

# CPU\_CTRL\_XMC4

Home

File List

Globals

## File List

Here is a list of all documented files with brief descriptions:

 [CPU\\_CTRL\\_XMC4.c](#)

 [CPU\\_CTRL\\_XMC4.h](#)



# CPU\_CTRL\_XMC4

<a href="#">Home</a>		
<a href="#">File List</a>	<a href="#">Globals</a>	
<a href="#">Code</a>		

[Functions](#)

## CPU\_CTRL\_XMC4.c

### File Reference

## Detailed Description

### Date

2015-08-31 NOTE: This file is generated by DAVE. Any manual modification done to this file will be lost when the code is regenerated.

Definition in file [CPU\\_CTRL\\_XMC4.c](#).

```
#include "cpu_ctrl_xmc4.h"
```

## Functions

---

DAVE\_APP\_VERSION\_t **CPU\_CTRL\_XMC4\_GetAppVersion** (void)  
Get CPU\_CTRL\_XMC4 APP version.  
[More...](#)

void **CPU\_CTRL\_XMC4\_MPU\_Enable** (uint32\_t options)  
Enables MPU memory region/background memory region. [More...](#)

void **CPU\_CTRL\_XMC4\_MPU\_Disable** (void)  
Disables MPU settings. [More...](#)

[Go to the source code of this file.](#)



# CPU\_CTRL\_XMC4

<a href="#">Home</a>		
<a href="#">File List</a>	<a href="#">Globals</a>	
<a href="#">Code</a>		

## CPU\_CTRL\_XMC4.h File Reference

## Detailed Description

### Date

2015-10-10 NOTE: This file is generated by DAVE. Any manual modification done to this file will be lost when the code is regenerated.

Definition in file [CPU\\_CTRL\\_XMC4.h](#).

```
#include <xmc_common.h> #include <DAVE_Common.h>
#include "cpu_ctrl_xmc4_conf.h"
#include "cpu_ctrl_xmc4_extern.h"
```

## Functions

---

DAVE\_APP\_VERSION\_t **CPU\_CTRL\_XMC4\_GetAppVersion** (void)  
Get CPU\_CTRL\_XMC4 APP version. [More...](#)

---

void **CPU\_CTRL\_XMC4\_MPU\_Enable** (uint32\_t c  
Enables MPU memory region/background me  
region. [More...](#)

---

void **CPU\_CTRL\_XMC4\_MPU\_Disable** (void)  
Disables MPU settings. [More...](#)

---

enum **CPU\_CTRL\_XMC4\_STATUS** {  
**CPU\_CTRL\_XMC4\_STATUS\_SUCCESS** = C  
**CPU\_CTRL\_XMC4\_STATUS\_FAILURE** = 1U

---

enum **CPU\_CTRL\_XMC4\_MPU** {  
**CPU\_CTRL\_XMC4\_MPU\_ENABLE\_BGREG**  
MPU\_CTRL\_PRIVDEFENA\_Msk,  
**CPU\_CTRL\_XMC4\_MPU\_ENABLE\_NMI\_H**  
= MPU\_CTRL\_HFNMIENA\_Msk }

[Go to the source code of this file.](#)



# CPU\_CTRL\_XMC4

Home			
File List	<b>Globals</b>		
<b>All</b>	Functions	Enumerations	Enumerator

Here is a list of all documented functions, variables, defines, enums, and typedefs with links to the documentation:

- [CPU\\_CTRL\\_XMC4\\_GetAppVersion\(\)](#) : [CPU\\_CTRL\\_XMC4.c](#) , [CPU\\_CTRL\\_XMC4.h](#)
- [CPU\\_CTRL\\_XMC4\\_MPU](#) : [CPU\\_CTRL\\_XMC4.h](#)
- [CPU\\_CTRL\\_XMC4\\_MPU\\_Disable\(\)](#) : [CPU\\_CTRL\\_XMC4.c](#) , [CPU\\_CTRL\\_XMC4.h](#)
- [CPU\\_CTRL\\_XMC4\\_MPU\\_Enable\(\)](#) : [CPU\\_CTRL\\_XMC4.c](#) , [CPU\\_CTRL\\_XMC4.h](#)
- [CPU\\_CTRL\\_XMC4\\_MPU\\_ENABLE\\_BGREGION](#) : [CPU\\_CTRL\\_XMC4.h](#)
- [CPU\\_CTRL\\_XMC4\\_MPU\\_ENABLE\\_NMI\\_HARDFAULT](#) : [CPU\\_CTRL\\_XMC4.h](#)
- [CPU\\_CTRL\\_XMC4\\_STATUS](#) : [CPU\\_CTRL\\_XMC4.h](#)
- [CPU\\_CTRL\\_XMC4\\_STATUS\\_FAILURE](#) : [CPU\\_CTRL\\_XMC4.h](#)
- [CPU\\_CTRL\\_XMC4\\_STATUS\\_SUCCESS](#) : [CPU\\_CTRL\\_XMC4.h](#)





