C²I² Systems

CCII Systems (Pty) Ltd Registration No. 1990/005058/07

Communications
Computer Intelligence
Integration

User Manual

for the

PMC High-Speed Serial Windows NT 4 Driver

C²I² Systems Document No.	CCII/HSS/6-MAN/004
Document Issue	1.1
Issue Date	2009-05-27
Print Date	2009-05-28
File Name	P:\HSS\TECH\MAN\CHSMAN04.WPD
Distribution List No.	

Document prepared by and for C²I² Systems, Cape Town

Signature Sheet

Name	Signature	Date
Completed by R. Bennett	pp Project Engineer HSS C²I² Systems	2009-05-27
Accepted by	Project Manager HSS C²I² Systems	2009-05-27
Accepted by X. Kruger	Quality Assurance C²I² Systems	2009-05-27

Amendment History

Issue	Description	Date	ECP No.
1.0	Initial version.	2003-07-18	-
1.1	Changed title.	2009-05-27	CCII/BLPGEN/6-ECP/015

Contents

List of Illustrations

Abbreviations and Acronyms

API	Application Program Interface
BIT	Built-In-Test
BRG	Baudrate Generator
CCII	Communications, Computer Intelligence, Integration
C^2I^2	C^2I^2 Systems (Pty) Ltd
HDD	Hard Diskdrive
HSS	High Speed Serial (Acronym for the C^2I^2 PMC Serial I/O card project)
I/O	Input/Output
PC	Personal Computer
PCI	Peripheral Component Interconnect
PMC	PCI Mezzanine Card
PnP	Plug and Play
SBC	Single Board Computer
SCC	Serial Communications Controller
SDK	Software Development Kit
SIO	Serial Input/Output
SMC	Serial Management Controller
TBD	To Be Determined
WDM	Windows Driver Model
WMI	Windows Management Instrumentation

1. Scope

1.1 Identification

This document is the User's Manual for the C²I² Systems' Peripheral Component Interconnect (PCI) Mezzanine Card (PMC) High Speed Serial Windows NT 4 Driver.

1.2 Introduction

The PMC High Speed Serial (HSS) driver is a low level, device-dependant interface for transferring data over a C²I² Systems' HSS PCI Mezzanine Card (PMC). The driver binaries are provided with explicit installation instructions.

The driver will also run as a legacy driver under Windows 2000, but does not support Plug and Play (PnP), Windows Management Instrumentation (WMI) or power management.

The driver software distribution consists of (at least) the following files :

hss4ntxyz.zip	An archive file containing all the files required for the driver installation. XYZ is the revision number for this driver release.
Setup.exe	Install wizard application extracting the following files to the desired locations...
hss4wnt.sys	HSS 4 Port Device Driver
hss4port.cpl	Control Panel Applet
flashprog.exe	Flash update application
hssReadme.txt	General information
hssRelease_notes.txt	Release notes and revision history. Please check this file for information on the latest updates.

2. Applicable Documents

2.1 Specifications

Not applicable.

2.2 Standards

2.2.1 DI-IPSC-81443 : Data Item Description for a Software User Manual.

2.3 Other Documents

2.3.1 MPC860 PowerQUICC™ User's Manual Rev. 1, http://e-www.motorola.com

2.3.2 MSDN Communications Resources,
http://msdn.microsoft.com/library/default.asp?url=/library/en-us/devio/base/communications_resources.asp

3. Configuration Procedure

This paragraph describes the installation procedure for the HSS driver.

3.1 Installing the Driver Files

Unzip the file “hss4ntxyz.zip” to any suitable folder on your local PC Hard disk drive (HDD). You must have administrative privileges on the PC. Run the application “Setup.exe” (Figure 1). All the driver files will be extracted to the required locations, i.e. the device driver file will be stored in “WINNT\system32\drivers”, the control panel applet in “WINNT\system32" and the flash update application in “Program Files\C2I2Systems\SIO4Port”.

As this is not a Windows Driver Model (WDM) driver, Windows 2000 will report device conflicts between the driver and a “Unknown PCI bridge” device. This is due to the NT driver not supporting PnP. This reported conflict will not affect the performance of the driver under Windows 2000.

Figure # 1 : Installation Wizard

3.2 Uninstalling the Driver

In the Start menu, select Control Panel from the Settings menu. Click on “Add/Remove Programs”. Select the “C²I² SIO 4 Port Driver” from the list and click the Remove button. Answer yes to delete all files from the HDD.

3.3 Updating the Device Firmware

Always ensure that when a new driver is installed, the corresponding firmware revision on the device is identical. There might be incompatibilities between different device driver and firmware versions. The Engine version reported by the control panel applet must match the firmware version of all devices in the system.

The flash update application is located in the “SIO4Port” folder. Before running the application, ensure that all ports on the device are closed. The firmware images are located on the supplied CD-ROM or the C²I² Systems’ website.

Do not remove the power from the PC until the flash programming is completed.

The syntax for the application is as follows...

Flashprog.exe # [b s u] filename

#	The port number residing on the device which will be updated. [1]
b	The filename following this flag is a binary image.[2]
s	The filename following this flag is a Motorola S-record file.[2]
u	No firmware updates will be done, the current firmware image will be stored in filename.

Note (1) : Only one port per device must be specified.
Note (2) : Whenever the firmware is updated, the current firmware will be stored in the file “ccbackup.bin”. Rename this file before using the application again.

Example :

flashprog 1 s ccHssvxyz.hex
Will update the firmware of the first device with the S-record file ccHssvxyz.hex. Using port number 7 will update the firmware on the second device.

Inspect the HSS PMC card and find the oscillator module. This is a silver can about 1cm x 1cm (0.4" x 0.4") in size. The oscillator frequency is marked on the can. On most boards, including the front panel board, it will be 39.3216. Use the corresponding S-record file matching your oscillator frequency.

