Donald Knuth’s EASY assembler for the ElectroData/Burroughs Datatron 205. This card-based assembler, along with its magnetic tape-based successor, MEASY, were used by Knuth to develop his Algol-58 compiler for the 205. Both were written during the summer of 1960. This source code was transcribed from a PDF of a scanned listing, available from:

http://archive.computerhistory.org/resources/text/Knuth_Don_X4100/PDF_index/k-2-pdf/k-2-c1037-EASY.pdf

another scanned listing, with forward references resolved, but without the symbol table and initialization code, is available from the same site:

http://archive.computerhistory.org/resources/text/Knuth_Don_X4100/PDF_index/k-2-pdf/k-2-u2435-EASY-doc.pdf

The first part of the source code below is the main line of the assembler that reads a card and begins parsing it. Entry to the program is near the end of the listing at address 1200. This assembly assumes the 205’s memory has been cleared to zeroes before loading the program.

Paul Kimpel
retro-205 Emulator Project
5 April 2015

1			1000	START		EASY ASSEMBLER.	1000	Start assembled code at address 1000
1	PROC	1	6005	CRD		READ INPUT CARD	1000 0 0010 44 6005	Read next card into high-speed loop address 6005. The Cardatron format band will parse the columns of the card into locations 6005-6018.
1			1600	BF6		MAKE TWO COPIES	1001 0 0000 26 1600	Block copy the contents of the 6000 loop to addresses 1600-1619 to duplicate the card buffer. The copy in the 6000 loop is where the assembled code will be placed. The duplicate in 1600 serves to retain the original values from the card during assembly
1			6012	CAD		IS THE OP CODE	1002 0 0000 64 6012	Load the symbolic op field to the A register
1				CNZ	OP	SYMBOLIC.	1003 0 0000 04 1024	If the symbolic op is non-zero (i.e., not blank), branch to label OP (address 1024)
1	ADDR		6011	CAD		IS THE ADDRESS	1004 0 0000 64 6011	Load the symbolic address field to A
1				CNZ	SYMBL	SYMBOLIC.	1005 0 0000 04 1048	If the symbolic address is non-zero (non-blank), branch to label SYMBL (1048)
1			6009	CAD		IS THE ADDRESS	1006 0 0000 64 6009	Load the second character of the program-point address (B/F/+/-)
1				CNZ	NUMAD	N+ N- NF OR NB.	1007 0 0000 04 5006	If the PP address is non-zero (non-blank), branch to label NUMAD (5006)
1	FLAG			SRT			1008 0 0000 13 0000	Shift A right zero places. This is a no-op, but this instruction will be replaced from the BUN 1037 at 1046 by the code at 1035-1035 during the processing of a START or END instruction, then restored from the SRT at 1045 by the code at 1037-1038. Effectively this causes the assembler to bypass checking the symbolic address field and program-point location field, along with the code that builds the assembled instruction at 1013-1023.
1	LOCAT		6017	CAD		IS THE LOCATION	1009 0 0000 64 6017	Load the symbolic location field
1				CNZ	SLOC	SYMBOLIC.	1010 0 0000 04 1056	If the symbolic location field is non-zero (non-blank), branch to label SLOC (1056)
1			6018	CAD		IS THE LOCATION	1011 0 0000 64 6018	Load the program-point location field
1				CNZ	NLOC	A PROGRAM POINT	1012 0 0000 04 1070	If the PP location field is non-zero (non-blank), branch to label NLOC (1070)
1	FINIS		1613	CAA			1013 0 0000 66 1613	Load absolute value of the assembled numeric address field
1			4	SRT		SYNTHESIZE	1014 0 0000 13 0004	Shift the four address digits into R
1			1608	CAD		INSTRUCTION.	1015 0 0000 64 1608	Load the high-speed loop tag field into A
1				CNZ	1F	CHECK HSL TAG	1016 0 0000 04 1020	If the HS loop tag is non-zero (non-blank), branch forward to PP label 1
1	2		1614	CAD		TACK ON	1017 0 0000 64 1614	If the HS loop tag is zero/blank, replace it in A with the numeric op code
1			2	SRT		OPERATION CODE.	1018 0 0000 13 0002	Shift the two op code digits into R with the address digits
1				BUN	HSLF		1019 0 0000 20 7010	Branch to label HSLF (7010) to finish assembling the instruction
1	1		1	SLT		INSERT	1020 0 0000 14 0001	To here if the HS loop tag field is non-blank: shift the high-order address digit from R to

							A
1		6008	CAD		HSL TAG DIGIT	1021 0 0000 64 6008	Load the HS loop tag field into A, replacing the original high-order address digit
1		1	SRT			1022 0 0000 13 0001	Shift the high-order address digit back into R
1			BUN	2B		1023 0 0000 20 1017	Branch back to PP label 2 to tack on the op code and finish assembling the instruction
1	OP		ADD	SEVEN	LOOK UP	1024 0 0000 74 5008	Handle a symbolic op code: add 7 to the alpha op code in A
1			CUR	TABLE	SYMBOLIC OPCODE	1025 0 0000 21 4000	Call the TABLE subroutine (CUR stores the return address in top four digits of R)
1		8421	HLT		HALT IF	1026 0 0000 08 8421	Return to here if the op code is not in the symbol table: HALT 8421
1			BUN	PROC	UNDEFINED.	1027 0 0000 20 1000	When restarted by the operator, abort this card and branch to read another
1		1614	STA		BUT IF DEFINED	1028 0 0000 12 1614	Return from TABLE to here if the op code was defined: store table result (the numeric op code corresponding to the symbolic op code) in the numeric op code field
1		2	SRT		CHECK FOR	1029 0 0000 13 0002	Shift the op code digits from the TABLE result into R
1		1614	CNZ		PSUDO OPERATION	1030 0 0000 04 1614	If the high-order 8 digits of the TABLE result are non-zero, branch to 1614, which holds the word returned by TABLE. The result will be non-zero only for START and END pseudo-ops, and their table words contain branches to the labels START (1034) and END (1032), respectively. See locations 2464/3464 (for START) and 2599/3599 (for END)
1			BUN	ADDR		1031 0 0000 20 1004	If the H.O. 8 digits of the TABLE result are zero (normal case), branch back to label ADDR (1004) to resume parsing fields from the card
1	END		CAD	3F	SET UP	1032 0 0000 64 1046	To here if an END pseudo-op is encountered: load the CUB PLOAD instruction word at 1046
1			STA	PROC	TO FINISH OFF.	1033 0 0000 12 1000	Store the CUB PLOAD word at 1000, overwriting the CDR 6005 instruction that was there. This will cause the the final phase of the assembler to be entered after the current instruction is finished and the logic branches back to label PROC, intending to read the next (non-existent) card. The CUB will block-transfer 20 words from 1708-1727 to the 7000 HS loop (wrapping around so that the word at 1720 goes into 7000) and then branch to 7008 in the loop
1	START		CAD	2F	CALCULATE	1034 0 0000 64 1047	To here if a START pseudo-op is encountered and fall through to common code for an END pseudo-op: load the BUN 5B word from 1047
1			STC	FLAG	ADDRESS	1035 0 0000 02 1008	Overwrite the SRT at 1008 with the BUN 5B. This will cause special processing of the instruction being assembled, as it’s a pseudo-op
1			BUN	ADDR	EQUIVALENT.	1036 0 0000 20 1004	Branch back to ADDR to begin parsing the address fields from the card
1	5		CAD	1F		1037 0 0000 64 1045	To here when control passes through address 1008: the BUN that replaced the SRT at that address for special pseudo-op handling will branch to here: load the SRT at 1045 in preparation to restore the instruction at 1008:
1			STA	FLAG		1038 0 0000 12 1008	Store the SRT at 1008, overwriting the BUN that was placed there for pseudo-op processing, and restoring the original no-op.
1		1613	CAD		CHANGE LOCATION	1039 0 0000 64 1613	Load the assembled effective address for the instruction
1		6002	STA		COUNTER	1040 0 0000 12 6002	Store in the location counter word to make this the address for the next word (for START) or the addressing of the loading routine (for END)
1		6003	ADD		AND INSERT	1041 0 0000 74 6003	Apply the skeleton CRD 0810 44 0000 at 6003 to the new location counter value
1		6000	STA		TRANSFER	1042 0 0000 12 6000	Store the CRD instruction at 6000. This will form the last word on the output card, which will cause the next-assembled instruction to load at the new address
1		6004	CAD		INSTRUCTION.	1043 0 0000 64 6004	Load a literal 6
1			BUN	PUNCH		1044 0 0000 20 7017	Branch to the output routine to set the 6-sign and punch the output card
1	1		SRT			1045 0 0000 13 0000	SRT instruction word used to restore location 1008. Not executed at this location
1	3		CUB	PLOAD		1046 0 0000 30 1708	CUB instruction word used to overwrite location 1000 when and END pseudo-op is encountered. Not executed at this location
1	2		BUN	5B		1047 0 0000 20 1037	BUN instruction word used to overwrite location 1008 when a START or END pseudo-op is encountered. Not executed at this location
1	SYMBL		CUR	TABLE	LOOK SYMBOL UP.	1048 0 0000 21 4000	To here from the main line if there is a symbolic address on the card. Call (change-and-

