

ISD ChipCorder 1700 library for AVR uC
Version : 1.0

Generated by Doxygen 1.7.4

Sat Jul 30 2011 12:22:46

Contents

1	Todo List	1
2	Bug List	1
3	File Documentation	1
3.1	isd.h File Reference	1
3.1.1	Detailed Description	4
3.1.2	License	4
3.1.3	Target	5
3.1.4	More informations	5
3.1.5	Function Documentation	6

1 Todo List

File isd.h isdRecord() : should have an option to use or not the LED indicator (like isdPlay())
isdWriteAPC1()
isd_IsFull()
isd_IsEOM()

2 Bug List

Global **isd_EraseX(uint8_t nb)** Not reliable

Global **isd_ForceGlobalErase(void)** May never return, experimental

3 File Documentation

3.1 isd.h File Reference

SPI functions for the ISD ChipCorder 1700 series from Nuvoton.

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

```
#include "spi/spi.h"
```

Defines

Define AnalogPathControl register (1 byte), default settings are in bold

- #define `isd_APC_DEFAULT` 0x0440
Default value.
- #define `isd_APC_VOL0` 0
*Volume, **000** is maximum, 111 is minimum, -4dBpr step.*
- #define `isd_APC_VOL1` 1
Default value.
- #define `isd_APC_VOL2` 2
Default value.
- #define `isd_APC_MONITOR_INPUT` 3
*Monitor output while recording, **0** disable, 1 enable.*
- #define `isd_APC_MIX_INPUT` 4
In conjunction with `isd_SPI_FT` Default : 0.
- #define `isd_APC_SE_EDITING` 5
*Sound effect in standalone mode, **0** enable, 1 disable.*
- #define `isd_APC_SPI_FT` 6
Enable/disable FT function while in SPI mode 0 on, 1 off.
- #define `isd_APC_AUD_AUX` 7
*Select AUD or AUX, **0** AUD, 1 AUX.*
- #define `isd_APC_PWM_SPK` 8
*PWM speaker output, **0** on, 1 off.*
- #define `isd_APC_PU_ANALOG_OUTPUT` 9
*PowerUp analog output, **0** off, 1 on.*
- #define `isd_APC_VALERT` 10
*vAlert, **0** on, 1 off*
- #define `isd_APC_EOM_ENABLE` 11
*SetPlay will stop at EOM position rather than End Address, **0** off, 1 on.*

Functions

- void `isd_CheckMemory` (void)
CheckCircular Memory command, place record and play pointers.
- void `isd_ClearInterrupt` (void)
Clear the interrupt flag.
- uint8_t `isd_CurrentOperation` (void)
Return the current operation.
- void `isd_Erase` (void)

- Erase the message at current record pointer.*

 - void [isd_EraseX](#) (uint8_t nb)
Erase message nb.
 - void [isd_ForceGlobalErase](#) (void)
Force a global erase.
 - void [isd_Forward](#) (void)
Advances the PLAY_PTR to next message.
 - void [isd_GlobalErase](#) (void)
Global erase.
 - uint8_t [isd_Int](#) (void)
Return the state of the interrupt.
 - uint8_t [isd_IsCMDError](#) (void)
Check if previous operation returned an error.
 - uint8_t [isd_IsErase](#) (void)
Check if current operation is Erase.
 - uint8_t [isd_IsPlay](#) (void)
Check if current operation is Play.
 - uint8_t [isd_IsPowerUp](#) (void)
Check if in PowerUp mode.
 - uint8_t [isd_IsReady](#) (void)
Check if in ready state (idle)
 - uint8_t [isd_IsRecord](#) (void)
Check if current operation is Record.
 - void [isd_LoadNvCfg](#) (void)
Load the configuration from the non volatile RAM.
 - uint8_t [isd_MaxPowerUp](#) (uint8_t retry)
Retry to send the command PowerUp retry times.
 - uint8_t [isd_NbMessages](#) (void)
 - void [isd_Play](#) (uint8_t led)
Play from current location until EOM or STOP command.
 - void [isd_PlayX](#) (uint8_t nb)
Play message nb.
 - void [isd_PowerDown](#) (void)
Power Down command.
 - void [isd_PowerUp](#) (void)
Power Up command.
 - uint16_t [isd_ReadAPC](#) (void)
Return the Analog Path Configuration register.
 - uint8_t [isd_ReadDeviceID](#) (void)
Return the ID for the ISD.

