7I69 REMOTE 48 BIT TTL I/O MODULE

V1.14

Table of Contents

GENERAL 1

DESCRIPTION 1

HARDWARE CONFIGURATION 2

GENERAL 2
SETUP/OPERATE MODE 2
HARDWARE PROCESS DATA MODE 2
SERIAL PORT TERMINATION 3
RS-422 CABLE POWER OPTION 3

CONNECTORS 4

7I69 CONNECTOR LOCATIONS AND DEFAULT JUMPER POSITIONS 4
I/O CONNECTORS 5
P2 CONNECTOR PINOUT 5
P3 CONNECTOR PINOUT 6
SERIAL PORT PINOUT 7
RS-422 CABLE POWER 7
5V POWER CONNECTOR 7

OPERATION 8
HOST INTERFACE 8
HOSTMOT2 7I69 INTERFACE 8
ANYTHING I/O INTERFACE DAUGHTER CARD 8
I/O CHARACTERISTICS 8
ALL I/O IS ACTIVE LOW 8
USING I/O PINS AS INPUTS 9
WATCHDOG AND FAULTS 9
STATUS LEDS 9
PARAMETERS 9
NON-VOLATILE PARAMETERS 10
OPERATE MODE BAUD RATE 10
WATCHDOG TIMEOUT 10
RPD, WPD, AND UFLBP 11
HOST PC SERIAL ADAPTER 12
MINIMAL HOST PC SERIAL ADAPTER 12
SOFTWARE PROCESS DATA MODES 13

Table of Contents

REFERENCE INFORMATION 14
SSLBP 14
GENERAL 14
REGISTER MAP 14
PROCESS INTERFACE REGISTERS 14
COMMAND REGISTER 15
COMMAND REGISTER WRITE IGNORE 15
DATA REGISTER 16
LOCAL READ OPERATIONS 16
LOCAL WRITE OPERATIONS 16
LOCAL PARAMETERS 17
NORMAL START 18
STOP ALL 19
STOP INDIVIDUAL CHANNELS 19
DOIT 19
PER CHANNEL INTERFACE DATA REGISTERS 20
PER CHANNEL CONTROL AND STATUS REGISTERS 20
REMOTE MODES 20
INTERFACE AND CS REGISTER CONTENTS AT START 20
CS REGISTER AFTER START 21
CS REGISTER AFTER DOIT 22
PROCESS DATA DISCOVERY 23
PROCESS TABLE OF CONTENTS 23
PROCESS DATA DESCRIPTOR 24
PROCESS DATA DESCRIPTOR FIELDS 24
RECORD_TYPE 24
DATA_LENGTH 24
DATA_TYPE 25
DATA_DIRECTION 25
PARAMETER_MIN 25
PARAMETER_MAX 25
UNIT_STRING 26
NAME_STRING 26
NUMERIC PROCESS DATA SCALING 26
MODE DESCRIPTOR 26
MODE TYPES 26
PROCESS ELEMENT PACKING AND UNPACKING 27
7I69 SPECIFIC PROCESS DATA EXAMPLE 28

Table of Contents

REFERENCE INFORMATION 30
SSLBP 30
NORMAL MODE OPERATION 30
SETUP START 31
SETUP MODE OPERATION 31
REMOTE READ EXAMPLE 31
REMOTE WRITE EXAMPLE 32
DISCOVERY SEQUENCE 33

LBP 36
LBP DATA READ/WRITEWCOMMAND 36
EXAMPLE COMMANDS 37
LOCAL LBP COMMANDS 38
LOCAL LBP READ COMMANDS 38
LOCAL LBP WRITE COMMANDS 40
RPC COMMANDS 40
EXAMPLE RPC COMMAND LIST 42
CRC 43
FRAMING 43
SSERIAL REMOTE RPCS 44

SPECIFICATIONS 45
DRAWINGS 46

GENERAL

DESCRIPTION

The 7I69 is a remote real time digital I/O card with 48 digital I/O bits. All I/O bits are TTL compatible open collector type drivers with 15 mA sink capability per I/O pin. When output pins are set high, they may be used as inputs. Each I/O pin has a 3.3K pullup for direct connection with switches, optosensors etc. I/O connectors are 50 pin headers with I/O module rack compatible pinouts.

The pinout and drive capabilities make the 7I69 compatible with Opto22 I/O module racks. The pinout is also compatible with Mesa’s standard FPGA daughter cards allowing the 7I69 to drive up to 2 simple isolator daughter cards like the 7I37.

The RS-422 interface at 2.5M Baud is compatible with HostMot2s SSLBP smart serial interface which can support as many as 32 7I69 cards for a total of 1536 I/O bits with real time update rates up to 10 KHz and cable lengths up to 100 feet.

HARDWARE CONFIGURATION

GENERAL

Hardware setup jumper positions assume that the 7I69 card is oriented in an upright position, that is, with the I/O and power connectors towards the top of the card, away from the person doing the configuration;.

SETUP/OPERATE MODE

The 7I69 can run in setup mode or operate mode. In setup mode, the serial interface baud rate is fixed at 115.2 KBaud. In the operate mode, the baud rate is set to 2.5 Mbaud (default). Setup mode is also less critical of host interface timing and enables a normal PC serial port or USB serial adaptor to communicate with the 7I69 for setup purposes. W3 controls the setup/normal mode selection.

W3	**MODE**	**BAUD RATE**
UP	Operate mode	2.5 Mbps (default, BR can be changed)
DOWN	Setup Mode	115.2 Kbps (fixed)

HARDWARE PROCESS DATA MODE

The 7I69 has 2 different jumper selectable process data modes for I/O pin-out. These are Mesa Pinout_Mode and OPTO-22_Pinout_mode. These modes differ in the pin-out of the 50 pin connectors. The modes reverse the order of the I/O bits. This is illustrated in the connector section of this manual.

W1	**MODE**	**50 PIN HEADER PINOUT**
DOWN	Mesa_Mode	ASCENDING (default)
UP	OPTO_22_MODE	DESCENDING

Note: W2, TP1, and TP2 are currently unused

OPERATION

SERIAL PORT TERMINATION

The RS-422 serial port on the 7I69 can be terminated or un-terminated. Normally the 7I69 is the serial cable endpoint so the port must be terminated. W6 and W7 enable and disable the termination,

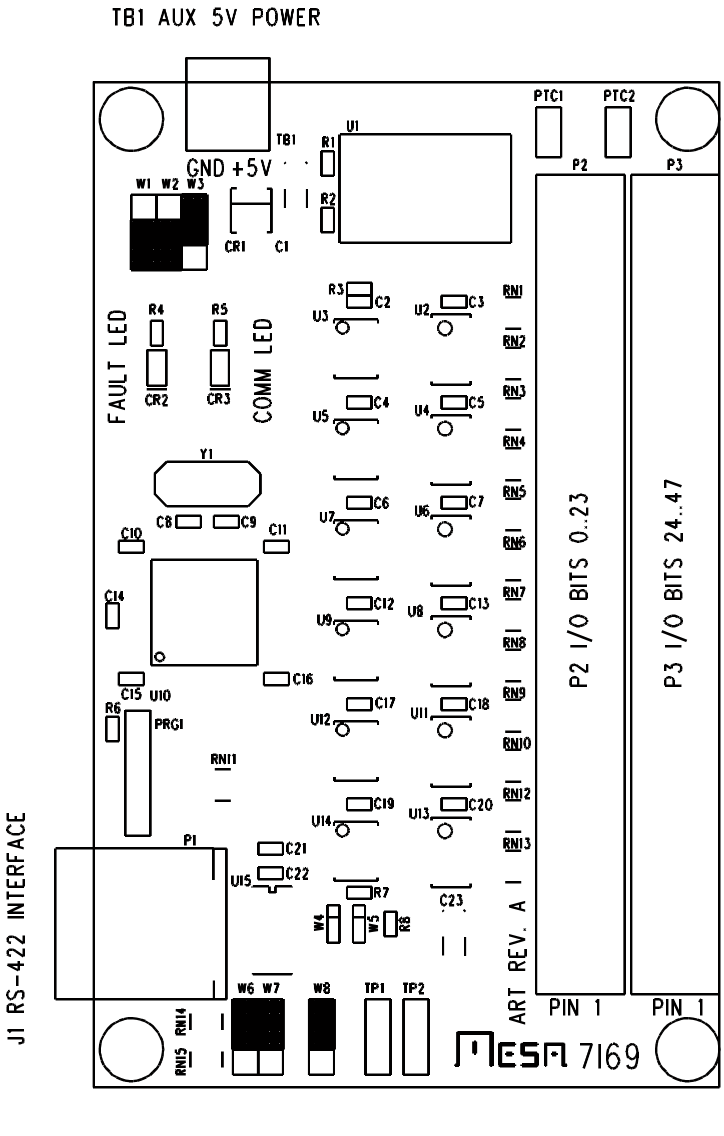
W6,W7	MODE
UP,UP	Terminated (default)
DOWN.DOWN	Unterminated

RS-422 CABLE POWER OPTION

The 7I69 can get its 5V power from its RS-422 connector P1 if desired. If cable power is selected, the 7I69 can be powered via a CAT5 interface cable. Cable power is compatible with Mesa serial daughtercards and allows the 7I69 to be powerd via the same cable that provide the serial interface. W8 controls the cable power option. When W8 is in the up position, cable power is enabled, that is 5V power is sourced from P1 pins 7 and 8. When W8 is in the down position, P1 pins 7 and 8 are unconnected and 7I69 5V power must be supplied via TB1.