								record) the TABLE subroutine. Symbol to look up is in A. Return address is in R1-4
1		1	1900	STC		IF UNDEFINED.	1049 1 0000 02 1900	TABLE returns here if symbol is undefined with the offset of the next available entry in B. Store the symbol word at 1900+B
1				BUN	LOOP7	DEFINE IT.	1050 0 0000 20 7000	Branch to LOOP7 to continue defining the symbol
1				OSD	7+	IF DEFINED AND	1051 0 0000 73 1058	TABLE returns here if the symbol is defined, with the offset of the entry in B and the symbol value in A. The symbol value will be negative if it's been referenced before being defined, so compare its sign to that of the word at the control address+7 (the BUN at 1058, which is a positive word). If the signs are different (the symbol value is negative), overflow will be set
1				BOF	LOOP7	FORWARD GO TO	1052 0 0000 28 7000	If the sign of the symbol value was negative, branch to LOOP7 to finish defining the forward reference
1	EQ		6013	ADD		SPECIAL ROUTINE	1053 0 0000 74 6013	Otherwise, if the symbol was previously defined and not forward, add the numeric address from the card to the symbol value in A to get the effective address. Also enters here from program-point address operands at 1810-1844
1			1613	STA		OTHERWISE SET	1054 0 0000 12 1613	Store the updated effective address back into the numeric address field for the instruction
1				BUN	FLAG	EQUIVALENT.	1055 0 0000 20 1008	Branch back to FLAG to continue parsing the address fields from the card (note that the instruction at FLAG is modified)
1	SLOC			CUR	TABLE	LOOK FOR SYMBOL	1056 0 0000 21 4000	To here from the main line if there is a symbol location on the card. Call the TABLE subroutine. Symbol to look up is in A. Return address is in R1-4
1		1	1900	STC			1057 1 0000 02 1900	TABLE returns here if the symbol is undefined with the offset of the next available entry in B. Store the symbol word at 1900+B
1				BUN	A3	UNDEFINED LOC.	1058 0 0000 20 1064	Branch to A3 to continue defining the symbol.
1				OSD	7-		1059 0 0000 73 1052	TABLE returns here if the symbol is defined, with the offset of the entry in B and the symbol value in A. The value will be negative if it's been referenced before being defined, so compare its sign to that of the word at the control address-7 (the BOF at 1052, which is a positive word). If the signs are different (the symbol value is negative), overflow will be set.
1				BOF	A3	DEFINED LOC.	1060 0 0000 28 1064	If the sign of the symbol value was negative, branch to A3 to finish defining the forward reference.
1			6017	CAD		HALT IF SYMBL	1061 0 0000 64 6017	Otherwise, the symbol has been previously defined. Load the symbol from the symbolic location field on the card to A
1			9669	HLT		OCCURS TWICE	1062 0 0000 08 9669	Halt the processor with the duplicate symbol definition in A
1				BUN	PROC	IN LOCATION	1063 0 0000 20 1000	If the operator continued after the hald, branch to PROC to read the next card
1	A3			STA	TEMP		1064 0 0000 12 4019	To here to finish defining the symbol for a symbolic instruction location. Store the symbol's currently-defined value (which will be zero for new or negative for a forward reference) in TEMP at 4019
1			6002	CAD		IF FORWARD	1065 0 0000 64 6002	Load the current location counter to A
1		1	2900	STA		REFERENCE WAS	1066 1 0000 12 2900	Store the current location counter as the new or forward-referenced symbol's value at 2900+B
1	A2			CAD	TEMP	MADE, MAKE	1067 0 0000 64 4019	Reload the symbol's former value from TEMP
1				CNZ	A1	NEW TABLE ENTRY	1068 0 0000 04 5011	If the former value is not zero (i.e., it's negative for a forward reference), branch to A1 to deal with the forward-reference chain
1				BUN	FINIS		1069 0 0000 20 1013	Otherwise, we're done with the symbolic location – branch to FINIS to put the assembled instruction together and output it
1	NLOC		6018	LDB		PROGRAM POINT	1070 0 0000 72 6018	To here from the main line if there is a program-point location on the card. Load B with the one-digit PP numeric value from the card
1		1	3950	CAD		LOCATION.	1071 1 0000 64 3950	Load A from 3950+B: this is the pseudo-symbol value for the PP "forward" location identified by the value in B, which is the last address defined for that PP
1				STA	TEMP	UNDEFINE	1072 0 0000 12 4019	Store the current PP "forward" location value in TEMP
1			6002	CAD		FORWARD P.P.	1073 0 0000 64 6002	Load A with the current location counter

