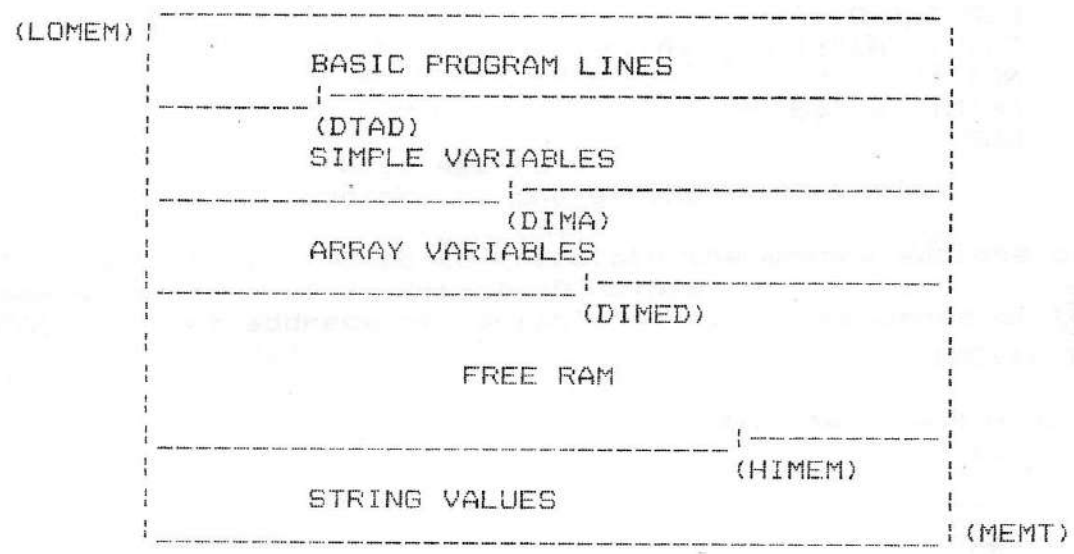
BIT90 APPLICATION NOTE NO.2

1. The memory layout in BIT90 computer is shown below:



BIT90 program lines occupy the low end of read/write memory starting at LOMEM. Numeric variables and string pointers are stored directly above the program lines.Arrays are stored above the simple variables.String values are stored at the top memory,starting at MEMT. The following memory locations are pointers for these area:

NAME	BYTE	ADDRESS	COMMENT
LOMEM	2	71E6H	LOWEST POINTER OF PROGRAM MEMORY
HIMEM	2	71E8H	HIGHEST POINTER OF AVAILABLE RAM
MEMT	2	71EAH	TOP MEMORY ADDRESS OF PROGRAM MEMORY
DIMED	2	7231H	END POINTER OF ARRAY AREA
DIMA	2	7233H	START POINTER OF ARRAY AREA
DTAD	2	7235H	START ADDRESS OF VARIABLE TABLE

When we edit or run the program,DIMED,DIMA and DTAD will automatically adjust upward,while HIMEM adjust downward.

2. STORING DATA ON CASSETTE

To save or load data by cassette,all variables must be declared first. The following program demonstrates how to store data on cassette,and how to recall them back in.

```
 10     CLEAR
 20     DIM A(10)
 30     HI=29233:REM DIMED
 40     LO=29237:REM DTAD
 50     I=0:ST=0:EN=0:D=0:G=0
 60     ST=PEEK( LO)+PEEK( LO+1):REM-START ADDRESS FOR BSAVE
 70     EN=PEEK( HI)+PEEK( HI+1):REM-END ADDRESS FOR BSAVE
 80     HOME
 90     PRINT "1.SAVE"
100     PRINT "2.LOAD"
110     INPUT "SELECT (1/2)",D
120     IF D<> 1AND D<> 2THEN GOTO 110
130     IF D=1THEN GOTO 140ELSE GOTO 230
140     PRINT :PRINT "ENTER DATA FOR ARRAY A(10)"
150     FOR I=1TO 10
160     PRINT I,
170     INPUT A(I)
180     NEXT
190     INPUT "ENTER DATA FOR VARIABLE G="
200     PRINT "SAVE";
210     BSAVE "DATA",ST,EN-ST:REM-SAVE SIMPLE AND ARRAY VARIABLES
```