CONNECTORS

7I69 CONNECTOR LOCATIONS AND DEFAULT JUMPER POSITIONS



CONNECTORS

IO CONNECTORS

I/O connectors P2 and P3 are 50 pin headers with I/O module rack pinouts. IO connector P2 has I/O bits 0 through 23 and connector P3 has I/O bits 24 through 47. I/O bit sequence depends on 7I69 operating mode. Both Mesa mode and OPTO22 mode pinouts are shown. Note that the modes simply reverse the numbering of the per connector I/O bits.

P2 CONNECTOR PINOUT

P2 PIN	MESA	OPTO22	P2 PIN	MESA	OPTO22
1	I/O0	I/O23	25	I/O12	I/O11
3	I/O1	I/O22	27	I/O13	I/O10
5	I/O2	I/O21	29	I/O14	I/O9
7	I/O3	I/O20	31	I/O15	I/O8
9	I/O4	I/O19	33	I/O16	I/O7
11	I/O5	I/O18	35	I/O17	I/O6
13	I/O6	I/O17	37	I/O18	I/O5
I15	I/O7	I/O16	39	I/O19	I/O4
17	I/O8	I/O15	41	I/O20	I/O3
19	I/O9	I/O14	43	I/O21	I/O2
21	I/O10	I/O13	45	I/O22	I/O1
23	I/O11	I/O12	47	I/O23	I/O0
			49	5V power	

Note: all even pins are grounded

CONNECTORS

P3 CONNECTOR PINOUT

P3 PIN	MESA	OPTO22	P3 PIN	MESA	OPTO22
1	I/O24	I/O47	25	I/O36	I/O35
3	I/O25	I/O46	27	I/O37	I/O34
5	I/O26	I/O45	29	I/O38	I/O33
7	I/O27	I/O44	31	I/O39	I/O32
9	I/O28	I/O43	33	I/O40	I/O31
11	I/O29	I/O42	35	I/O41	I/O30
13	I/O30	I/O41	37	I/O42	I/O29
I15	I/O31	I/O40	39	I/O43	I/O28
17	I/O32	I/O39	41	I/O44	I/O27
19	I/O33	I/O38	43	I/O45	I/O26
21	I/O34	I/O37	45	I/O46	I/O25
23	I/O35	I/O36	47	I/O47	I/O24
			49	5V power	

Note: all even P3 pins are grounded

CONNECTORS

SERIAL PORT PINOUT

J1 is the 7I69s serial interface. J1 is a RJ-45 jack. The serial interface pinout is compatible with standard 8 wire CAT5 Ethernet cables. J1 pinout is as follows:

1 RXA

2 RXB

3 TXA

4 GND

5 GND

6 TXB

7 +5V

8 +5V

J1s pinout is designed to match breakout cards like the 7I44 and 7I74. A standard CAT5 or CAT5E cable can be used to connect the 7I69 to a 7I44/7I74. CAT5E cable is suggested if the serial cable is used for powering the 7I69, as the larger wire size result in lower voltage drop.

RS422 CABLE POWER

Normally the 7I69 gets its power from the serial cable (W8 up). Daughtercards normally condition the 7I69's RS-422 signals for the FPGA controller and supply power to the 7I69. Depending on external 5V load and CAT5/CAT5E cable length it may be necessary to supply 5V power the 7I69 directly at the 7I69 card.

5V POWER CONNECTOR

If the serial cable power supply results in too much voltage drop, 5V power can be supplied directly to the 7I69 via terminal block TB1. +5 and ground are marked on board.

1 5V (SQUARE PAD)

2 GND

If 5V power is suppled to TB1, W8 should be in the down position.

OPERATION

HOST INTERFACE

HOSTMOT2 7I69 INTERFACE

The Hostmot2 interface to the 7I69 is a smart serial interface for Mesa's Anything I/O series of FPGA cards that encapsulates the LBP serial protocol details and presents a simple parallel register set to the host computer. Interface registers for input data, output data and communication status are provided for all connected 7I69 cards.

The 7I69 Hostmot2 interface is a SSerial module with specific firmware (SSLBP) for 7I69 card or other LBP interfaced cards. Each SSerial module can support up to eight 7I69 cards. Up to four sserial modules can be used in a single FPGA configuration. The sserial module supports the standard LBP 2.5 M Baud communication rate and process data update rate to 10 KHz. With the default configuration, the Hostmot2 interface sends 48 bits of output process data to the 7I69 and reads 48 bits of input process data from the 7I69 for each transfer request.

ANYTHING I/O INTERFACE DAUGHTER CARDS

7I69 compatible daughter cards are available to simplify connecting the 7I69 to Mesa's Anything I/O FPGA cards. Two RJ45 compatible RS-422 daughter cards are available, the 7I44 and the 7I74. The 7I44 and 7I74 provide 8 channels of RS-422/RS-485 serial communication interfaces. The 7I44 and 7I74 use RJ-45 connectors for the serial interface. These connectors are compatible with the 7I69 so a common CAT5 or CAT5E cable may be used to connect from the 7I44 or 7I74 to the 7I69. The 7I44 and 7I74 can also provide 5V power to the 7I69 subject to the CAT5/5E cable length restrictions.

Other compatible daughtercards are the 7I34, 7I47, 7I47S. 7I52, 7I53, 7I76, 7I77, 7I78. Other than the 7I34 all these daughtercards have screw terminal blocks. Mesa can supply matching serial cables for these daughter cards if desired.

I/O CHARACTERISTICS

ALL I/O IS ACTIVE LOW

All TTL I/O bits are active low, that is, a low output state is regarded as the 'on' state. This is for compatibility with I/O module racks and for OPTO-coupler/SSR (Solid State Relay) driving by sinking current.

Input and outputs bits are presented at the host interface in inverted form (active high) so a low input is reported as a '1' bit, and a '1' output register bit drives the corresponding TTL I/O bit on (low).

Because of the active low design, all I/O pins will be HIGH at power up, or after a watchdog bite. If SSRs are driven, they should always be driven ON by a low output. This means that the SSR+ input should connect to the 7I69s 5V power (pin 49 on the 50 pin connector) and the SSRs -input should connect to the 7I69 I/O pin.

OPERATION

I/O CHARACTERISTICS

USING I/O PINS AS INPUTS

All I/O pin may be used as input or output. To be able to use and I/O bit as an input the corresponding output bit must be off. This means the output must be 0 at the host interface (this means high = open drain with pullups at the I/O pins)

WATCHDOG AND FAULTS

The 7I69 has a watchdog timer that will set all I/O bits to the OFF state and set a fault flag if host communication does not occur at a minimum rate. Default watchdog time is 50 mS which means if not accessed at a greater than 20 Hz rate, the watchdog will bite and disable the outputs.

When a fault flag is set, outputs can not longer be set and the host must first clear the fault before normal operation can continue. This is also the 7I69s startup condition, meaning the host must first clear the fault before starting normal operation. This is normally handled by SSLBP.

LEDS

The 7I69 has two status LEDs, CR2 and CR3. CR2 (red) is a fault indicator. It will be illuminated in case of a watchdog fault (loss of communication). At startup, CR2 will be illuminated until the fault is cleared and communications established. CR3 (green) is the serial interface activity indicator. At slow communication rates, CR3 will blink at each serial transaction. When high speed communications are established, CR3 will blink at the packet rate/1024 or about 1Hz for a 1 KHz update rate.

PARAMETERS

The 7I69 has several user settable parameters, but normally only a very few need be changed in normal operation.

