Start-Up Guide

Software Development Tool

Start-Up Guide
Software Development Tool

- This product is sold to Members of Net Yaroze according to the terms of the membership agreement. Net Yaroze is operated by Sony Computer Entertainment Inc.
- The  symbol, 'PlayStation' and 'Net Yaroze' are trademarks of Sony Computer Entertainment Inc.
- The names of companies and products referred to in Net Yaroze are trademarks of their respective companies. Company and product names recorded in/on this product are generally trademarks of each company. Note that the symbols '® 'and 'TM' are not used explicitly.

Published March 1997

Written and produced by :
Sony Computer Entertainment Inc.
Akasaka Oji Building
8-1-22 Akasaka, Minato-ku, Tokyo, Japan 107
Inquiries to: Network Business Project
E-mail:ny-info@scei.co.jp
TEL:+81 (0) 3-3475-1711

Sony Computer Entertainment Europe
Waverley House
7-12 Noel Street
London W1V 4HH, England
Inquiries to: The Yaroze Team
E-mail: yaroze-info@scee.sony.co.uk
TEL: +44 (0)-171-447-1616

Sony Computer Entertainment America
919 E. Hillsdale Blvd., 2nd Floor
Foster City, CA 94404
USA
Inquiries to: The Yaroze Team
E-mail: yaroze@interactive.sony.com
TEL: +1-415-655-3600

Table of Contents

About Net Yaroze

What You Need to Know

In order to get started with Net Yaroze, you should have ***C programming*** experience at some level of competence and you should be familiar with ***2D graphic creation/editing tools***. You should also have a basic understanding of ***3D modelling packages*** and ***sound creation/editing tools.*** Together these will help you get the most out of your Net Yaroze System.

The Net Yaroze Manual Set

There are three books in the Net Yaroze manual set.

- *Start-Up Guide* (this document)
  An introductory booklet explaining the contents and requirements of the Net Yaroze Starter Kit. The *Start-Up Guide* gives step-by-step instructions on setting up the Net Yaroze software on your PC. It also explains how to run software on the Net Yaroze system.

- *User's Guide*
  A reference manual explaining how to create software for the Net Yaroze system.

- *Library Reference*
  A manual listing and describing the functions and structures in the Net Yaroze libraries.

Additional Reading

See the *Additional Reading* section at the end of this manual for further information.

Introduction

The Net Yaroze Starter Kit provides a complete development environment for creating PlayStation software on your personal computer. Software that you create on your PC can be downloaded and played on a special Net Yaroze PlayStation. The Net Yaroze PlayStation is only available to Members of Net Yaroze.

Net Yaroze Members can share their creations and knowledge with other Members via a unique Members-only Web site. This Web site is provided by Sony Computer Entertainment.

Members-Only Web Site

Connecting to the Members-Only Web Site

Members are provided with access to one of three Net Yaroze Web sites as shown below.

http://www.scei.co.jp/net/	for Members in Japan
http://www.scee.sony.co.uk/yaroze/	for Members in Europe
http://www.scea.sony.com/net/	for Members in North America

Browser Software Requirements

Netscape version 2.0 or later is the recommended browser software necessary for connecting to the Members-only Web site.

ID and Password

The ID and password required for accessing the Members-only Web site will be provided with your Welcome Pack.

Participating and Contributing to Net Yaroze

A key part of Net Yaroze is the participation of its Members on the Net Yaroze Web site. Members can participate on the Net Yaroze Web site by sharing their work with others, enjoying work created by other Members, working together on projects and many other activities.

Obtaining Additional Information

Sony Computer Entertainment will provide additional technical information and data that is not included in the documentation. Sony will make this information available on the Net Yaroze Web site. Please ask for any additional information that you may need.

Contents of the Net Yaroze Package

The following items are included in the Net Yaroze package:

1. DTL-H3001 Net Yaroze Member's PlayStation (for USA) or
   DTL-H3002 Net Yaroze Member's PlayStation (for Europe)
   (includes power cable and AV (RFU) cable)
2. DTL-H3010 Controllers x 2
3. DTL-H3020 Access card
4. DTL-S3035 Net Yaroze boot disk
5. DTL-S3045 Net Yaroze software development disk
6. DTL-H3050 Communications cable
7. DTL-D3065 Start-Up Guide (this manual)
8. DTL-D3075 User's Guide
9. DTL-D3095 Library Reference

Notes

Please note that the model numbers of the package you receive may vary slightly depending on which of the three Net Yaroze Members' regions that you live in.

'Net Yaroze' is also referred to simply as 'Yaroze'. Net Yaroze is a trademark of Sony Computer Entertainment, Inc.

Operating Environment

The following section describes the hardware and software needed to use the Net Yaroze development environment.

