Rockwell Semiconductor Systems

WaveArtist™ Host Interface

Reference Manual (Preliminary)

Order No. 1112
August 5, 1996

NOTICE

Table of Contents

1. INTRODUCTION 1-1

2. PNP INTERFACE 2-1

2.1.1 Auto-Configuration Registers 2-1
2.1.2 PnP Registers 2-1
Logical Device 0: WaveArtist Command Interface Requirement 2-1
Logical Device 1: Sound Blaster Pro Interface Requirement 2-1
Logical Device 2: MPU401 Interface Requirement 2-2
Logical Device 3: Modem COM Port Interface Requirement 2-2
Logical Device 4: EIDE CD-ROM Interface Requirement 2-2
Logical Device 5: Game Port Interface Requirement 2-2
2.2 Operation 2-2
2.2.1 Configuring 2-2
2.2.2 Interrupt Requests 2-2
2.2.3 PnP Resource Data 2-6

3. SOUND BLASTER PRO EMULATION INTERFACE 3-1

3.1 Sound Blaster Pro Registers 3-1
3.2 Sound Blaster Pro Registers Description 3-2
SB+0, +2, +8: FM Music Status Registers (R)/ Address Registers (W) 3-2
SB+1, +3, +9: FM Music Data Registers (R/W) 3-2
SB+4: Mixer Address Register (R/W) 3-2
SB+5 - SB Mixer Data Register (R/W) 3-3
SB+6: SB Reset Register (W) 3-3
SB+A: SB Read Data Register (R) 3-3
SB+C: SB Command/Data Register (W) 3-3
SB+C: SB Write Buffer Status Register (R) 3-3
SB+E: SB Read Buffer Status Register (R) 3-3

4. MPU-401 INTERFACE 4-1

4.1 MPU-401 Interface Registers 4-1
4.2 MPU-401 Registers Description 4-1
MB+0: MPU-401 Data Register (R/W) 4-1
MB+1: MPU-401 Command Register (W) 4-1
MB+2: MPU-401 Status Register (R) 4-2
4.3 MIDI Supported Messages 4-2
4.4 Rockwell System Exclusive (SYSEX) Sample Download Format. 4-6

5. GAME PORT INTERFACE 5-1

5.1 Game Port Register 5-1
5.2 Game Port Register Description 5-1
JB: Game Port Register 5-1

6. WAVEARTIST COMMAND/STATUS INTERFACE 6-1

6.1 WaveArtist Registers 6-1
6.2 WaveArtist Registers Description 6-2
WB+0: WaveArtist Command /Response Register (CMDR) 6-2
WB+2: WaveArtist Data Register (DATR) 6-2
WB+4: WaveArtist Control Register (CTRLR) 6-2
WB+5: WaveArtist Status Register (STATR) 6-3
WB+C: WaveArtist Expansion Control Register 1 (IRQSTAT) 6-4
6.3 WaveArtist Commands 6-5

6.4 WaveArtist Command Set Definitions 6-6
00H - System ID and Revision Request 6-6
01H - Downloadable DRAM Size Request 6-6
N09H - WaveArtist ROM, RAM, and Timer Tests 6-6
10H - Set Sample Width/Format for Data Path to PC 6-7
11H - Set Number of Channels for Data Path to PC 6-7
12H - Set Sample Speed Time Constant for Data Path to PC 6-8
13H - Assign ADC Data Transfer to PC Path 6-8
14H - Set Sample Count for Block Data Transfer to PC 6-9
15H - Enable ADC Data Transfer to PC 6-10
16H - Pause ADC Data Transfer to PC 6-10
17H - Stop ADC Data Transfer to PC 6-10
18H - Resume ADC Data Transfer to PC 6-10
20H - Set Sample Width/Format for Data Path from PC 6-11
21H - Set Number of Channels for Data Path from PC 6-11
22H - Set Sample Speed Time Constant for Data Path from PC 6-12
23H - Assign DAC Data Transfer from PC Path 6-12
24H - Set Sample Count for Block Data Transfer from PC 6-13
25H - Enable DAC Data Transfer from PC 6-14
26H - Pause DAC Data Transfer from PC 6-14
27H - Stop DAC Data Transfer from PC 6-14
28H - Resume DAC Data Transfer from PC 6-15
NN42H - Clear DMA Status Indicator(s) 6-15
6.5 WaveArtist Mixer and Volume Control Commands 6-16
NN30H - Read Mixer Control Word or Volume Setting 6-16
NN31H - Write New Volume Setting Pair 6-18
32H - Write New Mixer Control Setting Pair 6-18
33H - Reset Mixer Settings 6-18
NN34H - Select Mono Record Source 6-19
6.6 Extended Interface Commands 6-19
NN50H - Select MIDI Output Path 6-19
NN62N - Initiate DRAM Download 6-20
63H - Terminate DRAM Download 6-20

7. WINDOWS DRIVER SUPPORT FOR WAVETABLE DRAM DOWNLOAD 7-1

7.1 DRV_RWA_DOWNLOAD Message 7-1
7.2 RWA_DLOADINFO Structure 7-2
7.3 File Download Example 7-4

List of Tables

Table 2-1. PnP Auto-Configuration Ports 2-3
Table 2-2. WaveArtist PnP Registers........ 2-3
Table 2-3. WaveArtist PnP Register Descriptions 2-4
Table 3-1. Sound Blaster Pro Interface Register Map 3-1
Table 3-2. Mixer Index Registers........ 3-2
Table 4-1. MIDI Interface Register Map........ 4-1
Table 4-2. MPU-401 Commands 4-1
Table 4-3. MIDI Implementation Chart........ 4-3
Table 4-4. General MIDI Sound Set Program Number Assignment 4-4
Table 4-5. General MIDI Drum Kit Note Assignment........ 4-5
Table 5-1. Joystick Interface Register Map 5-1
Table 6-1. WaveArtist Interface Register Map........ 6-1
Table 6-2. WaveArtist Commands........ 6-5
Table 6-3. Mixer Control Bit Definitions........ 6-17

This page is intentionally blank.

1. INTRODUCTION

1.1 Scope

This manual defines the WaveArtist registers and associated commands, status, and data accessible by the host PC over the ISA bus. The registers include the PnP registers as well as the registers associated with the supported logical devices.

The Windows driver support for wavetable DRAM download is also described.

1.2 Rockwell Reference Documents

Title	Order No.
WaveArtist 100, WaveArtist 200, and WaveArtist 300 Audio System Devices Designer's Guide	1104
WaveArtist 010 Audio System Device Designer's Guide	1101

This page is intentionally blank.

2. PnP INTERFACE

This section summarizes the WaveArtist PnP registers. This summary refers to general operation of a PnP card interface as stated in the PnP ISA specification and supported by the WaveArtist. Please refer to the Plug and Play ISA Specification for additional information.

2.1.1 Auto-Configuration Registers

Three 8-bit registers are used by the host software to access the configuration space on each PnP ISA card. They are the ADDRESS port, the READ_DATA port, and the WRITE_DATA port. These ports are listed in Table 2-1.

The PnP registers are accessed by first writing the address of the desired register to the ADDRESS port, followed by a read of data from the READ_DATA port or a write of data to the WRITE_DATA port. A write to the ADDRESS port may be followed by any number of WRITE_DATA or READ_DATA accesses to the same register location without the need to re-write to the ADDRESS port before each access.

The ADDRESS port is also the write destination of the initiation key.

The WRITE_DATA port is used to write information to the PnP registers. The destination of the data is determined by the last setting of the ADDRESS port.

The READ_DATA port is used to read information from the PnP registers. The source of the data is determined by the last setting of the ADDRESS port.

The address of the READ_DATA port is set by writing the proper value to Set RD_DATA register.

2.1.2 PnP Registers

The PnP registers and reserved addresses are listed in Table 2-2. PnP registers functional in the WaveArtist are described in Table 2-3.

PnP card register space is divided into three parts: card control, logical device control, and logical device configuration.

Card control registers are used for global functions that control the entire card. One set of these registers is provided on a PnP card.

Logical device control registers and configuration registers are repeated for each logical device supported by a PnP card. Logical device configuration registers are used to program the device's ISA bus resource use.

The WaveArtist (RWA010, RWA011, RWA100 and RWA300) supports six logical devices:

- Logical Device 0: WaveArtist Command Interface (8-bit and 16-bit stereo audio full-duplex, mixer and volume controls, wavetable sample memory bus access and diagnostic).
- Logical Device 1: Sound Blaster Pro Interface (8-bit stereo audio with FM synthesis for DOS game compatibility)
- Logical Device 2: MPU401 Interface (MIDI UART and controller based wavetable synthesis (RWA300 only)
- Logical Device 3: Modem COM Port Interface
- Logical Device 4: EIDE CD-ROM Interface
- Logical Device 5: Game Port Interface

Logical Device 0: WaveArtist Command Interface Requirement

- 16 contiguous register addresses
- A 16-bit DMA channel (an 8-bit DMA channel shared with Sound Blaster Pro for full-duplex) from a choice of channels 5, 6, or 7
- An IRQ (shared with Sound Blaster Pro but not requested in this logical device descriptor)

Logical Device 1: Sound Blaster Pro Interface Requirement

- Two address ranges: 16 contiguous registers (typically the base address is either at 0220h or 0240h) and 4 contiguous registers at 0388h for FM synthesis AdLib compatibility.
  - An 8-bit DMA channel (hardwired to channel 1 for backward compatibility with Sound Blaster)
  - An IRQ (5, 7, or 10) (This IRQ is shared with WaveArtist Command Interface logical device)

Logical Device 2: MPU401 Interface Requirement

- Four contiguous addresses (only the lower two addresses are used and the rest are alias).
- An IRQ (typically 5, 7, 9, 10, 11 or 15 with Win95, less choices with standard Win 3.1 MPU401 driver).
  - IRQ 9 is also a possible choice; however, it is in the suboptimal choice to avoid problems with a mouse using IRQ2 and IRQ9 was not properly reported as a conflict.
- Standard Win 3.1 MPU 401 driver only supports the 8 bit ISA slot IRQ choices.

Logical Device 3: Modem COM Port Interface Requirement

- 8 contiguous registers (COM1 to COM4 standard addresses)COM3 address choice is listed as first preference to avoid standard serial ports (COM1 and COM2) and possibly some video cards (COM4) conflict.
- An IRQ (3, 4, 5, 7, 10, 11 or 15)
- IRQ 5, 7, 10, 11 or 15 are listed as first choices to avoid standard serial ports conflict. Non standard COM IRQ choices are available for Win95 COM driver to increase the chance of first time installation.
- Manual configuration of COM port selection may have to be done if non of the above choices are available.
- Some old PnP BIOSs can be trouble some to configure the modem properly if the standard COM1 or COM2 are listed as first choices

Logical Device 4: EIDE CD-ROM Interface Requirement

- Currently being stubbed out and requesting no resources to avoid problems with secondary IDE support on the mother board or other peripheral cards.
- If enabled the supported resources are:
  - Two address ranges for control, status, and data registers (typically based at 0170h and 0376h)
  - An IRQ choice of 3, 4, 5, 7, 9, 10, 11 or 15 (typically 15)
  - Optional 16 bit DMA channel 5, 6, or 7

Logical Device 5: Game Port Interface Requirement

- One register address (typically at 0201h)

2.2 Operation

2.2.1 Configuring

Upon reset, the WaveArtist starts at the WAIT FOR KEY state. It detects the Initiation Key sequence issued by the ISA host and goes to the SLEEP state. The PC then sets the WaveArtist card and any other PnP conforming cards in the system in ISOLATION state. In this state, the WaveArtist reports the Serial ID stored in the Serial EEPROM. If it wins out in the isolation, the PC gives it a Card Select Number and puts it in the CONFIGURATION state. The PC reads the Resource Data in the Serial EEPROM, and configures the I/O Base Address, and Interrupt Registers.

2.2.2 Interrupt Requests

The WaveArtist can drive the following IRQ lines: IRQ3, IRQ4, IRQ5, IRQ7, IRQ9, IRQ10, IRQ11, and IRQ15.

The interrupt request level for the logical device is selected by writing the interrupt request level number to the Interrupt Level Select 0 register. This select number represents the number of the interrupt on the ISA bus. If no interrupt is assigned, the Interrupt Level Select register must be set to 0 (default).

The IRQ output type is fixed at High and Level in the Interrupt Request Type Select 0 register.

Table 2-1. PnP Auto-Configuration Ports

ISA Bus Address (Hex.)	ISA Bus Access (R/W)	Name	Comments
0279	W	ADDRESS	
0A79	W	WRITE_DATA	The WRITE_DATA port is used to write data to the PnP registers. The data destination is determined by the last setting of the ADDRESS port.
0203 to 03FF	R	READ_DATA	The READ_DATA port is used to read data from the PnP registers. The data source is determined by the last setting of the ADDRESS port. The address of the READ_DATA port is determined by the Set RD_DATA Port register.
* Relocatable from 0203 to 03FF. Bits 1 and 0 are fixed at 1.			

