RCSL: 44-RT 551

Author: Jens Falkenberg
Andersen

Edited: 8/11 - 1972

RC 3600 BINARY LOADER

Key: RC 3600, Binary Loader, Automatic Program Load.

Abstract: The RC 3600 Binary Loader is a routine used to load the absolute binary tapes produced as output by the Assembler. The loader is available in a special formattet tape: RCSL: 44-RT 550.

This tape can be loadet by the bootstrap program in ROM 007 and ROM 008.

ASCII tape: RCSL: 44-RT 552.

CONTENTS PAGE

1. REQUIREMENTS

1.1 Memory

2 K or larger alterable memory.

1.2 Equipment

Teletype ASR or paper tape reader.

1.3 External Subroutines

None.

1.4 Other

None.

2. OPERATING PROCEDURE

2.1 Calling Sequence

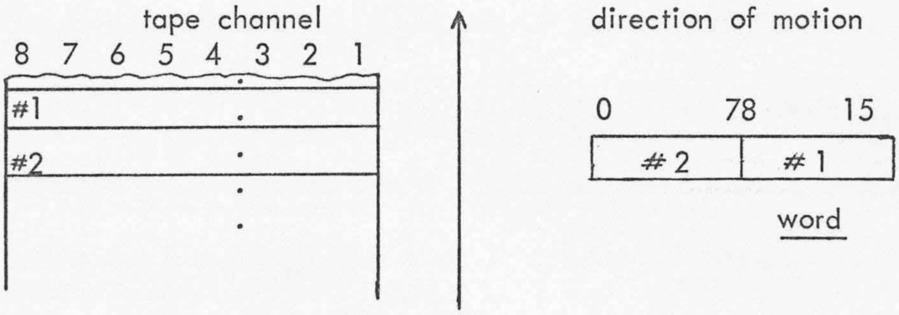
The Binary Loader must be loaded by using the Automatic Program Load procedure described in "How to Use Nova Computers". Special format tape RCSL: 44-RT 550 must be used.

The Binary Loader is started by entering SX777 in the data switches and depressing START. "X" represents the two most significant octal digits of the highest memory address available. For example, X = 07 for a 4K system and 17 for an 8K system. "S" represents bit 0 of the data switches and should be set if input is to be via the paper tape reader and reset if via the teletype.

2.2 Input Format

The input to the Loader is an absolute binary tape. The tape is punched in blocks separated by null (all zero) characters. The Loader reads two tape characters to form a 16-bit word.

The format is as follows:



In other words, the first tape character forms bits 8 - 15 of the data word, and the second tape character forms bits 0 - 7 of the data word.

The first non-null tape character indicates the start of a new block. Four different block types, data, multiple data, start, and error, are defined.
The block type is determined by the first word of the block. A description of each block type follows.

The first word, WC, of a Data Block is in the range $0 < WC \leq 20_8$.
Its format is:

word		
1	-WC	
2	ADDRESS	
3	CHECKSUM	
4	DATA WD 1	
5	DATA WD 2	
"	"	
"	"	WC = N
"	"	
3+N	DATA WD N	

The two's complement of WC is given in the first word. Normally sixteen data words will be punched per data block, but the .END and .LOC pseudo-ops to the Assembler may cause short blocks to be punched. The second word contains the address at which the first data word is to be loaded. Subsequent data words are loaded in sequentially ascending locations. The third word contains a checksum. This number is computed so that the binary sum of all words in the block should give a zero result. The remaining words are the data to be loaded.

The first word, WC, of a Multiple Data Block is in the range

$$20_8 < WC \leq 77777_8$$

Its format is:

word	
1	-WC
2	ADDRESS
3	CHECKSUM
4	DATA WD

where again the two's complement of WC is given in the first word. This block type is used to indicate that 16_{10} or more data words, all identical to the one data word punched, are to be loaded sequentially into memory locations beginnig at the absolute address, ADDRESS. In this case, the number of identical data words, n, is given by the formula

$$n = WC - 1$$

i.e. if the first word of the block is -17_{10}, the data is to be repeated 16_{10} times (note that WC is the absolute value of the first word). The checksum is computed in the same manner as an ordinary Data Block.

