



REAL WORLD SIGNAL PROCESSING™

PRU Message Handler for IO Link Master

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

Document License

This work is licensed under the Creative Commons Attribution-NoDerivs 3.0 Unported License. To view a copy of this license, visit http://creativecommons.org/licenses/by-nd/3.0/ or send a letter to Creative Commons, 171 Second Street, Suite 300, San Francisco, California, 94105, USA.

Contributors to this document

Copyright (C) 2019 Texas Instruments Incorporated - http://www.ti.com/



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

Document Version 0.3 Page 1 of 18



Document Version 0.3 Page 2 of 18



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	

Document Version 0.3 Page 3 of 18



TABLE OF CONTENTS

HISTORY	
TABLE OF CONTENTS	4
1. OVERVIEW	
1.1. STANDARDS AND REFERENCES	
2. PRU MESSAGE HANDLER	11
3. INTERFACE DESCRIPTION	14
3.1. Interrupt Controller	17
4 PROGRAMMERS MODEL	18



LIST OF FIGURES

FIGURE 1 STRUCTURE AND SERVICES OF A MASTER	9
FIGURE 2 IO-LINK PHYSICAL LAYER COMMUNICATION TIMING	9
FIGURE 3 M-SEQUENCE TYPE 2 V	10
FIGURE 4 - PRU MESSAGE HANDLER FOR ONE CHANNEL	12



L	.IST	OF	TAB	LES
---	------	----	------------	-----

Table 1 Memory start address of interface

Document Version 0.3 Page 6 of 18



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

Document Version 0.3 Page 7 of 18



1. OVERVIEW

This document describes the IO Link message handler interface between Programmable Real-time Unit (PRU) and ARM MPU on SitaraTM 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 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.

Document Version 0.3 Page 8 of 18



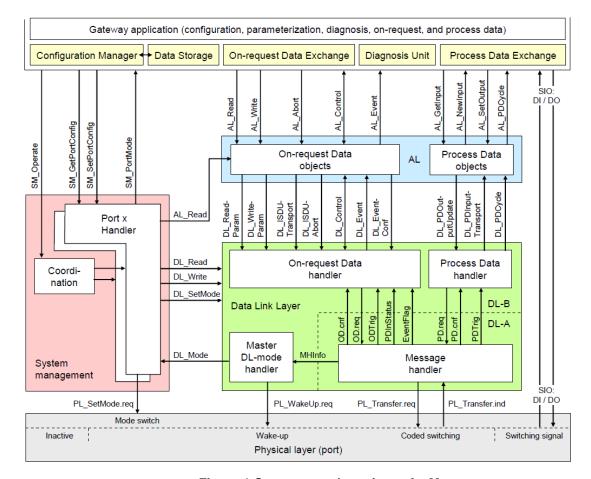


Figure 1 Structure and services of a Master

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.

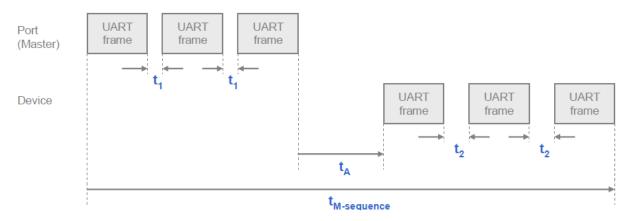


Figure 2 IO-Link physical layer communication timing

Document Version 0.3 Page 9 of 18



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.

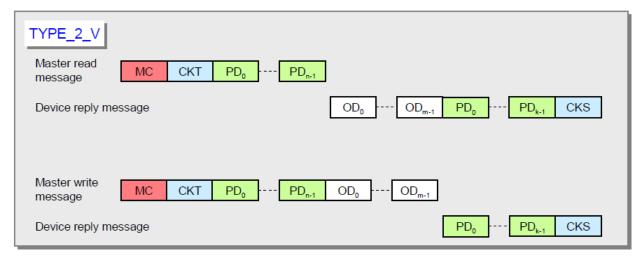


Figure 3 M-sequence TYPE_2_V

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]				

Document Version 0.3 Page 10 of 18



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).

