WL_tskLstEnd   DS       MAXTASK-1@TRECSZ+WL_taskList
02177    2177     00006 1 WL_numTasks    DS       MAXTASK-1      ; init loop counter for all tasks
02200    2200     00005 1 WL_numTasks1   DS       MAXTASK-2      ; init loop counter for all tasks - 1

02201    2201     37777 1 WL_maxVal      DS       MAXVAL
02202    2202     27340 0 WL_maxDelay    DS       MAXDELAY
02203    2203     10440 0 WL_maxTimeOut  DS       MAXTIMEOUT


                        ;------------------------------------------------------------------------------
                        ; WL_initWL - INITIALIZE WAITLIST
                        ;
                        ; Subroutine initializes the eraseable memory segment for WAITLIST.
                        ; Necessary in case the AGC is restarted.
                        ;
                        ; Note: the valid range for TIME3 is 10440 to 37777 (which spans
                        ;   12000 (base 10) ticks, which corresponds to 120 seconds)
                        ;   positive overflow occurs at 40000, which triggers T3RUPT.
                        ;   TIME3 values of 0 to 10437 are illegal; these values occur
                        ;   after timeout when the counter overflows. TIME3 values in this
                        ;   range indicate that timeout has occurred and that T3RUPT is
                        ;   presently occuring, or is pending.
                        ;------------------------------------------------------------------------------

                        WL_initWL      EQU      *
02204    2204 3  0,0001 0              XCH      Q
02205    2205 5  0,0075 0              TS       WL_IN_saveQ    ; save return address

02206    2206 3  1,2203 1              CAF      WL_maxTimeOut
02207    2207 5  0,0037 0              TS       TIME3

                        ; Iterate through task list and initialize all records to NIL

02210    2210 3  1,2175 0              CAF      WL_tskLstStart ; init pointer to start of list
02211    2211 5  0,0076 0              TS       WL_IN_taskPtr

02212    2212 3  1,2177 1              CAF      WL_numTasks    ; loop for number of tasks
                        WL_IN_loop     EQU      *
02213    2213 5  0,0077 1              TS       WL_IN_loopCnt

02214    2214 3  1,2202 0              CAF      WL_maxDelay
02215    2215 2  0,0076 1              INDEX    WL_IN_taskPtr
02216    2216 5  0,0000 1              TS       TSKTIME

02217    2217 3  1,2050 0              CAF      ZERO
02220    2220 2  0,0076 1              INDEX    WL_IN_taskPtr
02221    2221 5  0,0001 0              TS       TSKADDR

02222    2222 3  0,0076 0              XCH      WL_IN_taskPtr  ; bump task pointer back 1 record
02223    2223 6  1,2174 1              AD       WL_taskRecSize
02224    2224 5  0,0076 0              TS       WL_IN_taskPtr

02225    2225 1  0,0077 0              CCS      WL_IN_loopCnt  ; done checking task list?
02226    2226 0  1,2213 0              TC       WL_IN_loop     ; not yet

02227    2227 3  0,0075 0              XCH      WL_IN_saveQ
02230    2230 5  0,0001 0              TS       Q              ; restore return address
02231    2231 0  0,0000 0              RETURN

                        ;------------------------------------------------------------------------------
                        ; WAITLIST - ADD TASK TO WAITLIST
                        ;
                        ; Subroutine adds a task to WL_taskList. The following conditions are
                        ; true upon entry.
                        ; 1) The task list is sorted so the next task scheduled for execution
                        ;    is at the front of the list.
                        ; 2) If no tasks are currently scheduled, the task record at the front
                        ;    of the list will be NIL.
                        ; 3) Unused (NIL) records in the task list have their time fields set to
                        ;    MAXDELAY and their address fields set to zero.
                        ; 4) If any tasks are on the waitlist, the time field in that task's
                        ;    record will contain the remaining time AFTER the next timeout. The
                        ;    task scheduled for execution at timeout will have a time remaining
                        ;    of zero.
                        ;    Any other tasks that will execute at that time will also have a time of
                        ;    zero. Tasks that will execute some time in the future AFTER timeout
                        ;    will have nonzero times; these times indicate the additional time
                        ;    needed after the next timeout.
                        ;
                        ; This is the only 'public' function. It can be called from a job or from
                        ; a task or other interrupt. It disables interrupts to maintain the integrity
                        ; of the taskList.
```

```
                        ;-----------------------------------------------------------------------

                        WAITLIST        EQU     *
02232    2232 5  0,0102 1              TS      WL_AT_newTime  ; save task time
02233    2233 3  0,0001 0              XCH     Q
02234    2234 5  0,0100 0              TS      WL_AT_saveQ    ; save return address-1

02235    2235 3  1,2050 0              CAF     ZERO
02236    2236 2  1,2176 1              INDEX   WL_tskLstEnd
02237    2237 6  0,0001 0              AD      TSKADDR
02240    2240 1  0,0000 0              CCS     A              ; list full?
02241    2241 0  1,2343 1              TC      WL_AT_done     ; >0 yes, so give up

                        ; Calculate time remaining until currently scheduled time-out.

02242    2242 3  1,2050 0              CAF     ZERO
02243    2243 6  0,0037 0              AD      TIME3          ; get time
02244    2244 5  0,0103 0              TS      WL_AT_timeLeft ; save it, temporarily

                        ; Did TIME3 recently overflow? If so, we are inside T3RUPT, or T3RUPT
                        ; is pending. TIME3 values from 0 - 10437 are not legal, so they
                        ; indicate that an overflow has occurred.

02245    2245 4  1,2203 0              CS      WL_maxTimeOut
02246    2246 6  0,0103 0              AD      WL_AT_timeLeft
02247    2247 1  0,0000 0              CCS     A              ; TIME3 recently overflowed?
02250    2250 0  1,2264 0              TC      WL_AT_noOvf    ; >0 no
02251    2251 0  1,2264 0              TC      WL_AT_noOvf    ; +0 no
02252    2252 0  1,2254 0              TC      *+2            ; <0 yes
02253    2253 0  1,2264 0              TC      WL_AT_noOvf    ; -0 no

                        ; TIME3 already timed-out, so we must be inside T3RUPT, or T3RUPT
                        ; is pending. Just add the new task to the list. No time correction
                        ; is necessary; the epoch is NOW.

02254    2254 3  1,2050 0              CAF     ZERO
02255    2255 6  0,0102 1              AD      WL_AT_newTime
02256    2256 5  0,0122 0              TS      WL_IS_newTime  ; set time field in new task record

02257    2257 2  0,0100 1              INDEX   WL_AT_saveQ    ; indirectly address WL_AT_saveQ
02260    2260 3  0,0000 1              CAF     0
02261    2261 5  0,0123 1              TS      WL_IS_newAddr  ; set addr field in new task record

02262    2262 0  1,2473 0              TCR     WL_insert      ; add new task to task list
02263    2263 0  1,2343 1              TC      WL_AT_done

                        ; TIME3 has not timed out yet. Calculate time remaining until timeout
                        ; (timeout occurs when TIME3 overflows)

                        WL_AT_noOvf     EQU     *
02264    2264 4  0,0103 1              CS      WL_AT_timeLeft ; get -TIME3
02265    2265 6  1,2201 0              AD      WL_maxVal
02266    2266 6  1,2051 1              AD      ONE
02267    2267 5  0,0103 0              TS      WL_AT_timeLeft ; time left = -TIME3 + %37777 + 1

                        ; Compare that time against the timeout for the new task.

                        WL_AT_chkOrder  EQU     *
02270    2270 4  0,0102 0              CS      WL_AT_newTime
02271    2271 6  0,0103 0              AD      WL_AT_timeLeft
02272    2272 1  0,0000 0              CCS     A              ; compare new task to current
02273    2273 0  1,2306 0              TC      WL_AT_mkFirst  ; >0 (make new task 1st)
02274    2274 0  1,2276 0              TC      *+2            ; +0
02275    2275 0  1,2276 0              TC      *+1            ; <0

                        ; The new task does not need to run before the current time-out, so
                        ; just add it to the list. Subtract the remaining time interval from the
                        ; new task's time, so the new task will have the same epoch as the other
                        ; tasks on the list.

02276    2276 4  0,0103 1              CS      WL_AT_timeLeft
02277    2277 6  0,0102 1              AD      WL_AT_newTime  ; make epoch correction
02300    2300 5  0,0122 0              TS      WL_IS_newTime  ; set time field in new task record

02301    2301 2  0,0100 1              INDEX   WL_AT_saveQ    ; indirectly address WL_AT_saveQ
02302    2302 3  0,0000 1              CAF     0
02303    2303 5  0,0123 1              TS      WL_IS_newAddr  ; set addr field in new task record

02304    2304 0  1,2473 0              TCR     WL_insert      ; add new task to task list
02305    2305 0  1,2343 1              TC      WL_AT_done

                        ; The new task needs to run prior to the current time-out. Add the time
                        ; remaining to all tasks currently on the list to change their epoch
                        ; to NOW.

                        WL_AT_mkFirst   EQU     *
02306    2306 3  1,2175 0              CAF     WL_tskLstStart ; set pointer to front of list
```

```
02307    2307 5  0,0101 1               TS       WL_AT_taskPtr

02310    2310 3  1,2177 1               CAF      WL_numTasks    ; loop for number of tasks
                         WL_AT_loop     EQU      *
02311    2311 5  0,0104 1               TS       WL_AT_loopCnt

02312    2312 3  1,2050 0               CAF      ZERO
02313    2313 2  0,0101 0               INDEX    WL_AT_taskPtr
02314    2314 6  0,0001 0               AD       TSKADDR
02315    2315 1  0,0000 0               CCS      A              ; end of list?
02316    2316 0  1,2320 1               TC       *+2            ; >0 no, so keep going
02317    2317 0  1,2333 0               TC       WL_AT_schTsk   ; +0 yes, add the new task

02320    2320 3  1,2050 0               CAF      ZERO
02321    2321 2  0,0101 0               INDEX    WL_AT_taskPtr
02322    2322 6  0,0000 1               AD       TSKTIME
02323    2323 6  0,0103 0               AD       WL_AT_timeLeft ; time-out = time-out + timeLeft
02324    2324 2  0,0101 0               INDEX    WL_AT_taskPtr
02325    2325 5  0,0000 1               TS       TSKTIME

02326    2326 3  0,0101 1               XCH      WL_AT_taskPtr  ; bump task pointer back 1 record
02327    2327 6  1,2174 1               AD       WL_taskRecSize
02330    2330 5  0,0101 1               TS       WL_AT_taskPtr

02331    2331 1  0,0104 0               CCS      WL_AT_loopCnt  ; done fixing the times?
02332    2332 0  1,2311 0               TC       WL_AT_loop     ; not yet

                         ; Now that the tasks all share the same epoch, add the new task to the
                         ; list and call the scheduler to schedule the next task.

                         WL_AT_schTsk   EQU      *
02333    2333 3  1,2050 0               CAF      ZERO
02334    2334 6  0,0102 1               AD       WL_AT_newTime
02335    2335 5  0,0122 0               TS       WL_IS_newTime  ; set time field in new task record

02336    2336 2  0,0100 1               INDEX    WL_AT_saveQ    ; indirectly address WL_AT_saveQ
02337    2337 3  0,0000 1               CAF      0
02340    2340 5  0,0123 1               TS       WL_IS_newAddr  ; set addr field in new task record

02341    2341 0  1,2473 0               TCR      WL_insert      ; add new task to task list

02342    2342 0  1,2417 1               TCR      WL_schedTask   ; schedule the next task

                         WL_AT_done     EQU      *
02343    2343 3  0,0100 0               XCH      WL_AT_saveQ
02344    2344 6  1,2051 1               AD       ONE
02345    2345 5  0,0001 0               TS       Q              ; restore return address
02346    2346 0  0,0000 0               RETURN

                         ;--------------------------------------------------------------------------
                         ; WL_TIME3task - T3 TIMEOUT
                         ;
                         ; Perform WAITLIST activities when TIME3 times-out. Called by the
                         ; T3 interrupt handler.
                         ;--------------------------------------------------------------------------

                         WL_TIME3task   EQU      *
02347    2347 3  0,0001 0               XCH      Q
02350    2350 5  0,0105 0               TS       WL_T3_saveQ    ; save return address
02351    2351 3  0,0015 0               XCH      BANK
02352    2352 5  0,0106 0               TS       WL_T3_oldBank  ; save current bank

                         ; Execute all timed-out tasks.

02353    2353 0  1,2362 1               TCR      WL_runTasks

                         ; Set up TIME3 to overflow at the next task's time-out.
                         ; Adjust the time-outs for all remaining tasks.

02354    2354 0  1,2417 1               TCR      WL_schedTask

02355    2355 3  0,0106 0               XCH      WL_T3_oldBank
02356    2356 5  0,0015 0               TS       BANK           ; restore previous bank
02357    2357 3  0,0105 0               XCH      WL_T3_saveQ
02360    2360 5  0,0001 0               TS       Q              ; restore return address
02361    2361 0  0,0000 0               RETURN

                         ;--------------------------------------------------------------------------
                         ; WL_runTasks - RUN TIMED-OUT TASK(S)
                         ;
                         ; Runs all tasks timed-out on WL_taskList. Tasks are removed
                         ; from the list before they are run.
                         ;--------------------------------------------------------------------------

                         WL_runTasks    EQU      *
02362    2362 3  0,0001 0               XCH      Q
02363    2363 5  0,0113 1               TS       WL_RT_saveQ    ; save return address
```

```
                        ; loop, checking the task on the front of the list. If it is
                        ; timed out, remove it from the list and run it.

                        WL_RT_loop      EQU     *
02364   2364 3  1,2050 0                CAF     ZERO
02365   2365 2  1,2175 1                INDEX   WL_tskLstStart
02366   2366 6  0,0000 1                AD      TSKTIME
02367   2367 1  0,0000 0                CCS     A               ; task timed out?
02370   2370 0  1,2414 1                TC      WL_RT_done      ; >0 no, so we are done
02371   2371 0  1,2373 1                TC      *+2             ; +0
02372   2372 0  1,2373 1                TC      *+1             ; <0

                        ; This task has timed out, so run it.

02373   2373 0  1,2565 0                TCR     WL_remove       ; remove task from list
02374   2374 5  0,0114 0                TS      WL_RT_runAddr   ; save 14-bit address of task to run

                        ; The task address is always 14-bit, so check whether the address falls
                        ; within erasable or fixed-fixed memory. If so, use it as-is; otherwise,
                        ; set the bank register and change the address to 12-bit.

02375   2375 4  0,0000 0                COM                     ; -(14bitAddr)+%6000
02376   2376 6  1,2102 0                AD      bankAddr
02377   2377 1  0,0000 0                CCS     A               ; task is bank addressed?
02400   2400 0  1,2411 1                TC      WL_RT_runIt     ; >0 no, just run it, as is
02401   2401 0  1,2403 1                TC      *+2             ; +0 yes
02402   2402 0  1,2403 1                TC      *+1             ; <0 yes

02403   2403 3  1,2050 0                CAF     ZERO
02404   2404 6  0,0114 0                AD      WL_RT_runAddr
02405   2405 5  0,0015 0                TS      BANK            ; set the bank

02406   2406 7  1,2103 0                MASK    lowAddr         ; get lowest 10-bits of address
02407   2407 6  1,2102 0                AD      bankAddr        ; set bits 11,12 for fixed-switchable
02410   2410 5  0,0114 0                TS      WL_RT_runAddr

                        WL_RT_runIt     EQU     *
02411   2411 2  0,0114 1                INDEX   WL_RT_runAddr   ; apply indirect address to next instr.
02412   2412 0  0,0000 1                TC      0               ; run the task

                        TASKOVER        EQU     *               ; task returns here
02413   2413 0  1,2364 1                TC      WL_RT_loop      ; check next task on list

                        WL_RT_done      EQU     *
02414   2414 3  0,0113 1                XCH     WL_RT_saveQ
02415   2415 5  0,0001 0                TS      Q               ; restore return address
02416   2416 0  0,0000 0                RETURN

                        ;-----------------------------------------------------------------------
                        ; WL_schedTask - SCHEDULE NEXT TASK
                        ;
                        ; Schedule task on the front of list for the next time-out. Adjust the
                        ; time-out for all other tasks on the list, so they contain the remaining
                        ; time after the next timeout.
                        ;-----------------------------------------------------------------------

                        WL_schedTask    EQU     *
02417   2417 3  0,0001 0                XCH     Q
02420   2420 5  0,0107 1                TS      WL_ST_saveQ     ; save return address

02421   2421 3  1,2050 0                CAF     ZERO
02422   2422 2  1,2175 1                INDEX   WL_tskLstStart
02423   2423 6  0,0001 0                AD      TSKADDR
02424   2424 1  0,0000 0                CCS     A               ; task scheduled?
02425   2425 0  1,2427 1                TC      *+2             ; >0 yes
02426   2426 0  1,2466 1                TC      WL_ST_noTask    ; +0 no, so we are done

02427   2427 3  1,2050 0                CAF     ZERO
02430   2430 2  1,2175 1                INDEX   WL_tskLstStart
02431   2431 6  0,0000 1                AD      TSKTIME
02432   2432 5  0,0111 0                TS      WL_ST_newTime   ; save the new task's time-out

                        ; Iterate through all tasks on the list. Subtract the time-out time
                        ; from each task. (The 1st task on the list will now have a time-out
                        ; of zero)

02433   2433 3  1,2175 0                CAF     WL_tskLstStart ; set pointer to front of list
02434   2434 5  0,0110 1                TS      WL_ST_taskPtr

02435   2435 3  1,2177 1                CAF     WL_numTasks     ; loop for number of tasks
                        WL_ST_loop      EQU     *
02436   2436 5  0,0112 0                TS      WL_ST_loopCnt

02437   2437 3  1,2050 0                CAF     ZERO
02440   2440 2  0,0110 0                INDEX   WL_ST_taskPtr
02441   2441 6  0,0001 0                AD      TSKADDR
```

```
02442    2442 1  0,0000 0               CCS      A              ; end of list?
02443    2443 0  1,2445 0               TC       *+2            ; >0 no, so keep going
02444    2444 0  1,2461 0               TC       WL_ST_setT3    ; +0 yes, set TIME3

02445    2445 3  1,2050 0               CAF      ZERO
02446    2446 2  0,0110 0               INDEX    WL_ST_taskPtr
02447    2447 6  0,0000 1               AD       TSKTIME
02450    2450 2  0,0000 1               EXTEND
02451    2451 6  0,0111 0               SU       WL_ST_newTime  ; time-out = time-out - newtime
02452    2452 2  0,0110 0               INDEX    WL_ST_taskPtr
02453    2453 5  0,0000 1               TS       TSKTIME

02454    2454 3  0,0110 1               XCH      WL_ST_taskPtr  ; bump task pointer back 1 record
02455    2455 6  1,2174 1               AD       WL_taskRecSize
02456    2456 5  0,0110 1               TS       WL_ST_taskPtr

02457    2457 1  0,0112 1               CCS      WL_ST_loopCnt  ; done fixing the times?
02460    2460 0  1,2436 1               TC       WL_ST_loop     ; not yet

                         ; Set TIME3 to overflow at the time-out of the task on the front
                         ; of the list: TIME3 = %37777 - WL_ST_newTime + 1

                         WL_ST_setT3    EQU      *
02461    2461 4  0,0111 1               CS       WL_ST_newTime
02462    2462 6  1,2201 0               AD       WL_maxVal
02463    2463 6  1,2051 1               AD       ONE
02464    2464 5  0,0037 0               TS       TIME3          ; overflow at new time-out time
02465    2465 0  1,2470 0               TC       WL_ST_done

                         WL_ST_noTask   EQU      *
02466    2466 3  1,2203 1               CAF      WL_maxTimeOut
02467    2467 5  0,0037 0               TS       TIME3          ; nothing scheduled, reset the clock

                         WL_ST_done     EQU      *
02470    2470 3  0,0107 1               XCH      WL_ST_saveQ
02471    2471 5  0,0001 0               TS       Q              ; restore return address
02472    2472 0  0,0000 0               RETURN

                         ;--------------------------------------------------------------------------
                         ; WL_insert - INSERT TASK INTO SORTED LIST
                         ;
                         ; Insert a task record into the sorted list. Use 'WL_IS_newTime' and
                         ; 'WL_IS_newAddr' to set the fields of record to be inserted.
                         ; Performs an insertion sort, with the records sorted by time.
                         ; Lowest times are at the front of the list. If several records
                         ; have the same time, the records inserted first will appear first
                         ; in the list. NIL records have a time of NOTASK and a address
                         ; of positive zero.
                         ;--------------------------------------------------------------------------

                         WL_insert      EQU      *
02473    2473 3  0,0001 0               XCH      Q
02474    2474 5  0,0124 0               TS       WL_IS_saveQ    ; save return address

02475    2475 3  1,2176 0               CAF      WL_tskLstEnd   ; set pointer to back of list
02476    2476 5  0,0125 1               TS       WL_IS_taskPtr

02477    2477 2  0,0000 1               EXTEND
02500    2500 6  1,2174 1               SU       WL_taskRecSize ; set pointer to rec in front of it
02501    2501 5  0,0126 1               TS       WL_IS_taskPtr2

02502    2502 3  1,2050 0               CAF      ZERO
02503    2503 2  0,0125 0               INDEX    WL_IS_taskPtr
02504    2504 6  0,0001 0               AD       TSKADDR
02505    2505 1  0,0000 0               CCS      A              ; list full?
02506    2506 0  1,2562 1               TC       WL_IS_done     ; >0 yes

                         ; Work from the back of the list to the front, pushing each record
                         ; to the back until the insertion point is found.

02507    2507 3  1,2200 1               CAF      WL_numTasks1   ; loop for number of tasks minus 1
                         WL_IS_loop     EQU      *
02510    2510 5  0,0127 0               TS       WL_IS_loopCnt

02511    2511 3  1,2050 0               CAF      ZERO
02512    2512 2  0,0126 0               INDEX    WL_IS_taskPtr2
02513    2513 6  0,0001 0               AD       TSKADDR
02514    2514 1  0,0000 0               CCS      A              ; previous record is NIL?
02515    2515 0  1,2517 0               TC       *+2            ; no, so check it
02516    2516 0  1,2541 0               TC       WL_IS_bumpPtr  ; yes, so skip to next record


                         ; Is this the insertion point?

02517    2517 4  0,0122 1               CS       WL_IS_newTime
02520    2520 2  0,0126 0               INDEX    WL_IS_taskPtr2
02521    2521 6  0,0000 1               AD       TSKTIME
```

```
02522    2522 1  0,0000 0               CCS      A              ; found insertion point?
02523    2523 0  1,2527 0               TC       *+4            ; >0 no, keep checking
02524    2524 0  1,2552 1               TC       WL_IS_insRec   ; +0 yes
02525    2525 0  1,2552 1               TC       WL_IS_insRec   ; <0 yes
02526    2526 0  1,2552 1               TC       WL_IS_insRec   ; -0 yes

                        ; No, bump the record toward the back of the list.

02527    2527 3  1,2050 0               CAF      ZERO
02530    2530 2  0,0126 0               INDEX    WL_IS_taskPtr2
02531    2531 6  0,0000 1               AD       TSKTIME
02532    2532 2  0,0125 0               INDEX    WL_IS_taskPtr
02533    2533 5  0,0000 1               TS       TSKTIME        ; copy time field

02534    2534 3  1,2050 0               CAF      ZERO
02535    2535 2  0,0126 0               INDEX    WL_IS_taskPtr2
02536    2536 6  0,0001 0               AD       TSKADDR
02537    2537 2  0,0125 0               INDEX    WL_IS_taskPtr
02540    2540 5  0,0001 0               TS       TSKADDR        ; copy address field

                        WL_IS_bumpPtr  EQU      *
02541    2541 3  0,0125 1               XCH      WL_IS_taskPtr  ; bump task pointer forward 1 record
02542    2542 2  0,0000 1               EXTEND
02543    2543 6  1,2174 1               SU       WL_taskRecSize
02544    2544 5  0,0125 1               TS       WL_IS_taskPtr

02545    2545 2  0,0000 1               EXTEND
02546    2546 6  1,2174 1               SU       WL_taskRecSize ; set pointer to record in front of it
02547    2547 5  0,0126 1               TS       WL_IS_taskPtr2

02550    2550 1  0,0127 1               CCS      WL_IS_loopCnt  ; done bumping tasks backward?
02551    2551 0  1,2510 1               TC       WL_IS_loop     ; not yet

                        ; Insert new record.

                        WL_IS_insRec   EQU      *
02552    2552 3  1,2050 0               CAF      ZERO
02553    2553 6  0,0122 0               AD       WL_IS_newTime
02554    2554 2  0,0125 0               INDEX    WL_IS_taskPtr
02555    2555 5  0,0000 1               TS       TSKTIME        ; set time field

02556    2556 3  1,2050 0               CAF      ZERO
02557    2557 6  0,0123 1               AD       WL_IS_newAddr
02560    2560 2  0,0125 0               INDEX    WL_IS_taskPtr
02561    2561 5  0,0001 0               TS       TSKADDR        ; set address field

                        WL_IS_done     EQU      *
02562    2562 3  0,0124 0               XCH      WL_IS_saveQ
02563    2563 5  0,0001 0               TS       Q              ; restore return address
02564    2564 0  0,0000 0               RETURN

                        ;------------------------------------------------------------------------
                        ; WL_remove - REMOVE TASK FROM FRONT OF LIST
                        ;
                        ; Returns the address of the task in register A. If the list is
                        ; empty, it returns zero in A. If a task is removed from the list,
                        ; the remaining tasks are moved up to the front.
                        ;------------------------------------------------------------------------

                        WL_remove      EQU      *
02565    2565 3  0,0001 0               XCH      Q
02566    2566 5  0,0115 1               TS       WL_RM_saveQ    ; save return address

02567    2567 3  1,2175 0               CAF      WL_tskLstStart ; set pointer to front of list
02570    2570 5  0,0116 1               TS       WL_RM_taskPtr

02571    2571 6  1,2174 1               AD       WL_taskRecSize ; set pointer to next rec behind it
02572    2572 5  0,0117 0               TS       WL_RM_taskPtr2

                        ; Save the address of record at the front of the list.

02573    2573 3  1,2050 0               CAF      ZERO
02574    2574 2  0,0116 0               INDEX    WL_RM_taskPtr
02575    2575 6  0,0001 0               AD       TSKADDR
02576    2576 5  0,0121 0               TS       WL_RM_retval   ; get address of 1st task

02577    2577 1  0,0000 0               CCS      A              ; list empty?
02600    2600 0  1,2602 1               TC       *+2            ; >0, no
02601    2601 0  1,2636 0               TC       WL_RM_done     ; +0, yes, so exit

                        ; Loop through the remaining records in the task list and
                        ; bubble them up to the front.

02602    2602 3  1,2200 1               CAF      WL_numTasks1   ; loop for number of tasks minus 1
                        WL_RM_loop     EQU      *
02603    2603 5  0,0120 1               TS       WL_RM_loopCnt
```

```
02604    2604 3  1,2050 0               CAF      ZERO
02605    2605 2  0,0117 1               INDEX    WL_RM_taskPtr2
02606    2606 6  0,0000 1               AD       TSKTIME
02607    2607 2  0,0116 0               INDEX    WL_RM_taskPtr
02610    2610 5  0,0000 1               TS       TSKTIME        ; copy time field

02611    2611 3  1,2050 0               CAF      ZERO
02612    2612 2  0,0117 1               INDEX    WL_RM_taskPtr2
02613    2613 6  0,0001 0               AD       TSKADDR
02614    2614 2  0,0116 0               INDEX    WL_RM_taskPtr
02615    2615 5  0,0001 0               TS       TSKADDR        ; copy address field

02616    2616 1  0,0000 0               CCS      A              ; remainder of list empty?
02617    2617 0  1,2621 0               TC       *+2            ; >0, no
02620    2620 0  1,2636 0               TC       WL_RM_done     ; +0, yes, so exit

02621    2621 3  0,0116 1               XCH      WL_RM_taskPtr  ; bump task pointer back 1 record
02622    2622 6  1,2174 1               AD       WL_taskRecSize
02623    2623 5  0,0116 1               TS       WL_RM_taskPtr

02624    2624 6  1,2174 1               AD       WL_taskRecSize ; set pointer to record behind it
02625    2625 5  0,0117 0               TS       WL_RM_taskPtr2

02626    2626 1  0,0120 0               CCS      WL_RM_loopCnt  ; done bumping tasks upward?
02627    2627 0  1,2603 0               TC       WL_RM_loop     ; not yet

                         ; Since we removed a record, the last record on the list
                         ; should be NIL.

02630    2630 3  1,2202 0               CAF      WL_maxDelay
02631    2631 2  0,0116 0               INDEX    WL_RM_taskPtr
02632    2632 5  0,0000 1               TS       TSKTIME        ; set time field to NIL

02633    2633 3  1,2050 0               CAF      ZERO
02634    2634 2  0,0116 0               INDEX    WL_RM_taskPtr
02635    2635 5  0,0001 0               TS       TSKADDR        ; set address field to NIL

                       WL_RM_done      EQU      *
02636    2636 3  0,0115 1               XCH      WL_RM_saveQ
02637    2637 5  0,0001 0               TS       Q              ; restore return address
02640    2640 3  0,0121 0               XCH      WL_RM_retval   ; return task address in A
02641    2641 0  0,0000 0               RETURN
                                        INCL     exec_f.asm     ; EXEC
                         ;==============================================================================
                         ; EXEC (file:exec_f.asm)
                         ;
                         ; Version:  1.0
                         ; Author:   John Pultorak
                         ; Date:     04/26/2002
                         ;
                         ; PURPOSE:
                         ; Constants and source code for EXEC.
                         ;
                         ; Non-preemptive multitasking routines, originally implemented by J. H.
                         ; Laning, Jr. for AGC3 and later adapted for AGC4. Briefly discussed in
                         ; R-393, which gives some of the software interfaces into the
                         ; multitasking. This is my own recreation, and it only includes the job
                         ; scheduling. The original EXEC also includes memory management for the
                         ; eraseable memory; this is not reproduced here.
                         ;
                         ; Overview: scheduled elements are called 'jobs'. Up to 7 jobs can be
                         ; concurrently scheduled. An 8th 'dummy' job is always scheduled. Each
                         ; job has an assigned priority (1-n, where 1 is the lowest priority).
                         ; The highest priority job always executes. When that job terminates,
                         ; the next highest priority job is selected for execution. If several
                         ; jobs have the same priority, they are executed round-robin.
                         ;
                         ; A job is scheduled for execution by calling 'NOVAC' and
                         ; furnishing the job priority and starting address.
                         ;       L       XCH     JOB_PRIORITY
                         ;       L+1     TC      NOVAC
                         ;       L+2     DS      JOB_ADDRESS
                         ;       L+3     ... execution resumes here
                         ;
                         ; JOB_PRIORITY = a positive integer from %3 - %37776 where a higher number
                         ;    indicates higher priority. Priorities below 3 are reserved for
                         ;    internal EXEC use: 0=no job, 1=sleeping job, 2=dummy job.
                         ;    Priority %37777 is also reserved for woken jobs.
                         ; JOB_ADDRESS = starting address of the job.
                         ;
                         ; **** WARNING **** If NOVAC is not being called from an interrupt, be sure to
                         ; inhibit interrupts before calling it to protect the integrity of the list.
                         ;
                         ; When a new job is added, the new job's record (core set) is
                         ; initialized with a copy of the current job's record (MPAC and other
                         ; parameters), except for the new job priority and address, which are
                         ; set by the 'add job' routine. Therefore, data can be stored into
```

```
                        ; MPAC prior to starting a new job as a method of passing data into
                        ; the new job.
                        ;
                        ; Jobs terminate themselves by jumping to ENDOFJOB. This removes them
                        ; from the EXEC scheduler:
                        ;       TC      ENDOFJOB
                        ;
                        ; Jobs can suspend themselves (yield to a higher priority job) by
                        ; executing the following sequence. If there is no other job of
                        ; higher priority, executing of the yielded job resumes at L+2
                        ;       L       CCS     newJob
                        ;       L+1     TC      CHANG1
                        ;       L+2     ... execution resumes here
                        ;
                        ; If there is no other job of equal or higher priority, the branch is
                        ; not taken.
                        ;
                        ; Jobs can put themselves to sleep by calling JOBSLEEP. The address
                        ; where execution of the sleeping job should resume must be in register
                        ; A before calling JOBSLEEP. The job will remain sleeping until JOBWAKE
                        ; is called:
                        ;
                        ;       L       CAF     WAKECADR
                        ;       L+1     TC      JOBSLEEP
                        ;       (does not return from JOBSLEEP)
                        ;
                        ; Sleeping jobs are awakened by calling JOBWAKE. The address where
                        ; execution of the sleeping job should resume must be in register A.
                        ; JOBWAKE returns to the address after the call and execution continues
                        ; for the calling job. The job that was sleeping will now be the next
                        ; job to execute.
                        ;
                        ;       L       CAF     WAKECADR
                        ;       L+1     TC      JOBWAKE
                        ;       L+2     ... execution continues here
                        ;
                        ;==============================================================================

02642   2642    37777 1 EX_WAKE_PRIO    DS      %37777          ; waking job priority (highest)
02643   2643    00002 0 EX_DUMMY_PRIO   DS      %00002          ; dummy job priority (lowest runnable)
02644   2644    00001 0 EX_SLEEP_PRIO   DS      %00001          ; sleeping job; must be < dummy

02645   2645    00130 0 EX_jobCurStart DS       EX_currentJob   ; starting address for current job

02646   2646    00015 0 EX_jobRecSize   DS      JRECSZ          ; size of a job record (words)
02647   2647    00300 1 EX_jobLstStart DS       EX_jobList      ; starting address for jobList
02650   2650    00307 0 EX_jobLstEnd    DS      MAXJOBS+EX_jobList
02651   2651    00306 1 EX_jobLstEnd1   DS      MAXJOBS-1+EX_jobList
02652   2652    00006 1 EX_numJobs      DS      MAXJOBS-1       ; init loop counter for all jobs
02653   2653    00005 1 EX_numJobs1     DS      MAXJOBS-2       ; init loop counter for all jobs - 1

                        ; enumerated types for setting change flag:
02654   2654    00001 0 EX_changeJob    DS      CHGJOB          ; change job
02655   2655    00000 1 EX_keepJob      DS      KEEPJOB         ; keep job


                        ;------------------------------------------------------------------------
                        ; EX_exec -- EXEC SCHEDULER
                        ;
                        ; Executes the highest priority job. Enables interrupts while the job is
                        ; running. Once called, this function never returns.
                        ;------------------------------------------------------------------------

                        EX_exec         EQU     *               ; entry point

                        ; Add a dummy job (lowest priority) that never terminates.

02656   2656 3  1,2643 1                CAF     EX_DUMMY_PRIO   ; job priority
02657   2657 0  1,3162 1                TC      NOVAC
02660   2660     03510 0                DS      dumJob          ; 14 bit job address
02661   2661 2  0,0000 0                INHINT                  ; inhibit RUPTs enab by addJob

                        ; Get the next job to run.

                        EX_MN_findJob   EQU     *
02662   2662 0  1,3410 1                TCR     EX_remove

                        ; compare priority of current job to priority of next waiting job.
                        ; If next job has same priority as current job, set the newJob
                        ; flag so they will be scheduled round-robin.

02663   2663 4  0,0143 0                CS      PRIORITY        ; get priority of current job

02664   2664 2  1,2647 1                INDEX   EX_jobLstStart
02665   2665 2  0,0000 0                INDEX   0
02666   2666 6  0,0143 1                AD      PRIORITY        ; compare with priority of next job
```

```
02667    2667 1  0,0000 0                CCS      A              ; next job has equal priority?
02670    2670 0  1,2677 0                TC       EX_MN_setFlg   ; >0 (error!)
02671    2671 0  1,2677 0                TC       EX_MN_setFlg   ; +0 yes, set flag
02672    2672 0  1,2674 0                TC       *+2            ; <0 no,  clear flag
02673    2673 0  1,2677 0                TC       EX_MN_setFlg   ; -0 yes, set flag

02674    2674 3  1,2655 0                CAF      EX_keepJob     ; clear change flag
02675    2675 5  0,0307 0                TS       newJob
02676    2676 0  1,2701 0                TC       EX_MN_runJob

                         EX_MN_setFlg   EQU      *
02677    2677 3  1,2654 1                CAF      EX_changeJob   ; set change flag
02700    2700 5  0,0307 0                TS       newJob


                         ; Start the job. Interrupts are reenabled before 'EX_curJobPtr' is
                         ; referenced, but the interrupts can only call 'NOVAC' which does
                         ; not change 'EX_curJobPtr'.

                         ; The job address is always 14-bit, so check whether the address falls
                         ; within erasable or fixed-fixed memory. If so, use it as-is; otherwise,
                         ; set the bank register and change the address to 12-bit.

                         EX_MN_runJob   EQU      *
02701    2701 3  1,2050 0                CAF      ZERO
02702    2702 6  0,0140 1                AD       LOC
02703    2703 5  0,0333 1                TS       EX_MN_runAddr  ; save job's 14 bit address

02704    2704 4  0,0000 0                COM
02705    2705 6  1,2102 0                AD       bankAddr       ; -(14bitAddr)+%6000
02706    2706 1  0,0000 0                CCS      A              ; job is bank addressed?
02707    2707 0  1,2720 0                TC       EX_MN_runIt    ; >0 no, just run it, as is
02710    2710 0  1,2712 1                TC       *+2            ; +0 yes
02711    2711 0  1,2712 1                TC       *+1            ; <0 yes

02712    2712 3  1,2050 0                CAF      ZERO
02713    2713 6  0,0333 1                AD       EX_MN_runAddr
02714    2714 5  0,0015 0                TS       BANK           ; set the bank

02715    2715 7  1,2103 0                MASK     lowAddr        ; get lowest 10-bits of address
02716    2716 6  1,2102 0                AD       bankAddr       ; set bits 11,12 for fixed-switchable
02717    2717 5  0,0333 1                TS       EX_MN_runAddr

                         EX_MN_runIt    EQU      *
02720    2720 2  0,0000 1                RELINT                  ; enable interrupts
02721    2721 2  0,0333 0                INDEX    EX_MN_runAddr  ; apply indirect address to next instr.
02722    2722 0  0,0000 1                TC       0              ; run the job

                         ; Job is terminated. Delete the job record.

                         ENDOFJOB       EQU      *
02723    2723 2  0,0000 0                INHINT                  ; inhibit interrupts
02724    2724 0  1,2662 1                TC       EX_MN_findJob  ; get next job

                         ; job is sleeping. Keep the job record, but drop the priority so it
                         ; is below the priority of the dummy job. This will keep the job
                         ; from running until JOBWAKE is called. The address where it should
                         ; resume running when awoken is in register A.

                         JOBSLEEP       EQU      *
02725    2725 2  0,0000 0                INHINT                  ; inhibit interrupts
02726    2726 5  0,0140 1                TS       LOC            ; save restart address

02727    2727 3  1,2644 0                CAF      EX_SLEEP_PRIO
02730    2730 5  0,0143 1                TS       PRIORITY       ; set sleeping priority
02731    2731 5  0,0346 0                TS       EX_IS_newPrio

02732    2732 0  1,2757 0                TC       EX_MN_mvRec    ; finish up


                         ; Job is suspended. Keep the job record, but update the address, so
                         ; execution will resume at the point after suspension.

                         CHANG1         EQU      *
02733    2733 2  0,0000 0                INHINT                  ; inhibit interrupts
02734    2734 3  0,0001 0                XCH      Q
02735    2735 5  0,0333 1                TS       EX_MN_runAddr  ; save job's 12 bit restart address

02736    2736 4  0,0000 0                COM
02737    2737 6  1,2102 0                AD       bankAddr       ; -(12bitAddr)+%6000
02740    2740 1  0,0000 0                CCS      A              ; job is bank addressed?
02741    2741 0  1,2750 1                TC       EX_MN_notBank  ; >0 no, just save it, as is
02742    2742 0  1,2744 1                TC       *+2            ; +0 yes
02743    2743 0  1,2744 1                TC       *+1            ; <0 yes

02744    2744 4  1,2102 1                CS       bankAddr       ; 12bitAddr - %6000
02745    2745 6  0,0333 1                AD       EX_MN_runAddr
```

```
02746    2746 6  0,0015 0               AD       BANK           ; make it a 14-bit address
02747    2747 0  1,2752 0               TC       EX_MN_saveIt

                         EX_MN_notBank  EQU      *
02750    2750 3  1,2050 0               CAF      ZERO
02751    2751 6  0,0333 1               AD       EX_MN_runAddr  ; get restart address

                         EX_MN_saveIt   EQU      *
02752    2752 5  0,0140 1               TS       LOC            ; save job's new starting address

02753    2753 3  1,2050 0               CAF      ZERO
02754    2754 6  0,0144 0               AD       JOBPRIOBASE
02755    2755 5  0,0143 1               TS       PRIORITY
02756    2756 5  0,0346 0               TS       EX_IS_newPrio  ; restore job priority to nominal value

                         ; given the priority, find the insertion point in the list. Copy
                         ; the current job into the list at the correct insertion point.

                         EX_MN_mvRec    EQU      *
02757    2757 0  1,3332 0               TCR      EX_findIns
02760    2760 5  0,0352 0               TS       EX_IS_jobPtr   ; save address of insertion point

                         ; copy all fields in current record to list

02761    2761 3  1,2646 1               XCH      EX_jobRecSize
02762    2762 5  0,0334 0               TS       EX_MN_field

                         EX_MN_loop3    EQU      *
02763    2763 1  0,0334 1               CCS      EX_MN_field    ; done?
02764    2764 0  1,2766 1               TC       *+2            ; not yet
02765    2765 0  1,3002 0               TC       EX_MN_done3    ; yes
02766    2766 5  0,0334 0               TS       EX_MN_field

                         ; copy this field to list

02767    2767 3  1,2050 0               CAF      ZERO
02770    2770 2  0,0352 1               INDEX    EX_IS_jobPtr
02771    2771 6  0,0000 1               AD       0              ; get index to record in list
02772    2772 6  0,0334 0               AD       EX_MN_field    ; add field displacement
02773    2773 5  0,0335 1               TS       EX_MN_findx    ; save index to field in list

02774    2774 3  1,2050 0               CAF      ZERO
02775    2775 2  0,0334 1               INDEX    EX_MN_field
02776    2776 6  0,0130 0               AD       EX_currentJob  ; get field from current job
02777    2777 2  0,0335 0               INDEX    EX_MN_findx
03000    3000 5  0,0130 0               TS       EX_currentJob  ; copy field to list

03001    3001 0  1,2763 1               TC       EX_MN_loop3

                         EX_MN_done3    EQU      *
03002    3002 0  1,2662 1               TC       EX_MN_findJob  ; get next job


                         ;------------------------------------------------------------------------
                         ; JOBWAKE - wake up the job identified by address in register A
                         ;
                         ; Search jobList for a job with address matching the address in A.
                         ; If found, bump the priority up to the highest level, so the job
                         ; will be the next to run.
                         ;
                         ; This is a 'public' function. It assumes that interrupts are already
                         ; disabled before it is called. Disabling interrupts during JOBWAKE
                         ; is necessary to preserve the integrity of the joblist.
                         ;------------------------------------------------------------------------

                         JOBWAKE        EQU      *
03003    3003 5  0,0312 1               TS       EX_JW_CADR     ; save job address
03004    3004 3  0,0001 0               XCH      Q
03005    3005 5  0,0310 0               TS       EX_JW_saveQ    ; save return address

                         ; Search the joblist for the job to wake (job address matches
                         ; EX_JW_CADR).

03006    3006 3  1,2050 0               CAF      ZERO
03007    3007 5  0,0313 0               TS       EX_JW_foundit  ; clear 'found it' flag

03010    3010 3  1,2651 1               CAF      EX_jobLstEnd1  ; set pointer to back of list
03011    3011 5  0,0314 1               TS       EX_JW_jobPtr

03012    3012 6  1,2046 1               AD       NEG1           ; set pointer to rec in front of it
03013    3013 5  0,0315 0               TS       EX_JW_jobPtr2

03014    3014 3  1,2653 0               CAF      EX_numJobs1    ; loop for number of jobs minus 1
                         EX_JW_loop     EQU      *
03015    3015 5  0,0311 1               TS       EX_JW_loopCnt

                         ; if foundit=0, job has not been found yet. Keep searching toward
```

```
                        ;   the front of the list.
                        ; if foundit=1, the job has been found and removed from the list.
                        ;   push all jobs in front of the removed job one step to the back
                        ;   to fill in the gap and to make room at the front of the list
                        ;   for the awoken job.

03016    3016 1  0,0313 1               CCS      EX_JW_foundit  ; already found job to wake?
03017    3017 0  1,3035 1               TC       EX_JW_moveRec  ; >0, yes

                        ; Is this the job?

03020    3020 4  0,0312 0               CS       EX_JW_CADR
03021    3021 2  0,0314 0               INDEX    EX_JW_jobPtr
03022    3022 2  0,0000 0               INDEX    0
03023    3023 6  0,0140 1               AD       LOC
03024    3024 1  0,0000 0               CCS      A              ; found job to wake?
03025    3025 0  1,3041 1               TC       EX_JW_bumpPtr  ; >0, no
03026    3026 0  1,3030 1               TC       *+2            ; +0, yes
03027    3027 0  1,3041 1               TC       EX_JW_bumpPtr  ; <0, no

                        ; found the job to wake.

03030    3030 3  1,2051 1               CAF      ONE
03031    3031 5  0,0313 0               TS       EX_JW_foundit  ; set 'found it' flag

                        ; save record index for awoken job

03032    3032 2  0,0314 0               INDEX    EX_JW_jobPtr
03033    3033 3  0,0000 1               XCH      0
03034    3034 5  0,0316 0               TS       EX_JW_fndIndx  ; index for awoken job

                        ; bump prior record back

                        EX_JW_moveRec  EQU      *
03035    3035 2  0,0315 1               INDEX    EX_JW_jobPtr2
03036    3036 3  0,0000 1               XCH      0
03037    3037 2  0,0314 0               INDEX    EX_JW_jobPtr
03040    3040 3  0,0000 1               XCH      0

                        EX_JW_bumpPtr  EQU      *
03041    3041 3  0,0314 1               XCH      EX_JW_jobPtr   ; bump job pointer forward 1 record
03042    3042 6  1,2046 1               AD       NEG1
03043    3043 5  0,0314 1               TS       EX_JW_jobPtr

03044    3044 6  1,2046 1               AD       NEG1           ; set pointer to record in front of it
03045    3045 5  0,0315 0               TS       EX_JW_jobPtr2

03046    3046 1  0,0311 0               CCS      EX_JW_loopCnt  ; done bumping jobs backward?
03047    3047 0  1,3015 0               TC       EX_JW_loop     ; not yet

03050    3050 1  0,0313 1               CCS      EX_JW_foundit  ; found job to wake?
03051    3051 0  1,3053 1               TC       *+2            ; >0, yes
03052    3052 0  1,3056 1               TC       EX_JW_done     ; no

03053    3053 3  0,0316 0               XCH      EX_JW_fndIndx  ; put awoken job on front of list
03054    3054 2  1,2647 1               INDEX    EX_jobLstStart
03055    3055 5  0,0000 1               TS       0

                        EX_JW_done     EQU      *

                        ; Is the awoken job at the front of the list?
                        ; (If it was already there before we started searching, 'foundIt'
                        ; will be false (0) so we need to make this test).

03056    3056 4  0,0312 0               CS       EX_JW_CADR
03057    3057 2  1,2647 1               INDEX    EX_jobLstStart
03060    3060 2  0,0000 0               INDEX    0
03061    3061 6  0,0140 1               AD       LOC
03062    3062 1  0,0000 0               CCS      A              ; woken job at front of list?
03063    3063 0  1,3074 1               TC       EX_JW_return   ; >0, no
03064    3064 0  1,3066 1               TC       *+2            ; +0, yes
03065    3065 0  1,3074 1               TC       EX_JW_return   ; <0, no

                        ; set awoken priority and change job flag

03066    3066 3  1,2642 0               CAF      EX_WAKE_PRIO
03067    3067 2  1,2647 1               INDEX    EX_jobLstStart
03070    3070 2  0,0000 0               INDEX    0
03071    3071 5  0,0143 1               TS       PRIORITY       ; set waking priority

03072    3072 3  1,2654 1               CAF      EX_changeJob   ; set the change flag
03073    3073 5  0,0307 0               TS       newJob

                        EX_JW_return   EQU      *
03074    3074 0  0,0310 0               TC       EX_JW_saveQ    ; return
```

```
                        ;-----------------------------------------------------------------------------
                        ; SPVAC - ADD A JOB TO THE JOBLIST
                        ;
                        ; Similar to NOVAC, but used by VERB 37. The job CADR is in register A.
                        ; The job priority is in NEWPRIO. Return to the address in Q.
                        ;
                        ; NOVAC differs from SPVAC, because NOVAC has the job CADR at the address
                        ; in Q, and returns to Q+1. Also, in NOVAC the job priority is in A.
                        ;
                        ; This is a 'public' function. It can be called from a job
                        ; or from an interrupt.
                        ;-----------------------------------------------------------------------------

                        SPVAC           EQU     *
03075    3075 5  0,0350 1               TS      EX_IS_newLoc    ; store new job address
03076    3076 3  0,0001 0               XCH     Q
03077    3077 5  0,0317 1               TS      EX_AJ_saveQ     ; save return address


                        ; add new job to end of list

03100    3100 3  1,2050 0               CAF     ZERO
03101    3101 6  0,0360 1               AD      NEWPRIO
03102    3102 5  0,0346 0               TS      EX_IS_newPrio
03103    3103 5  0,0347 1               TS      EX_IS_newPrioB ; store new job priority

03104    3104 0  1,3332 0               TCR     EX_findIns      ; find insertion point in list
03105    3105 5  0,0352 0               TS      EX_IS_jobPtr    ; save address of insertion point

                        ; Initialize relevant fields in new job. The remaining fields
                        ; should already be zeroed.


                        ; Initialize fields for new job record. New job inherits copy of
                        ; MPAC from current job, so copy all fields in current job to new
                        ; job in list

03106    3106 3  1,2646 1               XCH     EX_jobRecSize
03107    3107 5  0,0323 0               TS      EX_AJ_field

                        EX_SP_loop1     EQU     *
03110    3110 1  0,0323 1               CCS     EX_AJ_field     ; done?
03111    3111 0  1,3113 1               TC      *+2             ; not yet
03112    3112 0  1,3127 0               TC      EX_SP_done1     ; yes
03113    3113 5  0,0323 0               TS      EX_AJ_field

                        ; copy this field to list

03114    3114 3  1,2050 0               CAF     ZERO
03115    3115 2  0,0352 1               INDEX   EX_IS_jobPtr
03116    3116 6  0,0000 1               AD      0               ; get index to record in list
03117    3117 6  0,0323 0               AD      EX_AJ_field     ; add field displacement
03120    3120 5  0,0324 1               TS      EX_AJ_findx     ; save index to field in list

03121    3121 3  1,2050 0               CAF     ZERO
03122    3122 2  0,0323 1               INDEX   EX_AJ_field
03123    3123 6  0,0130 0               AD      EX_currentJob   ; get field from current job
03124    3124 2  0,0324 0               INDEX   EX_AJ_findx
03125    3125 5  0,0130 0               TS      EX_currentJob   ; copy field to list

03126    3126 0  1,3110 1               TC      EX_SP_loop1

                        ; now, overwrite fields in the record with the priority
                        ; and location unique to this job.

                        EX_SP_done1     EQU     *
03127    3127 3  1,2050 0               CAF     ZERO
03130    3130 6  0,0346 0               AD      EX_IS_newPrio
03131    3131 2  0,0352 1               INDEX   EX_IS_jobPtr
03132    3132 2  0,0000 0               INDEX   0
03133    3133 5  0,0143 1               TS      PRIORITY        ; set priority field

03134    3134 3  1,2050 0               CAF     ZERO
03135    3135 6  0,0347 1               AD      EX_IS_newPrioB
03136    3136 2  0,0352 1               INDEX   EX_IS_jobPtr
03137    3137 2  0,0000 0               INDEX   0
03140    3140 5  0,0144 0               TS      JOBPRIOBASE     ; set nominal priority field

03141    3141 3  1,2050 0               CAF     ZERO
03142    3142 6  0,0350 1               AD      EX_IS_newLoc
03143    3143 2  0,0352 1               INDEX   EX_IS_jobPtr
03144    3144 2  0,0000 0               INDEX   0
03145    3145 5  0,0140 1               TS      LOC             ; set address field


                        ; Set changeflag if priority of new job >= priority of current job
```

```
                         EX_SP_testFlg  EQU      *
03146    3146 4  0,0143 0               CS       PRIORITY       ; get -priority of current job

03147    3147 6  0,0321 1               AD       EX_AJ_jobPrio  ; add positive priority of new job
03150    3150 1  0,0000 0               CCS      A              ; new job is highest priority?
03151    3151 0  1,3154 1               TC       *+3            ; >0, yes
03152    3152 0  1,3154 1               TC       *+2            ; +0, yes
03153    3153 0  1,3156 0               TC       EX_SP_done2    ; <0, no, current job is higher priority

03154    3154 3  1,2654 1               CAF      EX_changeJob   ; set the change flag
03155    3155 5  0,0307 0               TS       newJob

                         EX_SP_done2    EQU      *
03156    3156 3  0,0317 1               XCH      EX_AJ_saveQ
03157    3157 5  0,0001 0               TS       Q
03160    3160 0  0,0000 0               RETURN

                         ;------------------------------------------------------------------------
                         ; FINDVAC - not implemented
                         ;
                         ;------------------------------------------------------------------------

03161    3161 0  0,0001 0 FINDVAC        TC       Q              ; just return


                         ;------------------------------------------------------------------------
                         ; NOVAC - ADD A JOB TO THE JOBLIST
                         ;
                         ; Search jobList for an empty slot. If found, put the new job in the
                         ; empty slot. If the new job has the same, or higher, priority than the
                         ; current job, set the change flag to 'CHGJOB' (change jobs at the next
                         ; opportunity).
                         ;
                         ; This is a 'public' function. It can be called from a job
                         ; or from an interrupt.
                         ;------------------------------------------------------------------------

                         NOVAC          EQU      *
03162    3162 5  0,0321 1               TS       EX_AJ_jobPrio  ; save job priority
03163    3163 3  0,0001 0               XCH      Q
03164    3164 5  0,0317 1               TS       EX_AJ_saveQ    ; save return address-1


                         ; add new job to end of list

03165    3165 3  1,2050 0               CAF      ZERO
03166    3166 6  0,0321 1               AD       EX_AJ_jobPrio
03167    3167 5  0,0346 0               TS       EX_IS_newPrio
03170    3170 5  0,0347 1               TS       EX_IS_newPrioB ; store new job priority

03171    3171 2  0,0317 0               INDEX    EX_AJ_saveQ    ; indirectly address addJobQ
03172    3172 3  0,0000 1               CAF      0
03173    3173 5  0,0350 1               TS       EX_IS_newLoc   ; store new job address

03174    3174 0  1,3332 0               TCR      EX_findIns     ; find insertion point in list
03175    3175 5  0,0352 0               TS       EX_IS_jobPtr   ; save address of insertion point

                         ; Initialize relevant fields in new job. The remaining fields
                         ; should already be zeroed.



