TABLE OF CONTENTS

			Page
I.	BASIC CONCEPTS		
	A.	Introduction	1
	B.	Architecture	1
		1. Description of Memory	1
		2. Addressing Scheme	1
		3. 290 Front Panel Controls	2
	C.	Octal Representation of Binary Numbers	3
	D.	Conversion	3
	E.	Mnemonics	4
II.	NUMERICAL CONCEPTS IMPORTANT TO THE 1080		
	A.	Complements	4
	B.	Two's Complement Notation	4
	C.	The Logical AND	5
	D.	Registers Used in the 1080	7
III.	ADDRESSING MODES		
	A.	Immediate	7
	B.	Direct	7
	C.	Indirect	9
IV.	DESCRIPTION OF INSTRUCTIONS		
	A.	Group I Instructions	10
	B.	Jump Instruction	10
	C.	JMS Instruction	11
	D.	Shift Instructions	11
	E.	Skip Instructions	12
	F.	Miscellaneous Instructions	13
	G.	Input/Output Instructions	13
	H.	Hardware Instructions	14
	I.	Automatic Arithmetic Instructions	15
V.	THE ASSEMBLER		
	A.	Introduction	17
	B.	Special Characters	18

V. THE ASSEMBLER (continued)

C. Syntax 18
D. Addresses 19
E. Loading and Startup 20
F. Assembler Commands 20
G. Editor Commands 21
H. Cautionary Notes

VI. A PROGRAMMING EXAMPLE

A. Apologia 23
B. Outline of Good Programming Practices 23
C. Definition of a Sample Program 24
D. Flow Chart of the Program 25
E. Discussion 26
F. Conclusion 26

VII. TABLE SEARCHING ROUTINE 27

TABLES

I. 1080 Assembler Instructions 6
II. Assembler Commands 22

APPENDICES

I. ASCII Character Codes 29
II. Modifying the Assembler 31
III. Status Word Interpretation 32
IV. Decimal Octal Conversion Table 33

PROGRAMMING THE NICOLET 1080 STORED PROGRAM COMPUTER

I. BASIC CONCEPTS

A. Introduction

The Nicolet Instrument Corporation 1080 computer operates in both a wired program and a stored program mode. In the wired program mode, wired logic directs the analog to digital conversion, memory addition, sweep counting and display. In the stored program mode, another set of wired circuitry can be directed to perform certain elementary operations, called the Instruction Set, by the setting of certain bits in the Instruction Register. The Instruction Register (IR) is set by calling binary numbers from a preselected portion of the computer memory. Thus, the binary numbers stored in the 1080 memory can be either data or program instructions, depending on how they are addressed.

B. Architecture

1. Description of Memory

The 1080 consists of a wired processor and a programmable processor sharing the same memory and some of the same registers. The memory consists of planes of ferrite cores which can be magnetized into one of two states, representing either binary 1 or 0. The cores are logically divided into units called words. Each word is a string of 20 cores or 20 bits, and can be considered a logical unit containing one number, either data or an instruction.

Since each bit can be one or zero, it is convenient to consider each word a 20-bit binary number. Thus, any number between 0 and $2^{20}-1$ can be represented in a 20-bit word. The bits in each memory word, and in the registers, are numbered from 0 to 19, corresponding to the power of two they represent. Thus the rightmost bit is bit 0 and the leftmost bit, bit 19.

2. Addressing Scheme

For mechanical reasons, memory is provided in stacks of 4096 words, or 4K, each. The typical 1080 system consists of one stack for program storage and one or more stacks for data storage. The words in memory are addressed (or numbered) sequentially with the program memory starting at address 0000000 and the data memory at 100000_8.

This is an arbitrary distinction and the number of stacks in data memory vs. program memory can be adjusted internally. The only difference between the two segments is that wired program data storage begins at

100000 and proceeds through as many stacks as are indicated by the Measure Memory Allocation buttons. The Protect Program button protects the stored program segment (the first 4K) from being destroyed by the wired program. This could happen if the number of data points selected exceeded the number of data stacks in the machine.

Nothing in the design of the 1080 prevents one from storing and executing programs in the data section of memory, as long as the wired program is prevented from overwriting them.

3. 290 Front Panel Controls

The panel of the 1080 system containing the row of 20 toggle switches, labeled 290 Display Control, is extremely useful to the programmer. The row of seven pushbuttons to the left of the Execute and Stop buttons can be effectively used during program checkout and are described below. The function of each of these buttons is enabled by depressing it and then pressing Execute.

START - starts execution of program instructions at location 0. Note that this is used only for programs whose starting address (SA) is zero. All other programs must be started by LOAD PC followed by CONTINUE.

CONTINUE - begins execution at the location set in the Program Counter (PC).

LOAD PC - the contents of the 20 toggle switches (the switch register) are loaded into the PC. Each switch in the up position has the value 1. Thus, to start a program with a SA other than 0, set the SA in the switch register, depress LOAD PC, press Execute, depress CONTINUE and press Execute.

SINGLE INS - depression of this switch immediately halts the stored program if running, and each time the Execute button is pressed, one instruction is executed. The AC, PC and IR may then be examined to determine the result of each instruction. Remember that the address in the PC represents that of the next instruction to be executed.

EXAMINE - the contents of the location whose address is in the PC are loaded into the AC for examination. If STEP is depressed, the next sequential location is loaded into the AC each time Execute is pressed.

DEPOSIT - the number set in the switch register is deposited in the location set in the PC. If STEP is depressed, the PC is incremented so that one need only load the PC once to deposit numbers in successive locations.

The STOP button on the 290 panel has the same function as the Stored Program Stop button. It does not halt the wired program.

C. Octal Representation of Binary Numbers

Since binary numbers are tedious to write out, it is common to abbreviate them in the octal (base 8) numbering system. This has the advantage over the decimal system that each group of three bits represents an octal number, making interconversion easy. There are two concepts one must learn to understand the octal system: first, $7 + 1 = 10_8$ and second, the interconversion table given below:

Octal	Binary
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111
10	1 000

To reiterate, binary numbers, if divided into groups of three can be virtually instantaneously converted into their octal equivalent. For example,

BINARY	OCTAL	DECIMAL
0 0 1 1 1 0 0 0 0 1 1 0	1 6 0 6	902
1 1 1 0 0 1 1 0 0 1 0 1 0 0 0	7 1 4 5 0	29,486
0 0 0 0 0 1 1 1 1 0 0 1	1 7 1	121
0 1 1 1 0 1 1 0 0 0 1 1	3 5 4 3	1,891

D. Conversion

Conversion between number bases is a common operation discussed in great detail in many elementary mathematics texts. We will just remind the reader that a number in any base simply represents a series of multipliers times successive powers of the base.

Thus, the octal number 71450 means

$$\begin{aligned} 7 \times 8^4 = 7 \times 4096 &= 28,678 \\ + 1 \times 8^3 = 1 \times 612 &= 512 \\ + 4 \times 8^2 = 4 \times 64 &= 256 \\ + 5 \times 8^1 = 5 \times 8 &= 40 \\ + 0 \times 8^0 = 0 \times 1 &= 0 \\ & \overline{29,486_{10}} \end{aligned}$$

The binary number 0 1 1 0 1 0 means

$$
\begin{aligned}
0 \times 2^5 &= 0 \times 32 = \\
1 \times 2^4 &= 1 \times 16 = 16 \\
1 \times 2^3 &= 1 \times 8 = 8 \\
0 \times 2^2 &= 0 \times 4 = 0 \\
1 \times 2^1 &= 1 \times 2 = 2 \\
0 \times 2^0 &= 0 \times 1 = 0 \\
& \qquad\qquad\quad 26_{10}
\end{aligned}
$$

E. Mnemonics

Since strings of 20-bit binary numbers are extremely cumbersome for the average person to assimilate, there exists a program called the Assembler, which converts simple alphabetic abbreviations for the computer instructions into their binary equivalents. Thus, it is not generally necessary to learn the actual binary numbers which cause the computer to perform certain operations; it is merely necessary to learn a few simple abbreviations called mnemonics. The complete list of mnemonics accepted by the Assembler is given in Table I. They are explained in more detail following the list.

II. NUMERICAL CONCEPTS IMPORTANT TO THE 1080

A. Complements

The ones complement (or simply the complement) of a number is defined as the complementing of each bit: all 1's become 0's and all 0's become 1's. Thus, the 20-bit octal number 2341610 (10 011 100 001 110 001 000) becomes 1436167 (01 100 011 110 001 110 111) when complemented.

B. Two's Complement Notation

The two's complement of a number is defined as the one's complement plus one. The NIC-1080 is a two's complement machine, meaning that it considers the two's complement of a positive number its negative. This may be more clearly illustrated by the following example:

```
 3 0000003
 2 0000002
 1 0000001
 0 0000000
-1 3777777
-2 3777776
-3 3777775
```

By convention, then, half of all possible integers are considered positive and half negative. The half which are considered negative always have bit 19 (the left most bit) = 1, so that bit 19 is generally considered the "sign bit."

Now, since we have defined the negative of a number as its two's complement, let us see what the result of addition of positive and negative numbers is.

```
  3     0000003
 -1     3777777     (Remember that 7 + 1 = 10, so 7 + 3 = 12)
  2   1 0000002
```

The sum of 3 and -1 is found to be 2. The one at the extreme left is the "carry out" or overflow to the 21st bit. This value will be found in the Link, which acts as a 21st bit of the accumulator. Testing of the Link may be used to detect carry out so that one may discover when a number has become too large for storage in a single word, or when "subtraction" has occurred.

C. The Logical AND

The instructions in Table I are more or less self-defining. However, the AND instruction may be less familiar to the user. In this operation, if both arguments are 1, the result is 1, and if the two arguments are different or zero, the result is zero. Using the exclamation point to represent the AND operation, 1!1 = 1 and 1! 0 = 0!1 = 0!0 = 0 or

	0	1
0	0	0
1	0	1

The AND operation is always carried out between the AC and a memory location and may utilize any of the three addressing modes. It is most useful for masking out unnecessary bits so that one can examine only those of interest. For instance, if one were interested only in bits 0-2 of a number, he could mask that number as follows:

Number	2163015	10 001 110 011 000 001 101
Mask for bits 0-2	0000007	00 000 000 000 000 000 111
Result	0000005	00 000 000 000 000 000 101

This is carried out in the computer by the instructions:

```
     MEMA NUMBER      /PUT THE NUMBER IN THE AC
     ANDA MASK        /PERFORM THE LOGICAL AND
     ACCM RESULT      /STORE THE RESULT
NUMBER, 2163015
MASK, 7
RESULT, 0             /THE NUMBER 5 IS DEPOSITED HERE
```

TABLE I

GROUP I INSTRUCTIONS

Octal	Mnemonic	Source (Operator)
0500000	A+M	Add accumulator (AC) and memory
0520000	AMP	Add accumulator, memory and 1
0460000	A-M	Subtract memory from the accumulator
0320000	M-A	Subtract the AC from memory
0440000	ACM	Accumulator plus the complement of memory
0300000	CAM	Complement of the AC plus memory
0000000	AND	Logical AND between accumulator and memory
0100000	MEM	Take the contents of memory
0120000	MPO	Memory plus 1
0700000	MMO	Memory minus 1
0060000	MNG	Negative of memory
0040000	ACC	Accumulator
0420000	APO	Accumulator plus 1
0540000	AMO	Accumulator minus 1
0200000	ACP	Complement of the accumulator
0220000	ANG	Negative of the accumulator
0160000	ZER	Take the number zero*
0020000	ONE	Take the number 1
0140000	MON	Take the number -1
0740000	MTO	Take the number -2*
0000000	JMP	Jump
2000000	JMS	Jump to a subroutine, leave PC in first address

Destination (Suffix)

0010000	A	Accumulator
0004000	M	Memory
0002000	Z	Zero test unit Skip if result is 0

*These instructions, if loaded into the AC, will change the state of the Link.

