FcbExtractor.py - usage

Input parameter:

- the .s19 generated from a build including the FCB table

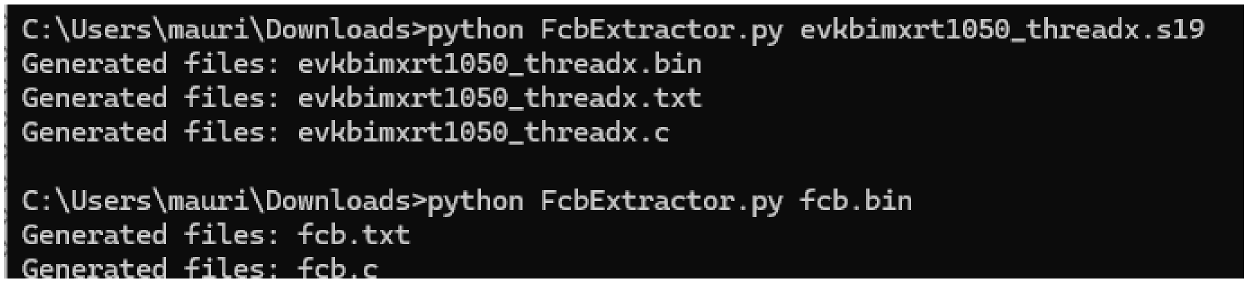
  or

- the .bin FCB table data generated with Secure Provisioning Tool (SPT)

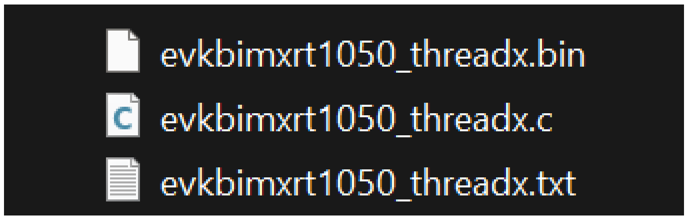
Output:

- the Serial NOR configuration block including the FCB (Flexspi Configuration Block) in the following formats:
  - .bin (if a .s19 is provided as input parameter). Same format managed by the SPT.
  - .txt containing an encoded text format array of hex data (ready to be included in a c array)
