

sarmfsw: SMFSW Toolbox (for ARM, STM32)

0.8

Generated by Doxygen 1.8.11

## Contents

<b>1</b>	<b>Class Index</b>	<b>1</b>
1.1	Class List . . . . .	1
<b>2</b>	<b>File Index</b>	<b>2</b>
2.1	File List . . . . .	2
<b>3</b>	<b>Class Documentation</b>	<b>2</b>
3.1	StructBitfield16 Struct Reference . . . . .	2
3.1.1	Detailed Description . . . . .	3
3.1.2	Member Data Documentation . . . . .	3
3.2	StructBitfield32 Struct Reference . . . . .	5
3.2.1	Detailed Description . . . . .	6
3.2.2	Member Data Documentation . . . . .	6
3.3	StructBitfield64 Struct Reference . . . . .	9
3.3.1	Detailed Description . . . . .	12
3.3.2	Member Data Documentation . . . . .	12
3.4	StructBitfield8 Struct Reference . . . . .	17
3.4.1	Detailed Description . . . . .	18
3.4.2	Member Data Documentation . . . . .	18
3.5	UnionByte Union Reference . . . . .	19
3.5.1	Detailed Description . . . . .	19
3.5.2	Member Data Documentation . . . . .	19
3.6	UnionDWord Union Reference . . . . .	20
3.6.1	Detailed Description . . . . .	21
3.6.2	Member Data Documentation . . . . .	21
3.7	UnionLWord Union Reference . . . . .	22
3.7.1	Detailed Description . . . . .	23
3.7.2	Member Data Documentation . . . . .	23
3.8	UnionWord Union Reference . . . . .	25
3.8.1	Detailed Description . . . . .	26
3.8.2	Member Data Documentation . . . . .	26

<b>4 File Documentation</b>	<b>27</b>
4.1 arm_attributes.h File Reference . . . . .	27
4.1.1 Detailed Description . . . . .	27
4.1.2 Macro Definition Documentation . . . . .	28
4.2 arm_cmsis.h File Reference . . . . .	28
4.2.1 Detailed Description . . . . .	28
4.3 arm_macros.h File Reference . . . . .	29
4.3.1 Detailed Description . . . . .	31
4.3.2 Macro Definition Documentation . . . . .	32
4.3.3 Function Documentation . . . . .	35
4.4 arm_stdclib.h File Reference . . . . .	36
4.4.1 Detailed Description . . . . .	37
4.4.2 Macro Definition Documentation . . . . .	37
4.5 arm_stm32.h File Reference . . . . .	38
4.5.1 Detailed Description . . . . .	38
4.5.2 Macro Definition Documentation . . . . .	39
4.6 arm_typedefs.h File Reference . . . . .	39
4.6.1 Detailed Description . . . . .	41
4.6.2 Typedef Documentation . . . . .	42
4.6.3 Enumeration Type Documentation . . . . .	43
4.7 sarmfsw.h File Reference . . . . .	43
4.7.1 Detailed Description . . . . .	44
4.7.2 Typedef Documentation . . . . .	44
4.7.3 Enumeration Type Documentation . . . . .	44
<b>Index</b>	<b>45</b>

## 1 Class Index

### 1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<a href="#">StructBitfield16</a>	
Bitfield 16b	2
<a href="#">StructBitfield32</a>	
Bitfield 32b	5
<a href="#">StructBitfield64</a>	
Bitfield 64b	9
<a href="#">StructBitfield8</a>	
Bitfield 8b	17
<a href="#">UnionByte</a>	
Union for BYTE	19
<a href="#">UnionDWord</a>	
Union for DWORD	20
<a href="#">UnionLWord</a>	
Union for LWORD	22
<a href="#">UnionWord</a>	
Union for WORD	25

## 2 File Index

### 2.1 File List

Here is a list of all files with brief descriptions:

<a href="#">arm_attributes.h</a>	
ARM common gcc attributes	27
<a href="#">arm_cmsis.h</a>	
ARM link with CMSIS files	28
<a href="#">arm_macros.h</a>	
ARM common macros	29
<a href="#">arm_stdclib.h</a>	
ARM common standard c library wrapper macros	36
<a href="#">arm_stm32.h</a>	
ARM common macros for STM32	38
<a href="#">arm_typedefs.h</a>	
ARM common typedefs	39
<a href="#">sarmfsw.h</a>	
ARM common headers for projects	43

## 3 Class Documentation

### 3.1 StructBitfield16 Struct Reference

Bitfield 16b.

```
#include <arm_typedefs.h>
```

## Public Attributes

- [WORD b0:1](#)  
*Bit 0 (LSB)*
- [WORD b1:1](#)  
*Bit 1.*
- [WORD b2:1](#)  
*Bit 2.*
- [WORD b3:1](#)  
*Bit 3.*
- [WORD b4:1](#)  
*Bit 4.*
- [WORD b5:1](#)  
*Bit 5.*
- [WORD b6:1](#)  
*Bit 6.*
- [WORD b7:1](#)  
*Bit 7.*
- [WORD b8:1](#)  
*Bit 8.*
- [WORD b9:1](#)  
*Bit 9.*
- [WORD b10:1](#)  
*Bit 10.*
- [WORD b11:1](#)  
*Bit 11.*
- [WORD b12:1](#)  
*Bit 12.*
- [WORD b13:1](#)  
*Bit 13.*
- [WORD b14:1](#)  
*Bit 14.*
- [WORD b15:1](#)  
*Bit 15 (MSB)*

### 3.1.1 Detailed Description

Bitfield 16b.

### 3.1.2 Member Data Documentation

#### 3.1.2.1 WORD StructBitfield16::b0

Bit 0 (LSB)

#### 3.1.2.2 WORD StructBitfield16::b1

Bit 1.

**3.1.2.3 WORD StructBitfield16::b10**

Bit 10.

**3.1.2.4 WORD StructBitfield16::b11**

Bit 11.

**3.1.2.5 WORD StructBitfield16::b12**

Bit 12.

**3.1.2.6 WORD StructBitfield16::b13**

Bit 13.

**3.1.2.7 WORD StructBitfield16::b14**

Bit 14.

**3.1.2.8 WORD StructBitfield16::b15**

Bit 15 (MSB)

**3.1.2.9 WORD StructBitfield16::b2**

Bit 2.

**3.1.2.10 WORD StructBitfield16::b3**

Bit 3.

**3.1.2.11 WORD StructBitfield16::b4**

Bit 4.

**3.1.2.12 WORD StructBitfield16::b5**

Bit 5.

**3.1.2.13 WORD StructBitfield16::b6**

Bit 6.

**3.1.2.14 WORD StructBitfield16::b7**

Bit 7.

## 3.1.2.15 WORD StructBitfield16::b8

Bit 8.

## 3.1.2.16 WORD StructBitfield16::b9

Bit 9.

The documentation for this struct was generated from the following file:

- [arm\\_typedefs.h](#)

## 3.2 StructBitfield32 Struct Reference

Bitfield 32b.

```
#include <arm_typedefs.h>
```

## Public Attributes

- [DWORD b0](#):1  
*Bit 0 (LSB)*
- [DWORD b1](#):1  
*Bit 1.*
- [DWORD b2](#):1  
*Bit 2.*
- [DWORD b3](#):1  
*Bit 3.*
- [DWORD b4](#):1  
*Bit 4.*
- [DWORD b5](#):1  
*Bit 5.*
- [DWORD b6](#):1  
*Bit 6.*
- [DWORD b7](#):1  
*Bit 7.*
- [DWORD b8](#):1  
*Bit 8.*
- [DWORD b9](#):1  
*Bit 9.*
- [DWORD b10](#):1  
*Bit 10.*
- [DWORD b11](#):1  
*Bit 11.*
- [DWORD b12](#):1  
*Bit 12.*
- [DWORD b13](#):1  
*Bit 13.*
- [DWORD b14](#):1

- Bit 14.*
- [DWORD b15:1](#)
- Bit 15.*
- [DWORD b16:1](#)
- Bit 16.*
- [DWORD b17:1](#)
- Bit 17.*
- [DWORD b18:1](#)
- Bit 18.*
- [DWORD b19:1](#)
- Bit 19.*
- [DWORD b20:1](#)
- Bit 20.*
- [DWORD b21:1](#)
- Bit 21.*
- [DWORD b22:1](#)
- Bit 22.*
- [DWORD b23:1](#)
- Bit 23.*
- [DWORD b24:1](#)
- Bit 24.*
- [DWORD b25:1](#)
- Bit 25.*
- [DWORD b26:1](#)
- Bit 26.*
- [DWORD b27:1](#)
- Bit 27.*
- [DWORD b28:1](#)
- Bit 28.*
- [DWORD b29:1](#)
- Bit 29.*
- [DWORD b30:1](#)
- Bit 30.*
- [DWORD b31:1](#)
- Bit 31 (MSB)*

### 3.2.1 Detailed Description

Bitfield 32b.

### 3.2.2 Member Data Documentation

#### 3.2.2.1 **DWORD StructBitfield32::b0**

Bit 0 (LSB)

#### 3.2.2.2 **DWORD StructBitfield32::b1**

Bit 1.



**3.2.2.3   DWORD StructBitfield32::b10**

Bit 10.

**3.2.2.4   DWORD StructBitfield32::b11**

Bit 11.

**3.2.2.5   DWORD StructBitfield32::b12**

Bit 12.

**3.2.2.6   DWORD StructBitfield32::b13**

Bit 13.

**3.2.2.7   DWORD StructBitfield32::b14**

Bit 14.

**3.2.2.8   DWORD StructBitfield32::b15**

Bit 15.

**3.2.2.9   DWORD StructBitfield32::b16**

Bit 16.

**3.2.2.10   DWORD StructBitfield32::b17**

Bit 17.

**3.2.2.11   DWORD StructBitfield32::b18**

Bit 18.

**3.2.2.12   DWORD StructBitfield32::b19**

Bit 19.

**3.2.2.13   DWORD StructBitfield32::b2**

Bit 2.

**3.2.2.14   DWORD StructBitfield32::b20**

Bit 20.

**3.2.2.15   DWORD StructBitfield32::b21**

Bit 21.