                         ; Initialize fields for new job record. New job inherits copy of
                         ; MPAC from current job, so copy all fields in current job to new
                         ; job in list

03176    3176 3  1,2646 1               XCH      EX_jobRecSize
03177    3177 5  0,0323 0               TS       EX_AJ_field

                         EX_AJ_loop1    EQU      *
03200    3200 1  0,0323 1               CCS      EX_AJ_field    ; done?
03201    3201 0  1,3203 0               TC       *+2            ; not yet
03202    3202 0  1,3217 0               TC       EX_AJ_done1    ; yes
03203    3203 5  0,0323 0               TS       EX_AJ_field

                         ; copy this field to list

03204    3204 3  1,2050 0               CAF      ZERO
03205    3205 2  0,0352 1               INDEX    EX_IS_jobPtr
03206    3206 6  0,0000 1               AD       0              ; get index to record in list
03207    3207 6  0,0323 0               AD       EX_AJ_field    ; add field displacement
03210    3210 5  0,0324 1               TS       EX_AJ_findx    ; save index to field in list

03211    3211 3  1,2050 0               CAF      ZERO
03212    3212 2  0,0323 1               INDEX    EX_AJ_field
03213    3213 6  0,0130 0               AD       EX_currentJob  ; get field from current job
03214    3214 2  0,0324 0               INDEX    EX_AJ_findx
```

```
03215    3215 5  0,0130 0               TS       EX_currentJob  ; copy field to list

03216    3216 0  1,3200 0               TC       EX_AJ_loop1

                         ; now, overwrite fields in the record with the priority
                         ; and location unique to this job.

                         EX_AJ_done1    EQU      *
03217    3217 3  1,2050 0               CAF      ZERO
03220    3220 6  0,0346 0               AD       EX_IS_newPrio
03221    3221 2  0,0352 1               INDEX    EX_IS_jobPtr
03222    3222 2  0,0000 0               INDEX    0
03223    3223 5  0,0143 1               TS       PRIORITY       ; set priority field

03224    3224 3  1,2050 0               CAF      ZERO
03225    3225 6  0,0347 1               AD       EX_IS_newPrioB
03226    3226 2  0,0352 1               INDEX    EX_IS_jobPtr
03227    3227 2  0,0000 0               INDEX    0
03230    3230 5  0,0144 0               TS       JOBPRIOBASE    ; set nominal priority field

03231    3231 3  1,2050 0               CAF      ZERO
03232    3232 6  0,0350 1               AD       EX_IS_newLoc
03233    3233 2  0,0352 1               INDEX    EX_IS_jobPtr
03234    3234 2  0,0000 0               INDEX    0
03235    3235 5  0,0140 1               TS       LOC            ; set address field


                         ; Set changeflag if priority of new job >= priority of current job

                         EX_AJ_testFlg  EQU      *
03236    3236 4  0,0143 0               CS       PRIORITY       ; get -priority of current job

03237    3237 6  0,0321 1               AD       EX_AJ_jobPrio  ; add positive priority of new job
03240    3240 1  0,0000 0               CCS      A              ; new job is highest priority?
03241    3241 0  1,3244 0               TC       *+3            ; >0, yes
03242    3242 0  1,3244 0               TC       *+2            ; +0, yes
03243    3243 0  1,3246 1               TC       EX_AJ_done2    ; <0, no, current job is higher priority

03244    3244 3  1,2654 1               CAF      EX_changeJob   ; set the change flag
03245    3245 5  0,0307 0               TS       newJob

                         EX_AJ_done2    EQU      *
03246    3246 3  0,0317 1               XCH      EX_AJ_saveQ
03247    3247 6  1,2051 1               AD       ONE
03250    3250 5  0,0001 0               TS       Q
03251    3251 0  0,0000 0               RETURN


                         ;-----------------------------------------------------------------------
                         ; EX_initEX - INITIALIZE EXEC
                         ;
                         ; Initialize the eraseable memory segment for EXEC. Necessary in
                         ; case the AGC is restarted.
                         ;-----------------------------------------------------------------------

                         EX_initEX      EQU      *
03252    3252 3  0,0001 0               XCH      Q
03253    3253 5  0,0325 0               TS       EX_IN_saveQ    ; save return address

03254    3254 3  1,2655 0               CAF      EX_keepJob     ; clear change flag
03255    3255 5  0,0307 0               TS       newJob

03256    3256 3  1,2050 0               CAF      ZERO
03257    3257 5  0,0143 1               TS       PRIORITY       ; set current job record to NIL

                         ; Iterate through jobList, initialize each element on the list so it
                         ; points to its own job record.

03260    3260 3  1,2647 0               CAF      EX_jobLstStart ; init pointer to start of list
03261    3261 5  0,0327 1               TS       EX_IN_jobPtr

03262    3262 3  1,2050 0               CAF      ZERO
03263    3263 6  1,2646 1               AD       EX_jobRecSize
03264    3264 5  0,0330 1               TS       EX_IN_recIndex

03265    3265 3  1,2652 1               CAF      EX_numJobs     ; loop for number of jobs
                         EX_IN_loop1    EQU      *
03266    3266 5  0,0326 0               TS       EX_IN_loopCnt

03267    3267 3  0,0330 1               XCH      EX_IN_recIndex
03270    3270 2  0,0327 0               INDEX    EX_IN_jobPtr
03271    3271 5  0,0000 1               TS       0              ; initialize record index
03272    3272 6  1,2646 1               AD       EX_jobRecSize
03273    3273 5  0,0330 1               TS       EX_IN_recIndex ; bump index to next record

03274    3274 3  0,0327 1               XCH      EX_IN_jobPtr   ; bump job pointer back 1 record
03275    3275 6  1,2051 1               AD       ONE
```

```
03276    3276 5  0,0327 1               TS       EX_IN_jobPtr

03277    3277 1  0,0326 1               CCS      EX_IN_loopCnt  ; done clearing jobList?
03300    3300 0  1,3266 0               TC       EX_IN_loop1    ; not yet


                         ; Iterate through job records, initialize each field to zero.

03301    3301 3  1,2647 0               CAF      EX_jobLstStart ; init pointer to start of list
03302    3302 5  0,0327 1               TS       EX_IN_jobPtr

03303    3303 3  1,2652 1               CAF      EX_numJobs     ; loop for number of jobs
                         EX_IN_loop2    EQU      *
03304    3304 5  0,0326 0               TS       EX_IN_loopCnt

                         ; loop for number of fields in each record

03305    3305 3  1,2646 1               XCH      EX_jobRecSize
03306    3306 5  0,0331 0               TS       EX_IN_field

                         EX_IN_loop3    EQU      *
03307    3307 1  0,0331 1               CCS      EX_IN_field    ; done?
03310    3310 0  1,3312 1               TC       *+2            ; not yet
03311    3311 0  1,3324 1               TC       EX_IN_done     ; yes
03312    3312 5  0,0331 0               TS       EX_IN_field

                         ; set the field to zero

03313    3313 3  1,2050 0               CAF      ZERO
03314    3314 2  0,0327 0               INDEX    EX_IN_jobPtr
03315    3315 6  0,0000 1               AD       0              ; get index to record
03316    3316 6  0,0331 0               AD       EX_IN_field    ; add field displacement
03317    3317 5  0,0332 0               TS       EX_IN_findx    ; save index to field
03320    3320 3  1,2050 0               CAF      ZERO
03321    3321 2  0,0332 1               INDEX    EX_IN_findx
03322    3322 5  0,0130 0               TS       EX_currentJob  ; clear field

03323    3323 0  1,3307 0               TC       EX_IN_loop3

                         ; done clearing all fields in record, so do next record

                         EX_IN_done     EQU      *
03324    3324 3  0,0327 1               XCH      EX_IN_jobPtr   ; bump job pointer back 1 record
03325    3325 6  1,2051 1               AD       ONE
03326    3326 5  0,0327 1               TS       EX_IN_jobPtr

03327    3327 1  0,0326 1               CCS      EX_IN_loopCnt  ; done clearing jobList?
03330    3330 0  1,3304 0               TC       EX_IN_loop2    ; not yet

03331    3331 0  0,0325 0               TC       EX_IN_saveQ    ; return


                         ;------------------------------------------------------------------------
                         ; EX_findIns - FIND INSERTION POINT INTO SORTED LIST
                         ;
                         ; Insert a job record into the sorted list. Use 'EX_IS_newPrio',
                         ; EX_IS_newPrioB and 'EX_IS_newLoc' to set the fields of record to
                         ; be inserted.
                         ; Performs an insertion sort, with the records sorted by priority.
                         ; Highest priority is at the front of the list. If several records
                         ; have the same priority, the records inserted first will appear first
                         ; in the list. NIL records have a priority of zero.
                         ;------------------------------------------------------------------------

                         EX_findIns     EQU      *
03332    3332 3  0,0001 0               XCH      Q
03333    3333 5  0,0351 0               TS       EX_IS_saveQ    ; save return address

03334    3334 3  1,2651 1               CAF      EX_jobLstEnd1  ; set pointer to back of list
03335    3335 5  0,0352 0               TS       EX_IS_jobPtr

03336    3336 6  1,2046 1               AD       NEG1           ; set pointer to rec in front of it
03337    3337 5  0,0353 1               TS       EX_IS_jobPtr2

03340    3340 3  1,2050 0               CAF      ZERO
03341    3341 2  0,0352 1               INDEX    EX_IS_jobPtr
03342    3342 2  0,0000 0               INDEX    0
03343    3343 6  0,0143 1               AD       PRIORITY       ; check last record on list

03344    3344 1  0,0000 0               CCS      A              ; list full?
03345    3345 0  1,3405 0               TC       EX_FI_done     ; >0 yes

                         ; Work from the back of the list to the front, pushing each record
                         ; to the back until the insertion point is found.

03346    3346 3  1,2653 0               CAF      EX_numJobs1    ; loop for number of jobs minus 1
                         EX_FI_loop     EQU      *
```

```
03347    3347 5  0,0354 0               TS       EX_IS_loopCnt

03350    3350 3  1,2050 0               CAF      ZERO
03351    3351 2  0,0353 0               INDEX    EX_IS_jobPtr2
03352    3352 2  0,0000 0               INDEX    0
03353    3353 6  0,0143 1               AD       PRIORITY
03354    3354 1  0,0000 0               CCS      A              ; previous record is NIL?
03355    3355 0  1,3357 0               TC       *+2            ; no, so check it
03356    3356 0  1,3376 0               TC       EX_FI_bumpPtr  ; yes, so skip to next record


                        ; Is this the insertion point?

03357    3357 4  0,0346 1               CS       EX_IS_newPrio
03360    3360 2  0,0353 0               INDEX    EX_IS_jobPtr2
03361    3361 2  0,0000 0               INDEX    0
03362    3362 6  0,0143 1               AD       PRIORITY
03363    3363 1  0,0000 0               CCS      A              ; found insertion point?
03364    3364 0  1,3405 0               TC       EX_FI_insRec   ; >0 yes
03365    3365 0  1,3405 0               TC       EX_FI_insRec   ; +0 yes
03366    3366 0  1,3370 0               TC       *+2            ; <0 no, keep checking
03367    3367 0  1,3405 0               TC       EX_FI_insRec   ; -0 yes

                        ; No, bump the record toward the back of the list.

03370    3370 2  0,0353 0               INDEX    EX_IS_jobPtr2
03371    3371 3  0,0000 1               XCH      0
03372    3372 2  0,0352 1               INDEX    EX_IS_jobPtr
03373    3373 3  0,0000 1               XCH      0
03374    3374 2  0,0353 0               INDEX    EX_IS_jobPtr2
03375    3375 3  0,0000 1               XCH      0

                        EX_FI_bumpPtr  EQU      *
03376    3376 3  0,0352 0               XCH      EX_IS_jobPtr   ; bump job pointer forward 1 record
03377    3377 6  1,2046 1               AD       NEG1
03400    3400 5  0,0352 0               TS       EX_IS_jobPtr

03401    3401 6  1,2046 1               AD       NEG1           ; set pointer to record in front of it
03402    3402 5  0,0353 1               TS       EX_IS_jobPtr2

03403    3403 1  0,0354 1               CCS      EX_IS_loopCnt  ; done bumping jobs backward?
03404    3404 0  1,3347 1               TC       EX_FI_loop     ; not yet

                        ; New record should be inserted at EX_IS_jobPtr.

                        EX_FI_insRec   EQU      *

                        EX_FI_done     EQU      *
03405    3405 3  1,2050 0               CAF      ZERO
03406    3406 6  0,0352 0               AD       EX_IS_jobPtr   ; get insertion spot in list
03407    3407 0  0,0351 0               TC       EX_IS_saveQ    ; return


                        ;------------------------------------------------------------------------
                        ; EX_remove - REMOVE JOB FROM FRONT OF LIST
                        ;
                        ; Remove job from front of list and copy it to the current job. Bubble
                        ; any remaining jobs toward the front of the list.
                        ;------------------------------------------------------------------------

                        EX_remove      EQU      *
03410    3410 3  0,0001 0               XCH      Q
03411    3411 5  0,0336 1               TS       EX_RM_saveQ    ; save return address

03412    3412 3  1,2647 0               CAF      EX_jobLstStart ; set pointer to front of list
03413    3413 5  0,0337 0               TS       EX_RM_jobPtr

03414    3414 6  1,2051 1               AD       ONE            ; set pointer to next rec behind it
03415    3415 5  0,0340 0               TS       EX_RM_jobPtr2

                        ; Dequeue the record at the top of the list (the next job to run).
                        ; Make it the current job by copying it to the current job record.

03416    3416 3  1,2646 1               XCH      EX_jobRecSize
03417    3417 5  0,0344 1               TS       EX_RM_field

                        EX_RM_loop1    EQU      *
03420    3420 1  0,0344 0               CCS      EX_RM_field    ; done?
03421    3421 0  1,3423 1               TC       *+2            ; not yet
03422    3422 0  1,3437 1               TC       EX_RM_done1    ; yes
03423    3423 5  0,0344 1               TS       EX_RM_field

                        ; copy field from list to current job

03424    3424 3  1,2050 0               CAF      ZERO
03425    3425 2  0,0337 1               INDEX    EX_RM_jobPtr
03426    3426 6  0,0000 1               AD       0              ; get index to record
```

```
03427    3427 6  0,0344 1               AD       EX_RM_field    ; add field displacement
03430    3430 5  0,0345 0               TS       EX_RM_findx    ; save index to field
03431    3431 3  1,2050 0               CAF      ZERO
03432    3432 2  0,0345 1               INDEX    EX_RM_findx
03433    3433 6  0,0130 0               AD       EX_currentJob  ; get field
03434    3434 2  0,0344 0               INDEX    EX_RM_field
03435    3435 5  0,0130 0               TS       EX_currentJob  ; move to current job

03436    3436 0  1,3420 1               TC       EX_RM_loop1

                         ; done copying record for current job. Restore the current job to
                         ; its default priority, in case it was previously elevated.

                         EX_RM_done1    EQU      *
03437    3437 3  1,2050 0               CAF      ZERO
03440    3440 6  0,0144 0               AD       JOBPRIOBASE
03441    3441 5  0,0143 1               TS       PRIORITY

03442    3442 2  0,0337 1               INDEX    EX_RM_jobPtr
03443    3443 3  0,0000 1               XCH      0
03444    3444 5  0,0341 1               TS       EX_RM_savePtr  ; so we can move it to the end later


                         ; Loop through the remaining records in the job list and
                         ; bubble them up to the front.

03445    3445 3  1,2653 0               CAF      EX_numJobs1    ; loop for number of jobs minus 1
                         EX_RM_loop2    EQU      *
03446    3446 5  0,0342 1               TS       EX_RM_loopCnt


03447    3447 2  0,0340 1               INDEX    EX_RM_jobPtr2
03450    3450 3  0,0000 1               XCH      0
03451    3451 2  0,0337 1               INDEX    EX_RM_jobPtr
03452    3452 5  0,0000 1               TS       0

03453    3453 1  0,0000 0               CCS      A              ; remainder of list empty?
03454    3454 0  1,3456 0               TC       *+2            ; >0, no
03455    3455 0  1,3465 0               TC       EX_RM_done2    ; +0, yes, so exit

03456    3456 3  0,0337 0               XCH      EX_RM_jobPtr   ; bump job pointer back 1 record
03457    3457 6  1,2051 1               AD       ONE
03460    3460 5  0,0337 0               TS       EX_RM_jobPtr

03461    3461 6  1,2051 1               AD       ONE            ; set pointer to record behind it
03462    3462 5  0,0340 0               TS       EX_RM_jobPtr2

03463    3463 1  0,0342 0               CCS      EX_RM_loopCnt  ; done bumping jobs upward?
03464    3464 0  1,3446 1               TC       EX_RM_loop2    ; not yet

                         ; Since we removed a record, the last record on the list
                         ; should be NIL.

                         EX_RM_done2    EQU      *
03465    3465 3  0,0341 1               XCH      EX_RM_savePtr
03466    3466 2  0,0337 1               INDEX    EX_RM_jobPtr   ; move the index for the top record
03467    3467 5  0,0000 1               TS       0              ; to the bottom of the list

                         ; set all fields in NIL record to zero

03470    3470 3  1,2646 1               XCH      EX_jobRecSize
03471    3471 5  0,0344 1               TS       EX_RM_field

                         EX_RM_loop3    EQU      *
03472    3472 1  0,0344 0               CCS      EX_RM_field    ; done?
03473    3473 0  1,3475 1               TC       *+2            ; not yet
03474    3474 0  1,3507 0               TC       EX_RM_done3    ; yes
03475    3475 5  0,0344 1               TS       EX_RM_field

                         ; set this field to zero

03476    3476 3  1,2050 0               CAF      ZERO
03477    3477 2  0,0337 1               INDEX    EX_RM_jobPtr
03500    3500 6  0,0000 1               AD       0              ; get index to record
03501    3501 6  0,0344 1               AD       EX_RM_field    ; add field displacement
03502    3502 5  0,0345 0               TS       EX_RM_findx    ; save index to field
03503    3503 3  1,2050 0               CAF      ZERO
03504    3504 2  0,0345 1               INDEX    EX_RM_findx
03505    3505 5  0,0130 0               TS       EX_currentJob  ; clear field

03506    3506 0  1,3472 0               TC       EX_RM_loop3

                         EX_RM_done3    EQU      *
03507    3507 0  0,0336 1               TC       EX_RM_saveQ    ; return


                         ;------------------------------------------------------------------------
```

```
                        ; DUMMY JOB - runs at the lowest priority and never terminates. Ensures
                        ; that there is always at least one job executing. Sleeping jobs are
                        ; given a lower priority than the dummy job.
                        ;
                        ; The dummy job controls the computer activity light on the DSKY. When
                        ; the dummy job is running, the light is off. When the dummy job is
                        ; preempted by a higher priority job, the light is on.
                        ;
                        ; I couldn't find good information on the computer activity light
                        ; in COLOSSUS, so this is my best guess concerning its operation. It
                        ; seems consistent witht the MPEG video of the Apollo 11 DSKY.
                        ;------------------------------------------------------------------------


                        ; entering dummy job -- turn off computer activity light

                        dumJob          EQU     *
03510    3510 3  1,2050 0               CAF     ZERO
03511    3511 6  0,0011 1               AD      DSALMOUT
03512    3512 7  1,3525 1               MASK    NOTACTLT
03513    3513 5  0,0011 1               TS      DSALMOUT        ; turn bit1 off

                        ; runtime loop for dummy job

                        dumJob1         EQU     *
03514    3514 1  0,0307 1               CCS     newJob          ; check for context switch
03515    3515 0  1,3517 1               TC      dumJob2         ; yes
03516    3516 0  1,3514 1               TC      dumJob1         ; no

                        ; exiting dummy job -- turn on computer activity light

                        dumJob2         EQU     *
03517    3517 4  0,0011 0               CS      DSALMOUT        ; inclusive OR bit 1 with 1 using
03520    3520 7  1,3525 1               MASK    NOTACTLT        ; Demorgan's theorem
03521    3521 4  0,0000 0               COM
03522    3522 5  0,0011 1               TS      DSALMOUT

03523    3523 0  1,2733 1               TC      CHANG1          ; exit to run higher priority job
03524    3524 0  1,3510 0               TC      dumJob          ; job done, return here, light off again

03525    3525     77776 1 NOTACTLT       DS      %77776          ; 1's compliment of bit1 (comp activity light)
                                        INCL    bank_f.asm      ; bank intercommunication routines
                        ;=============================================================================
                        ; BANK INTERCOMMUNICATION (file:bank_f.asm)
                        ;
                        ; Version:  1.0
                        ; Author:   John Pultorak
                        ; Date:     01/19/2002
                        ;
                        ; PURPOSE:
                        ; Contains bank intercommunication routines.
                        ; The source is missing from my (incomplete) listing of COLOSSUS. The
                        ; implementation here is inferred from the usage in the COLOSSUS pinball
                        ; routines. Some of these routines could probably be combined or optimized
                        ; away if I understood the pinball software architecture a little better.
                        ;=============================================================================

                        ;-----------------------------------------------------------------------
                        ; DXCHJUMP
                        ; Do a bank jump to the CADR in register A. After the bank jump, the return
                        ; CADR is in register A. Contents of register Q are destroyed.
                        ; This is my attempt to implement the block I equivalent for
                        ;       DCA     MY2CADR
                        ;       DXCH    Z
                        ;... which is used in some places in COLOSSUS to implement bank jumps. In that
                        ; implementation, MY2CADR has the lower portion of the address in MYCADR and
                        ; the bank portion in MY2CADR+1. DCA loads the lower address into A and the
                        ; bank address into L. DXCH loads the lower address into Z and the bank portion
                        ; into BB (both bank register), thereby doing a bank call. After the call,
                        ; the lower return address is in A and the return bank is in L.
                        ;-----------------------------------------------------------------------

                        DXCHJUMP        EQU     *
03526    3526 5  0,0576 0               TS      ADDRWD1         ; save 14-bit destination address

03527    3527 3  0,0001 0               XCH     Q
03530    3530 5  0,0617 1               TS      DCRET           ; save old return address

03531    3531 3  0,0015 0               XCH     BANK
03532    3532 5  0,0616 0               TS      DCBANK          ; save old bank

                        ; put the 12-bit destination address in ADDRWD1

03533    3533 4  0,0576 1               CS      ADDRWD1         ; -(14bitAddr)+%6000
03534    3534 6  1,2102 0               AD      bankAddr
03535    3535 1  0,0000 0               CCS     A               ; CADR is bank addressed?
03536    3536 0  1,3547 1               TC      DODXCHCALL      ; >0 no, just run it, as is
```

```
03537    3537 0  1,3541 1               TC       *+2            ; +0 yes
03540    3540 0  1,3541 1               TC       *+1            ; <0 yes

03541    3541 3  1,2050 0               CAF      ZERO
03542    3542 6  0,0576 0               AD       ADDRWD1
03543    3543 5  0,0015 0               TS       BANK           ; set the bank

03544    3544 7  1,2103 0               MASK     lowAddr        ; get lowest 10-bits of address
03545    3545 6  1,2102 0               AD       bankAddr       ; set bits 11,12 for fixed-switchable
03546    3546 5  0,0576 0               TS       ADDRWD1        ; save 12-bit destination address

                         ; put the 14-bit return CADR into A.

                         DODXCHCALL     EQU      *
03547    3547 4  0,0617 0               CS       DCRET          ; get 12-bit return address
03550    3550 6  1,2102 0               AD       bankAddr       ; -(12bitAddr)+%6000
03551    3551 1  0,0000 0               CCS      A              ; return address is bank addressed?
03552    3552 0  1,3561 0               TC       DC_NOTBANK     ; >0 no, just use it, as is
03553    3553 0  1,3555 1               TC       *+2            ; +0 yes
03554    3554 0  1,3555 1               TC       *+1            ; <0 yes

03555    3555 4  1,2102 1               CS       bankAddr       ; 12bitAddr - %6000
03556    3556 6  0,0617 1               AD       DCRET
03557    3557 6  0,0616 0               AD       DCBANK         ; put return CADR in A
03560    3560 0  1,3563 1               TC       *+3

                         DC_NOTBANK     EQU      *
03561    3561 3  1,2050 0               CAF      ZERO
03562    3562 6  0,0617 1               AD       DCRET          ; put return CADR in A

03563    3563 2  0,0576 1               INDEX    ADDRWD1        ; apply indirect address to next instr.
03564    3564 0  0,0000 1               TC       0              ; make the jump


                         ;-----------------------------------------------------------------------
                         ; BANKCALL
                         ; Do a bank jump to the location referenced by the 14-bit address referenced
                         ; in Q. Does not affect register A (but assumes A does not contain an
                         ; overflow). Functionally identical to POSTJUMP.
                         ; Usage:
                         ;       TC       BANKCALL         ; bank jump to CADR
                         ;       DS       MYCADR           ; the 14-bit address
                         ; returns here if MYCADR calls TC Q.
                         ;
                         ; Inferred from the AGC Block II COLOSSUS rev 249 assembly listing,
                         ; Oct 28, 1968.
                         ;-----------------------------------------------------------------------

                         BANKCALL       EQU      *
03565    3565 5  0,0612 1               TS       BCA            ; save A

03566    3566 2  0,0001 1               INDEX    Q              ; load the CADR into A
03567    3567 3  0,0000 1               CAF      0
03570    3570 5  0,0576 0               TS       ADDRWD1        ; save 14-bit destination address

03571    3571 3  0,0001 0               XCH      Q
03572    3572 5  0,0611 1               TS       BCRET          ; save old return address-1

03573    3573 3  0,0015 0               XCH      BANK
03574    3574 5  0,0610 0               TS       BCBANK         ; save old bank

03575    3575 4  0,0576 1               CS       ADDRWD1        ; -(14bitAddr)+%6000
03576    3576 6  1,2102 0               AD       bankAddr
03577    3577 1  0,0000 0               CCS      A              ; CADR is bank addressed?
03600    3600 0  1,3611 1               TC       DOBANKCALL     ; >0 no, just run it, as is
03601    3601 0  1,3603 1               TC       *+2            ; +0 yes
03602    3602 0  1,3603 1               TC       *+1            ; <0 yes

03603    3603 3  1,2050 0               CAF      ZERO
03604    3604 6  0,0576 0               AD       ADDRWD1
03605    3605 5  0,0015 0               TS       BANK           ; set the bank

03606    3606 7  1,2103 0               MASK     lowAddr        ; get lowest 10-bits of address
03607    3607 6  1,2102 0               AD       bankAddr       ; set bits 11,12 for fixed-switchable
03610    3610 5  0,0576 0               TS       ADDRWD1

                         DOBANKCALL     EQU      *
03611    3611 3  0,0612 1               XCH      BCA            ; restore A
03612    3612 2  0,0576 1               INDEX    ADDRWD1        ; apply indirect address to next instr.
03613    3613 0  0,0000 1               TC       0              ; make the jump

                         ; Jump returns here; restore the old bank and return

03614    3614 5  0,0612 1               TS       BCA            ; save A
03615    3615 3  0,0610 0               XCH      BCBANK
03616    3616 5  0,0015 0               TS       BANK
```

```
03617    3617 3  0,0611 1               XCH      BCRET
03620    3620 6  1,2051 1               AD       ONE            ; skip CADR
03621    3621 5  0,0001 0               TS       Q

03622    3622 3  0,0612 1               XCH      BCA            ; restore A
03623    3623 0  0,0000 0               RETURN


                         ;-----------------------------------------------------------------------
                         ; MYBANKCALL
                         ; Functionally identical to BANKCALL. Used for converting the FLASHON/FLASHOFF
                         ; COLOSSUS block II code to block I. In Block II, the V/N flash is controlled by
                         ; setting a bit in an I/O channel. In Block I, a bit in the display table must
                         ; be set using _11DSPIN. Because _11DSPIN is in fixed/switchable memory, but is
                         ; called from fixed/fixed, a bank call function is needed. The original BANKCALL
                         ; could not be used because it is not reentrant and I dont understand its usage
                         ; in COLOSSUS well enough to be certain  that FLASHON/FLASHOFF isn't already
                         ; being called somewhere through  BANKCALL.
                         ;-----------------------------------------------------------------------

                         MYBANKCALL     EQU      *
03624    3624 5  0,0615 0               TS       MBCA           ; save A

03625    3625 2  0,0001 1               INDEX    Q              ; load the CADR into A
03626    3626 3  0,0000 1               CAF      0
03627    3627 5  0,0576 0               TS       ADDRWD1        ; save 14-bit destination address

03630    3630 3  0,0001 0               XCH      Q
03631    3631 6  1,2051 1               AD       ONE            ; skip CADR
03632    3632 5  0,0614 1               TS       MBCRET         ; save old return address

03633    3633 3  0,0015 0               XCH      BANK
03634    3634 5  0,0613 0               TS       MBCBANK        ; save old bank

03635    3635 3  1,2050 0               CAF      ZERO
03636    3636 6  0,0576 0               AD       ADDRWD1
03637    3637 5  0,0015 0               TS       BANK           ; set the bank

03640    3640 7  1,2103 0               MASK     lowAddr        ; get lowest 10-bits of address
03641    3641 6  1,2102 0               AD       bankAddr       ; set bits 11,12 for fixed-switchable
03642    3642 5  0,0576 0               TS       ADDRWD1

03643    3643 3  0,0615 0               XCH      MBCA           ; restore A
03644    3644 2  0,0576 1               INDEX    ADDRWD1        ; apply indirect address to next instr.
03645    3645 0  0,0000 1               TC       0              ; make the jump

                         ; Jump returns here; restore the old bank and return

03646    3646 5  0,0615 0               TS       MBCA           ; save A
03647    3647 3  0,0613 0               XCH      MBCBANK
03650    3650 5  0,0015 0               TS       BANK

03651    3651 3  0,0615 0               XCH      MBCA           ; restore A
03652    3652 0  0,0614 1               TC       MBCRET


                         ;-----------------------------------------------------------------------
                         ; POSTJUMP
                         ; Do a bank jump to the location referenced by the 14-bit address referenced
                         ; in Q.  Does not affect register A (but assumes A does not contain an
                         ; overflow). Functionally identical to BANKCALL
                         ; Usage:
                         ;       TC       POSTJUMP         ; bank jump to CADR
                         ;       DS       MYCADR           ; the 14-bit address
                         ; returns here if MYCADR calls TC Q.
                         ;
                         ; Inferred from the AGC Block II COLOSSUS rev 249 assembly listing,
                         ; Oct 28, 1968.
                         ;-----------------------------------------------------------------------

                         POSTJUMP       EQU      *
03653    3653 5  0,0607 0               TS       PJA            ; save A

03654    3654 2  0,0001 1               INDEX    Q              ; load the CADR into A
03655    3655 3  0,0000 1               CAF      0
03656    3656 5  0,0576 0               TS       ADDRWD1        ; save 14-bit destination address

03657    3657 3  0,0001 0               XCH      Q
03660    3660 5  0,0606 1               TS       PJRET          ; save old return address-1

03661    3661 3  0,0015 0               XCH      BANK
03662    3662 5  0,0605 1               TS       PJBANK         ; save old bank

03663    3663 4  0,0576 1               CS       ADDRWD1        ; -(14bitAddr)+%6000
03664    3664 6  1,2102 0               AD       bankAddr
03665    3665 1  0,0000 0               CCS      A              ; CADR is bank addressed?
03666    3666 0  1,3677 1               TC       DOPOSTJUMP     ; >0 no, just run it, as is
```

```
03667    3667 0  1,3671 1               TC       *+2            ; +0 yes
03670    3670 0  1,3671 1               TC       *+1            ; <0 yes

03671    3671 3  1,2050 0               CAF      ZERO
03672    3672 6  0,0576 0               AD       ADDRWD1
03673    3673 5  0,0015 0               TS       BANK           ; set the bank

03674    3674 7  1,2103 0               MASK     lowAddr        ; get lowest 10-bits of address
03675    3675 6  1,2102 0               AD       bankAddr       ; set bits 11,12 for fixed-switchable
03676    3676 5  0,0576 0               TS       ADDRWD1

                        DOPOSTJUMP     EQU      *
03677    3677 3  0,0607 0               XCH      PJA            ; restore A
03700    3700 2  0,0576 1               INDEX    ADDRWD1        ; apply indirect address to next instr.
03701    3701 0  0,0000 1               TC       0              ; make the jump

                        ; Jump returns here; restore the old bank and return

03702    3702 5  0,0607 0               TS       PJA            ; save A

03703    3703 3  0,0605 1               XCH      PJBANK
03704    3704 5  0,0015 0               TS       BANK

03705    3705 3  0,0606 1               XCH      PJRET
03706    3706 6  1,2051 1               AD       ONE            ; skip CADR
03707    3707 5  0,0001 0               TS       Q

03710    3710 3  0,0607 0               XCH      PJA            ; restore A
03711    3711 0  0,0000 0               RETURN


                        ;-----------------------------------------------------------------------
                        ; BANKJUMP
                        ; Do a bank jump to the location referenced by the 14-bit address in A.
                        ; Usage:
                        ; CADRSTOR       DS       MYCADR
                        ;
                        ;                CAF      CADRSTOR         ; load the 14-bit address
                        ;                TC       BANKJUMP         ; bank jump to CADR
                        ; returns here if MYCADR calls TC Q
                        ;
                        ; Inferred from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968.
                        ;-----------------------------------------------------------------------

                        BANKJUMP       EQU      *
03712    3712 5  0,0576 0               TS       ADDRWD1        ; save 14-bit destination address

03713    3713 3  0,0001 0               XCH      Q
03714    3714 5  0,0604 0               TS       BJRET          ; save old return address

03715    3715 3  0,0015 0               XCH      BANK
03716    3716 5  0,0603 1               TS       BJBANK         ; save old bank

03717    3717 4  0,0576 1               CS       ADDRWD1        ; -(14bitAddr)+%6000
03720    3720 6  1,2102 0               AD       bankAddr
03721    3721 1  0,0000 0               CCS      A              ; CADR is bank addressed?
03722    3722 0  1,3733 0               TC       DOBANKJUMP     ; >0 no, just run it, as is
03723    3723 0  1,3725 1               TC       *+2            ; +0 yes
03724    3724 0  1,3725 1               TC       *+1            ; <0 yes

03725    3725 3  1,2050 0               CAF      ZERO
03726    3726 6  0,0576 0               AD       ADDRWD1
03727    3727 5  0,0015 0               TS       BANK           ; set the bank

03730    3730 7  1,2103 0               MASK     lowAddr        ; get lowest 10-bits of address
03731    3731 6  1,2102 0               AD       bankAddr       ; set bits 11,12 for fixed-switchable
03732    3732 5  0,0576 0               TS       ADDRWD1

                        DOBANKJUMP     EQU      *
03733    3733 2  0,0576 1               INDEX    ADDRWD1        ; apply indirect address to next instr.
03734    3734 0  0,0000 1               TC       0              ; make the jump

                        ; Jump returns here; restore the old bank and return

03735    3735 3  0,0603 1               XCH      BJBANK
03736    3736 5  0,0015 0               TS       BANK

03737    3737 3  0,0604 0               XCH      BJRET
03740    3740 5  0,0001 0               TS       Q
03741    3741 0  0,0000 0               RETURN


                        ;-----------------------------------------------------------------------
                        ; DATACALL
                        ; Retrieve memory contents at location referenced by the 14-bit address in A.
                        ; Usage:
```

```
                        ; CADRSTOR       DS       MYCADR
                        ;
                        ;                CAF      CADRSTOR         ; load the 14-bit address
                        ;                TC       DATACALL         ; return data in A
                        ;
                        ; Inferred from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968.
                        ;--------------------------------------------------------------------------

                        DATACALL       EQU      *
03742    3742 5  0,0576 0               TS       ADDRWD1        ; save 14-bit address

03743    3743 3  0,0001 0               XCH      Q
03744    3744 5  0,0604 0               TS       BJRET          ; save old return address

03745    3745 3  0,0015 0               XCH      BANK
03746    3746 5  0,0603 1               TS       BJBANK         ; save old bank

03747    3747 4  0,0576 1               CS       ADDRWD1        ; -(14bitAddr)+%6000
03750    3750 6  1,2102 0               AD       bankAddr
03751    3751 1  0,0000 0               CCS      A              ; CADR is bank addressed?
03752    3752 0  1,3763 0               TC       DODATACALL     ; >0 no, just use it, as is
03753    3753 0  1,3755 0               TC       *+2            ; +0 yes
03754    3754 0  1,3755 0               TC       *+1            ; <0 yes

03755    3755 3  1,2050 0               CAF      ZERO
03756    3756 6  0,0576 0               AD       ADDRWD1
03757    3757 5  0,0015 0               TS       BANK           ; set the bank

03760    3760 7  1,2103 0               MASK     lowAddr        ; get lowest 10-bits of address
03761    3761 6  1,2102 0               AD       bankAddr       ; set bits 11,12 for fixed-switchable
03762    3762 5  0,0576 0               TS       ADDRWD1

                        DODATACALL     EQU      *
03763    3763 3  1,2050 0               CAF      ZERO
03764    3764 2  0,0576 1               INDEX    ADDRWD1        ; apply indirect address to next instr.
03765    3765 6  0,0000 1               AD       0              ; load the word

03766    3766 3  0,0603 1               XCH      BJBANK         ; restore the old bank
03767    3767 5  0,0015 0               TS       BANK

03770    3770 3  0,0603 1               XCH      BJBANK         ; get the word
03771    3771 0  0,0604 0               TC       BJRET          ; return



                                       INCL     T4rupt_f.asm   ; T4RUPT handler
                        ;==========================================================================
                        ; T4RUPT (file:T4rupt_f.asm)
                        ;
                        ; Version:  1.0
                        ; Author:   John Pultorak
                        ; Date:     01/09/2002
                        ;
                        ; PURPOSE:
                        ; Contains T4RUPT handler and DSPOUT subroutine to update DSKY.
                        ;==========================================================================

                        ; RELTAB is a packed table. RELAYWORD code in upper 4 bits, RELAY code
                        ; in lower 5 bits. In COLOSSUS, p. 129.

                        RELTAB         EQU      *
03772    3772     04025 1               DS       %04025
03773    3773     10003 0               DS       %10003
03774    3774     14031 0               DS       %14031
03775    3775     20033 0               DS       %20033
03776    3776     24017 1               DS       %24017
03777    3777     30036 1               DS       %30036
04000    4000     34034 1               DS       %34034
04001    4001     40023 1               DS       %40023
04002    4002     44035 1               DS       %44035
04003    4003     50037 0               DS       %50037
04004    4004     54000 0               DS       %54000
04005    4005     60000 1 RELTAB11      DS       %60000


                        ;--------------------------------------------------------------------------
                        ; DK_initDK - INITIALIZE DSKY
                        ;
                        ; Subroutine initializes the eraseable memory segment for DSKY displays.
                        ; Blank DSKY registers program, verb, noun, R1, R2, R3.
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, Fresh Start and Restart, p.187.
                        ;--------------------------------------------------------------------------

04006    4006     05265 0 DKTESTINIT     DS       %5265          ; init DSKY to all zeroes (TEST ONLY)
```

```
                        DK_initDK       EQU     *
04007   4007 3  0,0001 0                XCH     Q
04010   4010 5  0,0546 0                TS      DK_IN_saveQ     ; save return address

04011   4011 3  1,2060 0                CAF     TEN             ; blank DSKY registers
04012   4012 5  0,0130 0 DSPOFF         TS      MPAC

04013   4013 4  1,2065 1                CS      BIT12
                        ;                 CS      DKTESTINIT        ; set display to '0'

04014   4014 2  0,0130 1                INDEX   MPAC
04015   4015 5  0,0512 1                TS      DSPTAB
04016   4016 1  0,0130 1                CCS     MPAC
04017   4017 0  2,4012 0                TC      DSPOFF

                        ; followed by additional DSKY initialization p 187, 188)

04020   4020 3  1,2050 0                CAF     ZERO
04021   4021 5  0,0465 0                TS      DSPCNT
04022   4022 5  0,0531 0                TS      CADRSTOR
04023   4023 5  0,0502 0                TS      REQRET
04024   4024 5  0,0504 0                TS      CLPASS
04025   4025 5  0,0501 0                TS      DSPLOCK
04026   4026 5  0,0507 0                TS      MONSAVE         ; kill monitor
04027   4027 5  0,0510 0                TS      MONSAVE1
04030   4030 5  0,0470 1                TS      VERBREG
04031   4031 5  0,0471 0                TS      NOUNREG
04032   4032 5  0,0532 0                TS      DSPLIST

04033   4033 3  1,2105 1                CAF     NOUTCON
04034   4034 5  0,0505 1                TS      NOUT

                        ; set DSKY display bit (sign bit). Word must be negative, but
                        ; not minus zero (find out where they do this in COLOSSUS)

04035   4035 4  1,2051 0                CS      ONE
04036   4036 5  0,0355 1                TS      FLAGWRD5

                        ; initialize DSPCNT (index into DSPTAB).

04037   4037 3  1,2050 0                CAF     ZERO
04040   4040 6  2,4072 0                AD      TABLNTH
04041   4041 5  0,0465 0                TS      DSPCNT

                        ; schedule 1st T4RUPT

04042   4042 3  2,4074 0                CAF     _120MRUPT       ; reschedule interrupt for 120 mSec
04043   4043 5  0,0040 0                TS      TIME4

04044   4044 3  0,0546 0                XCH     DK_IN_saveQ
04045   4045 5  0,0001 0                TS      Q               ; restore return address
04046   4046 0  0,0000 0                RETURN

                        ;----------------------------------------------------------------------
                        ; T4PROG -- T4RUPT PROGRAM
                        ;
                        ; Performs T4RUPT (DSRUPT) functions.
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.129.
                        ;----------------------------------------------------------------------

                        T4PROG          EQU     *
04047   4047 3  0,0001 0                XCH     Q
04050   4050 5  0,0544 1                TS      T4RET           ; save return address

04051   4051 0  2,4116 0                TC      DSPOUT          ; update DSKY display

04052   4052 3  2,4074 0                CAF     _120MRUPT       ; reschedule interrupt for 120 mSec
04053   4053 5  0,0040 0                TS      TIME4

04054   4054 3  0,0544 1                XCH     T4RET
04055   4055 5  0,0001 0                TS      Q               ; restore return address
04056   4056 0  0,0000 0                RETURN

                        ;----------------------------------------------------------------------
                        ; DSPOUT -- PUTS OUT DISPLAYS
                        ;
                        ; Writes changes in the software display buffer to the AGC DSKY hardware
                        ; display.
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.131.
                        ;----------------------------------------------------------------------

                        DSPOUTSR        EQU     *
```

```
04057    4057 5  0,0505 1                TS       NOUT           ; decrement NOUT

04060    4060 4  1,2050 1                CS       ZERO
04061    4061 5  0,0370 0                TS       DSRUPTEM       ; set to -0 for 1st pass thru DSPTAB

04062    4062 3  0,0465 0                XCH      DSPCNT
04063    4063 6  1,2045 1                AD       NEG0           ; to prevent +0
04064    4064 5  0,0465 0                TS       DSPCNT

                         DSPSCAN        EQU      *
04065    4065 2  0,0465 1                INDEX    DSPCNT
04066    4066 1  0,0512 0                CCS      DSPTAB         ; test sign of DSPTAB + DSPCNT

04067    4067 1  0,0465 1                CCS      DSPCNT         ; >0, already displayed, test DSPCNT
04070    4070 0  2,4063 0                TC       DSPSCAN-2      ; if DSPCNT +, again
04071    4071 0  2,4103 1                TC       DSPLAY         ; <0, not yet displayed

04072    4072     00012 1 TABLNTH        DS       %12            ; dec 10, length of DSPTAB
04073    4073 1  0,0370 1                CCS      DSRUPTEM       ; if DSRUPTEM=+0, 2nd pass thru DSPTAB

04074    4074     37764 0 _120MRUPT      DS       16372          ; (DSPCNT=0), +0 into NOUT

04075    4075 5  0,0505 1                TS       NOUT           ; DSRUPTEM=+0, every table entry was checked
04076    4076 0  2,4126 0                TC       DSPOUTEXIT     ; return

04077    4077 5  0,0370 0                TS       DSRUPTEM       ; DSRUPTEM=-0, 1st pass thru DSPTAB
04100    4100 3  2,4072 0                CAF      TABLNTH        ; (DSPCNT=0), +0 into DSRUPTEM, pass again
04101    4101 0  2,4064 1                TC       DSPSCAN-1

04102    4102 0  2,4126 0                TC       DSPOUTEXIT     ; return


                         DSPLAY         EQU      *
04103    4103 6  1,2051 1                AD       ONE
04104    4104 2  0,0465 1                INDEX    DSPCNT
04105    4105 5  0,0512 1                TS       DSPTAB         ; replace positively
04106    4106 7  2,4672 1                MASK     LOW11          ; remove bits 12 to 15
04107    4107 5  0,0370 0                TS       DSRUPTEM
04110    4110 3  2,4666 0                CAF      HI5
04111    4111 2  0,0465 1                INDEX    DSPCNT
04112    4112 7  1,3772 1                MASK     RELTAB         ; pick up bits 12 to 15 of RELTAB entry
04113    4113 6  0,0370 0                AD       DSRUPTEM
04114    4114 5  0,0010 0                TS       OUT0           ; was EXTEND/WRITE OUT0 in block II

04115    4115 0  2,4126 0                TC       DSPOUTEXIT     ; return


                         DSPOUT         EQU      *
04116    4116 3  0,0001 0                XCH      Q
04117    4117 5  0,0545 0                TS       DSPOUTRET      ; save return address

04120    4120 1  0,0355 0                CCS      FLAGWRD5       ; no display unless DSKY flag (sign bit) on
04121    4121 3  1,2050 0                CAF      ZERO           ; >0, DSKY disabled
04122    4122 0  2,4126 0                TC       NODSPOUT       ; +0, DSKY disabled
04123    4123 1  0,0505 0                CCS      NOUT           ; <0, DSKY enabled, so test NOUT
04124    4124 0  2,4057 1                TC       DSPOUTSR       ; >0, handle display requests
04125    4125 0  2,4126 0                TC       NODSPOUT       ; +0, no display requests

                         NODSPOUT       EQU      *
                         DSPOUTEXIT     EQU      *
04126    4126 3  0,0545 0                XCH      DSPOUTRET      ; return to calling routine
04127    4127 5  0,0001 0                TS       Q
04130    4130 0  0,0000 0                RETURN

                                        INCL     keyrupt_f.asm  ; KEYRUPT handler
                         ;==============================================================================
                         ; KEYRUPT (file:keyrupt_f.asm)
                         ;
                         ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                         ; Oct 28, 1968, pp. 77.
                         ;==============================================================================

                         ;------------------------------------------------------------------------------
                         ; KEYRUPT -- KEYBOARD INTERRUPT HANDLER
                         ;
                         ; Performs keyRUPT functions. Triggered by a keyboard key entry. N-key
                         ; rollover, implemented as follows: When an interrupt occurs, the current
                         ; job record is saved and then restored when the job resumes after the
                         ; interrupt. The job record includes MPAC, a set of general purpose
                         ; registers assigned to the job. When the keyboard interrupt occurs, the
                         ; interrupt handler stores the keyboard character in MPAC. A job is then
                         ; started to process the character. The new job copies its MPAC fields from
                         ; the current job, so the character is copied to storage owned by the job.
                         ; When additional keyboard interrupts occur, they start their own jobs.
                         ; Up to 7 jobs can be waiting in a queue for execution, so as many as
                         ; 7 keyboard characters can be enqueued for processing. Since all keyboard
                         ; jobs have the same priority, they are enqueued in the order received.
```

```
                        ; Its OK for the keyboard handler to modify the MPAC of the interrupted job
                        ; because the interrupted job's record is restored at the end of the
                        ; interrupt service routine.
                        ;
                        ; Not included in my partial AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, so I had to improvise it from the original flow charts in
                        ; E-1574, p.77.
                        ;-----------------------------------------------------------------------------

04131    4131     37776 0 CHRPRIO        DS       %37776         ; priority of CHARIN job (highest)

                        KEYPROG        EQU      *
04132    4132 3  0,0001 0               XCH      Q
04133    4133 5  0,0601 0               TS       KEYRET         ; save return address

                        ; prepare to EXEC a job to handle the keystroke.

04134    4134 3  0,0004 0               XCH      IN0
04135    4135 7  2,4664 0               MASK     LOW5
04136    4136 3  0,0130 0               XCH      MPAC           ; save keyboard code
04137    4137 3  0,0551 0               XCH      KP_MPAC        ; save previous MPAC

                        ; create the job. It terminates when it finishes processing the key.

04140    4140 3  2,4131 0               CAF      CHRPRIO        ; CHARIN job priority
04141    4141 0  1,3162 1               TC       NOVAC
04142    4142     12000 1               CADR     CHARIN         ; 14 bit CHARIN job address

04143    4143 3  0,0551 0               XCH      KP_MPAC
04144    4144 3  0,0130 0               XCH      MPAC           ; restore previous MPAC

04145    4145 3  0,0601 0               XCH      KEYRET
04146    4146 5  0,0001 0               TS       Q              ; restore return address
04147    4147 0  0,0000 0               RETURN



                                       INCL     math_f.asm     ; DP math routines
                        ;=============================================================================
                        ; MATH LIBRARY (file:math_f.asm)
                        ;
                        ; Version:  1.0
                        ; Author:   John Pultorak
                        ; Date:     03/01/2002
                        ;
                        ; PURPOSE:
                        ; Contains double precision math routines.
                        ;=============================================================================

                        ;-----------------------------------------------------------------------------
                        ; TPAGREE
                        ; Force the signs in a triple precision (TP) word to agree. The word is
                        ; in MPAC, MPAC+1, MPAC+2
                        ;
                        ; The sign of the corrected number is always the sign of the most-significant
                        ; non-zero word.
                        ;
                        ; This isn't included in my partial COLOSSUS listing, so I had to invent
                        ; my own version.
                        ;-----------------------------------------------------------------------------


                        TPAGREE        EQU      *
04150    4150 3  0,0001 0               XCH      Q
04151    4151 5  0,0577 1               TS       MATH_Q         ; return address

                        ; Find the sign to convert to. It will be the sign
                        ; of the most significant non-zero word.

                        TPA_SGN0       EQU      *
04152    4152 1  0,0130 1               CCS      MPAC
04153    4153 0  2,4157 0               TC       TPA_P0         ; >0, sign will be +
04154    4154 0  2,4241 1               TC       TPA_SGN1       ; +0, still don't know sign, check MPAC+1
04155    4155 0  2,4210 0               TC       TPA_M0         ; <0, sign will be -
04156    4156 0  2,4241 1               TC       TPA_SGN1       ; -0, still don't know sign, check MPAC+1


                        ; MPAC is non-zero positive, so reconcile signs to a positive number.

                        TPA_P0         EQU      *
04157    4157 1  0,0131 0               CCS      MPAC+1
04160    4160 0  2,4250 1               TC       TPA_P1+2       ; >0, MPAC+1 is OK, check MPAC+2
04161    4161 0  2,4167 0               TC       TPA_PZ0        ; +0,
04162    4162 0  2,4164 0               TC       *+2            ; <0, fix MPAC+1
04163    4163 0  2,4167 0               TC       TPA_PZ0        ; -0,

04164    4164 3  2,4317 0               CAF      TPA_MPAC0      ; borrow from MPAC to correct MPAC+1
```

```
04165    4165 0  2,4337 1               TC       TPA_FIXP
04166    4166 0  2,4250 1               TC       TPA_P1+2       ; MPAC+1 is now non-zero positive; check
MPAC+2

                         ; MPAC is non-zero positive, MPAC+1 is zero

                         TPA_PZ0        EQU      *
04167    4167 1  0,0132 0               CCS      MPAC+2
04170    4170 0  2,4175 0               TC       *+5            ; >0, zero MPAC+1, MPAC+2 is OK
04171    4171 0  2,4173 0               TC       *+2            ; +0, MPAC+1, +2 both zero
04172    4172 0  2,4200 1               TC       TPA_PZ0FIX     ; <0,

04173    4173 3  1,2050 0               CAF      ZERO           ; make sure they're both +0
04174    4174 5  0,0132 1               TS       MPAC+2
04175    4175 3  1,2050 0               CAF      ZERO
04176    4176 5  0,0131 1               TS       MPAC+1
04177    4177 0  0,0577 1               TC       MATH_Q

                         ; MPAC is non-zero positive, MPAC+1 is zero, MPAC+2 is non-zero negative.
                         ; Solution: borrow from MPAC, transfer borrowed value to MPAC+1, but also
                         ; borrow from MPAC+1, use borrowed value to correct MPAC+2.

                         TPA_PZ0FIX     EQU      *
04200    4200 3  0,0132 1               XCH      MPAC+2         ; move MPAC+2 to MPAC+1 so we can use
04201    4201 5  0,0131 1               TS       MPAC+1         ; our standard correction function

04202    4202 3  2,4317 0               CAF      TPA_MPAC0      ; borrow from MPAC to correct MPAC+1
04203    4203 0  2,4337 1               TC       TPA_FIXP

04204    4204 3  2,4315 1               CAF      MAXPOS         ; move corrected value from MPAC+1 back
04205    4205 3  0,0131 1               XCH      MPAC+1         ; to MPAC+2. Set MPAC+1 to correct value
04206    4206 5  0,0132 1               TS       MPAC+2         ; borrowed from MPAC.
04207    4207 0  0,0577 1               TC       MATH_Q


                         ; The MPAC is non-zero negative, so reconcile signs to a negative number.

                         TPA_M0         EQU      *
04210    4210 1  0,0131 0               CCS      MPAC+1
04211    4211 0  2,4215 0               TC       *+4            ; >0, fix MPAC+1
04212    4212 0  2,4220 0               TC       TPA_MZ0        ; +0,
04213    4213 0  2,4264 0               TC       TPA_M1+2       ; <0, MPAC+1 is OK, check MPAC+2
04214    4214 0  2,4220 0               TC       TPA_MZ0        ; -0,

04215    4215 3  2,4317 0               CAF      TPA_MPAC0      ; borrow from MPAC to correct MPAC+1
04216    4216 0  2,4321 0               TC       TPA_FIXM
04217    4217 0  2,4264 0               TC       TPA_M1+2

                         ; MPAC is non-zero negative, MPAC+1 is zero

                         TPA_MZ0        EQU      *
04220    4220 1  0,0132 0               CCS      MPAC+2
04221    4221 0  2,4231 0               TC       TPA_MZ0FIX     ; >0,
04222    4222 0  2,4224 1               TC       *+2            ; +0, MPAC+1, +2 both zero
04223    4223 0  2,4226 0               TC       *+3            ; <0, zero MPAC+1, MPAC+2 is OK

04224    4224 3  1,2045 1               CAF      NEG0           ; make sure they're both -0
04225    4225 5  0,0132 1               TS       MPAC+2
04226    4226 3  1,2045 1               CAF      NEG0
04227    4227 5  0,0131 1               TS       MPAC+1
04230    4230 0  0,0577 1               TC       MATH_Q

                         ; MPAC is non-zero negative, MPAC+1 is zero, MPAC+2 is non-zero positive
                         ; Solution: borrow from MPAC, transfer borrowed value to MPAC+1, but also
                         ; borrow from MPAC+1, use borrowed value to correct MPAC+2.

