

Minikombajn Pomiarowy

1.0

Arkadiusz Hudzikowski

Wygenerowano przez Doxygen 1.7.3

Fri Dec 30 2011 17:49:29

Rozdział 1

Indeks plików

1.1 Lista plików

Tutaj znajduje się lista wszystkich udokumentowanych plików z ich krótkimi opisami:

ADC.c (Plik obsługi przetwornika ADC)	3
ADC.h (Plik nagłówek obsługi przetwornika ADC)	7
Analizator.c (Plik podprogramu analizatora)	10
Analizator.h (Plik nagłówek podprogramu analizatora)	14
AnalizatorStLog.c (Plik podprogramu analizatora stanów logicznych)	16
AnalizatorStLog.h (Plik nagłówek podprogramu analizatora stanów logicznych)	19
avr_compiler.h (This file implements some macros that makes the IAR C-compiler and avr-gcc work with the same code base for the AVR architecture)	20
clksys_driver.c (XMEGA Clock System driver source file)	21
clksys_driver.h (XMEGA Clock System driver header file)	23
DAC.c (Plik obsługi przetwornika DAC)	26
DAC.h (Plik nagłówek obsługi przetwornika DAC)	28
font5x7.h (Plik zawierający tablice czcionek)	29
Generator.c (Plik podprogramu generatora)	30
Generator.h (Plik nagłówek podprogramu generatora)	33
Grafika.c (Plik funkcji graficznych)	34
Grafika.h (Plik nagłówek funkcji graficznych)	41
Keyboard.c (Plik obsługi klawiatury)	48
Keyboard.h (Plik nagłówek obsługi klawiatury)	50
lcd132x64.c (Plik obsługi wyświetlacza)	52
lcd132x64.h (Plik nagłówek obsługi wyświetlacza)	57
main.c (Plik główny programu)	61
Multimetr.c (Plik podprogramu multimetru)	65
Multimetr.h (Plik nagłówek podprogramu multimetru)	66
Oscyloskop.c (Plik podprogramu oscyloskopu)	67
Oscyloskop.h (Plik nagłówek podprogramu oscyloskopu)	69

TransmisjaPC.c (Plik obsługi transmisji UART)	70
TransmisjaPC.h (Plik nagłowski obsługi transmisji UART)	76
usart_driver.c (XMEGA USART driver source file)	77
usart_driver.h (XMEGA USART driver header file)	78
Ustawienia.c (Plik funkcji ustawień)	85
Ustawienia.h (Plik nagłowski funkcji ustawień)	88
Wobuloskop.c (Plik podprogramu wobuloskopu)	89
Wobuloskop.h (Plik nagłowski podprogramu wobuloskopu)	91

Rozdział 2

Dokumentacja plików

2.1 Dokumentacja pliku ADC.c

Plik obsługi przetwornika ADC.

```
#include <avr/io.h>
#include <util/delay.h>
#include <avr/pgmspace.h>
#include <stddef.h>
#include <avr/eeprom.h>
#include "ADC.h"
```

Funkcje

- uint8_t [ReadCalibrationByte](#) (uint8_t index)
Funkcja odczytująca rejestry kalibracyjne ADC.
- void [ADCInit](#) (void)
Funkcja inicjująca ADC.
- void [ADCOff](#) (void)
Funkcja wyłączająca ADC.
- int16_t [ADCGetCh0](#) (void)
Funkcja odczytująca kanał 0 ADC.
- int16_t [ADCGetCh1](#) (void)
Funkcja odczytująca kanał 1 ADC.
- int16_t [ADCGetCh2](#) (void)

Funkcja odczytująca kanał 2 ADC.

- void [ADCSetPeroid](#) (uint8_t per)
Funkcja ustawiająca podstawę czasu.
- void [ADCSetGain](#) (uint8_t g1, uint8_t g2)
Funkcja ustawiająca wzmacnienie.
- void [ADCRunOffsetCal](#) (void)
Funkcja kalibrująca offset sygnału wejściowego.
- void [ADCOffsetCorrect](#) (int16_t *wsk, uint8_t channels, uint8_t vdiv1, uint8_t vdiv2)
Funkcja korygująca sygnał na podstawie danych kalibracyjnych.

Zmienne

- int16_t [kan1_in](#) [512]
- int16_t [kan2_in](#) [512]
Bufor przechowujący próbki danych kanału 2.
- EEMEM int8_t [e_offset_cal](#) [14]
- prog_uint16_t [Time_tab](#) [15]
Tablica dzielników dla wyboru podstawy czasu.

2.1.1 Opis szczegółowy

Plik obsługi przetwornika ADC.

Autor

Arkadiusz Hudzikowski

Wersja

1.0

Data

22.11.2011

2.1.2 Dokumentacja funkcji

2.1.2.1 int16_t ADCGetCh0 (void)

Funkcja odczytująca kanał 0 ADC.

Zwraca

int16_t : odczytana wartosc

2.1.2.2 int16_t ADCGetCh1 (void)

Funkcja odczytująca kanal 1 ADC.

Zwraca

int16_t : odczytana wartosc

2.1.2.3 int16_t ADCGetCh2 (void)

Funkcja odczytująca kanal 2 ADC.

Zwraca

int16_t : odczytana wartosc

2.1.2.4 void ADCInit (void)

Funkcja inicjująca ADC.

Zwraca

none

2.1.2.5 void ADCOff (void)

Funkcja wyłączająca ADC.

Zwraca

none

2.1.2.6 void ADCOffsetCorrect (int16_t * wsk, uint8_t channels, uint8_t vdiv1, uint8_t vdiv2)

Funkcja korygująca sygnał na podstawie danych kalibracyjnych.

Parametry

<i>*wsk</i>	: adres bufora sygnału
<i>channels</i>	: 0 - praca 1 kanalowa, 1 - praca dwukanalowa
<i>vdiv1</i>	: wzmacnienie kanału 1
<i>vdiv2</i>	: wzmacnienie kanału 2

Zwraca

none

2.1.2.7 void ADCRunOffsetCal (void)

Funkcja kalibrująca offset sygnału wejściowego.

Zwraca

none

2.1.2.8 void ADCSetGain (uint8_t g1, uint8_t g2)

Funkcja ustawiająca wzmocnienie.

wartosc wzmocnienia wynosi 2^{g_x} , gdzie g_x to parametr podawany w funkcji

Parametry

<i>g1</i>	: wartosc wzmocnienia dla kanalu 1 (ch0)
<i>g2</i>	: wartosc wzmocnienia dla kanalu 2 (ch1)

Zwraca

none

2.1.2.9 void ADCSetPeroid (uint8_t per)

Funkcja ustawiająca podstawe czasu.

Parametry

<i>per</i>	: numer wybranej podstawy czasu
------------	---------------------------------

Zwraca

none

2.1.2.10 uint8_t ReadCalibrationByte (uint8_t index)

Funkcja odczytująca rejestry kalibracyjne ADC.

Parametry

<i>index</i>	: adres rejestru
--------------	------------------

Zwraca

uint8_t : wartosc rejestru

2.1.3 Dokumentacja zmiennych**2.1.3.1 EEMEM int8_t e_offset_cal[14]**

Tablica wartosci kalibracji offsetu ADC

2.1.3.2 prog_uint16_t Time_tab[15]

Wartość początkowa:

```
{
    4,
    5,
    4,
    5,
    10,
    25,
    50,
    100,
    250,
    500,
    1000,
    2500,
    5000,
    10000,
    25000}
```

Tablica dzielnikow dla wyboru podstawy czasu.

2.2 Dokumentacja pliku ADC.h

Plik naglowkowy obsługi przetwornika ADC.

Funkcje

- void [ADCInit](#) (void)
Funkcja inicjujaca ADC.
- void [ADCOff](#) (void)
Funkcja wylaczajaca ADC.
- int16_t [ADCGetCh0](#) (void)
Funkcja odczytujaca kanal 0 ADC.
- int16_t [ADCGetCh1](#) (void)

Funkcja odczytująca kanał 1 ADC.

- int16_t [ADCGetCh2](#) (void)

Funkcja odczytująca kanał 2 ADC.

- void [ADCSetPeroid](#) (uint8_t per)

Funkcja ustawiająca podstawę czasu.

- void [ADCSetGain](#) (uint8_t g1, uint8_t g2)

Funkcja ustawiająca wzmocnienie.

- void [ADCRunOffsetCal](#) (void)

Funkcja kalibrująca offset sygnału wejściowego.

- void [ADCOffsetCorrect](#) (int16_t *wsk, uint8_t channels, uint8_t vdiv1, uint8_t vdiv2)

Funkcja korygująca sygnał na podstawie danych kalibracyjnych.

2.2.1 Opis szczegółowy

Plik nagłówekowy obsługi przetwornika ADC.

Autor

Arkadiusz Hudzikowski

Wersja

1.0

Data

22.11.2011

2.2.2 Dokumentacja funkcji

2.2.2.1 int16_t [ADCGetCh0](#) (void)

Funkcja odczytująca kanał 0 ADC.

Zwraca

int16_t : odczytana wartość

2.2.2.2 int16_t ADCGetCh1 (void)

Funkcja odczytująca kanał 1 ADC.

Zwraca

int16_t : odczytana wartosc

2.2.2.3 int16_t ADCGetCh2 (void)

Funkcja odczytująca kanał 2 ADC.

Zwraca

int16_t : odczytana wartosc

2.2.2.4 void ADCInit (void)

Funkcja inicjująca ADC.

Zwraca

none

2.2.2.5 void ADCOff (void)

Funkcja wyłączająca ADC.

Zwraca

none

2.2.2.6 void ADCOffsetCorrect (int16_t * wsk, uint8_t channels, uint8_t vdiv1, uint8_t vdiv2)

Funkcja korygująca sygnał na podstawie danych kalibracyjnych.

Parametry

<i>*wsk</i>	: adres bufora sygnału
<i>channels</i>	: 0 - praca 1 kanałowa, 1 - praca dwukanałowa
<i>vdiv1</i>	: wzmacnienie kanału 1
<i>vdiv2</i>	: wzmacnienie kanału 2

Zwraca

none

2.2.2.7 void ADCRunOffsetCal (void)

Funkcja kalibrująca offset sygnału wejściowego.

Zwraca

none

2.2.2.8 void ADCSetGain (uint8_t g1, uint8_t g2)

Funkcja ustawiająca wzmocnienie.

wartosc wzmocnienia wynosi 2^{gx} , gdzie gx to parametr podawany w funkcji

Parametry

<i>g1</i>	: wartosc wzmocnienia dla kanalu 1 (ch0)
<i>g2</i>	: wartosc wzmocnienia dla kanalu 2 (ch1)

Zwraca

none

2.2.2.9 void ADCSetPeroid (uint8_t per)

Funkcja ustawiająca podstawe czasu.

Parametry

<i>per</i>	: numer wybranej podstawy czasu
------------	---------------------------------

Zwraca

none

2.3 Dokumentacja pliku Analizator.c

Plik podprogramu analizatora.

```
#include <avr/io.h>
#include <avr/pgmspace.h>
#include <avr/interrupt.h>
#include "ADC.h"
#include "Keyboard.h"
#include "lcd132x64.h"
```

```
#include "Grafika.h"
```

Definicje

- #define `USE_ANALIZATOR_MODULE`
- #define `Np` 1024

Funkcje

- uint8_t `log2_u32` (uint32_t n)
Funkcja licząca logarytm o podstawie 2 dla liczb 32-bitowych.
- uint8_t `log2_u16` (uint16_t n)
Funkcja licząca logarytm o podstawie 2 dla liczb 16-bitowych.
- void `FFT2N` (int16_t *xwsk, int16_t *tmp_w)
Funkcja licząca 512-punktowe Real-FFT.
- void `FFT2N128` (int16_t *xwsk)
Funkcja licząca 128-punktowe Real-FFT.
- void `Analizator` (void)
Funkcja główna podprogramu analizatora.

Zmienne

- int16_t `kan1_in` [512]
- int16_t `kan2_in` [512]
- uint8_t `kan1_lcd` [128]
Bufor przechowujący próbki danych kanału 1 do wyświetlenia na lcd.
- prog_uint8_t `Gain_tab` [10]
- prog_uint16_t `Time_tab` [14]
Tablica dzielników dla wyboru podstawy czasu.
- prog_int16_t `sin_tab` [640]
Tablica funkcji sin [0, 2.5PI].
- uint16_t `Xpos` = 0
- uint8_t `Ypos` = 0
- uint8_t `Zoom` = 1
- uint16_t `Cursor` = 70

2.3.1 Opis szczegółowy

Plik podprogramu analizatora.

Autor

Arkadiusz Hudzikowski

Wersja

1.0

Data

22.11.2011

2.3.2 Dokumentacja definicji

2.3.2.1 #define Np 1024

liczba próbek wejściowych FFT

2.3.2.2 #define USE_ANALIZATOR_MODULE

Użyj tego podprogramu

2.3.3 Dokumentacja funkcji

2.3.3.1 void Analizator (void)

Funkcja główna podprogramu analizatora.

Zwraca

none

2.3.3.2 void FFT2N(int16_t * xwsk, int16_t * tmp_w)

Funkcja licząca 512-punktowe Real-FFT.

Rzeczywista szybka transformata Fouriera. Sygnał zostaje rozdzielony na próbki parzyste, które traktowane są jako rzeczywiste i nieparzyste, które zostają zapisane do części urojonej. Następnie zostaje obliczona zespolona transformata Fouriera oraz zostają rozdzielone widma wejściowych próbek parzystych i nieparzystych. Na końcu następuje synteza tych widm tworząc jedno 512 punktowe.

