REAL WORLD SIGNAL PROCESSING™

PRU Message Handler for IO Link Master

Applies to Product Release: 01.00.00.00
Publication Date: February 8, 2019

Texas Instruments, Incorporated
12500 TI Blvd
Dallas, TX 75243 USA

HISTORY

Version	Date	Author	Notes
0.1	2017/06/28	Thomas Leyrer	Initial version of document in evaluation stage
0.2			After feedback from master stack partner
0.3	2019/02/08	Hao Zhang	Re-format the document for SDK release

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS, ABBREVIATIONS AND DEFINITIONS

Acronym	Definition
PRU	Programmable Real-time Unit
ICSS	Industrial Communication SubSystem
MPU	Microprocessor Unit
CPU	Central Processing Unit
SDCI	Single-drop digital communication interface

1. OVERVIEW

This document describes the IO Link message handler interface between Programmable Real-time Unit (PRU) and ARM MPU on Sitara^{TM} processor. The massage handler off-loads the real-time functions on the ARM CPU in order to support full performance of up to 8 or 16 IO Link channels. Maximum performance for IO Link connection is expressed in a minimum cycle time of 400 us. Multiple channels may work on different cycle times which are asynchronous. For example channel 1 works on 400 us and channel two works on 700 us. The slowest cycle time of 132ms also needs to be supported.

The frame handler is written in a way that it supports the high channel count at full PRU speed and single/dual channel implementation with reduced PRU speed. PRU-ICSS on Sitara^{TM} supports different modes to read in serial data and emulate UART operation. For high channel count implementation the general purpose input/output (GPIO) mode can support for example 8 channels with one PRU. In this mode external GPIO pins are mapped to PRU registers R30/R31. For implementations of up to 3 channels the serial shift mode or 3 channel serial interface mode should be used. In this mode data is received through overclocked shift register.

The IO Link master services and structure is shown in figure 1. The message handler is part of the data link layer and executes data exchange for on-request data objects and process data objects. Inside the message handler there is a function which splits the message into single UART transfers.

