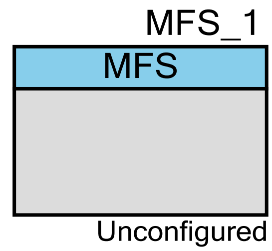
Multifunction Serial Interface (PDL_MFS)

1.0

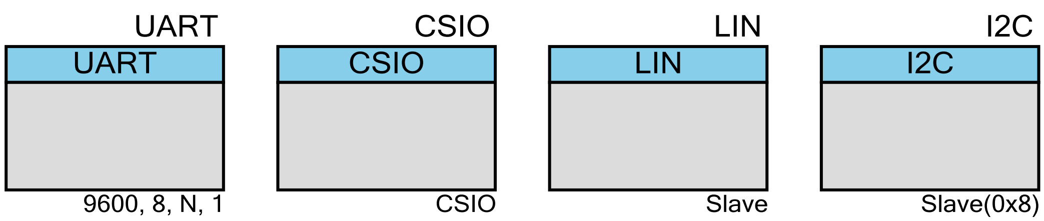


Features

- Configures the Multi-Function Serial (MFS) Interface to one of the following modes:
    - UART (Asynchronous normal serial interface)
    - Clock synchronous serial interface (SPI and I^2S can be supported)
    - LIN bus interface
    - I^2C bus interface

General Description

The Peripheral Driver Library (PDL) Multifunction Serial Interface (PDL_MFS) component is multifunction peripheral block that implements the following communication interfaces: UART, CSIO, LIN, I2C. Each is available as a pre-configured schematic macro in the PSoC Creator Component Catalog.



The base component in the catalog is setup in the unconfigured mode.

This component uses firmware drivers from the PDL_MFS module, which is automatically added to your project after a successful build.

When to Use a PDL_MFS Component

Use the PDL_MFS component to configure the initialization settings of the MFS peripheral block for UART, CSIO, LIN or I2C interface any time the device must connect to external buses.

Quick Start

Follow the quick start steps below to configure and initialize the MFS peripheral block.

1. Drag a PDL_MFS component from the Component Catalog FMx/Communication folder onto your schematic. The placed instance takes the name MFS_1.

2. Double-click to open the component's Configure dialog.
3. On the **Basic** tab set the configuration of the component.
4. Depending on the selected component configuration, will be added a tab with specific parameters.
5. Assign pins in your device using the Pin Editor. If you are creating a design for a development kit, refer the kit User Guide for suitable pin assignments.
6. Build the project to verify the correctness of your design. This will add the required PDL modules to the Workspace Explorer and generate configuration data for the MFS_1 instance.
7. In the *main.c* file, initialize the peripheral and start the application. The example assumes the MFS is configured in UART mode.

```
Mfs_Uart_Init(&MFS_1_HW, &MFS_1_Config);
MFS_1_SetPinFunc_SIN();
MFS_1_SetPinFunc_SOT();
Mfs_Uart_EnableFunc(&MFS_1_HW, UartTx);
Mfs_Uart_SendData(&MFS_1_HW, '1');
```

8. Build and program the device.

Component Parameters

The PDL_MFS component Configure dialog allows you to edit the configuration parameters for the component instance.

Basic Tab

This tab contains the component parameters used in the basic peripheral initialization settings.

Parameter Name	Description
bUseExtClk	Use an external clock on SCK pin
MFSConfig	Selects the MFS operating mode to one of the following modes: ▪ UART - configured to be in UART mode. ▪ CSIO - configured to be in CSIO mode. ▪ I2C - configured to be in I2C mode. ▪ LIN - configured to be in LIN mode. Unconfigured - This is the default mode. The component will not generate configuration structures to initialize the MFS block. The structures must be user provided.

FIFO Tab

This tab contains the FIFO configuration settings.

Parameter Name	Description
enFifoSel	Select the FIFO function
u8ByteCount1	Size of FIFO 1
u8ByteCount2	Size of FIFO 2

Interrupts Tab

This tab contains the Interrupts configuration settings.