1		1	3900	STC		DEFINE BACKWARD	1074 1 0000 02 3900	Store and clear A at 3900+B – this stores the new “backward” location for the PP
1		1	3950	STA			1075 1 0000 12 3950	Store the zeroed A at 3950+B – this stores the new “forward” location for the PP (i.e., there isn’t one yet)
1				BUN	A2		1076 0 0000 20 1067	Branch to A2 with the PP’s former “forward” location value in TEMP to handle the forward-reference chain
1	HALT			CAD	TRANS	PUNCH FINAL	1077 0 0000 64 1723	To here from the end of the loading routine generator – load the final branch instruction that will be executed from the last card in the load deck
1			7006	STA		CARD AND	1078 0 0000 12 7006	Store that final instruction in the last word of the last card
1		21	7000	CWR		STOP	1079 0 0210 54 7000	Punch the final card to unit 1 using format band 2
1			1111	HLT		UNCONDITIONALLY	1080 0 0000 08 1111	Halt. Assembly is finished
1				BUN	1-		1081 0 0000 20 1080	I said Halt and I meant Halt
1			1810	START			1810	Start of the routine to handle program-point address fields on the card (n+, n-, nF, nB). “n+” enters here from the BUN at NUMAD+1 (5007)
1			6002	CAD		N+	1810 0 0000 64 6002	Load the location counter
1			6010	ADD			1811 0 0000 74 6010	Add offset from the PP operand
1				BUN	EQ		1812 0 0000 20 1053	Branch to EQ to compute effective address
1			1820	START			1820	“n-“ enters here
1			6002	CAD		N-	1820 0 0000 64 6002	Load the location counter
1			6010	SUB			1821 0 0000 75 6010	Subtract offset from the PP operand
1				BUN	EQ		1822 0 0000 20 1053	Branch to EQ to compute effective address
1			1842	START			1842	“nB” enters here
1			6010	LDB		NB	1842 0 0000 72 6010	Load B with the PP number from the PP operand
1		1	3900	CAD			1843 1 0000 64 3900	Load A from 3900+B, which is the “backward” location for the PP
1				BUN	EQ		1844 0 0000 20 1053	Branch to EQ to compute effective address
1			1846	START			1846	“nF” enters here
1			6010	CAD		NF	1846 0 0000 64 6010	Load A with the PP number from the PP operand
1				ADD	FUDGE	PROCESS	1847 0 0000 74 5019	Add literal 1050 to the PP number in A
1				STA	TEMP	LIKE	1848 0 0000 12 4019	Store 1050+PP in TEMP for the LDB to use next
1				LDB	TEMP	SYMBOL	1849 0 0000 72 4019	Load B with the value 1050+PP
1		1	2900	CAD			1850 1 0000 64 2900	Load A with value at 2900+B (effectively the word at 3950+PP), which is the PP’s “forward” location value
1				BUN	LOOP7		1851 0 0000 20 7000	Branch to LOOP7 to handle the forward-reference chain for the PP
1			1708	START			1708	Start of loading routine generator. Note that the following 20 words are loaded to the 7000 loop by the CUB PLOAD at 1046, which actually overwrites and is executed from location PLOAD (1000) when the END card is encountered. The first step is to copy the code for the loading routine from 1761-1790 to cards
1	PLOAD	1	6002	CAD		PUNCH LOADING	1708 0 0001 64 6002	Load the current location counter (which would have been set by the operand address on the END card)
1			7000	ADD		ROUTINE.	1709 0 0000 74 7000	Add the word at 1720 (remember, we are executing this code out of the 7000 loop, so 1720 is now at location 7000). That is a skeleton LDB, so we end up with an LDB referencing the address on the END card
1			7000	STA		FIRST CARD	1710 0 0000 12 7000	Store the updated LDB back to location 7000
1			7002	EXT		LOADS B BOX	1711 0 0000 63 7002	Mask the LDB with the pattern 1111201111 from location 1722 (now 7002), which will preserve all of the numeric digits except those for the op code field. The high-order digit of the op code will be set to 2 and the low-order digit will be set to zero, so this converts the LDB to a BUN (20) instruction
1			7002	STA			1712 0 0000 12 7002	Store the generated BUN instruction back to location 7002
1			7006	CSU			1713 0 0000 65 7006	Load the negative value of the word at 1726 (now 7006), which is a CDR (44) for input unit 1 to address 0000. The negative sign will cause this word to be B-modified when it is executed

1 7001 STA	1714 0 0000 12 7001	Store the CDR at location 7001
1 21 7000 CWR	1715 0 0210 54 7000	Punch the first card of the loading routine from address 7000 to output unit 1 using format band 2. This card, when loaded, will put the following instructions into memory at the address specified on the END card: • LDB AAAA, where AAAA is the address from the END card • CDR 0000+B, which will read the next card to address AAAA • BUN AAAA, which will branch to the first word read from that next card
1 CAD 3F 7	1716 0 0000 64 7724	Load A with literal 23 from location 1724 (now 7004)
1 ADD PLACE	1717 0 0000 74 5014	Add the word at 5014, which is a STA 0009 instruction. This generates a STA 0032 instruction. That 32 is the offset to the start of the table of forward references that will be generated after the loading routine is output
1 4 SRT	1718 0 0000 13 0004	Shift the generated instruction right 4 digits, pushing the address field into the high-order digits of R
1 BUN 1F	1719 0 0000 20 1728	Branch to the next phase of the loading routine generator at 1728
1 LDB	1720 0 0000 72 0000	Skeleton LDB instruction used at 1709 above
1 4 0000 LDB	1721 0 4000 72 0000	Location for the generated CDR stored at 1714 above
1 SIGN6 11112 1111 SGN CHGED TO 6	1722 0 1111 20 1111	EXT mask used at 1711 above. Note that this word will have a sign of 6 applied to it during the initialization code at 1207-1212
1 TRANS 15 BUN	1723 0 0000 20 0015	Instruction used at end of loading routine generator at 1077
1 3 23	1724 0 0000 00 0023	Literal 23 used at 1716 above
1	1725 0 0000 00 0000	(word overwritten by loading routine generator at 1748 executing from 4008)
1 1 0000 CRD	1726 0 0010 44 0000	Skeleton CRD instruction used at 1713 above
1 7667 0005	1727 0 7667 00 0005	Literal: low-order four digits used by LDB at 1742 (executing from 4001)
1 1 CAD TOFIX	1728 0 0000 64 1776	Next phase of the loading routine generator: enter with end address of forward reference table in high-order digits of R (R1-4); load literal 64 from location TOFIX (1776), which is the op code for CAD
1 4 SLT	1729 0 0000 14 0004	Shift the end address of forward reference table left from R to form a complete CAD instruction word
1 STA TOFIX	1730 0 0000 12 1776	Store the constructed CAD instruction back to location TOFIX
1 1740 BT4	1731 0 0000 34 1740	Block words 1740-1759 to the 4000 loop
1 6004 CAD	1732 0 0000 64 6004	Load a literal 6 from location 6004
1 BUN 4F 4	1733 0 0000 20 4756	Branch to location 4756, which is congruent with 4000-loop address 4016, which is the word just blocked there from address 1756
1 1740 START	1740	What follows is the main loop to copy the loading routine instruction words to six punched cards
1 3 10 SRT LOOP TO	1740 0 0000 13 0010	Branch back to here from 1757 (now 4017) with number of cards left to punch in A: shift the number of cards left to the R register
1 7007 LDB PUNCH THE	1741 0 0000 72 7007	Load B from the lower four digits of the word at 7007 (a literal 5)
1 CAD 1F 4 LOADING ROUTINE	1742 0 0000 64 4745	Load A with the CAD 1755+B instruction word from 1745 (now 4005)
1 6004 SUB	1743 0 0000 75 6004	Subtract literal 6 in location 6004
1 STA 1F 4	1744 0 0000 12 4745	Store the updated CAD back in the next word at 1745 (now 4005)
1 1 1 1755 CAD	1745 1 0000 64 1755	Load A with the word at 1755+B, but note that the base address is increased by six every time through the loop (since this word is negative to enable B modification, subtracting 6 at 1743 (now 4003) actually increases the base address). Therefore the address is 1755 + 6*n + B, where n=1..6 for the six cards to be punched
1 1 7000 STA	1746 1 0000 12 7000	Store that word at 7000+B (7000-7007 is the buffer from which the card will be punched, but note that the Cardatron places the words backward on the card, so the word at 7000 will appear in columns 70-80 on the card)
1 DBB 1B 4	1747 0 0000 22 4745	If B > 0, decrement B and branch back to 4745=4005 (was 1745) to load the next word in sequence and store it into the buffer
1 7005 CAD	1748 0 0000 64 7005	Fall through to here after six words of the loading routine have been moved to the card buffer at 7000-7005: load the last word moved

