The Journal of National Chiao Tung University
Vol. 7, April 1980
pp. 27-44

設計與製作一程式可以控制之電視遊樂系統
Design and Implementation of a Programmable TV Game System

蔡中川　楊維邦　Jong-Chuang Tsay and Wei-Pang Yang

Department of Computer Engineering

(Received December 19, 1979)

Abstract——In this paper, a Programmable TV Game System, which has been designed and implemented on the Motorola M6800 micro-computer, is described. The design of the hardware is based on the M6800 microcomputer and the MTX-256**2 TV interface. The design of the software is centered on a Graphic FORTRAN language. It is a FORTRAN language with the addition of Graphic Statements for facilitating the description of a graphic entity such as a point or a line. Thus graphic-oriented statements are provided and the full capabilities of FORTRAN language are preserved in the Graphic FORTRAN language, and it is not difficult to write a Graphic FORTRAN program to specify a TV game. The structure of the Programmable TV Game System and its implementation method are presented in this paper. Some samples of TV game program and their running results are given for references.

I.Introduction

A host of TV game, many using microcomputers, promises the public more stimulating fun on leisure time [1]. The Electronic Industries Association estimates that some 10 million TV games were sold last year. With figures like these, integrated circuit chip suppliers such as General Instrument, Texas Instruments, National Semiconductor, and Rockwell are hard pressed just to keep up with this anticipated demands. Most games provide a host of devices to make them more interesting, such as different paddle sizes, choice of ball speeds, etc. The major consideration here is to minimize eventual boredom. A new generation of TV games will surely overcome this boredom: programmable TV games.

Fairchild Camera and Instrument developed programable TV games at end of 1976 with its "Viedo Entertainment". Here, one inserts what appers to be a tape cartridge (actually it is a solid-state circuit that contains game programs on ROM) into a slot on the game machine to select a particular game or games on TV receiver. Other leading manufacture on TV game is the General Instrument. The General Instrument game repertoire offers game manufactures a choice of approaches to the marketplace: GIMINI dedicated game chips and the GIMINI cassette programmable game set. The programmable game set based on a variant of GI's CP1600, an advanced 16 bit single chip microprocessor, provides maximum flexibility in implementing a programmable system. But how can we develop the program for a TV game?

In this paper, we describe and provide a language,which is easy and flexible to be used in developing the program for a TV game, for the user. This language is called Graphic FORTRAN language. It is a FORTRAN language with the addition of Graphic Statements for facilitating the description of a graphic entity such as a point or a line. In Section II, we describe the system hardware and the design of related hardware interface. In Section III, we present the software flowchart of the system and discuss the methods used to implement the system including graphic primitives and preprocessor design. In Section IV, we present three sample Graphic FORTRAN programs and their running results. Finally, Conclusion is made.

II. System Hardware Implementation

The system hardware we used to develop a TV game is shown in Fig. 1. The central part of the system is an M6800 microcomputer which has been attached by three standard peripherals: console, printer, and floppy disk [2]. These peripherals are provided for facilitating the development of user programs. In addition to these I/O devices, there are other devices attached to the microcomputer. These devices are a TV display, a speaker, a pair of control knobs and shot buttons, a reset button, and a timer.

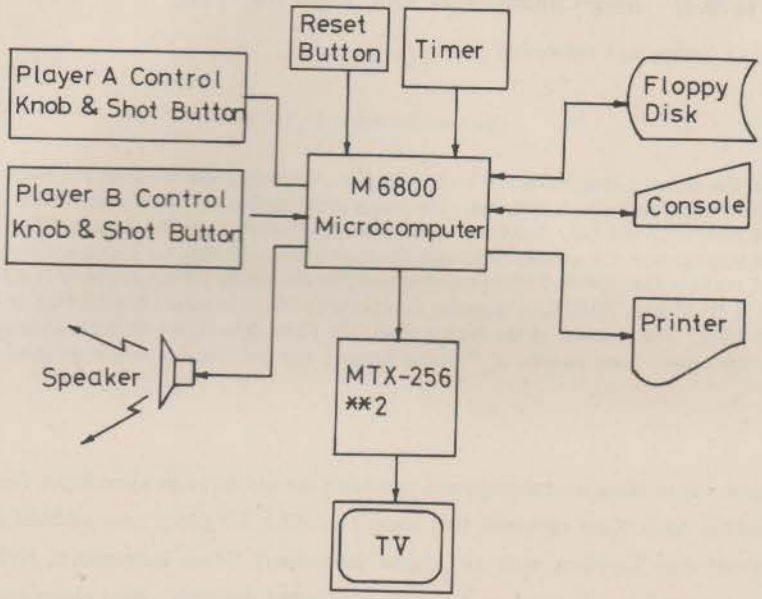


Fig. 1 Hardware Environment

The TV display is attached to the microcomputer via a MTX-256**2 interface. The MTX-256**2 is a modular system designed specifically for microcomputer graphics applications [3,4]. Fig. 2 shows the block diagram of the MTX-256**2.

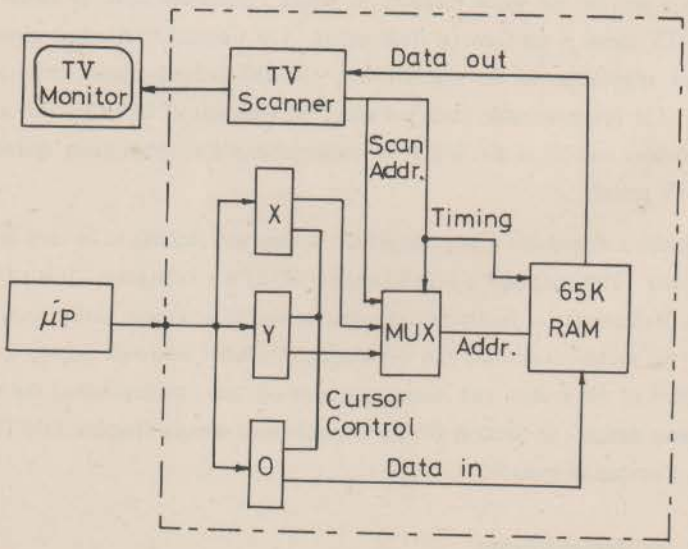


Fig. 2 Block Diagram of MTX-256**2

The speaker is used as an output device for generating sound. To let the TV game have a realistic effect, the system has a sound generator. It can be implemented as in Fig. 3. The software program can set 3 output bits of PIA and select one of the 8 sound generator circuits via a 4051 decoder. With reference to Electronic & Radio TV Technic [5], we implement two sound generators; one sounds like "whistle" of a police car, and the other one generates the sound of "boo-boo".

