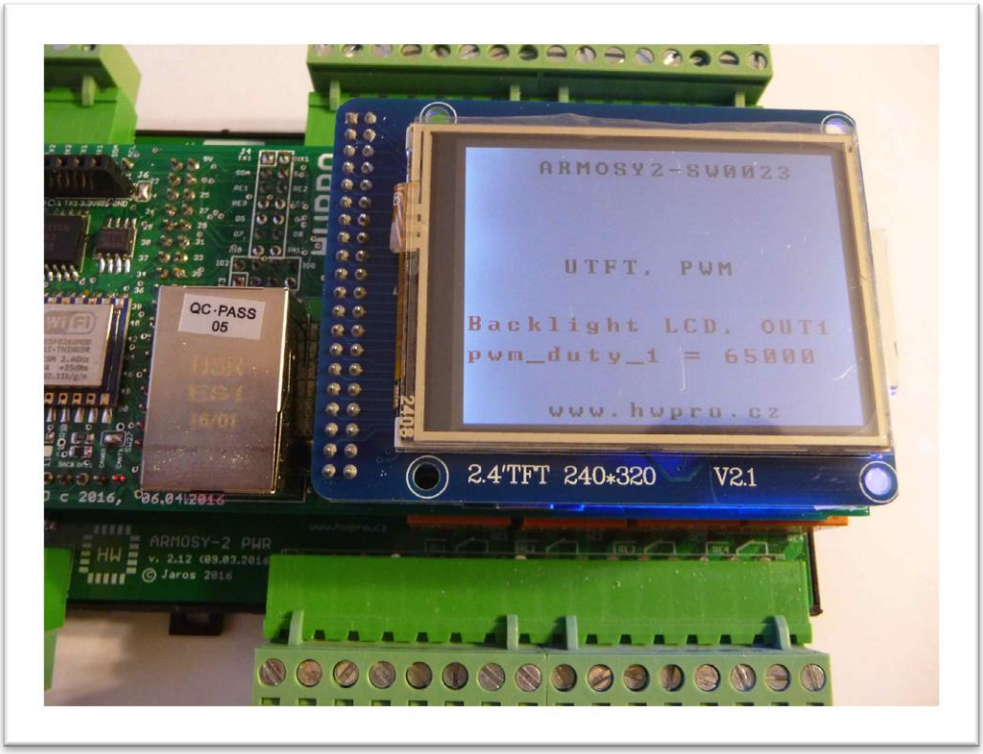
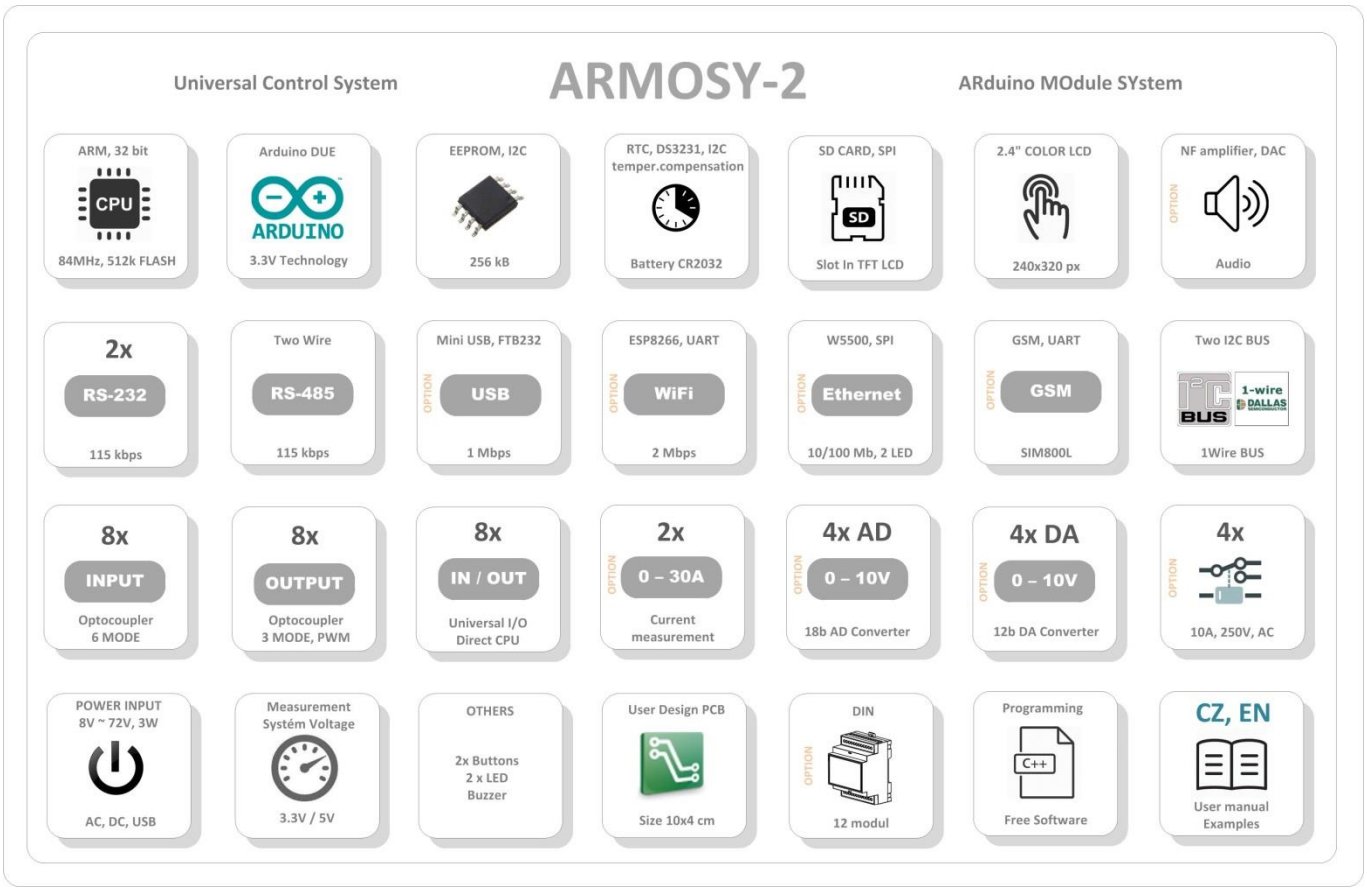
Example – SW0023