**3.2.2.16   DWORD StructBitfield32::b22**

Bit 22.

**3.2.2.17   DWORD StructBitfield32::b23**

Bit 23.

**3.2.2.18   DWORD StructBitfield32::b24**

Bit 24.

**3.2.2.19   DWORD StructBitfield32::b25**

Bit 25.

**3.2.2.20   DWORD StructBitfield32::b26**

Bit 26.

**3.2.2.21   DWORD StructBitfield32::b27**

Bit 27.

**3.2.2.22   DWORD StructBitfield32::b28**

Bit 28.

**3.2.2.23   DWORD StructBitfield32::b29**

Bit 29.

**3.2.2.24   DWORD StructBitfield32::b3**

Bit 3.

**3.2.2.25   DWORD StructBitfield32::b30**

Bit 30.

**3.2.2.26   DWORD StructBitfield32::b31**

Bit 31 (MSB)

**3.2.2.27** DWORD StructBitfield32::b4

Bit 4.

**3.2.2.28** DWORD StructBitfield32::b5

Bit 5.

**3.2.2.29** DWORD StructBitfield32::b6

Bit 6.

**3.2.2.30** DWORD StructBitfield32::b7

Bit 7.

**3.2.2.31** DWORD StructBitfield32::b8

Bit 8.

**3.2.2.32** DWORD StructBitfield32::b9

Bit 9.

The documentation for this struct was generated from the following file:

- [arm\\_typedefs.h](#)

**3.3** StructBitfield64 Struct Reference

Bitfield 64b.

```
#include <arm_typedefs.h>
```

**Public Attributes**

- [LWORD b0:1](#)  
*Bit 0 (LSB)*
- [LWORD b1:1](#)  
*Bit 1.*
- [LWORD b2:1](#)  
*Bit 2.*
- [LWORD b3:1](#)  
*Bit 3.*
- [LWORD b4:1](#)  
*Bit 4.*
- [LWORD b5:1](#)  
*Bit 5.*
- [LWORD b6:1](#)

- Bit 6.*
  - [LWORD b7:1](#)
- Bit 7.*
  - [LWORD b8:1](#)
- Bit 8.*
  - [LWORD b9:1](#)
- Bit 9.*
  - [LWORD b10:1](#)
- Bit 10.*
  - [LWORD b11:1](#)
- Bit 11.*
  - [LWORD b12:1](#)
- Bit 12.*
  - [LWORD b13:1](#)
- Bit 13.*
  - [LWORD b14:1](#)
- Bit 14.*
  - [LWORD b15:1](#)
- Bit 15.*
  - [LWORD b16:1](#)
- Bit 16.*
  - [LWORD b17:1](#)
- Bit 17.*
  - [LWORD b18:1](#)
- Bit 18.*
  - [LWORD b19:1](#)
- Bit 19.*
  - [LWORD b20:1](#)
- Bit 20.*
  - [LWORD b21:1](#)
- Bit 21.*
  - [LWORD b22:1](#)
- Bit 22.*
  - [LWORD b23:1](#)
- Bit 23.*
  - [LWORD b24:1](#)
- Bit 24.*
  - [LWORD b25:1](#)
- Bit 25.*
  - [LWORD b26:1](#)
- Bit 26.*
  - [LWORD b27:1](#)
- Bit 27.*
  - [LWORD b28:1](#)
- Bit 28.*
  - [LWORD b29:1](#)
- Bit 29.*
  - [LWORD b30:1](#)
- Bit 30.*
  - [LWORD b31:1](#)
- Bit 31.*

- [LWORD b32:1](#)  
*Bit 32.*
- [LWORD b33:1](#)  
*Bit 33.*
- [LWORD b34:1](#)  
*Bit 34.*
- [LWORD b35:1](#)  
*Bit 35.*
- [LWORD b36:1](#)  
*Bit 36.*
- [LWORD b37:1](#)  
*Bit 37.*
- [LWORD b38:1](#)  
*Bit 38.*
- [LWORD b39:1](#)  
*Bit 39.*
- [LWORD b40:1](#)  
*Bit 40.*
- [LWORD b41:1](#)  
*Bit 41.*
- [LWORD b42:1](#)  
*Bit 42.*
- [LWORD b43:1](#)  
*Bit 43.*
- [LWORD b44:1](#)  
*Bit 44.*
- [LWORD b45:1](#)  
*Bit 45.*
- [LWORD b46:1](#)  
*Bit 46.*
- [LWORD b47:1](#)  
*Bit 47.*
- [LWORD b48:1](#)  
*Bit 48.*
- [LWORD b49:1](#)  
*Bit 49.*
- [LWORD b50:1](#)  
*Bit 50.*
- [LWORD b51:1](#)  
*Bit 51.*
- [LWORD b52:1](#)  
*Bit 52.*
- [LWORD b53:1](#)  
*Bit 53.*
- [LWORD b54:1](#)  
*Bit 54.*
- [LWORD b55:1](#)  
*Bit 55.*
- [LWORD b56:1](#)  
*Bit 56.*
- [LWORD b57:1](#)

- Bit 57.*
- [LWORD b58:1](#)
- Bit 58.*
- [LWORD b59:1](#)
- Bit 59.*
- [LWORD b60:1](#)
- Bit 60.*
- [LWORD b61:1](#)
- Bit 61.*
- [LWORD b62:1](#)
- Bit 62.*
- [LWORD b63:1](#)
- Bit 63 (MSB)*

### 3.3.1 Detailed Description

Bitfield 64b.

### 3.3.2 Member Data Documentation

#### 3.3.2.1 LWORD StructBitfield64::b0

Bit 0 (LSB)

#### 3.3.2.2 LWORD StructBitfield64::b1

Bit 1.

#### 3.3.2.3 LWORD StructBitfield64::b10

Bit 10.

#### 3.3.2.4 LWORD StructBitfield64::b11

Bit 11.

#### 3.3.2.5 LWORD StructBitfield64::b12

Bit 12.

#### 3.3.2.6 LWORD StructBitfield64::b13

Bit 13.

#### 3.3.2.7 LWORD StructBitfield64::b14

Bit 14.

**3.3.2.8 LWORD StructBitfield64::b15**

Bit 15.

**3.3.2.9 LWORD StructBitfield64::b16**

Bit 16.

**3.3.2.10 LWORD StructBitfield64::b17**

Bit 17.

**3.3.2.11 LWORD StructBitfield64::b18**

Bit 18.

**3.3.2.12 LWORD StructBitfield64::b19**

Bit 19.

**3.3.2.13 LWORD StructBitfield64::b2**

Bit 2.

**3.3.2.14 LWORD StructBitfield64::b20**

Bit 20.

**3.3.2.15 LWORD StructBitfield64::b21**

Bit 21.

**3.3.2.16 LWORD StructBitfield64::b22**

Bit 22.

**3.3.2.17 LWORD StructBitfield64::b23**

Bit 23.

**3.3.2.18 LWORD StructBitfield64::b24**

Bit 24.

**3.3.2.19 LWORD StructBitfield64::b25**

Bit 25.

**3.3.2.20 LWORD StructBitfield64::b26**

Bit 26.

**3.3.2.21 LWORD StructBitfield64::b27**

Bit 27.

**3.3.2.22 LWORD StructBitfield64::b28**

Bit 28.

**3.3.2.23 LWORD StructBitfield64::b29**

Bit 29.

**3.3.2.24 LWORD StructBitfield64::b3**

Bit 3.

**3.3.2.25 LWORD StructBitfield64::b30**

Bit 30.

**3.3.2.26 LWORD StructBitfield64::b31**

Bit 31.

**3.3.2.27 LWORD StructBitfield64::b32**

Bit 32.

**3.3.2.28 LWORD StructBitfield64::b33**

Bit 33.

**3.3.2.29 LWORD StructBitfield64::b34**

Bit 34.

**3.3.2.30 LWORD StructBitfield64::b35**

Bit 35.

**3.3.2.31 LWORD StructBitfield64::b36**

Bit 36.



**3.3.2.32 LWORD StructBitfield64::b37**

Bit 37.

**3.3.2.33 LWORD StructBitfield64::b38**

Bit 38.

**3.3.2.34 LWORD StructBitfield64::b39**

Bit 39.

**3.3.2.35 LWORD StructBitfield64::b4**

Bit 4.

**3.3.2.36 LWORD StructBitfield64::b40**

Bit 40.

**3.3.2.37 LWORD StructBitfield64::b41**

Bit 41.

**3.3.2.38 LWORD StructBitfield64::b42**

Bit 42.

**3.3.2.39 LWORD StructBitfield64::b43**

Bit 43.

**3.3.2.40 LWORD StructBitfield64::b44**

Bit 44.

**3.3.2.41 LWORD StructBitfield64::b45**

Bit 45.

**3.3.2.42 LWORD StructBitfield64::b46**

Bit 46.

**3.3.2.43 LWORD StructBitfield64::b47**

Bit 47.

**3.3.2.44 LWORD StructBitfield64::b48**

Bit 48.

**3.3.2.45 LWORD StructBitfield64::b49**

Bit 49.

**3.3.2.46 LWORD StructBitfield64::b5**

Bit 5.

**3.3.2.47 LWORD StructBitfield64::b50**

Bit 50.

**3.3.2.48 LWORD StructBitfield64::b51**

Bit 51.

**3.3.2.49 LWORD StructBitfield64::b52**

Bit 52.

**3.3.2.50 LWORD StructBitfield64::b53**

Bit 53.

**3.3.2.51 LWORD StructBitfield64::b54**

Bit 54.

**3.3.2.52 LWORD StructBitfield64::b55**

Bit 55.

**3.3.2.53 LWORD StructBitfield64::b56**

Bit 56.

**3.3.2.54 LWORD StructBitfield64::b57**

Bit 57.

**3.3.2.55 LWORD StructBitfield64::b58**

Bit 58.

**3.3.2.56 LWORD StructBitfield64::b59**

Bit 59.

**3.3.2.57 LWORD StructBitfield64::b6**

Bit 6.

**3.3.2.58 LWORD StructBitfield64::b60**

Bit 60.

