NRC7292 Application Note
(Host SPI)

Ultra-low power & Long-range Wi-Fi Module

Ver 1.0
Mar 28, 2019

Newracom, Inc.

NRC7292 Application Note (Host SPI)
Ultra-low power & Long-range Wi-Fi Module

Office
Newracom, Inc.
25361 Commercentre Drive, Lake Forest, CA 92630 USA
http://www.newracom.com

Contents

List of Tables

List of Figures

1 Overview

This document describes the architecture, basic operation and registers associated with host serial peripheral interface (HSPI) of the NRC7292.

2 Host SPI

NRC7292 contains SPI slave engine for a host interface. The SPI slave engine consists of two separate domain, device and host side, as shown in Figure 2.1. This separation is mainly for power save. In a deep-sleep mode, most parts of NRC7292 including device side are turned off, but the host side keeps awake to monitor wake-up trigger from the external host.

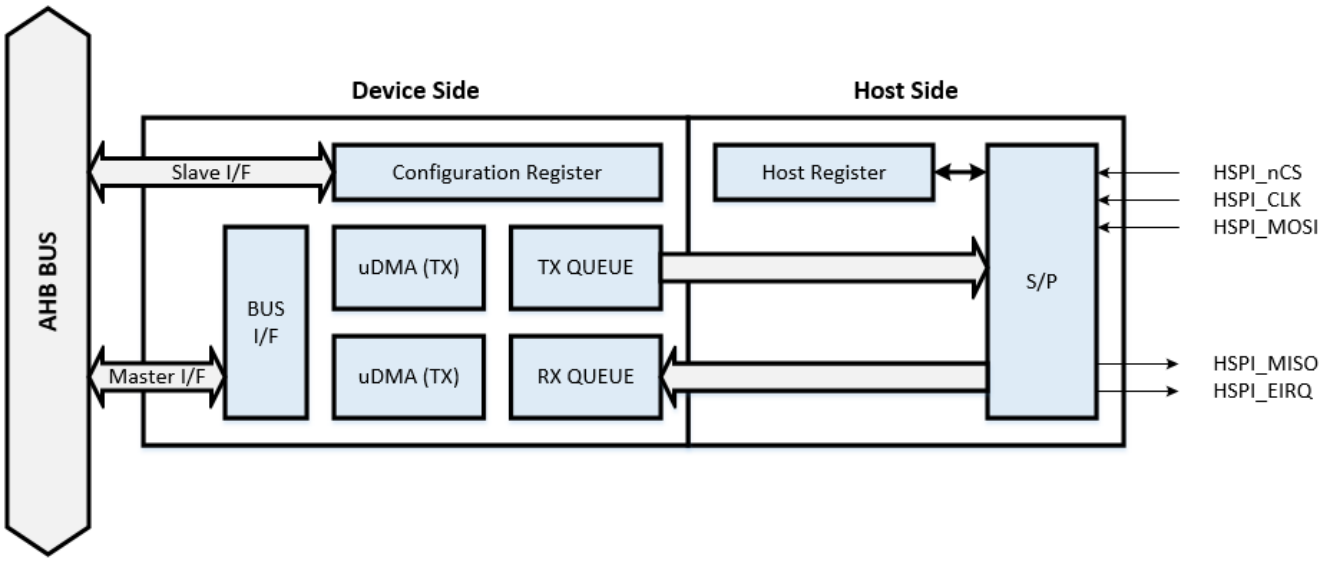

Figure 2.1 Block diagram of SPI slave engine of NRC7292

Table 2.1 HSPI pin description

Pin Name	I/O	Description
HSPI_EIRQ	Output	Interrupt to external host
HSPI_MOSI	Input	Master out Slave in
HSPI_MISO	Output	Master in Slave out
HSPI_nCS	Input	Chip select (active low)
HSPI_CLK	Input	Clock

A total of 5 dedicated pins are assigned for HSPI interface as presented in Table 2.1.

To use HSPI interface, the BOOT mode must be configured to HOST BOOT mode. The HOST BOOT mode can be selected by tying MODE_01 and MODE_02 pins to ground and MODE_00 pin to DVDDE_MODE. The DVDDE_MODE is the power supply for MODE_xx pins and the DVDDE_HSPI is for HSPI IOs such as HSPI_EIRQ and HSPI_MISO.