PWM 2Hz-40kHz, OUT1-4, Backlight LCD





```
/* ■■■■■■■■■■■■■■ ARMOSY-2 Example ■■■■■■■■■■■■■
Iniciale UTFT 2.4" 240x320px ILI9325D
+ TEXT
+ COLOR
+ PWM
+ Backlight LCD (setings => pwm_freq1, pwm_duty_1)
HW Settings: Backlight Control OUT1 => LCD, ARMOSY SW1 (2,3)= ON

Hardware: ARMOSY-2
Version: HW: 2.21
Create: 18.08.2017 */

// ■ LIBRARY
#include <UTFT.h> // Original Library for UTFT

// ■ DECLARATIONS
UTFT  myGLCD(ITDB24,38,39,40,41); // RS, WR, CS, REST
extern uint8_t SmallFont[]; // Type Font Small
extern uint8_t BigFont[]; // Type Font Big
byte OUT_1 = 6; // HW pin OUT1

// ■■■■■■■■■■■■■■ SETUP ■■■■■■■■■■■■■

void setup()
{

// ■ UTFT
  myGLCD.InitLCD(); // Initialization LCD
  myGLCD.clrScr(); // Clear Screen
  myGLCD.fillScr(VGA_WHITE);  // VGA Background Transparency
  myGLCD.setColor(0, 0, 0); // Fond Black Colors (R, G, B) BLACK
  myGLCD.setBackColor(255, 255, 255); // White Background
  myGLCD.setFont(BigFont); // Settings Size Fonts
  myGLCD.print("UTFT, PWM", 80, 100); // Text, position x, position y
  myGLCD.setColor(255, 0, 0); // Fond Black Colors (R, G, B) RED
  myGLCD.print("Backlight LCD, OUT1", 0, 150); // Text, position x, position y
  myGLCD.setColor(0, 0, 0); // Fond Black Colors (R, G, B) BLACK
  myGLCD.print("ARMOSY2-SW0023", 60, 10);
  myGLCD.print("www.hwpro.cz", CENTER, 220);

// ■ I/O
  pinMode(OUT_1, OUTPUT);

// ■ PWM setup
  pwm_set_resolution(16);               // pwm resolution (16 bit, 8bit)
  uint32_t  pwm_duty_1 = 65000;         // duty (Backlight value 65535 min, 1 max.)
  uint32_t  pwm_freq1 = 100;            // frequency Hz
  pwm_setup( OUT_1, pwm_freq1, 1);      // pwm_setup (pin, freq, iclock 2 ist Clock B)
  pwm_write_duty( OUT_1, pwm_duty_1 ); // pin, duty_cycle_pro_phase
}


// ■■■■■■■■■■■■■■ MAIN ■■■■■■■■■■■■■

void loop()
{
}
```

```
// ■■■■■■■■■■■■■■ PWM LBR ■■■■■■■■■■■■■

/*
Library: pwm01.h (version 01)
Date: 2/11/2013
Written By: randomvibe
Free Download: https://github.com/it-workshop/pwm01.h by randomvibe/blob/master/PWM01/pwm01.cpp

Purpose:
Setup one or two unique PWM frequenices directly, set PWM duty cycle, and stop PWM function.
User Functions: pwm_set_resolution(int res) ~  setup
PWM resolution; up to 16 bit
PWM_setup( uint32_t pwm_pin, uint32_t pwm_freq, int iclock) ~ Setup PWM on clock-A (iclock=1) or clock-B
(iclock-2)
pwm_write_duty( uint32_t pwm_pin, uint32_t pwm_duty) ~ Write PWM duty cycle   pwm_stop( uint32_t pwm_pin) ~
Force PWM stopNotes:
- Applies to Arduino-Due board, PWM pins 6, 7, 8 & 9.
- Libary Does not operate on the TIO pins.
- Unique frequencies set via PWM Clock-A ("CLKA") and Clock-B ("CLKB")
Therefore, up to two unique frequencies allowed.
- Set max duty cycle counts (pwm_max_duty_Ncount) equal to 255 per Arduino approach.
This value is best SUITED for low frequency     applications (2hz to 40,000hz) such as PWM motor drivers,