`1              CNZ     2+ 4                  1749 0 0000 04 4751`	If that last word is not zero skip the next instruction and branch to 4751=4011 (was 1751)
`1              BUN  HALT                     1750 0 0000 20 1077`	If the last word is zero, branch to HALT at 1077 to punch the last card and halt
`1         7006 CAD                          1751 0 0000 64 7006`	Load the word at 7006, a skeleton CRD (44) for unit 1, and part of the card buffer
`1         6004 ADD                          1752 0 0000 74 6004`	Add a literal 6 from address 6004 to update the address in the CRD instruction
`1         7006 STA                          1753 0 0000 12 7006`	Store the CRD instruction back to address 7006
`1      21 7000 CWR                          1754 0 0210 54 7000`	Write the buffer at 7000-7007 to Cardatron output unit 1 using format 1. Note that the high order digits at 7007 are 766 – the format band wll cause these digits will appear in columns 1-3 of the card
`1           10 SLT        AFTER FIRST       1755 0 0000 14 0010`	Shift back the number of cards left to punch from the R register
`1  4           SUB  ONE   SIX CARDS,        1756 0 0000 75 5005`	Decrement the number of loading routine cards left to punch (enter here initially from 1733 with a 6 in the A register)
`1              CNZ     3B 4 DUMP FORWARD     1757 0 0000 04 4740`	If A = 0, fall through to begin punching the forward reference table, otherwise branch back to 4740=4000 (was originally at 1740) to punch the next card
`1              CAD     8F   REFERENCE TABLE  1758 0 0000 64 1760`	Start of routine to dump the forward reference table: Load A from the skeleton CAD 0000+B instruction word from 1760
`1              STA     1B 4                  1759 0 0000 12 4745`	1. Store the CAD 0000+B instruction word at 4745=4005 (was 1745). This resets the base address from which words will be copied to the output card deck so that the table of forward references will follow the loading routine that has just been punched to cards. 2. Since this routine is running out of the 4000 loop, this instruction is at address 4759=4019. The next instruction will be taken from 4760=4000, so the program implicitly loops in the high-speed loop back to the instruction originally at 1740. Tricky…
`1  8     1 0000 CAD                          1760 1 0000 64 0000`	Skeleton CAD 0000+B instruction word
`1         1761 START                                            1761`	Start of the loading routine that is copied to cards at the end of the assembly by the code above. Note that the following code is not executed at assembly time, but when the assembled deck is loaded into the 205 at run time. • The first card of the loading routine (created by the code at 1708-1715 running out of the 7000 loop) is loaded at the effective address on the END card. The code coming from the card is this (where “aaaa” is the load address from the END card): 0 0000 72 aaaa – LDB aaaa 1 0010 44 0000 – CRD 0000+B 6 0000 20 aaaa – BUN aaaa • Since the third word has a sign of 6, it does not get loaded to memory, but instead terminates the read and branches to the load address, where the first word has already been stored. • The first word, when executed, loads B from the low-order four digits of the word at the load address, but since this first word is at the load address, it loads B from itself, and thus B is loaded with the load address that was on the END card. • The second word, when executed, reads the next card from unit 1 to address 0000+B. But B now has the load address, so that read will overwrite the two words just loaded and executed. This bootstraps the loading routine into memory. Subsequent cards end with a sign-7 word with a CAD instruction to continue loading the routine, and the last card ends with a sign-7 word with a BUN instruction that will terminate the loading process and branch to aaaa+15, which is equivalent to the word at label TOFIX (1776) below. • Note that the word at TOFIX was updated by the code at 1728-1730 to be a CAD xxxx, where xxxx is the offset to the end of the table of forward references generated by the assembler and punched to the output deck after the loading routine. • Also note that many of the words in the loading routine below have signs of 2, which will cause them to be adjusted by the value of the B register and the sign changed to 0 before being stored in memory. Since these words are being read at deck load time with the value of aaaa (the load address from the END card) in B, that relocates their addresses to the area of memory where the loading routine has itself been loaded. These addresses will be noted below as “+aaaa”.

						• Note that execution of the following code begins at 15+aaaa (@1776)
1		1	0001 CAD	LOADING ROUTINE	1761 1 0000 64 0001	Load the word at 1+B to the A register. This loads the next word in the chain of instructions requiring adjustment to a certain forward address. Note that B was decremented at 10+aaaa (@1771) before branching here, so 1+B effectively reverses that decrement.
1		2	0030 STA	CLEANS UP	1762 2 0000 12 0030	Store the instruction to be adjusted at 30+aaaa (@1791). This is done to save the chain-link address in the low-order for digits of the instruction word
1		2	0029 EXT	FORWARD	1763 2 0000 63 0029	Extract using the word at 29+aaaa (@1790 = 01111110000) which will clear the sign and address digits in the A register, leaving the rest of the digits intact
1		2	0031 SUB	REFERENCES.	1764 2 0000 75 0031	Subtract word at 31+aaaa. This subtracts the new address from zero in the address field of the instruction. Note, however, that the word being subtracted is negative, so this adds the new address to zero, effectively inserting the new address into the address field of the instruction
1			9 SRT		1765 0 0000 13 0009	Shift A right by 9 digits into R without shifting the sign. This leaves the high-order digit of the instruction word in A, right-justified over zeroes.
1		2	0021 CNZ		1766 2 0000 04 0021	If the high-order digit of the word is non-zero, branch to 21+aaaa (@1782) to handle address adjustment for the high-speed loops; otherwise fall through and continue
1			1 SRT		1767 0 0000 13 0001	The high-order digit is zero: shift it to R with the rest of the original word from A
1		2	0030 CAD		1768 2 0000 64 0030	Load the word at 30+aaaa. This is a copy of the original instruction word to be adjusted
1			10 SLT		1769 0 0000 14 0010	Shift 10 digits left from R into A, leaving only the sign digit from the original instruction word at 30+aaaa. Effectively, the last two instructions have preserved the sign of the original instruction word
1		1	0001 STC		1770 1 0000 02 0001	Store that word back to its location at 1+B
1		2	0030 LDB		1771 2 0000 72 0030	Load B from the four low-order digits at 30+aaaa. For the first word of a chain to be adjusted, this is the current entry word from the table of forward references; the low-order four digits are the address of the first instruction word in the chain that needs its address field adjusted. For subsequent words in the chain, it is a copy of the last word that needed to be adjusted, which has the chain-link address in its low-order four digits. Thus, either we load the address of the first word in the chain or the next word in the chain
1		2	0000 DBB		1772 2 0000 22 0000	If B=0, we have reached the end of this chain of forward references, so fall through and continue; otherwise, decrement B and branch to 0+aaaa (@1761), which will iterate through the chain of instruction words being adjusted to the same forward address
1		2	0015 CAD		1773 2 0000 64 0015	Load the word at 15+aaaa (@1776, label TOFIX) This is a CAD with the address of the current entry in the table of forward references
1		2	0028 SUB		1774 2 0000 75 0028	Subtract 1 from the address of the current entry (28+aaaa = @1789)
1		2	0015 STA		1775 2 0000 12 0015	Store the CAD with the decremented address back to 15+aaaa (TOFIX), the next word
1	TOFIX	2	0064		1776 2 0000 00 0064	This word is now a CAD xxxx, where xxxx is the address of the current entry word in the table of forward references. This loads the current entry to A. The low-order four digits in each entry are the address of the first word in a chain of words to be adjusted to the same address. The next higher four digits in the entry word are the address to which each word in the chain should be adjusted. Note that the sign digit in all of the entry words is a 1, *so the values are all negative*, and the sign is preserved during SRT and SLT operations.
1		2	0030 STA		1777 2 0000 12 0030	Store this table entry word at 30+aaaa
1			4 SRT		1778 0 0000 13 0004	Shift the four address digits of the first word in the chain right into the R register
1		2	0031 STA		1779 2 0000 12 0031	Store the remaining digits (the address to be applied to each word in the chain) at 31+aaaa. Note that *this word is negative*, and that all digits except the low-order four are zero
1		2	0010 CNZ		1780 2 0000 04 0010	If the address to be applied is zero, we are at the end of the table of forward references, so fall through and halt. This will occur when the address in the CAD at TOFIX is decremented to 32+aaaa, a stopper word of all zeroes. If the address to be applied is not zero, branch to 10+aaaa (@1771) to begin adjusting this chain of words with their new addresses