PARAMETER	TYPE	FUNCTION
NVBAUDRATE	UINT	Sets operate mode baudrate
NVUNITNUMBER	ULONG	Non-volatile unit number
UNITNUMBER	ULONG	Working unit number
NVWATCHDOGTIME	UINT	Non-volatile watchdog time in mS
WATCHDOGTIME	UINT	Working watchdog time in ms

OPERATION

PARAMETERS

INPUT	UDOUBLE	48 bits of input data (right justified)
OUTPUT	UDOUBLE	48 bits of output data (right justified)
FAULT	UINT	7I69 fault register
STATUS	UINT	7I69 status register

NON-VOLATILE PARAMETERS

All non volatile parameters start with the letters NV. Non-volatile parameters are stored permanently in the processors EEPROM and are copied to the volatile working parameters at power-up. Because of this, non-volatile parameters only take affect after a 7I69 power cycle.

OPERATE MODE BAUD RATE

The operate mode baud rate default is 2.5 MBaud. This should not be changed unless needed for non-standard applications. Baud rates are selected by writing an index value to the NVBAUDRATE parameter. The index numbers for available baud rates are as follows:

INDEX	**BAUD**	**INDEX**	**BAUD**	**INDEX**	**BAUD**
0	9600B	1	19200B	2	38400B
3	57600B	4	115200B	5	230400B
6	460800B	7	921600B	8	1.25MB
9	2.5MB*	10	5MB	11	10MB

WATCHDOG TIMEOUT

The default watchdog period is 50 mS but can be set to different periods to suit the application. Watchdog timeout units are mS. A watchdog timeout value of 0 will disable the watchdog. The watch dog is a safety feature that warns that the 7I69 input data has not been read and should normally not be disabled nor set to long timeout periods. The non-volatile watchdog timeout is set via the NVWATCHDOGTIMEOUT parameter. The working watchdog timeout is set with the WATCHDOGTIME parameter.

OPERATION

RPD, WPD, AND UFLBP

The RPD, WPD, and UFLBP are command line utilities allow reading and writing volatile and non-volatile 7I69 parameters, and updating the firmware on the 7I69 To use these utilities on most operating systems, the 7I69 must be in the setup mode or the operate mode baud rate must be 115200 KBaud or less

RPD, WPD, and UFLBP need environment variables preset before they will work. For Windows and 115200 baud, the following environment variables should be set:

SET BAUDRATE=115200

SET BAUDRATEMUL=1

SET PROTOCOL=LBP

SET INTERFACE=OSDEVICE

Example setting NVWATCHDOGTIMEOUT to 100 ms:

WPD NVWATCHDOGTIME 100

Note this is permanent change in the 7I69s watchdog timeout and like all non-volatile parameters, will only be applied after the 7I69 has been power cycled

Example reading 7I69 faults in Hexadecimal:

RPD FAULT H

Example of temporarily disabling watchdog and the setting every other output on:

WPD WATCHDOGTIME 0

WPD OUTPUT AAAAAAAAAAAA H

Example of updating 7I69 firmware with UFLBP

UFLBP 7I69.BIN

Note the 7I69 MUST be in setup mode for UFLBP to work properly.

OPERATION

HOST PC ADAPTER

In order to run any of the command line utilities a RS-422 adapter is needed. Mesa can provide a suitable adapter. Two such adapters are 3I21 or 3I22. These adapters connects the RJ-45 RS-422 interface on the 7I69 to a DB9 serial port (3I21) or USB port (3I22) and provide 5V link power.

MINIMAL HOST PC ADAPTER

A simple home made host adapter can be made by directly connecting RS-232 signals from a 9 pin PC serial port or USB RS-232 adapter to the 7I69s RS-422 signals via a one ended CAT5 cable. A single resistor between RS-232 TXD and RS-422 RXB is needed to prevent overloading the RS-232 TXD output

CAT5 PIN	DE-9F PIN	CAT5 SIGNAL	DE-9F SIGNAL	CAT5 COLOR
1	5	RXA	GND	ORANGE WHITE
2	3	RXB (1)	TXD (1)	ORANGE
3	XX	TXA	XX	GREEN WHITE
4	5	GND	GND	BLUE
5	5	GND	GND	BLUE WHITE
6	2	TXB	RXD	GREEN
7	XX	+5V (2)	XX	BROWN WHITE
8	XX	+5V (2)	XX	BROWN

Notes:

1. Connect via 470 Ohm 1/4 watt resistor. All other signals directly connected

2. +5V power must be supplied for the 7I69s serial link

OPERATION

SOFTWARE PROCESS DATA MODES

The 7I69 has five software selectable process data modes. These different modes select different sets of 7I69 data to be transferred between the host and the 7I69 during real time process data exchanges. For high speed applications, choosing the correct mode can reduce the data transfer sizes, resulting in higher maximum update rates.

MODE 0	Bidirectional mode (48 bits in 48 bits out)
MODE 1	Input only mode (48 bits in)
MODE 2	Output only mode (48 bits out)
MODE 3	24/24mode (24 bits in = bits 0..23 and 24 bits out = bits 24..47)
MODE 4	Bidirectional mode (48 bits in 48 bits out) plus 4 MPG encoder channels on inputs 0 through 7

All software modes can work in either hardware mode (Opto22 or Mesa Pin-out) for a total of 10 possible operational modes.

REFERENCE INFORMATION

Note that the following interface details presented here are not normally needed for users, as all register level interface details are handed by the driver code. This information is presented here for use by interface and driver developers.

SSLBP

GENERAL

SSLBP is a firmware option to HostMot2s SSERIAL serial interface that allows simple communication to LBP based peripherals like the 7I69. SSERIAL is a part of the HostMot2 motion interface firmware for MESA's Anything-I/O FPGA cards.

REGISTER MAP

SSLBP has two global processor interface registers and four per channel remote device interface registers. For more details on mapping of these registers in HostMot2 memory space, see the REGMAP file that is included with the HostMot2 source distribution.

PROCESSOR INTERFACE REGISTERS

There are two processor interface registers, the COMMAND register and the DATA register. These registers allow low level communication to SSLBP's interface processor for issuing global commands, discovery, and debug operations.

REFERENCE INFORMATION

SSLBP

COMMAND REGISTER

The commands register is a 16 bit register (right justified in the 32 bit interface) with the following format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
W	M	R	D	S	T	T	T	N	N	N	N	N	N	N	N

W = BIT 15 Write bit, set high for control data write commands

M = BIT 14 ROM enable/ reset bit, set high to reset processor / download ROM

R = BIT 13 Request bit, set high for read or write command

D = BIT 12 DoIt bit, set high for DoIt commands

S = BIT 11 Start/Stop bit, actual operation depends on T:

ST = 1,0,0,0 Stop LBP interface = 0x08NN

ST = 1,0,0,1 Start LBP interface in normal mode = 0x09NN

ST = 1,1,1,1 Start LBP interface in setup mode = 0x0FNN

N bits determine which channels start or do data transfer with remote device. A set bit indicates that the corresponding channel will start or do a data transfer.

A command is started when written to the command register. Command completion is signaled by the command register being cleared (to 0x0000) by the internal SSLBP firmware. If the command register is read before the command is complete, it will reflect the previously written command. *The command register should not be written when non-zero or unpredictable behavior may result.* There are two exceptions to this rule:

1. A STOP ALL command can always be written to reset the SSLBP interface.

2. Command writes with the ignore bit set can always be written (see below)

COMMAND REGISTER WRITE IGNORE

The command register has a feature that any command written with the MSB (bit 31) set will be ignored. This is for compatibility with DMA driven interfaces or any interfaces that use a fixed address list for low level hardware access so cannot skip writes

REFERENCE INFORMATION

SSLBP

DATA REGISTER

SSLBP has a global 8 bit data register for debug and custom setup purposes. This register allows access to internal SSLBP parameters. The data register is right justified in the 32 bit Hostmot2 register.