Document Version 0.3 Page 11 of 18



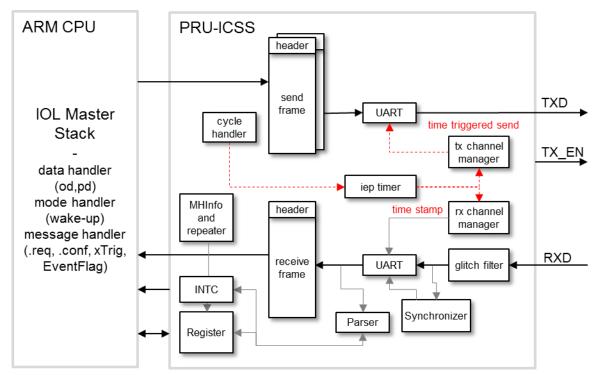


Figure 4 - PRU message handler for one channel

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.

Document Version 0.3 Page 12 of 18



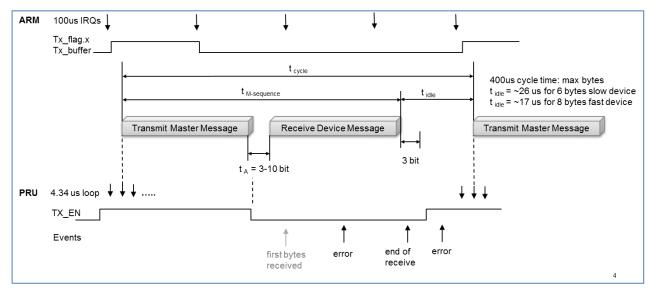


Figure 5 - Handshake between ARM and PRU on communication cycle

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.

Document Version 0.3 Page 13 of 18



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 (t₁) 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 SitaraTM 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 0x5444_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	After boot firmware is loaded
			0: firmware not start	and started by ARM CPU
			1: message handler ready	
			Bit 1-7: reserved	
Global_Control	0x0001	R/W	Bit 0	ARM triggers operation of
			0: message handler disabled	message handler for all

Document Version 0.3 Page 14 of 18



			1: message handler enabled	channels
			Bit 1-7: reserved	
Firmware_Revision	0x0002	R	Bit0-7: minor rev	16 bit firmware revision
			Bit 8-15: major rev	number
CH0_Enable	0x0004	R/W	Bit 0	Channel 0 enable register
			0: disabled	move all channels to one
			1: enabled	byte, arm to reset after 100us
			Bit 1-7: reserved	
CH0_TX_Mode	0x0005	R/W	Bit 0	The message in send buffer is
			0: single shot	either sent once or continuous
			1: cyclic mode	with every cycle. Continuous
			Bit 1:	can be from same buffer or
			0: tx buffer 0	using double buffer mode. Bit
			1: tx buffer 1	one indicates which buffer is
			Bit 2-7: reserved	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	Message cycle time per
			0-3: not supported	channel according to V1.1.2
			4-1328: 400 us 132.8 ms	standard. Min and max not
			>1328: not supported	checked by firmware!
CH0_Baud_Rate	8000x0	R/W	8 bit value	Values for baud rate
			V1: 4.8 kbit/s	configuration.
			V2: 38.4 kbit/s	1
			V3: 230.4 kbit/s	2
			V4: reserved (1 Mbit)	3
			!= V1-V4: not supported	
CH0_TX_Delay	0x0009	R/W	8 bit value	Referred as t ₁ in standard.
			0 = no delay	Time gap between two UART
			1 = 1/8 T _{BIT} delay	framers on transmit
			<u> </u>	
			7 = 7/8 T _{BIT} delay	(not supported by firmware
		<u> </u>	8 = 1 T _{BIT} delay	today, use 0 delay)
CH0_MHinfo	0x000A	R	If bit is 0 than no error	Message Handler info for
			occurred	received message
			Bit 0: 1 - lost communication	(no start =bit 0, length error
			Bit 1: 1 - illegal msg type	bit 1, parity 1/2,
			Bit 2: 1 – checksum error	
OLIO B	0.0000	_	Bit 3-7: reserved	
CH0_Repeat_cnt	0x000B	R	8 bit value: number of	In operational state message
			repeats	is repeated in case of error.
				Up to two times according to
CHO DV TO	0.0000	D	Dit0 15: 100 ::: 00:::::	standard.
CH0_RX_TS	0x000C	R	Bit0-15: 100 us count	Time stamp of incoming
			Bit16-31: 5 ns count	message from device.
			TA in [hite]	Reference point is start bit of
			TA in [bits]	first byte detected. Bit timer in PRU only.
CH0_RX_Status	0x0010	R	8 bit value:	In every cycle the status of
10_10.C_0tatao	0,0010	``	0: receive empty/invalid	receive buffer is indicated.
			1: receive pending	
		1	1 occiro poriding	