**3.3.2.59 LWORD StructBitfield64::b61**

Bit 61.

**3.3.2.60 LWORD StructBitfield64::b62**

Bit 62.

**3.3.2.61 LWORD StructBitfield64::b63**

Bit 63 (MSB)

**3.3.2.62 LWORD StructBitfield64::b7**

Bit 7.

**3.3.2.63 LWORD StructBitfield64::b8**

Bit 8.

**3.3.2.64 LWORD StructBitfield64::b9**

Bit 9.

The documentation for this struct was generated from the following file:

- [arm\\_typedefs.h](#)

**3.4 StructBitfield8 Struct Reference**

Bitfield 8b.

```
#include <arm_typedefs.h>
```

## Public Attributes

- **BYTE b0**:1  
*Bit 0 (LSB)*
- **BYTE b1**:1  
*Bit 1.*
- **BYTE b2**:1  
*Bit 2.*
- **BYTE b3**:1  
*Bit 3.*
- **BYTE b4**:1  
*Bit 4.*
- **BYTE b5**:1  
*Bit 5.*
- **BYTE b6**:1  
*Bit 6.*
- **BYTE b7**:1  
*Bit 7 (MSB)*

### 3.4.1 Detailed Description

Bitfield 8b.

### 3.4.2 Member Data Documentation

#### 3.4.2.1 **BYTE StructBitfield8::b0**

Bit 0 (LSB)

#### 3.4.2.2 **BYTE StructBitfield8::b1**

Bit 1.

#### 3.4.2.3 **BYTE StructBitfield8::b2**

Bit 2.

#### 3.4.2.4 **BYTE StructBitfield8::b3**

Bit 3.

#### 3.4.2.5 **BYTE StructBitfield8::b4**

Bit 4.

#### 3.4.2.6 **BYTE StructBitfield8::b5**

Bit 5.

## 3.4.2.7 BYTE StructBitfield8::b6

Bit 6.

## 3.4.2.8 BYTE StructBitfield8::b7

Bit 7 (MSB)

The documentation for this struct was generated from the following file:

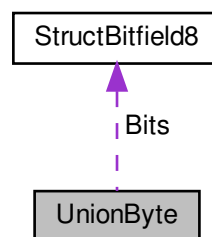
- [arm\\_typedefs.h](#)

## 3.5 UnionByte Union Reference

Union for BYTE.

```
#include <arm_typedefs.h>
```

Collaboration diagram for UnionByte:



## Public Attributes

- [BYTE Byte](#)  
*BYTE.*
- [sBitfield8 Bits](#)  
*Bits.*

## 3.5.1 Detailed Description

Union for BYTE.

## 3.5.2 Member Data Documentation

## 3.5.2.1 sBitfield8 UnionByte::Bits

Bits.

### 3.5.2.2 BYTE UnionByte::Byte

BYTE.

The documentation for this union was generated from the following file:

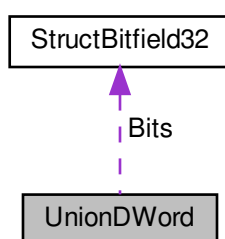
- [arm\\_typedefs.h](#)

## 3.6 UnionDWord Union Reference

Union for DWORD.

```
#include <arm_typedefs.h>
```

Collaboration diagram for UnionDWord:



### Public Attributes

- [DWORD DWord](#)  
*32b*
- [WORD Word](#) [2]  
*Words tab.*
- [BYTE Byte](#) [4]  
*Bytes tab.*
- struct {  
    [WORD W0](#):16  
    *W0 LSWord.*  
    [WORD W1](#):16  
    *W1 MSWord.*  
} [Words](#)
- struct {  
    [BYTE B0](#):8  
    *B0 LSByte.*  
    [BYTE B1](#):8  
    *B1.*  
    [BYTE B2](#):8  
    *B2.*  
    [BYTE B3](#):8  
    *B3 MSByte.*  
} [Bytes](#)
- [sBitfield32 Bits](#)  
*Bits.*

### 3.6.1 Detailed Description

Union for DWORD.

### 3.6.2 Member Data Documentation

#### 3.6.2.1 BYTE UnionDWord::B0

B0 LSByte.

#### 3.6.2.2 BYTE UnionDWord::B1

B1.

#### 3.6.2.3 BYTE UnionDWord::B2

B2.

#### 3.6.2.4 BYTE UnionDWord::B3

B3 MSByte.

#### 3.6.2.5 sBitfield32 UnionDWord::Bits

Bits.

#### 3.6.2.6 BYTE UnionDWord::Byte[4]

Bytes tab.

#### 3.6.2.7 struct { ... } UnionDWord::Bytes

#### 3.6.2.8 DWORD UnionDWord::DWord

32b

#### 3.6.2.9 WORD UnionDWord::W0

W0 LSWord.

#### 3.6.2.10 WORD UnionDWord::W1

W1 MSWord.

#### 3.6.2.11 WORD UnionDWord::Word[2]

Words tab.

### 3.6.2.12 struct { ... } UnionDWord::Words

The documentation for this union was generated from the following file:

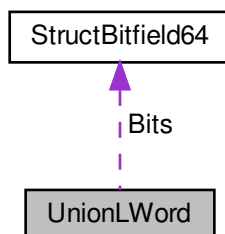
- [arm\\_typedefs.h](#)

## 3.7 UnionLWord Union Reference

Union for LWORD.

```
#include <arm_typedefs.h>
```

Collaboration diagram for UnionLWord:



### Public Attributes

- [LWORD LWord](#)  
*64b*
- [DWORD DWord](#) [2]  
*DWords tab.*
- [WORD Word](#) [4]  
*Words tab.*
- [BYTE Byte](#) [8]  
*Bytes tab.*
- struct {  
    [DWORD D0](#):32  
    *DW0 LSDWord.*  
    [DWORD D1](#):32  
    *DW1 MSDWord.*  
} [DWords](#)



- struct {
  - WORD W0:16  
W0 LSWord.
  - WORD W1:16  
W1.
  - WORD W2:16  
W2.
  - WORD W3:16  
W3 MSWord.
} Words
- struct {
  - BYTE B0:8  
B0 LSByte.
  - BYTE B1:8  
B1.
  - BYTE B2:8  
B2.
  - BYTE B3:8  
B3.
  - BYTE B4:8  
B4.
  - BYTE B5:8  
B5.
  - BYTE B6:8  
B6.
  - BYTE B7:8  
B7 MSByte.
} Bytes
- sBitfield64 Bits
  - Bits.

### 3.7.1 Detailed Description

Union for LWORD.

### 3.7.2 Member Data Documentation

#### 3.7.2.1 BYTE UnionLWord::B0

B0 LSByte.

#### 3.7.2.2 BYTE UnionLWord::B1

B1.

#### 3.7.2.3 BYTE UnionLWord::B2

B2.

**3.7.2.4 BYTE UnionLWord::B3**

B3.

**3.7.2.5 BYTE UnionLWord::B4**

B4.

**3.7.2.6 BYTE UnionLWord::B5**

B5.

**3.7.2.7 BYTE UnionLWord::B6**

B6.

**3.7.2.8 BYTE UnionLWord::B7**

B7 MSByte.

**3.7.2.9 sBitfield64 UnionLWord::Bits**

Bits.

**3.7.2.10 BYTE UnionLWord::Byte[8]**

Bytes tab.

**3.7.2.11 struct { ... } UnionLWord::Bytes****3.7.2.12 DWORD UnionLWord::D0**

DW0 LSDWord.

**3.7.2.13 DWORD UnionLWord::D1**

DW1 MSDWord.

**3.7.2.14 DWORD UnionLWord::DWord[2]**

DWords tab.

**3.7.2.15 struct { ... } UnionLWord::DWords****3.7.2.16 LWORD UnionLWord::LWord**

64b

### 3.7.2.17 WORD UnionLWord::W0

W0 LSWord.

### 3.7.2.18 WORD UnionLWord::W1

W1.

### 3.7.2.19 WORD UnionLWord::W2

W2.

### 3.7.2.20 WORD UnionLWord::W3

W3 MSWord.

### 3.7.2.21 WORD UnionLWord::Word[4]

Words tab.

### 3.7.2.22 struct { ... } UnionLWord::Words

The documentation for this union was generated from the following file:

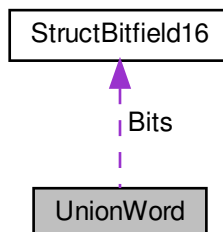
- [arm\\_typedefs.h](#)

## 3.8 UnionWord Union Reference

Union for WORD.

```
#include <arm_typedefs.h>
```

Collaboration diagram for UnionWord:



## Public Attributes

- [WORD Word](#)  
*16b*
- [BYTE Byte](#) [2]  
*Bytes tab.*
- struct {  
    [BYTE B0](#):8  
    *LSByte.*  
    [BYTE B1](#):8  
    *MSByte.*  
} [Bytes](#)
- [sBitfield16 Bits](#)  
*Bits.*

### 3.8.1 Detailed Description

Union for WORD.

### 3.8.2 Member Data Documentation

#### 3.8.2.1 [BYTE UnionWord::B0](#)

*LSByte.*

#### 3.8.2.2 [BYTE UnionWord::B1](#)

*MSByte.*

#### 3.8.2.3 [sBitfield16 UnionWord::Bits](#)

*Bits.*

#### 3.8.2.4 [BYTE UnionWord::Byte\[2\]](#)

*Bytes tab.*

#### 3.8.2.5 struct { ... } [UnionWord::Bytes](#)

#### 3.8.2.6 [WORD UnionWord::Word](#)

*16b*

The documentation for this union was generated from the following file:

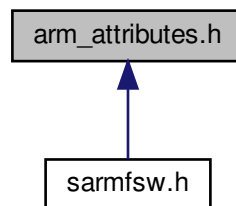
- [arm\\_typedefs.h](#)

## 4 File Documentation

### 4.1 arm\_attributes.h File Reference

ARM common gcc attributes.

This graph shows which files directly or indirectly include this file:



#### Macros

- #define [INLINE\\_\\_](#)
- #define [WEAK\\_\\_](#)
- #define [PACK\\_\\_](#)

#### 4.1.1 Detailed Description

ARM common gcc attributes.