Parameter Name	Description
bTouchNvic	Install interrupts in NVIC
bRxIrq	Receive interrupt enable
pfnRxIrqCb	Receive interrupt callback. Note: this generates a declaration only - USER must implement the function
bTxIrq	Transmit interrupt enable
pfnTxIrqCb	Transmit interrupt callback. Note: this generates a declaration only - USER must implement the function
bTxFifoIrq	Transmit FIFO interrupt enable
pfnTxFifoIrqCb	Transmit FIFO interrupt callback. Note: this generates a declaration only - USER must implement the function
bTxIdleIrq	Transmit idle interrupt enable
pfnTxIdleIrqCb	Transmit idle interrupt callback. Note: this generates a declaration only - USER must implement the function.
bCsErrIrq	Chip Select interrupt enable. It is visible when CSIO mode is selected
pfnCsErrIrqCb	Chip Select interrupt callback. Note: this generates a declaration only - USER must implement the function. It is visible when CSIO mode is selected
bSerialTimerIrq	Serial Timer interrupt enable. It is visible when CSIO mode is selected
pfnSerialTimerIrq	Serial Timer interrupt callback. Note: this generates a declaration only - USER must implement the function. It is visible when CSIO mode is selected
bWaitSelection	Generate interrupt after ACK (wait) or before ACK. It is visible when I2C mode is selected
bStopDetectIrq	Stop condition interrupt enable. It is visible when I2C mode is selected
pfnStopDetectIrqCb	Stop condition interrupt callback. Note: this generates a declaration only - USER must implement the function. It is visible when I2C mode is selected
bLinBreakIrq	LIN break field interrupt enable. It is visible when LIN mode is selected

Parameter Name	Description
pfnLinBreakIrq	LIN break field interrupt callback. Note: this generates a declaration only - USER must implement the function. It is visible when LIN mode is selected

Chip Select Tab

This tab contains the Chip Select configuration settings. Visible when CSIO mode is selected.

Parameter Name	Description
bActiveHold	Hold active status until all bytes are transferred
bScs*n*En	SCS*n* Enable, n = 0..3
enCsClkDiv	CS serial timer clock divider
enCsEndPin	Chip select pin selection
enCsStartPin	Chip select pin selection
enLevel	Active level selection, only apply to SCS0
u16CsDeselectTime	Chip select deselection time (gap between two bytes's transfer)
u8CsHoldDelayTime	Chip select hold time
u8CsSetupDelayTime	Chip select setup time
u8Scs*n*TransferByteCnt	SCS*n* Transfer byte count, n = 0..3

CSIO Tab

This tab contains the general CSIO configuration settings. Visible when CSIO mode is selected.

Parameter Name	Description
bInvertClk	SCK Mark level high (false) or low (true)
enActMode	CSIO Active mode
enCsioBitDirection	CSIO bit direction
enCsioDataLength	CSIO data length
enCsioMsMode	CSIO Master or Slave mode
enSyncWaitTime	CSIO wait time insertion
u32CsioBaudRate	Baud rate in bps

Serial Timer Tab

This tab contains the Serial Timer configuration settings in CSIO mode. Visible when CSIO mode is selected.

Parameter Name	Description
enStClkDiv	CSIO serial timer clock divider
u16CompareValue	Compare value to control transfer start
u8TransferByteCnt	Transfer byte count

UART Tab

This tab contains the UART configuration settings. Visible when UART mode is selected.

Parameter Name	Description
bHwFlow	Hardware flow control
bInvertData	Encode data as NRZ
enMode	UART Mode
enParity	UART parity
enStopBit	UART stop bits
enUartBitDirection	UART bit direction
enUartDataLength	UART data length
u32UartBaudRate	Baud rate in bps

I2C Tab

This tab contains the I2C configuration settings. Visible when I2C mode is selected.

Parameter Name	Description
bDmaEnable	Use DMA
enI2cMsMode	I2C Master or Slave mode
u32I2cBaudRate	Baud rate in bps
u8SlaveAddress	I2C Slave address (not used in Master mode)
u8SlaveMaskAddr	I2C Slave Mask (not used in Master mode)

LIN Tab

This tab contains the LIN configuration settings. Visible when LIN mode is selected.

Parameter Name	Description
enBreakLength	Break Generation Length (only applicable in LIN master mode)
enDelimiterLength	Break Delimiter Length (only applicable in LIN master mode)
enLinMsMode	LIN Master or Slave mode
enStopBits	LIN stop bit length
u32LinBaudRate	Baud rate in bps

Component Usage

After a successful build, firmware drivers from the PDL_MFS module are added to your project in the pdl/driver/mfs folder. Pass the generated data structures to the associated PDL functions in your application initialization code to configure the peripheral.

Generated Data

The PDL_MFS component populates the following peripheral initialization data structure(s). The generated code is placed in C source and header files that are named after the instance of the component (e.g. *MFS_1_config.c*). Each variable is also prefixed with the instance name of the component.