LOCAL READ OPERATIONS

The sequence used for reading a local SSLBP variable is as follows:

1. The parameter address ORed with the Request bit (bit 13) is written to the command register.

2. The host polls the command register until it reads as zero.

3. The host reads the parameter byte from the data register

LOCAL WRITE OPERATIONS

The sequence used for writing a local SSLBP variable is as follows:

1. The host polls the command register until it reads as zero.

2. The host writes the data byte to the data register

3. The host writes the command register with the the parameter address Ored with both the Request bit (bit 13) and the Write bit (bit 15)

REFERENCE INFORMATION

SSLBP

LOCAL PARAMETERS

There are a number of local SSLBP read only parameters that are useful for interface software and drivers to access using the local read operations:

LOCAL PARAMETER	ADDRESS	DESCRIPTION
INTERFACE_TYPE	0x0000	0x12 for SSLBP
INTERFACE_WIDTH	0x0001	Data port width (8)
MAJORREV	0x0002	Major SSLBP firmware revision
MINORREV	0x0003	Minor SSLBP firmware revision
GP_INPUTS	0x0004	Number of GP input bits (0 for SSLBP)
GP_OUTPUTS	0x0005	Number of GP output bits (0 for SSLBP)
PROCESSOR_TYPE	0x0006	0xD8 for Dumb8
CHANNELS	0x0007	1 to 8 depending on configuration

REFERENCE INFORMATION

SSLBP

NORMAL START

When the FPGA is first configured or after a STOP command, all local communication, error and status parameters are initialized and all LBP communication channels are idle. A normal START command begins to establish communications with all remote LBP devices. A normal start command is issued by writing a Start bit with type bits of 0,0,1 with a bit mask of the desired channels to start in the low byte, This is 0x9NN hex where NN is the bitmask of channels to start. This command is written to the command register to start the selected channels.

Once a start command has been issued, all channels that are selected in the bit mask will be probed to determine if a LBP device exists. If a device exists on a channel, the SSLBP firmware will acquire the device name, and device unit number, and pointers to process data information from the remote device..

A normal start command also does a standard set of remote device setup operations when it detects a remote device. This setup includes clearing any faults, setting remote operational mode, and setting the outputs off. If no errors have occurred and all faults are clearable, the SSLBP firmware enters a "chatter" loop where it repeatedly sends output data of all 0's. This keeps the remote devices watchdog fed while waiting for the first DOIT command.

When the command completes (the command register is clear), the data register can be read to determine if all selected channels have started. A 1 bit in any position in the data register indicates that the corresponding channel has failed to start. If a channel has failed to start, more information about the failure can be determined by reading the CS register of the failed channel.

Once a DOIT command has been executed, the firmware no longer "chatters" and it becomes the responsibility of the host interface to continue sending DOIT commands at a rate sufficient to feed the remote devices watchdog (faster than 20 Hz with the default 50 mS watchdog timeout period). If this is not done, the remote device’s watchdog will bite, disabling its outputs and setting the fault flag. This will require a channel stop followed by a channel start to resume normal operations.

REFERENCE INFORMATION

SSLBP

STOP ALL

A STOPALL command is issued to stop all channel communication. *STOPALL resets all channel variables and should always be issued by a driver when initializing the SSLBP interface.* A STOPALL followed by a START command can be used after a fault condition to re-establish communication with the remote LBP devices. Device discovery is only done once when START command is issued to a STOPed SSLBP. This means that if cabling, devices, or device hardware modes are are changed, a STOPALL command followed by a START command must be issued by the host to detect the changes. A STOPALL command is 0x0800.

STOP INDIVIDUAL CHANNELS

In addition to stopping all channels, a individual stop command can be issued. A individual stop command include a bitmask of the channels to stop in the least significant 8 bits of the command (the N bits), that is a stop channel 1 command would be 0x802. The intended use of individual stop is per channel error recovery. It should not be used for normal interface startup as it does not reset channel variables, that is a 0x8FF command (stop all individual channels) is not equivalent to a 0X800 (STOPALL) command.

DOIT

In normal operation SSLBP is designed to send host data from the interface registers to the remote device and request data from the remote device for presentation in the interface registers to the host. This SSLBP function is designed for high speed real time operation. Synchronization with the host is accomplished with the DOIT command.

When the host writes a DOIT command,, all outgoing process data from the host is sent to the remote devices and incoming process data is requested. Completion of the DOIT command is signaled by SSLBP clearing the COMMAND register. A DOIT command is completed when al requested channel transfers have completed or timed out. After the completion of a successful DOIT command, the incoming process data from the remote can be read.

A DOIT command contains the DOIT bit and an 8 bit mask in the 8 LSBs that selects the channels that will be requested to transfer data. *A DOIT should not be requested on an inactive channel, that is a channel that did not start*. After DOIT command completion the data register will contain a bit mask of channel status data. If any bit is set in the data register, it indicates a problem with the transfer (all zeros indicates no faults or errors).

The data register contents returned after a DOIT command can be used to minimize host access cycles by avoiding the need to read the per channel status registers. If detailed fault information is desired, the CS register can be read on any channel that shows a failed transfer.

REFERENCE INFORMATION

SSLBP

PER CHANNEL INTERFACE DATA REGISTERS

SSLBP supports three 32 bit interface data registers per channel. These are called interface register 0, interface register 1, and interface register 2. These are read/write registers with independent incoming and outgoing data. These registers are used for both setup/discovery data when starting a data link and process data once the link is running. When a start command is issued and has successfully completed, per channel setup data will be available in the interface registers.

PER CHANNEL CONTROL AND STATUS REGISTERS

SSLBP has a 32 bit control and status register for each channel. Like the interface data registers, these registers are used both for data link startup information and for status when the link is in operation.

REMOTE MODES

Some remote devices have software selectable modes that determine the specific data transferred for each DOIT command. These modes are selected by writing the mode number to the most significant byte of the remote channels CSR before a START or SETUP START command is issued. A default value of 0x00000000 should be written to all CSRs if MODE is not used.

REMOTE MODE IS WRITTEN TO CSR MS BYTE BEFORE START

CS REG	MODE	0	0	0.

INTERFACE AND CS REGISTER DATA AT START

After a successful start command (either setup start or normal start), Interface register 0 reports the remote device's unit number. This is the number printed on the card label. Interface register 1 reports the remote device's 4 letter name (LSB first). Interface register 2 reports the remote devices global table of contents pointer (GTOCP) and process table of contents pointer (PTOCP) for the currently selected remote device mode. The GTOCP and PTOCP will be 0x0000 for devices that do not support process data discovery. *Note that the setup data will be overwritten with process data once the first DOIT command is issued.*

READ DATA FROM PER CHANNEL INTERFACE REGISTERS AFTER START

CS REG	X	COM_STATE	STATUS	LOCAL FLT.
INTERFACE 0	UNIT# BYTE 3	UNIT# BYTE 2	UNIT# BYTE 1	UNIT# BYTE 0
INTERFACE 1	NAME BYTE 3	NAME BYTE 2	NAME BYTE 1	NAME BYTE 0
INTERFACE 2	GTOCP BYTE1	GTOCP BYTE 0	PTOCP BYTE1	PTOCP BYTE 0

REFERENCE INFORMATION

SSLBP

CS REGISTER AFTER START

The CS register is used for local SSLBP, and remote LBP device status and control information. Read access returns status information in both normal and setup mode. In normal mode, writes to the CS register are not used. After a normal start or setup start the CS register has the following format:

Byte3 = X undefined for SSLBP versions < 29, remote fault for versions >28 (See CS REGISTER AFTER DOIT section)

Byte2 = COM_STATE Communication state code (debug only)

Byte1 = Communication status code (0x00 for OK)

Bit 7 = CommunicationNotReady

Bit 6 = NoRemoteID

Bit 5 = CommunicationError

Bit 0 = RemoteFault

Byte0 = Local Communication faults (sticky, cleared only by STOP)

Bit 7 = TooManyerrors

Bit 6 = RemoteFault

Bit 5 = SerialBreakError

Bit 4 = ExtraCharacterError

Bit 3 = TimeoutError

Bit 2 = OverrunError

Bit 1 = InvalidCookieError

Bit 0 = CRCError

REFERENCE INFORMATION

SSLBP

CS REGISTER AFTER DOIT

After a successful DOIT command, or normal start with SSLBP versions >28 bytes 0 through 2 of CS register are the same as after a start command but in addition, the previously invalid byte 3 of the CS register contains remote fault information:

Byte3 = REMOTE_FAULTS

Bit 7 = LBPCOMFault

Bit 6 = IllegalMode Fault

Bit 5 = LowVoltageFault

Bit 4 = HighVoltageFault

Bit 3 = OverCurrentFault

Bit 2 = OverTempFault

Bit 1 = NoEnableFault

Bit 0 = WatchdogFault

REFERENCE INFORMATION

SSLBP

PROCESS DATA DISCOVERY

The SSLBP interface provides information to allow the host to determine the name, number, units, sizes, types, directions, and scaling of process data elements. This information is read from the remote device via a setup mode start followed by a series of remote read operations.