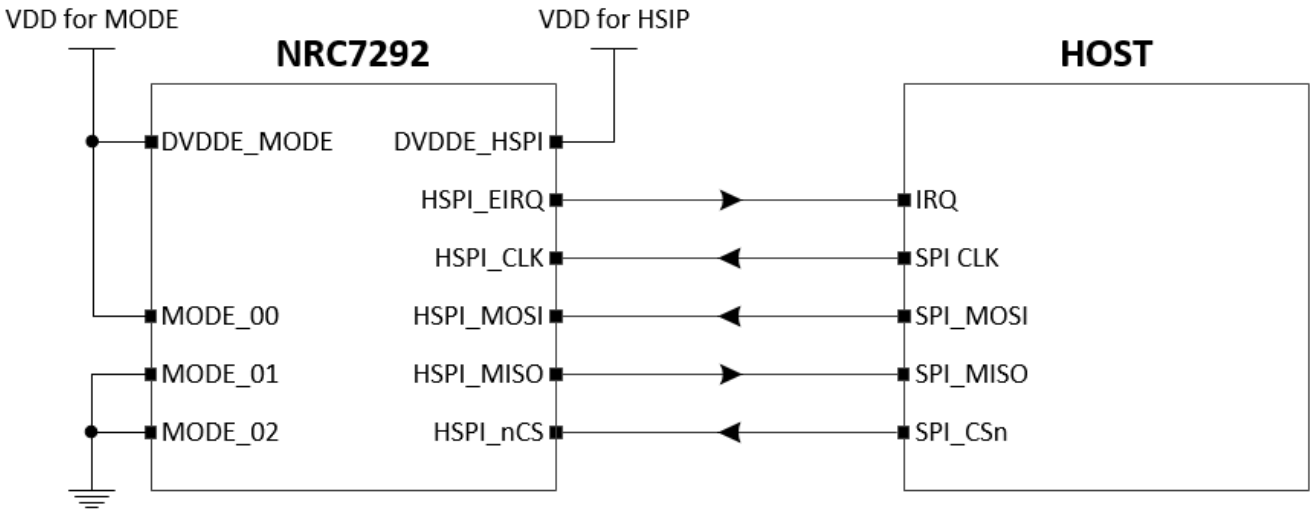

Figure 2.2 An example of H/W configuration for HSPI interface

SPI slave engine asserts HSPI_EIRQ interrupt when its TX QUEUE has data to send or the status of RX QUEUE is changed. Therefore, when the HSPI_EIRQ is asserted, the external host needs to check what triggers the interrupt after clearing the interrupt by reading EIRQ_CLEAR register(0x12).

As shown in Figure 2.3, SPI slave engine supports SPI mode 0 (CPOL = 0 & CPHA = 0). The HSPI_nCS must be held low for entire read/write cycle and must be taken high at least one clock period after the read/write cycle completed.