Hardware Requirements

A DOS-compatible PC/AT with a 486DX2 66MHz CPU or faster.

The PC should have one or more serial ports and a modem (at least 14,400 bps) for connection to the Net Yaroze Web site.

- ***Hard Disk***

  At least 10MB of free disk space is needed to install the basic development environment.

- ***Memory***

  At least 4MB of available RAM

- ***CD-ROM Drive***

- ***Display***

For the PC	SVGA monitor
For the PlayStation	A standard NTSC TV monitor with a video input terminal. When working with a PAL game, you must use a PAL compatible Color System television or monitor.

- ***Mouse***

Software Requirements

MS-DOS Version 5.0 or later.

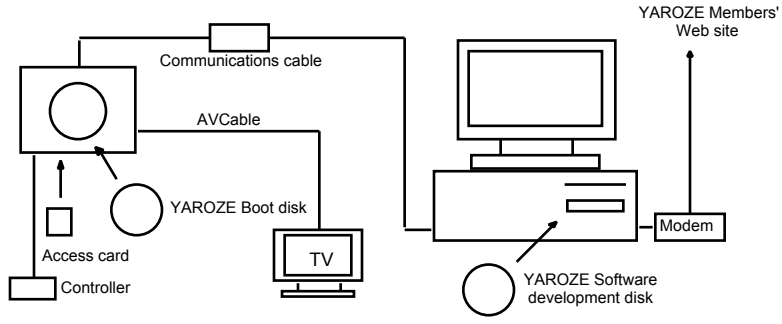
Windows 3.1 or Windows 95.

Connecting the Hardware

Make sure that the PC and Net Yaroze PlayStation power switches are turned off. Connect the serial port of the PC to the PlayStation using the Communications cable, as shown in the diagram below. Insert the Access card in the PlayStation's Memory card Slot 1.

Next, connect the PlayStation to the TV monitor with the supplied AV cable.

In order to access the Net Yaroze Members-only Web site via the Internet, you need a separate modem and telephone line connection, a contract with an Internet service provider and an Internet Web browser installed on your PC.

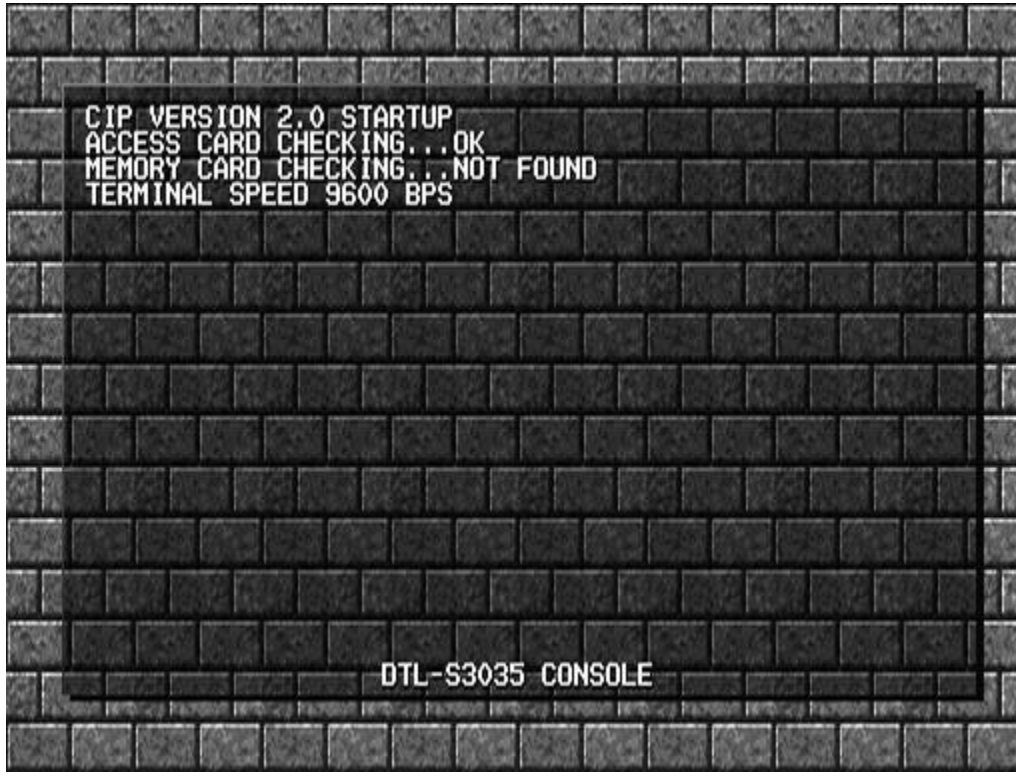


Setting Up the Software

For the PlayStation

No special set-up is necessary. The development environment is loaded automatically when you place the Net Yaroze boot disk in the Net Yaroze PlayStation, insert the Access card in Memory card Slot 1 and switch the power on.