Document Version 0.3 Page 15 of 18



CH0_TX_Status CH0_TX_Status CH0_Filter_Data CH0_Filter_Data		I	1	Or receive complete	Interrupt after 1 st byte
CH0_TX_Status				2: receive complete	
CH0_Fitter_Data CH0_Fitter_Data CH0_Fitter_Position CH0_Fitter_Positi	OHO TV Otatus	00044	_		
CH0_Filter_Data Ox0012 R/W Solit value: data byte to compare incoming message against CH0_Filter_Position Ox0013 R/W Solit value: data byte to compare incoming message against CH0_Filter_Position Ox0013 R/W Solit value: position on data byte for comparison. CH0_Filter_Mask Ox0014 R/W Solit value: bit mask for data comparison. If bit is set then the corresponding bit is not compared CH0_INT_TX_Status Ox0015 Reserved Ox0015 Reserved Ox0017 Reserved Ox0017 R/W Solit 1-7: reserved Ox0018 R/W Solit 1-7: reserved Ox0019 R/W Solit 0 Ox0019 R/W Solit 0 Ox0019	CHO_1 X_Status	UXUUTT	K		
CH0_Filter_Data Ox0012 R/W 8 bit value: position on data byte for compare incoming message against CH0_Filter_Position Ox0013 R/W 8 bit value: position on data byte for comparison. CH0_Filter_Mask Ox0014 R/W 8 bit value: position on data comparison. If bit is set then the corresponding bit is not compared Reserved Ox0015 Ox0017 R 8 bit 0 CH0_INT_TX_Status Ox0018 R Bit 0 Ox0019 R/W Bit 0 Ox no action 1: Clears TX status bit in Bit 1-7: reserved CH0_INT_TX_Mask Ox0018 R/W Bit 0 Ox no action 1: TX interrupt is masked Bit 1-7: reserved CH0_INT_RX_Status CH0_INT_RX_Status Ox001B Reserved Ox001B Reserved Ox001B Reserved CH0_INT_RX_Status Ox001B Reserved Ox001B Reserved CH0_INT_RX_Status Ox001C R CH0_INT_RX_Status Ox001C R CH0_INT_RX_Status Ox001E R/W CH0_INT_RX_Mask Ox001E R/W CH0_INT_RX_Mask Ox001E R/W CH0_INT_FIT_Status Ox0020 R CH0_INT_FIT_Status Ox0021 R/W Reserved Ox0023 Reserved Ox0023 Reserved Ox0026 R/W Reserved Ox0027 Reserved Ox0027 Reserved Ox0028 Reserved Ox0028 Reserved Ox0027 Reserved Ox0028 Reserved Ox0028 Reserved Ox0029 Reserved Ox0029 Reserved Ox0029 Reserved Ox0029 Reserved Ox0027 Reserved Ox0027 Reserved Ox0028 Reserved Ox0027 Reserved Ox0027 Reserved Ox0028 Reserved Ox0028 Reserved Ox0027 Reserved Ox0028 Reserved Ox0029 Reserved Ox002				1: transmit done	transmit buller is indicated.
CH0_Filter_Data Ox0012 R/W Source Bit value: data byte to compare incoming message parased early on before whole message is received.					
CH0_Filter_Position Ox0013 R/W 8 bit value: position on data byte for comparison. CH0_Filter_Mask Ox0014 R/W 8 bit value: bit mask for data comparison. If bit is set then the corresponding bit is not compared Reserved Ox0015- Ox0017 Reserved Dx0018 Reserved Ox0018 Reserved CH0_INT_TX_Status CH0_INT_TX_Status CH0_INT_TX_CLR Ox0019 R/W Bit 0 Ox no action 1: TX interrupt cocurred Bit 1-7: reserved Dx001B Rit 1-7: reserved CH0_INT_TX_Mask Ox001A R/W Bit 0 Ox no action 1: TX interrupt is masked Bit 1-7: reserved CH0_INT_RX_Status Ox001B Reserved Ox001B Reserved Ox001B Reserved CH0_INT_RX_Status Ox001C R CH0_INT_RX_Status Ox001B Reserved Ox001B Reserved CH0_INT_RX_Status Ox001C R CH0_INT_RX_Status Ox001C R CH0_INT_RX_Status Ox001E R/W CH0_INT_RX_Mask Ox001E R/W CH0_INT_FIT_CLR Ox0020 R CH0_INT_FIT_Status Ox0020 R CH0_INT_FIT_Status Ox0022 R/W Reserved Ox0023 Reserved Ox0025 R/W Reserved Ox0026 R/W Reserved Ox0027 Reserved Ox0027 Reserved Ox0027 Reserved Ox0028 Reserved Ox0028 Reserved Ox0027 Reserved Ox0027 Reserved Ox0027 Reserved Ox0028 Ox0028 Reserved CH1_Enable Ox0004 Reserved Ox0027 Reserved Ox0028 Ox0028 Reserved Ox0029 R	CH0 Filter Data	0x0012	R/W		I message received can be