The S-record filename will be

ccHss4vx.y.z-39.3216.hex (39.3216 MHz oscillator on PMC card)
ccHss4vx.y.z-39.9360.hex (39.9360 MHz oscillator on PMC card)
ccHss4vx.y.z-40.0000.hex (40.0000 MHz oscillator on PMC card)
ccHss4vx.y.z-48.0000.hex (48.0000 MHz oscillator on PMC card)
ccHss4vx.y.z-50.0000.hex (50.0000 MHz oscillator on PMC card)

Where x.y.z is the version of the firmware.

3.4 Using the Event Viewer

The Windows administrative tool, “Event Viewer” can be used to inspect the event logs. The HSS driver logs certain information and fatal errors to the event log. Refer to the event log when an operation does not function as expected.

4. Application Program Interface (API)

The HSS Windows NT driver complies to most of the Windows 32 API for Serial devices. Refer to the Platform SDK in [2.3.2] for the communications resource documentation. The serial function prototypes can be found in the SDK file “winbase.h”.

The device ports are named HSSx. HSS1-HSS6 are the ports of the first device in the system, HSS7-HSS12 are the ports of the second device.

4.1 Windows SDK Serial Functions

This paragraph lists the serial functions supported by the HSS device driver :

- BuildCommDCB
- BuildCommDCBAndTimeouts
- ClearCommError
- CommConfigDialog
- GetCommConfig
- GetCommMask
- GetCommProperties
- GetCommState
- GetCommTimeouts
- GetDefaultCommConfig
- PurgeComm
- SetCommConfig
- SetCommMask
- SetCommState
- SetCommTimeouts
- SetDefaultCommConfig
- SetupComm
- WaitCommEvent
- CreateFile
- ReadFile
- WriteFile
- CloseHandle

4.2 Windows SDK Serial Structures

This paragraph lists the serial structures supported by the HSS device driver :

- COMMCONFIG
- COMMPROP
- COMMTIMEOUTS
- COMSTAT
- DCB

4.3 Function Limitations

Not all settings are supported for each serial function. This paragraph will mention all the exceptions.

4.3.1 ClearCommError

Only CE_FRAME and CE_RXPARITY are supported.

4.3.2 GetCommMask

Only the EV_RXCHAR event mask is supported.

4.3.3 SetCommMask

Only the EV_RXCHAR event mask is supported.

4.3.4 WaitCommEvent

Only the EV_RX_CHAR event mask is supported.

4.3.5 Overlapped Writes

When a port is opened in overlapped (non-blocking) mode, overlapped writes might not behave as expected. An overlapped read will return immediately, and the event will be signalled once data has been received. For an overlapped write the function will not return immediately. The device driver has to send the data to the card, where it will be transmitted. This transfer does require a finite amount of time. Changing the driver architecture to match the overlapped read operation would degrade its throughput performance on transmission.

4.3.6 FlushFileBuffers

Not supported.

4.4 Structure Limitations

Not all the fields of the serial structures are used by the HSS device driver. This paragraph will mention all the exceptions.

4.4.1 COMMCONFIG

dwProviderSubType : none of the types makes provision for a device that is both RS232 and RS422 capable. No provider-specific data is supplied.

4.4.2 COMMPROP

dwProvSubType, dwProvSpec1, dwProvSpec2 and *wcProvChar* are not supported.

4.4.3 COMMTIMEOUTS

ReadIntervalTimeout is not supported. The write timeout value is used by the HSS driver to flush its internal transmitter queue. The internal queue is only used when one byte is transmitted at a time.

4.4.4 COMSTAT

Only *cbInQue* and *cbOutQue* are supported in this structure.

4.4.5 DCB

The following fields are not supported : *fOutxDsrFlow, fDtrControl , fDsrSensitivity, fTXContinueOnXoff, fOutX, fInX, fErrorChar, fNull, fRtsControl, fAbortOnError, XonLim, XoffLim, XonChar, XoffChar, ErrorChar, EofChar, EvtChar.*

StopBits of ONE5STOPBITS is not supported by the HSS device driver.

Note : The XON, XOFF flow control is not supported by the HSS device driver. To use no flowcontrol, the *fOutxCtsFlow* field must be set to FALSE.

5. Driver Protocol Settings

The HSS PMC has four serial communications controllers (SCC's) [Ports 1-4] that support UART and HDLC/SDLC protocols, and two serial management controllers (SMC's) [Ports 5&6] that support only asynchronous UART.

The control panel applet allows the user to set all the protocol-specific options available on the HSS PMC communication controller chip (the MPC860 PowerQUICC™). For available options for each of the fields, see [2.3.1].

This section details the information used by each protocol and explains the use and limitations of every member.

5.1 Protocol Selection

Each port must be configured to use a protocol and electrical interface.

Protocol :

UART
HDLC
BISYNC
SMC_UART

Elec. Interface : (only used for HSS Front Panel boards)

RS485	[RS485/422]
RS232 (int. control)	[RS232 : control lines (RTS, CTS, CD) are connected internally]
RS232 (ext. control)	[RS232 : control lines (RTS, CTS, CD) need to be connected externally]

The protocol settings for the device can be set through a control panel applet. Access the applet by clicking on the Start menu and selecting the Control Panel from the Settings option. Click on the icon shown in Figure 2.

Figure # 2 : Applet Icon

5.2 Using the Control Panel to Change Port Settings

The control panel applet selection boxes list available options for the specific protocols. Options that are not available for the selected protocol are grayed out. All the options (including the options in the Advanced dialog) must be entered to create a valid protocol setting. Always click the Apply button before closing any of the dialog windows. The settings are applied to the HSS port when it is reopened.

When “Lock Settings” is checked, all requests via the Win32 API will be ignored. Any change in baudrate or parity will then not be updated, the driver will keep the setting as specified in the control panel. When the settings are not locked, the baudrate, etc. may be updated via the Win32 API functions.

When using the HDLC or BISYNC protocols, it is recommended to use the “Lock Settings” option.

The settings are stored in the Windows registry, under the keys :
“HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SIO4P\DeviceX\PortX”

5.3 Using the Control Panel to Obtain the Current Version Information

Clicking on the Information icon near the bottom of the dialog screen will display the driver information dialog shown in Figure 3 .

