

VS1000/Interfacing Reference Manual

Generated by Doxygen 1.4.5

Thu Jul 3 13:30:00 2008

Contents

1	VS1000/Interfacing Data Structure Index	1
2	VS1000/Interfacing File Index	2
3	VS1000/Interfacing Page Index	3
4	VS1000/Interfacing Data Structure Documentation	4
5	VS1000/Interfacing File Documentation	34
6	VS1000/Interfacing Example Documentation	189
7	VS1000/Interfacing Page Documentation	190

1 VS1000/Interfacing Data Structure Index

1.1 VS1000/Interfacing Data Structures

Here are the data structures with brief descriptions:

__sfpos	4
AUDIOPTR	4
Codec	5
CodecServices	6
EARSPEAKER	10
FATINFO	11
FmfMeta	14
FRAGMENT	15
FsMapper	15
FsMapperFlash	17
FsMapperTiny	19
FsNandPhys	21
FsPhysical	22

KeyMapping	24
Player	25
scsicdb10variant1	27
scsicdb10variant2	28
scsicdb6variant1	29
scsicdb6variant2	29
scsicdb6variant3	30
usbpkt	31
USBVARS	32

2 VS1000/Interfacing File Index

2.1 VS1000/Interfacing File List

Here is a list of all files with brief descriptions:

assert.h	34
audio.h	35
c-nand.s	41
c-restart.s	41
c-spi.s	41
codec.h	41
codecmminiwav.h	46
codecvorbis.h	48
ctype.h	49
dev1000.h	51
errno.h	68
fat.h	73
float.h	74
mapper.h	77

<code>mapperflash.h</code>	79
<code>mappertiny.h</code>	82
<code>math.h</code>	84
<code>minifat.h</code>	88
<code>physical.h</code>	94
<code>player.h</code>	95
<code>romfont.h</code>	102
<code>scsi.h</code> (Common SCSI definitions)	104
<code>stdarg.h</code>	111
<code>stddef.h</code>	112
<code>stdio.h</code>	113
<code>stdlib.h</code>	118
<code>string.h</code>	121
<code>usb.h</code>	126
<code>usblowlib.h</code>	135
<code>vectors.h</code>	149
<code>vs1000.h</code>	150
<code>vsasm.h</code>	179
<code>vsNand.h</code>	179
<code>vstypes.h</code>	183

3 VS1000/Interfacing Page Index

3.1 VS1000/Interfacing Related Pages

Here is a list of all related documentation pages:

Bug List	190
-----------------	------------

4 VS1000/Interfacing Data Structure Documentation

4.1 `__sfpos` Struct Reference

```
#include <stdio.h>
```

Data Fields

- `short _pos [4]`

4.1.1 Detailed Description

Definition at line 17 of file `stdio.h`.

4.1.2 Field Documentation

4.1.2.1 `short __sfpos::_pos[4]`

Definition at line 18 of file `stdio.h`.

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

- `stdio.h`

4.2 `AUDIOPTR` Struct Reference

```
#include <audio.h>
```

Data Fields

- `__ys_int16 * wr`
- `__ys_int16 * rd`
- `u_int16 forwardModulo`
- `s_int16 leftVol`
- `s_int16 rightVol`
- `s_int16 underflow`

4.2.1 Detailed Description

Definition at line 61 of file `audio.h`.

4.2.2 Field Documentation

4.2.2.1 `u_int16 AUDIOPTR::forwardModulo`

Definition at line 64 of file audio.h.

4.2.2.2 `s_int16 AUDIOPTR::leftVol`

Definition at line 65 of file audio.h.

4.2.2.3 `__y s_int16* AUDIOPTR::rd`

Definition at line 63 of file audio.h.

4.2.2.4 `s_int16 AUDIOPTR::rightVol`

Definition at line 66 of file audio.h.

4.2.2.5 `s_int16 AUDIOPTR::underflow`

Definition at line 67 of file audio.h.

4.2.2.6 `__y s_int16* AUDIOPTR::wr`

Definition at line 62 of file audio.h.

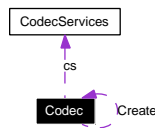
The documentation for this struct was generated from the following file:

- `audio.h`

4.3 Codec Struct Reference

```
#include <codec.h>
```

Collaboration diagram for Codec:



Data Fields

- `u_int16 version`
- `Codec>(* Create)(void)`
- `enum CodecError(* Decode)(struct Codec *cod, struct CodecServices *cs, const char **errorString)`
- `void(* Delete)(struct Codec *cod)`
- `CodecServices * cs`

4.3.1 Detailed Description

Standard Codec wrap-up structure

Definition at line 154 of file codec.h.

4.3.2 Field Documentation

4.3.2.1 `struct Codec>(* Codec::Create)(void)`

Create and allocate space for codec.

4.3.2.2 `struct CodecServices* Codec::cs`

A pointer that the codec may or may not fill or use.

Definition at line 168 of file codec.h.

4.3.2.3 `enum CodecError(* Codec::Decode)(struct Codec *cod, struct CodecServices *cs, const char **errorString)`

Decode file. Upon success or a negative number, Codec has succeeded. With a positive number, there has been an error. Upon return, an error string is also returned.

4.3.2.4 `void(* Codec::Delete)(struct Codec *cod)`

Free all resources allocated for codec.

4.3.2.5 `u_int16 Codec::version`

Version number. 8 MSBs contain version number, 8 LSBs size of the structure in words.

Definition at line 157 of file codec.h.

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

- `codec.h`

4.4 CodecServices Struct Reference

```
#include <codec.h>
```

Data Fields

- `u_int16 version`
- `u_int16(* Read)(struct CodecServices *cs, u_int16 *ptr, u_int16 firstOdd, u_int16 bytes)`
- `u_int32(* Skip)(struct CodecServices *cs, u_int32 bytes)`

- **s_int16(* Seek)**(struct CodecServices *cs, s_int32 offset, s_int16 whence)
- **s_int32(* Tell)**(struct CodecServices *cs)
- **s_int16(* Output)**(struct CodecServices *cs, s_int16 *data, s_int16 n)
- **void(* Comment)**(struct CodecServices *cs, u_int16 c)
- **void(* Spectrum)**(struct CodecServices *cs, s_int16 __y *data, s_int16 n, s_int16 ch)
- **u_int32 fileSize**
- **u_int32 fileLeft**
- **u_int16 goTo**
- **s_int16 cancel**
- **s_int32 playTimeSeconds**
- **s_int32 playTimeSamples**
- **u_int32 playTimeTotal**
- **u_int32 sampleRate**
- **u_int16 channels**
- **enum ChannelMatrix matrix** [MAX_SOURCE_CHANNELS]
- **u_int32 avgBitRate**
- **u_int32 currBitRate**
- **u_int32 peakBitRate**
- **s_int16 gain**
- **u_int16 fastForward**

4.4.1 Detailed Description

Codec(p. 5) Services structure. The caller must provide values for elements *Read*, *Seek*, *Tell* and *fileSize*. The caller may also set *move* and *cancel* for control operations. Note that the prototypes for *Read*, *Seek* and *Tell* are on purpose the same as they are for **stdio.h**(p. 113) functions **fread**()(p. 117), **fseek**()(p. 117) and **ftell**()(p. 117).

Definition at line 53 of file codec.h.

4.4.2 Field Documentation

4.4.2.1 u_int32 CodecServices::avgBitRate

Average bitrate. Updated by the codec. May change during playback.

Definition at line 123 of file codec.h.

4.4.2.2 s_int16 CodecServices::cancel

Request codec to cancel playing current file. To request cancellation, set this to a positive value. When the codec has finished cancelling, it will clear this value and return ceCancelled.

Definition at line 105 of file codec.h.

4.4.2.3 `u_int16 CodecServices::channels`

Number of channels. Updated by the codec.

Definition at line 119 of file `codec.h`.

4.4.2.4 `void(* CodecServices::Comment)(struct CodecServices *cs, u_int16 c)`

Offers comments fields one character at the time. Special code `0x4000U` is reserved for end-of-line. `0xC000U` means end of comments.

4.4.2.5 `u_int32 CodecServices::currBitRate`

Current bitrate. Updated by the codec. May change during playback.

Definition at line 125 of file `codec.h`.

4.4.2.6 `u_int16 CodecServices::fastForward`

Request codec to fast forward current file. If set, playback is requested at fast-Forward times normal playback speed. To stop fast forwarding, set value to 1 or 0.

Definition at line 134 of file `codec.h`.

4.4.2.7 `u_int32 CodecServices::fileLeft`

Bytes left in a file. If set to `0xFFFFFFFFU`, then the file is a stream and never ends.

Definition at line 87 of file `codec.h`.

4.4.2.8 `u_int32 CodecServices::fileSize`

File size in bytes. If set to `0xFFFFFFFFU`, then the file is a stream and the file size is not known.

Definition at line 84 of file `codec.h`.

4.4.2.9 `s_int16 CodecServices::gain`

Volume gain recommended by codec in 1/2 dB steps. Updated by the codec.

Definition at line 130 of file `codec.h`.

4.4.2.10 `u_int16 CodecServices::goTo`

Point to move to in an audio file in seconds. When set to anything other than `0xFFFFU`, the codec has the responsibility to jump to that point. The codec has the freedom of deciding for itself the actual landing point in the file. When the codec has reached its destination it will clear this variable, and the actual position in the file can be read from *playTimeSeconds*. If the codec cannot

jump to a given point (e.g. the file is a stream and the jump point would require jumping backwards), `goTo` is silently cleared to `0xFFFFFU`.

Definition at line 99 of file `codec.h`.

4.4.2.11 `enum ChannelMatrix CodecServices::matrix[MAX_SOURCE_CHANNELS]`

Channel matrix. Updated by the codec.

Definition at line 121 of file `codec.h`.

4.4.2.12 `s_int16(* CodecServices::Output)(struct CodecServices *cs, s_int16 *data, s_int16 n)`

Output to audio file. *data* is a pointer to the data. There are *n chan* channel samples (Example: *n* = 32, *cs->chan* = 2, there are a total of 64 samples in *data*).

4.4.2.13 `u_int32 CodecServices::peakBitRate`

Peak bitrate of the file. Updated by the codec. May change during playback.

Definition at line 128 of file `codec.h`.

4.4.2.14 `s_int32 CodecServices::playTimeSamples`

Samples played since last full second. Updated by the codec. If set to -1, the codec doesn't know where it is in the file.

Definition at line 111 of file `codec.h`.

4.4.2.15 `s_int32 CodecServices::playTimeSeconds`

Playback time from beginning of file in seconds. Updated by the codec. If set to -1, the codec doesn't know where it is in the file.

Definition at line 108 of file `codec.h`.

4.4.2.16 `u_int32 CodecServices::playTimeTotal`

Total playback time in seconds. Updated by the codec. May be a changing estimate if an exact figure isn't available. `0xFFFFFFFFFU` if there is no information available.

Definition at line 115 of file `codec.h`.

4.4.2.17 `u_int16(* CodecServices::Read)(struct CodecServices *cs, u_int16 *ptr, u_int16 firstOdd, u_int16 bytes)`

Read data from a file. If *firstOdd* is set, only the LSB of the first word is filled. If the last word is not completely filled (either *firstOdd* set or *bytes* is odd, but not both), only the MSB is changed.

4.4.2.18 `u_int32 CodecServices::sampleRate`

Sample rate. Updated by the codec. 0 if unknown.

Definition at line 117 of file codec.h.

4.4.2.19 `s_int16(* CodecServices::Seek)(struct CodecServices *cs, s_int32 offset, s_int16 whence)`

Seek in a file. *offset* and *whence* are equivalent with their `fseek()` (p. 117) counterparts. If *Seek* is NULL, the input is a stream and cannot be seeked.

4.4.2.20 `u_int32(* CodecServices::Skip)(struct CodecServices *cs, u_int32 bytes)`

Skip data in a file. This should also be supported for streams.

4.4.2.21 `void(* CodecServices::Spectrum)(struct CodecServices *cs, s_int16 __y *data, s_int16 n, s_int16 ch)`

Spectrum analyzer hook.

4.4.2.22 `s_int32(* CodecServices::Tell)(struct CodecServices *cs)`

Tell location in a file (in bytes).

4.4.2.23 `u_int16 CodecServices::version`

Version number. 8 MSBs contain version number, 8 LSBs size of the structure in words.

Definition at line 56 of file codec.h.

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

- `codec.h`

4.5 EARSPEAKER Struct Reference

```
#include <audio.h>
```

Data Fields

- `u_int16 Freq`
- `u_int16 Disable`

- `u_int16` Setting
- `s_int16` Old
- `u_int16` longFrames

4.5.1 Detailed Description

Definition at line 92 of file audio.h.

4.5.2 Field Documentation

4.5.2.1 `u_int16` EARSPEAKER::Disable

Definition at line 94 of file audio.h.

4.5.2.2 `u_int16` EARSPEAKER::Freq

Definition at line 93 of file audio.h.

4.5.2.3 `u_int16` EARSPEAKER::longFrames

Definition at line 97 of file audio.h.

4.5.2.4 `s_int16` EARSPEAKER::Old

Definition at line 96 of file audio.h.

4.5.2.5 `u_int16` EARSPEAKER::Setting

Definition at line 95 of file audio.h.

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

- `audio.h`

4.6 FATINFO Struct Reference

```
#include <fat.h>
```

Data Fields

- `u_int16` IS_FAT_32
- `u_int32` fatStart
- `u_int32` rootStart
- `u_int32` dataStart
- `u_int32` currentSector
- `u_int32` fileSize
- `u_int16` fatSectorsPerCluster

- `u_int16 BPB_RootEntCnt`
- `u_int16 FilSysType`
- `s_int32 totSize`
- `u_int16 fileName [6]`
- `u_int16 gFileNum [2]`
- `s_int32 filePos`
- `s_int32 parentDir`
- `const u_int32 * supportedSuffixes`
- `u_int16 longFileName [FAT_LFN_SIZE/2]`

4.6.1 Detailed Description

Definition at line 44 of file fat.h.

4.6.2 Field Documentation

4.6.2.1 `u_int16 FATINFO::BPB_RootEntCnt`

Definition at line 52 of file fat.h.

4.6.2.2 `u_int32 FATINFO::currentSector`

Sector in minifatBuffer

Definition at line 49 of file fat.h.

4.6.2.3 `u_int32 FATINFO::dataStart`

File Allocation Table start

Definition at line 48 of file fat.h.

4.6.2.4 `u_int16 FATINFO::fatSectorsPerCluster`

Definition at line 51 of file fat.h.

4.6.2.5 `u_int32 FATINFO::fatStart`

Fat start sector number

Definition at line 46 of file fat.h.

4.6.2.6 `u_int16 FATINFO::fileName[6]`

Current file name

Definition at line 55 of file fat.h.

4.6.2.7 s_int32 FATINFO::filePos

Current file byte read position

Definition at line 57 of file fat.h.

4.6.2.8 u_int32 FATINFO::fileSize

Current file size

Definition at line 50 of file fat.h.

4.6.2.9 u_int16 FATINFO::FilSysType

"\p21" for FAT12

Definition at line 53 of file fat.h.

4.6.2.10 u_int16 FATINFO::gFileNum[2]

File number counters

Definition at line 56 of file fat.h.

4.6.2.11 u_int16 FATINFO::IS_FAT_32

FAT32 or FAT16/FAT12

Definition at line 45 of file fat.h.

4.6.2.12 u_int16 FATINFO::longFileName[FAT_LFN_SIZE/2]

Long filename, if exists

Definition at line 61 of file fat.h.

4.6.2.13 s_int32 FATINFO::parentDir

Parent directory start sector

Definition at line 58 of file fat.h.

4.6.2.14 u_int32 FATINFO::rootStart

Root director start

Definition at line 47 of file fat.h.

4.6.2.15 const u_int32* FATINFO::supportedSuffixes

NULL or list of supported suffixes.

Definition at line 59 of file fat.h.

4.6.2.16 s_int32 FATINFO::totSize

Total sectors

Definition at line 54 of file fat.h.

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

- fat.h

4.7 FmfMeta Struct Reference

```
#include <mapperflash.h>
```

Data Fields

- u_int16 ecc01
- u_int16 ecc2AndType
- u_int16 reservedAndBadBlock
- u_int16 unused
- u_int32 logicalPageNo
- s_int32 newBranch

4.7.1 Detailed Description

Meta data

Definition at line 30 of file mapperflash.h.

4.7.2 Field Documentation

4.7.2.1 u_int16 FmfMeta::ecc01

Definition at line 31 of file mapperflash.h.

4.7.2.2 u_int16 FmfMeta::ecc2AndType

Definition at line 32 of file mapperflash.h.

4.7.2.3 u_int32 FmfMeta::logicalPageNo

Definition at line 35 of file mapperflash.h.

4.7.2.4 s_int32 FmfMeta::newBranch

Definition at line 36 of file mapperflash.h.

4.7.2.5 u_int16 FmfMeta::reservedAndBadBlock

Definition at line 33 of file mapperflash.h.

4.7.2.6 u_int16 FmfMeta::unused

Definition at line 34 of file mapperflash.h.

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

- **mapperflash.h**

4.8 FRAGMENT Struct Reference

```
#include <fat.h>
```

Data Fields

- **u_int32 start**
- **u_int16 size**

4.8.1 Detailed Description

Definition at line 67 of file fat.h.

4.8.2 Field Documentation

4.8.2.1 u_int16 FRAGMENT::size

fragment size in sectors

Definition at line 69 of file fat.h.

4.8.2.2 u_int32 FRAGMENT::start

high bit set if last fragment

Definition at line 68 of file fat.h.

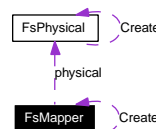
The documentation for this struct was generated from the following file:

- **fat.h**

4.9 FsMapper Struct Reference

```
#include <mapper.h>
```

Collaboration diagram for FsMapper:



Data Fields

- **u_int16 version**
- **u_int16 blockSize**
- **u_int32 blocks**
- **u_int16 cacheBlocks**
- **FsMapper *(* Create)**(struct **FsPhysical** *physical, **u_int16** cacheSize)
- **s_int16(* Delete)**(struct **FsMapper** *map)
- **s_int16(* Read)**(struct **FsMapper** *map, **u_int32** firstBlock, **u_int16** blocks, **u_int16** *data)
- **s_int16(* Write)**(struct **FsMapper** *map, **u_int32** firstBlock, **u_int16** blocks, **u_int16** *data)
- **s_int16(* Free)**(struct **FsMapper** *map, **u_int32** firstBlock, **u_int32** blocks)
- **s_int16(* Flush)**(struct **FsMapper** *map, **u_int16** hard)
- **FsPhysical** * physical

4.9.1 Detailed Description

File system Mapper layer structure. Each Mapper should begin its own internal structure with this common structure.

Definition at line 38 of file mapper.h.

4.9.2 Field Documentation

4.9.2.1 u_int32 FsMapper::blocks

How many usable blocks in the whole system

Definition at line 45 of file mapper.h.

4.9.2.2 u_int16 FsMapper::blockSize

How many 16-bit words in a block

Definition at line 43 of file mapper.h.

4.9.2.3 u_int16 FsMapper::cacheBlocks

How many blocks can be cached by the mapper

Definition at line 47 of file mapper.h.

4.9.2.4 struct FsMapper *(* FsMapper::Create)(struct FsPhysical *physical, u_int16 cacheSize)

Create a mapper.

4.9.2.5 s_int16(* FsMapper::Delete)(struct FsMapper *map)

Delete a mapper

4.9.2.6 s_int16(* FsMapper::Flush)(struct FsMapper *map, u_int16 hard)

Flush all data. if *hard* is non-zero, all potential journals are also flushed.

4.9.2.7 s_int16(* FsMapper::Free)(struct FsMapper *map, u_int32 firstBlock, u_int32 blocks)

Free blocks (implementation must be able to go fastly through large free areas.

4.9.2.8 struct FsPhysical* FsMapper::physical

Pointer to this Mapper's Physical layer.

Definition at line 65 of file mapper.h.

4.9.2.9 s_int16(* FsMapper::Read)(struct FsMapper *map, u_int32 firstBlock, u_int16 blocks, u_int16 *data)

Read blocks

4.9.2.10 u_int16 FsMapper::version

Version number. 8 MSBs contain version number, 8 LSBs size of the structure in words.

Definition at line 41 of file mapper.h.

4.9.2.11 s_int16(* FsMapper::Write)(struct FsMapper *map, u_int32 firstBlock, u_int16 blocks, u_int16 *data)

Write blocks

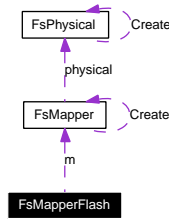
The documentation for this struct was generated from the following file:

- **mapper.h**

4.10 FsMapperFlash Struct Reference

```
#include <mapperflash.h>
```

Collaboration diagram for FsMapperFlash:



Data Fields

- **FsMapper m**
- **u_int32 root**
- **s_int16 blocksPerErase**
- **s_int32 lastUsed**
- **FmfCache * cache**
- **s_int32 physPages**
- **s_int32 emptyBlock [FS_MAP_NON_FULL]**
- **s_int16 nonFullLimit**
- **s_int32 skipped**
- **s_int32 freed**

4.10.1 Detailed Description

A Flash Mapper specific structure that contains required extensions to the basic Mapper structure.

Definition at line 46 of file mapperflash.h.

4.10.2 Field Documentation

4.10.2.1 s_int16 FsMapperFlash::blocksPerErase

Logical blocks in erase unit.

Definition at line 52 of file mapperflash.h.

4.10.2.2 struct FmfCache* FsMapperFlash::cache

Internal cache.

Definition at line 56 of file mapperflash.h.

4.10.2.3 s_int32 FsMapperFlash::emptyBlock[FS_MAP_NON_FULL]

Blocks that are not (almost) completely full with FMF_TYPE_DATA

Definition at line 60 of file mapperflash.h.

4.10.2.4 s_int32 FsMapperFlash::freed

How many blocks have been cleaned

Definition at line 66 of file mapperflash.h.

4.10.2.5 s_int32 FsMapperFlash::lastUsed

Last new physical address.

Definition at line 54 of file mapperflash.h.

4.10.2.6 struct FsMapper FsMapperFlash::m

Public structure that is common to all mappers.

Definition at line 48 of file mapperflash.h.

4.10.2.7 s_int16 FsMapperFlash::nonFullLimit

How many pages in a block must be free for the block to be non-full

Definition at line 62 of file mapperflash.h.

4.10.2.8 s_int32 FsMapperFlash::physPages

Total of physical pages.

Definition at line 58 of file mapperflash.h.

4.10.2.9 u_int32 FsMapperFlash::root

Root node physical address.

Definition at line 50 of file mapperflash.h.

4.10.2.10 s_int32 FsMapperFlash::skipped

How many blocks have been skipped while cleaning.

Definition at line 64 of file mapperflash.h.

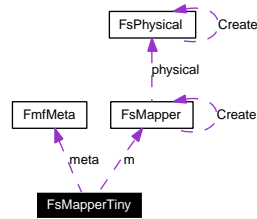
The documentation for this struct was generated from the following file:

- **mapperflash.h**

4.11 FsMapperTiny Struct Reference

```
#include <mappertiny.h>
```

Collaboration diagram for FsMapperTiny:



Data Fields

- **FsMapper m**
- **s_int32 root**
- **s_int16 blocksPerErase**
- **s_int32 firstBlock**
- **s_int32 lastBlock**
- **s_int32 logToPhys**
- **FmfMeta meta**

4.11.1 Detailed Description

A Tiny Flash Mapper specific structure that contains required extensions to the basic Tiny Flash Mapper structure. The Tiny Flash Mapper is a read-only system that has been optimized to access consecutive blocks using only a few words of memory. For such a file performance is very good, but if several file are open, performance may be slow.

Definition at line 36 of file mappertiny.h.

4.11.2 Field Documentation

4.11.2.1 s_int16 FsMapperTiny::blocksPerErase

Logical blocks in erase unit.

Definition at line 42 of file mappertiny.h.

4.11.2.2 s_int32 FsMapperTiny::firstBlock

First logical in access list.

Definition at line 44 of file mappertiny.h.

4.11.2.3 s_int32 FsMapperTiny::lastBlock

Last logical in access list.

Definition at line 46 of file mappertiny.h.

4.11.2.4 s_int32 FsMapperTiny::logToPhys

Logical block to physical page conversion number.

Definition at line 48 of file mappertiny.h.

4.11.2.5 struct FsMapper FsMapperTiny::m

Public structure that is common to all mappers.

Definition at line 38 of file mappertiny.h.

4.11.2.6 struct FmfMeta FsMapperTiny::meta

Meta info (buffer space)

Definition at line 50 of file mappertiny.h.

4.11.2.7 s_int32 FsMapperTiny::root

Root node physical address.

Definition at line 40 of file mappertiny.h.

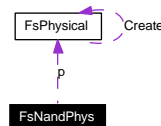
The documentation for this struct was generated from the following file:

- **mappertiny.h**

4.12 FsNandPhys Struct Reference

```
#include <vsNand.h>
```

Collaboration diagram for FsNandPhys:

**Data Fields**

- **FsPhysical p**
- **u_int16 nandType**
- **u_int16 waitns**

4.12.1 Detailed Description

Definition at line 33 of file vsNand.h.

4.12.2 Field Documentation

4.12.2.1 `u_int16 FsNandPhys::nandType`

NAND type, small/large-page, number of address bytes

Definition at line 37 of file vsNand.h.

4.12.2.2 `struct FsPhysical FsNandPhys::p`

Physical basic structure

Definition at line 35 of file vsNand.h.

4.12.2.3 `u_int16 FsNandPhys::waitns`

read/write time in ns

Definition at line 38 of file vsNand.h.

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

- `vsNand.h`

4.13 FsPhysical Struct Reference

```
#include <physical.h>
```

Collaboration diagram for FsPhysical:



Data Fields

- `u_int16 version`
- `u_int16 pageSize`
- `u_int16 eraseBlockSize`
- `u_int16 eraseBlocks`
- `FsPhysical>(* Create)(u_int16 param)`
- `s_int16(* Delete)(struct FsPhysical *p)`
- `s_int16(* Read)(struct FsPhysical *p, s_int32 firstPage, u_int16 pages, u_int16 *data, u_int16 *meta)`
- `s_int16(* Write)(struct FsPhysical *p, s_int32 firstPage, u_int16 pages, u_int16 *data, u_int16 *meta)`
- `s_int16(* Erase)(struct FsPhysical *p, s_int32 page)`
- `s_int16(* FreeBus)(struct FsPhysical *p)`
- `s_int16(* Reinitialize)(struct FsPhysical *p)`

4.13.1 Detailed Description

File system Physical layer basic structure. Each Physical layer internal structure should begin with this.

Definition at line 28 of file physical.h.

4.13.2 Field Documentation

4.13.2.1 `struct FsPhysical>(* FsPhysical::Create)(u_int16 param)`

Creates a physical layer. `param` is a device-specific parameter, usually 0.

4.13.2.2 `s_int16(* FsPhysical::Delete)(struct FsPhysical *p)`

Delete a physical layer

4.13.2.3 `s_int16(* FsPhysical::Erase)(struct FsPhysical *p, s_int32 page)`

Erase block. *firstPage* is the page number of the first page in the block.

4.13.2.4 `u_int16 FsPhysical::eraseBlocks`

The size of the memory unit in erasable blocks

Definition at line 37 of file physical.h.

4.13.2.5 `u_int16 FsPhysical::eraseBlockSize`

In pages

Definition at line 35 of file physical.h.

4.13.2.6 `s_int16(* FsPhysical::FreeBus)(struct FsPhysical *p)`

Frees the bus for other devices

4.13.2.7 `u_int16 FsPhysical::pageSize`

In 16-bit words

Definition at line 33 of file physical.h.

4.13.2.8 `s_int16(* FsPhysical::Read)(struct FsPhysical *p, s_int32 firstPage, u_int16 pages, u_int16 *data, u_int16 *meta)`

Read pages. `meta` is physical-specific data and not necessarily used. If either `data` or `meta` is NULL, that information is not returned. Setting both pointers to NULL is an error condition.

4.13.2.9 s_int16(* FsPhysical::Reinitialize)(struct FsPhysical *p)

Re-initializes bus after a fatal error (eg memory card removal)

4.13.2.10 u_int16 FsPhysical::version

Version number. 8 MSBs contain version number, 8 LSBs size of the structure in words.

Definition at line 31 of file physical.h.

4.13.2.11 s_int16(* FsPhysical::Write)(struct FsPhysical *p, s_int32 firstPage, u_int16 pages, u_int16 *data, u_int16 *meta)

Write pages. meta is physical-specific data and not necessarily used. If either data or meta is NULL, that information is not written. Setting both pointers to NULL is an error condition.

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

- **physical.h**

4.14 KeyMapping Struct Reference

```
#include <player.h>
```

Data Fields

- **u_int16 key**
- **enum keyEvent event**

4.14.1 Detailed Description

Definition at line 53 of file player.h.

4.14.2 Field Documentation**4.14.2.1 enum keyEvent KeyMapping::event**

event to trigger when key detected

Definition at line 55 of file player.h.

4.14.2.2 u_int16 KeyMapping::key

bitmask for key combination

Definition at line 54 of file player.h.

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

- `player.h`

4.15 Player Struct Reference

```
#include <player.h>
```

Data Fields

- `s_int16 totalFiles`
- `s_int16 currentFile`
- `s_int16 nextFile`
- `s_int16 nextStep`
- `s_int16 pauseOn`
- `s_int16 randomOn`
- `s_int16 volume`
- `s_int16 volumeOffset`
- `u_int16 offDelay`
- `u_int16 ffCount`
- `u_int16 maxClock`

4.15.1 Detailed Description

Structure used by the default firmwares play loop.

Definition at line 14 of file `player.h`.

4.15.2 Field Documentation

4.15.2.1 `s_int16 Player::currentFile`

current playing file

Definition at line 16 of file `player.h`.

4.15.2.2 `u_int16 Player::ffCount`

fast forward/rewind counter

Definition at line 24 of file `player.h`.

4.15.2.3 `u_int16 Player::maxClock`

max clock used by player: 7=3.5x, 6=3.0x etc. Default = 7, depends on core voltage.

Definition at line 25 of file `player.h`.

4.15.2.4 s_int16 Player::nextFile

next file to play, changed by key events like ke_next

Definition at line 17 of file player.h.

4.15.2.5 s_int16 Player::nextStep

skip direction, 1 = next, -1 = previous

Definition at line 18 of file player.h.

4.15.2.6 u_int16 Player::offDelay

pause timeout in 5 sec increments -> power off

Definition at line 23 of file player.h.

4.15.2.7 s_int16 Player::pauseOn

non-zero for pause mode

Definition at line 19 of file player.h.

4.15.2.8 s_int16 Player::randomOn

non-zero for random play mode

Definition at line 20 of file player.h.

4.15.2.9 s_int16 Player::totalFiles

total number of matching files

Definition at line 15 of file player.h.

4.15.2.10 s_int16 Player::volume

volume -24 to 180 in 0.5dB steps

Definition at line 21 of file player.h.

4.15.2.11 s_int16 Player::volumeOffset

volume offset, set by replayGain, default -12

Definition at line 22 of file player.h.

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

- **player.h**

4.16 scsicdb10variant1 Struct Reference

```
#include <scsi.h>
```

Data Fields

- `u_int16 length__opcode`
- `u_int16 res__lbab3`
- `u_int16 lbab2__lbab1`
- `u_int16 lbab0__res`
- `u_int16 wLength`
- `u_int16 control__null`

4.16.1 Detailed Description

Definition at line 43 of file scsi.h.

4.16.2 Field Documentation

4.16.2.1 `u_int16 scsicdb10variant1::control__null`

Definition at line 49 of file scsi.h.

4.16.2.2 `u_int16 scsicdb10variant1::lbab0__res`

Definition at line 47 of file scsi.h.

4.16.2.3 `u_int16 scsicdb10variant1::lbab2__lbab1`

Definition at line 46 of file scsi.h.

4.16.2.4 `u_int16 scsicdb10variant1::length__opcode`

Definition at line 44 of file scsi.h.

4.16.2.5 `u_int16 scsicdb10variant1::res__lbab3`

Definition at line 45 of file scsi.h.

4.16.2.6 `u_int16 scsicdb10variant1::wLength`

Definition at line 48 of file scsi.h.

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

- `scsi.h`

4.17 scsicdb10variant2 Struct Reference

```
#include <scsi.h>
```

Data Fields

- `u_int16 length__opcode`
- `u_int16 flags__lbab3`
- `u_int16 lbab2__lbab1`
- `u_int16 lbab0__res`
- `u_int16 wLength`
- `u_int16 control__null`

4.17.1 Detailed Description

Definition at line 52 of file scsi.h.

4.17.2 Field Documentation

4.17.2.1 `u_int16 scsicdb10variant2::control__null`

Definition at line 58 of file scsi.h.

4.17.2.2 `u_int16 scsicdb10variant2::flags__lbab3`

Definition at line 54 of file scsi.h.

4.17.2.3 `u_int16 scsicdb10variant2::lbab0__res`

Definition at line 56 of file scsi.h.

4.17.2.4 `u_int16 scsicdb10variant2::lbab2__lbab1`

Definition at line 55 of file scsi.h.

4.17.2.5 `u_int16 scsicdb10variant2::length__opcode`

Definition at line 53 of file scsi.h.

4.17.2.6 `u_int16 scsicdb10variant2::wLength`

Definition at line 57 of file scsi.h.

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

- `scsi.h`

4.18 scsicdb6variant1 Struct Reference

```
#include <scsi.h>
```

Data Fields

- `u_int16 length__opcode`
- `u_int16 flags__pageCode`
- `u_int16 allocationLength`
- `u_int16 control__null`

4.18.1 Detailed Description

Definition at line 22 of file scsi.h.

4.18.2 Field Documentation

4.18.2.1 `u_int16 scsicdb6variant1::allocationLength`

Definition at line 25 of file scsi.h.

4.18.2.2 `u_int16 scsicdb6variant1::control__null`

Definition at line 26 of file scsi.h.

4.18.2.3 `u_int16 scsicdb6variant1::flags__pageCode`

Definition at line 24 of file scsi.h.

4.18.2.4 `u_int16 scsicdb6variant1::length__opcode`

Definition at line 23 of file scsi.h.

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

- `scsi.h`

4.19 scsicdb6variant2 Struct Reference

```
#include <scsi.h>
```

Data Fields

- `u_int16 length__opcode`
- `u_int16 flags__pageCode`
- `u_int16 res__allocationLength`
- `u_int16 control__null`

4.19.1 Detailed Description

Definition at line 29 of file scsi.h.

4.19.2 Field Documentation

4.19.2.1 `u_int16 scsicdb6variant2::control__null`

Definition at line 33 of file scsi.h.

4.19.2.2 `u_int16 scsicdb6variant2::flags__pageCode`

Definition at line 31 of file scsi.h.

4.19.2.3 `u_int16 scsicdb6variant2::length__opcode`

Definition at line 30 of file scsi.h.

4.19.2.4 `u_int16 scsicdb6variant2::res__allocationLength`

Definition at line 32 of file scsi.h.

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

- `scsi.h`

4.20 scsicdb6variant3 Struct Reference

```
#include <scsi.h>
```

Data Fields

- `u_int16 length__opcode`
- `u_int16 flags__res`
- `u_int16 res__allocationLength`
- `u_int16 control__null`

4.20.1 Detailed Description

Definition at line 36 of file scsi.h.

4.20.2 Field Documentation

4.20.2.1 `u_int16 scsicdb6variant3::control__null`

Definition at line 40 of file scsi.h.

4.20.2.2 `u_int16 scsicdb6variant3::flags__res`

Definition at line 38 of file scsi.h.

4.20.2.3 `u_int16 scsicdb6variant3::length__opcode`

Definition at line 37 of file scsi.h.

4.20.2.4 `u_int16 scsicdb6variant3::res__allocationLength`

Definition at line 39 of file scsi.h.

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

- `scsi.h`

4.21 `usbpkt` Struct Reference

```
#include <usbblowlib.h>
```

Data Fields

- `u_int16 length`
- `u_int16 payload [(ENDPOINT_SIZE_1+1)>>1]`

4.21.1 Detailed Description

Holding space for one received USB packet.

Definition at line 136 of file usbblowlib.h.

4.21.2 Field Documentation**4.21.2.1** `u_int16 usbpkt::length`

packet length in bytes

Definition at line 137 of file usbblowlib.h.

4.21.2.2 `u_int16 usbpkt::payload[(ENDPOINT_SIZE_1+1)>>1]`

packet data

Definition at line 138 of file usbblowlib.h.

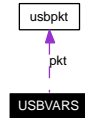
The documentation for this struct was generated from the following file:

- `usbblowlib.h`

4.22 USBVARS Struct Reference

```
#include <usbblowlib.h>
```

Collaboration diagram for USBVARS:



Data Fields

- `const u_int16 * descriptorTable` [6]
- `u_int16 configurationDescriptorSize`
- `USBPacket pkt`
- `u_int32 totbytes`
- `u_int16 ExtraZeroLengthPacketNeeded` [4]
- `const u_int16 * XmitBuf` [4]
- `u_int16 XmitLength` [4]
- `u_int16 EPReady` [4]
- `u_int16 lastSofTimeout`
- `u_int16 configuration`
- `u_int16 interfaces`
- `u_int16 lastSofFill`
- `u_int32 lastSofTime`

4.22.1 Detailed Description

Definition at line 149 of file `usbblowlib.h`.

4.22.2 Field Documentation

4.22.2.1 `u_int16 USBVARS::configuration`

Stores current configuration. Only used for `USB_REQUEST_GET_CONFIGURATION`.

Definition at line 189 of file `usbblowlib.h`.

4.22.2.2 `u_int16 USBVARS::configurationDescriptorSize`

Length of Configuration Descriptor. (needed because configuration descriptor is actually a collection of many descriptors so the first octet does not specify length of the entire descriptor)

Definition at line 168 of file `usbblowlib.h`.

4.22.2.3 const u_int16* USBVARS::descriptorTable[6]

Descriptor Pointer Table Members are:

- *stringDescriptor0
- *stringDescriptor1
- *stringDescriptor2,
- *stringDescriptor3,
- *deviceDescriptor,
- *configurationDescriptor

For others than configurationDescriptor, descriptor size is first octet of descriptor.

Definition at line 162 of file usblowlib.h.

4.22.2.4 u_int16 USBVARS::EPReady[4]

Is endpoint ready to transmit new block?

Definition at line 185 of file usblowlib.h.

4.22.2.5 u_int16 USBVARS::ExtraZeroLengthPacketNeeded[4]

Is an extra zero-length packet needed after transmission?

Definition at line 176 of file usblowlib.h.

4.22.2.6 u_int16 USBVARS::interfaces

Stores current and alternate setting. Only used for USB_REQUEST_GET_INTERFACE.

Definition at line 190 of file usblowlib.h.

4.22.2.7 u_int16 USBVARS::lastSofFill

Audio buf fullness at last SOF

Definition at line 191 of file usblowlib.h.

4.22.2.8 u_int32 USBVARS::lastSofTime

When last SOF was received

Definition at line 192 of file usblowlib.h.

4.22.2.9 `u_int16 USBVARS::lastSofTimeout`

Used for suspend detection, although a bit inaccurate.

Definition at line 187 of file `usbblowlib.h`.

4.22.2.10 `USBPacket USBVARS::pkt`

Holding space for one received USB packet.

Definition at line 171 of file `usbblowlib.h`.

4.22.2.11 `u_int32 USBVARS::totbytes`

Total transferred bytes.

Definition at line 173 of file `usbblowlib.h`.

4.22.2.12 `const u_int16* USBVARS::XmitBuf[4]`

Current USB Endpoint transmit buffer pointers

Definition at line 179 of file `usbblowlib.h`.

4.22.2.13 `u_int16 USBVARS::XmitLength[4]`

Current USB Endpoints' bytes left to transmit

Definition at line 182 of file `usbblowlib.h`.

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

- `usbblowlib.h`

5 VS1000/Interfacing File Documentation

5.1 `assert.h` File Reference

Defines

- `#define assert(exp) ((exp) ? 0 : __assert(#exp, __FILE__, __LINE__))`

5.1.1 Detailed Description

Standard C header file.

Definition in file `assert.h`.

5.1.2 Define Documentation

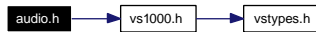
5.1.2.1 `#define assert(exp) ((exp) ? 0 : __assert(#exp, __FILE__, __LINE__))`

Definition at line 9 of file assert.h.

5.2 audio.h File Reference

```
#include "vs1000.h"
```

Include dependency graph for audio.h:



Data Structures

- struct **AUDIOPTR**
- struct **EARSPEAKER**

Defines

- `#define WITH_EARSPEAKER`
- `#define DIRECT_VORBIS_BLOCKSIZE`
- `#define USE_TIMER`
- `#define TIMER_TICKS 100`
- `#define DEFAULT_AUDIO_BUFFER_SAMPLES 2048`
- `#define DAC_DEFAULT_SAMPLERATE 8000`
- `#define DAC_DRIVER_ON_DELAY (DAC_DEFAULT_SAMPLERATE/10)`
- `#define APPL_RESET 0`
- `#define APPL_AUDIO 1`
- `#define APPL_BITSTREAM 10`

Functions

- `u_int32 ReadTimeCount (void)`
- `void InitAudio (void)`
- `auto void StereoCopy (register __i2 s_int16 *s, register __a0 u_int16 n)`
- `s_int16 AudioBufFill (void)`
- `s_int16 AudioBufFree (void)`
- `auto void SetRate (register __c1 u_int16 rate)`
- `auto void RealSetRate (register __c1 u_int16 rate)`
- `auto void SetVolume (void)`
- `auto void RealSetVolume (void)`

- auto void **AudioOutputSamples** (s_int16 *p, s_int16 samples)
- u_int16 **UartDiv** (void)

Variables

- __y volatile u_int32 **timeCount**
- s_int16(* **applAddr**)(s_int16 register __i0 **d, s_int16 register __a1 mode, s_int16 register __a0 n)
- __y s_int16 **audioBuffer** [2 *DEFAULT_AUDIO_BUFFER_SAMPLES]
- __y struct **AUDIOPTR** **audioPtr**
- u_int16 **earSpeakerReg**
- __y u_int16 **earSpeakerDisable**
- u_int16 **volumeReg**
- u_int16 **bassReg**
- __y u_int16 **extClock4KHz**
- __y u_int16 **clockX**
- u_int32 __y **curFctl**
- __y u_int16 **hwSampleRate**
- __y u_int16 **uiTime**
- __y u_int16 **uiTrigger**
- s_int16 __y **timeToRemovePDown2**
- u_int32 __y **haltTime**
- __y u_int16 **uartByteSpeed**
- __y struct **EARSPEAKER** **earSpeaker**

5.2.1 Detailed Description

Routines related to audio.

Definition in file **audio.h**.

5.2.2 Define Documentation

5.2.2.1 #define APPL_AUDIO 1

Definition at line 30 of file **audio.h**.

5.2.2.2 #define APPL_BITSTREAM 10

Definition at line 31 of file **audio.h**.

5.2.2.3 #define APPL_RESET 0

Definition at line 29 of file **audio.h**.

5.2.2.4 #define DAC_DEFAULT_SAMPLERATE 8000

Definition at line 26 of file audio.h.

5.2.2.5 #define DAC_DRIVER_ON_DELAY (DAC_DEFAULT_SAMPLERATE/10)

Definition at line 27 of file audio.h.

5.2.2.6 #define DEFAULT_AUDIO_BUFFER_SAMPLES 2048

Definition at line 25 of file audio.h.

5.2.2.7 #define DIRECT_VORBIS_BLOCKSIZE

Definition at line 10 of file audio.h.

5.2.2.8 #define TIMER_TICKS 100

Definition at line 14 of file audio.h.

5.2.2.9 #define USE_TIMER

Definition at line 12 of file audio.h.

5.2.2.10 #define WITH_EARSPEAKER

Definition at line 9 of file audio.h.

5.2.3 Function Documentation**5.2.3.1 s_int16 AudioBufFill (void)**

Tells the fill state of audio buffer in stereo samples.

5.2.3.2 s_int16 AudioBufFree (void)

Tells how many stereo samples still fits into the audio buffer without waiting.
Note: the buffer should never be completely filled because the same state means empty.

5.2.3.3 auto void AudioOutputSamples (s_int16 * p, s_int16 samples)

High-level audio output routine. If the samples do not fit into the audio buffer, this routine automatically waits for some room (calls `Sleep()`(p.178)).

Parameters:

p Pointer to interleaved stereo samples.

samples The number of stereo samples.

5.2.3.4 void InitAudio (void)

Initializes audio structures and configures the PLL.

5.2.3.5 u_int32 ReadTimeCount (void)

ReadTimeCount Reads the timecount variable safely, no interrupt disable is needed.

5.2.3.6 auto void RealSetRate (register __c1 u_int16 rate)

Sets new samplerate and/or new PLL setting (according to clockX variable).

Parameters:

rate New samplerate.

5.2.3.7 auto void RealSetVolume (void)

Sets the hardware volume according to volumeReg.

5.2.3.8 auto void SetRate (register __c1 u_int16 rate)

Hook: Sets new samplerate and/or new PLL setting. Default: RealSetRate.

Parameters:

rate New samplerate.

5.2.3.9 auto void SetVolume (void)

Hook: Sets the hardware volume according to volumeReg. Default: RealSetVolume

5.2.3.10 auto void StereoCopy (register __i2 s_int16 * s, register __a0 u_int16 n)

Internal low-level audio output routine that puts samples into the audio buffer. Applies audioPtr.leftVol and audioPtr.rightVol to the data. Setting audioPtr.rightVol to -audioPtr.leftVol after **InitAudio()**(p.38) has been called activates differential output mode, where the phase of right channel is inverted compared to the left channel. This routine does not check the buffer fullness.

Parameters:

s Pointer to interleaved stereo samples

n The number of stereo samples

5.2.3.11 `u_int16 UartDiv (void)`

Calculates UART divider from clockX and uartByteSpeed. When **SetRate()**(p. 38) changes PLL settings, the uart divider is automatically changed.

Returns:

UART divider for the current clockX and uartByteSpeed.

5.2.4 Variable Documentation**5.2.4.1** `s_int16(* applAddr)(s_int16 register __i0 **d, s_int16 register __a1 mode, s_int16 register __a0 n)`

applAddr A hook function to process samples before they are put into the audio buffer.

Parameters:

d A pointer to pointer to interleaved stereo samples

mode If APPL_AUDIO, samples available

n Number of stereo samples

Returns:

For APPL_AUDIO the number of output stereo samples, return n otherwise.

5.2.4.2 `__y s_int16 audioBuffer[2 *DEFAULT_AUDIO_BUFFER_SAMPLES]`

Audio FIFO. The length of the area used for audio can change depending on the state of the earSpeaker setting. Earspeaker can not be active when long vorbis frames are used.

5.2.4.3 `__y struct AUDIOPTR audioPtr`

Audio structure containing the audio FIFO read and write pointers, FIFO size, software volume settings, and the FIFO underflow flag. By negating the other volume field you can get mono differential drive from the DAC for connecting a speaker between LEFT and RIGHT outputs.

5.2.4.4 `u_int16 bassReg`

Bass and treble controls, not used by rom firmware. See VS10xx datasheets for details.

5.2.4.5 `__y u_int16 clockX`

Current clock multiplier in 0.5x steps. Is used to program the PLL at the next **SetRate()**(p. 38).

5.2.4.6 u_int32 __y curFctl

Current DAC adder value.

5.2.4.7 __y struct EARSPEAKER earSpeaker**5.2.4.8 __y u_int16 earSpeakerDisable**

Long vorbis frames disable EarSpeaker automatically.

5.2.4.9 u_int16 earSpeakerReg

Current EarSpeaker setting.

5.2.4.10 __y u_int16 extClock4KHz

Crystal/4000. Normally 3000 (for 12MHz).

5.2.4.11 u_int32 __y haltTime

The number of cycles spent in HALT since last check. Used for automatic clock management.

5.2.4.12 __y u_int16 hwSampleRate

Current samplerate.

5.2.4.13 __y volatile u_int32 timeCount

timeCount counts TIMER_TICKS regardless of clockX.

5.2.4.14 s_int16 __y timeToRemovePDown2

Delay until analog drivers are enabled.

5.2.4.15 __y u_int16 uartByteSpeed

UART speed in bps / 10. UART divider is automatically updated when PLL value is changed.

5.2.4.16 __y u_int16 uiTime

Free-running counter for UI.

5.2.4.17 __y u_int16 uiTrigger

Is non-zero 16 times per second when audio is played.

5.2.4.18 u_int16 volumeReg

Left and right volume 0x00(loudest)..0xff (off).

5.3 c-nand.s File Reference

5.3.1 Detailed Description

Startup module for programs booting from NAND-FLASH or UART. This startup does not touch stack pointer or other registers. This makes it possible to return to the caller. This is useful when the default player main loop can be used but some routines need to be patched. The user can perform the patches and return to the caller.

Definition in file **c-nand.s**.