                         TPA_MZ0FIX     EQU      *
04231    4231 3  0,0132 1               XCH      MPAC+2         ; move MPAC+2 to MPAC+1 so we can use
04232    4232 5  0,0131 1               TS       MPAC+1         ; our standard correction function

04233    4233 3  2,4317 0               CAF      TPA_MPAC0      ; borrow from MPAC to correct MPAC+1
04234    4234 0  2,4321 0               TC       TPA_FIXM

04235    4235 3  2,4316 1               CAF      MAXNEG         ; move corrected value from MPAC+1 back
04236    4236 3  0,0131 1               XCH      MPAC+1         ; to MPAC+2. Set MPAC+1 to correct value
04237    4237 5  0,0132 1               TS       MPAC+2         ; borrowed from MPAC.
04240    4240 0  0,0577 1               TC       MATH_Q


                         ; MPAC was zero, so we still don't know the sign. Check MPAC+1.

                         TPA_SGN1       EQU      *
04241    4241 1  0,0131 0               CCS      MPAC+1
04242    4242 0  2,4246 0               TC       TPA_P1         ; >0, sign will be +
04243    4243 0  2,4277 1               TC       TPA_SGN2       ; +0, still don't know sign, check MPAC+2
04244    4244 0  2,4262 0               TC       TPA_M1         ; <0, sign will be -
04245    4245 0  2,4277 1               TC       TPA_SGN2       ; -0, still don't know sign, check MPAC+2
```

```
                        ; MPAC+1 is non-zero positive, so reconcile signs to a positive number.

                        TPA_P1          EQU     *
04246    4246 3  1,2050 0               CAF     ZERO
04247    4247 5  0,0130 0               TS      MPAC            ; set MPAC to +0

04250    4250 1  0,0132 0               CCS     MPAC+2
04251    4251 0  0,0577 1               TC      MATH_Q          ; >0, all words are positive
04252    4252 0  0,0577 1               TC      MATH_Q          ; +0, all words are positive
04253    4253 0  2,4257 0               TC      *+4             ; <0, MPAC+2 is nonzero -
04254    4254 3  1,2050 0               CAF     ZERO            ; -0, change to +0 and we're done
04255    4255 5  0,0132 1               TS      MPAC+2
04256    4256 0  0,0577 1               TC      MATH_Q

04257    4257 3  2,4320 1               CAF     TPA_MPAC1       ; borrow from MPAC+1 to correct MPAC+2
04260    4260 0  2,4337 1               TC      TPA_FIXP
04261    4261 0  0,0577 1               TC      MATH_Q

                        ; MPAC+1 is non-zero negative, so reconcile signs to a negative number.

                        TPA_M1          EQU     *
04262    4262 3  1,2045 1               CAF     NEG0
04263    4263 5  0,0130 0               TS      MPAC            ; set MPAC to -0

04264    4264 1  0,0132 0               CCS     MPAC+2
04265    4265 0  2,4274 1               TC      *+7             ; >0, MPAC+2 is nonzero +
04266    4266 0  2,4271 1               TC      *+3             ; +0, change to -0 and we're done
04267    4267 0  0,0577 1               TC      MATH_Q          ; <0, all words are negative
04270    4270 0  0,0577 1               TC      MATH_Q          ; -0, all words are negative

04271    4271 3  1,2045 1               CAF     NEG0            ; +0, change to -0 and we're done
04272    4272 5  0,0132 1               TS      MPAC+2
04273    4273 0  0,0577 1               TC      MATH_Q

04274    4274 3  2,4320 1               CAF     TPA_MPAC1       ; borrow from MPAC+1 to correct MPAC+2
04275    4275 0  2,4321 0               TC      TPA_FIXM
04276    4276 0  0,0577 1               TC      MATH_Q

                        ; MPAC and MPAC+1 were both zero, so we still don't know the sign.
                        ; Check MPAC+2.

                        TPA_SGN2        EQU     *
04277    4277 1  0,0132 0               CCS     MPAC+2
04300    4300 0  2,4304 1               TC      TPA_P2          ; >0, sign is +
04301    4301 0  2,4310 1               TC      TPA_P3          ; +0, number is all zeros
04302    4302 0  2,4306 0               TC      TPA_M2          ; <0, sign is -
04303    4303 0  2,4310 1               TC      TPA_P3          ; -0, number is all zeros

04304    4304 3  1,2050 0 TPA_P2        CAF     ZERO
04305    4305 0  2,4312 0               TC      *+5             ; set MPAC, MPAC+1 to +0

04306    4306 3  1,2045 1 TPA_M2        CAF     NEG0            ; set MPAC, MPAC+1 to -0
04307    4307 0  2,4312 0               TC      *+3

04310    4310 3  1,2050 0 TPA_P3        CAF     ZERO
04311    4311 5  0,0132 1               TS      MPAC+2          ; set MPAC, MPAC+1, MPAC+2 to +0

04312    4312 5  0,0131 1               TS      MPAC+1
04313    4313 5  0,0130 0               TS      MPAC
04314    4314 0  0,0577 1               TC      MATH_Q



04315    4315     37777 1 MAXPOS        DS      %37777          ; largest non-overflow pos number
04316    4316     40000 0 MAXNEG        DS      %40000          ; largest non-overflow neg number

04317    4317     00130 0 TPA_MPAC0     DS      MPAC
04320    4320     00131 1 TPA_MPAC1     DS      MPAC+1


                        ;------------------------------------------------------------------------
                        ; TPA_FIXM
                        ; Reconcile the signs in a double precision word. The most significant word
                        ; is in C(A), the lesser word in C(A+1). Reconcilliation occurs by borrowing
                        ; from C(A) and adding the borrowed amount to C(A+1). C(A) is assumed to be
                        ; negative non-zero number and C(A+1) positive non-zero. The reconciliation
                        ; makes both numbers negative.
                        ;
                        ; This is part of my implementation of TPAGREE.
                        ;------------------------------------------------------------------------

                        TPA_FIXM        EQU     *
04321    4321 5  0,0576 0               TS      ADDRWD1

04322    4322 2  0,0576 1               INDEX   ADDRWD1
04323    4323 4  0,0000 0               CS      0               ; borrow from 1st word
```

```
04324    4324 1  0,0000 0               CCS      A
04325    4325 4  0,0000 0               COM
04326    4326 2  0,0576 1               INDEX    ADDRWD1
04327    4327 5  0,0000 1               TS       0

04330    4330 3  2,4316 1               CAF      MAXNEG
04331    4331 6  1,2046 1               AD       NEG1           ; create negative overflow
04332    4332 2  0,0576 1               INDEX    ADDRWD1
04333    4333 6  0,0001 0               AD       1              ; correct 2nd word, changes sign
04334    4334 2  0,0576 1               INDEX    ADDRWD1
04335    4335 5  0,0001 0               TS       1
04336    4336 0  0,0001 0               TC       Q


                        ;-----------------------------------------------------------------------
                        ; TPA_FIXP
                        ; Reconcile the signs in a double precision word. The most significant word
                        ; is in C(A), the lesser word in C(A+1). Reconcilliation occurs by borrowing
                        ; from C(A) and adding the borrowed amount to C(A+1). C(A) is assumed to be
                        ; positive non-zero number and C(A+1) negative non-zero. The reconciliation
                        ; makes both numbers positive.
                        ;
                        ; This is part of my implementation of TPAGREE.
                        ;-----------------------------------------------------------------------

                        TPA_FIXP       EQU      *
04337    4337 5  0,0576 0               TS       ADDRWD1

04340    4340 2  0,0576 1               INDEX    ADDRWD1
04341    4341 1  0,0000 0               CCS      0              ; borrow from 1st word
04342    4342 2  0,0576 1               INDEX    ADDRWD1
04343    4343 5  0,0000 1               TS       0

04344    4344 3  2,4315 1               CAF      MAXPOS
04345    4345 6  1,2051 1               AD       ONE            ; create positive overflow
04346    4346 2  0,0576 1               INDEX    ADDRWD1
04347    4347 6  0,0001 0               AD       1              ; correct 2nd word, changes sign
04350    4350 2  0,0576 1               INDEX    ADDRWD1
04351    4351 5  0,0001 0               TS       1
04352    4352 0  0,0001 0               TC       Q


                        ;-----------------------------------------------------------------------
                        ; SHORTMP -- MULTIPLY DOUBLE WORD BY A SINGLE WORD
                        ; Multiply C(MPAC, MPAC+1) by the contents of A. Put the product in MPAC,
                        ; MPAC+1, MPAC+2.
                        ;;
                        ; These aren't included in my partial COLOSSUS listing, so I had to invent
                        ; my own version.
                        ;-----------------------------------------------------------------------

                        SHORTMP        EQU      *
04353    4353 5  0,0573 0               TS       SHORTMP_A

                        ; MPAC+2 = MPAC+1 * A

04354    4354 2  0,0000 1               EXTEND
04355    4355 4  0,0131 0               MP       MPAC+1
04356    4356 5  0,0574 1               TS       SHORTMP_OVFL
04357    4357 3  0,0003 1               XCH      LP
04360    4360 5  0,0132 1               TS       MPAC+2

                        ; MPAC+1 = (MPAC * A) + overflow

04361    4361 3  0,0573 0               XCH      SHORTMP_A
04362    4362 2  0,0000 1               EXTEND
04363    4363 4  0,0130 1               MP       MPAC
04364    4364 5  0,0575 0               TS       SHORTMP_OVFH
04365    4365 3  0,0003 1               XCH      LP
04366    4366 6  0,0574 1               AD       SHORTMP_OVFL
04367    4367 5  0,0131 1               TS       MPAC+1         ; skip on overflow
04370    4370 3  1,2050 0               CAF      ZERO           ; otherwise, make interword carry=0

                        ; MPAC = overflow

04371    4371 6  0,0575 0               AD       SHORTMP_OVFH
04372    4372 5  0,0130 0               TS       MPAC

04373    4373 0  0,0001 0               TC       Q              ; return


                        ;-----------------------------------------------------------------------
                        ; DMP -- DOUBLE PRECISION MULTIPLY
                        ; Multiply val, val+1 with C(MPAC, MPAC+1). 'ADDRWD1' contains the
                        ; address of 'val'. The product appears in MPAC, MPAC+1, MPAC+2
                        ;
                        ; This isn't included in my partial COLOSSUS listing, but is taken from
```

```
                        ; the double precision math examples in R-393.
                        ;-----------------------------------------------------------------------------

                        DMP             EQU     *
04374    4374 2  0,0001 1               INDEX   Q
04375    4375 3  0,0000 1               CAF     0
04376    4376 6  2,5777 0               AD      EXTENDER
04377    4377 5  0,0576 0               TS      ADDRWD1

04400    4400 3  0,0131 1               XCH     MPAC+1
04401    4401 5  0,0034 0               TS      OVCTR
04402    4402 2  0,0576 1               INDEX   ADDRWD1
04403    4403 4  0,0001 1               MP      1
04404    4404 3  0,0034 0               XCH     OVCTR

04405    4405 2  0,0576 1               INDEX   ADDRWD1
04406    4406 4  0,0000 0               MP      0
04407    4407 3  0,0034 0               XCH     OVCTR
04410    4410 6  0,0003 1               AD      LP
04411    4411 3  0,0130 0               XCH     MPAC
04412    4412 5  0,0132 1               TS      MPAC+2
04413    4413 2  0,0576 1               INDEX   ADDRWD1
04414    4414 4  0,0001 1               MP      1
04415    4415 3  0,0034 0               XCH     OVCTR
04416    4416 3  0,0130 0               XCH     MPAC
04417    4417 6  0,0003 1               AD      LP
04420    4420 3  0,0132 1               XCH     MPAC+2

04421    4421 2  0,0576 1               INDEX   ADDRWD1
04422    4422 4  0,0000 0               MP      0
04423    4423 3  0,0034 0               XCH     OVCTR
04424    4424 6  0,0130 0               AD      MPAC
04425    4425 6  0,0003 1               AD      LP
04426    4426 3  0,0131 1               XCH     MPAC+1
04427    4427 3  0,0034 0               XCH     OVCTR
04430    4430 5  0,0130 0               TS      MPAC

04431    4431 3  0,0001 0               XCH     Q               ; skip next word on return
04432    4432 6  1,2051 1               AD      ONE
04433    4433 5  0,0001 0               TS      Q
04434    4434 0  0,0001 0               TC      Q


                        BANKFF_1        EQU     *


                        ;-----------------------------------------------------------------------------
                        ; PINBALL
                        ;
                        ; Now, do the "pinball game" (DSKY) routines.
                        ;
                        ; Mimic the bank assignments in COLOSSUS. Since this is a block I AGC that
                        ; has fewer banks, different bank numbers are used, but the sequence and
                        ; relative allocation of routines to various banks is preserved.
                        ;-----------------------------------------------------------------------------

                        ; don't change BANK04_1 without also changing V37BANK

                        BANK04_1        EQU     BANK4           ; was BANK 04 in COLOSSUS

                        BANK40_1        EQU     BANK5           ; was BANK 40 in COLOSSUS
                        BANK41_1        EQU     BANK6           ; was BANK 41 in COLOSSUS
                        BANK42_1        EQU     BANK7           ; was BANK 42 in COLOSSUS
                        BANK43_1        EQU     BANK10          ; was BANK 43 in COLOSSUS

                        ; start of COLOSSUS routines

                                        ORG     BANK04_1        ; COLOSSUS pp. 192-204
                                        INCL    bank04_1.asm
                        ;=============================================================================
                        ; MAJOR MODE CHANGE (file:bank04_1.asm)
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, pp. 192-204.
                        ;=============================================================================

                        ;-----------------------------------------------------------------------------
                        ; VERB 37
                        ;
                        ; In COLOSSUS, a successful V37 apparently also restarts the AGC. Here,
                        ; we implement a subset of COLOSSUS to kick off a job associated with the
                        ; verb.
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.192-204.
                        ;-----------------------------------------------------------------------------
```

```
                        V37           EQU      *              ; verb 37
10000  4,0000 5  0,0464 1              TS       MMNUMBER       ; save major mode

                        ; ** skipped quite a bit of guidance system-related code **

10001  4,0001 0  4,6005 1              TC       CHECKTAB

                        ; ** skipped more guidance system-related code **

                        V37BAD        EQU      *
10002  4,0002 0  2,5003 1              TC       RELDSP         ; releases display from astronaut
10003  4,0003 0  1,3653 1              TC       POSTJUMP       ; bring back last normal display if there
10004  4,0004     05067 0              CADR     PINBRNCH       ; was one, OY

                        ; Search table for entry matching major mode number. Table entries
                        ; are sorted by major mode number, so the search occurs in order from
                        ; lowest number to highest.

                        CHECKTAB      EQU      *
10005  4,0005 3  1,2050 0              CAF      ZERO           ; was CA NOV37MM in Block II
10006  4,0006 6  4,6046 0              AD       NOV37MM        ; the no. of MM in table (minus 1)

10007  4,0007 5  0,0131 1 AGAINMM      TS       MPAC+1

10010  4,0010 3  1,2050 0              CAF      ZERO           ; was CA PREMM1 in Block II
10011  4,0011 2  0,0131 0              INDEX    MPAC+1
10012  4,0012 6  4,6037 0              AD       PREMM1         ; obtain which MM this is for
10013  4,0013 7  1,2101 1              MASK     LOW7
10014  4,0014 4  0,0000 0              COM
10015  4,0015 6  0,0464 1              AD       MMNUMBER
10016  4,0016 1  0,0000 0              CCS      A              ; MMNUMBER - current table MM number

10017  4,0017 1  0,0131 0              CCS      MPAC+1         ; if GR, see if anymore in list
10020  4,0020 0  4,6007 0              TC       AGAINMM        ; yes, get next one (was TCF)
10021  4,0021 0  4,6026 0              TC       V37NONO        ; last time or passed MM (was TCF)

                        ; Found the index into the major mode table for entry matching the
                        ; major mode number input by the user.

10022  4,0022 3  1,2050 0              CAF      ZERO           ; was CA MPAC+1 in Block II
10023  4,0023 6  0,0131 1              AD       MPAC+1
10024  4,0024 5  0,0463 0              TS       MINDEX         ; save index for later

10025  4,0025 0  1,2147 1              TC       goMMchange     ; in Block II, jumped to restart AGC

                        ; Requested MM doesn't exist

                        V37NONO       EQU      *
10026  4,0026 0  2,4701 0              TC       FALTON         ; come here if MM requested doesn't exist
10027  4,0027 0  4,6002 0              TC       V37BAD



                        ;-----------------------------------------------------------------------------
                        ; FCADRMM
                        ;
                        ; For verb 37, two tables are maintained. Each table has an entry for each
                        ; major mode that can be started from the keyboard. The entries are put
                        ; into the table with the entry for the highest major mode coming first,
                        ; to the lowest major mode which is the last entry in the table.
                        ;
                        ; The FCADRMM table contains the FCADR of the starting job of the major mode.
                        ;
                        ; The entries in this table must match the entries in PREMM1 below.
                        ;-----------------------------------------------------------------------------

                        FCADRMM1      EQU      *
10030  4,0030     22147 0              CADR     P79
10031  4,0031     22142 0              CADR     P78
10032  4,0032     22127 0              CADR     P04
10033  4,0033     22066 1              CADR     P03
10034  4,0034     22036 1              CADR     P02
10035  4,0035     22022 1              CADR     P01
10036  4,0036     22000 1              CADR     P00
                        ; etc *********

                        ;-----------------------------------------------------------------------------
                        ; PREMM1
                        ;
                        ; The PREMM1 table contains the E-bank, major mode, and priority information.
                        ; It is in the following form:
                        ;
                        ; PPP PPE EEM MMM MMM
                        ;
                        ; Where,
                        ;       the 7 'M' bits contain the major mode number
```

```
                        ;       the 3 'E' bits contain the E-bank number (ignored in Block I)
                        ;       the 5 'P' bits contain the priority at which the job is to be started
                        ;
                        ; The entries in this table must match the entries in FCADRMM1 above.
                        ;-----------------------------------------------------------------------------

                        PREMM1          EQU     *
10037  4,0037     26117 1               DS      %26117          ; MM 79, PRIO 13
10040  4,0040     26116 0               DS      %26116          ; MM 78, PRIO 13
10041  4,0041     26004 1               DS      %26004          ; MM 04, PRIO 13
10042  4,0042     26003 0               DS      %26003          ; MM 03, PRIO 13
10043  4,0043     26002 1               DS      %26002          ; MM 02, PRIO 13
10044  4,0044     26001 1               DS      %26001          ; MM 01, PRIO 13
10045  4,0045     26000 0               DS      %26000          ; MM 00, PRIO 13
                        ; etc *********
                        EPREMM1         EQU     *

10046  4,0046     00006 1 NOV37MM       DS      EPREMM1-PREMM1-1 ; number of entries in table (minus 1)

                        BANK04_2        EQU     *

                                        ORG     BANK40_1        ; COLOSSUS pp. 310-317
                                        INCL    bank40_1.asm
                        ;=============================================================================
                        ; PINBALL GAME (file:bank40_1.asm)
                        ;
                        ; AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.310-317.
                        ;=============================================================================

                        ;-----------------------------------------------------------------------------
                        ; CHARIN -- PROCESS KEYBOARD CHARACTER FROM KEYRUPT
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.310.
                        ;-----------------------------------------------------------------------------

                        CHARIN          EQU     *
12000  5,0000 3  1,2051 1               CAF     ONE             ; block display syst
12001  5,0001 3  0,0501 0               XCH     DSPLOCK         ; make dsp syst busy, but save old
12002  5,0002 5  0,0412 0               TS      _2122REG        ; C(DSPLOCK) for error light reset
12003  5,0003 1  0,0531 1               CCS     CADRSTOR        ; all keys except ER turn on KR lite if
12004  5,0004 0  5,6006 1               TC      *+2             ; CADRSTOR is full. This reminds operator
12005  5,0005 0  5,6016 0               TC      CHARIN2         ; to re-establish a flashing display
12006  5,0006 4  5,6062 1               CS      ELRCODE1        ; which he has obscured with displays of
12007  5,0007 6  0,0130 0               AD      MPAC            ; his own (see remarks preceding routine
                        ; VBRELDSP).

12010  5,0010 1  0,0000 0               CCS     A               ; was BZF CHARIN2 in Block II
12011  5,0011 0  5,6015 0               TC      *+4             ; >0
12012  5,0012 0  5,6014 1               TC      *+2             ; +0
12013  5,0013 0  5,6015 0               TC      *+2             ; <0
12014  5,0014 0  5,6016 0               TC      CHARIN2         ; -0

12015  5,0015 0  2,4713 0               TC      RELDSPON

                        CHARIN2         EQU     *
12016  5,0016 3  0,0130 0               XCH     MPAC
12017  5,0017 5  0,0414 0               TS      CHAR
12020  5,0020 2  0,0000 0               INDEX   A
12021  5,0021 0  5,6022 1               TC      *+1             ; input_code function
12022  5,0022 0  5,7307 1               TC      CHARALRM        ; 0
12023  5,0023 0  5,6101 1               TC      NUM             ; 1
12024  5,0024 0  5,6101 1               TC      NUM             ; 2
12025  5,0025 0  5,6101 1               TC      NUM             ; 3
12026  5,0026 0  5,6101 1               TC      NUM             ; 4
12027  5,0027 0  5,6101 1               TC      NUM             ; 5
12030  5,0030 0  5,6101 1               TC      NUM             ; 6
12031  5,0031 0  5,6101 1               TC      NUM             ; 7
12032  5,0032 0  5,6065 1               TC      _89TEST
12033  5,0033 0  5,6065 1               TC      _89TEST         ; 11                9
12034  5,0034 0  5,7307 1               TC      CHARALRM        ; 12
12035  5,0035 0  5,7307 1               TC      CHARALRM        ; 13
12036  5,0036 0  5,7307 1               TC      CHARALRM        ; 14
12037  5,0037 0  5,7307 1               TC      CHARALRM        ; 15
12040  5,0040 0  5,7307 1               TC      CHARALRM        ; 16
12041  5,0041 0  5,7307 1               TC      CHARALRM        ; 17
12042  5,0042 0  5,6077 1               TC      NUM-2           ; 20                0
12043  5,0043 0  5,6272 0               TC      VERB            ; 21                VERB
12044  5,0044 0  5,7462 0               TC      ERROR           ; 22                ERROR LIGHT RESET
12045  5,0045 0  5,7307 1               TC      CHARALRM        ; 23
12046  5,0046 0  5,7307 1               TC      CHARALRM        ; 24
12047  5,0047 0  5,7307 1               TC      CHARALRM        ; 25
12050  5,0050 0  5,7307 1               TC      CHARALRM        ; 26
12051  5,0051 0  5,7307 1               TC      CHARALRM        ; 27
12052  5,0052 0  5,7307 1               TC      CHARALRM        ; 30
12053  5,0053 0  5,7327 0               TC      VBRELDSP        ; 31                KEY RELEASE
```

```
12054  5,0054 0  5,6326 0                TC       POSGN          ; 32                +
12055  5,0055 0  5,6312 1                TC       NEGSGN         ; 33                -
12056  5,0056 0  5,6063 1                TC       ENTERJMP       ; 34                ENTER
12057  5,0057 0  5,7307 1                TC       CHARALRM       ; 35
12060  5,0060 0  5,6412 0                TC       CLEAR          ; 36                CLEAR
12061  5,0061 0  5,6306 1                TC       NOUN           ; 37                NOUN

12062  5,0062     00022 1 ELRCODE1       DS       %22
12063  5,0063 0  1,3653 1 ENTERJMP       TC       POSTJUMP
12064  5,0064     14002 0                DS       ENTER

                          _89TEST        EQU      *
12065  5,0065 1  0,0466 1                CCS      DSPCOUNT
12066  5,0066 0  5,6072 1                TC       *+4            ; >0
12067  5,0067 0  5,6072 1                TC       *+3            ; +0
12070  5,0070 0  1,2723 0                TC       ENDOFJOB       ; <0, block data in if DSPCOUNT is <0 or -0
12071  5,0071 0  1,2723 0                TC       ENDOFJOB       ; -0

12072  5,0072 3  1,2053 0                CAF      THREE
12073  5,0073 7  0,0467 0                MASK     DECBRNCH
12074  5,0074 1  0,0000 0                CCS      A
12075  5,0075 0  5,6101 1                TC       NUM            ; if DECBRNCH is +, 8 or 9 OK
12076  5,0076 0  5,7307 1                TC       CHARALRM       ; if DECBRNCH is +0, reject 8 or 9

                          ;-----------------------------------------------------------------------
                          ; NUM -- PROCESS NUMERICAL KEYBOARD CHARACTER
                          ; Assembles octal, 3 bits at a time. For decimal, it converts incoming word
                          ; as a fraction, keeping results to DP (double precision).
                          ; Octal results are left in XREG, YREG, or ZREG. High part of DEC in XREG,
                          ; YREG, ZREG; the low parts in XREGLP, YREGLP, or ZREGLP).
                          ; DECBRNCH is left at +0 for octal, +1 for +DEC, +2 for -DEC.
                          ; If DSPCOUNT was left -, no more data is accepted.
                          ;
                          ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                          ; Oct 28, 1968, p.311.
                          ;-----------------------------------------------------------------------

12077  5,0077 3  1,2050 0                CAF      ZERO
12100  5,0100 5  0,0414 0                TS       CHAR

                          NUM            EQU      *
12101  5,0101 1  0,0466 1                CCS      DSPCOUNT
12102  5,0102 0  5,6106 0                TC       *+4            ; >0
12103  5,0103 0  5,6106 0                TC       *+3            ; +0
12104  5,0104 0  5,6105 0                TC       *+1            ; <0, block datain if DSPCOUNT is <0 or -0
12105  5,0105 0  1,2723 0                TC       ENDOFJOB       ; -0
12106  5,0106 0  5,6241 0                TC       GETINREL
12107  5,0107 1  0,0504 1                CCS      CLPASS         ; if CLPASS is >0 or +0, make it +0
12110  5,0110 3  1,2050 0                CAF      ZERO
12111  5,0111 5  0,0504 0                TS       CLPASS
12112  5,0112 0  5,6113 1                TC       *+1
12113  5,0113 2  0,0414 1                INDEX    CHAR
12114  5,0114 3  1,3772 0                CAF      RELTAB
12115  5,0115 7  2,4664 0                MASK     LOW5
12116  5,0116 5  0,0421 0                TS       CODE
12117  5,0117 3  1,2050 0                CAF      ZERO           ; was CA DSPCOUNT in Block II
12120  5,0120 6  0,0466 0                AD       DSPCOUNT
12121  5,0121 5  0,0440 1                TS       COUNT
12122  5,0122 0  5,7161 0                TC       DSPIN
12123  5,0123 3  1,2053 0                CAF      THREE
12124  5,0124 7  0,0467 0                MASK     DECBRNCH
12125  5,0125 1  0,0000 0                CCS      A              ; +0=octal, +1=+dec, +2=-dec
12126  5,0126 0  5,6137 1                TC       DECTOBIN       ; >0
12127  5,0127 2  0,0434 0                INDEX    INREL          ; +0 (octal)
12130  5,0130 3  0,0470 1                XCH      VERBREG
12131  5,0131 5  0,0022 1                TS       CYL
12132  5,0132 4  0,0022 0                CS       CYL
12133  5,0133 4  0,0022 0                CS       CYL
12134  5,0134 3  0,0022 1                XCH      CYL
12135  5,0135 6  0,0414 0                AD       CHAR
12136  5,0136 0  5,6155 0                TC       ENDNMTST

                          DECTOBIN       EQU      *
12137  5,0137 2  0,0434 0                INDEX    INREL
12140  5,0140 3  0,0470 1                XCH      VERBREG
12141  5,0141 5  0,0130 0                TS       MPAC           ; sum x 2EXP-14 in MPAC
12142  5,0142 3  1,2050 0                CAF      ZERO
12143  5,0143 5  0,0131 1                TS       MPAC+1
12144  5,0144 3  1,2060 0                CAF      TEN            ; 10 x 3EXP-14
12145  5,0145 0  2,4353 0                TC       SHORTMP        ; 10SUM x 2EXP-28 in MPAC, MPAC+1

12146  5,0146 3  0,0131 1                XCH      MPAC+1
12147  5,0147 6  0,0414 0                AD       CHAR
12150  5,0150 5  0,0131 1                TS       MPAC+1
12151  5,0151 0  5,6155 0                TC       ENDNMTST       ; no overflow

12152  5,0152 6  0,0130 0                AD       MPAC           ; overflow, must be 5th character
```

```
12153  5,0153 5  0,0130 0                TS       MPAC

12154  5,0154 0  5,6176 1                TC       DECEND

                         ENDNMTST        EQU      *
12155  5,0155 2  0,0434 0                INDEX    INREL
12156  5,0156 5  0,0470 1                TS       VERBREG
12157  5,0157 4  0,0466 1                CS       DSPCOUNT
12160  5,0160 2  0,0434 0                INDEX    INREL
12161  5,0161 6  5,6232 1                AD       CRITCON

12162  5,0162 1  0,0000 0                CCS      A              ; was BZF ENDNUM in Block II
12163  5,0163 0  5,6167 1                TC       *+4            ; >0
12164  5,0164 0  5,6166 0                TC       *+2            ; +0, DSPCOUNT = CRITCON
12165  5,0165 0  5,6167 1                TC       *+2            ; <0
12166  5,0166 0  5,6170 1                TC       ENDNUM         ; -0

12167  5,0167 0  5,6227 0                TC       MORNUM         ; - , DSPCOUNT G/ CRITCON

                         ENDNUM          EQU      *
12170  5,0170 3  1,2053 0                CAF      THREE
12171  5,0171 7  0,0467 0                MASK     DECBRNCH
12172  5,0172 1  0,0000 0                CCS      A
12173  5,0173 0  5,6176 1                TC       DECEND

                         ENDALL          EQU      *
12174  5,0174 4  0,0466 1                CS       DSPCOUNT       ; block NUMIN by placing DSPCOUNT
12175  5,0175 0  5,6230 0                TC       MORNUM+1       ; negatively



                         DECEND          EQU      *
12176  5,0176 4  1,2051 0                CS       ONE
12177  5,0177 6  0,0434 1                AD       INREL

12200  5,0200 1  0,0000 0                CCS      A              ; was BZMF ENDALL in Block II
12201  5,0201 0  5,6205 0                TC       *+4            ; >0
12202  5,0202 0  5,6204 1                TC       *+2            ; +0, INREL=0,1(VBREG,NNREG), leave whole
12203  5,0203 0  5,6204 1                TC       *+1            ; <0, INREL=0,1(VBREG,NNREG), leave whole
12204  5,0204 0  5,6174 0                TC       ENDALL         ; -0, INREL=0,1(VBREG,NNREG), leave whole

12205  5,0205 0  2,4374 0                TC       DMP            ; if INREL=2,3,4(R1,R2,R3), convert to frac
                         ; mult sum x2EXP-28 in MPAC, MPAC+1 by
12206  5,0206     06237 1                ADRES    DECON          ; 2EXP14/10EPX5. Gives(sum/10EXP5)x2EXP-14
                         ; in MPAC, +1, +2
12207  5,0207 3  1,2053 0                CAF      THREE
12210  5,0210 7  0,0467 0                MASK     DECBRNCH
12211  5,0211 2  0,0000 0                INDEX    A
12212  5,0212 0  5,6212 0                TC       *+0
12213  5,0213 0  5,6220 1                TC       PDECSGN

12214  5,0214 4  0,0131 0                CS       MPAC+1         ; - case (was DCS, DXCH in Block II)
12215  5,0215 5  0,0131 1                TS       MPAC+1
12216  5,0216 4  0,0132 0                CS       MPAC+2
12217  5,0217 5  0,0132 1                TS       MPAC+2

                         PDECSGN         EQU      *
12220  5,0220 3  0,0132 1                XCH      MPAC+2
12221  5,0221 2  0,0434 0                INDEX    INREL
12222  5,0222 5  0,0473 1                TS       XREGLP-2
12223  5,0223 3  0,0131 1                XCH      MPAC+1
12224  5,0224 2  0,0434 0                INDEX    INREL
12225  5,0225 5  0,0470 1                TS       VERBREG
12226  5,0226 0  5,6174 0                TC       ENDALL

                         MORNUM          EQU      *
12227  5,0227 1  0,0466 1                CCS      DSPCOUNT       ; decrement DSPCOUNT
12230  5,0230 5  0,0466 0                TS       DSPCOUNT
12231  5,0231 0  1,2723 0                TC       ENDOFJOB

                         CRITCON         EQU      *
12232  5,0232     00022 1                DS       %22            ; dec 18
12233  5,0233     00020 0                DS       %20            ; dec 16
12234  5,0234     00012 1                DS       %12            ; dec 10
12235  5,0235     00005 1                DS       %5
12236  5,0236     00000 1                DS       %0

                         DECON           EQU      *
12237  5,0237     05174 0                DS       %05174         ; 2EXP14/10EXP5 = .16384 DEC
12240  5,0240     13261 0                DS       %13261

                         ;---------------------------------------------------------------------
                         ; GETINREL
                         ; Gets proper data register relative address for current C(DSPCOUNT) and
                         ; puts into INREL: +0 VERBREG, 1 NOUNREG, 2 XREG, 3 YREG, 4 ZREG
                         ;
```

```
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.313.
                        ;-----------------------------------------------------------------------------

                        GETINREL        EQU     *
12241  5,0241 2  0,0466 1               INDEX   DSPCOUNT
12242  5,0242 3  5,6245 1               CAF     INRELTAB
12243  5,0243 5  0,0434 1               TS      INREL           ; (A TEMP, REG)
12244  5,0244 0  0,0001 0               TC      Q

                        INRELTAB        EQU     *
12245  5,0245     00004 0               DS      %4              ; R3D5, 0 = DSPCOUNT
12246  5,0246     00004 0               DS      %4              ; R3D4, 1
12247  5,0247     00004 0               DS      %4              ; R3D3, 2
12250  5,0250     00004 0               DS      %4              ; R3D2, 3
12251  5,0251     00004 0               DS      %4              ; R3D1, 4
12252  5,0252     00003 1               DS      %3              ; R2D5, 5
12253  5,0253     00003 1               DS      %3              ; R2D4, 6
12254  5,0254     00003 1               DS      %3              ; R2D3, 7
12255  5,0255     00003 1               DS      %3              ; R2D2, 8D
12256  5,0256     00003 1               DS      %3              ; R2D1, 9D
12257  5,0257     00002 0               DS      %2              ; R1D5, 10D
12260  5,0260     00002 0               DS      %2              ; R1D4, 11D
12261  5,0261     00002 0               DS      %2              ; R1D3, 12D
12262  5,0262     00002 0               DS      %2              ; R1D2, 13D
12263  5,0263     00002 0               DS      %2              ; R1D1, 14D
12264  5,0264 0  5,6271 0               TC      CCSHOLE         ; no DSPCOUNT numbers
12265  5,0265     00001 0               DS      %1              ; ND2, 16D
12266  5,0266     00001 0               DS      %1              ; ND1, 17D
12267  5,0267     00000 1               DS      %0              ; VD2, 18D
12270  5,0270     00000 1               DS      %0              ; VD1, 19D


12271  5,0271 0  1,2723 0 CCSHOLE        TC      ENDOFJOB        ; can't find this anywhere; best guess


                        ;-----------------------------------------------------------------------------
                        ; VERB
                        ; Verb key was pressed; prepare to enter a 2 decimal digit verb.
                        ; Blank the verb display and call ENDOFJOB.
                        ;
                        ; NOUN
                        ; Noun key was pressed; prepare to enter a 2 decimal digit noun.
                        ; Blank the noun display and call ENDOFJOB.
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.314.
                        ;-----------------------------------------------------------------------------

                        VERB            EQU     *
12272  5,0272 3  1,2050 0               CAF     ZERO
12273  5,0273 5  0,0470 1               TS      VERBREG
12274  5,0274 3  2,4675 1               CAF     VD1

                        NVCOM           EQU     *
12275  5,0275 5  0,0466 0               TS      DSPCOUNT
12276  5,0276 0  5,6540 0               TC      _2BLANK
12277  5,0277 3  1,2051 1               CAF     ONE
12300  5,0300 5  0,0467 1               TS      DECBRNCH        ; set for dec V/N code
12301  5,0301 3  1,2050 0               CAF     ZERO
12302  5,0302 5  0,0502 0               TS      REQRET          ; set for ENTPAS0
12303  5,0303 3  2,4553 0               CAF     ENDINST         ; if DSPALARM occurs before first ENTPAS0
12304  5,0304 5  0,0433 0               TS      ENTRET          ; or NVSUB, ENTRET must already be set
                        ; to TC ENDOFJOB
12305  5,0305 0  1,2723 0               TC      ENDOFJOB

                        NOUN            EQU     *
12306  5,0306 3  1,2050 0               CAF     ZERO
12307  5,0307 5  0,0471 0               TS      NOUNREG
12310  5,0310 3  2,4676 1               CAF     ND1             ; ND1, OCT 21 (DEC 17)
12311  5,0311 0  5,6275 1               TC      NVCOM


                        ;-----------------------------------------------------------------------------
                        ; NEGSGN
                        ; Turn the minus sign on for the register selected by DSPCOUNT.
                        ; Call ENDOFJOB when done.
                        ;
                        ; POSGN
                        ; Turn the plus sign on for the register selected by DSPCOUNT.
                        ; Call ENDOFJOB when done.
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.314.
                        ;-----------------------------------------------------------------------------

                        NEGSGN          EQU     *
```

```
12312  5,0312 0  5,6367 0                TC       SIGNTEST
12313  5,0313 0  5,6353 1                TC       M_ON
12314  5,0314 3  1,2052 1                CAF      TWO

                         BOTHSGN         EQU      *
12315  5,0315 2  0,0434 0                INDEX    INREL           ; set DEC compu bit to 1 (in DECBRNCH)
12316  5,0316 6  1,2072 0                AD       BIT7            ; Bit 5 for R1, bit 4 for R2, bit 3 for R3
12317  5,0317 6  0,0467 1                AD       DECBRNCH
12320  5,0320 5  0,0467 1                TS       DECBRNCH

                         PIXCLPAS        EQU      *
12321  5,0321 1  0,0504 1                CCS      CLPASS          ; if CLPASS is + or +0, make it +0
12322  5,0322 3  1,2050 0                CAF      ZERO
12323  5,0323 5  0,0504 0                TS       CLPASS
12324  5,0324 0  5,6325 0                TC       *+1
12325  5,0325 0  1,2723 0                TC       ENDOFJOB

                         POSGN           EQU      *
12326  5,0326 0  5,6367 0                TC       SIGNTEST
12327  5,0327 0  5,6332 0                TC       P_ON
12330  5,0330 3  1,2051 1                CAF      ONE
12331  5,0331 0  5,6315 0                TC       BOTHSGN


                         ;-----------------------------------------------------------------------
                         ; P_ON
                         ; Turn the plus sign on for register selected by DSPCOUNT.
                         ; Return when done.
                         ;
                         ; M_ON
                         ; Turn the minus sign on for register selected by DSPCOUNT.
                         ; Return when done.

                         ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                         ; Oct 28, 1968, p.314.
                         ;-----------------------------------------------------------------------

                         P_ON            EQU      *
12332  5,0332 3  0,0001 0                XCH      Q               ; was LXCH Q in block II
12333  5,0333 5  0,0547 1                TS       LXCH_LPRET      ; save return address in faux LP

12334  5,0334 0  5,6241 0                TC       GETINREL
12335  5,0335 2  0,0434 0                INDEX    INREL
12336  5,0336 3  5,6362 0                CAF      SGNTAB-2
12337  5,0337 5  0,0420 1                TS       SGNOFF
12340  5,0340 6  1,2051 1                AD       ONE
12341  5,0341 5  0,0417 0                TS       SGNON

                         SGNCOM          EQU      *
12342  5,0342 3  1,2050 0                CAF      ZERO
12343  5,0343 5  0,0421 0                TS       CODE
12344  5,0344 3  0,0420 1                XCH      SGNOFF
12345  5,0345 0  5,7253 1                TC       _11DSPIN

12346  5,0346 3  1,2066 0                CAF      BIT11
12347  5,0347 5  0,0421 0                TS       CODE
12350  5,0350 3  0,0417 0                XCH      SGNON
12351  5,0351 0  5,7253 1                TC       _11DSPIN

12352  5,0352 0  0,0547 1                TC       LXCH_LPRET      ; return

                         M_ON            EQU      *
12353  5,0353 3  0,0001 0                XCH      Q               ; was LXCH Q in block II
12354  5,0354 5  0,0547 1                TS       LXCH_LPRET      ; save return address in faux LP

12355  5,0355 0  5,6241 0                TC       GETINREL
12356  5,0356 2  0,0434 0                INDEX    INREL
12357  5,0357 3  5,6362 0                CAF      SGNTAB-2
12360  5,0360 5  0,0417 0                TS       SGNON
12361  5,0361 6  1,2051 1                AD       ONE
12362  5,0362 5  0,0420 1                TS       SGNOFF
12363  5,0363 0  5,6342 1                TC       SGNCOM

                         SGNTAB          EQU      *
12364  5,0364     00005 1                DS       %5              ; -R1
12365  5,0365     00003 1                DS       %3              ; -R2
12366  5,0366     00000 1                DS       %0              ; -R3


                         ;-----------------------------------------------------------------------
                         ; SIGNTEST
                         ; Test whether this is a valid point for entering a + or - sign character.
                         ; Returns if valid; calls ENDOFJOB if invalid.
                         ;
                         ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                         ; Oct 28, 1968, p.314.
                         ;-----------------------------------------------------------------------
```

```
                          SIGNTEST       EQU      *
12367  5,0367 3  0,0001 0                XCH      Q              ; was LXCH Q in block II
12370  5,0370 5  0,0547 1                TS       LXCH_LPRET     ; save return address in faux LP

                          ; allows +,- only when DSPCOUNT=R1D1
12371  5,0371 3  1,2053 0                CAF      THREE          ; R2D1, or D3D1. Allows only first of
12372  5,0372 7  0,0467 0                MASK     DECBRNCH       ; consecutive +/- characters.
12373  5,0373 1  0,0000 0                CCS      A              ; if low2 bits of DECBRNCH not=0, sign
12374  5,0374 0  1,2723 0                TC       ENDOFJOB       ; for this word already in, reject.

12375  5,0375 4  2,4635 1                CS       R1D1
12376  5,0376 0  5,6404 1                TC       SGNTST1        ; DSPCOUNT is R1D1?
12377  5,0377 4  2,4636 1                CS       R2D1
12400  5,0400 0  5,6404 1                TC       SGNTST1
12401  5,0401 4  2,4637 0                CS       R3D1
12402  5,0402 0  5,6404 1                TC       SGNTST1
12403  5,0403 0  1,2723 0                TC       ENDOFJOB       ; no match found, sign illegal

                          SGNTST1        EQU      *
12404  5,0404 6  0,0466 0                AD       DSPCOUNT

12405  5,0405 1  0,0000 0                CCS      A              ; was BZF *+2 in Block II
12406  5,0406 0  0,0001 0                TC       Q              ; >0, no match, check next register
12407  5,0407 0  0,0547 1                TC       LXCH_LPRET     ; +0, match found, sign is legal
12410  5,0410 0  0,0001 0                TC       Q              ; <0, no match, check next register
12411  5,0411 0  0,0547 1                TC       LXCH_LPRET     ; -0, match found, sign is legal


                          ;-----------------------------------------------------------------------
                          ; CLEAR -- PROCESS CLEAR KEY
                          ; Clear blanks which R1, R2, R3 is current or last to be displayed (pertinent
                          ; XREG, YREG, ZREG is cleared). Successive clears take care of each RX L/
                          ; RC until R1 is done, then no further action.
                          ;
                          ; The single component load verbs allow only the single RC that is appropriate
                          ; to be cleared.
                          ;
                          ; CLPASS = 0, PASSO, can be backed up
                          ; CLPASS = +NZ, HIPASS, can be backed up
                          ; CLPASS = -NZ, PASSO, cannot be backed up
                          ;
                          ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                          ; Oct 28, 1968, p.316.
                          ;-----------------------------------------------------------------------

                          CLEAR          EQU      *
12412  5,0412 1  0,0466 1                CCS      DSPCOUNT
12413  5,0413 6  1,2051 1                AD       ONE
12414  5,0414 0  5,6416 1                TC       *+2
12415  5,0415 6  1,2051 1                AD       ONE
12416  5,0416 2  0,0000 0                INDEX    A              ; do not change DSPCOUNT because may later
12417  5,0417 3  5,6245 1                CAF      INRELTAB       ; fail LEGALTST
12420  5,0420 5  0,0434 1                TS       INREL          ; must set INREL, even for HIPASS
12421  5,0421 1  0,0504 1                CCS      CLPASS
12422  5,0422 0  5,6431 1                TC       CLPASHI        ; +
12423  5,0423 0  5,6425 1                TC       *+2            ; +0, if CCLPASS is +0 or -, it is PASS0
12424  5,0424 0  5,6425 1                TC       *+1            ; -
12425  5,0425 3  1,2050 0                CAF      ZERO           ; was CA INREL in Block II
12426  5,0426 6  0,0434 1                AD       INREL
12427  5,0427 0  5,6464 1                TC       LEGALTST
12430  5,0430 0  5,6454 1                TC       CLEAR1

                          CLPASHI        EQU      *
12431  5,0431 1  0,0434 0                CCS      INREL
12432  5,0432 5  0,0434 1                TS       INREL
12433  5,0433 0  5,6464 1                TC       LEGALTST

12434  5,0434 3  5,6536 1                CAF      DOUBLK+2       ; +3 to - number, backs data requests
12435  5,0435 6  0,0502 0                AD       REQRET         ; was ADS REQRET in Block II
12436  5,0436 5  0,0502 0                TS       REQRET

12437  5,0437 3  1,2050 0                CAF      ZERO           ; was CA INREL in Block II
12440  5,0440 6  0,0434 1                AD       INREL
12441  5,0441 5  0,0422 0                TS       MIXTEMP        ; temp storage for INREL

12442  5,0442 1  0,0470 0                CCS      VERBREG        ; was DIM VERBREG in Block II
12443  5,0443 0  5,6446 1                TC       *+3
12444  5,0444 0  5,6446 1                TC       *+2
12445  5,0445 0  5,6446 1                TC       *+1
12446  5,0446 5  0,0470 1                TS       VERBREG        ; decrement VERB and redisplay

12447  5,0447 0  1,3565 1                TC       BANKCALL
12450  5,0450     14327 1                DS       UPDATVB

12451  5,0451 3  1,2050 0                CAF      ZERO           ; was CA MIXTEMP in Block II
12452  5,0452 6  0,0422 0                AD       MIXTEMP
```

```
12453  5,0453 5  0,0434 1                TS       INREL          ; restore INREL

                          CLEAR1          EQU      *
12454  5,0454 0  5,6461 1                TC       CLR5

12455  5,0455 3  0,0504 0                XCH      CLPASS         ; was INCR CLPASS in Block II
12456  5,0456 6  1,2051 1                AD       ONE
12457  5,0457 5  0,0504 0                TS       CLPASS         ; only if CLPASS is + or +0

12460  5,0460 0  1,2723 0                TC       ENDOFJOB       ; set for higher pass


                          ;-----------------------------------------------------------------------
                          ; CLR5
                          ; blanks 5 char display word by calling _5BLANK, but avoids TC GETINREL.
                          ; Returns when done.
                          ;
                          ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                          ; Oct 28, 1968, p.316.
                          ;-----------------------------------------------------------------------

                          CLR5            EQU      *
12461  5,0461 3  0,0001 0                XCH      Q              ; was LXCH Q in block II
12462  5,0462 5  0,0547 1                TS       LXCH_LPRET     ; save return address in faux LP
12463  5,0463 0  5,6476 1                TC       _5BLANK+3      ; uses _5BLANK, but avoids its TC GETINREL


                          ;-----------------------------------------------------------------------
                          ; LEGALTST
                          ; Returns if LEGAL, calls ENDOFJOB if illegal.
                          ;
                          ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                          ; Oct 28, 1968, p.316.
                          ;-----------------------------------------------------------------------

                          LEGALTST        EQU      *
12464  5,0464 6  1,2047 0                AD       NEG2
12465  5,0465 1  0,0000 0                CCS      A
12466  5,0466 0  0,0001 0                TC       Q              ; LEGAL, INREL G/ 2
12467  5,0467 0  5,6271 0                TC       CCSHOLE
12470  5,0470 0  1,2723 0                TC       ENDOFJOB       ; ILLEGAL, INREL = 0, 1
12471  5,0471 0  0,0001 0                TC       Q              ; LEGAL, INREL = 2


                          ;-----------------------------------------------------------------------
                          ; _5BLANK
                          ; blanks 5 char display word in R1,R2,or R3. It also zeroes XREG, YREG or
                          ; ZREG. Place any + DSPCOUNT number for pertinent RC into DSPCOUNT.
                          ; DSPCOUNT is left set to left most DSP numb for RC just blanked.
                          ; Returns when done.
                          ;
                          ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                          ; Oct 28, 1968, p.316.
                          ;-----------------------------------------------------------------------

12472  5,0472 5  0,0466 0                TS       DSPCOUNT       ; needed for BLANKSUB

                          _5BLANK         EQU      *
12473  5,0473 3  0,0001 0                XCH      Q              ; was LXCH Q in block II
12474  5,0474 5  0,0547 1                TS       LXCH_LPRET     ; save return address in faux LP

12475  5,0475 0  5,6241 0                TC       GETINREL
12476  5,0476 3  1,2050 0                CAF      ZERO
12477  5,0477 2  0,0434 0                INDEX    INREL
12500  5,0500 5  0,0470 1                TS       VERBREG        ; zero X, Y, Z reg
12501  5,0501 2  0,0434 0                INDEX    INREL
12502  5,0502 5  0,0473 1                TS       XREGLP-2
12503  5,0503 5  0,0421 0                TS       CODE
12504  5,0504 2  0,0434 0                INDEX    INREL          ; zero pertinent DEC comp bit
12505  5,0505 4  1,2072 1                CS       BIT7
12506  5,0506 7  0,0467 0                MASK     DECBRNCH
12507  5,0507 7  5,6537 1                MASK     BRNCHCON       ; zero low 3 bits
12510  5,0510 5  0,0467 1                TS       DECBRNCH
12511  5,0511 2  0,0434 0                INDEX    INREL
12512  5,0512 3  5,6527 1                CAF      SINBLANK-2     ; blank isolated char separately
12513  5,0513 5  0,0440 1                TS       COUNT
12514  5,0514 0  5,7161 0                TC       DSPIN

                          _5BLANK1        EQU      *
12515  5,0515 2  0,0434 0                INDEX    INREL
12516  5,0516 3  5,6532 0                CAF      DOUBLK-2
12517  5,0517 5  0,0466 0                TS       DSPCOUNT
12520  5,0520 0  5,6540 0                TC       _2BLANK

12521  5,0521 4  1,2052 0                CS       TWO
12522  5,0522 6  0,0466 0                AD       DSPCOUNT       ; was ADS DSPCOUNT in Block II
12523  5,0523 5  0,0466 0                TS       DSPCOUNT
```

```
12524  5,0524 0  5,6540 0                TC       _2BLANK
12525  5,0525 2  0,0434 0                INDEX    INREL
12526  5,0526 3  2,4633 0                CAF      R1D1-2
12527  5,0527 5  0,0466 0                TS       DSPCOUNT       ; set DSPCOUNT to leftmost DSP number
12530  5,0530 0  0,0547 1                TC       LXCH_LPRET     ; of REG, just blanked


                          SINBLANK       EQU      *
12531  5,0531     00016 0                DS       %16            ; DEC 14
12532  5,0532     00005 1                DS       %5
12533  5,0533     00004 0                DS       %4

                          DOUBLK         EQU      *
12534  5,0534     00015 0                DS       %15            ; DEC 13
12535  5,0535     00011 1                DS       %11            ; DEC 9
12536  5,0536     00003 1                DS       %3

12537  5,0537     77774 0 BRNCHCON       DS       %77774


                          ;----------------------------------------------------------------------------
                          ; _2BLANK
                          ; blanks 2 char, place DSP number of left char of the pair into DSPCOUNT.
                          ; This number is left in DSPCOUNT. Returns when done.
                          ;
                          ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                          ; Oct 28, 1968, p.317.
                          ;----------------------------------------------------------------------------


                          _2BLANK        EQU      *
12540  5,0540 3  0,0001 0                XCH      Q
12541  5,0541 5  0,0602 0                TS       SAVEQ

12542  5,0542 3  1,2050 0                CAF      ZERO           ; was CA DSPCOUNT in Block II
12543  5,0543 6  0,0466 0                AD       DSPCOUNT
12544  5,0544 5  0,0021 1                TS       SR
12545  5,0545 4  5,6563 0                CS       BLANKCON

12546  5,0546 2  0,0000 0                INHINT
12547  5,0547 2  0,0021 0                INDEX    SR
12550  5,0550 3  0,0512 1                XCH      DSPTAB

12551  5,0551 1  0,0000 0                CCS      A              ; was BZMF *+2 in Block II
12552  5,0552 0  5,6556 1                TC       *+4            ; >0
12553  5,0553 0  5,6555 1                TC       *+2            ; +0, if old contents -, NOUT OK
12554  5,0554 0  5,6555 1                TC       *+1            ; <0, if old contents -, NOUT OK
12555  5,0555 0  5,6557 0                TC       *+2            ; -0, if old contents -, NOUT OK

12556  5,0556 3  0,0505 1                XCH      NOUT           ; was INCR NOUT in Block II
12557  5,0557 6  1,2051 1                AD       ONE
12560  5,0560 5  0,0505 1                TS       NOUT           ; if old contents +, +1 to NOUT
12561  5,0561 2  0,0000 1                RELINT

12562  5,0562 0  0,0602 0                TC       SAVEQ

12563  5,0563     04000 0 BLANKCON       DS       %4000


                          BANK40_2       EQU      *

                                         ORG      BANK41_1       ; COLOSSUS pp. 318-329
                                         INCL     bank41_1.asm
                          ;============================================================================
                          ; DISPLAY ROUTINES (file:bank41_1.asm)
                          ;
                          ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                          ; Oct 28, 1968, pp. 318-329.
                          ;============================================================================

                          ;----------------------------------------------------------------------------
                          ; ENTER -- PROCESS ENTER KEY
                          ; Enter pass 0 is the execute function. Higher order enters are to load
                          ; data. The sign of REQRET determines the pass, + for pass 0, - for higher
                          ; passes.
                          ; Machine CADR to be specified (MCTBS) nouns desire an ECADR to be loaded
                          ; when used with load verbs, monitor verbs, or display verbs (except
                          ; verb = fixed memory display, which requires a FCADR).
                          ;
                          ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                          ; Oct 28, 1968, p.318.
                          ;----------------------------------------------------------------------------

14000  6,0000 0  6,7505 0 NVSUBR         TC       NVSUB1         ; standard lead-ins, don't move
14001  6,0001 0  6,6723 1 LOADLV1        TC       LOADLV
```

```
                          ENTER          EQU      *
14002  6,0002 3  1,2050 0               CAF      ZERO
14003  6,0003 5  0,0504 0               TS       CLPASS
14004  6,0004 3  2,4553 0               CAF      ENDINST
14005  6,0005 5  0,0433 0               TS       ENTRET

14006  6,0006 1  0,0502 1               CCS      REQRET
14007  6,0007 0  6,6040 0               TC       ENTPAS0        ; if +, pass 0
14010  6,0010 0  6,6040 0               TC       ENTPAS0        ; if +, pass 0
14011  6,0011 0  6,6012 1               TC       *+1            ; if -, not pass 0


                          ; not first pass thru ENTER, so enter data word

                          ENTPASHI       EQU      *
14012  6,0012 3  6,6036 1               CAF      MMADREF
14013  6,0013 6  0,0502 0               AD       REQRET         ; if L/2 char in for MM code, alarm

14014  6,0014 1  0,0000 0               CCS      A              ; and recycle (decide at MMCHANG+1)
14015  6,0015 0  6,6021 1               TC       *+4            ; >0
14016  6,0016 0  6,6020 0               TC       *+2            ; +0
14017  6,0017 0  6,6021 1               TC       *+2            ; <0
14020  6,0020 0  6,6032 0               TC       ACCEPTWD       ; -0, was BZF ACCEPTWD in Block II

14021  6,0021 3  1,2053 0               CAF      THREE          ; if DEC, alarm if L/5 char in for data,
14022  6,0022 7  0,0467 0               MASK     DECBRNCH       ; but leave REQRET - and flash on, so
14023  6,0023 1  0,0000 0               CCS      A              ; operator can supply missing numerical
14024  6,0024 0  6,6026 0               TC       *+2            ; characters and continue.
14025  6,0025 0  6,6032 0               TC       ACCEPTWD       ; octal, any number of char OK.
14026  6,0026 1  0,0466 1               CCS      DSPCOUNT
14027  6,0027 0  6,6341 1               TC       GODSPALM       ; less than 5 char DEC(DSPCOUNT is +)
14030  6,0030 0  6,6341 1               TC       GODSPALM       ; less than 5 char DEC(DSPCOUNT is +)
14031  6,0031 0  6,6032 0               TC       *+1            ; 5 char in (DSPCOUNT is -)

                          ACCEPTWD       EQU      *
14032  6,0032 4  0,0502 1               CS       REQRET         ; 5 char in (DSPCOUNT is -)
14033  6,0033 5  0,0502 0               TS       REQRET         ; set REQRET +
14034  6,0034 0  2,4770 0               TC       FLASHOFF
14035  6,0035 0  0,0502 0               TC       REQRET

                          ENTEXIT        EQU      ENTRET

14036  6,0036     15357 1 MMADREF        DS       MMCHANG+1      ; assumes TC REGMM at MMCHANG

14037  6,0037     00034 0 LOWVERB        DS       28             ; lower verb that avoids nount test.