GROUP II INSTRUCTIONS

Shift Group

0005000	LASH n	Left arithmetic shift of AC, Link unaffected
0005020	RASH n	Right arithmetic shift
0005040	LLSH n	Left logical shift
0005060	RLSH n	Right logical shift

$0 \leq n \leq 17_8$

Skip Group

0005100	SKIP on	0400020	ZAC	Zero AC
		0420000	MOAC	Minus one AC
		0540020	POAC	Plus one AC
0005140	EXCT on	0000010	AC$\emptyset$	AC bit $\emptyset$ = 1
		0000004	AC19	AC bit 19 = 1
		0000001	L	Link = 1

Miscellaneous Group

0005220	STOP	Processor halts
0005210	CLL	Clear the link
0005204	STL	Set the link = 1
0005202	TLAC	Transfer the link to bit 19 of the AC, bits 0-18 and link unchanged
0005201	TACL	Transfer bit 19 of the AC to the link, bit 19 unchanged

Input-Output Instructions

0006454	TTYRF	Skip when the Teletype keyboard reader is ready
0044453	RDTTY	Read the Teletype keyboard-reader buffer into the AC
0006444	TTYPF	Skip when the Teletype printer is ready for a new character
0004443	PRTTY	Print the character contained in the AC
0006464	HSRF	Skip if the high speed reader is ready
0044463	RHSR	Read the next character from the high speed reader into the AC
0006474	HSPF	Skip if the high speed punch is ready
0004473	RHSP	Punch the contents of the AC

Hardware Access Instructions

0105000	VDSH	Shift AC by amount controlled by vertical display scale switch
0004371	REDS	Reset Data Scaler to start new digitization
0004372	STDG	Start digitization
0004364	RDG	Read Digitizer result into AC
0006362	DWSK	Skip on Dwell Time flag
0004361	ASRMP	Advance Sweep Ramp
0004364	RSWP	Reset Sweep Ramp
0044034	STATUS	Read hardware settings into AC. See Appendix III.

Automatic Arithmetic Instructions

0505320	MULT	Multiply contents of MQ by next location
0465300	DIVD	Divide AC-MQ by contents of next location
0004354	TACMQ	Transfer AC to MQ
0004343	TMQAC	Transfer MQ to AC
0044354	ZRAM	Zero AC and MQ
0004347	BITINV	Bit invert the AC
0405000	RISH	Right integer shift of AC

Display Instructions

0214001	TACXD	Transfer AC to X display register (AC is complemented)
0214012	TACYD	Transfer AC to Y display register
0214014	INCSC	Increment x-axis
0214011	INTENS	Intensify display

D. Registers Used in the 1080

From the programmer's viewpoint, all arithmetic operations take place in the accumulator (AC). Memory data is loaded into the AC for calculation from memory through the memory buffer. The Program Counter indicates the address of the next instruction. The PC is one greater than the current instruction address unless a jump or skip is generated.

The Group I operands describe the source of data, for example, A+M means accumulator plus memory. The suffixes define the destination of this data: A, the Accumulator; M, Memory; and Z, the Zero Test Register. If zero is loaded into the zero test register, the next instruction is skipped. As is apparent from Figure 1, the suffixes merely open "gates" allowing the calculated quantity to be loaded into various registers.

It is important to note that all such transfers are non-destructive. The source is never harmed by the operation. Thus, A+MM TEMP adds the AC and memory to memory without changing the AC, and A-MZ TEMP loads the zero test register without changing either the AC or memory. No matter what register or combination of registers are loaded from the AC, the AC is never cleared.

III. ADDRESSING MODES

A. Immediate

The Nicolet 1080 has three addressing modes, immediate, direct and indirect. The immediate mode is the simplest to understand. In the Assembler the immediate mode is symbolized by the left parenthesis before the number or tag to be operated on.

```
MEMA (123       /PLACE 123 IN THE AC
```

The above instruction is in the immediate mode. The right hand 10 bits of the instruction are the data. Thus any number from -2000_8 to 2000_8 can be represented:

```
MPOA (1777      /SET AC = 2000
MNGA (36        /SET AC = -36
A+MA (15        /ADD 15 TO THE AC
MCPA (1777      /SET AC = -2000
```

B. Direct

In the direct mode, each instruction refers to another location which is tagged with some label. A label may have up to six characters and may contain

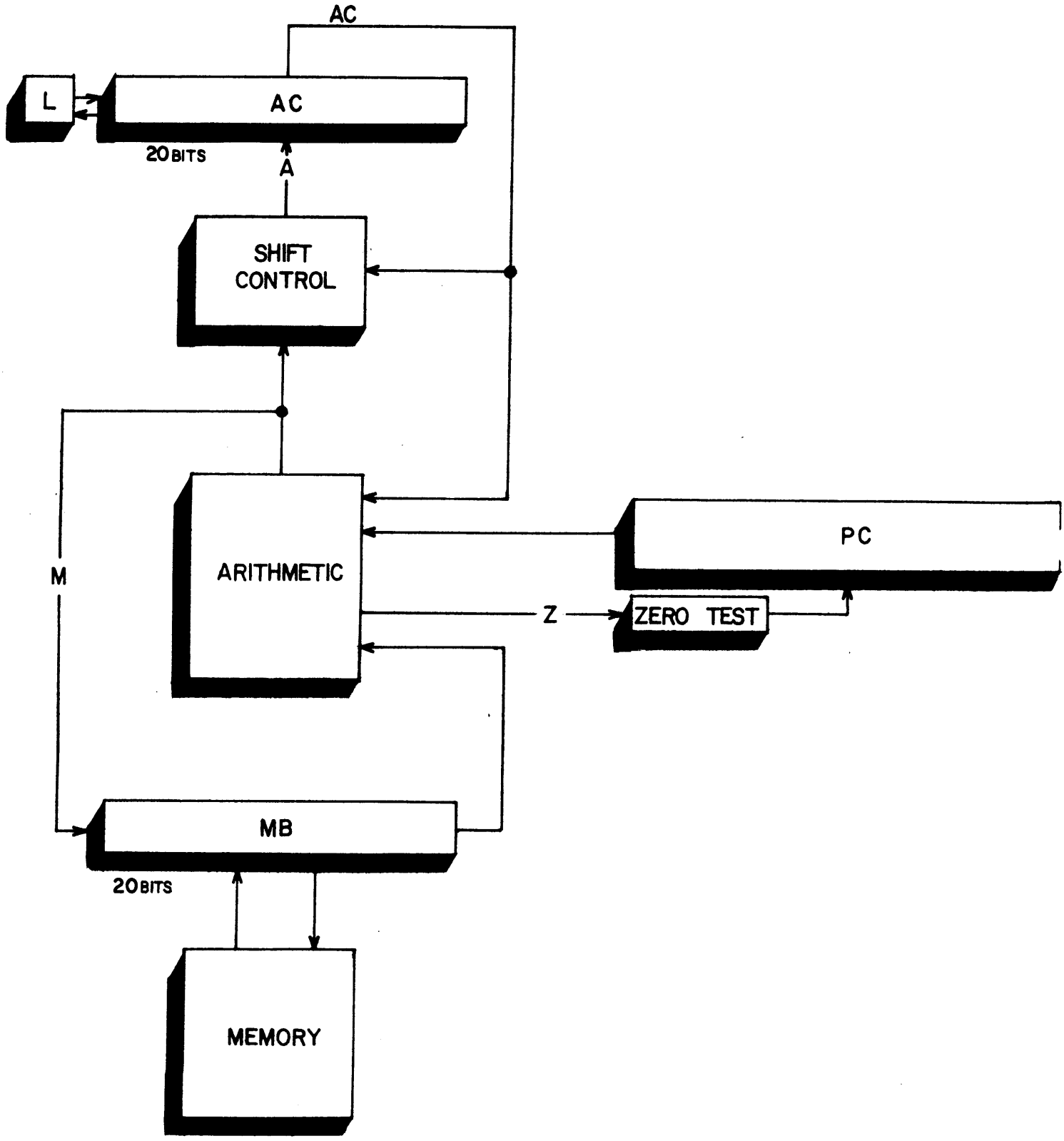

AC
L
AC
20 BITS
A
SHIFT
CONTROL
M
ARITHMETIC
PC
Z
ZERO TEST
MB
20 BITS
MEMORY

FIGURE – 1

any combination of letters and numbers. However, the first character must be alphabetic. In the direct mode, the left ten bits comprise the actual instruction as before. However, the right ten bits represent a relative address.

Since one can only represent 2000_8 (1024_{10}) addresses with ten bits, the convention has been adopted that these refer to a relative address on that memory page. Each memory page consists, therefore, of 2000_8 words, beginning at even octal thousands. In the lowest 4K, for example, the pages start at 0000000, 0002000, 0004000 and 0006000. Thus, one can address any of the 2000_8 locations on this memory page directly:

```
        *2000           /START AT ADDRESS 2000

MEMA T1         /PUT CONTENTS OF LOCATION T1 IN AC
LASH 3          /SHIFT 3 PLACES LEFT: OR MULTIPLY BY 8
ACCM T2         /AND STORE RESULT IN T2

STOP
T1,         5631
T2,         0
```

Note that a tagged location is defined by any one to six character name, followed by a comma.

C. Indirect Addressing

The indirect addressing mode is provided to address locations off the current memory page. The location ultimately addressed may contain a constant or be the destination of a jump. In the indirect mode, there must be a pointer location on the current page, which contains the address of the location indirectly addressed:

```
        *2000

MEMA @ PTEMP        /GET THE CONTENTS OF LOCATION TEMP
                    /INDIRECTLY

ACCM LOCAL          /AND STORE IT
JMP @ POINTER       /JUMP TO ROUTINE PNTER ON P. 6000

PTEMP,  TEMP
LOCAL,  Ø
POINTR, PNTER

        *6000

TEMP,  12345
PNTER, RLSH 15      /ROUTINE CONTINUES FROM HERE, SHIFT AC
    .               /15 BITS RIGHT
    .
    .
```

IV. DESCRIPTION OF INSTRUCTIONS

A. Group I Instructions

All Group I instructions involve a transfer of data from a source to a destination. There may be one or more destinations, specified by the suffixes A, M and Z but without at least one suffix, the instruction is meaningless and causes an error message from the Assembler. (This does not apply to JMP and JMS)

The extreme versatility of the 1080 instruction set is exemplified by the instruction A+MMZ TEMP, in which the contents of the accumulator are added to the contents of the location tagged TEMP, stored in TEMP and the result tested for zero. If this sum is zero, the next instruction is skipped. The accumulator remains unchanged. Note that it is legal to test for zero without putting the result anywhere except in the zero test register: A-MZ TEMP2. This leaves both the AC and memory unchanged, but generates a skip if the result is zero.

B. Jump Instruction

The instruction JMP DESTN simply causes jump to the location tagged DESTN. A typical example of the utility of the jump might be the following program for adding 20_8 sequential numbers stored starting at DATA1.

```
MEMA (20                  /GET THE NUMBER 20
ACCM COUNT                /AND PLACE IN THE REGISTER COUNT
MEMA PDATA                /GET THE ADDRESS OF THE FIRST WORD
                          /OF DATA
ACCM POINT                /AND SAVE AS POINTER

/NOW ADD TOGETHER 20 LOCATIONS STARTING AT DATA1
ZERA                      /ZERO THE AC
REPEAT, A+MA @ POINT      /ADD NEXT LOCATION TO AC
MPOM POINT                /INCREMENT DATA POINTER
MMOMZ COUNT               /DECREMENT COUNT AND SKIP WHEN IT
                          /REACHES ZERO
JMP REPEAT                /JUMP BACK TO ADD ANOTHER VALUE
ACCM SUM                  /PLACE SUM IN LOCATION "SUM"
STOP
COUNT, 0                  /COUNTER
PDATA, DATA1              /POINTER TO FIRST DATA LOCATION
POINT, 0                  /POINTER TO CURRENT DATA LOCATION
SUM, 0                    /CONTAINS FINAL SUM
DATA1, 0                  /FIRST DATA POINT IN LIST
     .
     .
     .
```

C. JMS Instruction

The JMS or jump to subroutine instruction is the most useful instruction in the 1080's repertoire. It provides a method of writing reusable routines which can be called from many different spots in memory without any modification of the actual routine. When a JMS SUBRTN instruction is executed, the address of the instruction following JMS SUBRTN is placed in the first location of the routine SUBRTN. This provides a pointer address for the return from the subroutine to the main program. Consider the following example:

```
        *2000
      /THIS IS THE MAIN PROGRAM
2000    START, MEMA (324    /PUT THE ASCII EQUIVALENT OF "T" IN
                            /THE AC
2001    JMS TYPE            /JUMP TO THE TYPE SUBROUTINE
2002    MEMA (305
              .
              .
              .
2100    TYPE, 0             /THE FIRST LOCATION OF A SUBROUTINE IS
                            /ALWAYS ZERO.  THE ADDRESS 2002 IS AUTO-
                            /MATICALLY DEPOSITED IN LOCATION 2100
                            /WHEN TYPE IS CALLED FROM ADDRESS 2001
2101    T1, TTYPF           /WAIT FOR THE TELETYPE FLAG, SKIP WHEN
                            /READY
2102    JMP T1              /WAIT IN THIS LOOP UNTIL TTY IS READY TO
                            /PRINT
2103    PRTTY               /PRINT THE ASCII CHARACTER IN THE AC
2104    JMP @ TYPE          /EXIT FROM TYPE BY INDIRECT JUMP THROUGH
                            /POINTER IN LOCN 2100
```

In the above example, the routine TYPE is called from location 2001 by JMS TYPE. The address following the call, in this case 2002, is placed in location TYPE (address 2100) by the JMS instruction. Execution of instructions then begins automatically at the second location of the subroutine. The location TYPE thus contains the return address and the instruction JMP @ TYPE causes a JMP to the location pointed to by TYPE, or location 2002. Thus, an automatic return is generated. Note that it is not necessary to know what location the subroutine was called from in advance, or indeed, what page it was called from. The instruction JMS @ PTYPE where PTYPE points to location TYPE on some other page will still cause the address following that instruction to be deposited in location TYPE.