  - .c a flexspi_nor_config_t stuct ready too be included in the source code

The c array obtained with this extraction and conversion process from .s19 can be compared with the original one present in the source code that generated the .s19 file taken as input, for validation.

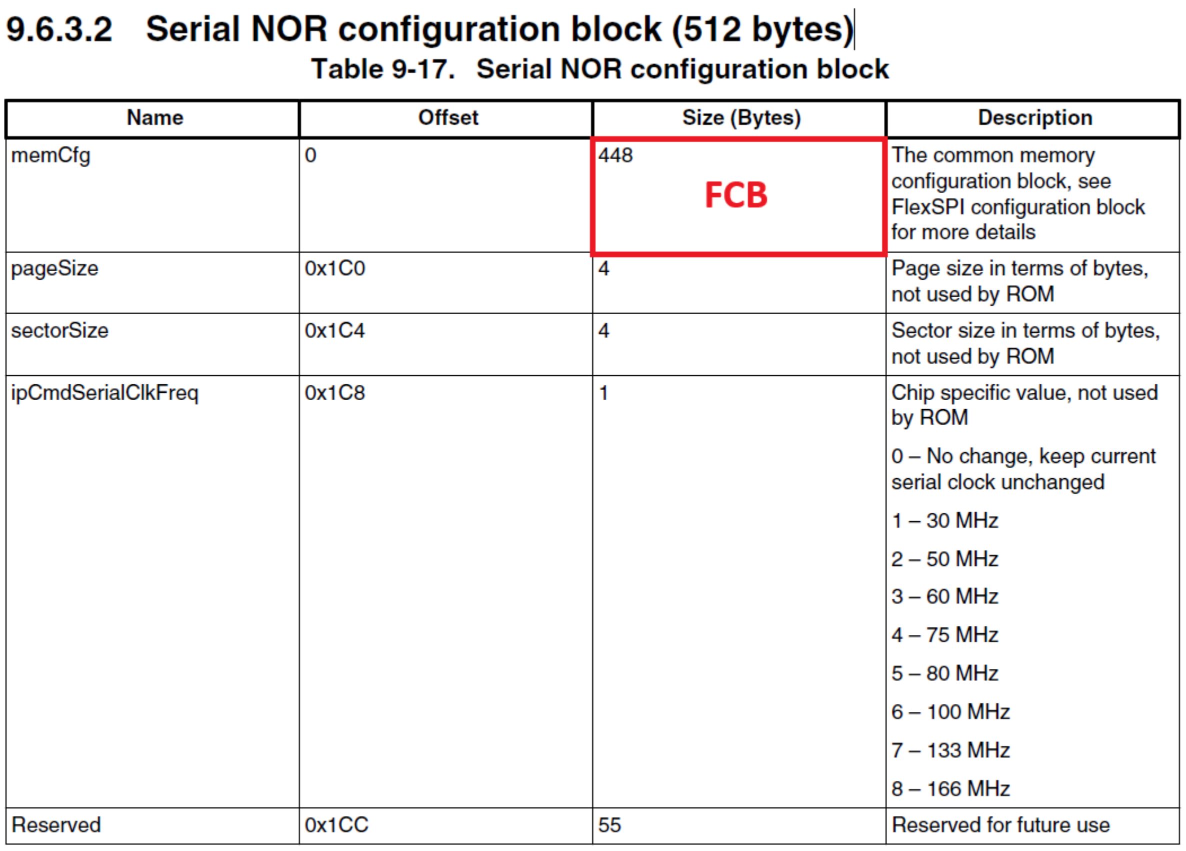


```
C:\Users\mauri\Downloads>python FcbExtractor.py evkbimxrt1050_threadx.s19
Generated files: evkbimxrt1050_threadx.bin
Generated files: evkbimxrt1050_threadx.txt
Generated files: evkbimxrt1050_threadx.c

C:\Users\mauri\Downloads>python FcbExtractor.py fcb.bin
Generated files: fcb.txt
Generated files: fcb.c
```