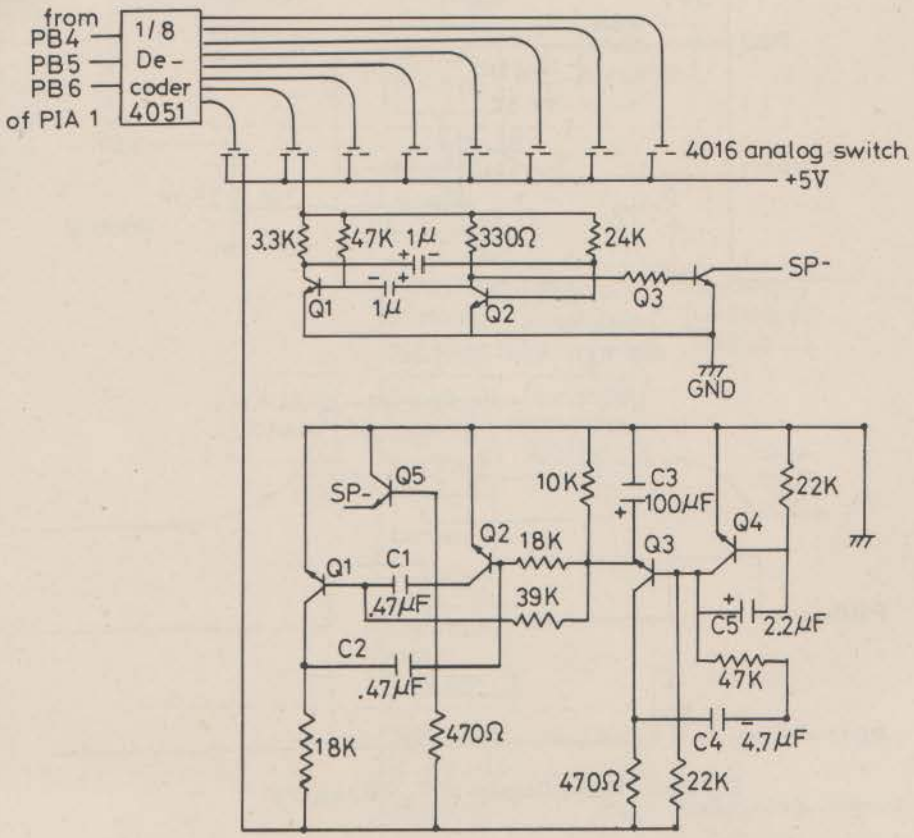


Fig. 3 Sound Generator Circuit

Control knobs are in pairs to allow for two players. Set values of the control knob can be read into the microcomputer by calling appropriate subroutines. A cheapest and handiest conversion circuit is shown in Fig. 4(a), where LM311 is a voltage comparator. PB0 and PB2 are assigned as output pins of PIA1, while PB1 and PB3 are assigned as input. The operation principle of the circuit can be understood by referring to Fig. 4(b): the timing diagram of V_c, PB0, and PB1. The width of the pulse on PB0 indicates the value of V_{in}

In general, a TV game system should provide interrupt functions. It can be used in many cases. For example, the proceeding submarine is shooting torpedo suddenly; the game is reset; or the graphic symbol (e.g., a ball) is moved one step. Thus, we design four external interrupt request circuits. They are: reset, A shot, B shot, and timer interrupt.

All the specially designed devices are attached to the microcomputer via a Peripheral Interface Adapter (PIA) chip.

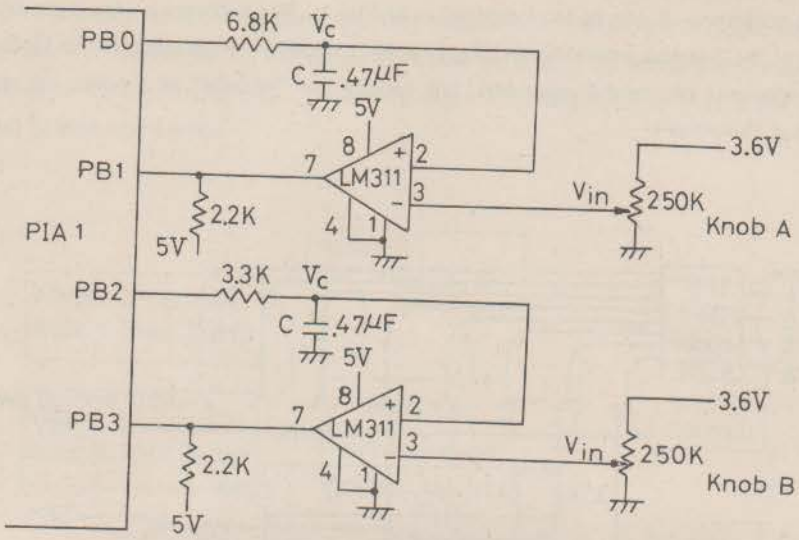


Fig. 4(a) Knob interface

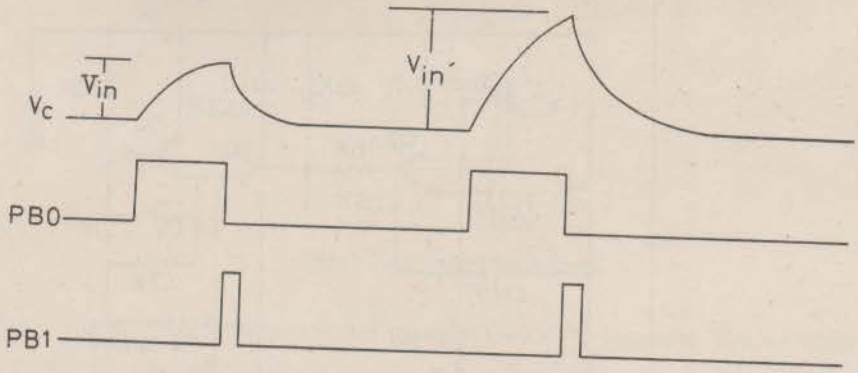


Fig. 4(b) Timing Diagram of V_c, PBO and PB1

III. System Software Implementation

The software block diagram of the Programmable TV Game System is shown in Fig. 5. The Graphic FORTRAN program includes FORTRAN program and Graphic Statements. The Preprocessor is used to translate Graphic FORTRAN program into FORTRAN source which is compilable by the Resident FORTRAN Compiler. The result of the compilation is a FORTRAN object.

Since some operation is easier to be done in assembly language, part of this system program is written in assembly language. The assembly language source is translated by the macro assembler into assembly language object. This object is then linked with FORTRAN object and FORTRAN library to generate executable game program. This program, after running, will display the graphic symbol on TV and accept the control from the external control circuits: control knob, shot button, reset button, and timer.