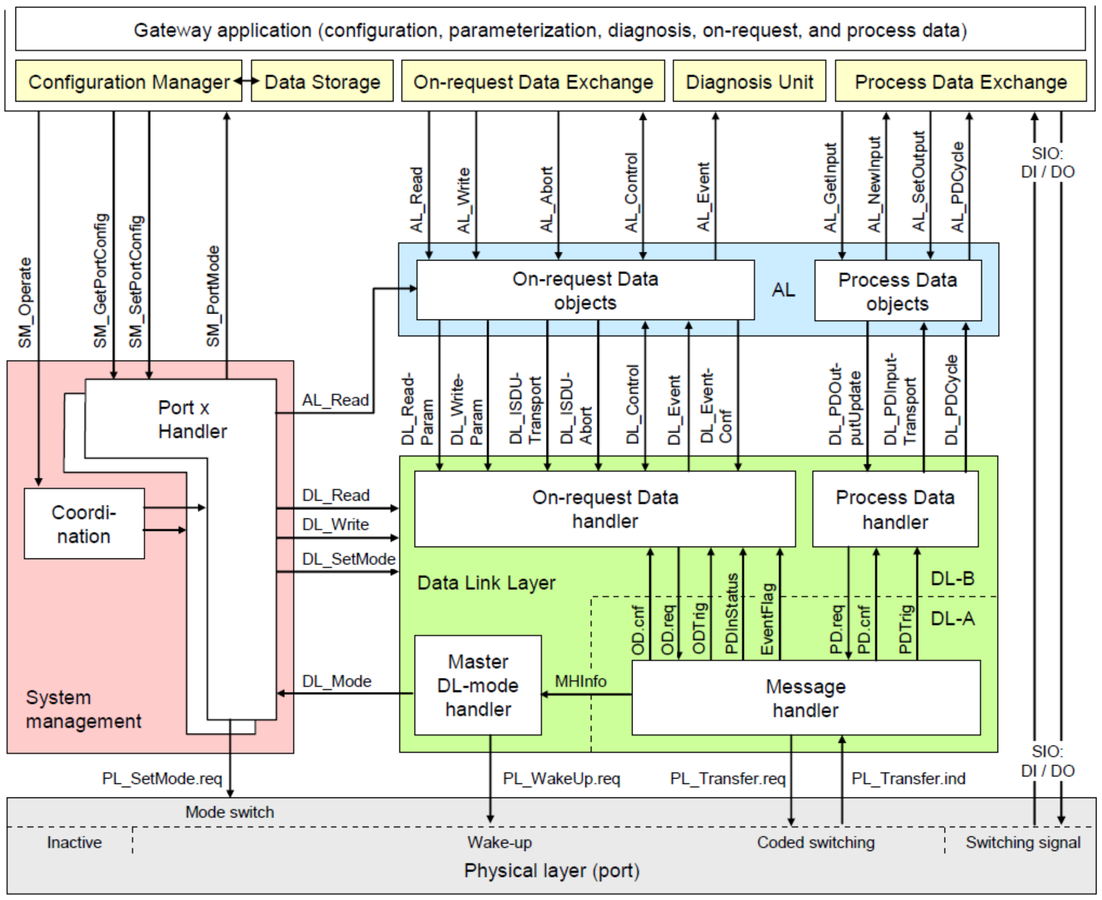

Figure 1 Structure and services of a Master

The timing parameters for the message and UART are specified in the IO-Link standard [1]. The definition includes the UART baud rate, time gaps between UART frames and messages, response time from device, message sequence time and cycle time. Figure 2 shows the basic timing parameters.

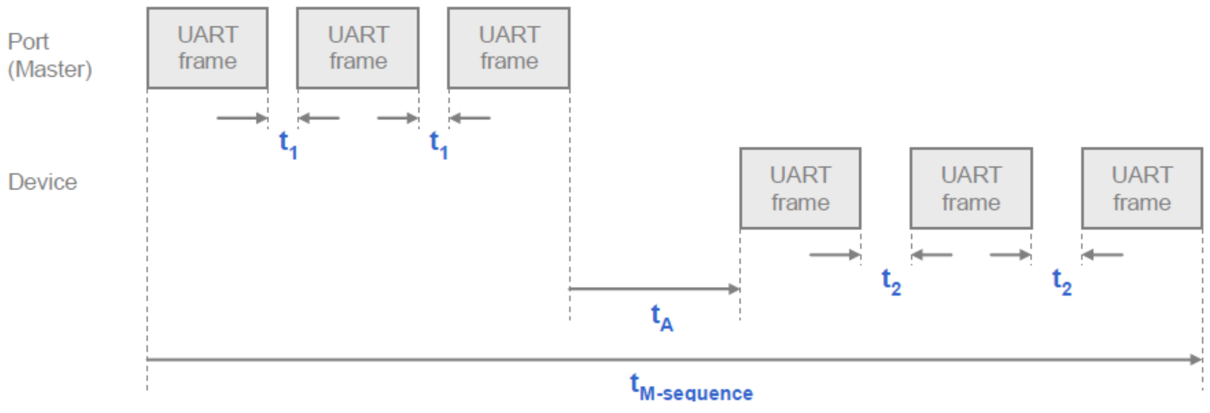

Figure 2 IO-Link physical layer communication timing

A new communication cycle begins with a message from the master followed by a response message from the device. A message is variable in length and structure. Common to all messages is the M-sequence control (MC) byte and a checksum byte (CKT/CKS). The number of process data bytes and on-request data bytes can be up to 32 bytes. A message from the master which contains maximum size of process data (PD) and on-request data (OD) takes 66 bytes. On device side the maximum size is 65 bytes. An example of message exchange between master and device is given in figure 3. Each octet is translated to a UART frame. The checksum byte on master message covers the whole message from MC to last byte of PD/OD.