After boot-up the following screen is displayed on the TV monitor.



For the PC

Copying Folders

Place the enclosed Net Yaroze software development CD (hereafter 'CD-ROM') in the PC CD-ROM drive. Copy the contents of each folder to the hard disk using the DOS XCOPY command or the Windows Program Manager/Explorer. (At the root of the CD-ROM there are two folders called PSX and GNU. You can copy these folders anywhere on the hard disk.) In the example below, the PSX and GNU folders are copied onto the root of the C drive.

Modifying config.sys

Check the contents of config.sys in the root of the start-up drive and confirm whether or not ansi.sys is included. If it is not included, add the following line to config.sys and reboot the system.

```
devicehigh=C:\WINDOWS\ansi.sys
```

Modifying the Batch File

There is a file called djsetup.bat in the PSX folder which you have copied to the hard disk. Its contents are shown below.

```
@echo off
set DJGPP=c:/psx/djgpp.env
PATH %path%;c:\psx\bin;c:\gnu\bin
set TMP=c:\tmp
set DTLH3000=0x3f8,4,9600
```

If you did not copy the PSX and GNU folders onto the root of the C drive, modify this file so the contents correspond to the location and names (if you changed these) of the copied folders. This procedure is described below.

Note

Because the file was copied from the CD-ROM, its access mode is read-only. Before editing, change this in DOS as shown below.

```
C:\PSX> attrib -r djsetup.bat
```
(where 'C:\PSX' is the path)

- The second line sets the environment variable DJGPP. This variable specifies the compiler's environment-setting file. If appropriate, modify the c:/psx part so it refers to the correct path and name of the copied folder. Note that you must use a '/' (forward slash) to delimit the path for *this* environment variable only.

- The third line sets the environment variable PATH. Modify the c:\psx and c:\gnu parts to specify the name and path of the copied folder, if appropriate.

- The fourth line sets the working folder. Create and allocate a folder if c:/tmp is not appropriate. For example, change the line to:

  `set tmp=c:\yaroze\wrk` (where 'wrk' is your preferred working folder).

- The fifth line sets the communication parameters for the link with the Net Yaroze PlayStation. Specify the port address, IRQ and speed (in this order) separated by commas. The defaults, 0x3F8 and 4, are the port address and IRQ of COM1.

- If you use COM2, specify 0x2F8 and 3 as the port address and IRQ, respectively.

- Communication speeds of up to 115200 bps are supported. Specify either 9600, 19200, 38400, 57600 or 115200 according to the processing capacity of your PC.

- (Please note that if you wish to use speeds other than 9600 baud, you should place a standard-formatted PlayStation Memory card in your Net Yaroze PlayStation's Memory card Slot 2 and change the rate using SIOCONS - See Chapter 17 of the Net Yaroze *User's Guide* for a description of SIOCONS.)

Executing the Environment-Setting Batch File

Activate the MS-DOS prompt from Windows (or directly from DOS). Change the current path to PSX (or your chosen folder name) using the CD command. Then execute the modified batch file, djsetup.bat, as shown below in bold type.

```
C:\PSX>djsetup
```
(where 'PSX' is your chosen Net Yaroze folder name)

This completes the set-up of the development environment.

Getting Started

The Net Yaroze CD-ROM contains some sample and quick-start programs which you can compile and run to test your Net Yaroze system.

The *Quick-Start* section below provides a step-by-step guide through Net Yaroze software development. Experienced programmers may wish to skip the explanations of the *Quick Start* and go directly to the section later in this chapter titled *Compiling and Executing the Sample Program*.

Quick-Start

The 'quick' directory on the Net Yaroze CD-ROM contains the quick-start test programs. You probably copied it to your PC's hard disk together with all the Net Yaroze software.

Using the quick-start programs you can quickly check that everything is working on your Net Yaroze system.