5.4 c-restart.s File Reference

5.4.1 Detailed Description

Startup module for flashing programs. If the user program returns, execution restarts the firmware.

Definition in file **c-restart.s**.

5.5 c-spi.s File Reference

5.5.1 Detailed Description

Startup module for programs loaded from SPI EEPROM. If the user program returns, execution continues from the point just after the spi boot command.

Definition in file **c-spi.s**.

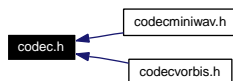
5.6 codec.h File Reference

```
#include <vstypes.h>
```

Include dependency graph for codec.h:



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



Data Structures

- struct `CodecServices`
- struct `Codec`

Defines

- #define `MAX_SOURCE_CHANNELS` 6
- #define `CODEC_VERSION` 0x0105
- #define `FS_CODEC_SERVICES_VERSION` 0x0125
- #define `FS_CODSER_COMMENT_END_OF_LINE` 0x4000U
- #define `FS_CODSER_COMMENT_END_OF_COMMENTS` ((u_int16)(0x8000U))
- #define `CS_VERSION_OFFSET` 0
- #define `CS_READ_OFFSET` 1
- #define `CS_SKIP_OFFSET` 2
- #define `CS_SEEK_OFFSET` 3
- #define `CS_TELL_OFFSET` 4
- #define `CS_OUTPUT_OFFSET` 5
- #define `CS_COMMENT_OFFSET` 6
- #define `CS_SPECTRUM_OFFSET` 7
- #define `CS_FILE_SIZE_OFFSET` 8
- #define `CS_FILE_LEFT_OFFSET` 10
- #define `CS_GO_TO_OFFSET` 12
- #define `CS_CANCEL_OFFSET` 13
- #define `CS_PLAY_TIME_SECONDS_OFFSET` 14
- #define `CS_PLAY_TIME_SAMPLES_OFFSET` 16
- #define `CS_PLAY_TIME_TOTAL_OFFSET` 18
- #define `CS_SAMPLE_RATE_OFFSET` 20
- #define `CS_CHANNELS_OFFSET` 22
- #define `CS_MATRIX_OFFSET` 23
- #define `CS_AVG_BIT_RATE_OFFSET` 29
- #define `CS_CURR_BIT_RATE_OFFSET` 31
- #define `CS_PEAK_BIT_RATE_OFFSET` 33
- #define `CS_GAIN_OFFSET` 35
- #define `CS_FAST_FORWARD_OFFSET` 36
- #define `CODEC_VERSION_OFFSET` 0
- #define `CODEC_CREATE_OFFSET` 1
- #define `CODEC_DECODE_OFFSET` 2
- #define `CODEC_DELETE_OFFSET` 3
- #define `CODEC_CS_OFFSET` 4

Enumerations

- enum `CodecError` {
`ceFastForward` = -1, `ceOk` = 0, `ceFormatNotFound`, `ceFormatNotSupported`,
`ceUnexpectedFileEnd`, `ceCancelled`, `ceOtherError` }

5.6.1 Detailed Description

Codec(p. 5) interfaces.

Definition in file `codec.h`.

5.6.2 Define Documentation

5.6.2.1 `#define CODEC_CREATE_OFFSET 1`

Definition at line 198 of file `codec.h`.

5.6.2.2 `#define CODEC_CS_OFFSET 4`

Definition at line 201 of file `codec.h`.

5.6.2.3 `#define CODEC_DECODE_OFFSET 2`

Definition at line 199 of file `codec.h`.

5.6.2.4 `#define CODEC_DELETE_OFFSET 3`

Definition at line 200 of file `codec.h`.

5.6.2.5 `#define CODEC_VERSION 0x0105`

Current version number. 8 MSBs contain version number, 8 LSBs size of the structure in words.

Version number history:

1. 0x0105 First version

Definition at line 26 of file `codec.h`.

5.6.2.6 `#define CODEC_VERSION_OFFSET 0`

Definition at line 197 of file `codec.h`.

5.6.2.7 `#define CS_AVG_BIT_RATE_OFFSET 29`

Definition at line 191 of file `codec.h`.

5.6.2.8 `#define CS_CANCEL_OFFSET 13`

Definition at line 184 of file `codec.h`.

5.6.2.9 `#define CS_CHANNELS_OFFSET 22`

Definition at line 189 of file `codec.h`.

5.6.2.10 #define CS_COMMENT_OFFSET 6

Definition at line 179 of file codec.h.

5.6.2.11 #define CS_CURR_BIT_RATE_OFFSET 31

Definition at line 192 of file codec.h.

5.6.2.12 #define CS_FAST_FORWARD_OFFSET 36

Definition at line 195 of file codec.h.

5.6.2.13 #define CS_FILE_LEFT_OFFSET 10

Definition at line 182 of file codec.h.

5.6.2.14 #define CS_FILE_SIZE_OFFSET 8

Definition at line 181 of file codec.h.

5.6.2.15 #define CS_GAIN_OFFSET 35

Definition at line 194 of file codec.h.

5.6.2.16 #define CS_GO_TO_OFFSET 12

Definition at line 183 of file codec.h.

5.6.2.17 #define CS_MATRIX_OFFSET 23

Definition at line 190 of file codec.h.

5.6.2.18 #define CS_OUTPUT_OFFSET 5

Definition at line 178 of file codec.h.

5.6.2.19 #define CS_PEAK_BIT_RATE_OFFSET 33

Definition at line 193 of file codec.h.

5.6.2.20 #define CS_PLAY_TIME_SAMPLES_OFFSET 16

Definition at line 186 of file codec.h.

5.6.2.21 #define CS_PLAY_TIME_SECONDS_OFFSET 14

Definition at line 185 of file codec.h.

5.6.2.22 #define CS_PLAY_TIME_TOTAL_OFFSET 18

Definition at line 187 of file codec.h.

5.6.2.23 #define CS_READ_OFFSET 1

Definition at line 174 of file codec.h.

5.6.2.24 #define CS_SAMPLE_RATE_OFFSET 20

Definition at line 188 of file codec.h.

5.6.2.25 #define CS_SEEK_OFFSET 3

Definition at line 176 of file codec.h.

5.6.2.26 #define CS_SKIP_OFFSET 2

Definition at line 175 of file codec.h.

5.6.2.27 #define CS_SPECTRUM_OFFSET 7

Definition at line 180 of file codec.h.

5.6.2.28 #define CS_TELL_OFFSET 4

Definition at line 177 of file codec.h.

5.6.2.29 #define CS_VERSION_OFFSET 0

Definition at line 173 of file codec.h.

5.6.2.30 #define FS_CODEC_SERVICES_VERSION 0x0125

Current version number. 8 MSBs contain version number, 8 LSBs size of the structure in words.

Version number history:

1. 0x0125 First version

Definition at line 37 of file codec.h.

5.6.2.31 #define FS_CODSER_COMMENT_END_OF_COMMENTS ((u_int16)(0x8000U))

Definition at line 41 of file codec.h.

5.6.2.32 `#define FS_CODSER_COMMENT_END_OF_LINE 0x4000U`

Definition at line 40 of file codec.h.

5.6.2.33 `#define MAX_SOURCE_CHANNELS 6`

Definition at line 13 of file codec.h.

5.6.3 Enumeration Type Documentation

5.6.3.1 enum CodecError

Codec(p. 5) error codes.

Enumerator:

- ceFastForward* Fast forwarded through file end
- ceOk* No errors.
- ceFormatNotFound* Data file was not in known format for the codec
- ceFormatNotSupported* Data file subformat is not supported.
- ceUnexpectedFileEnd* Unexpectedly early end of file
- ceCancelled* Playback cancel was requested
- ceOtherError* Unspecific error

Definition at line 142 of file codec.h.

```

142         {
143     ceFastForward = -1,
144     ceOk = 0,
145     ceFormatNotFound,
146     ceFormatNotSupported,
147     ceUnexpectedFileEnd,
148     ceCancelled,
149     ceOtherError
150 };

```

5.7 codecminiwav.h File Reference

```
#include <vstypes.h>
```

```
#include <codec.h>
```

Include dependency graph for codecminiwav.h:



Functions

- **Codec * CodMiniWavCreate** (void)
- void **CodMiniWavDelete** (struct **Codec** *cod)
- enum **CodecError CodMiniWavDecode** (struct **Codec** *cod, struct **CodecServices** *cs, const char **errorString)

5.7.1 Detailed Description

MinimalWav **Codec**(p. 5). At this moment the Wav **Codec**(p. 5) supports 8-bit and 16-bit stereo and mono files. Random access for seekable supported for all audio formats.

Definition in file **codecminiwav.h**.

5.7.2 Function Documentation

5.7.2.1 struct Codec* CodMiniWavCreate (void)

Create and allocate space for codec.

Returns:

A Wav **Codec**(p. 5) structure.

5.7.2.2 enum CodecError CodMiniWavDecode (struct Codec * cod, struct CodecServices * cs, const char ** errorString)

Decode file. Upon success or a negative number, **Codec**(p. 5) has succeeded. With a positive number, there has been an error. Upon return, an error string is also returned.

Parameters:

cod A Wav **Codec**(p. 5) structure.

cs User-supplied codec services with appropriate fields filled. The fields that are not to be filled by the user should be zero initialized.

errorString A pointer to a char pointer. The codec may return its error status here.

Returns:

Error code.

5.7.2.3 void CodMiniWavDelete (struct Codec * cod)

Free all resources allocated for codec.

Parameters:

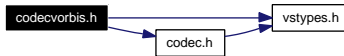
cod A Wav **Codec**(p. 5) structure.

5.8 codecvorbis.h File Reference

```
#include <vstypes.h>
```

```
#include <codec.h>
```

Include dependency graph for codecvorbis.h:



Defines

- `#define USE_ COMMENTS`

Functions

- `Codec * CodVorbisCreate` (void)
- `void CodVorbisDelete` (struct `Codec` *cod)
- `enum CodecError CodVorbisDecode` (struct `Codec` *cod, struct `CodecServices` *cs, const char **errorString)
- `auto s_int16 CodVBlockSize` (register `__i0` struct `Codec` *c, register `__a0` s_int16 bType)

5.8.1 Detailed Description

Vorbis 1.0 `Codec`(p. 5). The Vorbis `Codec`(p. 5) supports windows upto 4096 samples (8192 sample windows are NOT supported). Also random access is not supported.

Definition in file `codecvorbis.h`.

5.8.2 Define Documentation

5.8.2.1 `#define USE_ COMMENTS`

Definition at line 13 of file `codecvorbis.h`.

5.8.3 Function Documentation

5.8.3.1 `auto s_int16 CodVBlockSize` (register `__i0` struct `Codec` *c, register `__a0` s_int16 bType)

5.8.3.2 `struct Codec* CodVorbisCreate` (void)

Create and allocate space for codec.

Returns:

A Vorbis **Codec**(p. 5) structure.

5.8.3.3 enum CodecError CodVorbisDecode (struct Codec * *cod*, struct CodecServices * *cs*, const char ** *errorString*)

Decode file. Upon success or a negative number, **Codec**(p. 5) has succeeded. With a positive number, there has been an error. Upon return, an error string is also returned.

Parameters:

cod A Vorbis **Codec**(p. 5) structure.

cs User-supplied codec services with appropriate fields filled. The fields that are not to be filled by the user should be zero initialized.

errorString A pointer to a char pointer. The codec may return its error status here.

Returns:

Error code.

5.8.3.4 void CodVorbisDelete (struct Codec * *cod*)

Free all resources allocated for codec.

Parameters:

cod A Vorbis **Codec**(p. 5) structure.

5.9 ctype.h File Reference**Functions**

- int **isalnum** (int register __i0 c)
- int **isalpha** (int register __i0 c)
- int **iscntrl** (int register __i0 c)
- int **isdigit** (int register __i0 c)
- int **isgraph** (int register __i0 c)
- int **islower** (int register __i0 c)
- int **isprint** (int register __i0 c)
- int **ispunct** (int register __i0 c)
- int **isspace** (int register __i0 c)
- int **isupper** (int register __i0 c)
- int **isxdigit** (int register __i0 c)
- int register __i0 **tolower** (int register __i0 c)
- int register __i0 **toupper** (int register __i0 c)

5.9.1 Detailed Description

Standard C header file.

Definition in file `ctype.h`.

5.9.2 Function Documentation

5.9.2.1 `int isalnum (int register __i0 c)`

return TRUE for alphanumerical.

5.9.2.2 `int isalpha (int register __i0 c)`

return TRUE for alphabetic character.

5.9.2.3 `int iscntrl (int register __i0 c)`

return TRUE for control characters.

5.9.2.4 `int isdigit (int register __i0 c)`

return TRUE for digits 0-9.

5.9.2.5 `int isgraph (int register __i0 c)`

return TRUE for printable characters except space.

5.9.2.6 `int islower (int register __i0 c)`

return TRUE for lowercase characters.

5.9.2.7 `int isprint (int register __i0 c)`

return TRUE for printable characters, including space.

5.9.2.8 `int ispunct (int register __i0 c)`

return TRUE for printable characters that is not a space or alphanumeric.

5.9.2.9 `int isspace (int register __i0 c)`

return TRUE for white-space characters.

5.9.2.10 `int isupper (int register __i0 c)`

return TRUE for uppercase characters.

5.9.2.11 int isxdigit (int register __i0 c)

return TRUE for hexadecimal digits.

5.9.2.12 int register __i0 tolower (int register __i0 c)

convert letter c to lower case if possible.

5.9.2.13 int register __i0 toupper (int register __i0 c)

convert letter c to upper case if possible.

5.10 dev1000.h File Reference

```
#include <vstypes.h>
```

Include dependency graph for dev1000.h:

**Defines**

- #define PATCH_TEST_UNIT_READY
- #define MMC_MISO_BIT 8
- #define MMC_MOSI_BIT 12
- #define MMC_CLK (1<<9)
- #define MMC_MOSI (1<<MMC_MOSI_BIT)
- #define MMC_MISO (1<<MMC_MISO_BIT)
- #define MMC_XCS (1<<11)
- #define MMC_GO_IDLE_STATE 0
- #define MMC_SEND_OP_COND 1
- #define MMC_SEND_IF_COND 8
- #define MMC_SEND_CSD 9
- #define MMC_SEND_CID 10
- #define MMC_SEND_STATUS 13
- #define MMC_SET_BLOCKLEN 16
- #define MMC_READ_SINGLE_BLOCK 17
- #define MMC_WRITE_BLOCK 24
- #define MMC_PROGRAM_CSD 27
- #define MMC_SET_WRITE_PROT 28
- #define MMC_CLR_WRITE_PROT 29
- #define MMC_SEND_WRITE_PROT 30
- #define MMC_TAG_SECTOR_START 32
- #define MMC_TAG_SECTOR_END 33
- #define MMC_UNTAG_SECTOR 34
- #define MMC_TAG_ERASE_GROUP_START 35

- #define MMC_TAG_ERASE_GROUP_END 36
- #define MMC_UNTAG_ERASE_GROUP 37
- #define MMC_ERASE 38
- #define MMC_READ_OCR 58
- #define MMC_CRC_ON_OFF 59
- #define MMC_R1_BUSY 0x80
- #define MMC_R1_PARAMETER 0x40
- #define MMC_R1_ADDRESS 0x20
- #define MMC_R1_ERASE_SEQ 0x10
- #define MMC_R1_COM_CRC 0x08
- #define MMC_R1_ILLEGAL_COM 0x04
- #define MMC_R1_ERASE_RESET 0x02
- #define MMC_R1_IDLE_STATE 0x01
- #define MMC_STARTBLOCK_READ 0xFE
- #define MMC_STARTBLOCK_WRITE 0xFE
- #define MMC_STARTBLOCK_MWRITE 0xFC
- #define MMC_STOPTRAN_WRITE 0xFD
- #define MMC_DE_MASK 0x1F
- #define MMC_DE_ERROR 0x01
- #define MMC_DE_CC_ERROR 0x02
- #define MMC_DE_ECC_FAIL 0x04
- #define MMC_DE_OUT_OF_RANGE 0x04
- #define MMC_DE_CARD_LOCKED 0x04
- #define MMC_DR_MASK 0x1F
- #define MMC_DR_ACCEPT 0x05
- #define MMC_DR_REJECT_CRC 0x0B
- #define MMC_DR_REJECT_WRITE_ERROR 0x0D
- #define KEY_6 (1<<5)
- #define RC5_SYS_TVSET 0
- #define RC5_SYS_TELETEXT 2
- #define RC5_SYS_VCR 5
- #define RC5_SYS_EXPERIMENTAL7 7
- #define RC5_SYS_PREAMP 16
- #define RC5_SYS_RECEIVERTUNER 17
- #define RC5_SYS_TAPERECORDER 18
- #define RC5_SYS_EXPERIMENTAL19 19
- #define RC5_SYS_CDPLAYER 20
- #define RC5_CMD_0 0
- #define RC5_CMD_1 1
- #define RC5_CMD_2 2
- #define RC5_CMD_3 3
- #define RC5_CMD_4 4
- #define RC5_CMD_5 5
- #define RC5_CMD_6 6
- #define RC5_CMD_7 7
- #define RC5_CMD_8 8

- #define **RC5_CMD_9** 9
- #define **RC5_CMD_ONETWODIGITS** 10
- #define **RC5_CMD_STANDBY** 12
- #define **RC5_CMD_MUTE** 13
- #define **RC5_CMD_PRESET** 14
- #define **RC5_CMD_DISPLAY** 15
- #define **RC5_CMD_VOLUME_UP** 16
- #define **RC5_CMD_VOLUME_DOWN** 17
- #define **RC5_CMD_BRIGHT_UP** 18
- #define **RC5_CMD_BRIGHT_DOWN** 19
- #define **RC5_CMD_COLOR_UP** 20
- #define **RC5_CMD_COLOR_DOWN** 21
- #define **RC5_CMD_BASS_UP** 22
- #define **RC5_CMD_BASS_DOWN** 23
- #define **RC5_CMD_TREBLE_UP** 24
- #define **RC5_CMD_TREBLE_DOWN** 25
- #define **RC5_CMD_BALANCE_LEFT** 26
- #define **RC5_CMD_BALANCE_RIGHT** 27
- #define **RC5_CMD_CONTRAST_UP** 28
- #define **RC5_CMD_CONTRAST_DOWN** 29
- #define **RC5_CMD_PROGRAM_UP** 32
- #define **RC5_CMD_PROGRAM_DOWN** 33
- #define **RC5_CMD_ALTERNATE** 34
- #define **RC5_CMD_LANGUAGE** 35
- #define **RC5_CMD_EXPAND** 36
- #define **RC5_CMD_TIMER** 38
- #define **RC5_CMD_STORE** 41
- #define **RC5_CMD_CLOCK** 42
- #define **RC5_CMD_FINETUNE_PLUS** 43
- #define **RC5_CMD_FINETUNE_MINUS** 44
- #define **RC5_CMD_CROSS** 45
- #define **RC5_CMD_MIX** 46
- #define **RC5_CMD_PAUSE** 48
- #define **RC5_CMD_FAST_REVERSE** 50
- #define **RC5_CMD_FAST_FORWARD** 52
- #define **RC5_CMD_PLAY** 53
- #define **RC5_CMD_STOP** 54
- #define **RC5_CMD_RECORD** 55
- #define **RC5_CMD_SWITCH** 56
- #define **RC5_CMD_MENU** 59
- #define **RC5_CMD_TEXT** 60
- #define **RC5_CMD_SYSTEM_SELECT** 63

Functions

- auto **u_int16 SpiSendReceiveMmc** (register __a0 **u_int16** dataTopAligned, register __a1 bits)
- auto void **SpiSendClocks** (void)
- auto **u_int16 MmcCommand** (register __b0 **s_int16** cmd, register __d **u_int32** arg)
- void **PatchMSCPacketFromPC** (void *)
- void **ScsiTestUnitReady** (void)
- auto **u_int16 Fat12OpenFile** (register __c0 **u_int16** fileNum)
- auto **u_int16 OpenFileBaseName** (register __i2 const **u_int16** *packedName)
- void **PlayRangeSet** (**u_int32** start, **u_int32** end)
- void **PlayRange** (void)
- void **puthex** (**u_int16** d)
- void **KeyScan7** (void)
- void **Suspend7** (void)
- auto **u_int16 MapperlessReadDiskSector** (register __i0 **u_int16** *buffer, register __reg_a **u_int32** sector)
- void **NandPutCommand** (register __a0 **u_int16** command)
- void **NandPutAddressOctet** (register __a0 **u_int16** address)
- void **NandGetOctets** (register __c0 **s_int16** length, register __i2 **u_int16** *buf)
- void **NandPutOctets** (register __c0 **s_int16** length, register __i2 **u_int16** *buf)
- void **NandSetWaits** (register __a0 **u_int16** waitns)
- **u_int32 ReadIRam** (register __i0 **u_int16** addr)
- void **WriteIRam** (register __i0 **u_int16** addr, register __a **u_int32** ins)
- void **InterruptStub0** (void)
- void **InterruptStub1** (void)
- void **InterruptStub2** (void)
- void **InterruptStub3** (void)
- auto void **Interrupt0** (void)
- auto void **Interrupt1** (void)
- auto void **Interrupt2** (void)
- auto void **Interrupt3** (void)
- void **Rc5Init** (**u_int16** vector)
- **u_int16 Rc5GetFIFO** (void)

5.10.1 Detailed Description

Development library header file.

Definition in file **dev1000.h**.

5.10.2 Define Documentation

5.10.2.1 `#define KEY_6 (1<<5)`

Definition at line 122 of file dev1000.h.

5.10.2.2 `#define MMC_CLK (1<<9)`

Definition at line 12 of file dev1000.h.

5.10.2.3 `#define MMC_CLR_WRITE_PROT 29`

Definition at line 28 of file dev1000.h.

5.10.2.4 `#define MMC_CRC_ON_OFF 59`

Definition at line 38 of file dev1000.h.

5.10.2.5 `#define MMC_DE_CARD_LOCKED 0x04`

Definition at line 56 of file dev1000.h.

5.10.2.6 `#define MMC_DE_CC_ERROR 0x02`

Definition at line 53 of file dev1000.h.

5.10.2.7 `#define MMC_DE_ECC_FAIL 0x04`

Definition at line 54 of file dev1000.h.

5.10.2.8 `#define MMC_DE_ERROR 0x01`

Definition at line 52 of file dev1000.h.

5.10.2.9 `#define MMC_DE_MASK 0x1F`

Definition at line 51 of file dev1000.h.

5.10.2.10 `#define MMC_DE_OUT_OF_RANGE 0x04`

Definition at line 55 of file dev1000.h.

5.10.2.11 `#define MMC_DR_ACCEPT 0x05`

Definition at line 58 of file dev1000.h.

5.10.2.12 `#define MMC_DR_MASK 0x1F`

Definition at line 57 of file dev1000.h.

5.10.2.13 #define MMC_DR_REJECT_CRC 0x0B

Definition at line 59 of file dev1000.h.

5.10.2.14 #define MMC_DR_REJECT_WRITE_ERROR 0x0D

Definition at line 60 of file dev1000.h.

5.10.2.15 #define MMC_ERASE 38

Definition at line 36 of file dev1000.h.

5.10.2.16 #define MMC_GO_IDLE_STATE 0

Definition at line 17 of file dev1000.h.

5.10.2.17 #define MMC_MISO (1<<MMC_MISO_BIT)

Definition at line 14 of file dev1000.h.

5.10.2.18 #define MMC_MISO_BIT 8

Definition at line 10 of file dev1000.h.

5.10.2.19 #define MMC_MOSI (1<<MMC_MOSI_BIT)

Definition at line 13 of file dev1000.h.

5.10.2.20 #define MMC_MOSI_BIT 12

Definition at line 11 of file dev1000.h.

5.10.2.21 #define MMC_PROGRAM_CSD 27

Definition at line 26 of file dev1000.h.

5.10.2.22 #define MMC_R1_ADDRESS 0x20

Definition at line 41 of file dev1000.h.

5.10.2.23 #define MMC_R1_BUSY 0x80

Definition at line 39 of file dev1000.h.

5.10.2.24 #define MMC_R1_COM_CRC 0x08

Definition at line 43 of file dev1000.h.

5.10.2.25 `#define MMC_R1_ERASE_RESET 0x02`

Definition at line 45 of file dev1000.h.

5.10.2.26 `#define MMC_R1_ERASE_SEQ 0x10`

Definition at line 42 of file dev1000.h.

5.10.2.27 `#define MMC_R1_IDLE_STATE 0x01`

Definition at line 46 of file dev1000.h.

5.10.2.28 `#define MMC_R1_ILLEGAL_COM 0x04`

Definition at line 44 of file dev1000.h.

5.10.2.29 `#define MMC_R1_PARAMETER 0x40`

Definition at line 40 of file dev1000.h.

5.10.2.30 `#define MMC_READ_OCR 58`

Definition at line 37 of file dev1000.h.

5.10.2.31 `#define MMC_READ_SINGLE_BLOCK 17`

Definition at line 24 of file dev1000.h.

5.10.2.32 `#define MMC_SEND_CID 10`

Definition at line 21 of file dev1000.h.

5.10.2.33 `#define MMC_SEND_CSD 9`

Definition at line 20 of file dev1000.h.

5.10.2.34 `#define MMC_SEND_IF_COND 8`

Definition at line 19 of file dev1000.h.

5.10.2.35 `#define MMC_SEND_OP_COND 1`

Definition at line 18 of file dev1000.h.

5.10.2.36 `#define MMC_SEND_STATUS 13`

Definition at line 22 of file dev1000.h.

5.10.2.37 #define MMC_SEND_WRITE_PROT 30

Definition at line 29 of file dev1000.h.

5.10.2.38 #define MMC_SET_BLOCKLEN 16

Definition at line 23 of file dev1000.h.

5.10.2.39 #define MMC_SET_WRITE_PROT 28

Definition at line 27 of file dev1000.h.

5.10.2.40 #define MMC_STARTBLOCK_MWRITE 0xFC

Definition at line 49 of file dev1000.h.

5.10.2.41 #define MMC_STARTBLOCK_READ 0xFE

Definition at line 47 of file dev1000.h.

5.10.2.42 #define MMC_STARTBLOCK_WRITE 0xFE

Definition at line 48 of file dev1000.h.

5.10.2.43 #define MMC_STOPTRAN_WRITE 0xFD

Definition at line 50 of file dev1000.h.

5.10.2.44 #define MMC_TAG_ERARE_GROUP_END 36

Definition at line 34 of file dev1000.h.

5.10.2.45 #define MMC_TAG_ERASE_GROUP_START 35

Definition at line 33 of file dev1000.h.

5.10.2.46 #define MMC_TAG_SECTOR_END 33

Definition at line 31 of file dev1000.h.

5.10.2.47 #define MMC_TAG_SECTOR_START 32

Definition at line 30 of file dev1000.h.

5.10.2.48 #define MMC_UNTAG_ERASE_GROUP 37

Definition at line 35 of file dev1000.h.

5.10.2.49 #define MMC_UNTAG_SECTOR 34

Definition at line 32 of file dev1000.h.

5.10.2.50 #define MMC_WRITE_BLOCK 24

Definition at line 25 of file dev1000.h.

5.10.2.51 #define MMC_XCS (1<<11)

Definition at line 15 of file dev1000.h.

5.10.2.52 #define PATCH_TEST_UNIT_READY

Definition at line 9 of file dev1000.h.

5.10.2.53 #define RC5_CMD_0 0

Definition at line 205 of file dev1000.h.

5.10.2.54 #define RC5_CMD_1 1

Definition at line 206 of file dev1000.h.

5.10.2.55 #define RC5_CMD_2 2

Definition at line 207 of file dev1000.h.

5.10.2.56 #define RC5_CMD_3 3

Definition at line 208 of file dev1000.h.

5.10.2.57 #define RC5_CMD_4 4

Definition at line 209 of file dev1000.h.

5.10.2.58 #define RC5_CMD_5 5

Definition at line 210 of file dev1000.h.

5.10.2.59 #define RC5_CMD_6 6

Definition at line 211 of file dev1000.h.

5.10.2.60 #define RC5_CMD_7 7

Definition at line 212 of file dev1000.h.

5.10.2.61 #define RC5_CMD_8 8

Definition at line 213 of file dev1000.h.

5.10.2.62 #define RC5_CMD_9 9

Definition at line 214 of file dev1000.h.

5.10.2.63 #define RC5_CMD_ALTERNATE 34

Definition at line 236 of file dev1000.h.

5.10.2.64 #define RC5_CMD_BALANCE_LEFT 26

Definition at line 230 of file dev1000.h.

5.10.2.65 #define RC5_CMD_BALANCE_RIGHT 27

Definition at line 231 of file dev1000.h.

5.10.2.66 #define RC5_CMD_BASS_DOWN 23

Definition at line 227 of file dev1000.h.

5.10.2.67 #define RC5_CMD_BASS_UP 22

Definition at line 226 of file dev1000.h.

5.10.2.68 #define RC5_CMD_BRIGHT_DOWN 19

Definition at line 223 of file dev1000.h.

5.10.2.69 #define RC5_CMD_BRIGHT_UP 18

Definition at line 222 of file dev1000.h.

5.10.2.70 #define RC5_CMD_CLOCK 42

Definition at line 241 of file dev1000.h.

5.10.2.71 #define RC5_CMD_COLOR_DOWN 21

Definition at line 225 of file dev1000.h.

5.10.2.72 #define RC5_CMD_COLOR_UP 20

Definition at line 224 of file dev1000.h.

5.10.2.73 #define RC5_CMD_CONTRAST_DOWN 29

Definition at line 233 of file dev1000.h.

5.10.2.74 #define RC5_CMD_CONTRAST_UP 28

Definition at line 232 of file dev1000.h.

5.10.2.75 #define RC5_CMD_CROSS 45

Definition at line 244 of file dev1000.h.

5.10.2.76 #define RC5_CMD_DISPLAY 15

Definition at line 219 of file dev1000.h.

5.10.2.77 #define RC5_CMD_EXPAND 36

Definition at line 238 of file dev1000.h.

5.10.2.78 #define RC5_CMD_FAST_FORWARD 52

Definition at line 248 of file dev1000.h.

5.10.2.79 #define RC5_CMD_FAST_REVERSE 50

Definition at line 247 of file dev1000.h.

5.10.2.80 #define RC5_CMD_FINETUNE_MINUS 44

Definition at line 243 of file dev1000.h.

5.10.2.81 #define RC5_CMD_FINETUNE_PLUS 43

Definition at line 242 of file dev1000.h.

5.10.2.82 #define RC5_CMD_LANGUAGE 35

Definition at line 237 of file dev1000.h.

5.10.2.83 #define RC5_CMD_MENU 59

Definition at line 253 of file dev1000.h.

5.10.2.84 #define RC5_CMD_MIX 46

Definition at line 245 of file dev1000.h.

5.10.2.85 #define RC5_CMD_MUTE 13

Definition at line 217 of file dev1000.h.

5.10.2.86 #define RC5_CMD_ONETWODIGITS 10

Definition at line 215 of file dev1000.h.

5.10.2.87 #define RC5_CMD_PAUSE 48

Definition at line 246 of file dev1000.h.

5.10.2.88 #define RC5_CMD_PLAY 53

Definition at line 249 of file dev1000.h.

5.10.2.89 #define RC5_CMD_PRESET 14

Definition at line 218 of file dev1000.h.

5.10.2.90 #define RC5_CMD_PROGRAM_DOWN 33

Definition at line 235 of file dev1000.h.

5.10.2.91 #define RC5_CMD_PROGRAM_UP 32

Definition at line 234 of file dev1000.h.

5.10.2.92 #define RC5_CMD_RECORD 55

Definition at line 251 of file dev1000.h.

5.10.2.93 #define RC5_CMD_STANDBY 12

Definition at line 216 of file dev1000.h.

5.10.2.94 #define RC5_CMD_STOP 54

Definition at line 250 of file dev1000.h.

5.10.2.95 #define RC5_CMD_STORE 41

Definition at line 240 of file dev1000.h.

5.10.2.96 #define RC5_CMD_SWITCH 56

Definition at line 252 of file dev1000.h.

5.10.2.97 #define RC5_CMD_SYSTEM_SELECT 63

Definition at line 255 of file dev1000.h.

5.10.2.98 #define RC5_CMD_TEXT 60

Definition at line 254 of file dev1000.h.

5.10.2.99 #define RC5_CMD_TIMER 38

Definition at line 239 of file dev1000.h.

5.10.2.100 #define RC5_CMD_TREBLE_DOWN 25

Definition at line 229 of file dev1000.h.

5.10.2.101 #define RC5_CMD_TREBLE_UP 24

Definition at line 228 of file dev1000.h.

5.10.2.102 #define RC5_CMD_VOLUME_DOWN 17

Definition at line 221 of file dev1000.h.

5.10.2.103 #define RC5_CMD_VOLUME_UP 16

Definition at line 220 of file dev1000.h.

5.10.2.104 #define RC5_SYS_CDPLAYER 20

Definition at line 203 of file dev1000.h.

5.10.2.105 #define RC5_SYS_EXPERIMENTAL19 19

Definition at line 202 of file dev1000.h.

5.10.2.106 #define RC5_SYS_EXPERIMENTAL7 7

Definition at line 198 of file dev1000.h.

5.10.2.107 #define RC5_SYS_PREAMP 16

Definition at line 199 of file dev1000.h.

5.10.2.108 #define RC5_SYS_RECEIVERTUNER 17

Definition at line 200 of file dev1000.h.

5.10.2.109 #define RC5_SYS_TAPERECORDER 18

Definition at line 201 of file dev1000.h.

5.10.2.110 #define RC5_SYS_TELETEXT 2

Definition at line 196 of file dev1000.h.

5.10.2.111 #define RC5_SYS_TVSET 0

Definition at line 195 of file dev1000.h.

5.10.2.112 #define RC5_SYS_VCR 5

Definition at line 197 of file dev1000.h.

5.10.3 Function Documentation**5.10.3.1 auto u_int16 Fat12OpenFile (register __c0 u_int16 *fileNum*)**

Replacement for FatOpenFile that is normally in OpenFile hook. This version disables subdirectories for FAT12 partitions so OpenFile will not go into infinite recursion trying to handle it. Is not needed for VS1000d, but is compatible with it.

```
SetHookFunction((u_int16)OpenFile, Fat12OpenFile);
```

Parameters:

fileNum the same as for OpenFile

Returns:

the same as OpenFile

5.10.3.2 auto void Interrupt0 (void)

Called by InterruptStub0.

5.10.3.3 auto void Interrupt1 (void)

Called by InterruptStub1.

5.10.3.4 auto void Interrupt2 (void)

Called by InterruptStub2.

5.10.3.5 auto void Interrupt3 (void)

Called by InterruptStub3.

5.10.3.6 void InterruptStub0 (void)

Four interrupt stubs that the programmer can use without using ASM. For example to set up the first interrupt stub to GPIO0 interrupt, use the following:

```
WriteIRam(0x20+INTV_GPIO0, ReadIRam((u_int16)InterruptStub0));
```

Then you can enable GPIO0 interrupt and it will call your routine **Interrupt0()** (p. 64) whenever GPIO0 interrupt request is generated.

```
PERIP(GPIO0_INT_FALL) |= DISPLAY_XCS;
PERIP(INT_ENABLEL) |= INTF_GPIO0;
```

5.10.3.7 void InterruptStub1 (void)

Stub function that can be plugged to any interrupt vector. Calls **Interrupt1()** (p. 64), which must be provided by the user.

5.10.3.8 void InterruptStub2 (void)

Stub function that can be plugged to any interrupt vector. Calls **Interrupt2()** (p. 64), which must be provided by the user.

5.10.3.9 void InterruptStub3 (void)

Stub function that can be plugged to any interrupt vector. Calls **Interrupt3()** (p. 64), which must be provided by the user.

5.10.3.10 void KeyScan7 (void)

Replacement routine for **KeyScan()** (p. 99) to read GPIO[5:0] and power button instead of just GPIO[4:0] and power button. See also **Suspend7()** (p. 68).

5.10.3.11 auto u_int16 MapperlessReadDiskSector (register __i0 u_int16 * buffer, register __reg_a u_int32 sector)**5.10.3.12 auto u_int16 MmcCommand (register __b0 s_int16 cmd, register __d u_int32 arg)**

Send MMC/SD command. The CRC that is sent will always be 0x95.

Parameters:

- cmd* The MMC command, must include the start bit (0x40).
- arg* The command argument.

Returns:

the result code or 0xff (8th bit set) for timeout.

5.10.3.13 void NandGetOctets (register __c0 s_int16 *length*, register __i2 u_int16 * *buf*)

Writes packed data to NFIO. You must handle NFCS yourself.

5.10.3.14 void NandPutAddressOctet (register __a0 u_int16 *address*)

writes 8 bits with ALE=1. You must handle NFCS yourself.

5.10.3.15 void NandPutCommand (register __a0 u_int16 *command*)

Writes 8 bits with CLE=1. You must handle NFCS yourself.

5.10.3.16 void NandPutOctets (register __c0 s_int16 *length*, register __i2 u_int16 * *buf*)

Reads packed data from NFIO. You must handle NFCS yourself.

5.10.3.17 void NandSetWaits (register __a0 u_int16 *waitns*)

Sets NAND-interface waitstates in nanoseconds. Uses current clockX value for calculation.

5.10.3.18 auto u_int16 OpenFileName (register __i2 const u_int16 * *packedName*)

Open a file based on the name. The suffix is not changed, the caller must select the right suffix and restore the original after the call.

Parameters:

packedName The 8-character name must be in upper case and in packed format.

Returns:

The file index or 0xffffU if file is not found.

5.10.3.19 void PatchMSCPacketFromPC (void *)

LBAB patch – removes 4G restriction from USB (SCSI). Is not needed for VS1000d, but is compatible with it.

5.10.3.20 void PlayRange (void)

Plays from a previously set start position to end position. Handles player and cs structure initialization. You must call **PlayRangeSet()**(p.67) before calling **PlayRange()**(p.66).

5.10.3.21 void PlayRangeSet (u_int32 start, u_int32 end)

Sets the start and end positions for the **PlayRange()**(p.66) function.

Parameters:

start Play start time in 1/65536th of a second resolution.

end Play end time in 1/65536th of a second resolution. Set to 0x7fffffffUL to disable end time.

5.10.3.22 void puthex (u_int16 d)

For debugging with vs3emu: print 4-digit hex number.

5.10.3.23 u_int16 Rc5GetFIFO (void)

Returns 0 if no values are available, received 14-bit value otherwise.

5.10.3.24 void Rc5Init (u_int16 vector)

Initializes RC5 structure and installs interrupt handler. RC5 receiver uses one of the interruptable GPIO pins as input. Interrupt must be generated on both edges. The polarity of the receiver does not matter. Currently receives only codes that have two start bits.

```
Rc5Init(INTV_GPIO0);
PERIP(GPIO0_INT_FALL) |= (1<<14);
PERIP(GPIO0_INT_RISE) |= (1<<14);
PERIP(INT_ENABLEL) |= INTF_GPIO0;
while (1) {
    register u_int16 t = Rc5GetFIFO();
    if (t) {
        puthex(t);
        puts("=got");
    }
}
```

See also example code rc5.c .

5.10.3.25 u_int32 ReadIRam (register __i0 u_int16 addr)

Reads instruction RAM.

5.10.3.26 void ScsiTestUnitReady (void)

Hook called by **PatchMSCPacketFromPC()**(p.66).

5.10.3.27 auto void SpiSendClocks (void)

Send MMC clocks with XCS high.

5.10.3.28 `auto u_int16 SpiSendReceiveMmc (register __a0 u_int16 dataTopAligned, register __a1 bits)`

Send and receive bits from MMC/SD.

Parameters:

dataTopAligned The first bit to send must be in the most-significant bit.

bits The number of bits to send/receive.

Returns:

If less than 16 bits are sent, the high bits of the result will be the low bits of the `dataTopAligned` parameter.

5.10.3.29 `void Suspend7 (void)`

Replacement routine for `USBSuspend()`(p.101) to wake up from GPIO[5:0] and power button (and USB pins) instead of just GPIO[4:0] and power button (and USB pins). Also puts the analog to powerdown for the duration of USB suspend/low-power pause. See also `KeyScan7()`(p.65).

5.10.3.30 `void WriteIRam (register __i0 u_int16 addr, register __a u_int32 ins)`

Writes instruction RAM.

5.11 `errno.h` File Reference

Defines

- `#define EPERM 1`
- `#define ENOENT 2`
- `#define ESRCH 3`
- `#define EINTR 4`
- `#define EIO 5`
- `#define ENXIO 6`
- `#define E2BIG 7`
- `#define ENOEXEC 8`
- `#define EBADF 9`
- `#define ECHILD 10`
- `#define EDEADLK 11`
- `#define ENOMEM 12`
- `#define EACCES 13`
- `#define EFAULT 14`
- `#define ENOTBLK 15`
- `#define EBUSY 16`
- `#define EEXIST 17`
- `#define EXDEV 18`

- `#define ENODEV` 19
- `#define ENOTDIR` 20
- `#define EISDIR` 21
- `#define EINVAL` 22
- `#define ENFILE` 23
- `#define EMFILE` 24
- `#define ENOTTY` 25
- `#define ETXTBSY` 26
- `#define EFBIG` 27
- `#define ENOSPC` 28
- `#define ESPIPE` 29
- `#define EROFS` 30
- `#define EMLINK` 31
- `#define EPIPE` 32
- `#define EDOM` 33
- `#define ERANGE` 34
- `#define EAGAIN` 35
- `#define EWOULDBLOCK` `EAGAIN`
- `#define EINPROGRESS` 36
- `#define EALREADY` 37

Variables

- `__near int errno`

5.11.1 Detailed Description

Standard C header file. Used with `stdio` functions and some math and `strto*` functions.

Definition in file `errno.h`.

5.11.2 Define Documentation

5.11.2.1 `#define E2BIG` 7

Definition at line 14 of file `errno.h`.

5.11.2.2 `#define EACCES` 13

Definition at line 21 of file `errno.h`.

5.11.2.3 `#define EAGAIN` 35

Definition at line 47 of file `errno.h`.

5.11.2.4 `#define EALREADY 37`

Definition at line 50 of file `errno.h`.

5.11.2.5 `#define EBADF 9`

Definition at line 16 of file `errno.h`.

5.11.2.6 `#define EBUSY 16`

Definition at line 24 of file `errno.h`.

5.11.2.7 `#define ECHILD 10`

Definition at line 17 of file `errno.h`.

5.11.2.8 `#define EDEADLK 11`

Definition at line 18 of file `errno.h`.

5.11.2.9 `#define EDOM 33`

Definition at line 43 of file `errno.h`.

5.11.2.10 `#define EEXIST 17`

Definition at line 25 of file `errno.h`.

5.11.2.11 `#define EFAULT 14`

Definition at line 22 of file `errno.h`.

5.11.2.12 `#define EFBIG 27`

Definition at line 35 of file `errno.h`.

5.11.2.13 `#define EINPROGRESS 36`

Definition at line 49 of file `errno.h`.

5.11.2.14 `#define EINTR 4`

Definition at line 11 of file `errno.h`.

5.11.2.15 `#define EINVAL 22`

Definition at line 30 of file `errno.h`.

5.11.2.16 `#define EIO 5`

Definition at line 12 of file `errno.h`.

5.11.2.17 `#define EISDIR 21`

Definition at line 29 of file `errno.h`.

5.11.2.18 `#define EMFILE 24`

Definition at line 32 of file `errno.h`.

5.11.2.19 `#define EMLINK 31`

Definition at line 39 of file `errno.h`.

5.11.2.20 `#define ENFILE 23`

Definition at line 31 of file `errno.h`.

5.11.2.21 `#define ENODEV 19`

Definition at line 27 of file `errno.h`.

5.11.2.22 `#define ENOENT 2`

Definition at line 9 of file `errno.h`.

5.11.2.23 `#define ENOEXEC 8`

Definition at line 15 of file `errno.h`.

5.11.2.24 `#define ENOMEM 12`

Definition at line 20 of file `errno.h`.

5.11.2.25 `#define ENOSPC 28`

Definition at line 36 of file `errno.h`.

5.11.2.26 `#define ENOTBLK 15`

Definition at line 23 of file `errno.h`.

5.11.2.27 `#define ENOTDIR 20`

Definition at line 28 of file `errno.h`.

5.11.2.28 `#define ENOTTY 25`

Definition at line 33 of file `errno.h`.

5.11.2.29 `#define ENXIO 6`

Definition at line 13 of file `errno.h`.

5.11.2.30 `#define EPERM 1`

Definition at line 8 of file `errno.h`.

5.11.2.31 `#define EPIPE 32`

Definition at line 40 of file `errno.h`.

5.11.2.32 `#define ERANGE 34`

Definition at line 44 of file `errno.h`.

5.11.2.33 `#define EROFS 30`

Definition at line 38 of file `errno.h`.

5.11.2.34 `#define ESPIPE 29`

Definition at line 37 of file `errno.h`.

5.11.2.35 `#define ESRCH 3`

Definition at line 10 of file `errno.h`.

5.11.2.36 `#define ETXTBSY 26`

Definition at line 34 of file `errno.h`.

5.11.2.37 `#define EWOULDBLOCK EAGAIN`

Definition at line 48 of file `errno.h`.

5.11.2.38 `#define EXDEV 18`

Definition at line 26 of file `errno.h`.

5.11.3 Variable Documentation**5.11.3.1** `__near int errno`

5.12 fat.h File Reference

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



Data Structures

- struct **FATINFO**
- struct **FRAGMENT**

Defines

- `#define FATINFO_IN_Y`
- `#define MAX_FRAGMENTS 35`
- `#define FAT_LFN_SIZE (2*13 *2)`
- `#define FAT_MKID(a, b, c) ((a)|((b)<<8)|((u_int32)(c)<<16))`
- `#define LAST_FRAGMENT 0x80000000UL`

Variables

- `__y struct FRAGMENT minifatFragments [MAX_FRAGMENTS]`
- `__y struct FATINFO minifatInfo`
- `u_int16 minifatBuffer [256]`

5.12.1 Detailed Description

FAT definitions.

Definition in file `fat.h`.

5.12.2 Define Documentation

5.12.2.1 `#define FAT_LFN_SIZE (2*13 *2)`

Definition at line 10 of file `fat.h`.

5.12.2.2 `#define FAT_MKID(a, b, c) ((a)|((b)<<8)|((u_int32)(c)<<16))`

Definition at line 41 of file `fat.h`.

5.12.2.3 `#define FATINFO_IN_Y`

Definition at line 7 of file `fat.h`.

5.12.2.4 #define LAST_FRAGMENT 0x80000000UL

Definition at line 66 of file fat.h.

5.12.2.5 #define MAX_FRAGMENTS 35

Definition at line 8 of file fat.h.

5.12.3 Variable Documentation**5.12.3.1 u_int16 minifatBuffer[256]**

holds one sector of data

5.12.3.2 __y struct FRAGMENT minifatFragments[MAX_FRAGMENTS]**5.12.3.3 __y struct FATINFO minifatInfo****5.13 float.h File Reference****Defines**

- #define FLT_RADIX 2
- #define FLT_ROUNDS -1
- #define FLT_MANT_DIG 15
- #define FLT_EPSILON 6.10351562E-5F
- #define FLT_DIG 4
- #define FLT_MIN_EXP (-1021)
- #define FLT_MIN 2.2250738585072014E-308
- #define FLT_MIN_10_EXP (-307)
- #define FLT_MAX_EXP 1024
- #define FLT_MAX 1.7976931348623157E+308
- #define FLT_MAX_10_EXP 308
- #define DBL_MANT_DIG 31
- #define DBL_EPSILON 4.6566128752458E-10
- #define DBL_DIG 9
- #define DBL_MIN_EXP (-1021)
- #define DBL_MIN 4.45014771494214E-308
- #define DBL_MIN_10_EXP (-307)
- #define DBL_MAX_EXP 1024
- #define DBL_MAX 1.79769313486232E+308
- #define DBL_MAX_10_EXP 308
- #define LDBL_MANT_DIG DBL_MANT_DIG
- #define LDBL_EPSILON DBL_EPSILON
- #define LDBL_DIG DBL_DIG

- `#define LDBL_MIN_EXP DBL_MIN_EXP`
- `#define LDBL_MIN DBL_MIN`
- `#define LDBL_MIN_10_EXP DBL_MIN_10_EXP`
- `#define LDBL_MAX_EXP DBL_MAX_EXP`
- `#define LDBL_MAX DBL_MAX`
- `#define LDBL_MAX_10_EXP DBL_MAX_10_EXP`

5.13.1 Detailed Description

Standard C header file. The float type should not be use with VCC (only 15-bit accuracy).

Definition in file `float.h`.

5.13.2 Define Documentation

5.13.2.1 `#define DBL_DIG 9`

Definition at line 25 of file `float.h`.

5.13.2.2 `#define DBL_EPSILON 4.6566128752458E-10`

Definition at line 24 of file `float.h`.

5.13.2.3 `#define DBL_MANT_DIG 31`

Definition at line 23 of file `float.h`.