The Driver version is referring to the device driver itself. It has the format X.Y.Z. A change in X would indicate that the driver has an added feature. Y would mean that a new driver engine is used in the device driver. Z indicates any corrections to problems in the driver.

The Engine version must be identical to the firmware version for all cards in the PC. Normally this would only be different when the driver version is checked after a firmware update and the driver itself has not been updated.

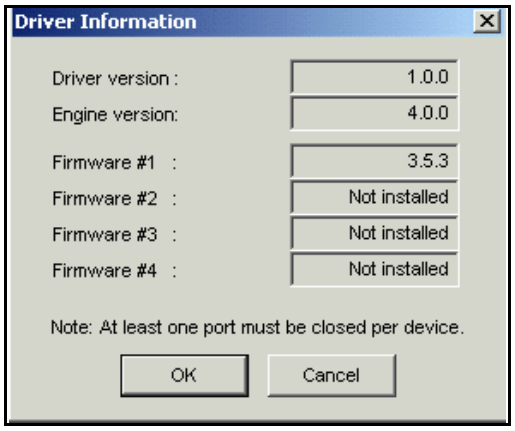

Figure # 3 : Driver Information Dialog

5.4 UART Mode

This protocol may only be used with the four SCC ports : HSS1-4. The UART Dialog window is shown in Figures 4 and 5. The settings are described in Paragraph 5.4.1.