                          ; first pass thru ENTER, so execute VERB/NOUN

                          ENTPAS0        EQU      *
14040  6,0040 3  1,2050 0               CAF      ZERO           ; noun verb sub enters here
14041  6,0041 5  0,0467 1               TS       DECBRNCH
14042  6,0042 4  2,4675 0               CS       VD1            ; block further num char, so that stray
14043  6,0043 5  0,0466 0               TS       DSPCOUNT       ; char do not get into verb or nount lights.

                          ; test VERB

                          TESTVB         EQU      *
14044  6,0044 4  0,0470 0               CS       VERBREG        ; if verb is G/E LOWVB, skip noun test
14045  6,0045 5  0,0530 1               TS       VERBSAVE       ; save verb for possible recycle.
14046  6,0046 6  6,6037 0               AD       LOWVERB        ; LOWVERB - VB

14047  6,0047 1  0,0000 0               CCS      A              ; was BZMF VERBFAN in Block II
14050  6,0050 0  6,6054 0               TC       *+4            ; >0
14051  6,0051 0  6,6053 1               TC       *+2            ; +0, VERB G/E LOWVERB
14052  6,0052 0  6,6053 1               TC       *+1            ; <0, VERB G/E LOWVERB
14053  6,0053 0  6,6151 1               TC       VERBFAN        ; -0, VERB G/E LOWVERB

                          ; test NOUN

                          TESTNN         EQU      *

                          ; set MIXBR and put the noun address into NNADTEM
                          ; MIXBR is an enumerated type:
                          ; 1 = normal nouns
                          ; 2 = mixed nouns

14054  6,0054 3  6,6124 0               CAF      LODNNLOC       ; was DCA LODNNLOC, DXCH Z in Block II
14055  6,0055 0  1,3526 0               TC       DXCHJUMP       ; bank jump to noun table read rtne

14056  6,0056 2  0,0435 1               INDEX    MIXBR          ; computed GOTO
14057  6,0057 0  6,6057 0               TC       *+0

14060  6,0060 0  6,6062 0               TC       *+2            ; returns here for normal noun
14061  6,0061 0  6,6237 1               TC       MIXNOUN        ; returns here for mixed noun
```

```
                        ; normal noun, so test noun address table entry (NNADTEM)

14062  6,0062 1  0,0443 0               CCS      NNADTEM        ; normal
14063  6,0063 0  6,6147 0               TC       VERBFAN-2      ; normal if +
14064  6,0064 0  6,6341 1               TC       GODSPALM       ; not in use if +0
14065  6,0065 0  6,6073 0               TC       REQADD         ; specify machine CADR if -

                        ; NNADTEM was -0, so just increment noun address (in NOUNCADR) and
                        ; set the result in NOUNADD

14066  6,0066 3  0,0506 1               XCH      NOUNCADR       ; augment machine CADR if -0
14067  6,0067 6  1,2051 1               AD       ONE
14070  6,0070 5  0,0506 1               TS       NOUNCADR       ; was INCR NOUNCADR in Block II

14071  6,0071 0  2,4625 1               TC       SETNADD        ; set NOUNADD
14072  6,0072 0  6,6132 1               TC       INTMCTBS+3

                        ; NNADTEM was -, so noun address needs to be specified (loaded).

                        REQADD         EQU      *
14073  6,0073 3  1,2062 1               CAF      BIT15          ; set CLPASS for pass0 only
14074  6,0074 5  0,0504 0               TS       CLPASS
14075  6,0075 4  2,4553 1               CS       ENDINST        ; test if reach here from internal or
14076  6,0076 6  0,0433 0               AD       ENTEXIT        ; from external

14077  6,0077 1  0,0000 0               CCS      A              ; was BZF *+2 in Block II
14100  6,0100 0  6,6104 1               TC       *+4            ; >0
14101  6,0101 0  6,6103 0               TC       *+2            ; +0
14102  6,0102 0  6,6104 1               TC       *+2            ; <0
14103  6,0103 0  6,6105 0               TC       *+2            ; -0, external mach CADR to be specified

14104  6,0104 0  6,6127 0               TC       INTMCTBS

14105  6,0105 0  6,6274 0               TC       REQDATZ        ; external mach CADR to be specified

14106  6,0106 1  0,0467 0               CCS      DECBRNCH       ; alarm and recycle if decimal used
14107  6,0107 0  2,4474 1               TC       ALMCYCLE       ; for MCTBS
14110  6,0110 4  2,4675 0               CS       VD1            ; octal used OK
14111  6,0111 5  0,0466 0               TS       DSPCOUNT       ; block num char in

14112  6,0112 1  0,0531 1               CCS      CADRSTOR
14113  6,0113 0  6,6116 1               TC       *+3            ; external MCTBS display will leave flash
14114  6,0114 0  6,6117 0               TC       USEADD         ; on if ENDIDLE not = +0
14115  6,0115 0  6,6116 1               TC       *+1
14116  6,0116 0  2,4760 1               TC       FLASHON

                        ; noun address has now been loaded into the Z register. Copy it into
                        ; NOUNCADR and NOUNADD and then jump to the VERBFAN.

                        USEADD         EQU      *
14117  6,0117 3  0,0474 0               XCH      ZREG
14120  6,0120 0  2,4616 1               TC       SETNCADR       ; ECADR into NOUNCADR, set EB, NOUNADD

14121  6,0121 3  6,6124 0               CAF      LODNNLOC       ; was DCA LODNNLOC, DXCH Z in Block II
14122  6,0122 0  1,3526 0               TC       DXCHJUMP       ; bank jump to noun table read rtne

14123  6,0123 0  6,6151 1               TC       VERBFAN

14124  6,0124     16114 1 LODNNLOC      DS       LODNNTAB       ; *** uses 2 words in Block II ***
14125  6,0125     00000 1               DS       0




14126  6,0126     77772 0 NEG5          DS       -5

                        ; If external (keyboard input), noun address is in register A.
                        ; If internal (S/W input), noun address is in MPAC+2.
                        ; Store the noun address into NOUNCADR and NOUNADD. If the verb
                        ; is O5. go directly to the VERBFAN; for all other verbs, display
                        ; the noun address in R3 and then go to the VERBFAN.

                        INTMCTBS       EQU      *

                        ; entry point for internal:

14127  6,0127 3  1,2050 0               CAF      ZERO           ; was CA MPAC+2 in Block II
14130  6,0130 6  0,0132 1               AD       MPAC+2         ; internal mach CADR to be specified

                        ; entry point for external (keyboard input):

14131  6,0131 0  2,4616 1               TC       SETNCADR       ; store addr (A) into NOUNCADR and NOUNADD

14132  6,0132 4  1,2055 1               CS       FIVE           ; NVSUB call left CADR in MAPC+2 for mach
14133  6,0133 6  0,0470 1               AD       VERBREG        ; CADR to be specified.
```

```
14134  6,0134 1  0,0000 0                CCS      A              ; was BZF VERBFAN in Block II
14135  6,0135 0  6,6141 0                TC       *+4            ; >0
14136  6,0136 0  6,6140 1                TC       *+2            ; +0
14137  6,0137 0  6,6141 0                TC       *+2            ; <0
14140  6,0140 0  6,6151 1                TC       VERBFAN        ; -0, don't display CADR if verb = 05

14141  6,0141 3  2,4637 1                CAF      R3D1           ; verb not = 05, display CADR
14142  6,0142 5  0,0466 0                TS       DSPCOUNT

14143  6,0143 3  1,2050 0                CAF      ZERO           ; was CA NOUNCADR in Block II
14144  6,0144 6  0,0506 1                AD       NOUNCADR
14145  6,0145 0  6,7310 1                TC       DSPOCTWD
14146  6,0146 0  6,6151 1                TC       VERBFAN

                         ; NNADTEM was + (normal), so just use the noun address straight from the
                         ; noun table (currently in A). The CCS instruction used to test the
                         ; address also decremented it, so we add one to restore the correct address.

14147  6,0147 6  1,2051 1                AD       ONE
14150  6,0150 0  2,4616 1                TC       SETNCADR       ; store addr (A) into NOUNCADR and NOUNADD


                         ; noun address is currently in NOUNCADR and NOUNADD.

                         VERBFAN        EQU      *
14151  6,0151 4  6,6163 1                CS       LST2CON
14152  6,0152 6  0,0470 1                AD       VERBREG        ; verb-LST2CON

14153  6,0153 1  0,0000 0                CCS      A
14154  6,0154 6  1,2051 1                AD       ONE            ; ver G/ LST2CON
14155  6,0155 0  6,6157 1                TC       *+2
14156  6,0156 0  6,6164 1                TC       VBFANDIR       ; verb L/ LST2CON
14157  6,0157 5  0,0130 0                TS       MPAC
14160  6,0160 0  2,5003 1                TC       RELDSP         ; release display syst
14161  6,0161 0  1,3653 1                TC       POSTJUMP       ; go to GOEXTVB with VB-40 in MPAC
14162  6,0162     20000 0                DS       GOEXTVB

14163  6,0163     00050 1 LST2CON        DS       40             ; first list2 verb (extended verb)

                         VBFANDIR       EQU      *
14164  6,0164 2  0,0470 0                INDEX    VERBREG
14165  6,0165 3  6,6167 1                CAF      VERBTAB
14166  6,0166 0  1,3712 0                TC       BANKJUMP

                         VERBTAB        EQU      *
14167  6,0167     14341 1                CADR     GODSPALM       ; VB00 Illegal
14170  6,0170     14355 1                CADR     DSPA           ; VB01 display oct comp 1 (R1)
14171  6,0171     14363 1                CADR     DSPB           ; VB02 display oct comp 2 (R1)
14172  6,0172     14370 0                CADR     DSPC           ; VB03 display oct comp 3 (R1)
14173  6,0173     14350 1                CADR     DSPAB          ; VB04 display oct comp 1,2 (R1,R2)
14174  6,0174     14343 0                CADR     DSPABC         ; VB05 display oct comp 1,2,3 (R1,R2,R3)
14175  6,0175     14510 0                CADR     DECDSP         ; VB06 decimal display
14176  6,0176     12704 1                CADR     DSPDPDEC       ; VB07 DP decimal display (R1,R2)
14177  6,0177     14341 1                CADR     GODSPALM       ; VB08 spare
14200  6,0200     14341 1                CADR     GODSPALM       ; VB09 spare
14201  6,0201     14341 1                CADR     GODSPALM       ; VB10 spare
14202  6,0202     15146 0                CADR     MONITOR        ; VB11 monitor oct comp 1 (R1)
14203  6,0203     15146 0                CADR     MONITOR        ; VB12 monitor oct comp 2 (R2)
14204  6,0204     15146 0                CADR     MONITOR        ; VB13 monitor oct comp 3 (R3)
14205  6,0205     15146 0                CADR     MONITOR        ; VB14 monitor oct comp 1,2 (R1,R2)
14206  6,0206     15146 0                CADR     MONITOR        ; VB15 monitor oct comp 1,2,3 (R1,R2,R3)
14207  6,0207     15146 0                CADR     MONITOR        ; VB16 monitor decimal
14210  6,0210     15146 0                CADR     MONITOR        ; VB17 monitor DP decimal (R1,R2)
14211  6,0211     14341 1                CADR     GODSPALM       ; VB18 spare
14212  6,0212     14341 1                CADR     GODSPALM       ; VB19 spare
14213  6,0213     14341 1                CADR     GODSPALM       ; VB20 spare
14214  6,0214     14663 1                CADR     ALOAD          ; VB21 load comp 1 (R1)
14215  6,0215     14673 0                CADR     BLOAD          ; VB22 load comp 2 (R2)
14216  6,0216     14707 1                CADR     CLOAD          ; VB23 load comp 3 (R3)
14217  6,0217     14635 1                CADR     ABLOAD         ; VB24 load comp 1,2 (R1,R2)
14220  6,0220     14600 1                CADR     ABCLOAD        ; VB25 load comp 1,2,3 (R1,R2,R3)
14221  6,0221     14341 1                CADR     GODSPALM       ; VB26 spare
14222  6,0222     15301 1                CADR     DSPFMEM        ; VB27 fixed memory display
14223  6,0223     14341 1                CADR     GODSPALM       ; VB28 spare
14224  6,0224     14341 1                CADR     GODSPALM       ; VB29 spare
14225  6,0225     15420 0                CADR     VBRQEXEC       ; VB30 request executive
14226  6,0226     15446 0                CADR     VBRQWAIT       ; VB31 request waitlist
14227  6,0227     13325 1                CADR     VBRESEQ        ; VB32 resequence
14230  6,0230     13315 1                CADR     VBPROC         ; VB33 proceed (without data)
14231  6,0231     13323 1                CADR     VBTERM         ; VB34 terminate
14232  6,0232     15572 0                CADR     VBTSTLTS       ; VB35 test lights
14233  6,0233     02126 0                CADR     SLAP1          ; VB36 fresh start
14234  6,0234     15356 0                CADR     MMCHANG        ; VB37 change major mode
14235  6,0235     14341 1                CADR     GODSPALM       ; VB38 spare
14236  6,0236     14341 1                CADR     GODSPALM       ; VB39 spare
```

```
                        ;----------------------------------------------------------------------------
                        ; MIXNOUN
                        ; NNADTAB contains a relative address, IDADDREL(in low 10 bits), referring
                        ; to where 3 consecutive addresses are stored (in IDADDTAB).
                        ; MIXNOUN gets data and stores in MIXTEMP, +1, +2. It sets NOUNADD for
                        ; MIXTEMP.
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.322.
                        ;----------------------------------------------------------------------------

                        MIXNOUN         EQU     *
14237  6,0237 0  6,6341 1               TC      GODSPALM        ; not currently implemented

                        ; ************ BUNCH OF MISSING STUFF ************


                        ;----------------------------------------------------------------------------
                        ; DPTEST
                        ; enter with SF routine code number (SF ROUT) in A. Returns to L+1 if no DP.
                        ; Returns to L+2 if DP.
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.322. Also, see p. 263.
                        ;----------------------------------------------------------------------------

                        DPTEST          EQU     *
14240  6,0240 5  0,0552 0               TS      DPTEST_A
14241  6,0241 3  0,0001 0               XCH     Q
14242  6,0242 5  0,0553 1               TS      DPTEST_Q

14243  6,0243 2  0,0552 1               INDEX   DPTEST_A
14244  6,0244 0  6,6245 1               TC      *+1
14245  6,0245 0  0,0553 1               TC      DPTEST_Q        ; octal only, no DP
14246  6,0246 0  0,0553 1               TC      DPTEST_Q        ; straight fractional, no DP
14247  6,0247 0  0,0553 1               TC      DPTEST_Q        ; CDU degrees (XXX.XX), no DP
14250  6,0250 0  0,0553 1               TC      DPTEST_Q        ; arithmetic SF, no DP
14251  6,0251 0  6,6262 1               TC      DPTEST1         ; DP1OUT
14252  6,0252 0  6,6262 1               TC      DPTEST1         ; DP2OUT
14253  6,0253 0  0,0553 1               TC      DPTEST_Q        ; Y OPTICS DEGREES, no DP
14254  6,0254 0  6,6262 1               TC      DPTEST1         ; DP3OUT
14255  6,0255 0  0,0553 1               TC      DPTEST_Q        ; HMS, no DP
14256  6,0256 0  0,0553 1               TC      DPTEST_Q        ; MS, no DP
14257  6,0257 0  6,6262 1               TC      DPTEST1         ; DP4OUT
14260  6,0260 0  0,0553 1               TC      DPTEST_Q        ; arith1, no DP
14261  6,0261 0  0,0553 1               TC      DPTEST_Q        ; 2INTOUT, no DP to get hi part in MPAC

                        DPTEST1         EQU     *
14262  6,0262 2  0,0553 0               INDEX   DPTEST_Q
14263  6,0263 0  0,0001 0               TC      1               ; return to L+2


                        ;----------------------------------------------------------------------------
                        ; REQDATX, REQDATY, REQDATZ
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.323.
                        ;----------------------------------------------------------------------------

                        REQDATX         EQU     *
14264  6,0264 3  0,0001 0               XCH     Q
14265  6,0265 5  0,0554 0               TS      REQ_Q

14266  6,0266 3  2,4635 0               CAF     R1D1
14267  6,0267 0  6,6277 0               TC      REQCOM

                        REQDATY         EQU     *
14270  6,0270 3  0,0001 0               XCH     Q
14271  6,0271 5  0,0554 0               TS      REQ_Q

14272  6,0272 3  2,4636 0               CAF     R2D1
14273  6,0273 0  6,6277 0               TC      REQCOM

                        REQDATZ         EQU     *
14274  6,0274 3  0,0001 0               XCH     Q
14275  6,0275 5  0,0554 0               TS      REQ_Q

14276  6,0276 3  2,4637 1               CAF     R3D1

14277  6,0277 5  0,0466 0 REQCOM        TS      DSPCOUNT
14300  6,0300 4  0,0554 1               CS      REQ_Q
14301  6,0301 5  0,0502 0               TS      REQRET
14302  6,0302 0  1,3565 1               TC      BANKCALL
14303  6,0303     12473 1               DS      _5BLANK

14304  6,0304 0  2,4760 1               TC      FLASHON
```

```
                                ENDRQDAT        EQU     *
14305  6,0305 0  0,0433 0                      TC      ENTEXIT


                                ;----------------------------------------------------------------------------
                                ; UPDATNN, UPDATVB
                                ;
                                ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                                ; Oct 28, 1968, p.323.
                                ;----------------------------------------------------------------------------

14306  6,0306 5  0,0471 0                      TS      NOUNREG
                                UPDATNN         EQU     *
14307  6,0307 3  0,0001 0                      XCH     Q
14310  6,0310 5  0,0414 0                      TS      UPDATRET

14311  6,0311 3  6,6124 0                      CAF     LODNNLOC        ; was DCA LODNNLOC, DXCH Z in Block II
14312  6,0312 0  1,3526 0                      TC      DXCHJUMP        ; bank jump to noun table read rtne

14313  6,0313 1  0,0443 0                      CCS     NNADTEM
14314  6,0314 6  1,2051 1                      AD      ONE             ; >0, normal
14315  6,0315 0  6,6320 0                      TC      PUTADD          ; +0, normal
14316  6,0316 0  6,6321 1                      TC      PUTADD+1        ; <0, MCTBS don't change NOUNADD
14317  6,0317 0  6,6321 1                      TC      PUTADD+1        ; -0, MCTBI don't change NOUNADD

                                PUTADD          EQU     *
14320  6,0320 0  2,4616 1                      TC      SETNCADR        ; ECADR into NOUNCADR, sets NOUNADD

14321  6,0321 3  2,4676 1                      CAF     ND1
14322  6,0322 5  0,0466 0                      TS      DSPCOUNT

14323  6,0323 3  1,2050 0                      CAF     ZERO            ; was CA NOUNREG in Block II
14324  6,0324 6  0,0471 0                      AD      NOUNREG
14325  6,0325 0  6,6335 1                      TC      UPDAT1


14326  6,0326 5  0,0470 1                      TS      VERBREG
                                UPDATVB         EQU     *
14327  6,0327 3  0,0001 0                      XCH     Q
14330  6,0330 5  0,0414 0                      TS      UPDATRET

14331  6,0331 3  2,4675 1                      CAF     VD1
14332  6,0332 5  0,0466 0                      TS      DSPCOUNT

14333  6,0333 3  1,2050 0                      CAF     ZERO            ; was CA VERBREG in Block II
14334  6,0334 6  0,0470 1                      AD      VERBREG

                                UPDAT1          EQU     *
14335  6,0335 0  1,3653 1                      TC      POSTJUMP        ; can't use SWCALL to go to DSPDECVN, since
14336  6,0336     13156 1                      DS      GOVNUPDT        ; UPDATVB can itself be called by SWCALL
14337  6,0337 0  0,0414 0                      TC      UPDATRET


                                ;----------------------------------------------------------------------------
                                ; GOALMCYC, GODSPALM
                                ;
                                ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                                ; Oct 28, 1968, p.324.
                                ;----------------------------------------------------------------------------

14340  6,0340 0  2,4474 1 GOALMCYC        TC      ALMCYCLE        ; needed because bankjump cant handle F/F


14341  6,0341 0  1,3653 1 GODSPALM        TC      POSTJUMP
14342  6,0342     13267 0                      DS      DSPALARM


                                ;----------------------------------------------------------------------------
                                ; DISPLAY VERBS
                                ;
                                ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                                ; Oct 28, 1968, p.326.
                                ;----------------------------------------------------------------------------

                                DSPABC          EQU     *
14343  6,0343 4  1,2052 0                      CS      TWO
14344  6,0344 0  6,6414 0                      TC      COMPTEST
14345  6,0345 2  0,0442 1                      INDEX   NOUNADD
14346  6,0346 4  0,0002 1                      CS      2
14347  6,0347 3  0,0427 0                      XCH     BUF+2

                                DSPAB           EQU     *
14350  6,0350 4  1,2051 0                      CS      ONE
14351  6,0351 0  6,6414 0                      TC      COMPTEST
14352  6,0352 2  0,0442 1                      INDEX   NOUNADD
14353  6,0353 4  0,0001 1                      CS      1
```

```
14354  6,0354 3  0,0426 1                XCH      BUF+1

                          DSPA            EQU      *
14355  6,0355 0  6,6435 0                TC       DECTEST
14356  6,0356 0  6,6462 1                TC       TSTFORDP
14357  6,0357 2  0,0442 1                INDEX    NOUNADD
14360  6,0360 4  0,0000 0                CS       0

                          DSPCOM1         EQU      *
14361  6,0361 3  0,0425 1                XCH      BUF
14362  6,0362 0  6,6375 0                TC       DSPCOM2

                          DSPB            EQU      *
14363  6,0363 4  1,2051 0                CS       ONE
14364  6,0364 0  6,6430 0                TC       DCOMPTST
14365  6,0365 2  0,0442 1                INDEX    NOUNADD
14366  6,0366 4  0,0001 1                CS       1
14367  6,0367 0  6,6361 0                TC       DSPCOM1

                          DSPC            EQU      *
14370  6,0370 4  1,2052 0                CS       TWO
14371  6,0371 0  6,6430 0                TC       DCOMPTST
14372  6,0372 2  0,0442 1                INDEX    NOUNADD
14373  6,0373 4  0,0002 1                CS       2
14374  6,0374 0  6,6361 0                TC       DSPCOM1

                          DSPCOM2         EQU      *
14375  6,0375 4  1,2052 0                CS       TWO             ;  A  B  C AB ABC
14376  6,0376 6  0,0470 1                AD       VERBREG         ; -1 -0 +1 +2 +3   IN A
14377  6,0377 1  0,0000 0                CCS      A               ; +0 +0 +0 +1 +2   IN A AFTER CCS
14400  6,0400 0  6,6403 0                TC       DSPCOM3
14401  6,0401 0  0,0433 0                TC       ENTEXIT
14402  6,0402 0  6,6403 0                TC       *+1

                          DSPCOM3         EQU      *
14403  6,0403 5  0,0417 0                TS       DISTEM          ; +0, +1, +2 into DISTEM
14404  6,0404 2  0,0000 0                INDEX    A
14405  6,0405 3  2,4635 0                CAF      R1D1
14406  6,0406 5  0,0466 0                TS       DSPCOUNT
14407  6,0407 2  0,0417 1                INDEX    DISTEM
14410  6,0410 4  0,0425 0                CS       BUF
14411  6,0411 0  6,7310 1                TC       DSPOCTWD
14412  6,0412 3  0,0417 0                XCH      DISTEM
14413  6,0413 0  6,6377 1                TC       DSPCOM2+2


                          ;-----------------------------------------------------------------------
                          ; COMPTEST
                          ; alarms if component number of verb (load or oct display) is
                          ; greater than the highest component number of noun.
                          ;
                          ; DCOMPTST
                          ; alarms if decimal only bit (bit 4 of comp code number) = 1.
                          ; If not, it performs regular COMPTEST.
                          ;
                          ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                          ; Oct 28, 1968, p.326.
                          ;-----------------------------------------------------------------------

                          COMPTEST        EQU      *
14414  6,0414 5  0,0420 1                TS       SFTEMP1         ; - verb comp
14415  6,0415 3  0,0001 0                XCH      Q               ; was LXCH Q in block II
14416  6,0416 5  0,0547 1                TS       LXCH_LPRET      ; save return address in faux LP

                          COMPTST1        EQU      *
14417  6,0417 0  6,6501 0                TC       GETCOMP
14420  6,0420 0  2,4647 0                TC       LEFT5
14421  6,0421 7  1,2053 1                MASK     THREE           ; noun comp
14422  6,0422 6  0,0420 1                AD       SFTEMP1         ; noun comp - verb comp
14423  6,0423 1  0,0000 0                CCS      A
14424  6,0424 0  0,0547 1                TC       LXCH_LPRET      ; noun comp G/ verb comp; return
14425  6,0425 0  5,6271 0                TC       CCSHOLE
14426  6,0426 0  6,6341 1                TC       GODSPALM        ; noun comp L/ verb comp

                          NDOMPTST        EQU      *
14427  6,0427 0  0,0547 1                TC       LXCH_LPRET      ; noun comp = verb comp; return

                          DCOMPTST        EQU      *
14430  6,0430 5  0,0420 1                TS       SFTEMP1         ; - verb comp

14431  6,0431 3  0,0001 0                XCH      Q               ; was LXCH Q in block II
14432  6,0432 5  0,0547 1                TS       LXCH_LPRET      ; save return address in faux LP

14433  6,0433 0  6,6435 0                TC       DECTEST
14434  6,0434 0  6,6417 0                TC       COMPTST1

                          ;-----------------------------------------------------------------------
```

```
                        ; DECTEST
                        ; alarms if dec only bit = 1 (bit 4 of comp code number1). Returns if not.
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.327.
                        ;-----------------------------------------------------------------------------

                        DECTEST         EQU     *
14435  6,0435 3  0,0001 0               XCH     Q               ; was QXCH MPAC+2 in block II
14436  6,0436 5  0,0132 1               TS      MPAC+2

14437  6,0437 0  6,6501 0               TC      GETCOMP
14440  6,0440 7  1,2063 1               MASK    BIT14

14441  6,0441 1  0,0000 0               CCS     A
14442  6,0442 0  6,6341 1               TC      GODSPALM
14443  6,0443 0  0,0132 1               TC      MPAC+2

                        ;-----------------------------------------------------------------------------
                        ; DCTSTCYC
                        ; alarms and recycles if dec only bit = 1  (bit 4 of comp code number).
                        ; Returns if not. Used by load verbs.
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.327.
                        ;-----------------------------------------------------------------------------

                        DCTSTCYC        EQU     *
14444  6,0444 3  0,0001 0               XCH     Q               ; was LXCH Q in block II
14445  6,0445 5  0,0547 1               TS      LXCH_LPRET      ; save return address in faux LP

14446  6,0446 0  6,6501 0               TC      GETCOMP
14447  6,0447 7  1,2063 1               MASK    BIT14

14450  6,0450 1  0,0000 0               CCS     A
14451  6,0451 0  2,4474 1               TC      ALMCYCLE
14452  6,0452 0  0,0547 1               TC      LXCH_LPRET

                        ;-----------------------------------------------------------------------------
                        ; NOUNTEST
                        ; alarms if no-load bit (bit 5 of comp code number) = 1
                        ; if not, it returns.
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.327.
                        ;-----------------------------------------------------------------------------

                        NOUNTEST        EQU     *
14453  6,0453 3  0,0001 0               XCH     Q               ; was LXCH Q in block II
14454  6,0454 5  0,0547 1               TS      LXCH_LPRET      ; save return address in faux LP

14455  6,0455 0  6,6501 0               TC      GETCOMP

14456  6,0456 1  0,0000 0               CCS     A
14457  6,0457 0  0,0547 1               TC      LXCH_LPRET
14460  6,0460 0  0,0547 1               TC      LXCH_LPRET
14461  6,0461 0  6,6341 1               TC      GODSPALM

                        ;-----------------------------------------------------------------------------
                        ; TSTFORDP
                        ; test for DP. If so, get minor part only.
                        ; The Block II version had some code that checked for a -1 in NNADTEM
                        ; which meant use an I/O channel instead of memory. This was removed
                        ; for the Block I.
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.327.
                        ;-----------------------------------------------------------------------------


                        TSTFORDP        EQU     *
14462  6,0462 3  0,0001 0               XCH     Q               ; was LXCH Q in block II
14463  6,0463 5  0,0547 1               TS      LXCH_LPRET      ; save return address in faux LP

14464  6,0464 2  0,0435 1               INDEX   MIXBR
14465  6,0465 0  6,6465 0               TC      *
14466  6,0466 0  6,6470 1               TC      *+2             ; normal
14467  6,0467 0  0,0547 1               TC      LXCH_LPRET      ; mixed case already handled in MIXNOUN

14470  6,0470 0  6,6762 1               TC      SFRUTNOR
14471  6,0471 0  6,6240 1               TC      DPTEST
14472  6,0472 0  0,0547 1               TC      LXCH_LPRET      ; no DP

14473  6,0473 3  0,0442 0               XCH     NOUNADD         ; was INCR NOUNADD in Block II
14474  6,0474 6  1,2051 1               AD      ONE             ; DP  E+1 into NOUNADD for minor part
14475  6,0475 5  0,0442 0               TS      NOUNADD
```

```
14476  6,0476 0  0,0547 1                TC       LXCH_LPRET

                          ;----------------------------------------------------------------------------
                          ; GETCOMP
                          ;
                          ; noun address is in NNADTEM
                          ; noun type is in NNTYPTEM
                          ;
                          ; MIXBR is an enumerated type:
                          ; 1 = normal nouns
                          ; 2 = mixed nouns
                          ;
                          ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                          ; Oct 28, 1968, p.328.
                          ;----------------------------------------------------------------------------

14477  6,0477     00444 0 COMPICK        DS       NNTYPTEM
14500  6,0500     00443 1                DS       NNADTEM

                          GETCOMP        EQU      *
14501  6,0501 2  0,0435 1                INDEX    MIXBR           ; normal             mixed
14502  6,0502 3  6,6476 1                CAF      COMPICK-1       ; ADRES NNTYPTEM     ADRES NNADTEM

14503  6,0503 2  0,0000 0                INDEX    A
14504  6,0504 4  0,0000 0                CS       0               ; C(NNTYPTEM)                C(NNADTEM)
14505  6,0505 4  0,0000 0                COM                      ; was CA 0 in Block II
14506  6,0506 7  2,4666 1                MASK     HI5

14507  6,0507 0  0,0001 0                TC       Q

                          ;----------------------------------------------------------------------------
                          ; DECDSP -- DECIMAL DISPLAY
                          ;
                          ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                          ; Oct 28, 1968, p.328.
                          ;----------------------------------------------------------------------------

                          DECDSP         EQU      *
14510  6,0510 0  6,6501 0                TC       GETCOMP
14511  6,0511 0  2,4647 0                TC       LEFT5
14512  6,0512 7  1,2053 1                MASK     THREE
14513  6,0513 5  0,0414 0                TS       DECOUNT         ; comp number into DECOUNT

                          DSPDCGET       EQU      *
14514  6,0514 5  0,0417 0                TS       DECTEM          ; picks up data
14515  6,0515 6  0,0442 0                AD       NOUNADD         ; DECTEM  1COMP +0, 2COMP +1, 3COMP +2
14516  6,0516 2  0,0000 0                INDEX    A
14517  6,0517 4  0,0000 0                CS       0
14520  6,0520 2  0,0417 1                INDEX    DECTEM
14521  6,0521 3  0,0472 0                XCH      XREG            ; cant use BUF since DMP uses it
14522  6,0522 1  0,0417 1                CCS      DECTEM
14523  6,0523 0  6,6514 1                TC       DSPDCGET        ; more to get

                          DSPDCPUT       EQU      *
14524  6,0524 3  1,2050 0                CAF      ZERO            ; displays data
14525  6,0525 5  0,0131 1                TS       MPAC+1          ; DECOUNT 1COMP +0, 2COMP +1, 3COMP +2
14526  6,0526 5  0,0132 1                TS       MPAC+2
14527  6,0527 2  0,0414 1                INDEX    DECOUNT
14530  6,0530 3  2,4635 0                CAF      R1D1
14531  6,0531 5  0,0466 0                TS       DSPCOUNT
14532  6,0532 2  0,0414 1                INDEX    DECOUNT
14533  6,0533 4  0,0472 1                CS       XREG
14534  6,0534 5  0,0130 0                TS       MPAC
14535  6,0535 0  6,7003 0                TC       SFCONUM         ; 2X (SF CON NUMB) in A
14536  6,0536 5  0,0420 1                TS       SFTEMP1

14537  6,0537 3  6,6550 1                CAF      GTSFOUTL        ; was DCA GTSFOUTL, DXCH Z in Block II
14540  6,0540 0  1,3526 0                TC       DXCHJUMP        ; bank jump to SF constant table read rtne

14541  6,0541 2  0,0435 1                INDEX    MIXBR
14542  6,0542 0  6,6542 1                TC       *+0
14543  6,0543 0  6,6546 0                TC       DSPSFNOR
14544  6,0544 0  6,6770 1                TC       SFRUTMIX
14545  6,0545 0  6,6560 1                TC       DECDSP3

                          DSPSFNOR       EQU      *
14546  6,0546 0  6,6762 1                TC       SFRUTNOR
14547  6,0547 0  6,6560 1                TC       DECDSP3
14550  6,0550     16162 0 GTSFOUTL       DS       GTSFOUT

                          DSPDCEND       EQU      *
14551  6,0551 0  1,3565 1                TC       BANKCALL        ; all SFOUT routines end here
14552  6,0552     13064 1                DS       DSPDECWD
14553  6,0553 1  0,0414 1                CCS      DECOUNT
14554  6,0554 0  6,6556 1                TC       *+2
14555  6,0555 0  0,0433 0                TC       ENTEXIT
14556  6,0556 5  0,0414 0                TS       DECOUNT
```

```
14557  6,0557 0  6,6524 1                TC       DSPDCPUT       ; more to display

                           DECDSP3        EQU      *
14560  6,0560 2  0,0000 0                INDEX    A
14561  6,0561 3  6,6563 1                CAF      SFOUTABR
14562  6,0562 0  1,3712 0                TC       BANKJUMP

                           SFOUTABR       EQU      *
14563  6,0563     13265 1                CADR     PREDSPAL       ; 0, alarm if dec display with octal only noun
14564  6,0564     14551 0                CADR     DSPDCEND       ; 1
14565  6,0565     12564 0                CADR     DEGOUTSF       ; 2
14566  6,0566     12644 1                CADR     ARTOUTSF       ; 3
14567  6,0567     00000 1                CADR     0              ; 4 **********
14570  6,0570     00000 1                CADR     0              ; 5 **********
14571  6,0571     00000 1                CADR     0              ; 6 **********
14572  6,0572     00000 1                CADR     0              ; 7 **********
14573  6,0573     16000 0                CADR     HMSOUT         ; 8
14574  6,0574     00000 1                CADR     0              ; 9 **********
14575  6,0575     00000 1                CADR     0              ; 10 *********
14576  6,0576     00000 1                CADR     0              ; 11 *********
14577  6,0577     00000 1                CADR     0              ; 12 *********



                           BANK41_2       EQU      *

                                          ORG      BANK40_2       ; COLOSSUS pp. 330-332
                                          INCL     bank40_2.asm
                           ;==============================================================================
                           ; SCALE FACTOR ROUTINES (file:bank40_2.asm)
                           ;
                           ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                           ; Oct 28, 1968, pp. 330-332.
                           ;==============================================================================

                           DEGOUTSF       EQU      *
12564  5,0564 3  1,2050 0                CAF      ZERO
12565  5,0565 5  0,0132 1                TS       MPAC+2         ; set index for full scale
12566  5,0566 0  5,6603 1                TC       FIXRANGE
12567  5,0567 0  5,6571 1                TC       *+2            ; no augment needed (SFTEMP1 and 2 are 0)
12570  5,0570 0  5,6572 1                TC       SETAUG         ; set augmenter according to C(MPAC+2)
12571  5,0571 0  5,6616 0                TC       DEGCOM

                           SETAUG         EQU      *
12572  5,0572 3  1,2050 0                CAF      ZERO           ; loads SFTEMP1 and SFTEMP2 with the
12573  5,0573 2  0,0132 0                INDEX    MPAC+2         ; DP augmenter constant
12574  5,0574 6  5,6640 0                AD       DEGTAB         ; was DCA DEGTAB, DXCH SFTEMP1 in Block II
12575  5,0575 3  0,0420 1                XCH      SFTEMP1

12576  5,0576 3  1,2050 0                CAF      ZERO
12577  5,0577 2  0,0132 0                INDEX    MPAC+2
12600  5,0600 6  5,6641 1                AD       DEGTAB+1
12601  5,0601 3  0,0421 0                XCH      SFTEMP1+1

12602  5,0602 0  0,0001 0                TC       Q

                           FIXRANGE       EQU      *
12603  5,0603 3  0,0001 0                XCH      Q
12604  5,0604 5  0,0563 1                TS       FR_RETQ

12605  5,0605 1  0,0130 1                CCS      MPAC           ; if MPAC is +, return to L+1
12606  5,0606 0  0,0563 1                TC       FR_RETQ        ; if MPAC is -, return to L+2 after
12607  5,0607 0  0,0563 1                TC       FR_RETQ        ; masking out the sign bit
12610  5,0610 0  5,6611 1                TC       *+1            ; was TCF *+1 in Block II
12611  5,0611 4  1,2062 0                CS       BIT15
12612  5,0612 7  0,0130 1                MASK     MPAC
12613  5,0613 5  0,0130 0                TS       MPAC
12614  5,0614 2  0,0563 0                INDEX    FR_RETQ
12615  5,0615 0  0,0001 0                TC       1

                           DEGCOM         EQU      *
12616  5,0616 3  1,2050 0                CAF      ZERO           ; was INDEX MPAC+2, DCA DEGTAB, DXCH MPAC in
Block II
12617  5,0617 2  0,0132 0                INDEX    MPAC+2         ; loads multiplier, does SHORTMP, and
12620  5,0620 6  5,6641 1                AD       DEGTAB+1       ; adds augmenter
12621  5,0621 3  0,0131 1                XCH      MPAC+1         ; adjusted angle in A

12622  5,0622 3  1,2050 0                CAF      ZERO
12623  5,0623 2  0,0132 0                INDEX    MPAC+2
12624  5,0624 6  5,6640 0                AD       DEGTAB
12625  5,0625 3  0,0130 0                XCH      MPAC

12626  5,0626 0  2,4353 0                TC       SHORTMP

12627  5,0627 3  0,0421 0                XCH      SFTEMP1+1      ; was DXCH SFTEMP1, DAS MPAC in Block II
12630  5,0630 6  0,0131 1                AD       MPAC+1
```

```
12631  5,0631 5  0,0131 1                TS       MPAC+1          ; skip on overflow
12632  5,0632 3  1,2050 0                CAF      ZERO            ; otherwise, make interword carry=0

12633  5,0633 6  0,0420 1                AD       SFTEMP1
12634  5,0634 6  0,0130 0                AD       MPAC
12635  5,0635 5  0,0130 0                TS       MPAC            ; skip on overflow
12636  5,0636 3  1,2050 0                CAF      ZERO            ; otherwise, make interword carry=0


12637  5,0637 0  5,6651 0                TC       SCOUTEND

                          DEGTAB         EQU      *
12640  5,0640     05605 1                DS       %05605          ; Hi part of .18
12641  5,0641     03656 1                DS       %03656          ; Lo part of .18
12642  5,0642     16314 0                DS       %16314          ; Hi part of .45
12643  5,0643     31463 1                DS       %31463          ; Lo part of .45


                          ARTOUTSF       EQU      *
12644  5,0644 3  0,0421 0                XCH      SFTEMP1+1       ; was DXCH SFTEMP1, DXCH MPAC in Block II
12645  5,0645 3  0,0131 1                XCH      MPAC+1          ; assumes point at left of DP SFCON
12646  5,0646 3  0,0420 1                XCH      SFTEMP1
12647  5,0647 3  0,0130 0                XCH      MPAC

12650  5,0650 0  2,4740 0                TC       PRSHRTMP        ; if C(A) = -0, SHORTMP fails to give -0
12651  5,0651 0  1,3653 1 SCOUTEND       TC       POSTJUMP
12652  5,0652     14551 0                CADR     DSPDCEND



                          ;-----------------------------------------------------------------------------
                          ; READLO
                          ; Picks up fresh data for both HI and LO and leaves it in MPAC, MPAC+1.
                          ; This is needed for time display. It zeroes MPAC+2, but does not force
                          ; TPAGREE.
                          ;
                          ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                          ; Oct 28, 1968, p.332.
                          ;-----------------------------------------------------------------------------

                          READLO         EQU      *
12653  5,0653 3  0,0001 0                XCH      Q
12654  5,0654 5  0,0441 0                TS       TEM4            ; save return address

12655  5,0655 2  0,0435 1                INDEX    MIXBR
12656  5,0656 0  5,6656 1                TC       *

12657  5,0657 0  5,6701 1                TC       RDLONOR         ; MIXBR=1, so normal noun

12660  5,0660 3  1,2050 0                CAF      ZERO            ; MIXBR=2, so mixed noun
12661  5,0661 2  0,0414 1                INDEX    DECOUNT         ; was INDEX DECOUNT, CA IDAD1TEM in Block II
12662  5,0662 6  0,0445 1                AD       IDAD1TEM        ; get IDADDTAB entry for comp K of noun
12663  5,0663 7  2,4672 1                MASK     LOW11           ; E bank
12664  5,0664 0  2,4633 0                TC       SETEBANK        ; set EB, leave E address in A


                          ; Dereference noun address to move components of noun into MPAC, MPAC+1
                          ; mixed           normal
                          ; C(E SUBK)       C(E)
                          ; C((E SUBK)+1)   C(E+1)

                          READLO1        EQU      *
12665  5,0665 5  0,0576 0                TS       ADDRWD1         ; temp store addr for immediate use below

12666  5,0666 3  1,2050 0                CAF      ZERO            ; was INDEX A, DCA Q, DXCH MPAC in Block II
12667  5,0667 2  0,0576 1                INDEX    ADDRWD1
12670  5,0670 6  0,0000 1                AD       0
12671  5,0671 5  0,0130 0                TS       MPAC

12672  5,0672 3  1,2050 0                CAF      ZERO
12673  5,0673 2  0,0576 1                INDEX    ADDRWD1
12674  5,0674 6  0,0001 0                AD       1
12675  5,0675 5  0,0131 1                TS       MPAC+1

12676  5,0676 3  1,2050 0                CAF      ZERO
12677  5,0677 5  0,0132 1                TS       MPAC+2

12700  5,0700 0  0,0441 0                TC       TEM4            ; return

12701  5,0701 3  1,2050 0 RDLONOR        CAF      ZERO            ; was CA NOUNADD in Block II
12702  5,0702 6  0,0442 0                AD       NOUNADD
12703  5,0703 0  5,6665 1 ENDRDLO        TC       READLO1

                          BANK40_3       EQU      *

                                         ORG      BANK42_1        ; COLOSSUS pp. 333-336
                                         INCL     bank42_1.asm
```

```
                        ;================================================================================
                        ; DISPLAY ROUTINES (file:bank42_1.asm)
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, pp. 333-336.
                        ;================================================================================

                        ;--------------------------------------------------------------------------------
                        ; HMSOUT -- OUTPUT SCALE FACTOR ROUTINE
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.333.
                        ;--------------------------------------------------------------------------------

                        HMSOUT          EQU     *
16000  7,0000 0  1,3565 1               TC      BANKCALL        ; read fresh data for HI and LO into MPAC,
16001  7,0001     12653 1               DS      READLO          ; MPAC+1.

16002  7,0002 0  2,4150 1               TC      TPAGREE         ; make DP data agree
16003  7,0003 0  7,6053 1               TC      SEPSECNR        ; leave frac sec/60 in MPAC, MPAC+1, leave
                        ; whole min in bit13 of LOWTEMOUT and above
16004  7,0004 0  2,4374 0               TC      DMP             ; use only fract sec/60 mod 60
16005  7,0005     06043 0               ADRES   SECON2          ; mult by .06

16006  7,0006 3  2,4637 1               CAF     R3D1            ; gives CENT1-SEC/10EXP5 mod 60
16007  7,0007 5  0,0466 0               TS      DSPCOUNT

16010  7,0010 0  1,3565 1               TC      BANKCALL        ; display sec mod 60
16011  7,0011     13064 1               DS      DSPDECWD

16012  7,0012 0  7,6074 1               TC      SEPMIN          ; remove rest of seconds
16013  7,0013 3  7,6045 0               CAF     MINCON2         ; leave fract min/60 in MPAC+1, leave
16014  7,0014 3  0,0130 0               XCH     MPAC            ; whole hours in MPAC
16015  7,0015 5  0,0476 1               TS      HITEMOUT        ; save whole hours
16016  7,0016 3  7,6046 0               CAF     MINCON2+1
16017  7,0017 3  0,0131 1               XCH     MPAC+1          ; use only fract min/60 mod 60
16020  7,0020 0  2,4740 0               TC      PRSHRTMP        ; if C(A) = -0, SHORTMP fails to give -0.
                        ; mult by .0006
16021  7,0021 3  2,4636 0               CAF     R2D1            ; gives min/10EXP5 mod 60
16022  7,0022 5  0,0466 0               TS      DSPCOUNT

16023  7,0023 0  1,3565 1               TC      BANKCALL        ; display min mod 60
16024  7,0024     13064 1               DS      DSPDECWD


16025  7,0025 3  7,6051 0               CAF     HRCON1          ; was DCA HRCON1, DXCH MPAC in Block II
16026  7,0026 5  0,0130 0               TS      MPAC
16027  7,0027 3  7,6052 0               CAF     HRCON1+1
16030  7,0030 5  0,0131 1               TS      MPAC+1          ; minutes, seconds have been removed

16031  7,0031 3  1,2050 0               CAF     ZERO            ; was CA HITEMOUT in Block II
16032  7,0032 6  0,0476 1               AD      HITEMOUT        ; use whole hours
16033  7,0033 0  2,4740 0               TC      PRSHRTMP        ; if C(A) = -0, SHORTMP fails to give -0.
                        ; mult by .16384
16034  7,0034 3  2,4635 0               CAF     R1D1            ; gives hours/10EXP5
16035  7,0035 5  0,0466 0               TS      DSPCOUNT

16036  7,0036 0  1,3565 1               TC      BANKCALL        ; use regular DSPDECWD, with round off
16037  7,0037     13064 1               DS      DSPDECWD

16040  7,0040 0  0,0433 0               TC      ENTEXIT

16041  7,0041     25660 0 SECON1         DS      %25660          ; 2EXP12/6000
16042  7,0042     31742 1               DS      %31742

16043  7,0043     01727 1 SECON2         DS      %01727          ; .06 for seconds display
16044  7,0044     01217 1               DS      %01217

16045  7,0045     00011 1 MINCON2        DS      %00011          ; .0006 for minutes display
16046  7,0046     32445 0               DS      %32445

16047  7,0047     02104 0 MINCON1        DS      %02104          ; .066..66 upped by 2EXP-28
16050  7,0050     10422 1               DS      %10422

16051  7,0051     05174 0 HRCON1         DS      %05174          ; .16384 decimal
16052  7,0052     13261 0               DS      %13261


                        ; ************* missing stuff ****************

                        SEPSECNR        EQU     *
16053  7,0053 3  0,0001 0               XCH     Q               ; this entry avoid rounding by .5 secs
16054  7,0054 5  0,0441 0               TS      SEPSCRET

16055  7,0055 0  2,4374 0               TC      DMP             ; mult by 2EXP12/6000
16056  7,0056     06041 1               ADRES   SECON1          ; gives fract sec/60 in bit12 of MPAC+1
```

```
16057  7,0057 3  1,2050 0               CAF      ZERO           ; was DCA MPAC, DXCH HITEMOUT in Block II
16060  7,0060 6  0,0130 0               AD       MPAC           ; save minutes and hours
16061  7,0061 3  0,0476 1               XCH      HITEMOUT

16062  7,0062 3  1,2050 0               CAF      ZERO
16063  7,0063 6  0,0131 1               AD       MPAC+1
16064  7,0064 3  0,0477 0               XCH      HITEMOUT+1

16065  7,0065 0  2,4721 1               TC       TPSL1
16066  7,0066 0  2,4721 1               TC       TPSL1          ; gives fract sec/60 in MPAC+1, MPAC+2

16067  7,0067 3  1,2050 0               CAF      ZERO
16070  7,0070 3  0,0132 1               XCH      MPAC+2         ; leave fract sec/60 in MPAC, MPAC+1
16071  7,0071 3  0,0131 1               XCH      MPAC+1
16072  7,0072 3  0,0130 0               XCH      MPAC

16073  7,0073 0  0,0441 0               TC       SEPSCRET


                         SEPMIN        EQU      *
16074  7,0074 3  0,0001 0               XCH      Q              ; finds whole minutes in bit13
16075  7,0075 5  0,0441 0               TS       SEPMNRET       ; of LOWTEMOUT and above.

16076  7,0076 3  1,2050 0               CAF      ZERO
16077  7,0077 6  0,0477 0               AD       LOTEMOUT       ; removes rest of seconds

16100  7,0100 2  0,0000 1               EXTEND                  ; leaves fract min/60 in MPAC+1
16101  7,0101 4  1,2076 0               MP       BIT3           ; leaves whole hours in MPAC
16102  7,0102 2  0,0000 1               EXTEND                  ; SR 12, throw away LP
16103  7,0103 4  1,2064 0               MP       BIT13          ; SR 2?, take from LP. = SL 12

16104  7,0104 3  0,0003 1               XCH      LP             ; was LXCH MPAC+1 in Block II
16105  7,0105 5  0,0131 1               TS       MPAC+1         ; this forces bits 12-1 to 0 if +,
                         ; forces bits 12-1 to 1 if -.