5.13.2.4 `#define DBL_MAX 1.79769313486232E+308`

Definition at line 30 of file `float.h`.

5.13.2.5 `#define DBL_MAX_10_EXP 308`

Definition at line 31 of file `float.h`.

5.13.2.6 `#define DBL_MAX_EXP 1024`

Definition at line 29 of file `float.h`.

5.13.2.7 `#define DBL_MIN 4.45014771494214E-308`

Definition at line 27 of file `float.h`.

5.13.2.8 `#define DBL_MIN_10_EXP (-307)`

Definition at line 28 of file `float.h`.

5.13.2.9 #define DBL_MIN_EXP (-1021)

Definition at line 26 of file float.h.

5.13.2.10 #define FLT_DIG 4

Definition at line 15 of file float.h.

5.13.2.11 #define FLT_EPSILON 6.10351562E-5F

Definition at line 14 of file float.h.

5.13.2.12 #define FLT_MANT_DIG 15

Definition at line 13 of file float.h.

5.13.2.13 #define FLT_MAX 1.7976931348623157E+308

Definition at line 20 of file float.h.

5.13.2.14 #define FLT_MAX_10_EXP 308

Definition at line 21 of file float.h.

5.13.2.15 #define FLT_MAX_EXP 1024

Definition at line 19 of file float.h.

5.13.2.16 #define FLT_MIN 2.2250738585072014E-308

Definition at line 17 of file float.h.

5.13.2.17 #define FLT_MIN_10_EXP (-307)

Definition at line 18 of file float.h.

5.13.2.18 #define FLT_MIN_EXP (-1021)

Definition at line 16 of file float.h.

5.13.2.19 #define FLT_RADIX 2

Definition at line 10 of file float.h.

5.13.2.20 #define FLT_ROUNDS -1

Definition at line 11 of file float.h.

5.13.2.21 #define LDBL_DIG DBL_DIG

Definition at line 35 of file float.h.

5.13.2.22 #define LDBL_EPSILON DBL_EPSILON

Definition at line 34 of file float.h.

5.13.2.23 #define LDBL_MANT_DIG DBL_MANT_DIG

Definition at line 33 of file float.h.

5.13.2.24 #define LDBL_MAX DBL_MAX

Definition at line 40 of file float.h.

5.13.2.25 #define LDBL_MAX_10_EXP DBL_MAX_10_EXP

Definition at line 41 of file float.h.

5.13.2.26 #define LDBL_MAX_EXP DBL_MAX_EXP

Definition at line 39 of file float.h.

5.13.2.27 #define LDBL_MIN DBL_MIN

Definition at line 37 of file float.h.

5.13.2.28 #define LDBL_MIN_10_EXP DBL_MIN_10_EXP

Definition at line 38 of file float.h.

5.13.2.29 #define LDBL_MIN_EXP DBL_MIN_EXP

Definition at line 36 of file float.h.

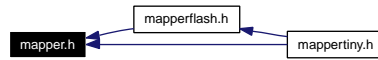
5.14 mapper.h File Reference

```
#include <vstypes.h>
```

Include dependency graph for mapper.h:



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



Data Structures

- struct **FsMapper**

Defines

- #define **FS_MAPPER_VERSION** 0x010C
- #define **MAP_VERSION_OFFSET** 0
- #define **MAP_BLOCK_SIZE_OFFSET** 1
- #define **MAP_BLOCKS_OFFSET** 2
- #define **MAP_CACHE_BLOCKS_OFFSET** 4
- #define **MAP_CREATE_OFFSET** 5
- #define **MAP_DELETE_OFFSET** 6
- #define **MAP_READ_OFFSET** 7
- #define **MAP_WRITE_OFFSET** 8
- #define **MAP_FREE_OFFSET** 9
- #define **MAP_FLUSH_OFFSET** 10
- #define **MAP_PHYSICAL_OFFSET** 11

5.14.1 Detailed Description

File system layer 5: Mapper layer. The mapper layer takes care of mapping between logical and physical blocks. Also, if there is a difference between read/write block size and erase page size, it is hidden by the mapper. Also caching is performed by the mapper.

With simple file systems, like MMC, where the MMC card itself takes care of mapping and erasing, caching is the only operation needed by the mapper.

Definition in file **mapper.h**.

5.14.2 Define Documentation

5.14.2.1 #define FS_MAPPER_VERSION 0x010C

Current version number. 8 MSBs contain version number, 8 LSBs size of the structure in words.

Version number history:

1. 0x010C First version

Definition at line 28 of file **mapper.h**.

5.14.2.2 #define MAP_BLOCK_SIZE_OFFSET 1

Definition at line 70 of file mapper.h.

5.14.2.3 #define MAP_BLOCKS_OFFSET 2

Definition at line 71 of file mapper.h.

5.14.2.4 #define MAP_CACHE_BLOCKS_OFFSET 4

Definition at line 72 of file mapper.h.

5.14.2.5 #define MAP_CREATE_OFFSET 5

Definition at line 73 of file mapper.h.

5.14.2.6 #define MAP_DELETE_OFFSET 6

Definition at line 74 of file mapper.h.

5.14.2.7 #define MAP_FLUSH_OFFSET 10

Definition at line 78 of file mapper.h.

5.14.2.8 #define MAP_FREE_OFFSET 9

Definition at line 77 of file mapper.h.

5.14.2.9 #define MAP_PHYSICAL_OFFSET 11

Definition at line 79 of file mapper.h.

5.14.2.10 #define MAP_READ_OFFSET 7

Definition at line 75 of file mapper.h.

5.14.2.11 #define MAP_VERSION_OFFSET 0

Definition at line 69 of file mapper.h.

5.14.2.12 #define MAP_WRITE_OFFSET 8

Definition at line 76 of file mapper.h.

5.15 mapperflash.h File Reference

```
#include <vstypes.h>
```

```
#include <mapper.h>
```


Include dependency graph for mapperflash.h:



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



Data Structures

- struct **FmfMeta**
- struct **FsMapperFlash**

Defines

- #define **FS_MAP_FLASH_PAGE_SIZE** 256
- #define **FS_MAP_FLASH_MAX_ERASE_PAGES** 256
- #define **FS_MAP_NON_FULL** 4
- #define **memcpyXY** memcpy
- #define **memcpyYX** memcpy
- #define **memcpyYY** memcpy
- #define **memsetY** memset
- #define **qsorty** qsort

Functions

- **FsMapper * FsMapFICreate** (struct **FsPhysical** *physical, **u_int16** format)
- **s_int16 FsMapFIDelete** (struct **FsMapper** *map)
- **s_int16 FsMapFIRead** (struct **FsMapper** *map, **u_int32** firstLogicalBlock, **u_int16** logicalBlocks, **u_int16** *data)
- **s_int16 FsMapFIWrite** (struct **FsMapper** *map, **u_int32** firstLogicalBlock, **u_int16** logicalBlocks, **u_int16** *data)
- **s_int16 FsMapFIFlush** (struct **FsMapper** *map, **u_int16** hard)
- **s_int16 FsMapFIFree** (struct **FsMapper** *m, **u_int32** logicalBlockNo, **u_int32** logicalBlocks)
- void **FsMapFIDump** (struct **FsMapper** *map, **s_int32** maxBlocks)
- void **FsMapFICacheDump** (struct **FsMapper** *map)
- void **FsMapFIPrint** (**s_int32** page)

5.15.1 Detailed Description

File System: Flash Mapper.

5.15.2 Introduction

The Flash Mapper creates a wear-leveling buffer between a file system and a Flash memory physical layer.

Version:

1.0

Date:

2006-xx-xx

Author:

Henrik Herranen

Definition in file **mapperflash.h**.

5.15.3 Define Documentation

5.15.3.1 **#define FS_MAP_FLASH_MAX_ERASE_PAGES 256**

4 x FS_MAP_FLASH_MAX_ERASE_PAGES + 1 words of memory is required

Definition at line 21 of file mapperflash.h.

5.15.3.2 **#define FS_MAP_FLASH_PAGE_SIZE 256**

Definition at line 19 of file mapperflash.h.

5.15.3.3 **#define FS_MAP_NON_FULL 4**

Definition at line 40 of file mapperflash.h.

5.15.3.4 **#define memcpyXY memcpy**

Definition at line 97 of file mapperflash.h.

5.15.3.5 **#define memcpyYX memcpy**

Definition at line 98 of file mapperflash.h.

5.15.3.6 **#define memcpyYY memcpy**

Definition at line 99 of file mapperflash.h.

5.15.3.7 **#define memsetY memset**

Definition at line 100 of file mapperflash.h.

5.15.3.8 #define qsorty qsort

Definition at line 101 of file mapperflash.h.

5.15.4 Function Documentation

5.15.4.1 void FsMapFICacheDump (struct FsMapper * *map*)

5.15.4.2 struct FsMapper* FsMapFICreate (struct FsPhysical * *physical*, u_int16 *format*)

Create a mapper

5.15.4.3 s_int16 FsMapFIDelete (struct FsMapper * *map*)

Delete a mapper

5.15.4.4 void FsMapFIDump (struct FsMapper * *map*, s_int32 *maxBlocks*)

5.15.4.5 s_int16 FsMapFIFlush (struct FsMapper * *map*, u_int16 *hard*)

Flush all cached data. if *hard* is non-zero, all potential journals are also flushed.

5.15.4.6 s_int16 FsMapFIFree (struct FsMapper * *m*, u_int32 *logicalBlockNo*, u_int32 *logicalBlocks*)

Free blocks.

5.15.4.7 void FsMapFIPrint (s_int32 *page*)

5.15.4.8 s_int16 FsMapFIRead (struct FsMapper * *map*, u_int32 *firstLogicalBlock*, u_int16 *logicalBlocks*, u_int16 * *data*)

Read blocks

5.15.4.9 s_int16 FsMapFIWrite (struct FsMapper * *map*, u_int32 *firstLogicalBlock*, u_int16 *logicalBlocks*, u_int16 * *data*)

Write blocks

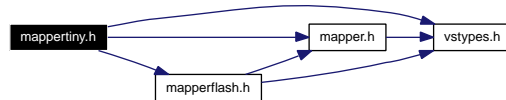
5.16 mappertiny.h File Reference

```
#include <vstypes.h>
```

```
#include <mapper.h>
```

```
#include <mapperflash.h>
```

Include dependency graph for mappertiny.h:



Data Structures

- struct **FsMapperTiny**

Functions

- **FsMapper * FsMapTnCreate** (struct **FsPhysical** *physical, u_int16 cacheSize)
- s_int16 **FsMapTnDelete** (struct **FsMapper** *map)
- s_int16 **FsMapTnRead** (struct **FsMapper** *map, u_int32 firstLogicalBlock, u_int16 logicalBlocks, u_int16 *data)
- s_int16 **FsMapTnWrite** (struct **FsMapper** *map, u_int32 firstLogicalBlock, u_int16 logicalBlocks, u_int16 *data)
- s_int16 **FsMapTnFlush** (struct **FsMapper** *map, u_int16 hard)
- s_int16 **FsMapTnFree** (struct **FsMapper** *m, u_int32 logicalBlockNo, u_int32 logicalBlocks)
- s_int16 **FsMapFNullFail** ()
- s_int16 **FsMapFNullOk** ()

5.16.1 Detailed Description

File System: Tiny Flash Mapper.

5.16.2 Introduction

The Tiny Flash Mapper is a read-only mapper that reads a wear-levelled buffers created by the Flash Mapper ([mapperflash.h](#)(p.79)). It acts as a converter tool between a file system logical and Flash physical layers.

This tiny mapper is not a very efficient implementation. For every logical block read there are max five physical read operations.

Version:

1.0

Date:

2006-08-25

Definition in file `mappertiny.h`.

5.16.3 Function Documentation

5.16.3.1 `s_int16 FsMapFInNullFail ()`

5.16.3.2 `s_int16 FsMapFInNullOk ()`

5.16.3.3 `struct FsMapper* FsMapTnCreate (struct FsPhysical * physical, u_int16 cacheSize)`

Create a tiny mapper

5.16.3.4 `s_int16 FsMapTnDelete (struct FsMapper * map)`

Delete a tiny mapper

5.16.3.5 `s_int16 FsMapTnFlush (struct FsMapper * map, u_int16 hard)`

Flush all cached data. if *hard* is non-zero, all potential journals are also flushed.

5.16.3.6 `s_int16 FsMapTnFree (struct FsMapper * m, u_int32 logicalBlockNo, u_int32 logicalBlocks)`

Free blocks.

5.16.3.7 `s_int16 FsMapTnRead (struct FsMapper * map, u_int32 firstLogicalBlock, u_int16 logicalBlocks, u_int16 * data)`

Read blocks

5.16.3.8 `s_int16 FsMapTnWrite (struct FsMapper * map, u_int32 firstLogicalBlock, u_int16 logicalBlocks, u_int16 * data)`

Write blocks

5.17 math.h File Reference

Defines

- `#define frexp __builtin_frexp`
- `#define ldexp __builtin_ldexp`
- `#define fabs __builtin_fabs`
- `#define modf(v, p) _modf(v,p)`
- `#define _oldmodf(v, p) (*(p)=(long)(v),(v)-(double)(long)(v))`

- #define **HUGE_VAL** 3.402823466385288598e38
- #define **EDOM** 33
- #define **ERANGE** 34
- #define **log2**(x) (log(x)/log(2.))
- #define **logdb**(x) (x ? (int)(log(x)/log(2.)*6.0)-96 : -96)
- #define **MulSh24**(a, b) ((long)((long long)(a) * (b) >> 24))
- #define **MulSh20**(a, b) ((long)((long long)(a) * (b) >> 20))

Functions

- `__near double` **frexp** (double value, `__near int *eptr`)
- `__near double` **ldexp** (double value, int exp)
- `__near double` **fabs** (double value)
- `__near double` **__modf** (double value, `__near double *iptr`)
- `__near double` **fmod** (double x, double y)
- `__near double` **acos** (double)
- `__near double` **asin** (double)
- `__near double` **atan** (double)
- `__near double` **cos** (double)
- `__near double` **cosh** (double)
- `__near double` **exp** (double)
- `__near double` **log10** (double)
- `__near double` **log** (double)
- `__near double` **sin** (double)
- `__near double` **sinh** (double)
- `__near double` **sqrt** (double)
- `__near double` **tan** (double)
- `__near double` **tanh** (double)
- `__near double` **floor** (double)
- `__near double` **ceil** (double)
- `__near double` **pow** (double, double)
- `__near double` **atan2** (double, double)
- `__near auto double` **SqrtF** (double)
- `__near auto unsigned short` **SqrtI** (register `__c` unsigned long x)
- `__near auto double` **SqrtF32** (double)
- `__near auto unsigned long` **SqrtI32** (register `__c` unsigned long x)
- `__near double` **ISqrt** (double x)
- `__near unsigned short` **LongLog2** (register `__a` long x)
- unsigned long **__divide16unsigned** (register `__b0` unsigned short dividend, register `__a0` unsigned short D)
- unsigned long **__divide16signed** (register `__b0` unsigned short dividend, register `__a0` unsigned short D)

5.17.1 Define Documentation

5.17.1.1 #define `_oldmodf(v, p) (*p)=(long)(v),(v)-(double)(long)(v)`

Definition at line 13 of file math.h.

5.17.1.2 #define `EDOM 33`

Definition at line 19 of file math.h.

5.17.1.3 #define `ERANGE 34`

Definition at line 20 of file math.h.

5.17.1.4 #define `fabs __builtin_fabs`

Definition at line 9 of file math.h.

5.17.1.5 #define `frexp __builtin_frexp`

Definition at line 7 of file math.h.

5.17.1.6 #define `HUGE_VAL 3.402823466385288598e38`

Definition at line 17 of file math.h.

5.17.1.7 #define `ldexp __builtin_ldexp`

Definition at line 8 of file math.h.

5.17.1.8 #define `log2(x) (log(x)/log(2.))`

Definition at line 52 of file math.h.

5.17.1.9 #define `logdb(x) (x ? (int)(log(x)/log(2.)*6.0)-96 : -96)`

Definition at line 53 of file math.h.

5.17.1.10 #define `modf(v, p) _modf(v,p)`

Definition at line 12 of file math.h.

5.17.1.11 #define `MulSh20(a, b) ((long)((long long)(a) * (b) >> 20))`

Definition at line 58 of file math.h.

5.17.1.12 `#define MulSh24(a, b) ((long)((long long)(a) * (b) >> 24))`

Definition at line 57 of file math.h.

5.17.2 Function Documentation

5.17.2.1 `unsigned long _divide16signed (register __b0 unsigned short dividend, register __a0 unsigned short D)`

5.17.2.2 `unsigned long _divide16unsigned (register __b0 unsigned short dividend, register __a0 unsigned short D)`

5.17.2.3 `__near double _modf (double value, __near double * iptr)`

5.17.2.4 `__near double acos (double)`

5.17.2.5 `__near double asin (double)`

5.17.2.6 `__near double atan (double)`

5.17.2.7 `__near double atan2 (double, double)`

5.17.2.8 `__near double ceil (double)`

5.17.2.9 `__near double cos (double)`

5.17.2.10 `__near double cosh (double)`

5.17.2.11 `__near double exp (double)`

5.17.2.12 `__near double fabs (double value)`

5.17.2.13 `__near double floor (double)`

5.17.2.14 `__near double fmod (double x, double y)`

5.17.2.15 `__near double frexp (double value, __near int * eptr)`

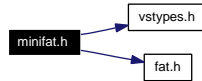
- 5.17.2.16 `__near double ISqrt (double x)`
- 5.17.2.17 `__near double ldexp (double value, int exp)`
- 5.17.2.18 `__near double log (double)`
- 5.17.2.19 `__near double log10 (double)`
- 5.17.2.20 `__near unsigned short LongLog2 (register __a long x)`
- 5.17.2.21 `__near double pow (double, double)`
- 5.17.2.22 `__near double sin (double)`
- 5.17.2.23 `__near double sinh (double)`
- 5.17.2.24 `__near double sqrt (double)`
- 5.17.2.25 `__near auto double SqrtF (double)`
- 5.17.2.26 `__near auto double SqrtF32 (double)`
- 5.17.2.27 `__near auto unsigned short SqrtI (register __c unsigned long x)`
- 5.17.2.28 `__near auto unsigned long SqrtI32 (register __c unsigned long x)`
- 5.17.2.29 `__near double tan (double)`
- 5.17.2.30 `__near double tanh (double)`

5.18 minifat.h File Reference

```
#include <vstypes.h>
```

```
#include "fat.h"
```

Include dependency graph for minifat.h:



Defines

- #define **MINIFAT_H**

Typedefs

- typedef **s_int16**(* **freeSectorCallback**)(void *private, **u_int32** sector, **u_int32** numSecs)

Functions

- auto **u_int16** **FatGetByte** (register __c0 **u_int16** n)
- auto **u_int16** **FatGetWord** (register __c0 **u_int16** n)
- auto **u_int32** **FatGetLong** (register __c0 **u_int16** n)
- auto **u_int16** **FatInitFileSystem** (void)
- auto __y struct **FRAGMENT** * **FatFragmentList** (register __i2 __y struct **FRAGMENT** *frag, register __reg_b **u_int32** fatCluster)
- auto **s_int16** **FatHandleDir** (register __y struct **FRAGMENT** *currentFragment, __y struct **FRAGMENT** *nextFragment)
- auto **s_int16** **FatOpenFile** (register __c0 **u_int16** fileNum)
- auto **s_int16** **FatReadFile** (register __i3 **u_int16** *buf, register __c1 **s_int16** byteOff, register __c0 **s_int16** byteSize)
- **u_int32** **FatTell** (void)
- **u_int32** **FatSeek** (register __reg_a **u_int32** pos)
- auto **u_int32** **FatFindSector** (register __reg_d **u_int32** pos)
- auto **s_int16** **FatCheckFileType** (register __reg_a **u_int32** suffix)
- void **MemCopyPackedBigEndian** (register __i0 **u_int16** *dst, register __a0 **u_int16** dstidx, register __i1 **u_int16** *src, register __a1 **u_int16** srcidx, register __b0 **u_int16** byteSize)
- void **MemCopyPackedLittleEndian** (register __i0 **u_int16** *dst, register __a0 **u_int16** dstidx, register __i1 **u_int16** *src, register __a1 **u_int16** srcidx, register __b0 **u_int16** byteSize)
- void **MemWritePacked** (register __a0 void *dst, register __a1 **u_int16** dstidx, register __b0 **u_int16** dat)
- **u_int16** **MemReadPacked** (register __a0 const void *src, register __a1 **u_int16** srcidx)
- void **MemWritePackedY** (register __a0 __y void *dst, register __a1 **u_int16** dstidx, register __b0 **u_int16** dat)
- **u_int16** **MemReadPackedY** (register __a0 __y const void *src, register __a1 **u_int16** srcidx)
- **s_int16** **FatIterateOverFreeSectors** (**freeSectorCallback** callBackFunction, void *private)

- auto **u_int16 ReadDiskSector** (register __i0 **u_int16** *buffer, register __reg_a **u_int32** sector)

5.18.1 Detailed Description

MiniFAT contains functions to handle one simultaneous open file.

Definition in file **minifat.h**.

5.18.2 Define Documentation

5.18.2.1 #define MINIFAT_H

Definition at line 7 of file minifat.h.

5.18.3 Typedef Documentation

5.18.3.1 typedef s_int16(* freeSectorCallback)(void *private, u_int32 sector, u_int32 numSecs)

Definition at line 126 of file minifat.h.

5.18.4 Function Documentation

5.18.4.1 auto s_int16 FatCheckFileType (register __reg_a u_int32 *suffix*)

Internal function to compare the 24-bit parameter to allowed suffixes.

Parameters:

suffix The suffix to compare.

Returns:

non-zero if suffix matches, or minifatInfo.supportedSuffixes is NULL.

5.18.4.2 auto u_int32 FatFindSector (register __reg_d u_int32 *pos*)

Internal function to locate the right sector for the current read position.

Parameters:

pos Byte position to find.

Returns:

the FAT sector that corresponds to the pos.

5.18.4.3 `auto __y struct FRAGMENT* FatFragmentList (register __i2 __y struct FRAGMENT * frag, register __reg_b u_int32 fatCluster)`

Creates a list of fragments in a file, starting from a specified FAT cluster.

Parameters:

frag fragment array entry to start filling information from.

fatCluster fat cluster to start from.

Returns:

the next free fragment array entry.

5.18.4.4 `auto u_int16 FatGetByte (register __c0 u_int16 n)`

Reads byte values from minifatBuffer.

Parameters:

n byte offset of the value inside minifatBuffer

Returns:

byte value

5.18.4.5 `auto u_int32 FatGetLong (register __c0 u_int16 n)`

Reads 32-bit long values from minifatBuffer.

Parameters:

n byte offset of the little-endian value inside minifatBuffer

Returns:

long value

5.18.4.6 `auto u_int16 FatGetWord (register __c0 u_int16 n)`

Reads 16-bit word values from minifatBuffer.

Parameters:

n byte offset of the little-endian value inside minifatBuffer

Returns:

word value

5.18.4.7 auto s_int16 FatHandleDir (register __y struct FRAGMENT * *curFragment*, __y struct FRAGMENT * *nextFragment*)

Internal function that scans files until the right file is found. Uses minifatInfo.gFileNum[0] and minifatInfo.gFileNum[1] for file number counts.

Parameters:

curFragment The clusters in the directory to scan.

nextFragment The first free fragment table entry, used for recursion.

Returns:

< 0 for file found, otherwise the total number of files (can be 0).

5.18.4.8 auto u_int16 FatInitFileSystem (void)

Initializes the file system and checks if FAT present.

Returns:

0 for success.

5.18.4.9 s_int16 FatIterateOverFreeSectors (freeSectorCallback *callBackFunction*, void * *private*)

Finds free areas from the FAT disk and executes a callback function.

Parameters:

callBackFunction Is called for each free area. If callback returns non-zero, the iteration is ended prematurely.

private a private parameter to be passed to the callBackFunction.

5.18.4.10 auto s_int16 FatOpenFile (register __c0 u_int16 *fileNum*)

Opens the specified file for reading. Only counts files that match a suffix set in the array set to minifatInfo.supportedSuffixes, or all files if minifatInfo.supportedSuffixes is NULL.

Parameters:

fileNum The number of file to open, starting from 0.

Returns:

< 0 for file found, otherwise the total number of files (can be 0).

5.18.4.11 auto s_int16 FatReadFile (register __i3 u_int16 * *buf*, register __c1 s_int16 *byteOff*, register __c0 s_int16 *byteSize*)

5.18.4.12 u_int32 FatSeek (register __reg_a u_int32 pos)

Changes the current read byte position of the file.

Parameters:

pos Absolute position for the new read byte position.

Returns:

The old read position.

5.18.4.13 u_int32 FatTell (void)

Returns the current read byte position of the file.

Returns:

Current read position.

5.18.4.14 void MemCopyPackedBigEndian (register __i0 u_int16 * dst, register __a0 u_int16 dstidx, register __i1 u_int16 * src, register __a1 u_int16 srcidx, register __b0 u_int16 byteSize)

Copies big-endian packed byte strings with arbitrary alignments (in X memory).

Parameters:

dst destination word pointer

dstidx destination byte offset

src source word pointer

srcidx source byte offset

byteSize the number of bytes to copy

5.18.4.15 void MemCopyPackedLittleEndian (register __i0 u_int16 * dst, register __a0 u_int16 dstidx, register __i1 u_int16 * src, register __a1 u_int16 srcidx, register __b0 u_int16 byteSize)

Not in VS1000B ROM!

5.18.4.16 u_int16 MemReadPacked (register __a0 const void * src, register __a1 u_int16 srcidx)

Not in VS1000B ROM!

5.18.4.17 u_int16 MemReadPackedY (register __a0 __y const void * src, register __a1 u_int16 srcidx)

Not in VS1000B ROM!

5.18.4.18 void MemWritePacked (register __a0 void * *dst*, register __a1 u_int16 *dstidx*, register __b0 u_int16 *dat*)

Not in VS1000B ROM!

5.18.4.19 void MemWritePackedY (register __a0 __y void * *dst*, register __a1 u_int16 *dstidx*, register __b0 u_int16 *dat*)

Writes bytes to big-endian packed byte array in Y memory.

Parameters:

- dst* destination word pointer
- dstidx* destination byte offset
- dat* byte to write, the value should be 0..255

5.18.4.20 auto u_int16 ReadDiskSector (register __i0 u_int16 * *buffer*, register __reg_a u_int32 *sector*)

Outside service that must be provided for minifat. VS1000 provides this function through IRAM hook ReadDiskSector. The default value for it is **Mapper-ReadDiskSector()**(p. 176), which uses map->Read() to implement the sector read.

Parameters:

- buffer* the buffer for the sector data.
- sector* the sector to read.

5.19 physical.h File Reference

```
#include <vstypes.h>
```

Include dependency graph for physical.h:



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



Data Structures

- struct FsPhysical

Defines

- #define FS_PHYSICAL_VERSION 0x010B

5.19.1 Detailed Description

File system layer 6: Physical layer. This is the lowest layer of a file system. This layer takes care of the actual, physical connection to a device.

Definition in file `physical.h`.

5.19.2 Define Documentation

5.19.2.1 `#define FS_PHYSICAL_VERSION 0x010B`

Current version number. 8 MSBs contain version number, 8 LSBs size of the structure in words.

Version number history:

1. 0x010B

Definition at line 21 of file `physical.h`.

5.20 `player.h` File Reference

```
#include <vstypes.h>
```

Include dependency graph for `player.h`:



Data Structures

- struct `Player`
- struct `KeyMapping`

Defines

- `#define LED1 4`
- `#define LED2 8`
- `#define KEY_1 1`
- `#define KEY_2 2`
- `#define KEY_3 4`
- `#define KEY_4 8`
- `#define KEY_5 16`
- `#define KEY_POWER 256`
- `#define KEY_RELEASED 0x4000`
- `#define KEY_LONG_PRESS 0x8000`
- `#define KEY_LONG_ONESHOT 0x8000`
- `#define SHORT_LIMIT 16`
- `#define OFF_LIMIT 32`

Enumerations

- enum `keyEvent` {
`ke_null = 0, ke_previous, ke_next, ke_rewind,`
`ke_forward, ke_volumeUp, ke_volumeDown, ke_earSpeaker,`
`ke_earSpeakerToggle, ke_randomToggle, ke_randomToggle-`
`NewSong, ke_pauseToggle,`
`ke_powerOff, ke_ff_faster, ke_ff_slower, ke_ff_off,`
`ke_volumeUp2, ke_volumeDown2 }`

Functions

- void `putch` (register `__a0` short dat)
- void `KeyScan` (void)
- auto void `CleanDisk` (register `__c1` `u_int16` tryBoot)
- void `PlayerVolume` (void)
- auto `u_int16` `USBIsAttached` (void)
- auto void `MassStorage` (void)
- auto void `RealMassStorage` (void)
- void `KeyEventHandler` (enum `keyEvent` event)
- void `RealKeyEventHandler` (enum `keyEvent` event)
- auto `u_int16` `ReadGPIO` (void)
- void `USBSuspend` (`u_int16` timeOut)
- void `RealUSBSuspend` (`u_int16` timeOut)
- void `UserInterfaceIdleHook` (void)
- `u_int16` `CsRead` (struct `CodecServices` *cs, `u_int16` *data, `u_int16` firstOdd, `u_int16` bytes)
- `s_int16` `CsSeek` (struct `CodecServices` *cs, `s_int32` offset, `s_int16` whence)
- `s_int16` `CsOutput` (struct `CodecServices` *cs, `s_int16` *data, `s_int16` n)

Variables

- `s_int16` `tmpBuf` [2 *32]
- `Player` `player`
- const struct `KeyMapping` * `currentKeyMap`
- const struct `KeyMapping` `sixKeyMap` []
- const struct `KeyMapping` `fiveKeyMap` []
- const struct `KeyMapping` `shiftFourKeyMap` []
- const struct `KeyMapping` `threeKeyMap` []
- `u_int16` `keyOld`
- `s_int16` `keyOldTime`
- `__y` `u_int16` `mallocAreaY` []
- `u_int16` `mallocAreaX` []

- `const u_int32 * supportedFiles`
- `const u_int32 defSupportedFiles []`
- `u_int16 keyCheck`
- `__y u_int16 vs1000d_BitReverse [256]`
- `__y u_int16 vs1000d_Latin1 [256 *3]`

5.20.1 Detailed Description

Routines related to the default player implementation.

Definition in file `player.h`.

5.20.2 Define Documentation

5.20.2.1 `#define KEY_1 1`

Definition at line 63 of file `player.h`.

5.20.2.2 `#define KEY_2 2`

Definition at line 64 of file `player.h`.

5.20.2.3 `#define KEY_3 4`

Definition at line 65 of file `player.h`.

5.20.2.4 `#define KEY_4 8`

Definition at line 66 of file `player.h`.

5.20.2.5 `#define KEY_5 16`

Definition at line 67 of file `player.h`.

5.20.2.6 `#define KEY_LONG_ONESHOT 0x8000`

Definition at line 71 of file `player.h`.

5.20.2.7 `#define KEY_LONG_PRESS 0x8000`

Definition at line 70 of file `player.h`.

5.20.2.8 `#define KEY_POWER 256`

Definition at line 68 of file `player.h`.

5.20.2.9 `#define KEY_RELEASED 0x4000`

Definition at line 69 of file `player.h`.

5.20.2.10 `#define LED1 4`

Definition at line 28 of file `player.h`.

5.20.2.11 `#define LED2 8`

Definition at line 29 of file `player.h`.

5.20.2.12 `#define OFF_LIMIT 32`

Definition at line 76 of file `player.h`.

5.20.2.13 `#define SHORT_LIMIT 16`

Definition at line 75 of file `player.h`.

5.20.3 Enumeration Type Documentation**5.20.3.1** `enum keyEvent`

Actions for keypresses.

Enumerator:

ke_null

ke_previous skip to previous song or start of song (if ≥ 5 seconds played)

ke_next goto next song

ke_rewind skip 5 seconds back

ke_forward skip 5 seconds forward

ke_volumeUp increase volume by 0.5dB

ke_volumeDown decrease volume by 0.5dB

ke_earSpeaker rotate earSpeaker setting: 0, 16000U, 38000U, 54000U

ke_earSpeakerToggle toggle earSpeaker setting: 0, 38000U

ke_randomToggle toggle random play mode

ke_randomToggleNewSong toggle random play mode, start new song immediately when activated

ke_pauseToggle toggle pause mode

ke_powerOff power off the unit

ke_ff_faster increase play speed (needs *ke_ff_off* as release event)

ke_ff_slower decrease play speed (needs *ke_ff_off* as release event)

ke_ff_off back to normal play speed

ke_volumeUp2 increase volume by 1.0dB

ke_volumeDown2 decrease volume by 1.0dB

Definition at line 32 of file `player.h`.

```

32     {
33     ke_null = 0,
34     ke_previous,
35     ke_next,
36     ke_rewind,
37     ke_forward,
38     ke_volumeUp,
39     ke_volumeDown,
40     ke_earSpeaker,
41     ke_earSpeakerToggle,
42     ke_randomToggle,
43     ke_randomToggleNewSong,
44     ke_pauseToggle,
45     ke_powerOff,
46     ke_ff_faster,
47     ke_ff_slower,
48     ke_ff_off,
49     ke_volumeUp2,
50     ke_volumeDown2,
51 };

```

5.20.4 Function Documentation

5.20.4.1 auto void CleanDisk (register __c1 u_int16 *tryBoot*)

Cleans unused sectors from FAT16/FAT32 disks. If *tryBoot* is non-zero, checks for VS1000_B.RUN.

5.20.4.2 s_int16 CsOutput (struct CodecServices * *cs*, s_int16 * *data*, s_int16 *n*)

Implements codec service Output function. Implements pause mode (*player.pauseOn*) by inserting zero samples, and low-power pause mode (**USB-Suspend()**(p. 101)) between key presses. Updates audio sample rate. Converts mono input to stereo for **AudioOutputSamples()**(p. 37).

5.20.4.3 u_int16 CsRead (struct CodecServices * *cs*, u_int16 * *data*, u_int16 *firstOdd*, u_int16 *bytes*)

Implements codec service Read function. Uses the **Read()** hook function.

5.20.4.4 s_int16 CsSeek (struct CodecServices * *cs*, s_int32 *offset*, s_int16 *whence*)

Implements codec service Seek function. Uses **Seek()**(p. 177) and **Tell()**(p. 178) hook functions.

5.20.4.5 void KeyEventHandler (enum keyEvent *event*)

Hook: performs actions for key events. Default: **RealKeyEventHandler**.

5.20.4.6 void KeyScan (void)

Performs key scanning of 5 keys on GPIO0 and the power key. Processes the results to generate short and long key press events, then calls the `KeyEventHandler` with the appropriate key events. Short press is generated if press was shorter than 16 ui ticks (1 second). Long presses are generated when press lasts longer than 16 ui ticks. Long press generates events 16 times a second unless the `KEY_LONG_ONESHOT` is added. Any key combination containing power key requires longer press (2 seconds).

`KeyScan` also cancels play and resets pause mode if USB is attached.

5.20.4.7 `auto void MassStorage (void)`

Hook: Implements `MassStorage`. Default: `RealMassStorage`.

5.20.4.8 `void PlayerVolume (void)`

Updates volume using `player.volume` and `player.volumeOffset`.

5.20.4.9 `void putch (register __a0 short dat)`

Waits for UART TX ready, then writes next byte.

5.20.4.10 `auto u_int16 ReadGPIO (void)`

Read GPIO0[0:7] values. Turns the bits to GPIO mode and restores the state afterwards.

5.20.4.11 `void RealKeyEventHandler (enum keyEvent event)`

Performs actions for key events.

5.20.4.12 `auto void RealMassStorage (void)`

Implements `MassStorage`.

5.20.4.13 `void RealUSBSuspend (u_int16 timeOut)`

Activates low-power mode. Wakes up if changes in USB pin or GPIO0[0:4] pins are detected. Restores USB voltages.

Parameters:

timeOut 0, or calls `PowerOff()`(p. 176) when timeout reached.

5.20.4.14 `auto u_int16 USBIsAttached (void)`

Non-zero if USBN is low for at least two cycles in 20000 cycles. Note: this routine may be too slow in some applications, so you can check `(PERIP(SCI_ - STATUS)(p. 167) & SCISTF_USB_DN) == 0` directly.

5.20.4.15 `void USBSuspend (u_int16 timeOut)`

Hook: Activate low-power mode. Default: RealUSBsuspend.

Parameters:

timeOut 0, or calls `PowerOff()`(p.176) when timeout reached.

5.20.4.16 `void UserInterfaceIdleHook (void)`

The default idle hook function that scans keys 16 timer per second.

5.20.5 Variable Documentation**5.20.5.1** `const struct KeyMapping* currentKeyMap`**5.20.5.2** `const u_int32 defSupportedFiles[]`**5.20.5.3** `const struct KeyMapping fiveKeyMap[]`**5.20.5.4** `u_int16 keyCheck`**5.20.5.5** `u_int16 keyOld`**5.20.5.6** `s_int16 keyOldTime`**5.20.5.7** `u_int16 mallocAreaX[]`**5.20.5.8** `__y u_int16 mallocAreaY[]`**5.20.5.9** `struct Player player`

Structure used by the default firmwares play loop.

5.20.5.10 `const struct KeyMapping shiftFourKeyMap[]`**5.20.5.11** `const struct KeyMapping sixKeyMap[]`**5.20.5.12** `const u_int32* supportedFiles`**5.20.5.13** `const struct KeyMapping threeKeyMap[]`**5.20.5.14** `s_int16 tmpBuf[2 *32]`

5.20.5.15 `__y u_int16 vs1000d_BitReverse[256]`

Helper array to quickly reverse bits in a byte.

5.20.5.16 `__y u_int16 vs1000d_Latin1[256 *3]`

Latin1-compatible font stored in Vertical format, LSB is the top pixel. 3 words (6 bytes) per character, high 8 bits first, add one empty line as space between characters. Contains block characters and special symbols in codes 0..31 and 128..159.

5.21 romfont.h File Reference**Defines**

- `#define SYMBOL_SPACE 32`
- `#define SYMBOL_NUMBER 48`
- `#define SYMBOL_ALPHA 65`
- `#define SYMBOL_ALPHA_LOW 97`
- `#define SYMBOL_LEFTSPK 128`
- `#define SYMBOL_RIGHTSPK 129`
- `#define SYMBOL_USB 130`
- `#define SYMBOL_DISKDRIVE 131`
- `#define SYMBOL_PU 132`
- `#define SYMBOL_KATAKANA 133`
- `#define SYMBOL_DIGIT 197`
- `#define SYMBOL_NOTE 207`
- `#define SYMBOL_PLAY 208`
- `#define SYMBOL_PAUSE 209`
- `#define SYMBOL_STOP 210`

Variables

- `const u_int16 fontPtrs []`
- `const u_int16 fontData []`

5.21.1 Define Documentation**5.21.1.1** `#define SYMBOL_ALPHA 65`

Definition at line 9 of file romfont.h.

5.21.1.2 `#define SYMBOL_ALPHA_LOW 97`

Definition at line 10 of file romfont.h.

5.21.1.3 #define SYMBOL_DIGIT 197

Definition at line 17 of file romfont.h.

5.21.1.4 #define SYMBOL_DISKDRIVE 131

Definition at line 14 of file romfont.h.

5.21.1.5 #define SYMBOL_KATAKANA 133

Definition at line 16 of file romfont.h.

5.21.1.6 #define SYMBOL_LEFTSPK 128

Definition at line 11 of file romfont.h.

5.21.1.7 #define SYMBOL_NOTE 207

Definition at line 18 of file romfont.h.

5.21.1.8 #define SYMBOL_NUMBER 48

Definition at line 8 of file romfont.h.

5.21.1.9 #define SYMBOL_PAUSE 209

Definition at line 20 of file romfont.h.

5.21.1.10 #define SYMBOL_PLAY 208

Definition at line 19 of file romfont.h.

5.21.1.11 #define SYMBOL_PU 132

Definition at line 15 of file romfont.h.

5.21.1.12 #define SYMBOL_RIGHTSPK 129

Definition at line 12 of file romfont.h.

5.21.1.13 #define SYMBOL_SPACE 32

Definition at line 7 of file romfont.h.

5.21.1.14 #define SYMBOL_STOP 210

Definition at line 21 of file romfont.h.

5.21.1.15 `#define SYMBOL_USB 130`

Definition at line 13 of file romfont.h.

5.21.2 Variable Documentation**5.21.2.1** `const u_int16 fontData[]`**5.21.2.2** `const u_int16 fontPtrs[]`**5.22** scsi.h File Reference

Common SCSI definitions.

Data Structures

- struct `scsicdb6variant1`
- struct `scsicdb6variant2`
- struct `scsicdb6variant3`
- struct `scsicdb10variant1`
- struct `scsicdb10variant2`

Defines

- `#define OPERATION_CODE 0`
- `#define SCSI_INQUIRY 0x12`
- `#define SCSI_FORMAT_UNIT 0x04`
- `#define SCSI_READ_6 0x08`
- `#define SCSI_READ_10 0x28`
- `#define SCSI_READ_12 0xa8`
- `#define ATAPI_READ_FORMAT_CAPACITIES 0x23`
- `#define SCSI_READ_CAPACITY_10 0x25`
- `#define SCSI_READ_CAPACITY_16 0x9e`
- `#define SCSI_READ_CAPACITY_16_2 0x10`
- `#define SCSI_RECEIVE_DIAGNOSTIC_RESULTS 0x1c`
- `#define SCSI_REPORT_LUNS 0xA0`
- `#define SCSI_REQUEST_SENSE 0x03`
- `#define SCSI_SEND_DIAGNOSTIC 0x1d`
- `#define SCSI_TEST_UNIT_READY 0x00`
- `#define SCSI_WRITE_6 0x0a`
- `#define SCSI_WRITE_10 0x2a`
- `#define SCSI_WRITE_12 0xaa`
- `#define SCSI_MODE_SENSE_6 0x1a`
- `#define SCSI_MODE_SENSE_10 0x5a`
- `#define SCSI_SYNCHRONIZE_CACHE 0x35`

- #define **SCSI_PREVENT_ALLOW_MEDIUM_REMOVAL** 0x1e
- #define **SCSI_VERIFY** 0x2f
- #define **SCSI_START_STOP_UNIT** 0x1b
- #define **SCSI_MODE_SELECT** 0x15
- #define **SK_NO_SENSE** 0
- #define **SK_RECOVERED_ERROR** 1
- #define **SK_NOT_READY** 2
- #define **SK_MEDIUM_ERROR** 3
- #define **SK_HARDWARE_ERROR** 4
- #define **SK_ILLEGAL_REQUEST** 5
- #define **SK_UNIT_ATTENTION** 6
- #define **SK_DATA_PROTECT** 7
- #define **SK_BLANK_CHECK** 8
- #define **SK_VENDOR_SPECIFIC** 9
- #define **SK_COPY_ABORTED** 10
- #define **SK_ABORTED_COMMAND** 11
- #define **SK_EQUAL** 12
- #define **SK_VOLUME_OVERFLOW** 13
- #define **SK_MISCOMPARE** 14

Typedefs

- typedef scsicdb6variant1 ScsiInquiryCdb
- typedef scsicdb6variant2 ScsiModeSense6Cdb
- typedef scsicdb6variant3 ScsiRequestSenseCdb
- typedef scsicdb10variant1 ScsiRead10Cdb
- typedef scsicdb10variant2 ScsiWrite10Cdb

Enumerations

- enum **SCSIStageEnum** {
SCSI_UNINITIALIZED = -1, **SCSI_READY_FOR_COMMAND** = 0, **SCSI_DATA_TO_HOST**, **SCSI_TRANSMITTING**,
SCSI_DATA_FROM_HOST, **SCSI_SEND_STATUS**, **SCSI_INVALID_CBW** }
- enum **SCSIStatusEnum** { **SCSI_OK** = 0, **SCSI_REQUEST_ERROR** = 1, **SCSI_PHASE_ERROR** = 2 }

Functions

- void **ScsiTaskHandler** (void)
- void **RealScsiTaskHandler** (void)
- void **DiskProtocolCommand** (u_int16 *cmd)
- void **ScsiReset** ()
- void **DiskDataReceived** (int length, u_int16 *datablock)
- enum **SCSIStageEnum** **ScsiState** (void)
- u_int16 **ScsiOrBlock** (register __i0 u_int16 *buffer, register __a0 s_int16 size)

5.22.1 Detailed Description

Common SCSI definitions.

Definition in file `scsi.h`.

5.22.2 Define Documentation

5.22.2.1 #define ATAPI_READ_FORMAT_CAPACITIES 0x23

Definition at line 88 of file `scsi.h`.

5.22.2.2 #define OPERATION_CODE 0

Definition at line 79 of file `scsi.h`.

5.22.2.3 #define SCSI_FORMAT_UNIT 0x04

Definition at line 84 of file `scsi.h`.

5.22.2.4 #define SCSI_INQUIRY 0x12

Definition at line 83 of file `scsi.h`.

5.22.2.5 #define SCSI_MODE_SELECT 0x15

Definition at line 106 of file `scsi.h`.

5.22.2.6 #define SCSI_MODE_SENSE_10 0x5a

Definition at line 101 of file `scsi.h`.

5.22.2.7 #define SCSI_MODE_SENSE_6 0x1a

Definition at line 100 of file `scsi.h`.

5.22.2.8 `#define SCSI_PREVENT_ALLOW_MEDIUM_REMOVAL 0x1e`

Definition at line 103 of file scsi.h.

5.22.2.9 `#define SCSI_READ_10 0x28`

Definition at line 86 of file scsi.h.

5.22.2.10 `#define SCSI_READ_12 0xa8`

Definition at line 87 of file scsi.h.

5.22.2.11 `#define SCSI_READ_6 0x08`

Definition at line 85 of file scsi.h.

5.22.2.12 `#define SCSI_READ_CAPACITY_10 0x25`

Definition at line 89 of file scsi.h.

5.22.2.13 `#define SCSI_READ_CAPACITY_16 0x9e`

Definition at line 90 of file scsi.h.

5.22.2.14 `#define SCSI_READ_CAPACITY_16_2 0x10`

Definition at line 91 of file scsi.h.

5.22.2.15 `#define SCSI_RECEIVE_DIAGNOSTIC_RESULTS 0x1c`

Definition at line 92 of file scsi.h.

5.22.2.16 `#define SCSI_REPORT_LUNS 0xA0`

Definition at line 93 of file scsi.h.

5.22.2.17 `#define SCSI_REQUEST_SENSE 0x03`

Definition at line 94 of file scsi.h.

5.22.2.18 `#define SCSI_SEND_DIAGNOSTIC 0x1d`

Definition at line 95 of file scsi.h.

5.22.2.19 `#define SCSI_START_STOP_UNIT 0x1b`

Definition at line 105 of file scsi.h.

5.22.2.20 `#define SCSI_SYNCHRONIZE_CACHE 0x35`

Definition at line 102 of file scsi.h.

5.22.2.21 `#define SCSI_TEST_UNIT_READY 0x00`

Definition at line 96 of file scsi.h.

5.22.2.22 `#define SCSI_VERIFY 0x2f`

Definition at line 104 of file scsi.h.

5.22.2.23 `#define SCSI_WRITE_10 0x2a`

Definition at line 98 of file scsi.h.

5.22.2.24 `#define SCSI_WRITE_12 0xaa`

Definition at line 99 of file scsi.h.

5.22.2.25 `#define SCSI_WRITE_6 0x0a`

Definition at line 97 of file scsi.h.

5.22.2.26 `#define SK_ABORTED_COMMAND 11`

Definition at line 120 of file scsi.h.

5.22.2.27 `#define SK_BLANK_CHECK 8`

Definition at line 117 of file scsi.h.

5.22.2.28 `#define SK_COPY_ABORTED 10`

Definition at line 119 of file scsi.h.

5.22.2.29 `#define SK_DATA_PROTECT 7`

Definition at line 116 of file scsi.h.

5.22.2.30 `#define SK_EQUAL 12`

Definition at line 121 of file scsi.h.

5.22.2.31 `#define SK_HARDWARE_ERROR 4`

Definition at line 113 of file scsi.h.

5.22.2.32 #define SK_ILLEGAL_REQUEST 5

Definition at line 114 of file scsi.h.

5.22.2.33 #define SK_MEDIUM_ERROR 3

Definition at line 112 of file scsi.h.

5.22.2.34 #define SK_MISCOMPARE 14

Definition at line 123 of file scsi.h.

5.22.2.35 #define SK_NO_SENSE 0

Definition at line 109 of file scsi.h.

5.22.2.36 #define SK_NOT_READY 2

Definition at line 111 of file scsi.h.

5.22.2.37 #define SK_RECOVERED_ERROR 1

Definition at line 110 of file scsi.h.

5.22.2.38 #define SK_UNIT_ATTENTION 6

Definition at line 115 of file scsi.h.

5.22.2.39 #define SK_VENDOR_SPECIFIC 9

Definition at line 118 of file scsi.h.