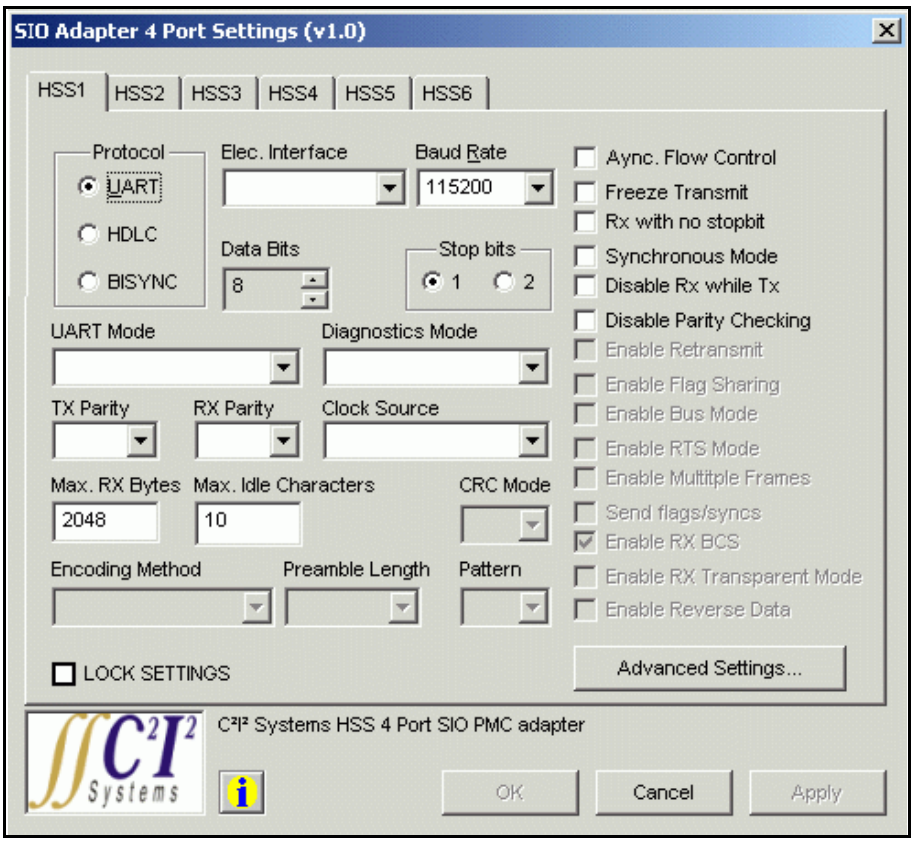

Figure # 4 : UART Dialog

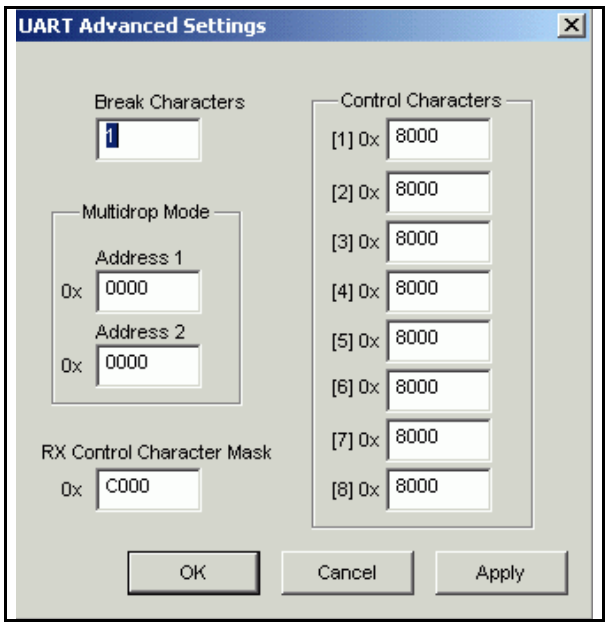

Figure # 5 : UART Advanced Dialog

5.4.1 UART Protocol Information

Name	Options		Description
Baud Rate	1200 - 115kbps (RS232) 1200 - 2.4Mbps (RS422/RS485) Any values permissible.		This member is used to specify a single baudrate for both transmitter and receiver.
Clock Source	CLOCK_DEFAULT		CLOCK_DEFAULT connects BRG[1-4] to Port[1-4]. For **synchronous UART**: when transmit clock is set to CLOCK_BRG[1-4], then receive clock is still set to CLOCK_EXT[1-4] for Port[1-4]. For **asynchronous UART**: transmit & receive clocks can be set to one of CLOCK_BRG[1-4] or CLOCK_EXT[1-4].
	CLOCK_BRG1 CLOCK_BRG2 CLOCK_BRG3 CLOCK_BRG4	Baudrate Generators [1-4].	
	CLOCK_EXT1 CLOCK_EXT2 CLOCK_EXT3 CLOCK_EXT4	External Clocks connected on Pins: RXCLK[1-4] (RS232) or CLKIN[1-4] (RS485/RS422). **Note:** CLOCK_EXT[1-2] can only be used for SCC Port[1&2], while CLOCK_EXT[3-4] can only be used for SCC Port[2&3].	
Async. Flow Control			Normal or asynchronous flow control.
Stop bits	ONE TWO		Number of full stop bits.
Data Bits	5 6 7 8		Number of data bits.
UART Mode	NORMAL MAN MM AUTO MM		Select UART mode: normal, manual multidrop or automatic multidrop mode.
Freeze Transmit			Pause (freeze) transmission. Transmission continues when set back to normal.
Rx with no stopbit			If set, the receiver receives data without stop bits.
Sync. Mode			Select asynchronous (normal) or synchronous mode.
Disable Rx while Tx			Enable (normal) or disable receiver while transmitting. Used in multidrop mode to prevent reception of own messages.
Disable Parity Checking			Enable or disable parity checking.
TX parity, RX parity	ODD LOW EVEN HIGH		Receive and transmit parity. Parity will only be checked if parity is enabled.

Diagnostics Mode	NORMAL	Normal operation. Use this for **external loopback**.	Set diagnostic mode. **External loopback - RS485:** connect TXD+ to RXD+, TXD- to RXD-, (TXCLK+ to RXCLK+ and TXCLK- to RXCLK- for synchronous mode). **External loopback - RS232:** connect TXD to RXD, (TXCLK to RXCLK for synchronous mode) and RTS to CTS & CD. For HSS Front Panel I/O Board: program *elec_interface= RS232 (int.control)* and connect TXD to RXD, (TXCLK to RXCLK for synchronous mode). Ignore RTS, CTL & CD.
	LOOPBACK	**Internal loopback:** TXD & RXD are connected internally. The value on RXD, CTS & CD is ignored. The transmitter and receiver share the same clock source.	
	ECHO	The transmitter automatically resends received data bit-by-bit.	
	LOOPBACK_ECHO	Loopback and echo operation occur simultaneously.	
Max. RX Bytes	1 to 2048 (default)		Maximum number of bytes that may be copied into a buffer.
Max. Idle Characters	0 to 2048 (default)		Maximum idle characters. When a character is received, the receiver begins counting idle characters. If max_idl idle characters are received before the next data character, an idle timeout occurs and the buffer is closed. Thus, max_idl offers a way to demarcate frames. To disable the feature, clear max_idl. The bit length of an idle character is calculated as follows: 1 + data length (5-9) + 1 (if parity is used) + number of stop bits (1-2). For 8 data bits, no parity, and 1 stop bit, the character length is 10 bits.
Break Characters	0 - 2048		Number of break characters sent by transmitter. For 8 data bits, no parity, 1 stop bit, and 1 start bit, each break character consists of 10 zero bits.
Address1, Address2	0x0000 - 0x00FF		Address in multidrop mode. Only the lower 8 bits are used so the upper 8 bits should be cleared.
Control Characters[8]	0b00------cccccccc - 0b10------cccccccc -	valid entry entry not valid and is not used.	Control character 1 to 8. These characters can be used to delimit received messages. ------ (6 bits) - reserved. Initialise to zero. cccccccc (8 bits) - defines control characters to be compared to the incoming character.
RX Control Character Mask	0b11------00000000 - 0b11------11111111 -	ignore these bits when comparing incoming character enable comparing the incoming character to cc[n].	Receive control character mask. A one enables comparison and a zero masks it.

5.5 HDLC Mode

This protocol may only be used with the four SCC ports : HSS1-4. The HDLC Dialog windows is shown in Figures 6 and 7. The settings are described in Paragraph 5.5.1.