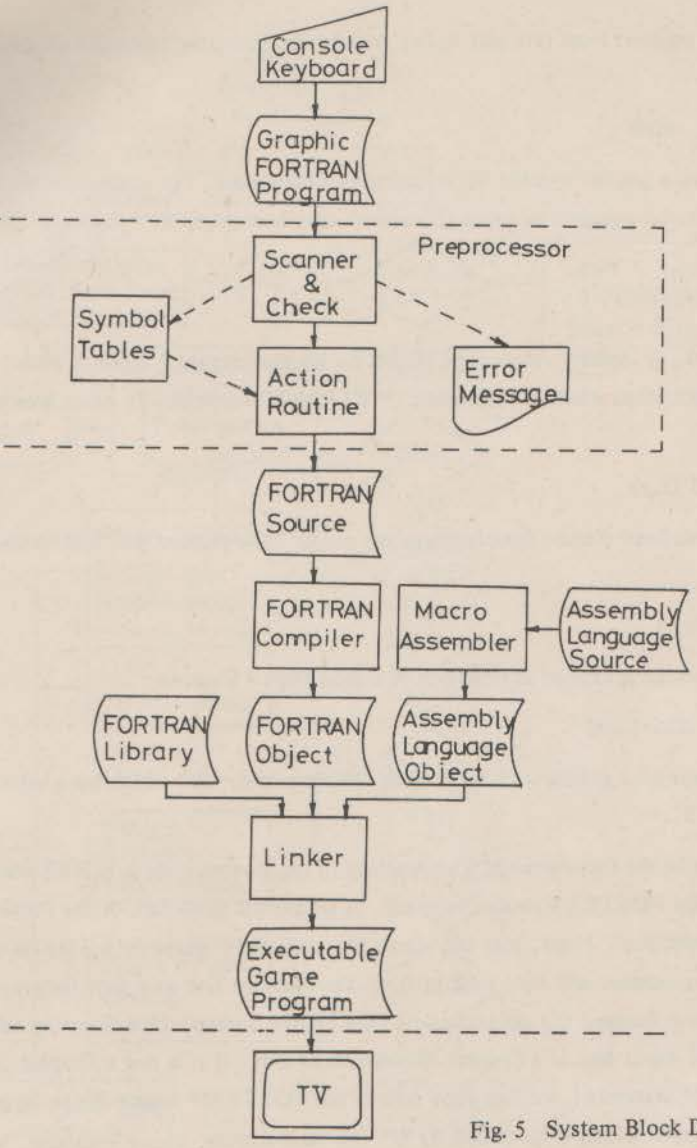


Fig. 5 System Block Diagram

In order to understand the details of the Preprocessor, let us describe the Graphic FORTRAN language first. It is a FORTRAN language embedded with Graphic Statements [6]. The Graphic Statements which we have designed for the system have the following formats :

1) *POINT (x,y)[.int]

Plot or erase a point at location (x,y) depending on intensity 'int' to be 1 or not. 'int' has a default value of 1. The addressable points of the display are [0,0] to [255, 255].

2) *LINE (x0,y0), (x1, y1) { ,(x_i, y_i)} [.int]

Draw line segments from (x0, y0) to (x1, y1), then from (x1, y1) to (x2, y2), and so on, if any. There are a number of methods to draw a line segment. We choose a method derived from Jordan's algoritihm [7].

3) *DASHLINE (x0, y0), (x1, y1), d1, d2 [.int]

Draw a dash line segment from (x0, y0) to (x1, y1) by repeating the operation of displaying d1 points and blanking d2 points.

4) *STRANS sx, sy, rotate

User can transform a graphic symbol by including this statement. For example, if we choose sx=sy=2, rotate=60, then the symbol is to be enlarged to twice its original size, and rotate 60^{o} clockwise about the origin of the coordinate system.

5) *RTRANS

The "*STRANS sx, sy, rotate" statement effects on all the graphic symbol(s) which are described by following statements. The effect stops when the statement "*RTRANS" appears. In other words, *RTRANS is equivalent to *STRANS 1,1,0.

6) *DISPLAY pic AT (x,y)

The 'pic' is a predefined picture (graphic symbol) name. The picture 'pic' will be displayed on the TV screen at location (x,y).

7) *ERASE pic AT (x,y)

The picture 'pic' which is located at (x,y) will be erased from TV screen.

8) *VECLINE (x,y), dir,n [.int]

Draw a line segment of n points from (x,y) with direction 'dir'. 'dir', which has a value of 0 - 15, specifies one of the eight directions [3].

Now we can describe the Preprocessor. The function of the Preprocessor is to translate a Graphic FORTRAN program into a compilable FORTRAN source program. A simplified flowchart of the Preprocessor is given in Fig. 6. In the "Game File Selection" block, user can select the desired TV game by keying its corresponding file name on the console. The Preprocessor will then read in from the file one line at a time for preprocessing. If an end of file is encountered, we have finished the generation of FORTRAN source. If it is not an end, the Preprocessor will go to check whether the input line is a Graphic Statement, or not. If it is not a Graphic Statement, then it must be a standard FORTRAN Statement, we just copy it into the FORTRAN source file to be generated. If it is a Graphic Statement, then the input which is identified by '*' on its columns 6 and 7 should be preprocessed as follows:

Firstly, SCANNER subroutine is called. It just parses the statement, recongnizes the basic elements and calls appropriate action routines. If there is anything wrong, error messages will be displayed on console; otherwise, a new line will be created to FORTRAN source file. The preprocessor then goes back to read in next Graphic FORTRAN statement.

The function of the Scanner is to create a symbol table (SYMT) by parsing the Graphic FORTRAN statement ; recognizing , the basic element; then jumping to appropriate action routines. For example, the symbol table created for the statement. "DISPLAY BALL AT (x,y)" is shown in Fig. 7.