*256

```
180       NEXT
190       INPUT "ENTER DATA FOR VARIABLE G="
200       PRINT "SAVE";
210       BSAVE "DATA",ST,EN-ST:REM-SAVE SIMPLE AND ARRAY VARIABLES
220       GOTO 290
230       PRINT "LOAD";:REM-RECALL DATA FROM CASSETTE
240       BLOAD "DATA"
250       FOR I=1TO 10
260       PRINT "A(";I;")=";A(I)
270       NEXT
280       PRINT "G=";G
290       END

READY

>RUN
1.SAVE
2.LOAD
SELECT (1/2)1

ENTER DATA FOR ARRAY A(10)
1         ?111
2         ?222
3         ?333
4         ?444
5         ?555
6         ?666
7         ?777
8         ?888
9         ?999
10        ?1000
ENTER DATA FOR VARIABLE G=575
SAVE
READY?Y
*TAPE DUMP:DATA.M
* END *

READY

>RUN
1.SAVE
2.LOAD
SELECT (1/2)2
LOAD
READY?
*TAPE LOAD:
DATA.M
* END *
A(1)=111
A(2)=222
A(3)=333
A(4)=444
A(5)=555
A(6)=666
A(7)=777
A(8)=888
A(9)=999
A(10)=1000
G=575

READY

>
```

3. DATA FORMAT:
   a. Numeric variables:

byte	usage
1	first ASCII code for name
2	second ASCII code for name
3	exponent with bit 7 set for negative value and bit 6 set for negative exponent

byte	usage
1	first ASCII code for name
2	second ASCII code for name
3	exponent with bit 7 set for negative value and bit 6 set for negative exponent.
4-6	mantissa.

b. String pointers:

byte	usage
1	first ASCII code with bit 7 set for name
2	second ASCII code for name
3	length
4	always set to 0
5-6	address of the string

The following formula is used to calculate the memory address of a variable:

memory address of a variable=DTAD+N*6

Where DTAD is start address of variable area,N is sequence of the variable which is first time used.

JL SUBROUTINES IN BIT90 BASIC:

NAME	ADDRESS(HEX.)	COMMENT
SETMOD	252BH	TRANSFER VDPR0 AND VDPR1 INTO VIDEO DISPLAY PROCESSOR
CIN	3303H	RETURNS A ASCII CODE WITH C FLAG SET
CLIN	346BH	GET A LINE FROM KEYBOARD INTO 'IOBUF'
COUT	36E0H	DISPLAY A ASCII CODE ON SCREEN AT CURRENT CURSOR POSITION
CLOUT	38ECH	CONVERT TOKEN CODE TO ASCII CHARACTERS AND DISPLAY ON THE SCREEN FOR A LINE IN IOBUF
SCRNWR	3870H	WRITE A ASCII CHARACTER IN ACCUMULATOR ON SCREEN
SCRNRD	387DH	READ A CHARACTER WITH POSITION IN HL REGISTER TO ACCUMULATOR
INIEP1	0380H	INITIALIZE MEMORY ARRANGEMENT AND EXPRESSION
LONUMB	0675H	LOCATE POINTER OF A LINE NUMBER
VRAMWR	30F9H	WRITE A BYTE FROM ACCUMULATOR TO VRAM WITH POINTER IN HL REGISTER
VRAMRD	30E2H	READ A BYTE FROM VRAM WITH POINTER IN HL INTO ACCUMULATOR
INITIAL	3278H	INITIALIZE VDP,PSG,AND RESET SYSTEM FLAG
LSTCHR	35C4H	FIND POSITION OF LAST CHARACTER TO HL REGISTER
CURSOR	3652H	DISPLAY CURSOR ON 'CURPOS' WHEN BIT 6 OF 'FLSCUR' IS SET
EOL	39D2H	END OF LINE CONTROL(MOVE CURSOR TO 1ST POSITION OF NEXT ROW)
FREA	2F80H	CHECK FREE RAM AND DISPLAY AN ERROR MESSAGE WHEN MEMORY IS INSUFFICENT