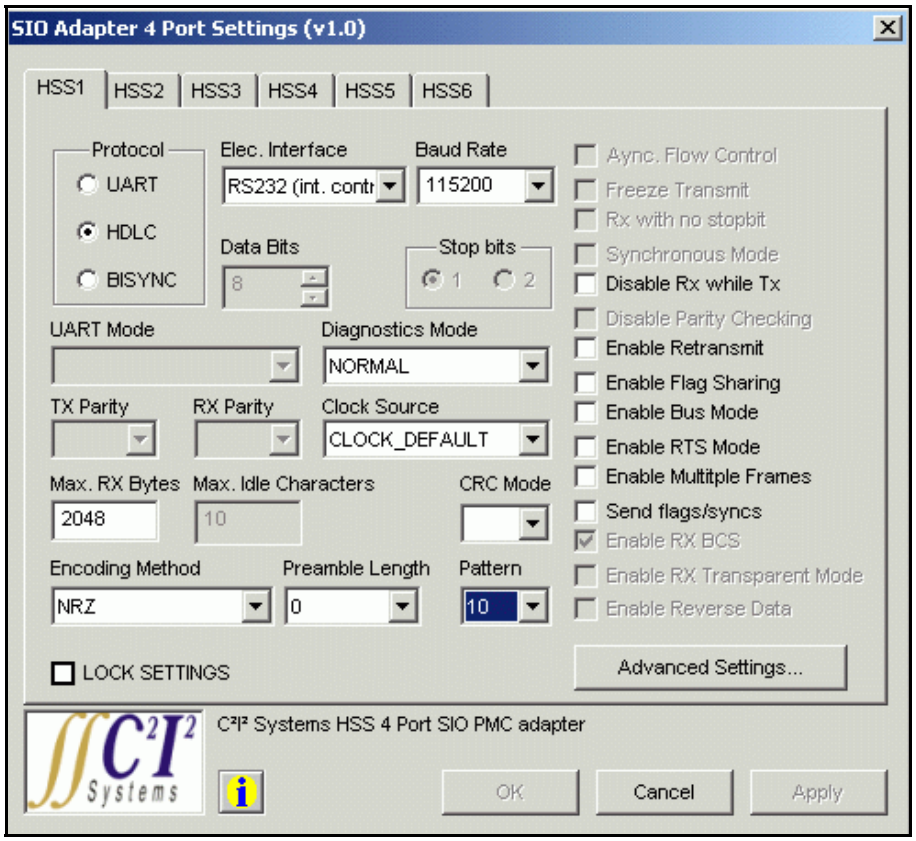

Figure # 6 : HDLC Dialog

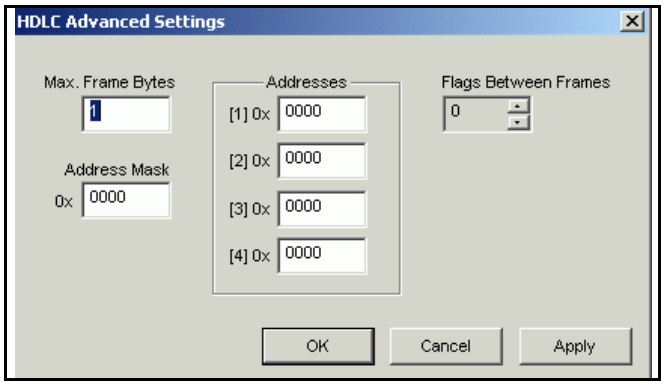

Figure # 7 : HDLC Advanced Dialog

5.5.1 HDLC Protocol Information Members

Name	Options		Description
Baud Rate	1200 - 115kbps (RS232) 1200 - 12Mbps (RS422/RS485) Any values permissible.		
Clock Source	CLOCK_DEFAULT		CLOCK_DEFAULT connects BRG[1-4] to Port[1-4]. For **NRZ/NRZI**: when transmit clock is set to CLOCK_BRG[1-4], then receive clock is still set to CLOCK_EXT[1-4] for Port[1-4]. For **FM0/1, Manchester & Diff. Manchester**: transmit & receive clocks can be set to one of CLOCK_BRG[1-4] or CLOCK_EXT[1-4].
	CLOCK_BRG1 CLOCK_BRG2 CLOCK_BRG3 CLOCK_BRG4	Baudrate Generators [1-4].	
	CLOCK_EXT1 CLOCK_EXT2 CLOCK_EXT3 CLOCK_EXT4	External Clocks connected on Pins: RXCLK[1-4] (RS232) or CLKIN[1-4] (RS485/RS422). **Note:** CLOCK_EXT[1-2] can only be used for SCC Port[1&2], while CLOCK_EXT[3-4] can only be used for SCC Port[3&4].	
CRC Mode	16 bit 32 bit		HDLC CRC mode.
Diagnostics Mode	NORMAL	Normal operation. Use this for **external loopback**.	Set diagnostic mode. **External loopback - RS485:** connect TXD+ to RXD+, TXD- to RXD-, (TXCLK+ to RXCLK+ and TXCLK- to RXCLK- for synchronous mode). **External loopback - RS232:** connect TXD to RXD, (TXCLK to RXCLK for synchronous mode) and RTS to CTS & CD. For HSS Front Panel I/O Board: program *elec_interface= RS232 (int. control)* and connect TXD to RXD, (TXCLK to RXCLK for synchronous mode). Ignore RTS, CTL & CD. For synchronous mode: see **encoding_method**.
	LOOPBACK	**Internal loopback:** TXD & RXD are connected internally. The value on RXD, CTS & CD is ignored. The transmitter and receiver share the same clock source.	
	ECHO	The transmitter automatically resends received data bit-by-bit.	
	LOOPBACK_ECHO	Loopback and echo operation occur simultaneously.	
Max. RX Bytes	1 to (2048 - CRC bytes (2 or 4)) (default)		Maximum number of bytes to receive before closing buffer. Set equal to max_frame_bytes.
Max. Frame Bytes	1 to 2048 (default)		Maximum number of bytes per frame. Set equal to the number of data bytes plus the number of CRC bytes (either 2 or 4) per frame.
Address Mask	0x0000 - 0xFFFF		HDLC address mask. A one enables comparison and a zero masks it.

Address[4]	0x0000 - 0xFFFF	Four address registers for address recognition. The SCC reads the frame address from the HDLC receiver, compares it with the address registers, and masks the result with address_mask. For example, to recognize a frame that begins 0x7E (flag), 0x68, 0xAA, using 16-bit address recognition, the address registers should contain 0xAA68 and address_mask should contain 0xFFFF. For 8-bit addresses, clear the eight high-order address bits.
Flags between Frames	0 - 15	Minimum number of flags between or before frames.
Enable Retransmit		Enable re-transmit.
Enable Flag Sharing		Enable flag sharing.
Disable Rx while Tx		Disable receive during transmit.
Enable Bus Mode		Enable bus mode.
Enable RTS Mode		Enable special RTS operation in HDLC bus mode.
Enable Multiple Frames		Enable multiple frames in transmit FIFO.
Encoding Method	NRZ NRZI_MARK NRZI_SPACE FM0 FM1 MANCHESTER DIFF_MANCHESTER	Rx / Tx encoding method. NRZ and NRZI use no DPLL. FM0/1, Manchester & Diff_Manchester use the DPLL for clock recovery.The clock rate is 16x when the DPLL is used.
Preamble Length	0 8 16 32 48 64 128	Determines the length of the preamble pattern.
Pattern	00 10 01 11	Determines what bit pattern precedes each Tx frame.
Send flags/sync		Send either idles or flags/syncs between frames as defined by the protocol. For HDLC the flag is defined as 0x7E. NRZI encoding methods may only be used with flags/syncs.

5.5.2 Preamble Requirements

Decoding Method	Preamble Pattern	Minimum Preamble Length Required
NRZI Mark	All zeros	8-bit
NRZI Space	All ones	8-bit
FM0	All ones	8-bit
FM1	All zeros	8-bit
Manchester	101010...10	8-bit
Differential Manchester	All ones	8-bit

5.6 BISYNC Mode

This protocol may only be used with the four SCC ports : HSS1-4. The BISYNC Dialog windows is shown in Figures 8 and 9. The settings are described in Paragraph 5.6.1.