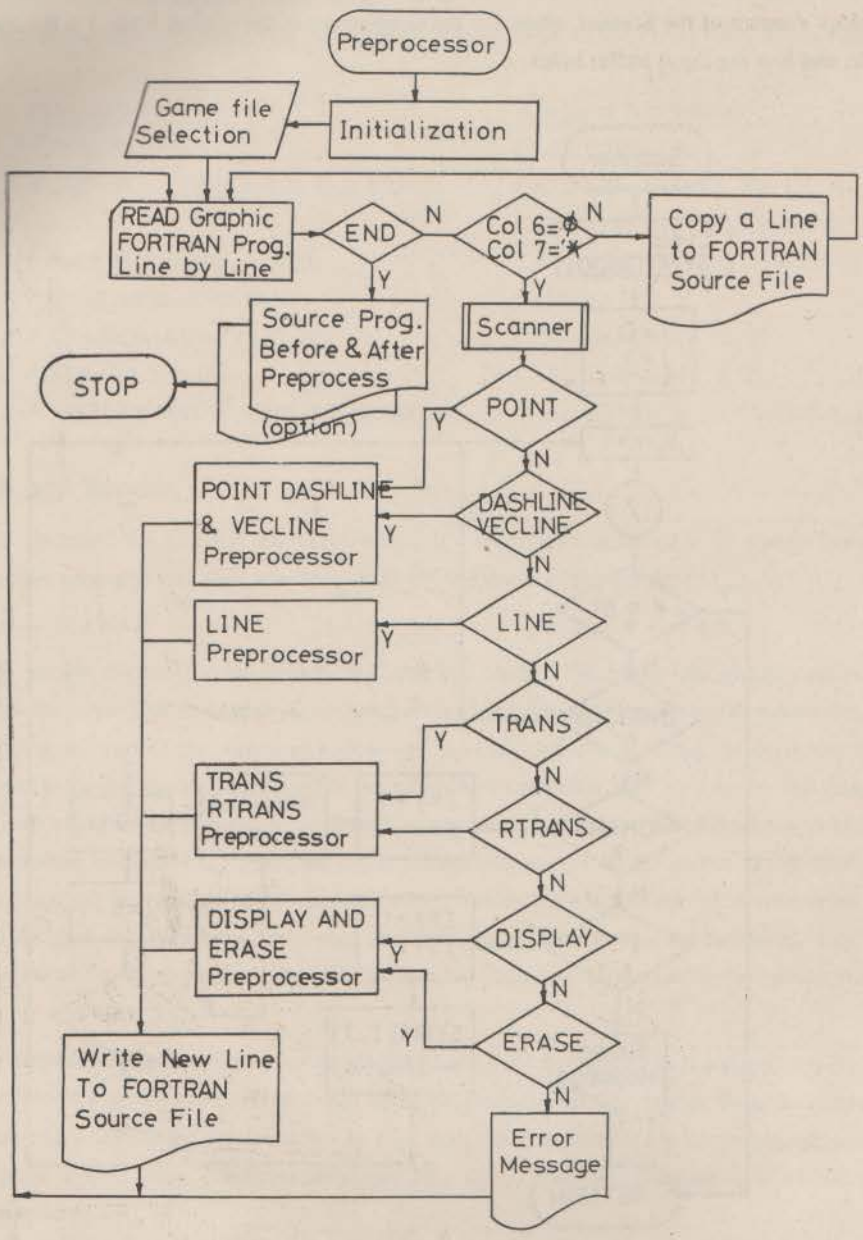


Fig. 6 Simplified Flowchart of tne Preprocessor

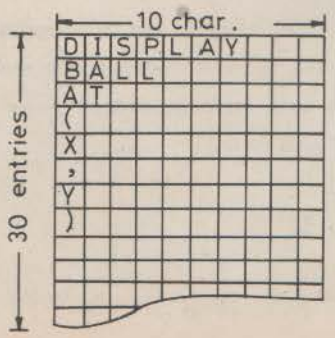


Fig. 7 Organization of Symbol Table (SYMT)

The basic elements or tokens are delimited by blanks, comma, period, left parenthesis or right parenthesis. Fig. 8 shows the block diagram of the Scanner, where I is the entry index of the symbol table, J is the character index of the symbol table, and K is the input buffer index.

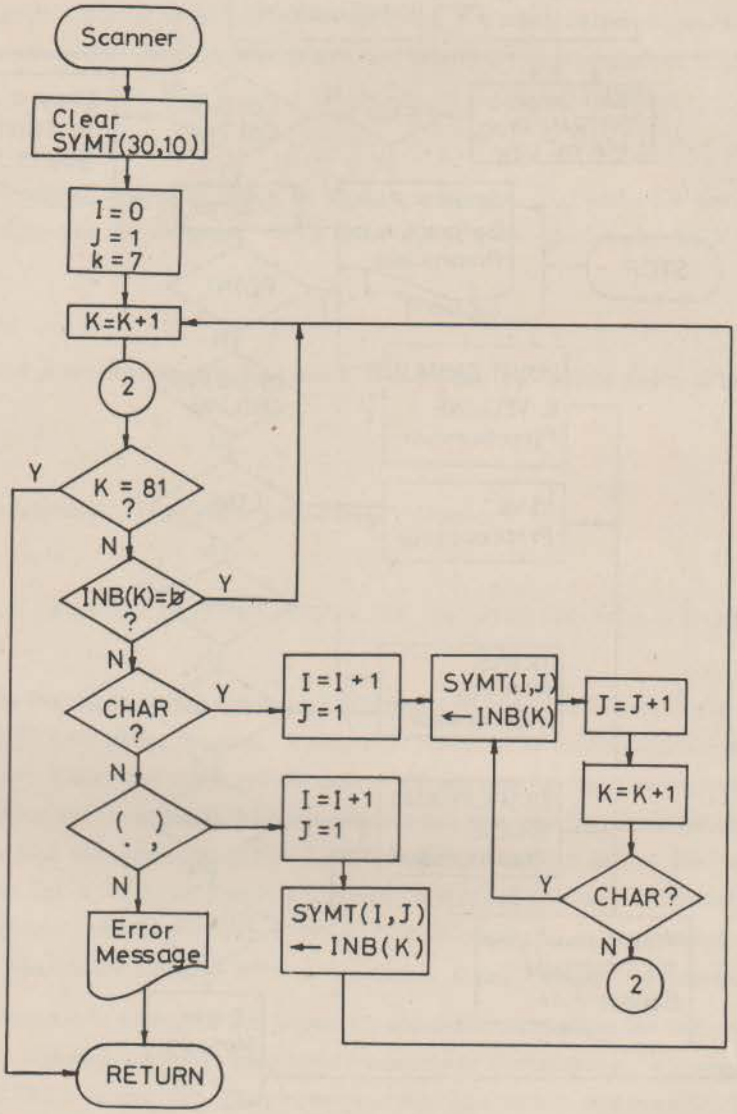


Fig. 8 Flowchart of Scanner

Graphic Statements of the Graphic FORTRAN program can be recognized by noticing a blank and * in column 6 and 7 of that statement. When it is recognized, a corresponding action routine is entered. Since the first entry of SYMT tells us the type of the Graphic Statement, according to the type we can decide which of the four action routines generates CALL statement of the following form:

CALL name (a_1, a_2, a_3,..., a_n, int)

where: name is the name of subroutine subprogram. a_1, a_2, a_3,..., a_n are the actual arguments that are being supplied

to the subroutine subprogram, int is intensity.

The CALL statement(s) are saved in an output buffer and then written to the FORTRAN source File. For examples,

Graphic Statements	Results of Preprocessor
* POINT (30, IY)	CALL POINT (30, IY 1)
* LINE (X1, Y1), (X2, Y2), (X3, Y3)	CALL LINE (X1, Y1, X2, Y2, 1)
	CALL LINE (X2, Y2, X3, Y3, 1)
*VECLIN (100, 25), 3, 50,0	CALL VECLIN (100, 25, 3, 50, 0)
* DISPLAY BAT AT (X, Y)	CALL BAT (X, Y, 1)
* ERASE BALL AT (220, IY)	CALL BALL (220, IY, 0)
* STRANS 2, 2, 0	CALL STRANS (2, 2, 0)
* DISPLAY SCORE AT (5, 30), IASCORE	CALL SCORE (5, 30, IASCORE, 1)

IV. Operation and Results

After having described the Graphic FORTRAN language and the Programmable TV Game System, we can now present three sample programs and their results to illustrate the usage of our system [8].

(1) Sample One —— SEAWAR

A typical TV screen display for the SEAWAR Game is shown in Fig. 9(a). One player controls the horizontal movement of the destroyer and the other player controls the horizontal movement of the submarine. The cargo ship moves across the upper part of the screen at a fixed speed. One player controlling the destroyer must protect the cargo ship and depth charge the submarine while the submarine controller tries to torpedo the cargo ship or the destroyer. Only one torpedo will appear on the screen at any time, rising from the submarine to either strike a ship and cause an explosion or disappear. Only one depth charge will appear on the screen at any time, falling from the destroyer to explode on the submarine or disappear when hitting the sea bottom. The submarine player fires torpedoes to score 2 points for hitting the cargo ship and score 3 points for hitting the destroyer. The destroyer player drops depth charges at the submarine and scores 5 points for hitting an area close to the submarine or a hit on the submarine.