# CPU\_CTRL\_XMC4

Home			
File List	Globals		
All	Functions	Enumerations	Enumerator

- CPU\_CTRL\_XMC4\_GetAppVersion() : [CPU\\_CTRL\\_XMC4.c](#) , [CPU\\_CTRL\\_XMC4.h](#)
- CPU\_CTRL\_XMC4\_MPU\_Disable() : [CPU\\_CTRL\\_XMC4.h](#) , [CPU\\_CTRL\\_XMC4.c](#)
- CPU\_CTRL\_XMC4\_MPU\_Enable() : [CPU\\_CTRL\\_XMC4.h](#) , [CPU\\_CTRL\\_XMC4.c](#)



# CPU\_CTRL\_XMC4

Home			
File List	Globals		
All	Functions	Enumerations	Enumerator

- CPU\_CTRL\_XMC4\_MPU : [CPU\\_CTRL\\_XMC4.h](#)
- CPU\_CTRL\_XMC4\_STATUS : [CPU\\_CTRL\\_XMC4.h](#)



# CPU\_CTRL\_XMC4

Home			
File List	Globals		
All	Functions	Enumerations	Enumerator

- CPU\_CTRL\_XMC4\_MPU\_ENABLE\_BGREGION : [CPU\\_CTRL\\_XMC4.h](#)
- CPU\_CTRL\_XMC4\_MPU\_ENABLE\_NMI\_HARDFAULT : [CPU\\_CTRL\\_XMC4.h](#)
- CPU\_CTRL\_XMC4\_STATUS\_FAILURE : [CPU\\_CTRL\\_XMC4.h](#)
- CPU\_CTRL\_XMC4\_STATUS\_SUCCESS : [CPU\\_CTRL\\_XMC4.h](#)



# CPU\_CTRL\_XMC4

Home	
File List	Globals
Code	

## CPU\_CTRL\_XMC4.c

[Go to the documentation of this file.](#)

1

```
55 /*****
```

```
56 * HEADER FILES
```

```
57
```

```
*****
```

```
58 #include "cpu_ctrl_xmc4.h"
```

```
59 /*****
```

```
60 * MACROS
```

```
61
```

```
*****
```

```
62
```

```
63 /*****
```

```
64 * LOCAL DATA
```

```
65
```

```
*****
```

```
66
```

```
67 /*****
```

```
68 * LOCAL ROUTINES
```

```
69
```

```
*****
```

```
70
```

```
71 /*****
```

```
72 * API IMPLEMENTATION
```

```
73 *****
```

```
74 /*
```

```
75 * API to retrieve the version of the CPU_CTRL_XMC4 APP
```

```

76 */
77 DAVE_APP_VERSION_t CPU_CTRL_XMC4_GetAppVersion(void)
78 {
79     DAVE_APP_VERSION_t version;
80
81     version.major = CPU_CTRL_XMC4_MAJOR_VERSION;
82     version.minor = CPU_CTRL_XMC4_MINOR_VERSION;
83     version.patch = CPU_CTRL_XMC4_PATCH_VERSION;
84
85     return (version);
86 }
87 /* Dummy Init API to maintain backward compatibility */
88 CPU_CTRL_XMC4_STATUS_t
CPU_CTRL_XMC4_Init(CPU_CTRL_XMC4_t *const handler)
89 {
90     (void)handler;
91     return CPU_CTRL_XMC4_STATUS_SUCCESS;
92 }
93
94 /*
95  * API to enable the MPU
96  */
97 void CPU_CTRL_XMC4_MPU_Enable(uint32_t options)
98 {
99     MPU->CTRL = MPU_CTRL_ENABLE_Msk | options;
100     __DSB(); /* Ensure MPU settings take effect */
101     __ISB(); /* Sequence instruction fetches using updated settings */
102 }
103
104 /*
105  * API to disable the MPU
106  */
107 void CPU_CTRL_XMC4_MPU_Disable(void)
108 {
109     __DMB(); /* make sure all transfers are done */
110     MPU->CTRL = 0;
111 }