Table 2-2. WaveArtist PnP Registers

Register Address (Hex.)	ISA Bus Access (R/W)	Name	Comments
Card Control Registers			
00	W	Set RD_DATA Port	Returns value 00 on read.
01	R	Serial Isolation	
02	W	Config Control	Returns value 00 on read.
03	W	Wake[CSN]	Returns value 00 on read.
04	R	Resource Data	
05	R	Status	
06	R/W	Card Select Number	
07	R/W	Logical Device Number	Only value of 00-05 is valid.
08 - 2F	-	None	No storage elements. Returns value 00 on read.
Logical Device Control Registers			
30	R/W	Activate	
31	R/W	I/O Range Check	
32 - 3F	-	None	No storage elements. Returns value 00 on read.
Logical Device Configuration Registers			
40 - 5F	-	Memory Space Configuration	No storage elements. Returns value 00 on read.
60	R/W	I/O Port Base Address Bits 15:8 Descriptor 0	
61	R/W	I/O Port Base Address Bits 7:0 Descriptor 0	
62	R/W	I/O Port Base Address Bits 15:8 Descriptor 1	
63	R/W	I/O Port Base Address Bits 7:0 Descriptor 1	
64 - 6F	-	None	No storage elements. Returns value 00 on read.
70	R/W	Interrupt Request Level Select 0	Bits 7:4 are 0 on reads. For Bits 3:0, only value of 3, 4, 5, 7, 9, 10, 11, or 15 is valid.
71	R	Interrupt Request Type Select 0	Hard-wired to 03. Interrupt type is High and Level.
72 - 73	-	None	No storage elements. Returns value 00 on read.
74	R/W	DMA Channel Selector 0	Bits 7:3 are 0 on reads. For Bits 2:0, only value of 1, 5, 6, or 7 is valid.
75 - FE	-	Logical Device Configuration	No storage elements. Returns value 00 on read.
FF	-	Reserved	No storage elements. Returns value 00 on read.

Table 2-3. WaveArtist PnP Register Descriptions

Name	Register Address (Hex.)	ISA Bus Access (R/W)	Definition
Set RD_DATA Port	00	W	The RD_DATA Port command sets the address of the READ_DATA Port for all PnP cards. Write data bits 7:0 is used as ISA I/O bus address bits[09:02]. The ISA bus address bits 1:0 are fixed at binary "11." This command can only be used in the Isolation state. Returns value 00 on read. This register must be reinitialized after a RESET or Reset CSN command.
Serial Isolation	01	R	A read from the Serial Isolation register causes PnP cards in the Isolation state to respond to the ISA bus read cycle in accordance with the PnP Isolation Protocol. Cards that "lose" the isolation protocol will enter the Sleep state.
Config Control	02	W	The Config Control register consists of three independent commands which are activated by writing a "1" to their corresponding register bits. These bits are automatically reset to "0" by the WaveArtist after the command executes. Bit 2 Reset CSN command. Resets CSN to 0. Bit 1 Wait for Key command. Returns to the Wait for Key state. The CSN is preserved and logical device is not affected. Bit 0 Reset command. Resets the logical device and restores configuration registers to their power-up values. The logical device enters its default state and the CSN is preserved.
Wake[CSN]	03	W	A write to this register will cause all cards that have a CSN that matches the write data[7:0] to go from the Sleep state to the either the Isolation state if the write data is zero or the Config state if the write data is not zero. Also, the pointer to the Serial EEPROM is reset so that the next read of resource data will be from byte 0 of the EEPROM.
Resource Data	04	R	A read from this address reads the next byte of resource information from the Serial EEPROM when in the Config state.
Status	05	R	Bit[0], when set, indicates that the next byte of resource data is available to be read from the Resource Data register.
Card Select Number	06	R/W	A write to this port sets a card's CSN. The CSN is a value uniquely assigned to each ISA PnP card after the serial identification process so that each card may be individually selected during a Wake[CSN] command. A Card Select Number of zero represents an unidentified card. Valid Card Select Numbers for identified ISA cards range from 1 to 255 and are assigned sequentially starting from 1. The CSN is never set to zero using the CSN register. The CSN is set to zero by RESET or Reset CSN command.
Logical Device Number	07	R/W	The logical device number written to this register selects which logical device the following configuration commands will operate on. All reads and writes of memory, I/O, and interrupt configuration information access the registers of the logical device pointed to by this register. Since the WaveArtist supports six logical devices, 00-05 are the only valid numbers.
Activate	30	R/W	The Activate register is used to activate a logical device. An active logical device responds to all ISA bus cycles as per its normal operation. An inactive logical device does not respond to nor drive any ISA bus signals. Bit 0 A 1 selects logical device active; a 0 selects logical device inactive. Before a logical device is activated, I/O range check must be disabled. Bits 7:1 Reserved and returns 0 on read.

Table 2-3. WaveArtist PnP Register Descriptions (Cont'd)

Name	Register Address (Hex.)	ISA Bus Access (R/W)	Definition
I/O Range Check	31	R/W	The I/O Range Check register allows the PnP host software to determine if another card conflicts with the I/O port range that has been assigned to a logical device. The I/O range check works by having all I/O ranges that would be used by a logical device return 0x55 then 0xAA on I/O read commands.
I/O port base address bits 15:8 descriptor 0	60	R/W	Read/write value indicating the selected I/O lower limit address bits 15:8 for I/O descriptor 0. Bits[15:10] are not supported and return value 0 on read.
I/O port base address bits 7:0 descriptor 0	61	R/W	Read/write value indicating the selected I/O lower limit address bits[7:0] for I/O descriptor 0.
I/O port base address bits 15:8 descriptor 1	62	R/W	Read/write value indicating the selected I/O lower limit address bits 15:8 for I/O descriptor 1. Bits[15:10] are not supported and return value 0 on read.
I/O port base address bits 7:0 descriptor 1	63	R/W	Read/write value indicating the selected I/O lower limit address bits[7:0] for I/O descriptor 1.
Interrupt Request Level Select 0	70	R/W	Read/write value indicating the selected interrupt level. Bits 7:4 0 on reads. Bits 3:0 Selects which interrupt level is used for Interrupt 0. Only value of 3, 4, 5, 7, 9, 10, 11, or 15 is valid.
Interrupt Request Type Select 0	71	R	Read/write value indicating which type of interrupt is used for the Request Level selected above. Bit 1 Level: 1 = high (fixed) Bit 0 Type: 1 = level (fixed)
DMA Channel Selector 0	74	R/W	Read/write value indicating the selected DMA channel. Bits 7:3 0 on reads. Bits 2:0 Selects which DMA channel is used for DMA Channel Selector 0. Only value of 1, 5, 6, or 7 is valid.

2.2.3 PnP Resource Data

An example of PnP resource data is shown below.

Byte No. (Dec.)	**Data (Hex.)**	**Description**
1	4A	Vendor ID Name :
2	73	RSS
3	50	Vendor ID Product Number : 5000
4	00	
5	01	Serial Number: 00000001
6	00	
7	00	
8	00	
9	69	Checksum (LFSR)
10	0A	PnP Version and Card String ID
11	10	PnP Ver 1.0
12	00	Rockwell ver 0
13	82	Identifier string ANSI
14	13	LSB length
15	00	MSB length
16	52	R
17	6F	o
18	63	c
19	6B	k
20	77	w
21	65	e
22	6C	l
23	6C	l
24	20	
25	57	W
26	61	a
27	76	v
28	65	e
29	41	A
30	72	r
31	74	t
32	69	i
33	73	s
34	74	t
35	15	Logical Device ID
36	4A	Logical Device ID 0 WaveArtist Wave Audio
37	73	
38	50	RSS 5000
39	00	
40	02	IO Range Check Enabled Reg 31 No boot
41	30	Dependent Function Tag
42	47	I/O Port Descriptor
43	01	16 bit address decode
44	50	Min base low
45	02	Min base high (0250)
46	F0	Max base low
47	03	Max base high (03F0)
48	10	Alignment for min base address
49	10	Range length
50	2A	DMA Format
51	E0	Channel 5 or 6 or 7
52	12	16 bit, count by word
53	38	End Dependent Function Tag
54	15	Logical Device ID
55	4A	Logical Device ID 1 Sound Blaster
56	73	
57	50	RSS 5001
58	01	
59	02	IO Range Check Enabled Reg 31 No boot

60	30	Dependent Function Tag
61	2A	DMA Format
62	02	Channel 1
63	08	8 bit, count by byte
64	47	I/O Port Descriptor
65	01	16 bit address decode
66	20	Min base low
67	02	Min base high (0220)
68	40	Max base low
69	02	Max base high (0240)
70	20	Alignment for min base address
71	10	Range length
72	47	I/O Port Descriptor
73	01	16 bit address decode
74	88	Min base low
75	03	Min base high (0388)
76	88	Max base low
77	03	Max base high (0388)
78	04	Alignment for min base address
79	04	Range length
80	22	IRQ Format
81	A0	IRQ 5 or 7 or 10
82	04	
83	38	End Dependent Function Tag
84	15	Logical Device ID
85	4A	Logical Device ID 2 MPU401
86	73	
87	50	RSS 5002
88	02	
89	02	IO Range Check Enabled Reg 31 No boot
90	31	Dependent Function Tag
91	00	Preferred Configuration
92	47	I/O Port Descriptor
93	01	16 bit address decode address 330
94	30	Min base low
95	03	Min base high (0330)
96	30	Max base low
97	03	Max base high (0330)
98	04	Alignment for min base address
99	04	Range length
100	22	IRQ Format
101	A0	IRQ 5 or 7
102	8C	or IRQ 10 or 11 or 15
103	31	Dependent Function Tag
104	01	Acceptable Configuration
105	47	I/O Port Descriptor
106	01	16 bit address decode address
107	00	Min base low
108	03	Min base high (0300)
109	00	Max base low
110	03	Max base high (0300)
111	04	Alignment for min base address
112	04	Range length
113	22	IRQ Format
114	A0	IRQ 5 or 7
115	8C	or IRQ 10 or 11 or 15
116	31	Dependent Function Tag
117	02	Suboptimal Configuration
118	47	I/O Port Descriptor
119	01	16 bit address decode address
120	00	Min base low
121	03	Min base high (0300)
122	30	Max base low
123	03	Max base high (0330)
124	30	Alignment for min base address
125	04	Range length

126	22	IRQ Format
127	A0	IRQ 5 or 7
128	8E	or IRQ 9 or 10 or 11 or 15
129	38	End Dependent Function Tag
130	15	Logical Device ID
131	4A	Logical Device ID 3 Modem
132	73	RC288ACF/SP
133	01	RSS 0160
134	60	
135	02	IO Range Check Enabled Reg 31 No boot
136	1C	Compatible device ID:PNPC11E
137	41	
138	D0	
139	C1	
140	1E	
141	31	Dependent Function Tag COM3 more IRQ
142	00	Most preferred
143	47	I/O Port Descriptor
144	01	16 bit address decode
145	E8	Min base low
146	03	Min base high (03E8)
147	E8	Max base low
148	03	Max base high (03E8)
149	08	Alignment for min base address
150	08	Range length
151	22	IRQ Format
152	A0	IRQ 5,7,10,11,15
153	8C	
154	31	Dependent Function Tag COM4 more IRQ
155	00	Most preferred
156	47	I/O Port Descriptor
157	01	16 bit address decode
158	E8	Min base low
159	02	Min base high (02E8)
160	E8	Max base low
161	02	Max base high (02E8)
162	08	Alignment for min base address
163	08	Range length
164	22	IRQ Format
165	A0	IRQ 5,7,10,11,15
166	8C	
167	31	Dependent Function Tag COM1
168	01	Acceptable
169	47	I/O Port Descriptor
170	01	16 bit address decode
171	F8	Min base low
172	03	Min base high (03F8)
173	F8	Max base low
174	03	Max base high (03F8)
175	08	Alignment for min base address
176	08	Range length
177	22	IRQ Format
178	B0	IRQ 4,5,7,10,11,15
179	8C	
180	31	Dependent Function Tag COM2
181	01	Acceptable
182	47	I/O Port Descriptor
183	01	16 bit address decode
184	F8	Min base low
185	02	Min base high (02F8)
186	F8	Max base low
187	02	Max base high (02F8)
188	08	Alignment for min base address
189	08	Range length
190	22	IRQ Format
191	A8	IRQ 3,5,7,10,11,15

192	8C	
193	31	Dependent Function Tag COM3
194	01	Acceptable
195	47	I/O Port Descriptor
196	01	16 bit address decode
197	E8	Min base low
198	03	Min base high (03E8)
199	E8	Max base low
200	03	Max base high (03E8)
201	08	Alignment for min base address
202	08	Range length
203	22	IRQ Format
204	B0	IRQ 4,5,7,10,11,15
205	8C	
206	31	Dependent Function Tag COM4
207	01	Acceptable
208	47	I/O Port Descriptor
209	01	16 bit address decode
210	E8	Min base low
211	02	Min base high (02E8)
212	E8	Max base low
213	02	Max base high (02E8)
214	08	Alignment for min base address
215	08	Range length
216	22	IRQ Format
217	A8	IRQ 3,5,7,10,11,15
218	8C	
219	38	End Dependent Function Tag
220	15	Logical Device ID
221	4A	Logical Device ID 4 CDROM
222	73	Dummy placer with no resource requested
223	50	RSS 5003
224	03	
225	00	No IO Range Check Enabled Reg 31 No boot
226	30	Dependent Function Tag No real resources
227	22	IRQ Format
228	00	
229	00	No IRQ
230	38	End Dependent Function Tag
231	15	Logical Device ID
232	4A	Logical Device ID 5 Game Port
233	73	
234	50	RSS 5004
235	04	
236	02	IO Range Check Enabled Reg 31 No boot
237	47	I/O Port Descriptor
238	01	16 bit address decode
239	01	Min base low
240	02	Min base high (0201)
241	01	Max base low
242	02	Max base high (0201)
243	01	Alignment for min base address
244	01	Range length
245	79	End Tag
246	93	Check Sum

This page is intentionally blank.

