

Teaching online electronics, microcontrollers and programming in Higher Education

Programing of embedded systems

7. Przetwornik A/C

Lead Partner: Warsaw University of Technology

Authors: Daniel Król

University of Applied Sciences in Tarnow



Declaration

This report has been prepared in the context of the ENGINE project. Where other published and unpublished source materials have been used, these have been acknowledged.

Copyright

© Copyright 2021 - 2023 the ENGINE Consortium

Warsaw University of Technology (Poland)

International Hellenic University (IHU) (Greece)

European Lab for Educational Technology- EDUMOTIVA (Greece)

University of Padova (Italy)

University of Applied Sciences in Tarnow (Poland)

All rights reserved.



This document is licensed to the public under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.

Funding Disclaimer

This project has been funded with support from the European Commission. This report reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Programing of embedded systems

7. Przetwornik A/C

I. Wyświetlacz OLED

1. Stwórz nowy projekt dla płyty *LPCXpresso804*. Nazwij projekt np. *ZPSW_Lab07* i dodaj sterowniki *ADC*, *CTIMER* oraz *I2C*:

ct name: 7PSW Lak			
ZF3W_Lat	b07	Project name suffix: Search	
Use default location			
ation: /Users/daniel/D	Documents/MCUXpressoIDE_11.5.0_7232/workspace_LPC55s6		
vice Packages	Board	Project Type	Project Options
LPC804M101JDH24	 Default board files 	C Project O C++ Project	SDK Debug Console 🔵 Semihost 🧿 UAR
LPC804M101JDH20	 Empty board files 	C Static Library C++ Static Libra	CMSIS-Core
LPC804M111JDH24			Import other files
mponents		8	Components selection summary
d or remove SDK softw	vare components		
perating Systems Drive	ers CMSIS Drivers Utilities Middleware Board Componen	nts Abstraction Layer Software Components	
Drivore			Name Description Versio II
Drivers			Name Description Versio In
Drivers type to filter			Name Description Versio In
Drivers type to filter Name	Description	Version Info	Name Description Versio In > © Drivers > © Operating Systems > © Utilities
Drivers type to filter Name	Description ADC Driver	Version Info 2.5.0	Name Description Versio In Description Versio In Description Versio In Description Versio In Utilities
Drivers type to filter Name Mame Mane Compared adc Compared adc	Description ADC Driver CAPT Driver	Version Info 2.5.0 2.1.0	Name Description Versio In > E Drivers E E Utilities
Drivers type to filter Name Image: Contemportation of the second seco	Description ADC Driver CAPT Driver Clock Driver	Version Info 2.5.0 2.3.1	Name Description Versio II > EDrivers > EOperating Systems > EUtilities
Drivers type to filter Name Image: State of the	Description ADC Driver CAPT Driver Clock Driver COMMON Driver	Version Info 2.5.0 2.10 2.3.1 2.3.1	Name Description Versio II > E Drivers > E Operating Systems > E Utilities Utilities >
Drivers type to filter Name	Description ADC Driver CAPT Driver Clock Driver COMMON Driver CTimer Driver	Version Info 2.5.0 2.1.0 2.3.1 2.3.1 2.2.1	Name Description Versio II > ■ Drivers > ■ Operating Systems > ■ Utilities ■
Drivers type to filter Name Image: Image and the second	Description ADC Driver CAPT Driver Clock Driver COMMON Driver CTimer Driver DAC Driver	Version Info 25.0 2.10 2.31 2.31 2.31 2.31 2.31 2.31 2.31 2.31	Name Description Versio II > EDrivers > EOperating Systems > Utilities
brivers type to filter Name	Description ADC Driver CAPT Driver Clock Driver COMMON Driver CTimer Driver DAC Driver GPID Driver 120 Driver	Version Info 2.5.0 2.3.1 2.3.1 2.3.1 2.2.2 2.12 2.12 2.12	Name Description Versio II > ≣ Drivers > ≣ Operating Systems > ≣ Utilities
Drivers type to filter Name Image: Second	Description ADC Driver CAPT Driver Clock Driver COMMON Driver CTimer Driver DAC Driver GPIO Driver I2D Driver I2D Driver	Version Info 250 250 210 231 231 231 221 202 217 210 206	Name Description Versio II > EDrivers > EOpraring Systems > Utilities
brivers type to filter Name	Description ADC Driver CAPT Driver Clock Driver COMMON Driver OTIME Driver GPIQ Driver 12D Driver 14D Driver 14D Driver 14D Driver	Version Info 2.50 2.10 2.31 2.31 2.21 2.02 2.17 2.10 2.02 2.17 2.10 2.03 2.01	Name Description Versio II > Operating Systems > Utilities
brivers type to filter Name	Description ADC Driver CAPT Driver Clock Driver COMMON Driver CTimer Driver DAC Driver GPIO Driver 120 Driver 120 Driver 120 Driver 120 Driver 120 Driver 120 Driver	Version info 2.50 2.10 2.31 2.21 2.10	Name Description Versio II > EDrivers > EOperating Systems > Utilities
Drivers type to filter Name	Description ADC Driver CAPT Driver Clock Driver CTimer Driver CTimer Driver GP(0 Driver GP(0 Driver I2D Driver IAP Driver IAP Driver ICCON Driver LPC_ACOMP Driver CRC Driver	Version Info 250 210 210 2.31 2.31 2.31 2.21 2.02 2.17 210 2.06 2.01 2.01 2.01 2.10 2.01 2.11 2.11	Name Description Versio II > EDrivers > EOprating Systems > Utilities