5.22.2.40 #define SK_VOLUME_OVERFLOW 13

Definition at line 122 of file scsi.h.

5.22.3 Typedef Documentation

5.22.3.1 typedef struct scsicdb6variant1 ScsiInquiryCdb

5.22.3.2 typedef struct scsicdb6variant2 ScsiModeSense6Cdb

5.22.3.3 typedef struct scsicdb10variant1 ScsiRead10Cdb

5.22.3.4 typedef struct scsicdb6variant3 ScsiRequestSenseCdb

5.22.3.5 typedef struct scsicdb10variant2 ScsiWrite10Cdb

5.22.4 Enumeration Type Documentation

5.22.4.1 enum SCISStageEnum

Enumerator:

```
SCSI_UNINITIALIZED  
SCSI_READY_FOR_COMMAND  
SCSI_DATA_TO_HOST  
SCSI_TRANSMITTING  
SCSI_DATA_FROM_HOST  
SCSI_SEND_STATUS  
SCSI_INVALID_CBW
```

Definition at line 6 of file scsi.h.

```
6     {  
7     SCSI_UNINITIALIZED = -1,  
8     SCSI_READY_FOR_COMMAND = 0,  
9     SCSI_DATA_TO_HOST,  
10    SCSI_TRANSMITTING,  
11    SCSI_DATA_FROM_HOST,  
12    SCSI_SEND_STATUS,  
13    SCSI_INVALID_CBW  
14 } SCISStageEnum;
```

5.22.4.2 enum SCISStatusEnum

Enumerator:

```
SCSI_OK  
SCSI_REQUEST_ERROR  
SCSI_PHASE_ERROR
```

Definition at line 16 of file scsi.h.

```
16     {  
17     SCSI_OK = 0,  
18     SCSI_REQUEST_ERROR = 1,  
19     SCSI_PHASE_ERROR = 2  
20 } SCISStatusEnum;
```

5.22.5 Function Documentation

5.22.5.1 void DiskDataReceived (int *length*, u_int16 * *datablock*)

Process data from host.

5.22.5.2 void DiskProtocolCommand (u_int16 * cmd)

Process a SCSI command block.

5.22.5.3 void RealScsiTaskHandler (void)

RealScsiTaskHandler

5.22.5.4 u_int16 ScsiOrBlock (register __i0 u_int16 * buffer, register __a0 s_int16 size)**5.22.5.5 void ScsiReset ()**

Reset SCSI state.

5.22.5.6 enum SCSIStageEnum ScsiState (void)**5.22.5.7 void ScsiTaskHandler (void)**

Handle any pending SCSI operation.

ScsiTaskHandler

5.23 stdarg.h File Reference**Defines**

- #define **va_start**(ap, last) (ap = ((va_list)&(last)))
- #define **va_arg**(ap, type) ((type *) (ap -= sizeof(type)))[0]
- #define **va_end**(ap)

Typedefs

- typedef char * **va_list**

5.23.1 Define Documentation**5.23.1.1 #define va_arg(ap, type) ((type *) (ap -= sizeof(type)))[0]**

Definition at line 6 of file stdarg.h.

5.23.1.2 #define va_end(ap)

Definition at line 7 of file stdarg.h.

5.23.1.3 `#define va_start(ap, last) (ap = ((va_list)&(last)))`

Definition at line 5 of file `stdarg.h`.

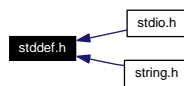
5.23.2 Typedef Documentation

5.23.2.1 `typedef char* va_list`

Definition at line 3 of file `stdarg.h`.

5.24 `stddef.h` File Reference

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



Defines

- `#define NULL 0`
- `#define offsetof(type, member) ((size_t)(&((type *)0) → member))`

Typedefs

- `typedef unsigned int size_t`
- `typedef signed int ptrdiff_t`
- `typedef short wchar_t`

5.24.1 Define Documentation

5.24.1.1 `#define NULL 0`

Definition at line 12 of file `stddef.h`.

5.24.1.2 `#define offsetof(type, member) ((size_t)(&((type *)0) → member))`

Definition at line 15 of file `stddef.h`.

5.24.2 Typedef Documentation

5.24.2.1 `typedef signed int ptrdiff_t`

Definition at line 8 of file `stddef.h`.

5.24.2.2 `typedef unsigned int size_t`

Definition at line 6 of file `stddef.h`.

5.24.2.3 `typedef short wchar_t`

Definition at line 9 of file `stddef.h`.

5.25 `stdio.h` File Reference

```
#include <stddef.h>
```

Include dependency graph for `stdio.h`:

**Data Structures**

- `struct __sfp`

Defines

- `#define EOF` (-1)
- `#define FOPEN_MAX` 20
- `#define FILENAME_MAX` 256
- `#define SEEK_SET` 0
- `#define SEEK_CUR` 1
- `#define SEEK_END` 2
- `#define stdin` (`FILE *`)1
- `#define stdout` (`FILE *`)2
- `#define stderr` (`FILE *`)3

Typedefs

- `typedef __sfp fpos_t`
- `typedef void FILE`

Functions

- `__near FILE` register `__a0` * `fopen` (`__near` const char register `__i0` *filename, `__near` const char register `__i1` *mode)
- `__near FILE` register `__a0` * `freopen` (`__near` const char register `__i0` *filename, `__near` const char register `__i1` *mode, `__near FILE` register `__a0` *stream)
- int register `__a0` `fflush` (`__near FILE` register `__a0` *stream)
- int register `__a0` `fclose` (`__near FILE` register `__a0` *stream)

- int register __a0 **remove** (__near const char register __i0 *filename)
- int register __a0 **rename** (__near const char register __i0 *oldname, __near const char register __i1 *newname)
- int register __a0 **fgetc** (__near **FILE** register __a1 *stream)
- __near char register __a0 * **fgets** (__near char register __i0 *s, int register __a0 n, __near **FILE** register __a1 *stream)
- int register __a0 **fputc** (int register __a0 c, __near **FILE** register __a1 *stream)
- int register __a0 **fputs** (__near const char register __i0 *s, __near **FILE** register __a1 *stream)
- int register __a0 **getc** (__near **FILE** register __a1 *stream)
- int register __a0 **getchar** (void)
- __near char register __a0 * **gets** (__near char register __i0 *s)
- int register __a0 **putc** (int register __a0 c, __near **FILE** register __a1 *stream)
- int register __a0 **putchar** (int register __a0 c)
- int register __a0 **puts** (__near const char register __i0 *s)
- int register __a0 **ungetc** (int register __a0 c, __near **FILE** register __a1 *stream)
- **size_t** register __a0 **fread** (__near void register __i0 *ptr, **size_t** register __b0 size, **size_t** register __b1 nobj, __near **FILE** register __a1 *stream)
- **size_t** register __a0 **fwrite** (__near const void register __i0 *ptr, **size_t** register __b0 size, **size_t** register __b1 nobj, __near **FILE** register __a1 *stream)
- int register __a0 **fseek** (__near **FILE** register __a0 *stream, long register __b offset, int register __a1 origin)
- long register __a **ftell** (__near **FILE** register __a0 *stream)
- void **rewind** (__near **FILE** register __a0 *stream)
- int register __a0 **fgetpos** (__near **FILE** register __a0 *stream, __near **fpos_t** register __i0 *ptr)
- int register __a0 **fsetpos** (__near **FILE** register __a0 *stream, __near const **fpos_t** register __i0 *ptr)
- void **clearerr** (__near **FILE** register __a0 *stream)
- int register __a0 **feof** (__near **FILE** register __a0 *stream)
- int register __a0 **ferror** (__near **FILE** register __a0 *stream)
- void **perror** (__near const char register __i0 *s)
- int **fprintf** (__near **FILE** *stream, __near const char *fmt,...)
- int **printf** (__near const char *fmt,...)
- int **sprintf** (__near char *s, __near const char *fmt,...)
- int **sscanf** (const char *str, const char *fmt,...)
- int **tinysprintf** (char *str, const char *fmt,...)
- int **tinyprintf** (const char *fmt,...)
- int **tinyprintf** (**FILE** *fp, const char *fmt,...)

5.25.1 Define Documentation

5.25.1.1 `#define EOF (-1)`

Definition at line 22 of file `stdio.h`.

5.25.1.2 `#define FILENAME_MAX 256`

Definition at line 27 of file `stdio.h`.

5.25.1.3 `#define FOPEN_MAX 20`

Definition at line 26 of file `stdio.h`.

5.25.1.4 `#define SEEK_CUR 1`

Definition at line 34 of file `stdio.h`.

5.25.1.5 `#define SEEK_END 2`

Definition at line 37 of file `stdio.h`.

5.25.1.6 `#define SEEK_SET 0`

Definition at line 31 of file `stdio.h`.

5.25.1.7 `#define stderr (FILE *)3`

Definition at line 42 of file `stdio.h`.

5.25.1.8 `#define stdin (FILE *)1`

Definition at line 40 of file `stdio.h`.

5.25.1.9 `#define stdout (FILE *)2`

Definition at line 41 of file `stdio.h`.

5.25.2 Typedef Documentation

5.25.2.1 `typedef void FILE`

Definition at line 20 of file `stdio.h`.

5.25.2.2 `typedef struct __sfp_pos_t`

5.25.3 Function Documentation

5.25.3.1 void clearerr (`__near FILE register __a0 * stream`)

5.25.3.2 int register `__a0` fclose (`__near FILE register __a0 * stream`)

5.25.3.3 int register `__a0` feof (`__near FILE register __a0 * stream`)

5.25.3.4 int register `__a0` ferror (`__near FILE register __a0 * stream`)

5.25.3.5 int register `__a0` fflush (`__near FILE register __a0 * stream`)

5.25.3.6 int register `__a0` fgetc (`__near FILE register __a1 * stream`)

5.25.3.7 int register `__a0` fgetpos (`__near FILE register __a0 * stream`, `__near fpos_t register __i0 * ptr`)

5.25.3.8 `__near char register __a0*` fgets (`__near char register __i0 * s`, int register `__a0 n`, `__near FILE register __a1 * stream`)

reads in at most one less than `n` from `stream` to buffer `s`. Reading stops at a newline or EOF. Newline is stored into the buffer. A NUL is stored after the last character in the buffer.

5.25.3.9 `__near FILE register __a0*` fopen (`__near const char register __i0 * filename`, `__near const char register __i1 * mode`)

5.25.3.10 int fprintf (`__near FILE * stream`, `__near const char * fmt`, ...)

5.25.3.11 int register `__a0` fputc (int register `__a0 c`, `__near FILE register __a1 * stream`)

writes the character `c` to `stream`.

5.25.3.12 int register __a0 fputs (__near const char register __i0 * s, __near FILE register __a1 * stream)

writes the string s to stream without the trailing NUL.

5.25.3.13 size_t register __a0 fread (__near void register __i0 * ptr, size_t register __b0 size, size_t register __b1 nobj, __near FILE register __a1 * stream)

Reads pairs of bytes from the stream and stores them into the word buffer. The first byte goes to the high bits of a word.

5.25.3.14 __near FILE register __a0* freopen (__near const char register __i0 * filename, __near const char register __i1 * mode, __near FILE register __a0 * stream)

5.25.3.15 int register __a0 fseek (__near FILE register __a0 * stream, long register __b offset, int register __a1 origin)

5.25.3.16 int register __a0 fsetpos (__near FILE register __a0 * stream, __near const fpos_t register __i0 * ptr)

5.25.3.17 long register __a ftell (__near FILE register __a0 * stream)

5.25.3.18 size_t register __a0 fwrite (__near const void register __i0 * ptr, size_t register __b0 size, size_t register __b1 nobj, __near FILE register __a1 * stream)

Writes pairs of bytes to the stream. The high bits of a word are written to the first byte.

5.25.3.19 int register __a0 getc (__near FILE register __a1 * stream)

equivalent to fgetc()(p.116), but may be macro.

5.25.3.20 int register __a0 getchar (void)

5.25.3.21 __near char register __a0* gets (__near char register __i0 * s)

5.25.3.22 void perror (__near const char register __i0 * s)

5.25.3.23 int printf (__near const char * fmt, ...)

5.25.3.24 int register __a0 putc (int register __a0 c, __near FILE register __a1 * *stream*)

5.25.3.25 int register __a0 putchar (int register __a0 c)

5.25.3.26 int register __a0 puts (__near const char register __i0 * *s*)

5.25.3.27 int register __a0 remove (__near const char register __i0 * *filename*)

5.25.3.28 int register __a0 rename (__near const char register __i0 * *oldname*, __near const char register __i1 * *newname*)

5.25.3.29 void rewind (__near FILE register __a0 * *stream*)

5.25.3.30 int sprintf (__near char * *s*, __near const char * *fmt*, ...)

5.25.3.31 int sscanf (const char * *str*, const char * *fmt*, ...)

5.25.3.32 int tinyprintf (FILE * *fp*, const char * *fmt*, ...)

5.25.3.33 int tinyprintf (const char * *fmt*, ...)

5.25.3.34 int tinsprintf (char * *str*, const char * *fmt*, ...)

5.25.3.35 int register __a0 ungetc (int register __a0 c, __near FILE register __a1 * *stream*)

5.26 stdlib.h File Reference

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



Defines

- #define **NULL** 0
- #define **RAND_MAX** 0x7fff
- #define **abort()** exit(-1)
- #define **EXIT_FAILURE** 1
- #define **EXIT_SUCCESS** 0
- #define **abs(a)** __builtin_abs(a)
- #define **labs(a)** __builtin_labs(a)
- #define **strtoul(a, b, c)** (unsigned long)strtol(a,b,c)
- #define **RANDOM_MAX** 0x7fffffffL
- #define **qsorty(b, n, s, c)** qsort((b),(n),-(s),(c))

Typedefs

- typedef unsigned int **size_t**

Functions

- int **rand** (void)
- void **srand** (register __a0 unsigned int seed)
- void **exit** (register __a0 int exitValue)
- __near int **atoi** (__near const char *s)
- __near long **strtol** (__near const char *s, __near char * __near *endp, int base)
- __near long **random** (void)
- __near void **srandom** (register __a unsigned long x)
- unsigned short **QsortLog2** (register __a0 short x)
- void **qsort** (void *base, int nmemb, int size, int(*compar)(const void *, const void *))
- short **CountBitsLong** (register __a unsigned long val)

Variables

- __y short **rand_seed**
- __near long **random_state**

5.26.1 Define Documentation**5.26.1.1 #define abort() exit(-1)**

Definition at line 19 of file stdlib.h.

5.26.1.2 #define abs(a) __builtin_abs(a)

Definition at line 24 of file stdlib.h.

5.26.1.3 #define EXIT_FAILURE 1

Definition at line 21 of file stdlib.h.

5.26.1.4 #define EXIT_SUCCESS 0

Definition at line 22 of file stdlib.h.

5.26.1.5 #define labs(a) __builtin_labs(a)

Definition at line 25 of file stdlib.h.

5.26.1.6 #define NULL 0

Definition at line 10 of file stdlib.h.

5.26.1.7 #define qsorty(b, n, s, c) qsort((b),(n),-(s),(c))

Definition at line 40 of file stdlib.h.

5.26.1.8 #define RAND_MAX 0x7fff

Definition at line 13 of file stdlib.h.

5.26.1.9 #define RANDOM_MAX 0x7fffffffL

Definition at line 31 of file stdlib.h.

5.26.1.10 #define strtoul(a, b, c) (unsigned long)strtol(a,b,c)

Definition at line 29 of file stdlib.h.

5.26.2 Typedef Documentation**5.26.2.1 typedef unsigned int size_t**

Definition at line 6 of file stdlib.h.

5.26.3 Function Documentation**5.26.3.1 __near int atoi (__near const char * s)****5.26.3.2 short CountBitsLong (register __a unsigned long val)**

Counts the number of 1-bits in a 32-bit value. Takes 12 to 84 cycles.

5.26.3.3 void exit (register __a0 int *exit Value*)

5.26.3.4 void `qsort` (void * *base*, int *nmemb*, int *size*, int(*)(*const* void *, *const* void *) *compar*)

5.26.3.5 unsigned short `QsortLog2` (register __a0 short *x*)

5.26.3.6 int `rand` (void)

5.26.3.7 __near long `random` (void)

5.26.3.8 void `srand` (register __a0 unsigned int *seed*)

5.26.3.9 __near void `srandom` (register __a unsigned long *x*)

5.26.3.10 __near long `strtol` (__near const char * *s*, __near char * __near * *endp*, int *base*)

5.26.4 Variable Documentation

5.26.4.1 __y short `rand_seed`

5.26.4.2 __near long `random_state`

5.27 string.h File Reference

```
#include <stddef.h>
```

Include dependency graph for string.h:



Defines

- #define `strcoll` `strcmp`

Functions

- __near char register __i0 * `strcpy` (__near char register __i0 **s*, __near const char register __i1 **ct*)
- __near char register __i0 * `strncpy` (__near char register __i0 **s*, __near const char register __i1 **ct*, `size_t` register __a0 *n*)
- __near char register __i0 * `strcat` (__near char register __i0 **s*, __near const char register __i1 **ct*)

- `__near char register __i0 * strncat (__near char register __i0 *s, __near const char register __i1 *ct, size_t register __a0 n)`
- `int register __a0 strcmp (__near const char register __i0 *cs, __near const char register __i1 *ct)`
- `int register __a0 strncmp (__near const char register __i0 *cs, __near const char register __i1 *ct, size_t register __a0 n)`
- `__near char register __i0 * strchr (__near const char register __i0 *cs, int register __a0 c)`
- `__near char register __i0 * strrchr (__near const char register __i0 *cs, int register __a0 c)`
- `size_t register __a0 strspn (__near const char register __i0 *cs, __near const char register __i1 *ct)`
- `size_t register __a0 strcspn (__near const char register __i0 *cs, __near const char register __i1 *ct)`
- `__near char register __i0 * strpbrk (__near const char register __i0 *cs, __near const char register __i1 *ct)`
- `__near char register __i0 * strstr (__near const char register __i0 *cs, __near const char register __i1 *ct)`
- `size_t register __a0 strlen (__near const char register __i0 *cs)`
- `__near char register __i0 * strerror (int register __a0 n)`
- `__near char register __i0 * strtok (__near char register __i0 *s, __near const char register __i1 *ct)`
- `__near void register __i0 * memcpy (__near void register __i0 *d, __near const void register __i1 *s, size_t register __a0 n)`
- `__near __y void register __i0 * memcpyXY (__near __y void register __i0 *d, __near const void register __i1 *s, size_t register __a0 n)`
- `__near void register __i0 * memcpyYX (__near void register __i0 *s, __near __y const void register __i1 *ct, size_t register __a0 n)`
- `__near __y void register __i0 * memcpyYY (__near __y void register __i0 *s, __near __y const void register __i1 *ct, size_t register __a0 n)`
- `__near void register __i0 * memmove (__near void register __i0 *s, const __near void register __i1 *ct, unsigned short register __a0 n)`
- `int register __a0 memcmp (__near const void register __i0 *cs, __near const void register __i1 *ct, size_t register __a0 n)`
- `int register __a0 memcmpY (__near const void register __i0 *cs, __near const void register __i1 *ct, size_t register __a0 n)`
- `__near void register __i0 * memchr (__near const void register __i0 *cs, int register __a0 c, size_t register __a1 n)`
- `__near void register __i0 * memset (__near void register __i0 *s, int register __a1 c, size_t register __a0 n)`
- `__near void register __i0 * memsetY (__near __y void register __i0 *s, int register __a1 c, size_t register __a0 n)`
- `__near void memclearXY (register __i0 unsigned short *p, register __a0 short c)`
- `size_t register __a0 strxfrm (__near char register __i0 *s1, __near const char register __i1 *s2, size_t register __a0 n)`

- void **memswap** (register __i0 void *a, register __i1 void *b, register __a0 size_t size)
- void **memswapy** (register __i0 __y void *a, register __i1 __y void *b, register __a0 size_t size)
- void **memswapxy** (register __i0 void *a, register __i1 __y void *b, register __a0 size_t size)
- __near void **MemCopyPackedBigEndian** (register __i0 __near unsigned short *dst, register __a0 unsigned short dstidx, register __i1 __near unsigned short *src, register __a1 unsigned short srcidx, register __b0 unsigned short byteSize)

5.27.1 Define Documentation

5.27.1.1 #define strcoll strcmp

Definition at line 7 of file string.h.

5.27.2 Function Documentation

5.27.2.1 __near void register __i0* memchr (__near const void register __i0 * cs, int register __a0 c, size_t register __a1 n)

5.27.2.2 __near void memclearXY (register __i0 unsigned short * p, register __a0 short c)

Clears c words in both X and Y memory starting from p.

5.27.2.3 int register __a0 memcmp (__near const void register __i0 * cs, __near const void register __i1 * ct, size_t register __a0 n)

5.27.2.4 int register __a0 memcmpY (__near const void register __i0 * cs, __near const void register __i1 * ct, size_t register __a0 n)

5.27.2.5 __near void MemCopyPackedBigEndian (register __i0 __near unsigned short * dst, register __a0 unsigned short dstidx, register __i1 __near unsigned short * src, register __a1 unsigned short srcidx, register __b0 unsigned short byteSize)

Copies bytes from one word buffer to another. The bytes are packed in big-endian format, i.e. the first byte to the high bits of the word and the next byte to the low bits of the word.

5.27.2.6 __near void register __i0* memcpy (__near void register __i0 * d, __near const void register __i1 * s, size_t register __a0 n)

Copies n words from s to d . The memory areas should not overlap. Returns a pointer to d .

5.27.2.7 `__near __y void register __i0* memcpyXY (__near __y void register __i0 * d, __near const void register __i1 * s, size_t register __a0 n)`

Copies n words from s to d , where the destination is in Y memory. Returns a pointer to d .

5.27.2.8 `__near void register __i0* memcpyYX (__near void register __i0 * s, __near __y const void register __i1 * ct, size_t register __a0 n)`

Copies n words from s to d , where the source is in Y memory. Returns a pointer to d .

5.27.2.9 `__near __y void register __i0* memcpyYY (__near __y void register __i0 * s, __near __y const void register __i1 * ct, size_t register __a0 n)`

Copies n words from s to d , where both the source and the destination is in Y memory. The memory areas should not overlap. Returns a pointer to d .

5.27.2.10 `__near void register __i0* memmove (__near void register __i0 * s, const __near void register __i1 * ct, unsigned short register __a0 n)`

5.27.2.11 `__near void register __i0* memset (__near void register __i0 * s, int register __a1 c, size_t register __a0 n)`

5.27.2.12 `__near void register __i0* memsetY (__near __y void register __i0 * s, int register __a1 c, size_t register __a0 n)`

5.27.2.13 `void memswap (register __i0 void * a, register __i1 void * b, register __a0 size_t size)`

Swaps two memory areas. The size gives the length of the area.

5.27.2.14 `void memswapxy (register __i0 void * a, register __i1 __y void * b, register __a0 size_t size)`

Swaps two memory areas, the second is in Y memory. The size gives the length of the area.

5.27.2.15 void memswapy (register __i0 __y void * a, register __i1 __y void * b, register __a0 size_t size)

Swaps two memory areas, both are in Y memory. The size gives the length of the area.

5.27.2.16 __near char register __i0* strcat (__near char register __i0 * s, __near const char register __i1 * ct)

5.27.2.17 __near char register __i0* strchr (__near const char register __i0 * cs, int register __a0 c)

5.27.2.18 int register __a0 strcmp (__near const char register __i0 * cs, __near const char register __i1 * ct)

5.27.2.19 __near char register __i0* strcpy (__near char register __i0 * s, __near const char register __i1 * ct)

5.27.2.20 size_t register __a0 strcspn (__near const char register __i0 * cs, __near const char register __i1 * ct)

5.27.2.21 __near char register __i0* strerror (int register __a0 n)

5.27.2.22 size_t register __a0 strlen (__near const char register __i0 * cs)

5.27.2.23 __near char register __i0* strncat (__near char register __i0 * s, __near const char register __i1 * ct, size_t register __a0 n)

5.27.2.24 int register __a0 strncmp (__near const char register __i0 * cs, __near const char register __i1 * ct, size_t register __a0 n)

5.27.2.25 __near char register __i0* strncpy (__near char register __i0 * s, __near const char register __i1 * ct, size_t register __a0 n)

5.27.2.26 __near char register __i0* strpbrk (__near const char register __i0 * cs, __near const char register __i1 * ct)

5.27.2.27 `__near char register __i0* strrchr (__near const char register __i0 * cs, int register __a0 c)`

5.27.2.28 `size_t register __a0 strspn (__near const char register __i0 * cs, __near const char register __i1 * ct)`

5.27.2.29 `__near char register __i0* strstr (__near const char register __i0 * cs, __near const char register __i1 * ct)`

5.27.2.30 `__near char register __i0* strtok (__near char register __i0 * s, __near const char register __i1 * ct)`

5.27.2.31 `size_t register __a0 strxfrm (__near char register __i0 * s1, __near const char register __i1 * s2, size_t register __a0 n)`

5.28 usb.h File Reference

Defines

- `#define USB_SET_ADDRESS_ENA 0xD0`
- `#define USB_SET_ENDP_ENA 0xD8`
- `#define USB_SET_MODE 0xF3`
- `#define USB_SET_DMA 0xFB`
- `#define USB_READ_INT 0xF4`
- `#define USB_SELECT_ENDP_CO 0x00`
- `#define USB_SELECT_ENDP_CI 0x01`
- `#define USB_SELECT_ENDP_E1O 0x02`
- `#define USB_SELECT_ENDP_E1I 0x03`
- `#define USB_SELECT_ENDP_E2O 0x04`
- `#define USB_SELECT_ENDP_E2I 0x05`
- `#define USB_ENDP_ST_CO 0x40`
- `#define USB_ENDP_ST_CI 0x41`
- `#define USB_ENDP_ST_E1O 0x42`
- `#define USB_ENDP_ST_E1I 0x43`
- `#define USB_ENDP_ST_E2O 0x44`
- `#define USB_ENDP_ST_E2I 0x45`
- `#define USB_ACCESS_BUF 0xF0`
- `#define USB_ACK_SETUP 0xF1`
- `#define USB_CLEAR_BUF 0xF2`
- `#define USB_VALIDATE_BUF 0xFA`
- `#define USB_SEND_RESUME 0xF6`
- `#define USB_READ_FRAME_NUMBER 0xF5`
- `#define D12_INT_EOT 0x100`
- `#define D12_INT_SUSPENDCHANGE 0x80`

- #define **D12_INT_BUSRESET** 0x40
- #define **D12_INT_ENDP2IN** 0x20
- #define **D12_INT_ENDP2OUT** 0x10
- #define **D12_INT_ENDP1IN** 0x08
- #define **D12_INT_ENDP1OUT** 0x04
- #define **D12_INT_ENDP0IN** 0x02
- #define **D12_INT_ENDP0OUT** 0x01
- #define **USB_ENDPOINT_HALT** 0
- #define **USB_ENDPOINT_DIRECTION_MASK** 0x80
- #define **USB_MAX_ENDPOINTS** 0x03
- #define **D12_FULLEEMPTY** 0x01
- #define **D12_STALL** 0x02
- #define **D12_SETUPPACKET** 0x20
- #define **EP0_TX_FIFO_SIZE** 16
- #define **EP0_RX_FIFO_SIZE** 16
- #define **EP0_PACKET_SIZE** 16
- #define **EP1_TX_FIFO_SIZE** 16
- #define **EP1_RX_FIFO_SIZE** 16
- #define **EP1_PACKET_SIZE** 16
- #define **EP2_TX_FIFO_SIZE** 64
- #define **EP2_RX_FIFO_SIZE** 64
- #define **EP2_PACKET_SIZE** 64
- #define **USB_REQUEST_MASK** 0x0F
- #define **USB_REQUEST_GET_STATUS** 0x00
- #define **USB_REQUEST_CLEAR_FEATURE** 0x01
- #define **USB_REQUEST_SET_FEATURE** 0x03
- #define **USB_REQUEST_SET_ADDRESS** 0x05
- #define **USB_REQUEST_GET_DESCRIPTOR** 0x06
- #define **USB_REQUEST_SET_DESCRIPTOR** 0x07
- #define **USB_REQUEST_GET_CONFIGURATION** 0x08
- #define **USB_REQUEST_SET_CONFIGURATION** 0x09
- #define **USB_REQUEST_GET_INTERFACE** 0x0A
- #define **USB_REQUEST_SET_INTERFACE** 0x0B
- #define **USB_REQUEST_SYNC_FRAME** 0x0C
- #define **USB_REQUEST_TYPE_MASK** 0x60
- #define **USB_STANDARD_REQUEST** 0x00
- #define **USB_INTERFACE_REQUEST** 0x01
- #define **USB_ENDPOINT_REQUEST** 0x02
- #define **USB_CLASS_REQUEST** 0x20
- #define **USB_VENDOR_REQUEST** 0x40
- #define **USB_DEVICE_DESCRIPTOR_TYPE** 0x01
- #define **USB_CONFIGURATION_DESCRIPTOR_TYPE** 0x02
- #define **USB_STRING_DESCRIPTOR_TYPE** 0x03
- #define **USB_INTERFACE_DESCRIPTOR_TYPE** 0x04
- #define **USB_ENDPOINT_DESCRIPTOR_TYPE** 0x05
- #define **USB_POWER_DESCRIPTOR_TYPE** 0x06

- #define **USB_CLASS_CODE_TEST_CLASS_DEVICE** 0xDC
- #define **USB_SUBCLASS_CODE_TEST_CLASS_D12** 0xA0
- #define **USB_PROTOCOL_CODE_TEST_CLASS_D12** 0xB0
- #define **USB_RECIPIENT** 0x1f
- #define **USB_RECIPIENT_DEVICE** 0x00
- #define **USB_RECIPIENT_INTERFACE** 0x01
- #define **USB_RECIPIENT_ENDPOINT** 0x02
- #define **USB_SP_REQUEST** 0
- #define **USB_SP_VALUE** 1
- #define **USB_SP_INDEX** 2
- #define **USB_SP_LENGTH** 3

5.28.1 Detailed Description

Miscellaneous USB-related definitions. Not all seem to be VS1000-related.

Definition in file **usb.h**.

5.28.2 Define Documentation

5.28.2.1 #define **D12_FULLEEMPTY** 0x01

Definition at line 47 of file **usb.h**.

5.28.2.2 #define **D12_INT_BUSRESET** 0x40

Definition at line 33 of file **usb.h**.

5.28.2.3 #define **D12_INT_ENDP0IN** 0x02

Definition at line 38 of file **usb.h**.

5.28.2.4 #define **D12_INT_ENDP0OUT** 0x01

Definition at line 39 of file **usb.h**.

5.28.2.5 #define **D12_INT_ENDP1IN** 0x08

Definition at line 36 of file **usb.h**.

5.28.2.6 #define **D12_INT_ENDP1OUT** 0x04

Definition at line 37 of file **usb.h**.

5.28.2.7 #define **D12_INT_ENDP2IN** 0x20

Definition at line 34 of file **usb.h**.

5.28.2.8 #define D12_INT_ENDP2OUT 0x10

Definition at line 35 of file usb.h.

5.28.2.9 #define D12_INT_EOT 0x100

Definition at line 31 of file usb.h.

5.28.2.10 #define D12_INT_SUSPENDCHANGE 0x80

Definition at line 32 of file usb.h.

5.28.2.11 #define D12_SETUPPACKET 0x20

Definition at line 50 of file usb.h.

5.28.2.12 #define D12_STALL 0x02

Definition at line 48 of file usb.h.

5.28.2.13 #define EP0_PACKET_SIZE 16

Definition at line 54 of file usb.h.

5.28.2.14 #define EP0_RX_FIFO_SIZE 16

Definition at line 53 of file usb.h.

5.28.2.15 #define EP0_TX_FIFO_SIZE 16

Definition at line 52 of file usb.h.

5.28.2.16 #define EP1_PACKET_SIZE 16

Definition at line 58 of file usb.h.

5.28.2.17 #define EP1_RX_FIFO_SIZE 16

Definition at line 57 of file usb.h.

5.28.2.18 #define EP1_TX_FIFO_SIZE 16

Definition at line 56 of file usb.h.

5.28.2.19 #define EP2_PACKET_SIZE 64

Definition at line 62 of file usb.h.

5.28.2.20 `#define EP2_RX_FIFO_SIZE 64`

Definition at line 61 of file usb.h.

5.28.2.21 `#define EP2_TX_FIFO_SIZE 64`

Definition at line 60 of file usb.h.

5.28.2.22 `#define USB_ACCESS_BUF 0xF0`

Definition at line 24 of file usb.h.

5.28.2.23 `#define USB_ACK_SETUP 0xF1`

Definition at line 25 of file usb.h.

5.28.2.24 `#define USB_CLASS_CODE_TEST_CLASS_DEVICE 0xDC`

Definition at line 93 of file usb.h.

5.28.2.25 `#define USB_CLASS_REQUEST 0x20`

Definition at line 83 of file usb.h.

5.28.2.26 `#define USB_CLEAR_BUF 0xF2`

Definition at line 26 of file usb.h.

5.28.2.27 `#define USB_CONFIGURATION_DESCRIPTOR_TYPE 0x02`

Definition at line 87 of file usb.h.

5.28.2.28 `#define USB_DEVICE_DESCRIPTOR_TYPE 0x01`

Definition at line 86 of file usb.h.

5.28.2.29 `#define USB_ENDP_ST_CI 0x41`

Definition at line 19 of file usb.h.

5.28.2.30 `#define USB_ENDP_ST_CO 0x40`

Definition at line 18 of file usb.h.

5.28.2.31 `#define USB_ENDP_ST_E1I 0x43`

Definition at line 21 of file usb.h.

5.28.2.32 `#define USB_ENDP_ST_E1O 0x42`

Definition at line 20 of file usb.h.

5.28.2.33 `#define USB_ENDP_ST_E2I 0x45`

Definition at line 23 of file usb.h.

5.28.2.34 `#define USB_ENDP_ST_E2O 0x44`

Definition at line 22 of file usb.h.

5.28.2.35 `#define USB_ENDPOINT_DESCRIPTOR_TYPE 0x05`

Definition at line 90 of file usb.h.

5.28.2.36 `#define USB_ENDPOINT_DIRECTION_MASK 0x80`

Definition at line 42 of file usb.h.

5.28.2.37 `#define USB_ENDPOINT_HALT 0`

Definition at line 41 of file usb.h.

5.28.2.38 `#define USB_ENDPOINT_REQUEST 0x02`

Definition at line 81 of file usb.h.

5.28.2.39 `#define USB_INTERFACE_DESCRIPTOR_TYPE 0x04`

Definition at line 89 of file usb.h.

5.28.2.40 `#define USB_INTERFACE_REQUEST 0x01`

Definition at line 80 of file usb.h.

5.28.2.41 `#define USB_MAX_ENDPOINTS 0x03`

Definition at line 43 of file usb.h.

5.28.2.42 `#define USB_POWER_DESCRIPTOR_TYPE 0x06`

Definition at line 91 of file usb.h.

5.28.2.43 `#define USB_PROTOCOL_CODE_TEST_CLASS_D12 0xB0`

Definition at line 95 of file usb.h.

5.28.2.44 `#define USB_READ_FRAME_NUMBER 0xF5`

Definition at line 29 of file usb.h.

5.28.2.45 `#define USB_READ_INT 0xF4`

Definition at line 11 of file usb.h.

5.28.2.46 `#define USB_RECIPIENT 0x1f`

Definition at line 97 of file usb.h.

5.28.2.47 `#define USB_RECIPIENT_DEVICE 0x00`

Definition at line 98 of file usb.h.

5.28.2.48 `#define USB_RECIPIENT_ENDPOINT 0x02`

Definition at line 100 of file usb.h.

5.28.2.49 `#define USB_RECIPIENT_INTERFACE 0x01`

Definition at line 99 of file usb.h.

5.28.2.50 `#define USB_REQUEST_CLEAR_FEATURE 0x01`

Definition at line 67 of file usb.h.

5.28.2.51 `#define USB_REQUEST_GET_CONFIGURATION 0x08`

Definition at line 72 of file usb.h.

5.28.2.52 `#define USB_REQUEST_GET_DESCRIPTOR 0x06`

Definition at line 70 of file usb.h.

5.28.2.53 `#define USB_REQUEST_GET_INTERFACE 0x0A`

Definition at line 74 of file usb.h.

5.28.2.54 `#define USB_REQUEST_GET_STATUS 0x00`

Definition at line 66 of file usb.h.

5.28.2.55 `#define USB_REQUEST_MASK 0x0F`

Definition at line 65 of file usb.h.

5.28.2.56 `#define USB_REQUEST_SET_ADDRESS 0x05`

Definition at line 69 of file usb.h.

5.28.2.57 `#define USB_REQUEST_SET_CONFIGURATION 0x09`

Definition at line 73 of file usb.h.

5.28.2.58 `#define USB_REQUEST_SET_DESCRIPTOR 0x07`

Definition at line 71 of file usb.h.

5.28.2.59 `#define USB_REQUEST_SET_FEATURE 0x03`

Definition at line 68 of file usb.h.

5.28.2.60 `#define USB_REQUEST_SET_INTERFACE 0x0B`

Definition at line 75 of file usb.h.

5.28.2.61 `#define USB_REQUEST_SYNC_FRAME 0x0C`

Definition at line 76 of file usb.h.

5.28.2.62 `#define USB_REQUEST_TYPE_MASK 0x60`

Definition at line 78 of file usb.h.

5.28.2.63 `#define USB_SELECT_ENDP_CI 0x01`

Definition at line 13 of file usb.h.

5.28.2.64 `#define USB_SELECT_ENDP_CO 0x00`

Definition at line 12 of file usb.h.

5.28.2.65 `#define USB_SELECT_ENDP_E1I 0x03`

Definition at line 15 of file usb.h.

5.28.2.66 `#define USB_SELECT_ENDP_E1O 0x02`

Definition at line 14 of file usb.h.

5.28.2.67 `#define USB_SELECT_ENDP_E2I 0x05`

Definition at line 17 of file usb.h.

5.28.2.68 `#define USB_SELECT_ENDP_E2O 0x04`

Definition at line 16 of file usb.h.

5.28.2.69 `#define USB_SEND_RESUME 0xF6`

Definition at line 28 of file usb.h.

5.28.2.70 `#define USB_SET_ADDRESS_ENA 0xD0`

Definition at line 7 of file usb.h.

5.28.2.71 `#define USB_SET_DMA 0xFB`

Definition at line 10 of file usb.h.

5.28.2.72 `#define USB_SET_ENDP_ENA 0xD8`

Definition at line 8 of file usb.h.

5.28.2.73 `#define USB_SET_MODE 0xF3`

Definition at line 9 of file usb.h.

5.28.2.74 `#define USB_SP_INDEX 2`

Definition at line 104 of file usb.h.

5.28.2.75 `#define USB_SP_LENGTH 3`

Definition at line 105 of file usb.h.

5.28.2.76 `#define USB_SP_REQUEST 0`

Definition at line 102 of file usb.h.

5.28.2.77 `#define USB_SP_VALUE 1`

Definition at line 103 of file usb.h.

5.28.2.78 `#define USB_STANDARD_REQUEST 0x00`

Definition at line 79 of file usb.h.

5.28.2.79 #define USB_STRING_DESCRIPTOR_TYPE 0x03

Definition at line 88 of file usb.h.

5.28.2.80 #define USB_SUBCLASS_CODE_TEST_CLASS_D12 0xA0

Definition at line 94 of file usb.h.

5.28.2.81 #define USB_VALIDATE_BUF 0xFA

Definition at line 27 of file usb.h.

5.28.2.82 #define USB_VENDOR_REQUEST 0x40

Definition at line 84 of file usb.h.

5.29 usblowlib.h File Reference

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

**Data Structures**

- struct **usbpkt**
- struct **USBVARS**

Defines

- #define **AUDIO_ISOC_OUT_EP** 0x01
- #define **MSC_BULK_OUT_ENDPOINT** 0x03
- #define **MSC_BULK_IN_ENDPOINT** 0x02
- #define **ENDPOINT_SIZE_0** 64
- #define **ENDPOINT_SIZE_1** 512
- #define **BRESET_INT** 0x8000
- #define **SOF_INT** 0x4000
- #define **RX_INT** 0x2000
- #define **TX_HOLD_INT** 0x1000
- #define **TX_INT** 0x0800
- #define **NAK_SENT_INT** 0x0400
- #define **SETUP_INFO** 0x0080
- #define **P_SETUP** 0x10
- #define **P_DATA** 0x20
- #define **PID_SOF** 0x05
- #define **PID_SETUP** 0x0D

- #define **PID_IN** 0x09
- #define **PID_OUT** 0x01
- #define **PID_DATA0** 0x03
- #define **PID_DATA1** 0x0B
- #define **PID_ACK** 0x02
- #define **PID_NACK** 0x0A
- #define **PID_STALL** 0x0E
- #define **USB_CONFIG** (USB_BASE)
- #define **USB_CONTROL** (USB_BASE+1)
- #define **USB_STATUS** (USB_BASE+2)
- #define **USB_STF_BUS_RESET** (1<<15)
- #define **USB_STF_SOF** (1<<14)
- #define **USB_STF_RX** (1<<13)
- #define **USB_STF_TX_READY** (1<<12)
- #define **USB_STF_TX_EMPTY** (1<<11)
- #define **USB_STF_NAK** (1<<10)
- #define **USB_STF_SETUP** (1<<7)
- #define **USB_STM_LAST_EP** (15<<0)
- #define **USB_RDPTR** (USB_BASE+3)
- #define **USB_WRPTR** (USB_BASE+4)
- #define **USB_EP_SEND0** (USB_BASE+8)
- #define **USB_EP_SEND1** (USB_BASE+9)
- #define **USB_EP_SEND2** (USB_BASE+10)
- #define **USB_EP_SEND3** (USB_BASE+11)
- #define **USB_EP_ST0** (USB_BASE+16)
- #define **USB_EP_ST1** (USB_BASE+17)
- #define **USB_EP_ST2** (USB_BASE+18)
- #define **USB_EP_ST3** (USB_BASE+19)
- #define **USB_STF_OUT_BULK** (0<<14)
- #define **USB_STF_OUT_INT** (1<<14)
- #define **USB_STF_OUT_ISO** (3<<14)
- #define **USB_STF_OUT_ENABLE** (1<<13)
- #define **USB_STF_OUT_STALL** (1<<12)
- #define **USB_STF_OUT_STALL_SENT** (1<<11)
- #define **USB_STF_OUT_EP_SIZE** (1<<8)
- #define **USB_STF_IN_BULK** (0<<6)
- #define **USB_STF_IN_INT** (1<<6)
- #define **USB_STF_IN_ISO** (3<<6)
- #define **USB_STF_IN_ENABLE** (1<<5)
- #define **USB_STF_IN_STALL** (1<<4)
- #define **USB_STF_IN_STALL_SENT** (1<<3)
- #define **USB_STF_IN_NACK_SENT** (1<<2)
- #define **USB_STF_IN_EMPTY** (1<<1)
- #define **USB_STF_IN_FORCE_NACK** (1<<0)
- #define **RING_BUF_SIZE** 1024
- #define **DT_LANGUAGES** 0

- #define **DT_VENDOR** 1
- #define **DT_MODEL** 2
- #define **DT_SERIAL** 3
- #define **DT_DEVICE** 4
- #define **DT_CONFIGURATION** 5
- #define **USB_MASS_STORAGE** 1
- #define **USB_AUDIO** 2
- #define **AUDIO_DELAY_FRAMES** 12
- #define **AUDIO_DELAY_FRAMES_STR** "\x0c"

Typedefs

- typedef **usbpkt** **USBPacket**

Functions

- auto void **RingBufCopyX** (register __i2 **u_int16** *d, register __i0 const **u_int16** *s, register __a0 **u_int16** n)
- int **USBStartTransmission** (**u_int16** ep, const void *buf, **u_int16** length, **u_int16** requestedLength)
- void **USBContinueTransmission** (**u_int16** ep)
- void **InitUSBDescriptors** (**u_int16** initDescriptors)
- void **RealInitUSBDescriptors** (**u_int16** initDescriptors)
- void **InitUSB** (**u_int16** initDescriptors)
- void **USBResetEndpoint** (register __c0 int ep)
- **u_int16** **USBReceivePacket** (**USBPacket** *packet)
- void **USBSendZeroLengthPacketToEndpoint0** (void)
- void **USBHandler** (void)
- void **RealUSBHandler** ()
- void **DecodeSetupPacket** (void)
- void **RealDecodeSetupPacket** (void)
- void **USBCheckForSetupPacket** (void)
- **u_int16** **USBXmitLeft** (**u_int16** endpoint)
- void **USBSingleStallEndpoint** (register __c0 **u_int16** ep)
- void **USBStallEndpoint** (register __c0 int ep)
- void **USBResetStall** (register __c0 int ep)
- **u_int16** **USBIsEndpointStalled** (register int ep)
- **u_int16** **SwapWord** (register __a1 **u_int16** d)
- auto **u_int16** **USBIsAttached** (void)
- auto **u_int16** **USBIsDetached** (void)
- auto **u_int16** **USBWantsSuspend** (void)
- void **MSCPacketFromPC** (**USBPacket** *setupPacket)
- void **RealMSCPacketFromPC** (**USBPacket** *setupPacket)
- **u_int16** **MscSendCsw** (**u_int16** status)
- void **DiskProtocolError** (char errorcode)
- void **AudioPacketFromUSB** (**u_int16** *data, **s_int16** words)

Variables

- **USBVARS USB**

5.29.1 Detailed Description

VS1000 low-level USB definitions.

Definition in file `usblowlib.h`.

5.29.2 Define Documentation

5.29.2.1 `#define AUDIO_DELAY_FRAMES 12`

Definition at line 311 of file `usblowlib.h`.

5.29.2.2 `#define AUDIO_DELAY_FRAMES_STR "\x0c"`

Definition at line 312 of file `usblowlib.h`.

5.29.2.3 `#define AUDIO_ISOC_OUT_EP 0x01`

OUT endpoint used for audio.

Definition at line 8 of file `usblowlib.h`.

5.29.2.4 `#define BRESET_INT 0x8000`

Bus reset interrupt

Definition at line 18 of file `usblowlib.h`.

5.29.2.5 `#define DT_CONFIGURATION 5`

Definition at line 147 of file `usblowlib.h`.

5.29.2.6 `#define DT_DEVICE 4`

Definition at line 146 of file `usblowlib.h`.

5.29.2.7 `#define DT_LANGUAGES 0`

Definition at line 142 of file `usblowlib.h`.

5.29.2.8 `#define DT_MODEL 2`

Definition at line 144 of file `usblowlib.h`.

5.29.2.9 #define DT_SERIAL 3

Definition at line 145 of file usblowlib.h.

5.29.2.10 #define DT_VENDOR 1

Definition at line 143 of file usblowlib.h.

5.29.2.11 #define ENDPOINT_SIZE_0 64

The endpoint size supported for IN.

Definition at line 12 of file usblowlib.h.

5.29.2.12 #define ENDPOINT_SIZE_1 512

The max OUT packet size, mainly for audio.

Definition at line 13 of file usblowlib.h.

5.29.2.13 #define MSC_BULK_IN_ENDPOINT 0x02

IN endpoint used for SCSI.

Definition at line 10 of file usblowlib.h.

5.29.2.14 #define MSC_BULK_OUT_ENDPOINT 0x03

OUT endpoint used for SCSI.

Definition at line 9 of file usblowlib.h.

5.29.2.15 #define NAK_SENT_INT 0x0400

Packet NAKed

Definition at line 23 of file usblowlib.h.

5.29.2.16 #define P_DATA 0x20

Definition at line 30 of file usblowlib.h.

5.29.2.17 #define P_SETUP 0x10

Definition at line 29 of file usblowlib.h.

5.29.2.18 #define PID_ACK 0x02

Definition at line 37 of file usblowlib.h.

5.29.2.19 #define PID_DATA0 0x03

Definition at line 35 of file usblowlib.h.

5.29.2.20 #define PID_DATA1 0x0B

Definition at line 36 of file usblowlib.h.

5.29.2.21 #define PID_IN 0x09

Definition at line 33 of file usblowlib.h.

5.29.2.22 #define PID_NACK 0x0A

Definition at line 38 of file usblowlib.h.

5.29.2.23 #define PID_OUT 0x01

Definition at line 34 of file usblowlib.h.

5.29.2.24 #define PID_SETUP 0x0D

Definition at line 32 of file usblowlib.h.

5.29.2.25 #define PID_SOF 0x05

Definition at line 31 of file usblowlib.h.

5.29.2.26 #define PID_STALL 0x0E

Definition at line 39 of file usblowlib.h.

5.29.2.27 #define RING_BUF_SIZE 1024

Definition at line 97 of file usblowlib.h.

5.29.2.28 #define RX_INT 0x2000

OUT packet received

Definition at line 20 of file usblowlib.h.

5.29.2.29 #define SETUP_INFO 0x0080

Setup packet

Definition at line 24 of file usblowlib.h.