```

```
112
113 #if (HARDFFAULT_ENABLED == 1)
114
115 volatile uint32_t stacked_r0;
116 volatile uint32_t stacked_r1;
117 volatile uint32_t stacked_r2;
118 volatile uint32_t stacked_r3;
119 volatile uint32_t stacked_r12;
120 volatile uint32_t stacked_lr;
121 volatile uint32_t stacked_pc;
122 volatile uint32_t stacked_psr;
123 volatile uint32_t _CFSR;
124 volatile uint32_t _HFSR;
125 volatile uint32_t _DFSR;
126 volatile uint32_t _AFSR;
127 volatile uint32_t _BFAR;
128 volatile uint32_t _MMAR;
129
130 #if defined(__GNUC__)
131 #pragma GCC diagnostic ignored "-Wunused-but-set-variable"
132 #endif
133 __attribute__((naked)) void __HardFault_Handler(uint32_t args[])
134 {
135
136 // Configurable Fault Status Register
137 // Consists of MMSR, BFSR and UFSR
138 _CFSR = SCB->CFSR;
139
140 // Hard Fault Status Register
141 _HFSR = SCB->HFSR;
142
143 // Debug Fault Status Register
144 _DFSR = SCB->DFSR;
145
146 // Auxiliary Fault Status Register
147 _AFSR = SCB->AFSR;
148
149
```

```

158 // Read the Fault Address Registers. These may not contain valid
values.
159 // Check BFARVALID/MMARVALID to see if they are valid values
160 // MemManage Fault Address Register
161 _MMAR = SCB->MMFAR;
162
163 // Bus Fault Address Register
164 _BFAR = SCB->BFAR;
165
166 stacked_r0 = ((uint32_t)args[0]);
167 stacked_r1 = ((uint32_t)args[1]);
168 stacked_r2 = ((uint32_t)args[2]);
169 stacked_r3 = ((uint32_t)args[3]);
170 stacked_r12 = ((uint32_t)args[4]);
171 stacked_lr = ((uint32_t)args[5]);
172 stacked_pc = ((uint32_t)args[6]);
173 stacked_psr = ((uint32_t)args[7]);
174
175 __asm("BKPT 0\n") ; // Break into the debugger
176
177 }
178
179
180 /*KEIL*/
181 #if defined(__CC_ARM)
182 __asm void HardFault_Handler(void)
183 {
184 EXTERN __HardFault_Handler [CODE]
185
186 TST LR, #4
187 ITE EQ
188 MRSEQ R0, MSP
189 MRSNE R0, PSP
190 B __HardFault_Handler
191 }
192 #endif
193

```

```
194 /*IAR*/
195 #if defined(__ICCARM__)
196 void HardFault_Handler(void)
197 {
198     asm(" TST LR, #4 \n"
199     " ITE EQ \n"
200     " MRSEQ R0, MSP \n"
201     " MRSNE R0, PSP \n"
202     " B __HardFault_Handler\n");
203 }
204 #endif
205
206 /*TASKING*/
207 #if defined(__TASKING__)
208 void HardFault_Handler(void)
209 {
210     __asm(" TST LR, #4 \n"
211     " ITE EQ \n"
212     " MRSEQ R0, MSP \n"
213     " MRSNE R0, PSP \n"
214     " B __HardFault_Handler\n");
215 }
216
217 #endif
218
219 /*GCC*/
220 #if defined(__GNUC__)
221 __attribute__((naked)) void HardFault_Handler(void)
222 {
223     __asm(" TST LR, #4 \n"
224     " ITE EQ \n"
225     " MRSEQ R0, MSP \n"
226     " MRSNE R0, PSP \n"
227     " B __HardFault_Handler\n");
228 }
229 #endif
230 #endif
```