Note to the bewildered: process data discovery and its complications are not needed to access the 7I69 via SSLBP. In fact the 7I69's data can be accessed via SSLBP with no more than a few register reads and writes The sole purpose of process data discovery is to allow the driver to present nicely named and formatted data to the host without the driver having any built in knowledge of the remote device.

PROCESS TABLE OF CONTENTS

After a normal start or setup start command, the PTOCP word in the low word of interface register 2 is a pointer to the current process table of contents (PTOC) in the remote device.

If remote devices that do not support process device discovery are present, their PTOCP will be 0, and process data organization must be inferred from the remote device name.

Remote reads from this location will return the first entry in the PTOC. All PTOC entries are pointers with a size of 2 bytes. The end of the PTOC is marked with a 0 sentinel. Each PTOC entry points to a process data descriptor. Here is an example of a 5 entry PTOC (PDD is Process Data Descriptor)

ENTRY	ADDRESS	CONTENTS
0	PTOCP	POINTER TO PDD 0
1	PTOCP+2	POINTER TO PDD 1
2	PTOCP+4	POINTER TO PDD 2
3	PTOCP+6	POINTER TO PDD 3
4	PTOCP+8	POINTER TO PDD 4
5	PTOCP+10	0x0000 (END OF TABLE)

REFERENCE INFORMATION

SSLBP

PROCESS DATA DESCRIPTOR

Each PTOC entry points to a process data descriptor or a mode descriptor. Each process data descriptor is a record with fields for data size, data type, data direction, minimum and maximum values, the address of the process data and the unit name and process data name. Each process data element has a corresponding process data descriptor record. In addition there are mode descriptor records that indicate the current hardware and software modes of the remote device. The process data descriptor record structure is as follows:

FIELD NAME	FIELD LENGTH	DESCRIPTION
RECORD_TYPE	8 BITS	RECORD TYPE = 0xA0
DATA_SIZE	8 BITS	DATA SIZE IN BITS
DATA_TYPE	8 BITS	DATA ELEMENT TYPE
DATA_DIRECTION	8 BITS	DATA DIRECTION
PARAM_MIN	32 BITS	IEEE-754 FP PARM MIN
PARAM_MAX	32 BITS	IEEE-754 FP PARM MAX
PARAM_ADD	16 BITS	ADDRESS OF PARM
UNIT_STRING	VARIABLE	NULL TERM. STRING
NAME_STRING	VARIABLE	NULL TERM. STRING

PROCESS DATA DESCRIPTOR FIELDS

RECORD_TYPE

The RECORD_TYPE field is a single byte at the beginning of the process data descriptor for record typing and sanity checking. It is 0xA0 for process data records.

DATA_LENGTH

The DATA_LENGTH field is a single byte field that specifies the length of the process data element in bits. Minimum is 1 bit, maximum is 255 bits, however current SSLBP implementations are limited by the number of interface registers to a maximum of 96 bits.

REFERENCE INFORMATION

SSLBP

DATA_TYPE

The DATA_TYPE field is a single byte field that specifies the data type of the process data element. Data types are as follows:

NUMBER	DATA_TYPE	NOTE
0x00	PAD	To pad for byte alignment
0x01	BITS	Packed bits, LSB is BIT 0
0x02	UNSIGNED	Numeric unsigned
0x03	SIGNED	Numeric twos complement LSB first
0x04	NONVOL_UNSIGNED	Numeric unsigned
0x05	NONVOL_SIGNED	Numeric twos complement LSB first
0x06	STREAM	Continuous data stream
0x07	BOOLEAN	Any length non-zero = true

DATA_DIRECTION

The DATA_DIRECTION field is a single byte field that specifies the data direction. Valid Data direction bytes are as follows:

0x00	INPUT	(Read from remote)
0x40	BI_DIRECTIONAL	(Read from and written to remote)
0X80	OUTPUT	(Written to remote)

PARAMETER_MIN

The PARAMETER_MIN field is a 32 bit IEEE-754 floating point number that specifies the minimum value of the process data element. This is to allow the driver to present data in engineering units. Not valid for non-numeric data types

PARAMETER_MAX

The PARAMETER_MAX field is a 32 bit IEEE-754 floating point number that specifies the maximum value of the process data element. This is to allow the driver to present data in engineering units. Not valid for non-numeric data types.

REFERENCE INFORMATION

SSLBP

UNIT_STRING

The UNIT_STRING is a variable length null terminated string that specifies the units of the process data element

NAME_STRING

The NAME_STRING is a variable length null terminated string that begins immediately after the UNIT_STRING. It specifies the name of the process data element.

NUMERIC PROCESS DATA SCALING

Currently all numeric process data is simple unsigned or signed (twos complement) binary data. The process data element PARAM_MIN and PARAM_MAX values in conjunction with the DATA_SIZE can be used to scale this numeric data.

For unsigned data, PARAM_MIN corresponds to a value of 0 and PARAM_MAX corresponds to a value of (2 ^ DATA_SIZE) -1. Meaning scaled unsigned data is RAW_DATA*(PARAM_MAX-PARAM_MIN) / ((2 ^ DATA_SIZE) -1) +PARAM_MIN.

For signed data. PARAM_MIN corresponds the value -(2 ^ DATA_SIZE-1)-1 and PARAM_MAX corresponds the value (2 ^ DATA_SIZE-1)-1, meaning scaled signed data is RAW_DATA (PARAM_MAX-PARAM_MIN) / ((2 ^ DATA_SIZE-1) -1) +PARAM_MIN.

MODE DESCRIPTOR

In addition to the process data descriptors, the PTOC will have pointers to two mode descriptors. These are the currently selected hardware and software modes of the remote device.

FIELD NAME	FIELD LENGTH	DESCRIPTION
RECORD_TYPE	8 BITS	RECORD TYPE = 0xB0
MODE INDEX	8 BITS	WHICH MODE
MODE TYPE	8 BITS	MODE TYPE
UNUSED	8 BITS	UNUSED
MODE_NAME_STRING	VARIABLE	NULL TERM. STRING

MODE TYPES

Currently there are only two mode types, HWMODE = 0x00 and SWMODE = 0x01 these correspond to hardware (EEPROM or Jumper setting)and software (dynamically changeable operational modes)

REFERENCE INFORMATION

SSLBP

PROCESS DATA ELEMENT PACKING AND UNPACKING

Ultimately all process data is transferred to and from the host via the interface 0,1,2 registers.

The packing of outgoing process data elements into these interface registers and unpacking of incoming process data elements from these interface registers is done in the order of process data descriptors listed in the PTOC. Process data elements in PTOC order and process descriptor DATA_SIZE are packed into or unpacked from the interface registers from LSB to MSB and from interface register 0 through interface register 2.

Read data and bidirectional data is unpacked from the interface registers read by the host. Write data and bidirectional data is packed into the interface registers written by the host.

Before a DOIT command is written to start a data transfer cycle with the remote device, the host must write its packed outgoing process data (OPD in table below) to the interface registers. (The CS register not currently used for outgoing data/control so is not written)

HOST WRITES OUTGOING INTERFACE REGISTERS BEFORE DOIT

CS REG	MODE	X	X	X
INTERFACE 0	OPD BYTE 3	OPD BYTE 2	OPD BYTE 1	OPD BYTE 0
INTERFACE 1	OPD BYTE 7	OPD BYTE 6	OPD BYTE 5	OPD BYTE 4
INTERFACE 2	OPD BYTE 11	OPD BYTE 10	OPD BYTE 9	OPD BYTE 8

REFERENCE INFORMATION

SSLBP

PROCESS DATA ELEMENT PACKING AND UNPACKING

After the DOIT command has completed, the incoming process data (IPD in table below) can be read along with the local and remote faults.

HOST READS INCOMING INTERFACE REGISTERS AFTER DOIT

CS REG	REMOTE. FLT	COM_STATE	STATUS	LOCAL FLT.
INTERFACE 0	IPD BYTE 3	IPD BYTE 2	IPD BYTE 1	IPD BYTE 0
INTERFACE 1	IPD BYTE 7	IPD BYTE 6	IPD BYTE 5	IPD BYTE 4
INTERFACE 2	IPD BYTE 11	IPD BYTE 10	IPD BYTE 9	IPD BYTE 8

7I69 SPECIFIC PROCESS DATA EXAMPLE

Process data is remote device dependent and also dependent on remote device mode. The 7I69 supports 2 hardware modes and 4 software modes. The hardware and software modes can be used in any combination.

The hardware modes are the MESA and OPTO22 I/O bit order mode. These are selected by a jumper on the 7I69 card and cannot be changed dynamically.