16106  7,0106 3  1,2050 0               CAF      ZERO
16107  7,0107 6  0,0476 1               AD       HITEMOUT
16110  7,0110 5  0,0130 0               TS       MPAC

16111  7,0111 0  2,4374 0               TC       DMP            ; mult by 1/15
16112  7,0112     06047 1               ADRES    MINCON1        ; gives fract min/60 in MPAC+1
16113  7,0113 0  0,0441 0 ENDSPMIN      TC       SEPMNRET       ; gives whole hours in MPAC



                         BANK42_2      EQU      *

                                       ORG      BANK40_3       ; COLOSSUS pp. 336
                                       INCL     bank40_3.asm
                         ;==============================================================================
                         ; WORD DISPLAY ROUTINES (file:bank40_3.asm)
                         ;
                         ; AGC Block II COLOSSUS rev 249 assembly listing,
                         ; Oct 28, 1968, p.336.
                         ;==============================================================================

                         ;------------------------------------------------------------------------------
                         ; DSPDPDEC
                         ; This is a special purpose verb for displaying a double precision AGC
                         ; word as 10 decimal digits on the AGC display panel. It can be used with
                         ; any noun, except mixed nouns. It displays the contents of the register
                         ; NOUNADD is pointing to. If used with nouns which are inherently not DP
                         ; such as the CDU counters, the display will be garbage.
                         ; Display is in R1 and R2 only with the sign in R1.
                         ;
                         ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                         ; Oct 28, 1968, p.353.
                         ;------------------------------------------------------------------------------

                         DSPDPDEC      EQU      *
12704  5,0704 2  0,0435 1               INDEX    MIXBR
12705  5,0705 0  5,6705 0               TC       *+0
12706  5,0706 0  5,6710 1               TC       *+2            ; normal noun
12707  5,0707 0  5,7267 0               TC       DSPALARM

12710  5,0710 3  1,2050 0               CAF      ZERO
12711  5,0711 2  0,0442 1               INDEX    NOUNADD
12712  5,0712 6  0,0000 1               AD       0              ; was DCA 0, DXCH MPAC in Block II
12713  5,0713 5  0,0130 0               TS       MPAC

12714  5,0714 3  1,2050 0               CAF      ZERO
12715  5,0715 2  0,0442 1               INDEX    NOUNADD
12716  5,0716 6  0,0001 0               AD       1              ; was DCA 0, DXCH MPAC in Block II
12717  5,0717 5  0,0131 1               TS       MPAC+1

12720  5,0720 3  2,4635 0               CAF      R1D1
12721  5,0721 5  0,0466 0               TS       DSPCOUNT
```

```
12722  5,0722 3  1,2050 0                CAF      ZERO
12723  5,0723 5  0,0132 1                TS       MPAC+2

12724  5,0724 0  2,4150 1                TC       TPAGREE
12725  5,0725 0  5,7131 0                TC       DSP2DEC
12726  5,0726 0  0,0433 0 ENDDPDEC       TC       ENTEXIT



                           BANK40_4       EQU      *

                                          ORG      BANK41_2       ; COLOSSUS pp. 337-342
                                          INCL     bank41_2.asm
                           ;==============================================================================
                           ; DISPLAY ROUTINES (file:bank41_2.asm)
                           ;
                           ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                           ; Oct 28, 1968, pp. 337-342.
                           ;==============================================================================

                           ;==============================================================================
                           ; PINBALL GAME LOAD VERBS (file:bank41_2.asm)
                           ;
                           ; If alarm condition is detected during execute, check fail light is
                           ; turned on and ENDOFJOB. If alarm condition is detected during enter
                           ; of data, check fail is turned on and it recycles to execute of
                           ; original load verb. Recycle caused by 1) decimal machine CADR,
                           ; 2) mixture of octal/decimal data, 3) octal data into decimal only
                           ; noun, 4) decimal data into octal only noun, 5) data too large for
                           ; scale, 6) fewer than two data words loaded for HRS, MIN, SEC noun.
                           ; For #2-6, alarm and recycle occur at final enter of set; for #1,
                           ; alarm and recycle occur at enter of CADR.
                           ;
                           ; AGC Block II COLOSSUS rev 249 assembly listing,
                           ; Oct 28, 1968, p.337-343.
                           ;==============================================================================

                           ABCLOAD        EQU      *
14600  6,0600 4  1,2052 0                CS       TWO
14601  6,0601 0  6,6414 0                TC       COMPTEST
14602  6,0602 0  6,6453 0                TC       NOUNTEST       ; test if noun can be loaded

14603  6,0603 3  6,6733 0                CAF      VBSP1LD
14604  6,0604 0  6,6326 0                TC       UPDATVB-1
14605  6,0605 0  6,6264 1                TC       REQDATX

14606  6,0606 3  6,6734 1                CAF      VBSP2LD
14607  6,0607 0  6,6326 0                TC       UPDATVB-1
14610  6,0610 0  6,6270 1                TC       REQDATY

14611  6,0611 3  6,6735 0                CAF      VBSP3LD
14612  6,0612 0  6,6326 0                TC       UPDATVB-1
14613  6,0613 0  6,6274 0                TC       REQDATZ

                           PUTXYZ         EQU      *
14614  6,0614 4  1,2056 1                CS       SIX            ; test that the 3 data words loaded are
14615  6,0615 0  6,6736 0                TC       ALLDC_OC       ; all dec or all oct

14616  6,0616 3  6,6124 0                CAF      LODNNLOC       ; was DCA LODNNLOC, DXCH Z in Block II
14617  6,0617 0  1,3526 0                TC       DXCHJUMP       ; bank jump to noun table read rtne

14620  6,0620 3  1,2050 0                CAF      ZERO           ; X comp
14621  6,0621 0  6,7031 1                TC       PUTCOM
14622  6,0622 2  0,0442 1                INDEX    NOUNADD
14623  6,0623 5  0,0000 1                TS       0

14624  6,0624 3  1,2051 1                CAF      ONE            ; Y comp
14625  6,0625 0  6,7031 1                TC       PUTCOM
14626  6,0626 2  0,0442 1                INDEX    NOUNADD
14627  6,0627 5  0,0001 0                TS       1

14630  6,0630 3  1,2052 1                CAF      TWO            ; Z comp
14631  6,0631 0  6,7031 1                TC       PUTCOM
14632  6,0632 2  0,0442 1                INDEX    NOUNADD
14633  6,0633 5  0,0002 0                TS       2

                           ; *************** missing stuff *****************

                           ; Omitted a bunch of code from here that does special stuff if the noun=7.
                           ; (a noun that operates on I/O channels and flagbits)

14634  6,0634 0  6,6723 1                TC       LOADLV


                           ABLOAD         EQU      *
```

```
14635  6,0635 4  1,2051 0                CS       ONE
14636  6,0636 0  6,6414 0                TC       COMPTEST
14637  6,0637 0  6,6453 0                TC       NOUNTEST       ; test if noun can be loaded

14640  6,0640 3  6,6733 0                CAF      VBSP1LD
14641  6,0641 0  6,6326 0                TC       UPDATVB-1
14642  6,0642 0  6,6264 1                TC       REQDATX

14643  6,0643 3  6,6734 1                CAF      VBSP2LD
14644  6,0644 0  6,6326 0                TC       UPDATVB-1
14645  6,0645 0  6,6270 1                TC       REQDATY

                         PUTXY           EQU      *
14646  6,0646 4  1,2055 1                CS       FIVE           ; test that the 2 data words loaded are
14647  6,0647 0  6,6736 0                TC       ALLDC_OC       ; all dec or all oct

14650  6,0650 3  6,6124 0                CAF      LODNNLOC       ; was DCA LODNNLOC, DXCH Z in Block II
14651  6,0651 0  1,3526 0                TC       DXCHJUMP       ; bank jump to noun table read rtne

14652  6,0652 3  1,2050 0                CAF      ZERO           ; X comp
14653  6,0653 0  6,7031 1                TC       PUTCOM
14654  6,0654 2  0,0442 1                INDEX    NOUNADD
14655  6,0655 5  0,0000 1                TS       0

14656  6,0656 3  1,2051 1                CAF      ONE            ; Y comp
14657  6,0657 0  6,7031 1                TC       PUTCOM
14660  6,0660 2  0,0442 1                INDEX    NOUNADD
14661  6,0661 5  0,0001 0                TS       1

14662  6,0662 0  6,6723 1                TC       LOADLV

                         ALOAD           EQU      *
14663  6,0663 0  6,6264 1                TC       REQDATX

14664  6,0664 3  6,6124 0                CAF      LODNNLOC       ; was DCA LODNNLOC, DXCH Z in Block II
14665  6,0665 0  1,3526 0                TC       DXCHJUMP       ; bank jump to noun table read rtne

14666  6,0666 3  1,2050 0                CAF      ZERO           ; X comp
14667  6,0667 0  6,7031 1                TC       PUTCOM
14670  6,0670 2  0,0442 1                INDEX    NOUNADD
14671  6,0671 5  0,0000 1                TS       0
14672  6,0672 0  6,6723 1                TC       LOADLV

                         BLOAD           EQU      *
14673  6,0673 4  1,2051 0                CS       ONE
14674  6,0674 0  6,6414 0                TC       COMPTEST
14675  6,0675 3  1,2062 1                CAF      BIT15          ; set CLPASS for PASS0 only
14676  6,0676 5  0,0504 0                TS       CLPASS
14677  6,0677 0  6,6270 1                TC       REQDATY
14700  6,0700 3  6,6124 0                CAF      LODNNLOC       ; was DCA LODNNLOC, DXCH Z in Block II
14701  6,0701 0  1,3526 0                TC       DXCHJUMP       ; bank jump to noun table read rtne
14702  6,0702 3  1,2051 1                CAF      ONE
14703  6,0703 0  6,7031 1                TC       PUTCOM
14704  6,0704 2  0,0442 1                INDEX    NOUNADD
14705  6,0705 5  0,0001 0                TS       1
14706  6,0706 0  6,6723 1                TC       LOADLV

                         CLOAD           EQU      *
14707  6,0707 4  1,2052 0                CS       TWO
14710  6,0710 0  6,6414 0                TC       COMPTEST
14711  6,0711 3  1,2062 1                CAF      BIT15          ; set CLPASS for PASS0 only
14712  6,0712 5  0,0504 0                TS       CLPASS
14713  6,0713 0  6,6274 0                TC       REQDATZ
14714  6,0714 3  6,6124 0                CAF      LODNNLOC       ; was DCA LODNNLOC, DXCH Z in Block II
14715  6,0715 0  1,3526 0                TC       DXCHJUMP       ; bank jump to noun table read rtne
14716  6,0716 3  1,2052 1                CAF      TWO
14717  6,0717 0  6,7031 1                TC       PUTCOM
14720  6,0720 2  0,0442 1                INDEX    NOUNADD
14721  6,0721 5  0,0002 0                TS       2
14722  6,0722 0  6,6723 1                TC       LOADLV         ; yes, COLOSSUS actually did this

                         LOADLV          EQU      *
14723  6,0723 3  1,2050 0                CAF      ZERO
14724  6,0724 5  0,0467 1                TS       DECBRNCH
14725  6,0725 4  1,2050 1                CS       ZERO
14726  6,0726 5  0,0503 1                TS       LOADSTAT
14727  6,0727 4  2,4675 0                CS       VD1            ; to block numerical chars and
14730  6,0730 5  0,0466 0                TS       DSPCOUNT       ; clears after a completed load
14731  6,0731 0  1,3653 1                TC       POSTJUMP       ; after completed load, go to RECALTST
14732  6,0732     13413 0                DS       RECALTST       ; to see if there is RECALL from ENDIDLE

14733  6,0733     00025 0 VBSP1LD        DS       21             ; VB21 = ALOAD
14734  6,0734     00026 0 VBSP2LD        DS       22             ; VB22 = BLOAD
14735  6,0735     00027 1 VBSP3LD        DS       23             ; VB23 = CLOAD

                         ALLDC_OC        EQU      *
14736  6,0736 5  0,0414 0                TS       DECOUNT        ; test that data words loaded are either
```

```
14737  6,0737 3  0,0001 0                XCH      Q              ; (needed to handle TCF conversion below)
14740  6,0740 5  0,0556 1                TS       ALLDC_OC_Q     ; save return address

14741  6,0741 4  0,0467 0                CS       DECBRNCH       ; all dec or all oct; alarms if not
14742  6,0742 5  0,0021 1                TS       SR
14743  6,0743 4  0,0021 0                CS       SR
14744  6,0744 4  0,0021 0                CS       SR             ; shifted right 2
14745  6,0745 1  0,0000 0                CCS      A              ; dec comp bits in low 3
14746  6,0746 0  6,6750 0                TC       *+2            ; some ones in low 3 (was TCF in Block II)
14747  6,0747 0  0,0556 1                TC       ALLDC_OC_Q     ; all zeros, all oct, OK so return
14750  6,0750 6  0,0414 0                AD       DECOUNT        ; dec comp = 7 for 3comp, =6 for 2comp
                                  ; (but it has been decremented by CCS)
14751  6,0751 1  0,0000 0                CCS      A              ; must match 6 for 3comp, 5 for 2comp
14752  6,0752 0  6,6756 0                TC       *+4            ; >0
14753  6,0753 0  6,6755 0                TC       *+2            ; +0
14754  6,0754 0  6,6756 0                TC       *+2            ; <0
14755  6,0755 0  6,6757 1                TC       *+2            ; -0, was BZF *+2 in Block II

14756  6,0756 0  2,4474 1                TC       ALMCYCLE       ; alarm and recycle (does not return)

14757  6,0757 3  0,0556 1                XCH      ALLDC_OC_Q     ; restore return address
14760  6,0760 5  0,0001 0                TS       Q
14761  6,0761 0  0,0001 0 GOQ            TC       Q              ; all required are dec, OK

                           ;-----------------------------------------------------------------------------
                           ; SFRUTNOR
                           ; gets SF routine number for normal case.
                           ;
                           ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                           ; Oct 28, 1968, p.340.
                           ;-----------------------------------------------------------------------------

                           SFRUTNOR       EQU      *
14762  6,0762 3  0,0001 0                XCH      Q
14763  6,0763 5  0,0411 0                TS       EXITEM         ; can't use L for return. TESTFORDP uses L.
14764  6,0764 3  2,4665 0                CAF      MID5
14765  6,0765 7  0,0444 1                MASK     NNTYPTEM
14766  6,0766 0  2,4640 1                TC       RIGHT5
14767  6,0767 0  0,0411 0                TC       EXITEM         ; SF routine number in A


                           ;-----------------------------------------------------------------------------
                           ; SFRUTMIX
                           ; gets SF routine number for mixed case.
                           ;
                           ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                           ; Oct 28, 1968, p.340.
                           ;-----------------------------------------------------------------------------

                           SFRUTMIX       EQU      *
14770  6,0770 3  0,0001 0                XCH      Q              ; gets SF routine number for mixed case
14771  6,0771 5  0,0411 0                TS       EXITEM

14772  6,0772 2  0,0414 1                INDEX    DECOUNT
14773  6,0773 3  6,7022 0                CAF      DISPLACE       ; put TC GOQ, TC RIGHT5, or TC LEFT5 in L
14774  6,0774 5  0,0557 0                TS       SFRUTMIX_L

14775  6,0775 2  0,0414 1                INDEX    DECOUNT
14776  6,0776 3  2,4664 1                CAF      LOW5           ; LOW5, MID5, or HI5 in A
14777  6,0777 7  0,0450 1                MASK     RUTMXTEM       ; get HI5, MID5, or LOW5 of RUTMXTAB entry

15000  6,1000 2  0,0557 1                INDEX    SFRUTMIX_L
15001  6,1001 0  0,0000 1                TC       0

                           ; do TC GOQ (DECOUNT=0), do TC RIGHT5 (DECOUNT=1), do TC LEFT5 (DECOUNT=2)

15002  6,1002 0  0,0411 0 SFRET1         TC       EXITEM         ; SF routine number in A


                           SFCONUM        EQU      *
15003  6,1003 3  0,0001 0                XCH      Q              ; gets 2X (SF constant number)
15004  6,1004 5  0,0411 0                TS       EXITEM
15005  6,1005 2  0,0435 1                INDEX    MIXBR
15006  6,1006 0  6,7006 0                TC       *+0
15007  6,1007 0  6,7025 1                TC       CONUMNOR       ; normal noun
15010  6,1010 2  0,0414 1                INDEX    DECOUNT        ; mixed noun
15011  6,1011 3  6,7022 0                CAF      DISPLACE
15012  6,1012 5  0,0560 1                TS       SFCONUM_L      ; put TC GOQ, TC RIGHT5, or TC LEFT5 in L
15013  6,1013 2  0,0414 1                INDEX    DECOUNT
15014  6,1014 3  2,4664 1                CAF      LOW5
15015  6,1015 7  0,0444 1                MASK     NNTYPTEM
15016  6,1016 2  0,0560 0                INDEX    SFCONUM_L
15017  6,1017 0  0,0000 1                TC       0

                           ; do TC GOQ (DECOUNT=0), do TC RIGHT5 (DECOUNT=1), do TC LEFT5 (DECOUNT=2)
```

```
15020  6,1020 6  0,0000 1 SFRET          DOUBLE                  ; 2X (SF constant number) in A
15021  6,1021 0  0,0411 0                TC       EXITEM


                           DISPLACE       EQU      *
15022  6,1022 0  6,6761 1                TC       GOQ
15023  6,1023 0  2,4640 1                TC       RIGHT5
15024  6,1024 0  2,4647 0                TC       LEFT5

                           CONUMNOR       EQU      *
15025  6,1025 3  2,4664 1                CAF      LOW5           ; normal noun always gets low 5 of
15026  6,1026 7  0,0444 1                MASK     NNTYPTEM       ; NNTYPTAB for SF CONUM
15027  6,1027 6  0,0000 1                DOUBLE
15030  6,1030 0  0,0411 0                TC       EXITEM         ; 2X (SF constant number) in A

                           PUTCOM         EQU      *
15031  6,1031 5  0,0414 0                TS       DECOUNT
15032  6,1032 3  0,0001 0                XCH      Q
15033  6,1033 5  0,0412 0                TS       DECRET
15034  6,1034 3  1,2050 0                CAF      ZERO
15035  6,1035 5  0,0136 0                TS       MPAC+6
15036  6,1036 2  0,0414 1                INDEX    DECOUNT
15037  6,1037 3  0,0475 1                XCH      XREGLP
15040  6,1040 5  0,0131 1                TS       MPAC+1
15041  6,1041 2  0,0414 1                INDEX    DECOUNT
15042  6,1042 3  0,0472 0                XCH      XREG
15043  6,1043 5  0,0130 0                TS       MPAC
15044  6,1044 2  0,0435 1                INDEX    MIXBR
15045  6,1045 0  6,7045 1                TC       *
15046  6,1046 0  6,7077 0                TC       PUTNORM        ; normal noun

                           ; if mixnoun, place address for component K into NOUNADD, set EBANK bits.

15047  6,1047 2  0,0414 1                INDEX    DECOUNT        ; set IDADDTAB entry for component K
15050  6,1050 3  1,2050 0                CAF      ZERO           ; of noun
15051  6,1051 6  0,0445 1                AD       IDAD1TEM       ; was CA IDAD1TEM in Block II
15052  6,1052 7  2,4672 1                MASK     LOW11          ; (ECADR) SUBK for current comp of noun
15053  6,1053 0  2,4616 1                TC       SETNCADR       ; ECADR into NOUNCADR, sets EB, NOUNADD
15054  6,1054 2  0,0000 1                EXTEND                  ; C(NOUNADD) in A upon return
15055  6,1055 6  0,0414 0                SU       DECOUNT        ; place (ESUBK)-K into NOUNADD
15056  6,1056 5  0,0442 0                TS       NOUNADD
15057  6,1057 1  0,0467 0                CCS      DECBRNCH
15060  6,1060 0  6,7114 1                TC       PUTDECSF       ; + dec
15061  6,1061 0  6,6444 0                TC       DCTSTCYC       ; +0 octal
15062  6,1062 0  6,6770 1                TC       SFRUTMIX       ; test if dec only bit = 1. If so,
15063  6,1063 0  6,6240 1                TC       DPTEST         ; alarm and recycle. If not, continue.
15064  6,1064 0  6,7111 1                TC       PUTCOM2        ; no DP
                           ; test for DP scale for oct load. If so,
                           ; +0 into major part. Set NOUNADD for
                           ; loading octal word into minor part.

                           PUTDPCOM       EQU      *
15065  6,1065 3  1,2050 0                CAF      ZERO           ; was INCR NOUNADD in Block II
15066  6,1066 6  0,0442 0                AD       NOUNADD        ; DP (RSUBK)-K+1 or E+1
15067  6,1067 6  1,2051 1                AD       ONE
15070  6,1070 5  0,0442 0                TS       NOUNADD

15071  6,1071 6  0,0414 0                AD       DECOUNT        ; (ESUBK)+1 or E+1 into DECOUNT
15072  6,1072 5  0,0414 0                TS       DECOUNT        ; was ADS DECOUNT in Block II

15073  6,1073 3  1,2050 0                CAF      ZERO           ; NOUNADD set for minor part
15074  6,1074 2  0,0414 1                INDEX    DECOUNT
15075  6,1075 5 17,7776 0                TS       -1             ; zero major part (ESUBK or E1)
15076  6,1076 0  6,7111 1                TC       PUTCOM2

                           PUTNORM        EQU      *
15077  6,1077 0  2,4625 1                TC       SETNADD        ; ECADR from NOUNCADR, sets EB, NOUNADD
15100  6,1100 1  0,0467 0                CCS      DECBRNCH
15101  6,1101 0  6,7114 1                TC       PUTDECSF       ; +DEC
15102  6,1102 0  6,6444 0                TC       DCTSTCYC       ; +0 octal
15103  6,1103 0  6,6762 1                TC       SFRUTNOR       ; test if dec only bit = 1. If so,
15104  6,1104 0  6,6240 1                TC       DPTEST         ; alarm and recycle. If not, continue.
15105  6,1105 0  6,7111 1                TC       PUTNORM_1      ; no DP
15106  6,1106 3  1,2050 0                CAF      ZERO
15107  6,1107 5  0,0414 0                TS       DECOUNT
15110  6,1110 0  6,7065 0                TC       PUTDPCOM

                           PUTNORM_1      EQU      *              ; eliminated Block II CHANNEL LOAD code

                           PUTCOM2        EQU      *
15111  6,1111 3  0,0130 0                XCH      MPAC
15112  6,1112 0  0,0412 0                TC       DECRET

15113  6,1113     16176 0 GTSFINLC       DS       GTSFIN

                           ; *************** missing stuff *****************
```

```
                        ; PUTDECSF
                        ; Finds MIXBR and DECOUNT still set from PUTCOM

                        PUTDECSF        EQU     *
15114  6,1114 0  6,7003 0               TC      SFCONUM         ; 2X (SF CON NUM) in A
15115  6,1115 5  0,0420 1               TS      SFTEMP1

15116  6,1116 3  6,7113 0               CAF     GTSFINLC        ; was DCA GTSFINLC, DXCH Z in Block II
15117  6,1117 0  1,3526 0               TC      DXCHJUMP        ; bank jump to SF const table read rtne
                        ; loads SFTEMP1, SFTEMP2
15120  6,1120 2  0,0435 1               INDEX   MIXBR
15121  6,1121 0  6,7121 1               TC      *
15122  6,1122 0  6,7125 0               TC      PUTSFNOR
15123  6,1123 0  6,6770 1               TC      SFRUTMIX
15124  6,1124 0  6,7126 0               TC      PUTDCSF2
15125  6,1125 0  6,6762 1 PUTSFNOR      TC      SFRUTNOR

15126  6,1126 2  0,0000 0 PUTDCSF2      INDEX   A
15127  6,1127 3  6,7131 0               CAF     SFINTABR
15130  6,1130 0  1,3712 0               TC      BANKJUMP        ; switch banks for expansion room
15131  6,1131     14340 0 SFINTABR      CADR    GOALMCYC        ; 0, alarm and recycle if dec load
15132  6,1132     13011 0               CADR    BINROUND        ; 1
15133  6,1133     12727 0               CADR    DEGINSF         ; 2
15134  6,1134     12776 1               CADR    ARTHINSF        ; 3
15135  6,1135     00000 1               CADR    0               ; 4 **********
15136  6,1136     00000 1               CADR    0               ; 5 **********
15137  6,1137     00000 1               CADR    0               ; 6 **********
15140  6,1140     00000 1               CADR    0               ; 7 **********
15141  6,1141     00000 1               CADR    0               ; 8 **********
15142  6,1142     00000 1               CADR    0               ; 9 **********
15143  6,1143     00000 1               CADR    0               ; 10 *********
15144  6,1144     00000 1               CADR    0               ; 11 *********
15145  6,1145     00000 1               CADR    0               ; 12 *********


                        ; BUNCH OF TABLE ENTRIES GO HERE!!!!!

                        ; ************ NEED TO ADD THE REST *************


                        BANK41_3        EQU     *

                                        ORG     BANK40_4        ; COLOSSUS pp. 343-346
                                        INCL    bank40_4.asm
                        ;================================================================================
                        ; SCALE FACTOR ROUTINES (file:bank40_4.asm)
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, pp. 343-346.
                        ;================================================================================

                        DEGINSF         EQU     *
12727  5,0727 0  2,4374 0               TC      DMP             ; SF routine for dec degrees
12730  5,0730     06772 0               ADRES   DEGCON1         ; mult by 5.5 5(10)X2EXP-3
12731  5,0731 1  0,0131 0               CCS     MPAC+1          ; this rounds off MPAC+1 before shift
12732  5,0732 3  1,2066 0               CAF     BIT11           ; left 3, and causes 360.00 to OF/UF
12733  5,0733 0  5,6735 0               TC      *+2             ; when shifted left and alarm
12734  5,0734 4  1,2066 1               CS      BIT11
12735  5,0735 6  0,0131 1               AD      MPAC+1
12736  5,0736 0  5,7016 1               TC      _2ROUND+2
12737  5,0737 0  2,4721 1               TC      TPSL1           ; left 1
12740  5,0740 0  2,4721 1 DEGINSF2      TC      TPSL1           ; left 2
12741  5,0741 0  5,7025 1               TC      TESTOFUF
12742  5,0742 0  2,4721 1               TC      TPSL1           ; returns if no OF/UF (left 3)
12743  5,0743 1  0,0130 1               CCS     MPAC
12744  5,0744 0  5,6750 0               TC      SIGNFIX         ; if +, go to SIGNFIX
12745  5,0745 0  5,6750 0               TC      SIGNFIX         ; if +0, go to SIGNFIX
12746  5,0746 4  0,0000 0               COM                     ; if -, use -MAGNITUDE + 1
12747  5,0747 5  0,0130 0               TS      MPAC            ; -f -0; use +0
12750  5,0750 1  0,0136 1 SIGNFIX       CCS     MPAC+6
12751  5,0751 0  5,6766 0               TC      SGNTO1          ; if overflow
12752  5,0752 0  5,6762 1               TC      ENDSCALE        ; no overflow/underflow
12753  5,0753 1  0,0130 1               CCS     MPAC            ; if UF, force sign to 0 except -180
12754  5,0754 0  5,6271 0               TC      CCSHOLE
12755  5,0755 0  5,6764 1               TC      NEG180
12756  5,0756 0  5,6757 1               TC      *+1
12757  5,0757 3  0,0130 0               XCH     MPAC
12760  5,0760 7  1,2106 0               MASK    POSMAX
12761  5,0761 5  0,0130 0               TS      MPAC

                        ENDSCALE        EQU     *
12762  5,0762 0  1,3653 1               TC      POSTJUMP
12763  5,0763     15111 1               CADR    PUTCOM2

12764  5,0764 4  1,2106 0 NEG180        CS      POSMAX
12765  5,0765 0  5,6761 1               TC      ENDSCALE-1
```

```
                          SGNTO1         EQU      *
12766  5,0766 4  0,0130 1                CS       MPAC           ; if OV force sign to 1
12767  5,0767 7  1,2106 0                MASK     POSMAX
12770  5,0770 4  0,0000 0                CS       A
12771  5,0771 0  5,6761 1                TC       ENDSCALE-1

12772  5,0772     26161 0 DEGCON1        DS       %26161
12773  5,0773     30707 1                DS       %30707

12774  5,0774     21616 0 DEGCON2        DS       %21616
12775  5,0775     07071 0                DS       %07071

                          ; ************ missing stuff ***************

                          ARTHINSF       EQU      *
12776  5,0776 0  2,4374 0                TC       DMP            ; scales MPAC, +1 by SFTEMP1, SFTEMP2
12777  5,0777     00420 1                ADRES    SFTEMP1        ; assumes point between HI and LO parts
13000  5,1000 3  0,0132 1                XCH      MPAC+2         ; of SFCON, shifts results left by 14.
13001  5,1001 3  0,0131 1                XCH      MPAC+1         ; (by taking results from MPAC+1, MPAC+2)
13002  5,1002 3  0,0130 0                XCH      MPAC

13003  5,1003 1  0,0000 0                CCS      A              ; was BZF BINROUND in Block II
13004  5,1004 0  5,7010 1                TC       *+4            ; >0
13005  5,1005 0  5,7007 1                TC       *+2            ; +0
13006  5,1006 0  5,7010 1                TC       *+2            ; <0
13007  5,1007 0  5,7011 0                TC       BINROUND       ; -0

13010  5,1010 0  2,4474 1                TC       ALMCYCLE       ; too large a load, alarm and recycle

                          BINROUND       EQU      *
13011  5,1011 0  5,7014 0                TC       _2ROUND
13012  5,1012 0  5,7025 1                TC       TESTOFUF
13013  5,1013 0  5,6762 1                TC       ENDSCALE


                          ; ************ missing stuff ***************

                          _2ROUND        EQU      *
13014  5,1014 3  0,0131 1                XCH      MPAC+1
13015  5,1015 6  0,0000 1                DOUBLE
13016  5,1016 5  0,0131 1                TS       MPAC+1
13017  5,1017 0  0,0001 0                TC       Q              ; if MPAC+1 does not OF/UF
13020  5,1020 6  0,0130 0                AD       MPAC
13021  5,1021 5  0,0130 0                TS       MPAC
13022  5,1022 0  0,0001 0                TC       Q              ; if MPAC does not OF/UF
13023  5,1023 5  0,0136 0                TS       MPAC+6
13024  5,1024 0  0,0001 0 _2RNDEND       TC       Q

                          TESTOFUF       EQU      *
13025  5,1025 1  0,0136 1                CCS      MPAC+6         ; returns if no OF/UF
13026  5,1026 0  2,4474 1                TC       ALMCYCLE       ; OF, alarm and recycle
13027  5,1027 0  0,0001 0                TC       Q
13030  5,1030 0  2,4474 1                TC       ALMCYCLE       ; UF, alarm and recycle
                          BANK40_5       EQU      *

                                         ORG      BANK42_2
                          BANK42_3       EQU      *

                                         ORG      BANK41_3
                                         INCL     bank41_3.asm   ; COLOSSUS pp. 349-351
                          ;==============================================================================
                          ; DISPLAY ROUTINES (file:bank41_3.asm)
                          ;
                          ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                          ; Oct 28, 1968, pp. 349-351.
                          ;==============================================================================

                          ; MONITOR allows other keyboard activity. It is ended by verb TERMINATE
                          ; verb PROCEED WITHOUT DATA, verb RESEQUENCE, another monitor, or any
                          ; NVSUB call that passes DSPLOCK (provided that the operator has somehow
                          ; allowed the ending of a monitor which he has initiated through the
                          ; keyboard.
                          ;
                          ; MONITOR action is suspended, but not ended, by any keyboard action,
                          ; except error light reset. It begins again when KEY RELEASE is performed.
                          ; MONITOR saves the noun and appropriate display verb in MONSAVE. It saves
                          ; NOUNCADR in MONSAVE1, if noun = machine CADR to be specified. Bit 15 of
                          ; MONSAVE1 is the kill monitor signal (killer bit). Bit 14 of MONSAVE1
                          ; indicates the current monitor was externally initiated (external monitor
                          ; bit). It is turned off by RELDSP and KILMONON.
                          ;
                          ; MONSAVE indicates if MONITOR is on (+=ON, +0=OFF)
                          ; If MONSAVE is +, monitor enters no request, but turns killer bit off.
                          ; If MONSAVE is +0, monitor enters request and turns killer bit off.
                          ;
                          ; NVSUB (if external monitor bit is off), VB=PROCEED WITHOUT DATA,
```

```
                          ; VB=RESEQUENCE, and VB=TERMINATE turn kill monitor bit on.
                          ;
                          ; If killer bit is on, MONREQ enters no further requests, zeroes MONSAVE
                          ; and MONSAVE1 (turning off killer bit and external monitor bit).
                          ;
                          ; MONITOR doesn't test for MATBS since NVSUB can handle internal MATBS now.


                          MONITOR        EQU      *
15146  6,1146 4  6,7155 0               CS       BIT15_14
15147  6,1147 7  0,0506 0               MASK     NOUNCADR

                          MONIT1         EQU      *
15150  6,1150 5  0,0131 1               TS       MPAC+1         ; temp storage

15151  6,1151 4  0,0433 1               CS       ENTEXIT
15152  6,1152 6  2,4553 0               AD       ENDINST
15153  6,1153 1  0,0000 0               CCS      A
15154  6,1154 0  6,7164 0               TC       MONIT2

15155  6,1155     60000 1 BIT15_14       DS       %60000
15156  6,1156 0  6,7164 0               TC       MONIT2

15157  6,1157 3  1,2063 0               CAF      BIT14          ; externally initiated monitor
15160  6,1160 6  0,0131 1               AD       MPAC+1         ; was ADS MPAC+1 in Block II
15161  6,1161 5  0,0131 1               TS       MPAC+1         ; set bit 14 for MONSAVE1

15162  6,1162 3  1,2050 0               CAF      ZERO
15163  6,1163 5  0,0511 1               TS       MONSAVE2       ; zero NVMONOPT options

                          MONIT2         EQU      *
15164  6,1164 3  1,2101 0               CAF      LOW7
15165  6,1165 7  0,0470 0               MASK     VERBREG
15166  6,1166 0  2,4647 0               TC       LEFT5
15167  6,1167 5  0,0022 1               TS       CYL
15170  6,1170 4  0,0022 0               CS       CYL
15171  6,1171 3  0,0022 1               XCH      CYL
15172  6,1172 6  0,0471 0               AD       NOUNREG
15173  6,1173 5  0,0130 0               TS       MPAC           ; temp storage
15174  6,1174 3  1,2050 0               CAF      ZERO
15175  6,1175 5  0,0501 0               TS       DSPLOCK        ; +0 into DSPLOCK so monitor can run
15176  6,1176 1  0,0531 1               CCS      CADRSTOR       ; turn off KR lite if CADRSTOR and DSPLIST
15177  6,1177 0  6,7201 0               TC       *+2            ; are both empty. (Lite comes on if new
15200  6,1200 0  2,5026 0               TC       RELDSP1        ; monitor is keyed in over old monitor.)
15201  6,1201 2  0,0000 0               INHINT
15202  6,1202 1  0,0507 1               CCS      MONSAVE
15203  6,1203 0  6,7207 0               TC       *+4            ; if MONSAVE was +, no request

15204  6,1204 3  1,2051 1               CAF      ONE            ; if MONSAVE was 0, request MONREQ
15205  6,1205 0  1,2232 0               TC       WAITLIST
15206  6,1206     15215 0               CADR     MONREQ

15207  6,1207 3  0,0131 1               XCH      MPAC+1         ; was DXCH MPAC, DXCH MONSAVE
15210  6,1210 3  0,0510 0               XCH      MONSAVE+1

15211  6,1211 3  0,0130 0               XCH      MPAC           ; place monitor verb and noun into MONSAVE
15212  6,1212 3  0,0507 0               XCH      MONSAVE        ; zero the kill monitor bit


15213  6,1213 2  0,0000 1               RELINT                  ; set up external monitor bit
15214  6,1214 0  0,0433 0               TC       ENTRET

                          MONREQ         EQU      *
15215  6,1215 0  6,7300 0               TC       LODSAMPT       ; called by waitlist (see COLOSSUS p. 374)
15216  6,1216 1  0,0510 1               CCS      MONSAVE1       ; time is snatched in RUPT for NOUN 65
15217  6,1217 0  6,7223 0               TC       *+4            ; if killer bit = 0, enter requests
15220  6,1220 0  6,7223 0               TC       *+3            ; if killer bit = 0, enter requests
15221  6,1221 0  6,7232 0               TC       KILLMON        ; if killer bit = 1, no requests
15222  6,1222 0  6,7232 0               TC       KILLMON        ; if killer bit = 1, no requests

15223  6,1223 3  6,7236 1               CAF      MONDEL
15224  6,1224 0  1,2232 0               TC       WAITLIST       ; enter waitlist request for MONREQ
15225  6,1225     15215 0               CADR     MONREQ

15226  6,1226 3  2,4131 0               CAF      CHRPRIO
15227  6,1227 0  1,3162 1               TC       NOVAC          ; enter EXEC request for MONDO
15230  6,1230     15237 0               CADR     MONDO

15231  6,1231 0  1,2413 0               TC       TASKOVER

                          KILLMON        EQU      *
15232  6,1232 3  1,2050 0               CAF      ZERO           ; zero MONSAVE and turn killer bit off
15233  6,1233 5  0,0507 0               TS       MONSAVE
15234  6,1234 5  0,0510 0               TS       MONSAVE1       ; turn off kill monitor bit
15235  6,1235 0  1,2413 0               TC       TASKOVER       ; turn off external monitor bit

15236  6,1236     00144 0 MONDEL         DS       %144           ; for 1 sec monitor intervals
```

```
                          MONDO          EQU      *
15237  6,1237 1  0,0510 1                CCS      MONSAVE1       ; called by EXEC
15240  6,1240 0  6,7244 1                TC       *+4            ; if killer bit = 0, continue
15241  6,1241 0  6,7244 1                TC       *+3            ; if killer bit = 0, continue
15242  6,1242 0  1,2723 0                TC       ENDOFJOB       ; in case TERMINATE came since last MONREQ
15243  6,1243 0  1,2723 0                TC       ENDOFJOB       ; in case TERMINATE came since last MONREQ
15244  6,1244 1  0,0501 1                CCS      DSPLOCK
15245  6,1245 0  6,7276 0                TC       MONBUSY        ; NVSUB is busy
15246  6,1246 3  1,2101 0                CAF      LOW7
15247  6,1247 7  0,0507 1                MASK     MONSAVE
15250  6,1250 0  6,6306 1                TC       UPDATNN-1      ; place noun into NOUNREG and display it
15251  6,1251 3  2,4473 0                CAF      MID7
15252  6,1252 7  0,0507 1                MASK     MONSAVE        ; change monitor verb to display verb
15253  6,1253 6  6,7274 1                AD       MONREF         ; -DEC10, starting in bit5

15254  6,1254 5  0,0020 0                TS       CYR            ; shift right 7, was TS EDOP, CA EDOP in BII
15255  6,1255 4  0,0020 1                CS       CYR
15256  6,1256 4  0,0020 1                CS       CYR
15257  6,1257 4  0,0020 1                CS       CYR
15260  6,1260 4  0,0020 1                CS       CYR
15261  6,1261 4  0,0020 1                CS       CYR
15262  6,1262 4  0,0020 1                CS       CYR
15263  6,1263 3  0,0020 0                XCH      CYR
15264  6,1264 7  1,2101 1                MASK     LOW7

15265  6,1265 5  0,0470 1                TS       VERBREG
15266  6,1266 3  6,7275 0                CAF      MONBACK        ; set return to PASTEVB after data display
15267  6,1267 5  0,0433 0                TS       ENTRET
15270  6,1270 4  6,7155 0                CS       BIT15_14
15271  6,1271 7  0,0510 1                MASK     MONSAVE1
15272  6,1272 5  0,0132 1                TS       MPAC+2         ; display it and set NOUNCADR, NOUNADD,
15273  6,1273 0  6,6054 0 ENDMONDO       TC       TESTNN         ; EBANK


                          ; COLOSSUS switches to fixed/fixed memory and inserts PASTEVB here--
                          ; Probably, because their assembler couldn't handle forward references.

15274  6,1274     75377 0 MONREF         DS       %75377         ; -dec10, starting in bit8
15275  6,1275     04435 1 MONBACK        CADR     PASTEVB

15276  6,1276 0  2,4713 0 MONBUSY        TC       RELDSPON       ; turn key release light
15277  6,1277 0  1,2723 0                TC       ENDOFJOB

15300  6,1300 0  0,0001 0 LODSAMPT       TC       Q              ; ************************* FIX
**************************
                          BANK41_4       EQU      *

                                         ORG      BANKFF_1
                                         INCL     bankff_1.asm   ; COLOSSUS pp. 351
                          ;=============================================================================
                          ; DISPLAY ROUTINES (file:bankff_1.asm)
                          ;
                          ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                          ; Oct 28, 1968, pp. 351.
                          ;=============================================================================

                          PASTEVB        EQU      *
04435    4435 3  2,4473 0                CAF      MID7
04436    4436 7  0,0511 0                MASK     MONSAVE2       ; NVMONOPT paste option
04437    4437 5  0,0571 1                TS       PASTE_TMP

04440    4440 1  0,0000 0                CCS      A              ; was BZF *+2 in Block II
04441    4441 0  2,4443 0                TC       *+2            ; >0,
04442    4442 0  2,4444 1                TC       *+2            ; +0,
04443    4443 0  2,4445 0                TC       *+2            ; <0,
04444    4444 0  2,4447 1                TC       *+3            ; -0,

04445    4445 3  0,0571 1                XCH      PASTE_TMP
04446    4446 0  2,4451 0                TC       PASTEOPT       ; paste please verb for NVMONOPT

04447    4447 3  1,2050 0                CAF      ZERO           ; was CA MONSAVE in BII
04450    4450 6  0,0507 0                AD       MONSAVE        ; paste monitor verb - paste option is 0

                          PASTEOPT       EQU      *
04451    4451 5  0,0020 0                TS       CYR            ; shift right 7, was TS EDOP, CA EDOP in BII
04452    4452 4  0,0020 1                CS       CYR
04453    4453 4  0,0020 1                CS       CYR
04454    4454 4  0,0020 1                CS       CYR
04455    4455 4  0,0020 1                CS       CYR
04456    4456 4  0,0020 1                CS       CYR
04457    4457 4  0,0020 1                CS       CYR
04460    4460 3  0,0020 0                XCH      CYR
04461    4461 7  1,2101 1                MASK     LOW7           ; place monitor verb or please verb into

04462    4462 0  1,3565 1                TC       BANKCALL       ; VERBREG and display it.
04463    4463     14326 0                CADR     UPDATVB-1
```

```
04464    4464 3  1,2050 0               CAF      ZERO           ; zero REQRET so that pasted verbs can
04465    4465 5  0,0502 0               TS       REQRET         ; be executed by operator.

04466    4466 3  1,2050 0               CAF      ZERO
04467    4467 6  0,0511 1               AD       MONSAVE2       ; was CA MONSAVE2 in BII
04470    4470 0  2,4565 0               TC       BLANKSUB       ; process NVMONOPT blank option if any (p.
368)
04471    4471 0  2,4472 1               TC       *+1
04472    4472 0  1,2723 0 ENDPASTE      TC       ENDOFJOB

04473    4473     37600 0 MID7          DS       %37600
                         BANKFF_2      EQU      *

                                       ORG      BANK41_4
                                       INCL     bank41_4.asm   ; COLOSSUS pp. 352
                         ;==============================================================================
                         ; DISPLAY ROUTINES (file:bank41_4.asm)
                         ;
                         ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                         ; Oct 28, 1968, pp. 352.
                         ;==============================================================================

                         ;------------------------------------------------------------------------------
                         ; DSPFMEM -- DISPLAY FIXED MEMORY
                         ; Used to display (in octal) any fixed register. It is used with NOUN =
                         ; machine CADR to be specified. The FCADR of the desired location is then
                         ; punched in. It handles F/F (FCADR 4000-7777)
                         ;
                         ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                         ; Oct 28, 1968, p.352.
                         ;------------------------------------------------------------------------------

                         DSPFMEM        EQU      *
15301  6,1301 3  2,4635 0               CAF      R1D1           ; If F/F, DATACALL uses bank 02 or 03
15302  6,1302 5  0,0466 0               TS       DSPCOUNT

15303  6,1303 3  1,2050 0               CAF      ZERO           ; was CA NOUNCADR, TC SUPDACAL in Block II
15304  6,1304 6  0,0506 1               AD       NOUNCADR       ; original FCADR loaded still in NOUNCADR
15305  6,1305 0  1,3742 0               TC       DATACALL       ; call with FCADR in A

15306  6,1306 0  6,7310 1               TC       DSPOCTWD
15307  6,1307 0  1,2723 0 ENDSPF        TC       ENDOFJOB
                         BANK41_5      EQU      *

                                       ORG      BANK40_5       ; COLOSSUS pp. 353-355
                                       INCL     bank40_5.asm
                         ;==============================================================================
                         ; WORD DISPLAY ROUTINES (file:bank40_5.asm)
                         ;
                         ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                         ; Oct 28, 1968, pp. 353-355.
                         ;==============================================================================

                         DSPSIGN        EQU      *
13031  5,1031 3  0,0001 0               XCH      Q
13032  5,1032 5  0,0441 0               TS       DSPWDRET

13033  5,1033 1  0,0130 1               CCS      MPAC
13034  5,1034 0  5,7044 0               TC       *+8            ; >0, positive sign
13035  5,1035 0  5,7044 0               TC       *+7            ; +0, positive sign

13036  5,1036 6  1,2051 1               AD       ONE
13037  5,1037 5  0,0130 0               TS       MPAC
13040  5,1040 0  5,6353 1               TC       M_ON           ; display minus sign
13041  5,1041 4  0,0131 0               CS       MPAC+1
13042  5,1042 5  0,0131 1               TS       MPAC+1
13043  5,1043 0  0,0441 0               TC       DSPWDRET

13044  5,1044 0  5,6332 0               TC       P_ON           ; display plus sign

13045  5,1045 0  0,0441 0               TC       DSPWDRET       ; return


                         ;------------------------------------------------------------------------------
                         ; DSPRND
                         ; Round up decimal fraction by 5 EXP -6. This was entirely coded in
                         ; Block II instructions, so I translated it to the functional
                         ; equivalent in Block I code.
                         ;
                         ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                         ; Oct 28, 1968, p.353.
                         ;------------------------------------------------------------------------------

                         DSPRND         EQU      *
13046  5,1046 3  5,7117 1               CAF      DECROUND
13047  5,1047 6  0,0131 1               AD       MPAC+1
```

```
13050  5,1050 5  0,0131 1                TS       MPAC+1          ; skip on overflow
13051  5,1051 3  1,2050 0                CAF      ZERO            ; otherwise, make interword carry=0
13052  5,1052 6  0,0130 0                AD       MPAC
13053  5,1053 5  0,0130 0                TS       MPAC            ; skip on overflow
13054  5,1054 0  0,0001 0                TC       Q               ; return

13055  5,1055 3  5,7063 0                CAF      DPOSMAX+1       ; number overflows, so set to max
13056  5,1056 5  0,0131 1                TS       MPAC+1
13057  5,1057 3  5,7062 1                CAF      DPOSMAX
13060  5,1060 5  0,0130 0                TS       MPAC
13061  5,1061 0  0,0001 0                TC       Q               ; return

                          DPOSMAX        EQU      *               ; max positive decimal fraction
13062  5,1062     37777 1                DS       %37777
13063  5,1063     34000 0                DS       %34000


                          ;-----------------------------------------------------------------------
                          ; DSPDECTWD -- DISPLAY DECIMAL WORD
                          ; Converts C(MPAC, MPAC+1) into a sign and 5 char decimal starting in loc
                          ; specified in DSPCOUNT. it rounds by 5 exp 6.
                          ;
                          ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                          ; Oct 28, 1968, p.353.
                          ;-----------------------------------------------------------------------

                          DSPDECWD       EQU      *
13064  5,1064 3  0,0001 0                XCH      Q
13065  5,1065 5  0,0412 0                TS       WDRET

13066  5,1066 0  5,7031 1                TC       DSPSIGN
13067  5,1067 0  5,7046 1                TC       DSPRND
13070  5,1070 3  1,2054 1                CAF      FOUR

                          DSPDCWD1       EQU      *
13071  5,1071 5  0,0434 1                TS       WDCNT
13072  5,1072 3  2,4700 1                CAF      BINCON
13073  5,1073 0  2,4353 0                TC       SHORTMP

                          TRACE1         EQU      *
13074  5,1074 2  0,0130 1                INDEX    MPAC
13075  5,1075 3  1,3772 0                CAF      RELTAB
13076  5,1076 7  2,4664 0                MASK     LOW5
13077  5,1077 5  0,0421 0                TS       CODE
13100  5,1100 3  1,2050 0                CAF      ZERO
13101  5,1101 3  0,0132 1                XCH      MPAC+2
13102  5,1102 3  0,0131 1                XCH      MPAC+1
13103  5,1103 5  0,0130 0                TS       MPAC
13104  5,1104 3  0,0466 0                XCH      DSPCOUNT

                          TRACE1S        EQU      *
13105  5,1105 5  0,0440 1                TS       COUNT
13106  5,1106 1  0,0000 0                CCS      A               ; decrement DSPCOUNT except at +0
13107  5,1107 5  0,0466 0                TS       DSPCOUNT
13110  5,1110 0  5,7161 0                TC       DSPIN
13111  5,1111 1  0,0434 0                CCS      WDCNT
13112  5,1112 0  5,7071 0                TC       DSPDCWD1        ; >0, not done yet

13113  5,1113 4  2,4675 0                CS       VD1             ; +0
13114  5,1114 5  0,0466 0                TS       DSPCOUNT
13115  5,1115 0  0,0412 0                TC       WDRET           ; return

13116  5,1116     00000 1                DS       %00000
13117  5,1117     02476 0 DECROUND       DS       %02476


                          ;-----------------------------------------------------------------------
                          ; DSPDECNR
                          ; Converts C(MPAC, MPAC+1) into a sign and 5 char decimal starting in loc
                          ; specified in DSPCOUNT. It does not round.
                          ;
                          ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                          ; Oct 28, 1968, p.354.
                          ;-----------------------------------------------------------------------

                          DSPDECNR       EQU      *
13120  5,1120 3  0,0001 0                XCH      Q
13121  5,1121 5  0,0412 0                TS       WDRET
13122  5,1122 0  5,7031 1                TC       DSPSIGN
13123  5,1123 0  5,7070 1                TC       DSPDCWD1-1


                          ;-----------------------------------------------------------------------
                          ; DSPDC2NR
                          ; Converts C(MPAC, MPAC+1) into a sign and 2 char decimal starting in loc
                          ; specified by DSPCOUNT. It does not round.
                          ;
```

```
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.354.
                        ;-----------------------------------------------------------------------------

                        DSPDC2NR        EQU     *
13124  5,1124 3  0,0001 0               XCH     Q
13125  5,1125 5  0,0412 0               TS      WDRET
13126  5,1126 0  5,7031 1               TC      DSPSIGN
13127  5,1127 3  1,2051 1               CAF     ONE
13130  5,1130 0  5,7071 0               TC      DSPDCWD1

                        ;-----------------------------------------------------------------------------
                        ; DSP2DEC
                        ; Converts C(MPAC) and C(MPAC+1) into a sign and 10 char decimal starting
                        ; in the loc specified in DSPCOUNT.
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.354.
                        ;-----------------------------------------------------------------------------

                        DSP2DEC         EQU     *
13131  5,1131 3  0,0001 0               XCH     Q
13132  5,1132 5  0,0412 0               TS      WDRET
13133  5,1133 3  1,2050 0               CAF     ZERO
13134  5,1134 5  0,0421 0               TS      CODE
13135  5,1135 3  1,2053 0               CAF     THREE
13136  5,1136 0  5,7253 1               TC      _11DSPIN        ; -R2 off
13137  5,1137 3  1,2054 1               CAF     FOUR
13140  5,1140 0  5,7253 1               TC      _11DSPIN        ; +R2 off
13141  5,1141 0  5,7031 1               TC      DSPSIGN
13142  5,1142 3  2,4636 0               CAF     R2D1
13143  5,1143 0  5,7071 0 END2DEC       TC      DSPDCWD1


                        ;-----------------------------------------------------------------------------
                        ; DSPDECVN
                        ; Displays C(A) upon entry as a 2 char decimal beginning in the
                        ; loc specified in DSPCOUNT.
                        ; C(A) should be in form N x 2EXP-14. This is scaled to form N/100 before
                        ; display conversion.
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.353.
                        ;-----------------------------------------------------------------------------

                        DSPDECVN        EQU     *
13144  5,1144 2  0,0000 1               EXTEND
13145  5,1145 4  5,7155 0               MP      VNDSPCON        ; mult by .01

13146  5,1146 3  0,0003 1               XCH     LP              ; was LXCH MPAC in Block II
13147  5,1147 5  0,0130 0               TS      MPAC            ; take results from LP (mult by 2EXP14)

13150  5,1150 3  1,2050 0               CAF     ZERO
13151  5,1151 5  0,0131 1               TS      MPAC+1
13152  5,1152 3  0,0001 0               XCH     Q
13153  5,1153 5  0,0412 0               TS      WDRET
13154  5,1154 0  5,7127 1               TC      DSPDC2NR+3      ; no sign, no round, 2 char

13155  5,1155     00244 0 VNDSPCON       DS      %00244          ; .01 rounded up

                        GOVNUPDT        EQU     *
13156  5,1156 0  5,7144 1               TC      DSPDECVN        ; this is not for general use. Really part
13157  5,1157 0  1,3653 1               TC      POSTJUMP        ; of UPDATVB
13160  5,1160     14337 0               DS      UPDAT1+2

                        BANK40_6        EQU     *

                                        ORG     BANK41_5        ; COLOSSUS pp. 355-356
                                        INCL    bank41_5.asm
                        ;=============================================================================
                        ; DISPLAY ROUTINES (file:bank41_5.asm)
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, pp. 355-356.
                        ;=============================================================================

                        ;-----------------------------------------------------------------------------
                        ; DSPOCTWD -- DISPLAY OCTAL WORD
                        ; Displays C(A) upon entry as a 5 char octal starting in the DSP char
                        ; specified in DSPCOUNT. It stops after 5 char have been displayed.
                        ;
                        ;
                        ; DSP2BIT -- DISPLAY 2 OCTAL CHARS
                        ; Displays C(A) upon entry as a 2 char oct beginning in the DSP
                        ; loc specified in DSPCOUNT by pre-cycling right C(A) and using
                        ; the logic of the 5 char octal display.
                        ;
```

```
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.355/356.
                        ;------------------------------------------------------------------------

                        DSPOCTWD        EQU     *
15310  6,1310 5  0,0022 1               TS      CYL
15311  6,1311 3  0,0001 0               XCH     Q
15312  6,1312 5  0,0412 0               TS      WDRET           ; must use the same return as DSP2BIT

15313  6,1313 3  1,2063 0               CAF     BIT14           ; to blank signs
15314  6,1314 6  0,0466 0               AD      DSPCOUNT        ; was ADS DSPCOUNT in block II
15315  6,1315 5  0,0466 0               TS      DSPCOUNT

15316  6,1316 3  1,2054 1               CAF     FOUR

                        WDAGAIN         EQU     *
15317  6,1317 5  0,0434 1               TS      WDCNT
15320  6,1320 4  0,0022 0               CS      CYL
15321  6,1321 4  0,0022 0               CS      CYL
15322  6,1322 4  0,0022 0               CS      CYL
15323  6,1323 4  0,0000 0               CS      A

15324  6,1324 7  1,2057 0               MASK    DSPMSK
15325  6,1325 2  0,0000 0               INDEX   A
15326  6,1326 3  1,3772 0               CAF     RELTAB
15327  6,1327 7  2,4664 0               MASK    LOW5
15330  6,1330 5  0,0421 0               TS      CODE

15331  6,1331 3  0,0466 0               XCH     DSPCOUNT
15332  6,1332 5  0,0440 1               TS      COUNT
15333  6,1333 1  0,0000 0               CCS     A               ; decrement DSPCOUNT except at +0
15334  6,1334 5  0,0466 0               TS      DSPCOUNT        ; > 0
15335  6,1335 0  1,3653 1               TC      POSTJUMP        ; + 0
15336  6,1336     13261 0               DS      DSPOCTIN

                        OCTBACK         EQU     *
15337  6,1337 1  0,0434 0               CCS     WDCNT
15340  6,1340 0  6,7317 0               TC      WDAGAIN

                        DSPLW           EQU     *
15341  6,1341 4  2,4675 0               CS      VD1             ; to block numerical characters, clears
15342  6,1342 5  0,0466 0               TS      DSPCOUNT
15343  6,1343 0  0,0412 0               TC      WDRET           ; * return

                        DSPMSK          EQU     SEVEN

                        DSP2BIT         EQU     *
15344  6,1344 5  0,0020 0               TS      CYR
15345  6,1345 3  0,0001 0               XCH     Q
15346  6,1346 5  0,0412 0               TS      WDRET
15347  6,1347 3  1,2051 1               CAF     ONE
15350  6,1350 5  0,0434 1               TS      WDCNT
15351  6,1351 4  0,0020 1               CS      CYR
15352  6,1352 4  0,0020 1               CS      CYR
15353  6,1353 3  0,0020 0               XCH     CYR
15354  6,1354 5  0,0022 1               TS      CYL
15355  6,1355 0  6,7324 0               TC      WDAGAIN+5



                        BANK41_6        EQU     *

                                        ORG     BANK40_6        ; COLOSSUS pp. 356-358
                                        INCL    bank40_6.asm
                        ;================================================================================
                        ; DISPLAY ROUTINES (file:bank40_6.asm)
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, pp. 356-358.
                        ;================================================================================

                        ;--------------------------------------------------------------------------
                        ; DSPIN -- DISPLAY RELAY CODE
                        ;
                        ; For DSPIN, place 0-25 oct into COUNT to select the character (same as DSPCOUNT),
                        ; 5 bit relay code into CODE. Both are destroyed. If bit 14 of COUNT is 1, sign is
                        ; blanked with left char.
                        ; For DSPIN11, place 0,1 into CODE, 2 into COUNT, rel address of DSPTAB entry
                        ; into DSREL.
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.356.
                        ;--------------------------------------------------------------------------

                        DSPIN           EQU     *
13161  5,1161 3  0,0001 0               XCH     Q               ; cant use L for RETURN, since many of the
13162  5,1162 5  0,0411 0               TS      DSEXIT          ; routines calling DSPIN use L as RETURN
```

```
                        ; Set DSREL to index into DSPTAB; the index corresponds to the display character
                        ; referenced by COUNT (which is derived from DSPCOUNT)

13163  5,1163 3  2,4664 1               CAF      LOW5
13164  5,1164 7  0,0440 0               MASK     COUNT
13165  5,1165 5  0,0021 1               TS       SR             ; divides by 2
13166  5,1166 3  0,0021 1               XCH      SR
13167  5,1167 5  0,0436 0               TS       DSREL

                        ; Check COUNT (derived from DSPCOUNT) to find whether the character to be
                        ; displayed is in the right (Bits 5-1) or left (Bits 10-6) bits of the
                        ; DSPTAB word.