38khz infrared transmitters, etc.
- Future library versions will address high frequency applications.
- Arduino's "wiring_analog.c" function was very helpful in this effort.      */

#ifdef __arm__
#include "pwm01.h"

int       pwm_resolution_nbit = 16;
uint32_t  pwm_clockA_freq = 0;
uint32_t  pwm_clockB_freq = 0;
uint32_t  pwm_max_duty_Ncount = 255;

void pwm_set_resolution(int res)
{
  pwm_resolution_nbit = res;
}

void pwm_set_clockA_freq(uint32_t  val)
{
  pwm_clockA_freq = val;
}

void pwm_set_clockB_freq(uint32_t  val)
{
  pwm_clockB_freq = val;
}

static inline uint32_t mapResolution(uint32_t value, uint32_t from, uint32_t to)
{
    if (from == to)
        return value;
    if (from > to)
        return value >> (from-to);
    else
        return value << (to-from);
}
```

```
// MAIN PWM INITIALIZATION
//--------------------------------
void  pwm_setup( uint32_t  pwm_pin,  uint32_t  pwm_freq,  int iclock  )
{
    uint32_t  pwm_duty = 0;
    uint32_t  chan = g_APinDescription[pwm_pin].ulPWMChannel;

    // SET CLOCK FREQUENCY
    //------------------------
    if (iclock==1)  pwm_set_clockA_freq( pwm_max_duty_Ncount*pwm_freq );
    if (iclock==2)  pwm_set_clockB_freq( pwm_max_duty_Ncount*pwm_freq );

    if (pwm_pin>=6 && pwm_pin<=9)
    {
        // SET PWM RESOLUTION
        //------------------------
        pwm_duty = mapResolution( pwm_duty, pwm_resolution_nbit, PWM_RESOLUTION);

        // PWM STARTUP AND SETUP CLOCK
        //--------------------------------
        pmc_enable_periph_clk( PWM_INTERFACE_ID );
        PWMC_ConfigureClocks( pwm_clockA_freq, pwm_clockB_freq, VARIANT_MCK );

        // SETUP PWM FOR pwm_pin
        //------------------------
        PIO_Configure(
g_APinDescription[pwm_pin].pPort,  g_APinDescription[pwm_pin].ulPinType,  g_APinDescription[pwm
_pin].ulPin,  g_APinDescription[pwm_pin].ulPinConfiguration);
        if (iclock==1)  PWMC_ConfigureChannel(PWM_INTERFACE, chan, PWM_CMR_CPRE_CLKA, 0, 0);
        if (iclock==2)  PWMC_ConfigureChannel(PWM_INTERFACE, chan, PWM_CMR_CPRE_CLKB, 0, 0);
        PWMC_SetPeriod(PWM_INTERFACE, chan, pwm_max_duty_Ncount);
        PWMC_SetDutyCycle(PWM_INTERFACE, chan, pwm_duty);
        PWMC_EnableChannel(PWM_INTERFACE, chan);
    }
}

// WRITE DUTY CYCLE
//--------------------------------
void  pwm_write_duty( uint32_t  pwm_pin,  uint32_t  pwm_duty )
{
    if (pwm_pin>=6 && pwm_pin<=9)
    {
        pwm_duty = mapResolution( pwm_duty, pwm_resolution_nbit, PWM_RESOLUTION);
        uint32_t  chan = g_APinDescription[pwm_pin].ulPWMChannel;
        PWMC_SetDutyCycle(PWM_INTERFACE, chan, pwm_duty);
    }
};

// FORCE PWM STOP
//--------------------------------
void  pwm_stop( uint32_t  pwm_pin )
{
    pinMode(pwm_pin, OUTPUT);      // sets the digital pin as output
    digitalWrite(pwm_pin, LOW);    // sets the LED off
};

#endif
```