3. SOUND BLASTER PRO EMULATION INTERFACE

The WaveArtist hardware implements a Sound Blaster Pro compatible interface. The WaveArtist provides hardware emulation of the OPL2/OPL3 for Sound Blaster Pro and AdLib register compatibility.

3.1 Sound Blaster Pro Registers

The Sound Blaster Pro interface registers and bits are identified in Table 3-1.

Table 3-1. Sound Blaster Pro Interface Register Map

Address (Hex)	No. of Bits	R/W	Function	D7	D6	D5	D4	D3	D2	D1	D0
SB+0, SB+8, 388	8	W	FM Music 0 Address	MS0 A7	MS0 A6	MS0 A5	MS0 A4	MS0 A3	MS0 A2	MS0 A1	MS0 A0
SB+0, SB+8, 388	8	R	FM Music 0 Status	MS0 S7	MS0 S6	MS0 S5	MS0 S4	MS0 S3	MS0 S2	MS0 S1	MS0 S0
SB+1, SB+9, 389	8	W	FM Music 0 Data	MS0 D7	MS0 D7	MS0 D6	MS0 D4	MS0 D3	MS0 D2	MS0 D1	MS0 D0
SB+2, 38A	8	W	FM Music 1 Address	MS1 A7	MS1 A6	MS1 A5	MS1 A4	MS1 A3	MS1 A2	MS1 A1	MS1 A0
SB+2, 38A	8	R	FM Music 1 Status	MS1 S7	MS1 S6	MS1 S5	MS1 S4	MS1 S3	MS1 S2	MS1 S1	MS1 S0
SB+3, 38B	8	W	FM Music 1 Data	MS1 D7	MS1 D7	MS1 D6	MS1 D4	MS1 D3	MS1 D2	MS1 D1	MS1 D0
SB+4	8	W	Sound Blaster Mixer Address	—	—	MIX A5	—	MIX A3	MIX A2	MIX A1	—
SB+5	8	W/R	Sound Blaster Mixer Data	MIX D7	MIX D6	MIX D5	MIX D4	MIX D3	MIX D2	MIX D1	MIX D0
SB+6	8	W	Sound Blaster Reset	—	—	—	—	—	—	—	RST
SB+A	8	R	Sound Blaster Data	SB D7	SB D6	SB D5	SB D4	SB D3	SB D2	SB D1	SB D0
SB+C	8	W	Sound Blaster Command/Data	SB D7	SB D6	SB D5	SB D4	SB D3	SB D2	SB D1	SB D0
SB+C	8	R	Sound Blaster Write Buffer Status	WE	1	1	1	1	1	1	1
SB+E	8	R	Sound Blaster Read Buffer Status	RF	1	1	1	1	1	1	1

Notes:
SB = Sound Blaster Pro interface base register (typically 220 or 240 as assigned by PnP setup).

3.2 Sound Blaster Pro Registers Description

SB+0, +2, +8: FM Music Status Registers (R)/ Address Registers (W)

These status/address registers are used to read the OPL2/OPL3 and AdLib emulation status and to write the OPL2/OPL3 and AdLib emulation register address.

SB+1, +3, +9: FM Music Data Registers (R/W)

These data registers are used to write FM data to the OPL2/OPL3 and AdLib emulation.

SB+4: Mixer Address Register (R/W)

The Sound Blaster Mixer Registers are implemented in the WaveArtist for compatibility. All IO registers can be read/written via SB+4 and SB+5 registers.

The Mixer Address Register is used to program which index register to access. This register must be programmed first before writing data to the Mixer Data Register. The Sound Blaster Mixer Index Register mapping is shown in Table 3-2.

Table 3-2. Mixer Index Registers

Register	Bits 7 - 4				Bits 3 - 0			
00H	Data Reset							
04H	Left PCM Volume				Right PCM Volume			
0AH	X	X	X	X	X	Microphone Volume Mixing		
0CH	X	X	X	X	X	ADC Input Select		X
0EH	X	X	X	X	X	X	Stereo	X
22H	Left Master Volume				Right Master Volume			
26H	Left FM Volume				Right FM Volume			
28H	Left CD Volume				Right CD volume			
2EH	Left Line Volume				Right Line Volume			
X = Don't care.								

The Mixer Index Registers definitions are:

00H	Any value writes to this register will reset the mixer to its default value.
04H	This register may be used to change/report PCM volume for both the left/right channels. The default setting is 99H.
0AH	This register provides Microphone Mixing control.
0CH	This register is used to select ADC input sources.
0EH	This register is used to select PCM mode. Bit 1 = 1 selects STEREO. Bit 1 = 0 selects MONO.
22H	This register is used to set overall volume level.
26H	This register provides FM volume for left/right channels.
28H	This register provides CD volume for left/right channels.
2EH	This register provides Line-in volume for left/right channels.

SB+5 - SB Mixer Data Register (R/W)

The Mixer Data Register is used to write mixer data.

SB+6: SB Reset Register (W)

This register may be used to reset the WaveArtist by setting bit 0 = 1. It will remain reset until this bit is cleared (bit 0 = 0).

SB+A: SB Read Data Register (R)

The Data Register is used to check whether valid data is available when bit 7 is set. A read from SB+A register will clear the Data Available Flag.

SB+C: SB Command/Data Register (W)

The Command/Data Register is used to transfer commands/data to the Sound Blaster emulation.

SB+C: SB Write Buffer Status Register (R)

This register is used to determine when it's ready to write another command to the Command/Data register. Bit 7 is set whenever the PC writes a byte of data to the Data register and is cleared when the register is read by the WaveArtist.

SB+E: SB Read Buffer Status Register (R)

The Status Register is used to determine when the data is available to be read from port SB+A. Bit 7 is set whenever the WaveArtist writes a byte of data to the Data Register and is cleared when the Data Register is read by the PC.

This page is intentionally blank.

4. MPU-401 INTERFACE

The WaveArtist hardware implements an MPU-401 compatible interface which complies the MIDI 1.0 Detailed Specification.

4.1 MPU-401 Interface Registers

The MPU-401 registers which are used to pass MIDI transmitted/received data and commands between the PC and the MIDI device. The MPU-401 interface registers and bits are identified in Table 4-1.

Table 4-1. MIDI Interface Register Map

Address (Hex)	No. of Bits	R/W	Function	D7	D6	D5	D4	D3	D2	D1	D0
MB+0	8	R/W	MPU-401 Data Register	D7	D6	D5	D4	D3	D2	D1	D0
MB+1	8	W	MPU-401 Command Register	D7	D6	D5	D4	D3	D2	D1	D0
MB+1	8	R	MPU-401 Status Register	RF	WE	—	—	—	—	—	—
Notes: MB = MPU-401 interface base register (typically 300 or 330 as assigned by PnP setup).											

4.2 MPU-401 Registers Description

<u>MB+0: MPU-401 Data Register (R/W)</u>

This bi-directional data register is used to pass MIDI data between the PC and the WaveArtist. It is also used to read the acknowledge byte (**FEH**) from a previously sent command. There is a separate register for each direction.

<u>MB+1: MPU-401 Command Register (W)</u>

This command register is used for passing MIDI commands from the PC to the WaveArtist. Each write to this register must be read back with the appropriate acknowledge byte (**FEH)**.

The WaveArtist supports MPU-401 commands listed in Table 4-2.

Table 4-2. MPU-401 Commands

Command Number	Function	Response at Data Port
3FH	UART mode	FEH
ACH	Request version	15H (BCD for version 1.5)
ADH	Request revision	01H (for revision A)
FFH	Reset	FEH

The command definitions are:

3FH	This command sets the MPU-401 interface to UART mode. In UART mode, data written to the MPU-401 Data Register by the PC is routed out the MIDI_TX pin and data received on the MIDI_RX pin is routed to the MPU-401 Data Register for reading by the PC.
ACH	Request version.
ADH	Request revision.
FFH	This command causes the WaveArtist to respond with the acknowledge byte (FEH) and to reset bit 7 in the MIDI Status Register to a 0. The MIDI interrupt will be active until the MIDI Data Register is read.

MB+2: MPU-401 Status Register (R)

The Status Register reports the status of the Data Registers and the Command Register.

Bit 7	**Read Full (RF).** When cleared to a 0, the RF bit indicates that the WaveArtist has written a new data byte to the MPU-401 Data Register. A PC read of the MPU-401 Data Register will set the bit to a 1. The MIDI interrupt will be active when bit 7 is a 0.
Bit 6	**Write Empty (WE).** When cleared to a 0, the WE bit indicates that the PC can write a byte to the MPU-401 Command Register or MPU-401 Data Register to the MIDI command register. A PC write to either register will set the bit to a 1. The bit will be cleared to a 0 when the WaveArtist reads the corresponding register.
Bits 5-0	**Not Used.** Don’t care.

4.3 MIDI Supported Messages

Supported MIDI messages are described in Table 4-3, Table 4-4, and Table 4-5.

Table 4-3. MIDI Implementation Chart

Function		Description
Basic Channel	Default	1
	Changed	1-16
Velocity	Note ON	
	Note OFF	
After Touch	Channel	
Control Change	0,32	Bank Select
	1,33	Mod Wheel
	2,34	Breath
	6,38	Data entry
	7,39	Volume
	10,42	Pan
	11,43	Expression
	64	Sustain Pedal
	66	Sostenudo Pedal
	67	Soft Pedal
	96	Data Increment
	97	Data Decrement
	99	Non-Register Parameter LSB
	99	Non-Register Parameter MSB
	100	Register Parameter LSB
	101	Register Parameter MSB
	120	All Sound Off
	121	Reset All Controllers
	123	All Notes Off
Program Change	True #	0-127
System Exclusive		Note 1
Aux Messages	All Notes Off	

Notes:

1. Rockwell WaveArtist System Exclusive (SYSEX) sample download format.

F0	00 01 03	00 00	cc	pp pp pp	F7
	Rockwell ID	WaveArtist Model #	Command	Parameter	

Table 4-4. General MIDI Sound Set Program Number Assignment

Prog. No.	General MIDI Sound	Prog. No.	General MIDI Sound	Prog. No.	General MIDI Sound	Prog. No.	General MIDI Sound
0	Acoustic Grand Piano	32	Acoustic Bass	64	Soprano Sax	96	FX 1 (rain)
1	Bright Acoustic Piano	33	Electric Bass (finger)	65	Alto Sax	97	FX 2 (soundtrack)
2	Electric Grand Piano	34	Electric Bass (pick)	66	Tenor Sax	98	FX 3 (crystal)
3	Honky-tonk Piano	35	Fretless Bass	67	Baritone Sax	99	FX 4 (atmosphere)
4	Electric Piano 1	36	Slap Bass 1	68	Oboe	100	FX 5 (brightness)
5	Electric Piano 2	37	Slap Bass 2	69	English Horn	101	FX 6 (goblins)
6	Harpsichord	38	Synth Bass 1	70	Bassoon	102	FX 7 (echos)
7	Clavi	39	Synth Bass 2	71	Clarinet	103	FX 8 (sci-fi)
8	Celesta	40	Violin	72	Piccolo	104	Sitar
9	Glockenspiel	41	Viola	73	Flute	105	Banjo
10	Music Box	42	Cello	74	Recorder	106	Shamisen
11	Vibraphone	43	Contrabass	75	Pan Flute	107	Koto
12	Marimba	44	Tremolo Strings	76	Blown Bottle	108	Kalimba
13	Xylophone	45	Pizzicato Strings	77	Shakuhachi	109	Bag Pipe
14	Tubular Bells	46	Orchestral Harp	78	Whistle	110	Fiddle
15	Dulcimer	47	Timpani	79	Ocarina	111	Shanai
16	Drawbar Organ	48	String Ensemble 1	80	Lead 1 (square)	112	Tinkle Bell
17	Percussive Organ	49	String Ensemble 2	81	Lead 2 (sawtooth)	113	Agogo
18	Rock Organ	50	Synth Strings 1	82	Lead 3 (calliope)	114	Steel Drums
19	Church Organ	51	Synth Strings 2	83	Lead 4 (chiff)	115	Woodblock
20	Reed Organ	52	Choir Aahs	84	Lead 5 (charang)	116	Taiko Drum
21	Accordion	53	Voice Oohs	85	Lead 6 (voice)	117	Melodic Tom
22	Harmonica	54	Synth Voice	86	Lead 7 (fifths)	118	Synth Drum
23	Tango Accordion	55	Orchestra Hit	87	Lead 8 (bass + lead)	119	Reverse Cymbal
24	Acoustic Guitar (nylon)	56	Trumpet	88	Pad 1 (new age)	120	Guitar Fret Noise
25	Acoustic Guitar (steel)	57	Trombone	89	Pad 2 (warm)	121	Breath Noise
26	Electric Guitar (jazz)	58	Tuba	90	Pad 3 (polysynth)	122	Seashore
27	Electric Guitar (clean)	59	Muted Trumpet	91	Pad 4 (choir)	123	Bird Tweet
28	Electric Guitar (muted)	60	French Horn	92	Pad 5 (bowed)	124	Telephone Ring
29	Overdriven Guitar	61	Brass Section	93	Pad 6 (metallic)	125	Helicopter
30	Distortion Guitar	62	Synth Brass 1	94	Pad 7 (halo)	126	Applause
31	Guitar Harmonics	63	Synth Brass 2	95	Pad 8 (sweep)	127	Gunshot