Net Yaroze Program Development - Overview

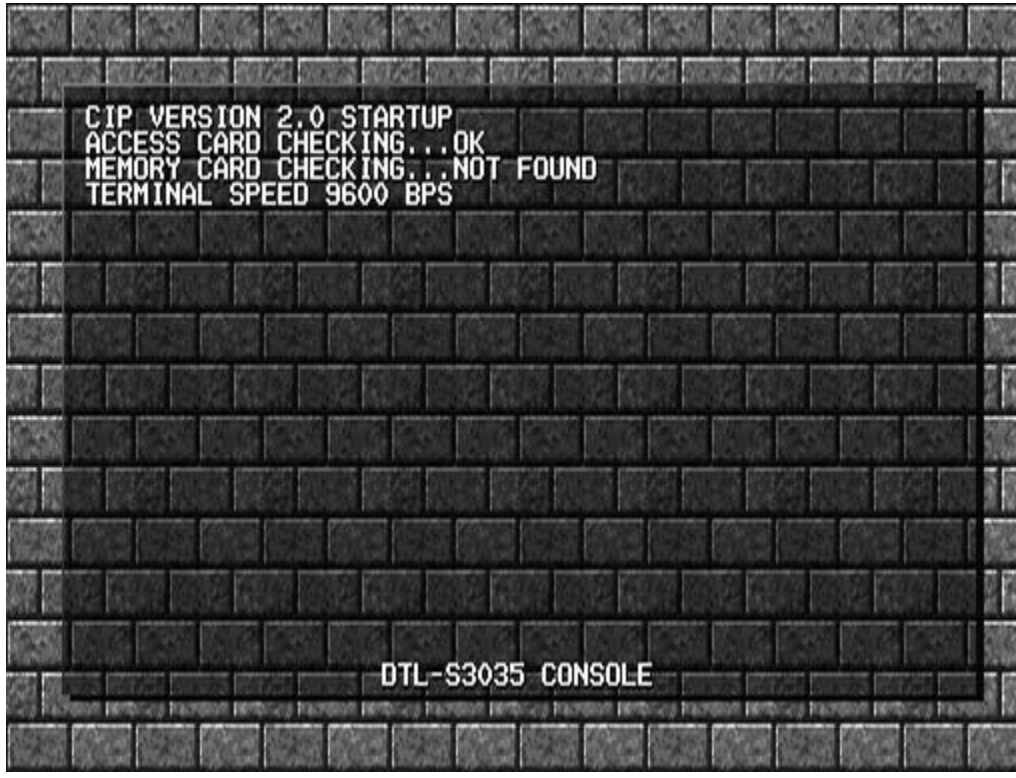
Here are the basic steps that you need to take to develop software on the Net Yaroze system. (See Chapter 13 of the Net Yaroze *User's Guide* for more details on the development cycle.)

1. Create/edit text files which contain the C code.
   Also known as 'source code', these files are usually named 'name.c' (where 'name' is the selected file name).
   (There are two source code files already in the quick-directory: tuto0.c and tuto1.c.)
2. Compile and link the C files to create the executable file (the file that is the finished program/application).
   (In the test example below, we use a makefile called by the command 'make' to do this.)
3. Use the SIOCONS console tool, provided as part of your Net Yaroze software to establish contact between your Net Yaroze PlayStation and PC.
4. Run the executable program.
5. (In this test example there are two batch files - batch0 and batch1 - which you can use to run the executable files (or 'executables') of the test programs.)

Compiling and Running the Quick-Start Programs - Step-by-Step

1. Set up your Net Yaroze PlayStation as described in the *Set-Up* section of this guide: connect the Net Yaroze PlayStation to your TV and PC, and insert a Controller in the PlayStation's Controller port 1.

2. Turn on the PC, TV and Net Yaroze PlayStation and wait for the ready-to-go screen to appear on the TV as shown below.



This screen should appear within 30 seconds of turning on or resetting the Net Yaroze PlayStation after the Sony and PlayStation logo screens appear.

3. Bring up an MS-DOS prompt on the PC. From the PSX folder (or your chosen Net Yaroze folder name) invoke the Net Yaroze environment-setting batch file as follows:

```
c:\PSX>djsetup
```
(where 'PSX' is your chosen Net Yaroze folder name)

4. Compile the quick-start programs. Use the 'make' utility to do this by typing 'make' at the DOS prompt. This will create the executable programs from the source files.

```
c:\psx\quick>make
```

5. Bring up the console tool, SIOCONS. (You must have set up the quick-directory as a path for SIOCONS in your autoexec.bat file.)

```
c:\psx\quick>siocons
```

or, if you have already changed the baud rate, type

`c:\psx\quick>siocons -B<baud rate>` (where '<baud rate>' is the value of your chosen rate)

6. Now you are in SIOCONS, the console tool used to communicate with the Net Yaroze PlayStation from the PC.

The PC monitor output should look something like this:

```
c:\psx\quick>siocons -B115200
siocons -- PlayStation debug system console prog
for DTLH3000 1996/05/10 00:00:03
type  F1   ----> display help
when hung up try type  ESC
I/O addr = 0x03F8, IRQ=4(vect=0x000C,8259=20)
BAUDRATE = 115200
```

7. Check that SIOCONS is working correctly by pressing return. If all is well, the SIOCONS prompt (a double-right arrow: `>>`) will appear.

8. Run the quick-start batch files.

First, press [F3] on the PC keyboard. This brings up the `Auto[1]:` prompt. Then enter the name of the batch file; **`batch0`**.