D. Shift Instructions

There are three kinds of shifts possible with the 1080: logical, arithmetic and integer. The number of places shifted is controlled by an integer following the instruction, which can vary from 0 to 17_8. A logical shift is an end-around

shift of the bits of the accumulator, so that the instruction RLSH 1 causes all bits to move one place right, and bit 0 to move around to bit 19. The direction is simply reversed by a left logical shift (LLSH).

The arithmetic shift is signed shift. It causes the sign bit to be propagated to the right during right shifts. During left arithmetic shifts, bits are dropped off the left end. Thus, if bit 19 (the sign bit) is 1, indicating a negative number, the instruction RASH 3 will cause bits 0 - 18 to be shifted three places right. Bits 0 - 2 will be lost, bit 18 will have moved to bit 15 and so forth. The sign bit will be copied into bits 18 - 16, making bits 19 - 16 all ones. This is illustrated below:

AC before instruction 10 001 010 110 011 100 101 (2126345 octal)

RASH 3

AC after instruction 11 110 001 010 110 011 100 (101 lost) 3612634_8

The RISH instruction (automatic arithmetic option) causes the bits of the AC to be shifted right without regard to sign, with the least significant bits "falling off" the end. While LISH does not exist, LASH has the same effect.

The number of shifts performed in one instruction can vary from 0 to 15_{10} or 0 to 17_8. There is no need for successive shift instructions, since up to 15 shifts can be performed in a single instruction. When the instruction is assembled, bits 0 - 3 contain the number of shifts to be performed. None of the shift instructions affect the Link.

E. Skip Instructions

The 1080 can test for several conditions and generate program branches when these conditions exist or do not exist. The test for zero is part of the Group I instructions and generates a skip of the next instruction if the calculated quantity is zero:

```
A-MZ TEST1      /SUBTRACT TEST1 FROM AC AND SKIP IF
                /RESULT IS ZERO
JMP A           /JUMP TO A IF NON-ZERO
JMP B           /JUMP TO B IF ZERO
```

The other conditions which one can test for are

ZAC	Zero accumulator
MOAC	Minus one accumulator
POAC	Plus one accumulator
ACØ	Bit Ø of the AC = 1, useful to test for odd or even numbers, or rotation overflow
AC19	The sign bit = 1, the number is negative
L	The Link is one

One can execute the next instruction (EXCT) or skip the next instruction (SKIP) for any of the above conditions. A typical sequence would be:

```
JMS READ          /READ A CHARACTER FROM THE TELETYPE
                  /KEYBOARD
A-MA (260         /SUBTRACT ASCII BIAS OF 260
EXCT AC19         /SKIP IF RESULT IS POSITIVE
JMP ERR           /GO TO ERROR ROUTINE IF LESS THAN 260 =
                  /AC NEGATIVE
JMP DOIT          /GO DO WHATEVER COMES NEXT
READ,Ø            /SUBROUTINE TO READ FROM THE KEYBOARD
                  /OR LSR
R1,TTYRF          /SKIP WHEN CHARACTER STRUCK
JMP R1            /WAIT LOOP UNTIL KB STRUCK
RDTTY             /READ TTY
JMP @ READ        /AND EXIT FROM SUBROUTINE
```

F. Miscellaneous Instructions

The miscellaneous group contains the instructions that operate on the Link. As mentioned earlier, the Link is a one-bit register which operates as an extension to the accumulator, so that when overflow or carryout occurs, the state of the Link changes. Since this change is only meaningful if the original state is known, the following instructions can be used on the Link:

CLL	Clear the Link: set it = 0
STL	Set the Link = 1
TLAC	Transfer the Link to AC bit 19. The Link and bits 0 - 18 of the AC are unchanged
TACL	Transfer bit 19 of the AC to the Link. Bit 19 is unchanged

Finally, the instruction STOP halts the stored program processor at the end of a memory cycle.

G. Input/Output Instructions

Any instruction which directs the computer to transfer data to or from an external device, such as a Teletype* is called an input/output or I/O instruction. Since these physical devices are much slower than the computer, they depend on the use of a flag to tell the computer when they are ready to transfer information. If the Teletype is not printing, the flag is up (=1). Thus, the Teletype printer flag goes up when it has finished printing the previous character,

*Teletype is a registered trademark of the Teletype Corporation.

indicating that it is ready to accept another. Since the Teletype keyboard and reader are physically linked, there is only one flag associated with them. It is normally down (=Ø) unless (a) the keyboard has been struck, or (b) there is tape in the reader and it is turned to READ. It should be noted that when the Teletype is switched to Line, striking a character does not cause it to be typed unless there is a program in operation to do this. The following code is necessary for the computer to read and type a character from the Teletype:

```
JMS ECHO          /READ AND TYPE A CHARACTER
ACCM KBD          /AND STORE IT AWAY FOR LATER USE
     .
     .
     .
ECHO, Ø
KWAIT,TTYRF       /WAIT FOR CHARACTER
JMP KWAIT
RDTTY             /READ IT INTO THE AC
PWAIT,TTYPF       /WAIT FOR PRINTER READY
JMP PWAIT
PRTTY             /PRINT IT
JMP @ ECHO        /AND EXIT WITH CHAR IN AC
```

To reiterate, the keyboard and low speed tape reader of the ASR-33 are physically linked on-line as well as off. The keyboard and printer mechanisms are entirely separate and must be driven under software control. Setting bit 14 of the IR during an I/O instruction causes the AC to be cleared before the transfer. This is generally done doing read instructions: RDTTY (44453), RHSR (44463). If a read command is issued without bit 14 set, the reader buffer is ORed into the AC.

H. Hardware Instructions

The following instructions cause the stored program to interact with devices normally associated with the wired program mode:

1. Digitizer Instructions

REDS	Reset data scaler; reset digitizer for new value
STDG	Start digitizing new number
RDG	Read digitized result. There must be a 20 usec delay between the STDG and the RDG commands. This is equivalent to five 4-usec instructions.

The REDS, STDG and RDG commands are micro-combinable. This is the same as saying that they can be issued simultaneously, causing all three events to happen during the same memory cycle. This is done by entering both on the same line in the program. Thus, the instruction RDG REDS STDG causes the value in the digitizer to be read into the AC, and a new digitization to begin.

2. Display Instructions

TACXD	Transfer AC bits 0 - 13 to the x-axis display register (14 bit). If 12 bit, AC bits 2-13 are used. The AC is complemented.
TACYD	Transfer AC bits 6 - 19 to the y-axis display register (14 bit). If 12 bit, AC bits 8-19 are used.
INTENS	Intensify the point held in the x and y registers
INCXD	Increment the x-axis display register
VDSH	Vertical display scale shift. Shifts the AC left by the number of positions set on the vertical display scale knob of the 1080

3. Sweep Ramp Instructions

In the NIC 1080, the sweep ramp is a separate digital to analog converter than that used by the display. It can be advanced and reset separately. Its instructions are:

RSWP	Reset sweep ramp
ASRMP	Advance (increment) sweep ramp

The timing of the sweep is, of course, set by the Dwell Time thumb wheel. Whenever a pulse from this clock is received, the dwell signal level goes high and a skip can be generated by the instruction DWSK/skip on dwell.

I. Automatic Arithmetic Instructions

The Automatic Arithmetic logic utilizes an additional register, called the Multiplier-Quotient Register or MQ. It is used to extend the accumulator to contain double precision integers. The instructions are described below.

TACMQ	Transfer the AC to the MQ, AC unaffected
TMQAC	Transfer the MQ to the AC, MQ unaffected
BITINV	Bit Invert the AC (used in Fourier transform routines) By bit inversion it is meant that bit 19 is interchanged with bit 0, bit 18 with bit 1 and so forth.
ZRAM	Zero the AC and MQ
MULT	The 20-bit number contained in the MQ is multiplied by the number contained in the location following the MULT instruction. The state of the AC is unimportant. At the completion of the instruction, the result is contained in the AC and MQ, with the high order part in the AC and the low order 20 bits in the MQ.

To multiply 3 by 4 the following code would be used.

```
MEMA (4                 /GET 4
ACCM MPLCND             /STORE IN LOCN FOLLOWING MULT
MEMA (3                 /GET 3
TACMQ                   /PLACE IN MQ
MULT                    /PERFORM MULTIPLICATION
MPLCND, ø               /LOCATION OF MULTIPLICAND
ACCM HIWORD             /HIGH WORD IN AC; STORE IT
TMQAC                   /GET LOW WORD
ACCM LOWORD             /AND STORE IT
```

RISH n — Right Integer shift. Right shift of AC, with least significant bits dropped off right end. ($ø \leq$ shifts $\leq 17_8$)

DIVD — Integer divide. The 38 bit dividend placed in the AC and MQ left shifted one place, is divided by the contents of the location following the instruction. At the conclusion of the operation, the quotient is in the MQ and the remainder in the AC. The remainder is left shifted one bit, the quotient is correct as it appears.

The reason for the shifting instructions is that it makes the treatment of numbers in the floating point format more efficient. This is utilized in the Floating Point Package version NIC 80/S-7005f,g. Only a 38 bit number can be accurately divided because there is no 20-bit quotient which can be produced by a 39 dividend and a 20-bit divisor.

For single precision division, especially in cases where the remainder is unimportant, the simple code below is representative:

/ROUTINE TO DIVIDE SINGLE PREC # DIVDND BY DIVSOR

```
MEMA DIVSOR             /GET THE DIVISOR
ACCM D1                 /PUT IT IN DIVD LOCATION + 1
MEMA DIVDND             /GET THE DIVIDEND
LASH 1                  /AND LEFT SHIFT IT
DIVD                    /DIVIDE BY D1
D1, ø
TMQAC                   /GET THE QUOTIENT, IGNORE REMAINDER
ACCM QUOT               /AND STORE IT
```

A typical double precision integer divide routine is given below:

```
/ROUTINE TO DIVIDE THE DOUBLE PRECISION NUMBER DBLDVH-DBLDVL
/BY DIVISOR

        CLL
        MEMA DBLDVL
        EXCT AC19
        STL             /SET LINK AS FLAG FOR OVERFLOW
        LASH 1          /SHIFT LEFT ONE PLACE
        TACMQ           /PLACE IN MQ
        MEMA DBLDVH     /GET HIGH-ORDER WORD
        LASH 1          /AND SHIFT IT
        EXCT L          /TEST LINK
        APOA            /INCREMENT AC IF DBLDVL OVERFLOWED
        DIVD            /EXECUTE THE DIVISION
        DIVSOR, nnnnnn  /NUMBER ALREADY STORED HERE FOR
                        /DIVISOR

        RISH 1
        ACCM REMNDR     /SHIFT AND SAVE THE REMAINDER
        TMQAC           /GET THE QUOTIENT
        ACCM QUOT       /AND SAVE IT
```

It should be noted that the hardware multiplication and division is unsigned, and that the signs must be tested for independently by the user.