evkbimxrt1050_threadx.bin
evkbimxrt1050_threadx.c
evkbimxrt1050_threadx.txt

Further information



9.6.3.2 Serial NOR configuration block (512 bytes)

Table 9-17. Serial NOR configuration block

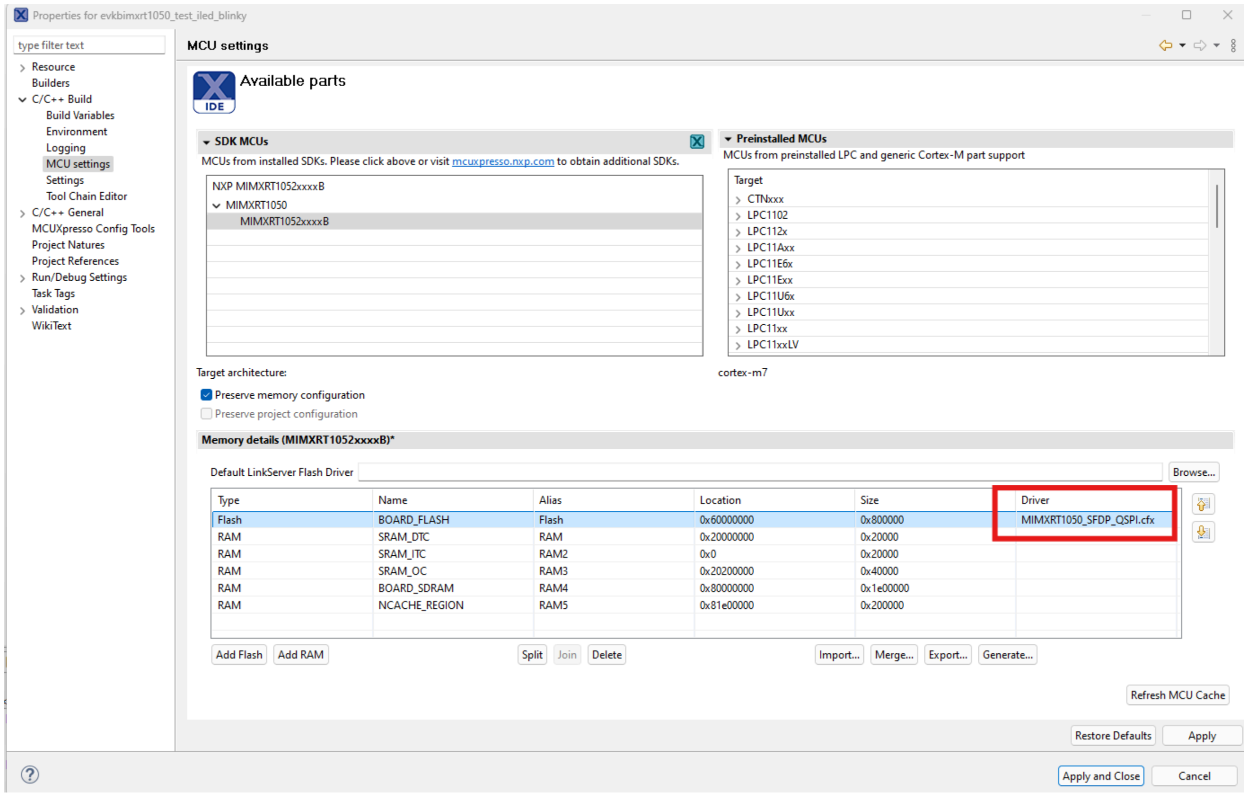
Name	Offset	Size (Bytes)	Description
memCfg	0	448 FCB	The common memory configuration block, see FlexSPI configuration block for more details
pageSize	0x1C0	4	Page size in terms of bytes, not used by ROM
sectorSize	0x1C4	4	Sector size in terms of bytes, not used by ROM
ipCmdSerialClkFreq	0x1C8	1	Chip specific value, not used by ROM 0 – No change, keep current serial clock unchanged 1 – 30 MHz 2 – 50 MHz 3 – 60 MHz 4 – 75 MHz 5 – 80 MHz 6 – 100 MHz 7 – 133 MHz 8 – 166 MHz
Reserved	0x1CC	55	Reserved for future use