5.29.2.30 #define SOF_INT 0x4000

Start of Frame interrupt

Definition at line 19 of file usblowlib.h.

5.29.2.31 #define TX_HOLD_INT 0x1000

Copied packet to IN registers

Definition at line 21 of file usblowlib.h.

5.29.2.32 #define TX_INT 0x0800

Successfully transmitted packet

Definition at line 22 of file usblowlib.h.

5.29.2.33 #define USB_AUDIO 2

Definition at line 298 of file usblowlib.h.

5.29.2.34 #define USB_CONFIG (USB_BASE)

Definition at line 45 of file usblowlib.h.

5.29.2.35 #define USB_CONTROL (USB_BASE+1)

Definition at line 46 of file usblowlib.h.

5.29.2.36 #define USB_EP_SEND0 (USB_BASE+8)

Definition at line 61 of file usblowlib.h.

5.29.2.37 #define USB_EP_SEND1 (USB_BASE+9)

Definition at line 62 of file usblowlib.h.

5.29.2.38 #define USB_EP_SEND2 (USB_BASE+10)

Definition at line 63 of file usblowlib.h.

5.29.2.39 #define USB_EP_SEND3 (USB_BASE+11)

Definition at line 64 of file usblowlib.h.

5.29.2.40 #define USB_EP_ST0 (USB_BASE+16)

Definition at line 70 of file usblowlib.h.

5.29.2.41 `#define USB_EP_ST1 (USB_BASE+17)`

Definition at line 71 of file usblowlib.h.

5.29.2.42 `#define USB_EP_ST2 (USB_BASE+18)`

Definition at line 72 of file usblowlib.h.

5.29.2.43 `#define USB_EP_ST3 (USB_BASE+19)`

Definition at line 73 of file usblowlib.h.

5.29.2.44 `#define USB_MASS_STORAGE 1`

Definition at line 297 of file usblowlib.h.

5.29.2.45 `#define USB_RDPTR (USB_BASE+3)`

USB receive buffer read pointer.

Definition at line 58 of file usblowlib.h.

5.29.2.46 `#define USB_STATUS (USB_BASE+2)`

Definition at line 47 of file usblowlib.h.

5.29.2.47 `#define USB_STF_BUS_RESET (1<<15)`

Definition at line 49 of file usblowlib.h.

5.29.2.48 `#define USB_STF_IN_BULK (0<<6)`

Definition at line 86 of file usblowlib.h.

5.29.2.49 `#define USB_STF_IN_EMPTY (1<<1)`

Definition at line 93 of file usblowlib.h.

5.29.2.50 `#define USB_STF_IN_ENABLE (1<<5)`

Definition at line 89 of file usblowlib.h.

5.29.2.51 `#define USB_STF_IN_FORCE_NACK (1<<0)`

Definition at line 94 of file usblowlib.h.

5.29.2.52 `#define USB_STF_IN_INT (1<<6)`

Definition at line 87 of file usblowlib.h.

5.29.2.53 `#define USB_STF_IN_ISO (3<<6)`

Definition at line 88 of file usblowlib.h.

5.29.2.54 `#define USB_STF_IN_NACK_SENT (1<<2)`

Definition at line 92 of file usblowlib.h.

5.29.2.55 `#define USB_STF_IN_STALL (1<<4)`

Definition at line 90 of file usblowlib.h.

5.29.2.56 `#define USB_STF_IN_STALL_SENT (1<<3)`

Definition at line 91 of file usblowlib.h.

5.29.2.57 `#define USB_STF_NAK (1<<10)`

Definition at line 54 of file usblowlib.h.

5.29.2.58 `#define USB_STF_OUT_BULK (0<<14)`

Definition at line 79 of file usblowlib.h.

5.29.2.59 `#define USB_STF_OUT_ENABLE (1<<13)`

Definition at line 82 of file usblowlib.h.

5.29.2.60 `#define USB_STF_OUT_EP_SIZE (1<<8)`

Definition at line 85 of file usblowlib.h.

5.29.2.61 `#define USB_STF_OUT_INT (1<<14)`

Definition at line 80 of file usblowlib.h.

5.29.2.62 `#define USB_STF_OUT_ISO (3<<14)`

Definition at line 81 of file usblowlib.h.

5.29.2.63 `#define USB_STF_OUT_STALL (1<<12)`

Definition at line 83 of file usblowlib.h.

5.29.2.64 `#define USB_STF_OUT_STALL_SENT (1<<11)`

Definition at line 84 of file usblowlib.h.

5.29.2.65 `#define USB_STF_RX (1<<13)`

Definition at line 51 of file usblowlib.h.

5.29.2.66 `#define USB_STF_SETUP (1<<7)`

Definition at line 55 of file usblowlib.h.

5.29.2.67 `#define USB_STF_SOF (1<<14)`

Definition at line 50 of file usblowlib.h.

5.29.2.68 `#define USB_STF_TX_EMPTY (1<<11)`

Definition at line 53 of file usblowlib.h.

5.29.2.69 `#define USB_STF_TX_READY (1<<12)`

Definition at line 52 of file usblowlib.h.

5.29.2.70 `#define USB_STM_LAST_EP (15<<0)`

Definition at line 56 of file usblowlib.h.

5.29.2.71 `#define USB_WRPTR (USB_BASE+4)`

USB receive buffer write pointer.

Definition at line 59 of file usblowlib.h.

5.29.3 Typedef Documentation**5.29.3.1** `typedef struct usbpkt USBPacket`

Holding space for one received USB packet.

5.29.4 Function Documentation**5.29.4.1** `void AudioPacketFromUSB (u_int16 * data, s_int16 words)`

Called each time a packet is received to the audio endpoint. The function must be provided.

5.29.4.2 `void DecodeSetupPacket (void)`

Hook: handles setup packets. Default: DecodeSetupPacket.

5.29.4.3 `void DiskProtocolError (char errorcode)`

5.29.4.4 void InitUSB (u_int16 *initDescriptors*)

Initializes USB subsystem.

Parameters:

initDescriptors 0=no init, 1=Mass Storage, 2=Audio

5.29.4.5 void InitUSBDescriptors (u_int16 *initDescriptors*)

Hook: Initializes USB descriptor table. Default: RealInitUSBDescriptors.

Parameters:

initDescriptors 0=no init, 1=Mass Storage, 2=Audio

5.29.4.6 void MSCPacketFromPC (USBPacket * *setupPacket*)

Hook: handle a mass storage packet. Default: RealMSCPacketFromPC.

5.29.4.7 u_int16 MscSendCsw (u_int16 *status*)**5.29.4.8 void RealDecodeSetupPacket (void)**

Processes setup packets.

5.29.4.9 void RealInitUSBDescriptors (u_int16 *initDescriptors*)

Initializes USB descriptor table.

Parameters:

initDescriptors 0=no init, 1=Mass Storage, 2=Audio

5.29.4.10 void RealMSCPacketFromPC (USBPacket * *setupPacket*)

Handles a mass storage packet.

5.29.4.11 void RealUSBHandler ()

Polled handling of USB packets. Both Mass Storage and Audio are processed, but the descriptor decides which one the host uses.

5.29.4.12 auto void RingBufCopyX (register __i2 u_int16 * *d*, register __i0 const u_int16 * *s*, register __a0 u_int16 *n*)

Copies packet from USB receive memory ring buffer to X memory buffer. The data is in big-endian format, i.e. the high 8 bits of a word are the first byte, the low 8 bits the second byte. Does not touch USB peripheral registers.

Parameters:

- d* Destination pointer.
- s* Source pointer, must point to the USB receive memory.
- n* The number of words to copy.

5.29.4.13 u_int16 SwapWord (register __a1 u_int16 d)

Swaps byte order

5.29.4.14 void USBCheckForSetupPacket (void)

Checks if any setup packets are waiting and discards any packets that appear before it in the input queue.

5.29.4.15 void USBContinueTransmission (u_int16 ep)

Continues sending a packet.

Parameters:

- ep* The endpoint.

5.29.4.16 void USBHandler (void)

Hook: polled handling of USB packets. Default: RealUSBHandler.

5.29.4.17 auto u_int16 USBIsAttached (void)

Uses USB pull-up enable and USB_DN to detect if USB is attached. USB is attached if the USB pull-up is enabled. USB is not attached if USB_DN is high for every sampling in 20000 clock cycles, and is attached if USB_DN is low for even one sampling.

Returns:

- 0 is USB is not attached, non-zero if USB is attached.

5.29.4.18 auto u_int16 USBIsDetached (void)

Checks USB_DP and USB_DN. If both are high, the unit is probably detached. You **MUST** also use the **USBWantsSuspend()**(p.148) function to see that there has not been SOF's for a while. Otherwise you will get false detections.

Returns:

- 0 is USB is still attached, non-zero if it might be detached.

5.29.4.19 `u_int16 USBIsEndpointStalled (register int ep)`

Returns the stall state of an endpoint.

Parameters:

ep 0x80|endpoint for IN endpoints.

Returns:

non-zero if endpoint is stalled.

5.29.4.20 `u_int16 USBReceivePacket (USBPacket * packet)`

Fetches a packet from USB receive FIFO. Also clears RX_INT and SETUP_INFO.

Parameters:

packet pointer to the packet structure

Returns:

endpoint number

5.29.4.21 `void USBResetEndpoint (register __c0 int ep)`

Resets an endpoint. Data toggles are also reset and possible IN packets waiting for transmit are discarded.

Parameters:

ep 0 for control endpoint, 0x80|endpoint for IN.

5.29.4.22 `void USBResetStall (register __c0 int ep)`

Resets the stall state of an endpoint.

Parameters:

ep 0x80|endpoint for IN endpoints.

5.29.4.23 `void USBSendZeroLengthPacketToEndpoint0 (void)`

Sends an empty packet to the control endpoint. Returns when the packet has been sent or a bus reset is received.

Bug

: currently is satisfied if any active IN is finished.

5.29.4.24 void USBSingleStallEndpoint (register __c0 u_int16 ep)

Sends (at least) one STALL to the specified endpoint.

Parameters:

ep 0x80|endpoint for IN endpoints.

5.29.4.25 void USBStallEndpoint (register __c0 int ep)

Sets the endpoint to stall state.

Parameters:

ep Endpoint number for OUT endpoint, 0x80|endpoint for IN endpoints.

5.29.4.26 int USBStartTransmission (u_int16 ep, const void * buf, u_int16 length, u_int16 requestedLength)

Starts transmission of data in endpoint-size -sized parts. Internall calls USBContinueTransmission once to start the transfer.

Parameters:

ep The endpoint to use.

buf Data to send.

length Length of data to send in bytes.

requestedLength Data size that was requested in bytes.

Returns:

0 if successful, non-zero if endpoint was busy.

5.29.4.27 auto u_int16 USBWantsSuspend (void)

Checks if there has been no SOF for 10..20ms.

Bug

USB suspend time is 3ms. Because of our software timer accuracy we detect suspend when there has been no SOF for 10..20ms.

This routine requires that the unit is configured before returning non-zero.

Returns:

0 if the communication is proceeding normally, non-zero if suspend state is detected.

5.29.4.28 u_int16 USBXmitLeft (u_int16 endpoint)

Checks how many bytes of a packet are waiting to be sent.

Parameters:

endpoint the endpoint to check.

Returns:

the number of bytes left

5.29.5 Variable Documentation**5.29.5.1 struct USBVARS USB****5.30 vectors.h File Reference**

```
#include "usbblowlib.h"
```

Include dependency graph for vectors.h:

**Functions**

- void **USBHandler** (void)
- void **RealUSBHandler** (void)
- void **DecodeSetupPacket** (void)
- void **RealDecodeSetupPacket** (void)
- void **MSCPacketFromPC** (USBPacket *inPacket)
- void **RealMSCPacketFromPC** (USBPacket *inPacket)

5.30.1 Function Documentation

5.30.1.1 void DecodeSetupPacket (void)

5.30.1.2 void MSCPacketFromPC (USBPacket * *inPacket*)

5.30.1.3 void RealDecodeSetupPacket (void)

5.30.1.4 void RealMSCPacketFromPC (USBPacket * *inPacket*)

5.30.1.5 void RealUSBHandler (void)

5.30.1.6 void USBHandler (void)

5.31 vs1000.h File Reference

```
#include <vstypes.h>
```

Include dependency graph for vs1000.h:



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



Defines

- #define **IROM_START** 0x4000
- #define **IROM_SIZE** 0x4a80
- #define **XROM_START** 0x4000
- #define **XROM_SIZE** 0x0c00
- #define **YROM_START** 0x4000
- #define **YROM_SIZE** 0x2800
- #define **XRAM_START** 0x0
- #define **XRAM_SIZE** 0x3400
- #define **YRAM_START** 0x0
- #define **YRAM_SIZE** 0x4000
- #define **IRAM_START** 0x0
- #define **IRAM_SIZE** 0x800
- #define **DCT_START** 0x1000
- #define **STACK_START** 0x1800
- #define **STACK_SIZE** 0x200
- #define **DEBUG_STACK** (STACK_START+STACK_SIZE-32)
- #define **OTHERS_START** 0x0800
- #define **YPREV0_START** 0x0000
- #define **YPREV1_START** 0x0400
- #define **AUDIO_START** 0x0000
- #define **INTV_GPIO1** 10
- #define **INTV_GPIO0** 9
- #define **INTV_REGU** 8
- #define **INTV_TIM1** 7
- #define **INTV_TIM0** 6
- #define **INTV_RX** 5
- #define **INTV_TX** 4
- #define **INTV_NFLSH** 3
- #define **INTV_USB** 2
- #define **INTV_SPI** 1

- #define **INTV_DAC** 0
- #define **INTF_GPIO1** (1<<INTV_GPIO1)
- #define **INTF_GPIO0** (1<<INTV_GPIO0)
- #define **INTF_REGU** (1<<INTV_REGU)
- #define **INTF_TIM1** (1<<INTV_TIM1)
- #define **INTF_TIM0** (1<<INTV_TIM0)
- #define **INTF_RX** (1<<INTV_RX)
- #define **INTF_TX** (1<<INTV_TX)
- #define **INTF_NFLSH** (1<<INTV_NFLSH)
- #define **INTF_USB** (1<<INTV_USB)
- #define **INTF_SPI** (1<<INTV_SPI)
- #define **INTF_DAC** (1<<INTV_DAC)
- #define **INT_EN_NONE** 0
- #define **INT_EN_GPIO1** INTV_GPIO1
- #define **INT_EN_GPIO0** INTV_GPIO0
- #define **INT_EN_REGU** INTV_REGU
- #define **INT_EN_TIM1** INTV_TIM1
- #define **INT_EN_TIM0** INTV_TIM0
- #define **INT_EN_RX** INTV_RX
- #define **INT_EN_TX** INTV_TX
- #define **INT_EN_NFLSH** INTV_NFLSH
- #define **INT_EN_USB** INTV_USB
- #define **INT_EN_SPI** INTV_SPI
- #define **INT_EN_DAC** INTV_DAC
- #define **SCI_SYSTEM** 0xC000
- #define **SCISYSF_CLKDIV** 0x8000
- #define **SCISYSF_AVDD** (1<<10)
- #define **SCISYSF_IOVDD** (1<<5)
- #define **SCISYSF_CVDD** (1<<0)
- #define **SCI_STATUS** 0xC001
- #define **SCISTF_SLOW_CLKMODE** (1<<15)
- #define **SCISTF_USB_DN_OUT** (1<<14)
- #define **SCISTF_USB_DP_OUT** (1<<13)
- #define **SCISTF_USB_DDR** (1<<12)
- #define **SCISTF_VCM_OVERLOAD** (1<<11)
- #define **SCISTF_VCM_DISABLE** (1<<10)
- #define **SCISTF_USB_DP** (1<<9)
- #define **SCISTF_USB_DN** (1<<8)
- #define **SCISTF_USB_DIFF_ENA** (1<<7)
- #define **SCISTF_USB_PULLUP_ENA** (1<<6)
- #define **SCISTF_REGU_POWERLOW** (1<<5)
- #define **SCISTF_REGU_POWERBUT** (1<<4)
- #define **SCISTF_ANADRV_PDOWN** (1<<3)
- #define **SCISTF_ANA_PDOWN** (1<<2)
- #define **SCISTF_REGU_CLOCK** (1<<1)
- #define **SCISTF_REGU_SHUTDOWN** (1<<0)

- #define **SCI_DEBUG** 0xC002
- #define **GPIO0_MODE** 0xC010
- #define **GPIO1_MODE** 0xC011
- #define **DAC_VOL** 0xC012
- #define **FREQCTL** 0xC013
- #define **FREQCTLH** 0xC014
- #define **FCH_MUL0_B** 4
- #define **FCH_MUL1_B** 5
- #define **FCH_MUL2_B** 6
- #define **FCH_MUL3_B** 7
- #define **FCH_DIV_INCLK_B** 8
- #define **FCH_FORCE_PLL_B** 9
- #define **FCH_VCO_OUT_ENA_B** 11
- #define **FCH_PLL_SET_LOCK_B** 12
- #define **FCH_PLL_LOCKED_B** 13
- #define **FCH_MUL0** (1<<FCH_MUL0_B)
- #define **FCH_MUL1** (1<<FCH_MUL1_B)
- #define **FCH_MUL2** (1<<FCH_MUL2_B)
- #define **FCH_MUL3** (1<<FCH_MUL3_B)
- #define **FCH_DIV_INCLK** (1<<FCH_DIV_INCLK_B)
- #define **FCH_FORCE_PLL** (1<<FCH_FORCE_PLL_B)
- #define **FCH_VCO_OUT_ENA** (1<<FCH_VCO_OUT_ENA_B)
- #define **FCH_PLL_SET_LOCK** (1<<FCH_PLL_SET_LOCK_B)
- #define **FCH_PLL_LOCKED** (1<<FCH_PLL_LOCKED_B)
- #define **DAC_LEFT** 0xC015
- #define **DAC_RIGHT** 0xC016
- #define **WDOG_CONFIG** 0xC020
- #define **WDOG_RESET** 0xC021
- #define **WDOG_DUMMY** 0xC022
- #define **WDOG_RESET_VAL** 0x4ea9
- #define **UART_STATUS** 0xC028
- #define **UART_DATA** 0xC029
- #define **UART_DATAH** 0xC02A
- #define **UART_DIV** 0xC02B
- #define **UART_ST_RXORUN** (1<<3)
- #define **UART_ST_RXFULL** (1<<2)
- #define **UART_ST_TXFULL** (1<<1)
- #define **UART_ST_TXRUNNING** (1<<0)
- #define **TIMER_CONFIG** 0xC030
- #define **TIMER_ENABLE** 0xC031
- #define **TIMER_T0L** 0xC034
- #define **TIMER_T0H** 0xC035
- #define **TIMER_T0CNTL** 0xC036
- #define **TIMER_T0CNTH** 0xC037
- #define **TIMER_T1L** 0xC038

- #define **TIMER_T1H** 0xC039
- #define **TIMER_T1CNTL** 0xC03A
- #define **TIMER_T1CNTH** 0xC03B
- #define **GPIO0_DDR** 0xC040
- #define **GPIO0_ODATA** 0xC041
- #define **GPIO0_IDATA** 0xC042
- #define **GPIO0_INT_FALL** 0xC043
- #define **GPIO0_INT_RISE** 0xC044
- #define **GPIO0_INT_PEND** 0xC045
- #define **GPIO0_SET_MASK** 0xC046
- #define **GPIO0_CLEAR_MASK** 0xC047
- #define **GPIO0_BIT_CONF** 0xC048
- #define **GPIO0_BIT_ENG0** 0xC049
- #define **GPIO0_BIT_ENG1** 0xC04A
- #define **GPIO0_READY** 0x0100
- #define **GPIO0_RD** 0x0200
- #define **GPIO0_CS1** 0x0400
- #define **GPIO0_WR** 0x0800
- #define **GPIO0_CLE** 0x1000
- #define **GPIO0_ALE** 0x2000
- #define **GPIO1_DDR** 0xC050
- #define **GPIO1_ODATA** 0xC051
- #define **GPIO1_IDATA** 0xC052
- #define **GPIO1_INT_FALL** 0xC053
- #define **GPIO1_INT_RISE** 0xC054
- #define **GPIO1_INT_PEND** 0xC055
- #define **GPIO1_SET_MASK** 0xC056
- #define **GPIO1_CLEAR_MASK** 0xC057
- #define **GPIO1_BIT_CONF** 0xC058
- #define **GPIO1_BIT_ENG0** 0xC059
- #define **GPIO1_BIT_ENG1** 0xC05A
- #define **SPI0_CONFIG** 0xC068
- #define **SPI0_CLKCONFIG** 0xC069
- #define **SPI0_STATUS** 0xC06A
- #define **SPI0_DATA** 0xC06B
- #define **SPI0_FSYNC** 0xC06C
- #define **SPI_CF_INTXCS** (0<<6)
- #define **SPI_CF_FALLXCS** (2<<6)
- #define **SPI_CF_RISEXCS** (3<<6)
- #define **SPI_CF_MASTER** (1<<5)
- #define **SPI_CF_SLAVE** (0<<5)
- #define **SPI_CF_DLEN** (1<<1)
- #define **SPI_CF_DLEN8** (7<<1)
- #define **SPI_CF_DLEN16** (15<<1)
- #define **SPI_CF_FSIDLE1** (1<<0)
- #define **SPI_CF_FSIDLE0** (0<<0)

- #define **SPI_CC_CLKDIV** (1<<2)
- #define **SPI_ST_BREAK** (1<<5)
- #define **SPI_ST_RXORUN** (1<<4)
- #define **SPI_ST_RXFULL** (1<<3)
- #define **SPI_ST_TXFULL** (1<<2)
- #define **SPI_ST_TXRUNNING** (1<<1)
- #define **SPI_ST_TXURUN** (1<<0)
- #define **NFLSH_CTRL** 0xC060
- #define **NFLSH_CF_LCD_CE_MODE** (1<<8)
- #define **NFLSH_CF_INT_ENABLE** (1<<7)
- #define **NFLSH_CF_NF_RESET** (1<<6)
- #define **NFLSH_CF_WAITSTATES** (1<<0)
- #define **NFLSH_LPL** 0xC061
- #define **NFLSH_CP_LPH** 0xC062
- #define **NFLSH_DATA** 0xC063
- #define **NFLSH_NFIF** 0xC064
- #define **NFLSH_NB_BYTECNT** (8)
- #define **NFLSH_NF_BYTECNT** (1<<NFLSH_NB_BYTECNT)
- #define **NFLSH_NF_USE_DBUF** (1<<7)
- #define **NFLSH_NF_POINTER** (1<<2)
- #define **NFLSH_NF_START** (1<<1)
- #define **NFLSH_NF_READ** (1<<0)
- #define **NFLSH_DSPIF** 0xC065
- #define **NFLSH_DB_POINTER** (4)
- #define **NFLSH_DF_POINTER** (1<<NFLSH_DB_POINTER)
- #define **NFLSH_DF_ENA_DBUF** (1<<3)
- #define **NFLSH_DF_READ** (1<<2)
- #define **NFLSH_DF_ECC_CALC** (1<<1)
- #define **NFLSH_DF_ECC_RESET** (1<<0)
- #define **NFLSH_ECC_CNT** 0xC066
- #define **INT_ENABLE** 0xC070
- #define **INT_ENABLEL** 0xC070
- #define **INT_ENABLEH** 0xC072
- #define **INT_ORIGIN** 0xC074
- #define **INT_VECTOR** 0xC076
- #define **INT_ENCOUNT** 0xC077
- #define **INT_GLOB_DIS** 0xC078
- #define **INT_GLOB_EN** 0xC079
- #define **USB_BASE** 0xC080U
- #define **USB_RECV_MEM** 0x2C00
- #define **USB_SEND_MEM** 0x3000
- #define **PERIP_IN_X**
- #define **PERIP(x)** USEX(x)

Enumerations

- enum **voltIdx** {
voltCorePlayer = 0, **voltIoPlayer**, **voltAnaPlayer**, **voltCoreUSB**,
voltIoUSB, **voltAnaUSB**, **voltCoreSuspend**, **voltIoSuspend**,
voltAnaSuspend, **voltCoreUser**, **voltIoUser**, **voltAnaUser**,
voltEnd }

Functions

- void **SpiBoot** (register __a0 short clkConf, register __i2 short addr, register __i0 short m24)
- void **SpiLoad** (register __i2 short startAddr, register __i0 short m24)
- void **SpiDelay** (register __a0 u_int16 wait)
- auto u_int16 **SpiSendReceive** (register __a0 u_int16 data)
- void **Restart** (void)
- void **IdleHook** (void)
- auto u_int16 **InitFileSystem** (void)
- auto s_int16 **OpenFile** (register __c0 u_int16 fileNum)
- auto s_int16 **ReadFile** (register __i3 u_int16 *buf, register __c1 s_int16 byteOff, register __c0 s_int16 byteSize)
- u_int32 **Seek** (register __reg_a u_int32 pos)
- u_int32 **Tell** (void)
- auto u_int16 **ReadDiskSector** (register __i0 u_int16 *buffer, register __a u_int32 sector)
- auto u_int16 **MapperReadDiskSector** (register __i0 u_int16 *buffer, register __a u_int32 sector)
- void **Disable** (void)
- void **Enable** (void)
- void **Sleep** (void)
- void **NullHook** (void)
- void * **SetHookFunction** (register __i0 u_int16 hook, register __a0 void *newFunc)
- void **BootFromX** (register __i0 u_int16 *start)
- void **SinTest** (void)
- void **MemTests** (register short __b0 muxTestResult)
- void **BusyWait10** (void)
- void **PowerSetVoltages** (u_int16 volt[3])
- void **PowerOff** (void)
- void **RealPowerOff** (void)
- u_int16 **PlayCurrentFile** (void)
- u_int16 **RealPlayCurrentFile** (void)
- void **LoadCheck** (struct **CodecServices** *cs, s_int16 n)
- void **RealLoadCheck** (struct **CodecServices** *cs, s_int16 n)
- u_int16 **UnsupportedFile** (struct **CodecServices** *cs)
- u_int16 **DefUnsupportedFile** (struct **CodecServices** *cs)

- **FsMapper * FsMapRamCreate** (struct **FsPhysical** *physical, **u_int16** cacheSize)
- **void patch** (register __a0 **s_int16** ch)
- **s_int16 getch** (void)

Variables

- **u_int16 voltages** [voltEnd]
- **u_int16 g_dctlo** [2048]
- **__y u_int16 g_dcthi** [2048]
- **s_int16 g_others** [2048]
- **s_int16 g_yprev0** [1024]
- **s_int16 g_yprev1** [1024]

5.31.1 Detailed Description

VS1000 hardware register definitions and general functions.

Definition in file **vs1000.h**.

5.31.2 Define Documentation

5.31.2.1 **#define AUDIO_START 0x0000**

Definition at line 45 of file vs1000.h.

5.31.2.2 **#define DAC_LEFT 0xC015**

Definition at line 145 of file vs1000.h.

5.31.2.3 **#define DAC_RIGHT 0xC016**

Definition at line 146 of file vs1000.h.

5.31.2.4 **#define DAC_VOL 0xC012**

Definition at line 117 of file vs1000.h.

5.31.2.5 **#define DCT_START 0x1000**

Definition at line 38 of file vs1000.h.

5.31.2.6 **#define DEBUG_STACK (STACK_START+STACK_SIZE-32)**

Definition at line 41 of file vs1000.h.

5.31.2.7 #define FCH_DIV_INCLK (1<<FCH_DIV_INCLK_B)

Definition at line 137 of file vs1000.h.

5.31.2.8 #define FCH_DIV_INCLK_B 8

Definition at line 126 of file vs1000.h.

5.31.2.9 #define FCH_FORCE_PLL (1<<FCH_FORCE_PLL_B)

Definition at line 138 of file vs1000.h.

5.31.2.10 #define FCH_FORCE_PLL_B 9

Definition at line 127 of file vs1000.h.

5.31.2.11 #define FCH_MUL0 (1<<FCH_MUL0_B)

Definition at line 133 of file vs1000.h.

5.31.2.12 #define FCH_MUL0_B 4

Definition at line 122 of file vs1000.h.

5.31.2.13 #define FCH_MUL1 (1<<FCH_MUL1_B)

Definition at line 134 of file vs1000.h.

5.31.2.14 #define FCH_MUL1_B 5

Definition at line 123 of file vs1000.h.

5.31.2.15 #define FCH_MUL2 (1<<FCH_MUL2_B)

Definition at line 135 of file vs1000.h.

5.31.2.16 #define FCH_MUL2_B 6

Definition at line 124 of file vs1000.h.

5.31.2.17 #define FCH_MUL3 (1<<FCH_MUL3_B)

Definition at line 136 of file vs1000.h.

5.31.2.18 #define FCH_MUL3_B 7

Definition at line 125 of file vs1000.h.

5.31.2.19 `#define FCH_PLL_LOCKED (1<<FCH_PLL_LOCKED_B)`

Definition at line 142 of file vs1000.h.

5.31.2.20 `#define FCH_PLL_LOCKED_B 13`

Definition at line 131 of file vs1000.h.

5.31.2.21 `#define FCH_PLL_SET_LOCK (1<<FCH_PLL_SET_LOCK_B)`

Definition at line 141 of file vs1000.h.

5.31.2.22 `#define FCH_PLL_SET_LOCK_B 12`

Definition at line 130 of file vs1000.h.

5.31.2.23 `#define FCH_VCO_OUT_ENA (1<<FCH_VCO_OUT_ENA_B)`

Definition at line 140 of file vs1000.h.

5.31.2.24 `#define FCH_VCO_OUT_ENA_B 11`

Definition at line 129 of file vs1000.h.

5.31.2.25 `#define FREQCTLH 0xC014`

Definition at line 120 of file vs1000.h.

5.31.2.26 `#define FREQCTLL 0xC013`

Definition at line 119 of file vs1000.h.

5.31.2.27 `#define GPIO0_ALE 0x2000`

Definition at line 192 of file vs1000.h.

5.31.2.28 `#define GPIO0_BIT_CONF 0xC048`

Definition at line 183 of file vs1000.h.

5.31.2.29 `#define GPIO0_BIT_ENG0 0xC049`

Definition at line 184 of file vs1000.h.

5.31.2.30 `#define GPIO0_BIT_ENG1 0xC04A`

Definition at line 185 of file vs1000.h.

5.31.2.31 #define GPIO0_CLE 0x1000

Definition at line 191 of file vs1000.h.

5.31.2.32 #define GPIO0_CLEAR_MASK 0xC047

Definition at line 182 of file vs1000.h.

5.31.2.33 #define GPIO0_CS1 0x0400

Definition at line 189 of file vs1000.h.

5.31.2.34 #define GPIO0_DDR 0xC040

Definition at line 175 of file vs1000.h.

5.31.2.35 #define GPIO0_IDATA 0xC042

Definition at line 177 of file vs1000.h.

5.31.2.36 #define GPIO0_INT_FALL 0xC043

Definition at line 178 of file vs1000.h.

5.31.2.37 #define GPIO0_INT_PEND 0xC045

Definition at line 180 of file vs1000.h.

5.31.2.38 #define GPIO0_INT_RISE 0xC044

Definition at line 179 of file vs1000.h.

5.31.2.39 #define GPIO0_MODE 0xC010

Definition at line 114 of file vs1000.h.

5.31.2.40 #define GPIO0_ODATA 0xC041

Definition at line 176 of file vs1000.h.

5.31.2.41 #define GPIO0_RD 0x0200

Definition at line 188 of file vs1000.h.

5.31.2.42 #define GPIO0_READY 0x0100

Definition at line 187 of file vs1000.h.

5.31.2.43 `#define GPIO0_SET_MASK 0xC046`

Definition at line 181 of file vs1000.h.

5.31.2.44 `#define GPIO0_WR 0x0800`

Definition at line 190 of file vs1000.h.

5.31.2.45 `#define GPIO1_BIT_CONF 0xC058`

Definition at line 202 of file vs1000.h.

5.31.2.46 `#define GPIO1_BIT_ENG0 0xC059`

Definition at line 203 of file vs1000.h.

5.31.2.47 `#define GPIO1_BIT_ENG1 0xC05A`

Definition at line 204 of file vs1000.h.

5.31.2.48 `#define GPIO1_CLEAR_MASK 0xC057`

Definition at line 201 of file vs1000.h.

5.31.2.49 `#define GPIO1_DDR 0xC050`

Definition at line 194 of file vs1000.h.

5.31.2.50 `#define GPIO1_IDATA 0xC052`

Definition at line 196 of file vs1000.h.

5.31.2.51 `#define GPIO1_INT_FALL 0xC053`

Definition at line 197 of file vs1000.h.

5.31.2.52 `#define GPIO1_INT_PEND 0xC055`

Definition at line 199 of file vs1000.h.

5.31.2.53 `#define GPIO1_INT_RISE 0xC054`

Definition at line 198 of file vs1000.h.

5.31.2.54 `#define GPIO1_MODE 0xC011`

Definition at line 115 of file vs1000.h.

5.31.2.55 `#define GPIO1_ODATA 0xC051`

Definition at line 195 of file vs1000.h.

5.31.2.56 `#define GPIO1_SET_MASK 0xC056`

Definition at line 200 of file vs1000.h.

5.31.2.57 `#define INT_EN_DAC INTV_DAC`

Definition at line 82 of file vs1000.h.

5.31.2.58 `#define INT_EN_GPIO0 INTV_GPIO0`

Definition at line 73 of file vs1000.h.

5.31.2.59 `#define INT_EN_GPIO1 INTV_GPIO1`

Definition at line 72 of file vs1000.h.

5.31.2.60 `#define INT_EN_NFLSH INTV_NFLSH`

Definition at line 79 of file vs1000.h.

5.31.2.61 `#define INT_EN_NONE 0`

Definition at line 71 of file vs1000.h.

5.31.2.62 `#define INT_EN_REGU INTV_REGU`

Definition at line 74 of file vs1000.h.

5.31.2.63 `#define INT_EN_RX INTV_RX`

Definition at line 77 of file vs1000.h.

5.31.2.64 `#define INT_EN_SPI INTV_SPI`

Definition at line 81 of file vs1000.h.

5.31.2.65 `#define INT_EN_TIM0 INTV_TIM0`

Definition at line 76 of file vs1000.h.

5.31.2.66 `#define INT_EN_TIM1 INTV_TIM1`

Definition at line 75 of file vs1000.h.

5.31.2.67 `#define INT_EN_TX INTV_TX`

Definition at line 78 of file vs1000.h.

5.31.2.68 `#define INT_EN_USB INTV_USB`

Definition at line 80 of file vs1000.h.

5.31.2.69 `#define INT_ENABLE 0xC070`

Definition at line 263 of file vs1000.h.

5.31.2.70 `#define INT_ENABLEH 0xC072`

Definition at line 265 of file vs1000.h.

5.31.2.71 `#define INT_ENABLEL 0xC070`

Definition at line 264 of file vs1000.h.

5.31.2.72 `#define INT_ENCOUNTER 0xC077`

Definition at line 268 of file vs1000.h.

5.31.2.73 `#define INT_GLOB_DIS 0xC078`

Definition at line 269 of file vs1000.h.

5.31.2.74 `#define INT_GLOB_EN 0xC079`

Definition at line 270 of file vs1000.h.

5.31.2.75 `#define INT_ORIGIN 0xC074`

Definition at line 266 of file vs1000.h.

5.31.2.76 `#define INT_VECTOR 0xC076`

Definition at line 267 of file vs1000.h.

5.31.2.77 `#define INTF_DAC (1<<INTV_DAC)`

Definition at line 69 of file vs1000.h.

5.31.2.78 `#define INTF_GPIO0 (1<<INTV_GPIO0)`

Definition at line 60 of file vs1000.h.

5.31.2.79 `#define INTF_GPIO1 (1<<INTV_GPIO1)`

Definition at line 59 of file vs1000.h.

5.31.2.80 `#define INTF_NFLSH (1<<INTV_NFLSH)`

Definition at line 66 of file vs1000.h.

5.31.2.81 `#define INTF_REGU (1<<INTV_REGU)`

Definition at line 61 of file vs1000.h.

5.31.2.82 `#define INTF_RX (1<<INTV_RX)`

Definition at line 64 of file vs1000.h.

5.31.2.83 `#define INTF_SPI (1<<INTV_SPI)`

Definition at line 68 of file vs1000.h.

5.31.2.84 `#define INTF_TIM0 (1<<INTV_TIM0)`

Definition at line 63 of file vs1000.h.

5.31.2.85 `#define INTF_TIM1 (1<<INTV_TIM1)`

Definition at line 62 of file vs1000.h.

5.31.2.86 `#define INTF_TX (1<<INTV_TX)`

Definition at line 65 of file vs1000.h.

5.31.2.87 `#define INTF_USB (1<<INTV_USB)`

Definition at line 67 of file vs1000.h.

5.31.2.88 `#define INTV_DAC 0`

Definition at line 57 of file vs1000.h.

5.31.2.89 `#define INTV_GPIO0 9`

Definition at line 48 of file vs1000.h.

5.31.2.90 `#define INTV_GPIO1 10`

Definition at line 47 of file vs1000.h.

5.31.2.91 #define INTV_NFLSH 3

Definition at line 54 of file vs1000.h.

5.31.2.92 #define INTV_REGU 8

Definition at line 49 of file vs1000.h.

5.31.2.93 #define INTV_RX 5

Definition at line 52 of file vs1000.h.

5.31.2.94 #define INTV_SPI 1

Definition at line 56 of file vs1000.h.

5.31.2.95 #define INTV_TIM0 6

Definition at line 51 of file vs1000.h.

5.31.2.96 #define INTV_TIM1 7

Definition at line 50 of file vs1000.h.

5.31.2.97 #define INTV_TX 4

Definition at line 53 of file vs1000.h.

5.31.2.98 #define INTV_USB 2

Definition at line 55 of file vs1000.h.

5.31.2.99 #define IRAM_SIZE 0x800

Definition at line 31 of file vs1000.h.

5.31.2.100 #define IRAM_START 0x0

Definition at line 30 of file vs1000.h.

5.31.2.101 #define IROM_SIZE 0x4a80

Definition at line 20 of file vs1000.h.

5.31.2.102 #define IROM_START 0x4000

Definition at line 19 of file vs1000.h.

5.31.2.103 `#define NFLSH_CF_INT_ENABLE (1<<7)`

Definition at line 240 of file vs1000.h.

5.31.2.104 `#define NFLSH_CF_LCD_CE_MODE (1<<8)`

Definition at line 239 of file vs1000.h.

5.31.2.105 `#define NFLSH_CF_NF_RESET (1<<6)`

Definition at line 241 of file vs1000.h.

5.31.2.106 `#define NFLSH_CF_WAITSTATES (1<<0)`

Definition at line 242 of file vs1000.h.

5.31.2.107 `#define NFLSH_CP_LPH 0xC062`

Definition at line 244 of file vs1000.h.

5.31.2.108 `#define NFLSH_CTRL 0xC060`

Definition at line 238 of file vs1000.h.

5.31.2.109 `#define NFLSH_DATA 0xC063`

Definition at line 245 of file vs1000.h.

5.31.2.110 `#define NFLSH_DB_POINTER (4)`

Definition at line 254 of file vs1000.h.

5.31.2.111 `#define NFLSH_DF_ECC_CALC (1<<1)`

Definition at line 258 of file vs1000.h.

5.31.2.112 `#define NFLSH_DF_ECC_RESET (1<<0)`

Definition at line 259 of file vs1000.h.

5.31.2.113 `#define NFLSH_DF_ENA_DBUF (1<<3)`

Definition at line 256 of file vs1000.h.

5.31.2.114 `#define NFLSH_DF_POINTER (1<<(NFLSH_DB_POINTER))`

Definition at line 255 of file vs1000.h.

5.31.2.115 `#define NFLSH_DF_READ (1<<2)`

Definition at line 257 of file vs1000.h.

5.31.2.116 `#define NFLSH_DSPIF 0xC065`

Definition at line 253 of file vs1000.h.

5.31.2.117 `#define NFLSH_ECC_CNT 0xC066`

Definition at line 260 of file vs1000.h.

5.31.2.118 `#define NFLSH_LPL 0xC061`

Definition at line 243 of file vs1000.h.

5.31.2.119 `#define NFLSH_NB_BYTECNT (8)`

Definition at line 247 of file vs1000.h.

5.31.2.120 `#define NFLSH_NF_BYTECNT (1<<NFLSH_NB_BYTECNT)`

Definition at line 248 of file vs1000.h.

5.31.2.121 `#define NFLSH_NF_POINTER (1<<2)`

Definition at line 250 of file vs1000.h.

5.31.2.122 `#define NFLSH_NF_READ (1<<0)`

Definition at line 252 of file vs1000.h.

5.31.2.123 `#define NFLSH_NF_START (1<<1)`

Definition at line 251 of file vs1000.h.

5.31.2.124 `#define NFLSH_NF_USE_DBUF (1<<7)`

Definition at line 249 of file vs1000.h.

5.31.2.125 `#define NFLSH_NFIF 0xC064`

Definition at line 246 of file vs1000.h.

5.31.2.126 `#define OTHERS_START 0x0800`

Definition at line 42 of file vs1000.h.

5.31.2.127 `#define PERIP(x) USEX(x)`

Definition at line 284 of file vs1000.h.

5.31.2.128 `#define PERIP_IN_X`

Definition at line 281 of file vs1000.h.

5.31.2.129 `#define SCI_DEBUG 0xC002`

Definition at line 112 of file vs1000.h.

5.31.2.130 `#define SCI_STATUS 0xC001`

Definition at line 91 of file vs1000.h.

5.31.2.131 `#define SCI_SYSTEM 0xC000`

Definition at line 86 of file vs1000.h.

5.31.2.132 `#define SCISTF_ANA_PDOWN (1<<2)`

Definition at line 108 of file vs1000.h.

5.31.2.133 `#define SCISTF_ANADRV_PDOWN (1<<3)`

Definition at line 107 of file vs1000.h.

5.31.2.134 `#define SCISTF_REGU_CLOCK (1<<1)`

Definition at line 109 of file vs1000.h.

5.31.2.135 `#define SCISTF_REGU_POWERBUT (1<<4)`

Definition at line 106 of file vs1000.h.

5.31.2.136 `#define SCISTF_REGU_POWERLOW (1<<5)`

Definition at line 105 of file vs1000.h.

5.31.2.137 `#define SCISTF_REGU_SHUTDOWN (1<<0)`

Definition at line 110 of file vs1000.h.

5.31.2.138 `#define SCISTF_SLOW_CLKMODE (1<<15)`

Definition at line 93 of file vs1000.h.

5.31.2.139 #define SCISTF_USB_DDR (1<<12)

Definition at line 96 of file vs1000.h.

5.31.2.140 #define SCISTF_USB_DIFF_ENA (1<<7)

Definition at line 103 of file vs1000.h.

5.31.2.141 #define SCISTF_USB_DN (1<<8)

Definition at line 102 of file vs1000.h.

5.31.2.142 #define SCISTF_USB_DN_OUT (1<<14)

Definition at line 94 of file vs1000.h.

5.31.2.143 #define SCISTF_USB_DP (1<<9)

Definition at line 101 of file vs1000.h.

5.31.2.144 #define SCISTF_USB_DP_OUT (1<<13)

Definition at line 95 of file vs1000.h.

5.31.2.145 #define SCISTF_USB_PULLUP_ENA (1<<6)

Definition at line 104 of file vs1000.h.

5.31.2.146 #define SCISTF_VCM_DISABLE (1<<10)

Definition at line 99 of file vs1000.h.

5.31.2.147 #define SCISTF_VCM_OVERLOAD (1<<11)

Definition at line 98 of file vs1000.h.

5.31.2.148 #define SCISYSF_AVDD (1<<10)

Definition at line 88 of file vs1000.h.

5.31.2.149 #define SCISYSF_CLKDIV 0x8000

Definition at line 87 of file vs1000.h.

5.31.2.150 #define SCISYSF_CVDD (1<<0)

Definition at line 90 of file vs1000.h.

5.31.2.151 #define SCISYSF_IOVDD (1<<5)

Definition at line 89 of file vs1000.h.

5.31.2.152 #define SPI0_CLKCONFIG 0xC069

Definition at line 207 of file vs1000.h.

5.31.2.153 #define SPI0_CONFIG 0xC068

Definition at line 206 of file vs1000.h.

5.31.2.154 #define SPI0_DATA 0xC06B

Definition at line 209 of file vs1000.h.

5.31.2.155 #define SPI0_FSYNC 0xC06C

Definition at line 210 of file vs1000.h.

5.31.2.156 #define SPI0_STATUS 0xC06A

Definition at line 208 of file vs1000.h.

5.31.2.157 #define SPI_CC_CLKDIV (1<<2)

Definition at line 228 of file vs1000.h.

5.31.2.158 #define SPI_CF_DLEN (1<<1)

Definition at line 217 of file vs1000.h.

5.31.2.159 #define SPI_CF_DLEN16 (15<<1)

Definition at line 219 of file vs1000.h.

5.31.2.160 #define SPI_CF_DLEN8 (7<<1)

Definition at line 218 of file vs1000.h.

5.31.2.161 #define SPI_CF_FALLXCS (2<<6)

Definition at line 213 of file vs1000.h.

5.31.2.162 #define SPI_CF_FSIDLE0 (0<<0)

Definition at line 221 of file vs1000.h.

5.31.2.163 `#define SPI_CF_FSIDLE1 (1<<0)`

Definition at line 220 of file vs1000.h.

5.31.2.164 `#define SPI_CF_INTXCS (0<<6)`

Definition at line 212 of file vs1000.h.

5.31.2.165 `#define SPI_CF_MASTER (1<<5)`

Definition at line 215 of file vs1000.h.

5.31.2.166 `#define SPI_CF_RISEXCS (3<<6)`

Definition at line 214 of file vs1000.h.

5.31.2.167 `#define SPI_CF_SLAVE (0<<5)`

Definition at line 216 of file vs1000.h.

5.31.2.168 `#define SPI_ST_BREAK (1<<5)`

Definition at line 230 of file vs1000.h.

5.31.2.169 `#define SPI_ST_RXFULL (1<<3)`

Definition at line 232 of file vs1000.h.

5.31.2.170 `#define SPI_ST_RXORUN (1<<4)`

Definition at line 231 of file vs1000.h.

5.31.2.171 `#define SPI_ST_TXFULL (1<<2)`

Definition at line 233 of file vs1000.h.

5.31.2.172 `#define SPI_ST_TXRUNNING (1<<1)`

Definition at line 234 of file vs1000.h.

5.31.2.173 `#define SPI_ST_TXURUN (1<<0)`

Definition at line 235 of file vs1000.h.

5.31.2.174 `#define STACK_SIZE 0x200`

Definition at line 40 of file vs1000.h.

5.31.2.175 `#define STACK_START 0x1800`

Definition at line 39 of file vs1000.h.

5.31.2.176 `#define TIMER_CONFIG 0xC030`

Definition at line 164 of file vs1000.h.

5.31.2.177 `#define TIMER_ENABLE 0xC031`

Definition at line 165 of file vs1000.h.

5.31.2.178 `#define TIMER_T0CNTH 0xC037`

Definition at line 169 of file vs1000.h.

5.31.2.179 `#define TIMER_T0CNTL 0xC036`

Definition at line 168 of file vs1000.h.

5.31.2.180 `#define TIMER_T0H 0xC035`

Definition at line 167 of file vs1000.h.

5.31.2.181 `#define TIMER_T0L 0xC034`

Definition at line 166 of file vs1000.h.

5.31.2.182 `#define TIMER_T1CNTH 0xC03B`

Definition at line 173 of file vs1000.h.

5.31.2.183 `#define TIMER_T1CNTL 0xC03A`

Definition at line 172 of file vs1000.h.

5.31.2.184 `#define TIMER_T1H 0xC039`

Definition at line 171 of file vs1000.h.

5.31.2.185 `#define TIMER_T1L 0xC038`

Definition at line 170 of file vs1000.h.

5.31.2.186 `#define UART_DATA 0xC029`

Definition at line 154 of file vs1000.h.

5.31.2.187 `#define UART_DATAH 0xC02A`

Definition at line 155 of file vs1000.h.

5.31.2.188 `#define UART_DIV 0xC02B`

Definition at line 156 of file vs1000.h.

5.31.2.189 `#define UART_ST_RXFULL (1<<2)`

Definition at line 159 of file vs1000.h.

5.31.2.190 `#define UART_ST_RXORUN (1<<3)`

Definition at line 158 of file vs1000.h.

5.31.2.191 `#define UART_ST_TXFULL (1<<1)`

Definition at line 160 of file vs1000.h.

5.31.2.192 `#define UART_ST_TXRUNNING (1<<0)`

Definition at line 161 of file vs1000.h.

5.31.2.193 `#define UART_STATUS 0xC028`

Definition at line 153 of file vs1000.h.

5.31.2.194 `#define USB_BASE 0xC080U`

Definition at line 275 of file vs1000.h.

5.31.2.195 `#define USB_RECV_MEM 0x2C00`

USB receive memory, ring buffer.

Definition at line 279 of file vs1000.h.