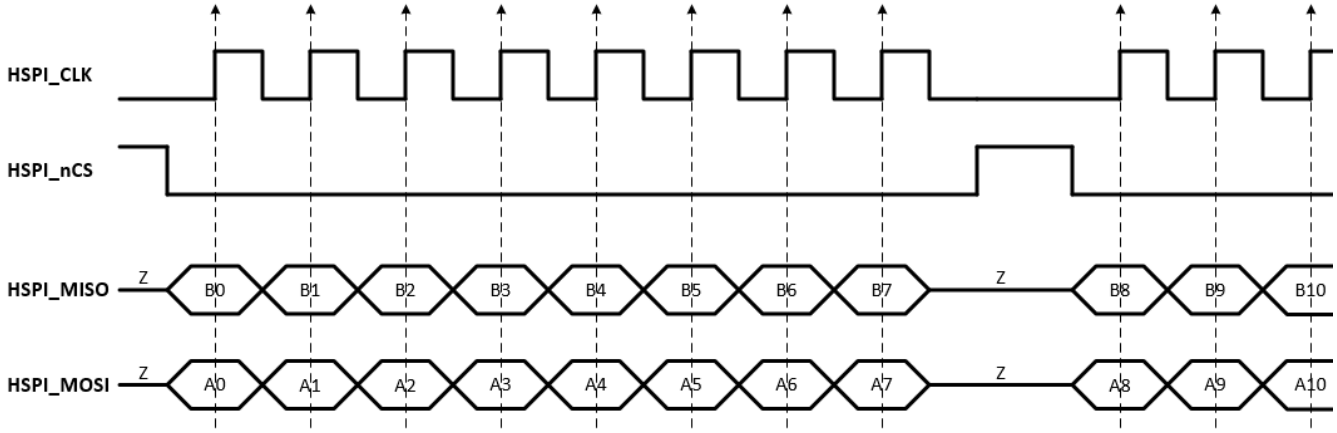

Figure 2.3 Timing diagram of HSPI interface

2.1 Single transfer mode

The single and burst transfer modes are defined. In single transfer mode, only one byte of data can be transferred from host to slave and vice versa. Figure 2.4 and Figure 2.5 represent the frame format for HSPI master write and read, respectively in single transfer mode. The single transfer mode starts with the command period and ends with the response period. The command period is composed of a 32-bits argument and a 16-bits cyclic redundancy check (CRC) part. The host (HSPI master) should check the acknowledgment (ACK) before sending the next command when writing data to HSPI slave. The ACK in response period is sent from HSPI slave to HSPI master to inform that it receives command correctly. When reading data, the HSPI slave transmit the data in response period.

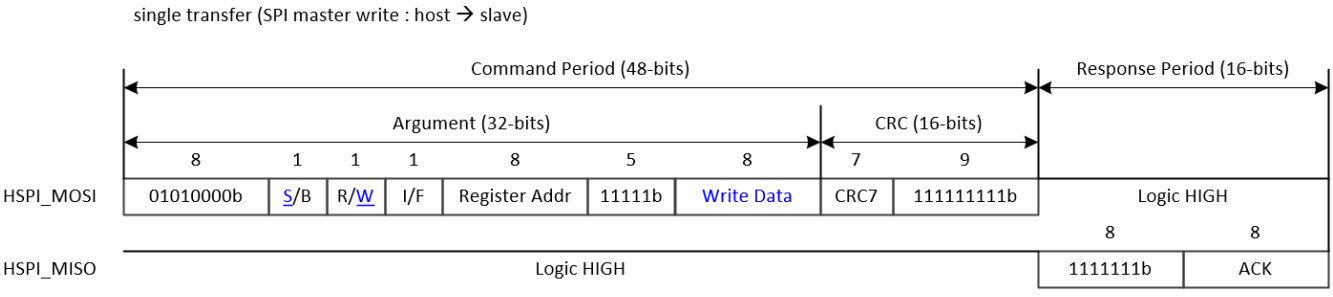

Figure 2.4 Frame format of HSPI master write in single transfer mode

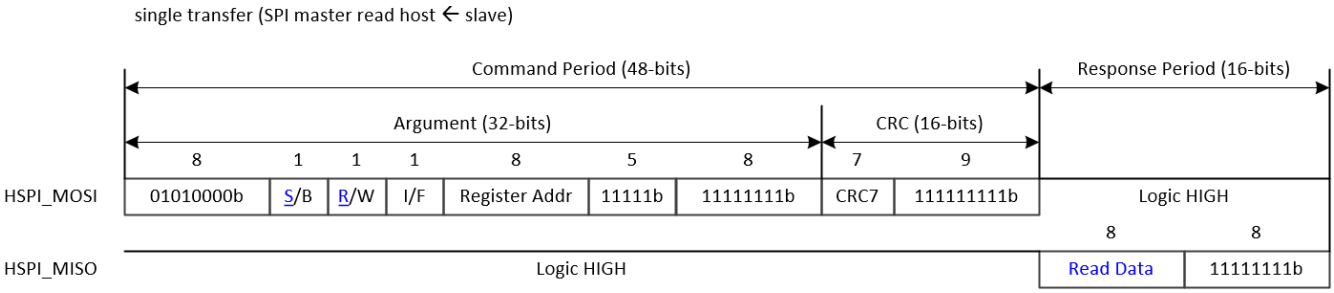

Figure 2.5 Frame format of HSPI master read in single transfer mode

Table 2.2 Field description of frame in single transfer mode

Field		# bits	Description
Argument	01010000b	8	header of the command period
	S/B	1	0 : single transfer 1 : burst transfer
	R/W	1	0 : read 1 : write
	I/F	1	0 : address increment 1 : address fix
	Register Addr	8	register address to access
	11111b	5	stuff bits
	Write Data / Stuff bits	8	when R/W = 0 : 0xFF when R/W = 1 : write data
CRC	CRC7	7	7 bits CRC calculation based on 32-bits argument
	111111111b	9	stuff bits
Response	Read Data / Stuff bits	8	when R/W = 0 : read data @ register address when R/W = 1 : 0xFF
	ACK	8	0x47