- Nor Serial Flash Datasheet: with special reference to the implemented commands

In light of the information contained in the RM, what has been generated is not simply an FCB to manage the external serial NOR Flash, but a 512 byte structure, called “Serial NOR configuration block” which includes it in its “memCfg” field.

FCB Block Synthesis and link with the application

This is how to act on the preprocessor directives and the linker to ensure that the application includes the external flash and the related FCB block to initialize it and be able to perform XIP:



MCU settings

Available parts

SDK MCUs

MCUs from installed SDKs. Please click above or visit mcuxpresso.nxp.com to obtain additional SDKs.

NXP MIMXRT1052xxxxB
MIMXRT1050
MIMXRT1052xxxxB

Preinstalled MCUs

MCUs from preinstalled LPC and generic Cortex-M part support

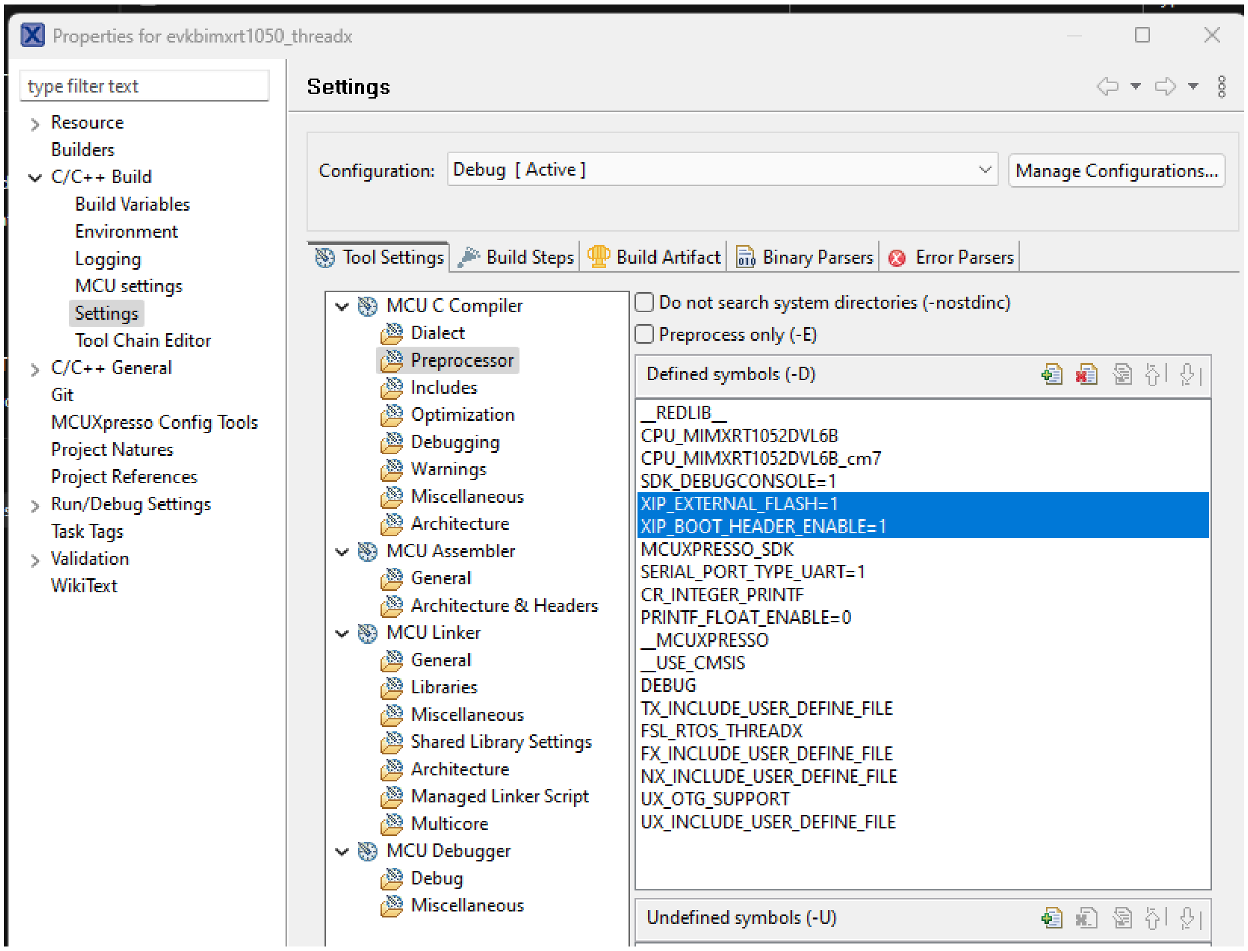
Target architecture: cortex-m7

☑ Preserve memory configuration
☐ Preserve project configuration

Memory details (MIMXRT1052xxxxB)*

Default LinkServer Flash Driver

Type	Name	Alias	Location	Size	Driver
Flash	BOARD_FLASH	Flash	0x60000000	0x800000	MIMXRT1050_SFDP_QSPI.cfx
RAM	SRAM_DTC	RAM	0x20000000	0x20000	
RAM	SRAM_ITC	RAM2	0x0	0x20000	
RAM	SRAM_OC	RAM3	0x20200000	0x40000	
RAM	BOARD_SDRAM	RAM4	0x80000000	0x1e00000	
RAM	NCACHE_REGION	RAM5	0x81e00000	0x200000	



Configuration: Debug [Active]