5.31.2.196 `#define USB_SEND_MEM 0x3000`

USB send memory

Definition at line 280 of file vs1000.h.

5.31.2.197 `#define WDOG_CONFIG 0xC020`

Definition at line 148 of file vs1000.h.

5.31.2.198 `#define WDOG_DUMMY 0xC022`

Definition at line 150 of file vs1000.h.

5.31.2.199 `#define WDOG_RESET 0xC021`

Definition at line 149 of file vs1000.h.

5.31.2.200 `#define WDOG_RESET_VAL 0x4ea9`

Definition at line 151 of file vs1000.h.

5.31.2.201 `#define XRAM_SIZE 0x3400`

Definition at line 27 of file vs1000.h.

5.31.2.202 `#define XRAM_START 0x0`

Definition at line 26 of file vs1000.h.

5.31.2.203 `#define XROM_SIZE 0x0c00`

Definition at line 22 of file vs1000.h.

5.31.2.204 `#define XROM_START 0x4000`

Definition at line 21 of file vs1000.h.

5.31.2.205 `#define YPREV0_START 0x0000`

Definition at line 43 of file vs1000.h.

5.31.2.206 `#define YPREV1_START 0x0400`

Definition at line 44 of file vs1000.h.

5.31.2.207 `#define YRAM_SIZE 0x4000`

Definition at line 29 of file vs1000.h.

5.31.2.208 `#define YRAM_START 0x0`

Definition at line 28 of file vs1000.h.

5.31.2.209 `#define YROM_SIZE 0x2800`

Definition at line 24 of file vs1000.h.

5.31.2.210 `#define YROM_START 0x4000`

Definition at line 23 of file vs1000.h.

5.31.3 Enumeration Type Documentation

5.31.3.1 enum voltIdx

enumerations for voltage array members.

Enumerator:

voltCorePlayer core voltage in player mode
voltIoPlayer IO voltage in player mode
voltAnaPlayer Analog voltage in player mode
voltCoreUSB core voltage in USB mode
voltIoUSB IO voltage in USB mode
voltAnaUSB Analog voltage in USB mode
voltCoreSuspend core voltage in suspend/low-power pause mode
voltIoSuspend IO voltage in suspend/low-power pause mode
voltAnaSuspend Analog voltage in suspend/low-power pause mode
voltCoreUser core voltage in user-defined mode
voltIoUser IO voltage in user-defined mode
voltAnaUser Analog voltage in user-defined mode
voltEnd

Definition at line 401 of file vs1000.h.

```

401     {
402     /* for VS1000B */
403     voltCorePlayer = 0,
404     voltIoPlayer,
405     voltAnaPlayer,
406     voltCoreUSB,
407     voltIoUSB,
408     voltAnaUSB,
409     voltCoreSuspend,
410     voltIoSuspend,
411     voltAnaSuspend,
412     voltCoreUser,
413     voltIoUser,
414     voltAnaUser,
415     voltEnd
416 };

```

5.31.4 Function Documentation

5.31.4.1 void BootFromX (register __i0 u_int16 * start)

Reads and handles I, X, Y, and execute records from X memory.

5.31.4.2 void BusyWait10 (void)

Waits 120000 cycles, which is 10ms if clock is 1.0x.

5.31.4.3 u_int16 DefUnsupportedFile (struct CodecServices * cs)

Called when file is not Ogg Vorbis. Dummy function.

5.31.4.4 void Disable (void)

Disable interrupts. Call **Enable()**(p.175) an equal number of time to enable interrupts. Should only be used for critical code sections when other exclusion methods are not possible.

5.31.4.5 void Enable (void)

Enable interrupts.

5.31.4.6 struct FsMapper* FsMapRamCreate (struct FsPhysical * physical, u_int16 cacheSize)

Creates a mapper for a RAM disk using mallocAreaY.

Parameters:

physical is a pointer to mallocAreaY

cacheSize not used, RAM disk is always 36 sectors.

Returns:

pointer to mapper structure

5.31.4.7 s_int16 getch (void)

raw polled UART receive

5.31.4.8 void IdleHook (void)

Hook: Called by **Sleep()**(p.178) before halt state is entered. Default: **UserInterfaceIdleHook()**(p.101).

5.31.4.9 auto u_int16 InitFileSystem (void)

Hook: Initializes filesystem. Default: **FatInitFileSystem()**(p.92).

Returns:

0 for success.

5.31.4.10 void LoadCheck (struct CodecServices * cs, s_int16 n)

Hook: Decreases or increases the system clock. Default: **RealLoadCheck()**(p.177).

5.31.4.11 `auto u_int16 MapperReadDiskSector (register __i0 u_int16 * buffer, register __a u_int32 sector)`

Reads one sector through the mapper interface (`map->Read()`).

Parameters:

buffer the buffer for the sector data.

sector the sector to read.

5.31.4.12 `void MemTests (register short __b0 muxTestResult)`

5.31.4.13 `void NullHook (void)`

IdleHook that does nothing.

5.31.4.14 `auto s_int16 OpenFile (register __c0 u_int16 fileNum)`

Hook: Opens a specified file. Default: `FatOpenFile()`(p. 92).

Parameters:

fileNum the number of the file to open.

Returns:

-1 for success, the number of files otherwise.

5.31.4.15 `u_int16 PlayCurrentFile (void)`

Hook: play the currently open file. Default: `RealPlayCurrentFile()`(p. 177).

5.31.4.16 `void PowerOff (void)`

Hook: Turns power off. Default: `RealPowerOff()`(p. 177).

5.31.4.17 `void PowerSetVoltages (u_int16 volt[3])`

Sets voltages according to parameter values.

5.31.4.18 `void putch (register __a0 s_int16 ch)`

raw polled UART send

5.31.4.19 `auto u_int16 ReadDiskSector (register __i0 u_int16 * buffer, register __a u_int32 sector)`

Hook: Read a sector. Default: `MapperReadDiskSector()`(p. 176).

Parameters:

buffer the buffer for the sector data.

sector the sector to read.

5.31.4.20 `auto s_int16 ReadFile (register __i3 u_int16 * buf, register __c1 s_int16 byteOff, register __c0 s_int16 byteSize)`

5.31.4.21 `void RealLoadCheck (struct CodecServices * cs, s_int16 n)`

5.31.4.22 `u_int16 RealPlayCurrentFile (void)`

Tries to play the current file. Turns on maximum player mode clock (by default 3.5x), and tries to decode the file with the Ogg Vorbis decoder. After decoding has ended feeds zeros to audio buffer to ensure earspeaker echo has decayed. If decoding returned with `ceFormatNotFound`, calls the `UnsupportedFile` hook. Before returning turns on maximum player mode clock to speed up locating the next file.

5.31.4.23 `void RealPowerOff (void)`

Power off routine. Sets 1.0x mode (PLL off), 100Hz samplerate, disables interrupts, turns off analog drivers and LED's. Then waits until the power button is released. Then shuts off the regulators.

5.31.4.24 `void Restart (void)`

5.31.4.25 `u_int32 Seek (register __reg_a u_int32 pos)`

Hook: Change the read position of a file. Default: `FatSeek()`(p.93).

Parameters:

pos Byte position to find.

Returns:

the FAT sector that corresponds to the pos. Sets pos, returns old pos

5.31.4.26 `void* SetHookFunction (register __i0 u_int16 hook, register __a0 void * newFunc)`

Sets new IRAM Hook function.

Parameters:

hook The address of the Hook in IRAM

newFunc The new hook function address.

Returns:

The old hook function address is returned.

5.31.4.27 void SinTest (void)

5.31.4.28 void Sleep (void)

Call the idle hook, then wait for the next interrupt (HALT mode).

5.31.4.29 void SpiBoot (register `__a0` short *clkConf*, register `__i2` short *addr*, register `__i0` short *m24*)

5.31.4.30 void SpiDelay (register `__a0` u_int16 *wait*)

5.31.4.31 void SpiLoad (register `__i2` short *startAddr*, register `__i0` short *m24*)

5.31.4.32 auto u_int16 SpiSendReceive (register `__a0` u_int16 *data*)

5.31.4.33 u_int32 Tell (void)

Hook: Return the current read position. Default: `FatTell()`(p.93).

Returns:

Current read position. Gets pos

5.31.4.34 u_int16 UnsupportedFile (struct CodecServices * *cs*)

Hook: called when file is not Ogg Vorbis. Default: `DefUnsupportedFile()`(p.175).

5.31.5 Variable Documentation

5.31.5.1 `__y` u_int16 `g_dcthi`[2048]

5.31.5.2 u_int16 `g_dctlo`[2048]

5.31.5.3 s_int16 `g_others`[2048]

5.31.5.4 s_int16 `g_yprev0`[1024]

5.31.5.5 s_int16 `g_yprev1`[1024]

5.31.5.6 u_int16 `voltages`[`voltEnd`]

5.32 vsasm.h File Reference

Defines

- `#define MAKEMOD64(stp, bufisz) (0x4000|(((bufisz)/64-1)&31)|(((stp)&255)<<6))`
- `#define MAKEMOD(stp, bufisz) (0x2000|(((stp)&127)<<6)|(((bufisz)-1)&63))`
- `#define MAKEMODF(bufisz) (0x8000|(((bufisz)-1)&8191))`
- `#define MAKEMODB(bufisz) (0xA000|(((bufisz)-1)&8191))`

5.32.1 Detailed Description

Standard VSDSP assembler macros.

Definition in file `vsasm.h`.

5.32.2 Define Documentation

5.32.2.1 `#define MAKEMOD(stp, bufisz) (0x2000|(((stp)&127)<<6)|(((bufisz)-1)&63))`

Creates a modifier register with *step* = -64..63, *bufisz* = 1..64.

Definition at line 14 of file `vsasm.h`.

5.32.2.2 `#define MAKEMOD64(stp, bufisz) (0x4000|(((bufisz)/64-1)&31)|(((stp)&255)<<6))`

Creates a modifier register with *step* = -128..127, *bufisz* N*64

Definition at line 12 of file `vsasm.h`.

5.32.2.3 `#define MAKEMODB(bufisz) (0xA000|(((bufisz)-1)&8191))`

Creates a backwards modifier register, *bufisz* = 1..8192.

Definition at line 18 of file `vsasm.h`.

5.32.2.4 `#define MAKEMODF(bufisz) (0x8000|(((bufisz)-1)&8191))`

Creates a forwards modifier register, *bufisz* = 1..8192.

Definition at line 16 of file `vsasm.h`.

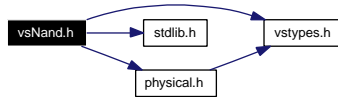
5.33 vsNand.h File Reference

```
#include <vstypes.h>
```

```
#include <stdlib.h>
```

```
#include <physical.h>
```

Include dependency graph for vsNand.h:



Data Structures

- struct **FsNandPhys**

Defines

- #define **NAND_OP_READ_SIGNATURE** 0x90
- #define **NAND_OP_READ_STATUS** 0x70
- #define **NAND_OP_READ_A** 0x00
- #define **NAND_OP_READ_C** 0x50
- #define **NAND_OP_COMMIT_DATA_ADDRESS** 0x30
- #define **NAND_OP_PREPARE_TO_PROGRAM** 0x80
- #define **NAND_OP_PERFORM_PROGRAM** 0x10

Functions

- **s_int16 FsPhNandErase** (struct **FsPhysical** *p, **s_int32** block)
- **FsPhysical * FsPhNandCreate** (**u_int16** param)
- **s_int16 FsPhNandDelete** (struct **FsPhysical** *p)
- **s_int16 FsPhNandFreeBus** (struct **FsPhysical** *p)
- **s_int16 FsPhNandReinitialize** (struct **FsPhysical** *p)
- **s_int16 FsPhNandRead** (struct **FsPhysical** *p, **s_int32** firstPage, **u_int16** pages, **u_int16** *data, **u_int16** *meta)
- **s_int16 FsPhNandWrite** (struct **FsPhysical** *p, **s_int32** firstPage, **u_int16** pages, **u_int16** *data, **u_int16** *meta)
- **s_int16 NandCountBits** (register __a **u_int32** val)
- **s_int16 NandMingle** (register __a **u_int32** val)
- **void NandSwapBad** (register __i0 **u_int16** *spare)
- **void NandWaitIdle** (void)

5.33.1 Detailed Description

NAND FLASH routines.

Definition in file **vsNand.h**.

5.33.2 Define Documentation

5.33.2.1 `#define NAND_OP_COMMIT_DATA_ADDRESS 0x30`

Nand Flash Opcode: Commit read address

Definition at line 24 of file vsNand.h.

5.33.2.2 `#define NAND_OP_PERFORM_PROGRAM 0x10`

Nand Flash Opcode: Execute Programming

Definition at line 30 of file vsNand.h.

5.33.2.3 `#define NAND_OP_PREPARE_TO_PROGRAM 0x80`

Nand Flash Opcode: Prepare to Program

Definition at line 27 of file vsNand.h.

5.33.2.4 `#define NAND_OP_READ_A 0x00`

Nand Flash Opcode: Read A - data area

Definition at line 18 of file vsNand.h.

5.33.2.5 `#define NAND_OP_READ_C 0x50`

Nand Flash Opcode: Read C - spare area

Definition at line 21 of file vsNand.h.

5.33.2.6 `#define NAND_OP_READ_SIGNATURE 0x90`

Nand Flash Opcode: Read Signature

Definition at line 12 of file vsNand.h.

5.33.2.7 `#define NAND_OP_READ_STATUS 0x70`

Nand Flash Opcode: Read Signature

Definition at line 15 of file vsNand.h.

5.33.3 Function Documentation

5.33.3.1 `struct FsPhysical* FsPhNandCreate (u_int16 param)`

Creates a physical layer.

5.33.3.2 s_int16 FsPhNandDelete (struct FsPhysical * p)

Free resources allocated by FsPhNandCreate and release HW

5.33.3.3 s_int16 FsPhNandErase (struct FsPhysical * p, s_int32 block)

Erase the block that starts from page "block".

5.33.3.4 s_int16 FsPhNandFreeBus (struct FsPhysical * p)

Free hardware bus for possible other devices

5.33.3.5 s_int16 FsPhNandRead (struct FsPhysical * p, s_int32 firstPage, u_int16 pages, u_int16 * data, u_int16 * meta)

Read pages. If meta is non-NULL, will use error correction. To read both sector and spare areas without error correction, you need two reads: first read only sector datas (meta = NULL), then read only spares (data = NULL).

5.33.3.6 s_int16 FsPhNandReinitialize (struct FsPhysical * p)

Reinitialize bus

5.33.3.7 s_int16 FsPhNandWrite (struct FsPhysical * p, s_int32 firstPage, u_int16 pages, u_int16 * data, u_int16 * meta)

Write pages. If meta is non-NULL, will generate error correction.

5.33.3.8 s_int16 NandCountBits (register __a u_int32 val)

Support: Count the number of 1-bits

5.33.3.9 s_int16 NandMingle (register __a u_int32 val)

Support: takes every other bit from the value.

5.33.3.10 void NandSwapBad (register __i0 u_int16 * spare)

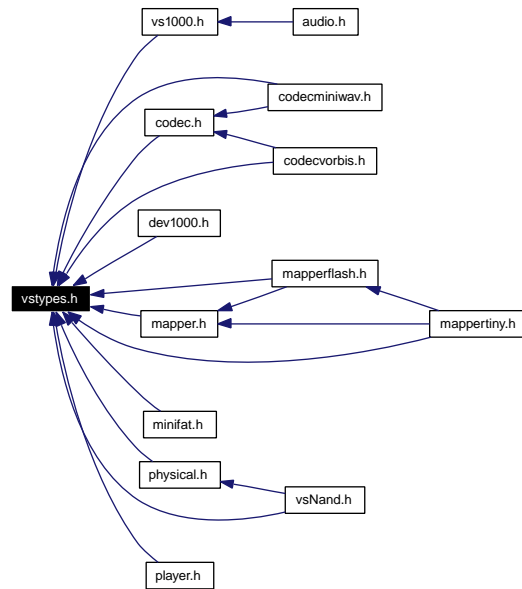
Support: reorder large-page bad block indicators to be compatible with small-page bad block indicators.

5.33.3.11 void NandWaitIdle (void)

Support: Waits until the NAND FLASH interface is idle.

5.34 vstypes.h File Reference

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



Defines

- #define **MR_NONE** 0
- #define **MR_INT** 0x200
- #define **MR_SAT** 0x400
- #define **FMT_S16** "d"
- #define **FMT_U16** "u"
- #define **FMT_H16** "x"
- #define **FMT_S32** "d"
- #define **FMT_U32** "u"
- #define **FMT_H32** "x"
- #define **register**
- #define **__reg_a**
- #define **__reg_b**
- #define **__reg_c**
- #define **__reg_d**
- #define **__a0**
- #define **__a1**
- #define **__b0**
- #define **__b1**
- #define **__c0**
- #define **__c1**
- #define **__d0**

- #define `__d1`
- #define `__i0`
- #define `__i1`
- #define `__i2`
- #define `__i3`
- #define `__mem_x`
- #define `__y`
- #define `__near`
- #define `__far`
- #define `auto`
- #define `FMUL32(a, b) (f_int32)((long long)(f_int32)(a) * (f_int32)(b) >> 31)`
- #define `FMUL16(a, b) (f_int16)((long)(f_int16)(a) * (f_int16)(b) >> 15)`
- #define `FDIV32(a, b) (f_int32)(((long long)(f_int32)(a)<<31) / (f_int32)(b))`
- #define `FDIV16(a, b) (f_int16)(((long)(f_int16)(a)<<15) / (f_int16)(b))`
- #define `DBL2F32(a) (f_int32)((a)*(32768.0*65536.0))`
- #define `DBL2F16(a) (f_int16)((a)*32768.0)`
- #define `SqrtI(a) ((u_int16)sqrt(a))`
- #define `SqrtI32(a) ((u_int32)(sqrt(a)*65536.0))`
- #define `USEX(x) *((__x volatile u_int16 *) (u_int16)(x))`
- #define `USEY(x) *((__y volatile u_int16 *) (u_int16)(x))`

Typedefs

- typedef short `s_int16`
- typedef unsigned short `u_int16`
- typedef int `s_int32`
- typedef unsigned int `u_int32`
- typedef `s_int16 f_int16`
- typedef `s_int32 f_int32`

5.34.1 Detailed Description

Standard VSDSP types and definitions. This standard file contains standard definitions and types. It also makes it easier to compile the same source code with both VSDSP and Unix compilers by undefining symbols that are only understood by VSDSP C compilers.

Definition in file `vstypes.h`.

5.34.2 Define Documentation

5.34.2.1 #define `__a0`

Definition at line 62 of file `vstypes.h`.

5.34.2.2 #define __a1

Definition at line 63 of file vstypes.h.

5.34.2.3 #define __b0

Definition at line 64 of file vstypes.h.

5.34.2.4 #define __b1

Definition at line 65 of file vstypes.h.

5.34.2.5 #define __c0

Definition at line 66 of file vstypes.h.

5.34.2.6 #define __c1

Definition at line 67 of file vstypes.h.

5.34.2.7 #define __d0

Definition at line 68 of file vstypes.h.

5.34.2.8 #define __d1

Definition at line 69 of file vstypes.h.

5.34.2.9 #define __far

Definition at line 77 of file vstypes.h.

5.34.2.10 #define __i0

Definition at line 70 of file vstypes.h.

5.34.2.11 #define __i1

Definition at line 71 of file vstypes.h.

5.34.2.12 #define __i2

Definition at line 72 of file vstypes.h.

5.34.2.13 #define __i3

Definition at line 73 of file vstypes.h.

5.34.2.14 `#define __mem_x`
Definition at line 74 of file vstypes.h.

5.34.2.15 `#define __near`
Definition at line 76 of file vstypes.h.

5.34.2.16 `#define __reg_a`
Definition at line 58 of file vstypes.h.

5.34.2.17 `#define __reg_b`
Definition at line 59 of file vstypes.h.

5.34.2.18 `#define __reg_c`
Definition at line 60 of file vstypes.h.

5.34.2.19 `#define __reg_d`
Definition at line 61 of file vstypes.h.

5.34.2.20 `#define __y`
Definition at line 75 of file vstypes.h.

5.34.2.21 `#define auto`
Definition at line 78 of file vstypes.h.

5.34.2.22 `#define DBL2F16(a) (f_int16)((a)*32768.0)`
16-bit fractional double-to-fract conversion (Unix version).
Definition at line 114 of file vstypes.h.

5.34.2.23 `#define DBL2F32(a) (f_int32)((a)*(32768.0*65536.0))`
32-bit fractional double-to-fract conversion (Unix version).
Definition at line 112 of file vstypes.h.

5.34.2.24 `#define FDIV16(a, b) (f_int16)((((long)(f_int16)(a)<<15) / (f_int16)(b))`
16-bit fractional division (Unix version).
Definition at line 110 of file vstypes.h.

5.34.2.25 `#define FDIV32(a, b) (f_int32)(((long long)(f_int32)(a)<<31) / (f_int32)(b))`

32-bit fractional division (Unix version).

Definition at line 108 of file vtypes.h.

5.34.2.26 `#define FMT_H16 "x"`

`printf()`(p. 117) format for 16-bit hex value

Definition at line 38 of file vtypes.h.

5.34.2.27 `#define FMT_H32 "x"`

`printf()`(p. 117) format for 32-bit hex value, UNIX version

Definition at line 53 of file vtypes.h.

5.34.2.28 `#define FMT_S16 "d"`

`printf()`(p. 117) format for signed 16-bit value

Definition at line 34 of file vtypes.h.

5.34.2.29 `#define FMT_S32 "d"`

`printf()`(p. 117) format for signed 32-bit value, UNIX version

Definition at line 49 of file vtypes.h.

5.34.2.30 `#define FMT_U16 "u"`

`printf()`(p. 117) format for unsigned 16-bit value

Definition at line 36 of file vtypes.h.

5.34.2.31 `#define FMT_U32 "u"`

`printf()`(p. 117) format for unsigned 32-bit value, UNIX version

Definition at line 51 of file vtypes.h.

5.34.2.32 `#define FMUL16(a, b) (f_int16)((long)(f_int16)(a) * (f_int16)(b) >> 15)`

16-bit fractional multiplication (Unix version).

Definition at line 106 of file vtypes.h.

5.34.2.33 `#define FMUL32(a, b) (f_int32)((long long)(f_int32)(a) * (f_int32)(b) >> 31)`

32-bit fractional multiplication (Unix version).

Definition at line 104 of file vstypes.h.

5.34.2.34 `#define MR_INT 0x200`

Mode register: Integer mode

Definition at line 17 of file vstypes.h.

5.34.2.35 `#define MR_NONE 0`

Mode register: Zero is fractional, non-saturated mode

Definition at line 15 of file vstypes.h.

5.34.2.36 `#define MR_SAT 0x400`

Mode register: Saturation mode

Definition at line 19 of file vstypes.h.

5.34.2.37 `#define register`

Definition at line 57 of file vstypes.h.

5.34.2.38 `#define SqrtI(a) ((u_int16)sqrt(a))`

Definition at line 119 of file vstypes.h.

5.34.2.39 `#define SqrtI32(a) ((u_int32)(sqrt(a)*65536.0))`

Definition at line 120 of file vstypes.h.

5.34.2.40 `#define USEX(x) *((__x volatile u_int16 *) (u_int16)(x))`

Direct memory write to X memory

Definition at line 157 of file vstypes.h.

5.34.2.41 `#define USEY(x) *((__y volatile u_int16 *) (u_int16)(x))`

Direct memory write to Y memory

Definition at line 161 of file vstypes.h.

5.34.3 Typedef Documentation

5.34.3.1 `typedef s_int16 f_int16`

Default 16-bit fractional type. This is the Non-VSDSP definition. The VSDSP definition reads `typedef __fract short f_int16`. Addition and subtraction can be used as normal with this type, but some operations must be executed using macros: **FMUL16()**(p. 187) for multiplication, **FDIV16()**(p. 186) for division and **DBL2F16()**(p. 186) for double-to-fract conversion.

Definition at line 93 of file `vstypes.h`.

5.34.3.2 `typedef s_int32 f_int32`

Default 32-bit fractional type. This is the Non-VSDSP definition. The VSDSP definition reads `typedef __fract long f_int32`. Addition and subtraction can be used as normal with this type, but some operations must be executed using macros: **FMUL32()**(p. 187) for multiplication, **FDIV32()**(p. 187) for division and **DBL2F32()**(p. 186) for double-to-fract conversion.

Definition at line 102 of file `vstypes.h`.

5.34.3.3 `typedef short s_int16`

Default signed 16-bit integer type

Definition at line 29 of file `vstypes.h`.

5.34.3.4 `typedef int s_int32`

Default signed 32-bit integer type

Definition at line 44 of file `vstypes.h`.

5.34.3.5 `typedef unsigned short u_int16`

Default unsigned 16-bit integer type

Definition at line 31 of file `vstypes.h`.

5.34.3.6 `typedef unsigned int u_int32`

Default unsigned 32-bit integer type

Definition at line 46 of file `vstypes.h`.

6 VS1000/Interfacing Example Documentation

6.1 LoadCheck

Decreases or increases the system clock. Also handles Replay Gain. If `cs == NULL` turns on maximum allowed clock (if `n` is zero, `player.maxClock` for player mode, otherwise 4.0x for USB mode). Otherwise audio underflows or too empty audio buffer (<256 stereo samples) cause an increase in clock, and too much idle time

Parameters:

- cs* **CodecServices**(p.6) struct pointer or NULL.
- n* The number of samples or 0 or 1. (NULL, 0); will turn on maximum player-mode clock.

6.2 tmpBuf

Reads bytes from the current file position.

Parameters:

- buf* Packed buffer to read bytes to.
- byteOff* Packed byte offset of the first byte. Even = high part of word, odd = low part of word.
- byteSize* The number of bytes to read.

Returns:

the number of bytes actually read. [0] = 0; FatReadFile(tmpBuf, 1, 1); reads one byte to the low part of tmpBuf[0].

7 VS1000/Interfacing Page Documentation

7.1 Bug List

Global USBSendZeroLengthPacketToEndpoint0(p.147) : currently is satisfied if any active IN is finished.

Global USBWantsSuspend(p.148) USB suspend time is 3ms. Because of our software timer accuracy we detect suspend when there has been no SOF for 10..20ms.

This routine requires that the unit is configured before returning non-zero.

Returns:

0 if the communication is proceeding normally, non-zero if suspend state is detected.

Index

- __a0
 - vstypes.h, 184
- __a1
 - vstypes.h, 184
- __b0
 - vstypes.h, 185
- __b1
 - vstypes.h, 185
- __c0
 - vstypes.h, 185
- __c1
 - vstypes.h, 185
- __d0
 - vstypes.h, 185
- __d1
 - vstypes.h, 185
- __far
 - vstypes.h, 185
- __i0
 - vstypes.h, 185
- __i1
 - vstypes.h, 185
- __i2
 - vstypes.h, 185
- __i3
 - vstypes.h, 185
- __mem_x
 - vstypes.h, 185
- __near
 - vstypes.h, 186
- __reg_a
 - vstypes.h, 186
- __reg_b
 - vstypes.h, 186
- __reg_c
 - vstypes.h, 186
- __reg_d
 - vstypes.h, 186
- __sfpos, 4
- __pos, 4
- __y
 - vstypes.h, 186
- _divide16signed
 - math.h, 87
- _divide16unsigned
 - math.h, 87
- _modf
 - math.h, 87
- _oldmodf
 - math.h, 86
- _pos
 - __sfpos, 4
- abort
 - stdlib.h, 119
- abs
 - stdlib.h, 119
- acos
 - math.h, 87
- allocationLength
 - scsicdb6variant1, 29
- APPL_AUDIO
 - audio.h, 36
- APPL_BITSTREAM
 - audio.h, 36
- APPL_RESET
 - audio.h, 36
- applAddr
 - audio.h, 39
- asin
 - math.h, 87
- assert
 - assert.h, 35
- assert.h, 34
 - assert, 35
- atan
 - math.h, 87
- atan2
 - math.h, 87
- ATAPI_READ_FORMAT_-
CAPACITIES
 - scsi.h, 106
- atoi
 - stdlib.h, 120
- audio.h, 35
 - APPL_AUDIO, 36
 - APPL_BITSTREAM, 36
 - APPL_RESET, 36
 - applAddr, 39
 - audioBuffer, 39
 - AudioBufFill, 37
 - AudioBufFree, 37
 - AudioOutputSamples, 37
 - audioPtr, 39

- bassReg, 39
- clockX, 39
- curFctl, 39
- DAC_DEFAULT_-
 SAMPLERATE, 36
- DAC_DRIVER_ON_DELAY,
 37
- DEFAULT_AUDIO_-
 BUFFER_SAMPLES,
 37
- DIRECT_VORBIS_-
 BLOCKSIZE, 37
- earSpeaker, 40
- earSpeakerDisable, 40
- earSpeakerReg, 40
- extClock4KHz, 40
- haltTime, 40
- hwSampleRate, 40
- InitAudio, 38
- ReadTimeCount, 38
- RealSetRate, 38
- RealSetVolume, 38
- SetRate, 38
- SetVolume, 38
- StereoCopy, 38
- timeCount, 40
- TIMER_TICKS, 37
- timeToRemovePDown2, 40
- uartByteSpeed, 40
- UartDiv, 38
- uiTime, 40
- uiTrigger, 40
- USE_TIMER, 37
- volumeReg, 40
- WITH_EARSPEAKER, 37
- AUDIO_DELAY_FRAMES
 usbblowlib.h, 138
- AUDIO_DELAY_FRAMES_STR
 usbblowlib.h, 138
- AUDIO_ISOC_OUT_EP
 usbblowlib.h, 138
- AUDIO_START
 vs1000.h, 156
- audioBuffer
 audio.h, 39
- AudioBuffFill
 audio.h, 37
- AudioBuffFree
 audio.h, 37
- AudioOutputSamples
 audio.h, 37
- AudioPacketFromUSB
 usbblowlib.h, 144
- AUDIOPTR, 4
 forwardModulo, 5
 leftVol, 5
 rd, 5
 rightVol, 5
 underflow, 5
 wr, 5
- audioPtr
 audio.h, 39
- auto
 vstypes.h, 186
- avgBitRate
 CodecServices, 7
- bassReg
 audio.h, 39
- blocks
 FsMapper, 16
- blockSize
 FsMapper, 16
- blocksPerErase
 FsMapperFlash, 18
 FsMapperTiny, 20
- BootFromX
 vs1000.h, 174
- BPB_RootEntCnt
 FATINFO, 12
- BRESET_INT
 usbblowlib.h, 138
- BusyWait10
 vs1000.h, 174
- c-nand.s, 41
- c-restart.s, 41
- c-spi.s, 41
- cache
 FsMapperFlash, 18
- cacheBlocks
 FsMapper, 16
- cancel
 CodecServices, 7
- ceCancelled
 codec.h, 46
- ceFastForward
 codec.h, 46
- ceFormatNotFound
 codec.h, 46

- ceFormatNotSupported
 - codec.h, 46
- ceil
 - math.h, 87
- ceOk
 - codec.h, 46
- ceOtherError
 - codec.h, 46
- ceUnexpectedFileEnd
 - codec.h, 46
- channels
 - CodecServices, 7
- CleanDisk
 - player.h, 99
- clearerr
 - stdio.h, 116
- clockX
 - audio.h, 39
- Codec, 5
 - Create, 6
 - cs, 6
 - Decode, 6
 - Delete, 6
 - version, 6
- codec.h, 41
 - ceCancelled, 46
 - ceFastForward, 46
 - ceFormatNotFound, 46
 - ceFormatNotSupported, 46
 - ceOk, 46
 - ceOtherError, 46
 - ceUnexpectedFileEnd, 46
 - CODEC_CREATE_OFFSET, 43
 - CODEC_CS_OFFSET, 43
 - CODEC_DECODE_OFFSET, 43
 - CODEC_DELETE_OFFSET, 43
 - CODEC_VERSION, 43
 - CODEC_VERSION_OFFSET, 43
 - CodecError, 46
 - CS_AVG_BIT_RATE_OFFSET, 43
 - CS_CANCEL_OFFSET, 43
 - CS_CHANNELS_OFFSET, 43
 - CS_COMMENT_OFFSET, 43
 - CS_CURR_BIT_RATE_OFFSET, 44
 - CS_FAST_FORWARD_OFFSET, 44
 - CS_FILE_LEFT_OFFSET, 44
 - CS_FILE_SIZE_OFFSET, 44
 - CS_GAIN_OFFSET, 44
 - CS_GO_TO_OFFSET, 44
 - CS_MATRIX_OFFSET, 44
 - CS_OUTPUT_OFFSET, 44
 - CS_PEAK_BIT_RATE_OFFSET, 44
 - CS_PLAY_TIME_SAMPLES_OFFSET, 44
 - CS_PLAY_TIME_SECONDS_OFFSET, 44
 - CS_PLAY_TIME_TOTAL_OFFSET, 44
 - CS_READ_OFFSET, 45
 - CS_SAMPLE_RATE_OFFSET, 45
 - CS_SEEK_OFFSET, 45
 - CS_SKIP_OFFSET, 45
 - CS_SPECTRUM_OFFSET, 45
 - CS_TELL_OFFSET, 45
 - CS_VERSION_OFFSET, 45
 - FS_CODEC_SERVICES_VERSION, 45
 - FS_CODSER_COMMENT_END_OF_COMMENTS, 45
 - FS_CODSER_COMMENT_END_OF_LINE, 45
 - MAX_SOURCE_CHANNELS, 46
 - CODEC_CREATE_OFFSET
 - codec.h, 43
 - CODEC_CS_OFFSET
 - codec.h, 43
 - CODEC_DECODE_OFFSET
 - codec.h, 43
 - CODEC_DELETE_OFFSET
 - codec.h, 43
 - CODEC_VERSION
 - codec.h, 43
 - CODEC_VERSION_OFFSET
 - codec.h, 43
 - CodecError
 - codec.h, 46
 - codecmidiwav.h, 46

- CodMiniWavCreate, 47
- CodMiniWavDecode, 47
- CodMiniWavDelete, 47
- CodecServices, 6
- CodecServices
 - avgBitRate, 7
 - cancel, 7
 - channels, 7
 - Comment, 8
 - currBitRate, 8
 - fastForward, 8
 - fileLeft, 8
 - fileSize, 8
 - gain, 8
 - goTo, 8
 - matrix, 9
 - Output, 9
 - peakBitRate, 9
 - playTimeSamples, 9
 - playTimeSeconds, 9
 - playTimeTotal, 9
 - Read, 9
 - sampleRate, 10
 - Seek, 10
 - Skip, 10
 - Spectrum, 10
 - Tell, 10
 - version, 10
- codecvorbis.h, 48
 - CodVBlockSize, 48
 - CodVorbisCreate, 48
 - CodVorbisDecode, 49
 - CodVorbisDelete, 49
 - USE_COMMENTS, 48
- CodMiniWavCreate
 - codecminiwav.h, 47
- CodMiniWavDecode
 - codecminiwav.h, 47
- CodMiniWavDelete
 - codecminiwav.h, 47
- CodVBlockSize
 - codecvorbis.h, 48
- CodVorbisCreate
 - codecvorbis.h, 48
- CodVorbisDecode
 - codecvorbis.h, 49
- CodVorbisDelete
 - codecvorbis.h, 49
- Comment
 - CodecServices, 8
- configuration
 - USBVARS, 32
- configurationDescriptorSize
 - USBVARS, 32
- control__null
 - scsicdb10variant1, 27
 - scsicdb10variant2, 28
 - scsicdb6variant1, 29
 - scsicdb6variant2, 30
 - scsicdb6variant3, 30
- cos
 - math.h, 87
- cosh
 - math.h, 87
- CountBitsLong
 - stdlib.h, 120
- Create
 - Codec, 6
 - FsMapper, 16
 - FsPhysical, 23
- cs
 - Codec, 6
- CS_AVG_BIT_RATE_OFFSET
 - codec.h, 43
- CS_CANCEL_OFFSET
 - codec.h, 43
- CS_CHANNELS_OFFSET
 - codec.h, 43
- CS_COMMENT_OFFSET
 - codec.h, 43
- CS_CURR_BIT_RATE_OFFSET
 - codec.h, 44
- CS_FAST_FORWARD_OFFSET
 - codec.h, 44
- CS_FILE_LEFT_OFFSET
 - codec.h, 44
- CS_FILE_SIZE_OFFSET
 - codec.h, 44
- CS_GAIN_OFFSET
 - codec.h, 44
- CS_GO_TO_OFFSET
 - codec.h, 44
- CS_MATRIX_OFFSET
 - codec.h, 44
- CS_OUTPUT_OFFSET
 - codec.h, 44
- CS_PEAK_BIT_RATE_OFFSET
 - codec.h, 44
- CS_PLAY_TIME_SAMPLES_-
OFFSET

- codec.h, 44
- CS_PLAY_TIME_SECONDS_-
 OFFSET
 codec.h, 44
- CS_PLAY_TIME_TOTAL_-
 OFFSET
 codec.h, 44
- CS_READ_OFFSET
 codec.h, 45
- CS_SAMPLE_RATE_OFFSET
 codec.h, 45
- CS_SEEK_OFFSET
 codec.h, 45
- CS_SKIP_OFFSET
 codec.h, 45
- CS_SPECTRUM_OFFSET
 codec.h, 45
- CS_TELL_OFFSET
 codec.h, 45
- CS_VERSION_OFFSET
 codec.h, 45
- CsOutput
 player.h, 99
- CsRead
 player.h, 99
- CsSeek
 player.h, 99
- ctype.h, 49
 - isalnum, 50
 - isalpha, 50
 - iscntrl, 50
 - isdigit, 50
 - isgraph, 50
 - islower, 50
 - isprint, 50
 - ispunct, 50
 - isspace, 50
 - isupper, 50
 - isxdigit, 50
 - tolower, 51
 - toupper, 51
- curFctl
 audio.h, 39
- currBitRate
 CodecServices, 8
- currentFile
 Player, 25
- currentKeyMap
 player.h, 101
- currentSector
 FATINFO, 12
- D12_FULLEMPY
 usb.h, 128
- D12_INT_BUSRESET
 usb.h, 128
- D12_INT_ENDP0IN
 usb.h, 128
- D12_INT_ENDP0OUT
 usb.h, 128
- D12_INT_ENDP1IN
 usb.h, 128
- D12_INT_ENDP1OUT
 usb.h, 128
- D12_INT_ENDP2IN
 usb.h, 128
- D12_INT_ENDP2OUT
 usb.h, 128
- D12_INT_EOT
 usb.h, 129
- D12_INT_SUSPENDCHANGE
 usb.h, 129
- D12_SETUPPACKET
 usb.h, 129
- D12_STALL
 usb.h, 129
- DAC_DEFAULT_SAMPLERATE
 audio.h, 36
- DAC_DRIVER_ON_DELAY
 audio.h, 37
- DAC_LEFT
 vs1000.h, 156
- DAC_RIGHT
 vs1000.h, 156
- DAC_VOL
 vs1000.h, 156
- dataStart
 FATINFO, 12
- DBL2F16
 vstypes.h, 186
- DBL2F32
 vstypes.h, 186
- DBL_DIG
 float.h, 75
- DBL_EPSILON
 float.h, 75
- DBL_MANT_DIG
 float.h, 75
- DBL_MAX
 float.h, 75

- DBL_MAX_10_EXP
 - float.h, 75
- DBL_MAX_EXP
 - float.h, 75
- DBL_MIN
 - float.h, 75
- DBL_MIN_10_EXP
 - float.h, 75
- DBL_MIN_EXP
 - float.h, 75
- DCT_START
 - vs1000.h, 156
- DEBUG_STACK
 - vs1000.h, 156
- Decode
 - Codec, 6
- DecodeSetupPacket
 - usbblowlib.h, 144
 - vectors.h, 149
- DEFAULT_AUDIO_BUFFER_-
 - SAMPLES
 - audio.h, 37
- defSupportedFiles
 - player.h, 101
- DefUnsupportedFile
 - vs1000.h, 174
- Delete
 - Codec, 6
 - FsMapper, 16
 - FsPhysical, 23
- descriptorTable
 - USBVARS, 32
- dev1000.h, 51
 - Fat12OpenFile, 64
 - Interrupt0, 64
 - Interrupt1, 64
 - Interrupt2, 64
 - Interrupt3, 64
 - InterruptStub0, 64
 - InterruptStub1, 65
 - InterruptStub2, 65
 - InterruptStub3, 65
 - KEY_6, 55
 - KeyScan7, 65
 - MapperlessReadDiskSector, 65
 - MMC_CLK, 55
 - MMC_CLR_WRITE_PROT, 55
 - MMC_CRC_ON_OFF, 55
 - MMC_DE_CARD_LOCKED, 55
 - MMC_DE_CC_ERROR, 55
 - MMC_DE_ECC_FAIL, 55
 - MMC_DE_ERROR, 55
 - MMC_DE_MASK, 55
 - MMC_DE_OUT_OF_-
 - RANGE, 55
 - MMC_DR_ACCEPT, 55
 - MMC_DR_MASK, 55
 - MMC_DR_REJECT_CRC, 55
 - MMC_DR_REJECT_-
 - WRITE_ERROR, 56
 - MMC_ERASE, 56
 - MMC_GO_IDLE_STATE, 56
 - MMC_MISO, 56
 - MMC_MISO_BIT, 56
 - MMC_MOSI, 56
 - MMC_MOSI_BIT, 56
 - MMC_PROGRAM_CSD, 56
 - MMC_R1_ADDRESS, 56
 - MMC_R1_BUSY, 56
 - MMC_R1_COM_CRC, 56
 - MMC_R1_ERASE_RESET, 56
 - MMC_R1_ERASE_SEQ, 57
 - MMC_R1_IDLE_STATE, 57
 - MMC_R1_ILLEGAL_COM, 57
 - MMC_R1_PARAMETER, 57
 - MMC_READ_OCR, 57
 - MMC_READ_SINGLE_-
 - BLOCK, 57
 - MMC_SEND_CID, 57
 - MMC_SEND_CSD, 57
 - MMC_SEND_IF_COND, 57
 - MMC_SEND_OP_COND, 57
 - MMC_SEND_STATUS, 57
 - MMC_SEND_WRITE_PROT, 57
 - MMC_SET_BLOCKLEN, 58
 - MMC_SET_WRITE_PROT, 58
 - MMC_STARTBLOCK_-
 - MWRITE, 58
 - MMC_STARTBLOCK_READ, 58
 - MMC_STARTBLOCK_-
 - WRITE, 58
 - MMC_STOPTRAN_WRITE, 58

MMC_TAG_ERASE_-
GROUP_END, 58
MMC_TAG_ERASE_-
GROUP_START, 58
MMC_TAG_SECTOR_END,
58
MMC_TAG_SECTOR_-
START, 58
MMC_UNTAG_ERASE_-
GROUP, 58
MMC_UNTAG_SECTOR, 58
MMC_WRITE_BLOCK, 59
MMC_XCS, 59
MmcCommand, 65
NandGetOctets, 65
NandPutAddressOctet, 66
NandPutCommand, 66
NandPutOctets, 66
NandSetWaits, 66
OpenFileBaseName, 66
PATCH_TEST_UNIT_-
READY, 59
PatchMSCPacketFromPC, 66
PlayRange, 66
PlayRangeSet, 66
puthex, 67
RC5_CMD_0, 59
RC5_CMD_1, 59
RC5_CMD_2, 59
RC5_CMD_3, 59
RC5_CMD_4, 59
RC5_CMD_5, 59
RC5_CMD_6, 59
RC5_CMD_7, 59
RC5_CMD_8, 59
RC5_CMD_9, 60
RC5_CMD_ALTERNATE, 60
RC5_CMD_BALANCE_LEFT,
60
RC5_CMD_BALANCE_-
RIGHT, 60
RC5_CMD_BASS_DOWN, 60
RC5_CMD_BASS_UP, 60
RC5_CMD_BRIGHT_DOWN,
60
RC5_CMD_BRIGHT_UP, 60
RC5_CMD_CLOCK, 60
RC5_CMD_COLOR_DOWN,
60
RC5_CMD_COLOR_UP, 60
RC5_CMD_CONTRAST_-
DOWN, 60
RC5_CMD_CONTRAST_UP,
61
RC5_CMD_CROSS, 61
RC5_CMD_DISPLAY, 61
RC5_CMD_EXPAND, 61
RC5_CMD_FAST_FORWARD,
61
RC5_CMD_FAST_REVERSE,
61
RC5_CMD_FINETUNE_-
MINUS, 61
RC5_CMD_FINETUNE_-
PLUS, 61
RC5_CMD_LANGUAGE, 61
RC5_CMD_MENU, 61
RC5_CMD_MIX, 61
RC5_CMD_MUTE, 61
RC5_CMD_ONETWODIGITS,
62
RC5_CMD_PAUSE, 62
RC5_CMD_PLAY, 62
RC5_CMD_PRESET, 62
RC5_CMD_PROGRAM_-
DOWN, 62
RC5_CMD_PROGRAM_UP,
62
RC5_CMD_RECORD, 62
RC5_CMD_STANDBY, 62
RC5_CMD_STOP, 62
RC5_CMD_STORE, 62
RC5_CMD_SWITCH, 62
RC5_CMD_SYSTEM_-
SELECT, 62
RC5_CMD_TEXT, 63
RC5_CMD_TIMER, 63
RC5_CMD_TREBLE_DOWN,
63
RC5_CMD_TREBLE_UP, 63
RC5_CMD_VOLUME_DOWN,
63
RC5_CMD_VOLUME_UP, 63
RC5_SYS_CDPLAYER, 63
RC5_SYS_-
EXPERIMENTAL19, 63
RC5_SYS_EXPERIMENTAL7,
63
RC5_SYS_PREAMP, 63

- RC5_SYS_RECEIVERTUNER, 63
- RC5_SYS_TAPERECORDER, 63
- RC5_SYS_TELETEXT, 64
- RC5_SYS_TVSET, 64
- RC5_SYS_VCR, 64
- Rc5GetFIFO, 67
- Rc5Init, 67
- ReadIRam, 67
- ScsiTestUnitReady, 67
- SpiSendClocks, 67
- SpiSendReceiveMmc, 67
- Suspend7, 68
- WriteIRam, 68
- DIRECT_VORBIS_BLOCKSIZE
 - audio.h, 37
- Disable
 - EARSPEAKER, 11
 - vs1000.h, 175
- DiskDataReceived
 - scsi.h, 110
- DiskProtocolCommand
 - scsi.h, 110
- DiskProtocolError
 - usbblowlib.h, 144
- DT_CONFIGURATION
 - usbblowlib.h, 138
- DT_DEVICE
 - usbblowlib.h, 138
- DT_LANGUAGES
 - usbblowlib.h, 138
- DT_MODEL
 - usbblowlib.h, 138
- DT_SERIAL
 - usbblowlib.h, 138
- DT_VENDOR
 - usbblowlib.h, 139
- E2BIG
 - errno.h, 69
- EACCES
 - errno.h, 69
- EAGAIN
 - errno.h, 69
- EALREADY
 - errno.h, 69
- EARSPEAKER, 10
 - Disable, 11
 - Freq, 11
 - longFrames, 11
 - Old, 11
 - Setting, 11
- earSpeaker
 - audio.h, 40
- earSpeakerDisable
 - audio.h, 40
- earSpeakerReg
 - audio.h, 40
- EBADF
 - errno.h, 70
- EBUSY
 - errno.h, 70
- ecc01
 - FmfMeta, 14
- ecc2AndType
 - FmfMeta, 14
- ECHILD
 - errno.h, 70
- EDEADLK
 - errno.h, 70
- EDOM
 - errno.h, 70
 - math.h, 86
- EEXIST
 - errno.h, 70
- EFAULT
 - errno.h, 70
- EFBIG
 - errno.h, 70
- EINPROGRESS
 - errno.h, 70
- EINTR
 - errno.h, 70
- EINVAL
 - errno.h, 70
- EIO
 - errno.h, 70
- EISDIR
 - errno.h, 71
- EMFILE
 - errno.h, 71
- EMLINK
 - errno.h, 71
- emptyBlock
 - FsMapperFlash, 18
- Enable
 - vs1000.h, 175
- ENDPOINT_SIZE_0
 - usbblowlib.h, 139