Parametry

*xwsk	inout: wskaźnik do bufora 1024 wejściowych próbek 16-bitowych, 512 próbek wyjściowych
*tmp_w	Adres bufora pomocniczego 64 próbek 16-bitowych

Zwraca

none

2.3.3.3 void FFT2N128 (int16_t * xwsk)

Funkcja licząca 128-punktowe Real-FFT.

Rzeczywista szybka transformata Fouriera. Sygnał zostaje rozdzielony na próbki parzyste, które traktowane są jako rzeczywiste i nieparzyste, które zostają zapisane do części urojonej. Następnie zostaje obliczona zespolona transformata Fouriera oraz zostają rozdzielone widma wejściowych próbek parzystych i nieparzystych. Na końcu następuje synteza tych widm tworząc jedno 128 punktowe.

Parametry

*xwsk	inout: wskaźnik do bufora 256 wejściowych próbek 16-bitowych, 128 próbek wyjściowych bufor wykorzystuje dodatkowo 128 próbek pomocniczych, całkowita długość bufora wynosi 384 próbki 16-bitowe
-------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Zwraca

none

2.3.3.4 uint8_t log2_u16 (uint16_t n)

Funkcja licząca logarytm o podstawie 2 dla liczb 16-bitowych.

Wartość wyjściowa wyrażona jest wzorem: $4 * \log_2(n)$. Wartości wynikowe pomiędzy wielokrotnościami 4 są podawane z przybliżeniem.

Parametry

n	: zmienna wejściowa
---	---------------------

Zwraca

uint8_t : wartość wyjściowa (logarytm)

2.3.3.5 uint8_t log2_u32 (uint32_t n)

Funkcja licząca logarytm o podstawie 2 dla liczb 32-bitowych.

Wartość wyjściowa wyrażona jest wzorem: $4 * \log_2(n)$. Wartości wynikowe pomiędzy wielokrotnościami 4 są podawane z przybliżeniem.

Parametry

n	: zmienna wejściowa
---	---------------------

Zwraca

uint8_t : wartosc wyjsciowa (logarytm)

2.3.4 Dokumentacja zmiennych**2.3.4.1 uint16_t Cursor = 70**

kursor czestotliwosci

2.3.4.2 int16_t kan1_in[512]

Bufor kanalu 1

2.3.4.3 uint8_t kan1_lcd ()

Bufor przechowujacy probki danych kanalu 1 do wyswietlenia na lcd.

Bufor oscylogramu 1

2.3.4.4 int16_t kan2_in[512]

Bufor kanalu 2

2.3.4.5 uint16_t Xpos = 0

przesuwanie w poziomie

2.3.4.6 uint8_t Ypos = 0

przesuwanie w pionie

2.3.4.7 uint8_t Zoom = 1

zoom sygnalu (do implementacji)

2.4 Dokumentacja pliku Analizator.h

Plik naglowkowy podprogramu analizatora.

Funkcje

- uint8_t [log2_u32](#) (uint32_t n)

Funkcja licząca logarytm o podstawie 2 dla liczb 32-bitowych.

- void [Analizator](#) (void)
Funkcja główna podprogramu analizatora.
- void [FFT2N](#) (int16_t *xwsk, int16_t *ywsk)
Funkcja licząca 512-punktowe Real-FFT.
- void [FFT2N128](#) (int16_t *xwsk)
Funkcja licząca 128-punktowe Real-FFT.

2.4.1 Opis szczegółowy

Plik nagłówkowy podprogramu analizatora.

Autor

Arkadiusz Hudzikowski

Wersja

1.0

Data

22.11.2011

2.4.2 Dokumentacja funkcji

2.4.2.1 void Analizator (void)

Funkcja główna podprogramu analizatora.

Zwraca

none

2.4.2.2 void FFT2N (int16_t * xwsk, int16_t * tmp_w)

Funkcja licząca 512-punktowe Real-FFT.

Rzeczywista szybka transformata Fouriera. Sygnał zostaje rozdzielony na próbki parzyste, które traktowane są jako rzeczywiste i nieparzyste, które zostają zapisane do części urojonej. Następnie zostaje obliczona zespolona transformata Fouriera oraz zostają rozdzielone widma wejściowych próbek parzystych i nieparzystych. Na końcu następuje synteza tych widm tworząc jedno 512 punktowe.

Parametry

<i>*xwsk</i>	inout: wskaźnik do bufora 1024 wejściowych próbek 16-bitowych, 512 próbek wyjściowych
<i>*tmp_w</i>	: adres bufora pomocniczego 64 próbek 16-bitowych

Zwraca

none

2.4.2.3 void FFT2N128 (int16_t * xwsk)

Funkcja licząca 128-punktowe Real-FFT.

Rzeczywista szybka transformata Fouriera. Sygnał zostaje rozdzielony na próbki parzyste, które traktowane są jako rzeczywiste i nieparzyste, które zostają zapisane do części urojonej. Następnie zostaje obliczona zespolona transformata Fouriera oraz zostają rozdzielone widma wejściowych próbek parzystych i nieparzystych. Na końcu następuje synteza tych widm tworząc jedno 128 punktowe.

Parametry

<i>*xwsk</i>	inout: wskaźnik do bufora 256 wejściowych próbek 16-bitowych, 128 próbek wyjściowych bufor wykorzystuje dodatkowo 128 próbek pomocniczych, całkowita długość bufora wynosi 384 próbki 16-bitowe
--------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Zwraca

none

2.4.2.4 uint8_t log2_u32 (uint32_t n)

Funkcja licząca logarytm o podstawie 2 dla liczb 32-bitowych.

Wartość wyjściowa wyrażona jest wzorem: $4 \cdot \log_2(n)$. Wartości wynikowe pomiędzy wielokrotnościami 4 są podawane z przybliżeniem.

Parametry

<i>n</i>	: zmienna wejściowa
----------	---------------------

Zwraca

uint8_t : wartość wyjściowa (logarytm)

2.5 Dokumentacja pliku AnalizatorStLog.c

Plik podprogramu analizatora stanów logicznych.

```
#include <avr/io.h>
```

```
#include <util/delay.h>
#include <avr/pgmspace.h>
#include "lcd132x64.h"
#include "Grafika.h"
#include "clksys_driver.h"
#include "ADC.h"
#include "DAC.h"
#include "Keyboard.h"
#include "Oscyloskop.h"
#include "Generator.h"
#include "Wobuloskop.h"
#include "Analizator.h"
#include "Ustawienia.h"
```

Definicje

- `#define` [USE_ANALIZATOR_ST_LOG_MODULE](#)

Funkcje

- void [LCDStateGraph](#) (uint8_t *wsk, uint8_t cursor)
Funkcja wyswietlajaca przebiegi cyfrowe.
- [ISR](#) (PORTD_INT0_vect)
Przerwanie INT0 od wybranego pinu (pinow) portu D mikrokontrolera.
- void [AnalizatorStLog](#) (void)
Funkcja glowna podprogramu analizatora stanow logicznych.

Zmienne

- uint8_t **kan1_lcd** []
- int16_t **kan1_in** [1024]
- char **Trig_type_tab** [8] = {'X','0','1','/',92}

2.5.1 Opis szczegółowy

Plik podprogramu analizatora stanow logicznych.

Autor

Arkadiusz Hudzikowski

Wersja

1.0

Data

22.11.2011

2.5.2 Dokumentacja definicji**2.5.2.1 #define USE_ANALIZATOR_ST_LOG_MODULE**

Uzyj podprogramu

2.5.3 Dokumentacja funkcji**2.5.3.1 void AnalizatorStLog (void)**

Funkcja glowna podprogramu analizatora stanow logicznych.

Zwraca

none

2.5.3.2 ISR (PORTD_INT0_vect)

Przerwanie INT0 od wybranego pinu (pinow) portu D mikrokontrolera.

Przerwanie deklarowane z atrybutem 'naked' w celu szybszego wykonywania. Sluzy tylko do wybudzenia mikrokontrolera.

Parametry

<i>PORTD_ INT0_vect</i>	: wektor przerwania INT0 portu D
-----------------------------	----------------------------------

Zwraca

none

2.5.3.3 void LCDStateGraph (uint8_t * wsk, uint8_t cursor)

Funkcja wyswietlajaca przebiegi cyfrowe.

Parametry

<i>*wsk</i>	: adres bufora przechowującego próbki do wyświetlenia
<i>cursor</i>	: wartość ustawiająca kursor w wybranej pozycji

Zwraca

none

2.6 Dokumentacja pliku AnalizatorStLog.h

Plik nagłówkowy podprogramu analizatora stanów logicznych.

Funkcje

- void **PORT_DMAInit** (void)
- void **PORT_DMAOff** (void)
- void [AnalizatorStLog](#) (void)

Funkcja główna podprogramu analizatora stanów logicznych.

2.6.1 Opis szczegółowy

Plik nagłówkowy podprogramu analizatora stanów logicznych.

Autor

Arkadiusz Hudzikowski

Wersja

1.0

Data

22.11.2011

2.6.2 Dokumentacja funkcji

2.6.2.1 void AnalizatorStLog (void)

Funkcja główna podprogramu analizatora stanów logicznych.

Zwraca

none

2.7 Dokumentacja pliku avr_compiler.h

This file implements some macros that makes the IAR C-compiler and avr-gcc work with the same code base for the AVR architecture.

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

Definicje

- #define [F_CPU](#) 2000000UL
Define default CPU frequency, if this is not already defined.
- #define [AVR_ENTER_CRITICAL_REGION\(\)](#)
This macro will protect the following code from interrupts.
- #define [AVR_LEAVE_CRITICAL_REGION\(\)](#) SREG = saved_sreg;
This macro must always be used in conjunction with AVR_ENTER_CRITICAL_REGION so the interrupts are enabled again.

2.7.1 Opis szczegółowy

This file implements some macros that makes the IAR C-compiler and avr-gcc work with the same code base for the AVR architecture.

Documentation

For comprehensive code documentation, supported compilers, compiler settings and supported devices see [readme.html](#)

Autor

Atmel Corporation: <http://www.atmel.com>
Support email: avr@atmel.com

Revision:

2772

Date:

2009-09-11 12:40:26 +0200 (fr, 11 sep 2009)

Copyright (c) 2008, Atmel Corporation All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. 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.
3. The name of ATMEL may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY ATMEL "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 EXPRESSLY AND SPECIFICALLY DISCLAIMED. IN NO EVENT SHALL ATMEL 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.

2.7.2 Dokumentacja definicji

2.7.2.1 `#define AVR_ENTER_CRITICAL_REGION()`

Wartość:

```
uint8_t volatile saved_sreg = SREG; \
    cli();
```

This macro will protect the following code from interrupts.

2.8 Dokumentacja pliku `clksys_driver.c`

XMEGA Clock System driver source file.

```
#include "clksys_driver.h"
```

Funkcje

- void [CCPWrite](#) (volatile uint8_t *address, uint8_t value)

CCP write helper function written in assembly.

2.8.1 Opis szczegółowy

XMEGA Clock System driver source file. This file contains the function implementations for the XMEGA Clock System driver.

The driver is not intended for size and/or speed critical code, since most functions are just a few lines of code, and the function call overhead would decrease code performance. The driver is intended for rapid prototyping and documentation purposes for getting started with the XMEGA Clock System.

For size and/or speed critical code, it is recommended to copy the function contents directly into your application instead of making a function call.

Several functions use the following construct: "some_register = ... | (some_parameter ? SOME_BIT_bm : 0) | ..." Although the use of the ternary operator (if ? then : else) is discouraged, in some occasions the operator makes it possible to write pretty clean and neat code. In this driver, the construct is used to set or not set a configuration bit based on a boolean input parameter, such as the "some_parameter" in the example above.

Application note:

AVR1003: Using the XMEGA Clock System

Documentation

For comprehensive code documentation, supported compilers, compiler settings and supported devices see readme.html

Author

Atmel Corporation: <http://www.atmel.com>

Support email: avr@atmel.com

Revision:

2771

Date:

2009-09-11 11:54:26 +0200 (fr, 11 sep 2009)

Copyright (c) 2008, Atmel Corporation All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. 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.
3. The name of ATMEL may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY ATMEL "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 EXPRESSLY AND SPECIFICALLY DISCLAIMED. IN NO EVENT SHALL ATMEL 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.

2.8.2 Dokumentacja funkcji

2.8.2.1 void CCPWrite (volatile uint8_t * address, uint8_t value)

CCP write helper function written in assembly.

This function is written in assembly because of the timecritical operation of writing to the registers.

Parametry

<i>address</i>	A pointer to the address to write to.
<i>value</i>	The value to put in to the register.

2.9 Dokumentacja pliku `clksys_driver.h`

XMEGA Clock System driver header file.

```
#include "avr_compiler.h"
```

Definicje

- `#define CLKSYS_Enable(_oscSel) (OSC.CTRL |= (_oscSel))`
This macro enables the selected oscillator.
- `#define CLKSYS_IsReady(_oscSel) (OSC.STATUS & (_oscSel))`
This macro check if selected oscillator is ready.
- `#define CLKSYS_RTC_ClockSource_Disable() (CLK.RTCCTRL &= ~CLK_RTCEN_bm)`
This macro disables routing of clock signals to the Real-Time Counter (RTC).
- `#define CLKSYS_AutoCalibration_Disable(_clk) ((_clk).CTRL &= ~DFLL_ENABLE_bm)`

This macro disables the automatic calibration of the selected internal oscillator.