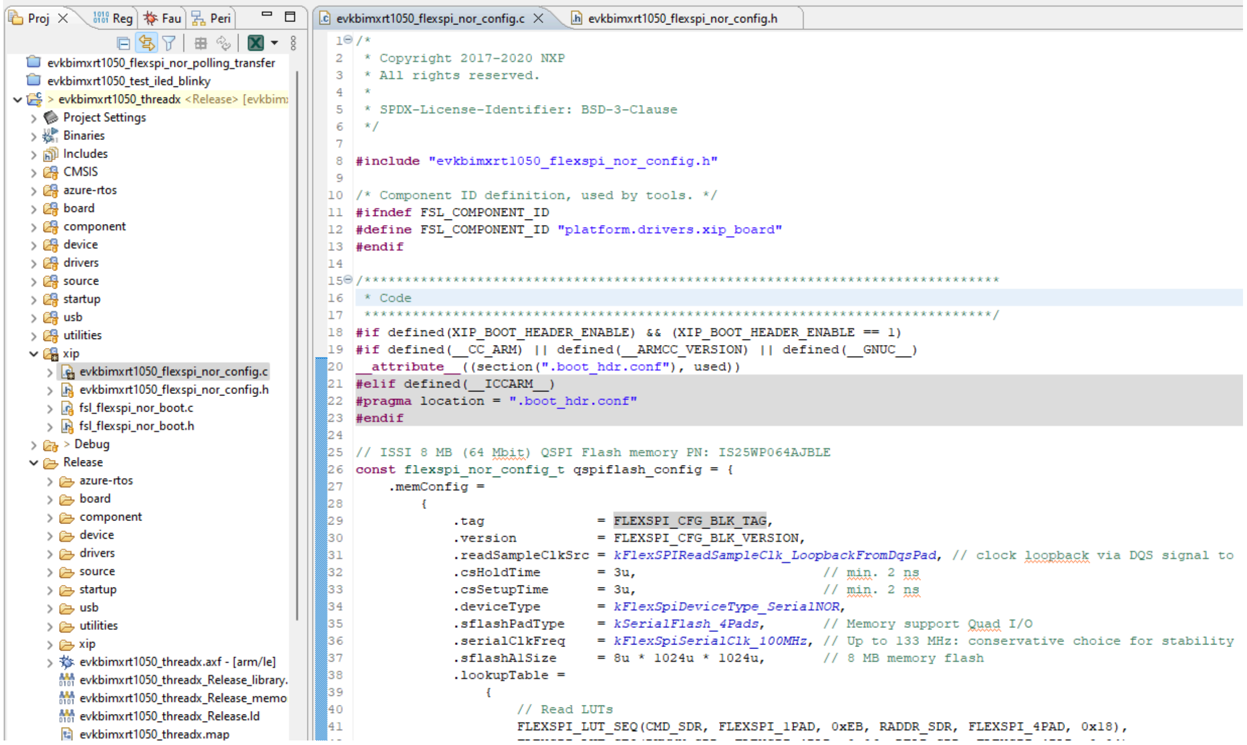
☐ Do not search system directories (-nostdinc)
☐ Preprocess only (-E)

Defined symbols (-D)

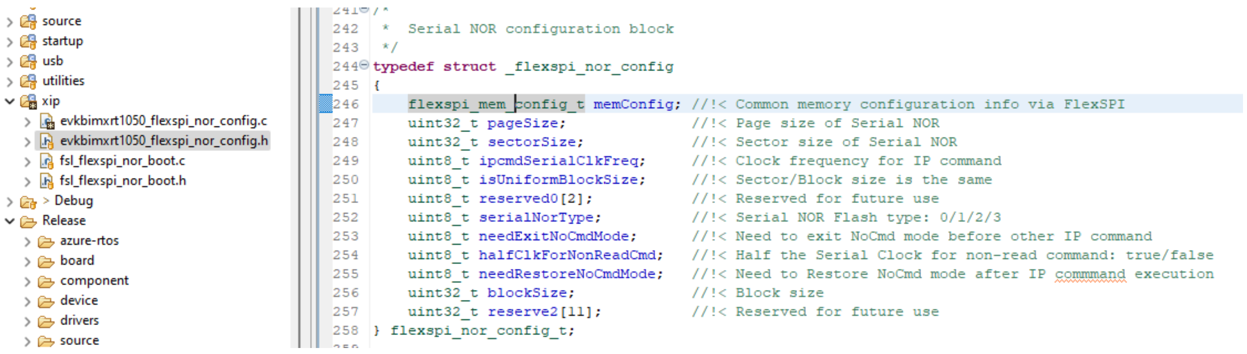
```
__REDLIB__
CPU_MIMXRT1052DVL6B
CPU_MIMXRT1052DVL6B_cm7
SDK_DEBUGCONSOLE=1
XIP_EXTERNAL_FLASH=1
XIP_BOOT_HEADER_ENABLE=1
MCUXPRESSO_SDK
SERIAL_PORT_TYPE_UART=1
CR_INTEGER_PRINTF
PRINTF_FLOAT_ENABLE=0
__MCUXPRESSO
__USE_CMSIS
DEBUG
TX_INCLUDE_USER_DEFINE_FILE
FSL_RTOS_THREADX
FX_INCLUDE_USER_DEFINE_FILE
NX_INCLUDE_USER_DEFINE_FILE
UX_OTG_SUPPORT
UX_INCLUDE_USER_DEFINE_FILE
```

Undefined symbols (-U)