- ENDPOINT_SIZE_1
 - usbblowlib.h, 139
- ENFILE
 - errno.h, 71
- ENODEV
 - errno.h, 71
- ENOENT
 - errno.h, 71
- ENOEXEC
 - errno.h, 71
- ENOMEM
 - errno.h, 71
- ENOSPC
 - errno.h, 71
- ENOTBLK
 - errno.h, 71
- ENOTDIR
 - errno.h, 71
- ENOTTY
 - errno.h, 71
- ENXIO
 - errno.h, 72
- EOF
 - stdio.h, 115
- EP0_PACKET_SIZE
 - usb.h, 129
- EP0_RX_FIFO_SIZE
 - usb.h, 129
- EP0_TX_FIFO_SIZE
 - usb.h, 129
- EP1_PACKET_SIZE
 - usb.h, 129
- EP1_RX_FIFO_SIZE
 - usb.h, 129
- EP1_TX_FIFO_SIZE
 - usb.h, 129
- EP2_PACKET_SIZE
 - usb.h, 129
- EP2_RX_FIFO_SIZE
 - usb.h, 129
- EP2_TX_FIFO_SIZE
 - usb.h, 130
- EPERM
 - errno.h, 72
- EPIPE
 - errno.h, 72
- EPReady
 - USBVARS, 33
- ERANGE
 - errno.h, 72
- math.h, 86
- Erase
 - FsPhysical, 23
- eraseBlocks
 - FsPhysical, 23
- eraseBlockSize
 - FsPhysical, 23
- EROFS
 - errno.h, 72
- errno
 - errno.h, 72
- errno.h, 68
 - E2BIG, 69
 - EACCES, 69
 - EAGAIN, 69
 - EALREADY, 69
 - EBADF, 70
 - EBUSY, 70
 - ECHILD, 70
 - EDEADLK, 70
 - EDOM, 70
 - EEXIST, 70
 - EFAULT, 70
 - EFBIG, 70
 - EINPROGRESS, 70
 - EINTR, 70
 - EINVAL, 70
 - EIO, 70
 - EISDIR, 71
 - EMFILE, 71
 - EMLINK, 71
 - ENFILE, 71
 - ENODEV, 71
 - ENOENT, 71
 - ENOEXEC, 71
 - ENOMEM, 71
 - ENOSPC, 71
 - ENOTBLK, 71
 - ENOTDIR, 71
 - ENOTTY, 71
 - ENXIO, 72
 - EPERM, 72
 - EPIPE, 72
 - ERANGE, 72
 - EROFS, 72
 - errno, 72
 - ESPIPE, 72
 - ESRCH, 72
 - ETXTBSY, 72
 - EWouldBlock, 72

- EXDEV, 72
- ESPIPE
 - errno.h, 72
- ESRCH
 - errno.h, 72
- ETXTBSY
 - errno.h, 72
- event
 - KeyMapping, 24
- EWOULDBLOCK
 - errno.h, 72
- EXDEV
 - errno.h, 72
- exit
 - stdlib.h, 120
- EXIT_FAILURE
 - stdlib.h, 119
- EXIT_SUCCESS
 - stdlib.h, 120
- exp
 - math.h, 87
- extClock4KHz
 - audio.h, 40
- ExtraZeroLengthPacketNeeded
 - USBVARS, 33
- f_int16
 - vstypes.h, 188
- f_int32
 - vstypes.h, 189
- fabs
 - math.h, 86, 87
- fastForward
 - CodecServices, 8
- fat.h, 73
 - FAT_LFN_SIZE, 73
 - FAT_MKID, 73
 - FATINFO_IN_Y, 73
 - LAST_FRAGMENT, 73
 - MAX_FRAGMENTS, 74
 - minifatBuffer, 74
 - minifatFragments, 74
 - minifatInfo, 74
- Fat12OpenFile
 - dev1000.h, 64
- FAT_LFN_SIZE
 - fat.h, 73
- FAT_MKID
 - fat.h, 73
- FatCheckFileType
 - minifat.h, 90
- FatFindSector
 - minifat.h, 90
- FatFragmentList
 - minifat.h, 90
- FatGetByte
 - minifat.h, 91
- FatGetLong
 - minifat.h, 91
- FatGetWord
 - minifat.h, 91
- FatHandleDir
 - minifat.h, 91
- FATINFO, 11
 - BPB_RootEntCnt, 12
 - currentSector, 12
 - dataStart, 12
 - fatSectorsPerCluster, 12
 - fatStart, 12
 - fileName, 12
 - filePos, 12
 - fileSize, 13
 - FilSysType, 13
 - gFileNum, 13
 - IS_FAT_32, 13
 - longFileName, 13
 - parentDir, 13
 - rootStart, 13
 - supportedSuffixes, 13
 - totSize, 13
- FATINFO_IN_Y
 - fat.h, 73
- FatInitFileSystem
 - minifat.h, 92
- FatIterateOverFreeSectors
 - minifat.h, 92
- FatOpenFile
 - minifat.h, 92
- FatReadFile
 - minifat.h, 92
- fatSectorsPerCluster
 - FATINFO, 12
- FatSeek
 - minifat.h, 92
- fatStart
 - FATINFO, 12
- FatTell
 - minifat.h, 93
- FCH_DIV_INCLK
 - vs1000.h, 156

- FCH_DIV_INCLK_B
 - vs1000.h, 157
- FCH_FORCE_PLL
 - vs1000.h, 157
- FCH_FORCE_PLL_B
 - vs1000.h, 157
- FCH_MUL0
 - vs1000.h, 157
- FCH_MUL0_B
 - vs1000.h, 157
- FCH_MUL1
 - vs1000.h, 157
- FCH_MUL1_B
 - vs1000.h, 157
- FCH_MUL2
 - vs1000.h, 157
- FCH_MUL2_B
 - vs1000.h, 157
- FCH_MUL3
 - vs1000.h, 157
- FCH_MUL3_B
 - vs1000.h, 157
- FCH_PLL_LOCKED
 - vs1000.h, 157
- FCH_PLL_LOCKED_B
 - vs1000.h, 158
- FCH_PLL_SET_LOCK
 - vs1000.h, 158
- FCH_PLL_SET_LOCK_B
 - vs1000.h, 158
- FCH_VCO_OUT_ENA
 - vs1000.h, 158
- FCH_VCO_OUT_ENA_B
 - vs1000.h, 158
- fclose
 - stdio.h, 116
- FDIV16
 - vstypes.h, 186
- FDIV32
 - vstypes.h, 186
- feof
 - stdio.h, 116
- ferror
 - stdio.h, 116
- ffCount
 - Player, 25
- fflush
 - stdio.h, 116
- fgetc
 - stdio.h, 116
- fgetpos
 - stdio.h, 116
- fgets
 - stdio.h, 116
- FILE
 - stdio.h, 115
- fileLeft
 - CodecServices, 8
- fileName
 - FATINFO, 12
- FILENAME_MAX
 - stdio.h, 115
- filePos
 - FATINFO, 12
- fileSize
 - CodecServices, 8
 - FATINFO, 13
- FilSysType
 - FATINFO, 13
- firstBlock
 - FsMapperTiny, 20
- fiveKeyMap
 - player.h, 101
- flags__lbab3
 - scsicdb10variant2, 28
- flags__pageCode
 - scsicdb6variant1, 29
 - scsicdb6variant2, 30
- flags__res
 - scsicdb6variant3, 30
- float.h, 74
 - DBL_DIG, 75
 - DBL_EPSILON, 75
 - DBL_MANT_DIG, 75
 - DBL_MAX, 75
 - DBL_MAX_10_EXP, 75
 - DBL_MAX_EXP, 75
 - DBL_MIN, 75
 - DBL_MIN_10_EXP, 75
 - DBL_MIN_EXP, 75
 - FLT_DIG, 76
 - FLT_EPSILON, 76
 - FLT_MANT_DIG, 76
 - FLT_MAX, 76
 - FLT_MAX_10_EXP, 76
 - FLT_MAX_EXP, 76
 - FLT_MIN, 76
 - FLT_MIN_10_EXP, 76
 - FLT_MIN_EXP, 76
 - FLT_RADIX, 76

- FLT_ROUNDS, 76
- LDBL_DIG, 76
- LDBL_EPSILON, 77
- LDBL_MANT_DIG, 77
- LDBL_MAX, 77
- LDBL_MAX_10_EXP, 77
- LDBL_MAX_EXP, 77
- LDBL_MIN, 77
- LDBL_MIN_10_EXP, 77
- LDBL_MIN_EXP, 77
- floor
 - math.h, 87
- FLT_DIG
 - float.h, 76
- FLT_EPSILON
 - float.h, 76
- FLT_MANT_DIG
 - float.h, 76
- FLT_MAX
 - float.h, 76
- FLT_MAX_10_EXP
 - float.h, 76
- FLT_MAX_EXP
 - float.h, 76
- FLT_MIN
 - float.h, 76
- FLT_MIN_10_EXP
 - float.h, 76
- FLT_MIN_EXP
 - float.h, 76
- FLT_RADIX
 - float.h, 76
- FLT_ROUNDS
 - float.h, 76
- Flush
 - FsMapper, 17
- FmfMeta, 14
- FmfMeta
 - ecc01, 14
 - ecc2AndType, 14
 - logicalPageNo, 14
 - newBranch, 14
 - reservedAndBadBlock, 14
 - unused, 14
- fmod
 - math.h, 87
- FMT_H16
 - vstypes.h, 187
- FMT_H32
 - vstypes.h, 187
- FMT_S16
 - vstypes.h, 187
- FMT_S32
 - vstypes.h, 187
- FMT_U16
 - vstypes.h, 187
- FMT_U32
 - vstypes.h, 187
- FMUL16
 - vstypes.h, 187
- FMUL32
 - vstypes.h, 187
- fontData
 - romfont.h, 104
- fontPtrs
 - romfont.h, 104
- fopen
 - stdio.h, 116
- FOPEN_MAX
 - stdio.h, 115
- forwardModulo
 - AUDIOPTR, 5
- fpos_t
 - stdio.h, 115
- fprintf
 - stdio.h, 116
- fputc
 - stdio.h, 116
- fputs
 - stdio.h, 116
- FRAGMENT, 15
 - size, 15
 - start, 15
- fread
 - stdio.h, 117
- Free
 - FsMapper, 17
- FreeBus
 - FsPhysical, 23
- freed
 - FsMapperFlash, 18
- freeSectorCallback
 - minifat.h, 90
- freopen
 - stdio.h, 117
- Freq
 - EARSPEAKER, 11
- FREQCTLH
 - vs1000.h, 158
- FREQCTL

- vs1000.h, 158
- frexp
 - math.h, 86, 87
- FS_CODECSERVICES_VERSION
 - codec.h, 45
- FS_CODSER_COMMENT_END_OF_COMMENTS
 - codec.h, 45
- FS_CODSER_COMMENT_END_OF_LINE
 - codec.h, 45
- FS_MAP_FLASH_MAX_ERASE_PAGES
 - mapperflash.h, 81
- FS_MAP_FLASH_PAGE_SIZE
 - mapperflash.h, 81
- FS_MAP_NON_FULL
 - mapperflash.h, 81
- FS_MAPPER_VERSION
 - mapper.h, 78
- FS_PHYSICAL_VERSION
 - physical.h, 95
- fseek
 - stdio.h, 117
- fsetpos
 - stdio.h, 117
- FsMapFICacheDump
 - mapperflash.h, 82
- FsMapFICreate
 - mapperflash.h, 82
- FsMapFIDelete
 - mapperflash.h, 82
- FsMapFIDump
 - mapperflash.h, 82
- FsMapFIFlush
 - mapperflash.h, 82
- FsMapFIFree
 - mapperflash.h, 82
- FsMapFINullFail
 - mappertiny.h, 84
- FsMapFINullOk
 - mappertiny.h, 84
- FsMapFIPrint
 - mapperflash.h, 82
- FsMapFIRead
 - mapperflash.h, 82
- FsMapFIWrite
 - mapperflash.h, 82
- FsMapper, 15
- FsMapper
 - blocks, 16
 - blockSize, 16
 - cacheBlocks, 16
 - Create, 16
 - Delete, 16
 - Flush, 17
 - Free, 17
 - physical, 17
 - Read, 17
 - version, 17
 - Write, 17
- FsMapperFlash, 17
- FsMapperFlash
 - blocksPerErase, 18
 - cache, 18
 - emptyBlock, 18
 - freed, 18
 - lastUsed, 19
 - m, 19
 - nonFullLimit, 19
 - physPages, 19
 - root, 19
 - skipped, 19
- FsMapperTiny, 19
- FsMapperTiny
 - blocksPerErase, 20
 - firstBlock, 20
 - lastBlock, 20
 - logToPhys, 20
 - m, 21
 - meta, 21
 - root, 21
- FsMapRamCreate
 - vs1000.h, 175
- FsMapTnCreate
 - mappertiny.h, 84
- FsMapTnDelete
 - mappertiny.h, 84
- FsMapTnFlush
 - mappertiny.h, 84
- FsMapTnFree
 - mappertiny.h, 84
- FsMapTnRead
 - mappertiny.h, 84
- FsMapTnWrite
 - mappertiny.h, 84
- FsNandPhys, 21
- FsNandPhys
 - nandType, 22
 - p, 22

- waitns, 22
- FsPhNandCreate
 - vsNand.h, 181
- FsPhNandDelete
 - vsNand.h, 181
- FsPhNandErase
 - vsNand.h, 182
- FsPhNandFreeBus
 - vsNand.h, 182
- FsPhNandRead
 - vsNand.h, 182
- FsPhNandReinitialize
 - vsNand.h, 182
- FsPhNandWrite
 - vsNand.h, 182
- FsPhysical, 22
- FsPhysical
 - Create, 23
 - Delete, 23
 - Erase, 23
 - eraseBlocks, 23
 - eraseBlockSize, 23
 - FreeBus, 23
 - pageSize, 23
 - Read, 23
 - Reinitialize, 23
 - version, 24
 - Write, 24
- ftell
 - stdio.h, 117
- fwrite
 - stdio.h, 117
- g_dcthi
 - vs1000.h, 178
- g_dctlo
 - vs1000.h, 178
- g_others
 - vs1000.h, 178
- g_yprev0
 - vs1000.h, 178
- g_yprev1
 - vs1000.h, 178
- gain
 - CodecServices, 8
- getc
 - stdio.h, 117
- getch
 - vs1000.h, 175
- getchar
 - stdio.h, 117
- gets
 - stdio.h, 117
- gFileNum
 - FATINFO, 13
- goTo
 - CodecServices, 8
- GPIO0_ALE
 - vs1000.h, 158
- GPIO0_BIT_CONF
 - vs1000.h, 158
- GPIO0_BIT_ENG0
 - vs1000.h, 158
- GPIO0_BIT_ENG1
 - vs1000.h, 158
- GPIO0_CLE
 - vs1000.h, 158
- GPIO0_CLEAR_MASK
 - vs1000.h, 159
- GPIO0_CS1
 - vs1000.h, 159
- GPIO0_DDR
 - vs1000.h, 159
- GPIO0_IDATA
 - vs1000.h, 159
- GPIO0_INT_FALL
 - vs1000.h, 159
- GPIO0_INT_PEND
 - vs1000.h, 159
- GPIO0_INT_RISE
 - vs1000.h, 159
- GPIO0_MODE
 - vs1000.h, 159
- GPIO0_ODATA
 - vs1000.h, 159
- GPIO0_RD
 - vs1000.h, 159
- GPIO0_READY
 - vs1000.h, 159
- GPIO0_SET_MASK
 - vs1000.h, 159
- GPIO0_WR
 - vs1000.h, 160
- GPIO1_BIT_CONF
 - vs1000.h, 160
- GPIO1_BIT_ENG0
 - vs1000.h, 160
- GPIO1_BIT_ENG1
 - vs1000.h, 160
- GPIO1_CLEAR_MASK

vs1000.h, 160
 GPIO1_DDR
 vs1000.h, 160
 GPIO1_IDATA
 vs1000.h, 160
 GPIO1_INT_FALL
 vs1000.h, 160
 GPIO1_INT_PEND
 vs1000.h, 160
 GPIO1_INT_RISE
 vs1000.h, 160
 GPIO1_MODE
 vs1000.h, 160
 GPIO1_ODATA
 vs1000.h, 160
 GPIO1_SET_MASK
 vs1000.h, 161

 haltTime
 audio.h, 40
 HUGE_VAL
 math.h, 86
 hwSampleRate
 audio.h, 40

 IdleHook
 vs1000.h, 175
 InitAudio
 audio.h, 38
 InitFileSystem
 vs1000.h, 175
 InitUSB
 usbblowlib.h, 144
 InitUSBDescriptors
 usbblowlib.h, 145
 INT_EN_DAC
 vs1000.h, 161
 INT_EN_GPIO0
 vs1000.h, 161
 INT_EN_GPIO1
 vs1000.h, 161
 INT_EN_NFLSH
 vs1000.h, 161
 INT_EN_NONE
 vs1000.h, 161
 INT_EN_REGU
 vs1000.h, 161
 INT_EN_RX
 vs1000.h, 161
 INT_EN_SPI
 vs1000.h, 161
 INT_EN_TIM0
 vs1000.h, 161
 INT_EN_TIM1
 vs1000.h, 161
 INT_EN_TX
 vs1000.h, 161
 INT_EN_USB
 vs1000.h, 162
 INT_ENABLE
 vs1000.h, 162
 INT_ENABLEH
 vs1000.h, 162
 INT_ENABLEL
 vs1000.h, 162
 INT_ENCOUNT
 vs1000.h, 162
 INT_GLOB_DIS
 vs1000.h, 162
 INT_GLOB_EN
 vs1000.h, 162
 INT_ORIGIN
 vs1000.h, 162
 INT_VECTOR
 vs1000.h, 162
 interfaces
 USBVARS, 33
 Interrupt0
 dev1000.h, 64
 Interrupt1
 dev1000.h, 64
 Interrupt2
 dev1000.h, 64
 Interrupt3
 dev1000.h, 64
 InterruptStub0
 dev1000.h, 64
 InterruptStub1
 dev1000.h, 65
 InterruptStub2
 dev1000.h, 65
 InterruptStub3
 dev1000.h, 65
 INTF_DAC
 vs1000.h, 162
 INTF_GPIO0
 vs1000.h, 162
 INTF_GPIO1
 vs1000.h, 162
 INTF_NFLSH

- vs1000.h, 163
- INTF_REGU
 - vs1000.h, 163
- INTF_RX
 - vs1000.h, 163
- INTF_SPI
 - vs1000.h, 163
- INTF_TIM0
 - vs1000.h, 163
- INTF_TIM1
 - vs1000.h, 163
- INTF_TX
 - vs1000.h, 163
- INTF_USB
 - vs1000.h, 163
- INTV_DAC
 - vs1000.h, 163
- INTV_GPIO0
 - vs1000.h, 163
- INTV_GPIO1
 - vs1000.h, 163
- INTV_NFLSH
 - vs1000.h, 163
- INTV_REGU
 - vs1000.h, 164
- INTV_RX
 - vs1000.h, 164
- INTV_SPI
 - vs1000.h, 164
- INTV_TIM0
 - vs1000.h, 164
- INTV_TIM1
 - vs1000.h, 164
- INTV_TX
 - vs1000.h, 164
- INTV_USB
 - vs1000.h, 164
- IRAM_SIZE
 - vs1000.h, 164
- IRAM_START
 - vs1000.h, 164
- IROM_SIZE
 - vs1000.h, 164
- IROM_START
 - vs1000.h, 164
- IS_FAT_32
 - FATINFO, 13
- isalnum
 - ctype.h, 50
- isalpha
 - ctype.h, 50
- iscntrl
 - ctype.h, 50
- isdigit
 - ctype.h, 50
- isgraph
 - ctype.h, 50
- islower
 - ctype.h, 50
- isprint
 - ctype.h, 50
- ispunct
 - ctype.h, 50
- ISqrt
 - math.h, 87
- isspace
 - ctype.h, 50
- isupper
 - ctype.h, 50
- isxdigit
 - ctype.h, 50
- ke_earSpeaker
 - player.h, 98
- ke_earSpeakerToggle
 - player.h, 98
- ke_ff_faster
 - player.h, 98
- ke_ff_off
 - player.h, 98
- ke_ff_slower
 - player.h, 98
- ke_forward
 - player.h, 98
- ke_next
 - player.h, 98
- ke_null
 - player.h, 98
- ke_pauseToggle
 - player.h, 98
- ke_powerOff
 - player.h, 98
- ke_previous
 - player.h, 98
- ke_randomToggle
 - player.h, 98
- ke_randomToggleNewSong
 - player.h, 98
- ke_rewind
 - player.h, 98

- ke_volumeDown
 - player.h, 98
- ke_volumeDown2
 - player.h, 98
- ke_volumeUp
 - player.h, 98
- ke_volumeUp2
 - player.h, 98
- key
 - KeyMapping, 24
- KEY_1
 - player.h, 97
- KEY_2
 - player.h, 97
- KEY_3
 - player.h, 97
- KEY_4
 - player.h, 97
- KEY_5
 - player.h, 97
- KEY_6
 - dev1000.h, 55
- KEY_LONG_ONESHOT
 - player.h, 97
- KEY_LONG_PRESS
 - player.h, 97
- KEY_POWER
 - player.h, 97
- KEY_RELEASED
 - player.h, 97
- keyCheck
 - player.h, 101
- keyEvent
 - player.h, 98
- KeyEventHandler
 - player.h, 99
- KeyMapping, 24
- KeyMapping
 - event, 24
 - key, 24
- keyOld
 - player.h, 101
- keyOldTime
 - player.h, 101
- KeyScan
 - player.h, 99
- KeyScan7
 - dev1000.h, 65
- labs
 - stdlib.h, 120
- LAST_FRAGMENT
 - fat.h, 73
- lastBlock
 - FsMapperTiny, 20
- lastSofFill
 - USBVARS, 33
- lastSofTime
 - USBVARS, 33
- lastSofTimeout
 - USBVARS, 33
- lastUsed
 - FsMapperFlash, 19
- lbab0__res
 - scsicdb10variant1, 27
 - scsicdb10variant2, 28
- lbab2__lbab1
 - scsicdb10variant1, 27
 - scsicdb10variant2, 28
- LDBL_DIG
 - float.h, 76
- LDBL_EPSILON
 - float.h, 77
- LDBL_MANT_DIG
 - float.h, 77
- LDBL_MAX
 - float.h, 77
- LDBL_MAX_10_EXP
 - float.h, 77
- LDBL_MAX_EXP
 - float.h, 77
- LDBL_MIN
 - float.h, 77
- LDBL_MIN_10_EXP
 - float.h, 77
- LDBL_MIN_EXP
 - float.h, 77
- ldexp
 - math.h, 86, 88
- LED1
 - player.h, 97
- LED2
 - player.h, 98
- leftVol
 - AUDIOPTR, 5
- length
 - usbpkt, 31
- length__opcode
 - scsicdb10variant1, 27
 - scsicdb10variant2, 28

- scsicdb6variant1, 29
- scsicdb6variant2, 30
- scsicdb6variant3, 31
- LoadCheck
 - vs1000.h, 175
- log
 - math.h, 88
- log10
 - math.h, 88
- log2
 - math.h, 86
- logdb
 - math.h, 86
- logicalPageNo
 - FmfMeta, 14
- logToPhys
 - FsMapperTiny, 20
- longFileName
 - FATINFO, 13
- longFrames
 - EARSPERAKER, 11
- LongLog2
 - math.h, 88
- m
 - FsMapperFlash, 19
 - FsMapperTiny, 21
- MAKEMOD
 - vsasm.h, 179
- MAKEMOD64
 - vsasm.h, 179
- MAKEMODB
 - vsasm.h, 179
- MAKEMODF
 - vsasm.h, 179
- mallocAreaX
 - player.h, 101
- mallocAreaY
 - player.h, 101
- MAP_BLOCK_SIZE_OFFSET
 - mapper.h, 78
- MAP_BLOCKS_OFFSET
 - mapper.h, 79
- MAP_CACHE_BLOCKS_OFFSET
 - mapper.h, 79
- MAP_CREATE_OFFSET
 - mapper.h, 79
- MAP_DELETE_OFFSET
 - mapper.h, 79
- MAP_FLUSH_OFFSET
 - mapper.h, 79
- MAP_FREE_OFFSET
 - mapper.h, 79
- MAP_PHYSICAL_OFFSET
 - mapper.h, 79
- MAP_READ_OFFSET
 - mapper.h, 79
- MAP_VERSION_OFFSET
 - mapper.h, 79
- MAP_WRITE_OFFSET
 - mapper.h, 79
- mapper.h, 79
- MAP_FREE_OFFSET
 - mapper.h, 79
- MAP_PHYSICAL_OFFSET
 - mapper.h, 79
- MAP_READ_OFFSET
 - mapper.h, 79
- MAP_VERSION_OFFSET
 - mapper.h, 79
- MAP_WRITE_OFFSET
 - mapper.h, 79
- mapper.h, 77
 - FS_MAPPER_VERSION, 78
 - MAP_BLOCK_SIZE_-
 - OFFSET, 78
 - MAP_BLOCKS_OFFSET, 79
 - MAP_CACHE_BLOCKS_-
 - OFFSET, 79
 - MAP_CREATE_OFFSET, 79
 - MAP_DELETE_OFFSET, 79
 - MAP_FLUSH_OFFSET, 79
 - MAP_FREE_OFFSET, 79
 - MAP_PHYSICAL_OFFSET, 79
 - MAP_READ_OFFSET, 79
 - MAP_VERSION_OFFSET, 79
 - MAP_WRITE_OFFSET, 79
- mapperflash.h, 79
 - FS_MAP_FLASH_MAX_-
 - ERASE_PAGES, 81
 - FS_MAP_FLASH_PAGE_-
 - SIZE, 81
 - FS_MAP_NON_FULL, 81
 - FsMapFlCacheDump, 82
 - FsMapFlCreate, 82
 - FsMapFlDelete, 82
 - FsMapFlDump, 82
 - FsMapFlFlush, 82
 - FsMapFlFree, 82
 - FsMapFlPrint, 82
 - FsMapFlRead, 82
 - FsMapFlWrite, 82
 - memcpyXY, 81
 - memcpyYX, 81
 - memcpyYY, 81
 - memsetY, 81
 - qsorty, 81
- MapperlessReadDiskSector
 - dev1000.h, 65
- MapperReadDiskSector
 - vs1000.h, 175

- mappertiny.h, 82
 - FsMapFlNullFail, 84
 - FsMapFlNullOk, 84
 - FsMapTnCreate, 84
 - FsMapTnDelete, 84
 - FsMapTnFlush, 84
 - FsMapTnFree, 84
 - FsMapTnRead, 84
 - FsMapTnWrite, 84
- MassStorage
 - player.h, 100
- math.h, 84
 - _divide16signed, 87
 - _divide16unsigned, 87
 - _modf, 87
 - _oldmodf, 86
 - acos, 87
 - asin, 87
 - atan, 87
 - atan2, 87
 - ceil, 87
 - cos, 87
 - cosh, 87
 - EDOM, 86
 - ERANGE, 86
 - exp, 87
 - fabs, 86, 87
 - floor, 87
 - fmod, 87
 - frexp, 86, 87
 - HUGE_VAL, 86
 - ISqrt, 87
 - ldexp, 86, 88
 - log, 88
 - log10, 88
 - log2, 86
 - logdb, 86
 - LongLog2, 88
 - modf, 86
 - MulSh20, 86
 - MulSh24, 86
 - pow, 88
 - sin, 88
 - sinh, 88
 - sqrt, 88
 - SqrtF, 88
 - SqrtF32, 88
 - SqrtI, 88
 - SqrtI32, 88
 - tan, 88
 - tanh, 88
- matrix
 - CodecServices, 9
- MAX_FRAGMENTS
 - fat.h, 74
- MAX_SOURCE_CHANNELS
 - codec.h, 46
- maxClock
 - Player, 25
- memchr
 - string.h, 123
- memclearXY
 - string.h, 123
- memcmp
 - string.h, 123
- memcmpY
 - string.h, 123
- MemCopyPackedBigEndian
 - minifat.h, 93
 - string.h, 123
- MemCopyPackedLittleEndian
 - minifat.h, 93
- memcpy
 - string.h, 123
- memcpyXY
 - mapperflash.h, 81
 - string.h, 124
- memcpyYX
 - mapperflash.h, 81
 - string.h, 124
- memcpyYY
 - mapperflash.h, 81
 - string.h, 124
- memmove
 - string.h, 124
- MemReadPacked
 - minifat.h, 93
- MemReadPackedY
 - minifat.h, 93
- memset
 - string.h, 124
- memsetY
 - mapperflash.h, 81
 - string.h, 124
- memswap
 - string.h, 124
- memswapxy
 - string.h, 124
- memswapy
 - string.h, 124

- MemTests
 - vs1000.h, 176
- MemWritePacked
 - minifat.h, 93
- MemWritePackedY
 - minifat.h, 94
- meta
 - FsMapperTiny, 21
- minifat.h, 88
 - FatCheckFileType, 90
 - FatFindSector, 90
 - FatFragmentList, 90
 - FatGetByte, 91
 - FatGetLong, 91
 - FatGetWord, 91
 - FatHandleDir, 91
 - FatInitFileSystem, 92
 - FatIterateOverFreeSectors, 92
 - FatOpenFile, 92
 - FatReadFile, 92
 - FatSeek, 92
 - FatTell, 93
 - freeSectorCallback, 90
 - MemCopyPackedBigEndian, 93
 - MemCopyPackedLittleEndian, 93
 - MemReadPacked, 93
 - MemReadPackedY, 93
 - MemWritePacked, 93
 - MemWritePackedY, 94
 - MINIFAT_H, 90
 - ReadDiskSector, 94
- MINIFAT_H
 - minifat.h, 90
- minifatBuffer
 - fat.h, 74
- minifatFragments
 - fat.h, 74
- minifatInfo
 - fat.h, 74
- MMC_CLK
 - dev1000.h, 55
- MMC_CLR_WRITE_PROT
 - dev1000.h, 55
- MMC_CRC_ON_OFF
 - dev1000.h, 55
- MMC_DE_CARD_LOCKED
 - dev1000.h, 55
- MMC_DE_CC_ERROR
 - dev1000.h, 55
- MMC_DE_ECC_FAIL
 - dev1000.h, 55
- MMC_DE_ERROR
 - dev1000.h, 55
- MMC_DE_MASK
 - dev1000.h, 55
- MMC_DE_OUT_OF_RANGE
 - dev1000.h, 55
- MMC_DR_ACCEPT
 - dev1000.h, 55
- MMC_DR_MASK
 - dev1000.h, 55
- MMC_DR_REJECT_CRC
 - dev1000.h, 55
- MMC_DR_REJECT_WRITE_ERROR
 - dev1000.h, 56
- MMC_ERASE
 - dev1000.h, 56
- MMC_GO_IDLE_STATE
 - dev1000.h, 56
- MMC_MISO
 - dev1000.h, 56
- MMC_MISO_BIT
 - dev1000.h, 56
- MMC_MOSI
 - dev1000.h, 56
- MMC_MOSI_BIT
 - dev1000.h, 56
- MMC_PROGRAM_CSD
 - dev1000.h, 56
- MMC_R1_ADDRESS
 - dev1000.h, 56
- MMC_R1_BUSY
 - dev1000.h, 56
- MMC_R1_COM_CRC
 - dev1000.h, 56
- MMC_R1_ERASE_RESET
 - dev1000.h, 56
- MMC_R1_ERASE_SEQ
 - dev1000.h, 57
- MMC_R1_IDLE_STATE
 - dev1000.h, 57
- MMC_R1_ILLEGAL_COM
 - dev1000.h, 57
- MMC_R1_PARAMETER
 - dev1000.h, 57
- MMC_READ_OCR
 - dev1000.h, 57
- MMC_READ_SINGLE_BLOCK
 - dev1000.h, 57

- MMC_SEND_CID
 - dev1000.h, 57
- MMC_SEND_CSD
 - dev1000.h, 57
- MMC_SEND_IF_COND
 - dev1000.h, 57
- MMC_SEND_OP_COND
 - dev1000.h, 57
- MMC_SEND_STATUS
 - dev1000.h, 57
- MMC_SEND_WRITE_PROT
 - dev1000.h, 57
- MMC_SET_BLOCKLEN
 - dev1000.h, 58
- MMC_SET_WRITE_PROT
 - dev1000.h, 58
- MMC_STARTBLOCK_MWRITE
 - dev1000.h, 58
- MMC_STARTBLOCK_READ
 - dev1000.h, 58
- MMC_STARTBLOCK_WRITE
 - dev1000.h, 58
- MMC_STOPTRAN_WRITE
 - dev1000.h, 58
- MMC_TAG_ERASE_GROUP_-
 - END
 - dev1000.h, 58
- MMC_TAG_ERASE_GROUP_-
 - START
 - dev1000.h, 58
- MMC_TAG_SECTOR_END
 - dev1000.h, 58
- MMC_TAG_SECTOR_START
 - dev1000.h, 58
- MMC_UNTAG_ERASE_GROUP
 - dev1000.h, 58
- MMC_UNTAG_SECTOR
 - dev1000.h, 58
- MMC_WRITE_BLOCK
 - dev1000.h, 59
- MMC_XCS
 - dev1000.h, 59
- MmcCommand
 - dev1000.h, 65
- modf
 - math.h, 86
- MR_INT
 - vstypes.h, 188
- MR_NONE
 - vstypes.h, 188
- MR_SAT
 - vstypes.h, 188
- MSC_BULK_IN_ENDPOINT
 - usbblowlib.h, 139
- MSC_BULK_OUT_ENDPOINT
 - usbblowlib.h, 139
- MSCPacketFromPC
 - usbblowlib.h, 145
- MscSendCsw
 - usbblowlib.h, 145
- MulSh20
 - math.h, 86
- MulSh24
 - math.h, 86
- NAK_SENT_INT
 - usbblowlib.h, 139
- NAND_OP_COMMIT_DATA_-
 - ADDRESS
 - vsNand.h, 181
- NAND_OP_PERFORM_-
 - PROGRAM
 - vsNand.h, 181
- NAND_OP_PREPARE_TO_-
 - PROGRAM
 - vsNand.h, 181
- NAND_OP_READ_A
 - vsNand.h, 181
- NAND_OP_READ_C
 - vsNand.h, 181
- NAND_OP_READ_SIGNATURE
 - vsNand.h, 181
- NAND_OP_READ_STATUS
 - vsNand.h, 181
- NandCountBits
 - vsNand.h, 182
- NandGetOctets
 - dev1000.h, 65
- NandMingle
 - vsNand.h, 182
- NandPutAddressOctet
 - dev1000.h, 66
- NandPutCommand
 - dev1000.h, 66
- NandPutOctets
 - dev1000.h, 66
- NandSetWaits
 - dev1000.h, 66
- NandSwapBad

- vsNand.h, 182
- nandType
 - FsNandPhys, 22
- NandWaitIdle
 - vsNand.h, 182
- newBranch
 - FmfMeta, 14
- nextFile
 - Player, 25
- nextStep
 - Player, 26
- NFLSH_CF_INT_ENABLE
 - vs1000.h, 164
- NFLSH_CF_LCD_CE_MODE
 - vs1000.h, 165
- NFLSH_CF_NF_RESET
 - vs1000.h, 165
- NFLSH_CF_WAITSTATES
 - vs1000.h, 165
- NFLSH_CP_LPH
 - vs1000.h, 165
- NFLSH_CTRL
 - vs1000.h, 165
- NFLSH_DATA
 - vs1000.h, 165
- NFLSH_DB_POINTER
 - vs1000.h, 165
- NFLSH_DF_ECC_CALC
 - vs1000.h, 165
- NFLSH_DF_ECC_RESET
 - vs1000.h, 165
- NFLSH_DF_ENA_DBUF
 - vs1000.h, 165
- NFLSH_DF_POINTER
 - vs1000.h, 165
- NFLSH_DF_READ
 - vs1000.h, 165
- NFLSH_DSPIF
 - vs1000.h, 166
- NFLSH_ECC_CNT
 - vs1000.h, 166
- NFLSH_LPL
 - vs1000.h, 166
- NFLSH_NB_BYTECNT
 - vs1000.h, 166
- NFLSH_NF_BYTECNT
 - vs1000.h, 166
- NFLSH_NF_POINTER
 - vs1000.h, 166
- NFLSH_NF_READ
 - vs1000.h, 166
- NFLSH_NF_START
 - vs1000.h, 166
- NFLSH_NF_USE_DBUF
 - vs1000.h, 166
- NFLSH_NFIF
 - vs1000.h, 166
- nonFullLimit
 - FsMapperFlash, 19
- NULL
 - stddef.h, 112
 - stdlib.h, 120
- NullHook
 - vs1000.h, 176
- OFF_LIMIT
 - player.h, 98
- offDelay
 - Player, 26
- offsetof
 - stddef.h, 112
- Old
 - EARSPEAKER, 11
- OpenFile
 - vs1000.h, 176
- OpenFileName
 - dev1000.h, 66
- OPERATION_CODE
 - scsi.h, 106
- OTHERS_START
 - vs1000.h, 166
- Output
 - CodecServices, 9
- P
 - FsNandPhys, 22
- P_DATA
 - usbblowlib.h, 139
- P_SETUP
 - usbblowlib.h, 139
- pageSize
 - FsPhysical, 23
- parentDir
 - FATINFO, 13
- PATCH_TEST_UNIT_READY
 - dev1000.h, 59
- PatchMSCPacketFromPC
 - dev1000.h, 66
- pauseOn
 - Player, 26

- payload
 - usbpkt, 31
- peakBitRate
 - CodecServices, 9
- PERIP
 - vs1000.h, 166
- PERIP_IN_X
 - vs1000.h, 167
- perror
 - stdio.h, 117
- physical
 - FsMapper, 17
- physical.h, 94
 - FS_PHYSICAL_VERSION, 95
- physPages
 - FsMapperFlash, 19
- PID_ACK
 - usbblowlib.h, 139
- PID_DATA0
 - usbblowlib.h, 139
- PID_DATA1
 - usbblowlib.h, 140
- PID_IN
 - usbblowlib.h, 140
- PID_NACK
 - usbblowlib.h, 140
- PID_OUT
 - usbblowlib.h, 140
- PID_SETUP
 - usbblowlib.h, 140
- PID_SOF
 - usbblowlib.h, 140
- PID_STALL
 - usbblowlib.h, 140
- pkt
 - USBVARS, 34
- PlayCurrentFile
 - vs1000.h, 176
- Player, 25
 - currentFile, 25
 - ffCount, 25
 - maxClock, 25
 - nextFile, 25
 - nextStep, 26
 - offDelay, 26
 - pauseOn, 26
 - randomOn, 26
 - totalFiles, 26
 - volume, 26
 - volumeOffset, 26
- player
 - player.h, 101
- player.h, 95
 - CleanDisk, 99
 - CsOutput, 99
 - CsRead, 99
 - CsSeek, 99
 - currentKeyMap, 101
 - defSupportedFiles, 101
 - fiveKeyMap, 101
 - ke_earSpeaker, 98
 - ke_earSpeakerToggle, 98
 - ke_ff_faster, 98
 - ke_ff_off, 98
 - ke_ff_slower, 98
 - ke_forward, 98
 - ke_next, 98
 - ke_null, 98
 - ke_pauseToggle, 98
 - ke_powerOff, 98
 - ke_previous, 98
 - ke_randomToggle, 98
 - ke_randomToggleNewSong, 98
 - ke_rewind, 98
 - ke_volumeDown, 98
 - ke_volumeDown2, 98
 - ke_volumeUp, 98
 - ke_volumeUp2, 98
 - KEY_1, 97
 - KEY_2, 97
 - KEY_3, 97
 - KEY_4, 97
 - KEY_5, 97
 - KEY_LONG_ONESHOT, 97
 - KEY_LONG_PRESS, 97
 - KEY_POWER, 97
 - KEY_RELEASED, 97
 - keyCheck, 101
 - keyEvent, 98
 - KeyEventHandler, 99
 - keyOld, 101
 - keyOldTime, 101
 - KeyScan, 99
 - LED1, 97
 - LED2, 98
 - mallocAreaX, 101
 - mallocAreaY, 101
 - MassStorage, 100
 - OFF_LIMIT, 98
 - player, 101

- PlayerVolume, 100
- putch, 100
- ReadGPIO, 100
- RealKeyEventHandler, 100
- RealMassStorage, 100
- RealUSBSuspend, 100
- shiftFourKeyMap, 101
- SHORT_LIMIT, 98
- sixKeyMap, 101
- supportedFiles, 101
- threeKeyMap, 101
- tmpBuf, 101
- USBIsAttached, 100
- USBSuspend, 100
- UserInterfaceIdleHook, 101
- vs1000d_BitReverse, 101
- vs1000d_Latin1, 102
- PlayerVolume
 - player.h, 100
- PlayRange
 - dev1000.h, 66
- PlayRangeSet
 - dev1000.h, 66
- playTimeSamples
 - CodecServices, 9
- playTimeSeconds
 - CodecServices, 9
- playTimeTotal
 - CodecServices, 9
- pow
 - math.h, 88
- PowerOff
 - vs1000.h, 176
- PowerSetVoltages
 - vs1000.h, 176
- printf
 - stdio.h, 117
- ptrdiff_t
 - stddef.h, 112
- putc
 - stdio.h, 117
- putch
 - player.h, 100
 - vs1000.h, 176
- putchar
 - stdio.h, 118
- puthex
 - dev1000.h, 67
- puts
 - stdio.h, 118
- qsort
 - stdlib.h, 120
- QsortLog2
 - stdlib.h, 121
- qsorty
 - mapperflash.h, 81
 - stdlib.h, 120
- rand
 - stdlib.h, 121
- RAND_MAX
 - stdlib.h, 120
- rand_seed
 - stdlib.h, 121
- random
 - stdlib.h, 121
- RANDOM_MAX
 - stdlib.h, 120
- random_state
 - stdlib.h, 121
- randomOn
 - Player, 26
- RC5_CMD_0
 - dev1000.h, 59
- RC5_CMD_1
 - dev1000.h, 59
- RC5_CMD_2
 - dev1000.h, 59
- RC5_CMD_3
 - dev1000.h, 59
- RC5_CMD_4
 - dev1000.h, 59
- RC5_CMD_5
 - dev1000.h, 59
- RC5_CMD_6
 - dev1000.h, 59
- RC5_CMD_7
 - dev1000.h, 59
- RC5_CMD_8
 - dev1000.h, 59
- RC5_CMD_9
 - dev1000.h, 60
- RC5_CMD_ALTERNATE
 - dev1000.h, 60
- RC5_CMD_BALANCE_LEFT
 - dev1000.h, 60
- RC5_CMD_BALANCE_RIGHT
 - dev1000.h, 60
- RC5_CMD_BASS_DOWN
 - dev1000.h, 60

- RC5_CMD_BASS_UP
 - dev1000.h, 60
- RC5_CMD_BRIGHT_DOWN
 - dev1000.h, 60
- RC5_CMD_BRIGHT_UP
 - dev1000.h, 60
- RC5_CMD_CLOCK
 - dev1000.h, 60
- RC5_CMD_COLOR_DOWN
 - dev1000.h, 60
- RC5_CMD_COLOR_UP
 - dev1000.h, 60
- RC5_CMD_CONTRAST_DOWN
 - dev1000.h, 60
- RC5_CMD_CONTRAST_UP
 - dev1000.h, 61
- RC5_CMD_CROSS
 - dev1000.h, 61
- RC5_CMD_DISPLAY
 - dev1000.h, 61
- RC5_CMD_EXPAND
 - dev1000.h, 61
- RC5_CMD_FAST_FORWARD
 - dev1000.h, 61
- RC5_CMD_FAST_REVERSE
 - dev1000.h, 61
- RC5_CMD_FINETUNE_MINUS
 - dev1000.h, 61
- RC5_CMD_FINETUNE_PLUS
 - dev1000.h, 61
- RC5_CMD_LANGUAGE
 - dev1000.h, 61
- RC5_CMD_MENU
 - dev1000.h, 61
- RC5_CMD_MIX
 - dev1000.h, 61
- RC5_CMD_MUTE
 - dev1000.h, 61
- RC5_CMD_ONETWODIGITS
 - dev1000.h, 62
- RC5_CMD_PAUSE
 - dev1000.h, 62
- RC5_CMD_PLAY
 - dev1000.h, 62
- RC5_CMD_PRESET
 - dev1000.h, 62
- RC5_CMD_PROGRAM_DOWN
 - dev1000.h, 62
- RC5_CMD_PROGRAM_UP
 - dev1000.h, 62
- RC5_CMD_RECORD
 - dev1000.h, 62
- RC5_CMD_STANDBY
 - dev1000.h, 62
- RC5_CMD_STOP
 - dev1000.h, 62
- RC5_CMD_STORE
 - dev1000.h, 62
- RC5_CMD_SWITCH
 - dev1000.h, 62
- RC5_CMD_SYSTEM_SELECT
 - dev1000.h, 62
- RC5_CMD_TEXT
 - dev1000.h, 63
- RC5_CMD_TIMER
 - dev1000.h, 63
- RC5_CMD_TREBLE_DOWN
 - dev1000.h, 63
- RC5_CMD_TREBLE_UP
 - dev1000.h, 63
- RC5_CMD_VOLUME_DOWN
 - dev1000.h, 63
- RC5_CMD_VOLUME_UP
 - dev1000.h, 63
- RC5_SYS_CDPLAYER
 - dev1000.h, 63
- RC5_SYS_EXPERIMENTAL19
 - dev1000.h, 63
- RC5_SYS_EXPERIMENTAL7
 - dev1000.h, 63
- RC5_SYS_PREAMP
 - dev1000.h, 63
- RC5_SYS_RECEIVERTUNER
 - dev1000.h, 63
- RC5_SYS_TAPERECORDER
 - dev1000.h, 63
- RC5_SYS_TELETEXT
 - dev1000.h, 64
- RC5_SYS_TVSET
 - dev1000.h, 64
- RC5_SYS_VCR
 - dev1000.h, 64
- Rc5GetFIFO
 - dev1000.h, 67
- Rc5Init
 - dev1000.h, 67
- rd
 - AUDIOPTR, 5
- Read
 - CodecServices, 9

- FsMapper, 17
- FsPhysical, 23
- ReadDiskSector
 - minifat.h, 94
 - vs1000.h, 176
- ReadFile
 - vs1000.h, 177
- ReadGPIO
 - player.h, 100
- ReadIRam
 - dev1000.h, 67
- ReadTimeCount
 - audio.h, 38
- RealDecodeSetupPacket
 - usbblowlib.h, 145
 - vectors.h, 149
- RealInitUSBDescriptors
 - usbblowlib.h, 145
- RealKeyEventHandler
 - player.h, 100
- RealLoadCheck
 - vs1000.h, 177
- RealMassStorage
 - player.h, 100
- RealMSCPacketFromPC
 - usbblowlib.h, 145
 - vectors.h, 149
- RealPlayCurrentFile
 - vs1000.h, 177
- RealPowerOff
 - vs1000.h, 177
- RealScsiTaskHandler
 - scsi.h, 111
- RealSetRate
 - audio.h, 38
- RealSetVolume
 - audio.h, 38
- RealUSBHandler
 - usbblowlib.h, 145
 - vectors.h, 149
- RealUSBSuspend
 - player.h, 100
- register
 - vstypes.h, 188
- Reinitialize
 - FsPhysical, 23
- remove
 - stdio.h, 118
- rename
 - stdio.h, 118
- res__allocationLength
 - scsicdb6variant2, 30
 - scsicdb6variant3, 31
- res__lbab3
 - scsicdb10variant1, 27
- reservedAndBadBlock
 - FmfMeta, 14
- Restart
 - vs1000.h, 177
- rewind
 - stdio.h, 118
- rightVol
 - AUDIOPTR, 5
- RING_BUF_SIZE
 - usbblowlib.h, 140
- RingBufCopyX
 - usbblowlib.h, 145
- romfont.h, 102
 - fontData, 104
 - fontPtrs, 104
 - SYMBOL_ALPHA, 102
 - SYMBOL_ALPHA_LOW, 102
 - SYMBOL_DIGIT, 102
 - SYMBOL_DISKDRIVE, 103
 - SYMBOL_KATAKANA, 103
 - SYMBOL_LEFTSPK, 103
 - SYMBOL_NOTE, 103
 - SYMBOL_NUMBER, 103
 - SYMBOL_PAUSE, 103
 - SYMBOL_PLAY, 103
 - SYMBOL_PU, 103
 - SYMBOL_RIGHTSPK, 103
 - SYMBOL_SPACE, 103
 - SYMBOL_STOP, 103
 - SYMBOL_USB, 103
- root
 - FsMapperFlash, 19
 - FsMapperTiny, 21
- rootStart
 - FATINFO, 13
- RX_INT
 - usbblowlib.h, 140
- s_int16
 - vstypes.h, 189
- s_int32
 - vstypes.h, 189
- sampleRate
 - CodecServices, 10
- SCI_DEBUG