#### Author

SMFSW

#### Version

v0.8

#### Date

2017

#### Copyright

MIT (c) 2017, SMFSW

## 4.1.2 Macro Definition Documentation

### 4.1.2.1 #define INLINE\_\_

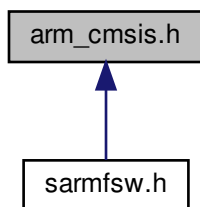
### 4.1.2.2 #define PACK\_\_

### 4.1.2.3 #define WEAK\_\_

## 4.2 arm\_cmsis.h File Reference

ARM link with CMSIS files.

This graph shows which files directly or indirectly include this file:



### 4.2.1 Detailed Description

ARM link with CMSIS files.

#### Author

SMFSW

#### Version

v0.8

#### Date

2017

#### Copyright

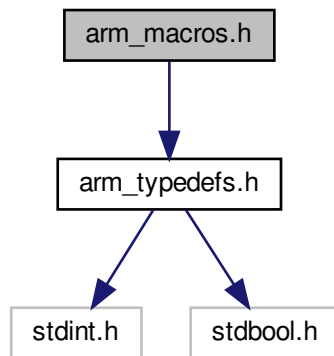
MIT (c) 2017, SMFSW

## 4.3 arm\_macros.h File Reference

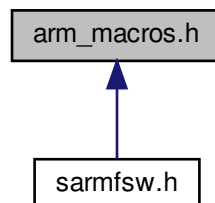
ARM common macros.

```
#include "arm_typedefs.h"
```

Include dependency graph for arm\_macros.h:



This graph shows which files directly or indirectly include this file:



#### Macros

- `#define Undefined -1`  
*Undefined value.*
- `#define Null 0`  
*Null Value.*
- `#define pNull (void *) 0`  
*Null pointer -> same as NULL in Stdlib.h.*
- `#define charNUL '\0'`  
*Null Char.*
- `#define MAKEWORD(b1, b2) (((WORD) (((BYTE) (b1)) | ((WORD) ((BYTE) (b2))) * 0x100))`

- Make WORD from **b1** and **b2** with **b1** as LSB.

  - #define MAKELONG(w1, w2) (((DWORD) (((WORD) (w1)) | ((DWORD) ((WORD) (w2))) \* 0x10000))
- Make LONG from **w1** and **w2** with **w1** as LSB.

  - #define LOWORD(l) ((WORD) (l))
- Get WORD LSW from LONG **l**.

  - #define HIWORD(l) ((WORD) ((DWORD) (l) / 0x10000))
- Get WORD MSW from LONG **l**.

  - #define LOBYTE(w) ((BYTE) (w))
- Get BYTE LSB from WORD **w**.

  - #define HIBYTE(w) ((BYTE) ((WORD) (w) / 0x100))
- Get BYTE MSB from WORD **w**.

  - #define BYTE\_TO\_PERC(b) ((BYTE) (((b) \* 100) / 255))
- Converts a BYTE **b** (0-255) to percent (0-100)

  - #define PERC\_TO\_BYTE(p) ((BYTE) (((p) > 100 ? 100 : (p)) \* 255 / 100))
- Converts a BYTE **p** percentage (0-100) to BYTE (0-255) with max checking.

  - #define TPSSUP\_MS(v, t) ((DWORD) (HAL\_GetTick() - (DWORD) (v)) > (DWORD) (t))
- Tests if **v** (a HStart variable) has reached time lapse stated in **t** (ms)

  - #define TPSINF\_MS(v, t) ((DWORD) (HAL\_GetTick() - (DWORD) (v)) < (DWORD) (t))
- Tests if **v** (a HStart variable) has not reached time lapse stated in **t** (ms)

  - #define OFFSETOF(typ, mbr) ((size\_t) &(((typ\*)0)->mbr))
- Computes the offset member **mbr** from struct **typ**.

  - #define CAT(a, b) a##b
- Preprocessor Name concatenation.

  - #define XCAT(a, b) CAT(a, b)
- Preprocessor Name concatenation (possible nesting)

  - #define STR(s) ("\"#s)
- Stringify an expression.

  - #define binEval(exp) ((exp) ? true : false)
- boolean evaluation of expression **exp**

  - #define nbinEval(exp) (!binEval(exp))
- complemented boolean evaluation of expression **exp**

  - #define max(a, b) ((a) >= (b) ? (a) : (b))
- Returns max value between **a** and **b**.

  - #define min(a, b) ((a) <= (b) ? (a) : (b))
- Returns min value between **a** and **b**.

  - #define MIN3(a, b, c) ((b) <= (c) ? ((a) <= (b) ? (a) : (b)) : ((a) <= (c) ? (a) : (c)))
- Returns max value between **a**, **b** and **c**.

  - #define MAX3(a, b, c) ((b) >= (c) ? ((a) >= (b) ? (a) : (b)) : ((a) >= (c) ? (a) : (c)))
- Returns min value between **a**, **b** and **c**.

  - #define CLAMP(v, min, max) ((v) < (min) ? (min) : ((v) > (max) ? (max) : (v)))
- Returns the value between **min** and **max** from **val**.

  - #define OneThird ((float) (1.0 / 3.0))
- 1/3 approximation

  - #define TwoThird ((float) (2.0 / 3.0))
- 2/3 approximation

  - #define Pi 3.141593f
- Approximate Pi calculation (4 \* atan(1))

  - #define RADIAN\_TO\_FLOAT(r) ((float) (((r) > 2\*Pi ? 2\*Pi : (r)) / 2\*Pi))
- #define FLOAT\_TO\_RADIAN(f) ((float) (((f) > 1.0f ? 1.0f : (f) < 0.0f ? 0.0f : (f)) \* 2\*Pi))
- #define DEGREE\_TO\_FLOAT(d) ((float) (((d) > 360.0f ? 360.0f : (d)) / 360.0f))
- #define FLOAT\_TO\_DEGREE(f) ((float) (((f) > 1.0f ? 1.0f : (f) < 0.0f ? 0.0f : (f)) \* 360.0f))



- #define `conv8upto16Bits`(val, nb) (((WORD) (((WORD) ((val) << (nb)) + (WORD) ((val) & (0xFF >> (8-(nb)))))
- converts **val** (8bits) to **8+nb** bits (nb must be comprised between 0 & 8 bits)
- #define `conv16upto32Bits`(val, nb) (((DWORD) (((DWORD) ((val) << (nb)) + (DWORD) ((val) & (0xFFFF >> (16-(nb)))))
- converts **val** (16bits) to **16+nb** bits (nb must be comprised between 0 & 16 bits)
- #define `SWAP_BYTE`(a, b) { BYTE c; c=(a); a=(b); b=c; }
- Swap BYTES **a** & **b**.
- #define `SWAP_WORD`(a, b) { WORD c; c=(a); a=(b); b=c; }
- Swap WORDs **a** & **b**.
- #define `SWAP_DWORD`(a, b) { DWORD c; c=(a); a=(b); b=c; }
- Swap DWORDs **a** & **b**.
- #define `SWAP_END16B`(w) (((WORD) (((WORD) (w) & 0x00FFU) \* 0x100) | (((WORD) (w) & 0xFF00U) / 0x100)))
- #define `SWAP_END32B`(d) (((DWORD) ((SWAP\_END16B((DWORD) (d) & 0xFFFFFUL) \* 0x10000) | (SWAP\_END16B((DWORD) (d) & 0xFFFF0000U) / 0x10000)))
- Swap 32bits endian.

## Functions

- void `INLINE__ SWAP_END16B_TAB` (uint16\_t tab[], uint16\_t nb)
- Swap endians of a 16b tab.
- void `INLINE__ SWAP_END32B_TAB` (uint32\_t tab[], uint16\_t nb)
- Swap endians of a 32b tab.
- bool `INLINE__ inTolerance` (int16\_t val, int16\_t ref, int16\_t tolerance)
- Checks if val given as parameter is in tolerance.
- bool `INLINE__ inRange` (int16\_t val, int16\_t low, int16\_t high)
- Checks if val given as parameter is in range.

### 4.3.1 Detailed Description

ARM common macros.

#### Author

SMFSW

#### Version

v0.8

#### Date

2017

#### Copyright

MIT (c) 2017, SMFSW

### 4.3.2 Macro Definition Documentation

#### 4.3.2.1 `#define binEval( exp ) ((exp) ? true : false)`

boolean evaluation of expression **exp**

#### 4.3.2.2 `#define BYTE_TO_PERC( b ) ((BYTE) (((b) * 100) / 255))`

Converts a BYTE **b** (0-255) to percent (0-100)

#### 4.3.2.3 `#define CAT( a, b ) a##b`

Preprocessor Name concatenation.

#### Warning

No possible nesting, use *XCAT* in this case

#### 4.3.2.4 `#define charNUL '\0'`

Null Char.

#### 4.3.2.5 `#define CLAMP( v, min, max ) ((v) < (min) ? (min) : ((v) > (max) ? (max) : (v)))`

Returns the value between **min** and **max** from **val**.

#### 4.3.2.6 `#define conv16upto32Bits( val, nb ) ((DWORD) ((DWORD) ((val) << (nb)) + (DWORD) ((val) & (0xFFFF >> (16-(nb)))))`

converts **val** (16bits) to **16+nb** bits (nb must be comprised between 0 & 16 bits)

#### Warning

conversion output shall not exceed 32bits (input shall strictly be unsigned 16bits)  
nb shall be in range 0-16 (note that using 0 doesn't change val)

#### 4.3.2.7 `#define conv8upto16Bits( val, nb ) ((WORD) ((WORD) ((val) << (nb)) + (WORD) ((val) & (0xFF >> (8-(nb)))))`

converts **val** (8bits) to **8+nb** bits (nb must be comprised between 0 & 8 bits)

#### Warning

conversion output shall not exceed 16bits (input shall strictly be unsigned 8bits)  
nb shall be in range 0-8 (note that using 0 doesn't change val)

4.3.2.8 `#define DEGREE_TO_FLOAT( d ) ((float) (((d) > 360.0f ? 360.0f : (d)) / 360.0f))`

4.3.2.9 `#define FLOAT_TO_DEGREE( f ) ((float) (((f) > 1.0f ? 1.0f : (f)) < 0.0f ? 0.0f : (f)) * 360.0f)`

4.3.2.10 `#define FLOAT_TO_RADIAN( f ) ((float) (((f) > 1.0f ? 1.0f : (f)) < 0.0f ? 0.0f : (f)) * 2*Pi)`

4.3.2.11 `#define HIBYTE( w ) ((BYTE) ((WORD) (w) / 0x100))`

Get BYTE MSB from WORD **w**.