- 2. Dodaj bibliotekę OLED i skonfiguruj obsługę wyświetlacza jak w poprzedniej instrukcji.
- 3. W Config Tools -> Clocks zmień częstotliwość generatora FRO_OSC na 30 MHz.
- 4. Przejdź do głównego pliku projektu i zmodyfikuj kod jak poniżej:

```
#include <stdio.h>
#include "board.h"
#include "peripherals.h"
#include "pin_mux.h"
#include "clock_config.h"
#include "Lock_config.h"
#include "fsl_debug_console.h"
#include "bl_d.b"
 #include "oled.h'
 char sbuff[32];
  volatile uint16_t adcValue = 0;
  * @brief Application entry point.
 int main(void) {
                    * Init board hardware. */
                  BOARD_InitBootPins();
BOARD_InitBootClocks();
 BOARD_InitBootPeripherals();
#ifndef BOARD_INIT_DEBUG_CONSOLE_PERIPHERAL
                                      de
                        Init
                  BOARD_InitDebugConsole();
 #endif
                      Initialize OLED *
                  OLED_Init(I2C0_PERIPHERAL);
                  while(1) {
                                   OLED_Clear_Screen(0);
sprintf(sbuff, "ADC: %5d", adcValue);
OLED_Puts(0, 1, sbuff);
OLED_Refresh_Gram();
                  }
                  return 0 ;
```

5. Podłącz wyświetlacz i sprawdź jego działanie.



II. Przetwornik A/C

1. Przejdź do Config Tool -> Pins i utwórz nowy preset o nazwie BOARD_InitADCPins:



2. Kliknij w blok *ADC* i podłącz sygnał *ADC0* (wyprowadzenie PIO0_1). Wyłącz domyślny *Pull-Up* ustawiając w polu *Mode* wartość *Inactive*:



3. Przejdź do *Clocks* i włącz sygnał zegarowy *ADC clock* 5 MHz dla przetwornika A/C:



4. Przejdź do ustawień przetwornika ADC i wprowadź poniższą konfigurację:

•••		o workspace_LPC55s69 - ZPSW_Lab07/source/ZPSW_Lab07.c - MCUXpresso IDE	
📑 • 🔜 🌇 🔛 : ZPSW_Lab07 💽 🌴 🛕	💈 Update Code 🔹 Functional Group	* BOARD_IntPeripherals 🔹 💌 🛤 🥥 🖓 🖓 💼 🖬 💷 🚳 🐘 🖷 🕱 🗞 🔅 🚳 - 🏪 🖓 - 💷 🖉 - 🖓 - 🕼	• > 🖻 🔍 👘 🖓 🔿 📰
Components 23 🦞 Peripherals 🙂 🗆	🔁 12C0 🔁 ADC 😫 🔁 CTIMER)	🗆 🌴 Overview 😫 🔂 Code Preview 📟 🗆
tune filter heet	12-bit ADC Controller ()	ADC) (Derinheral rhivers (Devine snarific))	Configuration - General Info
			Configuration - UW Info
Peripheral drivers (Device specific)	Name ADC	Custom name	Processor: LPC804
ADC CTIMERO 1200	Peripheral ADC		Part number: LPC804M101JDH24
	 General configuration 		Core: Cortex-M0P
Custom initialization	✓ Basic ADC configuration		Board: LPCXpressoB04
	Clock mode S	ystem clock - BOARD_BootClockFR018M: 15 MHz, BOARD_BootClockFR024M: 12 MHz, BOARD_BootClockFR030M: 15 MHz	SDK Version: ksdk2_0
	Clock source frequency 1	MHz (BOARD_BootClockFR018M)	> Project
	Clock divider number 0		V Peripherals
	Low power mode		Configures the initialization of the SDK peripheral drivers.
	 Configure threshold setting 	8	(t)
	Threshold values pair 0		
	Low value 0		
	High value 0		V Generated code
	Threshold values pair 1		Update code enabled
	Low value 0		board/peripherals.c
	High value 0		board/peripherals.h
	ADC conversion sequence A	0	✓ Functional groups
	Set high priority for conversion	iequence 🗹	🖶 BOARD_InitPeripherals 🐔
	Hardware trigger	CTIMER0_MAT3	
	Trigger polarity	A positive edge	V Other tools
	Synchronization bypassing		
	Single step mode		(二) (几)
	Interrupt source	Entire sequence	
	ADC conversion sequence B		
	Set high priority for conversion	lequence	
	Hardware trigger	Disabled	
	Trigger polarity	A negative edge	
	Synchronization bypassing		A Problems 23 B Y -
	Single step mode		type filter text
	Interrupt source	Each conversion Y	Level v Resource Issue
	Sampled channels Custom name Cha O CH,	+ X Intel number Channel (threshold pair Threshold interrupt mode Conversion sequence () (5) (5) (Threshold pair) interrupt stabled Sequence A	Warning CTMERR.ctime_match.3 Match output of the match 3 char Warning USART0 Peripheral USART0 is not initialize
		Sequence A interrupt	
	Interrupt sources	Overrun interrupt	
WARNINGS: ZPSW_Lab07: Match output of the match 3 channel of peri	Enable Sequence A interrupt	0	U NXP LPC804* (ZPSW Lab07)
	Interrupt	ADC_SEQA_IRQn	
	Interrupt request	Enabled in initialization	
	Enable priority initialization		
	Priority	0	
	Enable custom handler name	ADC ADC SED & IROHANDI ER	
	Handler template	Copy to clipboard	
	Enable Sequence B interrupt		
	Interrupt	ADC_SEQ8_IRQn v	
	Enable priority initialization	Enabled in initialization	
	Priority	0	
	Enable custom handler name		
	Interrupt handler name		
	Interrupt handler name Handler template		
	Interrupt handler name Handler template Enable Threshold compare inter	Copy to dipboard	
	Interrupt handler name Handler template Enable Threshold compare inter	Cary to rightwork vapt D D D D D D D D D D D D D D D D D D	
	Interrupt handler name Handler template Enable Threshold compare inter Interrupt Interrupt	Copy to clysteer rept ACC_THCM_RIDA	