Table 4-5. General MIDI Drum Kit Note Assignment

Note Number	Note	Sound	Note Number	Note	Sound
27	D#1	High Q	68	G#4	Low Agogo
28	E1	Slap	69	A4	Cabasa
29	F1	Scratch Push	70	A#4	Maracus
30	F#1	Scratch Pull	71	B4	Short Whistle
31	G1	Sticks	72	C5	Long Whistle
32	G#1	Square Click	73	C#5	Short Guiro
33	A1	Metronome Click	74	D5	Long Guiro
34	A#1	Metronome Bell	75	D#5	Claves
35	B1	Acoustic Bass Drum	76	E5	High Wood Block
36	C2	Bass Drum 1	77	F5	Low Wood Block
37	C#2	Side Stick	78	F#5	Mute Cuica
38	D2	Acoustic Snare	79	G5	Open Cuica
39	D#2	Hand Clap	80	G#5	Mute Triangle
40	E2	Electric Snare	81	A5	Open Triangle
41	F2	Low Floor Tom	82	A#5	Shaker
42	F#2	Closed High Hat	83	B5	Jingle Bell
43	G2	High Floor Tom	84	C6	Bell
44	G#2	Pedal High Hat	85	C#6	Castanets
45	A2	Low Tom	86	D6	Mute Surdo (Exc6)
46	A#2	Open High Hat	87	D#6	Open Surdo (Exc6)
47	B2	Low Mid Tom	88	E6	(No Sound)
48	C3	High Mid Tom	89	F6	(No Sound)
49	C#3	Crash Cymbal 1	90	F#6	(No Sound)
50	D3	High Tom	91	G6	(No Sound)
51	D#3	Ride Cymbal 1	92	G#6	(No Sound)
52	E3	Chinese Cymbal	93	A6	(No Sound)
53	F3	Ride Bell	94	A#6	(No Sound)
54	F#3	Tambourine	95	B6	(No Sound)
55	G3	Splash Cymbal	96	C7	(No Sound)
56	G#3	Cowbell	97	C#7	(No Sound)
57	A3	Crash Cymbal 2	98	D7	(No Sound)
58	A#3	Vibraslap	99	D#7	(No Sound)
59	B3	Ride Cymbal 2	100	E7	(No Sound)
60	C4	High Bongo	101	F7	(No Sound)
61	C#4	Low Bongo	102	F#7	(No Sound)
62	D4	Mute High Conga	103	G7	(No Sound)
63	D#4	Open High Conga	104	G#7	(No Sound)
64	E4	Low Conga	105	A7	(No Sound)
65	F4	High Timbale	106	A#7	(No Sound)
66	F#4	Low Timbale	107	B7	(No Sound)
67	G4	High Agogo	108	C8	(No Sound)

4.4 Rockwell System Exclusive (SYSEX) Sample Download Format.

The following are the SYSEX messages to turn on/off 3D in different WaveArtist configurations:

SYSEX to turn on 3D

RWA300 or (RWA010/RWA011 and RWA030)	F0 00 01 03 00 00 00 02 04 01 00 00 40 08 F7
RWA030 (Only)	F0 00 01 03 00 00 00 02 04 01 00 00 40 00 F7

SYSEX to turn off 3D

RWA300 or (RWA010/RWA011 and RWA030)	F0 00 01 03 00 00 00 02 04 01 00 00 00 08 F7
RWA030 (Only)	F0 00 01 03 00 00 00 02 04 01 00 00 00 00 F7

Note: Sending the wrong SYSEX message to the WaveArtist will cause the system to lock up.

5. GAME PORT INTERFACE

The WaveArtist hardware controls and indicates joystick interface using the Game Port Register.

5.1 Game Port Register

This Game Port Register (Table 5-1) is used to trigger (PC write) and indicate (PC read) the state of the four timer input pins (JAX, JAY, JBX, and JBY) which can support two joysticks or four paddles. This register also indicates the state of four button input pins (JA1, JA2, JB1, and JB2) which can support two buttons per joystick.

Table 5-1. Joystick Interface Register Map

Address (Hex)	No. of Bits	R/W	Function	D7	D6	D5	D4	D3	D2	D1	D0
JB	8	R/W	Game Port	JB2	JB1	JA2	JA1	JBY	JBX	JAY	JAX
Notes: JB = Game Port (Joystick) interface base register (typically 201 as assigned by PnP setup).											

5.2 Game Port Register Description

JB: Game Port Register

Bit 7 Joystick B Switch 2 (JB2)
This bit reflects the state of the Joystick B Switch 2 input to the WaveArtist.

Bit 6 Joystick B Switch 1 (JB1)
This bit reflects the state of the Joystick B Switch 1 input to the WaveArtist.

Bit 5 Joystick A Switch 2 (JA2)
This bit reflects the state of the Joystick A Switch 2 input to the WaveArtist.

Bit 4 Joystick A Switch 1 (JA1)
This bit reflects the state of the Joystick A Switch 1 input to the WaveArtist.

Bit 3 Joystick B Y Position Timer (JBY)
This bit reflects the state of the Joystick B Y position timer.

Bit 2 Joystick B Y Position Timer (JBX)
This bit reflects the state of the Joystick B X position timer.

Bit 1 Joystick A Y Position Timer(JAY)
This bit reflects the state of the Joystick A Y position timer.

Bit 0 Joystick A Y Position Timer (JAX)
This bit reflects the state of the Joystick A X position timer.

This page is intentionally blank.

6. WAVEARTIST COMMAND/STATUS INTERFACE

6.1 WaveArtist Registers

The Command/Status registers are used to control data I/O, mixer functions, and WaveArtist functions such as downloading wavetable samples. It is assumed that the base address of the register set is determined during bootstrap via PnP, or via user configuration options.

Some of these registers are 16-bits wide and others are 8-bit registers. The PC must use 16-bit instructions to read/write 16-bit registers and 8-bit instructions to access the single byte registers. The contents of the registers are identified in Table 6-1.

Table 6-1. WaveArtist Interface Register Map

Address (Hex)	No. of Bits	R/W	Function	D15 D7	D14 D6	D13 D5	D12 D4	D11 D3	D10 D2	D9 D1	D8 D0
WB+0, 1	16	R/W R/W	WaveArtist Command Low Byte WaveArtist Command High Byte	D7 D15	D6 D14	D5 D13	D4 D12	D3 D11	D2 D10	D1 D9	D0 D8
WB+2, 3	16	R/W R/W	WaveArtist Data Low Byte WaveArtist Data High Byte	D7 D15	D6 D14	D5 D13	D4 D12	D3 D11	D2 D10	D1 D9	D0 D8
WB+4	8	R/W	WaveArtist Control Register	CMD WEIE	CMD RFIE	DAT WEIE	DAT RFIE	RESET	FLG1	FLG0	IRQ ACK
WB+5	8	R	WaveArtist Status Register	CMD WE	CMD RF	DAT WE	DAT RF	IRQ REQ	FLG3	FLG2	ISA EWE
WB+6 to WB+B	8	R/W	WaveArtist Reserved	—	—	—	—	—	—	—	—
WB+C	8	R	WaveArtist Logical Device IRQ Status Register	—	—	—	CDROM IRQ	MODEM IRQ	MDP401 IRQ	SB IRQ	WA IRQ

Notes:

WB = WaveArtist interface base register (typically between 0250 and 03F0 as assigned by PnP setup).

6.2 WaveArtist Registers Description

WB+0: WaveArtist Command /Response Register (CMDR)

The PC can use this register to issue 16-bit I/O setup, mixer setup, and extended interface setup commands to the WaveArtist, and to receive 16-bit responses for a subset of these commands. The PC may enter a command into CMDR only if the CMD_WE bit is set in the STATR register. The PC may poll this bit according to its own scheme; this may or may not be in response to a PC interrupt generated by the WaveArtist as a result of the CMD_WEIE bit previously being set by the PC.

When the PC writes a value into CMDR, it must not attempt to write another command until the CMD_WE bit is set again, as the result of the WaveArtist reading the previous data. For commands which require the WaveArtist to send a response value through CMDR, the PC must wait on CMD_RF as an indicator that it may read CMDR to get the response. This may or may not be in response to the CMD_RFIE interrupt. The PC must not attempt to issue a command to the WaveArtist via CMDR during the period while it is waiting for a response from a previous command.

WB+2: WaveArtist Data Register (DATR)

The PC can use this register to send and receive 16-bit data values for parameters and/or programmed I/O. The PC may enter a value into DATR only if the DAT_WE bit is set in the STATR register. The PC may poll this bit according to its own scheme; this may or may not be in response to an interrupt request generated by the WaveArtist as a result of the DAT_WEIE bit previously being set by the PC.

When the PC writes a value into DATR, it must not attempt to write another command until the DAT_WE bit is set again, as the result of the WaveArtist reading the previous data. For data sent to the PC through DATR, the PC must wait on DAT_RF as an indicator that it may read DATR to get the response. This may or may not be in response to the DAT_RFIE interrupt.

WB+4: WaveArtist Control Register (CTRLR)

This register contains control fields which the WaveArtist reads to determine interrupt enable modes, and reset mode. This register may also be written so that bits may be accessed using a “read-modify-write” approach. This allows setting or clearing of individual bits without affecting the other bits within the control register. All bits are cleared by a PC reset. Each bit is described below:

Bit 7 Command Write Empty interrupt Enable (CMD_WEIE)

The interrupt is asserted immediately if the CMDR is empty and the CMD_WE is already set, when the PC sets CMD_WEIE. Otherwise the interrupt will assert every time the CMD_WE transitions from OFF to ON (CMDR not empty).

Bit 6 Command Read Full interrupt Enable (CMD_RFIE)

The interrupt will assert immediately If the CMDR is full and the CMD_RF is already set, when the PC sets CMD_RFIE. Otherwise the interrupt will happen every time the CMD_RFIE transitions from OFF to ON.

Bit 5 Data Write Empty interrupt Enable (DAT_WEIE)

The interrupt is asserted immediately if DATR is empty and the DAT_WE is already set, when the PC sets DAT_WEIE. Otherwise the interrupt will assert every time the DAT_WE transitions from OFF to ON.

Bit 4 Data Read Full interrupt Enable (DAT_RFIE)

The interrupt will assert immediately If the DATR is full and the DAT_RF is already set, when the PC sets DAT_RFIE. Otherwise the interrupt will happen every time the DAT_RFIE transitions from OFF to ON.

Bit 3 WaveArtist Reset (RESET)

The PC sets this bit whenever it wants to place the WaveArtist into its initial, known state. The PC must keep the RESET bit active for at least 7 ms or less than 300 ms. Upon clearing the RESET bit, the PC should wait for the CMD_RF bit in the STATR register to be set, then the PC should read the CMDR register, and should expect the value 0x55AA. If this value is not read, the reset failed.

Bit 2 DMA 1 Interrupt Enable (DMA1_IE)/(FLG1)

In DMA mode, the PC sets this bit and the DMA1 bit in the STATR register to enable the interrupt for the 16-bit DMA transfer. The PC should program the WaveArtist and the DMA controller for the proper size DMA buffers and interrupt count, as well as direction, before enabling this interrupt. If DMA has not been properly setup and initiated before enabling DMA1_IE, its behavior is undefined.

NOTE: *Disabling this bit does not halt DMA transfer, but merely inhibits the PC interrupt associated with reaching a preprogrammed count of samples for each transfer.*

Bit 1 DMA 0 Interrupt Enable (DMA0_IE) (FLG0)

In DMA mode, the PC sets this bit and the DMA0 bit in the STATR register to enable the interrupt for the 8-bit DMA transfer. The PC should program the WaveArtist and the DMA controller for the proper size DMA buffers and interrupt count, as well as direction, before enabling this interrupt. If DMA has not been properly setup and initiated before enabling DMA0_IE, its behavior is undefined.

NOTE: *This is the Sound Blaster Pro emulation's DMA channel. If necessary, this bit can be eliminated in favor of using the Sound Blaster control/status interface.*

Bit 0 Interrupt Request Acknowledge (IRQ_ACK)

The PC must set this bit and then clear it before exiting its interrupt service routine. Regardless of how many actual interrupts may have been serviced by the interrupt service routine (ISR), the IRQ_ACK bit must be toggled only once at the end of the ISR.

<u>WB+5: WaveArtist Status Register (STATR)</u>

The WaveArtist Status Register reports the status of various WaveArtist functions. The PC uses these bits to determine which events are pending. It may or may not poll this register in response to interrupts.

Bit 7 Command Write Empty (CMD_WE) (Default = 1)

This bit is set by the WaveArtist when no interrupts are pending. The PC may enter 8/16-bit command data into the CMDR register only if CMD_WE is set. This may, or may not be a result of an interrupt. After the data has been written, this bit will clear. The PC can not issue another command to the CMDR register again until the CMD_WE is set.