For example (after pressing [F3] on the PC keyboard):

```
Auto[1]:batch0
```

This will download batch0 and run tuto0.exe.

9. Immediately after starting the download, downloading messages will appear on the PC monitor similar to the following.

```
tuto0 [ .text] address:80100000-801001ef size:0001f0  0001f0:   1sec.
tuto0 [.rdata] address:801001f0-8010020f size:000020  000020:   1sec.
tuto0 [ .data] address:80100210-8010035f size:000150  000150:   2sec.
tuto0 [.sdata] address:80100360-801003df size:000080  000080:   2sec.
```

10. The TV screen shows downloading messages similar to those below:

```
BINARY DATA TRANSFER - BWR
ADRS = 80100000
SIZE = 000001F0 BYTES
INTO BURST TRANSFER  SUM:111F0/1F0 BYTES
DONE.
```

Running and Terminating a Program

Tuto0 (called by batch0) prints 'Hello World!' to the SIOCONS console on the PC. Once it completes it will terminate automatically.

When you have run and terminated a program, the Net Yaroze PlayStation resets itself and returns to the ready-to-download screen (as shown in [2] of *Compile and Run the Quick-Start Programs - Step-by-Step*, above). SIOCONS will then display its prompt again (>>).

Once the PlayStation and PC are reset you can run the second quick-start program, tuto1, by following the steps described in [8] and [9] of *Compile and Run the Quick-Start Programs - Step-by-Step*, above.

Tuto1 (called by batch1) prints 'Hello World!' to the TV screen. Terminate this program by pressing SELECT on the Controller.

Exiting SIOCONS

To quit SIOCONS and return to the MS-DOS prompt on the PC keyboard, press [F10] then [F2], *or* just [Esc].

Using the Test Files

The source files, makefile and batch files are all simple text files which you can easily edit (using the text editor in DOS, for example) to see their contents.

To edit or look at the tuto0.c source code using the text editor in DOS, type:

```
c:\>psx\quick>textedit tuto0.c
```

Look at the test files and other sample code to familiarize yourself with what the Net Yaroze system can do and how you can achieve this through C programming.

Quick-Directory Contents

Type	Name
C files for the sample programs	tuto0.c
	tuto1.c
	pad.c
Makefile	makefile
Batch files	batch0 (downloads and runs tuto0)
	batch1 (downloads and runs tuto1)
Subdirectories	
make	a help file on the program updating utility 'make' and some example makefiles
batch	a help file on batch (autoexecution) files and some example batch files
baudrate	a help file on changing the baud rate (rate of data downloading from the PC)

Compiling and Executing the Sample Program

In order to check your environment, try running the sample program provided. (The source code and executable file are in the psx\sample\check folder - where 'psx' is your chosen file name.)

Compile and execute the sample code as follows:

1. Execute Make

   Type **make** at the DOS prompt to test the paths listed in djsetup.bat, and ensure that the program is compiled and linked correctly as shown below (the text in bold indicates command line input).

```
C:\PSX\SAMPLE\CHECK>make
gcc -O1 -g   -c main.c -o main.o
gcc -Xlinker -Ttext -Xlinker 80140000 -o main main.o
```

2. Run SIOCONS

   After the program is compiled and linked, run the console monitor, SIOCONS, as shown below.

```
C:\PSX\SAMPLE\CHECK>siocons
siocons -- PlayStation debug system console program
   for DTLH3000 1996/05/10 00:00:00
   type  F1   ----> display help
   when hung up try type  Ctrl+C
 I/O addr = 0x03F8, IRQ=4(vect=0x000C,8259=20)
 BAUDRATE = 9600
```

3. Auto download

   Press the F3 function key to obtain the `Auto[1]` prompt. Type **auto** to indicate automatic download and press the enter key. (See below.)

```
Auto[1]: auto
main [ .text] address:80140000-80140b4f size:000b50  000b50:   1sec.
main [.rdata] address:80140b50-80140c7f size:000130  000130:   1sec.
main [ .data] address:80140c80-801412ef size:000670  000670:   2sec.
main [.sdata] address:801412f0-8014137f size:000090  000090:   2sec.
PC=80140990, GP=801492f0, SP=801ffff0

>>go
ResetGraph:jtb=80047dd0,env=80047e18
```

The sample program screen should now be displayed on the TV monitor.

The Controller button operation is as follows.