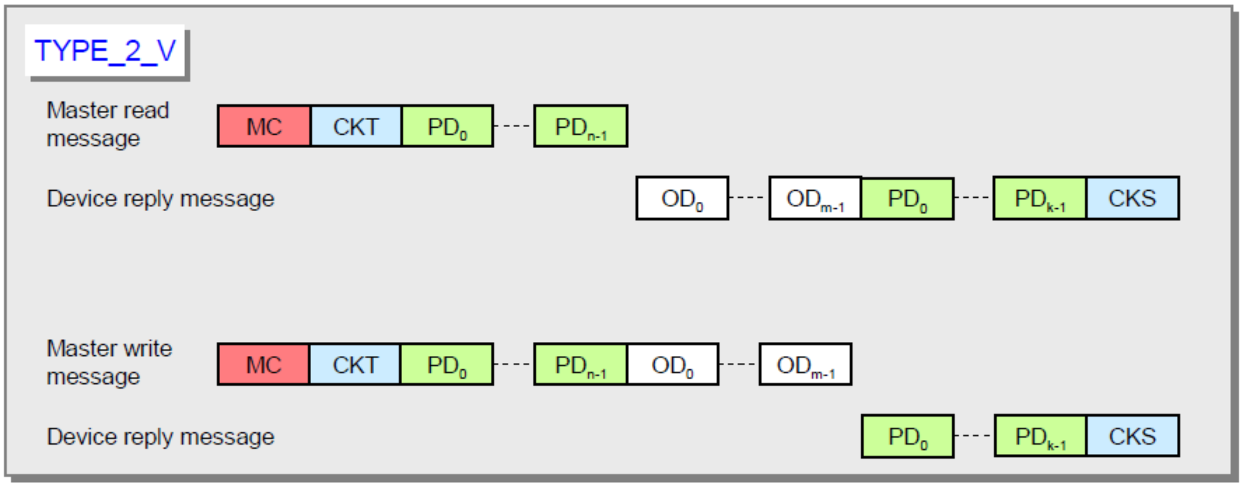

Figure 3 M-sequence TYPE_2_V

The transmission of messages needs to be scheduled according to configured timing parameters and tolerances of device message. Error checks are on UART interface with parity bit and valid frame parameter. The device reply message needs to have correct check-sum, length and timing. In addition to message time of a communication cycle there is an idle time which gives the device enough time to become ready for the next receive message.

1.1. Standards and References

Ref	Title / Source	Owner	Version
[1]	IO-Link Interface and System Specification	IO-Linik	V1.1.2
[2]			
[3]			
[4]			

2. PRU MESSAGE HANDLER

The message handler on PRU is a subset of the message handler defined in the standard. It supports a message interface for multiple channels and takes over the UART frame transmit and receive function.

Supported features are: (optional)

- 8 channel IO Link Master frame handler
- min 400us cycle time for 8 ports
- channel independent cycle time
- 68 bytes receive buffer
- two 68 bytes send buffer
- time control and monitoring per port
- (on the fly receive parser and notification)
- receive glitch filter and 8x oversampling
- 3% receive tolerance (baud rate)
- 0..3 bit receive symbol gap
- (check sum calculation and verification)
- MessageHandlerInfo (lost, illegal, error)
- (frame repeater on error)
- receive timing diagnostics

Figure 4 gives an overview of the message handler implementation on PRU. Before a frame is transferred by the message handler the host (ARM CPU) configures timing parameters for each channel. PRU message handler at this point in time has no understanding of message handler states like STARTUP, PREOPERATE or OPERATE. It only understands enable or disable. If enabled the message in the send buffer goes out with defined timing parameters. The header of the send buffer contains one byte with length of message in number of bytes. It is followed by MC and CKT bytes at a minimum. The header of the receive buffer is also written by host to indicate the expected number of received bytes. PRU will compare actual received bytes with the number of bytes provided by host and signal an error in case of miss match. PRU also verifies timing of received message (ta and t2).