A hit of the torpedo on the cargo ship or the destroyer or a hit of the depth charge on the submarine will cause the symbol of explosion displayed on TV and will cause an explosive sound. In Fig. 9(b), the screen shows that the destroyer is exploded when hit by the torpedo. In Fig. 9(c), the ships are not hit by the torpedo and the water pillar appears. The game is over when either player scores 30 points. The main program of SEAWAR game is given in Appendix for reference.

Fig. 9(a)
The Cargo Ship
is Exploded When
Hit by Torpedo.

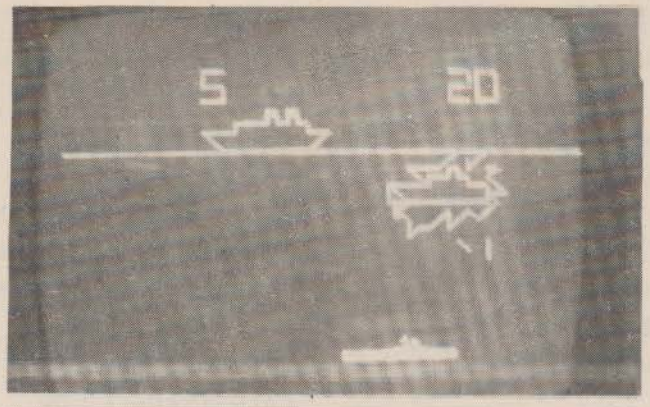


Fig. 9 (b)
The Destroyer
is Exploded.

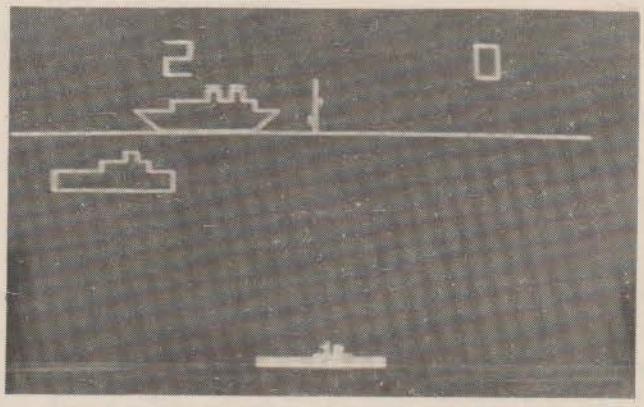


Fig. 9(c)
The Ships are not
Hit by the Torpedo
and Water Pillar
Appears.

(2) Sample Two –– TENNIS

With the tennis game the symbol on the television screen would be similar to Fig. 10(a), with one 'bat' per side, a top and bottom boundary, and a center net. The individual scores are counted and displayed automatically in the position shown. The speed of the ball and the length of the bat depend upon the selection of the options as entered from console. After the reset button has been applied, the scores will be 0, 0 and the ball will serve from one side at arbitrary angle by depressing one of the shot buttons. If the ball hits the top or bottom boundary, it will assume the angle of reflection and continue in play. The player being served must control his bat by knob to intersect the path of the ball. When a 'hit' is detected by the program, it will assume the angle of reflection and continue too.

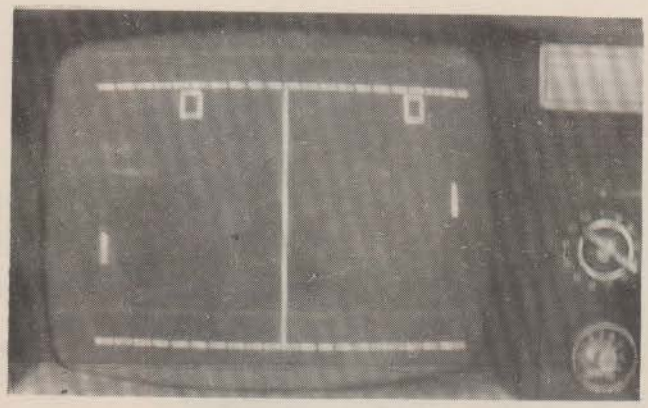


Fig. 10(a)
The Original
Screen of
TENNIS Game.

The ball will then traverse towards the other player, reflecting from the top or bottom as necessary until the other player makes his 'hit'. This action is repeated until one player misses the ball. The program then detects a 'score' and automatically increments the correct score counter and updates the score display. The ball will then be served by the player who has just won. This sequence is repeated until a score of 15 is reached by one side, whereupon the game is stopped. The ball will not be served again. While the game is in progress, two audio tones are output by the system to indicate top and bottom reflections and bat hits. Fig. 10 (b) and Fig. 10(c) show the small bat and the large bat, respectively.

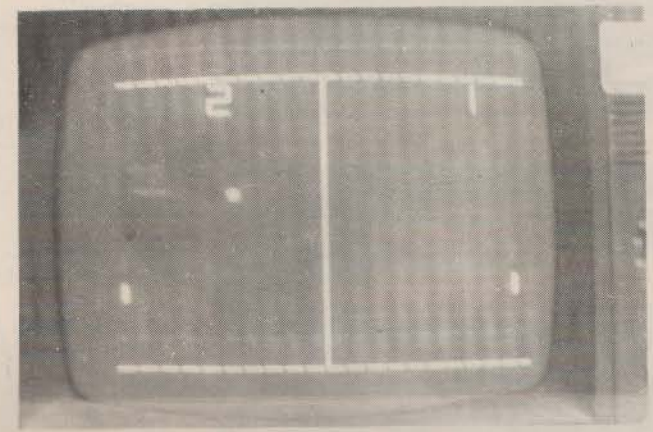


Fig. 10 (b)
The Size of Bat is Selected by Player.

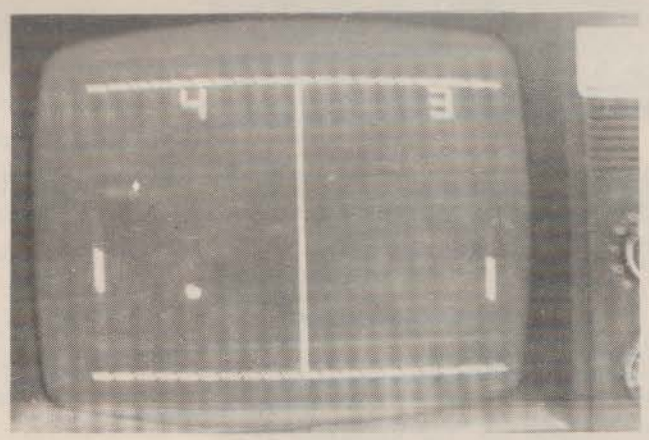


Fig. 10 (c)
A Larger Bat is Selected.