2.2 Burst transfer mode

Burst transfer mode also starts with the command period followed by a response period like the single transfer mode. However, additional data period follows the response period as shown in Figure 2.6 and Figure 2.7. In the data period, HSPI master writes or reads a number of data.

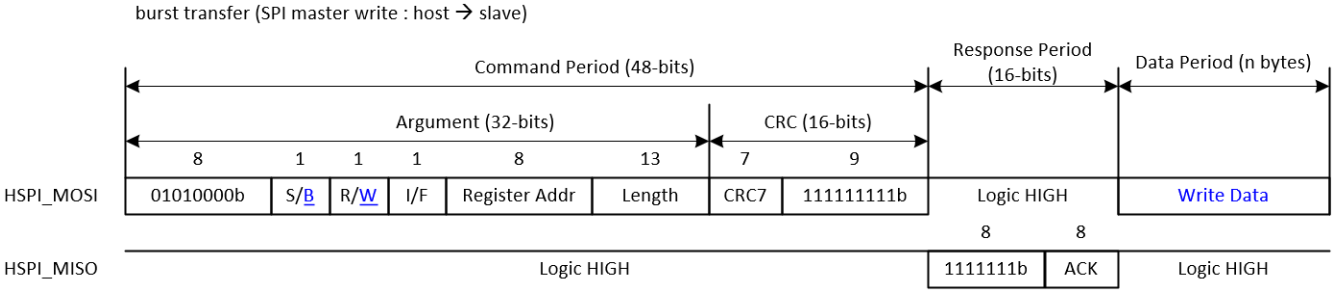

Figure 2.6 Frame format of HSPI master write in burst transfer mode

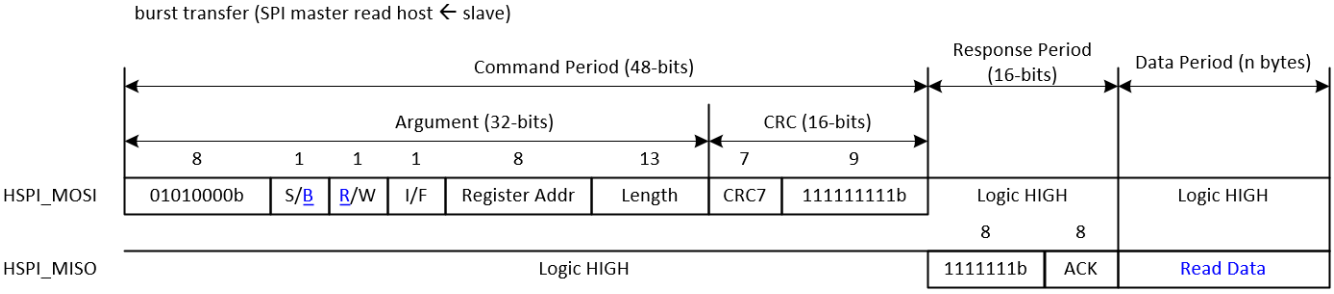

Figure 2.7 Frame format of HSPI master read in burst transfer mode

The frame structure of burst transfer mode is almost the same as the structure of the single transfer mode except for the length field at the end of the 32-bit argument. This 13-bit length field can represent up to 8K bytes and indicates the data size in byte.

Table 2.3 Field description of frame in burst transfer mode

Field		# bits	Description
Argument	01010000b	8	header of the command period
	S/B	1	0 : single transfer 1 : burst transfer
	R/W	1	0 : read 1 : write
	I/F	1	0 : address increment 1 : address fix
	Register Addr	8	register address to access
	Length	13	data length in byte
CRC	CRC7	7	7 bits CRC calculation based on 32-bits argument
	111111111b	9	stuff bits
Response	Read Data / Stuff bits	8	when R/W = 0 : read data @ register address when R/W = 1 : 0xFF
	ACK	8	0x47

2.3 Interrupt handler

The SPI slave engine asserts HSPI_EIRQ when the following events are occurred.