4.3.2.12 `#define HIWORD( l ) ((WORD) ((DWORD) (l) / 0x10000))`

Get WORD MSW from LONG **l**.

4.3.2.13 `#define LOBYTE( w ) ((BYTE) (w))`

Get BYTE LSB from WORD **w**.

4.3.2.14 `#define LOWORD( l ) ((WORD) (l))`

Get WORD LSW from LONG **l**.

4.3.2.15 `#define MAKELONG( w1, w2 ) ((DWORD) (((WORD) (w1)) | ((DWORD) ((WORD) (w2))) * 0x10000))`

Make LONG from **w1** and **w2** with **w1** as LSB.

4.3.2.16 `#define MAKEWORD( b1, b2 ) ((WORD) (((BYTE) (b1)) | ((WORD) ((BYTE) (b2))) * 0x100))`

Make WORD from **b1** and **b2** with **b1** as LSB.

4.3.2.17 `#define max( a, b ) ((a) >= (b) ? (a) : (b))`

Returns max value between **a** and **b**.

4.3.2.18 `#define MAX3( a, b, c ) ((b) >= (c) ? ((a) >= (b) ? (a) : (b)) : ((a) >= (c) ? (a) : (c))`

Returns min value between **a**, **b** and **c**.

4.3.2.19 `#define min( a, b ) ((a) <= (b) ? (a) : (b))`

Returns min value between **a** and **b**.

4.3.2.20 `#define MIN3( a, b, c ) ((b) <= (c) ? ((a) <= (b) ? (a) : (b)) : ((a) <= (c) ? (a) : (c))`

Returns max value between **a**, **b** and **c**.

4.3.2.21 `#define nbinEval( exp ) (!binEval(exp))`

complemented boolean evaluation of expression **exp**

4.3.2.22 `#define Null 0`

Null Value.

4.3.2.23 `#define OFFSETOF( typ, mbr ) ((size_t) &(((typ*)0)->mbr))`

Computes the offset member **mbr** from struct **typ**.

4.3.2.24 `#define OneThird ((float) (1.0 / 3.0))`

1/3 approximation

4.3.2.25 `#define PERC_TO_BYTE( p ) ((BYTE) (((p) > 100 ? 100 : (p)) * 255 / 100))`

Converts a BYTE **p** percentage (0-100) to BYTE (0-255) with max checking.

4.3.2.26 `#define Pi 3.141593f`

Approximate Pi calculation ( $4 * \text{atan}(1)$ )

4.3.2.27 `#define pNull (void *) 0`

Null pointer -> same as NULL in Stdlib.h.

4.3.2.28 `#define RADIAN_TO_FLOAT( r ) ((float) (((r) > 2*Pi ? 2*Pi : (r)) / 2*Pi))`4.3.2.29 `#define STR( s ) (" " #s)`

Stringify an expression.

4.3.2.30 `#define SWAP_BYTE( a, b ) { BYTE c; c=(a); a=(b); b=c; }`

Swap BYTES **a** & **b**.

4.3.2.31 `#define SWAP_DWORD( a, b ) { DWORD c; c=(a); a=(b); b=c; }`

Swap DWORDs **a** & **b**.

4.3.2.32 `#define SWAP_END16B( w ) ((WORD) (((WORD) (w) & 0x00FFU) * 0x100) | (((WORD) (w) & 0xFF00U) / 0x100)))`

Swap 16bits endian

4.3.2.33 `#define SWAP_END32B( d ) ((DWORD) ((SWAP_END16B((DWORD) (d) & 0xFFFFUL) * 0x10000) | (SWAP_END16B((DWORD) (d) & 0xFFFF0000U) / 0x10000)))`

Swap 32bits endian.

Warning

SWAP\_END32B needs some testing

4.3.2.34 `#define SWAP_WORD( a, b ) { WORD c; c=(a); a=(b); b=c; }`

Swap WORDs **a** & **b**.

4.3.2.35 `#define TPSINF_MS( v, t ) ((DWORD) (HAL_GetTick() - (DWORD) (v)) < (DWORD) (t))`

Tests if **v** (a HStart variable) has not reached time lapse stated in **t** (ms)

4.3.2.36 `#define TPSSUP_MS( v, t ) ((DWORD) (HAL_GetTick() - (DWORD) (v)) > (DWORD) (t))`

Tests if **v** (a HStart variable) has reached time lapse stated in **t** (ms)

4.3.2.37 `#define TwoThird ((float) (2.0 / 3.0))`

2/3 approximation

4.3.2.38 `#define Undefined -1`

Undefined value.

4.3.2.39 `#define XCAT( a, b ) CAT(a, b)`

Preprocessor Name concatenation (possible nesting)

### 4.3.3 Function Documentation

4.3.3.1 `bool INLINE__ inRange ( int16_t val, int16_t low, int16_t high ) [inline]`

Checks if val given as parameter is in range.

#### Parameters

in	<i>val</i>	- Value to check
in	<i>low</i>	- Low range boundary
in	<i>high</i>	- High range boundary

#### Returns

True if val is inRange

4.3.3.2 `bool INLINE__ inTolerance ( int16_t val, int16_t ref, int16_t tolerance ) [inline]`

Checks if val given as parameter is in tolerance.

#### Parameters

in	<i>val</i>	- Value to check
in	<i>ref</i>	- Reference value
in	<i>tolerance</i>	- Tolerance on reference value

**Returns**

True if val is inTolerance

**4.3.3.3** void **INLINE\_\_SWAP\_END16B\_TAB** ( uint16\_t *tab*[], uint16\_t *nb* ) [inline]

Swap endians of a 16b tab.

**Parameters**

in	<i>tab</i>	- tab of 16b values
in	<i>nb</i>	- nb of values in tab

**4.3.3.4** void **INLINE\_\_SWAP\_END32B\_TAB** ( uint32\_t *tab*[], uint16\_t *nb* ) [inline]

Swap endians of a 32b tab.

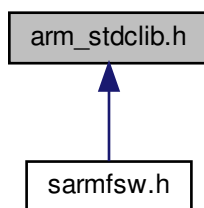
**Parameters**

in	<i>tab</i>	- tab of 32b values
in	<i>nb</i>	- nb of values in tab

**4.4 arm\_stdclib.h File Reference**

ARM common standard c library wrapper macros.

This graph shows which files directly or indirectly include this file:

**Macros**

- #define **printExpr**(e) (printf("%s = %d\n", #e, (e)))  
Print expression **e** and it's result **e** using printf.
- #define **verblnstr**(i) (printf(" #i), (i))  
Print instruction **e** and execute it.
- #define **str\_clr**(s) (s[0] = '\0')

- clear string **s** (fast way)*
  - #define `str_clr_safe(s)` (`memset('\0', s, sizeof(s))`)
- clear string **s** (safe way)*
  - #define `str_add_tab(s)` (`strcat(s, '\t')`)
- Adding tab to string using strcat.*
  - #define `str_add_cr(s)` (`strcat(s, '\n')`)
- Adding new line to string using strcat.*
  - #define `VerboseInc(x)` (`puts("Incrementing " #x), (x)++`)
- Increment example using puts.*
  - #define `TestMalloc(x)` (`((x) = malloc(sizeof(*x)), assert(x))`)
- Asserted malloc.*

#### 4.4.1 Detailed Description

ARM common standard c library wrapper macros.

##### Author

SMFSW

##### Version

v0.8

##### Date

2017

##### Copyright

MIT (c) 2017, SMFSW

#### 4.4.2 Macro Definition Documentation

##### 4.4.2.1 #define printExpr( *e* ) (printf("%s = %d\n", #e, (e)))

Print expression *e* and it's result *e* using printf.

##### 4.4.2.2 #define str\_add\_cr( *s* ) (strcat(s, '\n'))

Adding new line to string using strcat.

##### 4.4.2.3 #define str\_add\_tab( *s* ) (strcat(s, '\t'))

Adding tab to string using strcat.

##### 4.4.2.4 #define str\_clr( *s* ) (s[0] = '\0')

clear string *s* (fast way)

4.4.2.5 `#define str_clr_safe( s ) (memset('\0', s, sizeof(s)))`

clear string **s** (safe way)

4.4.2.6 `#define TestMalloc( x ) ((x) = malloc(sizeof(*x)), assert(x))`

Asserted malloc.

4.4.2.7 `#define verblnstr( i ) (printf(" %i), (i))`

Print instruction **e** and execute it.

4.4.2.8 `#define VerboseInc( x ) (puts("Incrementing " #x), (x)++)`

Increment example using puts.

## 4.5 arm\_stm32.h File Reference

ARM common macros for STM32.

### Macros

- `#define port(mnem) XCAT(mnem, _GPIO_Port)`  
*Wrapper for PORT Alias.*
- `#define pin(mnem) XCAT(mnem, _Pin)`  
*Wrapper for PIN Alias.*
- `#define gpio(mnem) port(mnem), pin(mnem)`  
*Wrapper for PORT/PIN Alias (when using HAL\_GPIO\_ReadPin for example)*
- `#define STM_HEADER(f) XCAT(<stm32, XCAT(f, xx.h>))`  
*concatenate <stm32(f)xx.h> name following stm family **f***
- `#define STM_CONF_HEADER(f) XCAT(<stm32, XCAT(f, xx_hal_conf.h>))`  
*concatenate <stm32(f)xx.h> name following stm family **f***
- `#define STM32_INC STM_HEADER(STM_FAMILY)`  
*Alias for STM32 include.*
- `#define STM32_CFG STM_CONF_HEADER(STM_FAMILY)`  
*Alias for STM32 include.*

### 4.5.1 Detailed Description

ARM common macros for STM32.