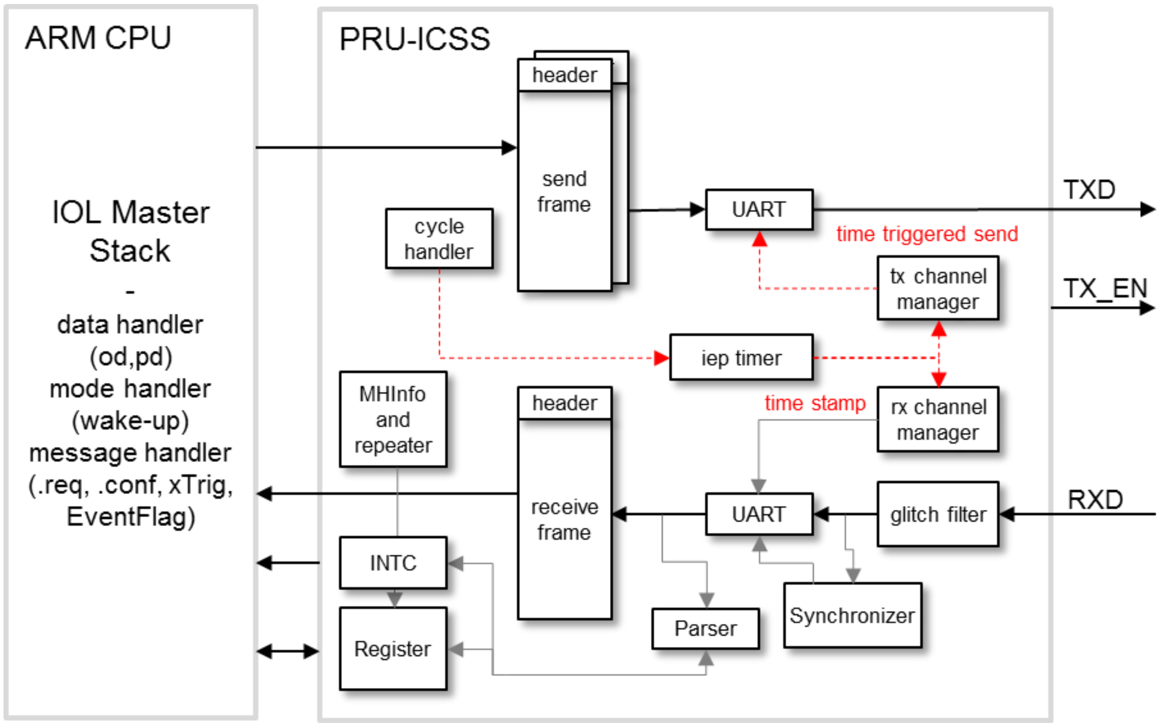

Figure 4 - PRU message handler for one channel

The start time of a message depends on the configured cycle time and is synchronized to a 100 us base clock which is common to all 8 channels. This allows running multiple channels synchronized with exactly the same cycle start time. PRU sends out the data using UART frame format with start bit, stop bit and parity bit. After the message was sent the TX_EN pin is toggled in the middle of a bit time, e.g. 2 us at max baud rate.

The response time from the device can be anything between 0 and 10 bit times. UART frames from the device may have a gap of up to 3 bit times. PRU receive processing is waiting for the start bit using 8x oversampling. A glitch filter is applied to the oversampled data and sample point is re-calculated for every UART frame. In case there is no valid response from device in given time window the message is repeated by the transmitter. The length of the received message is known by the master. PRU reads each UART frame data byte and writes it into receive buffer. An error is indicated in case the numbers of bytes do not match or the timing of received bytes is not within range. A start of message time stamp for each received message is recorded and can be appended to the frame. After complete frame has been received and interrupt is generated towards the ARM CPU to pick up the frame.

As shown in figure 5, ARM indicated new message to be sent by PRU through TX_flag.x bit. By setting multiple channels at the same time causes synchronous communication start of the selected channels. PRU communication of selected channels happens with 5ns accuracy. TX_flag.x is self-cleared by ARM on next 100 us interrupt. PRU polls for TX_flag.x bit with 4.3 us loop.

TX_EN is controlled by PRU and switches back to transmit after 3 bit times of last receive symbol. Before PRU controls the direction of physical layer in the middle of first bit time after transmit message. A new communication cycle will start in case there are no parameters changed in the interface and no errors occurred.