- TX credit updated
- TX QUEUE status updated
- RX QUEUE status updated

The TX credit is the number of available chucked buffer for each access category (AC) in the client. The host can afford to send data (IEEE 802.11ah uplink data) to the client only when it has TX credit. The TX (RX) QUEUE is a first in first out (FIFO) inside SPI slave engine. This QUEUE has multiple slots in it. The SPI slave engine asserts HSPI_EIRQ as well when the number of available slot is updated.

Figure 2.8 shows a basic operation of interrupt handling. Once the HSPI_EIRQ is asserted, the host needs to check what triggers the interrupt after clearing the interrupt by reading EIRQ_CLEAR (0x12) register. If the interrupt is caused by TX credit update, then the host initiates the TX Frame Handler to push TX data into TX-Queue (different from TX QUEUE in SPI slave engine). If TX QUEUE in SPI slave engine triggers the interrupt, the host updates the number of RX-slot and starts RX Thread Handler to receive RX data via HSPI interface. If there is an available slot in RX QUEUE in SPI slave engine, the host updates TX-slot, then TX Work Queue thread starts to send TX data to HSPI slave.

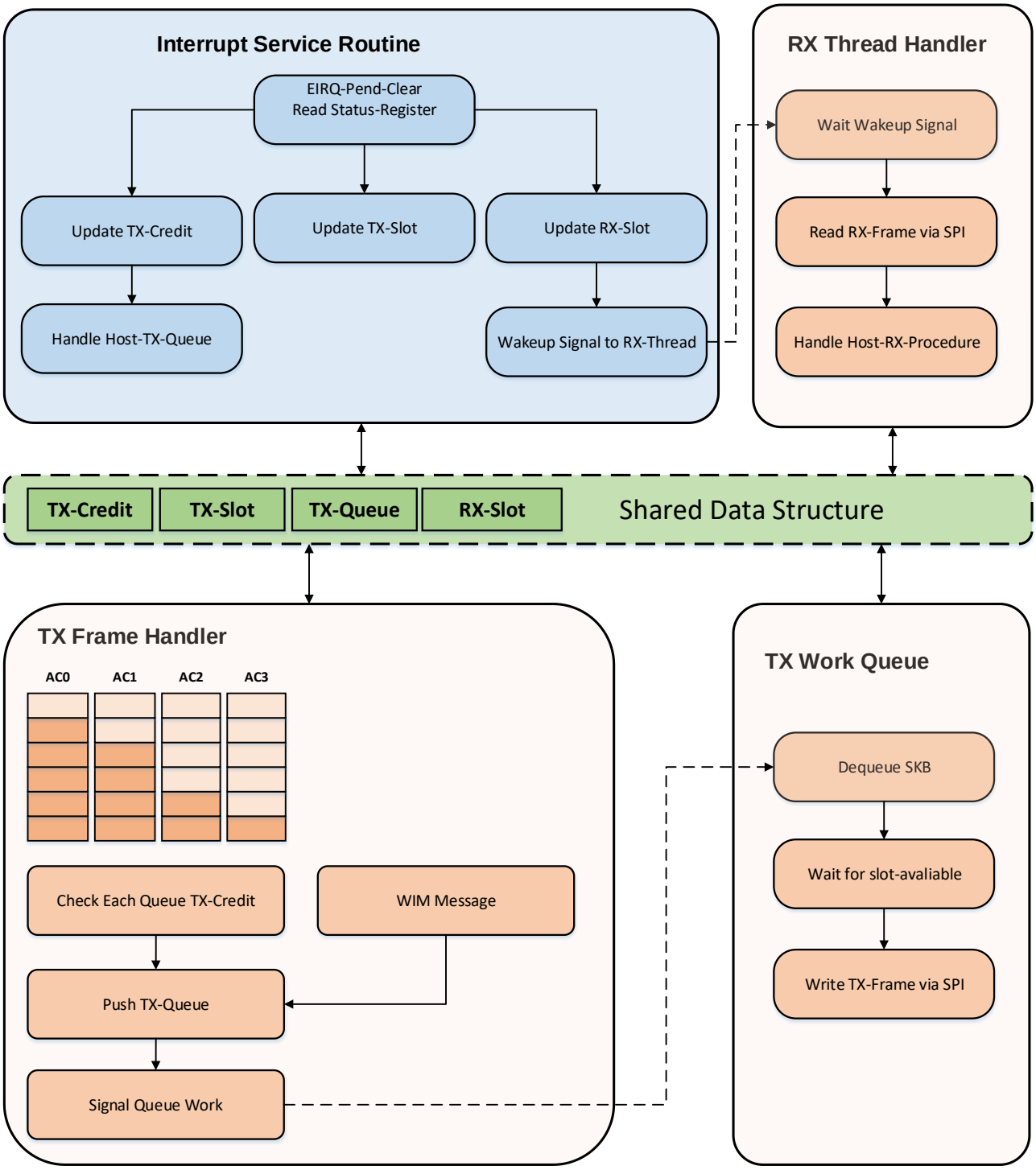

Figure 2.8 Handle interrupt

2.4 Register map

Table 2.4 Registers for HSPI

Address	Register	R/W	Description
System register			
0x00	WAKEUP	W	HSPI wakes up when writing 0x79
0x01	DEV_RESET	W	HSPI reset when writing 0xC8
IRQ register			
0x10	EIRQ_MODE	R/W	[07:03] Reserved [02] EIRQ_IO_EN (1: IO enable) [01] EIRQ_MODE1 (0: level trig., 1: edge trig.) [00] EIRQ_MODE2 (0: active low, 1: active high)
0x11	EIRQ_ENABLE	R/W	[07:04] Reserved [03] DEV_SLEEP_IRQ (1: enable) [02] DEV_READY_IRQ (1: enable) [01] TX_QUEUE_IRQ (1: enable) [00] RX_QUEUE_IRQ (1: enable)