During operations between numbers in the floating point format, the binary point is assumed between bits 19 and 18, and bit 19 is always zero since the division is unsigned. No shifts need be performed before division, and if bit 19 of the quotient is 1, the exponent is incremented by 1. A RISH 1 is then performed.

V. THE ASSEMBLER

A. Introduction

This section describes the actual use of the Assembler Program. It specifically refers to the tape NIC 80/S-7006E-B.

The Nicolet Assembler is especially suited for those with only low speed I/O equipment, but also recognizes the high speed reader and punch. It allows the user to create and store programs in data memory while occupying only locations 2000-4601. While it is a two-pass assembler, no tapes need be reloaded, since it operates entirely on the source text as stored in data memory. The release version allows only one data field of memory for text storage, but the size allowed can be changed using NICOBUG.

A typical storage arrangement would be:

0 - 1777	free for user programming
2000 - 4601	Assembler - Editor
4602 - 4777	free
5000 - 5215	Nicobug
5216 - 5347	free
5350 - 7577	Floating Point
7600 - 7777	Binary Loaders

B. Special Characters

The following special characters are recognized by the assembler:

/ Starts a comment. All characters beyond the slash except $ are ignored by the Assembler until a carriage return is encountered.

$ Signifies the end of the program. Cannot appear elsewhere in a tag, instruction or comment.

, Designates a tagged address; the first six characters following the previous Return and before the comma are taken as the name of the tag. A tag need not be six characters, but any after six are ignored.

(Designates immediate address mode.

space Used to separate operators from operands. There must not be a space between the operator and its suffix, if any.

@ Designates an indirect instruction. An indirect immediate instruction is flagged as an error.

* Designates the starting address of the code that follows.

= Allows the definition of symbols. For example, TTY2RF = 6434 defines the flag of a second Teletype, with I/O code 43.

The Assembler recognizes only printing characters as meaningful. All non-printing characters are ignored.

C. Syntax

1. Spaces may be used freely to improve legibility. They are not required anywhere except between a Group I instruction and its operand.

Their omission here will generate the error message IS (Illegal Suffix).

2. A comment may contain any character except a dollar sign.

3. All non-printing characters are ignored on input and are not stored.

4. Numbers can be entered only as positive octal integers.

5. Tags

a. Must be separated from the location contents by a comma. No space is required following the comma, but is recommended for legibility reasons.

b. Must start with an alphabetic character but may contain any combination of alphabetic and numeric characters after the first.

c. May contain up to six characters. Tags differing only in characters beyond the sixth are treated as identical.

d. May begin with three characters identical to a Group I instruction except when that tag is later referenced as a single address on a line by itself. In this case the message IS (Illegal Suffix) will be generated to remind the user of the potential ambiguity. The tag is assembled correctly as the address to which it refers in this case. (An example of a tag used in a list of addresses appears in the programming example in the next chapter.)

D. Addresses

The asterisk is used to indicate the starting address for a series of lines of code. The octal number following the asterisk is taken as that address. No test is made by the Assembler for code overflowing pages. The Assembler assumes that each line following the starting address contains code to be placed in successive locations unless (a) the line is blank - extra carriage returns may be inserted for legibility, (b) the line contains only a comment, (c) a new address is defined with an asterisk, or (d) only a symbol definition using an equals sign appears.

E. Loading and Startup

The Assembler tape is loaded using the standard Binary Loader. It is started at 2000 by

1. Place 2000 in the switch register (00 000 000 010 000 000 000)
2. Depress Load PC
3. Press Execute
4. Depress Continue
5. Press Execute

F. Assembler Commands

The program should start by typing a Return and Line Feed and type out ASSEMBLER. The program is then ready to receive commands:

R - Read a source tape into memory. The source tape is stored in the data memory and replaces any data or source programs stored there previously. The tape must end with a $-sign to return the Assembler to receive more commands. The tape is read from the Low Speed Reader (LSR) if the ADD button on the 1080 is depressed. If the SUBTRACT button is depressed the program is read from the High Speed Reader.

E - Perform an Error Analysis of the source code. The computer will seem to pause for up to several minutes if the program is lengthy. All errors will be typed out followed by their locations and the offending code. When the analysis is finished, a dollar sign will be typed. The errors detected are as follows:

IS - Illegal Suffix
II - Illegal Immediate
DU - Dubious Syntax
NL - No Label (or Tag) Found
DL - Duplicate Label - two identical tags are found

B - Punch out a binary tape of the program on the LSP. Prefixing the command by an H (see below) causes the output to be on the high speed punch. This tape can be read by the standard Binary Loader.

L - List the assembled code. The listing is divided into pages by a row of dashes. Each page is titled with the first line of code in the program. This line, therefore, should be a program title. The listing can be punched on the HSP by prefixing the L command with an H. Such a tape can then be listed off-line while the computer is doing something else.

H - Causes all output from the Assembler to be to the High Speed Punch, until the program returns to receive another command. All output includes the words ASSEMBLER and EDIT. To punch a binary tape on the HSP, type HB; to punch a listing type HL; to punch out the source tape type H,Control/E,W, and when the punching is done, FORM, before turning off the punch.

Control/E - Enter the Editor. Control/E (WRU) is produced by holding down the CTRL key and typing E. The program will respond by typing EDIT.

G. Editor Commands

W - Write out the source tape. If H was typed before entering the Editor, this will be done on the high speed punch.

A - Append more text to that already in memory. This differs from the Read command in that any previous text remains in storage, whereas the Read command erases all previous code. As with the Read command, if the ADD button is pressed, the code is input to the Low Speed Reader and if the SUBTRACT button is pressed input is from the HSR. A dollar sign must terminate the new code.

Pm nnnn - Print. The command P will be followed by either O or A depending on whether the current mode is Octal or Absolute. The user then types an octal number which represents the line to be printed. If the mode is octal, the line which has that octal address will be typed. If the mode is Absolute, each line, whether it contains code or not is numbered. The mode can be changed from Octal to Absolute or back again at any time by typing A or O immediately following the P (or D or I).

Dm nnnn - Delete line nnnn.

N - Type out the Next line following the one just operated on.

Im nnnn - Insert text just before line nnnn until Control/D(EOT) is typed.

Control/L - (FORM) Exit from the Editor and return to the Assembler.

The Editor remembers the last line number entered, so that it need not be retyped. Therefore to change line 37, the following is sufficient.

```
PA 37
MEMQ TEMP
DA                    (The underlined parts are typed by the EDITOR)
IA
MEMA TEMP
CTRL/D
```

TABLE II

Nicolet Assembler Command Summary

SA = 2000

R - Read in Source Tape
from LSR if ADD is depressed
from HSR if SUBTRACT is depressed

B - Punch Binary Tape

E - Error Analysis

L - List Assembled Code

H - Causes output of B, L, W to be on HSP

Control/E - (WRU) Enter Editor

Editor

W - Write out the Source tape

P - Print line
A - Absolute
O - Octal

I - Insert
EOT to exit (Control/D)

D - Delete

N - Print Next line

A - Append more text to buffer
from LSR if ADD is depressed
from HSR if SUBTRACT is depressed

FORM - (Control/L) exit to Assembler

H. Cautionary Notes

The Assembler can be restarted at any time by pressing STOP and restarting at location 2000.

Since neither Read nor Append echo at the keyboard, programs can best be created from scratch by typing R, Return, Return, $ to zero previous text and create a two-line blank program. One can then Insert all the text needed by starting with IA 1. The command IA Ø is not legal.

The first character read by the RDTTY command after turning on power is garbage, so it is good practice to start each program with an RDTTY command to clear the flag.

The Assembler recognizes the instruction

MEMA (TAG

as an instruction to get the relative address of the tagged location. The operand becomes the 10 least significant bits of the address TAG. While this has limited general use, it becomes extremely useful on page zero (locations 0-1777), where the relative and absolute addresses are identical and in calling the Floating Point routines.

VI. A PROGRAMMING EXAMPLE

A. Apologia

There is absolutely no way to learn assembly language programming short of doing it. No amount of reading or class attendance will be of any use until one plugs through a few examples which are of enough interest to encourage perseverance. Because personal interests and needs vary so widely, it is difficult to provide examples which will be stimulating to all readers. However, some general rules for good programming can be gleaned from this program and, hopefully, these at least will be rapidly assimilated.

B. Outline of Good Programming Practices

1. Define the task clearly

    a. In sentences, preferably on paper
    b. In flow charts, dividing the task into small, concise steps

2. Initialize all devices and registers

    a. Type a Carriage Return and Line Feed so the position of the Teletype carriage is known

b. Set all constants from an independent source, i.e., MEMA CSET, ACCM CTEMP. Zero all counters and pointers. Never assume that a location contains a zero. It won't unless you set it to zero.

c. Be sure the program is serially reusable.

d. Be sure the program is not "gullible." It should not expect only numbers in a certain range. Tests should be made for zero, negative and out of range values.

e. Read the keyboard buffer to clear it of power-on "garbage."

3. Write the program in small independent segments.

   a. Plan the exact relation between routines before you start writing them.

   b. Use recallable subroutines whenever possible.

   c. Be sure the program has no large volume point or part through which all the rest must flow.

4. Comment each section of the program thoroughly.

   a. For use in debugging.

   b. To aid others who will need to use it.

5. Obtain an accurate listing, and binary and source tapes whenever a new program verion is generated. The smallest changes may have the most astonishing effects. Be sure to date and number each version and discard obsolete versions to avoid confusion.

6. Most of the time spent in programming is spent in debugging. Be sure tl allow plenty of time for this. Write the program so that it is easily "debuggable." Plan on using NICOBUG (NIC 80/S-7110-B) whenever possible.

C. Definition of a Sample Program

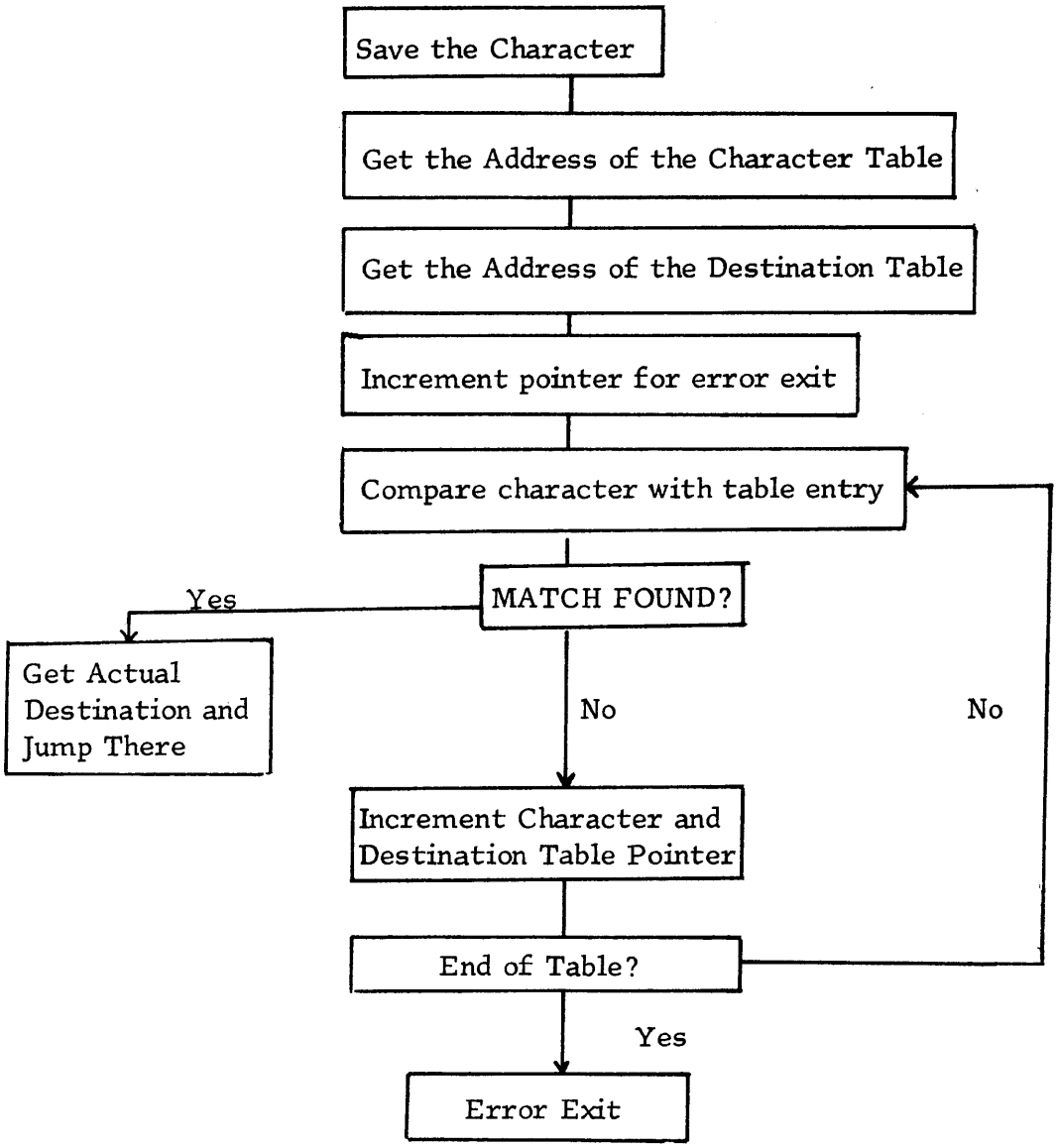
The problem is to search a table of characters for a match, and if a match is found, jump to a location found in a table of destinations. It must be called as a subroutine in the form

```
JMS TBSRCH      /ENTER WITH DESIRED CHARACTER IN AC
CTABLE          /STARTING ADDRESS OF CHARACTER TABLE
DTABLE          /STARTING ADDRESS OF DESTINATION TABLE
JMP ERROR       /RETURN HERE IF NO MATCH IN CHARACTER TABLE
```

If no match is found, the subroutine exits to a third location following the call. The tables may be of any length, but the value zero must be used as the last entry in the character table, indicating the end of the table.