	Interrupt handler name Handler template Enable Threshold compare inter Interrupt Interrupt request Enable priority initialization	Corp to statement Vet AC_TINEW_RDA Fondated in InStatement	
	Interrupt handler name Handler template Enable Threshold compare inter Interrupt Interrupt request Enable priority initialization Priority	Core to replaced Acc, Tricking, Rich Constrained in obtaination Constrained in obtaination Constrained in obtaination Constrained in obtaination Constrained in obtained	
	Interrupt handler handler Handler template Enable Threshold compare inter Interrupt request Enable priority initialization Priority Enable custom handler name	Core to espaced Section 2010 Control 2010	
	Interrupt handler name Handler template Enable Threshold compare inter Interrupt request Enable priority initialization Priority Enable custom handler name Interrupt handler name	City to replaced	
	Interrupt handler name Handler template Enable Threshold compare inter Interrupt Enable priority initialization Priority Enable custom handler name Interrupt handler name Handler template	Core to replacer Acc, throug, and Core to an advect of the second of t	
	Interrupt handler name Handler template Enable Threshold compare inter Interrupt Interrupt request Enable priority initialization Priority Enable custom handler name Interrupt handler name Interrupt handler name Handler template Enable Overrun error interrupt	Carp to stiploard	
	Interrupt handler name Handler template Enable Threshold compare inter- Interrupt Interrupt Interrupt Interrupt Enable control handler name Handler template Enable Control more interrupt Interrupt	Copy to replayer	
	Interrupt handler name Handler remptate Interrupt request Enable Threshold compare inter- Interrupt request Enable priority initialization Friority Enable custom handler name Interrupt handler name Interrupt Interrupt	Copy to stational	
	Interrupt handler name Handler rempilse Enable Threshold compare inter- Interrupt Interrupt Enable operating initialization Handler name Handler name Handler name merir interrupt Interrupt Interrupt request Enable Operating initialization	Copy to styleowil	
	Interrupt handler name Handler rempila Enable Threshold compare inter- Interrupt Interrupt register Darbe printry initialization Printry Enable custern handler name Handler tempilate Enable Overani error interrupt Interrupt handler Interrupt handler Enable Coverani frei Darbe printry initialization Printry	Corps to stratework	
	Interrupt handler name Handler inneght Interrupt nesent Date prinzip histolication prinzip Exable carton handler name Interrupt handler name Handler template Restand template Meterupt resent Exable Ceremon interrupt Meterupt resent Exable carton handler and Exable ceremon interrupt Meterupt resent	City to Stylevit	
	Interrupt handler name Handler remptate Interrupt regards Exables Threshold compare inter- Interrupt Targards Exable priority initialization Priority Exable Overane arror Interrupt Interrupt Targards Interrupt Targards Exable Overane arror Interrupt Interrupt request Exable Overane arror Interrupt Interrupt request Exable Overane arror Interrupt Interrupt request Exable Overane arror Interrupt Interrupt request Exable Constant function Priority Exable Constant function Priority	Carpe estatement SAC, Trucky, Rich Carbine estatement Carbine estatement Carpe e	

