SENSORAY CO., INC.

PC/104 Frame Grabber

Model 512

August 29, 2001

7313 SW Tech Center Dr.
Tigard, OR 97223
Phone 503.684.8073 • Fax 503.684.8164
sales@sensoray.com
www.sensoray.com

TABLE OF CONTENTS

1. Limited Warranty

Sensoray Company, Incorporated (Sensoray) warrants the model 512 hardware to be free from defects in material and workmanship and perform to applicable published Sensoray specifications for two years from the date of shipment to purchaser. Sensoray will, at its option, repair or replace equipment that proves to be defective during the warranty period. This warranty includes parts and labor.

The warranty provided herein does not cover equipment subjected to abuse, misuse, accident, alteration, neglect, or unauthorized repair or installation. Sensoray shall have the right of final determination as to the existence and cause of defect.

As for items repaired or replaced under warranty, the warranty shall continue in effect for the remainder of the original warranty period, or for ninety days following date of shipment by Sensoray of the repaired or replaced part, whichever period is longer.

A Return Material Authorization (RMA) number must be obtained from the factory and clearly marked on the outside of the package before any equipment will be accepted for warranty work. Sensoray will pay the shipping costs of returning to the owner parts which are covered by warranty.

Sensoray believes that the information in this manual is accurate. The document has been carefully reviewed for technical accuracy. In the event that technical or typographical errors exist, Sensoray reserves the right to make changes to subsequent editions of this document without prior notice to holders of this edition. The reader should consult Sensoray if errors are suspected. In no event shall Sensoray be liable for any damages arising out of or related to this document or the information contained in it.

EXCEPT AS SPECIFIED HEREIN, SENSORAY MAKES NO WARRANTIES, EXPRESS OR IMPLIED, AND SPECIFICALLY DISCLAIMS ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. CUSTOMER'S RIGHT TO RECOVER DAMAGES CAUSED BY FAULT OR NEGLIGENCE ON THE PART OF SENSORAY SHALL BE LIMITED TO THE AMOUNT THERETOFORE PAID BY THE CUSTOMER. SENSORAY WILL NOT BE LIABLE FOR DAMAGES RESULTING FROM LOSS OF DATA, PROFITS, USE OF PRODUCTS, OR INCIDENTAL OR CONSEQUENTIAL DAMAGES, EVEN IF ADVISED OF THE POSSIBILITY THEREOF.

2. Special Handling Instructions

The Model 512 board contains CMOS circuitry that is sensitive to Electrostatic Discharge (ESD). Special care should be taken in handling, transporting, and installing the 512 to prevent ESD damage to the board. In particular:

- Do not remove the 512 from its protective antistatic bag until you are ready to install it in your computer.
- Handle the 512 only at grounded, ESD protected stations.
- Always turn off the computer before installing or removing the 512 board

3. Introduction

The Sensoray Model 512 is an MPEG video encoder decoder board. Some of the feature include:

General

- Real time MPEG-2 and MPEG-1 video encoder and decoder
- Support for variable bit rate and constant bit rate
- IPB pictures to 15Mbps for constant bit rate and 10Mbps for variable bit rate
- Supports multiple resolutions (704x480, 640x480, 352x240, etc.)
- Support for NTSC, PAL
- Composite and S-Video inputs and outputs
- During encoding and standby, video input is fed to output for easy adjustment
- Onboard linear 16-bit audio CODEC (uncompressed audio)
- PC/104 form factor
- Either BNC connectors or header for video input and output
- Either mini phone jacks or header for audio input and output
- 3 digital TTL level digital inputs and/or outputs
- Low power
- Windows WDM driver

Video encoder

- Generates 13818 (MPEG-2) and 11172 (MPEG-1) compliant elementary streams (ES)
- Operates up to 30 frames per second
- Selectable bit rate

Video decoder

- Decodes both MPEG-1 and MPEG-2 streams
- Horizontal and vertical scaling

Driver features

- Windows WDM Sensoray sxDriver
    - Operation systems: Windows 98 (second edition), Windows 2000
    - Supports all Sensoray ISA, PC/104, PCI and Compact PCI devices
    - Easy calls to functions that return or send blocks of compressed video to and from the 512 hardware
    - Easy calls to functions that return or send blocks of uncompressed audio to and from the 512 hardware
    - Status functions
    - Digital I/O functions
    - Command line utility to set up video parameters such as bit rate, resolution, etc.