#### Author

SMFSW

#### Version

v0.8

#### Date

2017

#### Copyright

MIT (c) 2017, SMFSW



#### 4.5.2 Macro Definition Documentation

##### 4.5.2.1 #define gpio( *mnem* ) port(mnem), pin(mnem)

Wrapper for PORT/PIN Alias (when using HAL\_GPIO\_ReadPin for example)

##### 4.5.2.2 #define pin( *mnem* ) XCAT(mnem, \_Pin)

Wrapper for PIN Alias.

##### 4.5.2.3 #define port( *mnem* ) XCAT(mnem, \_GPIO\_Port)

Wrapper for PORT Alias.

##### 4.5.2.4 #define STM32\_CFG STM\_CONF\_HEADER(STM\_FAMILY)

Alias for STM32 include.

##### 4.5.2.5 #define STM32\_INC STM\_HEADER(STM\_FAMILY)

Alias for STM32 include.

##### 4.5.2.6 #define STM\_CONF\_HEADER( *f* ) XCAT(<stm32, XCAT(f, xx\_hal\_conf.h>))

concatenate <stm32(f)xx.h> name following stm family *f*

##### 4.5.2.7 #define STM\_HEADER( *f* ) XCAT(<stm32, XCAT(f, xx.h>))

concatenate <stm32(f)xx.h> name following stm family *f*

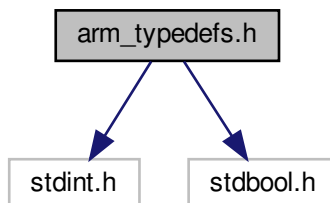
## 4.6 arm\_typedefs.h File Reference

ARM common typedefs.

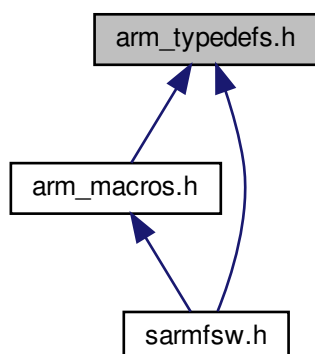
```
#include <stdint.h>
```

```
#include <stdbool.h>
```

Include dependency graph for arm\_typedefs.h:



This graph shows which files directly or indirectly include this file:



## Classes

- struct [StructBitfield8](#)  
*Bitfield 8b.*
- struct [StructBitfield16](#)  
*Bitfield 16b.*
- struct [StructBitfield32](#)  
*Bitfield 32b.*
- struct [StructBitfield64](#)  
*Bitfield 64b.*
- union [UnionByte](#)  
*Union for BYTE.*
- union [UnionWord](#)  
*Union for WORD.*
- union [UnionDWord](#)  
*Union for DWORD.*
- union [UnionLWord](#)  
*Union for LWORD.*

## Typedefs

- typedef char [CHAR](#)  
*Char typedef (8bits)*
- typedef uint8\_t [BYTE](#)  
*Unsigned Byte typedef (8bits)*
- typedef uint16\_t [WORD](#)  
*Unsigned Word typedef (16bits)*
- typedef uint32\_t [DWORD](#)  
*Unsigned dWord typedef (32bits)*
- typedef uint64\_t [LWORD](#)

- Unsigned IWord typedef (64bits)*
  - typedef int8\_t [SBYTE](#)
- Signed Byte typedef (8bits)*
  - typedef int16\_t [SWORD](#)
- Signed Word typedef (16bits)*
  - typedef int32\_t [SDWORD](#)
- Signed dWord typedef (32bits)*
  - typedef int64\_t [SLWORD](#)
- Signed IWord typedef (64bits)*
  - typedef enum [eState](#) [eState](#)
  - typedef enum [eEdge](#) [eEdge](#)
  - typedef struct [StructBitfield8](#) [sBitfield8](#)
  - typedef struct [StructBitfield16](#) [sBitfield16](#)
  - typedef struct [StructBitfield32](#) [sBitfield32](#)
  - typedef struct [StructBitfield64](#) [sBitfield64](#)
  - typedef union [UnionByte](#) [uByte](#)
  - typedef union [UnionWord](#) [uWord](#)
  - typedef union [UnionDWord](#) [uDWord](#)
  - typedef union [UnionLWord](#) [uLWord](#)

## Enumerations

- enum [eState](#) { [Off](#) = 0U, [On](#) = 1U }
- Activation state On, Off.*
- enum [eEdge](#) { [NoEdge](#) = 0, [Rising](#), [Falling](#) }
- Signal Edges.*

### 4.6.1 Detailed Description

ARM common typedefs.

#### Author

SMFSW

#### Version

v0.8

#### Date

2017

#### Copyright

MIT (c) 2017, SMFSW

## 4.6.2 Typedef Documentation

### 4.6.2.1 typedef uint8\_t BYTE

Unsigned Byte typedef (8bits)

### 4.6.2.2 typedef char CHAR

Char typedef (8bits)

### 4.6.2.3 typedef uint32\_t DWORD

Unsigned dWord typedef (32bits)

### 4.6.2.4 typedef enum eEdge eEdge

### 4.6.2.5 typedef enum eState eState

### 4.6.2.6 typedef uint64\_t LWORD

Unsigned lWord typedef (64bits)

### 4.6.2.7 typedef struct StructBitfield16 sBitfield16

### 4.6.2.8 typedef struct StructBitfield32 sBitfield32

### 4.6.2.9 typedef struct StructBitfield64 sBitfield64

### 4.6.2.10 typedef struct StructBitfield8 sBitfield8

### 4.6.2.11 typedef int8\_t SBYTE

Signed Byte typedef (8bits)

### 4.6.2.12 typedef int32\_t SDWORD

Signed dWord typedef (32bits)

### 4.6.2.13 typedef int64\_t SLWORD

Signed lWord typedef (64bits)

### 4.6.2.14 typedef int16\_t SWORD

Signed Word typedef (16bits)

4.6.2.15 typedef union **UnionByte** uByte

4.6.2.16 typedef union **UnionDWord** uDWord

4.6.2.17 typedef union **UnionLWord** uLWord

4.6.2.18 typedef union **UnionWord** uWord

4.6.2.19 typedef uint16\_t **WORD**

Unsigned Word typedef (16bits)

### 4.6.3 Enumeration Type Documentation

#### 4.6.3.1 enum eEdge

Signal Edges.

Enumerator

**NoEdge** No change.

**Rising** Rising edge.

**Falling** Falling edge.

#### 4.6.3.2 enum eState

Activation state On, Off.

Enumerator

**Off** Off / Clear.

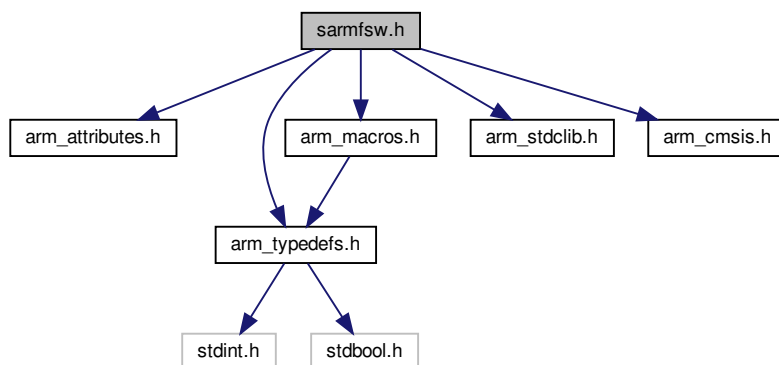
**On** On / Set.

## 4.7 sarmfsw.h File Reference

ARM common headers for projects.

```
#include "arm_attributes.h"
#include "arm_typedefs.h"
#include "arm_macros.h"
#include "arm_stdclib.h"
#include "arm_cmsis.h"
```

Include dependency graph for sarmfsw.h:



## Typedefs

- typedef enum [FW\\_target](#) [FW\\_target](#)

## Enumerations

- enum [FW\\_target](#) {  
    [DefSpecialTarget](#) = 0, [DefDebugTarget](#), [DefReleaseTarget](#), [DefFUBARTarget](#),  
    [DefUnknownTarget](#) = 0xFF }  
    *Firmware target types.*

### 4.7.1 Detailed Description

ARM common headers for projects.

#### Author

SMFSW

#### Version

v0.8

#### Date

2017

#### Copyright

MIT (c) 2017, SMFSW

### 4.7.2 Typedef Documentation

#### 4.7.2.1 typedef enum [FW\\_target](#) [FW\\_target](#)

### 4.7.3 Enumeration Type Documentation

#### 4.7.3.1 enum [FW\\_target](#)

Firmware target types.