2. USEFUL 'PEEK' AND 'POKE' LOCATIONS:

NAME	BYTE	ADDRESS	COMMENT
WARM	5	7000H	WARM START FLAG (="BIT90")
TOKENE	2	7005H	EXTERNAL TOKEN TABLE POINTER THE EXTERNAL TOKEN TABLE IS ORGANIZED AS BELOW: (TOKENE) ASCII CODE OF RESERVED WORDS TOKEN CODE WITH 7TH BIT SET ENTRY POINTER OF PROGRAM . . . 0FFH (END OF TABLE)
INTVEC	2	7007H	VECTOR OF INTERRUPT SERVICE ROUTINE
NMIVEC	2	700AH	VECTOR OF NONMASKABLE INTERRUPT SERVICE ROUTINE
CURPOS	2	700FH	CURSOR POSITION
VDPR0	1	7014H	VDP REGISTER 0(SEE TABLE 1)
VDPR1	1	7015H	VDP REGISTER 1(SEE TABLE 1)
IMMBUF	128	7018H	IMMEDIATE COMMAND BUFFER
IOBUF	128	7098H	LINE BUFFER FOR KEYBOARD INPUT, EXPRESSION BUFFER
LINEH	2	71E2H	MAXIMUM LINE NUMBER OF CURRENT PROGRAM
ADDR	2	71E4H	MAXIMUM POINTER OF CURRENT PROGRAM
LOMEM	2	71E6H	LOWEST POINTER OF PROGRAM MEMORY
HIMEM	2	71E8H	HIGHEST POINTER OF AVAILABLE RAM
MEMT	2	71EAH	TOP MEMORY ADDRESS OF PROGRAM MEMORY
PROMPT	1	71F2H	PROMPT CHARACTER
CURLIN	2	71FBH	CURRENT LINE NUMBER
DIMED	2	7231H	END POINTER OF DIMENSION AREA
DIMA	2	7233H	START POINTER OF DIMENSION AREA
DTAD	2	7235H	START ADDRESS OF VARIABLE TABLE
CPYFC	1	7340H	NUMBERS OF CHARACTERS PER ROW FOR CENTRONIC PRINTER INTERFACE
GETKEY	3	734DH	CONSOLE INPUT DRIVER
GETLIN	3	7350H	CONSOLE LINE INPUT DRIVER

GETKEY	3	734DH	CONSOLE INPUT DRIVER
GETLIN	3	7350H	CONSOLE LINE INPUT DRIVER
DSPCHR	3	7353H	CONSOLE OUTPUT DRIVER
DSPLIN	3	7356H	CONSOLE LINE OUTPUT DRIVER

TABLE 1:

REGISTER 0	REGISTER 1	COMMENT
0	192*	DISPLAY MODE 0
0	200*	DISPLAY MODE 2 (LOW RESOLUTION)
2	192*	DISPLAY MODE 1 (HIGH RESOLUTION)
ANY	128	BLANKING DISPLAY (BORDER COLOR ONLY)

* THIS VALUE +1 CAUSES SPRITES TO BE MAGNIFIED (2X).
  THIS VALUE +2 SELECTS SIZE 1 SPRITES (16X16 BITS).
  THIS VALUE +3 SELECTS SIZE 1 SPRITES AND MAGNIFIED.

NOTES:
1. IF SIZE 1 SPRITES ARE SELECTED, SPRITE PATTERN NUMBER MUST START ON 4XN. WHERE N IS FROM 0 TO 31.
2. EACH SCAN LINE ON SCREEN CAN DISPLAY MAX. 4 SPRITES.
3. AFTER POKING VALUES INTO VDPR0 AND VDPR1,ROUTINE 'SETMOD' (252BH) MUST BE CALLED.

USEFUL 'IN' AND 'OUT' ADDRESSES:

PORT ADDRESS	IN/OUT	COMMENT
255	OUT	TONE OR NOISE GEN. CONTROL
191	OUT	WRITE CONTROL BYTES TO VDP
190	IN	READ DATA FROM VIDEO RAM
	OUT	WRITE DATA INTO VIDEO RAM

A. TONE AND NOISE GENERATORS:
THE OUTPUT FREQUENCY IS DEFINED BY THE FOLLOWING EQUATION:

FREQ=3.58/(32*N)

WHERE N IS A VALUE RANGE FROM 1 TO 1023,AND THE RESULT FREQUENCY WILL BE FROM 109 Hz TO 111,875 Hz.
TO CONTROL OUTPUT OF TONE GENERATOR,THE FOLLOWING BYTES MUST BE SPECIFIED TO PORT 255.