CHO_Filter_Position CMO013 R/W S bit value: position on data byte for comparison.	orio_rinor_bata	OXOO 12	1000	-	
CH0_Filter_Position Ox0013 RW bit value: position on data byte for comparison. CH0_Filter_Mask Ox0014 RW a bit value: bit mask for data comparison. If bit is set then the corresponding bit is not compared Reserved Ox0015- Ox0017 Reserved Ox0017 Reserved Ox0018 R Bit 0 O: no TX interrupt occurred Bit 1-7: reserved D: no action 1: clears TX status bit in Bit 1-7: reserved CH0_INT_TX_Mask Ox001A RW Bit 0 O: no action 1: TX interrupt is masked Bit 1-7: reserved Reserved Ox001B Reserved Ox001B Reserved Ox001B Reserved Ox001C Reserved Ox001D RW Reserved Ox001D RW Reserved Ox001F Reserved Ox001F Reserved CH0_INT_ETX_Mask Ox001C R Reserved Ox001F RW Reserved Ox001F Reserved CH0_INT_ETY_Status Ox0000 R Reserved Ox001F Reserved CH0_INT_ETY_Status Ox0000 R CH0_INT_ETY_Status Ox0000 R CH0_INT_ETY_Status Ox0000 R Reserved Ox001F Reserved CH0_INT_ETY_Status Ox0000 R Reserved Ox001F Reserved CH0_INT_ETY_Status Ox0000 R CH0_INT_ETY_Mask Ox0000 R Reserved Ox00027 Reserved Ox0027 Reserved Ox0027 Reserved Ox0028 Ox0029 R Reserved Ox00027 Reserved Ox00027 Reserved Ox00028 Ox0029 R Reserved Ox00029 R Reserved Ox00029 R Reserved Ox00020 R Reserved Ox00027 Reserved Ox00028 Ox0029 R Reserved Ox00029 R Reserved Ox00029 R Reserved Ox00029 R Reserved Ox00020 R Reserved Ox0021 R Reserved Ox0022 R Reserved Ox0023 R Reserved Ox0026 R Reserved Ox0027 R Reserved Ox0027 R Reserved Ox0028 R Ox0029 R Reserved					
Dyte for comparison. Dyte received					
Dyte for comparison. Dyte received	CH0_Filter_Position	0x0013	R/W	8 bit value: position on data	Potion starts with 1 for first
Comparison. If bit is set then the corresponding bit is not compared				byte for comparison.	byte received.
Comparison. If bit is set then the corresponding bit is not compared					
Reserved	CH0_Filter_Mask	0x0014	R/W		the state of the s
Reserved				comparison. If bit is set then	from filter operation
Reserved 0x0015- 0x0017 CH0_INT_TX_Status 0x0018 CH0_INT_TX_Status 0x0018 CH0_INT_TX_CLR 0x0019 CH0_INT_TX_CLR 0x0019 CH0_INT_TX_CLR 0x001A CH0_INT_TX_Mask 0x001A CH0_INT_TX_Status 0x001C CH0_INT_RX_Status 0x001C CH0_INT_RX_Mask 0x001B CH0_INT_RX_Mask 0x001B CH0_INT_RX_Mask 0x001C CH0_INT_EX_Mask 0x001C CH0_INT_EX_CLR 0x001D CH0_INT_EX_CLR 0x001D CH0_INT_EX_CLR 0x001C CH0_INT_EX_C					
CH0_INT_TX_Status	December	00045			Aliana a sust a sustinua to 00 lait
CHO_INT_TX_Status	Reserved			Reserved	Align next section to 32 bit