The software modes determine whether the 7I69 card is in input/output, input only, output only or 24 in/24 out mode. In the default input/output mode the 7I69 process data consists of 48 input bits and 48 outputs bits. In input mode, process data consists of 48 input bits, in output only mode, process data consists of 48 output bits in 24in/24out mode, process data consists of 24 input bits and 24 output bits. The chief advantage of the input only output only or 24 in/24 out mode is higher possible loop rates where only 48 total bits of data are transferred each cycle rather than 96 bits.

REFERENCE INFORMATION

SSLBP

7I69 SPECIFIC PROCESS DATA EXAMPLE

In the default input/output mode the process data appears in the interface registers in the order shown:

7I69 OUTGOING PROCESS DATA FOR DEFAULT INPUT/OUTPUT MODE

CS REG	X	X	X	X
INTERFACE 0	P3 BITS 31..24	P2 BITS 23..16	P2 BITS 15..8	P2 BITS 7..0
INTERFACE 1	X	X	P3 BITS 47..40	P3 BITS 39..32
INTERFACE 2	X	X	X	X

7I69 INCOMING PROCESS DATA FOR DEFAULT INPUT/OUTPUT MODE

CS REG	REMOTE. FLT	COM_STATE	STATUS	LOCAL FLT.
INTERFACE 0	P3 BITS 31..24	P2 BITS 23..16	P2 BITS 15..8	P2 BITS 7..0
INTERFACE 1	X	X	P3 BITS 47..40	P3 BITS 39..32
INTERFACE 2	X	X	X	X

Note that this information is just for user convenience and the process data organization in the interface registers can be determined by process data discovery.

REFERENCE INFORMATION

SSLBP

NORMAL MODE OPERATION

In normal mode the sequence of operations for a cyclic access with write before read is as follows:

Note steps 1 through 5 are setup operations and are only done once per session

1. Issue STOP ALL command (0x800), wait for COMMAND register clear to verify stop command completion.

2. Issue normal START command (0x9NN) with bitmask (NN) of channels to start.

3. Wait for COMMAND register clear to verify start command completion. (may be many mS)

4. Read data register to verify that all selected channels started (a 1 in any channel position bit means a fault in the channel that the bit represents)

5. Read device unit number (This can only be read before DOIT has been asserted)

6. Check command register, if not clear, cycle time is too short.

(Note the command register should never be written to when not clear except to issue a stop command or when written with the command ignore bit set)

7. Check data register, any 1 bits indicate previous DOIT command failed for in the corresponding channels

8. Read per channel Interface register 0 and interface register 1 for input process data

9. Write per channel output process data (for 7I69) to interface 0 register and interface 1 register

10. Write DOIT command = 0x10NN where NN is the bit mask of channels to initiate transfers.

11. Wait for next cycle, at next cycle time, loop back to state 6

This sequence can be modified if a read-modify-write sequence is required, this requires polling the command register for send/receive completion. This will take a maximum of 100 uSec from the DOIT command to command register clear and valid input data.

REFERENCE INFORMATION

SETUP START

When the FPGA is first configured or after a stop all command, all LBP communication channels are idle. A SETUP START command first initializes and all local communication, error and status parameters and begins to establish communications with all remote LBP devices. Unlike the NORMAL START command, SETUP START does no device specific setup but instead creates a pass-through access mode that allows the host to read or write any remote LBP device parameter. This allows simple utilities to setup 7I69 volatile and non-volatile parameters, and allows the host to do process data discovery to determine the input and output process data information from the remote device.

SETUP MODE OPERATION

In setup mode the SSLBP interface is used as a passthrough device to allow reading and writing parameters to the remote LBP device.

REMOTE READ EXAMPLE:

For a remote word read, the sequence of operations is as follows:

1. Issue a STOPALL command (0x800), wait for COMMAND register clear to verify stop command completion.

2. Issue a setup START command (0xFNN) with bitmask (NN) of channels to start

3. Wait for COMMAND register clear to verify start command completion. (may be many mS)

4. Read data register to verify that all selected channels started (a 1 bit means a fault in the channel that the bit represents)

5. Write LBP word read command (0x45) in the MSByte ORed with the parameter address to the selected channels CS register. (0x4500PPPP)

6. Issue a DOIT Command

7. Wait for the command register to be clear

8. Check that the data register is clear, any set bits indicate an error

9. Read the returned data in the LS word of the selected channels Interface0 register

10. Repeat from step 5 for any additional remote data reads

Remote read byte, word, long and double are basically equivalent, the only difference being the LBP command (0x44.0x45,0x46,0x47 respectively) and the size of the data read from the interface register(s)

REFERENCE INFORMATION

SSLBP

REMOTE WRITE EXAMPLE:

For a remote word write, the sequence of operations is as follows:

1. Issue a STOPALL (0x800) command, wait for COMMAND register clear to verify stop command completion.

2. Issue a setup START command (0xFNN) with bitmask (NN) of channels to start

3. Wait for COMMAND register clear to verify start command completion. (may be many mS)

4. Read data register to verify that all selected channels started (a 1 bit means a fault in the channel that the bit represents)

5. Write the new parameter data to the selected channels Interface0 register (right justified)

6. Write LBP word write command (0x65) in the MSByte ORed with the parameter address to the selected channels CS register. (0x6500PPPP)

7. Issue a DOIT Command

8. Wait for the command register to be clear

9. Check that the data register is clear, any set bits indicate an error

. Repeat from step 5 for any additional remote parameter writes

Remote write byte, word, long and double are basically equivalent, the only difference being the LBP command (0x64,0x65,0x66,0x67 respectively) and the size of the data written to the interface register(s)

REFERENCE INFORMATION

SSLBP

DISCOVERY SEQUENCE:

for process data discovery (of one channel) the sequence of operations is as follows:

Note that the first section acquires the PTOC and the second section reads the records pointed to by the PTOC. For brevity, the remote read sequence (steps 5 through 9 of the remote read procedure) will be listed here as "remote read"

FIRST PART, ACQUIRE PTOC:

1. Issue a STOPALL (0x800) command, wait for COMMAND register clear to verify stop command completion.

2. Issue a setup START command (0xFNN) with bitmask (NN) of channels to start

3. Wait for COMMAND register clear to verify start command completion. (may be many mS)

4. Read data register to verify that the selected channels started (a 1 bit means a fault in the channel that the bit represents)

5. Read PTOCP from interface register 2, of selected channel, if zero, remote device does not support discovery

6. Remote read word at PTOCP

7. If word data is 0, PTOC collection is complete goto step 11

8. Save value in local PTOC table, and increment local PTOC table index

9. Increment PTOCP value by 2 (as it is a word pointer)

10. Repeat from step 6

REFERENCE INFORMATION

SSLBP

DISCOVERY SEQUENCE

SECOND PART, READ PROCESS DESCRIPTOR AND MODE DESCRIPTOR RECORDS:

11. For each PTOC entry acquired in the previous step:

12. Remote read byte at PTOC+0

12. If byte is 0xA0, proceed to step 16, reading process data descriptor

14 If byte is 0xB0, proceed to step 25 reading mode descriptor

15. If byte is neither, there is a error

16. Remote read byte at PTOC+1 This is DATA_SIZE

17. Remote read byte at PTOC+2 This is DATA_TYPE

18. Remote read byte at PTOC+3 This is DATA_DIRECTION

19. Remote read long at PTOC+4 This is PARAM_MIN.

20. Remote read long at PTOC+8 This is PARAM_MAX

21. Remote read word at PTOC+10 This is PARAM_ADD (not used normally)

22. Read UNIT_STRING starting at PTOC+12

    Initialize CharPointer to PTOC+12

    repeat (remote read byte at CharPointer, increment CharPointer, if byte is 0: done)

23 Read NAME_STRING starting at CharPointer

    repeat (remote read byte at CharPointer, increment CharPointer, if byte is 0: done)

24. Repeat with next PTOC = step 11

REFERENCE INFORMATION

SSLBP

DISCOVERY SEQUENCE

SECOND PART, READ PROCESS DESCRIPTOR AND MODE DESCRIPTOR RECORDS:

25. Remote read byte at PTOC+1 This is MODE_INDEX

26. Remote read byte at PTOC+2 This is MODE TYPE

27. Read MODE_NAME_STRING starting at PTOC+4

    Initialize CharPointer to PTOC+4

    repeat (remote read byte at CharPointer, increment CharPointer, if byte is 0: done)

28. Repeat with next PTOC = step 1

29. Select next channel # and repeat from step 5

REFERENCE INFORMATION

Note that the low level serial interface details presented here are not normally needed for users or driver writers, as all the low level serial protocol details are handed by the SSLBP code in the SSerial interface built into the FPGA. This information is presented here for completeness

LBP

LBP is a simple binary master slave protocol where the host sends read, write, or RPC commands to the 7I69, and the 7I69 responds. All controller communication to the 7I69 is done via LBP. LBP commands always start with a command header byte. This header specifies whether the command is a read or write or RPC, the number of address bytes(0, or 2), and the number of data bytes(1 through 8).The 0 address size option indicates that the current address pointer should be used. This address pointer will be post incremented by the data size if the auto increment bit is set. RPC commands allow any of up to 64 stored commands to be executed in response to the single byte command.