Bit 6 Command Read Full (CMD_RF) (Default = 0)

This bit is set by the WaveArtist when the interrupts are pending. The PC may read the CMDR register to obtain response data when the CMD_RF bit is set. After the data has been read, this bit will clear. The PC must not attempt to write another command until the response has been read and CMD_RF has cleared. This handshake is implemented to synchronize command/responses through CMDR.

Bit 5 Data Write Empty (DAT_WE) (Default = 1)

This bit is set by the WaveArtist when the Data Register is empty and is ready to accept new data. The PC may enter a 8/16-bit data into the DATR register only if the DAT_WE is set. This may, or may not be a result of an interrupt. After the data has been written, this bit will clear.

Bit 4 Data Read Full (DAT_RF) (Default = 0)

This bit is set by the WaveArtist whenever there are data available in the DATR register. The PC may read the data in the DATR when the DAT_RF bit set. This may or may not be a result of an interrupt. After the data has been read, this bit will clear.

NOTE: *The DAT_RF and the DAT_WE are totally asynchronous and may occur at different times and rates.*

Bit 3 Interrupt Request (IRQ_REQ) (Default = 0)

This bit is set by the WaveArtist when an interrupt has been asserted. The PC must service the appropriate interrupt and then set the IRQ_ACK bit in the CTLR register to indicate that the request has been serviced. Note that the PC does not require the IRQ_REQ bit.

Bit 2 DMA1 (FLG3) (Default = 0)

This bit is set by the WaveArtist whenever a preprogrammed number of 8-bit bytes has been transferred to or from PC memory on the 16-bit DMA channel. Normally, this number will be set to half of the DMA buffer size for this channel. If DMA1_IE is enabled, the WaveArtist will force the concurrent interrupt to the PC. The PC may or may not service this bit a result of the associated interrupt.

Bit 1 DMA0 (FLG2) (Default = 0)

This bit is set by the WaveArtist whenever a preprogrammed number of 8-bit bytes has been transferred to or from PC memory on the 8-bit DMA channel. Normally, this number will be set to half of the DMA buffer size for this channel. If DMA0_IE is enabled, the WaveArtist will force the concurrent interrupt to the PC. The PC may or may not service this bit a result of the associated interrupt.

Note: This is the Sound Blaster Pro emulation’s DMA channel. If necessary, this bit can be eliminated in favor of using the Sound Blaster control/status interface.

Bit 0 Reserved (Default = 0)

This bit is not used.

<u>WB+C: WaveArtist Expansion Control Register 1 (IRQSTAT)</u>

These active high IRQ status bits are set and reset automatically by the WaveArtist to reflect the current state of each logical device IRQ line to the ISA bus.

Bit 7 Reserved (Default = 0)

This bit is not used.

Bit 6 Reserved (Default = 0)

This bit is not used.

Bit 5 Reserved (Default = 0)

This bit is not used.

Bit 4 CD-ROM Logical Device IRQ (Default = 0)

This bit reflects the CD-ROM Logical Device IRQ output state (1 = IRQ asserted, 0 = IRQ deasserted).

Bit 3 Modem Logical Device IRQ (Default = 0)

This bit reflects the Modem Logical Device IRQ output state (1 = IRQ asserted, 0 = IRQ deasserted).

Bit 2 MPU-401 Logical Device IRQ (Default = 0)

This bit reflects the MPU-401 Logical Device IRQ output state (1 = IRQ asserted, 0 = IRQ deasserted).

Bit 1 Sound Blaster Pro Logical Device IRQ (Default = 0)

This bit reflects the Sound Blaster Pro Logical Device IRQ output state (1 = IRQ asserted, 0 = IRQ deasserted).

Bit 0 WaveArtist Logical Device IRQ (Default = 0)

This bit reflects the WaveArtist Logical Device IRQ output state (1 = IRQ asserted, 0 = IRQ deasserted).

6.3 WaveArtist Commands

The WaveArtist commands are listed in Table 6-2.

Table 6-2. WaveArtist Commands

Category and Function	Command No.	Description
Query and Diagnostic Commands	00h	System ID and Revision Request
	01h	Downloadable DRAM Size Request
Wave I/O Commands	n09h	WaveArtist ROM, RAM, and Timer Tests
	10h	Set Sample Width/Format for Data Path to PC
	11h	Set Number of Channels for Data Path to PC
	12h	Set Sample Speed Time Constant for Data Path to PC
	13h	Assign ADC Data Transfer to PC Path
	14h	Set Sample Count for Block Data Transfer to PC
	15h	Enable ADC Data Transfer to PC
	16h	Pause ADC Data Transfer to PC
	17h	Stop ADC Data Transfer to PC
	18h	Resume ADC Data Transfer to PC
	20h	Set Sample Width/Format for Data Path from PC
	21h	Set Number of Channels for Data Path from PC
	22h	Set Sample Speed Time Constant for Data Path from PC
	23h	Assign DAC Data Transfer from PC Path
	24h	Set Sample Count for Block Data Transfer from PC
	25h	Enable DAC Data Transfer from PC
	26h	Pause DAC Data Transfer from PC
	27h	Stop DAC Data Transfer from PC
	28h	Resume DAC Data Transfer from PC
	nn42h	Clear DMA Status Indicator(s)
Mixer and Volume Control Commands	nn30h	Read Mixer Control Word or Volume Setting
	nn31h	Write New Volume Setting Pair
	32h	Write New Mixer Control Setting Pair
	33h	Reset Mixer Settings
	nn34h	Select Mono Record Source
Extended Interface Commands	nn50h	Select MIDI Output Path
	nn62h	Initiate DRAM Download
	63h	Terminate DRAM Download

6.4 WaveArtist Command Set Definitions

Described below are WaveArtist commands which can be used to determine system configuration and capabilities.

00H - System ID and Revision Request

This command word may be used to request the WaveArtist product ID and revision code. The PC must write the command first to the CMDR register followed by reading Word 1 then Word 2.

Word 1: The low byte = Product ID. The high byte = Code Revision.

Word 2: Wavetable preset, if not the response is 0000H.

01H - Downloadable DRAM Size Request

This command word may be used to indicate the size of the DRAM. The PC must write the command first to the CMDR register followed by Word 1 or Word 2. The returned first word in the CMDR register is reserved and the contents are “don’t care”.

Word 1: Reserved.

Word 2: DRAM size

0080H = 4Mx16 DRAM
00A0H = 1Mx16 DRAM
00A8H = 256Kx16 DRAM

N09H - WaveArtist ROM, RAM, and Timer Tests

This command may be used to run diagnostic tests on WaveArtist internal ROM, RAM, and Timer. N is the parameter indicating which diagnostic test to run. The test results are written to the CMDR Register upon test completion.

N = 0; Run ROM, RAM, and Timer tests (Expected result = 31H, 80H, and 8DH)
N = 1; Run ROM checksum test only (Expected result = 31H)
N = 2; Run RAM checksum test only (Expected result = 80H)
N = 3; Run Timer test only (Expected result = 8DH)

A sample ROM Checksum Test Program is available.

10H - Set Sample Width/Format for Data Path to PC

This command may be used to set the ADC format to 8-bit or 16-bit samples. The PC must write the command word first in the CMDR Register followed by the parameter word described below. In each case, the CMD_WE bit must be set first before the PC writes. The PC must then read the response value from the CMDR register, after the CMD_RF bit is set, to verify whether the selected sample format has been accepted by the WaveArtist. The WaveArtist response will occur within 5µs.

Parameter Word:

0 = 8-bit signed PCM (Default).
1 = 16-bit signed PCM.
2 = 8-bit unsigned PCM.

Response Value:

0 = Unacceptable selection.
1 = OK.

Notes:

1. Acceptance of this command overrides the previous ADC format/width settings.
2. If used while the ADC path is fully active, the results are unpredictable.

11H - Set Number of Channels for Data Path to PC

This command may be used to select MONO or STEREO for the ADC data. The PC must write the command word first in the CMDR register followed by the parameter word described below. In each case, the CMD_WE bit must be set first before the PC writes. The PC must then read the response value from the CMDR register, after the CMD_RF bit is set, to verify whether the selected number of channels has been accepted by the WaveArtist. The WaveArtist response will occur within 5µs.

Parameter Word:

1 = MONO (Default).
2 = STEREO.

Response Value:

0 = Unacceptable selection.
1 = OK.

Notes:

1. Acceptance of this command overrides the previous ADC STEREO or MONO setting.
2. If used while the ADC path to the PC is fully active, the results are unpredictable

12H - Set Sample Speed Time Constant for Data Path to PC

This command may be used to set the ADC sample rate for the data stream coming to the PC. The PC must write the command word first in the CMDR register followed by the parameter word. The sample rate parameter is entered as described in the formula below. In each case, the CMD_WE bit must be set first before the PC writes. The PC must then read a response value from the CMDR register, after the CMD_RF bit is set, to verify whether the selected sample rate has been accepted by the WaveArtist. The WaveArtist response will occur within 5µs.

Parameter Formula:

K = (65536 * Sample Rate)/44100. The value of K is a 32-bit word, but only the lower 16-bit result is required.

For the nominal speeds supported, the formula works out to the hex values in the following table:

Speed (Hz)	K
8000	2E71
11,025	4000
22,050	8000
44,100	0000

Response Value:

0 = Unacceptable selection.

1 = OK.

Note: *Acceptance of this command overrides the previous ADC sample speed setting.*

13H - Assign ADC Data Transfer to PC Path

This command may be used to transfer the ADC data to the PC. The data can be 13H transferred by using the DMA channels, or the programmed I/O mode via the DATR register. The PC must write the command word first in the CMDR register followed immediately by the parameter word. In each case, the CMD_WE bit must be set first before the PC writes. The PC must then read a response value from the CMDR register, after the CMD_RF bit is set, to verify whether the selected path has been accepted by the WaveArtist. The WaveArtist response will occur within 5µs.

In the programmed I/O mode, 8 or 16-bit ADC data will be sent through the DATR register. Each byte or word transferred will be read by the PC if and only if the DAT_RF bit has been set. Notice that the sample count (see command 14H) associated with the transfer can still be used to generate interrupts less often than on every sample.

Parameter Word:

0 = Use 8-bit DMA Channel (Sound Blaster Pro DMA Channel)

1 = Use 16-bit DMA Channel (Default)

Response Value:

0 = Unacceptable Rate Selection.

1 = OK.

Note

If the command is issued while ADC data is being routed to the PC, then the results are unpredictable. PC should terminate the ADC path to the PC before issuing the command. If the selection is already in use by the DAC path the selection is rejected.

14H - Set Sample Count for Block Data Transfer to PC

This command may be used to set the “between interrupts” sample count for block of ADC data transferring to the PC. The ***sample*** count is an 8-bit byte or 16-bit word depending upon the currently selected sample width value; this number is between 1 and 64K.

Sample Definitions

8-bit Mono Mode

Each sample is one byte received from the single ADC channel for each sample period. Note that if the 16-bit DMA channel is in use, each DMA cycle can accommodate two 8-bit sample transfers to the PC and the WaveArtist must decrement its internal count twice per DMA cycle. If the 8-bit DMA channel is in use, each sample transfer to the PC occupies one DMA cycle.

16-bit Mono Mode

Each sample is a 16-bit word for each sample period. Note that if the 8-bit DMA channel is used, each 16-bit sample datum requires 2 DMA cycles; the WaveArtist’s internal count however is decrement only once for each 2 DMA cycles. If the 16-bit DMA channel is used in this mode, each sample item requires one DMA cycle, for which the WaveArtist’s internal count is decrement once.

8-bit Stereo Mode

Two single byte samples are sent, one from each channel, for the sample period. However, the count provided to the WaveArtist is still a total count of 8-bit samples per block, and thus is decrement by the WaveArtist twice per sample period. If the 8-bit DMA channel is in use, the WaveArtist alternates the left and right channel samples for each DMA cycle transfer, decrementing its internal count once per DMA cycle. If the 16-bit DMA channel is used, the WaveArtist packs both the left and right sample into each 16-bit DMA cycle transfer, decrementing its internal sample count twice per DMA cycle.

16-bit Stereo Mode

Two single word samples are sent, one from each channel, for the sample period. The count provided to the WaveArtist is a total count of 16-bit samples per block, and thus is decremented by the WaveArtist twice per sample period, identically to 8-bit STEREO mode. If the 8-bit DMA channel is in use however, the WaveArtist must use 4 DMA cycles each sample period, sending the low and high bytes of first the left channel, and then the low and high bytes of the right channel. The WaveArtist’s internal count will be decremented for each two DMA cycle transfers, or twice per sample period as mentioned above. If the 16-bit DMA channel is used, the WaveArtist sends first the left, and then the right channel 16-bit samples in 2 DMA cycle transfers, decrementing its count on each transfer.

The PC will most likely determine the count, sent to the WaveArtist as a function of the sample size and speed. Therefore, it is likely that the count will be sent every time the format and/or speed is changed.

Block Transfer Complete Notification

The WaveArtist must notify the PC when a block transfer is complete. It applies the count parameter to the currently selected data path as follows:

Notifying the PC at the end of a sample block utilizing the 8-bit DMA Channel

If the 8-bit DMA (Sound Blaster) channel is selected, the WaveArtist asserts DMA0 in the STATR register when the number of samples requested have been fully transferred to the PC via the 8-bit DMA channel. If and only if the DMA0_IE bit has been set by the PC, the WaveArtist additionally asserts the PC interrupt for this condition.