This sort of routine is commonly used to route a Teletype input command to the proper calculation routine. It can also be used to cause extensive branching on a calculated result.

D. Flow Chart of the Program



E. Discussion

Several points should be noted in examining the listing on the following pages, not the least of which is the extensive commenting. It is absolutely essential that every program be well documented, even if the author is the only one who will ever use it. Often a program will need change or updating long after it was written. Figuring out the method (or "algorithm") that a program uses is extremely tedious work without thorough commenting.

A second feature of interest is the instruction ZERZ which is used to generate an unconditional skip in locations where the non zero case requires more instructions than the zero case. It literally means "load the zero test register with zero and skip if the zero test register is zero."

Still a third useful feature is that of calling a subroutine with arguments. This is roughly equivalent to the FORTRAN subroutine call CALL SUBR (A,B) where A and B are values transmitted to the subroutine. In this case the two values transmitted are the addresses of the two tables. These addresses can thus vary with each call and are page independent of the location of TBSRCH or its call.

The location TBSRCH on the call JMS TBSRCH contains the address of the location following the call, or 4. The content of 4 is the address of CTABLE, or 35. Incrementing TBSRCH causes it to point to location 5, where the address of the destination table is stored. Thus, MEMA @ TBSRCH allows access to the address of CDEST, and incrementing TBSRCH again causes it to point to the return address, where "?" is typed.

Finally, the routine at DSET converts the pointer to the destination address to the address itself. If the character B were struck, DESTN would contain the address 43. MEMA @ DESTN produces the actual location of routine B on the AC, or in this case 47.

F. Conclusion

The above described program contains a large number of concepts crucial to effective assembly language programming. When the reader thoroughly understands this example, he should start trying his own simple routines.

Some possible programs might include

(a) A program to type out octal numbers from memory
(b) A program to pack two ASCII characters into a single word and unpack them again. (How could you pack three?)
(c) A program to draw a box on the scope.

GOOD LUCK!

VII. TABLE SEARCHING ROUTINE

```
/TABLE SEARCHING ROUTINE
*0  /STARTING ADDRESS = 0

    /DRIVER ROUTINE
/ADDRESS - CONTENTS

      0   44453  START,  RDTTY   /CLEAR READER BUFFER
      1 2000011  S1, JMS CRLF  /INITIALIZE TTY CARRIAGE
      2 2000017    JMS ECHO  /READ AND ECHO CHARACTER
      3 2000052    JMS TBSRCH /SEARCH TABLE FOR LEGAL COMMAND
      4      35    CTABLE     /ADDRESS OF START OF TABLE OF CHARS
      5      42    CDEST     /START OF TABLE OF DESTINATI S
      6  110277    MEMA (277   /ERROR EXIT IF NO MATCH FOUND
      7 2000030    JMS TYPE    /TYPE ?
     10       1    JMP S1    /AND RETURN FOR NEW CHARACTER

     11       0  CRLF, 0  /TYPES CARRIAGE RETURN AND LINE FEED
     12  110215    MEMA (215  /ASCII FOR CR
     13 2000030    JMS TYPE
     14  110212    MEMA (212  /ASCII FOR LF
     15 2000030    JMS TYPE
     16 1000011    JMP @ CRLF  /EXIT

     17       0  ECHO, 0  /READS AND ECHOES CHAR FROM TTY
     20 2000023    JMS READ
     21 2000030    JMS TYPE
     22 1000017    JMP @ ECHO

     23       0  READ, 0  /READS CHAR FROM TTY
     24    6454  R1, TTYRF  /WAIT FOR CHAR TO BE STRUCK
     25      24    JMP R1
     26   44453    RDTTY  /READ IT
     27 1000023    JMP @ READ  /AND EXIT WITH CHAR IN AC

     30       0  TYPE, 0  /TYPES CHAR IN AC
     31    6444  T1, TTYPF  /WAIT FOR PRINTER TO BE READY
     32      31    JMP T1
     33    4443    PRTTY
     34 1000030    JMP @ TYPE  /EXIT

/TABLE OF LEGAL INPUT CHARACTERS (ASCII CODES)
     35     301  CTABLE,  301  /A
     36     302    302 /B
     37     303    303 /C
     40     304    304 /D
     41       0    0   /ZERO MARKS END OF THE TABLE

/TABLE OF DESTINATIONS (ALL DUMMY)
     42      46  CDEST, A
     43      47    B
     44      50    C
     45      51    D
```

```
/DUMMY DESTINATIONS -ALL STOPS IN THIS EXAMPLE
     46     5220  A, STOP
     47     5220  B, STOP
     50     5220  C, STOP
     51     5220  D, STOP


/TABLE SEARCHING SUBROUTINE

     52        0  TBSRCH, 0
     53 2404076    ACCM CHAR     /STORE AC IN LOCN "CHAR"
     54 3110052    MEMA @ TBSRCH     /GET CONTENTS OF ADDRS FOLLOWING CAL
     55 2404077    ACCM TABL     /STORE STRING ADDRS OF CHAR TABLE
     56 2124052    MPOM TBSRCH   /INCREMENT POINTER TO 2ND LOCN
     57 3110052    MEMA @ TBSRCH   / GET CONTENTS OF 2ND LOCN
     60 2404100    ACCM DESTN     /STORE ADDRESS OF DESTINATION TABLE
     61 2124052    MPOM TBSRCH      /INCREMENT POINTER FOR ERROR EXIT
     62 3112077  TBLOOP,  MEMAZ @ TABL  /GET CHAR FROM TABLE; SKIP IF =0
     63  162000    ZERZ        /UNCONDITIONAL SKIP
     64 1000052    JMP @ TBSRCH  /ERROR EXIT IF CHAR = 0
     65 2322076    M-AZ CHAR    /COMPARE WITH CHAR JUST TYPED
     66  162000    ZERZ        /CREATE ZERO AND SKIP
     67      73    JMP DSET   /IF MATCH, GO TO THAT LOCATION
     70 2124077    MPOM TABL  /INCREMENT TABLE POINTER
     71 2124100    MPOM DESTN  /INCR DESTINATION POINTER
     72      62    JMP TBLOOP  /TRY NEXT CHARACTER IN TABLE

/CONVERT POINTER TO DESTINATION TO ACTUAL DESTINATION
     73 3110100  DSET, MEMA @ DESTN
     74 2404100    ACCM DESTN  /AND STORE IN SAME LOCATION
     75 1000100    JMP @ DESTN  /THEN JUMP THERE


     76        0  CHAR, 0  /STORE ACTUAL INPUT CHARACTER
     77        0  TABL, 0  /POINTS TO CHARACTER TABLE
    100        0  DESTN, 0  /POINTS TO DESTINATION TABLE
```

APPENDIX I

ASCII CHARACTER CODES

The following list contains the 8-bit octal codes produced by standard ASR-33 Teletypes. This code is known as ASCII (American Standard Code for Information Interchange).

Character	Code	Character	Code
A	301	!	241
B	302	"	242
C	303	#	243
D	304	$	244
E	305	%	245
F	306	&	246
G	307	'	247
H	310	(	250
I	311	)	251
J	312	*	252
K	313	+	253
L	314	,	254
M	315	-	255
N	316	.	256
O	317	/	257
P	320	:	272
Q	321	;	273
R	322	<	274
S	323	=	275
T	324	>	276
U	325	?	277
V	326	@	300
W	327	[	333
X	330	\	334
Y	331	]	335
Z	332	↑	336
0	260	←	337
1	261	EOT	204
2	262	WRU	205
3	263	RU	206
4	264	BELL	207
5	265	TAB	211
6	266	Line Feed	212
7	267	FORM	214
8	270	Return	215
9	271	Space	240
		ALT MODE	375
		Rub Out	377

Several things should be noted about this code:

(a) The integers are biased by 260.
(b) The alphabet starts at 301.
(c) Most non-printing characters are less than 240.
(d) The CTRL key removes bit 6 from whatever key is typed: E = 305, CTRL/E = 205.
(e) The Shift key adds bit 4 to whatever key is typed: N = 316, SHIFT/N = ↑ = 336.
(f) Leader-Trailer (200) tape can be generated by holding down SHIFT, CTRL, REPT and P. Release in opposite order.
(g) The characters [and] are generated by SHIFT/K and SHIFT/M respectively.

APPENDIX II

MODIFYING THE ASSEMBLER (NIC-80/S-7106e-B)

The following modifications may be desirable for certain advanced users and can be easily accomplished from the switch register or by using Nicobug.

1. Changing the size of data storage. The locations SIZE, KMAX and LLIMIT define the storage area for text as one 4096 word stack starting at address 100000. These locations are located as follows:

	Address	Contents	
LLIMIT,	2342	100000	/STARTING ADDRESS
SIZE,	2343	10000	/ONE MEMORY STACK
KMAX,	4461	10000	/PAGE 4000 COPY OF SIZE /MUST BE MODIFIED IF SIZE /IS CHANGED

2. Changing the Append Command to echo at the Teletype. This can be used to generate error free source tapes or as an alternative method to Insert for adding new code.