Funkcje

- void **CCPWrite** (volatile uint8_t *address, uint8_t value)
CCP write helper function written in assembly.

2.9.1 Opis szczegółowy

XMEGA Clock System driver header file. This file contains the function prototypes and enumerator definitions for various configuration parameters for the XMEGA Clock System driver.

The driver is not intended for size and/or speed critical code, since most functions are just a few lines of code, and the function call overhead would decrease code performance. The driver is intended for rapid prototyping and documentation purposes for getting started with the XMEGA Clock System.

For size and/or speed critical code, it is recommended to copy the function contents directly into your application instead of making a function call.

Application note:

AVR1003: Using the XMEGA Clock System

Documentation

For comprehensive code documentation, supported compilers, compiler settings and supported devices see readme.html

Autor

Atmel Corporation: <http://www.atmel.com>
Support email: avr@atmel.com

Revision:

1665

Date:

2008-06-05 09:21:50 +0200 (to, 05 jun 2008)

Copyright (c) 2008, Atmel Corporation All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. 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.

3. The name of ATMEL may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY ATMEL "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 EXPRESSLY AND SPECIFICALLY DISCLAIMED. IN NO EVENT SHALL ATMEL 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.

2.9.2 Dokumentacja definicji

2.9.2.1 `#define CLKSYS_AutoCalibration_Disable(_clk) ((_clk).CTRL &= ~DFLL_ENABLE_bm)`

This macro disables the automatic calibration of the selected internal oscillator.

Parametry

<code>_clk</code>	Clock source calibration to disable, either DFLLRC2M or DFLLRC32M.
-------------------	--------------------------------------------------------------------

2.9.2.2 `#define CLKSYS_Enable(_oscSel) (OSC.CTRL |= (_oscSel))`

This macro enables the selected oscillator.

Nota

Note that the oscillator cannot be used as a main system clock source without being enabled and stable first. Check the ready flag before using the clock. The function [CLKSYS_IsReady\(_oscSel \)](#) can be used to check this.

Parametry

<code>_oscSel</code>	Bitmask of selected clock. Can be one of the following <code>OSC_RC2MEN_bm</code> , <code>OSC_RC32MEN_bm</code> , <code>OSC_RC32KEN_bm</code> , <code>OSC_XOSCEN_bm</code> , <code>OSC_PPLEN_bm</code> .
----------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

2.9.2.3 #define CLKSYS_IsReady(_oscSel) (OSC.STATUS & (_oscSel))

This macro check if selected oscillator is ready.

This macro will return non-zero if is is running, regardless if it is used as a main clock source or not.

Parametry

<i>_oscSel</i>	Bitmask of selected clock. Can be one of the following OSC_RC2MEN_bm, OSC_RC32MEN_bm, OSC_RC32KEN_bm, OSC_XOSCEN_bm, OSC_PLEN_bm.
----------------	-----------------------------------------------------------------------------------------------------------------------------------

Zwraca

Non-zero if oscillator is ready and running.

2.9.2.4 #define CLKSYS_RTC_ClockSource_Disable() (CLK.RTCCTRL &= ~CLK_RTCEN_bm)

This macro disables routing of clock signals to the Real-Time Counter (RTC).

Disabling the RTC saves power if the RTC is not in use.

2.9.3 Dokumentacja funkcji

2.9.3.1 void CCPWrite (volatile uint8_t * address, uint8_t value)

CCP write helper function written in assembly.

This function is written in assembly because of the timecritical operation of writing to the registers.

Parametry

<i>address</i>	A pointer to the address to write to.
<i>value</i>	The value to put in to the register.

2.10 Dokumentacja pliku DAC.c

Plik obsługi przetwornika DAC.

```
#include <avr/io.h>
```

Funkcje

- void [DACInit](#) (void)

Funkcja inicjujaca DAC.

- void `DACOff` (void)
Funkcja wylaczajaca DAC.
- void `DACWriteCh0` (uint16_t val)
Funkcja wpisujaca wartosc do DAC.

Zmienne

- uint16_t `kan_out` [512]
Bufor przechowujacy probki sygnalu wyjscowego.

2.10.1 Opis szczegółowy

Plik obsługi przetwornika DAC.

Autor

Arkadiusz Hudzikowski

Wersja

1.0

Data

22.11.2011

2.10.2 Dokumentacja funkcji

2.10.2.1 void `DACInit` (void)

Funkcja inicjujaca DAC.

Zwraca

none

2.10.2.2 void `DACOff` (void)

Funkcja wylaczajaca DAC.

Zwraca

none

2.10.2.3 void DACWriteCh0 (uint16_t val)

Funkcja wpisująca wartość do DAC.

Parametry

<i>val</i>	: wartość wyjściowa
------------	---------------------

Zwraca

none

2.11 Dokumentacja pliku DAC.h

Plik nagłówkowy obsługi przetwornika DAC.

Funkcje

- void [DACInit](#) (void)
Funkcja inicjująca DAC.
- void [DACOff](#) (void)
Funkcja wyłączająca DAC.
- void [DACWriteCh0](#) (uint16_t val)
Funkcja wpisująca wartość do DAC.

2.11.1 Opis szczegółowy

Plik nagłówkowy obsługi przetwornika DAC.

Autor

Arkadiusz Hudzikowski

Wersja

1.0

Data

22.11.2011

2.11.2 Dokumentacja funkcji

2.11.2.1 void DACInit (void)

Funkcja inicjująca DAC.

Zwraca

none

2.11.2.2 void DACOff (void)

Funkcja wylaczajaca DAC.

Zwraca

none

2.11.2.3 void DACWriteCh0 (uint16_t val)

Funkcja wpisujaca wartosc do DAC.

Parametry

<i>val</i>	: wartosc wyjsciova
------------	---------------------

Zwraca

none

2.12 Dokumentacja pliku font5x7.h

Plik zawierajacy tablice czcionek.

Definicje

- #define **FONT_OFFSET** 32
Tablica czcionki 5x7.
- #define **FONT_WIDTH** 5

Zmienne

- prog_uint8_t **font5x7** []

2.12.1 Opis szczegółowy

Plik zawierajacy tablice czcionek.

Autor

Arkadiusz Hudzikowski

Wersja

1.0

Data

22.11.2011

2.12.2 Dokumentacja definicji**2.12.2.1 #define FONT_OFFSET 32**

Tablica czcionki 5x7.

Czcionka pochodzi z przykladów obsługi wyświetlacza ze sterownikiem S6B0724 Autor: Radosław Kwiecien

2.13 Dokumentacja pliku Generator.c

Plik podprogramu generatora.

```
#include <avr/io.h>
#include <util/delay.h>
#include <avr/pgmspace.h>
#include <avr/eeprom.h>
#include "DAC.h"
#include "ADC.h"
#include "Keyboard.h"
#include "lcd132x64.h"
#include "Grafika.h"
```

Definicje

- #define [USE_GENERATOR_MODULE](#)
- #define [F_clk](#) 2048000000
- #define [F_max](#) 32000000

Funkcje

- void [EepromTabRead](#) (uint8_t *wsk, uint8_t *e_wsk)
Funkcja odczytująca sygnał z pamięci EEPROM.
- void [EepromTabWrite](#) (uint8_t *e_wsk, uint8_t *wsk)
Funkcja zapisująca sygnał do pamięci EEPROM.

- void [GenSetParam](#) (uint16_t per, uint16_t tab, uint16_t gain, uint16_t duty, int16_t dc_shift, uint8_t type)

Funkcja ustawiajaca parametry generatora sygnalu (synteza)

- void [Generator](#) (void)

Funkcja glowna podprogramu generatora.

Zmienne

- uint16_t **kan_out** [512]
- uint8_t **kan1_lcd** [128]
- EEMEM uint8_t **e_kan_out** [768]
- prog_char [mod_tab](#) [3][3] = {"FM", "AM", "SW"}

Tablice napisow.

- prog_char [Wave_tab](#) [7][11] = {"sine ", "square ", "triangle ", "pink noise", "white noise", "arbitrary ", "none (off)"}

Tablice napisow.

- prog_int16_t [sin_tab](#) [640]

Tablica funkcji sin [0, 2.5PI].

- uint32_t **Freq** = 64516
- uint16_t **Gain** = 2000
- uint8_t **Duty** = 50
- uint8_t **Type** = 0
- int16_t **Dc_shift** = 0
- uint32_t **per_c** = 62
- uint32_t **tab_c** = 512

2.13.1 Opis szczegółowy

Plik podprogramu generatora.

Autor

Arkadiusz Hudzikowski

Wersja

1.0

Data

22.11.2011

2.13.2 Dokumentacja definicji

2.13.2.1 `#define F_clk 2048000000`

czestotliwosc taktowania timera x64 (rozdzielczosc 1/64Hz)

2.13.2.2 `#define F_max 32000000`

maksymalna wyjsciowa czestotliwosc x64 (0.5MHz)

2.13.2.3 `#define USE_GENERATOR_MODULE`

Uzyj tego podprogramu

2.13.3 Dokumentacja funkcji

2.13.3.1 `void EepromTabRead (uint8_t * wsk, uint8_t * e_wsk)`

Funkcja odczytujaca sygnal z pamieci EEPROM.

Parametry

<code>*wsk</code>	: wskaznik adresu bufora docelowego
<code>*e_wsk</code>	: wskaznik adresu bufora w pamieci EEPROM

Zwraca

none

2.13.3.2 `void EepromTabWrite (uint8_t * e_wsk, uint8_t * wsk)`

Funkcja zapisujaca sygnal do pamieci EEPROM.

Parametry

<code>*e_wsk</code>	: wskaznik adresu bufora w pamieci EEPROM
<code>*wsk</code>	: wskaznik adresu bufora wejsciowego

Zwraca

none

2.13.3.3 `void Generator (void)`

Funkcja glowna podprogramu generatora.

Zwraca

none

2.13.3.4 void GenSetParam (uint16_t *per*, uint16_t *tab*, uint16_t *gain*, uint16_t *duty*, int16_t *dc_shift*, uint8_t *type*)

Funkcja ustawiajaca parametry generatora sygnalu (synteza)

Parametry

<i>per</i>	: period - okres probkowania w (1/32M)s (okres timera taktowanego 32MHz) (32 - 65535)
<i>tab</i>	: table - dlugosc bufora sygnalu
<i>gain</i>	: wzmacnienie sygnalu (0 - 2047, do 10000 z przesterowaniem)
<i>duty</i>	: wypelnienie (symetria) sygnalu (prostokat, trojkat) (1 - 99)
<i>dc_shift</i>	: przesunienie (skladowa stala) sygnalu (-2000 - 2000)
<i>type</i>	: typ sygnalu: 0 - sin, 1 - prost., 2 - trojkat, 3 - szum b., 4 - szum r., 5 - arb., 6 - brak

Zwraca

none

2.14 Dokumentacja pliku Generator.h

plik naglowkowy podprogramu generatora

Funkcje

- void [Generator](#) (void)
Funkcja glowna podprogramu generatora.
- void [GenSetParam](#) (uint16_t per, uint16_t tab, uint16_t gain, uint16_t duty, int16_t dc_shift, uint8_t type)
Funkcja ustawiajaca parametry generatora sygnalu (synteza)

2.14.1 Opis szczegółowy

plik naglowkowy podprogramu generatora

Autor

Arkadiusz Hudzikowski

Wersja

1.0

Data

22.11.2011

2.14.2 Dokumentacja funkcji**2.14.2.1 void Generator (void)**

Funkcja glowna podprogramu generatora.

Zwraca

none

2.14.2.2 void GenSetParam (uint16_t per, uint16_t tab, uint16_t gain, uint16_t duty, int16_t dc_shift, uint8_t type)

Funkcja ustawiajaca parametry generatora sygnalu (synteza)

Parametry

<i>per</i>	: period - okres probkowania w (1/32M)s (okres timera taktowanego 32MHz) (32 - 65535)
<i>tab</i>	: table - dlugosc bufora sygnalu
<i>gain</i>	: wzmacnienie sygnalu (0 - 2047, do 10000 z przesterowaniem)
<i>duty</i>	: wypelnienie (symetria) sygnalu (prostokat, trojkat) (1 - 99)
<i>dc_shift</i>	: przesunienie (skladowa stala) sygnalu (-2000 - 2000)
<i>type</i>	: typ sygnalu: 0 - sin, 1 - prost., 2 - trojkat, 3 - szum b., 4 - szum r., 5 - arb., 6 - brak

Zwraca

none

2.15 Dokumentacja pliku Grafika.c

Plik funkcji graficznych.

```
#include <avr/io.h>
#include <avr/pgmspace.h>
#include "lcd132x64.h"
#include "Grafika.h"
```

Funkcje

- void **LCDU8** (uint8_t n)
Funkcja wyswietlajaca zmienna 8-bitowa bez znaku.
- void **LCDI8** (int8_t n)
Funkcja wyswietlajaca zmienna 8-bitowa ze znakiem.
- void **LCDI10** (int16_t n)
Funkcja wyswietlajaca zmienna 16-bitowa ze znakiem w zakresie do |999|.
- void **LCDU16** (uint16_t n)
Funkcja wyswietlajaca zmienna 16-bitowa bez znaku.
- void **LCDI16** (int16_t n)
Funkcja wyswietlajaca zmienna 16-bitowa ze znakiem.
- void **LCDU32** (uint32_t n)
Funkcja wyswietlajaca zmienna 32-bitowa bez znaku.
- void **LCDI32** (int32_t n)
Funkcja wyswietlajaca zmienna 32-bitowa ze znakiem.
- void **LCDUF6** (uint8_t n)
Funkcja wyswietlajaca 6-cio bitowa czesc ulamkowa.
- void **LCDU16mV** (uint16_t n)
Funkcja wyswietlajaca zmienna 16-bitowa bez znaku wyrazona w mV.
- void **LCDI16mV** (int16_t n)
Funkcja wyswietlajaca zmienna 16-bitowa ze znakiem, wyrazona w mV.
- void **LCDosc** (uint8_t *wsk, uint8_t *wsk2, uint8_t xpos, uint8_t ypos1, uint8_t ypos2, uint8_t cur1, uint8_t cur2)
Funkcja wyswietlajaca oscylogramy oraz kursory i wskazniki.
- void **LCDWriteScaleLine** (uint8_t s, uint8_t v)
Funkcja wyswietlajaca informacje o podstawie czasu i wzmacnienie.
- void **LCDWriteAnScaleLine** (uint8_t s, uint8_t v)
Funkcja wyswietlajaca informacje o podstawie czestotliwosci i wzmacnienie.
- void **LCDWritePositionLine** (int16_t x, int8_t y)
Funkcja wyswietlajaca informacje o przesunieciu przebiegu.
- void **LCDWriteTriggerLine** (uint8_t trig, int16_t lev)

2.15.2.2 void LCDI16 (int16_t n)

Funkcja wyswietlajaca zmienna 16-bitowa ze znakiem.