The first word of a Start Block is 000001. Its format is:

word	
1	000001
2	S ADDRESS
3	CHECKSUM

The second word uses bit 0 as a flag. If S = 1, the loader will transfer control after loading to the address in bits 1 - 15 of the second word. The checksum is the same as that for a Data Block.

The first word of an Error Block is greater than +1. Its format is:

word	
1	< 1
2	
.	IGNORED
.	
.	
.	
N	

The last byte of an error block is a rubout (377). An error block is ignored in its entirety by the Loader.

The binary tape to be loaded must be mounted in the input device selected by bit 0 of the data switches before starting the Loader.

2.3 Output Format

The output is a loaded routine ready for execution. If no starting address was given, the Loader will HALT at location XX741. Otherwise, control will be transferred to the loaded routine.

2.4 Error Returns

Two error conditions will cause the Loader to HALT at location XX727.

The first is a binary tape that attempts to overwrite the Loader. This is a fatal error, and the user must reassemble with a lower origin before loading will be successful.

The second is a checksum failure over the last block read. The binary tape should be repositioned to the beginning of the last block read andCONTINUE depressed. If this second attempt fails, the binary tape should be assumed to be incorrectly punched. The user must either reassemble to obtain a new binary tape, or he must proceed with the loading from the next block and after loading key in from the console the sixteen words of the block in error.

2.5 State of Active Registers upon Exit

If a checksum error occures, AC0 will contain the incorrect checksum.

If a binary tape attempted to overwrite the Loader, AC3 will contain the address which would have been overwritten.

2.6 Cautions to User

If possible, the user should write routines which do not destroy locations above XX635 (the start of the Loader). If he adheres to this practice, the Bootstrap and Binary Loaders will always be intact and need never be reloaded. Note that although the Loader will not load data above XX635, the user can write in this area during execution.

3. DISCUSSION

3.1 Algorithms

The binary loader reads in a frame of information at a time from the input device using a GTCHR routine. Once the start of a block has been detected (a non-null frame), the Loader assembles two frames at a time to construct a complete 16-bit word. The type of block is determine, i.e. start, data, multiple data, or error, and control is transferred to an appropriate processing routine. A start block terminates the loading process by causing control to be transferred to the starting address or causing the Loader to HALT.

3.2 Limitations and Accuracy

The Binary Loader will not permit itself to be overwritten.

3.3 Size and Timing

The Loader is 120 (octal) words in length, 116 of which immediately precede the Bootstrap Loader and the remaining two of which follow the Bootstrap.

The speed of the Loader is limited by the speed of the input device.

3.4 Flow Diagrams

Not applicable.

4. EXAMPLES AND APPLICATIONS

None.