O: no TX interrupt occurred 1: TX interrupt occurred 1: TX interrupt occurred Bit 1-7: reserved CH0_INT_TX_CLR Ox0019 R/W Bit 0 O: no action 1: clears TX status bit in Bit 1-7: reserved When message was sent and buffer is available for new message. CH0_INT_TX_Mask CH0_INT_TX_Mask Ox001A R/W Bit 0 O: no action 1: TX interrupt is masked Bit 1-7: reserved Bit 1-7: reserved CH0_INT_RX_Status Ox001B Reserved CH0_INT_RX_Status Ox001D RW CH0_INT_RX_CLR Ox001D RW CH0_INT_RX_Mask Ox001E Reserved Ox001F Reserved CH0_INT_Err_Status Ox001F Reserved CH0_INT_Err_Status Ox0020 CH0_INT_Err_Status Ox0021 CH0_INT_Err_LCLR Ox0021 CH0_INT_Err_Mask Ox0022 CH0_INT_FIt_Status Ox0023 CH0_INT_FIt_Status Ox0024 CH0_INT_FIt_Status Ox0025 CH0_INT_FIt_Status Ox0026 Reserved Ox0027 Reserved Ox0028 CH1_Enable Ox0034 Reserved Ox0034 Reserved Ox0034 Reserved Ox0034 Reserved Ox0028 Ox0027 Reserved Ox0028 Ox0028 Reserved Ox0028 Ox0029 CH1_Enable Ox0064	CHO INT TY Status		P	Rit 0	Status hit for transmit interrupt
1: TX interrupt occurred Bit 1-7: reserved Sit 1-7: reserved	CHO_INT_TX_Status	000016	^		
Bit 1-7: reserved buffer is available for new message					
CH0_INT_TX_CLR					
CH0_INT_TX_CLR Ox0019 R/W Bit 0 0: no action 1: clears TX status bit in Bit 1-7: reserved CH0_INT_TX_Mask CH0_INT_TX_Mask Ox001A R/W Bit 0 0: no action 1: clears TX status bit in 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 Ox001B Reserved CH0_INT_RX_Status CH0_INT_RX_CLR Ox001D RW CH0_INT_RX_Mask Ox001E Reserved CH0_INT_Err_Status Ox001F Reserved CH0_INT_Err_Status Ox001F Reserved CH0_INT_Err_Status Ox0020 R CH0_INT_Err_CLR Ox0021 CH0_INT_Err_Mask Ox0022 Reserved Ox0023 Reserved CH0_INT_FIt_Status Ox0024 Reserved CH0_INT_FIt_CLR Ox0025 RW CH0_INT_FIt_CLR Ox0027 Reserved Ox0028 Reserved Ox0027 Reserved CH1_Enable Ox0034 repeat same registers as on CH0 CH2_Enable Ox0064				Bit 17. Teserved	
CH0_INT_TX_Mask CH0_INT_TX_Mask CH0_INT_TX_Mask CH0_INT_TX_Mask Reserved CH0_INT_RX_Status CH0_INT_RX_Status CH0_INT_RX_Status CH0_INT_RX_Status CH0_INT_RX_CLR CH0_INT_RX_Mask CH0_INT_RX_Mask CH0_INT_EX_Mask CH0_INT_EX_Mask CH0_INT_EX_Mask CH0_INT_EX_Status CH0_INT_EX_Mask CH0_INT_EX_Status CH0_INT_EX_Mask CH0_INT_EX_Status CH0_INT_EX_CX_TANUS CH0_INT_EX_CX_TAN	CH0 INT TX CLR	0x0019	R/W	Bit 0	
1: clears TX status bit in Bit 1-7: reserved					
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_FIt_Status 0x0024 R CH0_INT_FIt_CLR 0x0025 R/W CH0_INT_FIt_Mask 0x0026 R/W Reserved 0x0027 Reserved Reserved 0x0028 Reserved 0x0027 Reserved Reserved CH1_Enable 0x0034 Reserved CH2_Enable 0x0064 Reserved					
O: no action 1: TX interrupt is masked Bit 1-7: reserved Complete. Still it can poll and clear the status bit				Bit 1-7: reserved	
1: TX interrupt is masked Bit 1-7: reserved	CH0_INT_TX_Mask	0x001A	R/W	Bit 0	When set ARM will not get an
Bit 1-7: reserved 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_FIt_Status 0x0024 R CH0_INT_FIt_CLR 0x0025 R/W CH0_INT_FIt_Mask 0x0025 R/W Reserved 0x0027 Reserved Reserved 0x0028 Reserved Reserved 0x0028 Reserved 0x002F Reserved 0x0034 repeat same registers as on CH0 Reserved 0x0064					
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 Reserved 0x002F Reserved 0x0034 repeat same registers as on CH0 0x0064					clear the status bit
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_Fit_Status 0x0024 R CH0_INT_Fit_CLR 0x0025 R/W CH0_INT_Fit_Mask 0x0026 R/W Reserved 0x0027 Reserved Reserved 0x0028 Reserved 0x002F Reserved 0x0034 repeat same registers as on CH0 0x0064 Image: Chi and the context of the				Reserved	