Notifying the PC at the end of a sample block utilizing the 16-bit DMA Channel

If the 16-bit DMA channel is selected, the WaveArtist asserts DMA1 in the STATR register when the number of samples requested have been transferred to the PC via the 16-bit DMA channel. If and only if the DMA1_IE bit has been set by the PC, the WaveArtist additionally asserts the PC interrupt for this condition.

Notifying the PC at the end of a sample block utilizing Programmed I/O

If the PIO mode is selected for ADC, the WaveArtist asserts whichever DMA status bit in STATR is currently unused by the DAC data path coming from the PC. If the 8-bit DMA channel is being used for DAC data, then the WaveArtist uses DMA1 in STATR, and the associated interrupt if DMA1_IE is selected to provide infrequent PIO interrupts. If the 16-bit DMA Channel is being used for DAC data, then the WaveArtist uses DMA0 in a likewise manner. These bits are allocated by the

WaveArtist similarly if the DAC path is setup for PIO. If the DAC path is currently unused, then the PIO ADC path defaults to using the DMA1 bit and the associated interrupt if enabled.

This mechanism is designed to allow the PC to perform programmed I/O without requiring a PC interrupt on every sample datum. When the PC receives the interrupt for the expired sample count, it will then get the data, 2 bytes (16-bits) at a time via the DATR and DAT_RF mechanism. If the sample width is 8-bits, this means that the WaveArtist must always pack 2 samples into DATR per programmed I/O transfer (as if the 16-bit DMA channel were in use).

This mechanism implies that the interrupt, for programmed I/O will be generated only after the WaveArtist has collected the full number of samples specified by the count in its internal memory, creating a software FIFO or “speed buffer” -- which implies that the WaveArtist must buffer the data -- which further implies that for programmed I/O, the sample count will likely be quite small. A better mechanism for PIO might employ WaveArtist FIFOs between the PC and the WaveArtist.

It is assumed that the specified count will be used by the WaveArtist in an auto-reinitialize manner. When the WaveArtist fulfills a block transfer, it automatically reinitializes its count and is ready to begin the next block. The PC can, at any time issue the PAUSE or TERMINATE commands, to stop the data flow. Since the WaveArtist can receive these commands at any time, it is currently considered unnecessary for it to support a “single block” transfer request (though this could be added if desired).

When the 14h is entered into CMDR, the WaveArtist then assumes that the desired sample item count will be entered immediately into CMDR by the PC. The PC writes only when CMD_WE is enabled. The PC must accept a response from the WaveArtist for this command via CMDR, when CMD_RF is set. The value read from CMDR indicates whether the selected count has been accepted by the WaveArtist WaveArtist/firmware. The WaveArtist response will occur within 5µs.

15H - Enable ADC Data Transfer to PC

This command may be used to enable ADC data transfer in the current data format and sample rate. The data can be transferred by using DMA or Programmed IO mode. Status bits and interrupts (if enabled) are set according to the mode.

16H - Pause ADC Data Transfer to PC

This command may be used to pause ADC data transfer to the PC. The WaveArtist stops sending data regardless of the format, or selected data path. When the RESUME command is given, the WaveArtist resumes sending the ADC data to the PC at the same position within the DMA block transfer; the sample item count is not reset by the PAUSE command. This command should be used if the current DMA buffer is to be completed after issuing the RESUME command.

17H - Stop ADC Data Transfer to PC

This command may be used to stop ADC data transfer to the PC. The WaveArtist stops sending data regardless of the format, or selected data path. When and if the ENABLE command is given, the WaveArtist begins sending the ADC data to the PC after resetting its internal position relative to the sample item count. This command should be used when the current DMA buffer is to be released.

18H - Resume ADC Data Transfer to PC

This command may be used to resume ADC data transfer in the current data format and sample speed through the selected DMA or Programmed IO data path to the PC. Status bits and interrupts (if enabled) are set according to the mode.

20H - Set Sample Width/Format for Data Path from PC

This command may be used to set the DAC format to 8-bit or 16-bit sample data coming from the PC. The PC must write the command word first in the CMDR register, followed by the parameter word. In each case, the CMD_WE bit must be set first before the PC writes. The PC must then read the response from the CMDR register, after the CMD_RF bit is set, to see whether the selected sample format has been accepted by the WaveArtist. The WaveArtist response will occur within 5µs.

Parameter Word:

0 = 8-bit PCM (Default).

1 = 16-bit PCM.

Response Value:

0 = Unacceptable selection.

1 = OK.

Notes:

1. Acceptance of this command overrides the previous DAC format/width settings.
2. If used while the DAC path is fully active, the results are unpredictable.

21H - Set Number of Channels for Data Path from PC

This command may be used to select MONO or STEREO for the DAC data path. The PC must write the command word first in the CMDR register followed by the parameter word. In each case, the CMD_WE bit must be set first before the PC writes. The PC must then read the response value from the CMDR register, after the CMD_RF bit is set, to verify whether the selected number of channels has been accepted by the WaveArtist. The WaveArtist response will occur within 5µs.

Parameter Word:

1 = MONO (Default).

2 = STEREO.

Response Value:

0 = Unacceptable selection.

1 = OK.

Notes:

1. Acceptance of this command overrides the previous ADC STEREO or MONO setting.
2. If used while the DAC path to the PC is fully active, the results are unpredictable

22H - Set Sample Speed Time Constant for Data Path from PC

This command may be used to set the DAC sample rate for the data stream coming to the PC. The PC must write the command word first in the CMDR register followed by the parameter word. The sample rate parameter is entered as described in the formula below. In each case, the CMD_WE bit must be set first before the PC writes. The PC must then read a response value from the CMDR register, after the CMD_RF bit is set, to verify whether the selected sample rate has been accepted by the WaveArtist. The WaveArtist response will occur within 5µs.

Parameter Formula:

K = (65536 * Sample Rate)/44100. The value of K is a 32-bit word, but only the lower 16-bit result is required.

For the nominal speeds supported, the formula works out to the hex values in the following table:

Speed (Hz)	K
8000	2E71
11,025	4000
22,050	8000
44,100	0000

Response Value:

0 = Unacceptable selection.
1 = OK.

Note:

Acceptance of this command overrides the previous ADC sample speed setting.

23H - Assign DAC Data Transfer from PC Path

This command may be used to transfer the DAC data to the PC. The data can be transferred by using the DMA channels, or the programmed I/O mode via the DATR register. The PC must write the command word first in the CMDR register followed immediately by the parameter word. In each case, the CMD_WE bit must be set first before the PC writes. The PC must then read a response value from the CMDR register, after the CMD_RF bit is set, to verify whether the selected path has been accepted by the WaveArtist. The WaveArtist response will occur within 5µs.

In the programmed I/O mode, 8 or 16-bit ADC data will be sent through the DATR register. Each byte or word transferred will be read if and only if the DAT_WE bit has been set. Notice that the sample count (see command 14H) associated with the transfer can still be used to generate interrupts less often than on every sample

Parameter Word:

0 = Use 8-bit DMA Channel (Sound Blaster Pro DMA Channel)
1 = Use 16-bit DMA Channel (Default)

Response Value:

0 = Unacceptable Rate Selection
1 = OK

Note:

If the command is issued while DAC data is being routed to the PC, then the results are unpredictable. PC should terminate the DAC path to the PC before issuing the command. If the selection is already in use by the ADC path the selection is rejected.

24H - Set Sample Count for Block Data Transfer from PC

This command may be used to set the "between interrupts" count used for the data stream coming from the PC. This is a number between 1 and 64K and refers to the number of ***samples*** to be transferred from the PC to the WaveArtist. For the purposes of this command, a ***sample*** is an 8-bit byte or 16-bit word depending upon the currently selected sample width value. Note, other sample widths may be possible in the future.

Parameter Word:

Number (1-64K) of samples to transfer from the PC before asserting the status bit and interrupt (if enabled) associated with fulfilling the count. See description.

Response Value:

1 = OK

0 = Unacceptable count

Sample Definition

The meaning of the sample count parameter is detailed below:

8-bit Mono Mode

Each sample is one byte sent to the single DAC channel. A byte is sent each sample period, thus the count is a byte count. Note that if the 16-bit DMA channel is in use, each DMA cycle can accommodate two 8-bit sample transfers from the PC and the WaveArtist must decrement its internal count twice per DMA cycle. If the 8-bit DMA channel is in use, each sample transfer from the PC occupies one DMA cycle.

16-bit Mono Mode

Each sample is a 16-bit word. A 16-bit sample is sent to the single DAC channel each sample period, thus the count is a word count. Note that if the 8-bit DMA channel is used, each 16-bit sample datum requires 2 DMA cycles; the WaveArtist's internal count however is decremented only once for the 2 DMA cycles. If the 16-bit DMA channel is used in this mode, each sample transfer from the PC requires one DMA cycle, for which the WaveArtist's internal count is decremented once.

8-bit Stereo Mode

Two single byte samples are sent, one on each DAC channel, for the sample period. However, the count provided to the WaveArtist is still a total count of 8-bit samples per block, and thus is decremented by the WaveArtist twice per sample period. If the 8-bit DMA channel is in use, the PC alternates the left and right channel samples for each DMA cycle transfer to the WaveArtist, which will decrement its internal count once per DMA cycle. If the 16-bit DMA channel is used, the PC packs both the left and right sample into each 16-bit DMA cycle transfer to the WaveArtist, which will decrement its internal sample count twice per DMA cycle.

16-bit Stereo Mode

Two single word samples are sent, one for each DAC channel, for the sample period. The count provided to the WaveArtist is a total count of 16-bit samples per block, and thus is decremented by the WaveArtist twice per sample period, identically to 8-bit STEREO mode. If the 8-bit DMA channel is in use however, the PC must use 4 DMA cycles each sample period, sending the low and high bytes of first the left channel, and then the low and high bytes of the right channel to the WaveArtist. The WaveArtist's internal count will be decremented for each two DMA cycle transfers from the PC, or twice per sample period as mentioned above. If the 16-bit DMA channel is used, the WaveArtist receives first the left, and then the right channel 16-bit samples in 2 DMA cycle transfers, decrementing its count on each transfer from the PC.

The PC will most likely determine the count, sent to the WaveArtist as a function of the sample size and speed. Therefore, it is likely that the count will be sent every time the format and/or speed is changed.

Transfer Complete Notification

The WaveArtist must notify the PC when a block transfer is complete. It applies the count parameter to the currently selected data path as follows:

Notifying the PC at the end of a sample block utilizing the 8-bit DMA Channel

If the 8-bit DMA (Sound Blaster) channel is selected, the WaveArtist asserts DMA0 in the STATR register when the number of samples requested have been fully transferred from the PC via the 8-bit DMA channel. If and only if the DMA0_IE bit has been set by the PC, the WaveArtist additionally asserts the PC interrupt for this condition.

Notifying the PC at the end of a sample block utilizing the 16-bit DMA Channel

If the 16-bit DMA channel is selected, the WaveArtist asserts DMA1 in the STATR register when the number of samples requested have been transferred from the PC via the 16-bit DMA channel. If and only if the DMA1_IE bit has been set by the PC, the WaveArtist additionally asserts the PC interrupt for this condition.

Notifying the PC at the end of a sample block utilizing Programmed I/O

If the PIO mode is selected for DAC, the WaveArtist asserts which ever DMA status bit in STATR is currently unused by the ADC data path going to the PC. If the 8-bit DMA channel is being used for ADC data, then the WaveArtist uses DMA1 in STATR, and the associated interrupt if DMA1_IE is selected, to provide infrequent PIO interrupts for DAC. If the 16-bit DMA Channel is being used for ADC data, then the WaveArtist uses DMA0 in a likewise manner. These bits are allocated by the WaveArtist similarly if the ADC path is setup for PIO. If the ADC path is currently unused, then the PIO DAC path defaults to using the DMA1 bit and the associated interrupt if enabled.

This mechanism is designed to allow the PC to perform programmed I/O without requiring a PC interrupt on every sample datum. When the PC receives the interrupt for the expired sample count, it will then send the data, 2 bytes (16-bits) at a time via the DATR and DAT_WE mechanism. If the sample width is 8-bits, this means that the PC must always pack 2 samples into DATR per programmed I/O transfer (as if the 16-bit DMA channel were in use), and the WaveArtist must read the 2 samples from DATR.

This mechanism implies that the interrupt, for programmed I/O will be generated only when the WaveArtist has collected enough buffer space to contain the full number of samples specified by the count in its internal memory, creating a software FIFO or “speed buffer” -- which implies that the WaveArtist must buffer the transmitted data -- which further implies that for programmed I/O, the sample count will likely be quite small. A better mechanism for PIO might employ WaveArtist FIFOs between the PC and the WaveArtist.

It is assumed that the specified count will be used by the WaveArtist in an auto-reinitialize manner. When the WaveArtist fulfills a block transfer, it automatically reinitializes its count and is ready to begin the next block. The PC can, at any time issue the PAUSE or TERMINATE commands, to stop the data flow. Since the WaveArtist can receive these commands at any time, it is currently considered unnecessary for it to support a “single block” transfer request (though this could be added if desired).

When the 24h is entered into CMDR, the WaveArtist then assumes that the desired sample item count will be entered immediately into CMDR by the PC. The PC writes only when CMD_WE is enabled. The PC must accept a response from the WaveArtist for this command via CMDR, when CMD_RF is set. The value read from CMDR indicates whether the selected count has been accepted by the WaveArtist WaveArtist/firmware. The WaveArtist response will occur within 5µs.