(3) Sample Three — BREAK OUT

BREAK OUT is a one player game which utilizes only one bat as shown in Fig. 11. The left counter displays the score of the player while the right counter shows the maximum score of the previous game. There are 8 rows and 8 columns of bricks. The player controls the horizontal movement of the bat. When the player pushes the shot button, the ball will serve arbitrarily from one side at an arbitrary angle. The ball will then traverse towards the bat, reflecting from the end boundary when necessary, until the player makes his 'hit'. When a 'hit' is detected, the section of the bat which made the hit is used to determine the new angle of the ball. A score is made when the ball hits a brick. 1 point is for hitting the 2 lowest rows, 3 points for hitting the third and fourth row, 5 points for the next two rows, and 7 points for the uppermost two rows. Then the ball will reflect back. This action is repeated until the player misses the ball. There are three balls for each player.

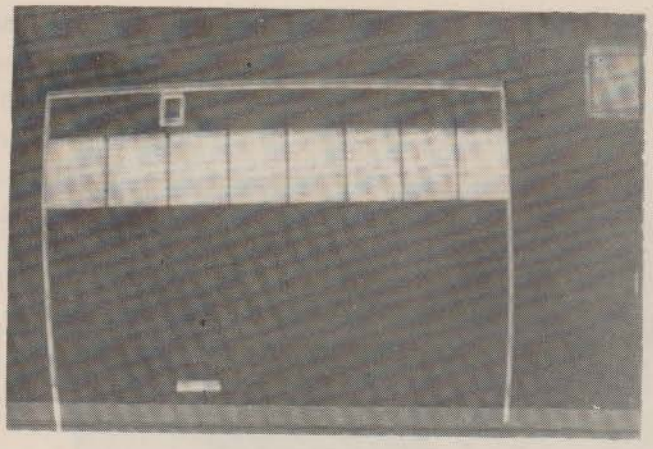


Fig. 11 (a)
The Original
Screen of
BREAK OUT.

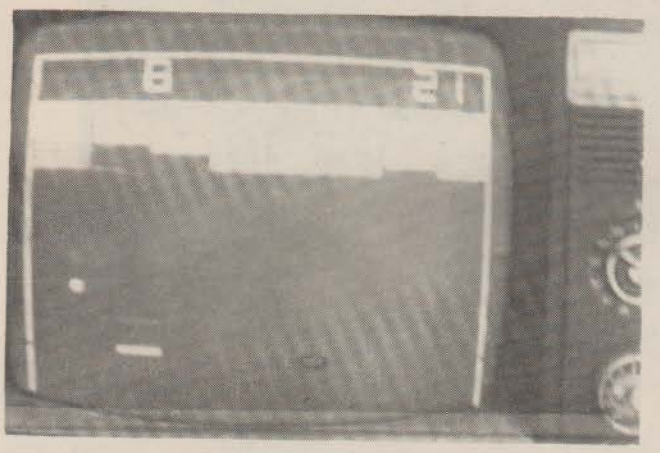


Fig. 11 (b)
The Speed of the
Ball Varies with
the Score.

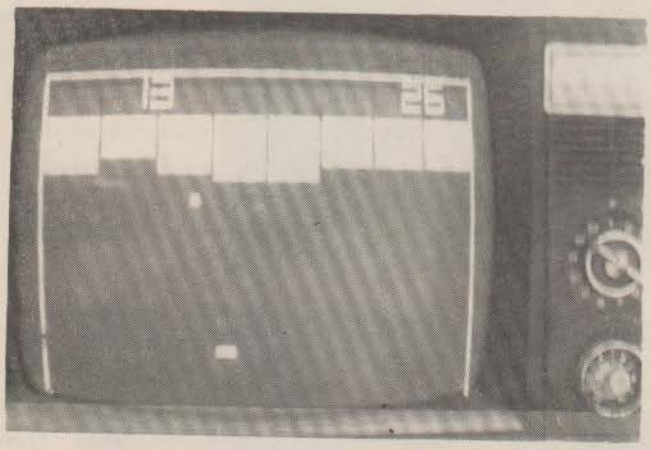


Fig. 11 (c)
After the Ball
Hits Tenth
Brick, the Bat
Becomes Smaller
Automatically.

V. Conclusion

A programmable TV game is a new tendency. But how can we develop the program of a TV game? In this paper, we present an easy method to reach this goal by designing and implementing a Programmable TV Game System. In the respect of software, we design a Graphic FORTRAN language. In the respect of hardware we design a simple interface for the player(s) including the low speed but low cost ADC.

The Programmable TV Game System has been implemented and applied to three TV games: SEAWAR, TENNIS

and BREAK OUT. In considering about the speed, the result is satisfactory for these games. However, for a more complicated TV game, the result may not be tolerable. In considering about the memory requirement, Fig. 12 illustrates the program size of the three games. The total program size of BREAK OUT game is 8779 bytes among which 4206 bytes are shared with another 2 games. Therefore, to implement this game only needs a main program of 3799 bytes and 744 symbol generation program.

Unit: Byte

	Main Program	Graphic Symbol	FORTRAN Subroutines	Assembly Subroutines	TOTAL
SEAWAR	2429	2955	3365	841	9590
TENNIS	2398	308	3365	841	6912
BREAK OUT	3799	774	3365	841	8779

Fig. 12

Acknowledgements

The authors are thankful to the Digital System and Microcomputer Laboratory of NCTU for providing various computing facilities.

References

1. Kris Jensen, "New 1978 Electronic Games", Popular Electronics, January 1978, pp. 33-43.
2. Motorola Semiconductor Products Inc., M6800 EXORciser User's Guide.
3. MATROX Electronic System Inc., MTX-256**2 Data Sheet.
4. MATROX Electronic Systems Inc., MTX-256**2 Application Note.
5. Electronic & Radio-TV Technic, Vol. 35, No. 10, 1976, p. 136.
6. A Hurwitz, J. P. Citron, and J. B. Yeaton, "GRAF: Graphic Additions to FORTRAN", Proc. AFIPS 1967 SJCC, Vol. 30, AFIPS Press, Montvale, N. J., pp. 553-557.
7. B. W. Jordan, W. J. Lennon, B. D. Holm, "An Improved Algorithm for the Generation of Nonparametric Curves", IEEE Trans. Comput., C-19, pp. 783-793, Sept. 1970.
8. General Instrument Corp., 1978 Data Catalog.