Parametry

<i>n</i>	: zmienna do wyswietlenia (-32768 - 32767)
----------	--------------------------------------------

Zwraca

none

2.15.2.3 void LCDI16mV (int16_t n)

Funkcja wyswietlajaca zmienna 16-bitowa ze znakiem, wyrazona w mV.

Parametry

<i>n</i>	: zmienna do wyswietlenia (-32768 - 32767)
----------	--------------------------------------------

Zwraca

none

2.15.2.4 void LCDI32 (int32_t n)

Funkcja wyswietlajaca zmienna 32-bitowa ze znakiem.

Parametry

<i>n</i>	: zmienna do wyswietlenia (-999999 - 999999)
----------	----------------------------------------------

Zwraca

none

2.15.2.5 void LCDI8 (int8_t n)

Funkcja wyswietlajaca zmienna 8-bitowa ze znakiem.

Parametry

<i>n</i>	: zmienna do wyswietlenia (-128 - 127)
----------	----------------------------------------

Zwraca

none

2.15.2.6 void LCDosc (uint8_t * wsk, uint8_t * wsk2, uint8_t xpos, uint8_t ypos1, uint8_t ypos2, uint8_t cur1, uint8_t cur2)

Funkcja wyswietlajaca oscylogramy oraz kursory i wskazniki.

Struktura bufora wyswietlacza:

B0.0 B8.0 ... B1048.0

B0.1 B8.1

B0.2 B8.2

B0.3 B8.3

B0.4 B8.4

B0.5 B8.5

B0.6 B8.6

B0.7 B8.7

B1.0 B8.0

B1.1 B8.1

... ..

B7.7 B15.7 ... B1055.7

Parametry

*wsk	: adres bufora kanalu 1
*wsk2	: adres bufora kanalu 2
xpos	: pozycja przebiegow w poziomie (0 - 255), pozycja zerowa - 62
ypos1	: pozycja przebiegu 1 w pionie (0 - 255), pozycja zerowa - 128
ypos2	: pozycja przebiegu 2 w pionie (0 - 254), pozycja zerowa - 128, wartosc 255 powoduje wygaszenie przebiegu 2
cur1	: pozycja kursora 1 (0 - 255)
cur2	: pozycja kursora 2 (0 - 255)

Zwraca

none

2.15.2.7 void LCDU16 (uint16_t n)

Funkcja wyswietlajaca zmienna 16-bitowa bez znaku.

Parametry

n	: zmienna do wyswietlenia (0 - 65535)
---	---------------------------------------

Zwraca

none

2.15.2.8 void LCDU16mV (uint16_t n)

Funkcja wyswietlajaca zmienna 16-bitowa bez znaku wyrazona w mV.

Parametry

n	: zmienna do wyswietlenia (0 - 65535)
-----	---------------------------------------

Zwraca

none

2.15.2.9 void LCDU32 (uint32_t n)

Funkcja wyswietlajaca zmienna 32-bitowa bez znaku.

Parametry

n	: zmienna do wyswietlenia (0 - 999999)
-----	----------------------------------------

Zwraca

none

2.15.2.10 void LCDU8 (uint8_t n)

Funkcja wyswietlajaca zmienna 8-bitowa bez znaku.

Parametry

n	: zmienna do wyswietlenia (0 - 255)
-----	-------------------------------------

Zwraca

none

2.15.2.11 void LCDUF6 (uint8_t n)

Funkcja wyswietlajaca 6-cio bitowa czesc ulamkowa.

Wartosc wyswietlana wynosi: $1/n$ z dokladnoscia 2 cyfr po przecinku

Parametry

n	: 6-cio bitowa czesc ulamkowa (0 - 63)
-----	----------------------------------------

Zwraca

none

2.15.2.12 void LCDWriteAnScaleLine (uint8_t s, uint8_t v)

Funkcja wyswietlajaca informacje o podstawie czestotliwosci i wzmacnienie.

Parametry

<i>s</i>	: podstawa czestotliwosci (0 - 14)
<i>v</i>	: wzmacnienie (0 - 6 lub 0 - 9 z programowym wzmacnieniem)

Zwraca

none

2.15.2.13 void LCDWriteFreqCursorLine (uint32_t freq, int8_t db)

Funkcja wyswietlajaca informacje o kursorze czestotliwosci.

Parametry

<i>freq</i>	: czestotliwosc wskazywana przez kursor
<i>db</i>	: sila sygnalu w miejscu wskazywanym przez kursor

Zwraca

none

2.15.2.14 void LCDWritePositionLine (int16_t x, int8_t y)

Funkcja wyswietlajaca informacje o przesunieciu przebiegu.

Parametry

<i>x</i>	: przesuniecie w poziomie
<i>y</i>	: przesuniecie w pionie

Zwraca

none

2.15.2.15 void LCDWriteScaleLine (uint8_t s, uint8_t v)

Funkcja wyswietlajaca informacje o podstawie czasu i wzmacnienie.

Parametry

<i>s</i>	: podstawa czasu (0 - 14)
<i>v</i>	: wzmacnienie (0 - 6 lub 0 - 9 z programowym wzmacnieniem)

Zwraca

none

2.15.2.16 void LCDWriteTimeCursorLine (int16_t *cur*, uint8_t *sd*)

Funkcja wyswietlajaca informacje o kursorach czasu.

Parametry

<i>cur</i>	: odleglosc kursora / miedzy kursorami
<i>sd</i>	: podstawa czasu

Zwraca

none

2.15.2.17 void LCDWriteTriggerLine (uint8_t *trig*, int16_t *lev*)

Funkcja wyswietlajaca informacje o wyzwalaniu.

bity odpowiadajace za ustawienie triggera

7 - ---

[6 5] - zaznacz: typ, zbocze, filtr, ---

4 - ---

3 - filtr: 'LF', 'HF'

2 - zbocze: '\', '/'

[1 0] - typ: '-', 'N', 'A', 'S'

Parametry

<i>trig</i>	: sposob wyzwalania jak wyzej
<i>lev</i>	: poziom wyzwalania

Zwraca

none

2.16 Dokumentacja pliku Grafika.h

Plik naglowkowy funkcji graficznych.

Wyliczenia

- enum **display_type** { **SV_DIV**, **XY_POS**, **TRIG**, **CURSORS** }

Funkcje

- void **LCDosc** (uint8_t *wsk, uint8_t *wsk2, uint8_t xpos, uint8_t ypos1, uint8_t ypos2, uint8_t cur1, uint8_t cur2)

Funkcja wyswietlajaca oscylogramy oraz kursory i wskazniki.

- void **lcd_osc4** (uint8_t *wsk, uint8_t rozdziel)
- void **LCDWriteScaleLine** (uint8_t s, uint8_t v)

Funkcja wyswietlajaca informacje o podstawie czasu i wzmacnienie.

- void **LCDWriteAnScaleLine** (uint8_t s, uint8_t v)

Funkcja wyswietlajaca informacje o podstawie czestotliwosci i wzmacnienie.

- void **LCDWritePositionLine** (int16_t x, int8_t y)

Funkcja wyswietlajaca informacje o przesunieciu przebiegu.

- void **LCDWriteTriggerLine** (uint8_t trig, int16_t lev)

Funkcja wyswietlajaca informacje o wyzwalaniu.

- void **LCDWriteTimeCursorLine** (int16_t cur, uint8_t sd)

Funkcja wyswietlajaca informacje o kursorach czasu.

- void **LCDWriteFreqCursorLine** (uint32_t freq, int8_t db)

Funkcja wyswietlajaca informacje o kursorze czestotliwosci.

- void **LCDU32** (uint32_t n)

Funkcja wyswietlajaca zmienna 32-bitowa bez znaku.

- void **LCDI32** (int32_t n)

Funkcja wyswietlajaca zmienna 32-bitowa ze znakiem.

- void **LCDU16** (uint16_t n)

Funkcja wyswietlajaca zmienna 16-bitowa bez znaku.

- void **LCDI16** (int16_t n)

Funkcja wyswietlajaca zmienna 16-bitowa ze znakiem.

- void **LCDU8** (uint8_t n)

Funkcja wyswietlajaca zmienna 8-bitowa bez znaku.

- void **LCDI8** (int8_t n)

Funkcja wyswietlajaca zmienna 8-bitowa ze znakiem.

- void **LCDI10** (int16_t n)

Funkcja wyswietlajaca zmienna 16-bitowa ze znakiem w zakresie do |999|.

- void **LCDUF6** (uint8_t n)

Funkcja wyswietlajaca 6-cio bitowa czesc ulamkowa.

- void **LCDU16mV** (uint16_t n)

Funkcja wyswietlajaca zmienna 16-bitowa bez znaku wyrazona w mV.

- void **LCDI16mV** (int16_t n)

Funkcja wyswietlajaca zmienna 16-bitowa ze znakiem, wyrazona w mV.

2.16.1 Opis szczegółowy

Plik naglowkowy funkcji graficznych.

Autor

Arkadiusz Hudzikowski

Wersja

1.0

Data

22.11.2011

2.16.2 Dokumentacja funkcji

2.16.2.1 void LCDI10 (int16_t n)

Funkcja wyswietlajaca zmienna 16-bitowa ze znakiem w zakresie do |999|.

Parametry

<i>n</i>	: zmienna do wyswietlenia (-999 - 999)
----------	----------------------------------------

Zwraca

none

2.16.2.2 void LCDI16 (int16_t n)

Funkcja wyswietlajaca zmienna 16-bitowa ze znakiem.

Parametry

<i>n</i>	: zmienna do wyswietlenia (-32768 - 32767)
----------	--------------------------------------------

Zwraca

none

2.16.2.3 void LCDI16mV (int16_t *n*)

Funkcja wyswietlajaca zmienna 16-bitowa ze znakiem, wyrazona w mV.

Parametry

<i>n</i>	: zmienna do wyswietlenia (-32768 - 32767)
----------	--------------------------------------------

Zwraca

none

2.16.2.4 void LCDI32 (int32_t *n*)

Funkcja wyswietlajaca zmienna 32-bitowa ze znakiem.

Parametry

<i>n</i>	: zmienna do wyswietlenia (-999999 - 999999)
----------	----------------------------------------------

Zwraca

none

2.16.2.5 void LCDI8 (int8_t *n*)

Funkcja wyswietlajaca zmienna 8-bitowa ze znakiem.

Parametry

<i>n</i>	: zmienna do wyswietlenia (-128 - 127)
----------	----------------------------------------

Zwraca

none

2.16.2.6 void LCDosc (uint8_t * wsk, uint8_t * wsk2, uint8_t xpos, uint8_t ypos1, uint8_t ypos2, uint8_t cur1, uint8_t cur2)

Funkcja wyswietlajaca oscylogramy oraz kursory i wskazniki.

Struktura bufora wyswietlacza:

B0.0 B8.0 ... B1048.0

B0.1 B8.1

B0.2 B8.2

B0.3 B8.3

B0.4 B8.4

B0.5 B8.5

B0.6 B8.6

B0.7 B8.7

B1.0 B8.0

B1.1 B8.1

... ..

B7.7 B15.7 ... B1055.7

Parametry

*wsk	: adres bufora kanalu 1
*wsk2	: adres bufora kanalu 2
xpos	: pozycja przebiegow w poziomie (0 - 255), pozycja zerowa - 62
ypos1	: pozycja przebiegu 1 w pionie (0 - 255), pozycja zerowa - 128
ypos2	: pozycja przebiegu 2 w pionie (0 - 254), pozycja zerowa - 128, wartosc 255 powoduje wygaszenie przebiegu 2
cur1	: pozycja kursora 1 (0 - 255)
cur2	: pozycja kursora 2 (0 - 255)

Zwraca

none

2.16.2.7 void LCDU16 (uint16_t n)

Funkcja wyswietlajaca zmienna 16-bitowa bez znaku.

Parametry

n	: zmienna do wyswietlenia (0 - 65535)
---	---------------------------------------

Zwraca

none

2.16.2.8 void LCDU16mV (uint16_t *n*)

Funkcja wyswietlajaca zmienna 16-bitowa bez znaku wyrazona w mV.

Parametry

<i>n</i>	: zmienna do wyswietlenia (0 - 65535)
----------	---------------------------------------

Zwraca

none

2.16.2.9 void LCDU32 (uint32_t *n*)

Funkcja wyswietlajaca zmienna 32-bitowa bez znaku.