25H - Enable DAC Data Transfer from PC

This ENABLE command may be used to begin DAC data transfer from the PC. The PC must select the data format and sample speed through the selected DMA or Programmed IO mode. Status bits and interrupts (if enabled) are set according to the mode. This command instructs the WaveArtist NOT to issue a “dummy” read to force an initial DMA request.

26H - Pause DAC Data Transfer from PC

This PAUSE command may be used to pause the DAC data transfer from the PC. The WaveArtist will stop sending the data regardless of the data format. When the PC issues RESUME command, the WaveArtist will send the DAC data at the same position within the DMA block transfer. The PC uses the RESUME command to complete the data transfer.

NOTE: The sample item count is not reset by the PAUSE command.

27H - Stop DAC Data Transfer from PC

This STOP command may be used to stop the DAC data transfer from the PC. The WaveArtist will stop sending the data regardless of the data format. When the PC issues ENABLE command, the WaveArtist will begin the DAC data after resetting its internal position relative to the sample item count. The PC uses the ENABLE command to release the current DMA buffer.

28H - Resume DAC Data Transfer from PC

This RESUME command may be used to resume DAC data transfer to the PC in the current data format and sample speed through the selected DMA or Programmed IO mode. Status bits and interrupts (if enabled) are set according to the selected mode. This command assumes that a valid DMA sample is in the WaveArtist DMA read register, and this sample is used rather than discarded.

NN42H - Clear DMA Status Indicator(s)

This command may be used to acknowledge that pending DMA0 and/or DMA1 status bits in the STATR register have been dealt with and will be cleared by the WaveArtist. The PC sends this command, after the IRQ_ACK bit is toggled, for a pending DMA interrupt. The ability to acknowledge both the 8-bit and 16-bit channel interrupts simultaneously allows the PC to service both interrupts during a single interrupt service routine invocation. This means that the PC may be in the midst of servicing one interrupt, say DMA0, and then if the DMA1 bit goes ON, the PC can loop back and service this one as well, without exiting its ISR. Upon completion, the PC ISR can acknowledge both interrupts, thus preventing the WaveArtist from raising the IRQ again. This command does not write a separate parameter into the CMDR register, but the selected parameter is stored in the upper byte of the command request as shown:

NN =	the upper byte of the 16-bit value written to the CMDR register. In this case, the upper byte is used to hold a parameter for the command which is an image of the DMA0 and DMA1 status bits in the STATR register identifying which DMA channel(s) are acknowledged.

6.5 WaveArtist Mixer and Volume Control Commands

The WaveArtist Mixer and Volume Control Commands are expanded to include enhanced features. The features include controlling wavetable functionality as well as downloading samples.

NN30H - Read Mixer Control Word or Volume Setting

This command may be used to determine the current Mixer Control Word or Volume Setting. Where NN, the upper byte of the 16-bit value written to the CMDR register, is used to hold a parameter for indexing mixer or volume setting.

The valid values of NN30H are defined as follows:

0030H -0930H	Return one Mixer control word in the range from 1 through 10. The upper byte is the 4-bit control word address minus one. Example: 0730H means return mixer control word 8.
0A30H	Return left PCM channel volume word.
0B30H	Return right PCM channel volume word.
0C30H	Return left FM channel volume word.
0D30H	Return right FM channel volume word.
0E30H	Return left wavetable channel volume word.
0F30H	Return right wavetable channel volume word.
1030H	Return left PCM expansion channel output volume word.
1130H	Return right PCM expansion channel output volume word.
1230H	Return left FM expansion channel output volume word.
1330H	Return right FM expansion channel output volume word.

The WaveArtist returns Current Settings of the Specified Mixer Control, expansion volume, PCM, FM, or wavetable Volume word as specified in the request. The formats of the ten Mixer control words are described in the Control Bit Definition table below. The format of the Volume words is a 15-bit unsigned scalar (0 - 7FFFH).

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
ADR				0	Control Word Specific Settings										

Table 6-3 details the WaveArtist Mixer control words.

Table 6-3. Mixer Control Bit Definitions

4 bit Address	11 Bit Contents	Function	Logic Level
1	Left_Line_Mixer_Gain [10:6]	+12 to -33.0 dB and mute	11111 = +12 dB 0 = mute
	Left_Aux1_Mixer_Gain [5:1]	+12 to -33.0 dB and mute	11111 = +12 dB 0 = mute
	Left_Line_Mute [0]	Mute left line output	0 = mute
2	Left_Aux2_Mixer_Gain [10:6]	+12 to -33.0 dB and mute	11111 = +12 dB 0 = mute
	Right_2_Left_Mic_Gain [5:1]	+12 to -33.0 dB and mute	11111 = +12 dB 0 = mute
	Mono_Mute [0]	Mute mono line output	0 = mute
3	Left_Mic_Mixer_Gain [10:6]	+12 to -33.0 dB and mute	11111 = +12 dB 0 = mute
	Left_Mic_Gain [5:4]	0, 10 or +20 dB	11 = 20 dB 0 = mute
	Left_Mixer_Gain [3:1]	3 dB step Mixer attenuation	111 = 0 dB 0 = mute
	Dither_Disable [0]	ADC dither disable	0 = enable
4	Left_Mixer_Input_Select [10:4]	Mux control to mixer	see Mixer Input Control
	Left_Rx_Filter_Gain [3]	0 or +12 dB gain	1 = 12 dB 0 = 0 dB
	Left_ADC_Gain [2:0]	0 to 10.5 dB in 1.5 dB steps	111 = 10.5 dB
5	Right_Line_Mixer_Gain [10:6]	+12 to -33.0 dB and mute	11111 = +12 dB 0 = mute
	Right_Aux1_Mixer__Gain [5:1]	+12 to -33.0 dB and mute	11111 = +12 dB 0 = mute
	Right_Line_Mute [0]	Mute right line output	0 = mute
6	Right_Aux2_Mixer_Gain [10:6]	+12 to -33.0 dB and mute	11111 = +12 dB 0 = mute
	Left_2_Right_Mic_Gain [5:1]	+12 to -33.0 dB and mute	11111 = +12 dB 0 = mute
	Test_1_Bitout [0]	Send ADC 1 bit out to pad	1 = enable
7	Right_Mic_Mixer_Gain [10:6]	+12 to -33.0 dB and mute	11111 = +12 dB 0 = mute
	Right_Mic_Gain [5:4]	0, 10 or +20 dB	11 = 20 dB 0 = mute
	Right_Mixer_Gain [3:1]	3 dB step Mixer attenuation	111 = 0 dB 0 = mute
	Right_Mixer_Bypass [0]	Bypass the mixer on DAC	1 = bypass
8	Right_Mixer_Select [10:4]	Mux control to mixer	see Mixer Input Control
	Right_Rx_Filter_Gain [3]	0 or +12 dB gain setting	1 = +12 dB 0 = 0 dB
	Right_ADC_Gain [2:0]	0 to 10.5 dB in 1.5 dB steps	111 = 10.5 dB
9	Mono_Mixer_Gain [10:6]	+12 to -33.0 dB and mute	11111 = +12 dB 0 = mute
	Right_ADC_Mux_Select [5:3]	Select 1 of 5 inputs to ADC	000 = AGND 011 = AUX2
	Left_ADC_Mux_Select [2:0]	Select 1 of 5 inputs to ADC	001 = MIXER 100 = AUX1
			010 = LINE 101 = MIC
10	Left_LB_ADC_2_DAC [10]	Loop back ADC to DAC	1 = active
	Left_Line_Out [9:8]	Left line output control	00 = L_mixer 10 = L_DAC
			01 = R_mixer 11 = R_DAC
	Right_LB_ADC_2_DAC [7]	Loop back ADC to DAC	1 = active
	ADC_Chop_DIS [6]		0 = enable
	Txfilter_Chop_Dis [5]		0 = enable
	Offset_Cancel_Disable [4]		0 = enable
	Enable_Serial _Test_Out [3]	Serial output of control register	1 = enable
	Left_LB_DAC_2_ADC_bar [2]	Loop back DAC to ADC	0 = active
	Right_LB_DAC_2_ADC_bar [1]	Loop back DAC to ADC	0 = active
	Over_Sampling_Ratio [0]	Choose 128 or 256 (Note 1)	1 = 256 0 = 128

Notes

1. Although the sampling rate can be changed from 128 to 256, the external clock rate remains the same.
2. Mixer Input Control: This 7-bit control word controls the mux to the mixer. Each bit is an input enable, high true. Bit order is as follows: <MSB> Tx_filter, Mono, cross channel Mic, Mic, Aux1, Aux2, Line <LSB>.
3. Power-up: All mute signals are defaulted to 0 (active) and all gain settings are defaulted to 0 (mute)

NN31H - Write New Volume Setting Pair

This command may be used to set output volume for PCM, FM, or wavetable playback. The PC always sends two 16-bit parameters (left and right settings, each in the range 0-7FFFh) to the CMDR register even if only one channel is being adjusted by the user. Each parameter requires a separate CMD_WE validation step.

The command options are:

nn31h	where "nn" is the upper byte of the 16-bit value written to CMDR. In this case, the upper byte is used to hold a parameter for the command which is an index selecting which volume setting is to be changed. The legal values of "nn31" are defined as follows:
0031h	Change Left and Right PCM volume settings.
0131h	Change Left and Right FM volume settings.
0231h	Change Left and Right wavetable volume settings.
0331h	Change Left and Right PCM expansion channel output volume settings.
0431h	Change Left and Right FM expansion channel output volume settings.

32H - Write New Mixer Control Setting Pair

This command may be used to change the current mixer settings. The PC always sends two 16-bit left and right settings to the CMDR register. Each parameter contains an Address Field and a Control Word corresponding to left and right mixer settings.

The Mixer control word format is:

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
ADR				1	Control Word Specific Settings										

The parameter pairs are:

Pair 1	The first parameter is Mixer control word 1 and the second parameter is Mixer control word 5.
Pair 2	The first parameter is Mixer control word 2 and the second parameter is Mixer control word 6.
Pair 3	The first parameter is Mixer control word 3 and the second parameter is Mixer control word 7.
Pair 4	The first parameter is Mixer control word 4 and the second parameter is Mixer control word 8.
Pair 5	The first parameter is Mixer control word 9 and the second parameter is Mixer control word 10.

33H - Reset Mixer Settings

This command, sent to the CMDR register, may be used to reset all Mixer and Volume controls to the default values.

NN34H - Select Mono Record Source

This command may be used to select either the left or right source for mono recording. The upper byte of the 16-bit value written to CMDR selects the source.

The command options are:

0034h	Use the left input of the selected recording source.
0134h	Use the right input of the selected recording source.

6.6 Extended Interface Commands

NN50H - Select MIDI Output Path

This command may be used to switch the MIDI output destination between wavetable, external UART, or both. The path selection will remain in effect until either a POR, hard reset command, or another set path command occurs. The default setting is to route data to both the wavetable and the external UART (or only the UART if wavetable is not available).

NN =	The upper byte of the 16-bit value written to CMDR. In this case, the upper byte is used to hold a parameter for the command which is used by the WaveArtist to determine where to route MIDI information coming from the PC through the MPU-401 interface. The legal values of “nn50” are defined as follows:

The command options are:

0050h	Route MIDI data to the external UART.
0150h	Route MIDI data to the wavetable.
0250h	Route MIDI data to both the wavetable and the external UART (default).

Response Value:

0 =	The WaveArtist configuration does not support the requested path selection.
1 =	OK

NN62N - Initiate DRAM Download

This command may be used to initiate the DRAM "download" mode and to specify where in sample memory the download should begin. All downloaded data are transferred via the DATR register 16-bits at a time. This means that 8-bit wave data, must be prepackaged into 16-bit quantities (and sign processed) by the PC software.

Before sending the NN62H command, the PC must ensure that the DAT_WE bit is set in the STATR register. The PC must then prime the DATR register with the first 16-bit quantity to be downloaded into the sample memory. After the NN62H command is sent, the WaveArtist uses the NN as the upper byte of the 24-bit download address. The lower 2 bytes of this address are then passed to the WaveArtist via the CMDR parameter after the CMD_WE bit is set. Once the parameter has been received by the WaveArtist, the value in the DATR register is copied to the indicated address in the sample memory. The WaveArtist then waits for the PC to place another 16-bit quantity in DATR, at which time it copies that value into the next word location in the sample memory. This process is driven by the event consisting of the PC placing the next 16-bit value into DATR. It continues indefinitely, until the PC has completed downloading all necessary data, at which time it issues the 63h (terminate download) command to the WaveArtist. The WaveArtist continues accepting 16-bit download values until it receives the 63H command.

Notes:

1. It is assumed that the WaveArtist can continue to process DMA wave I/O and MPU-401 MIDI output, while in the download mode. However, the capabilities of doing simultaneous wave, MIDI, and download operations may be limited by the PC software's ability to manage these tasks concurrently.
2. The product should be configured to support DRAM download and must be within range (< 24-bits).
3. The initial value for the programmed I/O download should be pre-loaded into the DATR register before issuing this command.

63H - Terminate DRAM Download

This command instructs the WaveArtist to terminate the DRAM "download" mode. It is to be issued after the last 16-bit quantity of download data has been placed in the sample memory.