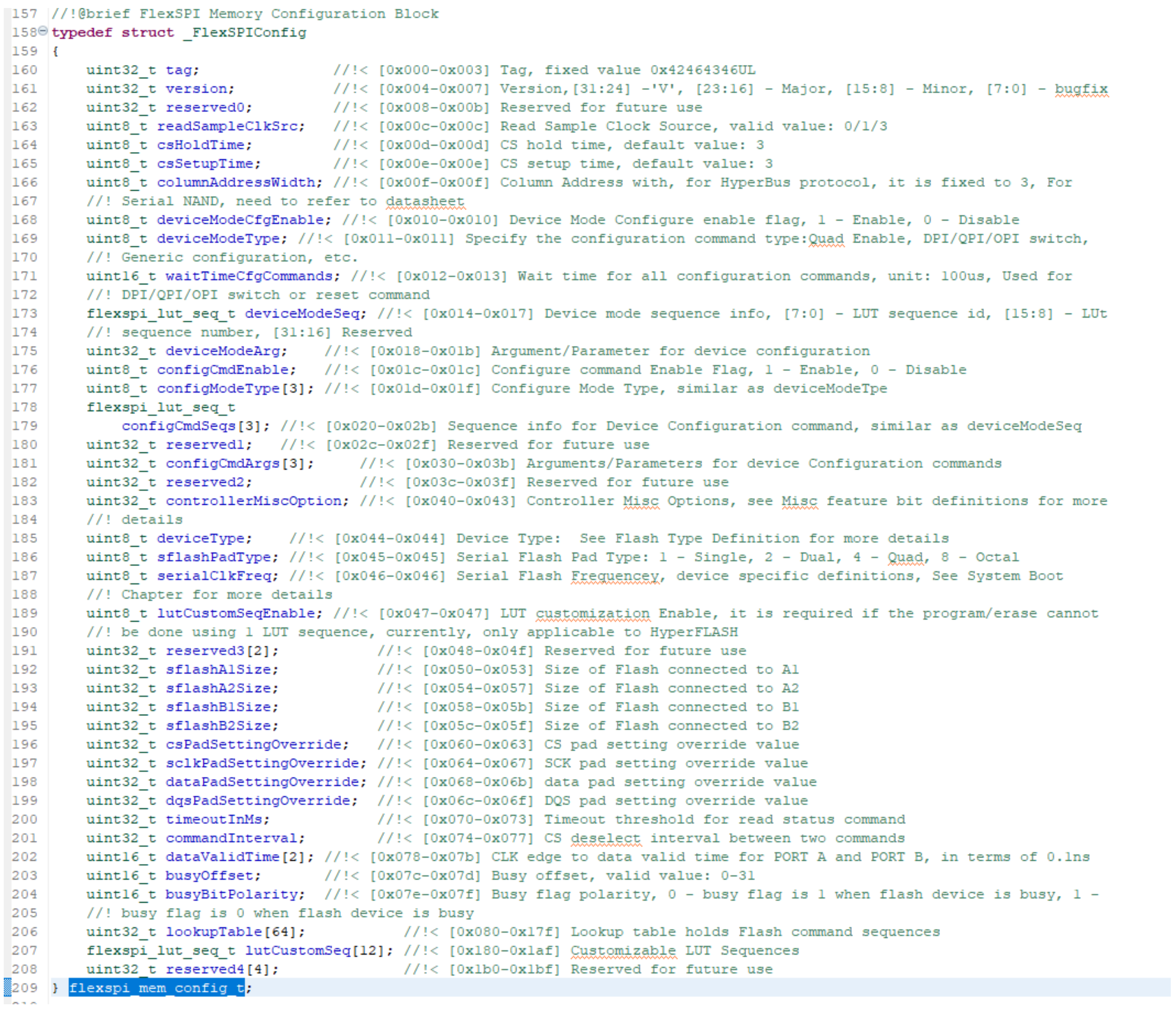
The structure on which to act is the following:


Proj
Reg
Fau
Peri
evkbimxrt1050_flexspi_nor_config.c
evkbimxrt1050_flexspi_nor_config.h
evkbimxrt1050_flexspi_nor_polling_transfer
evkbimxrt1050_test_iled_blinky
evkbimxrt1050_threadx <Release> [evkbim
Project Settings
Binaries
Includes
CMSIS
azure-rtos
board
component
device
drivers
source
startup
usb
utilities
xip
evkbimxrt1050_flexspi_nor_config.c
evkbimxrt1050_flexspi_nor_config.h
fsl_flexspi_nor_boot.c
fsl_flexspi_nor_boot.h
Debug
Release
azure-rtos
board
component
device
drivers
source
startup
usb
utilities
xip
evkbimxrt1050_threadx.axf - [arm/le]
evkbimxrt1050_threadx_Release_library.
evkbimxrt1050_threadx_Release_memo
evkbimxrt1050_threadx_Release.ld
evkbimxrt1050_threadx.map

```c
/*
 * Copyright 2017-2020 NXP
 * All rights reserved.
 *
 * SPDX-License-Identifier: BSD-3-Clause
 */

#include "evkbimxrt1050_flexspi_nor_config.h"

/* Component ID definition, used by tools. */
#ifndef FSL_COMPONENT_ID
#define FSL_COMPONENT_ID "platform.drivers.xip_board"
#endif

/*******************************************************************************
 * Code
 ******************************************************************************/
#if defined(XIP_BOOT_HEADER_ENABLE) && (XIP_BOOT_HEADER_ENABLE == 1)
#if defined(__CC_ARM) || defined(__ARMCC_VERSION) || defined(__GNUC__)
__attribute__((section(".boot_hdr.conf"), used))
#elif defined(__ICCARM__)
#pragma location = ".boot_hdr.conf"
#endif

// ISSI 8 MB (64 Mbit) QSPI Flash memory PN: IS25WP064AJBLE
const flexspi_nor_config_t qspiflash_config = {
    .memConfig =
        {
            .tag              = FLEXSPI_CFG_BLK_TAG,
            .version          = FLEXSPI_CFG_BLK_VERSION,
            .readSampleClkSrc = kFlexSPIReadSampleClk_LoopbackFromDqsPad, // clock loopback via DQS signal to
            .csHoldTime       = 3u,                          // min. 2 ns
            .csSetupTime      = 3u,                          // min. 2 ns
            .deviceType       = kFlexSpiDeviceType_SerialNOR,
            .sflashPadType    = kSerialFlash_4Pads,          // Memory support Quad I/O
            .serialClkFreq    = kFlexSpiSerialClk_100MHz, // Up to 133 MHz: conservative choice for stability
            .sflashA1Size     = 8u * 1024u * 1024u,          // 8 MB memory flash
            .lookupTable =
                {
                    // Read LUTs
                    FLEXSPI_LUT_SEQ(CMD_SDR, FLEXSPI_1PAD, 0xEB, RADDR_SDR, FLEXSPI_4PAD, 0x18),
```



```c
/*
 *  Serial NOR configuration block
 */
typedef struct _flexspi_nor_config
{
    flexspi_mem_config_t memConfig; //!< Common memory configuration info via FlexSPI
    uint32_t pageSize;              //!< Page size of Serial NOR
    uint32_t sectorSize;            //!< Sector size of Serial NOR
    uint8_t ipcmdSerialClkFreq;     //!< Clock frequency for IP command
    uint8_t isUniformBlockSize;     //!< Sector/Block size is the same
    uint8_t reserved0[2];           //!< Reserved for future use
    uint8_t serialNorType;          //!< Serial NOR Flash type: 0/1/2/3
    uint8_t needExitNoCmdMode;      //!< Need to exit NoCmd mode before other IP command
    uint8_t halfClkForNonReadCmd;   //!< Half the Serial Clock for non-read command: true/false
    uint8_t needRestoreNoCmdMode;   //!< Need to Restore NoCmd mode after IP commmand execution
    uint32_t blockSize;             //!< Block size
    uint32_t reserve2[11];          //!< Reserved for future use
} flexspi_nor_config_t;
```



```c
//!@brief FlexSPI Memory Configuration Block
typedef struct _FlexSPIConfig
{
    uint32_t tag;               //!< [0x000-0x003] Tag, fixed value 0x42464346UL
    uint32_t version;           //!< [0x004-0x007] Version,[31:24] -'V', [23:16] - Major, [15:8] - Minor, [7:0] - bugfix
    uint32_t reserved0;         //!< [0x008-0x00b] Reserved for future use
    uint8_t readSampleClkSrc;   //!< [0x00c-0x00c] Read Sample Clock Source, valid value: 0/1/3
    uint8_t csHoldTime;         //!< [0x00d-0x00d] CS hold time, default value: 3