Parametry

<i>n</i>	: zmienna do wyswietlenia (0 - 999999)
----------	----------------------------------------

Zwraca

none

2.16.2.10 void LCDU8 (uint8_t *n*)

Funkcja wyswietlajaca zmienna 8-bitowa bez znaku.

Parametry

<i>n</i>	: zmienna do wyswietlenia (0 - 255)
----------	-------------------------------------

Zwraca

none

2.16.2.11 void LCDUF6 (uint8_t *n*)

Funkcja wyswietlajaca 6-cio bitowa czesc ulamkowa.

Wartosc wyswietlana wynosi: 1/*n* z dokladnoscia 2 cyfr po przecinku

Parametry

<i>n</i>	: 6-cio bitowa czesc ulamkowa (0 - 63)
----------	----------------------------------------

Zwraca

none

2.16.2.12 void LCDWriteAnScaleLine (uint8_t s, uint8_t v)

Funkcja wyswietlajaca informacje o podstawie czestotliwosci i wzmacnienie.

Parametry

<i>s</i>	: podstawa czestotliwosci (0 - 14)
<i>v</i>	: wzmacnienie (0 - 6 lub 0 - 9 z programowym wzmacnieniem)

Zwraca

none

2.16.2.13 void LCDWriteFreqCursorLine (uint32_t freq, int8_t db)

Funkcja wyswietlajaca informacje o kursorze czestotliwosci.

Parametry

<i>freq</i>	: czestotliwosc wskazywana przez kursor
<i>db</i>	: sila sygnalu w miejscu wskazywanym przez kursor

Zwraca

none

2.16.2.14 void LCDWritePositionLine (int16_t x, int8_t y)

Funkcja wyswietlajaca informacje o przesunieciu przebiegu.

Parametry

<i>x</i>	: przesuniecie w poziomie
<i>y</i>	: przesuniecie w pionie

Zwraca

none

2.16.2.15 void LCDWriteScaleLine (uint8_t s, uint8_t v)

Funkcja wyswietlajaca informacje o podstawie czasu i wzmacnienie.

Parametry

<i>s</i>	: podstawa czasu (0 - 14)
<i>v</i>	: wzmacnienie (0 - 6 lub 0 - 9 z programowym wzmacnieniem)

Zwraca

none

2.16.2.16 void LCDWriteTimeCursorLine (int16_t *cur*, uint8_t *sd*)

Funkcja wyswietlajaca informacje o kursorach czasu.

Parametry

<i>cur</i>	: odleglosc kursora / miedzy kursorami
<i>sd</i>	: podstawa czasu

Zwraca

none

2.16.2.17 void LCDWriteTriggerLine (uint8_t *trig*, int16_t *lev*)

Funkcja wyswietlajaca informacje o wyzwalaniu.

bity odpowiadajace za ustawienie triggera

7 - ---

[6 5] - zaznacz: typ, zbocze, filtr, ---

4 - ---

3 - filtr: 'LF', 'HF'

2 - zbocze: '\', '/'

[1 0] - typ: '-', 'N', 'A', 'S'

Parametry

<i>trig</i>	: sposob wyzwalania jak wyzej
<i>lev</i>	: poziom wyzwalania

Zwraca

none

2.17 Dokumentacja pliku Keyboard.c

Plik obsługi klawiatury.

```
#include <avr/io.h>
```

```
#include <avr/interrupt.h>
```

```
#include <util/delay.h>
```

```
#include "Keyboard.h"
```

Definicje

- `#define KEYB_INT0 PORTB##_INT0_vect`

Funkcje

- void `KeybInit` (void)
Funkcja konfigurująca wyprowadzenia do obsługi klawiatury.
- `ISR` (KEYB_INT0)
Przerwanie od przycisku, z parametrem 'naked' w celu szybszego wykonywania.
- int16_t `ShiftValue` (uint8_t key, int16_t val, const int16_t min, const int16_t max, uint8_t step, const uint8_t key1, const uint8_t key2)
Funkcja wygodnej zmiany wartości zmiennych.
- uint8_t `Keyboard` (void)
Funkcja zwracająca kod naciśniętego przycisku.

2.17.1 Opis szczegółowy

Plik obsługi klawiatury.

Autor

Arkadiusz Hudzikowski

Wersja

1.0

Data

22.11.2011

2.17.2 Dokumentacja funkcji

2.17.2.1 `ISR (KEYB_INT0)`

Przerwanie od przycisku, z parametrem 'naked' w celu szybszego wykonywania.

Ustawiona zostaje flaga w rejestrze. Można ją wykorzystać do sygnalizacji przycisnięcia w funkcjach, w których nie ma czasu na obsługę klawiatury.

2.17.2.2 void Keyblnit (void)

Funkcja konfiguruje wyprowadzenia do obsługi klawiatury.

Zwraca

none

2.17.2.3 uint8_t Keyboard (void)

Funkcja zwracająca kod naciśniętego przycisku.

Zwraca

uint8_t : kod naciśniętego przycisku

2.17.2.4 int16_t ShiftValue (uint8_t key, int16_t val, const int16_t min, const int16_t max, uint8_t step, const uint8_t key1, const uint8_t key2)

Funkcja wygodnej zmiany wartości zmiennych.

Parametry

<i>key</i>	: kod przycisku
<i>val</i>	: wartość zmiennej
<i>min</i>	: minimalna wartość ustawienia
<i>max</i>	: maksymalna wartość ustawienia
<i>step</i>	: krok zmiany przy naciśnięciu przycisku OK
<i>key1</i>	: klawisz inkrementacji
<i>key2</i>	: klawisz dekrementacji

Zwraca

nowa wartość zmiennej

2.18 Dokumentacja pliku Keyboard.h

Plik nagłówkowy obsługi klawiatury.

Definicje

- #define **KEYB_PORT** PORTC
- #define **PORTK1** PORTB
- #define **PINK1** 1
- #define **PORTK2** PORTK1

- #define **PINK2** 3
- #define **P_LEFT** 1
- #define **P_RIGHT** 2
- #define **P_OK** 4
- #define **P_UP** 8
- #define **P_DOWN** 16
- #define **P_EXIT** 32
- #define **P_DIV** 64
- #define **P_XY** 96
- #define **P_TRIG** 128
- #define **P_CURS** 160

Funkcje

- void **KeybInit** (void)
Funkcja konfigurująca wyprowadzenia do obsługi klawiatury.
- int16_t **ShiftValue** (uint8_t key, int16_t val, const int16_t min, const int16_t max, uint8_t step, const uint8_t key1, const uint8_t key2)
Funkcja wygodnej zmiany wartości zmiennych.
- uint8_t **Keyboard** (void)
Funkcja zwracająca kod naciśniętego przycisku.

2.18.1 Opis szczegółowy

Plik nagłówkowy obsługi klawiatury.

Autor

Arkadiusz Hudzikowski

Wersja

1.0

Data

22.11.2011

2.18.2 Dokumentacja funkcji

2.18.2.1 void Keyblnit (void)

Funkcja konfigurująca wyprowadzenia do obsługi klawiatury.

Zwraca

none

2.18.2.2 uint8_t Keyboard (void)

Funkcja zwracająca kod naciśniętego przycisku.

Zwraca

uint8_t : kod naciśniętego przycisku

2.18.2.3 int16_t ShiftValue (uint8_t key, int16_t val, const int16_t min, const int16_t max, uint8_t step, const uint8_t key1, const uint8_t key2)

Funkcja wygodnej zmiany wartości zmiennych.

Parametry

<i>key</i>	: kod przycisku
<i>val</i>	: wartość zmiennej
<i>min</i>	: minimalna wartość ustawienia
<i>max</i>	: maksymalna wartość ustawienia
<i>step</i>	: krok zmiany przy naciśnięciu przycisku OK
<i>key1</i>	: klawisz inkrementacji
<i>key2</i>	: klawisz dekrementacji

Zwraca

nowa wartość zmiennej

2.19 Dokumentacja pliku lcd132x64.c

Plik obsługi wyświetlacza.

```
#include <avr/io.h>
#include <util/delay.h>
#include <avr/pgmspace.h>
#include <avr/eeprom.h>
#include <inttypes.h>
#include "lcd132x64.h"
#include "font5x7.h"
```

Definicje

- #define **SPLC501C_SET** IOSET0
- #define **SPLC501C_CLR** IOCLR0
- #define **SPLC501C_DIR** IODIR0

- #define **SPLC501C_PIN** IOPIN0
- #define **SPLC501C_RD** (1 << 0)
- #define **SPLC501C_WR** (1 << 1)
- #define **SPLC501C_A0** (1 << 2)
- #define **SPLC501C_RES** (1 << 3)
- #define **SPLC501C_CS1** (1 << 0)
- #define **LCD_CTRL_PORT** PORTA
- #define **LCD_CTRL_PORT2** PORTE
- #define **VCC_PORT** PORTA
- #define **VCC_PIN** (1 << 4)
- #define **DA_PIN** (1 << 1)
- #define **LCD_DATA_PORT** PORTC

Funkcje

- void **LCDInitPort** (void)
Funkcja konfigurująca wyprowadzenia LCD.
- void **LCDWriteData** (uint8_t data)
Funkcja wysyłająca jeden bajt do wyświetlacza.
- void **LCDWriteComm** (uint8_t comm)
Funkcja wysyłająca komendę do wyświetlacza.
- void **LCDInit** (void)
Funkcja konfigurująca wyświetlacz LCD.
- void **LCDOff** (void)
Funkcja wyłączająca wyświetlacz.
- uint8_t **LCDBright** (int8_t step)
Funkcja ustawiająca podświetlenie wyświetlacza.
- uint8_t **LCDContrast** (int8_t step)
Funkcja ustawiająca kontrast wyświetlacza.
- void **LCDGoTo** (uint8_t x, uint8_t y)
Funkcja ustawiająca kursor na wybranej pozycji.
- void **LCDClearScreen** (void)
Funkcja czyszcząca ekran wyświetlacza.
- void **LCDWriteChar** (uint8_t charCode)
Funkcja wysyłająca znak do wyświetlacza.
- void **LCDWriteCharNeg** (uint8_t charCode)

Funkcja wysylajaca znak do wyswietlacza z negacja kolorow.

- void **LCDText** (prog_char *string)

Funkcja wysylajaca lancuch znakow z pamieci FLASH do wyswietlacza.

- void **LCDTextNeg** (prog_char *string)

Funkcja wysylajaca lancuch znakow z pamieci FLASH do wyswietlacza z negacja kolorow.

Zmienne

- EEMEM uint8_t **e_contrast**
- EEMEM int8_t **e_bright**

2.19.1 Opis szczegółowy

Plik obsługi wyświetlacza.

Autor

Arkadiusz Hudzikowski

Wersja

1.0

Data

22.11.2011 Wyświetlacz 132x64 pikseli ze sterownikiem SPLC501C. Plik stworzony na podstawie biblioteki ze strony <http://en.radzio.dxp.pl/splc501c/>
Autor biblioteki: Radosław Kwiecien

2.19.2 Dokumentacja funkcji

2.19.2.1 uint8_t LCDBright (int8_t step)

Funkcja ustawiająca podświetlenie wyświetlacza.

Parametry

<i>step</i>	: wartosc zmiany podswietlenia (-100 - 100)
-------------	---------------------------------------------

Zwraca

uint8_t : nowa wartosc podswietlenia (0 - 100)

2.19.2.2 void LCDClearScreen (void)

Funkcja czyszczaca ekran wyswietlacza.

Zwraca

none

2.19.2.3 uint8_t LCDContrast (int8_t *step*)

Funkcja ustawiajaca kontrast wyswietlacza.

Parametry

<i>step</i>	: wartosc zmiany kontrastu (-20 - 20)
-------------	---------------------------------------

Zwraca

uint8_t : nowa wartosc kontrastu (10 - 30)

2.19.2.4 void LCDGoTo (uint8_t *x*, uint8_t *y*)

Funkcja ustawiajaca kursor na wybranej pozycji.

Parametry

<i>x</i>	: nr znaku w wierszu
<i>y</i>	: nr wiersza

Zwraca

none

2.19.2.5 void LCDInit (void)

Funkcja konfigurujaca wyswietlacz LCD.

Zwraca

none

2.19.2.6 void LCDInitPort (void)

Funkcja konfigurujaca wyprowadzenia LCD.

Zwraca

none

2.19.2.7 void LCDOff (void)

Funkcja wylaczajaca wyswietlacz.

Zwraca

none

2.19.2.8 void LCDText (prog_char * string)

Funkcja wysylajaca lancuch znakow z pamieci FLASH do wyswietlacza.

Parametry

<i>*string</i>	: adres lancucha znakow z pamieci flash
----------------	-----------------------------------------

Zwraca

none

2.19.2.9 void LCDTextNeg (prog_char * string)

Funkcja wysylajaca lancuch znakow z pamieci FLASH do wyswietlacza z negacja kolorow.

Parametry

<i>*string</i>	: adres lancucha znakow z pamieci flash
----------------	-----------------------------------------

Zwraca

none

2.19.2.10 void LCDWriteChar (uint8_t charCode)

Funkcja wysylajaca znak do wyswietlacza.

Parametry

<i>charCode</i>	: znak ASCII
-----------------	--------------

Zwraca

none

2.19.2.11 void LCDWriteCharNeg (uint8_t *charCode*)

Funkcja wysylajaca znak do wyswietlacza z negacja kolorow.