- uint16_t [isd_ReadPlayPtr](#) (void)
Return the pointer of the current message.
- uint16_t [isd_ReadRecordPtr](#) (void)
Return the pointer of the current recording position.
- uint16_t [isd_ReadSR0](#) (void)
Return the status of the SRO register.
- uint8_t [isd_ReadSR1](#) (void)
Return the status of the SR1 register.
- uint8_t [isd_ReadVolume](#) (void)
Read the volume control.
- void [isd_Record](#) (void)
Record from REC_PTR until stop or EOM.
- void [isd_Reset](#) (void)
Reset command.
- void [isd_SPIInit](#) (void)
Initialize SPI communication.
- void [isd_Stop](#) (void)
Stop command.
- void [isd_WaitRDY](#) (void)
Wait until ready state (idle)
- void [isd_WriteAPC2](#) (uint16_t apc)
Load the Analog Path Configuration register with volume settings.
- void [isd_WriteNvCfg](#) (void)
Write the configuration from APC to the non volatile RAM.
- void [isd_WriteVolume](#) (uint8_t volume)
Set the volume control (7 min, 0 max) -4dB steps.

3.1.1 Detailed Description

SPI functions for the ISD ChipCorder 1700 series from Nuvoton.

3.1.2 License

Copyright (c) 2010, 2011 Patrice Nadeau All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. Neither the name of Patrice Nadeau nor the names of its

contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL Patrice Nadeau BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

3.1.3 Target

- ATmega48
- ATmega88
- ATmega168 (Tested)
- ATmega328

3.1.4 More informations

Datasheet : <http://www.nuvoton.com/hq/enu/ProductAndSales/ProductLines/ConsumerElect>

Author

Patrice Nadeau
email : patricen@telwarwick.net
www : <http://nadeaup.homeip.net:8080/>

Example

```
isd_SPIInit();  
isd_PowerUp();  
check isd_RDY();
```

Todo

isdRecord() : should have an option to use or not the LED indicator (like isdPlay())
isdWriteAPC1()
isd_IsFull()
isd_IsEOM()

3.1.5 Function Documentation

3.1.5.1 uint8_t isd_CurrentOperation (void)

Return the current operation.

Returns

First 4 bits of SR1

3.1.5.2 void isd_Erase (void)

Erase the message at current record pointer.

Returns

The content of SR0

Warning

No validation is done to verify that the ISD chip has successfully responded to commands

3.1.5.3 void isd_EraseX (uint8_t nb)

Erase message *nb*.

Returns

The content of SR0 (from isd_Erase)

Bug

Not reliable

3.1.5.4 void isd_ForceGlobalErase (void)

Force a global erase.

Bug

May never return, experimental

3.1.5.5 void isd_GlobalErase (void)

Global erase.

Precondition

isd_IsReady == true

3.1.5.6 uint8_t isd_Int (void)

Return the state of the interrupt.

Return values

<i>true</i> <i>false</i>	Interrupt active
----------------------------	------------------

3.1.5.7 uint8_t isd_IsCMDError (void)

Check if previous operation returned an error.

Return values

<i>true</i> <i>false</i>	Previous operation return an error
----------------------------	------------------------------------

3.1.5.8 uint8_t isd_IsErase (void)

Check if current operation is Erase.

Return values

<i>true</i> <i>false</i>	Current operation is Erase
----------------------------	----------------------------

3.1.5.9 uint8_t isd_IsPlay (void)

Check if current operation is Play.

Return values

<i>true</i> <i>false</i>	Current operation is Play
----------------------------	---------------------------

3.1.5.10 uint8_t isd_IsPowerUp (void)

Check if in PowerUp mode.

Return values

<i>true</i> <i>false</i>	ISD is powered up
----------------------------	-------------------

3.1.5.11 uint8_t isd_IsReady (void)

Check if in ready state (idle)

Return values

<i>true</i> <i>false</i>	Ready state
----------------------------	-------------

3.1.5.12 `uint8_t isd_IsRecord (void)`

Check if current operation is Record.

Return values

<i>true</i> <i>false</i>	Current operation is Record
----------------------------	-----------------------------

3.1.5.13 `uint8_t isd_MaxPowerUp (uint8_t retry)`

Retry to send the command PowerUp *retry* times.

Parameters

<i>in</i>	<i>retry</i>	Number of times to retry the PowerUp command
-----------	--------------	--

Return values

<i>0</i>	Successful
----------	------------

3.1.5.14 `uint8_t isd_NbMessages (void)`

Return the number of messages recorded

Return values

<i>0 - 255</i>	Number of message(s)
----------------	----------------------

3.1.5.15 `void isd.Play (uint8_t led)`

Play from current location until EOM or STOP command.

Parameters

<i>in</i>	<i>led</i>	1 turn on the LED while active
-----------	------------	--------------------------------

3.1.5.16 `void isd.PlayX (uint8_t nb)`

Play message *nb*.