0x12	EIRQ_CLEAR	R	EIRQ clear register Clear all interrupt when read
0x13	EIRQ_STATUS	R	[07:04] Reserved [03] DEV_SLEEP_STATUS [02] DEV_READY_STATUS [01] TX_QUEUE_STATUS [00] RX_QUEUE_STATUS
0x14	QUEUE_STATUS	R	[07:00] TX QUEUE status [47:40]
0x15			[07:00] TX QUEUE status [39:32]
0x16			[07:00] TX QUEUE status [31:24]
0x17			[07:00] TX QUEUE status [23:16]
0x18			[07:00] TX QUEUE status [15:08]
0x19			[07:00] TX QUEUE status [07:00]
0x1A			[07:00] RX QUEUE status [47:40]
0x1B			[07:00] RX QUEUE status [39:32]
0x1C			[07:00] RX QUEUE status [31:24]
0x1D			[07:00] RX QUEUE status [23:16]
0x1E			[07:00] RX QUEUE status [15:08]
0x1F			[07:00] RX QUEUE status [07:00]
Message register			
0x20	DEV_MSG_00	R	[07:00] Device message [31:24]
0x21			[07:00] Device message [23:16]
0x22			[07:00] Device message [15:08]

0x23			[07:00] Device message [07:00]
0x24	DEV_MSG_01	R	[07:00] Device message [31:24]
0x25			[07:00] Device message [23:16]
0x26			[07:00] Device message [15:08]
0x27			[07:00] Device message [07:00]
0x28	DEV_MSG_02	R	[07:00] Device message [31:24]
0x29			[07:00] Device message [23:16]
0x2A			[07:00] Device message [15:08]
0x2B			[07:00] Device message [07:00]
0x2C	DEV_MSG_03	R	[07:00] Device message [31:24]
0x2D			[07:00] Device message [23:16]
0x2E			[07:00] Device message [15:08]
0x2F			[07:00] Device message [07:00]
RX QUEUE register			
0x31	RXQUEUE_WINDOW	W	[07:00] received data from HSPI master (host)
TX QUEUE register			
0x41	TXQUEUE_WINDOW	R	[07:00] transmitted data to HSPI master (host)

3 Revision History

Revision No	Date	Comments
Ver 1.0	03/28/2019	Initial version for customer release created