Up directional button	Increases number of balls displayed
Down directional button	Decreases number of balls displayed
L1 button	Pauses display
Select button	Terminates

Sample Program Makefile

```
CFLAGS  = -O1 -g
LINKER  = -Xlinker -Ttext -Xlinker 80140000
RM      = del

PROG    = main
OBJS    = main.o

all: $(PROG)

$(PROG): $(OBJS)
         $(CC) $(LINKER) -o $@ $?

main.o: main.c

clean:
       $(RM) $(PROG)
       $(RM) $(OBJS)
```

Sample Program Source Code

```
/*
 *   This sample draws sprites on the screen in a bouncing
 *   pattern. For each press of the up or down button of the
 *   controller the code dynamically maps a 16x16 ball
 *   texture onto each sprite using colors from the currently
 *   loaded color look-up table (CLUT). Pressing the up or down
 *   directional buttons on the Controller increases
 *   or decreases the number of displayed sprites.
 *
 *   This sample also reads sound data from the boot disk
 *   and plays it back as background music (BGM).
 *
 *
 *   Copyright (C) 1996 Sony Computer Entertainment Inc.
 *   All Rights Reserved
 *
 */


/*---- Includes ----*/
#include <libps.h>
#include "pad.h"
```

```
#include "balltex.h"  /* sprite ball texture pattern */


/*----    Macros ----*/

/* Basic setup */
#define KANJI          /*  Switch for displaying Kanji
*/
#define OT_LENGTH   1  /*  Ordering table will be 1 level deep  */
#define MAXOBJ 1500    /*  Maximum number of Sprites(balls)  */
#define MVOL   127     /*  Main volume level  */
#define SVOL   127     /*  SEQ volume level  */
#define DFILE  3       /*  File number  */

/*  Addresses where sound data will be loaded  */
#define VH_ADDR     0x80090000
#define VB_ADDR     0x800a0000
#define SEQ_ADDR    0x80110000

/*  Macros relating to display area */
#define FRAME_X     320 /* Display area size(horizontal) */
#define FRAME_Y     240 /* Display area size(vertical) */

#define WALL_X (FRAME_X-16) /* Sprite movement area size(horizontal)*/
#define WALL_Y (FRAME_Y-16) /* Sprite movement area size(vertical)  */

/*  Range check macro */
#define limitRange(x, l, h)
((x)=((x)<(l)?(l):(x)>(h)?(h):(x)))

/*---- Global variables ----*/

GsOT WorldOT[2];    /* Two ordering tables. One for each buffer */
GsOT_TAG  OTTags[2][1<<OT_LENGTH]; /* Ordering table entries */
                                   /* One array for each buffer */


/*  Primitive related variables */
PACKET    GpuPacketArea[2][MAXOBJ*(20+4)]; /* Workarea for primitives */
GsSPRITE  sprt[MAXOBJ];                    /* Sprite Primitives */

/*  Sprite movement related variables */
typedef struct {
     u_short x, y;   /* 2D Location */
     u_short dx, dy; /* Speed */
} POS;

/*  Controller related variables */
volatile u_char *bb0, *bb1;

/*  File reading related variables */
typedef struct {
     char *fname;
```

```
    void *addr;
    CdlFILE   finfo;
} FILE_INFO;

static FILE_INFO dfile[DFILE] = {
    {"\\DATA\\SOUND\\STD0.VH;1", (void *)VH_ADDR, 0},
    {"\\DATA\\SOUND\\STD0.VB;1", (void *)VB_ADDR, 0},
    {"\\DATA\\SOUND\\SAMPLE1.SEQ;1", (void *)SEQ_ADDR, 0},
};

short vab, seq;                          /* Sound related variables */

/*-- Prototypes ----*/
static void init_prim();     /* Sprite graphics relatedinitialization */
static void init_point(POS *pos);/* Sprite movement initialization */
static long pad_read(long n);    /* Controller state analysis */
static u_long PadRead(long id);  /* Get controller state */
static void datafile_search();   /* Retrieve file on CD-ROM */
static void datafile_read();     /* Read file on CD-ROM */
static void init_sound();   /* Sound on memory playback initilization */
static void play_sound();   /* Sound playback start  */
static void stop_sound();   /* Sound playback termination */

/*---- Main functions ----*/
main()
{
    int  nobj = 1;      /* Number of sprites displayed (from1)*/
    GsOT *ot;           /* Pointer to drawing OT */
    int  i, cnt, x, y;  /* Working variables*/
    int  activeBuff;
    GsSPRITE  *sp;
    POS  pos[MAXOBJ];
    POS  *pp;

    SetVideoMode( MODE_NTSC ); /* NTSC Mode */
/*  SetVideoMode( MODE_PAL );  /* PAL Mode (for European televisions*/

    GetPadBuf(&bb0, &bb1); /* Get controller reception buffer */
    datafile_search();     /* Data file retrieval  */
    datafile_read();       /* Data file reading  */

    GsInitGraph(320,240,4,0,0); /* Initializes the graphics system  */