1 1221 HLT	1781 0 0000 08 1221	Halt 1221. The program is now loaded and ready to run.
1 2 0032 STC	1782 2 0000 02 0032	To here if the high-order digit in the original instruction word is non-zero: HS loop address adjustment. Store the loop digit in 32+aaaa and clear the A register
1 6 SLT	1783 0 0000 14 0006	Shift R left into A for six digits. R has the low-order 9 digits of the original instruction word, so this positions the high-order digit of the address field in the low-order digit of A
1 2 0027 EXT	1784 2 0000 63 0027	Extract with the pattern at 27+aaaa (=1111111110). This will clear the low-order digit in A
1 2 0032 ADD	1785 2 0000 74 0032	Add the HS loop digit at 32+aaaa to A, effectively setting that digit as the high-order digit of the instruction word address field
1 7 SRT	1786 0 0000 13 0007	Shift 7 digits right into R. The entire 10 digits of the loop-adjusted instruction word are now in R
1 2 0007 BUN	1787 2 0000 20 0007	Branch back to 7+aaaa (@1768) to continue updating the instruction word in memory
1 1111111110	1788 0 1111 11 1110	Extract pattern used to clear the low-order digit in A
1 1	1789 0 0000 00 0001	Literal 1
1 1111110000	1790 0 1111 11 0000	Extract pattern used to clear the address field in A
1 4000 START	4000	Symbol table lookup routine. Enter with the symbol to be looked up (left-justified over zeroes with 7 added to the value). Each symbol table entry consists of two words. The first word is stored in the address range 1900-2899 and contains the symbol in the format just described for the parameter above. The second word, stored at the symbol's address plus 1000 in the range 2900-3899, contains the symbol value (typically either an op code or an address). Symbols are hashed as discussed below to determine an initial probe point in the table. Hash collisions are handled by sequentially searching backwards, wrapping around to the top of the table as necessary. If the symbol is not found in the table, returns to the location after the call. If the symbol is found, returns to the location after the call plus two words.
1 TABLE STC HOLD SYMBOL TABLE	4000 0 0000 02 5010	Store the parameter symbol and clear the A register
1 CAD 4F SEARCH ROUTINE.	4001 0 0000 64 7012	Load the instruction word at 7012 (STA 6020)
1 4 SLT	4002 0 0000 14 0004	Shift A and R left four digits. This shifts the return address from R into the low-order four digits of A and changes the instruction from STA (12) to BUN (20), which will branch to the return address
1 STA EXIT STORE EXIT.	4003 0 0000 12 4013	Store the crafted BUN instruction word at EXIT for later use
1 CAD HOLD GENERATE	4004 0 0000 64 5010	Reload the parameter symbol to A
1 MUL 10101 STARTING	4005 0 0000 60 4014	Multiply the symbol by 1001001001
1 STC TEMP PLACE, AND	4006 0 0000 02 4019	Store the A register in TEMP and clear the A register (the purpose is only to clear the A register)
1 3 SLT SEARCH FOR	4007 0 0000 14 0003	Shift the high-order three digits of the low-order word of the product from R into A. This is the hashed value of the symbol
1 STA TEMP EMPTY OR	4008 0 0000 12 4019	Store the hashed value in TEMP so it can be used by the LDB, next
1 LDB TEMP UNUSED PLACE	4009 0 0000 72 4019	Load B with the hashed value from the low-order four digits in TEMP
1 1 1 1900 CAD	4010 1 0000 64 1900	Load the word in the symbol table at 1900+B. This is a candidate symbol for matching to the parameter symbol
1 CNZ 2F	4011 0 0000 04 5000	If the word loaded from the symbol table is zero, the symbol is not in the table, so fall through in preparation to exit
1 CAD HOLD NOT IN TABLE	4012 0 0000 64 5010	Since the symbol was not found, reload the original parameter value to A
1 EXIT	4013 0 0000 00 0000	The crafted BUN instruction from 4003 was stored here. Branch back to the word after the call with the original symbol in A and the offset to the empty symbol table entry in B. This is a location where the new symbol can be stored
1 10101 1 1 1001	4014 0 1001 00 1001	Literal 1001001001 used to hash the symbol
1 3 CAD EXIT IF SYMBOL IN	4015 0 0000 64 4013	To here from 5001 if a matching symbol table entry was found. Load the crafted BUN instruction word with the return address from 4013
1 ADD TWO TABLE, ADD 2	4016 0 0000 74 5009	Add 2 to the return address
1 STA TEMP TO EXIT LINE	4017 0 0000 12 4019	Store the BUN instruction with the updated return address at TEMP, where we will run

	into in after the next instruction
`1           1      2900 CAD                AND DISPLAY         4018 1 0000 64 2900`	Load the corresponding value word for the symbol from 2900+B
`1     TEMP                                 EQUIVALENT          4019 0 0000 00 0000`	This location is used both as temporary storage and as the location for a BUN instruction with the return address, as crafted at 4015-4017. If the symbol was found, control will fall through to this location and exit the routine.
`1                  5000 START                                                5000`	Since the prior instruction was at the end of the 4000 loop, reset the location counter to the start of the 5000 loop
`1   2                   SUB   HOLD                              5000 0 0000 75 5010`	To here from 4011 if the current probe into the symbol table found a non-zero word. Subtract the parameter symbol from the from the symbol found in the table
`1                       NOR         3B                          5001 0 0000 15 4015`	Normalize the result of the subtraction. If the A register is zero (indicating the parameter matches this table entry), branch to 4015. This will also shift R into A and clear R, but that result is not used. If A is not zero, fall through to the next instruction. This will also shift R left into A until the high-order digit of A is non-zero, but that result is not used.
`1                       DBB         1B                          5002 0 0000 22 4010`	If B>0, decrement B and branch to 4010. This will step to the next entry in the symbol table. If B=0, we have reached the beginning of the table, so fall through to reposition to the top of the table and continue the search
`1                       LDB   999        CYCLE MOD           5003 0 0000 72 6019`	Load B with the literal value 999 to address the last entry in the table
`1                       BUN         1B   1000.               5004 0 0000 20 4010`	Branch to 4010 to continue the search
`1     ONE                  1                                   5005 0 0000 00 0001`	Literal 1
`1     NUMAD        6009 LDB                                    5006 0 0000 72 6009`	To here from main line if there is a program-point address on the card (n+, n-, nF, nB). Load B with the two-digit alpha value of the second character of the field (+,-,F,B)
`1           1      1800 BUN                                    5007 1 0000 20 1800`	Branch to the appropriate handling routine at 1800+B. Note that “+” (or “&”)=10, “-“=20, “B”=42, “F”=46. Therefore, this will branch to 1810, 1820, 1842, or 1846. Slick.
`1     SEVEN                7                                   5008 0 0000 00 0007`	Literal 7
`1     TWO                  2                                   5009 0 0000 00 0002`	Literal 2
`1     HOLD                                                     5010 0 0000 00 0000`	Temp cell used by TABLE
`1     A1                   4 SRT               PREPARE         5011 0 0000 13 0004`	To here from 1068 to create a new forward-reference table entry. Shift the symbol’s value in A right four digits into R. This will be either zero for a newly-defined symbol, or a backlink to a previous instruction word with a forward reference for that symbol
`1                  6002 CSU                    FORWARD         5012 0 0000 65 6002`	Load the negative of the current location counter (this is the address of the instruction word that will eventually need to have its address adjusted)
`1                     4 SLT                    REFERENCE       5013 0 0000 14 0004`	Shift the symbol’s current value back into A. Note the sign does not shift
`1     PLACE           9 STA                    TABLE           5014 0 0000 02 0009`	Store the entry into the next slot in the forward-reference table. This is initially address 0009, but the address gets bumped for each entry in the following instructions
`1                       CAD   PLACE            ENTRY           5015 0 0000 64 5014`	Load the store instruction just above
`1                       ADD   ONE                              5016 0 0000 74 5005`	Increment the table-entry address
`1                       STC   PLACE                            5017 0 0000 02 5014`	Store the instruction with the incremented table-entry address back in PLACE
`1                       BUN   FINIS                            5018 0 0000 20 1013`	Branch back to the main line to finish assembling the instruction
`1     FUDGE        1050                                        5019 0 0000 00 1050`	Constant used at 1847
`1                  6003 START                                                6003`	The 6000 loop (addresses 6000-6019) holds much of the data for the instruction currently being assembled. The source card is read into 6005-6018, but this is not the text from the card – it’s the data after it has been formatted by the Cardatron format band defined at 1400-1428. Note that the fields in memory are in reverse order with respect to their positions on a card, as was the case with Cardatron I/O. The layout of the 6000 loop is as follows: • 6000 = the assembled word, less its sign digit this will be punched into columns 70-80 on the output card for this instruction • 6001 = the sign (in the low-order digit of the word) of the assembled word • 6002 = current value of the location counter (address of the instruction being