Appendix

```
PAGE 001  SE        SA:1

C
C
C    MAIN PROGRAM OF SEAWAR GAME
C
      COMMON ITX,ITY,ISX,ISY,RD
      COMMON IWICH
      CALL ASMIRQ
      CALL MTXINT
C
C     RESET
C
  888 *ERASE ALLERA
      *RTRANS
      *LINE (0,80),(255,80)
```

```
        ISSCOR=0
        IDSCOR=0
        *STRANS 2,2,0
        *DISPLAY SCORE AT (50,35),ISSCOR
        *DISPLAY SCORE AT (185,35),IDSCOR
        ISSHOF=1
        IDSHOF=1
        JXD=0
        JXS=0
        ICC=-5
        ITT=0
        IGG=0
C
C       DISPLAY CARGO SHIP
C
        IXC=IRAND(100,200)
        *DISPLAY CARGO AT (IXC,69)
        JXC=IXC
C
C       LOOP
C
    1   IWICH=9
    5   CALL ASMCL1
        CALL ASMSEI
        IF(IWICH.EQ.9) GO TO 2
        GO TO (888,52,53,54,99),IWICH
    2   IXD=JVR(1)
        IXS=JVR(0)
        ITCNT=0
        IF(IXD.EQ.JXD) GO TO 3
        *ERASE DESTRO AT (JXD,100)
        IF(IXD.GT.200.OR.IXD.LT.0) GO TO 3
        *DISPLAY DESTRO AT (IXD,100)
        JXD=IXD
    3   IF(IXS.EQ.JXS) GO TO 4
        *ERASE SUBMAR AT (JXS,200)
        IF(IXS.GT.200.OR.IXS.LT.0) GO TO 4
        *DISPLAY SUBMAR AT (IXS,200)
        JXS=IXS
    4   *RTRANS
        *LINE (0,80),(255,80)
        GO TO 5
C
C       888,RESET  IWICH=1

PAGE 002  SE       .SA:1

C       52,  SUBMARINE SHOOT TORPEDO
C       53, DESTROYER SHOOT DEPTH-CHARGE
C       54, TIMER INTERRUPT CARGO,TORPEDO,DEPTH-CHARGE MOV
C
C 52
C
   52   IWICH=9
        IF(ISSHOF)99,99,62
   62   IF(IXS.GT.200.OR.IXS.LT.0) GO TO 2
        ISSHOF=-1
        IXT=IXS+25
        IYT=193
        *DISPLAY TORPED AT (IXT,IYT)
        JXT=IXT
        JYT=IYT
        ITT=-20
        CALL ASMSND(4,2)
        GO TO 2
```

```
C
C 53  DESTROYER SHOT DEPTH-CHARGE
C
   53 IWICH=9
      IF(IDSHOF)99,99,63
   63 IF(IXD.GT.200.OR.IXD.LT.0) GO TO 2
      IDSHOF=-1
      IXG=IXD+32
      IYG=110
      *DISPLAY DEPCHA AT (IXG,IYG)
      JXG=IXG
      JYG=IYG
      IGG=46
      CALL ASMSND(4,2)
      GO TO 2
C
C 54  TIMER INTERRUPT: MOVE CARGO,TORPEDO,DEPTH-CHARGE;
C
   54 IWICH=9
      IXC=JXC+ICC
      IF(IXC.GT.0) GO TO 541
      IXC=190
  541 *ERASE CARGO AT (JXC,69)
      *DISPLAY CARGO AT (IXC,69)
      JXC=IXC
C
C     MOVE TORPEDO, AND TEST
C
      IF(ISSHOF)542,55,55
  542 IYT=JYT+ITT
      IF(IYT.LT.80) GO TO 546
      IF(IYT.LT.110) GO TO 543
  549 *ERASE TORPED AT (JXT,JYT)
      *DISPLAY TORPED AT (IXT,IYT)
      JXT=IXT
      JYT=IYT
      GO TO 55
  543 IXDMX=IXD+50
      IF(IXT.GT.IXD.AND.IXT.LT.IXDMX) GO TO 544
      GO TO 549

PAGE 003  SE       .SA:1

  546 IXCMX=IXC+60
      IF(IXT.GT.IXC.AND.IXT.LT.IXCMX) GO TO 545
  547 *ERASE TORPED AT (JXT,JYT)
      *DISPLAY TORPE2 AT (IXT,IYT)
      CALL ASMSND(4,2)
      ISSHOF=1
      *ERASE TORPE2 AT (IXT,IYT)
      GO TO 55
  544 *DISPLAY EXPLOR AT (JXD,100)
      CALL ASMSND(4,3)
      ISSHOF=1
      ISSCOR=ISSCOR+3
      *STRANS 2,2,0
      *DISPLAY SCORE AT (50,35),ISSCOR
      *ERASE TORPED AT (JXT,JYT)
      *ERASE EXPLOR AT (JXD,100)
      *ERASE DESTRO AT (JXD,100)
      GO TO 548
  545 *DISPLAY EXPLOR AT (JXC,69)
      CALL ASMSND(4,3)
      ISSHOF=1
      ISSCOR=ISSCOR+2
      *STRANS 2,2,0
      *DISPLAY SCORE AT(50,35),ISSCOR
```

```
      *ERASE EXPLOR AT (JXC,69)
      *ERASE CARGO AT (JXC,69)
      *ERASE TORPED AT (JXT,JYT)
      GO TO 548
  548 IF(ISSCOR.LT.30) GO TO 55
      ISSHOF=0
      IDSHOF=0
      CALL ASMSND(4,5)
      GO TO 2
C
C 55  MOVE DEPTH-CHARGE AND TEST
C
   55 IF(IDSHOF)551,99,99
  551 IYG=JYG+IGG
      IXSMX=IXS+65
      IF(IXG.GT.IXS.AND.IXG.LT.IXSMX) GO TO 552
  554 IF(IYG.GT.250) GO TO 553
      *ERASE DEPCHA AT (JXG,JYG)
      *DISPLAY DEPCHA AT (IXG,IYG)
      JXG=IXG
      JYG=IYG
      GO TO 99
  553 *ERASE DEPCHA AT (JXG,JYG)
      CALL ASMSND(4,2)
      IDSHOF=1
      GO TO 99
  552 IF(IYG.LT.200) GO TO 554
      *DISPLAY EXPLOR AT (JXS,200)
      CALL ASMSND(4,3)
      IDSCOR=IDSCOR+5
      *STRANS 2,2,0
      *DISPLAY SCORE AT (185,35),IDSCOR
      IDSHOF=1
      *ERASE EXPLOR AT (JXS,200)

PAGE 004  SE       .SH:1

      *ERASE SUBMAR AT (JXS,200)
      *ERASE DEPCHA AT (JXG,JYG)
      IF(IDSCOR.LT.30) GO TO 99
      IDSHOF=0
      ISSHOF=0
      CALL ASMSND(4,5)
      GO TO 2
   99 GO TO 2
    9 GO TO 2
      STOP
C
C     END OF SEAWAR GAME MAIN PROGRAM
C
      END
```

In this program, we use some utility subprograms. These subprograms are provided for a user to develop his TV game program. The subprogram names and their functions are:

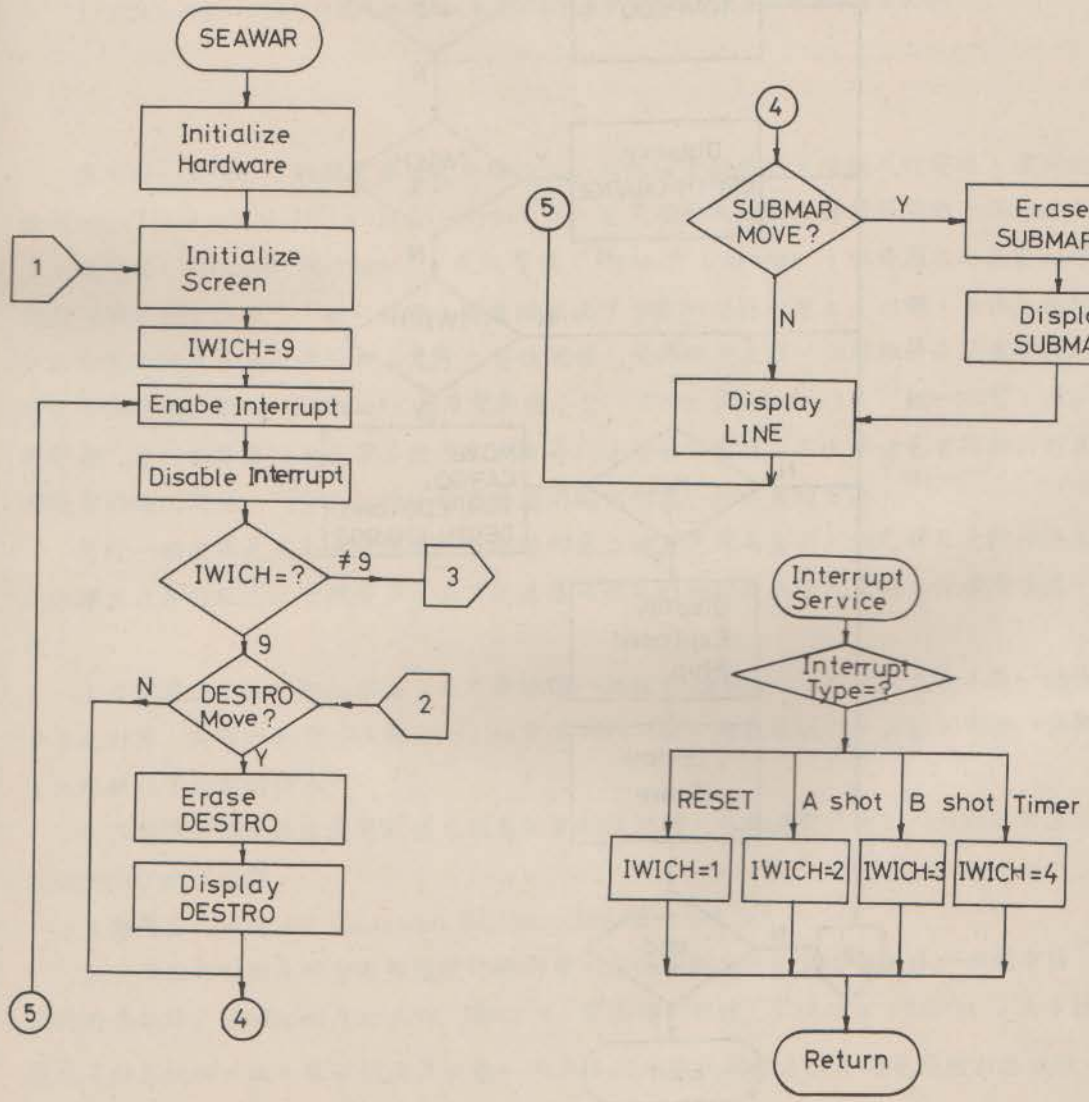
ASMCLI	A subroutine subprogram which enables the interrupt.
ASMSEI	A subroutine subprogram which disables the interrupt.
ASMSND (X,Y)	A subroutine subprogram which generates sound. X for sound selection, Y for sound duration.
JVR (I)	A function subprogram which returns the value of control knob of player I. Its range is [0, 255].
IRAND (A, B)	A function subprogram which returns random number of [A, B].

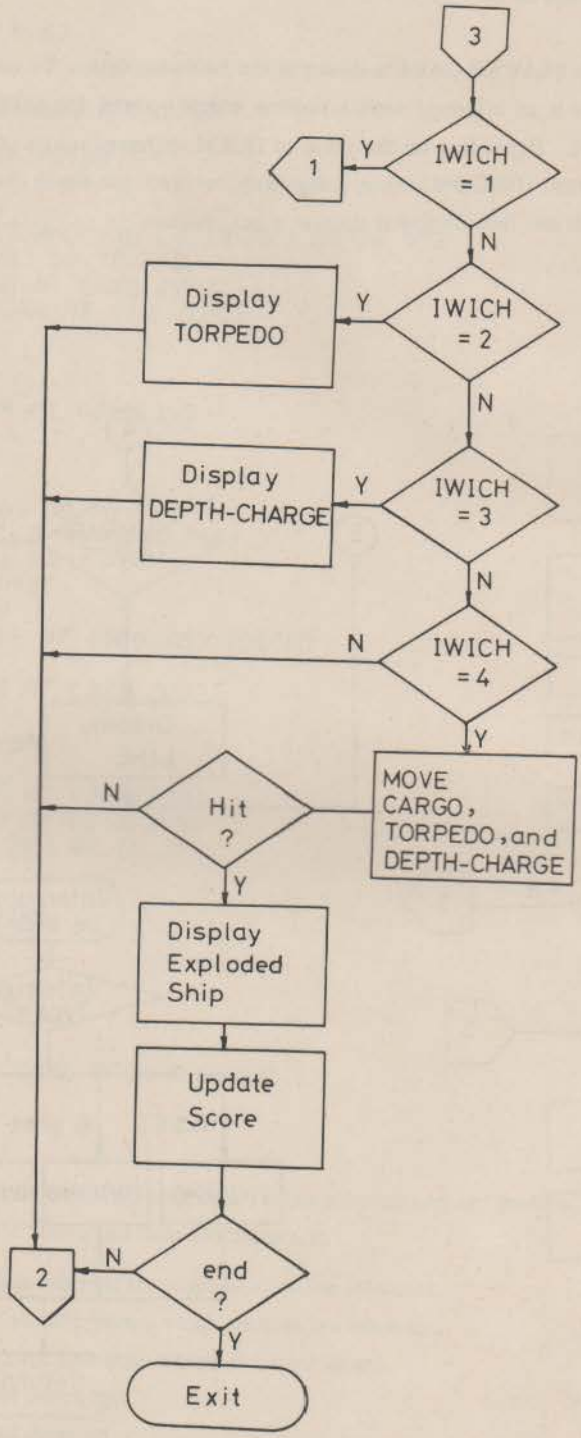
ASMIRQ A subroutine subprogram which initializes PIO's.

MTXINT A subroutine subprogram which initializes MTX-256**2 interface.

ASMSND (I, J) A subroutine subprogram which selects the Ith sound generator and generates the sound for a time interval depending on J.

A simplified flowchart for this SEAWAR GAME is shown in the following figure. To understand the flowchart, it is important to notice that there is an interrupt service routine which updates the COMMON variable IWICH according to the types of interrupt. Depending on the value of IWICH, different routes of the flowchart are taken. For example, when a timer interrupt (IWICH=4) occurs, cargo ship, torpedo, and depth charge should be moved. To move a cargo ship means to erase it and then display it again at other position.




3
IWICH = 1
Y
1
N
IWICH = 2
Y
Display TORPEDO
N
IWICH = 3
Y
Display DEPTH-CHARGE
N
IWICH = 4
N
Y
MOVE CARGO, TORPEDO, and DEPTH-CHARGE
Hit ?
N
Y
Display Exploded Ship
Update Score
end ?
N
2
Y
Exit