                  /* Turn on the GPU, Set background color to black */
                  /* and initializes screen coordinates. */

    GsDefDispBuff(0, 0, 0, 240); /* Initializes the double buffers */
                                 /* in memory and specifies clipping */
                                 /* parameters.  */
    /* For (0,0)-(320,240),   display (0,240)- (320,480)(db[0]) */
    /* For (0,240)-(320,480), display   (0,0)- (320,240)(db[0])*/

    /* Ordering table information setting */
    for (i = 0; i < 2; i++)
    {
```

```
        WorldOT[i].length = OT_LENGTH;
        WorldOT[i].org = OTTags[i];
    }

    /* Font setting */
#ifdef KANJI        /* In the case of Kanji display */
    KanjiFntOpen(160, 16, 256, 240, 704, 0, 768, 256, 0,
512);
#endif
    FntLoad(960, 256);
/* Load basic font pattern in frame buffer */
    FntOpen(16, 16, 256, 200, 0, 512);
/* Font display location setting */

    init_prim();   /* Primitive buffer initial setting */
    init_point(pos); /* sprite movement initial setting*/
    init_sound();  /* Sound initial setting  */
    play_sound();  /* Sound playback start  */

    /* Main loop */
    while ((nobj = pad_read(nobj)) > 0)
    {
        /* Double buffer switch */
        activeBuff = GsGetActiveBuff();
        GsSetWorkBase((PACKET *)GpuPacketArea[activeBuff]);

        /* Ordering table clear */
        GsClearOt(0, 0, &WorldOT[activeBuff]);

        /* Sprite location update and registration to OT */
        sp = sprt;     pp = pos;

        for (i = 0; i < nobj; i++, sp++, pp++)
        {
            /* Update the horizontal coordinate values */
            if ((x = (pp->x += pp->dx) % WALL_X*2) >=
WALL_X)
                x = WALL_X*2 - x;

            /* Vertical coordinate value update */
            if ((y = (pp->y += pp->dy) % WALL_Y*2) >=
WALL_Y)
                y = WALL_Y*2 - y;

           /* Set new coordinate value for sprite primitive
*/
           sp->x = x;     sp->y = y;

           /* Registration to sprite primitive ordering
table */
           GsSortFastSprite(sp, &WorldOT[activeBuff], 0);
        }

        DrawSync(0);    /* Waiting for end of drawing */
```

```
        /* Wait for vertical synchronisation interrupt */
        cnt = VSync(0);

        /* Swap double buffers to get ready for the next
frame */
        GsSwapDispBuff();

       /* Registers the clear command to the primitive
ordering table*/
        GsSortClear(60, 120, 120, &WorldOT[activeBuff]);

        /* Drawing of primitive registered in OT */
        GsDrawOt(&WorldOT[activeBuff]);

        /* Printing number of balls and elapsed time */
#ifdef KANJI
        KanjiFntPrint("Num  =%d\n", nobj);
        KanjiFntPrint("Time =%d\n", cnt);
        KanjiFntFlush(-1);
#endif
        FntPrint("sprite = %d\n", nobj);
        FntPrint("total time = %d\n\n\n", cnt);
        FntPrint("UP    : INCREASE\n");
        FntPrint("DOWN  : DECREASE\n");
        FntPrint("L1    : PAUSE\n");
        FntPrint("SELECT: END\n");
        FntFlush(-1);
    }    /* Main loop terminal */

    /* Execute this by pressing select button and verifying
*/
    stop_sound();  /* Sound playback termination */
    return(0);     /* Program termination */

} /* end main */


/* Ball pattern graphics related initialization */
static void init_prim()
{
    GsSPRITE  *sp;
    u_short   tpage;
    RECT rect;
    int  i;

    rect.x = 640;  rect.y = 0;
    rect.w = 16/4; rect.h = 16;
    /* Load the sprite pattern into video ram */
    LoadImage(&rect, ball16x16);
    /* Get a handle to the texture page */
    tpage = GetTPage(0, 0, 640, 0);

    for (i = 0; i < 32; i++)
```

```
        {
            rect.x = 0;    rect.y = 480+i;
            rect.w = 256;  rect.h = 1;
            LoadImage(&rect, ballcolor[i]);
        }

        /* Sprite initialization */
        for (sp = sprt, i = 0; i < MAXOBJ; i++, sp++)
        {
            sp->attribute = 0;
            sp->x = 0;
            sp->y = 0;
            sp->w = 16;
            sp->h = 16;
            sp->tpage = tpage;
            sp->u = 0;
            sp->v = 0;
            sp->cx = 0;
            sp->cy = 480+(i%32);
            sp->r = sp->g = sp->b = 0x80;
            sp->mx = 0;
            sp->my = 0;
            sp->scalex = ONE;
            sp->scaley = ONE;
            sp->rotate = 0;
        }
    } /* end init_prim */

    /* Ball pattern movement related initialization */
    static void init_point(POS *pos)