1.

ARMOSY_SW0023_2 | Arduino 1.6.5

Soubor Úpravy Projekt Nástroje Nápověda

ARMOSY_SW0023_2 PWM

```
24 extern uint8_t SmallFont[]; // Type Font Small
25 extern uint8_t BigFont[]; // Type Font Big
26 byte OUT_1 = 6; // HW pin OUT1
27
28 // ■■■■■■■■■■■■■■ SETUP ■■■■■■■■■■■■■
29
30 void setup()
31 {
32
33 // ■ UIFI
34
35   myGLCD.InitLCD(); // Initialization LCD
36   myGLCD.clrScr(); // Clear Screen
37   myGLCD.fillScr(VGA_WHITE);  // VGA Background Transparency
38   myGLCD.setColor(0, 0, 0); // Fond Black Colors (R, G, B) BLACK
39   myGLCD.setBackColor(255, 255, 255); // White Background
40   myGLCD.setFont(BigFont); // Settings Size Fonts
41   myGLCD.print("UIFI, PWM", 80, 100); // Text, position x, position y
42   myGLCD.setColor(255, 0, 0); // Fond Black Colors (R, G, B) RED
43   myGLCD.print("Backlight LCD, OUT1", 0, 150); // Text, position x, position y
44   myGLCD.setColor(0, 0, 0); // Fond Black Colors (R, G, B) BLACK
45   myGLCD.print("ARMOSY2-SW0023", 60, 10);
46   myGLCD.print("www.hwpro.cz", CENTER, 220);
47
48 // ■ I/O
49   pinMode(OUT_1, OUTPUT);
50
51 // ■ PWM setup
52   pwm_set_resolution(16);              // pwm resolution (16 bit, 8bit)
53   uint32_t  pwm_duty_1 = 65000;        // duty (Backlight LCD value 65535 min, 1 max.)
54   uint32_t  pwm_freq1 = 100;           // frequency Hz
55   pwm_setup( OUT_1, pwm_freq1, 1);     // pwm_setup (pin, freq, iclock 2 ist Clock B)
56   pwm_write_duty( OUT_1, pwm_duty_1 ); // pin, duty_cycle_pro_phase
57
58 }
59
60
61 // ■■■■■■■■■■■■■■ MAIN ■■■■■■■■■■■■■
62
63 void loop()
64
65 {
66 }
67
68
```