---





# CPU\_CTRL\_XMC4

Home	
File List	Globals
Code	

## CPU\_CTRL\_XMC4.h

[Go to the documentation of this file.](#)

1

```
57 #ifndef CPU_CTRL_XMC4_H
```

```
58 #define CPU_CTRL_XMC4_H
```

```
59
```

```
60 /*****
```

```
61 * HEADER FILES
```

```
62
```

```
*****/
```

```
63 #include <xmc_common.h>
```

```
64 #include <DAVE_Common.h>
```

```
65 #include "cpu_ctrl_xmc4_conf.h"
```

```
66 /*****
```

```
67 * MACROS
```

```
68
```

```
*****/
```

```
69
```

```
70 /*****
```

```
71 * ENUMS
```

```
72
```

```
*****/
```

```
77 /*
```

```
78 * @brief enumeration for CPU_CTRL_XMC4 APP
```

```
79 */
```

```
80 typedef enum CPU_CTRL_XMC4_STATUS
```

```
81 {
```

```
82 CPU_CTRL_XMC4_STATUS_SUCCESS = 0U,
```

```

83 CPU_CTRL_XMC4_STATUS_FAILURE = 1U
84 } CPU_CTRL_XMC4_STATUS_t;
85
86 typedef enum CPU_CTRL_XMC4_MPU
87 {
88 CPU_CTRL_XMC4_MPU_ENABLE_BGREGION =
MPU_CTRL_PRIVDEFENA_Msk,
89 CPU_CTRL_XMC4_MPU_ENABLE_NMI_HARDFAULT =
MPU_CTRL_HFNMIENA_Msk
90 } CPU_CTRL_XMC4_MPU_t;
91
92 /******
93 * DATA STRUCTURES
94 *****/
95 /*
96 * @brief Init structure for CPU_CTRL_XMC4 APP
97 */
98 typedef struct CPU_CTRL_XMC4
99 {
100 bool initialized;
101 } CPU_CTRL_XMC4_t;
102
103 /******
104 * API PROTOTYPES
105 *****/
106
107 #ifdef __cplusplus
108 extern "C" {
109 #endif
110
111 DAVE_APP_VERSION_t
CPU_CTRL_XMC4_GetAppVersion(void);
112
113 CPU_CTRL_XMC4_STATUS_t
CPU_CTRL_XMC4_Init(CPU_CTRL_XMC4_t *const handler);
114
115 void CPU_CTRL_XMC4_MPU_Enable(uint32_t options);
116
117 void CPU_CTRL_XMC4_MPU_Disable(void);

```

```
174
178 #ifdef __cplusplus
179 }
180 #endif
181
182
183 #include "cpu_ctrl_xmc4_extern.h"
184
185
186 #endif /* CPU_CTRL_XMC4_H */
187
```

---

A large empty rectangular box with a black border, likely a placeholder for content.

# CPU\_CTRL\_XMC4

[Home](#)

[Code](#) >

## Code Directory Reference

## Files

---

file [CPU\\_CTRL\\_XMC4.c](#) [code]

---

file [CPU\\_CTRL\\_XMC4.h](#) [code]

---



# CPU\_CTRL\_XMC4

[Home](#)

## Methods

DAVE\_APP\_VERSION\_t [CPU\\_CTRL\\_XMC4\\_GetAppVersion](#) (void)  
Get CPU\_CTRL\_XMC4 APP version.  
[More...](#)

void [CPU\\_CTRL\\_XMC4\\_MPU\\_Enable](#) (uint32\_t options)  
Enables MPU memory region/background memory region. [More...](#)

void [CPU\\_CTRL\\_XMC4\\_MPU\\_Disable](#) (void)  
Disables MPU settings. [More...](#)

## **Detailed Description**

### **Methods**



## Function Documentation

**DAVE\_APP\_VERSION\_t CPU\_CTRL\_XMC4\_GetAppVersion ( void )**

Get CPU\_CTRL\_XMC4 APP version.

### Returns

DAVE\_APP\_VERSION\_t APP version information (major, minor and patch number)

### Description:

The function can be used to check application software compatibility with a specific version of the APP.