5. ASSEMBLER LISTNING

```
                    ;RC 3600 BINARY LOADER                                 PAGE 1

                    ;
                    ; PREAMBLE FOR NEW BOOT PROGRAM
                    ;
      000777                .LOC     777       ; ANY NON PAGE ZERO WILL DO

      000030                GET=30

00777 000001                000001             ; TAPE SYNCHRONIZER
01000 177754                BEG-END-2          ; NEGATIVE WORD COUNY FOR PREAMBLE

01001 020421        BEG:    LDA      0,C4K     ; MEMORY SIZING INCREMENT
01002 176221                ADCZR    3,3,SKP   ; FORM HIGHEST ADDRESS
01003 116400        LOOP:   SUB      0,3       ; DECREMENT
01004 055400                STA      3,0,3     ; STORE ADDRESS
01005 031400                LDA      2,0,3     ; GET IT BACK
01006 172414                SUB#     3,2,SZR   ; SAME?
01007 000774                JMP      LOOP      ; NO - NO MEMORY
01010 004030                JSR      GET       ; GET
01011 044411                STA      1,C4K     ; SAVE COUNT OF BINLOADER
01012 133000                ADD      1,2       ; FORM FIRST ADDRESS
01013 151400                INC      2,2       ; INCREMENT ADDRESS
01014 004030                JSR      GET       ; GET
01015 045000                STA      1,0,2     ; SET INTO MEMORY
01016 010404                ISZ      C4K       ; BUMP COUNT
01017 000774                JMP      .-4       ; GO BACK
01020 063077                HALT               ; WHOA FAT HIPPO
01021 001000                JMP      0,2
01022 004000        C4K:    4000
01023 000756        END:    JMP      BEG       ; GETS CONTROL HERE
```

```
                 ;RC 3600 BINARY LOADER                          PAGE 2

                 ; START
                 ; BINARY BLOCK LOADER

                 ; SUBROUTINE TO ASSEMBLE A WORD INTO AC2, THIS WORD IS
                 ; ADDED INTO THE CHECKSUM HELD IN ACØ

      ØØ7635     .LOC    7635

07635 177636             BUILD-BEND-1      ; MINUS WORD COUNT FOR BIN LOADER

07636 Ø54512     BUILD:  STA     3,TEMP1   ; SAVE THE RETURN
07637 ØØ44Ø7             JSR     GTCHR     ; GET CHARACTER INTO AC3
0764Ø 1713ØØ             MOVS    3,2       ; AND SAVE IN THE LN OF AC2
07641 ØØ44Ø5             JSR     GTCHR     ; GET THE NEXT CHARACTER
07642 1733ØØ             ADDS    3,2       ; AND BUILD IN AC2
07643 143ØØØ             ADD     2,Ø       ; ADD INTO CHECKSUM
07644 ØØ25Ø4             JMP     @TEMP1    ; AND RETURN
07645 ØØØØØ4     DIFF:   4

                 ; SUBROUTINE TO GET A CHARACTER INTO AC3
                 ; IF SWITCHØ=Ø, USE TELETYPE, ELSE USE PTR
Ø7646 Ø545Ø3     GTCHR:  STA     3,TEMP2   ; SAVE THE RETURN
Ø7647 Ø345Ø3             LDA     3,SAVE    ; GET THE SWITCH WORD
Ø765Ø 1751Ø3             MOVL    3,3,SNC   ; AND TEST BIT Ø
Ø7651 ØØØ4Ø5             JMP     .+5       ; A Ø, USE THE TTI
Ø7652 Ø63612             SKPDN   PTR       ; A 1, USE THE PTR
Ø7653 ØØØ777             JMP     .-1
Ø7654 Ø74512             DIAS    3,PTR     ; READ INTO AC3 AND START
Ø7655 ØØ2474             JMP     @TEMP2    ; RETURN

Ø7656 Ø6361Ø             SKPDN   TTI       ; WAIT FOR TTI FLAG
Ø7657 ØØØ777             JMP     .-1
Ø766Ø Ø7451Ø             DIAS    3,TTI
Ø7661 ØØ247Ø             JMP     @TEMP2    ; EXIT

                 ; START OF THE LOADER
Ø7662 Ø62677     START:  IORST
Ø7663 Ø60477             READS   Ø         ; READ SWITCHES
Ø7664 Ø40466             STA     Ø,SAVE    ; AND SAVE THE WORD
Ø7665 Ø6Ø11Ø             NIOS    TTI       ; START BOTH READERS
Ø7666 Ø6Ø112             NIOS    PTR
```

```
                   ;RC 3600 BINARY LOADER                        PAGE 3

                   ; READ IN A BLOCK