Note that the low level serial interface details presented here are not normally needed for 7I69 card access, as all the low level details are handed by the SSLBP code in the SSerial interface built into the FPGA, but is presented here for completeness.

LBP DATA READ/WRITE COMMAND

0	1	WR	RID	AI	AS	DS1	DS0

Bit 7.. 6 **CommandType:** Must be 01b to specify data read/write command

Bit 5 **Write:** 1 to specify write, 0 to specify read

Bit 4 **RPCIncludesData:** 0 specifies that data is from stream, 1, that data is from RPC (RPC only, ignored for non RPC commands)

Bit 3 **AutoInc:** 0 leaves address unchanged, 1 specifies that address is post incremented by data size in bytes.

BIT 2 **AddressSize:** 0 to specify current address, 1 to specify 2 byte address.

Bit 1..0 **DataSize:** Specifies data size, 00b = 1 bytes, 01b = 2 bytes, 10 b= 4 bytes, 011b = 8 bytes.

When multiple bytes are specified in a read or write command, the bytes are always written to or read from successive addresses. That is, a 4 byte read at location 0x21 will read locations 0x21, 0x22, 0x23, 0x24. The address pointer is not modified after the command unless the AutoInc bit is set.

REFERENCE INFORMATION

LBP

EXAMPLE LBP COMMANDS

Write 4 bytes (0xAA, 0xBB,0xCC,0xDD) to addresses 0x010,0x011,0x012,0x013 with AutoInc so that the address pointer will be left at 0x014 when the command is completed:

COMMAND BITS	CT1	CT0	WR	RID	AI	AS	DS1	DS0
LBPWrite: 2 add 4 data	0	1	1	0	1	1	1	0
Write Address LSB	0	0	0	1	0	0	0	0
Write Address MSB	0	0	0	0	0	0	0	0
Write data 0	1	0	1	0	1	0	1	0
Write Data 1	1	0	1	1	1	0	1	1
Write Data 2	1	1	0	0	1	1	0	0
Write Data 3	1	1	0	1	1	1	0	1

Write 2 more bytes (0xEE,0xFF) at 0x014 and 0x015:

COMMAND BITS	CT1	CT0	WR	RID	AI	AS	DS1	DS0
LBPWrite: 0 add 2 data	0	1	1	0	0	0	0	1
Write data 0	1	1	1	0	1	1	1	0
Write data 1	1	1	1	1	1	1	1	1

Read 8 bytes at 0x010,0x011,0x012,0x013,0x014,0x015,0x016,0x017:

COMMAND BITS	CT1	CT0	WR	RID	AI	AS	DS1	DS0
LBPRead: 2 add 8 data	0	1	0	0	0	1	1	1
Read Address LSB	0	0	0	1	0	0	0	0
Read Address MSB	0	0	0	0	0	0	0	0

REFERENCE INFORMATION

LBP

LOCAL LBP COMMANDS

In addition to the basic data access commands, there are a set of commands that access LBP status and control the operation of LBP itself. These are organized as READ and WRITE commands

LOCAL LBP READ COMMANDS

(HEX), all of these commands return a single byte of data.

0xC0 Get unit address

0xC1 Get LBP status

LBP Status bit definitions:

BIT 7 Reserved

BIT 6 Command Timeout Error

BIT 5 Invalid write Error (attempted write to protected area)

BIT 4 Buffer overflow error

BIT 3 Watchdog timeout error

BIT 2 Reserved

BIT 1 Reserved

BIT 0 CRC error

0xC2 Get CRC enable status (note CRCs are always enabled on the 7I69)

0xC3 Get CRC error count

0xC4 .. 0xC9 Reserved

0xCA Get Enable_RPCMEM access flag

0xCB Get Command timeout (character times/10 for serial)

0xCC .. 0xCF Reserved

0xD0 .. 0xD3 4 character card name

REFERENCE INFORMATION

LBP

LOCAL LBP READ COMMANDS

0xD5 .. 0xD7 4 character configuration name (only on some configurations)

0xD8 Get low address

0xD9 Get high address

0xDA Get LBP version

0xDB Get LBP Unit ID (Serial only, not used with USB)

0xDC Get RPC Pitch

0xDD Get RPC SizeL (Low byte of RPCSize)

0xDE Get RPC SizeH (High byte of RPCSize)

0xDF Get LBP cookie (returns 0x5A)

REFERENCE INFORMATION

LBP

LOCAL LBP WRITE COMMANDS

(HEX), all of these commands except 0xFF expect a single byte of data.

0xE0 Reserved

0xE1 Set LBP status (0 to clear errors)

0xE2 Set CRC check enable (Flag non-zero to enable CRC checking)

0xE3 Set CRC error count

0xE4 .. 0xE9 Reserved

0xEA Set Enable_RPCMEM access flag (non zero to enable access to RPC memory)

0xEB Set Command timeout (in mS for USB and character times for serial)

0xEC .. 0xEF Reserved

0xF0 .. 0xF6 Reserved

0xF7 Write LEDs

0xF8 Set low address

0xF9 Set high address

0xFA Add byte to current address

0xFB .. 0xFC Reserved

0xFD Set unit ID (serial only)

0xFE Reset LBP processor if followed by 0x5A

0xFF Reset LBP parser (no data follows this command)

REFERENCE INFORMATION

LBP

RPC COMMANDS

RPC commands allow previously stored sequences of read/write commands to be executed with a single byte command. Up to 64 RPC's may be stored. RPC write commands may include data if desired, or the data may come from the serial data stream. RPCs allow significant command compression which improves communication bandwidth. When used with SSLBP, the 7I69s process data transfer uses an RPC for efficiency.

LBP RPC COMMAND

1	0	RPC5	RPC4	RPC3	RPC2	RPC1	RPC0

Bit 7..6 **CommandType:** must be 10b to specify RPC

Bit 5..0 **RPCNumber:** Specifies RPC 0 through 63

In the 7I69 LBP implementation, RPCPitch is 0x8 bytes so each RPC command has native size of 0x08 bytes and start 0x8 byte boundaries in the RPC table area. RPCs can cross RPCPitch boundaries if larger than RPCPitch RPCs are needed. The stored RPC commands consist of LBP headers and addresses, and possibly data if the command header has the RID bit set. RPC command lists are terminated by a 0 byte.

The RPC table is accessed at addresses 0 through RPCSize-1 This means with a RPCPitch of 0x8 bytes, RPC0 starts at 0x0000, RPC1 starts at 0x008, RPC2 starts at 0x0010 and so on.

Before RPC commands can be written to the RPC table,the RPCMEM access flag must be set. The RPCMEM access flag must be clear for normal operation.

REFERENCE INFORMATION

LBP

EXAMPLE RPC COMMAND LIST

This is an example stored RPC command list. Note RPC command lists must start at a RPCPitch boundary in the RPC table but an individual RPC list can extend until the end of the table. This particular RPC example contains 2 LBP commands and uses 7 bytes starting at 0x0028 (RPC5 for 0x08 pitch RPC table)

Command1. Writes two data bytes to address 0x10, 0x11 with 2 data bytes supplied by host

Command2. Reads two data bytes from address 0x12,0x13

COMMAND BITS	CT1	CT0	WR	RID	I	AS	DS1	DS0
LBPWrite: 2 add 2 data	0	1	1	0	0	1	0	1
Write Address LSB	0	0	0	1	0	0	0	0
Write Address MSB	0	0	0	0	0	0	0	0
LBPRead: 2 add 2 data	0	1	0	0	0	1	0	1
Read Address LSB	0	0	0	1	0	0	1	0
Read Address MSB	0	0	0	0	0	0	0	0
Terminator	0	0	0	0	0	0	0	0

The data stream for this RPC would consist of these 3 bytes:

COMMAND BITS	CT1	CT0	R5	R4	R3	R2	R1	R0
RPC 5	1	0	0	0	0	1	0	1
Data 0 for Command 1	0	1	0	1	0	1	0	1
Data 1 for Command 1	1	1	0	0	1	1	0	0

REFERENCE INFORMATION

CRC

LBP on the 7I69 uses CRC checking of all commands and data to insure validity. The CRC used is a 8 bit CRC using the same polynomial as the Dallas/Maxim one wire devices (X^8+X^5++X^4+X^0). The CRC must be appended to all LBP commands and all returned data will have a CRC byte appended. Commands with no returned data (writes or RPCs with no reads) will still cause a CRC byte to be returned, this CRC byte will always be 00H.