13170  5,1170 3  1,2100 1               CAF      BIT1
13171  5,1171 7  0,0440 0               MASK     COUNT
13172  5,1172 1  0,0000 0               CCS      A
13173  5,1173 0  5,7175 0               TC       *+2            ; >0, left if COUNT is odd
13174  5,1174 0  5,7205 1               TC       DSPIN1-1       ; +0, right if COUNT is even

                        ; Character to be displayed should be in the left bits (Bit 10-6), so
                        ; shift it left into bits 10-6.

13175  5,1175 3  0,0421 0               XCH      CODE
13176  5,1176 0  2,4656 0               TC       SLEFT5         ; does not use CYL
13177  5,1177 5  0,0421 0               TS       CODE

                        ; Set COUNT as an enumerated type; tells how to mask the new character
                        ; into the relay word.
                        ; 0 = mask new character into right side of relayword (bits 5-1)
                        ; 1 = mask into left side (bits 10-6) and leave old sign (bit 11) alone.
                        ; 2 = mask into left side (bits 10-6) and blank sign bit (bit 11)

13200  5,1200 3  1,2063 0               CAF      BIT14
13201  5,1201 7  0,0440 0               MASK     COUNT
13202  5,1202 1  0,0000 0               CCS      A
13203  5,1203 3  1,2052 1               CAF      TWO            ; >0, BIT14 = 1, blank sign
13204  5,1204 6  1,2051 1               AD       ONE            ; +0, BIT14 = 0, leave sign alone

13205  5,1205 5  0,0440 1               TS       COUNT

                        ; New display character in CODE has been bit-shifted into the correct (left
                        ; or right) bit position. All other bits are zeroed.

                        DSPIN1         EQU      *
13206  5,1206 2  0,0000 0               INHINT

                        ; Get the existing display word from DSPTAB. Words that have already been
                        ; displayed will be positive; words yet to be displayed will be negative.
                        ; Use CCS to load the absolute value of the display word. Since CCS decrements
                        ; it, we need to add 1 to restore the value.

13207  5,1207 2  0,0436 1               INDEX    DSREL
13210  5,1210 1  0,0512 0               CCS      DSPTAB
13211  5,1211 0  5,7213 0               TC       *+2            ; >0, old word already displayed
13212  5,1212 0  5,7245 0               TC       DSLV           ; +0, illegal DSPCOUNT (was TC CCSHOLE)
13213  5,1213 6  1,2051 1               AD       ONE            ; <0, old word not displayed yet

13214  5,1214 5  0,0437 1               TS       DSMAG          ; store the old relay word

                        ; Now, mask off the portion of the old relay word corresponding to the
                        ; new character. Subtract the new character from the old to see whether
                        ; they are the same.

13215  5,1215 2  0,0440 0               INDEX    COUNT
13216  5,1216 7  5,7247 0               MASK     DSMSK          ; mask with 00037, 01740, 02000, or 03740
13217  5,1217 2  0,0000 1               EXTEND
13220  5,1220 6  0,0421 0               SU       CODE

                        ; Old code same as new code? If so, we don't need to redisplay it.

13221  5,1221 1  0,0000 0               CCS      A              ; was BZF DSLV in Block II
13222  5,1222 0  5,7226 0               TC       DFRNT          ; >0
13223  5,1223 0  5,7245 0               TC       DSLV           ; +0, same, so return
13224  5,1224 0  5,7226 0               TC       DFRNT          ; <0
13225  5,1225 0  5,7245 0               TC       DSLV           ; -0, same, so return

                        ; New code is different.

                        DFRNT          EQU      *              ; different
13226  5,1226 2  0,0440 0               INDEX    COUNT
13227  5,1227 4  5,7247 0               CS       DSMSK          ; mask with 77740, 76037, 75777, or 74037
13230  5,1230 7  0,0437 0               MASK     DSMAG
13231  5,1231 6  0,0421 0               AD       CODE

                        ; Store new DSPTAB word and get the old (previous) word. If the old word is
                        ; negative, it had not been displayed yet, so NOUT (the count of undisplayed
```

```
                        ; words) has already been incremented for this DSPTAB word. If the old word
                        ; is positive, it has already been displayed, so we need to increment NOUT
                        ; to tell DSPOUT to display the new word.

13232  5,1232 4  0,0000 0               CS       A
13233  5,1233 2  0,0436 1               INDEX    DSREL
13234  5,1234 3  0,0512 1               XCH      DSPTAB

13235  5,1235 1  0,0000 0               CCS      A              ; was BZMF DSLV in Block II
13236  5,1236 0  5,7242 1               TC       *+4            ; >0
13237  5,1237 0  5,7241 1               TC       *+2            ; +0, DSPTAB entry was -
13240  5,1240 0  5,7241 1               TC       *+1            ; <0, DSPTAB entry was -
13241  5,1241 0  5,7245 0               TC       DSLV           ; -0, DSPTAB entry was -

13242  5,1242 3  0,0505 1               XCH      NOUT           ; DSPTAB entry was + (was INCR NOUT in Block
II)
13243  5,1243 6  1,2051 1               AD       ONE
13244  5,1244 5  0,0505 1               TS       NOUT

13245  5,1245 2  0,0000 1 DSLV          RELINT
13246  5,1246 0  0,0411 0               TC       DSEXIT         ; return


                        DSMSK          EQU      *
13247  5,1247     00037 0               DS       %00037         ; COUNT=0
13250  5,1250     01740 0               DS       %01740         ; COUNT=1
13251  5,1251     02000 0               DS       %02000         ; COUNT=2
13252  5,1252     03740 1               DS       %03740         ; COUNT=3

                        ; For 11DSPIN, put rel address of DSPTAB entry into A, 1 in BIT11 or 0 in
                        ; BIT11 of CODE. I changed the name to _11DSPIN because my assembler doesn't
                        ; like labels that start with a digit.

                        _11DSPIN       EQU      *
13253  5,1253 5  0,0436 0               TS       DSREL
13254  5,1254 3  1,2052 1               CAF      TWO
13255  5,1255 5  0,0440 1               TS       COUNT
13256  5,1256 3  0,0001 0               XCH      Q              ; must use same return as DSPIN
13257  5,1257 5  0,0411 0               TS       DSEXIT
13260  5,1260 0  5,7206 1               TC       DSPIN1

                        DSPOCTIN       EQU      *
13261  5,1261 0  5,7161 0               TC       DSPIN          ; so DSPOCTWD doesn't use SWCALL
13262  5,1262 3  5,7264 0               CAF      *+2
13263  5,1263 0  1,3712 0               TC       BANKJUMP
13264  5,1264     15337 1 ENDSPOCT      DS       OCTBACK


                        ; DSPALARM finds TC NVSUBEND in ENTRET for NVSUB initiated routines.
                        ; Abort with 01501.
                        ; DSPALARM finds TC ENDOFJOB in ENTRET for keyboard initiated routines.
                        ; do TC ENTRET.

                        PREDSPAL       EQU      *
13265  5,1265 4  2,4675 0               CS       VD1
13266  5,1266 5  0,0466 0               TS       DSPCOUNT

                        DSPALARM       EQU      *
13267  5,1267 4  5,7314 1               CS       NVSBENDL
13270  5,1270 6  0,0433 0               AD       ENTEXIT

13271  5,1271 1  0,0000 0               CCS      A              ; was BZF CHARALRM+2 in Block II
13272  5,1272 0  5,7276 0               TC       *+4            ; >0
13273  5,1273 0  5,7275 0               TC       *+2            ; +0
13274  5,1274 0  5,7276 0               TC       *+2            ; <0
13275  5,1275 0  5,7311 0               TC       CHARALRM+2     ; -0

13276  5,1276 4  5,7313 0               CS       MONADR         ; if this is a monitor, kill it
13277  5,1277 6  0,0433 0               AD       ENTEXIT

13300  5,1300 1  0,0000 0               CCS      A              ; was BZF *+2 in Block II
13301  5,1301 0  5,7305 0               TC       *+4            ; >0
13302  5,1302 0  5,7304 1               TC       *+2            ; +0
13303  5,1303 0  5,7305 0               TC       *+2            ; <0
13304  5,1304 0  5,7306 0               TC       *+2            ; -0

13305  5,1305 0  5,7307 1               TC       *+2
13306  5,1306 0  2,4536 0               TC       KILMONON

                        CHARALRM       EQU      *
13307  5,1307 0  2,4701 0               TC       FALTON         ; not NVSUB initiated, turn on OPR error
13310  5,1310 0  1,2723 0               TC       ENDOFJOB

13311  5,1311 0  2,5050 1               TC       POODOO
13312  5,1312     01501 1               DS       %01501
13313  5,1313     04435 1 MONADR        DS       PASTEVB
13314  5,1314 0  2,4532 1 NVSBENDL      TC       NVSUBEND
```

```
                        BANK40_7        EQU     *

                                        ORG     BANKFF_2        ; COLOSSUS pp. 358
                                        INCL    bankff_2.asm
                        ;================================================================================
                        ; DISPLAY ROUTINES (file:bankff_2.asm)
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, pp. 358.
                        ;================================================================================

                        ;--------------------------------------------------------------------------------
                        ; ALMCYCLE
                        ; Turns on check fail light, redisplays the original verb that was executed,
                        ; and recycles to execute the original verb/noun combination that was last
                        ; executed. Used for bad data during load verbs and by MCTBS. Also by MMCHANG
                        ; if 2 numerical chars were not punched in for MM code.
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.358.
                        ;--------------------------------------------------------------------------------

                        ALMCYCLE        EQU     *
04474    4474 0  2,4701 0               TC      FALTON          ; turn on check fail light
04475    4475 4  0,0530 0               CS      VERBSAVE        ; get original verb that was executed
04476    4476 5  0,0502 0               TS      REQRET          ; set for ENTPAS0
04477    4477 0  1,3565 1               TC      BANKCALL        ; puts original verb into VERBREG and
04500    4500     14326 0               DS      UPDATVB-1       ; displays it in verb lights
04501    4501 0  1,3653 1               TC      POSTJUMP
04502    4502     14002 0 ENDALM        DS      ENTER

                        BANKFF_3        EQU     *

                                        ORG     BANK41_6        ; COLOSSUS pp. 359-360
                                        INCL    bank41_6.asm
                        ;================================================================================
                        ; DISPLAY ROUTINES (file:bank41_6.asm)
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, pp. 359-360.
                        ;================================================================================

                        ;--------------------------------------------------------------------------------
                        ; MMCHANG -- MAJOR MODE CHANGE
                        ; Uses noun display until ENTER; then it uses MODE display. It goes to
                        ; MODROUT with the new MM code in A, but not displayed in MM lights.
                        ; It demands 2 numerical characters be punched in for new MM code.
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.359.
                        ;--------------------------------------------------------------------------------

                        MMCHANG         EQU     *
15356  6,1356 0  6,7404 0               TC      REQMM           ; ENTPASHI assumes the TC GRQMM at MMCHANG
                        ; if this moves at all, must change
                        ; MMADREF at ENTPASHI
15357  6,1357 3  1,2074 0               CAF     BIT5            ; OCT 20 = ND2
15360  6,1360 6  0,0466 0               AD      DSPCOUNT        ; DSPCOUNT must = -ND2

15361  6,1361 1  0,0000 0               CCS     A               ; was BZF *+2 in Block II
15362  6,1362 0  6,7366 0               TC      *+4             ; >0
15363  6,1363 0  6,7365 0               TC      *+2             ; +0
15364  6,1364 0  6,7366 0               TC      *+2             ; <0
15365  6,1365 0  6,7367 1               TC      *+2             ; -0

15366  6,1366 0  2,4474 1               TC      ALMCYCLE        ; DSPCOUNT not -ND2. Alarm and recycle.

15367  6,1367 3  1,2050 0               CAF     ZERO
15370  6,1370 3  0,0471 0               XCH     NOUNREG
15371  6,1371 5  0,0130 0               TS      MPAC

15372  6,1372 3  2,4676 1               CAF     ND1
15373  6,1373 5  0,0466 0               TS      DSPCOUNT

15374  6,1374 0  1,3565 1               TC      BANKCALL
15375  6,1375     12540 0               DS      _2BLANK

15376  6,1376 4  2,4675 0               CS      VD1             ; block num char in
15377  6,1377 5  0,0466 0               TS      DSPCOUNT

15400  6,1400 3  1,2050 0               CAF     ZERO            ; was CA MPAC in Block II
15401  6,1401 6  0,0130 0               AD      MPAC
15402  6,1402 0  1,3653 1               TC      POSTJUMP
15403  6,1403     10000 0               DS      MODROUTR        ; go thru standard loc.

                        MODROUTR        EQU     V37
```

```
                                REQMM           EQU     *
15404  6,1404 4  0,0001 1                      CS      Q
15405  6,1405 5  0,0502 0                      TS      REQRET
15406  6,1406 3  2,4676 1                      CAF     ND1
15407  6,1407 5  0,0466 0                      TS      DSPCOUNT
15410  6,1410 3  1,2050 0                      CAF     ZERO
15411  6,1411 5  0,0471 0                      TS      NOUNREG
15412  6,1412 0  1,3565 1                      TC      BANKCALL
15413  6,1413     12540 0                      DS      _2BLANK
15414  6,1414 0  2,4760 1                      TC      FLASHON
15415  6,1415 3  1,2051 1                      CAF     ONE
15416  6,1416 5  0,0467 1                      TS      DECBRNCH        ; set for dec
15417  6,1417 0  0,0433 0                      TC      ENTEXIT


                                ;-----------------------------------------------------------------------
                                ; VBRQEXEC -- REQUEST EXECUTIVE
                                ;
                                ; Enters request to EXEC for any address with any priority. It does ENDOFJOB
                                ; after entering request. Display syst is released. It assumes NOUN 26 has been
                                ; preloaded with:
                                ; COMPONENT 1 -- priority (bits 10-14), bit1=0 for NOVAC, bit1=1 for FINDVAC
                                ; COMPONENT 2 -- job CADR (14 bit; was 12 bit in Block II)
                                ; COMPONENT 3 -- not used (was BBCON in Block II)
                                ;
                                ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                                ; Oct 28, 1968, p.360.
                                ;-----------------------------------------------------------------------

                                VBRQEXEC        EQU     *
15420  6,1420 3  1,2100 1                      CAF     BIT1
15421  6,1421 7  0,0534 1                      MASK    DSPTEM1
15422  6,1422 1  0,0000 0                      CCS     A
15423  6,1423 0  6,7444 1                      TC      SETVAC          ; if bit1=1, FINDVAC
15424  6,1424 3  2,4667 1                      CAF     TCNOVAC         ; if bit1=0, NOVAC

                                ; sets up to call NOVAC or FINDVAC thru MPAC as follows:
                                ; MPAC          = TC NOVAC
                                ; MPAC+1        = job CADR
                                ; MPAC+2        = TC ENDOFJOB
                                ; MPAC+3        = temp store for job PRIO

                                REQEX1          EQU     *
15425  6,1425 5  0,0130 0                      TS      MPAC            ; TC NOVAC or TC FINDVAC into MPAC
15426  6,1426 4  1,2100 0                      CS      BIT1
15427  6,1427 7  0,0534 1                      MASK    DSPTEM1
15430  6,1430 5  0,0133 0                      TS      MPAC+3          ; PRIO into MPAC+3 as a temp (was +4)

                                REQUESTC        EQU     *
15431  6,1431 0  2,5003 1                      TC      RELDSP
15432  6,1432 3  1,2050 0                      CAF     ZERO            ; was CA ENDINST in Block II
15433  6,1433 6  2,4553 0                      AD      ENDINST
15434  6,1434 5  0,0132 1                      TS      MPAC+2          ; TC ENDOFJOB into MPAC+2 (was +3)

15435  6,1435 3  1,2050 0                      CAF     ZERO            ; set BBCON for Block II dropped
15436  6,1436 6  0,0535 1                      AD      DSPTEM1+1       ; job adres into MPAC+1
15437  6,1437 5  0,0131 1                      TS      MPAC+1

15440  6,1440 3  1,2050 0                      CAF     ZERO            ; was CA MPAC+4 in Block II
15441  6,1441 6  0,0133 0                      AD      MPAC+3          ; PRIO in A
15442  6,1442 2  0,0000 0                      INHINT
15443  6,1443 0  0,0130 0                      TC      MPAC

                                SETVAC          EQU     *
15444  6,1444 3  2,4671 0                      CAF     TCFINDVAC
15445  6,1445 0  6,7425 0                      TC      REQEX1


                                ;-----------------------------------------------------------------------
                                ; VBRQWAIT -- REQUEST WAITLIST
                                ;
                                ; Enters request to WAITLIST for any address with any delay. It does ENDOFJOB
                                ; after entering request. Display syst is released. It assumes NOUN 26 has been
                                ; preloaded with:
                                ; COMPONENT 1 -- delay (low bits)
                                ; COMPONENT 2 -- task CADR (14 bit; was 12 bit in Block II)
                                ; COMPONENT 3 -- not used (was BBCON in Block II)
                                ;
                                ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                                ; Oct 28, 1968, p.360.
                                ;-----------------------------------------------------------------------

                                VBRQWAIT        EQU     *
15446  6,1446 3  2,4670 1                      CAF     TCWAIT
15447  6,1447 5  0,0130 0                      TS      MPAC            ; TC WAITLIST into MPAC
```

```
15450  6,1450 3  1,2050 0                CAF      ZERO           ; was CA DSPTEM1 in Block II
15451  6,1451 6  0,0534 0                AD       DSPTEM1        ; time delay
15452  6,1452 0  6,7430 1 ENDRQWT        TC       REQUESTC-1

                           ; REQUESTC will put task address in MPAC+1, TC ENDOFJOB in MPAC+2.
                           ; It will take the time delay out of MPAC+3 and leave it in A, INHINT
                           ; and TC MPAC.

                           BANK41_7       EQU      *

                                          ORG      BANK40_7       ; COLOSSUS pp. 360-362
                                          INCL     bank40_7.asm
                           ;================================================================================
                           ; DISPLAY ROUTINES (file:bank40_7.asm)
                           ;
                           ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                           ; Oct 28, 1968, pp. 360-362.
                           ;================================================================================

                           ;--------------------------------------------------------------------------------
                           ; VBPROC -- PROCEED WITHOUT DATA
                           ; VBTERM -- TERMINATE
                           ;
                           ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                           ; Oct 28, 1968, p.360.
                           ;--------------------------------------------------------------------------------

                           VBPROC         EQU      *
13315  5,1315 3  1,2051 1                CAF      ONE            ; proceed without data
13316  5,1316 5  0,0503 1                TS       LOADSTAT
13317  5,1317 0  2,4536 0                TC       KILMONON       ; turn on kill monitor bit
13320  5,1320 0  2,5003 1                TC       RELDSP
13321  5,1321 0  2,4770 0                TC       FLASHOFF
13322  5,1322 0  5,7413 0                TC       RECALTST       ; see if there is any recall from endidle

                           VBTERM         EQU      *
13323  5,1323 4  1,2051 0                CS       ONE
13324  5,1324 0  5,7316 1                TC       VBPROC+1       ; term verb sets loadstat neg

                           ;--------------------------------------------------------------------------------
                           ; VBRESEQ
                           ; Wakes ENDIDLE at same line as final enter of load (L+3). Main use is
                           ; intended as response to internally initiated flashing displays in ENDIDLE.
                           ; Should not be used with load verbs, please perform, or please mark verbs
                           ; because they already use L+3 in another context.
                           ;
                           ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                           ; Oct 28, 1968, p.361.
                           ;--------------------------------------------------------------------------------

                           VBRESEQ        EQU      *
13325  5,1325 4  1,2050 1                CS       ZERO           ; make it look like data in.
13326  5,1326 0  5,7316 1                TC       VBPROC+1

                           ; flash is turned off by proceed without data, terminate,
                           ; resequence, end of load.


                           ;--------------------------------------------------------------------------------
                           ; VBRELDSP
                           ; This routine always turns off the UPACT light and always clears
                           ; DSPLOCK.
                           ;
                           ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                           ; Oct 28, 1968, p.362.
                           ;--------------------------------------------------------------------------------

                           VBRELDSP       EQU      *

                           ; some code here to turn off the UPACT light is omitted

13327  5,1327 1  0,0412 1                CCS      _2122REG       ; old DSPLOCK
13330  5,1330 3  1,2063 0                CAF      BIT14
13331  5,1331 7  0,0510 1                MASK     MONSAVE1       ; external monitor bit (EMB)
13332  5,1332 1  0,0000 0                CCS      A
13333  5,1333 0  5,7342 0                TC       UNSUSPEN       ; old DSPLOCK and EMB both 1, unsuspend

13334  5,1334 0  2,5003 1 TSTLTS4        TC       RELDSP         ; not unsuspending external monitor,
13335  5,1335 1  0,0531 1                CCS      CADRSTOR       ; release display system and
13336  5,1336 0  5,7340 1                TC       *+2            ; do reestablish if CADRSTOR is full
13337  5,1337 0  1,2723 0                TC       ENDOFJOB
13340  5,1340 0  1,3653 1                TC       POSTJUMP
13341  5,1341     05067 0                CADR     PINBRNCH

                           UNSUSPEN       EQU      *
13342  5,1342 3  1,2050 0                CAF      ZERO           ; external monitor is suspended
13343  5,1343 5  0,0501 0                TS       DSPLOCK        ; just unsuspend it by clearing DSPLOCK
```

```
13344  5,1344 1  0,0531 1                CCS       CADRSTOR       ; turn key release light off if both
13345  5,1345 0  1,2723 0                TC        ENDOFJOB       ; CADRSTOR and DSPLIST are empty
13346  5,1346 0  2,5026 0                TC        RELDSP1
13347  5,1347 0  1,2723 0                TC        ENDOFJOB


                          BANK40_8       EQU       *

                                         ORG       BANKFF_3       ; COLOSSUS pp. 363-364
                                         INCL      bankff_3.asm
                          ;=============================================================================
                          ; DISPLAY ROUTINES (file:bankff_3.asm)
                          ;
                          ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                          ; Oct 28, 1968, pp. 363-364.
                          ;=============================================================================

                          ; COLOSSUS p. 364 - comments are taken from the Block I flow charts with some
                          ; additional annotations by me.


                          NVSUB          EQU       *
04503    4503 5  0,0565 1                TS        NVSUB_A        ; more gymnastics for Block II conversion

04504    4504 3  1,2050 0                CAF       ZERO           ; was LXCH 7 in Block II
04505    4505 5  0,0564 0                TS        NVSUB_L        ; zero NVMONOPT options

                          ; save C(A). C(A) should be holding the noun/verb code; C(L) should
                          ; be holding NVMONOPT options. In this Block I version, the NVMONOPT
                          ; options should be placed in NVSUB_L before calling NVMONOPT.

04506    4506 3  0,0565 1                XCH       NVSUB_A

                          NVMONOPT       EQU       *
04507    4507 5  0,0420 1                TS        NVTEMP

                          ; Test DSPLOCK (+NZ=busy; +0=display system available)
                          ; Display is blocked by DSPLOCK=1 or external monitor bit set (bit 14)

04510    4510 3  1,2063 0                CAF       BIT14
04511    4511 7  0,0510 1                MASK      MONSAVE1       ; external monitor bit
04512    4512 6  0,0501 0                AD        DSPLOCK
04513    4513 1  0,0000 0                CCS       A
04514    4514 0  0,0001 0                TC        Q              ; dsp syst blocked, ret to 1, calling loc

                          ; Store calling line +2 in NVQTEM

04515    4515 3  1,2051 1                CAF       ONE            ; dsp syst available
04516    4516 6  0,0001 0 NVSBCOM        AD        Q
04517    4517 5  0,0526 0                TS        NVQTEM         ; 2+calling loc into NVQTEM

                          ; Force bit 15 of MONSAVE to 1, turn off bit 14.

04520    4520 3  0,0564 0                XCH       NVSUB_L        ; was LXCH MONSAVE2 in Block II
04521    4521 3  0,0511 1                XCH       MONSAVE2       ; store NVMONOPT options
04522    4522 5  0,0564 0                TS        NVSUB_L        ; replaces LXCH by working through A instead

04523    4523 0  2,4536 0                TC        KILMONON       ; turn on kill monitor bit

                          ; Store calling bank in NVBNKTEM
                          ; ** this was changed quite a bit from Block II **

                          NVSUBCOM       EQU       *
04524    4524 3  1,2050 0                CAF       ZERO
04525    4525 6  0,0015 0                AD        BANK
04526    4526 5  0,0527 1                TS        NVBNKTEM

04527    4527 0  1,3624 1                TC        MYBANKCALL     ; go to NVSUB1 thru standard loc
04530    4530     14000 1                CADR      NVSUBR

04531    4531     15505 0 NVSRRBNK       CADR      NVSUB1         ; ****** WHAT'S THIS FOR?? ********


                          ; Restore calling bank and TC NVQTEM
                          ; ** this was changed quite a bit from Block II **

                          NVSUBEND       EQU       *
04532    4532 3  1,2050 0                CAF       ZERO
04533    4533 6  0,0527 1                AD        NVBNKTEM
04534    4534 5  0,0015 0                TS        BANK           ; restore calling bank
04535    4535 0  0,0526 0                TC        NVQTEM

                          BANKFF_4       EQU       *

                                         ORG       BANK41_7       ; COLOSSUS pp. 365-366
                                         INCL      bank41_7.asm
                          ;=============================================================================
```

```
                        ; DISPLAY ROUTINES (file:bank41_7.asm)
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, pp. 365-366.
                        ;================================================================================


                        ; BLANKDSP blanks display according to option number in NVTEMP as follows:
                        ; -4 full blank, -3 leave mode, -2 leave mode and verb, -1 blank R-S only

                        BLANKDSP        EQU     *
15453  6,1453 6  1,2057 1               AD      SEVEN           ; 7,8,9,or 10 (A had 0,1,2,or 3)
15454  6,1454 2  0,0000 0               INHINT
15455  6,1455 5  0,0421 0               TS      CODE            ; blank specified DSPTABS
15456  6,1456 4  1,2065 1               CS      BIT12
15457  6,1457 2  0,0421 1               INDEX   CODE
15460  6,1460 3  0,0512 1               XCH     DSPTAB

15461  6,1461 1  0,0000 0               CCS     A
15462  6,1462 0  6,7501 1               TC      INCR_NOUT       ; was INCR NOUT in Block II
15463  6,1463 0  6,7464 0 INCR_NOUT_RET TC      *+1

15464  6,1464 1  0,0421 1               CCS     CODE
15465  6,1465 0  6,7455 1               TC      BLANKDSP+2
15466  6,1466 2  0,0000 1               RELINT
15467  6,1467 2  0,0420 0               INDEX   NVTEMP
15470  6,1470 0  6,7475 0               TC      *+5
15471  6,1471 0  6,7472 1               TC      *+1             ; NVTEMP has -4 (never touch MODREG)
15472  6,1472 5  0,0470 1               TS      VERBREG         ;            -3
15473  6,1473 5  0,0471 0               TS      NOUNREG         ;            -2
15474  6,1474 5  0,0504 0               TS      CLPASS          ;            -1

15475  6,1475 4  2,4675 0               CS      VD1
15476  6,1476 5  0,0466 0               TS      DSPCOUNT

15477  6,1477 0  2,4770 0               TC      FLASHOFF        ; protect against invisible flash
15500  6,1500 0  6,7540 1               TC      ENTSET-2        ; zeroes REQRET

                        INCR_NOUT       EQU     *
15501  6,1501 3  0,0505 1               XCH     NOUT            ; was INCR NOUT in Block II
15502  6,1502 6  1,2051 1               AD      ONE             ; have to make it a separate routine
15503  6,1503 5  0,0505 1               TS      NOUT            ; because it was nested inside
15504  6,1504 0  6,7463 1               TC      INCR_NOUT_RET   ; a CCS.

                        NVSUB1          EQU     *
15505  6,1505 3  6,7542 0               CAF     ENTSET          ; in bank
15506  6,1506 5  0,0433 0               TS      ENTRET          ; set return to NVSUBEND

15507  6,1507 1  0,0420 0               CCS     NVTEMP          ; what now
15510  6,1510 0  6,7514 0               TC      *+4             ; normal NVSUB call (execute VN or paste)
15511  6,1511 0  6,6341 1               TC      GODSPALM
15512  6,1512 0  6,7453 1               TC      BLANKDSP        ; blank display as specified
15513  6,1513 0  6,6341 1               TC      GODSPALM

15514  6,1514 3  1,2101 0               CAF     LOW7
15515  6,1515 7  0,0420 0               MASK    NVTEMP
15516  6,1516 5  0,0133 0               TS      MPAC+3          ; temp for noun (can't use MPAC, DSPDECVN
                        ;                 uses MPAC, +1, +2
15517  6,1517 3  1,2050 0               CAF     ZERO            ; was CA NVTEMP
15520  6,1520 6  0,0420 1               AD      NVTEMP

15521  6,1521 5  0,0020 0               TS      CYR             ; shift right 7, was TS EDOP, CA EDOP in BII
15522  6,1522 4  0,0020 1               CS      CYR
15523  6,1523 4  0,0020 1               CS      CYR
15524  6,1524 4  0,0020 1               CS      CYR
15525  6,1525 4  0,0020 1               CS      CYR
15526  6,1526 4  0,0020 1               CS      CYR
15527  6,1527 4  0,0020 1               CS      CYR
15530  6,1530 3  0,0020 0               XCH     CYR
15531  6,1531 7  1,2101 1               MASK    LOW7
15532  6,1532 5  0,0134 1               TS      MPAC+4          ; temp for verb (can't use MPAC, DSPDECVN
                        ;                 uses MPAC, +1, +2

15533  6,1533 1  0,0133 1               CCS     MPAC+3          ; test noun (+NZ or +0)
15534  6,1534 0  6,7543 1               TC      NVSUB2          ; if noun not +0, DC on

15535  6,1535 3  1,2050 0               CAF     ZERO            ; was CA MPAC+4 in Block II
15536  6,1536 6  0,0134 1               AD      MPAC+4
15537  6,1537 0  6,6326 0               TC      UPDATVB-1       ; if noun = +0, display verb then return

15540  6,1540 3  1,2050 0               CAF     ZERO            ; zero REQRET so that pasted verbs can
15541  6,1541 5  0,0502 0               TS      REQRET          ; be executed by operator

15542  6,1542 0  2,4532 1 ENTSET        TC      NVSUBEND
```

```
15543  6,1543 1  0,0134 0 NVSUB2         CCS      MPAC+4          ; test verb (+NZ or +0)
15544  6,1544 0  6,7551 1                TC       *+5             ; if verb not +0, go on

15545  6,1545 3  1,2050 0                CAF      ZERO            ; was CA MPAC+3 in Block II
15546  6,1546 6  0,0133 0                AD       MPAC+3
15547  6,1547 0  6,6306 1                TC       UPDATNN-1       ; if verb = +0, display noun, then return
15550  6,1550 0  2,4532 1                TC       NVSUBEND

15551  6,1551 3  1,2050 0                CAF      ZERO            ; was CA MPAC+2 in Block II
15552  6,1552 6  0,0132 1                AD       MPAC+2          ; temp for mach CADR to be spec, (DSPDECVN
15553  6,1553 5  0,0135 0                TS       MPAC+5          ;              uses MPAC, +1, +2

15554  6,1554 3  1,2050 0                CAF      ZERO            ; was CA MPAC+4 in Block II
15555  6,1555 6  0,0134 1                AD       MPAC+4
15556  6,1556 0  6,6326 0                TC       UPDATVB-1       ; if both noun and verb not +0, display

15557  6,1557 3  1,2050 0                CAF      ZERO            ; was CA MPAC+3 in Block II
15560  6,1560 6  0,0133 0                AD       MPAC+3          ; both and go to ENTPAS0
15561  6,1561 0  6,6306 1                TC       UPDATNN-1

15562  6,1562 3  1,2050 0                CAF      ZERO
15563  6,1563 5  0,0503 1                TS       LOADSTAT        ; set for waiting for data condition
15564  6,1564 5  0,0504 0                TS       CLPASS
15565  6,1565 5  0,0502 0                TS       REQRET          ; set request for pass 0

15566  6,1566 3  1,2050 0                CAF      ZERO            ; was CA MPAC+5 in Block II
15567  6,1567 6  0,0135 0                AD       MPAC+5          ; restores mach CADR to be spec to MPAC+2
15570  6,1570 5  0,0132 1                TS       MPAC+2          ; for use in INTMCTBS (in ENTPAS0)

15571  6,1571 0  6,6040 0 ENDNVSB1       TC       ENTPAS0

                         ; if internal mach CADR to be specified, MPAC+2 will be placed into
                         ; NOUNCADR in ENTPAS0 (INTMCTBS)


                         BANK41_8        EQU      *

                                         ORG      BANKFF_4        ; COLOSSUS pp. 366-368
                                         INCL     bankff_4.asm
                         ;================================================================================
                         ; DISPLAY ROUTINES (file:bankff_4.asm)
                         ;
                         ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                         ; Oct 28, 1968, pp. 366-368.
                         ;================================================================================


                         KILMONON        EQU      *               ; force bit 15 of MONSAVE1 to 1.
04536    4536 3  1,2062 1                CAF      BIT15           ; this is the kill monitor bit.
04537    4537 5  0,0510 0                TS       MONSAVE1        ; turn off bit 14, the external
04540    4540 0  0,0001 0                TC       Q               ; monitor bit.



                         ; COLOSSUS p. 367

                         ENDIDLE         EQU      *
04541    4541 3  0,0001 0                XCH      Q               ; was LXCH Q in Block II
04542    4542 5  0,0566 1                TS       ENDIDLE_L       ; return address into L

04543    4543 0  2,4554 1                TC       ISCADR_P0       ; abort if CADRSTOR not= +0
04544    4544 0  2,4560 0                TC       ISLIST_P0       ; abort if DSPLIST not= +0

04545    4545 3  1,2050 0                CAF      ZERO            ; was CA L in Block II
04546    4546 6  0,0566 1                AD       ENDIDLE_L       ; don't set DSPLOCK to 1 so can use
04547    4547 7  2,4674 1                MASK     LOW10           ; ENDIDLE with NVSUB initiated monitor.
04550    4550 6  0,0015 0                AD       BANK            ; same strategy for CADR as MAKECADR
04551    4551 5  0,0531 0                TS       CADRSTOR
04552    4552 0  1,2725 0                TC       JOBSLEEP

04553    4553 0  1,2723 0 ENDINST        TC       ENDOFJOB

                         ISCADR_P0       EQU      *
04554    4554 1  0,0531 1                CCS      CADRSTOR        ; aborts (code 1206 if CADRSTOR not= +0
04555    4555 0  2,4563 0                TC       DSPABORT        ; returns if CADRSTOR = +0
04556    4556 0  0,0001 0                TC       Q
04557    4557 0  2,4563 0                TC       DSPABORT

                         ISLIST_P0       EQU      *
04560    4560 1  0,0532 1                CCS      DSPLIST         ; aborts (code 1206 if DSPLIST not= +0
04561    4561 0  2,4563 0                TC       DSPABORT        ; returns if DSPLIST = +0
04562    4562 0  0,0001 0                TC       Q
04563    4563 0  2,5050 1 DSPABORT       TC       POODOO
04564    4564     01206 1                DS       %1206
```

```
                        ; BLANKSUB blanks any combination of R1, R2, R3. Call with blanking code in A.
                        ; BIT1=1 blanks R1, BIT2=1 blanks R2, BIT3=1 blanks R3. Any combination of these
                        ; three bits is accepted.
                        ;
                        ; DSPCOUNT is restored to the state it was in before BLANKSUB was executed.

                        BLANKSUB        EQU     *
04565    4565 7  1,2057 0               MASK    SEVEN
04566    4566 5  0,0420 1               TS      NVTEMP          ; store blanking code in NVTEMP
04567    4567 3  1,2063 0               CAF     BIT14
04570    4570 7  0,0510 1               MASK    MONSAVE1        ; external monitor bit
04571    4571 6  0,0501 0               AD      DSPLOCK
04572    4572 1  0,0000 0               CCS     A
04573    4573 0  0,0001 0               TC      Q               ; dsp syst blocked. Return to 1+calling loc
04574    4574 3  0,0001 0               XCH     Q               ; was INCR Q in Block II
04575    4575 6  1,2051 1               AD      ONE             ; set return for 2+calling location
04576    4576 5  0,0561 0               TS      BLANKSUB_Q      ; was TC Q in Block II

04577    4577 1  0,0420 0               CCS     NVTEMP
04600    4600 0  2,4602 1               TC      *+2             ; was TCF in Block II
04601    4601 0  0,0561 0               TC      BLANKSUB_Q      ; nothing to blank, Return to 2+calling loc


                        ; the return address+2 is now in BLANKSUB_Q. We need to call BLNKSUB1 in
                        ; in "bank 40", so we'll have to save the bank register so that we can
                        ; return to the address in BLANKSUB_Q. The block II code had a bunch of
                        ; tricky stuff involving the both bank bits and superbit. Block I doesn't
                        ; need to worry about that, so we can substitute this simplified code.
                        ; As in the Block II code, the return bank gets saved to BUF and the return
                        ; address+2 gets saved to BUF+1.

04602    4602 3  1,2050 0               CAF     ZERO
04603    4603 6  0,0561 0               AD      BLANKSUB_Q
04604    4604 3  0,0426 1               XCH     BUF+1           ; set return for 2+calling loc

04605    4605 3  1,2050 0               CAF     ZERO
04606    4606 6  0,0015 0               AD      BANK
04607    4607 3  0,0425 1               XCH     BUF             ; save return bank

04610    4610 3  2,4612 0               CAF     BSUB1ADDR
04611    4611 0  1,3526 0               TC      DXCHJUMP        ; bank jump to BLNKSUB1 rtne
04612    4612     13350 0 BSUB1ADDR     CADR    BLNKSUB1

                        ; this is my attempt to implement the return from BLNKSUB1. In BII, it executes
                        ; as part of the BLNKSUB1 routine:
                        ;     DXCH BUF
                        ;     TC SUPDXCHZ+1
                        ; to jump from the BLNKSUB1 bank to the calling bank.

                        BS_SUPDXCHZ     EQU     *
04613    4613 3  0,0425 1               XCH     BUF
04614    4614 3  0,0015 0               XCH     BANK            ; restore the calling bank bits
04615    4615 0  0,0426 1               TC      BUF+1           ; return to calling loc+2 (set in BLANKSUB)
                        BANKFF_5        EQU     *

                                        ORG     BANK04_2        ; COLOSSUS pp. 369
                                        INCL    bank04_2.asm
                        ;==============================================================================
                        ; DSPMM - DISPLAY MODREG (file: bank04_2.asm)
                        ;
                        ; DSPMM does not display MODREG directly. It puts EXEC request with
                        ; prio=CHARPRIO for DSPMMJB and returns to caller.
                        ;
                        ; If MODREG contains -0, DSPMMJB blanks the MODE lights.
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, pp. 369.
                        ;==============================================================================


                        DSPMM           EQU     *
10047  4,0047 3  0,0001 0               XCH     Q
10050  4,0050 5  0,0130 0               TS      MPAC
10051  4,0051 2  0,0000 0               INHINT
10052  4,0052 3  2,4131 0               CAF     CHRPRIO
10053  4,0053 0  1,3162 1               TC      NOVAC
10054  4,0054     13400 1               CADR    DSPMMJB
10055  4,0055 2  0,0000 1               RELINT
10056  4,0056 0  0,0130 0 ENDSPMM       TC      MPAC


                        BANK04_3        EQU     *

                                        ORG     BANK40_8        ; COLOSSUS pp. 369-371
                                        INCL    bank40_8.asm
                        ;==============================================================================
```

```
                        ; DISPLAY ROUTINES (file:bank40_8.asm)
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, pp. 369-371.
                        ;================================================================================

                        BLNKSUB1        EQU     *
13350  5,1350 3  1,2050 0               CAF     ZERO            ; was CA DSPCOUNT in Block II
13351  5,1351 6  0,0466 0               AD      DSPCOUNT        ; save old DSPCOUNT for later restoration
13352  5,1352 5  0,0427 0               TS      BUF+2

13353  5,1353 3  1,2100 1               CAF     BIT1            ; test bit 1. See if R1 to be blanked.
13354  5,1354 0  5,7373 1               TC      TESTBIT
13355  5,1355 3  2,4635 0               CAF     R1D1
13356  5,1356 0  5,6472 0               TC      _5BLANK-1

13357  5,1357 3  1,2077 0               CAF     BIT2            ; test bit 2. See if R2 to be blanked.
13360  5,1360 0  5,7373 1               TC      TESTBIT
13361  5,1361 3  2,4636 0               CAF     R2D1
13362  5,1362 0  5,6472 0               TC      _5BLANK-1

13363  5,1363 3  1,2076 1               CAF     BIT3            ; test bit 3. See if R3 to be blanked.
13364  5,1364 0  5,7373 1               TC      TESTBIT
13365  5,1365 3  2,4637 1               CAF     R3D1
13366  5,1366 0  5,6472 0               TC      _5BLANK-1

13367  5,1367 3  1,2050 0               CAF     ZERO            ; was CA BUF+2 in Block II
13370  5,1370 6  0,0427 0               AD      BUF+2           ; restore DSPCOUNT to state it had
13371  5,1371 5  0,0466 0               TS      DSPCOUNT        ; before BLANKSUB

13372  5,1372 0  2,4613 1               TC      BS_SUPDXCHZ     ; was DXCH BUF, TC SUPDXCHZ+1 in BII



                        TESTBIT         EQU     *
13373  5,1373 7  0,0420 0               MASK    NVTEMP          ; NVTEMP contains blanking code
13374  5,1374 1  0,0000 0               CCS     A
13375  5,1375 0  0,0001 0               TC      Q               ; if current bit = 1, return to L+1
13376  5,1376 2  0,0001 1               INDEX   Q               ; if current bit = 0, return to L+3
13377  5,1377 0  0,0002 0               TC      2

                        DSPMMJB         EQU     *
13400  5,1400 3  2,4677 0               CAF     MD1             ; gets here thru DSPMM
13401  5,1401 3  0,0466 0               XCH     DSPCOUNT
13402  5,1402 5  0,0435 0               TS      DSPMMTEM        ; save DSPCOUNT
13403  5,1403 1  0,0500 0               CCS     MODREG
13404  5,1404 6  1,2051 1               AD      ONE
13405  5,1405 0  5,7144 1               TC      DSPDECVN        ; if MODREG is + or +0, display MODREG
13406  5,1406 0  5,7410 0               TC      *+2             ; if MODREG is -NZ, do nothing
13407  5,1407 0  5,6540 0               TC      _2BLANK         ; if MODREG is -0, blank MM
13410  5,1410 3  0,0435 0               XCH     DSPMMTEM        ; restore DSPCOUNT
13411  5,1411 5  0,0466 0               TS      DSPCOUNT
13412  5,1412 0  1,2723 0               TC      ENDOFJOB

                        ;--------------------------------------------------------------------------
                        ; RECALTST
                        ; Entered directly after data is loaded (or resequence verb is executed),
                        ; terminate verb is executed, or proceed without data verb is executed.
                        ; It wakes up job that did TC ENDIDLE.
                        ; If CADRSTOR not = +0, it puts +0 into DSPLOCK, and turns off KEY RLSE
                        ; light if DSPLIST is empty (leaves KEY RLSE light alone if not empty).
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.370.
                        ;--------------------------------------------------------------------------

                        RECALTST        EQU     *
13413  5,1413 1  0,0531 1               CCS     CADRSTOR
13414  5,1414 0  5,7416 0               TC      RECAL1
13415  5,1415 0  1,2723 0               TC      ENDOFJOB        ; normal exit if keyboard initiated.

                        RECAL1          EQU     *
13416  5,1416 3  1,2050 0               CAF     ZERO
13417  5,1417 3  0,0531 0               XCH     CADRSTOR
13420  5,1420 2  0,0000 0               INHINT
13421  5,1421 0  1,3003 1               TC      JOBWAKE
13422  5,1422 1  0,0503 0               CCS     LOADSTAT
13423  5,1423 0  5,7450 1               TC      DOPROC          ; + proceed without data
13424  5,1424 0  1,2723 0               TC      ENDOFJOB        ; pathological case exit
13425  5,1425 0  5,7446 0               TC      DOTERM          ; - terminate
13426  5,1426 3  1,2052 1               CAF     TWO             ; -0, data in or resequence

                        RECAL2          EQU     *
13427  5,1427 2  0,0300 0               INDEX   LOCCTR
13430  5,1430 6  0,0140 1               AD      LOC             ; loc is + for basic jobs
13431  5,1431 2  0,0300 0               INDEX   LOCCTR
13432  5,1432 5  0,0140 1               TS      LOC
```

```
13433  5,1433 3  1,2050 0                CAF      ZERO           ; save verb in MPAC, noun in MPAC+1 at
13434  5,1434 6  0,0471 0                AD       NOUNREG        ; time of response to ENDIDLE for
13435  5,1435 2  0,0300 0                INDEX    LOCCTR         ; possible later testing by job that has
13436  5,1436 5  0,0131 1                TS       MPAC+1         ; been waked up

13437  5,1437 3  1,2050 0                CAF      ZERO
13440  5,1440 6  0,0470 1                AD       VERBREG
13441  5,1441 2  0,0300 0                INDEX    LOCCTR
13442  5,1442 5  0,0130 0                TS       MPAC

13443  5,1443 2  0,0000 1                RELINT

                          RECAL3         EQU      *
13444  5,1444 0  2,5003 1                TC       RELDSP
13445  5,1445 0  1,2723 0                TC       ENDOFJOB

                          DOTERM         EQU      *
13446  5,1446 3  1,2050 0                CAF      ZERO
13447  5,1447 0  5,7427 1                TC       RECAL2

                          DOPROC         EQU      *
13450  5,1450 3  1,2051 1                CAF      ONE
13451  5,1451 0  5,7427 1                TC       RECAL2
                          BANK40_8a      EQU      *

                                         ORG      BANKFF_5       ; COLOSSUS pp. 372-376
                                         INCL     bankff_5.asm
                          ;==============================================================================
                          ; DISPLAY ROUTINES (file:bankff_5.asm)
                          ;
                          ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                          ; Oct 28, 1968, pp. 372-376.
                          ;==============================================================================

                          ;------------------------------------------------------------------------------
                          ; MISCELLANEOUS SERVICE ROUTINES IN FIXED-FIXED
                          ;
                          ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                          ; Oct 28, 1968, p.372.
                          ;------------------------------------------------------------------------------

                          ; SETNCADR
                          ; Store the eraseable memory address from A into NOUNCADR and NOUNADD.
                          ; (changed from Block II, because there is no bank addressing for block I)

                          ; SETNADD
                          ; Get the eraseable memory address from NOUNCADR and store it into NOUNADD.
                          ; (changed from Block II, because there is no bank addressing for block I)
                          ;
                          ; SETEBANK
                          ; E CADR arrives in A. E ADRES is "derived" and left in A.
                          ; (changed from Block II, because there is no bank addressing for block I)

                          SETNCADR       EQU      *
04616    4616 3  0,0001 0                XCH      Q
04617    4617 5  0,0555 1                TS       SETNCADR_Q     ; save return address
04620    4620 3  0,0001 0                XCH      Q              ; restore A

04621    4621 5  0,0506 1                TS       NOUNCADR       ; store ECADR
04622    4622 7  2,4674 1                MASK     LOW10
04623    4623 5  0,0442 0                TS       NOUNADD        ; put E ADRES into NOUNADD
04624    4624 0  0,0555 1                TC       SETNCADR_Q

                          SETNADD        EQU      *
04625    4625 3  0,0001 0                XCH      Q
04626    4626 5  0,0555 1                TS       SETNCADR_Q     ; save return address
04627    4627 3  0,0001 0                XCH      Q              ; restore A

04630    4630 3  1,2050 0                CAF      ZERO
04631    4631 6  0,0506 1                AD       NOUNCADR       ; get NOUNCADR
04632    4632 0  2,4622 0                TC       SETNCADR+4

                          SETEBANK       EQU      *
04633    4633 7  2,4674 1                MASK     LOW10
04634    4634 0  0,0001 0                TC       Q



04635    4635     00016 0 R1D1           DS       %16            ; these 3 constants form a packed table
04636    4636     00011 1 R2D1           DS       %11            ; don't separate
04637    4637     00004 0 R3D1           DS       %4             ; must stay here

                          RIGHT5         EQU      *
04640    4640 5  0,0020 0                TS       CYR
04641    4641 4  0,0020 1                CS       CYR
04642    4642 4  0,0020 1                CS       CYR
```

```
04643    4643 4  0,0020 1                CS       CYR
04644    4644 4  0,0020 1                CS       CYR
04645    4645 3  0,0020 0                XCH      CYR
04646    4646 0  0,0001 0                TC       Q

                         LEFT5          EQU      *
04647    4647 5  0,0022 1                TS       CYL
04650    4650 4  0,0022 0                CS       CYL
04651    4651 4  0,0022 0                CS       CYL
04652    4652 4  0,0022 0                CS       CYL
04653    4653 4  0,0022 0                CS       CYL
04654    4654 3  0,0022 1                XCH      CYL
04655    4655 0  0,0001 0                TC       Q

                         SLEFT5         EQU      *
04656    4656 6  0,0000 1                DOUBLE
04657    4657 6  0,0000 1                DOUBLE
04660    4660 6  0,0000 1                DOUBLE
04661    4661 6  0,0000 1                DOUBLE
04662    4662 6  0,0000 1                DOUBLE
04663    4663 0  0,0001 0                TC       Q

04664    4664     00037 0 LOW5           DS       %00037          ; these 3 constants form a packed table
04665    4665     01740 0 MID5           DS       %01740          ; don't separate
04666    4666     76000 0 HI5            DS       %76000          ; must stay here

04667    4667 0  1,3162 1 TCNOVAC        TC       NOVAC
04670    4670 0  1,2232 0 TCWAIT         TC       WAITLIST
                         ;TCTSKOVR        TC       TASKOVER
04671    4671 0  1,3161 1 TCFINDVAC      TC       FINDVAC

                         ;CHRPRIO         DS       %30000           ; EXEC priority of CHARIN
04672    4672     03777 0 LOW11          DS       %3777
                         B12M1          EQU      LOW11
04673    4673     00377 1 LOW8           DS       %377
04674    4674     01777 1 LOW10          DS       %01777

04675    4675     00023 0 VD1            DS       %23             ; these 3 constants form a packed table
04676    4676     00021 1 ND1            DS       %21             ; don't separate
04677    4677     00025 0 MD1            DS       %25             ; must stay here

04700    4700     00012 1 BINCON         DS       10

                         ;**************** TURN ON/OFF OPERATOR ERROR LIGHT ******* p. 373

                         DSALMOUT       EQU      OUT1            ; channel 11 in Block II is OUT1 in Block I

                         FALTON         EQU      *
04701    4701 4  0,0011 0                CS       DSALMOUT        ; inclusive OR bit 7 with 1 using
04702    4702 7  2,4712 0                MASK     FALTOR          ; Demorgan's theorem
04703    4703 4  0,0000 0                COM
04704    4704 5  0,0011 1                TS       DSALMOUT        ; was bit 7 of channel 11 in Block II
04705    4705 0  0,0001 0                TC       Q

                         FALTOF         EQU      *
04706    4706 4  1,2072 1                CS       BIT7
04707    4707 7  0,0011 0                MASK     DSALMOUT
04710    4710 5  0,0011 1                TS       DSALMOUT        ; was bit 7 of channel 11 in Block II
04711    4711 0  0,0001 0                TC       Q

04712    4712     77677 1 FALTOR         DS       %77677          ; 1's compliment of bit 7


                         ;**************** TURN ON KEY RELEASE LIGHT ******* p. 373

                         RELDSPON       EQU      *
04713    4713 4  0,0011 0                CS       DSALMOUT        ; inclusive OR bit 5 with 1 using
04714    4714 7  2,4720 1                MASK     RELDSPOR        ; Demorgan's theorem
04715    4715 4  0,0000 0                COM
04716    4716 5  0,0011 1                TS       DSALMOUT        ; was bit 5 of channel 11 in Block II
04717    4717 0  0,0001 0                TC       Q

04720    4720     77757 1 RELDSPOR       DS       %77757          ; 1's compliment of bit 5


                         ; TPSL1
                         ; Shift triple word MPAC, MPAC+1, MPAC+2 left 1 bit

                         TPSL1          EQU      *
04721    4721 3  1,2050 0                CAF      ZERO
04722    4722 6  0,0132 1                AD       MPAC+2
04723    4723 6  0,0132 1                AD       MPAC+2
04724    4724 5  0,0132 1                TS       MPAC+2          ; skip on overflow

04725    4725 3  1,2050 0                CAF      ZERO            ; otherwise, make interword carry=0
04726    4726 6  0,0131 1                AD       MPAC+1
04727    4727 6  0,0131 1                AD       MPAC+1
```

```
04730    4730 5  0,0131 1               TS       MPAC+1          ; skip on overflow

04731    4731 3  1,2050 0               CAF      ZERO            ; otherwise, make interword carry=0
04732    4732 6  0,0130 0               AD       MPAC
04733    4733 6  0,0130 0               AD       MPAC
04734    4734 5  0,0130 0               TS       MPAC            ; skip on overflow

04735    4735 0  0,0001 0               TC       Q               ; no net OV/UF
04736    4736 5  0,0136 0               TS       MPAC+6          ; MPAC+6 set to +/- 1 for OV/UF
04737    4737 0  0,0001 0               TC       Q


                        ; PRSHRTMP
                        ; if MPAC, +1 are each +NZ or +0 and C(A)=-0, SHORTMP wrongly gives +0.
                        ; if MPAC, +1 are each -NZ or -0 and C(A)=+0, SHORTMP wrongly gives +0.
                        ; PRSHRTMP fixes first case only, by merely testing C(A) and if it = -0,
                        ; setting result to -0.
                        ; (Do not use PRSHRTMP unless MPAC, +1 are each +NZ or +0, as they are
                        ; when they contain the SF constants).