#### Enumerator

**[DefSpecialTarget](#)** Special FW target (same as debug, yet)

**[DefDebugTarget](#)** Debug FW target (default)

**[DefReleaseTarget](#)** Release FW target (No debug information)

**[DefFUBARTarget](#)** FUBAR FW target (shall be used only for stress/testing purposes)

**[DefUnknownTarget](#)** Unknown FW target (should never happen!)

## Index

- arm\_attributes.h, [27](#)
  - INLINE\_, [28](#)
  - PACK\_, [28](#)
  - WEAK\_, [28](#)
- arm\_cmsis.h, [28](#)
- arm\_macros.h, [29](#)
  - BYTE\_TO\_PERC, [32](#)
  - binEval, [32](#)
  - CAT, [32](#)
  - CLAMP, [32](#)
  - charNUL, [32](#)
  - conv16upto32Bits, [32](#)
  - conv8upto16Bits, [32](#)
  - DEGREE\_TO\_FLOAT, [32](#)
  - FLOAT\_TO\_DEGREE, [33](#)
  - FLOAT\_TO\_RADIAN, [33](#)
  - HIBYTE, [33](#)
  - HIWORD, [33](#)
  - inRange, [35](#)
  - inTolerance, [35](#)
  - LOBYTE, [33](#)
  - LOWORD, [33](#)
  - MAKELONG, [33](#)
  - MAKEWORD, [33](#)
  - MAX3, [33](#)
  - MIN3, [33](#)
  - max, [33](#)
  - min, [33](#)
  - nbinEval, [33](#)
  - Null, [33](#)
  - OFFSETOF, [34](#)
  - OneThird, [34](#)
  - PERC\_TO\_BYTE, [34](#)
  - pNull, [34](#)
  - Pi, [34](#)
  - RADIAN\_TO\_FLOAT, [34](#)
  - STR, [34](#)
  - SWAP\_BYTE, [34](#)
  - SWAP\_DWORD, [34](#)
  - SWAP\_END16B\_TAB, [36](#)
  - SWAP\_END16B, [34](#)
  - SWAP\_END32B\_TAB, [36](#)
  - SWAP\_END32B, [34](#)
  - SWAP\_WORD, [34](#)
  - TPSINF\_MS, [35](#)
  - TPSSUP\_MS, [35](#)
  - TwoThird, [35](#)
  - Undefined, [35](#)
  - XCAT, [35](#)
- arm\_stdclib.h, [36](#)
  - printExpr, [37](#)
  - str\_add\_cr, [37](#)
  - str\_add\_tab, [37](#)
  - str\_clr, [37](#)
  - str\_clr\_safe, [37](#)
  - TestMalloc, [38](#)
  - verbInstr, [38](#)
  - VerboseInc, [38](#)
- arm\_stm32.h, [38](#)
  - gpio, [39](#)
  - pin, [39](#)
  - port, [39](#)
  - STM32\_CFG, [39](#)
  - STM32\_INC, [39](#)
  - STM\_CONF\_HEADER, [39](#)
  - STM\_HEADER, [39](#)
- arm\_typedefs.h, [39](#)
  - BYTE, [42](#)
  - CHAR, [42](#)
  - DWORD, [42](#)
  - eEdge, [42](#), [43](#)
  - eState, [42](#), [43](#)
  - Falling, [43](#)
  - LWORD, [42](#)
  - NoEdge, [43](#)
  - Off, [43](#)
  - On, [43](#)
  - Rising, [43](#)
  - SBYTE, [42](#)
  - sBitfield16, [42](#)
  - sBitfield32, [42](#)
  - sBitfield64, [42](#)
  - sBitfield8, [42](#)
  - SDWORD, [42](#)
  - SLWORD, [42](#)
  - SWORD, [42](#)
  - uByte, [42](#)
  - uDWord, [43](#)
  - uLWord, [43](#)
  - uWord, [43](#)
  - WORD, [43](#)
- B0
  - UnionDWord, [21](#)
  - UnionLWord, [23](#)
  - UnionWord, [26](#)
- b0
  - StructBitfield16, [3](#)
  - StructBitfield32, [6](#)
  - StructBitfield64, [12](#)
  - StructBitfield8, [18](#)
- B1
  - UnionDWord, [21](#)
  - UnionLWord, [23](#)
  - UnionWord, [26](#)
- b1
  - StructBitfield16, [3](#)
  - StructBitfield32, [6](#)
  - StructBitfield64, [12](#)
  - StructBitfield8, [18](#)

- b10
  - StructBitfield16, [3](#)
  - StructBitfield32, [6](#)
  - StructBitfield64, [12](#)
- b11
  - StructBitfield16, [4](#)
  - StructBitfield32, [7](#)
  - StructBitfield64, [12](#)
- b12
  - StructBitfield16, [4](#)
  - StructBitfield32, [7](#)
  - StructBitfield64, [12](#)
- b13
  - StructBitfield16, [4](#)
  - StructBitfield32, [7](#)
  - StructBitfield64, [12](#)
- b14
  - StructBitfield16, [4](#)
  - StructBitfield32, [7](#)
  - StructBitfield64, [12](#)
- b15
  - StructBitfield16, [4](#)
  - StructBitfield32, [7](#)
  - StructBitfield64, [12](#)
- b16
  - StructBitfield32, [7](#)
  - StructBitfield64, [13](#)
- b17
  - StructBitfield32, [7](#)
  - StructBitfield64, [13](#)
- b18
  - StructBitfield32, [7](#)
  - StructBitfield64, [13](#)
- b19
  - StructBitfield32, [7](#)
  - StructBitfield64, [13](#)
- B2
  - UnionDWord, [21](#)
  - UnionLWord, [23](#)
- b2
  - StructBitfield16, [4](#)
  - StructBitfield32, [7](#)
  - StructBitfield64, [13](#)
  - StructBitfield8, [18](#)
- b20
  - StructBitfield32, [7](#)
  - StructBitfield64, [13](#)
- b21
  - StructBitfield32, [7](#)
  - StructBitfield64, [13](#)
- b22
  - StructBitfield32, [8](#)
  - StructBitfield64, [13](#)
- b23
  - StructBitfield32, [8](#)
  - StructBitfield64, [13](#)
- b24
  - StructBitfield32, [8](#)
- StructBitfield64, [13](#)
- b25
  - StructBitfield32, [8](#)
  - StructBitfield64, [13](#)
- b26
  - StructBitfield32, [8](#)
  - StructBitfield64, [13](#)
- b27
  - StructBitfield32, [8](#)
  - StructBitfield64, [14](#)
- b28
  - StructBitfield32, [8](#)
  - StructBitfield64, [14](#)
- b29
  - StructBitfield32, [8](#)
  - StructBitfield64, [14](#)
- B3
  - UnionDWord, [21](#)
  - UnionLWord, [23](#)
- b3
  - StructBitfield16, [4](#)
  - StructBitfield32, [8](#)
  - StructBitfield64, [14](#)
  - StructBitfield8, [18](#)
- b30
  - StructBitfield32, [8](#)
  - StructBitfield64, [14](#)
- b31
  - StructBitfield32, [8](#)
  - StructBitfield64, [14](#)
- b32
  - StructBitfield64, [14](#)
- b33
  - StructBitfield64, [14](#)
- b34
  - StructBitfield64, [14](#)
- b35
  - StructBitfield64, [14](#)
- b36
  - StructBitfield64, [14](#)
- b37
  - StructBitfield64, [14](#)
- b38
  - StructBitfield64, [15](#)
- b39
  - StructBitfield64, [15](#)
- B4
  - UnionLWord, [24](#)
- b4
  - StructBitfield16, [4](#)
  - StructBitfield32, [8](#)
  - StructBitfield64, [15](#)
  - StructBitfield8, [18](#)
- b40
  - StructBitfield64, [15](#)
- b41
  - StructBitfield64, [15](#)
- b42



- StructBitfield64, [15](#)
- b43
  - StructBitfield64, [15](#)
- b44
  - StructBitfield64, [15](#)
- b45
  - StructBitfield64, [15](#)
- b46
  - StructBitfield64, [15](#)
- b47
  - StructBitfield64, [15](#)
- b48
  - StructBitfield64, [15](#)
- b49
  - StructBitfield64, [16](#)
- B5
  - UnionLWord, [24](#)
- b5
  - StructBitfield16, [4](#)
  - StructBitfield32, [9](#)
  - StructBitfield64, [16](#)
  - StructBitfield8, [18](#)
- b50
  - StructBitfield64, [16](#)
- b51
  - StructBitfield64, [16](#)
- b52
  - StructBitfield64, [16](#)
- b53
  - StructBitfield64, [16](#)
- b54
  - StructBitfield64, [16](#)
- b55
  - StructBitfield64, [16](#)
- b56
  - StructBitfield64, [16](#)
- b57
  - StructBitfield64, [16](#)
- b58
  - StructBitfield64, [16](#)
- b59
  - StructBitfield64, [16](#)
- B6
  - UnionLWord, [24](#)
- b6
  - StructBitfield16, [4](#)
  - StructBitfield32, [9](#)
  - StructBitfield64, [17](#)
  - StructBitfield8, [18](#)
- b60
  - StructBitfield64, [17](#)
- b61
  - StructBitfield64, [17](#)
- b62
  - StructBitfield64, [17](#)
- b63
  - StructBitfield64, [17](#)
- B7
  - UnionLWord, [24](#)
- b7
  - StructBitfield16, [4](#)
  - StructBitfield32, [9](#)
  - StructBitfield64, [17](#)
  - StructBitfield8, [19](#)
- b8
  - StructBitfield16, [4](#)
  - StructBitfield32, [9](#)
  - StructBitfield64, [17](#)
- b9
  - StructBitfield16, [5](#)
  - StructBitfield32, [9](#)
  - StructBitfield64, [17](#)
- BYTE\_TO\_PERC
  - arm\_macros.h, [32](#)
- BYTE
  - arm\_typedefs.h, [42](#)
- binEval
  - arm\_macros.h, [32](#)
- Bits
  - UnionByte, [19](#)
  - UnionDWord, [21](#)
  - UnionLWord, [24](#)
  - UnionWord, [26](#)
- Byte
  - UnionByte, [19](#)
  - UnionDWord, [21](#)
  - UnionLWord, [24](#)
  - UnionWord, [26](#)
- Bytes
  - UnionDWord, [21](#)
  - UnionLWord, [24](#)
  - UnionWord, [26](#)
- CAT
  - arm\_macros.h, [32](#)
- CHAR
  - arm\_typedefs.h, [42](#)
- CLAMP
  - arm\_macros.h, [32](#)
- charNUL
  - arm\_macros.h, [32](#)
- conv16upto32Bits
  - arm\_macros.h, [32](#)
- conv8upto16Bits
  - arm\_macros.h, [32](#)
- D0
  - UnionLWord, [24](#)
- D1
  - UnionLWord, [24](#)
- DEGREE\_TO\_FLOAT
  - arm\_macros.h, [32](#)
- DWORD
  - arm\_typedefs.h, [42](#)
- DWord
  - UnionDWord, [21](#)
  - UnionLWord, [24](#)