1. UPDATE FREQUENCY:
FIRST BYTE=N-INT(N/16)*16+128+32*CH
SECOND BYTE=INT(N/16)

2. UPDATE ATTENUATOR:
SINGLE BYTE=ATN+128+32*CH+16

3. UPDATE NOISE SOURCE:
SINGLE BYTE=NSE+128+32*3

WHERE N: DIVIDER VALUE RANGED FROM 1 TO 1023.
ATN: ATTENUATION CONTROL BYTE RANGE FROM 0(MAX. VOLUME)

WHERE N: DIVIDER VALUE RANGED FROM 1 TO 1023.
ATN: ATTENUATION CONTROL BYTE RANGE FROM 0(MAX. VOLUME) TO 15(MIN. VOLUME)
CH: CHANNEL NUMBER,0 TO 2 FOR TONE AND 3 FOR NOISE SOURCE
NSE: NOISE CONTROL BYTE,0 TO 3 FOR PERIODIC NOISE AND 4 TO 7 FOR WHITE NOISE:

NSE	SHIFT RATE
0(4)	3.58/512
1(5)	3.58/1024
2(6)	3.58/2048
3(7)	OUTPUT OF CHANNEL 2

B. ACCESS VIDEO MEMORY:
THE VIDEO MEMORY IS ARRANGED AS BELOW:

ADDRESS	USAGE
0-2047	PATTERN TABLE FOR SCREEN MODE 0
0-6143	PATTERN TABLE FOR SCREEN MODE 1
2048-6143	PATTERN TABLE FOR SCREEN MODE 2
6144-7167	TEXT (SCREEN MODE 0)
7168-8191	NAME TABLE FOR SCREEN MODE 2
8192-8223	COLOR TABLE FOR SCREEN MODE 0
8192-14335	COLOR TABLE FOR SCREEN MODE 1
14336-15459	PATTERN TABLE FOR SPRITE
15360-16383	SPRITE ATTRIBUTE TABLE

TO READ FROM OR WRITE INTO VIDEO RAM:(AD=VIDEO RAM ADDRESS)
STEP 1. SET UP LOW BYTE ADDRESS
AD-INT(AD/256)*256
STEP 2. SET UP HIGH BYTE ADDRESS
INT(AD/256)
THE ABOVE TWO BYTES ARE OUTPUT TO PORT 191 BY USING 'OUT' STATEMENT.
STEP 3. READ FROM OR WRITE INTO PORT 190 BY 'IN' OR 'OUT' STATEMENT.
NOTE THAT SEQUENTIAL DATA READS OR WRITES REQUIRE ONLY STEP 3 SINCE ADDRESS IS ALREADY SET UP.

. APPLICATIONS:
A. SELECT SPRITE SIZE AND MAGNIFICATION:

```
 10      A=7*4096+16+5 :REM CALCULATE POKE ADDRESS FOR VDPR1
 15      REM DEFINE PATTERNS TO BE A FLOWER
 20      CALL SPRPTN(124,"C6EE6E106CEEDC10")
 30      CALL SPRPTN(125,"180CC7427A3A0703")
 40      CALL SPRPTN(126,"00000822085D0822")
 50      CALL SPRPTN(127,"488000061CF0C080")
 60      FOR B=192 TO 195:REM REPEAT DEMO FOR 4 SPRITE MODES
 70      POKE A,B:REM POKE INTO VDPR1
 80      CALL 2*4096+5*256+2*16+11:REM CALL SETMOD
 90      CALL SPRITE(40,40,124,15,1)
100      FOR J=1 TO 500:NEXT J:REM DELAY
110      NEXT B
120      GOTO 60
```