Example Usage:

```
#include <DAVE.h>
int main(void)
{
    DAVE_APP_VERSION_t version;
    // CPU_CTRL_XMC4_Init() is called from within DAVE_Init().
    init_status = DAVE_Init();
    version = CPU_CTRL_XMC4_GetAppVersion();
    if (version.major != 4U)
    {
    }
    // More code here
    while(1)
    {
    }
    return (0);
}
```

Definition at line **77** of file **CPU\_CTRL\_XMC4.c**.

**void CPU\_CTRL\_XMC4\_MPU\_Disable ( void )**

Disables MPU settings.

**Returns**

None

**Description:**

The function can be used to disable all the MPU options by writing zero.

Definition at line [107](#) of file [CPU\\_CTRL\\_XMC4.c](#).

**void CPU\_CTRL\_XMC4\_MPU\_Enable ( uint32\_t options )**

Enables MPU memory region/background memory region.

**Returns**

None

**Description:**

The function can be used to enable MPU options by settings background memory region or memory region

Definition at line [97](#) of file [CPU\\_CTRL\\_XMC4.c](#).

# CPU\_CTRL\_XMC4

[Home](#)

## CPU\_CTRL\_XMC4

```
enum CPU_CTRL_XMC4_MPU {  
    CPU_CTRL_XMC4_MPU_ENABLE_BGREGION =  
    MPU_CTRL_PRIVDEFENA_Msk,  
    CPU_CTRL_XMC4_MPU_ENABLE_NMI_HARDFULT =  
    MPU_CTRL_HFNMIENA_Msk };
```

void **CPU\_CTRL\_XMC4\_MPU\_Enable** (uint32\_t options)  
Enables MPU memory region/background memory region.  
[More...](#)

void **CPU\_CTRL\_XMC4\_MPU\_Disable** (void)  
Disables MPU settings. [More...](#)

## Detailed Description

### Enumeration Type Documentation

enum `CPU_CTRL_XMC4_MPU`

Enumerator	
<code>CPU_CTRL_XMC4_MPU_ENABLE_BGREGION</code>	Enables background memory region
<code>CPU_CTRL_XMC4_MPU_ENABLE_NMI_HARDFULT</code>	Enables MPU during NMI and hardfault handler

Definition at line **86** of file `CPU_CTRL_XMC4.h`.

## Function Documentation

**void CPU\_CTRL\_XMC4\_MPU\_Disable ( void )**

Disables MPU settings.

**Returns**

None

**Description:**

The function can be used to disable all the MPU options by writing zero.

Definition at line **107** of file **CPU\_CTRL\_XMC4.c**.

**void CPU\_CTRL\_XMC4\_MPU\_Enable ( uint32\_t options )**

Enables MPU memory region/background memory region.

**Returns**

None

**Description:**

The function can be used to enable MPU options by settings background memory region or memory region

Definition at line **97** of file **CPU\_CTRL\_XMC4.c**.



# CPU\_CTRL\_XMC4

[Home](#)

## Enumerations

```
enum CPU_CTRL_XMC4_STATUS {  
    CPU_CTRL_XMC4_STATUS_SUCCESS = 0U,  
    CPU_CTRL_XMC4_STATUS_FAILURE = 1U }
```

```
enum CPU_CTRL_XMC4_MPU {  
    CPU_CTRL_XMC4_MPU_ENABLE_BGREGION =  
    MPU_CTRL_PRIVDEFENA_Msk,  
    CPU_CTRL_XMC4_MPU_ENABLE_NMI_HARDFAULT =  
    MPU_CTRL_HFNMIENA_Msk }
```

## Detailed Description

## Enumeration Type Documentation

### enum CPU\_CTRL\_XMC4\_MPU

Enumerator	
<i>CPU_CTRL_XMC4_MPU_ENABLE_BGREGION</i>	Enables background memory region
<i>CPU_CTRL_XMC4_MPU_ENABLE_NMI_HARDFFAULT</i>	Enables MPU during NMI and hardfault handler

Definition at line **86** of file **CPU\_CTRL\_XMC4.h**.

### enum CPU\_CTRL\_XMC4\_STATUS

Enumerator	
<i>CPU_CTRL_XMC4_STATUS_SUCCESS</i>	APP initialization is success
<i>CPU_CTRL_XMC4_STATUS_FAILURE</i>	APP initialization is failure

Definition at line **80** of file **CPU\_CTRL\_XMC4.h**.