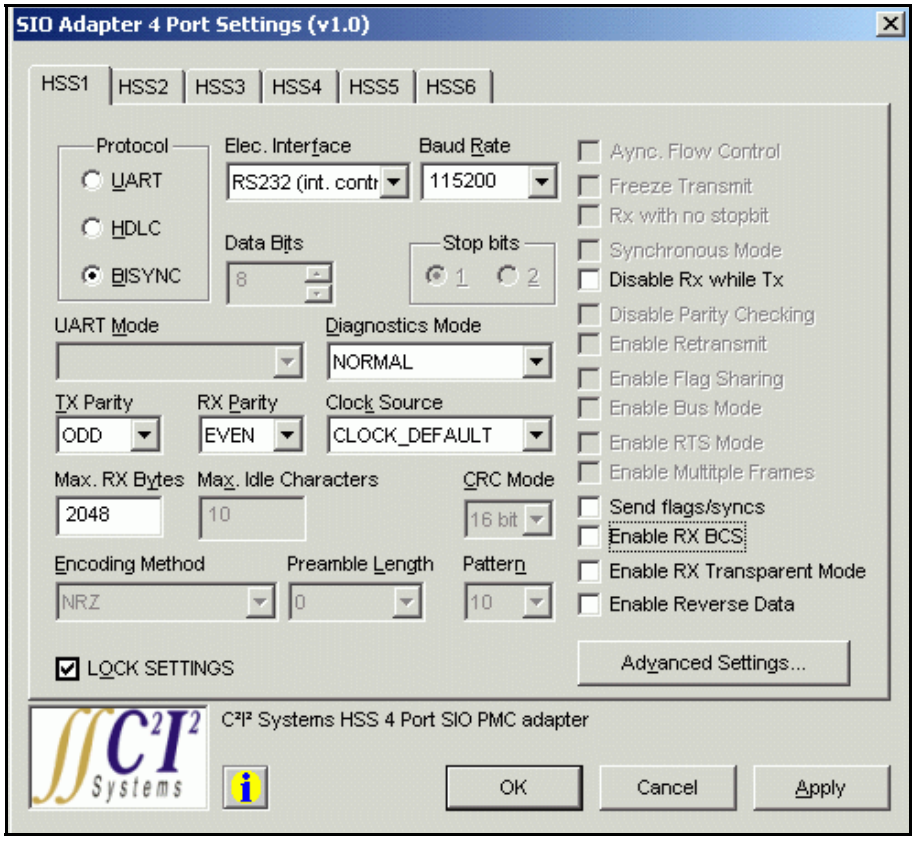

Figure # 8 : BISYNC Dialog

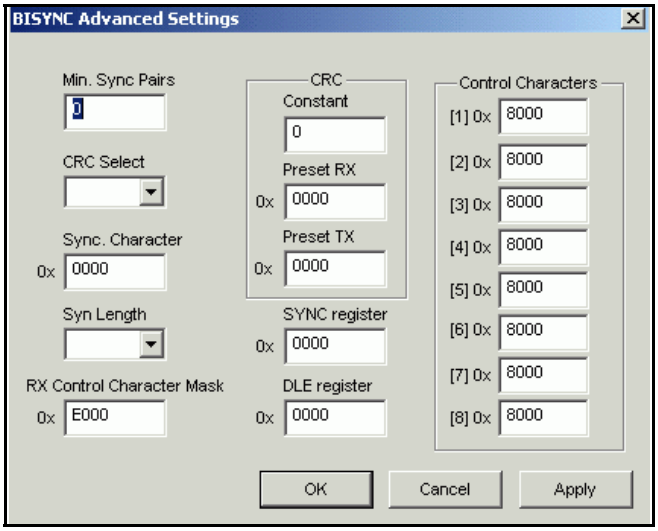

Figure # 9 : BISYNC Advanced Dialog

5.6.1 BISYNC Protocol Information Members

Name	Options		Description
Baud Rate	1200 - 115kbps (RS232) 1200 - 12Mbps (RS422/RS485) Any values permissible.		This member is used to specify a single baudrate for both transmitter and receiver.
Clock Source	CLOCK_DEFAULT		CLOCK_DEFAULT connects BRG[1-4] to Port[1-4]. When the transmit clock is set to CLOCK_BRG[1-4], then receive clock is still set to CLOCK_EXT[1-4] for Port[1-4].
	CLOCK_BRG1 CLOCK_BRG2 CLOCK_BRG3 CLOCK_BRG4	Baudrate Generators [1-4].	
	CLOCK_EXT1 CLOCK_EXT2 CLOCK_EXT3 CLOCK_EXT4	External Clocks connected on Pins: RXCLK[1-4] (RS232) or CLKIN[1-4] (RS485/RS422). **Note:** CLOCK_EXT[1-2] can only be used for SCC Port[1&2], while CLOCK_EXT[3-4] can only be used for SCC Port[3&4].	
Max. RX Bytes	1 to (2048 - 2 CRC bytes) (default)		Maximum number of bytes to receive before closing buffer.
Min. Sync Pairs	0b0000 (0 pairs) - 0b1111 (16 pairs)		Minimum number of SYN1-SYN2 pairs sent between or before messages. The entire pair is always sent, regardless of the **syn_length** variable.
CRC Select	16 LRC		CRC selection. 1: CRC16 (X16 + X15 + X2 + 1): initialise **prcrc** & **ptcrc** to all zeros or all ones. 2: LRC (sum check): for even LRC, initialise **prcrc** & **ptcrc** to zeros, for odd LRC initialise to ones.
Enable RX BCS			Enable Receive Block Check Sequence (BCS).
Enable RX Transparent Mode			Enable Receiver transparent mode. FALSE: normal receiver mode with SYNC stripping and control character recognition. TRUE: transparent receiver mode. SYNC's, DLE's and control characters are recognised only after the leading DLE character. The receiver calculates the CRC16 sequence even if it is programmed to LRC while in transparent mode. Initialize **prcrc** to the CRC16 preset value before setting **rx_transparant_mode**.
Enable Reverse Data			Enable Reverse data.
Disable Rx while Tx			Disable receiver while sending.