7. WINDOWS DRIVER SUPPORT FOR WAVETABLE DRAM DOWNLOAD

7.1 DRV_RWA_DOWNLOAD Message

Most of the messages handled by the WaveArtist Windows audio drivers are standard audio driver messages defined by the system. The normal way for an application program to deal with the driver is via the standard multimedia DLLs, such as provided by MMSYSTEM or MSMIXMGR, who in-turn, generate the required messages sent to the driver.

However, the wavetable download function is not a standard Windows multimedia function, and some method needed to be devised which allows an application program access to this capability. For this purpose, the installable interface (DriverProc()) message handler within the WaveArtist driver has been modified to process a new, “user defined” message, as allowed in the Windows documentation. This message is defined in the C portion of the driver as follows:

```
#define DRV_RWA_DOWNLOAD    (DRV_USER + 0)
```

The DRV_USER label is available in MMSYSTEM.H and is set by the system to the value 0x4000; the application should define this message identifier identically to the WaveArtist driver. The DRV_RWA_DOWNLOAD message is implemented as a “user defined message” which an application can send to the driver by using the “SendDriverMessage()” standard API function call. To use this message, an application first must call the standard function: OpenDriver() in order to obtain the correct handle for the WaveArtist driver. It then can use SendDriverMessage() to download DRAM data. When it has completed its download, it uses CloseDriver() to close off this special link to the driver.

Note that the SendDriverMessage() function has three parameters. The first, is the driver “handle” which is retrieved by using OpenDriver(). The second and third are optional 32-bit parameters for specific use by the driver. In this case, the first parameter (dwParam1) is setup as a pointer to the RWA_DLOADINFO structure. The DRV_RWA_DOWNLOAD message, when processed by the WaveArtist driver always results in one of two DWORD statuses:

1L	The requested download operation has completed successfully. Note, at present, no ability has been added into the driver to perform a “callback” function to inform the application of success. Instead, the entire operation is synchronous, in that the application does not receive control until the download is complete (or fails).
0L	The requested download operation has failed. Some or all of the download data did not get transferred to the WaveArtist. The WaveArtist may have a problem, or the parameters in the RWA_DLOADINFO structure may be erroneous.

7.2 RWA_DLOADINFO Structure

The application instructs the driver about specifics for the download operation by passing the driver a pointer to a "RWA_DLOADINFO" data structure, which is defined by the WaveArtist driver as follows:

```
typedef struct tagRWA_DLOADINFO
               {

                WORD  dwStructSize
                WORD  wBitsPerSample;
                LPSTR lpFileToOpen;
                DWORD dwDramAddr;
                DWORD dwDataSize;
                HPSTR hpData

               } RWA_DLOADINFO, FAR *LPRWA_DLOADINFO;
```

This structure must be properly allocated and filled out by the application software before sending the DRV_RWA_DOWNLOAD message. The fields within the RWA_DLOADINFO structure, a pointer to which must be passed to the driver as the first parameter in the SendDriverMessage() call, are utilized by the WaveArtist device driver and the application program as described below:

dwStructSize	This 16-bit field is used as a standard Windows sanity check. It must be set with the size of the RWA_DLOADINFO structure (for instance using the C language sizeof() function) by the application program before sending the download message. This runtime value must match the WaveArtist's own structure size for RWA_DLOADINFO, or the driver returns a 0L (failure) for the SendDriverMessage() function call.
wBitsPerSample	This 16-bit field is used by application if and only if the download operation is NOT for a standard RIFF format wave file. When a wave file is downloaded, the driver reads the sample width from the wave file header, and wBitsPerSample is ignored. If the file is a non-wave file however, or if the download operation is for a data buffer (lpFileToOpen is NULL) then wBitsPerSample is used by the driver to determine how to preprocess the data being downloaded. The WaveArtist driver supports only two values in this case:

16 (16-bit data, the default)	In this case, the driver will assume that all data passed to the WaveArtist, is already properly formatted as 16-bit quantities. If this is a buffer download, it is assumed that the dwDataSize parameter, which is always a "byte count" must be even, implying that dwDataSize/2 is the number of 16-bit words to be sent. In this case, the hpData parameter must point to the first 16-bit quantity. If these conditions are not met (for instance if dwDataSize is odd while wBitsPerSample is 16) then the driver returns a failure status (0L). If this is not a buffer download, but rather, a binary (non-wave) file download, the contents of the file will be treated as 16-bit word quantities. In this case the file will be broken into 64K chunks of download buffers (see lpFileToOpen description below). It is possible that the final chunk of this file will have an odd number of bytes. If wBitsPerSample is 16 in this situation, the driver will return a failed status after downloading all previous file chunks, since only the final chunk will have the odd size.

	8 (8-bit data)	In this case, the driver will assume that all data bytes passed to the WaveArtist must be sign converted. The dwDataSize parameter may be odd or even -- but if it is odd an extra byte of silence is padded since all transfers are still done 16-bits at a shot, through the DATR register. The sign conversion is applied to each byte of each 16-bit transfer. The value 0x80 is added (without carry) to each byte.

lpFileToOpen	The application sets this field to NULL if it wishes to request a download buffer operation. In that case, the hpData parameter is used by the driver to download data already present in PC memory. If however the lpFileToOpen field is not NULL, it is assumed to point to a zero terminated ASCII file name. In this case, the driver attempts to open this file, first as a RIFF wave file, and if that fails, as a binary file. If the file open fails anywhere along the line, the SendDriverMessage() return value is 0L. If a wave file is found, the wave header information is used to determine if the data is to be treated as 8-bit or 16-bit, and the driver processes the data as described under wBitsPerSample above, although the wBitsPerSample field is ignored. If this is not a wave file, the wBitsPerSample field is used. In either case, the driver attempts to download the data read from the file in 64K chunks (except for the last chunk). This allows very large files to be downloaded by the driver without needing to allocate more than 64K of memory. For 8-bit data, the final chunk may be padded with an extra silence byte. Note also, that the final chunk is sized to whatever data is left in the file, and is normally less than 64K bytes long.
dwDramAddr	This 32-bit field is used by the driver in constructing the nn62 “begin download” command to the WaveArtist. It specifies a 24-bit sample memory address where data is to be downloaded. It must be an even word address which falls within the 24-bit range, or a failure return is generated.
dwDataSize	This 32-bit field is the number of 8-bit bytes to be transferred in a buffer download request. It is ignored if this is a file download request. But if lpFileToOpen is NULL, then dwDataSize must be supplied. If wBitsPerSample is 8, then dwDataSize may be odd or even, but if this is specified as 16-bit data, the driver checks that the size is an even number of bytes. Also, a failure return is generated if “dwDataSize+dwDramAddr” would overflow the sample memory -- the requested download buffer should fit.
hpData	If this is a buffer download, the hpData field is assumed to be a “huge” pointer to the data being downloaded. This allows the data buffer to exceed 64K bytes in length if desired. For file downloads, this field is ignored by the driver.

7.3 File Download Example

The following example code shows how the DRV_RWA_DOWNLOAD message may be used by a C language application. Hopefully, other language examples will be intuitive.

```
#define DRV_RWA_DOWNLOAD (DRV_USER+0)

struct
{

  WORD  dwStructSize
  WORD  wBitsPerSample;
  LPSTR lpFileToOpen;
  DWORD dwDramAddr;
  DWORD dwDataSize;
  HPSTR hpData

} rwaDlInfo;

HDRVR rwaHANDLE;
DWORD dwDramAddr = 0;

int iCnt;


if ((rwaHANDLE = OpenDriver( "c:\\wac\\rwa.drv", NULL, NULL )) != NULL)
{

    rwaDlInfo.dwStructSize = sizeof(rwaDlInfo);
    rwaDlInfo.lpFileToOpen = (LPSTR) “TEST.WAV”;
    rwaDlInfo.dwDramAddr =   dwDramAddr;

    //would be used by driver only if not a wave file
    rwaDlInfo.wBitsPerSample = 16;
    rwaDlInfo.HpData = (HPSTR) NULL;

    //download file TEST.WAV to WaveArtist DRAM
    if (!SendDriverMessage(  rwaHANDLE,
                             DRV_RWA_DOWNLOAD,
                             (DWORD) (char _far *) &rwaDlInfo,
                             (DWORD) NULL)
       )
        MessageBox( hWnd,
                    szDLFile,
                    "Unable to Download file.",
                    MB_OK|MB_APPLMODAL|MB_ICONSTOP
                  );


    //now close down direct link to driver
    CloseDriver (rwaHANDLE, NULL, NULL);
  }
}
```

INSIDE BACK COVER NOTES

Headquarters
Rockwell Semiconductor Systems
4311 Jamboree Road,
P.O. Box C
Newport Beach, CA 92658-8902
Phone: (714) 221-4600
Fax: (714) 221-6375

European Headquarters
Rockwell Semiconductor Systems
S.A.R.L.
Les Taissounieres B1
Route des Dolines
Sophia Antipolis Cedex
06905 Valbonne
France
Phone: (33) 93 00 33 35
Fax: (33) 93 00 33 03

For more information:
Call 1-800-854-8099
International information:
Call 1-714-833-6996

URL Address:
http://www.nb.rockwell.com
E-Mail Address:
literature@nb.rockwell.com

REGIONAL SALES OFFICES

US Southwest Office
Rockwell Semiconductor Systems
5000 Birch Street
Suite 400
Newport Beach, CA 92660
Phone: (714) 222-9119
Fax: (714) 222-0620

US Southwest Satellite Office
Rockwell Semiconductor Systems
1000 Business Center Circle
Suite 215
Thousand Oaks, CA 91320
Phone: (805) 376-0559
Fax: (805) 376-8180

US South Central Office
Rockwell Semiconductor Systems
2001 North Collins Blvd
Suite 103
Richardson, TX 75080
Phone: (214) 379-9310
Fax: (214) 479-9317

US Southeast Office
Rockwell Semiconductor Systems
900 Ashwood Parkway
Suite 400
Atlanta, GA 30338
Phone: (770) 393-1830
Fax: (770) 395-1419

US Southeast Satellite Office
Rockwell Semiconductor Systems
Arbor Shoreline Office Park
19345 US 19 N.
Suite 108
Clearwater, FL 34624-3156
Phone: (813) 538-8837
Fax: (813) 531-3031

US Northwest Office
Rockwell Semiconductor Systems
US Northwest Office
3600 Pruneridge Avenue
Suite 100
Santa Clara, CA 95051
Phone: (408) 249-9696
Fax: (408) 249-7113

US North Central Office
Rockwell Semiconductor Systems
Two Pierce Place
Chancellory Park
Suite 810
Itasca, IL 60143
Phone: (708) 773-3454
Fax: (708) 773-3907

US Northeast Office
Rockwell Semiconductor Systems
239 Littleton Road
Suite 4A
Westford, MA 01886
Phone: (508) 692-7660
Fax: (508) 692-8185

Australia
Rockwell Semiconductor Systems
Rockwell Australia Limited
3 Thomas Holt Drive
P.O. Box 165
North Ryde, NSW 2113
Australia
Phone: (61-2) 805 5555
Fax: (61-2) 805 5599

Europe Mediterranean
Rockwell Semiconductor Systems
S.A.R.L.
c/o Allen Bradley Srl
Via Di Vittorio, 1
20017 Mazzo Di Rho (MI)
Italy
Phone: (39 2) 93179911
Fax (39 2) 93179913

Europe North
Rockwell Semiconductor Systems, Ltd.
Berkshire Court
Western Road
Bracknell
Berkshire RG12 1RE
England
Phone: +44 1344 486 444
Fax: +44 1344 486 555

Europe South
Rockwell Semiconductor Systems
S.A.R.L.
Tour GAN
Cedex 13
92082 Paris La Defense 2
France
Phone: (33-1) 49-06-3980
Fax: (33-1) 49-06-3990

Germany
Rockwell Semiconductor Systems
Rockwell Int'l GmbH Germany
Paul-Gerhardt-Allee 50 a
81245 Munchen
Germany
Phone: (49-89) 829-1320
Fax: (49-89) 834-2734

Hong Kong
Rockwell Int'l (Asia Pacific) Ltd.
13th Floor, Suites 8-10,
Harbour Centre
25 Harbour Road
Wanchai,
Hong Kong
Phone: (852) 2 827-0181
Fax: (852) 2 827-6488
Phone: (82.2) 565.2880
Fax: (82.2) 565.1440

Japan
Rockwell Int'l Japan Co., Ltd.
Shimomoto Bldg
1-46-3 Hatsudai, Shibuya-ku
Tokyo, 151
Japan
Phone: (81-3) 5371 1520
Fax: (81-3) 5371 1501

Korea
Rockwell-Collins Int'l, Inc.
Room No. 1508
Korea Textile Centre Building
944-31, Daechi-3dong
Kangnam P.O. Box 2037
Kangnam-ku
Seoul
Korea
Phone: (82-2) 565-2880
Fax: (82-2) 565-1440

Singapore
Rockwell-Collins Int'l, Inc.
230 Orchard Road #10-230/232
Faber House
Singapore 0923
Phone: (65) 732-2292
Fax: (65) 733-0835

Taiwan
Rockwell Int'l Taiwan Company, Ltd.
Room 2808 International Trade Bldg.
333, Keelung Road, Section I
Taipei,
Taiwan
10548 ROC
Phone: (886-2) 720-0282
Fax: (886-2) 757-6760

Rockwell
Semiconductor Systems