                        PRSHRTMP        EQU      *
04740    4740 5  0,0432 1               TS       MPTEMP
04741    4741 3  0,0001 0               XCH      Q
04742    4742 5  0,0600 1               TS       PRSHRTMP_Q

04743    4743 1  0,0432 0               CCS      MPTEMP
04744    4744 0  2,4754 0               TC       DOSHRTMP        ; C(A) +, do regular SHORTMP
04745    4745 0  2,4754 0               TC       DOSHRTMP        ; C(A) +0, do regular SHORTMP
04746    4746 0  2,4754 0               TC       DOSHRTMP        ; C(A) -, do regular SHORTMP
04747    4747 4  1,2050 1               CS       ZERO            ; C(A) -0, force result to -0 and return
04750    4750 5  0,0130 0               TS       MPAC
04751    4751 5  0,0131 1               TS       MPAC+1
04752    4752 5  0,0132 1               TS       MPAC+2
04753    4753 0  0,0600 1               TC       PRSHRTMP_Q

                        DOSHRTMP        EQU      *
04754    4754 3  1,2050 0               CAF      ZERO
04755    4755 6  0,0432 1               AD       MPTEMP
04756    4756 0  2,4353 0               TC       SHORTMP
04757    4757 0  0,0600 1               TC       PRSHRTMP_Q



                        ;**************** TURN ON/OFF V/N FLASH ******* p. 374
                        ; this is handled by setting a bit in channel 11 in Block II.
                        ; In Block I, it has to be set through the display table, so I
                        ; borrowed this method from SGNCOM (the DSKY +/- sign routine)
                        ; Uses MYBANKCALL because BANKCALL is not reentrant and I dont
                        ; understand its usage in COLOSSUS well enough to be certain
                        ; that FLASHON/FLASHOFF isn't being called somewhere through
                        ; BANKCALL.

                        FLASHON         EQU      *
04760    4760 3  0,0001 0               XCH      Q
04761    4761 5  0,0570 0               TS       FLASHRET

04762    4762 3  1,2066 0               CAF      BIT11
04763    4763 5  0,0421 0               TS       CODE
04764    4764 3  2,5000 1               CAF      FLSHTAB
04765    4765 0  1,3624 1               TC       MYBANKCALL
04766    4766     13253 1               CADR     _11DSPIN

04767    4767 0  0,0570 0               TC       FLASHRET

                        FLASHOFF        EQU      *
04770    4770 3  0,0001 0               XCH      Q
04771    4771 5  0,0570 0               TS       FLASHRET

04772    4772 3  1,2050 0               CAF      ZERO
04773    4773 5  0,0421 0               TS       CODE
04774    4774 3  2,5000 1               CAF      FLSHTAB
04775    4775 0  1,3624 1               TC       MYBANKCALL
04776    4776     13253 1               CADR     _11DSPIN

04777    4777 0  0,0570 0               TC       FLASHRET

05000    5000     00011 1 FLSHTAB        DS       %11             ; V/N flash



                        NVSUBUSY        EQU      *
05001    5001 0  1,3653 1               TC       POSTJUMP
05002    5002     13452 0               CADR     NVSUBSY1
```

```
                                BANKFF_5a       EQU     *

                                                ORG     BANK40_8a       ; COLOSSUS pp. 376
                                                INCL    bank40_8a.asm
                                ;================================================================================
                                ; DISPLAY ROUTINES (file:bank40_8a.asm)
                                ;
                                ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                                ; Oct 28, 1968, pp. 376.
                                ;================================================================================

                                ;--------------------------------------------------------------------------------
                                ; MISCELLANEOUS SERVICE ROUTINES IN FIXED-FIXED
                                ;
                                ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                                ; Oct 28, 1968, p.376.
                                ;--------------------------------------------------------------------------------

                                NVSUBSY1        EQU     *
13452  5,1452 5  0,0567 0                      TS      NBSUBSY1_L      ; save CADR
13453  5,1453 0  2,4554 1                      TC      ISCADR_P0       ; abort if CADRSTOR not = +0
13454  5,1454 0  2,4560 0                      TC      ISLIST_P0       ; abort if DSPLIST not = +0
13455  5,1455 0  2,4713 0                      TC      RELDSPON
13456  5,1456 3  1,2050 0                      CAF     ZERO            ; was CA L in Block II
13457  5,1457 6  0,0567 0                      AD      NBSUBSY1_L
13460  5,1460 5  0,0532 0                      TS      DSPLIST
13461  5,1461 0  1,2725 0 ENDNVBSY             TC      JOBSLEEP
                                BANK40_9        EQU     *

                                                ORG     BANKFF_5a       ; COLOSSUS pp. 376-378
                                                INCL    bankff_5a.asm
                                ;================================================================================
                                ; DISPLAY ROUTINES (file:bankff_5a.asm)
                                ;
                                ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                                ; Oct 28, 1968, pp. 376-378.
                                ;================================================================================

                                ;--------------------------------------------------------------------------------
                                ; MISCELLANEOUS SERVICE ROUTINES IN FIXED-FIXED
                                ;
                                ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                                ; Oct 28, 1968, p.376.
                                ;--------------------------------------------------------------------------------

                                ; RELDSP
                                ; used by VBPROC, VBTERM, VBRQEXEC, VBRQWAIT, VBRELDSP, EXTENDED VERB
                                ; DISPATCHER, VBRESEQ, and RECALTST.

                                ; RELDSP1
                                ; used by monitor set up, VBRELDSP

                                RELDSP          EQU     *
05003    5003 3  0,0001 0                      XCH     Q               ; set DSPLOCK to +0, turn RELDSP light
05004    5004 5  0,0441 0                      TS      RELRET          ; off, search DSPLIST
05005    5005 4  1,2063 1                      CS      BIT14
05006    5006 2  0,0000 0                      INHINT
05007    5007 7  0,0510 1                      MASK    MONSAVE1
05010    5010 5  0,0510 0                      TS      MONSAVE1        ; turn off external monitor bit
05011    5011 1  0,0532 1                      CCS     DSPLIST
05012    5012 0  2,5014 1                      TC      *+2
05013    5013 0  2,5017 1                      TC      RELDSP2         ; list empty
05014    5014 3  1,2050 0                      CAF     ZERO
05015    5015 3  0,0532 0                      XCH     DSPLIST
05016    5016 0  1,3003 1                      TC      JOBWAKE

                                RELDSP2         EQU     *
05017    5017 2  0,0000 1                      RELINT
05020    5020 4  1,2074 1                      CS      BIT5            ; turn off KEY RLSE light
05021    5021 7  0,0011 0                      MASK    DSALMOUT        ; was WAND DSALMOUT in Block II
05022    5022 5  0,0011 1                      TS      DSALMOUT

05023    5023 3  1,2050 0                      CAF     ZERO
05024    5024 5  0,0501 0                      TS      DSPLOCK
05025    5025 0  0,0441 0                      TC      RELRET

                                RELDSP1         EQU     *
05026    5026 3  0,0001 0                      XCH     Q               ; set DSPLOCK to +0, No DSPLIST search
05027    5027 5  0,0441 0                      TS      RELRET          ; turn KEY RLSE light off if DSPLIST is
                                ; empty. Leave KEY RLSE light alone if
                                ; DSPLIST is not empty.
05030    5030 1  0,0532 1                      CCS     DSPLIST
05031    5031 0  2,5033 1                      TC      *+2             ; + not empty, leave KEY RLSE light alone
05032    5032 0  2,5017 1                      TC      RELDSP2         ; +0, list empty, turn off KEY RLSE light
05033    5033 3  1,2050 0                      CAF     ZERO            ; - not empty, leave KEY RLSE light alone
```

```
05034    5034 5  0,0501 0               TS       DSPLOCK
05035    5035 0  0,0441 0               TC       RELRET


                        ;----------------------------------------------------------------------
                        ; NEWMODEA
                        ;
                        ; The new major mode is in register A. Store the major mode in MODREG and update
                        ; the major mode display.
                        ;
                        ; I couldn't find this in my COLOSSUS listing, so I borrowed it from UPDATVB-1
                        ; (but modified it to work with the major mode instead of the verb).
                        ;----------------------------------------------------------------------

                        NEWMODEA       EQU      *
05036    5036 5  0,0500 1               TS       MODREG          ; store new major mode
05037    5037 3  0,0001 0               XCH      Q
05040    5040 5  0,0572 1               TS       NEWMODEA_Q      ; save Q

05041    5041 3  2,4677 0               CAF      MD1
05042    5042 5  0,0466 0               TS       DSPCOUNT

05043    5043 3  1,2050 0               CAF      ZERO
05044    5044 6  0,0500 1               AD       MODREG

05045    5045 0  1,3565 1               TC       BANKCALL
05046    5046     13144 1               CADR     DSPDECVN

05047    5047 0  0,0572 1               TC       NEWMODEA_Q      ; return

                        ;----------------------------------------------------------------------
                        ; POODOO - Program alarm.
                        ;
                        ; Turn on program alarm light and store alarm code in FAILREG. The alarm code
                        ; is retrieved from the address pointed to by Q. The most recent code is stored
                        ; in FAILREG. Older codes are scrolled to FAILREG+1,+2. Older CADRs are
                        ; scrolled down.
                        ;
                        ; This was missing from my COLOSSUS listing, so I had to guess at the
                        ; implementation, based upon calling references in COLOSSUS, and textual
                        ; descriptions of normal noun 9 which retrieves alarm codes.
                        ;----------------------------------------------------------------------

                        POODOO         EQU      *
05050    5050 3  0,0001 0               XCH      Q
05051    5051 5  0,0130 0               TS       MPAC
05052    5052 4  0,0011 0               CS       DSALMOUT        ; inclusive OR bit 9 with 1 using
05053    5053 7  2,5066 0               MASK     NOTPALT         ; Demorgan's theorem
05054    5054 4  0,0000 0               COM
05055    5055 5  0,0011 1               TS       DSALMOUT        ; turn on PROG ALM light
05056    5056 3  0,0461 1               XCH      FAILREG+1       ; scroll previous codes down
05057    5057 5  0,0462 1               TS       FAILREG+2
05060    5060 3  0,0460 0               XCH      FAILREG
05061    5061 5  0,0461 1               TS       FAILREG+1
05062    5062 2  0,0130 1               INDEX    MPAC            ; indirectly address Q
05063    5063 3  0,0000 1               CAF      0               ; (gets alarm code)
05064    5064 5  0,0460 0               TS       FAILREG         ; store alarm code
05065    5065 0  1,2723 0               TC       ENDOFJOB

05066    5066     77377 1 NOTPALT       DS       %77377          ; 1's compliment of bit9 (PROG ALM)



                        ;----------------------------------------------------------------------
                        ; PINBRNCH
                        ;
                        ; This is supposed to restore the DSKY display to its former state in the
                        ; event of error. According to COLOSSUS, it works if you use "Margaret's"
                        ; code. I don't have that portion of the listing, so I just terminate
                        ; the job, which seems to be an acceptable work-around, even though the
                        ; old display is not restored.
                        ;----------------------------------------------------------------------

05067    5067 0  1,2723 0 PINBRNCH      TC       ENDOFJOB


                        BANKFF_6       EQU      *

                                       ORG      BANK41_8        ; COLOSSUS pp. 379-380
                                       INCL     bank41_8.asm
                        ;==============================================================================
                        ; DISPLAY ROUTINES (file:bank41_8.asm)
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, pp. 379-380.
                        ;==============================================================================
```

```
                          VBTSTLTS       EQU      *
15572  6,1572 2  0,0000 0                INHINT

                          ; heavily modified from the original Block II code...

15573  6,1573 4  0,0011 0                CS       DSALMOUT       ; turn on lights
15574  6,1574 7  6,7623 0                MASK     TSTCON1        ; inclusive OR light bits with 1's using
15575  6,1575 4  0,0000 0                COM                     ; Demorgan's theorem
15576  6,1576 5  0,0011 1                TS       DSALMOUT

15577  6,1577 3  1,2060 0                CAF      TEN

15600  6,1600 5  0,0414 0 TSTLTS1        TS       ERCNT
15601  6,1601 4  6,7621 1                CS       FULLDSP
15602  6,1602 2  0,0414 1                INDEX    ERCNT
15603  6,1603 5  0,0512 1                TS       DSPTAB

15604  6,1604 1  0,0414 1                CCS      ERCNT
15605  6,1605 0  6,7600 0                TC       TSTLTS1
15606  6,1606 4  6,7622 1                CS       FULLDSP1
15607  6,1607 5  0,0513 0                TS       DSPTAB+1       ; turn on 3 plus signs
15610  6,1610 5  0,0516 0                TS       DSPTAB+4
15611  6,1611 5  0,0520 0                TS       DSPTAB+6

15612  6,1612 3  1,2061 1                CAF      ELEVEN
15613  6,1613 5  0,0505 1                TS       NOUT

15614  6,1614 0  2,4760 1                TC       FLASHON

15615  6,1615 3  6,7624 0                CAF      SHOLTS
15616  6,1616 0  1,2232 0                TC       WAITLIST
15617  6,1617     15625 1                CADR     TSTLTS2

15620  6,1620 0  1,2723 0                TC       ENDOFJOB       ; DSPLOCK is left busy (from keyboard
                          ; action) until TSTLTS3 to ensure that
                          ; lights test will be seen.


15621  6,1621     05675 0 FULLDSP        DS       %05675         ; display all 8's
15622  6,1622     07675 1 FULLDSP1       DS       %07675         ; display all 8's and +

                          ; 1's Comp of UPTEL=bit3, KEY REL=bit5, oper err=bit7, PROG ALM=bit 9

15623  6,1623     77253 0 TSTCON1        DS       %77253

15624  6,1624     00764 1 SHOLTS         DS       %764           ; 5 sec

                          TSTLTS2        EQU      *
15625  6,1625 3  2,4131 0                CAF      CHRPRIO        ; called by WAITLIST
15626  6,1626 0  1,3162 1                TC       NOVAC
15627  6,1627     15631 1                CADR     TSTLTS3
15630  6,1630 0  1,2413 0                TC       TASKOVER

                          TSTLTS3        EQU      *
15631  6,1631 2  0,0000 0                INHINT
15632  6,1632 3  6,7623 1                CAF      TSTCON1        ; turn off lights
15633  6,1633 7  0,0011 0                MASK     DSALMOUT
15634  6,1634 5  0,0011 1                TS       DSALMOUT

15635  6,1635 2  0,0000 1                RELINT

15636  6,1636 0  1,3565 1                TC       BANKCALL       ; redisplay C(MODREG)
15637  6,1637     10047 0                CADR     DSPMM
15640  6,1640 0  2,4536 0                TC       KILMONON       ; turn on kill monitor bit
15641  6,1641 0  2,4770 0                TC       FLASHOFF       ; turn off V/N flash
15642  6,1642 0  1,3653 1                TC       POSTJUMP       ; does RELDSP and goes to PINBRNCH if
15643  6,1643     13334 1                CADR     TSTLTS4        ; ENDIDLE is awaiting operator response
                          BANK41_9       EQU      *

                                         ORG      BANK40_9       ; COLOSSUS pp. 381-382
                                         INCL     bank40_9.asm
                          ;=============================================================================
                          ; DISPLAY ROUTINES (file:bank40_9.asm)
                          ;
                          ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                          ; Oct 28, 1968, pp. 381-382.
                          ;=============================================================================

                          ;-----------------------------------------------------------------------------
                          ; ERROR - Error light reset.
                          ;
                          ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                          ; Oct 28, 1968, p.381.
                          ;-----------------------------------------------------------------------------
```

```
                                ERROR           EQU     *
13462  5,1462 3  0,0412 0                      XCH     _2122REG        ; restore original C(DSPLOCK), thus error
13463  5,1463 5  0,0501 0                      TS      DSPLOCK         ; light reset leaves DSPLOCK unchanged

                                ; omitted some stuff in COLOSSUS here

13464  5,1464 4  5,7520 0                      CS      ERCON           ; turn off UPTL, OPER ERR, PROG ALM
13465  5,1465 7  0,0011 0                      MASK    DSALMOUT
13466  5,1466 5  0,0011 1                      TS      DSALMOUT

13467  5,1467 3  2,4700 1 TSTAB                CAF     BINCON          ; dec 10
13470  5,1470 5  0,0414 0                      TS      ERCNT           ; ERCNT = count
13471  5,1471 2  0,0000 0                      INHINT
13472  5,1472 2  0,0414 1                      INDEX   ERCNT
13473  5,1473 1  0,0512 0                      CCS     DSPTAB
13474  5,1474 6  1,2051 1                      AD      ONE
13475  5,1475 0  5,7502 1                      TC      ERPLUS
13476  5,1476 6  1,2051 1                      AD      ONE

13477  5,1477 4  0,0000 0 ERMINUS              CS      A
13500  5,1500 7  5,7517 1                      MASK    NOTBIT12
13501  5,1501 0  5,7505 0                      TC      ERCOM

13502  5,1502 4  0,0000 0 ERPLUS               CS      A
13503  5,1503 7  5,7517 1                      MASK    NOTBIT12
13504  5,1504 4  0,0000 0                      CS      A

13505  5,1505 2  0,0414 1 ERCOM                INDEX   ERCNT
13506  5,1506 5  0,0512 1                      TS      DSPTAB
13507  5,1507 2  0,0000 1                      RELINT
13510  5,1510 1  0,0414 1                      CCS     ERCNT
13511  5,1511 0  5,7470 0                      TC      TSTAB+1

13512  5,1512 3  1,2050 0                      CAF     ZERO            ; clear the error codes for PROG ALM
13513  5,1513 5  0,0460 0                      TS      FAILREG
13514  5,1514 5  0,0461 1                      TS      FAILREG+1
13515  5,1515 5  0,0462 1                      TS      FAILREG+2

13516  5,1516 0  1,2723 0                      TC      ENDOFJOB

13517  5,1517     73777 1 NOTBIT12             DS      %73777


13520  5,1520     00504 0 ERCON                DS      %504            ; channel 11 bits 3,7,9
                                BANK40_10       EQU     *

                                ; end of PINBALL routines

                                ; PINBALL NOUN tables

                                                ORG     BANK42_3
                                                INCL    bank42_3.asm    ; COLOSSUS pp. 263-279
                                ;================================================================================
                                ; PINBALL NOUN TABLES (file:bank42_3.asm)
                                ;
                                ; The following routines are for reading the noun tables and the scale
                                ; factor (SF) tables (which are in a separate bank from the rest of
                                ; PINBALL). These reading routines are in the same bank as the tables.
                                ; They are called by DXCH Z (translated to DXCHJUMP for Block I).
                                ;
                                ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                                ; Oct 28, 1968, pp. 263-279.
                                ;================================================================================

                                ;--------------------------------------------------------------------------------
                                ; Noun table info from COLOSSUS, p.325
                                ;
                                ; noun code < 40 : normal noun case
                                ; noun code >= 40: mixed noun case

                                ;--------------------------------------------------------------------------------
                                ; NNADTAB:
                                ; for normal noun case, NNADTAB contains one CADR for each noun.
                                ;       +entry   = noun CADR
                                ;       +0       = noun not used.
                                ;       -entry   = machine CADR (E or F) to be specified.
                                ;       -1       = channel to be specified (not used for Block I);
                                ;       -0       = augment of last machine CADR supplied.

                                ; for mixed noun case, NNADTAB contains one indirect address (IDADDREL)
                                ; in low 10 bits, and the component code number in the high 5 bits.

                                ; Examples:
                                ;       NNADTAB  = %00042 ; CADR for octal address 42
                                ;       NNADTAB  = %00000 ; noun not used
                                ;       NNADTAB  = %40000 ; specify machine address
                                ;       NNADTAB  = %77777 ; augment last address
```

```
;-----------------------------------------------------------------------------
; NNTYPETAB (normal case):
; a packed table of the form: MMMMM NNNNN PPPPP

; for the normal case:
;       MMMMM (bits 15-11): COMPONENT CODE NUMBER (p.263)
;                00000 = 1 component
;                00001 = 2 component
;                00010 = 3 component
;                X1XXX = bit4=1, decimal only
;                1XXXX = bit5=1, no load
;
;       NNNNN (bits 10-6): SF ROUTINE CODE NUMBER (p.263)
;                00000 = octal only
;                00001 = straight fractional (decimal)
;                00010 = CDU degrees (XXX.XX)
;                00011 = arithmetic SF
;                00100 = arith DP1, OUT(mult by 2EXP14 at end),      IN(straight)
;                00101 = arith DP2, OUT(straight),          IN(SL 7 at end)
;                00110 = Y optics degrees (XX.XXX max at 89.999)
;                00111 = arith DP3, OUT(SL 7 at end)                IN(straight)
;                01000 = whole hours in R1, whole minutes (mod 60) in R2,
;                         seconds (mod 60) 0XX.XX in R3   *** alarms if used with
octal
;
;       PPPPP (bits 5-1): SF CONSTANT CODE NUMBER (p.263)
;                00000 = whole,   use arith

; Examples:
;       NNTYPTAB = %00000 ; 1 comp, octal only
;       NNTYPTAB = %02000 ; 2 comp, octal only
;       NNTYPTAB = %04000 ; 3 comp, octal only
;       NNTYPTAB = %00040 ; 1 comp ,straight fractional
;       NNTYPTAB = %04040 ; 3 comp ,straight fractional

;-----------------------------------------------------------------------------
; NNTYPETAB (mixed case):
; a packed table of the form: MMMMM NNNNN PPPPP

; for the mixed case (3 component):
;       MMMMM (bits 15-11)        = SF constant3 code number.
;       NNNNN (bits 10-6)         = SF constant2 code number.
;       PPPPP (bits 5-1) = SF constant1 code number.

; for the mixed case (2 component):
;       NNNNN (bits 10-6)         = SF constant2 code number.
;       PPPPP (bits 5-1) = SF constant1 code number.

; for the mixed case (1 component):
;       PPPPP (bits 5-1) = SF constant1 code number.

;-----------------------------------------------------------------------------
; IDADDTAB (mixed case only):
; there is also an indirect address table for mixed case only.
; Each entry contains one ECADR. IDADDREL is the relative address of
; the first of these entries.

; There is one entry in this table for each component of a mixed noun.
; They are listed in order of ascending K.

;-----------------------------------------------------------------------------
; RUTMXTAB (mixed case only):
; there is also a scale factor routine number table for mixed case only.
; There is one entry per mixed noun. The form is: QQQQQ RRRRR SSSSS

; for the 3 component case
;       QQQQQ (bits 15-11)        = SF routine3 code number.
;       RRRRR (bits 10-6)         = SF routine2 code number.
;       SSSSS (bits 5-1) = SF routine1 code number.

; for the 2 component case
;       RRRRR (bits 10-6)         = SF routine2 code number.
;       SSSSS (bits 5-1) = SF routine1 code number.


; In octal display and load (oct or dec) verbs, exclude use of verbs whose
; component number is greater than the number of components in noun.
; (All machine address to be specified nouns are 3 component)
; In multi-component load verbs, no mixing of octal and decimal data
; component words is allowed; alarm if violation.

; In decimal loads of data, 5 numerical chars must be keyed in before
; each enter; if not, alarm.


;-----------------------------------------------------------------------------
```

```
                        ; LODNNTAB
                        ; loads NNADTEM with the NNADTAB entry, NNTYPTEM with the NNTYPTAB
                        ; entry. If the noun is mixed, IDAD1TEM is loaded with the first IDADTAB
                        ; entry, IDAD2TEM the second IDADTAB entry, IDAD3TEM the third IDADTAB
                        ; entry, RUTMXTEM with the RUTMXTAB entry. MIXBR is set for mixed=2
                        ; or normal=1 noun.
                        ;
                        ; NOTE: in BlockII, NNADTEM = -1 means use an I/O channel instead of a
                        ; memory address (channel specified in NOUNCADR). Block I does not have
                        ; I/O channels.
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.265.
                        ;-----------------------------------------------------------------------

                        LODNNTAB        EQU     *
16114  7,0114 5  0,0562 0               TS      GTSF_RET        ; save return CADR

16115  7,0115 2  0,0471 1               INDEX   NOUNREG
16116  7,0116 3  7,6210 1               CAF     NNADTAB
16117  7,0117 5  0,0443 1               TS      NNADTEM

16120  7,0120 2  0,0471 1               INDEX   NOUNREG
16121  7,0121 3  7,6354 0               CAF     NNTYPTAB
16122  7,0122 5  0,0444 0               TS      NNTYPTEM

16123  7,0123 4  0,0471 1               CS      NOUNREG
16124  7,0124 6  7,6161 1               AD      MIXCON
16125  7,0125 1  0,0000 0               CCS     A               ; was BZMF LODMIXNN in Block II
16126  7,0126 0  7,6132 1               TC      *+4             ; >0
16127  7,0127 0  7,6131 1               TC      *+2             ; +0, noun number G/E first mixed noun
16130  7,0130 0  7,6131 1               TC      *+1             ; <0, noun number G/E first mixed noun
16131  7,0131 0  7,6135 0               TC      LODMIXNN        ; -0, noun number G/E first mixed noun


16132  7,0132 3  1,2051 1               CAF     ONE             ; noun number L/ first mixed noun
16133  7,0133 5  0,0435 0               TS      MIXBR           ; normal, +1 into MIXBR

16134  7,0134 0  7,6156 0               TC      LODNLV

                        LODMIXNN        EQU     *
16135  7,0135 3  1,2052 1               CAF     TWO             ; mixed, +2 into MIXBR
16136  7,0136 5  0,0435 0               TS      MIXBR

16137  7,0137 2  0,0471 1               INDEX   NOUNREG
16140  7,0140 3  7,7054 1               CAF     RUTMXTAB-40     ; first mixed noun = 40
16141  7,0141 5  0,0450 0               TS      RUTMXTEM

16142  7,0142 3  2,4674 0               CAF     LOW10
16143  7,0143 7  0,0443 0               MASK    NNADTEM
16144  7,0144 5  0,0001 0               TS      Q               ; temp

16145  7,0145 2  0,0000 0               INDEX   A
16146  7,0146 3  7,6640 0               CAF     IDADDTAB
16147  7,0147 5  0,0445 1               TS      IDAD1TEM        ; load IDAD1TEM with first IDADDTAB entry

16150  7,0150 2  0,0001 1               INDEX   Q
16151  7,0151 3  7,6641 1               CAF     IDADDTAB+1
16152  7,0152 5  0,0446 1               TS      IDAD2TEM        ; load IDAD2TEM with 2nd IDADDTAB entry

16153  7,0153 2  0,0001 1               INDEX   Q
16154  7,0154 3  7,6642 1               CAF     IDADDTAB+2
16155  7,0155 5  0,0447 0               TS      IDAD3TEM        ; load IDAD3TEM with 3rd IDADDTAB entry

                        LODNLV          EQU     *
16156  7,0156 3  1,2050 0               CAF     ZERO
16157  7,0157 6  0,0562 0               AD      GTSF_RET        ; load return CADR
16160  7,0160 0  1,3526 0               TC      DXCHJUMP        ; return

16161  7,0161     00050 1 MIXCON         DS      %50             ; 1st mixed noun = 40 (DEC 40)

                        ;-----------------------------------------------------------------------
                        ; GTSFOUT
                        ; On entry, SFTEMP1 contains SFCONUM X 2.
                        ; Loads SFTEMP1, SFTEMP2 with the DP SFOUTAB entries
                        ;
                        ; GTSFIN
                        ; On entry, SFTEMP1 contains SFCONUM X 2.
                        ; Loads SFTEMP1, SFTEMP2 with the DP SFINTAB entries
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.266.
                        ;-----------------------------------------------------------------------

                        GTSFOUT         EQU     *
16162  7,0162 5  0,0562 0               TS      GTSF_RET        ; save return CADR
```

```
16163  7,0163 3  0,0420 1               XCH      SFTEMP1
16164  7,0164 5  0,0001 0               TS       Q              ; temp

16165  7,0165 2  0,0001 1               INDEX    Q
16166  7,0166 3  7,6570 0               CAF      SFOUTAB
16167  7,0167 5  0,0420 1               TS       SFTEMP1

16170  7,0170 2  0,0001 1               INDEX    Q
16171  7,0171 3  7,6571 1               CAF      SFOUTAB+1
16172  7,0172 5  0,0421 0               TS       SFTEMP2

                         SFCOM          EQU      *
16173  7,0173 3  1,2050 0               CAF      ZERO
16174  7,0174 6  0,0562 0               AD       GTSF_RET       ; load return CADR
16175  7,0175 0  1,3526 0               TC       DXCHJUMP       ; return

                         GTSFIN         EQU      *
16176  7,0176 5  0,0562 0               TS       GTSF_RET       ; save return CADR

16177  7,0177 3  0,0420 1               XCH      SFTEMP1
16200  7,0200 5  0,0001 0               TS       Q              ; temp

16201  7,0201 2  0,0001 1               INDEX    Q
16202  7,0202 3  7,6520 0               CAF      SFINTAB
16203  7,0203 5  0,0420 1               TS       SFTEMP1

16204  7,0204 2  0,0001 1               INDEX    Q
16205  7,0205 3  7,6521 1               CAF      SFINTAB+1
16206  7,0206 5  0,0421 0               TS       SFTEMP2

16207  7,0207 0  7,6173 1               TC       SFCOM

                         ;-----------------------------------------------------------------------
                         ; NOUN ADDRESS TABLE (NNADTAB)
                         ; Indexed by noun number (0-39 decimal for normal nouns).
                         ;
                         ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                         ; Oct 28, 1968, p.266.
                         ;-----------------------------------------------------------------------

                         NNADTAB        EQU      *              ; NN - NORMAL NOUNS
16210  7,0210     00000 1               DS       %0             ; 00 - not in use
16211  7,0211     40000 0               DS       %40000         ; 01 - specify machine address (fractional)
16212  7,0212     40000 0               DS       %40000         ; 02 - specify machine address (whole)
16213  7,0213     40000 0               DS       %40000         ; 03 - specify machine address (degrees)
16214  7,0214     00036 1               DS       %00036         ; 04 - spare ********** TEST, CHANGE TO ZERO
16215  7,0215     00000 1               DS       %0             ; 05 - spare
16216  7,0216     00000 1               DS       %0             ; 06 - spare
16217  7,0217     00000 1               DS       %0             ; 07 - spare
16220  7,0220     00000 1               DS       %0             ; 08 - spare
16221  7,0221     00460 0               ECADR    FAILREG        ; 09 - alarm codes
16222  7,0222     00000 1               DS       %0             ; 10 - spare
16223  7,0223     00000 1               DS       %0             ; 11 - spare
16224  7,0224     00000 1               DS       %0             ; 12 - spare
16225  7,0225     00000 1               DS       %0             ; 13 - spare
16226  7,0226     00000 1               DS       %0             ; 14 - spare
16227  7,0227     77777 0               DS       %77777         ; 15 - increment machine address
16230  7,0230     00000 1               DS       %0             ; 16 - spare
16231  7,0231     00000 1               DS       %0             ; 17 - spare
16232  7,0232     00000 1               DS       %0             ; 18 - spare
16233  7,0233     00000 1               DS       %0             ; 19 - spare
16234  7,0234     00000 1               DS       %0             ; 20 - spare
16235  7,0235     00000 1               DS       %0             ; 21 - spare
16236  7,0236     00000 1               DS       %0             ; 22 - spare
16237  7,0237     00000 1               DS       %0             ; 23 - spare
16240  7,0240     00000 1               DS       %0             ; 24 - spare
16241  7,0241     00000 1               DS       %0             ; 25 - spare
16242  7,0242     00534 0               ECADR    DSPTEM1        ; 26 - prio/delay, adres, BBCON
16243  7,0243     00000 1               DS       %0             ; 27 - spare
16244  7,0244     00000 1               DS       %0             ; 28 - spare
16245  7,0245     00000 1               DS       %0             ; 29 - spare
16246  7,0246     00000 1               DS       %0             ; 30 - spare
16247  7,0247     00000 1               DS       %0             ; 31 - spare
16250  7,0250     00000 1               DS       %0             ; 32 - spare
16251  7,0251     00000 1               DS       %0             ; 33 - spare
16252  7,0252     00000 1               DS       %0             ; 34 - spare
16253  7,0253     00000 1               DS       %0             ; 35 - spare
16254  7,0254     00035 1               ECADR    TIME2          ; 36 - time of AGC clock (hrs, min, sec)
16255  7,0255     00000 1               DS       %0             ; 37 - spare
16256  7,0256     00000 1               DS       %0             ; 38 - spare
16257  7,0257     00000 1               DS       %0             ; 39 - spare
                         ; end of normal nouns

                         ; start of mixed nouns

16260  7,0260     00000 1               DS       %0             ; 40 - spare
16261  7,0261     00000 1               DS       %0             ; 41 - spare
```

```
16262  7,0262     00000 1               DS       %0             ; 42 - spare
16263  7,0263     00000 1               DS       %0             ; 43 - spare
16264  7,0264     00000 1               DS       %0             ; 44 - spare
16265  7,0265     00000 1               DS       %0             ; 45 - spare
16266  7,0266     00000 1               DS       %0             ; 46 - spare
16267  7,0267     00000 1               DS       %0             ; 47 - spare
16270  7,0270     00000 1               DS       %0             ; 48 - spare
16271  7,0271     00000 1               DS       %0             ; 49 - spare
16272  7,0272     00000 1               DS       %0             ; 50 - spare
16273  7,0273     00000 1               DS       %0             ; 51 - spare
16274  7,0274     00000 1               DS       %0             ; 52 - spare
16275  7,0275     00000 1               DS       %0             ; 53 - spare
16276  7,0276     00000 1               DS       %0             ; 54 - spare
16277  7,0277     00000 1               DS       %0             ; 55 - spare
16300  7,0300     00000 1               DS       %0             ; 56 - spare
16301  7,0301     00000 1               DS       %0             ; 57 - spare
16302  7,0302     00000 1               DS       %0             ; 58 - spare
16303  7,0303     00000 1               DS       %0             ; 59 - spare
16304  7,0304     00000 1               DS       %0             ; 60 - spare
16305  7,0305     00000 1               DS       %0             ; 61 - spare
16306  7,0306     00000 1               DS       %0             ; 62 - spare
16307  7,0307     00000 1               DS       %0             ; 63 - spare
16310  7,0310     00000 1               DS       %0             ; 64 - spare
16311  7,0311     00000 1               DS       %0             ; 65 - spare
16312  7,0312     00000 1               DS       %0             ; 66 - spare
16313  7,0313     00000 1               DS       %0             ; 67 - spare
16314  7,0314     00000 1               DS       %0             ; 68 - spare
16315  7,0315     00000 1               DS       %0             ; 69 - spare
16316  7,0316     00000 1               DS       %0             ; 70 - spare
16317  7,0317     00000 1               DS       %0             ; 71 - spare
16320  7,0320     00000 1               DS       %0             ; 72 - spare
16321  7,0321     00000 1               DS       %0             ; 73 - spare
16322  7,0322     00000 1               DS       %0             ; 74 - spare
16323  7,0323     00000 1               DS       %0             ; 75 - spare
16324  7,0324     00000 1               DS       %0             ; 76 - spare
16325  7,0325     00000 1               DS       %0             ; 77 - spare
16326  7,0326     00000 1               DS       %0             ; 78 - spare
16327  7,0327     00000 1               DS       %0             ; 79 - spare
16330  7,0330     00000 1               DS       %0             ; 80 - spare
16331  7,0331     00000 1               DS       %0             ; 81 - spare
16332  7,0332     00000 1               DS       %0             ; 82 - spare
16333  7,0333     00000 1               DS       %0             ; 83 - spare
16334  7,0334     00000 1               DS       %0             ; 84 - spare
16335  7,0335     00000 1               DS       %0             ; 85 - spare
16336  7,0336     00000 1               DS       %0             ; 86 - spare
16337  7,0337     00000 1               DS       %0             ; 87 - spare
16340  7,0340     00000 1               DS       %0             ; 88 - spare
16341  7,0341     00000 1               DS       %0             ; 89 - spare
16342  7,0342     00000 1               DS       %0             ; 90 - spare
16343  7,0343     00000 1               DS       %0             ; 91 - spare
16344  7,0344     00000 1               DS       %0             ; 92 - spare
16345  7,0345     00000 1               DS       %0             ; 93 - spare
16346  7,0346     00000 1               DS       %0             ; 94 - spare
16347  7,0347     00000 1               DS       %0             ; 95 - spare
16350  7,0350     00000 1               DS       %0             ; 96 - spare
16351  7,0351     00000 1               DS       %0             ; 97 - spare
16352  7,0352     00000 1               DS       %0             ; 98 - spare
16353  7,0353     00000 1               DS       %0             ; 99 - spare
                        ; end of mixed nouns

                        ;-----------------------------------------------------------------------
                        ; NOUN TYPE TABLE (NNTYPTAB)
                        ; Indexed by noun number (0-39 decimal for normal nouns).
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.269.
                        ;-----------------------------------------------------------------------

                        NNTYPTAB       EQU      *              ; NN - NORMAL NOUNS
16354  7,0354     00000 1               DS       %0             ; 00 - not in use
16355  7,0355     04040 1               DS       %04040         ; 01 - 3 component (fractional)
16356  7,0356     04140 0               DS       %04140         ; 02 - 3 component (whole)
16357  7,0357     04102 0               DS       %04102         ; 03 - 3 component (CDU degrees)
16360  7,0360     00000 1               DS       %0             ; 04 - spare
16361  7,0361     00000 1               DS       %0             ; 05 - spare
16362  7,0362     00000 1               DS       %0             ; 06 - spare
16363  7,0363     00000 1               DS       %0             ; 07 - spare
16364  7,0364     00000 1               DS       %0             ; 08 - spare
16365  7,0365     04000 0               DS       %04000         ; 09 - 3 component, octal only
16366  7,0366     00000 1               DS       %0             ; 10 - spare
16367  7,0367     00000 1               DS       %0             ; 11 - spare
16370  7,0370     00000 1               DS       %0             ; 12 - spare
16371  7,0371     00000 1               DS       %0             ; 13 - spare
16372  7,0372     00000 1               DS       %0             ; 14 - spare
16373  7,0373     00000 1               DS       %0             ; 15 - 1 component, octal only
16374  7,0374     00000 1               DS       %0             ; 16 - spare
16375  7,0375     00000 1               DS       %0             ; 17 - spare
```

```
16376  7,0376     00000 1               DS       %0            ; 18 - spare
16377  7,0377     00000 1               DS       %0            ; 19 - spare
16400  7,0400     00000 1               DS       %0            ; 20 - spare
16401  7,0401     00000 1               DS       %0            ; 21 - spare
16402  7,0402     00000 1               DS       %0            ; 22 - spare
16403  7,0403     00000 1               DS       %0            ; 23 - spare
16404  7,0404     00000 1               DS       %0            ; 24 - spare
16405  7,0405     00000 1               DS       %0            ; 25 - spare
16406  7,0406     04000 0               DS       %04000        ; 26 - 3 component, octal only
16407  7,0407     00000 1               DS       %0            ; 27 - spare
16410  7,0410     00000 1               DS       %0            ; 28 - spare
16411  7,0411     00000 1               DS       %0            ; 29 - spare
16412  7,0412     00000 1               DS       %0            ; 30 - spare
16413  7,0413     00000 1               DS       %0            ; 31 - spare
16414  7,0414     00000 1               DS       %0            ; 32 - spare
16415  7,0415     00000 1               DS       %0            ; 33 - spare
16416  7,0416     00000 1               DS       %0            ; 34 - spare
16417  7,0417     00000 1               DS       %0            ; 35 - spare
16420  7,0420     24400 0               DS       %24400        ; 36 - 3 component, HMS, (dec only)
16421  7,0421     00000 1               DS       %0            ; 37 - spare
16422  7,0422     00000 1               DS       %0            ; 38 - spare
16423  7,0423     00000 1               DS       %0            ; 39 - spare
                        ; end of normal nouns

                        ; start of mixed nouns

16424  7,0424     00000 1               DS       %0            ; 40 - spare
16425  7,0425     00000 1               DS       %0            ; 41 - spare
16426  7,0426     00000 1               DS       %0            ; 42 - spare
16427  7,0427     00000 1               DS       %0            ; 43 - spare
16430  7,0430     00000 1               DS       %0            ; 44 - spare
16431  7,0431     00000 1               DS       %0            ; 45 - spare
16432  7,0432     00000 1               DS       %0            ; 46 - spare
16433  7,0433     00000 1               DS       %0            ; 47 - spare
16434  7,0434     00000 1               DS       %0            ; 48 - spare
16435  7,0435     00000 1               DS       %0            ; 49 - spare
16436  7,0436     00000 1               DS       %0            ; 50 - spare
16437  7,0437     00000 1               DS       %0            ; 51 - spare
16440  7,0440     00000 1               DS       %0            ; 52 - spare
16441  7,0441     00000 1               DS       %0            ; 53 - spare
16442  7,0442     00000 1               DS       %0            ; 54 - spare
16443  7,0443     00000 1               DS       %0            ; 55 - spare
16444  7,0444     00000 1               DS       %0            ; 56 - spare
16445  7,0445     00000 1               DS       %0            ; 57 - spare
16446  7,0446     00000 1               DS       %0            ; 58 - spare
16447  7,0447     00000 1               DS       %0            ; 59 - spare
16450  7,0450     00000 1               DS       %0            ; 60 - spare
16451  7,0451     00000 1               DS       %0            ; 61 - spare
16452  7,0452     00000 1               DS       %0            ; 62 - spare
16453  7,0453     00000 1               DS       %0            ; 63 - spare
16454  7,0454     00000 1               DS       %0            ; 64 - spare
16455  7,0455     00000 1               DS       %0            ; 65 - spare
16456  7,0456     00000 1               DS       %0            ; 66 - spare
16457  7,0457     00000 1               DS       %0            ; 67 - spare
16460  7,0460     00000 1               DS       %0            ; 68 - spare
16461  7,0461     00000 1               DS       %0            ; 69 - spare
16462  7,0462     00000 1               DS       %0            ; 70 - spare
16463  7,0463     00000 1               DS       %0            ; 71 - spare
16464  7,0464     00000 1               DS       %0            ; 72 - spare
16465  7,0465     00000 1               DS       %0            ; 73 - spare
16466  7,0466     00000 1               DS       %0            ; 74 - spare
16467  7,0467     00000 1               DS       %0            ; 75 - spare
16470  7,0470     00000 1               DS       %0            ; 76 - spare
16471  7,0471     00000 1               DS       %0            ; 77 - spare
16472  7,0472     00000 1               DS       %0            ; 78 - spare
16473  7,0473     00000 1               DS       %0            ; 79 - spare
16474  7,0474     00000 1               DS       %0            ; 80 - spare
16475  7,0475     00000 1               DS       %0            ; 81 - spare
16476  7,0476     00000 1               DS       %0            ; 82 - spare
16477  7,0477     00000 1               DS       %0            ; 83 - spare
16500  7,0500     00000 1               DS       %0            ; 84 - spare
16501  7,0501     00000 1               DS       %0            ; 85 - spare
16502  7,0502     00000 1               DS       %0            ; 86 - spare
16503  7,0503     00000 1               DS       %0            ; 87 - spare
16504  7,0504     00000 1               DS       %0            ; 88 - spare
16505  7,0505     00000 1               DS       %0            ; 89 - spare
16506  7,0506     00000 1               DS       %0            ; 90 - spare
16507  7,0507     00000 1               DS       %0            ; 91 - spare
16510  7,0510     00000 1               DS       %0            ; 92 - spare
16511  7,0511     00000 1               DS       %0            ; 93 - spare
16512  7,0512     00000 1               DS       %0            ; 94 - spare
16513  7,0513     00000 1               DS       %0            ; 95 - spare
16514  7,0514     00000 1               DS       %0            ; 96 - spare
16515  7,0515     00000 1               DS       %0            ; 97 - spare
16516  7,0516     00000 1               DS       %0            ; 98 - spare
16517  7,0517     00000 1               DS       %0            ; 99 - spare
                        ; end of mixed nouns
```

```
                        ;----------------------------------------------------------------------
                        ; SCALE FACTOR INPUT TABLE (SFINTAB)
                        ; Indexed by SF constant code number x 2 PPPPP (0-19 decimal; 0-23 octal)
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.272.
                        ;----------------------------------------------------------------------

                        SFINTAB         EQU     *
16520  7,0520     00006 1               DS      %00006          ; 00 - whole, DP time (sec)
16521  7,0521     03240 1               DS      %03240          ; 00

16522  7,0522     00000 1               DS      %00000          ; 01 - spare
16523  7,0523     00000 1               DS      %00000          ; 01

16524  7,0524     00000 1               DS      %00000          ; 02 - CDU degrees, Y optics degrees
16525  7,0525     00000 1               DS      %00000          ; 02   (SFCONs in DEGINSF, OPTDEGIN

16526  7,0526     00000 1               DS      %00000          ; 03
16527  7,0527     00000 1               DS      %00000          ; 03

16530  7,0530     00000 1               DS      %00000          ; 04
16531  7,0531     00000 1               DS      %00000          ; 04

16532  7,0532     00000 1               DS      %00000          ; 05
16533  7,0533     00000 1               DS      %00000          ; 05

16534  7,0534     00000 1               DS      %00000          ; 06
16535  7,0535     00000 1               DS      %00000          ; 06

16536  7,0536     00000 1               DS      %00000          ; 07
16537  7,0537     00000 1               DS      %00000          ; 07

16540  7,0540     00000 1               DS      %00000          ; 10
16541  7,0541     00000 1               DS      %00000          ; 10

16542  7,0542     00000 1               DS      %00000          ; 11
16543  7,0543     00000 1               DS      %00000          ; 11

16544  7,0544     00000 1               DS      %00000          ; 12
16545  7,0545     00000 1               DS      %00000          ; 12

16546  7,0546     00000 1               DS      %00000          ; 13
16547  7,0547     00000 1               DS      %00000          ; 13

16550  7,0550     00000 1               DS      %00000          ; 14
16551  7,0551     00000 1               DS      %00000          ; 14

16552  7,0552     00000 1               DS      %00000          ; 15
16553  7,0553     00000 1               DS      %00000          ; 15

16554  7,0554     00000 1               DS      %00000          ; 16
16555  7,0555     00000 1               DS      %00000          ; 16

16556  7,0556     00000 1               DS      %00000          ; 17
16557  7,0557     00000 1               DS      %00000          ; 17

16560  7,0560     00000 1               DS      %00000          ; 20
16561  7,0561     00000 1               DS      %00000          ; 20

16562  7,0562     00000 1               DS      %00000          ; 21
16563  7,0563     00000 1               DS      %00000          ; 21

16564  7,0564     00000 1               DS      %00000          ; 22
16565  7,0565     00000 1               DS      %00000          ; 22

16566  7,0566     00000 1               DS      %00000          ; 23
16567  7,0567     00000 1               DS      %00000          ; 23

                        ;----------------------------------------------------------------------
                        ; SCALE FACTOR OUTPUT TABLE (SFOUTAB)
                        ; Indexed by SF constant code number x 2 PPPPP (0-19 decimal; 0-23 octal)
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.273.
                        ;----------------------------------------------------------------------

                        SFOUTAB         EQU     *
16570  7,0570     05174 0               DS      %05174          ; 00 - whole, DP time (sec)
16571  7,0571     13261 0               DS      %13261          ; 00

16572  7,0572     00000 1               DS      %00000          ; 01 - spare
16573  7,0573     00000 1               DS      %00000          ; 01

16574  7,0574     00000 1               DS      %00000          ; 02 - CDU degrees, Y optics degrees
16575  7,0575     00000 1               DS      %00000          ; 02   (SFCONs in DEGOURSF, OPTDEGOUT
```

```
16576  7,0576     00000 1               DS       %00000         ; 03
16577  7,0577     00000 1               DS       %00000         ; 03

16600  7,0600     00000 1               DS       %00000         ; 04
16601  7,0601     00000 1               DS       %00000         ; 04

16602  7,0602     00000 1               DS       %00000         ; 05
16603  7,0603     00000 1               DS       %00000         ; 05

16604  7,0604     00000 1               DS       %00000         ; 06
16605  7,0605     00000 1               DS       %00000         ; 06

16606  7,0606     00000 1               DS       %00000         ; 07
16607  7,0607     00000 1               DS       %00000         ; 07

16610  7,0610     00000 1               DS       %00000         ; 10
16611  7,0611     00000 1               DS       %00000         ; 10

16612  7,0612     00000 1               DS       %00000         ; 11
16613  7,0613     00000 1               DS       %00000         ; 11

16614  7,0614     00000 1               DS       %00000         ; 12
16615  7,0615     00000 1               DS       %00000         ; 12

16616  7,0616     00000 1               DS       %00000         ; 13
16617  7,0617     00000 1               DS       %00000         ; 13

16620  7,0620     00000 1               DS       %00000         ; 14
16621  7,0621     00000 1               DS       %00000         ; 14

16622  7,0622     00000 1               DS       %00000         ; 15
16623  7,0623     00000 1               DS       %00000         ; 15

16624  7,0624     00000 1               DS       %00000         ; 16
16625  7,0625     00000 1               DS       %00000         ; 16

16626  7,0626     00000 1               DS       %00000         ; 17
16627  7,0627     00000 1               DS       %00000         ; 17

16630  7,0630     00000 1               DS       %00000         ; 20
16631  7,0631     00000 1               DS       %00000         ; 20

16632  7,0632     00000 1               DS       %00000         ; 21
16633  7,0633     00000 1               DS       %00000         ; 21

16634  7,0634     00000 1               DS       %00000         ; 22
16635  7,0635     00000 1               DS       %00000         ; 22

16636  7,0636     00000 1               DS       %00000         ; 23
16637  7,0637     00000 1               DS       %00000         ; 23


                          ; SCALE FACTOR INPUT ROUTINE TABLE is on pp. 342, 343 of COLOSSUS
                          ; SCALE FACTOR OUTPUT ROUTINE TABLE is on p. 329 of COLOSSUS

                          ;-----------------------------------------------------------------------
                          ; MIXED NOUN ADDRESS TABLE (IDADDTAB)
                          ;
                          ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                          ; Oct 28, 1968, p.274.
                          ;-----------------------------------------------------------------------

                          ; ** currently, the table is not populated **

                          IDADDTAB       EQU      *
16640  7,0640     00000 1               DS       %0             ; 40 - spare component
16641  7,0641     00000 1               DS       %0             ; 40 - spare component
16642  7,0642     00000 1               DS       %0             ; 40 - spare component

16643  7,0643     00000 1               DS       %0             ; 41 - spare component
16644  7,0644     00000 1               DS       %0             ; 41 - spare component
16645  7,0645     00000 1               DS       %0             ; 41 - spare component

16646  7,0646     00000 1               DS       %0             ; 42 - spare component
16647  7,0647     00000 1               DS       %0             ; 42 - spare component
16650  7,0650     00000 1               DS       %0             ; 42 - spare component

16651  7,0651     00000 1               DS       %0             ; 43 - spare component
16652  7,0652     00000 1               DS       %0             ; 43 - spare component
16653  7,0653     00000 1               DS       %0             ; 43 - spare component

16654  7,0654     00000 1               DS       %0             ; 44 - spare component
16655  7,0655     00000 1               DS       %0             ; 44 - spare component
16656  7,0656     00000 1               DS       %0             ; 44 - spare component

16657  7,0657     00000 1               DS       %0             ; 45 - spare component
```

```
16660  7,0660     00000 1              DS       %0            ; 45 - spare component
16661  7,0661     00000 1              DS       %0            ; 45 - spare component

16662  7,0662     00000 1              DS       %0            ; 46 - spare component
16663  7,0663     00000 1              DS       %0            ; 46 - spare component
16664  7,0664     00000 1              DS       %0            ; 46 - spare component

16665  7,0665     00000 1              DS       %0            ; 47 - spare component
16666  7,0666     00000 1              DS       %0            ; 47 - spare component
16667  7,0667     00000 1              DS       %0            ; 47 - spare component

16670  7,0670     00000 1              DS       %0            ; 48 - spare component
16671  7,0671     00000 1              DS       %0            ; 48 - spare component
16672  7,0672     00000 1              DS       %0            ; 48 - spare component

16673  7,0673     00000 1              DS       %0            ; 49 - spare component
16674  7,0674     00000 1              DS       %0            ; 49 - spare component
16675  7,0675     00000 1              DS       %0            ; 49 - spare component

16676  7,0676     00000 1              DS       %0            ; 50 - spare component
16677  7,0677     00000 1              DS       %0            ; 50 - spare component
16700  7,0700     00000 1              DS       %0            ; 50 - spare component

16701  7,0701     00000 1              DS       %0            ; 51 - spare component
16702  7,0702     00000 1              DS       %0            ; 51 - spare component
16703  7,0703     00000 1              DS       %0            ; 51 - spare component

16704  7,0704     00000 1              DS       %0            ; 52 - spare component
16705  7,0705     00000 1              DS       %0            ; 52 - spare component
16706  7,0706     00000 1              DS       %0            ; 52 - spare component

16707  7,0707     00000 1              DS       %0            ; 53 - spare component
16710  7,0710     00000 1              DS       %0            ; 53 - spare component
16711  7,0711     00000 1              DS       %0            ; 53 - spare component

16712  7,0712     00000 1              DS       %0            ; 54 - spare component
16713  7,0713     00000 1              DS       %0            ; 54 - spare component
16714  7,0714     00000 1              DS       %0            ; 54 - spare component

16715  7,0715     00000 1              DS       %0            ; 55 - spare component
16716  7,0716     00000 1              DS       %0            ; 55 - spare component
16717  7,0717     00000 1              DS       %0            ; 55 - spare component

16720  7,0720     00000 1              DS       %0            ; 56 - spare component
16721  7,0721     00000 1              DS       %0            ; 56 - spare component
16722  7,0722     00000 1              DS       %0            ; 56 - spare component