    uint8_t csSetupTime;        //!< [0x00e-0x00e] CS setup time, default value: 3
    uint8_t columnAddressWidth; //!< [0x00f-0x00f] Column Address with, for HyperBus protocol, it is fixed to 3, For
    //! Serial NAND, need to refer to datasheet
    uint8_t deviceModeCfgEnable; //!< [0x010-0x010] Device Mode Configure enable flag, 1 - Enable, 0 - Disable
    uint8_t deviceModeType; //!< [0x011-0x011] Specify the configuration command type:Quad Enable, DPI/QPI/OPI switch,
    //! Generic configuration, etc.
    uint16_t waitTimeCfgCommands; //!< [0x012-0x013] Wait time for all configuration commands, unit: 100us, Used for
    //! DPI/QPI/OPI switch or reset command
    flexspi_lut_seq_t deviceModeSeq; //!< [0x014-0x017] Device mode sequence info, [7:0] - LUT sequence id, [15:8] - LUt
    //! sequence number, [31:16] Reserved
    uint32_t deviceModeArg;    //!< [0x018-0x01b] Argument/Parameter for device configuration
    uint8_t configCmdEnable;   //!< [0x01c-0x01c] Configure command Enable Flag, 1 - Enable, 0 - Disable
    uint8_t configModeType[3]; //!< [0x01d-0x01f] Configure Mode Type, similar as deviceModeTpe
    flexspi_lut_seq_t
        configCmdSeqs[3]; //!< [0x020-0x02b] Sequence info for Device Configuration command, similar as deviceModeSeq
    uint32_t reserved1;   //!< [0x02c-0x02f] Reserved for future use
    uint32_t configCmdArgs[3];     //!< [0x030-0x03b] Arguments/Parameters for device Configuration commands
    uint32_t reserved2;            //!< [0x03c-0x03f] Reserved for future use
    uint32_t controllerMiscOption; //!< [0x040-0x043] Controller Misc Options, see Misc feature bit definitions for more
    //! details
    uint8_t deviceType;    //!< [0x044-0x044] Device Type:  See Flash Type Definition for more details
    uint8_t sflashPadType; //!< [0x045-0x045] Serial Flash Pad Type: 1 - Single, 2 - Dual, 4 - Quad, 8 - Octal
    uint8_t serialClkFreq; //!< [0x046-0x046] Serial Flash Frequencey, device specific definitions, See System Boot
    //! Chapter for more details
    uint8_t lutCustomSeqEnable; //!< [0x047-0x047] LUT customization Enable, it is required if the program/erase cannot
    //! be done using 1 LUT sequence, currently, only applicable to HyperFLASH
    uint32_t reserved3[2];           //!< [0x048-0x04f] Reserved for future use
    uint32_t sflashA1Size;           //!< [0x050-0x053] Size of Flash connected to A1
    uint32_t sflashA2Size;           //!< [0x054-0x057] Size of Flash connected to A2
    uint32_t sflashB1Size;           //!< [0x058-0x05b] Size of Flash connected to B1
    uint32_t sflashB2Size;           //!< [0x05c-0x05f] Size of Flash connected to B2
    uint32_t csPadSettingOverride;   //!< [0x060-0x063] CS pad setting override value
    uint32_t sclkPadSettingOverride; //!< [0x064-0x067] SCK pad setting override value
    uint32_t dataPadSettingOverride; //!< [0x068-0x06b] data pad setting override value
    uint32_t dqsPadSettingOverride;  //!< [0x06c-0x06f] DQS pad setting override value
    uint32_t timeoutInMs;            //!< [0x070-0x073] Timeout threshold for read status command
    uint32_t commandInterval;        //!< [0x074-0x077] CS deselect interval between two commands
    uint16_t dataValidTime[2]; //!< [0x078-0x07b] CLK edge to data valid time for PORT A and PORT B, in terms of 0.1ns
    uint16_t busyOffset;       //!< [0x07c-0x07d] Busy offset, valid value: 0-31
    uint16_t busyBitPolarity;  //!< [0x07e-0x07f] Busy flag polarity, 0 - busy flag is 1 when flash device is busy, 1 -
    //! busy flag is 0 when flash device is busy
    uint32_t lookupTable[64];           //!< [0x080-0x17f] Lookup table holds Flash command sequences
    flexspi_lut_seq_t lutCustomSeq[12]; //!< [0x180-0x1af] Customizable LUT Sequences
    uint32_t reserved4[4];              //!< [0x1b0-0x1bf] Reserved for future use
} flexspi_mem_config_t;
```

Note that the structure is byte-aligned so 1:1 with the documentation and that unvalued fields are initialized to zero.