						assembled) • 6003 = skeleton CRD instruction (0 0810 44 xxxx) that will be punched into columns 59-69 on the output card for this instruction. At load time, this will cause the next card in the output deck to be read, and the address (xxxx) will specify the location into which the instruction on that card should be stored • 6004 = literal 6: this is applied by the Cardatron output band to the CRD instruction above and also used as a literal 6 elsewhere in the assembler (e.g., at 1732) • 6005 = third word of comment field (columns 52-56) • 6006 = second word of comment field (columns 47-51) • 6007 = first word of comment field (columns 42-46) • 6008 = high-speed loop tag from column 40 of the card • 6009 = program-point address character 2 from column 38 on the card (+, -, F, B) • 6010 = program-point address character 1 from column 37 (a digit) • 6011 = symbolic address field from columns 31-15 on the card • 6012 = symbol op code field from columns 25-29 on the card • 6013 = numeric address field from columns 20-23 on the card • 6014 = numeric op code field from columns 18-19 on the card • 6015 = numeric control digits from columns 14-17 on the card • 6016 = numeric sign from column 13 on the card • 6017 = symbolic location from columns 7-11 on the card • 6018 = program-point address from column 5 on the card • 6019 = literal 1600000999. The low-order four digits are used by the TABLE routine during search wraparound to reload the B register. The high-order 16 gets punched into columns 1-2 in the output card to be used as Cardatron input format band digits when the assembled deck is read back in. Selecting format from column 1 allows the deck to be used as the source input into another assembly run. Selecting format from column 2 allows the deck to be used as a one-word-per-card object-load deck. This is a very clever use of the Cardatron
1		81	0000 CRD	CONSTANTS FOR	6003 0 0810 44 0000	Skeleton CRD instruction as mentioned above
1			6	PUNCHOUT	6004 0 0000 00 0006	Literal 6 for the sign of the CRD instruction as mentioned above
1			6019 START		6019	
1	999	16	0999		6019 0 1600 00 0999	Literal 999 for the TABLE routine, as mentioned above
1			7000 START		7000	Routine for adding a location to the chain of forward references for an undefined symbol. Enter with either a zero in A (for the first occurrence of a symbol) or the negative address (from the symbol table entry) of the prior instruction referencing this as-yet undefined symbol. B is the offset into the symbol table for the current symbol
1	LOOP7		1613 STA	SET UP CHAINING	7000 0 0000 12 1613	Store the address of the head instruction in the chain for this symbol (or a zero) in the numeric address field of the duplicate card buffer at 1600-1619
1			6002 CSU	PROCEDURE	7001 0 0000 65 6002	Load the negative value of the current location counter (i.e., the address of the current instruction being assembled)
1		1	2900 STC	FOR FORWARD	7002 1 0000 02 2900	Store the negative address of the current instruction as the current value for the symbol. This is effectively the head of the chain of instructions that need to be fixed up with the true address of the symbol
1			6015 CAD	REFERENCES	7003 0 0000 64 6015	Load the upper four (“control”) digits of the current instruction being assembled
1			3 SRT		7004 0 0000 13 0003	Shift the low-order three control digits right into R, leaving only the high-order digit in A
1			6008 CAD		7005 0 0000 64 6008	Load the value of the high-speed loop tag for the current instruction being assembled
1			3 SLT		7006 0 0000 14 0003	Shift the three low-order control digits from R back into A with the high-speed loop-tag digit
1			1615 STC		7007 0 0000 02 1615	Store the updated control digits into their field in the duplicate card buffer and clear the A register
1			1608 STA		7008 0 0000 12 1608	Store the cleared A register to the high-speed loop tag in the duplicate card buffer,

							setting it to zero
1			BUN	LOCAT		7009 0 0000 20 1009	Branch back to the main line to continue assembling the instruction
1	HSLF	1615	CAD		TACK ON CON-	7010 0 0000 64 1615	To here from 1019 to finish assembling the instruction and punch the output card. The high-order six digits of the R register have the op code and effective address of the instruction. Load the four high-order control digits from the duplicate card buffer
1		6	SLT		TROL DIGITS	7011 0 0000 14 0006	Shift the control digits in A and the op code plus address in R left six digits to put the entire instruction (less sign) in A
1	4	6020	STA			7012 0 0000 12 6020	Store the assembled instruction in the card buffer (note that address 6020=6000)
1		6002	CAD		INCREMENT	7013 0 0000 64 6002	Load the current location counter from the card buffer
1			ADD	ONE	LOCATION	7014 0 0000 74 5005	Bump the location counter for the next instruction
1		6002	STC		COUNTER	7015 0 0000 02 6002	Store the updated location counter back into the card buffer
1		6016	CAD			7016 0 0000 64 6016	Load the numeric sign field from the card buffer
1	PUNCH	6001	STC		SET SIGN	7017 0 0000 02 6001	Store the numeric sign into the assembled sign field in the card buffer
1		1 6000	CWR		PUNCH AND	7018 0 0010 54 6000	Punch the assembled instruction from the card buffer to Cardatron output unit 1
1			BUN	PROC	RECYCLE	7019 0 0000 20 1000	Branch back to the main line to read the next card
1		1400	START			1400	Cardatron input format band 1. This is used when reading a source card to distribute its fields to words at 6005-6018 in the card buffer
1		33333333333				1400 3 3333 33 3333	
1		33333333333			INPUT FORMAT	1401 3 3333 33 3333	
1		33333333333			BAND ONE	1402 3 3333 33 3333	
1		33333333333				1403 3 3333 33 3333	
1		11111113333				1404 1 1111 11 3333	
1		11111110111				1405 1 1111 11 0111	
1		11111110111				1406 1 1111 11 0111	
1		2221330111				1407 0 2221 33 0111	
1		11 0000				1408 0 0011 00 0000	
1		2221 0000				1409 0 2221 00 0000	
1		11111 0000				1410 1 1111 00 0000	
1		11113211111				1411 1 1113 21 1111	
1		13132111111				1412 1 3132 11 1111	
1		31313				1413 0 0000 03 1313	
1		31310				1414 0 0000 03 1310	
1		31313131000				1415 3 1313 13 1000	
1		2231 0000				1416 2 2310 00 0000	
1		111 0000				1417 1 1100 00 0000	
1		21321111111				1418 2 1321 11 1111	
1		3 22222222				1419 3 0022 22 2222	
1		33333333333				1420 3 3333 33 3333	
1		33333333333				1421 3 3333 33 3333	
1		33333333333				1422 3 3333 33 3333	
1		33333333333				1423 3 3333 33 3333	
1		33333333333				1424 3 3333 33 3333	
1		33333333333				1425 3 3333 33 3333	
1		33333333333				1426 3 3333 33 3333	
1		33333333333				1427 3 3333 33 3333	
1		33333333333				1428 3 3333 33 3333	
1			START	7+		1436	Cardatron output format band 1. This formats the assembled output card from words in the 6000 loop
1		222222 0000			OUTPUT FORMAT	1436 2 2222 20 0000	
1		33333232222			BAND ONE	1437 3 3333 23 2222	