16723  7,0723     00000 1              DS       %0            ; 57 - spare component
16724  7,0724     00000 1              DS       %0            ; 57 - spare component
16725  7,0725     00000 1              DS       %0            ; 57 - spare component

16726  7,0726     00000 1              DS       %0            ; 58 - spare component
16727  7,0727     00000 1              DS       %0            ; 58 - spare component
16730  7,0730     00000 1              DS       %0            ; 58 - spare component

16731  7,0731     00000 1              DS       %0            ; 59 - spare component
16732  7,0732     00000 1              DS       %0            ; 59 - spare component
16733  7,0733     00000 1              DS       %0            ; 59 - spare component

16734  7,0734     00000 1              DS       %0            ; 60 - spare component
16735  7,0735     00000 1              DS       %0            ; 60 - spare component
16736  7,0736     00000 1              DS       %0            ; 60 - spare component

16737  7,0737     00000 1              DS       %0            ; 61 - spare component
16740  7,0740     00000 1              DS       %0            ; 61 - spare component
16741  7,0741     00000 1              DS       %0            ; 61 - spare component

16742  7,0742     00000 1              DS       %0            ; 62 - spare component
16743  7,0743     00000 1              DS       %0            ; 62 - spare component
16744  7,0744     00000 1              DS       %0            ; 62 - spare component

16745  7,0745     00000 1              DS       %0            ; 63 - spare component
16746  7,0746     00000 1              DS       %0            ; 63 - spare component
16747  7,0747     00000 1              DS       %0            ; 63 - spare component

16750  7,0750     00000 1              DS       %0            ; 64 - spare component
16751  7,0751     00000 1              DS       %0            ; 64 - spare component
16752  7,0752     00000 1              DS       %0            ; 64 - spare component

16753  7,0753     00000 1              DS       %0            ; 65 - spare component
16754  7,0754     00000 1              DS       %0            ; 65 - spare component
16755  7,0755     00000 1              DS       %0            ; 65 - spare component

16756  7,0756     00000 1              DS       %0            ; 66 - spare component
16757  7,0757     00000 1              DS       %0            ; 66 - spare component
16760  7,0760     00000 1              DS       %0            ; 66 - spare component
```

```
16761  7,0761     00000 1              DS       %0            ; 67 - spare component
16762  7,0762     00000 1              DS       %0            ; 67 - spare component
16763  7,0763     00000 1              DS       %0            ; 67 - spare component

16764  7,0764     00000 1              DS       %0            ; 68 - spare component
16765  7,0765     00000 1              DS       %0            ; 68 - spare component
16766  7,0766     00000 1              DS       %0            ; 68 - spare component

16767  7,0767     00000 1              DS       %0            ; 69 - spare component
16770  7,0770     00000 1              DS       %0            ; 69 - spare component
16771  7,0771     00000 1              DS       %0            ; 69 - spare component

16772  7,0772     00000 1              DS       %0            ; 70 - spare component
16773  7,0773     00000 1              DS       %0            ; 70 - spare component
16774  7,0774     00000 1              DS       %0            ; 70 - spare component

16775  7,0775     00000 1              DS       %0            ; 71 - spare component
16776  7,0776     00000 1              DS       %0            ; 71 - spare component
16777  7,0777     00000 1              DS       %0            ; 71 - spare component

17000  7,1000     00000 1              DS       %0            ; 72 - spare component
17001  7,1001     00000 1              DS       %0            ; 72 - spare component
17002  7,1002     00000 1              DS       %0            ; 72 - spare component

17003  7,1003     00000 1              DS       %0            ; 73 - spare component
17004  7,1004     00000 1              DS       %0            ; 73 - spare component
17005  7,1005     00000 1              DS       %0            ; 73 - spare component

17006  7,1006     00000 1              DS       %0            ; 74 - spare component
17007  7,1007     00000 1              DS       %0            ; 74 - spare component
17010  7,1010     00000 1              DS       %0            ; 74 - spare component

17011  7,1011     00000 1              DS       %0            ; 75 - spare component
17012  7,1012     00000 1              DS       %0            ; 75 - spare component
17013  7,1013     00000 1              DS       %0            ; 75 - spare component

17014  7,1014     00000 1              DS       %0            ; 76 - spare component
17015  7,1015     00000 1              DS       %0            ; 76 - spare component
17016  7,1016     00000 1              DS       %0            ; 76 - spare component

17017  7,1017     00000 1              DS       %0            ; 77 - spare component
17020  7,1020     00000 1              DS       %0            ; 77 - spare component
17021  7,1021     00000 1              DS       %0            ; 77 - spare component

17022  7,1022     00000 1              DS       %0            ; 78 - spare component
17023  7,1023     00000 1              DS       %0            ; 78 - spare component
17024  7,1024     00000 1              DS       %0            ; 78 - spare component

17025  7,1025     00000 1              DS       %0            ; 79 - spare component
17026  7,1026     00000 1              DS       %0            ; 79 - spare component
17027  7,1027     00000 1              DS       %0            ; 79 - spare component

17030  7,1030     00000 1              DS       %0            ; 80 - spare component
17031  7,1031     00000 1              DS       %0            ; 80 - spare component
17032  7,1032     00000 1              DS       %0            ; 80 - spare component

17033  7,1033     00000 1              DS       %0            ; 81 - spare component
17034  7,1034     00000 1              DS       %0            ; 81 - spare component
17035  7,1035     00000 1              DS       %0            ; 81 - spare component

17036  7,1036     00000 1              DS       %0            ; 82 - spare component
17037  7,1037     00000 1              DS       %0            ; 82 - spare component
17040  7,1040     00000 1              DS       %0            ; 82 - spare component

17041  7,1041     00000 1              DS       %0            ; 83 - spare component
17042  7,1042     00000 1              DS       %0            ; 83 - spare component
17043  7,1043     00000 1              DS       %0            ; 83 - spare component

17044  7,1044     00000 1              DS       %0            ; 84 - spare component
17045  7,1045     00000 1              DS       %0            ; 84 - spare component
17046  7,1046     00000 1              DS       %0            ; 84 - spare component

17047  7,1047     00000 1              DS       %0            ; 85 - spare component
17050  7,1050     00000 1              DS       %0            ; 85 - spare component
17051  7,1051     00000 1              DS       %0            ; 85 - spare component

17052  7,1052     00000 1              DS       %0            ; 86 - spare component
17053  7,1053     00000 1              DS       %0            ; 86 - spare component
17054  7,1054     00000 1              DS       %0            ; 86 - spare component

17055  7,1055     00000 1              DS       %0            ; 87 - spare component
17056  7,1056     00000 1              DS       %0            ; 87 - spare component
17057  7,1057     00000 1              DS       %0            ; 87 - spare component

17060  7,1060     00000 1              DS       %0            ; 88 - spare component
17061  7,1061     00000 1              DS       %0            ; 88 - spare component
17062  7,1062     00000 1              DS       %0            ; 88 - spare component
```

```
17063  7,1063     00000 1               DS       %0             ; 89 - spare component
17064  7,1064     00000 1               DS       %0             ; 89 - spare component
17065  7,1065     00000 1               DS       %0             ; 89 - spare component

17066  7,1066     00000 1               DS       %0             ; 90 - spare component
17067  7,1067     00000 1               DS       %0             ; 90 - spare component
17070  7,1070     00000 1               DS       %0             ; 90 - spare component

17071  7,1071     00000 1               DS       %0             ; 91 - spare component
17072  7,1072     00000 1               DS       %0             ; 91 - spare component
17073  7,1073     00000 1               DS       %0             ; 91 - spare component

17074  7,1074     00000 1               DS       %0             ; 92 - spare component
17075  7,1075     00000 1               DS       %0             ; 92 - spare component
17076  7,1076     00000 1               DS       %0             ; 92 - spare component

17077  7,1077     00000 1               DS       %0             ; 93 - spare component
17100  7,1100     00000 1               DS       %0             ; 93 - spare component
17101  7,1101     00000 1               DS       %0             ; 93 - spare component

17102  7,1102     00000 1               DS       %0             ; 94 - spare component
17103  7,1103     00000 1               DS       %0             ; 94 - spare component
17104  7,1104     00000 1               DS       %0             ; 94 - spare component

17105  7,1105     00000 1               DS       %0             ; 95 - spare component
17106  7,1106     00000 1               DS       %0             ; 95 - spare component
17107  7,1107     00000 1               DS       %0             ; 95 - spare component

17110  7,1110     00000 1               DS       %0             ; 96 - spare component
17111  7,1111     00000 1               DS       %0             ; 96 - spare component
17112  7,1112     00000 1               DS       %0             ; 96 - spare component

17113  7,1113     00000 1               DS       %0             ; 97 - spare component
17114  7,1114     00000 1               DS       %0             ; 97 - spare component
17115  7,1115     00000 1               DS       %0             ; 97 - spare component

17116  7,1116     00000 1               DS       %0             ; 98 - spare component
17117  7,1117     00000 1               DS       %0             ; 98 - spare component
17120  7,1120     00000 1               DS       %0             ; 98 - spare component

17121  7,1121     00000 1               DS       %0             ; 99 - spare component
17122  7,1122     00000 1               DS       %0             ; 99 - spare component
17123  7,1123     00000 1               DS       %0             ; 99 - spare component
                        ; end of mixed noun address table

                        ;-----------------------------------------------------------------------------
                        ; MIXED NOUN SCALE FACTOR ROUTINE TABLE (RUTMXTAB)
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.278.
                        ;-----------------------------------------------------------------------------

                        ; ** currently, the table is not populated **

                        RUTMXTAB       EQU      *
17124  7,1124     00000 1               DS       %0             ; 40 - spare
17125  7,1125     00000 1               DS       %0             ; 41 - spare
17126  7,1126     00000 1               DS       %0             ; 42 - spare
17127  7,1127     00000 1               DS       %0             ; 43 - spare
17130  7,1130     00000 1               DS       %0             ; 44 - spare
17131  7,1131     00000 1               DS       %0             ; 45 - spare
17132  7,1132     00000 1               DS       %0             ; 46 - spare
17133  7,1133     00000 1               DS       %0             ; 47 - spare
17134  7,1134     00000 1               DS       %0             ; 48 - spare
17135  7,1135     00000 1               DS       %0             ; 49 - spare
17136  7,1136     00000 1               DS       %0             ; 50 - spare
17137  7,1137     00000 1               DS       %0             ; 51 - spare
17140  7,1140     00000 1               DS       %0             ; 52 - spare
17141  7,1141     00000 1               DS       %0             ; 53 - spare
17142  7,1142     00000 1               DS       %0             ; 54 - spare
17143  7,1143     00000 1               DS       %0             ; 55 - spare
17144  7,1144     00000 1               DS       %0             ; 56 - spare
17145  7,1145     00000 1               DS       %0             ; 57 - spare
17146  7,1146     00000 1               DS       %0             ; 58 - spare
17147  7,1147     00000 1               DS       %0             ; 59 - spare
17150  7,1150     00000 1               DS       %0             ; 60 - spare
17151  7,1151     00000 1               DS       %0             ; 61 - spare
17152  7,1152     00000 1               DS       %0             ; 62 - spare
17153  7,1153     00000 1               DS       %0             ; 63 - spare
17154  7,1154     00000 1               DS       %0             ; 64 - spare
17155  7,1155     00000 1               DS       %0             ; 65 - spare
17156  7,1156     00000 1               DS       %0             ; 66 - spare
17157  7,1157     00000 1               DS       %0             ; 67 - spare
17160  7,1160     00000 1               DS       %0             ; 68 - spare
17161  7,1161     00000 1               DS       %0             ; 69 - spare
17162  7,1162     00000 1               DS       %0             ; 70 - spare
```

```
17163  7,1163     00000 1               DS       %0            ; 71 - spare
17164  7,1164     00000 1               DS       %0            ; 72 - spare
17165  7,1165     00000 1               DS       %0            ; 73 - spare
17166  7,1166     00000 1               DS       %0            ; 74 - spare
17167  7,1167     00000 1               DS       %0            ; 75 - spare
17170  7,1170     00000 1               DS       %0            ; 76 - spare
17171  7,1171     00000 1               DS       %0            ; 77 - spare
17172  7,1172     00000 1               DS       %0            ; 78 - spare
17173  7,1173     00000 1               DS       %0            ; 79 - spare
17174  7,1174     00000 1               DS       %0            ; 80 - spare
17175  7,1175     00000 1               DS       %0            ; 81 - spare
17176  7,1176     00000 1               DS       %0            ; 82 - spare
17177  7,1177     00000 1               DS       %0            ; 83 - spare
17200  7,1200     00000 1               DS       %0            ; 84 - spare
17201  7,1201     00000 1               DS       %0            ; 85 - spare
17202  7,1202     00000 1               DS       %0            ; 86 - spare
17203  7,1203     00000 1               DS       %0            ; 87 - spare
17204  7,1204     00000 1               DS       %0            ; 88 - spare
17205  7,1205     00000 1               DS       %0            ; 89 - spare
17206  7,1206     00000 1               DS       %0            ; 90 - spare
17207  7,1207     00000 1               DS       %0            ; 91 - spare
17210  7,1210     00000 1               DS       %0            ; 92 - spare
17211  7,1211     00000 1               DS       %0            ; 93 - spare
17212  7,1212     00000 1               DS       %0            ; 94 - spare
17213  7,1213     00000 1               DS       %0            ; 95 - spare
17214  7,1214     00000 1               DS       %0            ; 96 - spare
17215  7,1215     00000 1               DS       %0            ; 97 - spare
17216  7,1216     00000 1               DS       %0            ; 98 - spare
17217  7,1217     00000 1               DS       %0            ; 99 - spare
                        ; end of mixed noun scale factor routine table

                        BANK42_4       EQU      *

                        ; extended verb tables

                                       ORG      BANK43_1
                                       INCL     bank43_1.asm   ; COLOSSUS pp. 230-232
                        ;==============================================================================
                        ; DISPLAY ROUTINES (file:bank43_1.asm)
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, pp. 230.
                        ;==============================================================================

                        ;------------------------------------------------------------------------------
                        ; GOEXTVB -- EXTENDED VERBS
                        ;
                        ; Adapted from the AGC Block II COLOSSUS rev 249 assembly listing,
                        ; Oct 28, 1968, p.230.
                        ;------------------------------------------------------------------------------

                        GOEXTVB        EQU      *
20000 10,0000 2  0,0130 1              INDEX    MPAC          ; verb-40 is in MPAC
20001 10,0001 0 10,6002 0              TC       LST2FAN       ; fan as before

                        LST2FAN        EQU      *
20002 10,0002 0 10,6076 0              TC       ALM_END       ; VB40 - spare
20003 10,0003 0 10,6076 0              TC       ALM_END       ; VB41 - spare
20004 10,0004 0 10,6076 0              TC       ALM_END       ; VB42 - spare
20005 10,0005 0 10,6076 0              TC       ALM_END       ; VB43 - spare
20006 10,0006 0 10,6076 0              TC       ALM_END       ; VB44 - spare
20007 10,0007 0 10,6076 0              TC       ALM_END       ; VB45 - spare
20010 10,0010 0 10,6076 0              TC       ALM_END       ; VB46 - spare
20011 10,0011 0 10,6076 0              TC       ALM_END       ; VB47 - spare
20012 10,0012 0 10,6076 0              TC       ALM_END       ; VB48 - spare
20013 10,0013 0 10,6076 0              TC       ALM_END       ; VB49 - spare
20014 10,0014 0 10,6076 0              TC       ALM_END       ; VB50 - spare
20015 10,0015 0 10,6076 0              TC       ALM_END       ; VB51 - spare
20016 10,0016 0 10,6076 0              TC       ALM_END       ; VB52 - spare
20017 10,0017 0 10,6076 0              TC       ALM_END       ; VB53 - spare
20020 10,0020 0 10,6076 0              TC       ALM_END       ; VB54 - spare
20021 10,0021 0 10,6076 0              TC       ALM_END       ; VB55 - spare
20022 10,0022 0 10,6076 0              TC       ALM_END       ; VB56 - spare
20023 10,0023 0 10,6076 0              TC       ALM_END       ; VB57 - spare
20024 10,0024 0 10,6076 0              TC       ALM_END       ; VB58 - spare
20025 10,0025 0 10,6076 0              TC       ALM_END       ; VB59 - spare
20026 10,0026 0 10,6076 0              TC       ALM_END       ; VB60 - spare
20027 10,0027 0 10,6076 0              TC       ALM_END       ; VB61 - spare
20030 10,0030 0 10,6076 0              TC       ALM_END       ; VB62 - spare
20031 10,0031 0 10,6076 0              TC       ALM_END       ; VB63 - spare
20032 10,0032 0 10,6076 0              TC       ALM_END       ; VB64 - spare
20033 10,0033 0 10,6076 0              TC       ALM_END       ; VB65 - spare
20034 10,0034 0 10,6076 0              TC       ALM_END       ; VB66 - spare
20035 10,0035 0 10,6076 0              TC       ALM_END       ; VB67 - spare
20036 10,0036 0 10,6076 0              TC       ALM_END       ; VB68 - spare
20037 10,0037 0 10,6076 0              TC       ALM_END       ; VB69 - spare
20040 10,0040 0 10,6076 0              TC       ALM_END       ; VB70 - spare
```

```
20041 10,0041 0 10,6076 0                TC       ALM_END        ; VB71 - spare
20042 10,0042 0 10,6076 0                TC       ALM_END        ; VB72 - spare
20043 10,0043 0 10,6076 0                TC       ALM_END        ; VB73 - spare
20044 10,0044 0 10,6076 0                TC       ALM_END        ; VB74 - spare
20045 10,0045 0 10,6076 0                TC       ALM_END        ; VB75 - spare
20046 10,0046 0 10,6076 0                TC       ALM_END        ; VB76 - spare
20047 10,0047 0 10,6076 0                TC       ALM_END        ; VB77 - spare
20050 10,0050 0 10,6076 0                TC       ALM_END        ; VB78 - spare
20051 10,0051 0 10,6076 0                TC       ALM_END        ; VB79 - spare
20052 10,0052 0 10,6076 0                TC       ALM_END        ; VB80 - spare
20053 10,0053 0 10,6076 0                TC       ALM_END        ; VB81 - spare
20054 10,0054 0 10,6076 0                TC       ALM_END        ; VB82 - spare
20055 10,0055 0 10,6076 0                TC       ALM_END        ; VB83 - spare
20056 10,0056 0 10,6076 0                TC       ALM_END        ; VB84 - spare
20057 10,0057 0 10,6076 0                TC       ALM_END        ; VB85 - spare
20060 10,0060 0 10,6076 0                TC       ALM_END        ; VB86 - spare
20061 10,0061 0 10,6076 0                TC       ALM_END        ; VB87 - spare
20062 10,0062 0 10,6076 0                TC       ALM_END        ; VB88 - spare
20063 10,0063 0 10,6076 0                TC       ALM_END        ; VB89 - spare
20064 10,0064 0 10,6076 0                TC       ALM_END        ; VB90 - spare
20065 10,0065 0 10,6076 0                TC       ALM_END        ; VB91 - spare
20066 10,0066 0 10,6076 0                TC       ALM_END        ; VB92 - spare
20067 10,0067 0 10,6076 0                TC       ALM_END        ; VB93 - spare
20070 10,0070 0 10,6076 0                TC       ALM_END        ; VB94 - spare
20071 10,0071 0 10,6076 0                TC       ALM_END        ; VB95 - spare
20072 10,0072 0 10,6076 0                TC       ALM_END        ; VB96 - spare
20073 10,0073 0 10,6076 0                TC       ALM_END        ; VB97 - spare
20074 10,0074 0 10,6076 0                TC       ALM_END        ; VB98 - spare
20075 10,0075 0 10,6076 0                TC       ALM_END        ; VB99 - spare


                         ALM_END         EQU      *
20076 10,0076 0  2,4701 0                TC       FALTON         ; turn on operator error light
20077 10,0077 0  1,3653 1 GOPIN          TC       POSTJUMP
20100 10,0100     05067 0                FCADR    PINBRNCH



                         BANK43_2        EQU      *

                         ;-----------------------------------------------------------------------------
                         ; TEST JOBS & TASKS
                         ;-----------------------------------------------------------------------------

                                         ORG      BANKFF_6

                         ;-----------------------------------------------------------------------------
                         ; MAJOR MODES
                         ;-----------------------------------------------------------------------------

                                         ORG      BANK11

                         ;-----------------------------------------------------------------------------
                         ; P00 CMC IDLE PROGRAM
                         ;
                         ; Does nothing
                         ;-----------------------------------------------------------------------------

                         P00             EQU      *

                         ; Start any jobs or tasks needed at AGC initialization.

22000 11,0000 3 11,6004 0                CAF      time1          ; add a test task
22001 11,0001 0  1,2232 0                TC       WAITLIST
22002 11,0002     22005 1                CADR     task1          ; 14-bit task address
22003 11,0003 0  1,2723 0                TC       ENDOFJOB


                         ; TEST CODE - task started by P00

22004 11,0004     01750 1 time1          DS       1000           ; 10 seconds

                         task1           EQU      *
22005 11,0005 3 11,6011 1                XCH      prio1          ; job priority
22006 11,0006 0  1,3162 1                TC       NOVAC
22007 11,0007     22012 1                CADR     job1           ; 14 bit job address

22010 11,0010 0  1,2413 0                TC       TASKOVER


                         ; TEST CODE - job started by task

22011 11,0011     00003 1 prio1          DS       %3             ; lowest priority

                         job1            EQU      *
22012 11,0012 3  1,2050 0                CAF      ZERO
```

```
22013 11,0013 6  0,0053 1               AD       %53
22014 11,0014 6  1,2051 1               AD       ONE
22015 11,0015 5  0,0053 1               TS       %53            ; incr data at this address
22016 11,0016 0  1,2723 0               TC       ENDOFJOB

                         ;-----------------------------------------------------------------------
                         ; P01 DEMO PROGRAM
                         ;
                         ; Calls pinball: verb 1, noun 4.
                         ;-----------------------------------------------------------------------

22017 11,0017     00204 1 nvcode1        DS       %0204          ; verb 01, noun 04
22020 11,0020     22024 1 restart1_addr  DS       P01_restart
22021 11,0021     00042 1 tcadr1         DS       %42


                         P01            EQU      *
22022 11,0022 3 11,6021 1               CAF      tcadr1         ; load 'machine address to be specified'
22023 11,0023 5  0,0132 1               TS       MPAC+2


                         P01_restart    EQU      *
22024 11,0024 3 11,6017 1               CAF      nvcode1
22025 11,0025 0  2,4503 0               TC       NVSUB

22026 11,0026 0 11,6030 1               TC       *+2            ; display busy
22027 11,0027 0  1,2723 0               TC       ENDOFJOB       ; execution of verb/noun succeeded

22030 11,0030 3 11,6020 0               CAF      restart1_addr
22031 11,0031 0  2,5001 0               TC       NVSUBUSY       ; go to sleep until display released

22032 11,0032 0  1,2723 0               TC       ENDOFJOB       ; error: another job is already waiting


                         ;-----------------------------------------------------------------------
                         ; P02 DEMO PROGRAM
                         ;
                         ; Calls pinball: verb 21, noun 2.
                         ;
                         ; Sleeps if DSKY is busy until KEYREL. Executes verb 21, noun 2 to do
                         ; an external load. Then it sleeps with ENDIDLE until the user loads
                         ; the data or terminatest the load with PROCEED or TERMINATE.
                         ; NOTE: routines that call ENDIDLE must be in fixed-switchable memory
                         ;-----------------------------------------------------------------------

22033 11,0033     05202 1 nvcode2        DS       %05202         ; verb 21, noun 02
22034 11,0034     22040 0 restart2_addr  DS       P02_restart
22035 11,0035     00042 1 tcadr2         DS       %42


                         P02            EQU      *
22036 11,0036 3 11,6035 1               CAF      tcadr2
22037 11,0037 5  0,0132 1               TS       MPAC+2

                         P02_restart    EQU      *
22040 11,0040 3 11,6033 1               CAF      nvcode2
22041 11,0041 0  2,4503 0               TC       NVSUB

22042 11,0042 0 11,6044 1               TC       *+2            ; display busy
22043 11,0043 0 11,6047 1               TC       P02_wait       ; execution of verb/noun succeeded

22044 11,0044 3 11,6034 0               CAF      restart2_addr
22045 11,0045 0  2,5001 0               TC       NVSUBUSY       ; go to sleep until display released
22046 11,0046 0  1,2723 0               TC       ENDOFJOB       ; another job is already sleeping

                         P02_wait       EQU      *
22047 11,0047 0  2,4541 0               TC       ENDIDLE
22050 11,0050 0 11,6060 1               TC       P02_ter        ; terminate
22051 11,0051 0 11,6055 1               TC       P02_pwd        ; proceed without data
22052 11,0052 3  1,2051 1               CAF      ONE            ; data in
22053 11,0053 5  0,0043 0               TS       %43            ; set loc=1
22054 11,0054 0  1,2723 0               TC       ENDOFJOB

                         P02_pwd        EQU      *              ; proceed without data
22055 11,0055 3  1,2052 1               CAF      TWO
22056 11,0056 5  0,0043 0               TS       %43            ; set loc=2
22057 11,0057 0  1,2723 0               TC       ENDOFJOB

                         P02_ter        EQU      *              ; terminate
22060 11,0060 3  1,2053 0               CAF      THREE
22061 11,0061 5  0,0043 0               TS       %43            ; set loc=3
22062 11,0062 0  1,2723 0               TC       ENDOFJOB


                         ;-----------------------------------------------------------------------
                         ; P03 DEMO PROGRAM
                         ;
```

```
                        ; Nearly identical to P02, except that the job does not go to sleep
                        ; waiting for the load with ENDIDLE. Instead, it busy-waits on LOADSTAT.
                        ; NOTE: routines that call ENDIDLE must be in fixed-switchable memory
                        ;-----------------------------------------------------------------------------

22063 11,0063     05202 1 nvcode3        DS       %05202         ; verb 21, noun 02
22064 11,0064     22070 0 restart3_addr  DS       P03_restart
22065 11,0065     00042 1 tcadr3         DS       %42


                        P03            EQU      *
22066 11,0066 3 11,6065 1              CAF      tcadr3
22067 11,0067 5  0,0132 1              TS       MPAC+2

                        P03_restart    EQU      *
22070 11,0070 3 11,6063 1              CAF      nvcode3
22071 11,0071 0  2,4503 0              TC       NVSUB

22072 11,0072 0 11,6074 1              TC       *+2            ; display busy
22073 11,0073 0 11,6077 1              TC       P03_wait       ; execution of verb/noun succeeded

22074 11,0074 3 11,6064 0              CAF      restart3_addr
22075 11,0075 0  2,5001 0              TC       NVSUBUSY       ; go to sleep until display released
22076 11,0076 0  1,2723 0              TC       ENDOFJOB       ; another job is already sleeping

                        P03_wait       EQU      *
22077 11,0077 1  0,0503 0              CCS      LOADSTAT
22100 11,0100 0 11,6115 1              TC       P03_pwd        ; >0, verb "proceed w/o data" has been keyed
in
22101 11,0101 0 11,6107 1              TC       P03_yield      ; +0, waiting for data
22102 11,0102 0 11,6120 1              TC       P03_ter        ; <0, verb "terminate" has been keyed in
22103 11,0103 3  0,0000 1              NOOP                    ; -0, load has been completed

22104 11,0104 3  1,2051 1              CAF      ONE            ; data in
22105 11,0105 5  0,0043 0              TS       %43            ; set loc=1
22106 11,0106 0  1,2723 0              TC       ENDOFJOB

                        P03_yield      EQU      *
22107 11,0107 3  1,2051 1              CAF      ONE
22110 11,0110 6  0,0043 0              AD       %43
22111 11,0111 5  0,0043 0              TS       %43            ; incr loc while busy-waiting

22112 11,0112 1  0,0307 1              CCS      newJob         ; yield to higher priority job?
22113 11,0113 0  1,2733 1              TC       CHANG1         ; yes
22114 11,0114 0 11,6077 1              TC       P03_wait       ; no, keep busy-waiting

                        P03_pwd        EQU      *              ; proceed without data
22115 11,0115 3  1,2052 1              CAF      TWO
22116 11,0116 5  0,0043 0              TS       %43            ; set loc=2
22117 11,0117 0  1,2723 0              TC       ENDOFJOB

                        P03_ter        EQU      *              ; terminate
22120 11,0120 3  1,2053 0              CAF      THREE
22121 11,0121 5  0,0043 0              TS       %43            ; set loc=3
22122 11,0122 0  1,2723 0              TC       ENDOFJOB


                        ;-----------------------------------------------------------------------------
                        ; P04 DEMO PROGRAM
                        ;
                        ; Calls pinball: monitor verb 11, noun 04.
                        ;-----------------------------------------------------------------------------

22123 11,0123     02604 1 nvcode4        DS       %02604         ; verb 11, noun 04
22124 11,0124     22131 1 restart4_addr  DS       P04_restart
22125 11,0125     00042 1 tcadr4         DS       %42
                        ;mon_option      DS        %6
22126 11,0126     02206 1 mon_option     DS       %2206


                        P04            EQU      *
22127 11,0127 3 11,6125 1              CAF      tcadr4         ; load 'machine address to be specified'
22130 11,0130 5  0,0132 1              TS       MPAC+2


                        P04_restart    EQU      *
22131 11,0131 3 11,6126 1              CAF      mon_option     ; paste verb 09, blank R2, R3
22132 11,0132 5  0,0564 0              TS       NVSUB_L

22133 11,0133 3 11,6123 1              CAF      nvcode4
22134 11,0134 0  2,4507 1              TC       NVMONOPT       ; was NVSUB

22135 11,0135 0 11,6137 1              TC       *+2            ; display busy
22136 11,0136 0  1,2723 0              TC       ENDOFJOB       ; execution of verb/noun succeeded

22137 11,0137 3 11,6124 0              CAF      restart4_addr
22140 11,0140 0  2,5001 0              TC       NVSUBUSY       ; go to sleep until display released
```

```
22141 11,0141 0  1,2723 0               TC       ENDOFJOB       ; error: another job is already waiting


                         ;------------------------------------------------------------------------
                         ; P78 DEMO PROGRAM
                         ;
                         ;------------------------------------------------------------------------

                         P78            EQU      *
22142 11,0142 3  1,2050 0               CAF      ZERO
22143 11,0143 6  0,0051 0               AD       %51
22144 11,0144 6  1,2051 1               AD       ONE
22145 11,0145 5  0,0051 0               TS       %51            ; incr data at this address
22146 11,0146 0  1,2723 0               TC       ENDOFJOB

                         ;------------------------------------------------------------------------
                         ; P79 DEMO PROGRAM
                         ;
                         ;------------------------------------------------------------------------

                         P79            EQU      *
22147 11,0147 3  1,2050 0               CAF      ZERO
22150 11,0150 6  0,0052 0               AD       %52
22151 11,0151 6  1,2051 1               AD       ONE
22152 11,0152 5  0,0052 0               TS       %52            ; incr data at this address
22153 11,0153 0  1,2723 0               TC       ENDOFJOB


Assembly complete. Errors = 0

Symbol table:
BANK0           000057   MAXTASK        000007   MAXVAL         037777
MAXDELAY        027340   MAXTIMEOUT     010440   TSKTIME        000000
TSKADDR         000001   TRECSZ         000002   WL_taskList    000057
WL_IN_saveQ     000075   WL_IN_taskPtr  000076   WL_IN_loopCnt  000077
WL_AT_saveQ     000100   WL_AT_taskPtr  000101   WL_AT_newTime  000102
WL_AT_timeLeft  000103   WL_AT_loopCnt  000104   WL_T3_saveQ    000105
WL_T3_oldBank   000106   WL_ST_saveQ    000107   WL_ST_taskPtr  000110
WL_ST_newTime   000111   WL_ST_loopCnt  000112   WL_RT_saveQ    000113
WL_RT_runAddr   000114   WL_RM_saveQ    000115   WL_RM_taskPtr  000116
WL_RM_taskPtr2  000117   WL_RM_loopCnt  000120   WL_RM_retval   000121
WL_IS_newTime   000122   WL_IS_newAddr  000123   WL_IS_saveQ    000124
WL_IS_taskPtr   000125   WL_IS_taskPtr2 000126   WL_IS_loopCnt  000127
MAXJOBS         000007   JRECSZ         000015   EX_currentJob  000130
MPAC            000130   MODE           000137   LOC            000140
BANKSET         000141   PUSHLOC        000142   PRIORITY       000143
JOBPRIOBASE     000144   JREC0          000145   JREC1          000162
JREC2           000177   JREC3          000214   JREC4          000231
JREC5           000246   JREC6          000263   EX_jobList     000300
LOCCTR          000300   CHGJOB         000001   KEEPJOB        000000
newJob          000307   EX_JW_saveQ    000310   EX_JW_loopCnt  000311
EX_JW_CADR      000312   EX_JW_foundit  000313   EX_JW_jobPtr   000314
EX_JW_jobPtr2   000315   EX_JW_fndIndx  000316   EX_AJ_saveQ    000317
EX_AJ_loopCnt   000320   EX_AJ_jobPrio  000321   EX_AJ_jobPtr   000322
EX_AJ_field     000323   EX_AJ_findx    000324   EX_IN_saveQ    000325
EX_IN_loopCnt   000326   EX_IN_jobPtr   000327   EX_IN_recIndex 000330
EX_IN_field     000331   EX_IN_findx    000332   EX_MN_runAddr  000333
EX_MN_field     000334   EX_MN_findx    000335   EX_RM_saveQ    000336
EX_RM_jobPtr    000337   EX_RM_jobPtr2  000340   EX_RM_savePtr  000341
EX_RM_loopCnt   000342   EX_RM_retval   000343   EX_RM_field    000344
EX_RM_findx     000345   EX_IS_newPrio  000346   EX_IS_newPrioB 000347
EX_IS_newLoc    000350   EX_IS_saveQ    000351   EX_IS_jobPtr   000352
EX_IS_jobPtr2   000353   EX_IS_loopCnt  000354   FLAGWRD5       000355
ITEMP1          000356   WAITEXIT       000356   EXECTEM1       000356
ITEMP2          000357   WAITBANK       000357   EXECTEM2       000357
ITEMP3          000360   RUPTSTOR       000360   WAITADR        000360
NEWPRIO         000360   ITEMP4         000361   WAITTEMP       000361
ITEMP5          000362   NEWLOC         000362   ITEMP6         000363
NEWLOCP1        000363   NEWJOB         000364   RUPTREG1       000365
RUPTREG2        000366   RUPTREG3       000367   RUPTREG4       000370
KEYTEMP1        000370   DSRUPTEM       000370   STATE          000371
FLAGFILL        000405   EMDOT          000405   STATEXIT       000407
INTB15P         000411   DSEXIT         000411   EXITEM         000411
BLANKRET        000411   INTBIT15       000412   WRDRET         000412
WDRET           000412   DECRET         000412   _2122REG       000412
ADDRWD          000413   POLISH         000414   UPDATRET       000414
CHAR            000414   ERCNT          000414   DECOUNT        000414
FIXLOC          000415   OVFIND         000416   VBUF           000417
SGNON           000417   NOUNTEM        000417   DISTEM         000417
DECTEM          000417   SGNOFF         000420   NVTEMP         000420
SFTEMP1         000420   HITEMIN        000420   CODE           000421
SFTEMP2         000421   LOWTEMIN       000421   MIXTEMP        000422
SIGNRET         000422   BUF            000425   BUF2           000430
INDEXLOC        000425   SWWORD         000425   SWBIT          000426
MPTEMP          000432   DMPNTEMP       000432   DOTINC         000433
```

DVSIGN	000433	ESCAPE	000433	ENTRET	000433
DOTRET	000434	DVNORMCT	000434	ESCAPE2	000434
WDCNT	000434	INREL	000434	MATINC	000435
MAXDVSW	000435	POLYCNT	000435	DSPMMTEM	000435
MIXBR	000435	TEM1	000436	POLYRET	000436
DSREL	000436	TEM2	000437	DSMAG	000437
IDADDTEM	000437	TEM3	000440	COUNT	000440
TEM4	000441	LSTPTR	000441	RELRET	000441
FREERET	000441	DSPWDRET	000441	SEPSCRET	000441
SEPMNRET	000441	TEM5	000442	NOUNADD	000442
NNADTEM	000443	NNTYPTEM	000444	IDAD1TEM	000445
IDAD2TEM	000446	IDAD3TEM	000447	RUTMXTEM	000450
DEXDEX	000437	DEX1	000440	DEX2	000441
RTNSAVER	000442	TERM1TMP	000430	RESTREG	000451
NVWORD	000452	MARXNV	000453	NVSAVE	000454
CADRFLSH	000455	CADRMARK	000456	TEMPFLSH	000457
FAILREG	000460	MINDEX	000463	MMNUMBER	000464
DSPCNT	000465	DSPCOUNT	000466	DECBRNCH	000467
VERBREG	000470	NOUNREG	000471	XREG	000472
YREG	000473	ZREG	000474	XREGLP	000475
YREGLP	000476	HITEMOUT	000476	ZREGLP	000477
LOTEMOUT	000477	MODREG	000500	DSPLOCK	000501
REQRET	000502	LOADSTAT	000503	CLPASS	000504
NOUT	000505	NOUNCADR	000506	MONSAVE	000507
MONSAVE1	000510	MONSAVE2	000511	DSPTAB	000512
NVQTEM	000526	NVBNKTEM	000527	VERBSAVE	000530
CADRSTOR	000531	DSPLIST	000532	EXTVRACT	000533
DSPTEM1	000534	DSPTEM2	000537	DSPTEMX	000537
NORMTEM1	000534	OPTIONX	000537	MMTEMP	000542
DSRUPTSW	000543	T4RET	000544	DSPOUTRET	000545
DK_IN_saveQ	000546	LXCH_LPRET	000547	LXCH_A	000550
KP_MPAC	000551	DPTEST_A	000552	DPTEST_Q	000553
REQ_Q	000554	SETNCADR_Q	000555	ALLDC_OC_Q	000556
SFRUTMIX_L	000557	SFCONUM_L	000560	BLANKSUB_Q	000561
GTSF_RET	000562	FR_RETQ	000563	NVSUB_L	000564
NVSUB_A	000565	ENDIDLE_L	000566	NBSUBSY1_L	000567
FLASHRET	000570	PASTE_TMP	000571	NEWMODEA_Q	000572
SHORTMP_A	000573	SHORTMP_OVFL	000574	SHORTMP_OVFH	000575
ADDRWD1	000576	MATH_Q	000577	PRSHRTMP_Q	000600
KEYRET	000601	SAVEQ	000602	BJBANK	000603
BJRET	000604	PJBANK	000605	PJRET	000606
PJA	000607	BCBANK	000610	BCRET	000611
BCA	000612	MBCBANK	000613	MBCRET	000614
MBCA	000615	DCBANK	000616	DCRET	000617
EXTENDER	005777	GOPROG	002000	T3RUPT	002004
ERRUPT	002010	DSRUPT	002014	KEYRUPT	002020
UPRUPT	002024	endRUPT	002030	goT3	002034
goER	002036	goDS	002037	goKEY	002041
goUP	002043	ofbit	002044	NEG0	002045
NEG1	002046	NEG2	002047	ZERO	002050
ONE	002051	TWO	002052	THREE	002053
FOUR	002054	FIVE	002055	SIX	002056
SEVEN	002057	TEN	002060	ELEVEN	002061
BIT15	002062	BIT14	002063	BIT13	002064
BIT12	002065	BIT11	002066	BIT10	002067
BIT9	002070	BIT8	002071	BIT7	002072
BIT6	002073	BIT5	002074	BIT4	002075
BIT3	002076	BIT2	002077	BIT1	002100
LOW7	002101	bankAddr	002102	lowAddr	002103
OCT1400	002104	NOUTCON	002105	POSMAX	002106
CLRMEM	002107	CLRMEM_CHK	002113	CLRMEM_WORD	002116
CLRMEM_VAL	002050	TIME3	000037	CLRMEM_BADDR	000037
CLRMEM_WC	002123	V37BANK	002124	SAMASK	002125
goMAIN	002126	SLAP1	002126	goMMchange	002147
V37XEQ	002153	V37XEQC	002166	WL_taskRecSize	002174
WL_tskLstStart	002175	WL_tskLstEnd	002176	WL_numTasks	002177
WL_numTasks1	002200	WL_maxVal	002201	WL_maxDelay	002202
WL_maxTimeOut	002203	WL_initWL	002204	WL_IN_loop	002213
WAITLIST	002232	WL_AT_noOvf	002264	WL_AT_chkOrder	002270
WL_AT_mkFirst	002306	WL_AT_loop	002311	WL_AT_schTsk	002333
WL_AT_done	002343	WL_TIME3task	002347	WL_runTasks	002362
WL_RT_loop	002364	WL_RT_runIt	002411	TASKOVER	002413
WL_RT_done	002414	WL_schedTask	002417	WL_ST_loop	002436
WL_ST_setT3	002461	WL_ST_noTask	002466	WL_ST_done	002470
WL_insert	002473	WL_IS_loop	002510	WL_IS_bumpPtr	002541
WL_IS_insRec	002552	WL_IS_done	002562	WL_remove	002565
WL_RM_loop	002603	WL_RM_done	002636	EX_WAKE_PRIO	002642
EX_DUMMY_PRIO	002643	EX_SLEEP_PRIO	002644	EX_jobCurStart	002645
EX_jobRecSize	002646	EX_jobLstStart	002647	EX_jobLstEnd	002650
EX_jobLstEnd1	002651	EX_numJobs	002652	EX_numJobs1	002653
EX_changeJob	002654	EX_keepJob	002655	EX_exec	002656
EX_MN_findJob	002662	EX_MN_setFlg	002677	EX_MN_runJob	002701
EX_MN_runIt	002720	ENDOFJOB	002723	JOBSLEEP	002725
CHANG1	002733	EX_MN_notBank	002750	EX_MN_saveIt	002752
EX_MN_mvRec	002757	EX_MN_loop3	002763	EX_MN_done3	003002
JOBWAKE	003003	EX_JW_loop	003015	EX_JW_moveRec	003035
EX_JW_bumpPtr	003041	EX_JW_done	003056	EX_JW_return	003074

SPVAC	003075	EX_SP_loop1	003110	EX_SP_done1	003127
EX_SP_testFlg	003146	EX_SP_done2	003156	FINDVAC	003161
NOVAC	003162	EX_AJ_loop1	003200	EX_AJ_done1	003217
EX_AJ_testFlg	003236	EX_AJ_done2	003246	EX_initEX	003252
EX_IN_loop1	003266	EX_IN_loop2	003304	EX_IN_loop3	003307
EX_IN_done	003324	EX_findIns	003332	EX_FI_loop	003347
EX_FI_bumpPtr	003376	EX_FI_insRec	003405	EX_FI_done	003405
EX_remove	003410	EX_RM_loop1	003420	EX_RM_done1	003437
EX_RM_loop2	003446	EX_RM_done2	003465	EX_RM_loop3	003472
EX_RM_done3	003507	dumJob	003510	dumJob1	003514
dumJob2	003517	NOTACTLT	003525	DXCHJUMP	003526
DODXCHCALL	003547	DC_NOTBANK	003561	BANKCALL	003565
DOBANKCALL	003611	MYBANKCALL	003624	POSTJUMP	003653
DOPOSTJUMP	003677	BANKJUMP	003712	DOBANKJUMP	003733
DATACALL	003742	DODATACALL	003763	RELTAB	003772
RELTAB11	004005	DKTESTINIT	004006	DK_initDK	004007
DSPOFF	004012	T4PROG	004047	DSPOUTSR	004057
DSPSCAN	004065	TABLNTH	004072	_120MRUPT	004074
DSPLAY	004103	DSPOUT	004116	NODSPOUT	004126
DSPOUTEXIT	004126	CHRPRIO	004131	KEYPROG	004132
TPAGREE	004150	TPA_SGN0	004152	TPA_P0	004157
TPA_PZ0	004167	TPA_PZ0FIX	004200	TPA_M0	004210
TPA_MZ0	004220	TPA_MZ0FIX	004231	TPA_SGN1	004241
TPA_P1	004246	TPA_M1	004262	TPA_SGN2	004277
TPA_P2	004304	TPA_M2	004306	TPA_P3	004310
MAXPOS	004315	MAXNEG	004316	TPA_MPAC0	004317
TPA_MPAC1	004320	TPA_FIXM	004321	TPA_FIXP	004337
SHORTMP	004353	DMP	004374	BANKFF_1	004435
BANK4	010000	BANK04_1	010000	BANK5	012000
BANK40_1	012000	BANK6	014000	BANK41_1	014000
BANK7	016000	BANK42_1	016000	BANK10	020000
BANK43_1	020000	V37	010000	V37BAD	010002
CHECKTAB	010005	AGAINMM	010007	V37NONO	010026
FCADRMM1	010030	PREMM1	010037	EPREMM1	010046
NOV37MM	010046	BANK04_2	010047	CHARIN	012000
CHARIN2	012016	ELRCODE1	012062	ENTERJMP	012063
_89TEST	012065	NUM	012101	DECTOBIN	012137
ENDNMTST	012155	ENDNUM	012170	ENDALL	012174
DECEND	012176	PDECSGN	012220	MORNUM	012227
CRITCON	012232	DECON	012237	GETINREL	012241
INRELTAB	012245	CCSHOLE	012271	VERB	012272
NVCOM	012275	NOUN	012306	NEGSGN	012312
BOTHSGN	012315	PIXCLPAS	012321	POSGN	012326
P_ON	012332	SGNCOM	012342	M_ON	012353
SGNTAB	012364	SIGNTEST	012367	SGNTST1	012404
CLEAR	012412	CLPASHI	012431	CLEAR1	012454
CLR5	012461	LEGALTST	012464	_5BLANK	012473
_5BLANK1	012515	SINBLANK	012531	DOUBLK	012534
BRNCHCON	012537	_2BLANK	012540	BLANKCON	012563
BANK40_2	012564	NVSUBR	014000	LOADLV1	014001
ENTER	014002	ENTPASHI	014012	ACCEPTWD	014032
ENTEXIT	000433	MMADREF	014036	LOWVERB	014037
ENTPAS0	014040	TESTVB	014044	TESTNN	014054
REQADD	014073	USEADD	014117	LODNNLOC	014124
NEG5	014126	INTMCTBS	014127	VERBFAN	014151
LST2CON	014163	VBFANDIR	014164	VERBTAB	014167
MIXNOUN	014237	DPTEST	014240	DPTEST1	014262
REQDATX	014264	REQDATY	014270	REQDATZ	014274
REQCOM	014277	ENDRQDAT	014305	UPDATNN	014307
PUTADD	014320	UPDATVB	014327	UPDAT1	014335
GOALMCYC	014340	GODSPALM	014341	DSPABC	014343
DSPAB	014350	DSPA	014355	DSPCOM1	014361
DSPB	014363	DSPC	014370	DSPCOM2	014375
DSPCOM3	014403	COMPTEST	014414	COMPTST1	014417
NDOMPTST	014427	DCOMPTST	014430	DECTEST	014435
DCTSTCYC	014444	NOUNTEST	014453	TSTFORDP	014462
COMPICK	014477	GETCOMP	014501	DECDSP	014510
DSPDCGET	014514	DSPDCPUT	014524	DSPSFNOR	014546
GTSFOUTL	014550	DSPDCEND	014551	DECDSP3	014560
SFOUTABR	014563	BANK41_2	014600	DEGOUTSF	012564
SETAUG	012572	FIXRANGE	012603	DEGCOM	012616
DEGTAB	012640	ARTOUTSF	012644	SCOUTEND	012651
READLO	012653	READLO1	012665	RDLONOR	012701
ENDRDLO	012703	BANK40_3	012704	HMSOUT	016000
SECON1	016041	SECON2	016043	MINCON2	016045
MINCON1	016047	HRCON1	016051	SEPSECNR	016053
SEPMIN	016074	ENDSPMIN	016113	BANK42_2	016114
DSPDPDEC	012704	ENDDPDEC	012726	BANK40_4	012727
ABCLOAD	014600	PUTXYZ	014614	ABLOAD	014635
PUTXY	014646	ALOAD	014663	BLOAD	014673
CLOAD	014707	LOADLV	014723	VBSP1LD	014733
VBSP2LD	014734	VBSP3LD	014735	ALLDC_OC	014736
GOQ	014761	SFRUTNOR	014762	SFRUTMIX	014770
SFRET1	015002	SFCONUM	015003	SFRET	015020
DISPLACE	015022	CONUMNOR	015025	PUTCOM	015031
PUTDPCOM	015065	PUTNORM	015077	PUTNORM_1	015111
PUTCOM2	015111	GTSFINLC	015113	PUTDECSF	015114

PUTSFNOR	015125	PUTDCSF2	015126	SFINTABR	015131
BANK41_3	015146	DEGINSF	012727	DEGINSF2	012740
SIGNFIX	012750	ENDSCALE	012762	NEG180	012764
SGNTO1	012766	DEGCON1	012772	DEGCON2	012774
ARTHINSF	012776	BINROUND	013011	_2ROUND	013014
_2RNDEND	013024	TESTOFUF	013025	BANK40_5	013031
BANK42_3	016114	MONITOR	015146	MONIT1	015150
BIT15_14	015155	MONIT2	015164	MONREQ	015215
KILLMON	015232	MONDEL	015236	MONDO	015237
ENDMONDO	015273	MONREF	015274	MONBACK	015275
MONBUSY	015276	LODSAMPT	015300	BANK41_4	015301
PASTEVB	004435	PASTEOPT	004451	ENDPASTE	004472
MID7	004473	BANKFF_2	004474	DSPFMEM	015301
ENDSPF	015307	BANK41_5	015310	DSPSIGN	013031
DSPRND	013046	DPOSMAX	013062	DSPDECWD	013064
DSPDCWD1	013071	TRACE1	013074	TRACE1S	013105
DECROUND	013117	DSPDECNR	013120	DSPDC2NR	013124
DSP2DEC	013131	END2DEC	013143	DSPDECVN	013144
VNDSPCON	013155	GOVNUPDT	013156	BANK40_6	013161
DSPOCTWD	015310	WDAGAIN	015317	OCTBACK	015337
DSPLW	015341	DSPMSK	002057	DSP2BIT	015344
BANK41_6	015356	DSPIN	013161	DSPIN1	013206
DFRNT	013226	DSLV	013245	DSMSK	013247
_11DSPIN	013253	DSPOCTIN	013261	ENDSPOCT	013264
PREDSPAL	013265	DSPALARM	013267	CHARALRM	013307
MONADR	013313	NVSBENDL	013314	BANK40_7	013315
ALMCYCLE	004474	ENDALM	004502	BANKFF_3	004503
MMCHANG	015356	MODROUTR	010000	REQMM	015404
VBRQEXEC	015420	REQEX1	015425	REQUESTC	015431
SETVAC	015444	VBRQWAIT	015446	ENDRQWT	015452
BANK41_7	015453	VBPROC	013315	VBTERM	013323
VBRESEQ	013325	VBRELDSP	013327	TSTLTS4	013334
UNSUSPEN	013342	BANK40_8	013350	NVSUB	004503
NVMONOPT	004507	NVSBCOM	004516	NVSUBCOM	004524
NVSRRBNK	004531	NVSUBEND	004532	BANKFF_4	004536
BLANKDSP	015453	INCR_NOUT_RET	015463	INCR_NOUT	015501
NVSUB1	015505	ENTSET	015542	NVSUB2	015543
ENDNVSB1	015571	BANK41_8	015572	KILMONON	004536
ENDIDLE	004541	ENDINST	004553	ISCADR_P0	004554
ISLIST_P0	004560	DSPABORT	004563	BLANKSUB	004565
BSUB1ADDR	004612	BS_SUPDXCHZ	004613	BANKFF_5	004616
DSPMM	010047	ENDSPMM	010056	BANK04_3	010057
BLNKSUB1	013350	TESTBIT	013373	DSPMMJB	013400
RECALTST	013413	RECAL1	013416	RECAL2	013427
RECAL3	013444	DOTERM	013446	DOPROC	013450
BANK40_8a	013452	SETNCADR	004616	SETNADD	004625
SETEBANK	004633	R1D1	004635	R2D1	004636
R3D1	004637	RIGHT5	004640	LEFT5	004647
SLEFT5	004656	LOW5	004664	MID5	004665
HI5	004666	TCNOVAC	004667	TCWAIT	004670
TCFINDVAC	004671	LOW11	004672	B12M1	004672
LOW8	004673	LOW10	004674	VD1	004675
ND1	004676	MD1	004677	BINCON	004700
OUT1	000011	DSALMOUT	000011	FALTON	004701
FALTOF	004706	FALTOR	004712	RELDSPON	004713
RELDSPOR	004720	TPSL1	004721	PRSHRTMP	004740
DOSHRTMP	004754	FLASHON	004760	FLASHOFF	004770
FLSHTAB	005000	NVSUBUSY	005001	BANKFF_5a	005003
NVSUBSY1	013452	ENDNVBSY	013461	BANK40_9	013462
RELDSP	005003	RELDSP2	005017	RELDSP1	005026
NEWMODEA	005036	POODOO	005050	NOTPALT	005066
PINBRNCH	005067	BANKFF_6	005070	VBTSTLTS	015572
TSTLTS1	015600	FULLDSP	015621	FULLDSP1	015622
TSTCON1	015623	SHOLTS	015624	TSTLTS2	015625
TSTLTS3	015631	BANK41_9	015644	ERROR	013462
TSTAB	013467	ERMINUS	013477	ERPLUS	013502
ERCOM	013505	NOTBIT12	013517	ERCON	013520
BANK40_10	013521	LODNNTAB	016114	LODMIXNN	016135
LODNLV	016156	MIXCON	016161	GTSFOUT	016162
SFCOM	016173	GTSFIN	016176	NNADTAB	016210
NNTYPTAB	016354	SFINTAB	016520	SFOUTAB	016570
IDADDTAB	016640	RUTMXTAB	017124	BANK42_4	017220
GOEXTVB	020000	LST2FAN	020002	ALM_END	020076
GOPIN	020077	BANK43_2	020101	BANK11	022000
P00	022000	time1	022004	task1	022005
prio1	022011	job1	022012	nvcode1	022017
restart1_addr	022020	tcadr1	022021	P01	022022
P01_restart	022024	nvcode2	022033	restart2_addr	022034
tcadr2	022035	P02	022036	P02_restart	022040
P02_wait	022047	P02_pwd	022055	P02_ter	022060
nvcode3	022063	restart3_addr	022064	tcadr3	022065
P03	022066	P03_restart	022070	P03_wait	022077
P03_yield	022107	P03_pwd	022115	P03_ter	022120
nvcode4	022123	restart4_addr	022124	tcadr4	022125
mon_option	022126	P04	022127	P04_restart	022131
P78	022142	P79	022147	ARUPT	000026
Q	000001	QRUPT	000027	TIME1	000036

TIME2	000035	BANK	000015	A	000000
TIME4	000040	OUT0	000010	IN0	000004
LP	000003	OVCTR	000034	CYL	000022
SR	000021	CYR	000020		