Change location 3677 from	2024403	ONEM SUPSWT
to	2164403	ZERM SUPSWT

3. Changing the High Speed Reader-Punch I/O device codes. High speed equipment installed by users may utilize different I/O codes than those used by NIC. These are located as follows:

4565	44463	RHSR
4566	6464	HSRF
4522	6474	HSPF
4523	4473	PHSP

APPENDIX III

STATUS WORD INTERPRETATION

READOUT MEMORY ALLOCATION:

STARTING

1K (pushbutton in = "1")	Bit 0
2K (pushbutton in = "1")	Bit 1
4K (pushbutton in = "1")	Bit 2
8K (pushbutton in = "1")	Bit 3

SIZE

1K (pushbutton out = "1")	Bit 4
2K (pushbutton out = "1")	Bit 5
4K (pushbutton out = "1")	Bit 6
8K (pushbutton out = "1")	Bit 7

MEASURE MEMORY ALLOCATION:

STARTING

1K (pushbutton in = "1")	Bit 8
2K (pushbutton in = "1")	Bit 9
4K (pushbutton in = "1")	Bit 10
8K (pushbutton in = "1")	Bit 11

SIZE

1K (pushbutton out = "1")	Bit 12
2K (pushbutton out = "1")	Bit 13
4K (pushbutton out = "1")	Bit 14
8K (pushbutton out = "1")	Bit 15

INPUT DATA ADD/SUBTRACT:

ADD (Pushbutton in = "1")	Bit 16

VIEW INPUT/MEMORY/CONTINUOUS

INPUT (pushbutton out = "1")	Bit 17
MEMORY (pushbutton out = "1")	Bit 18

MEASURE (pushbutton in = "1")	Bit 19

APPENDIX IV

DECIMAL-OCTAL CONVERSION TABLE

	0	1	2	3	4	5	6	7	8	9
0000	0000	0001	0002	0003	0004	0005	0006	0007	0010	0011
0010	0012	0013	0014	0015	0016	0017	0020	0021	0022	0023
0020	0024	0025	0026	0027	0030	0031	0032	0033	0034	0035
0030	0036	0037	0040	0041	0042	0043	0044	0045	0046	0047
0040	0050	0051	0052	0053	0054	0055	0056	0057	0060	0061
0050	0062	0063	0064	0065	0066	0067	0070	0071	0072	0073
0060	0074	0075	0076	0077	0100	0101	0102	0103	0104	0105
0070	0106	0107	0110	0111	0112	0113	0114	0115	0116	0117
0080	0120	0121	0122	0123	0124	0125	0126	0127	0130	0131
0090	0132	0133	0134	0135	0136	0137	0140	0141	0142	0143
0100	0144	0145	0146	0147	0150	0151	0152	0153	0154	0155
0110	0156	0157	0160	0161	0162	0163	0164	0165	0166	0167
0120	0170	0171	0172	0173	0174	0175	0176	0177	0200	0201
0130	0202	0203	0204	0205	0206	0207	0210	0211	0212	0213
0140	0214	0215	0216	0217	0220	0221	0222	0223	0224	0225
0150	0226	0227	0230	0231	0232	0233	0234	0235	0236	0237
0160	0240	0241	0242	0243	0244	0245	0246	0247	0250	0251
0170	0252	0253	0254	0255	0256	0257	0260	0261	0262	0263
0180	0264	0265	0266	0267	0270	0271	0272	0273	0274	0275
0190	0276	0277	0300	0301	0302	0303	0304	0305	0306	0307
0200	0310	0311	0312	0313	0314	0315	0316	0317	0320	0321
0210	0322	0323	0324	0325	0326	0327	0330	0331	0332	0333
0220	0334	0335	0336	0337	0340	0341	0342	0343	0344	0345
0230	0346	0347	0350	0351	0352	0353	0354	0355	0356	0357
0240	0360	0361	0362	0363	0364	0365	0366	0367	0370	0371
0250	0372	0373	0374	0375	0376	0377	0400	0401	0402	0403
0260	0404	0405	0406	0407	0410	0411	0412	0413	0414	0415
0270	0416	0417	0420	0421	0422	0423	0424	0425	0426	0427
0280	0430	0431	0432	0433	0434	0435	0436	0437	0440	0441
0290	0442	0443	0444	0445	0446	0447	0450	0451	0452	0453
0300	0454	0455	0456	0457	0460	0461	0462	0463	0464	0465
0310	0466	0467	0470	0471	0472	0473	0474	0475	0476	0477
0320	0500	0501	0502	0503	0504	0505	0506	0507	0510	0511
0330	0512	0513	0514	0515	0516	0517	0520	0521	0522	0523
0340	0524	0525	0526	0527	0530	0531	0532	0533	0534	0535
0350	0536	0537	0540	0541	0542	0543	0544	0545	0546	0547
0360	0550	0551	0552	0553	0554	0555	0556	0557	0560	0561
0370	0562	0563	0564	0565	0566	0567	0570	0571	0572	0573
0380	0574	0575	0576	0577	0600	0601	0602	0603	0604	0605
0390	0606	0607	0610	0611	0612	0613	0614	0615	0616	0617
0400	0620	0621	0622	0623	0624	0625	0626	0627	0630	0631
0410	0632	0633	0634	0635	0636	0637	0640	0641	0642	0643
0420	0644	0645	0646	0647	0650	0651	0652	0653	0654	0655
0430	0656	0657	0660	0661	0662	0663	0664	0665	0666	0667
0440	0670	0671	0672	0673	0674	0675	0676	0677	0700	0701
0450	0702	0703	0704	0705	0706	0707	0710	0711	0712	0713
0460	0714	0715	0716	0717	0720	0721	0722	0723	0724	0725
0470	0726	0727	0730	0731	0732	0733	0734	0735	0736	0737
0480	0740	0741	0742	0743	0744	0745	0746	0747	0750	0751
0490	0752	0753	0754	0755	0756	0757	0760	0761	0762	0763

	0	1	2	3	4	5	6	7	8	9
0500	0764	0765	0766	0767	0770	0771	0772	0773	0774	0775
0510	0776	0777	1000	1001	1002	1003	1004	1005	1006	1007
0520	1010	1011	1012	1013	1014	1015	1016	1017	1020	1021
0530	1022	1023	1024	1025	1026	1027	1030	1031	1032	1033
0540	1034	1035	1036	1037	1040	1041	1042	1043	1044	1045
0550	1046	1047	1050	1051	1052	1053	1054	1055	1056	1057
0560	1060	1061	1062	1063	1064	1065	1066	1067	1070	1071
0570	1072	1073	1074	1075	1076	1077	1100	1101	1102	1103
0580	1104	1105	1106	1107	1110	1111	1112	1113	1114	1115
0590	1116	1117	1120	1121	1122	1123	1124	1125	1126	1127
0600	1130	1131	1132	1133	1134	1135	1136	1137	1140	1141
0610	1142	1143	1144	1145	1146	1147	1150	1151	1152	1153
0620	1154	1155	1156	1157	1160	1161	1162	1163	1164	1165
0630	1166	1167	1170	1171	1172	1173	1174	1175	1176	1177
0640	1200	1201	1202	1203	1204	1205	1206	1207	1210	1211
0650	1212	1213	1214	1215	1216	1217	1220	1221	1222	1223
0660	1224	1225	1226	1227	1230	1231	1232	1233	1234	1235
0670	1236	1237	1240	1241	1242	1243	1244	1245	1246	1247
0680	1250	1251	1252	1253	1254	1255	1256	1257	1260	1261
0690	1262	1263	1264	1265	1266	1267	1270	1271	1272	1273
0700	1274	1275	1276	1277	1300	1301	1302	1303	1304	1305
0710	1306	1307	1310	1311	1312	1313	1314	1315	1316	1317
0720	1320	1321	1322	1323	1324	1325	1326	1327	1330	1331
0730	1332	1333	1334	1335	1336	1337	1340	1341	1342	1343
0740	1344	1345	1346	1347	1350	1351	1352	1353	1354	1355
0750	1356	1357	1360	1361	1362	1363	1364	1365	1366	1367
0760	1370	1371	1372	1373	1374	1375	1376	1377	1400	1401
0770	1402	1403	1404	1405	1406	1407	1410	1411	1412	1413
0780	1414	1415	1416	1417	1420	1421	1422	1423	1424	1425
0790	1426	1427	1430	1431	1432	1433	1434	1435	1436	1437
0800	1440	1441	1442	1443	1444	1445	1446	1447	1450	1451
0810	1452	1453	1454	1455	1456	1457	1460	1461	1462	1463
0820	1464	1465	1466	1467	1470	1471	1472	1473	1474	1475
0830	1476	1477	1500	1501	1502	1503	1504	1505	1506	1507
0840	1510	1511	1512	1513	1514	1515	1516	1517	1520	1521
0850	1522	1523	1524	1525	1526	1527	1530	1531	1532	1533
0860	1534	1535	1536	1537	1540	1541	1542	1543	1544	1545
0870	1546	1547	1550	1551	1552	1553	1554	1555	1556	1557
0880	1560	1561	1562	1563	1564	1565	1566	1567	1570	1571
0890	1572	1573	1574	1575	1576	1577	1600	1601	1602	1603
0900	1604	1605	1606	1607	1610	1611	1612	1613	1614	1615
0910	1616	1617	1620	1621	1622	1623	1624	1625	1626	1627
0920	1630	1631	1632	1633	1634	1635	1636	1637	1640	1641
0930	1642	1643	1644	1645	1646	1647	1650	1651	1652	1653
0940	1654	1655	1656	1657	1660	1661	1662	1663	1664	1665
0950	1666	1667	1670	1671	1672	1673	1674	1675	1676	1677
0960	1700	1701	1702	1703	1704	1705	1706	1707	1710	1711
0970	1712	1713	1714	1715	1716	1717	1720	1721	1722	1723
0980	1724	1725	1726	1727	1730	1731	1732	1733	1734	1735
0990	1736	1737	1740	1741	1742	1743	1744	1745	1746	1747

	0	1	2	3	4	5	6	7	8	9
1000	1750	1751	1752	1753	1754	1755	1756	1757	1760	1761
1010	1762	1763	1764	1765	1766	1767	1770	1771	1772	1773
1020	1774	1775	1776	1777	2000	2001	2002	2003	2004	2005
1030	2006	2007	2010	2011	2012	2013	2014	2015	2016	2017
1040	2020	2021	2022	2023	2024	2025	2026	2027	2030	2031
1050	2032	2033	2034	2035	2036	2037	2040	2041	2042	2043
1060	2044	2045	2046	2047	2050	2051	2052	2053	2054	2055
1070	2056	2057	2060	2061	2062	2063	2064	2065	2066	2067
1080	2070	2071	2072	2073	2074	2075	2076	2077	2100	2101
1090	2102	2103	2104	2105	2106	2107	2110	2111	2112	2113
1100	2114	2115	2116	2117	2120	2121	2122	2123	2124	2125
1110	2126	2127	2130	2131	2132	2133	2134	2135	2136	2137
1120	2140	2141	2142	2143	2144	2145	2146	2147	2150	2151
1130	2152	2153	2154	2155	2156	2157	2160	2161	2162	2163
1140	2164	2165	2166	2167	2170	2171	2172	2173	2174	2175
1150	2176	2177	2200	2201	2202	2203	2204	2205	2206	2207
1160	2210	2211	2212	2213	2214	2215	2216	2217	2220	2221
1170	2222	2223	2224	2225	2226	2227	2230	2231	2232	2233
1180	2234	2235	2236	2237	2240	2241	2242	2243	2244	2245
1190	2246	2247	2250	2251	2252	2253	2254	2255	2256	2257
1200	2260	2261	2262	2263	2264	2265	2266	2267	2270	2271
1210	2272	2273	2274	2275	2276	2277	2300	2301	2302	2303
1220	2304	2305	2306	2307	2310	2311	2312	2313	2314	2315
1230	2316	2317	2320	2321	2322	2323	2324	2325	2326	2327
1240	2330	2331	2332	2333	2334	2335	2336	2337	2340	2341
1250	2342	2343	2344	2345	2346	2347	2350	2351	2352	2353
1260	2354	2355	2356	2357	2360	2361	2362	2363	2364	2365
1270	2366	2367	2370	2371	2372	2373	2374	2375	2376	2377
1280	2400	2401	2402	2403	2404	2405	2406	2407	2410	2411
1290	2412	2413	2414	2415	2416	2417	2420	2421	2422	2423
1300	2424	2425	2426	2427	2430	2431	2432	2433	2434	2435
1310	2436	2437	2440	2441	2442	2443	2444	2445	2446	2447
1320	2450	2451	2452	2453	2454	2455	2456	2457	2460	2461
1330	2462	2463	2464	2465	2466	2467	2470	2471	2472	2473
1340	2474	2475	2476	2477	2500	2501	2502	2503	2504	2505
1350	2506	2507	2510	2511	2512	2513	2514	2515	2516	2517
1360	2520	2521	2522	2523	2524	2525	2526	2527	2530	2531
1370	2532	2533	2534	2535	2536	2537	2540	2541	2542	2543
1380	2544	2545	2546	2547	2550	2551	2552	2553	2554	2555
1390	2556	2557	2560	2561	2562	2563	2564	2565	2566	2567
1400	2570	2571	2572	2573	2574	2575	2576	2577	2600	2601
1410	2602	2603	2604	2605	2606	2607	2610	2611	2612	2613
1420	2614	2615	2616	2617	2620	2621	2622	2623	2624	2625
1430	2626	2627	2630	2631	2632	2633	2634	2635	2636	2637
1440	2640	2641	2642	2643	2644	2645	2646	2647	2650	2651
1450	2652	2653	2654	2655	2656	2657	2660	2661	2662	2663
1460	2664	2665	2666	2667	2670	2671	2672	2673	2674	2675
1470	2676	2677	2700	2701	2702	2703	2704	2705	2706	2707
1480	2710	2711	2712	2713	2714	2715	2716	2717	2720	2721
1490	2722	2723	2724	2725	2726	2727	2730	2731	2732	2733