Parameters

<i>in</i>	<i>nb</i>	Message number to play range : 1-255
-----------	-----------	---

3.1.5.17 void isd_PowerDown (void)

Power Down command.

Postcondition

Register affected : SR0: PU, SR1: RDY

3.1.5.18 uint16_t isd_ReadAPC (void)

Return the Analog Path Configuration register.

Returns

APC register (0000D11-D0)

3.1.5.19 uint8_t isd_ReadDeviceID (void)

Return the ID for the ISD.

Returns

ID

Return values

0	Not valid
---	-----------

3.1.5.20 uint16_t isd_ReadPlayPtr (void)

Return the pointer of the current message.

Returns

PLAY_PTR

3.1.5.21 uint16_t isd_ReadRecordPtr (void)

Return the pointer of the current recording position.

Returns

REC_PTR

3.1.5.22 uint16_t isd_ReadSR0 (void)

Return the status of the SRO register.

Returns

The content of the SRO register

3.1.5.23 `uint8_t isd_ReadSR1 (void)`

Return the status of the SR1 register.

Returns

The content of the SR1 register

3.1.5.24 `uint8_t isd_ReadVolume (void)`

Read the volume control.

Returns

Volume level

Return values

<code>0 - 7</code> 7 min, 0 max, in -4dB steps
--

3.1.5.25 `void isd.Record (void)`

Record from REC_PTR until stop or EOM.

Precondition

`isd_IsReady == true`

3.1.5.26 `void isd.SPInit (void)`

Initialize SPI communication.

Note

SPI : LSB, mode 3, clock/128, interrupt disabled

3.1.5.27 `void isd.Stop (void)`

Stop command.

Returns

`uint16_t` SR0

3.1.5.28 void isd_WriteAPC2 (uint16_t *apc*)

Load the Analog Path Configuration register with volume settings.

Parameters

<code>in</code>	<code>apc</code>	Analog Path Configuration register
-----------------	------------------	------------------------------------

3.1.5.29 void isd_WriteVolume (uint8_t *volume*)

Set the volume control (7 min, 0 max) -4dB steps.

Parameters

<code>in</code>	<code>volume</code>	level 0 - 7 : 7 min, 0 max in -4dB steps
-----------------	---------------------	---

Index

isd.h, 1

- isd_CurrentOperation, 5
- isd_Erase, 5
- isd_EraseX, 6
- isd_ForceGlobalErase, 6
- isd_GlobalErase, 6
- isd_Int, 6
- isd_IsCMDError, 6
- isd_IsErase, 7
- isd_IsPlay, 7
- isd_IsPowerUp, 7
- isd_IsReady, 7
- isd_IsRecord, 7
- isd_MaxPowerUp, 8
- isd_NbMessages, 8
- isd_Play, 8
- isd_PlayX, 8
- isd_PowerDown, 8
- isd_ReadAPC, 9
- isd_ReadDeviceID, 9
- isd_ReadPlayPtr, 9
- isd_ReadRecordPtr, 9
- isd_ReadSR0, 9
- isd_ReadSR1, 9
- isd_ReadVolume, 10
- isd_Record, 10
- isd_SPIInit, 10
- isd_Stop, 10
- isd_WriteAPC2, 10
- isd_WriteVolume, 11

isd_CurrentOperation

- isd.h, 5

isd_Erase

- isd.h, 5

isd_EraseX

- isd.h, 6

isd_ForceGlobalErase

- isd.h, 6

isd_GlobalErase

- isd.h, 6

isd_Int

- isd.h, 6

isd_IsCMDError

- isd.h, 6

isd_IsErase

- isd.h, 7

isd_IsPlay

- isd.h, 7

isd_IsPowerUp

- isd.h, 7

isd_IsReady

- isd.h, 7

isd_IsRecord

- isd.h, 7

isd_MaxPowerUp

- isd.h, 8

isd_NbMessages

- isd.h, 8

isd_Play

- isd.h, 8

isd_PlayX

- isd.h, 8

isd_PowerDown

- isd.h, 8

isd_ReadAPC

- isd.h, 9

isd_ReadDeviceID

- isd.h, 9

isd_ReadPlayPtr

- isd.h, 9

isd_ReadRecordPtr

- isd.h, 9

isd_ReadSR0

- isd.h, 9

isd_ReadSR1

- isd.h, 9

isd_ReadVolume

- isd.h, 10

isd_Record

- isd.h, 10

isd_SPIInit

- isd.h, 10

isd_Stop

- isd.h, 10

isd_WriteAPC2
 isd.h, [10](#)
isd_WriteVolume
 isd.h, [11](#)