1	33222233333				1438 3 3222 23 3333	
1	22333333333				1439 2 2333 33 3333	
1	31111232222				1440 3 1111 23 2222	
1	11111133333				1441 1 1111 13 3333	
1	11111131111				1442 1 1111 13 1111	
1	11111131111				1443 1 1111 13 1111	
1	311 1 11111				1444 3 1101 01 1111	
1	33113333333				1445 3 3113 33 3333	
1	11 13333333				1446 1 1013 33 3333	
1	11133333333				1447 1 1133 33 3333	
1	11 11111111				1448 1 1011 11 1111	
1	2 111111111				1449 2 0111 11 1111	
1	13333333222				1450 1 3333 33 3222	
1	33333333010				1451 3 3333 33 3010	
1	33 1 1 1013				1452 3 3010 10 1013	
1	3311 133333				1453 3 3110 13 3333	
1	11111333333				1454 1 1111 33 3333	
1	11 1 111111				1455 1 1010 11 1111	
1	33333333311				1456 3 3333 33 3311	
1	32233333				1457 0 0032 23 3333	
1	20	START	0+		1478	Cardatron output format band 2. This formats the loader routine and forward-reference table entries that are punched at the end of the assembly from words in the 7000 loop
1	222 0000			OUTPUT FORMAT	1478 2 2200 00 0000	
1	22222222222			BAND TWO	1479 2 2222 22 2222	
1	22222222222				1480 2 2222 22 2222	
1	22222222222				1481 2 2222 22 2222	
1	22222222222				1482 2 2222 22 2222	
1	22222222222				1483 2 2222 22 2222	
1	22222222222				1484 2 2222 22 2222	
1	33332222222				1485 3 3332 22 2222	
1	32222333				1486 0 0032 22 2333	
1	1967	START		SYMB. OP. TABLE	1967	First half of the symbol table. It actually starts at 1900. Each word in 1900-2899 contains the alphanumeric characters for a symbol, left-justified over zero digits (spaces) with a 7 added to the word. The value of the symbol is in the word 1000 locations after it in memory. Symbols are located by hashing the alphanumeric value and indexing into this table. Collisions are handled by searching backwards in memory from the initial probe address, wrapping around to the end of the table if necessary. A zero word indicates an unused entry
1	4343425907			CCBR	1967 0 4343 42 5907	
1	1980	START			1980	
1	4359440007			CRD	1980 0 4359 44 0007	
1	1982	START			1982	
1	4359460007			CRF	1982 0 4359 46 0007	
1	1985	START			1985	
1	4359490007			CRI	1985 0 4359 49 0007	
1	2003	START			2003	
1	5763664607			PTWF	2003 0 5763 66 4607	
1	2010	START			2010	
1	4449650007			DIV	2010 0 4449 65 0007	
1	2069	START			2069	
1	4364425907			CUBR	2069 0 4364 42 5907	

```
1                  2102 START                                               2102
1            5459560007                     MRO              2102 0 5459 56 0007
1                  2124 START                                               2124
1            4241  0007                     BA               2124 0 4241 00 0007
1                  2175 START                                               2175
1            4341410007                     CAA              2175 0 4341 41 0007
1                  2178 START                                               2178
1            4341440007                     CAD              2178 0 4341 44 0007
1                  2188 START                                               2188
1            4441440007                     DAD              2188 0 4441 44 0007
1            6259630007                     SRT              2189 0 6259 63 0007
1                  2208 START                                               2208
1            4641440007                     FAD              2208 0 4641 44 0007
1                  2259 START                                               2259
1            4162430007                     ASC              2259 0 4162 43 0007
1                  2277 START                                               2277
1            4362410007                     CSA              2277 0 4362 41 0007
1                  2286 START                                               2286
1            4442420007                     DBB              2286 0 4442 42 0007
1                  2294 START                                               2294
1            4942  0007                     IB               2294 0 4942 00 0007
1                  2300 START                                               2300
1            4362640007                     CSU              2300 0 4362 64 0007
1                  2330 START                                               2330
1            4662640007                     FSU              2330 0 4662 64 0007
1                  2344 START                                               2344
1            6441  0007                     UA               2344 0 6441 00 0007
1                  2377 START                                               2377
1            4343420007                     CCB              2377 0 4343 42 0007
1                  2394 START                                               2394
1            4343590007                     CCR              2394 0 4343 59 0007
1            4353590007                     CLR              2395 0 4353 59 0007
1                  2405 START                                               2405
1            4349594107                     CIRA             2405 0 4349 59 4107
1                  2410 START                                               2410
1            5662440007                     OSD              2410 0 5662 44 0007
1            4263840007                     BT4              2411 0 4263 84 0007
1            4263850007                     BT5              2412 0 4263 85 0007
1            4263860007                     BT6              2413 0 4263 86 0007
1            4263870007                     BT7              2414 0 4263 87 0007
1                  2449 START                                               2449
1            4853630007                     HLT              2449 0 4853 63 0007
1                  2456 START                                               2456
1            4144410007                     ADA              2456 0 4144 41 0007
1                  2459 START                                               2459
1            4144440007                     ADD              2459 0 4144 44 0007
1                  2464 START                                               2464
1            6263415970                     START            2464 0 6263 41 5970
1                  2469 START                                               2469
1            6262430007                     SSC              2469 0 6262 43 0007
1                  2479 START                                               2479
1            4364420007                     CUB              2479 0 4364 42 0007
1                  2482 START                                               2482
```