B. CURSOR POSITION CONTROL:

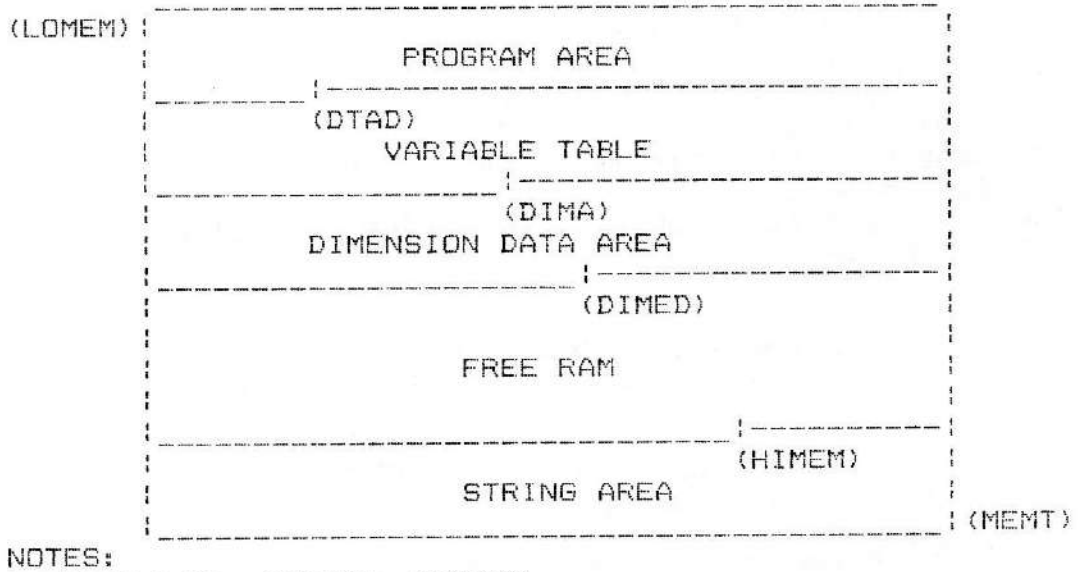
```
 10      HOME
 20      A=7*4096+15
 30      DIM A(10)
 40      CUR=34:REM INITIALIZE CURSOR POSITION
 50      FOR I=1 TO 10:REM ASSIGN VALUES TO ARRAY A(I)
 55      REM DEFINE CURSOR POSITION
 60      POKE A,CUR-INT(CUR/256)*256:POKE A+1,INT(CUR/256)
 70      PRINT "A(";I;")=";
 80      CUR=CUR+32:REM CALCULATE NEXT CURSOR POSITION
 90      NEXT
100      CUR=45:REM INITIALIZE INPUT CURSOR POSITION
110      FOR I=1 TO 10
120      POKE A,CUR-INT(CUR/256)*256:POKE A+1,INT(CUR/256)
130      INPUT ANS$:A(I)=VAL(ANS$):REM ASSIGN VALUES TO A(I)
140      CUR=CUR+32:REM POINT TO NEXT INPUT POSITION
150      NEXT
```

```
140     CUR=CUR+32:REM POINT TO NEXT INPUT POSITION
150     NEXT
160     FOR I=1 TO 10:CALCULATE SUM OF THE ARRAY
170     SUM=SUM+A(I)
180     NEXT
190     PRINT "SUM=";SUM
200     POKE 29160,PEEK(29162):POKE 29161,PEEK(29163):REM DEALLOCATE
        INPUT STRING:(MEMT)-->(HIMEM)
210     SUM=0:GOTO 40
```

C. TONE GENERATOR CONTROL

```
10      REM      *** BIT90 TONE TEST ***
20      PA=255
30      OUT PA,159:OUT PA,191:OUT PA,223:OUT PA,255:REM CLEAR OUTPUTS
40      REM TONE OUTPUT IS 3.58/32*I:ENTER I TO TEST TONE OUTPUT
50      PRINT "DIVIDER VALUE=";:INPUT " ",I
60      B=INT(I/16)
70      A=I-B*16+128
80      OUT PA,A:OUT PA,B:REM SET 2 BYTE TONE VALUE TO TONE GEN.
90      OUT PA,144:REM TURN ON TONE 1 OUTPUT
100     FOR X=0 TO 1000:NEXT:REM DELAY
110     GOTO 120
```

. DATA STRUCTURE:



NOTES:

1. FREE RAM= (HIMEM)-(DIMED)
2. STRING AREA IS ALLOCATED BY:
   A. THE RESULTS OF STRING EXPRESSIONS,SUCH AS '&' OR CHR$(N),...ETC.
   B. STRINGS IN 'MUSIC' STATEMENTS.
3. VARIABLE TABLE,DIMENSION AREA AND STRING AREA ARE DEALLOCATED BY 'CLEAR' STATEMENT.