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_Fit_Status 0x0024 R CH0_INT_Fit_CLR 0x0025 R/W CH0_INT_Fit_Mask 0x0026 R/W Reserved 0x0027 Reserved Reserved 0x0028 Reserved 0x002F Reserved 0x0034 repeat same registers as on CH0 0x0064 Ox0064					
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_Fit_Status 0x0024 R CH0_INT_Fit_CLR 0x0025 R/W CH0_INT_Fit_Mask 0x0026 R/W Reserved 0x0027 Reserved Reserved 0x0028 Reserved 0x002F Reserved Reserved CH1_Enable 0x0034 Image: Chapter of the cha					
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_FIt_Status 0x0024 R CH0_INT_FIt_CLR 0x0025 R/W CH0_INT_FIt_Mask 0x0026 R/W Reserved 0x0027 Reserved Reserved 0x0028 Reserved 0x002F Reserved 0x0034 repeat same registers as on CH0 0x0064 0x0064			R/W	Barrier	
CH0_INT_Err_CLR 0x0021 R/W CH0_INT_Err_Mask 0x0022 R/W Reserved 0x0023 Reserved CH0_INT_FIt_Status 0x0024 R CH0_INT_FIt_CLR 0x0025 R/W CH0_INT_FIt_Mask 0x0026 R/W Reserved 0x0027 Reserved Reserved 0x0028 Reserved 0x002F Reserved 0x0034 repeat same registers as on CH0 0x0064 0x0064			<u> </u>	Keserved	
CH0_INT_Err_Mask 0x0022 R/W Reserved 0x0023 Reserved CH0_INT_FIt_Status 0x0024 R CH0_INT_FIt_CLR 0x0025 R/W CH0_INT_FIt_Mask 0x0026 R/W Reserved 0x0027 Reserved Reserved 0x0028 Reserved 0x002F CH1_Enable 0x0034 repeat same registers as on CH0 CH2_Enable 0x0064				-	
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 Reserved CH1_Enable 0x0034 repeat same registers as on CH0 CH2_Enable 0x0064 ox0064			1		
CH0_INT_Fit_Status 0x0024 R CH0_INT_Fit_CLR 0x0025 R/W CH0_INT_Fit_Mask 0x0026 R/W Reserved 0x0027 Reserved Reserved 0x0028 Reserved 0x002F Reserved 0x0034 repeat same registers as on CH0 0x0064 Image: CH1_Enable on the control of			IT/VV	Posonyod	
CH0_INT_Fit_CLR 0x0025 R/W CH0_INT_Fit_Mask 0x0026 R/W Reserved 0x0027 Reserved Reserved 0x0028 Reserved 0x002F CH1_Enable 0x0034 repeat same registers as on CH0 0x0064 CH2_Enable 0x0064			Р	Veseiven	
CH0_INT_Fit_Mask 0x0026 R/W Reserved 0x0027 Reserved Reserved 0x0028 Reserved 0x002F CH1_Enable 0x0034 repeat same registers as on CH0 0x0064			1		
Reserved 0x0027 Reserved Reserved 0x0028 0x002F Reserved CH1_Enable repeat same registers as on CH0 0x0034 CH2_Enable 0x0064					
Reserved 0x0028 0x002F Reserved CH1_Enable repeat same registers as on CH0 0x0034 CH2_Enable 0x0064			17/ / /	Reserved	
Ox002F Ox0034 CH1_Enable 0x0034 repeat same registers as on CH0 0x0064 CH2_Enable 0x0064					
CH1_Enable 0x0034 repeat same registers as on CH0 CH2_Enable 0x0064	1.0001700			1.0301700	
repeat same registers as on CH0 CH2_Enable 0x0064	CH1 Enable				
registers as on CH0 CH2_Enable 0x0064		37.000			
CH2_Enable 0x0064					
		0x0064			

Document Version 0.3 Page 16 of 18



OHO Frakla	00004	1	1	
CH3_Enable	0x0094			
014 5 - 11	0.0004			
CH4_Enable	0x00C4			
	0.0054			
CH5_Enable	0x00F4			
	0.0101			
CH6_Enable	0x0124			
<u> </u>				
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

Document Version 0.3 Page 17 of 18



4. PROGRAMMERS MODEL

Document Version 0.3 Page 18 of 18