Rx Parity, Tx Parity	ODD LOW EVEN HIGH		Receive and transmit parity. Parity is ignored unless **crc_select** = LRC.
Diagnostics Mode	NORMAL	Normal operation. Use this for external loopback.	Set diagnostic mode. **External loopback - RS485:** connect TXD+ to RXD+, TXD- to RXD-, TXCLK+ to RXCLK+ and TXCLK- to RXCLK-. **External loopback - RS232:** connect TXD to RXD, TXCLK to RXCLK and RTS to CTS & CD. For HSS Front Panel I/O Board: program *elec_interface= RS232 (int. control)* and connect TXD to RXD, TXCLK to RXCLK. Ignore RTS, CTL & CD.
	LOOPBACK	**Internal loopback:** TXD & RXD are connected internally. The value on RXD, CTS & CD is ignored. The transmitter and receiver share the same clock source.	
	ECHO	The transmitter automatically resends received data bit-by-bit.	
	LOOPBACK_ECHO	Loopback and echo operation occur simultaneously.	
CRC Constant	0		CRC constant value.
CRC Preset RX CRC Preset TX	0x0000 or 0xFFFF		Preset receiver / transmitter CRC16/LRC. These values should be preset to all ones or zeros, depending on the BCS used.
SYNC register	0bv0000000ssssssss		BISYNC SYNC register. Contains the value of the SYNC character stripped from incoming data on receive once the receiver synchronizes to the data using the SYN1- SYN2 pair. v - if v = 1and the receiver is not in hunt mode when a SYNC character is received, this character is discarded. ssssssss (8 bits) - SYNC character. When using 7-bit characters with parity, the parity bit should be included in the SYNC register value.
DLE register	0bv0000000dddddddd		BISYNC DLE register. In transparent mode, the receiver discards any DLE character received. v - if v = 1and the receiver is not in hunt mode when a DLE character is received, this character is discarded. dddddddd (8 bits) - DLE character. This character tells the receiver that the next character is text.

Control Characters[8]	0b0bh-----cccccccc - valid entry 0b1bh-----cccccccc - entry not valid and is not used.	Control character 1 to 8. ----- (5 bits) - reserved. Initialise to zero. b - Block check sequence expected. A maskable interrupt is generated after the buffer is closed. b = 0: the character is written into the receive buffer and the buffer is closed. b = 1: the character is written into the receive buffer. The receiver waits for 1 LRC or 2 CRC bytes and then closes the buffer. h - Enables hunt mode when the current buffer is closed. h = 0: the BISYNC controller maintains character synchronisation after closing the buffer. h = 1: the BISYNC controller enters hunt mode after closing the buffer. When b = 1, the controller enters hunt mode after receiving LRC or CRC. cccccccc (8 bits) - defines control characters to be compared to the incoming character. When using 7-bit characters with parity, include the parity bit in the character value.
RX Control Character Mask	0b11------00000000 - ignore these bits when comparing incoming character 0b11------11111111 - enable comparing the incoming character to cc[n].	Receive control character mask. A one enables comparison and a zero masks it.
Sync. Character	0xssss (2 bytes)	SYNC character: should be programmed with the sync pattern.
Syn Length	8 16	SYNL_8: should be chosen to implement mono-sync protocol. The receiver synchronizes on an 8-bit sync pattern in **sync**. SYNL_16: The receiver synchronizes on a 16-bit sync pattern stored in **sync**.
Enable flags/syncs		Send either idles or flags/syncs between frames as defined by the protocol.The flag character is equal to sync.

5.7 SMC UART Mode

This protocol may only be used with the two SMC ports : HSS5&6. The SMC Dialog window is shown in Figure 10. The settings are described in Paragraph 5.7.1.