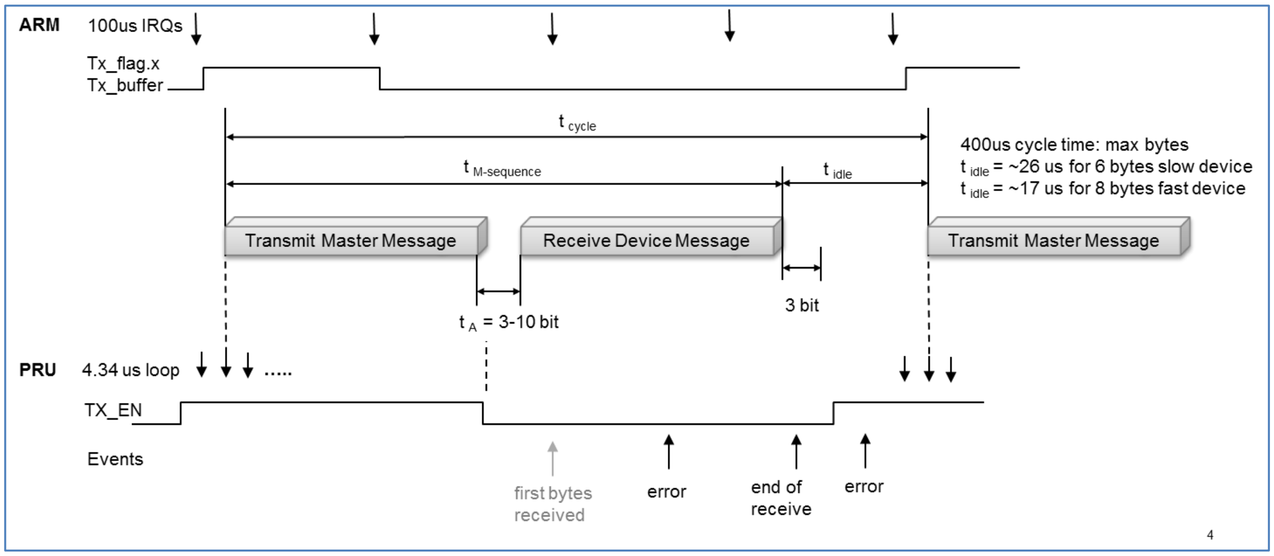

Figure 5 – Handshake between ARM and PRU on communication cycle

Exact 80 us wake-up pulse is generated by PRU and triggered by ARM. Initial communication sequence can be implemented using single mode where host configures new baud rate for each communication cycle.

3. INTERFACE DESCRIPTION

- Indication that PRU Message Handler was loaded and is ready for operation
- Firmware version register to identify features supported by given firmware release
- global control register to enable IO Link master message handler
- per channel enable register on byte boundaries – avoid read modify write operation
- per channel cycle time register in n x 100 us format; e.g. 4 = 400 us, 1000 = 10 ms
- per channel transmit mode register to set one-shot or continuous mode
- per channel baud rate register (T_{BIT}), v1 = 4.8k, v2 = 38.4k, v3=230.4k, v4= reserved
- per channel UART frame transmission delay (t_1) in 1/8th of bit time. Valid range is 0..8
- per channel MHinfo byte
- per channel repeat counter
- per channel receive time stamp for start of message (t_A – response time)
- per channel message transmit status: transmit pending, transmit done. Auto clear on new cycle
- per channel message receive status: receive empty, receive in progress, receive complete
- per channel message filter match – filter is tbd e.g. filter position, filter length, filter mask
- interrupt control register and status register and mask register for message sent
- interrupt control register and status register and mask register for message received
- interrupt control register and status register and mask register for filter match
- interrupt control register and status register and mask register for receive error
- per channel two send buffer, 68 bytes each
- per channel receive buffer, 68 bytes each

The registers and buffer memory are implemented using PRU-ICSS data memory. Depending on which Sitara™ device and which instance of PRU-ICSS is used the start address of PRU data memory varies. Here is the example of AM437x using PRU_ICSS0 and PRU0 Data memory. The global memory offset for PRU-ICSS is 0x5440_0000. Inside PRU-ICSS the data ram 0 of PRU-ICSS0 resides at 0x0004_0000 i.e. the ARM PRU sees this memory at address 0x544**4**_0000. The PRU0 sees the memory at address 0.