4. Hardware Configuration

The 512 is installed on a PC/104 stack. Four mounting holes are provided, one in each corner, for mechanical stability. The bus connectors are the large ones as seen on the bottom of figure 1.

The base address of the 512 must be selected using the base address selection jumper, as seen in the lower left hand corner of figure 1. Address selection will be discussed in a following section.

Video, audio, and digital I/O are present on five connectors. The header, J1, has all I/O signals, including S-Video, composite video, audio, and digital. J27, J28 are for composite video only and J26, J26 are for audio only.

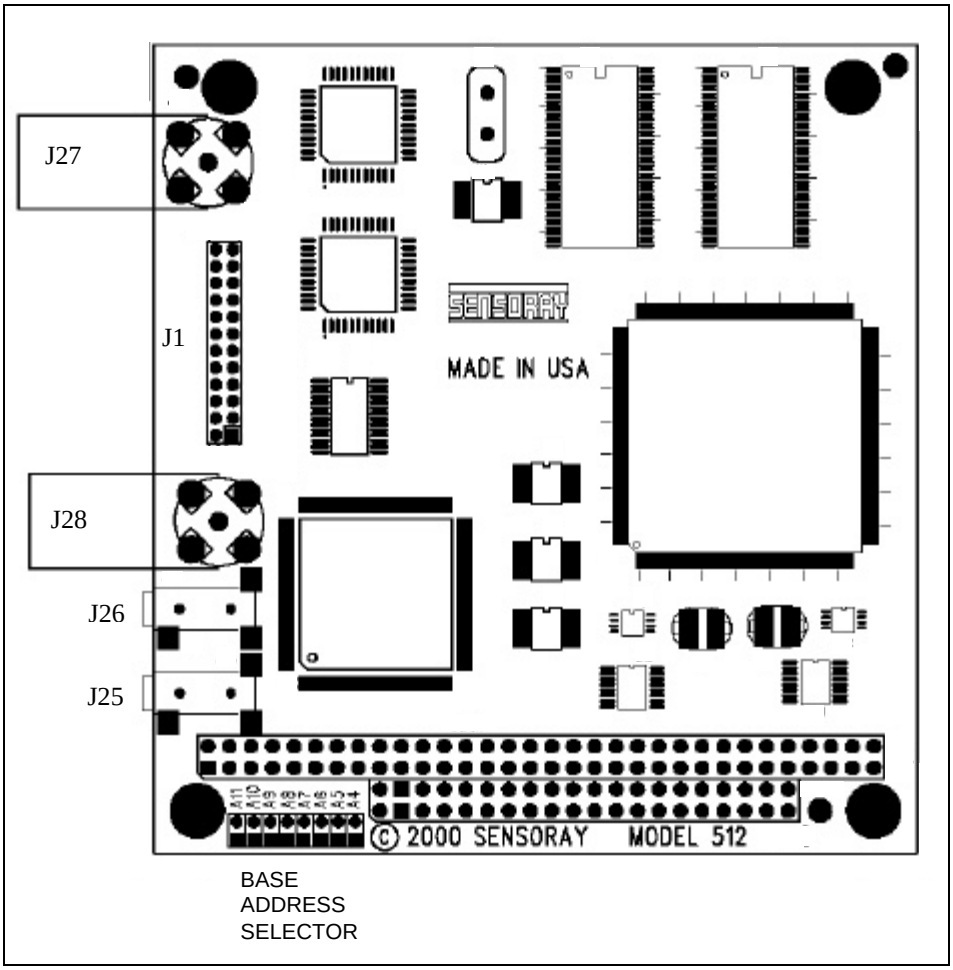

Figure 1 Board Layout

4.1 Base address selection

The jumpers marked A6 through A11 are for address selection. A “1” in the table below indicates an installed jumper. A ‘0’ is no jumper. Using the table the user can determine the base address of the 512 in hexadecimal.