Parametry

<i>charCode</i>	: znak ASCII
-----------------	--------------

Zwraca

none

2.19.2.12 void LCDWriteComm (uint8_t *comm*)

Funkcja wysylajaca komende do wyswietlacza.

Parametry

<i>comm</i>	: komenda
-------------	-----------

Zwraca

none

2.19.2.13 void LCDWriteData (uint8_t *data*)

Funkcja wysylajaca jeden bajt do wyswietlacza.

Parametry

<i>data</i>	: bajt danych
-------------	---------------

Zwraca

none

2.20 Dokumentacja pliku lcd132x64.h

Plik naglowkowy obslugi wyswietlacza.

```
#include <avr/pgmspace.h>
```

Definicje

- #define **SCREEN_WIDTH** 132
- #define **SCREEN_HEIGHT** 64
- #define **PIXELS_PER_PAGE** 8

- #define **SPLC501C_DISPLAY_ON** 0xAF
- #define **SPLC501C_DISPLAY_OFF** 0xAE
- #define **SPLC501C_START_LINE** 0x40
- #define **SPLC501C_PAGE_ADDRESS** 0xB0
- #define **SPLC501C_COLUMN_ADDRESS_HI** 0x10
- #define **SPLC501C_COLUMN_ADDRESS_LO** 0x00
- #define **SPLC501C_ADC_NORMAL** 0xA0
- #define **SPLC501C_ADC_REVERSE** 0xA1
- #define **SPLC501C_DISPLAY_NORMAL** 0xA6
- #define **SPLC501C_DISPLAY_REVERSE** 0xA7
- #define **SPLC501C_DISPLAY_ALL_ON** 0xA5
- #define **SPLC501C_DISPLAY_ALL_OFF** 0xA4
- #define **SPLC501C_BIAS_19** 0xA2
- #define **SPLC501C_BIAS_15** 0xA3
- #define **SPLC501C_RMW_START** 0xE0
- #define **SPLC501C_RMW_END** 0xEE
- #define **SPLC501C_RESET** 0xE2
- #define **SPLC501C_COM0** 0xC0
- #define **SPLC501C_COM63** 0xC8
- #define **SPLC501C_POWERON** 0x2F
- #define **SPLC501C_VOLTAGE_RATIO** 0x20
- #define **SPLC501C_VOLUME_MODE** 0x81
- #define **SPLC501C_VOLUME_SET** 0x00
- #define **SPLC501C_PAGE_BLINKING_MODE** 0xD5
- #define **SPLC501C_PAGE_BLINKING_0** 0x01
- #define **SPLC501C_PAGE_BLINKING_1** 0x02
- #define **SPLC501C_PAGE_BLINKING_2** 0x04
- #define **SPLC501C_PAGE_BLINKING_3** 0x08
- #define **SPLC501C_PAGE_BLINKING_4** 0x10
- #define **SPLC501C_PAGE_BLINKING_5** 0x20
- #define **SPLC501C_PAGE_BLINKING_6** 0x40
- #define **SPLC501C_PAGE_BLINKING_7** 0x80

Funkcje

- void **LCDOff** (void)
Funkcja wylaczajaca wyswietlacz.
- uint8_t **LCDBright** (int8_t step)
Funkcja ustawiajaca podswietlenie wyswietlacza.
- uint8_t **LCDContrast** (int8_t step)
Funkcja ustawiajaca kontrast wyswietlacza.
- void **LCDWriteData** (unsigned char dataToWrite)
- void **LCDGoTo** (unsigned char, unsigned char)

- void **LCDWriteChar** (uint8_t charCode)
Funkcja wysylajaca znak do wyswietlacza.
- void **LCDWriteCharNeg** (uint8_t charCode)
Funkcja wysylajaca znak do wyswietlacza z negacja kolorow.
- void **LCDText** (prog_char *string)
Funkcja wysylajaca lancuch znakow z pamieci FLASH do wyswietlacza.
- void **LCDTextNeg** (prog_char *string)
Funkcja wysylajaca lancuch znakow z pamieci FLASH do wyswietlacza z negacja kolorow.
- void **LCDInit** (void)
Funkcja konfigurujaca wyswietlacz LCD.
- void **LCDClearScreen** (void)
Funkcja czyszczaca ekran wyswietlacza.
- void **lcd_String_neg** (uint8_t val)

2.20.1 Opis szczegółowy

Plik naglowkowy obsługi wyświetlacza.

Autor

Arkadiusz Hudzikowski

Wersja

1.0

Data

22.11.2011 Wyświetlacz 132x64 pikseli ze sterownikiem SPLC501C. Plik stworzony na podstawie biblioteki ze strony <http://en.radzio.dxp.pl/splc501c/>
Autor biblioteki: Radosław Kwiecien

2.20.2 Dokumentacja funkcji

2.20.2.1 uint8_t LCDBright (int8_t step)

Funkcja ustawiająca podświetlenie wyświetlacza.

Parametry

<i>step</i>	: wartosc zmiany podswietlenia (-100 - 100)
-------------	---------------------------------------------

Zwraca

uint8_t : nowa wartosc podswietlenia (0 - 100)

2.20.2.2 void LCDClearScreen (void)

Funkcja czyszczaca ekran wyswietlacza.

Zwraca

none

2.20.2.3 uint8_t LCDContrast (int8_t step)

Funkcja ustawiajaca kontrast wyswietlacza.

Parametry

<i>step</i>	: wartosc zmiany kontrastu (-20 - 20)
-------------	---------------------------------------

Zwraca

uint8_t : nowa wartosc kontrastu (10 - 30)

2.20.2.4 void LCDInit (void)

Funkcja konfigurujaca wyswietlacz LCD.

Zwraca

none

2.20.2.5 void LCDOff (void)

Funkcja wylaczajaca wyswietlacz.

Zwraca

none

2.20.2.6 void LCDText (prog_char * string)

Funkcja wysylajaca lancuch znakow z pamieci FLASH do wyswietlacza.

Parametry

<i>*string</i>	: adres lancucha znakow z pamieci flash
----------------	-----------------------------------------

Zwraca

none

2.20.2.7 void LCDTextNeg (prog_char * string)

Funkcja wysylajaca lancuch znakow z pamieci FLASH do wyswietlacza z negacja kolorow.

Parametry

<i>*string</i>	: adres lancucha znakow z pamieci flash
----------------	-----------------------------------------

Zwraca

none

2.20.2.8 void LCDWriteChar (uint8_t charCode)

Funkcja wysylajaca znak do wyswietlacza.

Parametry

<i>charCode</i>	: znak ASCII
-----------------	--------------

Zwraca

none

2.20.2.9 void LCDWriteCharNeg (uint8_t charCode)

Funkcja wysylajaca znak do wyswietlacza z negacja kolorow.

Parametry

<i>charCode</i>	: znak ASCII
-----------------	--------------

Zwraca

none

2.21 Dokumentacja pliku main.c

Plik glowny programu.

```
#include <avr/io.h>
#include <util/delay.h>
#include <avr/pgmspace.h>
#include "lcd132x64.h"
#include "Grafika.h"
#include "clksys_driver.h"
#include "ADC.h"
#include "DAC.h"
#include "Keyboard.h"
#include "Oscyloskop.h"
#include "Generator.h"
#include "Wobuloskop.h"
#include "Multimetr.h"
#include "Analizator.h"
#include "AnalizatorStLog.h"
#include "TransmisjaPC.h"
#include "Ustawienia.h"
```

Funkcje

- `int16_t kan1_in[512] __attribute__((section(".data")))`
Bufor przechowujący próbki danych kanału 1.
- `void PrintMainMenu (uint8_t menu)`
Funkcja wyświetlająca menu na ekranie.
- `void CLKInit (void)`
Funkcja inicjująca petle PLL. Ustawienie częstotliwości 32MHz.
- `void CLKIdle (void)`
Funkcja wyłączająca zewnętrzne taktowanie.
- `int main (void)`
Funkcja main.

Zmienne

- `prog_char menu_tab [8][11]`

Tablica napisow menu.

2.21.1 Opis szczegółowy

Plik glowny programu.

Autor

Arkadiusz Hudzikowski

Wersja

1.0

Data

22.11.2011

First version: 02.01.2008 ATmega32 + ATmega8 + LCD Nokia3510i ADS830 2MS/s,
DAC0808 5,33MS/s

Second version (full): 21.10.2009 STM32F103RBT6 + LCD Siemens S65 ADS831
72MS/s, DAC0808 6,67MS/s

Third version (small): 14.07.2010 Xmega32A4 + LCD SPLC501C ADC-Internal 2MS/s,
DAC-Internal 1MS/s

2.21.2 Dokumentacja funkcji

2.21.2.1 `int16_t kan1_in [512] __attribute__((section(".data"))) ;`

Bufor przechowujacy probki danych kanalu 1.

Bufor przechowujacy probki danych kanalu 2 do wyswietlenia na lcd.

Bufor przechowujacy probki danych kanalu 1 do wyswietlenia na lcd.

Bufor przechowujacy probki sygnalu wyjsciowego.

Bufor przechowujacy probki danych kanalu 2.

2.21.2.2 `void CLKIdle (void)`

Funkcja wylaczajaca zewnetrzne taktowanie.

Uruchomiony zostaje wewnetrzny generator 32KHz

Zwraca

none

2.21.2.3 void CLKInit (void)

Funkcja inicjujaca petle PLL. Ustawienie czestotliosci 32MHz.

Zwraca

none

2.21.2.4 int main (void)

Funkcja main.

Zwraca

0

2.21.2.5 void PrintMainMenu (uint8_t menu)

Funkcja wyswietlajaca menu na ekranie.

Parametry

<i>menu</i>	: numer wskazywanej pozycji w menu
-------------	------------------------------------

Zwraca

none

2.21.3 Dokumentacja zmiennych

2.21.3.1 prog_char menu_tab[8][11]

Wartość początkowa:

```
{
    "Oscyloskop",
    "Generator ",
    "Analizator",
    "An.st.log.",
    "Wobuloskop",
    "Multimetr ",
    "RS232->PC ",
    "Ustawienia"
}
```

Tablica napisow menu.

2.22 Dokumentacja pliku Multimetr.c

Plik podprogramu multimetru.

```
#include <avr/io.h>
#include <util/delay.h>
#include <avr/pgmspace.h>
#include <avr/eeprom.h>
#include "lcd132x64.h"
#include "Grafika.h"
#include "clksys_driver.h"
#include "ADC.h"
#include "DAC.h"
#include "Keyboard.h"
#include "Oscyloskop.h"
#include "Generator.h"
#include "Wobuloskop.h"
#include "Analizator.h"
#include "Ustawienia.h"
```

Funkcje

- uint32_t [sqrt32](#) (uint32_t x)
Funkcja obliczająca pierwiastek kwadratowy 32-bitowej liczby.
- void [Multimetr](#) (void)
Funkcja główna podprogramu multimetru.

Zmienne

- int16_t [kan1_in](#) [1024]
- uint8_t [kan1_lcd](#) [128]
- prog_uint16_t [Time_tab](#) [15]
Tablica dzielników dla wyboru podstawy czasu.
- prog_uint8_t [Gain_tab](#) [10]
- EEMEM int8_t [e_offset_cal](#) [14]

2.22.1 Opis szczegółowy

Plik podprogramu multimetru.

Autor

Arkadiusz Hudzikowski

Wersja

1.0

Data

22.11.2011

2.22.2 Dokumentacja funkcji

2.22.2.1 void Multimetr (void)

Funkcja glowna podprogramu multimetru.

Zwraca

none

2.22.2.2 uint32_t sqrt32 (uint32_t x)

Funkcja obliczajaca pierwiastek kwadratowy 32-bitowej liczby.

Parametry

x	: zmienna wejsciova
-----	---------------------

Zwraca

uint32_t : wartosc pierwiastka

2.22.3 Dokumentacja zmiennych

2.22.3.1 EEMEM int8_t e_offset_cal[14]

Tablica wartosci kalibracji offsetu ADC

2.23 Dokumentacja pliku Multimetr.h

Plik naglowkowy podprogramu multimetru.

Funkcje

- void **Multimetr** (void)

Funkcja glowna podprogramu multimetru.

2.23.1 Opis szczegółowy

Plik naglowkowy podprogramu multimetru.

Autor

Arkadiusz Hudzikowski

Wersja

1.0

Data

22.11.2011

2.23.2 Dokumentacja funkcji

2.23.2.1 void **Multimetr** (void)

Funkcja glowna podprogramu multimetru.

Zwraca

none

2.24 Dokumentacja pliku Oscyloskop.c

Plik podprogramu oscyloskopu.

```
#include <avr/io.h>
#include <util/delay.h>
#include <avr/pgmspace.h>
#include <avr/interrupt.h>
#include "Grafika.h"
#include "ADC.h"
#include "Keyboard.h"
#include "lcd132x64.h"
```

Definicje

- `#define Sinc_tab0 1`
- `#define Sinc_tab2 -6`
- `#define Sinc_tab4 38`
- `#define Sinc_tab5 64`
- `#define Sinc_2tab0 0`
- `#define Sinc_2tab2 -1`
- `#define Sinc_2tab3 -4`
- `#define Sinc_2tab4 -5`
- `#define Sinc_2tab6 14`
- `#define Sinc_2tab7 36`
- `#define Sinc_2tab8 56`
- `#define Sinc_2tab9 64`

Funkcje

- **ISR** (DMA_CH1_vect)
- **ISR** (DMA_CH2_vect)
- void [Oscyloskop](#) (void)

Funkcja glowna oscyloskopu.