	0	1	2	3	4	5	6	7	8	9
1500	2734	2735	2736	2737	2740	2741	2742	2743	2744	2745
1510	2746	2747	2750	2751	2752	2753	2754	2755	2756	2757
1520	2760	2761	2762	2763	2764	2765	2766	2767	2770	2771
1530	2772	2773	2774	2775	2776	2777	3000	3001	3002	3003
1540	3004	3005	3006	3007	3010	3011	3012	3013	3014	3015
1550	3016	3017	3020	3021	3022	3023	3024	3025	3026	3027
1560	3030	3031	3032	3033	3034	3035	3036	3037	3040	3041
1570	3042	3043	3044	3045	3046	3047	3050	3051	3052	3053
1580	3054	3055	3056	3057	3060	3061	3062	3063	3064	3065
1590	3066	3067	3070	3071	3072	3073	3074	3075	3076	3077
1600	3100	3101	3102	3103	3104	3105	3106	3107	3110	3111
1610	3112	3113	3114	3115	3116	3117	3120	3121	3122	3123
1620	3124	3125	3126	3127	3130	3131	3132	3133	3134	3135
1630	3136	3137	3140	3141	3142	3143	3144	3145	3146	3147
1640	3150	3151	3152	3153	3154	3155	3156	3157	3160	3161
1650	3162	3163	3164	3165	3166	3167	3170	3171	3172	3173
1660	3174	3175	3176	3177	3200	3201	3202	3203	3204	3205
1670	3206	3207	3210	3211	3212	3213	3214	3215	3216	3217
1680	3220	3221	3222	3223	3224	3225	3226	3227	3230	3231
1690	3232	3233	3234	3235	3236	3237	3240	3241	3242	3243
1700	3244	3245	3246	3247	3250	3251	3252	3253	3254	3255
1710	3256	3257	3260	3261	3262	3263	3264	3265	3266	3267
1720	3270	3271	3272	3273	3274	3275	3276	3277	3300	3301
1730	3302	3303	3304	3305	3306	3307	3310	3311	3312	3313
1740	3314	3315	3316	3317	3320	3321	3322	3323	3324	3325
1750	3326	3327	3330	3331	3332	3333	3334	3335	3336	3337
1760	3340	3341	3342	3343	3344	3345	3346	3347	3350	3351
1770	3352	3353	3354	3355	3356	3357	3360	3361	3362	3363
1780	3364	3365	3366	3367	3370	3371	3372	3373	3374	3375
1790	3376	3377	3400	3401	3402	3403	3404	3405	3406	3407
1800	3410	3411	3412	3413	3414	3415	3416	3417	3420	3421
1810	3422	3423	3424	3425	3426	3427	3430	3431	3432	3433
1820	3434	3435	3436	3437	3440	3441	3442	3443	3444	3445
1830	3446	3447	3450	3451	3452	3453	3454	3455	3456	3457
1840	3460	3461	3462	3463	3464	3465	3466	3467	3470	3471
1850	3472	3473	3474	3475	3476	3477	3500	3501	3502	3503
1860	3504	3505	3506	3507	3510	3511	3512	3513	3514	3515
1870	3516	3517	3520	3521	3522	3523	3524	3525	3526	3527
1880	3530	3531	3532	3533	3534	3535	3536	3537	3540	3541
1890	3542	3543	3544	3545	3546	3547	3550	3551	3552	3553
1900	3554	3555	3556	3557	3560	3561	3562	3563	3564	3565
1910	3566	3567	3570	3571	3572	3573	3574	3575	3576	3577
1920	3600	3601	3602	3603	3604	3605	3606	3607	3610	3611
1930	3612	3613	3614	3615	3616	3617	3620	3621	3622	3623
1940	3624	3625	3626	3627	3630	3631	3632	3633	3634	3635
1950	3636	3637	3640	3641	3642	3643	3644	3645	3646	3647
1960	3650	3651	3652	3653	3654	3655	3656	3657	3660	3661
1970	3662	3663	3664	3665	3666	3667	3670	3671	3672	3673
1980	3674	3675	3676	3677	3700	3701	3702	3703	3704	3705
1990	3706	3707	3710	3711	3712	3713	3714	3715	3716	3717

	0	1	2	3	4	5	6	7	8	9
2000	3720	3721	3722	3723	3724	3725	3726	3727	3730	3731
2010	3732	3733	3734	3735	3736	3737	3740	3741	3742	3743
2020	3744	3745	3746	3747	3750	3751	3752	3753	3754	3755
2030	3756	3757	3760	3761	3762	3763	3764	3765	3766	3767
2040	3770	3771	3772	3773	3774	3775	3776	3777	4000	4001
2050	4002	4003	4004	4005	4006	4007	4010	4011	4012	4013
2060	4014	4015	4016	4017	4020	4021	4022	4023	4024	4025
2070	4026	4027	4030	4031	4032	4033	4034	4035	4036	4037
2080	4040	4041	4042	4043	4044	4045	4046	4047	4050	4051
2090	4052	4053	4054	4055	4056	4057	4060	4061	4062	4063
2100	4064	4065	4066	4067	4070	4071	4072	4073	4074	4075
2110	4076	4077	4100	4101	4102	4103	4104	4105	4106	4107
2120	4110	4111	4112	4113	4114	4115	4116	4117	4120	4121
2130	4122	4123	4124	4125	4126	4127	4130	4131	4132	4133
2140	4134	4135	4136	4137	4140	4141	4142	4143	4144	4145
2150	4146	4147	4150	4151	4152	4153	4154	4155	4156	4157
2160	4160	4161	4162	4163	4164	4165	4166	4167	4170	4171
2170	4172	4173	4174	4175	4176	4177	4200	4201	4202	4203
2180	4204	4205	4206	4207	4210	4211	4212	4213	4214	4215
2190	4216	4217	4220	4221	4222	4223	4224	4225	4226	4227
2200	4230	4231	4232	4233	4234	4235	4236	4237	4240	4241
2210	4242	4243	4244	4245	4246	4247	4250	4251	4252	4253
2220	4254	4255	4256	4257	4260	4261	4262	4263	4264	4265
2230	4266	4267	4270	4271	4272	4273	4274	4275	4276	4277
2240	4300	4301	4302	4303	4304	4305	4306	4307	4310	4311
2250	4312	4313	4314	4315	4316	4317	4320	4321	4322	4323
2260	4324	4325	4326	4327	4330	4331	4332	4333	4334	4335
2270	4336	4337	4340	4341	4342	4343	4344	4345	4346	4347
2280	4350	4351	4352	4353	4354	4355	4356	4357	4360	4361
2290	4362	4363	4364	4365	4366	4367	4370	4371	4372	4373
2300	4374	4375	4376	4377	4400	4401	4402	4403	4404	4405
2310	4406	4407	4410	4411	4412	4413	4414	4415	4416	4417
2320	4420	4421	4422	4423	4424	4425	4426	4427	4430	4431
2330	4432	4433	4434	4435	4436	4437	4440	4441	4442	4443
2340	4444	4445	4446	4447	4450	4451	4452	4453	4454	4455
2350	4456	4457	4460	4461	4462	4463	4464	4465	4466	4467
2360	4470	4471	4472	4473	4474	4475	4476	4477	4500	4501
2370	4502	4503	4504	4505	4506	4507	4510	4511	4512	4513
2380	4514	4515	4516	4517	4520	4521	4522	4523	4524	4525
2390	4526	4527	4530	4531	4532	4533	4534	4535	4536	4537
2400	4540	4541	4542	4543	4544	4545	4546	4547	4550	4551
2410	4552	4553	4554	4555	4556	4557	4560	4561	4562	4563
2420	4564	4565	4566	4567	4570	4571	4572	4573	4574	4575
2430	4576	4577	4600	4601	4602	4603	4604	4605	4606	4607
2440	4610	4611	4612	4613	4614	4615	4616	4617	4620	4621
2450	4622	4623	4624	4625	4626	4627	4630	4631	4632	4633
2460	4634	4635	4636	4637	4640	4641	4642	4643	4644	4645
2470	4646	4647	4650	4651	4652	4653	4654	4655	4656	4657
2480	4660	4661	4662	4663	4664	4665	4666	4667	4670	4671
2490	4672	4673	4674	4675	4676	4677	4700	4701	4702	4703

	0	1	2	3	4	5	6	7	8	9
3000	5670	5671	5672	5673	5674	5675	5676	5677	5700	5701
3010	5702	5703	5704	5705	5706	5707	5710	5711	5712	5713
3020	5714	5715	5716	5717	5720	5721	5722	5723	5724	5725
3030	5726	5727	5730	5731	5732	5733	5734	5735	5736	5737
3040	5740	5741	5742	5743	5744	5745	5746	5747	5750	5751
3050	5752	5753	5754	5755	5756	5757	5760	5761	5762	5763
3060	5764	5765	5766	5767	5770	5771	5772	5773	5774	5775
3070	5776	5777	6000	6001	6002	6003	6004	6005	6006	6007
3080	6010	6011	6012	6013	6014	6015	6016	6017	6020	6021
3090	6022	6023	6024	6025	6026	6027	6030	6031	6032	6033
3100	6034	6035	6036	6037	6040	6041	6042	6043	6044	6045
3110	6046	6047	6050	6051	6052	6053	6054	6055	6056	6057
3120	6060	6061	6062	6063	6064	6065	6066	6067	6070	6071
3130	6072	6073	6074	6075	6076	6077	6100	6101	6102	6103
3140	6104	6105	6106	6107	6110	6111	6112	6113	6114	6115
3150	6116	6117	6120	6121	6122	6123	6124	6125	6126	6127
3160	6130	6131	6132	6133	6134	6135	6136	6137	6140	6141
3170	6142	6143	6144	6145	6146	6147	6150	6151	6152	6153
3180	6154	6155	6156	6157	6160	6161	6162	6163	6164	6165
3190	6166	6167	6170	6171	6172	6173	6174	6175	6176	6177
3200	6200	6201	6202	6203	6204	6205	6206	6207	6210	6211
3210	6212	6213	6214	6215	6216	6217	6220	6221	6222	6223
3220	6224	6225	6226	6227	6230	6231	6232	6233	6234	6235
3230	6236	6237	6240	6241	6242	6243	6244	6245	6246	6247
3240	6250	6251	6252	6253	6254	6255	6256	6257	6260	6261
3250	6262	6263	6264	6265	6266	6267	6270	6271	6272	6273
3260	6274	6275	6276	6277	6300	6301	6302	6303	6304	6305
3270	6306	6307	6310	6311	6312	6313	6314	6315	6316	6317
3280	6320	6321	6322	6323	6324	6325	6326	6327	6330	6331
3290	6332	6333	6334	6335	6336	6337	6340	6341	6342	6343
3300	6344	6345	6346	6347	6350	6351	6352	6353	6354	6355
3310	6356	6357	6360	6361	6362	6363	6364	6365	6366	6367
3320	6370	6371	6372	6373	6374	6375	6376	6377	6400	6401
3330	6402	6403	6404	6405	6406	6407	6410	6411	6412	6413
3340	6414	6415	6416	6417	6420	6421	6422	6423	6424	6425
3350	6426	6427	6430	6431	6432	6433	6434	6435	6436	6437
3360	6440	6441	6442	6443	6444	6445	6446	6447	6450	6451
3370	6452	6453	6454	6455	6456	6457	6460	6461	6462	6463
3380	6464	6465	6466	6467	6470	6471	6472	6473	6474	6475
3390	6476	6477	6500	6501	6502	6503	6504	6505	6506	6507
3400	6510	6511	6512	6513	6514	6515	6516	6517	6520	6521
3410	6522	6523	6524	6525	6526	6527	6530	6531	6532	6533
3420	6534	6535	6536	6537	6540	6541	6542	6543	6544	6545
3430	6546	6547	6550	6551	6552	6553	6554	6555	6556	6557
3440	6560	6561	6562	6563	6564	6565	6566	6567	6570	6571
3450	6572	6573	6574	6575	6576	6577	6600	6601	6602	6603
3460	6604	6605	6606	6607	6610	6611	6612	6613	6614	6615
3470	6616	6617	6620	6621	6622	6623	6624	6625	6626	6627
3480	6630	6631	6632	6633	6634	6635	6636	6637	6640	6641
3490	6642	6643	6644	6645	6646	6647	6650	6651	6652	6653