A11	A10	A9	A8	First digit of address		A7	A6	Second digit of address		Third digit of address
0	0	0	0	F		0	0	C		0
0	0	0	1	E		0	1	8		
0	0	1	0	D		1	0	4		
0	0	1	1	C		1	1	0		
0	1	0	0	B						
0	1	0	1	A						
0	1	1	0	9						
0	1	1	1	8						
1	0	0	0	7						
1	0	0	1	6						
1	0	1	0	5						
1	0	1	1	4						
1	1	0	0	3						
1	1	0	1	2						
1	1	1	0	1						
1	1	1	1	0						

Table 1 Base Address Selection

Example 1:

A11, A10 not installed, A9, A8 installed (First digit is C)
A7,A6 not installed (Second digit is C)
The third digit is always zero, hence we have a base address of CC0 hex

Example 2:

We need a base address of F80 hex.
The first digit is F, therefore jumpers A11,A10,A9,A8 must be removed.
The second digit is 8, therefore jumper A7 must be removed and jumper A6 installed.

If you have two 512’s in your system they must have different base addresses. Be sure that you are not using an address used by another device in your system.

VERY IMPORTANT: For normal operation a jumper must be installed at position A5 and no jumper installed at position A4.

4.2. Connectors

The header J1 carries all the I/O signals. The pinout is given in Table 2.

Pin	Function	Pin	Function
1	Digital I/O 2	2	+5V
3	Digital I/O 1	4	Digital ground
5	Digital I/O 0	6	Digital ground
7	Audio in (Microphone level)	8	Audio ground
9	Audio out (Line level)	10	Audio ground
11	S-Video out – Y	12	Video ground
13	S-Video out – C	14	Video ground
15	Composite video out	16	Video ground
17	Composite video in 4 / S-Video 1 – C	18	Video ground
19	Composite video in 3 / S-Video 2 – C	20	Video ground
21	Composite video in 2 / S-Video 1 – Y	22	Video ground
23	Composite video in 1 / S-Video 2 – Y	24	Video ground

Table 2 Header J1 Pinout

The BNC connectors (if installed) are for composite video. J27 is composite video out and J28 is composite video in 1. The center pin is the signal and the out shell is video ground.

The 1/32” mini-phone jacks (if installed) are for audio. J25 is line level audio out and J26 is audio in.

5. Software Components

The driver's executable software components must be correctly installed on the target system to ensure proper functioning of the
Model 626 driver. The components that are required, and their locations in your system, are shown in Table 1.

5.1. Installation Procedure

The executable software components must be correctly installed on the target system to ensure proper functioning of the driver. The components that are required, and their locations in your system, are shown in Table 3.

Filename	Directory	Function
X12.DLL	SYSTEM	Model x12 API
SXDRVR98.SYS	SYSTEM32\DRIVERS	Kernel-mode driver

Table 3 Required driver components

1. Open "Control Panel | Add New Hardware" dialog, click "Next".
2. Select "No, I want to select the hardware from a list".
3. Select the type of hardware: "Other devices"
4. Click "Have Disk"
5. Browse the location of sx12.inf file
6. Select "Sensoray Model X12 Frame Grabber"
7. You will see the list of resources (Input/Output Range), acceptable for the device. Select one and set the same base address on the board (see Table 1).
8. Turn power off
9. Install the device on a PC/104 stack.
10. Reboot the system.