- vs1000.h, 167
- SCI_STATUS
 - vs1000.h, 167
- SCI_SYSTEM
 - vs1000.h, 167
- SCISTF_ANA_PDOWN
 - vs1000.h, 167
- SCISTF_ANADRV_PDOWN
 - vs1000.h, 167
- SCISTF_REGU_CLOCK
 - vs1000.h, 167
- SCISTF_REGU_POWERBUT
 - vs1000.h, 167
- SCISTF_REGU_POWERLOW
 - vs1000.h, 167
- SCISTF_REGU_SHUTDOWN
 - vs1000.h, 167
- SCISTF_SLOW_CLKMODE
 - vs1000.h, 167
- SCISTF_USB_DDR
 - vs1000.h, 167
- SCISTF_USB_DIFF_ENA
 - vs1000.h, 168
- SCISTF_USB_DN
 - vs1000.h, 168
- SCISTF_USB_DN_OUT
 - vs1000.h, 168
- SCISTF_USB_DP
 - vs1000.h, 168
- SCISTF_USB_DP_OUT
 - vs1000.h, 168
- SCISTF_USB_PULLUP_ENA
 - vs1000.h, 168
- SCISTF_VCM_DISABLE
 - vs1000.h, 168
- SCISTF_VCM_OVERLOAD
 - vs1000.h, 168
- SCISYSF_AVDD
 - vs1000.h, 168
- SCISYSF_CLKDIV
 - vs1000.h, 168
- SCISYSF_CVDD
 - vs1000.h, 168
- SCISYSF_IOVDD
 - vs1000.h, 168
- scsi.h, 104
 - ATAPI_READ_FORMAT_-
CAPACITIES, 106
 - DiskDataReceived, 110
 - DiskProtocolCommand, 110
 - OPERATION_CODE, 106
 - RealScsiTaskHandler, 111
 - SCSI_DATA_FROM_HOST,
110
 - SCSI_DATA_TO_HOST, 110
 - SCSI_FORMAT_UNIT, 106
 - SCSI_INQUIRY, 106
 - SCSI_INVALID_CBW, 110
 - SCSI_MODE_SELECT, 106
 - SCSI_MODE_SENSE_10, 106
 - SCSI_MODE_SENSE_6, 106
 - SCSI_OK, 110
 - SCSI_PHASE_ERROR, 110
 - SCSI_PREVENT_ALLOW_-
MEDIUM_REMOVAL,
106
 - SCSI_READ_10, 107
 - SCSI_READ_12, 107
 - SCSI_READ_6, 107
 - SCSI_READ_CAPACITY_10,
107
 - SCSI_READ_CAPACITY_16,
107
 - SCSI_READ_CAPACITY_-
16_2, 107
 - SCSI_READY_FOR_-
COMMAND, 110
 - SCSI_RECEIVE_-
DIAGNOSTIC_RESULTS,
107
 - SCSI_REPORT_LUNS, 107
 - SCSI_REQUEST_ERROR, 110
 - SCSI_REQUEST_SENSE, 107
 - SCSI_SEND_DIAGNOSTIC,
107
 - SCSI_SEND_STATUS, 110
 - SCSI_START_STOP_UNIT,
107
 - SCSI_SYNCHRONIZE_-
CACHE, 107
 - SCSI_TEST_UNIT_READY,
108
 - SCSI_TRANSMITTING, 110
 - SCSI_UNINITIALIZED, 110
 - SCSI_VERIFY, 108
 - SCSI_WRITE_10, 108
 - SCSI_WRITE_12, 108
 - SCSI_WRITE_6, 108
 - ScsiInquiryCdb, 109
 - ScsiModeSense6Cdb, 109

- ScsiOrBlock, 111
- ScsiRead10Cdb, 109
- ScsiRequestSenseCdb, 109
- ScsiReset, 111
- SCSIStageEnum, 110
- ScsiState, 111
- SCSIStatusEnum, 110
- ScsiTaskHandler, 111
- ScsiWrite10Cdb, 109
- SK_ABORTED_COMMAND, 108
- SK_BLANK_CHECK, 108
- SK_COPY_ABORTED, 108
- SK_DATA_PROTECT, 108
- SK_EQUAL, 108
- SK_HARDWARE_ERROR, 108
- SK_ILLEGAL_REQUEST, 108
- SK_MEDIUM_ERROR, 109
- SK_MISCOMPARE, 109
- SK_NO_SENSE, 109
- SK_NOT_READY, 109
- SK_RECOVERED_ERROR, 109
- SK_UNIT_ATTENTION, 109
- SK_VENDOR_SPECIFIC, 109
- SK_VOLUME_OVERFLOW, 109
- SCSI_DATA_FROM_HOST
scsi.h, 110
- SCSI_DATA_TO_HOST
scsi.h, 110
- SCSI_FORMAT_UNIT
scsi.h, 106
- SCSI_INQUIRY
scsi.h, 106
- SCSI_INVALID_CBW
scsi.h, 110
- SCSI_MODE_SELECT
scsi.h, 106
- SCSI_MODE_SENSE_10
scsi.h, 106
- SCSI_MODE_SENSE_6
scsi.h, 106
- SCSI_OK
scsi.h, 110
- SCSI_PHASE_ERROR
scsi.h, 110
- SCSI_PREVENT_ALLOW_-
MEDIUM_REMOVAL
scsi.h, 106
- SCSI_READ_10
scsi.h, 107
- SCSI_READ_12
scsi.h, 107
- SCSI_READ_6
scsi.h, 107
- SCSI_READ_CAPACITY_10
scsi.h, 107
- SCSI_READ_CAPACITY_16
scsi.h, 107
- SCSI_READ_CAPACITY_16_2
scsi.h, 107
- SCSI_READY_FOR_COMMAND
scsi.h, 110
- SCSI_RECEIVE_DIAGNOSTIC_-
RESULTS
scsi.h, 107
- SCSI_REPORT_LUNS
scsi.h, 107
- SCSI_REQUEST_ERROR
scsi.h, 110
- SCSI_REQUEST_SENSE
scsi.h, 107
- SCSI_SEND_DIAGNOSTIC
scsi.h, 107
- SCSI_SEND_STATUS
scsi.h, 110
- SCSI_START_STOP_UNIT
scsi.h, 107
- SCSI_SYNCHRONIZE_CACHE
scsi.h, 107
- SCSI_TEST_UNIT_READY
scsi.h, 108
- SCSI_TRANSMITTING
scsi.h, 110
- SCSI_UNINITIALIZED
scsi.h, 110
- SCSI_VERIFY
scsi.h, 108
- SCSI_WRITE_10
scsi.h, 108
- SCSI_WRITE_12
scsi.h, 108
- SCSI_WRITE_6
scsi.h, 108
- scsicdb10variant1, 27
control__null, 27
lbab0__res, 27
lbab2__lbab1, 27
length__opcode, 27

- res__lbab3, 27
- wLength, 27
- scsicdb10variant2, 28
 - control__null, 28
 - flags__lbab3, 28
 - lbab0__res, 28
 - lbab2__lbab1, 28
 - length__opcode, 28
 - wLength, 28
- scsicdb6variant1, 29
 - allocationLength, 29
 - control__null, 29
 - flags__pageCode, 29
 - length__opcode, 29
- scsicdb6variant2, 29
 - control__null, 30
 - flags__pageCode, 30
 - length__opcode, 30
 - res__allocationLength, 30
- scsicdb6variant3, 30
 - control__null, 30
 - flags__res, 30
 - length__opcode, 31
 - res__allocationLength, 31
- ScsiInquiryCdb
 - scsi.h, 109
- ScsiModeSense6Cdb
 - scsi.h, 109
- ScsiOrBlock
 - scsi.h, 111
- ScsiRead10Cdb
 - scsi.h, 109
- ScsiRequestSenseCdb
 - scsi.h, 109
- ScsiReset
 - scsi.h, 111
- SCSIStageEnum
 - scsi.h, 110
- ScsiState
 - scsi.h, 111
- SCSIStatusEnum
 - scsi.h, 110
- ScsiTaskHandler
 - scsi.h, 111
- ScsiTestUnitReady
 - dev1000.h, 67
- ScsiWrite10Cdb
 - scsi.h, 109
- Seek
 - CodecServices, 10
 - vs1000.h, 177
- SEEK_CUR
 - stdio.h, 115
- SEEK_END
 - stdio.h, 115
- SEEK_SET
 - stdio.h, 115
- SetHookFunction
 - vs1000.h, 177
- SetRate
 - audio.h, 38
- Setting
 - EARSPEAKER, 11
- SETUP_INFO
 - usbblowlib.h, 140
- SetVolume
 - audio.h, 38
- shiftFourKeyMap
 - player.h, 101
- SHORT_LIMIT
 - player.h, 98
- sin
 - math.h, 88
- sinh
 - math.h, 88
- SinTest
 - vs1000.h, 177
- sixKeyMap
 - player.h, 101
- size
 - FRAGMENT, 15
- size_t
 - stddef.h, 112
 - stdlib.h, 120
- SK_ABORTED_COMMAND
 - scsi.h, 108
- SK_BLANK_CHECK
 - scsi.h, 108
- SK_COPY_ABORTED
 - scsi.h, 108
- SK_DATA_PROTECT
 - scsi.h, 108
- SK_EQUAL
 - scsi.h, 108
- SK_HARDWARE_ERROR
 - scsi.h, 108
- SK_ILLEGAL_REQUEST
 - scsi.h, 108
- SK_MEDIUM_ERROR
 - scsi.h, 109

- SK_MISCOMPARE
 - scsi.h, 109
- SK_NO_SENSE
 - scsi.h, 109
- SK_NOT_READY
 - scsi.h, 109
- SK_RECOVERED_ERROR
 - scsi.h, 109
- SK_UNIT_ATTENTION
 - scsi.h, 109
- SK_VENDOR_SPECIFIC
 - scsi.h, 109
- SK_VOLUME_OVERFLOW
 - scsi.h, 109
- Skip
 - CodecServices, 10
- skipped
 - FsMapperFlash, 19
- Sleep
 - vs1000.h, 178
- SOF_INT
 - usbblowlib.h, 140
- Spectrum
 - CodecServices, 10
- SPIO_CLKCONFIG
 - vs1000.h, 169
- SPIO_CONFIG
 - vs1000.h, 169
- SPIO_DATA
 - vs1000.h, 169
- SPIO_FSYNC
 - vs1000.h, 169
- SPIO_STATUS
 - vs1000.h, 169
- SPI_CC_CLKDIV
 - vs1000.h, 169
- SPI_CF_DLEN
 - vs1000.h, 169
- SPI_CF_DLEN16
 - vs1000.h, 169
- SPI_CF_DLEN8
 - vs1000.h, 169
- SPI_CF_FALLXCS
 - vs1000.h, 169
- SPI_CF_FSIDLE0
 - vs1000.h, 169
- SPI_CF_FSIDLE1
 - vs1000.h, 169
- SPI_CF_INTXCS
 - vs1000.h, 170
- SPI_CF_MASTER
 - vs1000.h, 170
- SPI_CF_RISEXCS
 - vs1000.h, 170
- SPI_CF_SLAVE
 - vs1000.h, 170
- SPI_ST_BREAK
 - vs1000.h, 170
- SPI_ST_RXFULL
 - vs1000.h, 170
- SPI_ST_RXORUN
 - vs1000.h, 170
- SPI_ST_TXFULL
 - vs1000.h, 170
- SPI_ST_TXRUNNING
 - vs1000.h, 170
- SPI_ST_TXURUN
 - vs1000.h, 170
- SpiBoot
 - vs1000.h, 178
- SpiDelay
 - vs1000.h, 178
- SpiLoad
 - vs1000.h, 178
- SpiSendClocks
 - dev1000.h, 67
- SpiSendReceive
 - vs1000.h, 178
- SpiSendReceiveMmc
 - dev1000.h, 67
- sprintf
 - stdio.h, 118
- sqrt
 - math.h, 88
- SqrtF
 - math.h, 88
- SqrtF32
 - math.h, 88
- SqrtI
 - math.h, 88
 - vstypes.h, 188
- SqrtI32
 - math.h, 88
 - vstypes.h, 188
- srand
 - stdlib.h, 121
- srandom
 - stdlib.h, 121
- sscanf
 - stdio.h, 118

- STACK_SIZE
 - vs1000.h, 170
- STACK_START
 - vs1000.h, 170
- start
 - FRAGMENT, 15
- stdarg.h, 111
 - va_arg, 111
 - va_end, 111
 - va_list, 112
 - va_start, 111
- stddef.h, 112
 - NULL, 112
 - offsetof, 112
 - ptrdiff_t, 112
 - size_t, 112
 - wchar_t, 113
- stderr
 - stdio.h, 115
- stdin
 - stdio.h, 115
- stdio.h, 113
 - clearerr, 116
 - EOF, 115
 - fclose, 116
 - feof, 116
 - ferror, 116
 - fflush, 116
 - fgetc, 116
 - fgetpos, 116
 - fgets, 116
 - FILE, 115
 - FILENAME_MAX, 115
 - fopen, 116
 - FOPEN_MAX, 115
 - fpos_t, 115
 - fprintf, 116
 - fputc, 116
 - fputs, 116
 - fread, 117
 - freopen, 117
 - fseek, 117
 - fsetpos, 117
 - ftell, 117
 - fwrite, 117
 - getc, 117
 - getchar, 117
 - gets, 117
 - perror, 117
 - printf, 117
 - putc, 117
 - putchar, 118
 - puts, 118
 - remove, 118
 - rename, 118
 - rewind, 118
 - SEEK_CUR, 115
 - SEEK_END, 115
 - SEEK_SET, 115
 - sprintf, 118
 - sscanf, 118
 - stderr, 115
 - stdin, 115
 - stdout, 115
 - tinyprintf, 118
 - tinyprintf, 118
 - tinysprintf, 118
 - ungetc, 118
- stdlib.h, 118
 - abort, 119
 - abs, 119
 - atoi, 120
 - CountBitsLong, 120
 - exit, 120
 - EXIT_FAILURE, 119
 - EXIT_SUCCESS, 120
 - labs, 120
 - NULL, 120
 - qsort, 120
 - QsortLog2, 121
 - qsority, 120
 - rand, 121
 - RAND_MAX, 120
 - rand_seed, 121
 - random, 121
 - RANDOM_MAX, 120
 - random_state, 121
 - size_t, 120
 - srand, 121
 - srandom, 121
 - strtol, 121
 - strtoul, 120
- stdout
 - stdio.h, 115
- StereoCopy
 - audio.h, 38
- strcat
 - string.h, 125
- strchr
 - string.h, 125

- strcmp
 - string.h, 125
- strcoll
 - string.h, 123
- strcpy
 - string.h, 125
- strcspn
 - string.h, 125
- strerror
 - string.h, 125
- string.h, 121
 - memchr, 123
 - memclearXY, 123
 - memcmp, 123
 - memcmpY, 123
 - MemCopyPackedBigEndian, 123
 - memcpy, 123
 - memcpyXY, 124
 - memcpyYX, 124
 - memcpyYY, 124
 - memmove, 124
 - memset, 124
 - memsetY, 124
 - memswap, 124
 - memswapxy, 124
 - memswapy, 124
 - strcat, 125
 - strchr, 125
 - strcmp, 125
 - strcoll, 123
 - strcpy, 125
 - strcspn, 125
 - strerror, 125
 - strlen, 125
 - strncat, 125
 - strncmp, 125
 - strncpy, 125
 - strpbrk, 125
 - strrchr, 125
 - strspn, 126
 - strstr, 126
 - strtok, 126
 - strxfrm, 126
- strlen
 - string.h, 125
- strncat
 - string.h, 125
- strncmp
 - string.h, 125
- strncpy
 - string.h, 125
- string.h, 125
 - string.h, 125
- strpbrk
 - string.h, 125
- strrchr
 - string.h, 125
- strspn
 - string.h, 126
- strstr
 - string.h, 126
- strtok
 - string.h, 126
- strtol
 - stdlib.h, 121
- strtoul
 - stdlib.h, 120
- strxfrm
 - string.h, 126
- supportedFiles
 - player.h, 101
- supportedSuffixes
 - FATINFO, 13
- Suspend7
 - dev1000.h, 68
- SwapWord
 - usbblowlib.h, 146
- SYMBOL_ALPHA
 - romfont.h, 102
- SYMBOL_ALPHA_LOW
 - romfont.h, 102
- SYMBOL_DIGIT
 - romfont.h, 102
- SYMBOL_DISKDRIVE
 - romfont.h, 103
- SYMBOL_KATAKANA
 - romfont.h, 103
- SYMBOL_LEFTSPK
 - romfont.h, 103
- SYMBOL_NOTE
 - romfont.h, 103
- SYMBOL_NUMBER
 - romfont.h, 103
- SYMBOL_PAUSE
 - romfont.h, 103
- SYMBOL_PLAY
 - romfont.h, 103
- SYMBOL_PU
 - romfont.h, 103
- SYMBOL_RIGHTSPK
 - romfont.h, 103
- SYMBOL_SPACE
 - romfont.h, 103

- romfont.h, 103
- SYMBOL_STOP
 - romfont.h, 103
- SYMBOL_USB
 - romfont.h, 103
- tan
 - math.h, 88
- tanh
 - math.h, 88
- Tell
 - CodecServices, 10
 - vs1000.h, 178
- threeKeyMap
 - player.h, 101
- timeCount
 - audio.h, 40
- TIMER_CONFIG
 - vs1000.h, 171
- TIMER_ENABLE
 - vs1000.h, 171
- TIMER_T0CNTH
 - vs1000.h, 171
- TIMER_T0CNTL
 - vs1000.h, 171
- TIMER_T0H
 - vs1000.h, 171
- TIMER_T0L
 - vs1000.h, 171
- TIMER_T1CNTH
 - vs1000.h, 171
- TIMER_T1CNTL
 - vs1000.h, 171
- TIMER_T1H
 - vs1000.h, 171
- TIMER_T1L
 - vs1000.h, 171
- TIMER_TICKS
 - audio.h, 37
- timeToRemovePDown2
 - audio.h, 40
- tinyprintf
 - stdio.h, 118
- tinyprintf
 - stdio.h, 118
- tinysprintf
 - stdio.h, 118
- tmpBuf
 - player.h, 101
- tolower
 - ctype.h, 51
- totalFiles
 - Player, 26
- totbytes
 - USBVARS, 34
- totSize
 - FATINFO, 13
- toupper
 - ctype.h, 51
- TX_HOLD_INT
 - usbblowlib.h, 141
- TX_INT
 - usbblowlib.h, 141
- u_int16
 - vstypes.h, 189
- u_int32
 - vstypes.h, 189
- UART_DATA
 - vs1000.h, 171
- UART_DATAH
 - vs1000.h, 171
- UART_DIV
 - vs1000.h, 172
- UART_ST_RXFULL
 - vs1000.h, 172
- UART_ST_RXORUN
 - vs1000.h, 172
- UART_ST_TXFULL
 - vs1000.h, 172
- UART_ST_TXRUNNING
 - vs1000.h, 172
- UART_STATUS
 - vs1000.h, 172
- uartByteSpeed
 - audio.h, 40
- UartDiv
 - audio.h, 38
- uiTime
 - audio.h, 40
- uiTrigger
 - audio.h, 40
- underflow
 - AUDIOPTR, 5
- ungetc
 - stdio.h, 118
- UnsupportedFile
 - vs1000.h, 178
- unused
 - FmfMeta, 14

- USB
 - usbblowlib.h, 149
 - usb.h, 126
 - D12_FULLEEMPTY, 128
 - D12_INT_BUSRESET, 128
 - D12_INT_ENDP0IN, 128
 - D12_INT_ENDP0OUT, 128
 - D12_INT_ENDP1IN, 128
 - D12_INT_ENDP1OUT, 128
 - D12_INT_ENDP2IN, 128
 - D12_INT_ENDP2OUT, 128
 - D12_INT_EOT, 129
 - D12_INT_SUSPENDCHANGE, 129
 - D12_SETUPPACKET, 129
 - D12_STALL, 129
 - EP0_PACKET_SIZE, 129
 - EP0_RX_FIFO_SIZE, 129
 - EP0_TX_FIFO_SIZE, 129
 - EP1_PACKET_SIZE, 129
 - EP1_RX_FIFO_SIZE, 129
 - EP1_TX_FIFO_SIZE, 129
 - EP2_PACKET_SIZE, 129
 - EP2_RX_FIFO_SIZE, 129
 - EP2_TX_FIFO_SIZE, 130
 - USB_ACCESS_BUF, 130
 - USB_ACK_SETUP, 130
 - USB_CLASS_CODE_TEST_-
CLASS_DEVICE, 130
 - USB_CLASS_REQUEST, 130
 - USB_CLEAR_BUF, 130
 - USB_CONFIGURATION_-
DESCRIPTOR_TYPE,
130
 - USB_DEVICE_-
DESCRIPTOR_TYPE,
130
 - USB_ENDP_ST_CI, 130
 - USB_ENDP_ST_CO, 130
 - USB_ENDP_ST_E1I, 130
 - USB_ENDP_ST_E1O, 130
 - USB_ENDP_ST_E2I, 131
 - USB_ENDP_ST_E2O, 131
 - USB_ENDPOINT_-
DESCRIPTOR_TYPE,
131
 - USB_ENDPOINT_-
DIRECTION_MASK,
131
 - USB_ENDPOINT_HALT, 131
 - USB_ENDPOINT_REQUEST,
131
 - USB_INTERFACE_-
DESCRIPTOR_TYPE,
131
 - USB_INTERFACE_REQUEST,
131
 - USB_MAX_ENDPOINTS, 131
 - USB_POWER_-
DESCRIPTOR_TYPE,
131
 - USB_PROTOCOL_CODE_-
TEST_CLASS_D12, 131
 - USB_READ_FRAME_-
NUMBER, 132
 - USB_READ_INT, 132
 - USB_RECIPIENT, 132
 - USB_RECIPIENT_DEVICE,
132
 - USB_RECIPIENT_-
ENDPOINT, 132
 - USB_RECIPIENT_-
INTERFACE, 132
 - USB_REQUEST_CLEAR_-
FEATURE, 132
 - USB_REQUEST_GET_-
CONFIGURATION, 132
 - USB_REQUEST_GET_-
DESCRIPTOR, 132
 - USB_REQUEST_GET_-
INTERFACE, 132
 - USB_REQUEST_GET_-
STATUS, 132
 - USB_REQUEST_MASK, 132
 - USB_REQUEST_SET_-
ADDRESS, 133
 - USB_REQUEST_SET_-
CONFIGURATION, 133
 - USB_REQUEST_SET_-
DESCRIPTOR, 133
 - USB_REQUEST_SET_-
FEATURE, 133
 - USB_REQUEST_SET_-
INTERFACE, 133
 - USB_REQUEST_SYNC_-
FRAME, 133
 - USB_REQUEST_TYPE_-
MASK, 133
 - USB_SELECT_ENDP_CI, 133
 - USB_SELECT_ENDP_CO, 133

- USB_SELECT_ENDP_E1I, 133
- USB_SELECT_ENDP_E1O, 133
- USB_SELECT_ENDP_E2I, 133
- USB_SELECT_ENDP_E2O, 134
- USB_SEND_RESUME, 134
- USB_SET_ADDRESS_ENA, 134
- USB_SET_DMA, 134
- USB_SET_ENDP_ENA, 134
- USB_SET_MODE, 134
- USB_SP_INDEX, 134
- USB_SP_LENGTH, 134
- USB_SP_REQUEST, 134
- USB_SP_VALUE, 134
- USB_STANDARD_REQUEST, 134
- USB_STRING_-
 DESCRIPTOR_TYPE, 134
- USB_SUBCLASS_CODE_-
 TEST_CLASS_D12, 135
- USB_VALIDATE_BUF, 135
- USB_VENDOR_REQUEST, 135
- USB_ACCESS_BUF
 usb.h, 130
- USB_ACK_SETUP
 usb.h, 130
- USB_AUDIO
 usbblowlib.h, 141
- USB_BASE
 vs1000.h, 172
- USB_CLASS_CODE_TEST_-
 CLASS_DEVICE
 usb.h, 130
- USB_CLASS_REQUEST
 usb.h, 130
- USB_CLEAR_BUF
 usb.h, 130
- USB_CONFIG
 usbblowlib.h, 141
- USB_CONFIGURATION_-
 DESCRIPTOR_TYPE
 usb.h, 130
- USB_CONTROL
 usbblowlib.h, 141
- USB_DEVICE_DESCRIPTOR_-
 TYPE
- usb.h, 130
- USB_ENDP_ST_CI
 usb.h, 130
- USB_ENDP_ST_CO
 usb.h, 130
- USB_ENDP_ST_E1I
 usb.h, 130
- USB_ENDP_ST_E1O
 usb.h, 130
- USB_ENDP_ST_E2I
 usb.h, 131
- USB_ENDP_ST_E2O
 usb.h, 131
- USB_ENDPOINT_-
 DESCRIPTOR_TYPE
 usb.h, 131
- USB_ENDPOINT_DIRECTION_-
 MASK
 usb.h, 131
- USB_ENDPOINT_HALT
 usb.h, 131
- USB_ENDPOINT_REQUEST
 usb.h, 131
- USB_EP_SEND0
 usbblowlib.h, 141
- USB_EP_SEND1
 usbblowlib.h, 141
- USB_EP_SEND2
 usbblowlib.h, 141
- USB_EP_SEND3
 usbblowlib.h, 141
- USB_EP_ST0
 usbblowlib.h, 141
- USB_EP_ST1
 usbblowlib.h, 141
- USB_EP_ST2
 usbblowlib.h, 142
- USB_EP_ST3
 usbblowlib.h, 142
- USB_INTERFACE_-
 DESCRIPTOR_TYPE
 usb.h, 131
- USB_INTERFACE_REQUEST
 usb.h, 131
- USB_MASS_STORAGE
 usbblowlib.h, 142
- USB_MAX_ENDPOINTS
 usb.h, 131
- USB_POWER_DESCRIPTOR_-
 TYPE

- usb.h, 131
- USB_PROTOCOL_CODE_-
TEST_CLASS_D12
usb.h, 131
- USB_RDPTR
usb_lowlib.h, 142
- USB_READ_FRAME_NUMBER
usb.h, 132
- USB_READ_INT
usb.h, 132
- USB_RECIPIENT
usb.h, 132
- USB_RECIPIENT_DEVICE
usb.h, 132
- USB_RECIPIENT_ENDPOINT
usb.h, 132
- USB_RECIPIENT_INTERFACE
usb.h, 132
- USB_RECV_MEM
vs1000.h, 172
- USB_REQUEST_CLEAR_-
FEATURE
usb.h, 132
- USB_REQUEST_GET_-
CONFIGURATION
usb.h, 132
- USB_REQUEST_GET_-
DESCRIPTOR
usb.h, 132
- USB_REQUEST_GET_-
INTERFACE
usb.h, 132
- USB_REQUEST_GET_STATUS
usb.h, 132
- USB_REQUEST_MASK
usb.h, 132
- USB_REQUEST_SET_ADDRESS
usb.h, 133
- USB_REQUEST_SET_-
CONFIGURATION
usb.h, 133
- USB_REQUEST_SET_-
DESCRIPTOR
usb.h, 133
- USB_REQUEST_SET_FEATURE
usb.h, 133
- USB_REQUEST_SET_-
INTERFACE
usb.h, 133
- USB_REQUEST_SYNC_FRAME
usb.h, 133
- USB_REQUEST_TYPE_MASK
usb.h, 133
- USB_SELECT_ENDP_CI
usb.h, 133
- USB_SELECT_ENDP_CO
usb.h, 133
- USB_SELECT_ENDP_E1I
usb.h, 133
- USB_SELECT_ENDP_E1O
usb.h, 133
- USB_SELECT_ENDP_E2I
usb.h, 133
- USB_SELECT_ENDP_E2O
usb.h, 134
- USB_SEND_MEM
vs1000.h, 172
- USB_SEND_RESUME
usb.h, 134
- USB_SET_ADDRESS_ENA
usb.h, 134
- USB_SET_DMA
usb.h, 134
- USB_SET_ENDP_ENA
usb.h, 134
- USB_SET_MODE
usb.h, 134
- USB_SP_INDEX
usb.h, 134
- USB_SP_LENGTH
usb.h, 134
- USB_SP_REQUEST
usb.h, 134
- USB_SP_VALUE
usb.h, 134
- USB_STANDARD_REQUEST
usb.h, 134
- USB_STATUS
usb_lowlib.h, 142
- USB_STF_BUS_RESET
usb_lowlib.h, 142
- USB_STF_IN_BULK
usb_lowlib.h, 142
- USB_STF_IN_EMPTY
usb_lowlib.h, 142
- USB_STF_IN_ENABLE
usb_lowlib.h, 142
- USB_STF_IN_FORCE_NACK
usb_lowlib.h, 142
- USB_STF_IN_INT

- usbllib.h, 142
- USB_STF_IN_ISO
 - usbllib.h, 142
- USB_STF_IN_NACK_SENT
 - usbllib.h, 143
- USB_STF_IN_STALL
 - usbllib.h, 143
- USB_STF_IN_STALL_SENT
 - usbllib.h, 143
- USB_STF_NAK
 - usbllib.h, 143
- USB_STF_OUT_BULK
 - usbllib.h, 143
- USB_STF_OUT_ENABLE
 - usbllib.h, 143
- USB_STF_OUT_EP_SIZE
 - usbllib.h, 143
- USB_STF_OUT_INT
 - usbllib.h, 143
- USB_STF_OUT_ISO
 - usbllib.h, 143
- USB_STF_OUT_STALL
 - usbllib.h, 143
- USB_STF_OUT_STALL_SENT
 - usbllib.h, 143
- USB_STF_RX
 - usbllib.h, 143
- USB_STF_SETUP
 - usbllib.h, 144
- USB_STF_SOF
 - usbllib.h, 144
- USB_STF_TX_EMPTY
 - usbllib.h, 144
- USB_STF_TX_READY
 - usbllib.h, 144
- USB_STM_LAST_EP
 - usbllib.h, 144
- USB_STRING_DESCRIPTOR_-
TYPE
 - usb.h, 134
- USB_SUBCLASS_CODE_TEST_-
CLASS_D12
 - usb.h, 135
- USB_VALIDATE_BUF
 - usb.h, 135
- USB_VENDOR_REQUEST
 - usb.h, 135
- USB_WRPTR
 - usbllib.h, 144
- USBCheckForSetupPacket
 - usbllib.h, 146
- USBContinueTransmission
 - usbllib.h, 146
- USBHandler
 - usbllib.h, 146
 - vectors.h, 149
- USBsAttached
 - player.h, 100
 - usbllib.h, 146
- USBsDetached
 - usbllib.h, 146
- USBsEndpointStalled
 - usbllib.h, 146
- usbllib.h, 135
 - AUDIO_DELAY_FRAMES, 138
 - AUDIO_DELAY_FRAMES_-
STR, 138
 - AUDIO_ISOC_OUT_EP, 138
 - AudioPacketFromUSB, 144
 - BRESET_INT, 138
 - DecodeSetupPacket, 144
 - DiskProtocolError, 144
 - DT_CONFIGURATION, 138
 - DT_DEVICE, 138
 - DT_LANGUAGES, 138
 - DT_MODEL, 138
 - DT_SERIAL, 138
 - DT_VENDOR, 139
 - ENDPOINT_SIZE_0, 139
 - ENDPOINT_SIZE_1, 139
 - InitUSB, 144
 - InitUSBDescriptors, 145
 - MSC_BULK_IN_ENDPOINT,
139
 - MSC_BULK_OUT_-
ENDPOINT, 139
 - MSCPacketFromPC, 145
 - MscSendCsw, 145
 - NAK_SENT_INT, 139
 - P_DATA, 139
 - P_SETUP, 139
 - PID_ACK, 139
 - PID_DATA0, 139
 - PID_DATA1, 140
 - PID_IN, 140
 - PID_NACK, 140
 - PID_OUT, 140
 - PID_SETUP, 140
 - PID_SOF, 140
 - PID_STALL, 140

- RealDecodeSetupPacket, 145
- RealInitUSBDescriptors, 145
- RealMSCPacketFromPC, 145
- RealUSBHandler, 145
- RING_BUF_SIZE, 140
- RingBufCopyX, 145
- RX_INT, 140
- SETUP_INFO, 140
- SOF_INT, 140
- SwapWord, 146
- TX_HOLD_INT, 141
- TX_INT, 141
- USB, 149
- USB_AUDIO, 141
- USB_CONFIG, 141
- USB_CONTROL, 141
- USB_EP_SEND0, 141
- USB_EP_SEND1, 141
- USB_EP_SEND2, 141
- USB_EP_SEND3, 141
- USB_EP_ST0, 141
- USB_EP_ST1, 141
- USB_EP_ST2, 142
- USB_EP_ST3, 142
- USB_MASS_STORAGE, 142
- USB_RDPTR, 142
- USB_STATUS, 142
- USB_STF_BUS_RESET, 142
- USB_STF_IN_BULK, 142
- USB_STF_IN_EMPTY, 142
- USB_STF_IN_ENABLE, 142
- USB_STF_IN_FORCE_NACK, 142
- USB_STF_IN_INT, 142
- USB_STF_IN_ISO, 142
- USB_STF_IN_NACK_SENT, 143
- USB_STF_IN_STALL, 143
- USB_STF_IN_STALL_SENT, 143
- USB_STF_NAK, 143
- USB_STF_OUT_BULK, 143
- USB_STF_OUT_ENABLE, 143
- USB_STF_OUT_EP_SIZE, 143
- USB_STF_OUT_INT, 143
- USB_STF_OUT_ISO, 143
- USB_STF_OUT_STALL, 143
- USB_STF_OUT_STALL_SENT, 143
- USB_STF_RX, 143
- USB_STF_SETUP, 144
- USB_STF_SOF, 144
- USB_STF_TX_EMPTY, 144
- USB_STF_TX_READY, 144
- USB_STM_LAST_EP, 144
- USB_WRPTR, 144
- USBCheckForSetupPacket, 146
- USBContinueTransmission, 146
- USBHandler, 146
- USBIsAttached, 146
- USBIsDetached, 146
- USBIsEndpointStalled, 146
- USBPacket, 144
- USBReceivePacket, 147
- USBResetEndpoint, 147
- USBResetStall, 147
- USBSendZeroLengthPacketToEndpoint0, 147
- USBSingleStallEndpoint, 147
- USBStallEndpoint, 148
- USBStartTransmission, 148
- USBWantsSuspend, 148
- USBXmitLeft, 148
- USBPacket
 - usbblowlib.h, 144
- usbpkt, 31
 - length, 31
 - payload, 31
- USBReceivePacket
 - usbblowlib.h, 147
- USBResetEndpoint
 - usbblowlib.h, 147
- USBResetStall
 - usbblowlib.h, 147
- USBSendZeroLengthPacketToEndpoint0
 - usbblowlib.h, 147
- USBSingleStallEndpoint
 - usbblowlib.h, 147
- USBStallEndpoint
 - usbblowlib.h, 148
- USBStartTransmission
 - usbblowlib.h, 148
- USBSuspend
 - player.h, 100
- USBVARS, 32
 - configuration, 32
 - configurationDescriptorSize, 32
 - descriptorTable, 32
 - EPReady, 33

- ExtraZeroLengthPacketNeeded, 33
- interfaces, 33
- lastSofFill, 33
- lastSofTime, 33
- lastSofTimeout, 33
- pkt, 34
- totbytes, 34
- XmitBuf, 34
- XmitLength, 34
- USBWantsSuspend
 - usbblowlib.h, 148
- USBXmitLeft
 - usbblowlib.h, 148
- USE_COMMENTS
 - codecvorbis.h, 48
- USE_TIMER
 - audio.h, 37
- UserInterfaceIdleHook
 - player.h, 101
- USEX
 - vstypes.h, 188
- USEY
 - vstypes.h, 188
- va_arg
 - stdarg.h, 111
- va_end
 - stdarg.h, 111
- va_list
 - stdarg.h, 112
- va_start
 - stdarg.h, 111
- vectors.h, 149
 - DecodeSetupPacket, 149
 - MSCPacketFromPC, 149
 - RealDecodeSetupPacket, 149
 - RealMSCPacketFromPC, 149
 - RealUSBHandler, 149
 - USBHandler, 149
- version
 - Codec, 6
 - CodecServices, 10
 - FsMapper, 17
 - FsPhysical, 24
- voltages
 - vs1000.h, 178
- voltAnaPlayer
 - vs1000.h, 174
- voltAnaSuspend
 - vs1000.h, 174
- voltAnaUSB
 - vs1000.h, 174
- voltAnaUser
 - vs1000.h, 174
- voltCorePlayer
 - vs1000.h, 174
- voltCoreSuspend
 - vs1000.h, 174
- voltCoreUSB
 - vs1000.h, 174
- voltCoreUser
 - vs1000.h, 174
- voltEnd
 - vs1000.h, 174
- voltIdx
 - vs1000.h, 174
- voltIoPlayer
 - vs1000.h, 174
- voltIoSuspend
 - vs1000.h, 174
- voltIoUSB
 - vs1000.h, 174
- voltIoUser
 - vs1000.h, 174
- volume
 - Player, 26
- volumeOffset
 - Player, 26
- volumeReg
 - audio.h, 40
- vs1000.h, 150
 - AUDIO_START, 156
 - BootFromX, 174
 - BusyWait10, 174
 - DAC_LEFT, 156
 - DAC_RIGHT, 156
 - DAC_VOL, 156
 - DCT_START, 156
 - DEBUG_STACK, 156
 - DefUnsupportedFile, 174
 - Disable, 175
 - Enable, 175
 - FCH_DIV_INCLK, 156
 - FCH_DIV_INCLK_B, 157
 - FCH_FORCE_PLL, 157
 - FCH_FORCE_PLL_B, 157
 - FCH_MUL0, 157
 - FCH_MUL0_B, 157
 - FCH_MUL1, 157

FCH_MUL1_B, 157
FCH_MUL2, 157
FCH_MUL2_B, 157
FCH_MUL3, 157
FCH_MUL3_B, 157
FCH_PLL_LOCKED, 157
FCH_PLL_LOCKED_B, 158
FCH_PLL_SET_LOCK, 158
FCH_PLL_SET_LOCK_B, 158
FCH_VCO_OUT_ENA, 158
FCH_VCO_OUT_ENA_B, 158
FREQCTLH, 158
FREQCTLL, 158
FsMapRamCreate, 175
g_dcthi, 178
g_dctlo, 178
g_others, 178
g_yprev0, 178
g_yprev1, 178
getch, 175
GPIO0_ALE, 158
GPIO0_BIT_CONF, 158
GPIO0_BIT_ENG0, 158
GPIO0_BIT_ENG1, 158
GPIO0_CLE, 158
GPIO0_CLEAR_MASK, 159
GPIO0_CS1, 159
GPIO0_DDR, 159
GPIO0_IDATA, 159
GPIO0_INT_FALL, 159
GPIO0_INT_PEND, 159
GPIO0_INT_RISE, 159
GPIO0_MODE, 159
GPIO0_ODATA, 159
GPIO0_RD, 159
GPIO0_READY, 159
GPIO0_SET_MASK, 159
GPIO0_WR, 160
GPIO1_BIT_CONF, 160
GPIO1_BIT_ENG0, 160
GPIO1_BIT_ENG1, 160
GPIO1_CLEAR_MASK, 160
GPIO1_DDR, 160
GPIO1_IDATA, 160
GPIO1_INT_FALL, 160
GPIO1_INT_PEND, 160
GPIO1_INT_RISE, 160
GPIO1_MODE, 160
GPIO1_ODATA, 160
GPIO1_SET_MASK, 161
IdleHook, 175
InitFileSystem, 175
INT_EN_DAC, 161
INT_EN_GPIO0, 161
INT_EN_GPIO1, 161
INT_EN_NFLSH, 161
INT_EN_NONE, 161
INT_EN_REGU, 161
INT_EN_RX, 161
INT_EN_SPI, 161
INT_EN_TIM0, 161
INT_EN_TIM1, 161
INT_EN_TX, 161
INT_EN_USB, 162
INT_ENABLE, 162
INT_ENABLEH, 162
INT_ENABLEL, 162
INT_ENCOUNT, 162
INT_GLOB_DIS, 162
INT_GLOB_EN, 162
INT_ORIGIN, 162
INT_VECTOR, 162
INTF_DAC, 162
INTF_GPIO0, 162
INTF_GPIO1, 162
INTF_NFLSH, 163
INTF_REGU, 163
INTF_RX, 163
INTF_SPI, 163
INTF_TIM0, 163
INTF_TIM1, 163
INTF_TX, 163
INTF_USB, 163
INTV_DAC, 163
INTV_GPIO0, 163
INTV_GPIO1, 163
INTV_NFLSH, 163
INTV_REGU, 164
INTV_RX, 164
INTV_SPI, 164
INTV_TIM0, 164
INTV_TIM1, 164
INTV_TX, 164
INTV_USB, 164
IRAM_SIZE, 164
IRAM_START, 164
IROM_SIZE, 164
IROM_START, 164
LoadCheck, 175
MapperReadDiskSector, 175

MemTests, 176
 NFLASH_CF_INT_ENABLE,
 164
 NFLASH_CF_LCD_CE_MODE,
 165
 NFLASH_CF_NF_RESET, 165
 NFLASH_CF_WAITSTATES,
 165
 NFLASH_CP_LPH, 165
 NFLASH_CTRL, 165
 NFLASH_DATA, 165
 NFLASH_DB_POINTER, 165
 NFLASH_DF_ECC_CALC, 165
 NFLASH_DF_ECC_RESET, 165
 NFLASH_DF_ENA_DBUF, 165
 NFLASH_DF_POINTER, 165
 NFLASH_DF_READ, 165
 NFLASH_DSPIF, 166
 NFLASH_ECC_CNT, 166
 NFLASH_LPL, 166
 NFLASH_NB_BYTECNT, 166
 NFLASH_NF_BYTECNT, 166
 NFLASH_NF_POINTER, 166
 NFLASH_NF_READ, 166
 NFLASH_NF_START, 166
 NFLASH_NF_USE_DBUF, 166
 NFLASH_NFIF, 166
 NullHook, 176
 OpenFile, 176
 OTHERS_START, 166
 PERIP, 166
 PERIP_IN_X, 167
 PlayCurrentFile, 176
 PowerOff, 176
 PowerSetVoltages, 176
 putch, 176
 ReadDiskSector, 176
 ReadFile, 177
 RealLoadCheck, 177
 RealPlayCurrentFile, 177
 RealPowerOff, 177
 Restart, 177
 SCI_DEBUG, 167
 SCI_STATUS, 167
 SCI_SYSTEM, 167
 SCISTF_ANA_PDOWN, 167
 SCISTF_ANADRV_PDOWN,
 167
 SCISTF_REGU_CLOCK, 167
 SCISTF_REGU_POWERBUT,
 167
 SCISTF_REGU_POWERLOW,
 167
 SCISTF_REGU_SHUTDOWN,
 167
 SCISTF_SLOW_CLKMODE,
 167
 SCISTF_USB_DDR, 167
 SCISTF_USB_DIFF_ENA, 168
 SCISTF_USB_DN, 168
 SCISTF_USB_DN_OUT, 168
 SCISTF_USB_DP, 168
 SCISTF_USB_DP_OUT, 168
 SCISTF_USB_PULLUP_ENA,
 168
 SCISTF_VCM_DISABLE, 168
 SCISTF_VCM_OVERLOAD,
 168
 SCISYSF_AVDD, 168
 SCISYSF_CLKDIV, 168
 SCISYSF_CVDD, 168
 SCISYSF_IOVDD, 168
 Seek, 177
 SetHookFunction, 177
 SinTest, 177
 Sleep, 178
 SPI0_CLKCONFIG, 169
 SPI0_CONFIG, 169
 SPI0_DATA, 169
 SPI0_FSYNC, 169
 SPI0_STATUS, 169
 SPI_CC_CLKDIV, 169
 SPI_CF_DLEN, 169
 SPI_CF_DLEN16, 169
 SPI_CF_DLEN8, 169
 SPI_CF_FALLXCS, 169
 SPI_CF_FSIDLE0, 169
 SPI_CF_FSIDLE1, 169
 SPI_CF_INTXCS, 170
 SPI_CF_MASTER, 170
 SPI_CF_RISEXCS, 170
 SPI_CF_SLAVE, 170
 SPI_ST_BREAK, 170
 SPI_ST_RXFULL, 170
 SPI_ST_RXORUN, 170
 SPI_ST_TXFULL, 170
 SPI_ST_TXRUNNING, 170
 SPI_ST_TXURUN, 170
 SpiBoot, 178

- SpiDelay, 178
- SpiLoad, 178
- SpiSendReceive, 178
- STACK_SIZE, 170
- STACK_START, 170
- Tell, 178
- TIMER_CONFIG, 171
- TIMER_ENABLE, 171
- TIMER_T0CNTH, 171
- TIMER_T0CNTL, 171
- TIMER_T0H, 171
- TIMER_T0L, 171
- TIMER_T1CNTH, 171
- TIMER_T1CNTL, 171
- TIMER_T1H, 171
- TIMER_T1L, 171
- UART_DATA, 171
- UART_DATAH, 171
- UART_DIV, 172
- UART_ST_RXFULL, 172
- UART_ST_RXORUN, 172
- UART_ST_TXFULL, 172
- UART_ST_TXRUNNING, 172
- UART_STATUS, 172
- UnsupportedFile, 178
- USB_BASE, 172
- USB_RECV_MEM, 172
- USB_SEND_MEM, 172
- voltages, 178
- voltAnaPlayer, 174
- voltAnaSuspend, 174
- voltAnaUSB, 174
- voltAnaUser, 174
- voltCorePlayer, 174
- voltCoreSuspend, 174
- voltCoreUSB, 174
- voltCoreUser, 174
- voltEnd, 174
- voltIdx, 174
- voltIoPlayer, 174
- voltIoSuspend, 174
- voltIoUSB, 174
- voltIoUser, 174
- WDOG_CONFIG, 172
- WDOG_DUMMY, 172
- WDOG_RESET, 172
- WDOG_RESET_VAL, 173
- XRAM_SIZE, 173
- XRAM_START, 173
- XROM_SIZE, 173
- XROM_START, 173
- YPREV0_START, 173
- YPREV1_START, 173
- YRAM_SIZE, 173
- YRAM_START, 173
- YROM_SIZE, 173
- YROM_START, 173
- vs1000d_BitReverse
 - player.h, 101
- vs1000d_Latin1
 - player.h, 102
- vsasm.h, 179
 - MAKEMOD, 179
 - MAKEMOD64, 179
 - MAKEMODB, 179
 - MAKEMODF, 179
- vsNand.h, 179
- vsNand.h
 - FsPhNandCreate, 181
 - FsPhNandDelete, 181
 - FsPhNandErase, 182
 - FsPhNandFreeBus, 182
 - FsPhNandRead, 182
 - FsPhNandReinitialize, 182
 - FsPhNandWrite, 182
 - NAND_OP_COMMIT_-
 - DATA_ADDRESS, 181
 - NAND_OP_PERFORM_-
 - PROGRAM, 181
 - NAND_OP_PREPARE_TO_-
 - PROGRAM, 181
 - NAND_OP_READ_A, 181
 - NAND_OP_READ_C, 181
 - NAND_OP_READ_-
 - SIGNATURE, 181
 - NAND_OP_READ_STATUS,
 - 181
 - NandCountBits, 182
 - NandMingle, 182
 - NandSwapBad, 182
 - NandWaitIdle, 182
- vstypes.h, 183
 - __a0, 184
 - __a1, 184
 - __b0, 185
 - __b1, 185
 - __c0, 185
 - __c1, 185
 - __d0, 185
 - __d1, 185

- __far, 185
 - __i0, 185
 - __i1, 185
 - __i2, 185
 - __i3, 185
 - __mem_x, 185
 - __near, 186
 - __reg_a, 186
 - __reg_b, 186
 - __reg_c, 186
 - __reg_d, 186
 - __y, 186
 - auto, 186
 - DBL2F16, 186
 - DBL2F32, 186
 - f_int16, 188
 - f_int32, 189
 - FDIV16, 186
 - FDIV32, 186
 - FMT_H16, 187
 - FMT_H32, 187
 - FMT_S16, 187
 - FMT_S32, 187
 - FMT_U16, 187
 - FMT_U32, 187
 - FMUL16, 187
 - FMUL32, 187
 - MR_INT, 188
 - MR_NONE, 188
 - MR_SAT, 188
 - register, 188
 - s_int16, 189
 - s_int32, 189
 - SqrtI, 188
 - SqrtI32, 188
 - u_int16, 189
 - u_int32, 189
 - USEX, 188
 - USEY, 188
- waitns
- FsNandPhys, 22
- wchar_t
- stddef.h, 113
- WDOG_CONFIG
- vs1000.h, 172
- WDOG_DUMMY
- vs1000.h, 172
- WDOG_RESET
- vs1000.h, 172
- WDOG_RESET_VAL
- vs1000.h, 173
- WITH_EARSPEAKER
- audio.h, 37
- wLength
- scsicdb10variant1, 27
 - scsicdb10variant2, 28
- wr
- AUDIOPTR, 5
- Write
- FsMapper, 17
 - FsPhysical, 24
- WriteIRam
- dev1000.h, 68
- XmitBuf
- USBVARS, 34
- XmitLength
- USBVARS, 34
- XRAM_SIZE
- vs1000.h, 173
- XRAM_START
- vs1000.h, 173
- XROM_SIZE
- vs1000.h, 173
- XROM_START
- vs1000.h, 173
- YPREV0_START
- vs1000.h, 173
- YPREV1_START
- vs1000.h, 173
- YRAM_SIZE
- vs1000.h, 173
- YRAM_START
- vs1000.h, 173
- YROM_SIZE
- vs1000.h, 173
- YROM_START
- vs1000.h, 173