Zmienne

- uint8_t **kan1_lcd** []
- uint8_t [kan2_lcd](#) []

Bufor przechowujacy probki danych kanalu 2 do wyswietlenia na lcd.

- int16_t **kan1_in** [512]
- int16_t **kan2_in** [512]
- prog_uint16_t [Time_tab](#) [15]

Tablica dzielnikow dla wyboru podstawy czasu.

- static int16_t **ypos2** = 128
- static uint8_t **Vdiv2** = 0
- static uint8_t **Sdiv** = 7
- prog_uint8_t **Gain_tab** [10]

2.24.1 Opis szczegółowy

Plik podprogramu oscyloskopu.

Autor

Arkadiusz Hudzikowski

Wersja

1.0

Data

22.11.2011

2.24.2 Dokumentacja funkcji**2.24.2.1 void Oscyloskop (void)**

Funkcja glowna oscyloskopu.

Zwraca

none

2.24.3 Dokumentacja zmiennych**2.24.3.1 prog_uint8_t Gain_tab[10]****Wartość początkowa:**

```
{
    16,
    20,
    20,
    20,
    25,
    25,
    25,
    63,
    125,
    250}
```

2.25 Dokumentacja pliku Oscyloskop.h

Plik naglowkowy podprogramu oscyloskopu.

Funkcje

- void [Oscyloskop](#) (void)
Funkcja glowna oscyloskopu.

2.25.1 Opis szczegółowy

Plik naglowkowy podprogramu oscyloskopu.

Autor

Arkadiusz Hudzikowski

Wersja

1.0

Data

22.11.2011

2.25.2 Dokumentacja funkcji**2.25.2.1 void Oscyloskop (void)**

Funkcja glowna oscyloskopu.

Zwraca

none

2.26 Dokumentacja pliku TransmisjaPC.c

Plik obsługi transmisji UART.

```
#include <avr/io.h>
#include <util/delay.h>
#include <avr/pgmspace.h>
#include <avr/eeprom.h>
#include "lcd132x64.h"
#include "Grafika.h"
#include "clksys_driver.h"
#include "ADC.h"
#include "DAC.h"
#include "Keyboard.h"
#include "Oscyloskop.h"
#include "Generator.h"
#include "Wobuloskop.h"
#include "Analizator.h"
#include "Ustawienia.h"
#include "usart_driver.h"
```

Definicje

- #define `USE_RS232_MODULE`
- #define `USART` `USARTE0`

Funkcje

- `ISR` (`USARTE0_RXC_vect`)
Przerwanie od UART.
- `uint8_t SetRsSpeed` (`int8_t speed`)
Funkcja ustawiajaca predkosc transmisji.
- `void UARTInit` (`void`)
Funkcja konfigurujaca UART.
- `void UARTU16` (`uint16_t n`)
Funkcja wysylajaca zmienna 16-bitowa przez UART.
- `void UARTC` (`char c`)
Funkcja wysylajaca znak przez UART.
- `void UART_ADCSetGain` (`void`)
Funkcja ustawiajaca wzmacnienie kanalow ADC przez UART.
- `void UART_ADCSetPeriod` (`void`)
Funkcja ustawiajaca podstawe czasu ADC przez UART.
- `void UART_DACSetPeriod` (`void`)
Funkcja ustawiajaca podstawe czasu DAC przez UART.
- `void UART_ADCGetCh1` (`void`)
Funkcja wysylajaca bufor danych ADC przez UART.
- `void UARTSendKan1` (`void`)
Funkcja obslugi kanalu 1 ADC przez UART.
- `void UARTSendKan12` (`void`)
Funkcja obslugi obu kanalow ADC przez UART.
- `void UART_DACGetCh` (`void`)
Funkcja pobierajaca bufor danych DAC przez UART.
- `void UARTGetChOut` (`void`)
Funkcja obslugi DAC przez UART.

- void `ADC_DMA_UART` (void)
Funkcja obsługi kanału 1 ADC przez UART z wykorzystaniem DMA.
- void `TransmisjaPC` (void)
Funkcja główna obsługi transmisji PC.

Zmienne

- int16_t `kan1_in` [512]
- int16_t `kan2_in` [512]
- int16_t `kan_out` [512]
- EEMEM uint8_t `e_rsSpeed`
- prog_uint8_t `rsTab` [10]

2.26.1 Opis szczegółowy

Plik obsługi transmisji UART.

Autor

Arkadiusz Hudzikowski

Wersja

1.0

Data

22.11.2011

2.26.2 Dokumentacja definicji

2.26.2.1 #define USART USARTE0

Wykorzystywany USART

2.26.2.2 #define USE_RS232_MODULE

uzyj podprogramu

2.26.3 Dokumentacja funkcji

2.26.3.1 void ADC_DMA_UART (void)

Funkcja obsługi kanału 1 ADC przez UART z wykorzystaniem DMA.

Zwraca

none

2.26.3.2 ISR (USARTE0_RXC_vect)

Przerwanie od UART.

Tylko do wybudzenia uK

2.26.3.3 uint8_t SetRsSpeed (int8_t speed)

Funkcja ustawiajaca predkosc transmisji.

Parametry

<i>speed</i>	: wartosc zmiany (-9 - 9)
--------------	---------------------------

Zwraca

uint8_t : aktualna predkosc (0 - 10)

2.26.3.4 void TransmisjaPC (void)

Funkcja glowna obslugi transmisji PC.

Zwraca

none

2.26.3.5 void UART_ADCGetCh1 (void)

Funkcja wysylajaca bufor danych ADC przez UART.

Zwraca

none

2.26.3.6 void UART_ADCSetGain (void)

Funkcja ustawiajaca wzmacnienie kanalow ADC przez UART.

Zwraca

none

2.26.3.7 void UART_ADCSetPeriod (void)

Funkcja ustawiająca podstawę czasu ADC przez UART.

Zwraca

none

2.26.3.8 void UART_DACGetCh (void)

Funkcja pobierająca bufor danych DAC przez UART.

Zwraca

none

2.26.3.9 void UART_DACSetPeriod (void)

Funkcja ustawiająca podstawę czasu DAC przez UART.

Zwraca

none

2.26.3.10 void UARTC (char c)

Funkcja wysyłająca znak przez UART.

Parametry

<i>c</i>	: znak do wysłania
----------	--------------------

Zwraca

none

2.26.3.11 void UARTGetChOut (void)

Funkcja obsługi DAC przez UART.

Zwraca

none

2.26.3.12 void UARTInit (void)

Funkcja konfigurująca UART.

Zwraca

none

2.26.3.13 void UARTSendKan1 (void)

Funkcja obsługi kanału 1 ADC przez UART.

Zwraca

none

2.26.3.14 void UARTSendKan12 (void)

Funkcja obsługi obu kanałów ADC przez UART.

Zwraca

none

2.26.3.15 void UARTU16 (uint16_t n)

Funkcja wysyłająca zmienną 16-bitową przez UART.

Kodowanie little endian

Parametry

<i>n</i>	: zmienna 16-bitowa do wysłania
----------	---------------------------------

Zwraca

none

2.26.4 Dokumentacja zmiennych**2.26.4.1 prog_uint8_t rsTab[10]****Wartość początkowa:**

```
{  
    103,  
    51,  
    34,  
    ...  
}
```

```
16,  
246,  
107,  
37,  
24,  
53,  
19}
```

2.27 Dokumentacja pliku TransmisjaPC.h

Plik naglowskiy obsługi transmisji UART.

Funkcje

- uint8_t [SetRsSpeed](#) (int8_t speed)
Funkcja ustawiajaca predkosc transmisji.
- void [UARTInit](#) (void)
Funkcja konfigurujaca UART.
- void [TransmisjaPC](#) (void)
Funkcja glowna obsługi transmisji PC.

2.27.1 Opis szczegółowy

Plik naglowskiy obsługi transmisji UART.

Autor

Arkadiusz Hudzikowski

Wersja

1.0

Data

22.11.2011

2.27.2 Dokumentacja funkcji

2.27.2.1 uint8_t [SetRsSpeed](#) (int8_t speed)

Funkcja ustawiajaca predkosc transmisji.

Parametry

<i>speed</i>	: wartosc zmiany (-9 - 9)
--------------	---------------------------

Zwraca

uint8_t : aktualna predkosc (0 - 10)

2.27.2.2 void TransmisjaPC (void)

Funkcja glowna obslugi transmisji PC.

Zwraca

none

2.27.2.3 void UARTInit (void)

Funkcja konfigurujaca UART.

Zwraca

none

2.28 Dokumentacja pliku usart_driver.c

XMEGA USART driver source file.

```
#include "usart_driver.h"
```

2.28.1 Opis szczególowy

XMEGA USART driver source file. This file contains the function implementations the XMEGA interrupt and polled USART driver.

The driver is not intended for size and/or speed critical code, since most functions are just a few lines of code, and the function call overhead would decrease code performance. The driver is intended for rapid prototyping and documentation purposes for getting started with the XMEGA ADC module.

For size and/or speed critical code, it is recommended to copy the function contents directly into your application instead of making a function call.

Some functions use the following construct: "some_register = ... | (some_parameter ? SOME_BIT_bm : 0) | ..." Although the use of the ternary operator (if ? then : else) is discouraged, in some occasions the operator makes it possible to write pretty clean and neat code. In this driver, the construct is used to set or not set a configuration bit based on a boolean input parameter, such as the "some_parameter" in the example above.

Application note:

AVR1307: Using the XMEGA USART

Documentation

For comprehensive code documentation, supported compilers, compiler settings and supported devices see readme.html

Autor

Atmel Corporation: <http://www.atmel.com>
Support email: avr@atmel.com

Revision:

1694

Date:

2008-07-29 14:21:58 +0200 (ti, 29 jul 2008)

Copyright (c) 2008, Atmel Corporation All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. 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.
3. The name of ATMEL may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY ATMEL "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 EXPRESSLY AND SPECIFICALLY DISCLAIMED. IN NO EVENT SHALL ATMEL 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.

2.29 Dokumentacja pliku `usart_driver.h`

XMEGA USART driver header file.

```
#include "avr_compiler.h"
```

Definicje

- **#define USART_RX_BUFFER_SIZE** 4
- **#define USART_TX_BUFFER_SIZE** 4
- **#define USART_RX_BUFFER_MASK** (USART_RX_BUFFER_SIZE - 1)
- **#define USART_TX_BUFFER_MASK** (USART_TX_BUFFER_SIZE - 1)
- **#define USART_Format_Set**(_usart, _charSize, _parityMode, _twoStopBits)
Macro that sets the USART frame format.
- **#define USART_Baudrate_Set**(_usart, _bseIValue, _bScaleFactor)
Set USART baud rate.
- **#define USART_Rx_Enable**(_usart) ((_usart)->CTRLB |= USART_RXEN_bm)
Enable USART receiver.
- **#define USART_Rx_Disable**(_usart) ((_usart)->CTRLB &= ~USART_RXEN_bm)
Disable USART receiver.
- **#define USART_Tx_Enable**(_usart) ((_usart)->CTRLB |= USART_TXEN_bm)
Enable USART transmitter.
- **#define USART_Tx_Disable**(_usart) ((_usart)->CTRLB &= ~USART_TXEN_bm)
Disable USART transmitter.
- **#define USART_RxdInterruptLevel_Set**(_usart, _rxIntLevel) ((_usart)->CTRLA = ((_usart)->CTRLA & ~USART_RXCINTLVL_gm) | _rxIntLevel)
Set USART RXD interrupt level.
- **#define USART_TxdInterruptLevel_Set**(_usart, _txIntLevel) ((_usart)->CTRLA = ((_usart)->CTRLA & ~USART_TXCINTLVL_gm) | _txIntLevel)
Set USART TXD interrupt level.
- **#define USART_DreInterruptLevel_Set**(_usart, _dreIntLevel) ((_usart)->CTRLA = ((_usart)->CTRLA & ~USART_DREINTLVL_gm) | _dreIntLevel)
Set USART DRE interrupt level.
- **#define USART_SetMode**(_usart, _usartMode) ((_usart)->CTRLC = ((_usart)->CTRLC & (~USART_CMODE_gm)) | _usartMode)
Set the mode the USART run in.
- **#define USART_IsTXDataRegisterEmpty**(_usart) (((_usart)->STATUS & USART_DREIF_bm) != 0)
Check if data register empty flag is set.

- #define `USART_PutChar(_usart, _data)` ((`_usart`)->DATA = `_data`)
Put data (5-8 bit character).
- #define `USART_IsRXComplete(_usart)` (((`_usart`)->STATUS & USART_RXCIF_bm) != 0)
Checks if the RX complete interrupt flag is set.
- #define `USART_GetChar(_usart)` ((`_usart`)->DATA)
Get received data (5-8 bit character).

2.29.1 Opis szczegółowy

XMEGA USART driver header file. This file contains the function prototypes and enumerator definitions for various configuration parameters for the XMEGA USART driver.

The driver is not intended for size and/or speed critical code, since most functions are just a few lines of code, and the function call overhead would decrease code performance. The driver is intended for rapid prototyping and documentation purposes for getting started with the XMEGA ADC module.

For size and/or speed critical code, it is recommended to copy the function contents directly into your application instead of making a function call.

Application note:

AVR1307: Using the XMEGA USART

Documentation

For comprehensive code documentation, supported compilers, compiler settings and supported devices see [readme.html](#)

Autor

Atmel Corporation: <http://www.atmel.com>
Support email: avr@atmel.com

Revision:

1694

Date:

2008-07-29 14:21:58 +0200 (ti, 29 jul 2008)

Copyright (c) 2008, Atmel Corporation All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. 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.
3. The name of ATMEL may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY ATMEL "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 EXPRESSLY AND SPECIFICALLY DISCLAIMED. IN NO EVENT SHALL ATMEL 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.

