Excelon™ Ultra QSPI F-RAM Library User Guide

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
www.cypress.com

1. Quad SPI Driver Details

This user guide summarizes the APIs written in spifi_fram (.h/.c) file along with the LPCExpresso/uVision Example project provided by Cypress Semiconductors. Each function has been supplemented by user comments to help understand the usage. Revision 1.0 of this release covers limited features of the device. The detailed feature set of Cypress' Quad SPI F-RAM can be found in the device datasheet. The APIs listed here are for enabling easy usage of the Quad SPI F-RAM features and are not an official release of driver support from Cypress Semiconductors. Users are encouraged to leverage these APIs for building their end applications.

Application Programming Interface

The QSPI F-RAM APIs are declared in \QSPI\Inc\spifi_fram.h file and declared in \QSPI\Src\spifi_fram.c. Following table provides summary of all the supported APIs for Revision 1.0 of this release. Unless specified explicitly the description is applicable only for the Quad SPI F-RAM device.

Sr No	API	Description	Notes
1	void FRAM_Write(spifiFRAM_T *,uint32_t Addr,uint32_t *writeBuff,uint32_t bytes);	This API performs write operation (Opcode 0x02) to the device	Device must be write enabled (FRAM_WriteEnable()) prior to writing into the device. This API performs write of length "bytes" bytes stored in array pointed by "*writeBuff", starting at location "Addr" in the F-RAM. The API assumes that the controller is in a known operating mode (SPI, DPI or QPI)
2	void FRAM_Read(spifiFRAM_T *,uint32_t Addr,uint32_t *writeBuff,uint32_t bytes);	This API performs read operation (Opcode 0x03) to the device	This API performs read of length "bytes" bytes stored in array pointed by "*writeBuff", starting at location "Addr" in the F-RAM. The API assumes that the controller is in a known operating mode (SPI, DPI or QPI)
3	void FRAM_WriteEnable(spifiFRAM_T *fram);	Issues WREN (0x06) opcode to the device	
4	void FRAM_WriteDisable(spifiFRAM_T *fram);	Issues WRDI (0x04) opcode to the device	
5	void FRAM_StatusRegisterWriteDisable(spifiFRAM_T *fram,Bool_T);	This API will set/reset the Status register write enable bit in Status Register.	All these MPNs will read/write into Status or configuration registers. A WREN command should be issued for a write operation into a register.
6	void FRAM_TBPROT(spifiFRAM_T *fram,Bool_T);	This API will set/reset the	

Sr No	API	Description	Notes
		TBPROT bit in Status Register.	
7	void FRAM_BlockProtect(spifiFRAM_T *fram,BLOCK_PROTECT_T BP);	This API will modify the block protection bits in Status Register.	
8	WIP FRAM_WIP(spifiFRAM_T *fram);	This API will return the value of WIP bit in Status Register.	
9	void FRAM_SetLatency(spifiFRAM_T *fram,uint8_t);	This API will update the memory read Latency values.	
10	uint8_t FRAM_GetLatency(spifiFRAM_T *fram);	This API will return memory read latency values.	
11	void FRAM_SetQuadMode(spifiFRAM_T *fram,Bool_T);	This API will enable/disable Quad mode of device	
12	uint8_t FRAM_GetQuadMode(spifiFRAM_T *fram);	This API will return the Quad Mode bit value in configuration register 1	
13	FRAM_ERR_T FRAM_SetQPI(spifiFRAM_T *fram,Bool_T);	This API will enable/disable QPI mode	
14	void FRAM_SetIO3R(spifiFRAM_T *fram,Bool_T);	This API will Enbale/disbale the IO3 reset function	
15	FRAM_ERR_T FRAM_SetDPI(spifiFRAM_T *fram,Bool_T);	This API will enable/disable DPI mode	
16	void FRAM_SetOuputImpedance(spifiFRAM_T *fram,uint8_t OI);	This API will update the Outputimpedence value in CR4 register	
17	void FRAM_SetDPDPOR(spifiFRAM_T *fram,Bool_T);	This API will Enable/Disable the Deep power down on POR feature.	
18	void FRAM_SetRegisterLatency(spifiFRAM_T *fram,uint8_t);	This API will update the register read latency value.	

Sr No	API	Description	Notes
19	uint8_t SpifiFram_config(spifiFRAM_T *fram);	This API will modify all the mode bits and reset the device into SPI mode. Read all the registers and update the static register values	The F-RAM device can be configured in several operating modes (SPI, DPI or QPI) and can have several register settings. This API is written specifically to implement a “Go Home” feature in case the controller faces a miss-match of operating mode. For eg: during a sudden power cycle, the F-RAM device will retain its state but the controller will execute a power-up routine and will not be able to communicate with the F-RAM device. It is recommended to implement similar function in end application
20	void FRAM_WriteSerialNumber(spifiFRAM_T *,uint32_t *);	This API will write the Serial number register with the user required value	SN Register is an 8-byte register. Read function will read the device’s serial number and store it in SN_Reg array of operating_mode structure The argument passed to Write function must be a pointer to an 8-byte array containing value of new Serial number
21	void FRAM_ReadSerialNumber(spifiFRAM_T *,uint32_t *);	This API will read the serial number register	

Revision History

Document Revision History

Document Title: ExcelonTM Ultra QSPI F-RAM Library User Guide Document Number: 001-00000 Rev **			
Revision	**Issue Date**	**Origin of Change**	**Description of Change**
**		VINI	New Spec