Data Structure Type	Name	Description
stc_mfs_fifo_config_t	MFS_1_FifoConfig	FIFO configuration structure
stc_mfs_uart_config_t	MFS_1_Config	UART configuration structure. Generated if the UART mode selected
stc_mfs_i2c_config_t	MFS_1_Config	I2C configuration structure. Generated if the I2C mode selected
stc_csio_serial_timer_t	MFS_1_SerialTimer	Serial timer configuration. Generated if the CSIO mode selected
stc_csio_cs_t	MFS_1_CsConfig	Chip selection configuration. Generated if the CSIO mode selected
stc_mfs_csio_config_t	MFS_1_Config	CSIO configuration structure. Generated if the CSIO mode selected
stc_mfs_lin_config_t	MFS_1_Config	LIN configuration structure. Generated if the LIN mode selected

Once the component is initialized, the application code should use the peripheral functions provided in the referenced PDL files. Refer to the PDL documentation for the list of provided API functions. To access this document, right-click on the component symbol on the schematic and choose “**Open API Documentation…**” option in the drop-down menu.

Preprocessor Macros

The PDL_MFS component generates the following preprocessor macro(s). Note that each macro is prefixed with the instance name of the component (e.g. “MFS_1”).

Macro	Description
MFS_1_HW	Hardware pointer to the block instance in the device. This should be used in all API calls when specifying the block to access.
MFS_1_SetPinFunc_SIN()	Macro to assign MFS SIN signal in the device pin.
MFS_1_SetPinFunc_SOT()	Macro to assign MFS SOT signal in the device pin.
MFS_1_SetPinFuncSCK()	Macro to assign MFS SCK signal in the device pin.
MFS_1_SetPinFunc_SCS0()	Macro to assign MFS SCS0 signal in the device pin. Generated only in CSIO mode.
MFS_1_SetPinFunc_SCS1()	Macro to assign MFS SCS1 signal in the device pin. Generated only in CSIO mode.
MFS_1_SetPinFunc_SCS2()	Macro to assign MFS SCS2 signal in the device pin. Generated only in CSIO mode.
MFS_1_MFS_CONFIG	MFS interface. Is set to one of the following values based on the MFSConfig parameter option: MFS_1_MFS_UNCONFIGURED , MFS_1_MFS_UART, MFS_1_MFS_I2C, MFS_1_MFS_CSIO or MFS_1_MFS_LIN.

Data in RAM

The generated data may be placed in flash memory (const) or RAM. The former is the more memory-efficient choice if you do not wish to modify the configuration data at run-time. Under the **Built-In** tab of the Configure dialog set the parameter CONST_CONFIG to make your selection. The default option is to place the data in flash.

Interrupt Support

If the PDL_MFS component is specified to trigger interrupts, it will generate the callback function declaration that will be called from the MFS ISR. The user is then required to provide the actual callback code. If a null string is provided the struct is populated with zeroes and the callback declaration is not generated. In that case it is the user’s responsibility to modify the struct in firmware.

The component generates the following function declarations.

Function Callback	Description
MFS_1_TxIrqCb	TX interrupt callback function
MFS_1_RxIrqCb	RX interrupt callback function
MFS_1_TxIdleIrqCb	TX idle interrupt callback function

Function Callback	Description
MFS_1_TxFifoIrqCb	TX FIFO interrupt callback function
CsErrIrqCb	Chip Select interrupt callback. Note: this generates a declaration only - USER must implement the function. This interrupt callback function may be generated only in CSIO mode.
SerialTimerIrqCb	Serial Timer interrupt callback. Note: this generates a declaration only - USER must implement the function. This interrupt callback function may be generated only in CSIO mode.
StopDetectIrqCb	Stop condition interrupt callback. Note: this generates a declaration only - USER must implement the function. This interrupt callback function may be generated only in I2C mode.
TxLinBreakIrqCb	LIN break field interrupt callback. Note: this generates a declaration only - USER must implement the function. This interrupt callback function may be generated only in LIN mode.

Code Examples and Application Notes

There are numerous code examples that include schematics and example code available online at the Cypress Code Examples web page.

Cypress also provides a number of application notes describing how FMx devices can be integrated into your design. You can access the Cypress Application Notes search web page at www.cypress.com/appnotes.

Resources

The PDL_MFS component uses the Multi-Function Serial Interface (MFS) peripheral block.

References

- FM0+ Family of 32-bit ARM® Cortex®-M0+ Microcontrollers Peripheral Manuals
- Cypress FM0+ Family of 32-bit ARM® Cortex®-M0+ Microcontrollers

Component Changes

This section lists the major changes in the component from the previous version.

Version	Description of Changes	Reason for Changes / Impact
1.0.a	Minor datasheet edits.	
1.0	Initial Version	