07667 004757       BLOCK:  JSR     GTCHR    ; GET A CHARACTER
07670 171305               MOVS    3,2,SNR  ; AND TEST IT FOR ZERO
07671 000776               JMP     BLOCK    ; YES, STILL IN LEADER
07672 004754               JSR     GTCHR    ; OK, BUILD A WORD
07673 173300               ADDS    3,2      ; IN AC2
07674 141000               MOV     2,0      ; SET INTO THE CHECKSUM
07675 145000               MOV     2,1      ; SET THE COUNTER
07676 004740               JSR     BUILD    ; GO GET THE ADDRESS
07677 050477               STA     2,ADDRS  ; AND STORE IT
07700 004736               JSR     BUILD    ; READ THE CHECKSUM WORD
07701 125113               MOVL#   1,1,SNC  ; TEST THE COUNT
07702 000426               JMP     TEST     ; IT IS >0, IE A START OR IGNORE
07703 044450               STA     1,COUNT  ; BLOCK

                   ;READ IN THE DATA BLOCK
07704 030445               LDA     2,TEMP2  ; SEE IF STORAGE
07705 034740               LDA     3,DIFF
07706 172400               SUB     3,2
07707 034467               LDA     3,ADDRS  ; ADDRESS IS TOO BIG
07710 136400               SUB     1,3
07711 172023               ADCZ    3,2,SNC
07712 000414               JMP     CHKER    ; YES, HALT THE LOADER
07713 030441               LDA     2,C20
07714 147033               ADDZ#   2,1,SNC
07715 010436               ISZ     COUNT
07716 147022               ADDZ    2,1,SZC  ; REPEAT BLOCK?
07717 125113       STORE:  MOVL#   1,1,SNC
07720 004716               JSR     BUILD
07721 052455               STA     2,@ADDRS
07722 010454               ISZ     ADDRS
07723 010430               ISZ     COUNT
07724 000773               JMP     STORE
07725 101004               MOV     0,0,SZR  ; NOW, TEST THE CHECKSUM
07726 063077       CHKER:  HALT             ; CHECKSUM ERROR, AC0@VALUE
07727 000740               JMP     BLOCK    ; GO READ IN A BLOCK
```

```
                  ;RC 36ØØ BINARY LOADER                          PAGE 4

                  ; START BLOCK OR IGNORE BLOCK
Ø773Ø 125224      TEST:    MOVZR    1,1,SZR
Ø7731 ØØØ411               JMP      IGNOR    ; AN IGNORE BLOCK
Ø7732 1Ø1ØØ4               MOV      Ø,Ø,SZR  ; TEST THE CHECK SUM
Ø7733 ØØØ773               JMP      CHKER    ; ERROR
Ø7734 Ø3Ø442               LDA      2,ADDRS  ; GET THE ADDRESS
Ø7735 Ø62677               IORST             ; DO A RESET
Ø7736 151113               MOVL#    2,2,SNC  ; TEST BIT Ø
Ø7737 ØØ1ØØØ               JMP      Ø,2      ; Ø-START THE PROGRAM
Ø774Ø Ø63Ø77               HALT              ; Ø, HALT
Ø7741 ØØØ777               JMP      .-1
                  ;IGNORE ERROR MESSAGES BY READING UNTIL
                  ; A RUBOUT

Ø7742 ØØ47Ø4      IGNOR:   JSR      GTCHR    ; GET INTO AC3
Ø7743 Ø2Ø4Ø4               LDA      Ø,C377
Ø7744 1164Ø4               SUB      Ø,3,SZR
Ø7745 ØØØ775               JMP      IGNOR
Ø7746 ØØØ721               JMP      BLOCK    ; OK, GO INTO BLOCK MODE
Ø7747 ØØØ377      C377:    377
Ø775Ø ØØØØØØ      TEMP1:   Ø
Ø7751 ØØØØØØ      TEMP2:   Ø
Ø7752 ØØØØØØ      SAVE:    Ø
Ø7753 ØØØØØØ      COUNT:   Ø
Ø7754 ØØØØ2Ø      C2Ø:     2Ø                ; REPEAT BLOCKS HAVE WD > 2Ø(OCTAL

      ØØ7776               .LOC     .+21     ; SKIP BOTTSTRAP (OLD NOVA)
Ø7776 ØØØØØØ      ADDRS:   Ø
Ø7777 ØØØ663      BEND:    JMP      START

                           .END
```

```
ADDRS    007776
BEG      001001
BEND     007777
BLOCK    007667
BUILD    007636
C20      007754
C377     007747
C4K      001022
CHKER    007726
COUNT    007753
DIFF     007645
END      001023
GET      000030
GTCHR    007646
IGNOR    007742
LOOP     001003
SAVE     007752
START    007662
STORE    007717
TEMP1    007750
TEMP2    007751
TEST     007730
```