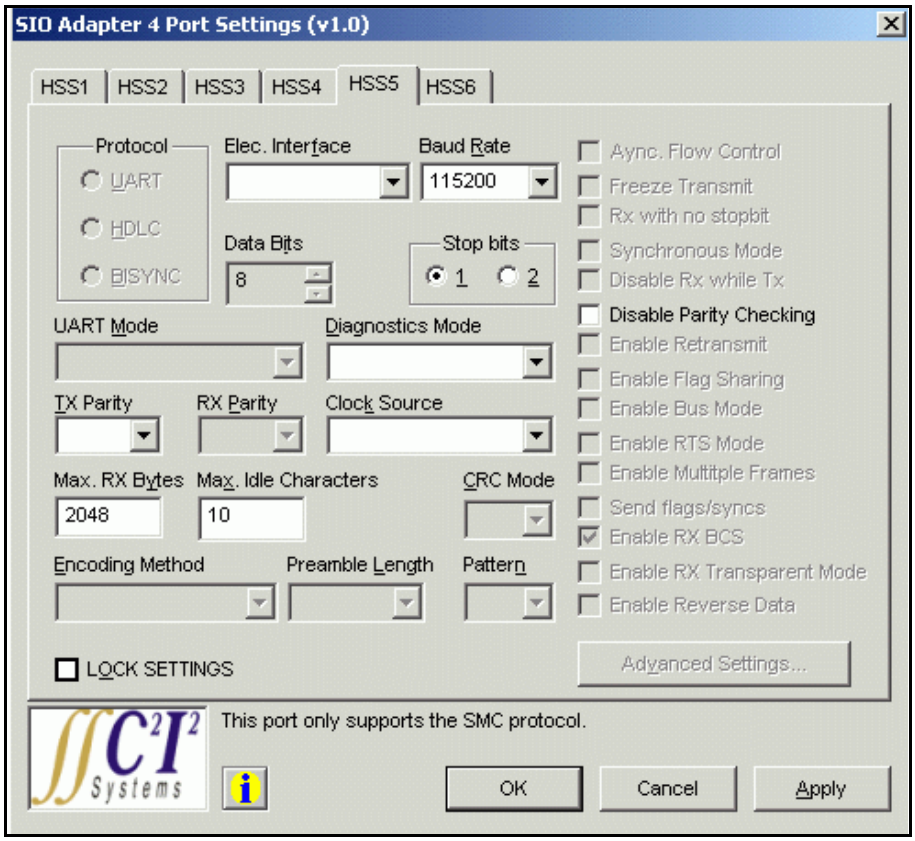

Figure # 10 : SMC Dialog

5.7.1 SMC UART Protocol Information Members

Name	Options		Description
Baud Rate	1200 - 115kbps (RS232/RS422/RS485) Any values permissible.		This member is used to specify a single baudrate for both transmitter and receiver.
Clock Source	CLOCK_DEFAULT		CLOCK_DEFAULT connects BRG[1-2] to Port[5-6]. Transmit & receive clocks can be set to one of CLOCK_BRG[1-4] or CLOCK_EXT[1-4].
	CLOCK_BRG1 CLOCK_BRG2 CLOCK_BRG3 CLOCK_BRG4	Baudrate Generators [1-4].	
	CLOCK_EXT1 CLOCK_EXT2 CLOCK_EXT3 CLOCK_EXT4	External Clocks connected on Pins: RXCLK[1-4] (RS232) or CLKIN[1-4] (RS485/RS422). **Note:** CLOCK_EXT[1-2] can only be used for SMC Port 5, while CLOCK_EXT[3-4] can only be used for SMC Port 6.	
Stop bits	ONE TWO		Number of full stop bits.

Data Bits	5 6 7 8 9 10 11 12 13 14		Number of data bits. Note only ports 5 & 6 (i.e. the SMC ports) can select 9 or more data bits.
Disable Parity Checking			Enable or disable parity checking.
TX Parity	ODD EVEN		Receive and transmit parity. Parity will only be checked if parity is enabled.
Diagnostics Mode	NORMAL	Normal operation. Use this for **external loopback**.	Set diagnostic mode. **External loopback - RS485:** connect TXD+ to RXD+ & TXD- to RXD-. **External loopback - RS232:** connect TXD to RXD.
	LOOPBACK	**Internal loopback:** TXD & RXD are connected internally. The value on RXD is ignored.	
	ECHO	The transmitter automatically resends received data bit-by-bit.	
	LOOPBACK_ECHO	Loopback and echo operation occur simultaneously.	
Max. RX Bytes	1 to 2048 (default)		Maximum number of bytes that may be copied into a buffer.
Max. Idle Characters	0 to 2048 (default)		Maximum idle characters. When a character is received, the receiver begins counting idle characters. If max_idl idle characters are received before the next data character, an idle timeout occurs and the buffer is closed. Thus, max_idl offers a way to demarcate frames. To disable the feature, clear max_idl. The bit length of an idle character is calculated as follows: 1 + data length (5-14) + 1 (if parity is used) + number of stop bits (1-2). For 8 data bits, no parity, and 1 stop bit, the character length is 10 bits.

6. Getting Started

This paragraph contains example code extracts for using the Win32 API to access the HSS device.

6.1 Normal Write Operation

```
HANDLE h_device;
DCB    dcb;
char   tx_buffer[100];
DWORD  bytes_send;

h_device = CreateFile("\\\\.\\HSS1",
                      GENERIC_READ | GENERIC_WRITE,
                      0,
                      NULL,
                      OPEN_EXISTING,
                      FILE_ATTRIBUTE_NORMAL,
                      NULL);

GetCommState(h_device, &dcb);

dcb.ByteSize     = 8;
dcb.Parity       = NOPARITY;
dcb.StopBits     = ONESTOPBIT;
dcb.BaudRate     = 115200;
dcb.fOutxCtsFlow = FALSE;

SetCommState(h_device, &dcb);
memset(tx_buffer, '*', sizeof(tx_buffer));

WriteFile(h_device, tx_buffer, sizeof(tx_buffer), &bytes_send, NULL);

CloseHandle(h_device);
```

6.2 Overlapped Read Operation

```
char       rx_buffer[100];
DWORD      bytes_read;
OVERLAPPED overlap;
DWORD      wait_event;
HANDLE     h_device;

h_device = CreateFile("\\\\.\\HSS1",
                      GENERIC_READ | GENERIC_WRITE,
                      0,
                      NULL,
                      OPEN_EXISTING,
                      FILE_ATTRIBUTE_NORMAL | FILE_FLAG_OVERLAPPED,
                      NULL);

overlap.Offset     = 0;
overlap.OffsetHigh = 0;
overlap.hEvent     = CreateEvent(NULL, FALSE, FALSE,  NULL);

ReadFile(h_device, rx_buffer, sizeof(rx_buffer), &bytes_read, &overlap);
wait_event = WaitForSingleObject(overlap.hEvent, INFINITE);
if (WAIT_OBJECT_0 == wait_event)
{
   printf("Data received.\n");
}
CloseHandle(overlap.hEvent);
CloseHandle(h_device);
```

7. Contact Details

7.1 Contact Person

Direct all correspondence and / or support queries to the Project Manager (HSS) at C²I² Systems.

7.2 Physical Address

CCII Systems (Pty) Ltd
Unit 3 Rosmead Place, 67 Rosmead Avenue
Kenilworth, 7708
Cape Town
South Africa

7.3 Postal Address

CCII Systems (Pty) Ltd
P.O. Box 171
Rondebosch
7701
South Africa

7.4 Voice and Electronic Contacts

Tel: (+27) (0)21 683 5490
Fax: (+27) (0)21 683 5435
Email: info@ccii.co.za
Email: support@ccii.co.za
URL: http://www.ccii.co.za/

7.5 Product Support

Support on C²I² Systems' products is available telephonically between Monday and Friday from 09:00 to 17:00 CAT. Central African Time (CAT = GMT + 2).

Email support is available at support@ccii.co.za