6. Building an Application with SX12 DLL

The following files are distributed with `sx12.dll`:
`sx12.h` – contains data types and constants definitions;
`sx12f.h` – contains exported functions declarations;
`sx12app.c` – contains exported functions and helper functions definitions.
When building an application with `sx12.dll`, it is necessary to include `sx12app.c` in the project.
If MFC is used, an option "Not using precompiled headers" must be set in the project settings for `sx12app.c`.
All modules containing calls to the `sx11.dll` functions must include `sx11f.h`. Please refer to the sample source code for an example of building an application with SX12 DLL.

7. Functions Reference

The SX12 DLL is designed to provide the application developer with full control over the frame grabber. All special data types used by the DLL are defined in sx12.h. The sample application illustrates the use of most of the functions and allows building a custom application within minutes.

7.1. DLL Load/Unload Functions

X12_DLLOpen

```
int X12_DLLOpen(void);
```

Parameters

None.

Return values

Returns 0 in case of success, or an error code.

Notes

Searches for the file SX12.DLL in the following sequence:

1. The directory from which the application loaded.
2. The current directory.
3. Windows 95:
   - The Windows SYSTEM directory.

   Windows NT:
   - The Windows SYSTEM32 directory.
   - The 16-bit Windows SYSTEM directory.
4. The Windows directory
5. The directories that are listed in the PATH environment variable.

X12_DLLClose

```
void X12_DLLClose(void);
```

Parameters

None.

Return values

None.

Notes

Closes the DLL.

7.2. DLL Exported Functions

X12_init

```
int X12_init(void);
```

Parameters

None.

Return values

Returns number of cards on success. Returns -1 if any errors were detected.

Notes

The function initializes the driver, searches for and initializes all boards supported by the DLL. This function has to be called **only once** per application. The system resources allocated by X12_init are released by a call to X12_cleanup.

X12_reset

```
int X12_reset(int CardId);
```

Parameters

CardId
: Card identifier. Must be between 0 and number of cards minus 1.

Return values

Returns 0 on success. Returns -1 if any errors were detected.

Notes

Resets the board. Writes new values to the board registers.

X12_set_input

```
void X12_set_input(int CardId,int input);
```

Parameters

CardId
: Card identifier. Must be between 0 and number of cards minus 1.

input
: Video input. Must be between 0 and 3.

Return values

None.

Notes

Select video input. Default is 0.

X12_set_input_type

```
void X12_set_input_type(int CardId,int type);
```

Parameters

CardId
: Card identifier. Must be between 0 and number of cards minus 1.

type
: Video input type. Must be 0 for Composite Video or 1 for S-Video.

Return values

None.

Notes

Sets video input type. Default is 0 (Composite Video).

X12_set_system

```
void X12_set_system(int CardId,int system);
```

Parameters

CardId
: Card identifier. Must be between 0 and number of cards minus 1.

system
: Video system. Must be PAL (1) or NTSC (2).

Return values

None.

Notes

Set video system NTSC or PAL. Default is 2 (NTSC).

X12_set_bitrate

```
void X12_set_bitrate(int CardId,int rate);
```

Parameters

CardId
: Card identifier. Must be between 0 and number of cards minus 1.

rate
: Video bit rate in bits per second. Must between 700,000 and 10,000,000.

Return values

None.

Notes

Sets desired bit rate. Default is 4000000.

X12_set_picsize

```
void X12_set_picsize(int CardId,int size);
```

Parameters

CardId
: Card identifier. Must be between 0 and number of cards minus 1.

size
: Picture size. D1 MPEG-2 (0), 1/2 D1 MPEG-2 (1), 2/3 D1 MPEG-2 (2), or CIF MPEG-1 (3).

Return values

None.

Notes

Sets picture size and compression system:

- **D1, MPEG-2:** NTSC-704x480, PAL-704x576
- **1/2 D1, MPEG-2:** NTSC-352x480, PAL-352x576
- **2/3 D1, MPEG-2:** NTSC-480x480, PAL-480x576
- **CIF, MPEG-1:** NTSC-352x240, PAL-352x288

Default is 0 (D1).