DWords  
     UnionLWord, 24  
 DefDebugTarget  
     sarmfsw.h, 44  
 DefFUBARTarget  
     sarmfsw.h, 44  
 DefReleaseTarget  
     sarmfsw.h, 44  
 DefSpecialTarget  
     sarmfsw.h, 44  
 DefUnknownTarget  
     sarmfsw.h, 44  
  
 eEdge  
     arm\_typedefs.h, 42, 43  
 eState  
     arm\_typedefs.h, 42, 43  
  
 FLOAT\_TO\_DEGREE  
     arm\_macros.h, 33  
 FLOAT\_TO\_RADIAN  
     arm\_macros.h, 33  
 FW\_target  
     sarmfsw.h, 44  
 Falling  
     arm\_typedefs.h, 43  
  
 gpio  
     arm\_stm32.h, 39  
  
 HIBYTE  
     arm\_macros.h, 33  
 HIWORD  
     arm\_macros.h, 33  
  
 INLINE\_\_  
     arm\_attributes.h, 28  
 inRange  
     arm\_macros.h, 35  
 inTolerance  
     arm\_macros.h, 35  
  
 LOBYTE  
     arm\_macros.h, 33  
 LOWORD  
     arm\_macros.h, 33  
 LWORD  
     arm\_typedefs.h, 42  
 LWord  
     UnionLWord, 24  
  
 MAKELONG  
     arm\_macros.h, 33  
 MAKEWORD  
     arm\_macros.h, 33  
 MAX3  
     arm\_macros.h, 33  
 MIN3  
     arm\_macros.h, 33  
 max  
     arm\_macros.h, 33  
     min  
         arm\_macros.h, 33  
  
 nbinEval  
     arm\_macros.h, 33  
 NoEdge  
     arm\_typedefs.h, 43  
 Null  
     arm\_macros.h, 33  
  
 OFFSETOF  
     arm\_macros.h, 34  
 Off  
     arm\_typedefs.h, 43  
 On  
     arm\_typedefs.h, 43  
 OneThird  
     arm\_macros.h, 34  
  
 PACK\_\_  
     arm\_attributes.h, 28  
 PERC\_TO\_BYTE  
     arm\_macros.h, 34  
 pNull  
     arm\_macros.h, 34  
 Pi  
     arm\_macros.h, 34  
 pin  
     arm\_stm32.h, 39  
 port  
     arm\_stm32.h, 39  
 printExpr  
     arm\_stdclib.h, 37  
  
 RADIAN\_TO\_FLOAT  
     arm\_macros.h, 34  
 Rising  
     arm\_typedefs.h, 43  
  
 SBYTE  
     arm\_typedefs.h, 42  
 sBitfield16  
     arm\_typedefs.h, 42  
 sBitfield32  
     arm\_typedefs.h, 42  
 sBitfield64  
     arm\_typedefs.h, 42  
 sBitfield8  
     arm\_typedefs.h, 42  
 SDWORD  
     arm\_typedefs.h, 42  
 SLWORD  
     arm\_typedefs.h, 42  
 STM32\_CFG  
     arm\_stm32.h, 39  
 STM32\_INC  
     arm\_stm32.h, 39  
 STM\_CONF\_HEADER

- arm\_stm32.h, [39](#)
- STM\_HEADER
  - arm\_stm32.h, [39](#)
- STR
  - arm\_macros.h, [34](#)
- SWAP\_BYTE
  - arm\_macros.h, [34](#)
- SWAP\_DWORD
  - arm\_macros.h, [34](#)
- SWAP\_END16B\_TAB
  - arm\_macros.h, [36](#)
- SWAP\_END16B
  - arm\_macros.h, [34](#)
- SWAP\_END32B\_TAB
  - arm\_macros.h, [36](#)
- SWAP\_END32B
  - arm\_macros.h, [34](#)
- SWAP\_WORD
  - arm\_macros.h, [34](#)
- SWORD
  - arm\_typedefs.h, [42](#)
- sarmfsw.h, [43](#)
  - DefDebugTarget, [44](#)
  - DefFUBARTarget, [44](#)
  - DefReleaseTarget, [44](#)
  - DefSpecialTarget, [44](#)
  - DefUnknownTarget, [44](#)
  - FW\_target, [44](#)
- str\_add\_cr
  - arm\_stdclib.h, [37](#)
- str\_add\_tab
  - arm\_stdclib.h, [37](#)
- str\_clr
  - arm\_stdclib.h, [37](#)
- str\_clr\_safe
  - arm\_stdclib.h, [37](#)
- StructBitfield16, [2](#)
  - b0, [3](#)
  - b1, [3](#)
  - b10, [3](#)
  - b11, [4](#)
  - b12, [4](#)
  - b13, [4](#)
  - b14, [4](#)
  - b15, [4](#)
  - b2, [4](#)
  - b3, [4](#)
  - b4, [4](#)
  - b5, [4](#)
  - b6, [4](#)
  - b7, [4](#)
  - b8, [4](#)
  - b9, [5](#)
- StructBitfield32, [5](#)
  - b0, [6](#)
  - b1, [6](#)
  - b10, [6](#)
  - b11, [7](#)
- b12, [7](#)
- b13, [7](#)
- b14, [7](#)
- b15, [7](#)
- b16, [7](#)
- b17, [7](#)
- b18, [7](#)
- b19, [7](#)
- b2, [7](#)
- b20, [7](#)
- b21, [7](#)
- b22, [8](#)
- b23, [8](#)
- b24, [8](#)
- b25, [8](#)
- b26, [8](#)
- b27, [8](#)
- b28, [8](#)
- b29, [8](#)
- b3, [8](#)
- b30, [8](#)
- b31, [8](#)
- b4, [8](#)
- b5, [9](#)
- b6, [9](#)
- b7, [9](#)
- b8, [9](#)
- b9, [9](#)
- StructBitfield64, [9](#)
  - b0, [12](#)
  - b1, [12](#)
  - b10, [12](#)
  - b11, [12](#)
  - b12, [12](#)
  - b13, [12](#)
  - b14, [12](#)
  - b15, [12](#)
  - b16, [13](#)
  - b17, [13](#)
  - b18, [13](#)
  - b19, [13](#)
  - b2, [13](#)
  - b20, [13](#)
  - b21, [13](#)
  - b22, [13](#)
  - b23, [13](#)
  - b24, [13](#)
  - b25, [13](#)
  - b26, [13](#)
  - b27, [14](#)
  - b28, [14](#)
  - b29, [14](#)
  - b3, [14](#)
  - b30, [14](#)
  - b31, [14](#)
  - b32, [14](#)
  - b33, [14](#)
  - b34, [14](#)

- b35, [14](#)
- b36, [14](#)
- b37, [14](#)
- b38, [15](#)
- b39, [15](#)
- b4, [15](#)
- b40, [15](#)
- b41, [15](#)
- b42, [15](#)
- b43, [15](#)
- b44, [15](#)
- b45, [15](#)
- b46, [15](#)
- b47, [15](#)
- b48, [15](#)
- b49, [16](#)
- b5, [16](#)
- b50, [16](#)
- b51, [16](#)
- b52, [16](#)
- b53, [16](#)
- b54, [16](#)
- b55, [16](#)
- b56, [16](#)
- b57, [16](#)
- b58, [16](#)
- b59, [16](#)
- b6, [17](#)
- b60, [17](#)
- b61, [17](#)
- b62, [17](#)
- b63, [17](#)
- b7, [17](#)
- b8, [17](#)
- b9, [17](#)
- StructBitfield8, [17](#)
  - b0, [18](#)
  - b1, [18](#)
  - b2, [18](#)
  - b3, [18](#)
  - b4, [18](#)
  - b5, [18](#)
  - b6, [18](#)
  - b7, [19](#)
- TPSINF\_MS
  - [arm\\_macros.h](#), [35](#)
- TPSSUP\_MS
  - [arm\\_macros.h](#), [35](#)
- TestMalloc
  - [arm\\_stdclib.h](#), [38](#)
- TwoThird
  - [arm\\_macros.h](#), [35](#)
- uByte
  - [arm\\_typedefs.h](#), [42](#)
- uDWord
  - [arm\\_typedefs.h](#), [43](#)
- uLWord
  - [arm\\_typedefs.h](#), [43](#)
- uWord
  - [arm\\_typedefs.h](#), [43](#)
- Undefined
  - [arm\\_macros.h](#), [35](#)
- UnionByte, [19](#)
  - Bits, [19](#)
  - Byte, [19](#)
- UnionDWord, [20](#)
  - B0, [21](#)
  - B1, [21](#)
  - B2, [21](#)
  - B3, [21](#)
  - Bits, [21](#)
  - Byte, [21](#)
  - Bytes, [21](#)
  - DWord, [21](#)
  - W0, [21](#)
  - W1, [21](#)
  - Word, [21](#)
  - Words, [21](#)
- UnionLWord, [22](#)
  - B0, [23](#)
  - B1, [23](#)
  - B2, [23](#)
  - B3, [23](#)
  - B4, [24](#)
  - B5, [24](#)
  - B6, [24](#)
  - B7, [24](#)
  - Bits, [24](#)
  - Byte, [24](#)
  - Bytes, [24](#)
  - D0, [24](#)
  - D1, [24](#)
  - DWord, [24](#)
  - DWords, [24](#)
  - LWord, [24](#)
  - W0, [24](#)
  - W1, [25](#)
  - W2, [25](#)
  - W3, [25](#)
  - Word, [25](#)
  - Words, [25](#)
- UnionWord, [25](#)
  - B0, [26](#)
  - B1, [26](#)
  - Bits, [26](#)
  - Byte, [26](#)
  - Bytes, [26](#)
  - Word, [26](#)
- verblnstr
  - [arm\\_stdclib.h](#), [38](#)
- VerboseInc
  - [arm\\_stdclib.h](#), [38](#)
- W0
  - UnionDWord, [21](#)

- UnionLWord, [24](#)
- W1
  - UnionDWord, [21](#)
  - UnionLWord, [25](#)
- W2
  - UnionLWord, [25](#)
- W3
  - UnionLWord, [25](#)
- WEAK\_\_
  - arm\_attributes.h, [28](#)
- WORD
  - arm\_typedefs.h, [43](#)
- Word
  - UnionDWord, [21](#)
  - UnionLWord, [25](#)
  - UnionWord, [26](#)
- Words
  - UnionDWord, [21](#)
  - UnionLWord, [25](#)
- XCAT
  - arm\_macros.h, [35](#)