	0	1	2	3	4	5	6	7	8	9
2500	4704	4705	4706	4707	4710	4711	4712	4713	4714	4715
2510	4716	4717	4720	4721	4722	4723	4724	4725	4726	4727
2520	4730	4731	4732	4733	4734	4735	4736	4737	4740	4741
2530	4742	4743	4744	4745	4746	4747	4750	4751	4752	4753
2540	4754	4755	4756	4757	4760	4761	4762	4763	4764	4765
2550	4766	4767	4770	4771	4772	4773	4774	4775	4776	4777
2560	5000	5001	5002	5003	5004	5005	5006	5007	5010	5011
2570	5012	5013	5014	5015	5016	5017	5020	5021	5022	5023
2580	5024	5025	5026	5027	5030	5031	5032	5033	5034	5035
2590	5036	5037	5040	5041	5042	5043	5044	5045	5046	5047
2600	5050	5051	5052	5053	5054	5055	5056	5057	5060	5061
2610	5062	5063	5064	5065	5066	5067	5070	5071	5072	5073
2620	5074	5075	5076	5077	5100	5101	5102	5103	5104	5105
2630	5106	5107	5110	5111	5112	5113	5114	5115	5116	5117
2640	5120	5121	5122	5123	5124	5125	5126	5127	5130	5131
2650	5132	5133	5134	5135	5136	5137	5140	5141	5142	5143
2660	5144	5145	5146	5147	5150	5151	5152	5153	5154	5155
2670	5156	5157	5160	5161	5162	5163	5164	5165	5166	5167
2680	5170	5171	5172	5173	5174	5175	5176	5177	5200	5201
2690	5202	5203	5204	5205	5206	5207	5210	5211	5212	5213
2700	5214	5215	5216	5217	5220	5221	5222	5223	5224	5225
2710	5226	5227	5230	5231	5232	5233	5234	5235	5236	5237
2720	5240	5241	5242	5243	5244	5245	5246	5247	5250	5251
2730	5252	5253	5254	5255	5256	5257	5260	5261	5262	5263
2740	5264	5265	5266	5267	5270	5271	5272	5273	5274	5275
2750	5276	5277	5300	5301	5302	5303	5304	5305	5306	5307
2760	5310	5311	5312	5313	5314	5315	5316	5317	5320	5321
2770	5322	5323	5324	5325	5326	5327	5330	5331	5332	5333
2780	5334	5335	5336	5337	5340	5341	5342	5343	5344	5345
2790	5346	5347	5350	5351	5352	5353	5354	5355	5356	5357
2800	5360	5361	5362	5363	5364	5365	5366	5367	5370	5371
2810	5372	5373	5374	5375	5376	5377	5400	5401	5402	5403
2820	5404	5405	5406	5407	5410	5411	5412	5413	5414	5415
2830	5416	5417	5420	5421	5422	5423	5424	5425	5426	5427
2840	5430	5431	5432	5433	5434	5435	5436	5437	5440	5441
2850	5442	5443	5444	5445	5446	5447	5450	5451	5452	5453
2860	5454	5455	5456	5457	5460	5461	5462	5463	5464	5465
2870	5466	5467	5470	5471	5472	5473	5474	5475	5476	5477
2880	5500	5501	5502	5503	5504	5505	5506	5507	5510	5511
2890	5512	5513	5514	5515	5516	5517	5520	5521	5522	5523
2900	5524	5525	5526	5527	5530	5531	5532	5533	5534	5535
2910	5536	5537	5540	5541	5542	5543	5544	5545	5546	5547
2920	5550	5551	5552	5553	5554	5555	5556	5557	5560	5561
2930	5562	5563	5564	5565	5566	5567	5570	5571	5572	5573
2940	5574	5575	5576	5577	5600	5601	5602	5603	5604	5605
2950	5606	5607	5610	5611	5612	5613	5614	5615	5616	5617
2960	5620	5621	5622	5623	5624	5625	5626	5627	5630	5631
2970	5632	5633	5634	5635	5636	5637	5640	5641	5642	5643
2980	5644	5645	5646	5647	5650	5651	5652	5653	5654	5655
2990	5656	5657	5660	5661	5662	5663	5664	5665	5666	5667

	0	1	2	3	4	5	6	7	8	9
3500	6654	6655	6656	6657	6660	6661	6662	6663	6664	6665
3510	6666	6667	6670	6671	6672	6673	6674	6675	6676	6677
3520	6700	6701	6702	6703	6704	6705	6706	6707	6710	6711
3530	6712	6713	6714	6715	6716	6717	6720	6721	6722	6723
3540	6724	6725	6726	6727	6730	6731	6732	6733	6734	6735
3550	6736	6737	6740	6741	6742	6743	6744	6745	6746	6747
3560	6750	6751	6752	6753	6754	6755	6756	6757	6760	6761
3570	6762	6763	6764	6765	6766	6767	6770	6771	6772	6773
3580	6774	6775	6776	6777	7000	7001	7002	7003	7004	7005
3590	7006	7007	7010	7011	7012	7013	7014	7015	7016	7017
3600	7020	7021	7022	7023	7024	7025	7026	7027	7030	7031
3610	7032	7033	7034	7035	7036	7037	7040	7041	7042	7043
3620	7044	7045	7046	7047	7050	7051	7052	7053	7054	7055
3630	7056	7057	7060	7061	7062	7063	7064	7065	7066	7067
3640	7070	7071	7072	7073	7074	7075	7076	7077	7100	7101
3650	7102	7103	7104	7105	7106	7107	7110	7111	7112	7113
3660	7114	7115	7116	7117	7120	7121	7122	7123	7124	7125
3670	7126	7127	7130	7131	7132	7133	7134	7135	7136	7137
3680	7140	7141	7142	7143	7144	7145	7146	7147	7150	7151
3690	7152	7153	7154	7155	7156	7157	7160	7161	7162	7163
3700	7164	7165	7166	7167	7170	7171	7172	7173	7174	7175
3710	7176	7177	7200	7201	7202	7203	7204	7205	7206	7207
3720	7210	7211	7212	7213	7214	7215	7216	7217	7220	7221
3730	7222	7223	7224	7225	7226	7227	7230	7231	7232	7233
3740	7234	7235	7236	7237	7240	7241	7242	7243	7244	7245
3750	7246	7247	7250	7251	7252	7253	7254	7255	7256	7257
3760	7260	7261	7262	7263	7264	7265	7266	7267	7270	7271
3770	7272	7273	7274	7275	7276	7277	7300	7301	7302	7303
3780	7304	7305	7306	7307	7310	7311	7312	7313	7314	7315
3790	7316	7317	7320	7321	7322	7323	7324	7325	7326	7327
3800	7330	7331	7332	7333	7334	7335	7336	7337	7340	7341
3810	7342	7343	7344	7345	7346	7347	7350	7351	7352	7353
3820	7354	7355	7356	7357	7360	7361	7362	7363	7364	7365
3830	7366	7367	7370	7371	7372	7373	7374	7375	7376	7377
3840	7400	7401	7402	7403	7404	7405	7406	7407	7410	7411
3850	7412	7413	7414	7415	7416	7417	7420	7421	7422	7423
3860	7424	7425	7426	7427	7430	7431	7432	7433	7434	7435
3870	7436	7437	7440	7441	7442	7443	7444	7445	7446	7447
3880	7450	7451	7452	7453	7454	7455	7456	7457	7460	7461
3890	7462	7463	7464	7465	7466	7467	7470	7471	7472	7473
3900	7474	7475	7476	7477	7500	7501	7502	7503	7504	7505
3910	7506	7507	7510	7511	7512	7513	7514	7515	7516	7517
3920	7520	7521	7522	7523	7524	7525	7526	7527	7530	7531
3930	7532	7533	7534	7535	7536	7537	7540	7541	7542	7543
3940	7544	7545	7546	7547	7550	7551	7552	7553	7554	7555
3950	7556	7557	7560	7561	7562	7563	7564	7565	7566	7567
3960	7570	7571	7572	7573	7574	7575	7576	7577	7600	7601
3970	7602	7603	7604	7605	7606	7607	7610	7611	7612	7613
3980	7614	7615	7616	7617	7620	7621	7622	7623	7624	7625
3990	7626	7627	7630	7631	7632	7633	7634	7635	7636	7637

	0	1	2	3	4	5	6	7	8	9
4000	7640	7641	7642	7643	7644	7645	7646	7647	7650	7651
4010	7652	7653	7654	7655	7656	7657	7660	7661	7662	7663
4020	7664	7665	7666	7667	7670	7671	7672	7673	7674	7675
4030	7676	7677	7700	7701	7702	7703	7704	7705	7706	7707
4040	7710	7711	7712	7713	7714	7715	7716	7717	7720	7721
4050	7722	7723	7724	7725	7726	7727	7730	7731	7732	7733
4060	7734	7735	7736	7737	7740	7741	7742	7743	7744	7745
4070	7746	7747	7750	7751	7752	7753	7754	7755	7756	7757
4080	7760	7761	7762	7763	7764	7765	7766	7767	7770	7771
4090	7772	7773	7774	7775	7776	7777	0000	0001	0002	0003
4100	0004	0005	0006	0007	0010	0011	0012	0013	0014	0015
4110	0016	0017	0020	0021	0022	0023	0024	0025	0026	0027
4120	0030	0031	0032	0033	0034	0035	0036	0037	0040	0041
4130	0042	0043	0044	0045	0046	0047	0050	0051	0052	0053
4140	0054	0055	0056	0057	0060	0061	0062	0063	0064	0065
4150	0066	0067	0070	0071	0072	0073	0074	0075	0076	0077
4160	0100	0101	0102	0103	0104	0105	0106	0107	0110	0111
4170	0112	0113	0114	0115	0116	0117	0120	0121	0122	0123
4180	0124	0125	0126	0127	0130	0131	0132	0133	0134	0135
4190	0136	0137	0140	0141	0142	0143	0144	0145	0146	0147
4200	0150	0151	0152	0153	0154	0155	0156	0157	0160	0161
4210	0162	0163	0164	0165	0166	0167	0170	0171	0172	0173
4220	0174	0175	0176	0177	0200	0201	0202	0203	0204	0205
4230	0206	0207	0210	0211	0212	0213	0214	0215	0216	0217
4240	0220	0221	0222	0223	0224	0225	0226	0227	0230	0231
4250	0232	0233	0234	0235	0236	0237	0240	0241	0242	0243
4260	0244	0245	0246	0247	0250	0251	0252	0253	0254	0255
4270	0256	0257	0260	0261	0262	0263	0264	0265	0266	0267
4280	0270	0271	0272	0273	0274	0275	0276	0277	0300	0301
4290	0302	0303	0304	0305	0306	0307	0310	0311	0312	0313
4300	0314	0315	0316	0317	0320	0321	0322	0323	0324	0325
4310	0326	0327	0330	0331	0332	0333	0334	0335	0336	0337
4320	0340	0341	0342	0343	0344	0345	0346	0347	0350	0351
4330	0352	0353	0354	0355	0356	0357	0360	0361	0362	0363
4340	0364	0365	0366	0367	0370	0371	0372	0373	0374	0375
4350	0376	0377	0400	0401	0402	0403	0404	0405	0406	0407
4360	0410	0411	0412	0413	0414	0415	0416	0417	0420	0421
4370	0422	0423	0424	0425	0426	0427	0430	0431	0432	0433
4380	0434	0435	0436	0437	0440	0441	0442	0443	0444	0445
4390	0446	0447	0450	0451	0452	0453	0454	0455	0456	0457
4400	0460	0461	0462	0463	0464	0465	0466	0467	0470	0471
4410	0472	0473	0474	0475	0476	0477	0500	0501	0502	0503
4420	0504	0505	0506	0507	0510	0511	0512	0513	0514	0515
4430	0516	0517	0520	0521	0522	0523	0524	0525	0526	0527
4440	0530	0531	0532	0533	0534	0535	0536	0537	0540	0541
4450	0542	0543	0544	0545	0546	0547	0550	0551	0552	0553
4460	0554	0555	0556	0557	0560	0561	0562	0563	0564	0565
4470	0566	0567	0570	0571	0572	0573	0574	0575	0576	0577
4480	0600	0601	0602	0603	0604	0605	0606	0607	0610	0611
4490	0612	0613	0614	0615	0616	0617	0620	0621	0622	0623