X12_set_m

```
void X12_set_m(int CardId,int m);
```

Parameters

CardId
: Card identifier. Must be between 0 and number of cards minus 1.

m
: Number of frames in Group of Pictures.

Return values

None.

Notes

Sets number of frames in Group of Pictures. Default is 3.

X12_set_n

```
void X12_set_n(int CardId,int n);
```

Parameters

CardId
: Card identifier. Must be between 0 and number of cards minus 1.

n
: Distance between I/P frames.

Return values

None.

Notes

Sets distance between I/P frames. Default is 15.

X12_set_vbr

```
void X12_set_vbr(int CardId,int vbr);
```

Parameters

CardId
: Card identifier. Must be between 0 and number of cards minus 1.

vbr
: Enables (1) or disables (0) variable bit rate compression

Return values

None.

Notes

Enables/disables variable bit rate compression. Default is 0.

Important:

All **X12_set_*xx*** functions do not change the actual values of the board registers. Use **X12_reset** function to refresh the values.

X12_read_gpio

```
int X12_read_gpio(int CardId);
```

Parameters

CardId
Card identifier. Must be between 0 and number of cards minus 1.

Return values

State of GPIO

Notes

Reads state of GPIO pins: active input is low (when input is pulled down bit is set to 1).

X12_write_gpio

```
void X12_write_gpio(int CardId,int state);
```

Parameters

CardId
Card identifier. Must be between 0 and number of cards minus 1.
state
State of GPIO

Return values

None.

Notes

Sets state of GPIO pins: active output is low (output is pulled down when bit is set to 1).

X12_get_framecount

```
int X12_get_framecount(int CardId);
```

Parameters

CardId
Card identifier. Must be between 0 and number of cards minus 1.

Return values

Compression frame count.

Notes

Returns compression frames count.

X12_get_displayframecount

```
int X12_get_displayframecount(int CardId);
```

Parameters

CardId
Card identifier. Must be between 0 and number of cards minus 1.

Return values

Decompression frame count.

Notes

Returns decompression frames count.

X12_get_droppedframes

```
int X12_get_droppedframes(int CardId);
```

Parameters

CardId
Card identifier. Must be between 0 and number of cards minus 1.

Return values

Number of dropped frames.

Notes

Returns number of frames dropped during compression.

X12_get_vqsize

```
int X12_get_vqsize(int CardId);
```

Parameters

CardId
Card identifier. Must be between 0 and number of cards minus 1.

Return values

Amount of data in video queue.

Notes

Returns amount of data in video queue.

X12_get_port

```
int X12_get_port(int CardId);
```

Parameters

CardId
Card identifier. Must be between 0 and number of cards minus 1.

Return values

I/O base port address on the bus for given board.

Notes

Returns base I/O port address set during installation process. Actual address of the board must be set manually (see Table 1).

X12_read

```
int X12_read(int CardId,char *buffer,int nbytes);
```

Parameters

CardId
Card identifier. Must be between 0 and number of cards minus 1.
buffer
Pointer to input video buffer.
nbytes
Number of bytes to input or -1 if any errors were detected.

Return values

None.

Notes

Attempts to read "nbytes" bytes of MPEG data from the SM2210 output FIFO. The actual amount read is returned. If no data is available “0” is returned unless blocking has been enabled. In which case this function will wait until at least one byte of data is made available. If this is the first read the encoder is initialized and started.

X12_write

```
int X12_write(int CardId,char *buffer, int nbytes);
```

Parameters

CardId

Card identifier. Must be between 0 and number of cards minus 1.

buffer

Pointer to output video buffer.

nbytes

Number of bytes to output or -1 if any errors were detected.

Return values

None.

Notes

Attempts to write "nbytes" bytes of MPEG data to the SM2210 input FIFO. The actual amount written is returned. If this is the first write the decoder is initialized and started.

X12_cleanup

```
void X12_cleanup(void);
```

Parameters

None.

Return values

None.

Notes

Resets all boards and closes the driver.