



# Software Development Seminar

Link Cable (Advanced)



Sony Computer Entertainment Inc.

**CONFIDENTIAL**

AT

---

# Link Cable



Sony Computer Entertainment Inc.

**CONFIDENTIAL**

**AT**

## Latest Revisions

---

### Description of revisions to the Link Cable Library

- Asynchronous write support  
Character-by-character transmission using DSR interrupt from the other machine
- Timeout monitoring function  
Callback from within internal loop for synchronous I/O
- Reduced detection frequency of receive-related errors  
CTS and DTR are turned off within the interrupt handler
- Public control line operation  
Added "\_comb\_control()" function
- Correction of incomplete "DeIDRV()" operation



## Link Cable Serial Controller (1)

- Asynchronous serial communications (incompatible with RS-232C)
- Effective data volume is equivalent to that of a memory card  
(occupies 3% of CPU time at 4K bytes/second)
- Communications specifications

Item	Settings
Character length:	5, 6, 7 or 8 bits
Stop bits:	0, 1, 1.5, or 2 bits
Parity checking:	None, even, odd
Communications speed:	1 to 2,073,600 bps (divisors of 2,073,600 only)



## Link Cable Serial Controller (2)

- Buffers: Receive: 8 bytes; transmit: 1 byte
- Control lines: Two pairs: DTR/DSR and RTS/CTS

Name on transmitting side	Name on receiving side	Function on receiving side
DTR	DSR	Receiving function automatically stops when off
RTS	CTS	Receiving function automatically stops when off

- Interrupts: 1 2, 4, or 8 bytes received  
+DSR/ level  
+errors



## Link Cable Library (1)

---

File name	Description
libcomb.h	Include header
libcomb.lib	Library



## Link Cable Library (2)

---

- Basic configuration  
Link Cable driver + Link Cable BIOS
- List of functions

AddCOMB	Initializes the Link Cable driver
DelCOMB	Deletes the Link Cable driver
ChangeClearSIO	Sets the Link Cable driver interrupt
_comb_control	Link Cable BIOS interface



# Link Cable BIOS

---

- Interface function: `_comb_control( )`
- Enables execution of control line operations, cancellation of I/O request, etc.
- Effective only after the Link Cable Driver is installed
- Not covered in this seminar





## Link Cable Driver: Overview

---

I/O provided through standard C procedures

Item	Description
Device name	sio
Block size: 1 byte	
Asynchronous mode: "O_NOWAIT"	Specified by

macro when device is opened



## Link Cable Driver: Related Events

- Related events

Source descriptor	Event type	Report contents
HwSIO	EvSpIOEW	End of asynchronous write
	EvSpIOER	End of asynchronous read
	EvSpERROR	Receive error occurred (asynchronous only)
	EvSpTIMOUT	Timeout during synchronous read/write



## Link Cable Driver: Initialization and Deletion

---

- Initialization: `AddCOMB( )`
- Deletion: `DelCOMB( )`

(Did not operate properly until  
Library version 3.4, due to a bug)



# Synchronous I/O: Overview

---

- Methods conform with standard C rules
- Functions terminate upon completion of I/O



## Synchronous I/O: Wait Callback

---

- Called during a synchronous "read()" or a synchronous "write()"
- Registration process: `_comb_control(4 , 0 , func)`  
(Default: Not registered)
- Deletion process: `_comb_control(4 , 0 , NULL)`
- Callback function specifications

Format: `long func( long spec, unsigned long count )`

Parameters: `spec`      1: Synchronous read in progress

                          2: Synchronous write in progress

`count` Current value of internal counter

Return values: "0" is returned when wait loop was terminated due to timeout,

"1" when wait continues.



# Synchronous I/O: Termination Conditions

---

- Synchronous read: Return value is normally the number of characters received
  - (1) Reception of specified number of characters is completed
  - (2) Detection of a reception error, such as a parity, overrun or frame error
  - (3) Value of wait callback function is "0"
- Synchronous write: Return value is normally the number of characters sent
  - (1) Transmission of specified number of characters is completed
  - (2) Value of wait callback function is "0"
  - (3) DTR from other machine is "0"  
(invalid for open mode "O\_NBLOCK")



# Asynchronous I/O: Overview

---

- Entails non-standard C methods and overhead
- Read and write functions terminate immediately only with I/O request registration
- Completion of I/O is reported through events



## Asynchronous I/O: Number of Characters in Receive Units

- Asynchronous I/O is interrupt driven => Source of overhead
- Asynchronous reads use an interrupt each time that 1, 2, 4, or 8 characters are received
- Asynchronous writes use DSR interrupts
- Overhead for writes  $\geq$  overhead for reads (write  $\geq$  read)





## Asynchronous I/O: Error Processing

---

- Error processing in asynchronous I/O

(1) Detection of receive error

(2) Cancellation of asynchronous read request

Function used:

```
/* Cancels the asynchronous read request and  
clears the serial controller error flag */
```

```
_comb_control( 2, 3, 0 );
```

(3) Delivery of "EvSpERROR" event



## Control Line Transitions (1)

Driver operation	DTR	RTS
Power on (No other machine present, or power for other machine is off)	1	1
(Other machine present, driver not initialized)	0	0
Driver initialization AddCOMB( )	0	0
Driver deletion DelCOMB( )	-	-

-: No change , 0:OFF , 1:ON



## Control Line Transitions (2)

Driver operation	DTR	RTS
Synchronous write		
open("sio",O_WRONLY);	-	-
write( ... );	-	-
write complete	-	-
close( );	-	-
Synchronous read		
open("sio",O_RDONLY);	-	-
read( ... );	1	1
read complete	0	0
close( );	-	-



## Control Line Transitions (3)

Driver operation	DTR	RTS	
Asynchronous write			
open("sio",O_WRONLY O_NOWAIT);	-	-	
write( ... );	-	-	
DSR interrupt generated	0	0	(save state)
DSR interrupt complete			(restore state)
write complete	-	-	
close();	-	-	



## Control Line Transitions (4)

Driver operation	DTR	RTS
Asynchronous read		
open("sio",O_RDONLY O_NOWAIT);	-	-
read( ... );	1	1
Receive interrupt generated	0	0 (save state)
Receive interrupt complete		(restore state)
read complete	0	0
close ();	-	-