2.29.2 Dokumentacja definicji

2.29.2.1 #define USART_Baudrate_Set(_usart, _bseIValue, _bScaleFactor)

Wartość:

```
(_usart)->BAUDCTRLA =(uint8_t)_bseIValue;
    \
    (_usart)->BAUDCTRLB =(_bScaleFactor << USART_BSCALE0_bp) | (_bseIValue >> 8
    )
```

Set USART baud rate.

Sets the USART's baud rate register.

UBRR_Value : Value written to UBRR ScaleFactor : Time Base Generator Scale Factor

Equation for calculation of BSEL value in asynchronous normal speed mode: If ScaleFactor ≥ 0 BSEL = $((I/O \text{ clock frequency}) / (2^{(ScaleFactor * 16 * Baudrate)})) - 1$ If ScaleFactor < 0 BSEL = $(1 / (2^{(ScaleFactor * 16)})) * (((I/O \text{ clock frequency}) / Baudrate) - 1)$

Nota

See XMEGA manual for equations for calculation of BSEL value in other modes.

Parametry

<i>_usart</i>	Pointer to the USART module.
<i>_bseIValue</i>	Value to write to BSEL part of Baud control register. Use uint16_t type.
<i>_-</i>	USART baud rate scale factor. Use uint8_t type
<i>bScaleFactor</i>	

2.29.2.2 `#define USART_DreInterruptLevel.Set(_usart, _dreIntLevel) (_usart->CTRLA = ((_usart->CTRLA & ~USART_DREINTLVL_gm) | _dreIntLevel`

Set USART DRE interrupt level.

Sets the interrupt level on Data Register interrupt.

Parametry

<code>_usart</code>	Pointer to the USART module.
<code>_dreIntLevel</code>	Interrupt level of the DRE interrupt. Use USART_DREINTLVL_t type.

2.29.2.3 `#define USART_Format.Set(_usart, _charSize, _parityMode, _twoStopBits)`

Wartość:

```
(_usart)->CTRLC = (uint8_t) _charSize | _parityMode | \
                  (_twoStopBits ? USART_SBMODE_bm : 0)
```

Macro that sets the USART frame format.

Sets the frame format, Frame Size, parity mode and number of stop bits.

Parametry

<code>_usart</code>	Pointer to the USART module
<code>_charSize</code>	The character size. Use USART_CHSIZE_t type.
<code>_parityMode</code>	The parity Mode. Use USART_PMODE_t type.
<code>_twoStopBits</code>	Enable two stop bit mode. Use bool type.

2.29.2.4 `#define USART_GetChar(_usart) ((_usart->DATA)`

Get received data (5-8 bit character).

This macro reads out the RX register. Use the macro USART_RX_Complete to check if anything is received.

Parametry

<code>_usart</code>	The USART module.
---------------------	-------------------

Zwracane wartości

<i>Received</i>	data.
-----------------	-------

2.29.2.5 `#define USART_IsRXComplete(_usart) (((_usart)->STATUS & USART_RXCIF_bm) != 0)`

Checks if the RX complete interrupt flag is set.

Checks if the RX complete interrupt flag is set.

Parametry

<code>_usart</code>	The USART module.
---------------------	-------------------

2.29.2.6 `#define USART_IsTXDataRegisterEmpty(_usart) (((_usart)->STATUS & USART_DREIF_bm) != 0)`

Check if data register empty flag is set.

Parametry

<code>_usart</code>	The USART module.
---------------------	-------------------

2.29.2.7 `#define USART_PutChar(_usart, _data) ((_usart)->DATA = _data)`

Put data (5-8 bit character).

Use the macro `USART_IsTXDataRegisterEmpty` before using this function to put data to the TX register.

Parametry

<code>_usart</code>	The USART module.
<code>_data</code>	The data to send.

2.29.2.8 `#define USART_Rx_Disable(_usart) ((_usart)->CTRLB &= ~USART_RXEN_bm)`

Disable USART receiver.

Parametry

<code>_usart</code>	Pointer to the USART module.
---------------------	------------------------------

2.29.2.9 `#define USART_Rx_Enable(_usart) ((_usart)->CTRLB |= USART_RXEN_bm)`

Enable USART receiver.

Parametry

<code>_usart</code>	Pointer to the USART module
---------------------	-----------------------------

2.29.2.10 `#define USART_RxdInterruptLevel_Set(_usart, _rxdIntLevel) ((_usart)->CTRLA = ((_usart)->CTRLA & ~USART_RXCINTLVL_gm) | _rxdIntLevel)`

Set USART RXD interrupt level.

Sets the interrupt level on RX Complete interrupt.

Parametry

<code>_usart</code>	Pointer to the USART module.
<code>_rxdIntLevel</code>	Interrupt level of the RXD interrupt. Use USART_RXCINTLVL_t type.

2.29.2.11 `#define USART_SetMode(_usart, _usartMode) ((_usart)->CTRLC = ((_usart)->CTRLC & (~USART_CMODE_gm)) | _usartMode)`

Set the mode the USART run in.

Set the mode the USART run in. The default mode is asynchronous mode.

Parametry

<code>_usart</code>	Pointer to the USART module register section.
<code>_usartMode</code>	Selects the USART mode. Use USART_CMODE_t type.

USART modes:

- 0x0 : Asynchronous mode.
- 0x1 : Synchronous mode.
- 0x2 : IrDA mode.
- 0x3 : Master SPI mode.

2.29.2.12 `#define USART_Tx_Disable(_usart) ((_usart)->CTRLB &= ~USART_TXEN_bm)`

Disable USART transmitter.

Parametry

<code>_usart</code>	Pointer to the USART module.
---------------------	------------------------------

2.29.2.13 `#define USART_Tx_Enable(_usart) ((_usart)->CTRLB |= USART_TXEN_bm)`

Enable USART transmitter.

Parametry

<code>_usart</code>	Pointer to the USART module.
---------------------	------------------------------

```
2.29.2.14 #define USART_TxdInterruptLevel_Set( _usart, _txdIntLevel ) (_usart)->CTRLA =
          ((_usart)->CTRLA & ~USART_TXCINTLVL_gm) | _txdIntLevel
```

Set USART TXD interrupt level.

Sets the interrupt level on TX Complete interrupt.

Parametry

<code>_usart</code>	Pointer to the USART module.
<code>_txdIntLevel</code>	Interrupt level of the TXD interrupt. Use USART_TXCINTLVL_t type.

2.30 Dokumentacja pliku Ustawienia.c

Plik funkcji ustawień.

```
#include <avr/io.h>
#include "Keyboard.h"
#include "lcd132x64.h"
#include "Grafika.h"
#include "ADC.h"
#include "Oscyloskop.h"
#include "TransmisjaPC.h"
#include <util/delay.h>
```

Definicje

- #define [MENU_UST_TAB_I](#) 3

Funkcje

- void [PrintMenu_ust](#) (uint8_t menu)
Funkcja wyświetlająca menu wyboru.
- void [Kalibracja](#) (void)
Funkcja ustawień kalibracji.
- void [RS232](#) (void)
Funkcja ustawiająca parametry transmisji UART.
- void [Wyswietlacz](#) (void)
Funkcja ustawiająca parametry wyświetlacza.
- void [Ustawienia](#) (void)

Funkcja glowna ustawien.

Zmienne

- prog_uint8_t `menu_ust_tab` [MENU_UST_TAB_I][11]
- prog_uint8_t `rsSpeedTab` [10][8]
- static uint8_t `dac_offset_cal` = 1

2.30.1 Opis szczegółowy

Plik funkcji ustawień.

Autor

Arkadiusz Hudzikowski

Wersja

1.0

Data

22.11.2011

2.30.2 Dokumentacja definicji

2.30.2.1 `#define MENU_UST_TAB_I 3`

Liczba pozycji w menu ustawień

2.30.3 Dokumentacja funkcji

2.30.3.1 void Kalibracja (void)

Funkcja ustawien kalibracji.

Zwraca

none

2.30.3.2 void PrintMenu_ust (uint8_t menu)

Funkcja wyswietlajaca menu wyboru.

Parametry

<i>menu</i>	: numer wskazywanej pozycji w menu
-------------	------------------------------------

Zwraca

none

2.30.3.3 void RS232 (void)

Funkcja ustawiajaca parametry transmisji UART.

Zwraca

none

2.30.3.4 void Ustawienia (void)

Funkcja glowna ustawien.

Zwraca

none

2.30.3.5 void Wyszwietlacz (void)

Funkcja ustawiajaca parametry wyswietlacza.

Zwraca

none

2.30.4 Dokumentacja zmiennych**2.30.4.1 prog_uint8_t menu_ust_tab[MENU_UST_TAB_I][11]****Wartość początkowa:**

```
{
    "Kalibracja",
    "RS232 - PC",
    "Wyszwietlac"
}
```

Napisy menu ustawień

2.30.4.2 prog_uint8_t rsSpeedTab[10][8]

Wartość początkowa:

```
{
    "19200  ",
    "38400  ",
    "57600  ",
    "115200 ",
    "230400 ",
    "460800 ",
    "921600 ",
    "1152000",
    "1500000",
    "2500000",
}
```

Napisy z predkosciami UART

2.31 Dokumentacja pliku Ustawienia.h

Plik naglowkowy funkcji ustawień.

Funkcje

- void [Ustawienia](#) (void)
Funkcja glowna ustawien.

2.31.1 Opis szczegółowy

Plik naglowkowy funkcji ustawień.

Autor

Arkadiusz Hudzikowski

Wersja

1.0

Data

22.11.2011

2.31.2 Dokumentacja funkcji

2.31.2.1 void Ustawienia (void)

Funkcja glowna ustawien.

Zwraca

none

2.32 Dokumentacja pliku Wobuloskop.c

Plik podprogramu wobuloskopu.

```
#include <avr/io.h>
#include <util/delay.h>
#include <math.h>
#include "ADC.h"
#include "Keyboard.h"
#include "lcd132x64.h"
#include "Grafika.h"
#include "Generator.h"
#include "Analizator.h"
```

Definicje

- #define **F_clk** 2048000000
- #define **F_max** 32000000

Funkcje

- void **Dirac** (void)
Funkcja obliczająca charakterystykę badanego układu pobudzonego impulsem Diraca.
- void **Noise** (void)
Funkcja obliczająca charakterystykę badanego układu pobudzanego szumem białym.
- void **Sweep** (float freqf, float step_freqf)
Funkcja obliczająca charakterystykę badanego układu przemiataniem częstotliwości.
- float **sqr2** (float a)
Funkcja obliczająca pierwiastek kwadratowy liczby zmiennoprzecinkowej.
- void **Wobuloskop** (void)
Funkcja główna wobuloskopu.

Zmienne

- `int16_t kan1_in` [512]
- `int16_t kan2_in` [512]
- `uint16_t kan_out` [512]
- `uint8_t kan1_lcd` [128]
- `uint8_t kan2_lcd` [128]
- `prog_int16_t sin_tab` [640]
Tablica funkcji \sin [0, 2.5PI].
- `uint32_t war` = 0x1

2.32.1 Opis szczegółowy

Plik podprogramu wobuloskopu.

Autor

Arkadiusz Hudzikowski

Wersja

1.0

Data

22.11.2011

2.32.2 Dokumentacja funkcji

2.32.2.1 `void Dirac (void)`

Funkcja obliczająca charakterystykę badanego układu pobudzonego impulsem Diraca.

Wygenerowany zostaje bardzo krótki impuls, następnie zostaje zapisana reakcja badanego układu na to pobudzenie (odpowiedź impulsowa) i na tej podstawie obliczona FFT.

Zwraca

none

2.32.2.2 `void Noise (void)`

Funkcja obliczająca charakterystykę badanego układu pobudzanego szumem białym.

Zostaje wygenerowany szum biały, a następnie zapisywana jest reakcja badanego układu na ten szum i na tej podstawie liczona jest FFT. Na koniec wyniki FFT zostają uśrednione.

Zwraca

none

2.32.2.3 float sqr2 (float a)

Funkcja obliczająca pierwiastek kwadratowy liczby zmiennoprzecinkowej.

Parametry

<i>a</i>	: zmienna, ktorej pierwiastek jest liczony
----------	--------------------------------------------

Zwraca

float : pierwiastek zmiennej a

2.32.2.4 void Sweep (float freqf, float step_freqf)

Funkcja obliczająca charakterystykę badanego układu przemiataniem częstotliwości.

Zostaje wygenerowany sygnał sinusoidalny o zmiennej częstotliwości i stałej amplitudzie. Dla kolejnych częstotliwości mierzona jest amplituda sygnału na wyjściu badanego układu.

Parametry

<i>freqf</i>	: częstotliwość początkowa
<i>step_freqf</i>	: krok zmiany częstotliwości

Zwraca

none

2.32.2.5 void Wobuloskop (void)

Funkcja główna wobuloskopu.

Zwraca

none

2.33 Dokumentacja pliku Wobuloskop.h

Plik nagłówkowy podprogramu wobuloskopu.

Funkcje

- void [Wobuloskop](#) (void)

Funkcja glowna wobuloskopu.

2.33.1 Opis szczegółowy

Plik naglowkowy podprogramu wobuloskopu.

Autor

Arkadiusz Hudzikowski

Wersja

1.0

Data

22.11.2011

2.33.2 Dokumentacja funkcji

2.33.2.1 void Wobuloskop (void)

Funkcja glowna wobuloskopu.

Zwraca

none