    {
        int  i;

        for (i = 0; i < MAXOBJ; i++)
        {
            pos->x  = rand();   /* Start coordinate X  */
            pos->y  = rand();   /* Start coordinate Y  */

            /* Movement distance X 1<=x<=4) */
            pos->dx = (rand() % 4) + 1;

            /* Movement distance Y 1<=y<=4) */
            pos->dy = (rand() % 4) + 1;
            pos++;
        }

    } /* end init_point */

    /*---- Reading and analysis of controller state ----*/

    /*   Return value -1  : At time of pressing select button and verifying
         1st argument + 4 : At time of pressing up button and verifying
         1st argument - 4 : At time of pressing down button and verifying
         Pause in function while pressing L1 key
```

```
*/

static long pad_read(long n)
{
     u_long padd = PadRead(1);     /* Controller reading */

     if (padd & PADLup)   /* Up directional button */
         n += 4;
     if (padd & PADLdown) /* Down directional button */
         n -= 4;

     if (padd & PADL1)    /* Pause */
         while (PadRead(1)&PADL1);

     if(padd & PADselect)
         return(-1);      /* Program termination */

     /* n is given value 1<=n<=(MAXOBJ-1) */
     limitRange(n, 1, MAXOBJ-1);

     return(n);
} /* end pad_read */

/* Controller state reading */
static u_long PadRead(long id)
{
    return(~(*(bb0+3) | *(bb0+2) << 8 | *(bb1+3) << 16 |
    *(bb1+2) << 24));
} /* end Pad_Read */

/*---- Reading the file on CD-ROM (DFILE) ----*/
static void datafile_search()
{
    int i, j;

    for (i = 0; i < DFILE; i++)
    {    /* Deal with DFILE file */

        for (j = 0; j < 10; j++)
        {    /* Return loop */
            if (CdSearchFile(&(dfile[i].finfo),
dfile[i].fname)!=0)
                break;

            /* Retry loop interruption on normal termination
*/
            else
                printf("%s not found.\n", dfile[i].fname);
        }
    }
} /* end datafile search */

/* CD-ROM file reading */
static void datafile_read()
```

```
{
     int i, j;
     int cnt;

     for (i = 0; i < DFILE; i++)
     {    /* Deal with DFILE file */
         for (j = 0; j < 10; j++)
         {   /* Retry loop */
             CdReadFile(dfile[i].fname,
                        dfile[i].addr,
                        dfile[i].finfo.size);

/* Normal processing can be executed by other side of read */
/* Here, remaining sector number is monitored until Read terminated */

             while ((cnt = CdReadSync(1, 0)) > 0 )
                 VSync(0); /* Waiting for vertical synchronisation */
                           /* interrupt (for time adjustment */

             if (cnt == 0)
                 break; /* Retry loop interruption on normal */
                        /* termination*/

         } /* end for number of read tries */
     } /* end for number of files */
} /* end datafile_read() */


/*---- Sound related ----*/

/* Sound data on memory playback preparation */
static void init_sound()
{
     /* VAB opening and transmission to sound buffer */
     vab = SsVabTransfer( (u_char*)VH_ADDR,
                        (u_char*)VB_ADDR, -1, 1 );
     if (vab < 0)
     {
         printf("SsVabTransfer failed (%d)\n", vab);
         return;
     }

     /* SEQ opening */
     seq = SsSeqOpen((u_long *)SEQ_ADDR, vab);

     if (seq < 0)
         printf("SsSeqOpen failed (%d)\n", seq);

} /* end init_sound */

/* Sound playback start */
static void play_sound()
{
     SsSetMVol(MVOL, MVOL);               /* Set Main volume */
```

```
     SsSeqSetVol(seq, SVOL, SVOL);         /* Set volume for each SEQ */
     SsSeqPlay(seq, SSPLAY_PLAY, SSPLAY_INFINITY);/*Playback switch ON*/
} /* end play_sound */

/* Sound playback termination */
static void stop_sound()
{
     SsSeqStop(seq);     /* Playback switch OFF */
     VSync(0);
     VSync(0);
     SsSeqClose(seq);    /* SEQ close */
     SsVabClose(vab);    /* VAB close */
} /* end stop_sound */

/* end main.c */
```

Additional Reading

The following books may be useful references to help you create Net Yaroze programs.

C Programming

Title	Publisher	ISBN
The C Programming Language	Prentice Hall	ISBN 0-13-110362-8
Teach Yourself C	McGraw-Hill	ISBN 0-07-882011-1
Programming with GNU Software	O'Reilly	ISBN 0-56592-112-7

Graphics

Title	Publisher	ISBN
3D Computer Graphics	O'Reilly	ISBN 0-201-63186-5
Encyclopedia of Graphic File Formats	Addison-Wesley	ISBN 1-56592-058-9