1	4264550007	BUN	2482 0 4264 55 0007	
1	2496 START		2496	
1	4364590007	CUR	2496 0 4364 59 0007	
1	2529 START		2529	
1	4644650007	FDV	2529 0 4644 65 0007	
1	4654640007	FMU	2530 0 4654 64 0007	
1	2536 START		2536	
1	5763590007	PTR	2536 0 5763 59 0007	
1	2543 START		2543	
1	5763660007	PTW	2543 0 5763 66 0007	
1	2568 START		2568	
1	6263410007	STA	2568 0 6263 41 0007	
1	2570 START		2570	
1	6263430007	STC	2570 0 6263 43 0007	
1	2577 START		2577	
1	5344420007	LDB	2577 0 5344 42 0007	
1	2589 START		2589	
1	6253630007	SLT	2589 0 6253 63 0007	
1	2599 START		2599	
1	4555440007	END	2599 0 4555 44 0007	
1	5464530007	MUL	2600 0 5464 53 0007	
1	2605 START		2605	
1	4355690007	CNZ	2605 0 4355 69 0007	
1	2668 START		2668	
1	6264410007	SUA	2668 0 6264 41 0007	
1	6264420007	SUB	2669 0 6264 42 0007	
1	2672 START		2672	
1	4256460007	BOF	2672 0 4256 46 0007	
1	2683 START		2683	
1	4366460007	CWF	2683 0 4366 46 0007	
1	2686 START		2686	
1	4366490007	CWI	2686 0 4366 49 0007	
1	2696 START		2696	
1	4366590007	CWR	2696 0 4366 59 0007	
1	2709 START		2709	
1	4246840007	BF4	2709 0 4246 84 0007	
1	4246850007	BF5	2710 0 4246 85 0007	
1	4246860007	BF6	2711 0 4246 86 0007	
1	4246870007	BF7	2712 0 4246 87 0007	
1	2740 START		2740	
1	5955440007	RND	2740 0 5955 44 0007	
1	2815 START		2815	
1	5556590007	NOR	2815 0 5556 59 0007	
1	2820 START		2820	
1	4567630007	EXT	2820 0 4567 63 0007	
1	2967 START	EQUIVALENTS.	2967	Start of the second half of the symbol table (it actually begins at 2900). Each word below is the value for the symbol in the word 1000 locations before it in memory
1	39	CCBR	2967 0 0000 00 0039	
1	2980 START		2980	
1	44	CRD	2980 0 0000 00 0044	
1	2982 START		2982	
1	48	CRF	2982 0 0000 00 0048	

```
1                  2985 START                                               2985
1                    45                     CRI              2985 0 0000 00 0045
1                  3003 START                                               3003
1                     7                     PTWF             3003 0 0000 00 0007
1                  3010 START                                               3010
1                    61                     DIV              3010 0 0000 00 0061
1                  3069 START                                               3069
1                    31                     CUBR             3069 0 0000 00 0031
1                  3102 START                                               3102
1                    70                     MRO              3102 0 0000 00 0070
1                  3124 START                                               3124
1                    11                     BA               3124 0 0000 00 0011
1                  3175 START                                               3175
1                    66                     CAA              3175 0 0000 00 0066
1                  3178 START                                               3178
1                    64                     CAD              3178 0 0000 00 0064
1                  3188 START                                               3188
1                    10                     DAD              3188 0 0000 00 0010
1                    13                     SRT              3189 0 0000 00 0013
1                  3208 START                                               3208
1                    80                     FAD              3208 0 0000 00 0080
1                  3259 START                                               3259
1                    16                     ASC              3259 0 0000 00 0016
1                  3277 START                                               3277
1                    67                     CSA              3277 0 0000 00 0067
1                  3286 START                                               3286
1                    22                     DBB              3286 0 0000 00 0022
1                  3294 START                                               3294
1                    32                     IB               3294 0 0000 00 0032
1                  3300 START                                               3300
1                    65                     CSU              3300 0 0000 00 0065
1                  3330 START                                               3330
1                    81                     FSU              3330 0 0000 00 0081
1                  3344 START                                               3344
1                     6                     UA               3344 0 0000 00 0006
1                  3377 START                                               3377
1                    38                     CCB              3337 0 0000 00 0038
1                  3394 START                                               3394
1                    29                     CCR              3394 0 0000 00 0029
1                    33                     CLR              3395 0 0000 00 0033
1                  3405 START                                               3405
1                     1                     CIRA             3405 0 0000 00 0001
1                  3410 START                                               3410
1                    73                     OSD              3410 0 0000 00 0073
1                    34                     BT4              3411 0 0000 00 0034
1                    35                     BT5              3412 0 0000 00 0035
1                    36                     BT6              3413 0 0000 00 0036
1                    37                     BT7              3414 0 0000 00 0037
1                  3449 START                                               3449
1                     8                     HLT              3449 0 0000 00 0008
1                  3456 START                                               3456
1                    76                     ADA              3456 0 0000 00 0076
1                  3459 START                                               3459
```

1	74		ADD	3459 0 0000 00 0074	
1	3464	START		3464	
1		BUN START	START	3464 0 0000 20 1034	Entry for the START pseudo op code. Note that this is a branch instruction to a special handling routine, not the value of the symbol
1	3469	START		3469	
1	17		SSC	3469 0 0000 00 0017	
1	3479	START		3479	
1	30		CUB	3479 0 0000 00 0030	
1	3482	START		3482	
1	20		BUN	3482 0 0000 00 0020	
1	3496	START		3496	
1	21		CUR	3496 0 0000 00 0021	
1	3529	START		3529	
1	83		FDV	3529 0 0000 00 0083	
1	82		FMU	3530 0 0000 00 0082	
1	3536	START		3536	
1			PTR	3536 0 0000 00 0000	
1	3543	START		3543	
1	3		PTW	3543 0 0000 00 0003	
1	3568	START		3568	
1	12		STA	3568 0 0000 00 0012	
1	3570	START		3570	
1	2		STC	3570 0 0000 00 0002	
1	3577	START		3577	
1	72		LDB	3577 0 0000 00 0072	
1	3589	START		3589	
1	14		SLT	3589 0 0000 00 0014	
1	3599	START		3599	
1		BUN END	END	3599 0 0000 20 1032	Entry for the END pseudo op code. Note that this is another branch instruction to a special handling routine, not the value of the symbol
1	60		MUL	3600 0 0000 00 0060	
1	3605	START		3605	
1	4		CNZ	3605 0 0000 00 0004	
1	3668	START		3668	
1	77		SUA	3668 0 0000 00 0077	
1	75		SUB	3669 0 0000 00 0075	
1	3672	START		3672	
1	28		BOF	3672 0 0000 00 0028	
1	3683	START		3683	
1	58		CWF	3683 0 0000 00 0058	
1	3686	START		3686	
1	55		CWI	3686 0 0000 00 0055	
1	3696	START		3696	
1	54		CWR	3696 0 0000 00 0054	
1	3709	START		3709	
1	24		BF4	3709 0 0000 00 0024	
1	25		BF5	3710 0 0000 00 0025	
1	26		BF6	3711 0 0000 00 0026	
1	27		BF7	3712 0 0000 00 0027	
1	3740	START		3740	
1	23		RND	3740 0 0000 00 0023	
1	3815	START		3815	
1	15		NOR	3815 0 0000 00 0015	

1		3820	START			3820	
1		63			EXT	3820 0 0000 00 0063	
1		1200	START			1200	Initialization code for the assembler. The loader deck should branch to here to start the program
1	1	1429	CWF		LOAD UP EASY	1200 0 0010 58 1429	Load Cardatron format band 1 on output unit 1 from the 29 words starting at 1429
1	1	1400	CRF		FROM THE DRUM	1201 0 0010 48 1400	Load Cardatron format band 1 on input unit 1 from the 29 words starting at 1400
1	21	1458	CWF			1202 0 0210 58 1458	Load Cardatron format band 2 on output unit 1 from the 29 words starting at 1458
1		1220	BT4			1203 0 0000 34 1220	Block transfer to the 4000-, 5000-, 6000-, and 7000 high speed loops starting from addresses 1220, 1260, 1240, and 1280 respectively. It’s not clear what the purpose of these instructions are, because if they are left as is when the program is assembled with itself, they overwrite the high-speed loops, rendering the program useless. It is possible this listing is from a version configured to be assembled with the mag-tape based MEASY assembler, whose loader ran out of the high-speed loops, so these instructions were necessary to pre-load the loops during initialization. In any case, they were changed to BF*n* instructions in order to get the program to work with the retro-205 emulator, which effectively made them no-ops
1		1260	BT5			1204 0 0000 35 1260	
1		1240	BT6			1205 0 0000 36 1240	
1		1280	BT7			1206 0 0000 37 1280	
1			CAD	SIGN6		1207 0 0000 64 1722	Load the EXT mask used at 1711
1		10	SRT			1208 0 0000 13 0010	Shift 10 digits into the R register
1		6004	CAD			1209 0 0000 64 6004	Load the literal 6
1		9	CIRA			1210 0 0000 01 0009	Rotate the A register 9+1 digits to the left, including sign. This will leave the 6 in the sign position
1		10	SLT			1211 0 0000 14 0010	Shift 10 digits from R into A without affecting the sign. This has effectively applied a sign of 6 to the EXT mask
1			STA	SIGN6		1212 0 0000 12 1722	Store the updated EXT mask back into the SIGN6 word
1			BUN	PROC		1213 0 0000 20 1000	Branch to the main line to read the first card of the source deck
1		1500	END			1500	End assembly. The loader routine and its forward reference table will be loaded into memory starting at location 1500