Start address of PRU message handler interface:

CORE	MEMORY ADDRESS	COMMENT
ARM Cortex A9	0x5444_0000	Used for driver and master stack
PRU-ICSS1 (other ICSS)	0x0004_0000	not used, e.g. Profinet IRT, EtherCAT
PRU_ICSS0 PRU0	0x0000	Low level message handler
PRU_ICSS0 PRU1	0x2000	Available for high level message handler

Table 1 Memory start address of interface

The data memory is split into first block used for registers followed by all receive buffers and then all transmit buffers. For easier address calculation the buffers are 32 byte aligned and take 96 bytes per buffer.

Note: yellow or red marked text not implemented in current version

Register Name	Offset	Host	Value	Comment
Global_Status	0x0000	R	Bit 0 0: firmware not start 1: message handler ready Bit 1-7: reserved	After boot firmware is loaded and started by ARM CPU
Global_Control	0x0001	R/W	Bit 0 0: message handler disabled	ARM triggers operation of message handler for all

			1: message handler enabled Bit 1-7: reserved	channels
Firmware_Revision	0x0002	R	Bit0-7: minor rev Bit 8-15: major rev	16 bit firmware revision number
CH0_Enable	0x0004	R/W	Bit 0 0: disabled 1: enabled Bit 1-7: reserved	Channel 0 enable register move all channels to one byte, arm to reset after 100us
CH0_TX_Mode	0x0005	R/W	Bit 0 0: single shot 1: cyclic mode Bit 1: 0: tx buffer 0 1: tx buffer 1 Bit 2-7: reserved	The message in send buffer is either sent once or continuous with every cycle. Continuous can be from same buffer or using double buffer mode. Bit one indicates which buffer is completed by host for PRU to send. ARM to indicate previous buffer on error interrupt
CH0_Cyle_Time	0x0006	R/W	16 bit value x 100 us 0-3: not supported 4-1328: 400 us .. 132.8 ms >1328: not supported	Message cycle time per channel according to V1.1.2 standard. Min and max not checked by firmware!
CH0_Baud_Rate	0x0008	R/W	8 bit value V1: 4.8 kbit/s V2: 38.4 kbit/s V3: 230.4 kbit/s V4: reserved (1 Mbit) != V1-V4: not supported	Values for baud rate configuration. 1 2 3
CH0_TX_Delay	0x0009	R/W	8 bit value 0 = no delay 1 = 1/8 T_{BIT} delay .. 7 = 7/8 T_{BIT} delay 8 = 1 T_{BIT} delay	Referred as t_1 in standard. Time gap between two UART framers on transmit (not supported by firmware today, use 0 delay)
CH0_MHinfo	0x000A	R	If bit is 0 than no error occurred Bit 0: 1 - lost communication Bit 1: 1 - illegal msg type Bit 2: 1 – checksum error Bit 3-7: reserved	Message Handler info for received message (no start =bit 0, length error bit 1, parity 1/2,
CH0_Repeat_cnt	0x000B	R	8 bit value: number of repeats	In operational state message is repeated in case of error. Up to two times according to standard.
CH0_RX_TS	0x000C	R	Bit0-15: 100 us count Bit16-31: 5 ns count TA in [bits]	Time stamp of incoming message from device. Reference point is start bit of first byte detected. Bit timer in PRU only.
CH0_RX_Status	0x0010	R	8 bit value: 0: receive empty/invalid 1: receive pending	In every cycle the status of receive buffer is indicated.