2.

```
50 void pwm_set_clockB_freq(uint32_t  val)
51 {
```

Jméno nového souboru: PWM OK Storno

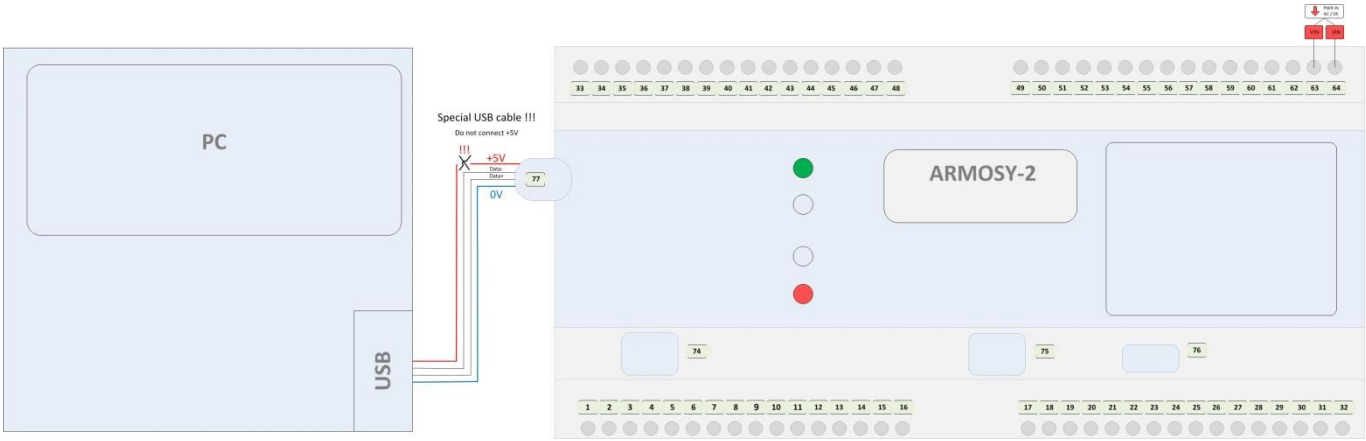
3.

ARMOSY_SW0023_2 | Arduino 1.6.5

Soubor Úpravy Projekt Nástroje Nápověda

ARMOSY_SW0023_2 PWM

```
1  // ■■■■■■■■■■■■■■ PWM LBR ■■■■■■■■■■■■■
2
3  /*
4
5  Library:      pwm01.h (version 01)
6  Date:         2/11/2013
7  Written By:   randomvibe
8  Free Download: https://github.com/it-workshop/pwm01.h_by_randomvibe/blob/master/PWM01/pwm01.cpp
9
10 Purpose:
11    Setup one or two unique PWM frequenices directly in sketch program,
12    set PWM duty cycle, and stop PWM function.
13
14 User Functions:
15    pwm_set_resolution(int res) ~  setup PWM resolution; up to 16 bit
16    pwm_setup( uint32_t pwm_pin, uint32_t pwm_freq, int iclock) ~ Setup PWM on clock-A (iclock=1) or clock-B (iclock-2)
17    pwm_write_duty( uint32_t pwm_pin, uint32_t pwm_duty) ~ Write PWM duty cycle
18    pwm_stop( uint32_t pwm_pin) ~ Force PWM stop
19
20 Notes:
21    - Applies to Arduino-Due board, PWM pins 6, 7, 8 & 9.
22    - Libary Does not operate on the TIO pins.
23    - Unique frequencies set via PWM Clock-A ("CLKA") and Clock-B ("CLKB")
24      Therefore, up to two unique frequencies allowed.
25    - Set max duty cycle counts (pwm_max_duty_Ncount) equal to 255 per Arduino approach.  This value is best SUITED for low frequency
26      applications (2hz to 40,000hz) such as PWM motor drivers, 38khz infrared transmitters, etc.
27    - Future library versions will address high frequency applications.
28    - Arduino's "wiring_analog.c" function was very helpful in this effort.
29
30 */
31
32 #ifdef __arm__
33 #include "pwm01.h"
34
35 int       pwm_resolution_nbit = 16;
36 uint32_t  pwm_clockA_freq = 0;
37 uint32_t  pwm_clockB_freq = 0;
38 uint32_t  pwm_max_duty_Ncount = 255;
39
40 void pwm_set_resolution(int res)
41 {
42   pwm_resolution_nbit = res;
43 }
44
45 void pwm_set_clockA_freq(uint32_t  val)
```



HWPRO
Vývoj a výroba elektronických zařízení
e-mail: info@hwpro.cz
web: www.hwpro.cz