FRAMING

Since LBP is a binary protocol with no special sync characters, the packet framing must be determined by other methods.

Framing is done by a combination of timing and pre-parsing the serial data. Timing based framing is used to reset the parser at gaps in the serial data stream. This provides fast resynchronization to allow robust operation in noisy environments. The actual timeout used needs to be optimized for the operating mode. In setup mode where a non real-time OS may be communicating with the remote device, the frame timing is set to its maximum value (25.5 character times). This is equivalent to 2.1 mS at 115200 baud. This means that host communications cannot have more than 2.1 mS delays between characters in a command sequence when in setup mode.

In operate mode, command timeout is set by SSLBP to be 4 character times (16 uSec at 2.5M baud). The SSLBP firmware always sends commands in bursts without inter-character gaps so will always meet this timing. The timing is set short so that the parser on the remote device will always be reset and ready for the next command at the highest repetition rates even if data has been corrupted by noise so that incomplete commands have been received.

REFERENCE INFORMATION

SSERIAL REMOTE RPCS

SSerial remote devices must implement three special RPCs to be compatible with the hosts FPGA SSLBP firmware. These RPCs may be normal in-memory RPCs or special hardwired RPCs for speed. Normal programmable RPCs are not required for compatibility with SSLBP so need not be implemented.

UNIT NUMBER RPC

The unit number RPC returns the 4 byte remote unit number. Like all LBP data this is sent LSB first. This RPC is 0xBB hex.

DISCOVERY RPC

The discovery RPC returns the total sizes of the receive and transmit process data in bytes and returns 16 bit pointers to the PTOC and GTOC (which are in turn tables of pointers to process data records and mode records). The discovery RPC is 0xBC hex.

Return data bytes are in the following order: RXSize, TXSize, PTOCLSB, PTOCMSB, GTOCLSB, GTOCMSB.

RXSize is host relative so this is the size of data that the remote transmits. Likewise TXSize is host relative so this is the size of process data the remote receives. Note that the remote should check its remote SW mode and remote HW mode flags and return size data and pointers appropriate for the currently selected mode. Note that the remote always sends remote fault data as the first byte of the process data sent to the host. This extra byte of data must be reflected in the RXSize byte.

PROCESS DATA RPC

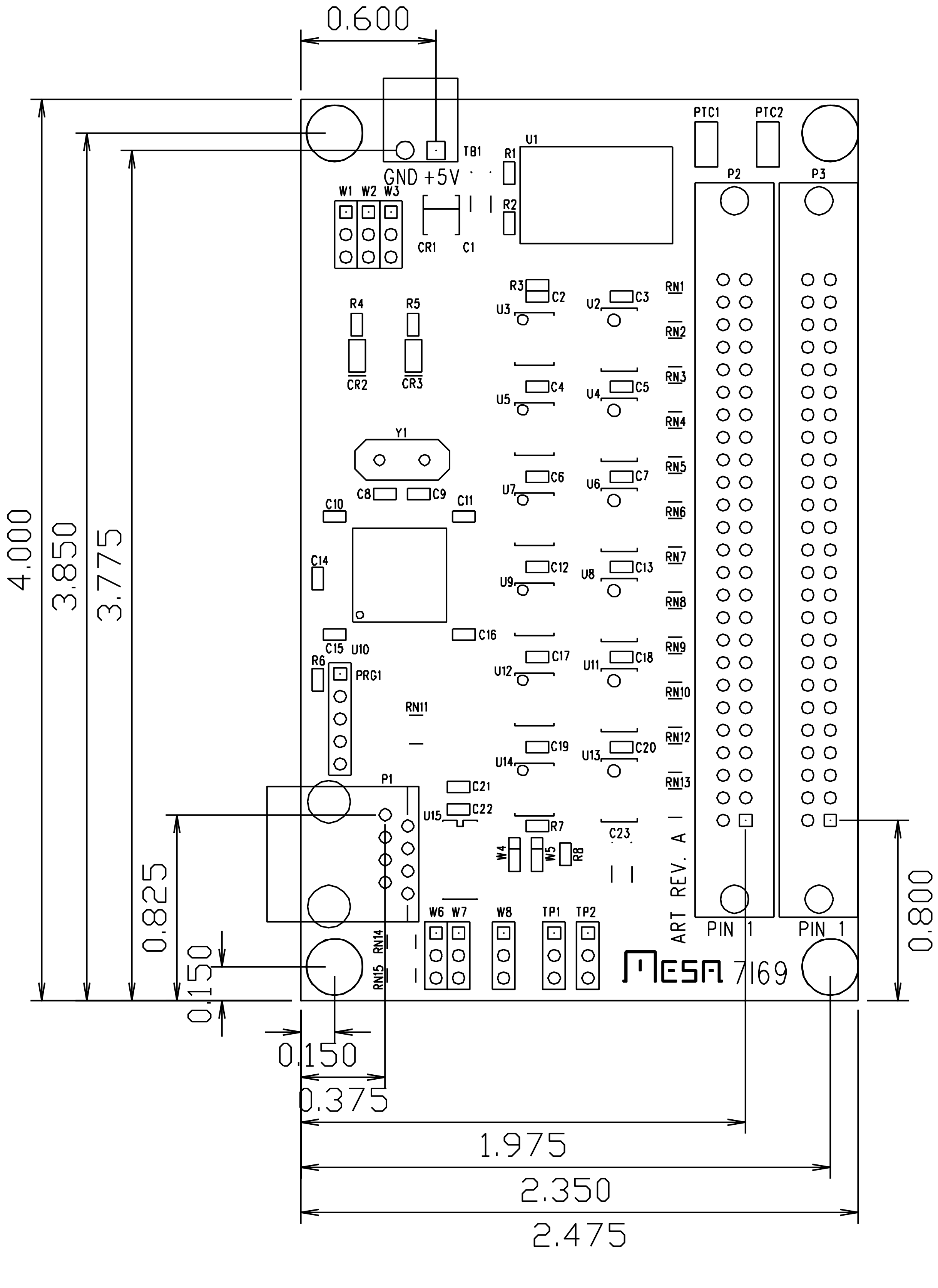
The Process data RPC is used to transfer process data to and from the host. The process data RPC should always receive and send the amount of RX and TX data that the Discovery RPC indicates. As mentioned above, the first byte of data sent from the remote to the host is always remote fault information as listed in CS REGISTER AFTER DOIT section of the manual. The process data RPC is 0xBD hex.

REFERENCE INFORMATION

SPECIFICATIONS

	MIN	MAX	NOTES
LOGIC SUPPLY VOLTAGE 5V	4.5V	5.5V	
5V CURRENT	----	200 mA	No external load.
I/O INPUT VOLTAGE RANGE	0V	5.5V	
OUTPUT SINK CURRENT	0	15mA	@. 0.4V VOL
OUTPUT SOURCE CURRENT	600uA	—	@ 2.4V VOH
	(3.3K pullups @ 4.5V VCC)		
TEMPERATURE -C VERSION	0^{o}C	70^{o}C	
TEMPERATURE -I VERSION	-40^{o}C	85^{o}C	

DRAWINGS


0.600
TB1
GND +5V
U1
R1
R2
PTC1
PTC2
P2
P3
W1 W2 W3
CR1
C1
R3
C2
C3
U3
U2
R4
R5
CR2
CR3
RN1
RN2
RN3
RN4
RN5
RN6
RN7
RN8
RN9
RN10
RN12
RN13
U5
C4
U4
C5
Y1
U7
C6
U6
C7
C8
C9
C10
C11
C14
U9
C12
U8
C13
C15 U10
C16
R6
PRG1
U12
C17
U11
C18
RN11
U14
C19
U13
C20
P1
C21
C22
U15
R7
C23
W4
W5
R8
ART REV. A
W6 W7
W8
TP1
TP2
RN14
RN15
PIN 1
PIN 1
MESA 7I69
4.000
3.850
3.775
0.825
0.150
0.150
0.375
1.975
2.350
2.475
0.800