			2: receive complete >3: reserved	Interrupt after 1st byte received (optional)
CH0_TX_Status	0x0011	R	8 bit value: 0: transmit pending 1: transmit done >2: reserved	In every cycle the status of transmit buffer is indicated.
CH0_Filter_Data	0x0012	R/W	8 bit value: data byte to compare incoming message against	I message received can be parsed early on before whole message is received
CH0_Filter_Position	0x0013	R/W	8 bit value: position on data byte for comparison.	Potion starts with 1 for first byte received.
CH0_Filter_Mask	0x0014	R/W	8 bit value: bit mask for data comparison. If bit is set then the corresponding bit is not compared	Mask certain bits to exclude from filter operation
Reserved	0x0015- 0x0017		Reserved	Align next section to 32 bit
CH0_INT_TX_Status	0x0018	R	Bit 0 0: no TX interrupt occurred 1: TX interrupt occurred Bit 1-7: reserved	Status bit for transmit interrupt occurred. This is set by PRU when message was sent and buffer is available for new message.
CH0_INT_TX_CLR	0x0019	R/W	Bit 0 0: no action 1: clears TX status bit in Bit 1-7: reserved	Host can clear the TX_Status bit in interrupt service routine.
CH0_INT_TX_Mask	0x001A	R/W	Bit 0 0: no action 1: TX interrupt is masked Bit 1-7: reserved	When set ARM will not get an interrupt for TX message complete. Still it can poll and clear the status bit
Reserved	0x001B		Reserved	
CH0_INT_RX_Status	0x001C	R		
CH0_INT_RX_CLR	0x001D	R/W		
CH0_INT_RX_Mask	0x001E	R/W		
Reserved	0x001F		Reserved	
CH0_INT_Err_Status	0x0020	R		
CH0_INT_Err_CLR	0x0021	R/W		
CH0_INT_Err_Mask	0x0022	R/W		
Reserved	0x0023		Reserved	
CH0_INT_Flt_Status	0x0024	R		
CH0_INT_Flt_CLR	0x0025	R/W		
CH0_INT_Flt_Mask	0x0026	R/W		
Reserved	0x0027		Reserved	
Reserved	0x0028 0x002F		Reserved	
CH1_Enable	0x0034			
… repeat same registers as on CH0				
CH2_Enable	0x0064			
…				

CH3_Enable	0x0094			
…				
CH4_Enable	0x00C4			
…				
CH5_Enable	0x00F4			
…				
CH6_Enable	0x0124			
…				
CH7_Enable	0x0154			
…				
CH0 Receive Buffer	0x0200	R/W	96 bytes, first byte is length received, followed by message (OD, PD and CKS)	First byte is written by host. It contains the number of expected bytes. The PRU firmware verifies the number of received bytes.
CH1 Receive Buffer	0x0260	R		
CH2 Receive Buffer	0x02C0	R		
CH3 Receive Buffer	0x0320	R		
CH4 Receive Buffer	0x0380	R		
CH5 Receive Buffer	0x03E0	R		
CH6 Receive Buffer	0x0440	R		
CH7 Receive Buffer	0x04A0	R		
Reserved				
CH0 Transmit buffer 0	0x0600	R	96 bytes, first byte is tx length, second byte is rx length followed by message (OD,PD, CKT)	The verifies received byte with length
CH0 Transmit Buffer 1	0x0660	R		Second buffer for channel 0
CH1 Transmit Buffer 0	0x06C0	R		
CH1 Transmit Buffer 1	0x0720	R		
CH2 Transmit Buffer 0	0x0780	R		
CH2 Transmit Buffer 1	0x07E0	R		
CH3 Transmit Buffer 0	0x0840	R		
CH3 Transmit Buffer 1	0x08A0	R		
CH4 Transmit Buffer 1	0x0900	R		
CH4 Transmit Buffer 1	0x0960	R		
CH5 Transmit Buffer 0	0x09C0	R		
CH5 Transmit Buffer 1	0x0A20	R		
CH6 Transmit Buffer 0	0x0A80	R		
CH6 Transmit Buffer 1	0x0AE0	R		
CH7 Transmit Buffer 0	0x0B40	R		
CH7 Transmit Buffer 1	0x0BA0	R		

3.1. Interrupt Controller

4. PROGRAMMERS MODEL