5. Przejdź do *Peripherals*, wybierz *CTIMER* i skonfiguruj go dla zmiany stanu wyjścia z częstotliwością 20 Hz:

•••		e workspace_LPC55s69 - ZPSW_Lab07/source/ZPSW_Lab07.c - MCUXpresso IDE		
📬 • 🔝 🐘 ZPSW_Lab07	💌 🌴 🔺	, 🛛 Update Code 📲 Functional Group 🌯 BOARD_InitPeripherals 💿 🗮 🛤 🗐 🖉 😒 👔 👔 🖬 🔯 👘 🔞 👘 🕼 🖷 3. 🗞 🔅 🚳 - 🍇 - 🦢 🔗 - 💷		ା 🖬 🕹 🕹 🖓 👘 🕴
Components 💱 🦞 Peripherals	- 0	12C0 12C0 12 CTIMERO 23		A Overview 🕅 🔁 Code Preview
type filter text	0	Standard counter/timer (Peripheral drivers (Device specific))		> Configuration - General Info
	<u> </u>			V Configuration - HW Info
Peripheral drivers (Device specific)	0	Name CTIMERO	Custom name	Processor: LPC804
400 CTIMEDO	1200	Mode Input Capture/Match Peripheral CTIMERO	٣	Part number: LPC804M101JDH24
ADC CINNERO	1200	v [≜] Timer/counter general configuration	Preset Custom 🔻	Core: Cortex-MOP
Custom initialization	0	Timer counter configuration		Board: LPCXpresso804
		Timer mode Timer (bus clock source)	T	SDK Version: ksdk2_0
		Bus clock source Svstem clock = R0ARD_RostClockFR018M-15 MHz_R0ARD_RostClockFR024M-12 MHz_R0ARD_RostClockFR030M-15 MHz		> Project
		Clock source frequency 15 MHz (BOARD BootClockE9018MI)		Nr. Deviekerale
		Time Insul frequency/prescaler 1		Configures the initialization of the SDK peripheral drivers
		Calculated prescaler 1		
		Calculated timer input frequency 15 MHz; 66.667 ns		(Ψ)
		Timer counter period 80 ms		
		Start timer in initialization code 🗹		
				V Generated code
		✓ [●] Match channels + X		Update code enabled
		Match_3		board/peripherals.c
		Channel ID Match. 3		board/peripherals.h
		Match channel 3	¥	and Record and an and
		Channel frequency/period/offset 750000		Functional groups
		Channel period (ticks) 750000		BOARD_InitPeripherals
		Calculated match frequency/period/offset 20 Hz; 50 ms		N. Other tools
		Enable counter reset on match		• Other tools
		Enable counter stop on match		
		Output control Toggle bit/output	•	
		Initial output value Low	•	
		Enable match interrupt request		
		✓ Capture channels + X		
		Add item two clicking on plus button		
		Multiple and the second s		
		meeting tente consistent of the second se		A Problems 🗱 🛛 🖪 🍸
		Interrupt		
				Level V Resource Issue
		Interruption Cliniceruption		& Warning CTIMER0.ctimer_match.3 Match output of the match
		Priority 0		S Warning USARTO Peripheral USARTO is not
		v v		
		Callback mode Callback disabled		
WARNINGS: ZPSW_Lab07: Match output of the ma	atch 3 channel of peri	pheral CTIMERO is nos not routed. Match output of the match 3 channel of peripheral CTIMERO is not routed. Writable Smart Insert 81: 32: 28	355	() NXP LPC804* (ZPSW Lab07)

Przetwornik ADC będzie wyzwalany tylko jednym zboczem, dlatego jego częstotliwość próbkowania będzie dwukrotnie niższa - czyli 10 Hz.

6. Przejdź do głównego pliku projektu i zmodyfikuj kod jak poniżej:

```
#include <stdio.h>
#include "board.h"
#include "peripherals.h"
#include "pin_mux.h"
#include "clock_config.h"
#include "LPC804.h"
#include "fsl_debug_console.h"
#include "fsl_power.h"
#include "oled.h"
static adc_result_info_t gAdcResultInfoStruct;
adc_result_info_t *volatile gAdcResultInfoPtr = &gAdcResultInfoStruct;
char sbuff[32];
 volatile uint16_t adcValue = 0;
 /* ADC SEOA IROn interrupt handler
 void ADC_ADC_SEQ_A_IRQHANDLER(void) {
                     Get status flags */
                if (kADC_ConvSeqAInterruptFlag == (kADC_ConvSeqAInterruptFlag & ADC_GetStatusFlags(ADC_PERIPHERAL))) {
    /* Place your interrupt code here */
                                ADC_GetChannelConversionResult(ADC_PERIPHERAL, 0, gAdcResultInfoPtr);
                                adcValue = gAdcResultInfoStruct.result;
/* Clear status flags */
ADC_ClearStatusFlags(ADC_PERIPHERAL, kADC_ConvSeqAInterruptFlag);
                }
 * @brief Application entry point.
int main(void) {
                /* Power on ADC. */
POWER_DisablePD(kPDRUNCFG_PD_ADC0);
                /* Init board hardware.
BOARD_InitBootPins();
                BOARD_InitBootClocks();
BOARD_InitBootPeripherals();
#ifndef BOARD_INIT_DEBUG_CONSOLE_PERIPHERAL
                               SL debug con
                BOARD_InitDebugConsole();
```

 Podłącz potencjometr do płytki, zaprogramuj układ i sprawdź działanie przykładu. Poruszając osią potencjometru, wyświetlana wartość powinna się zmieniać w zakresie 0-4095 (12-bitowa rozdzielczość) co odpowiada napięciu wejściowemu 0-3.3 V.



III. GUI - prosty wskaźnik analogowy

1. Zmodyfikuj kod projektu:

```
#include <stdio.h>
#include "board.h"
#include "peripherals.h"
#include "pin_mux.h"
#include "in_mux.h"
#include "fsl_debug_console.h"
#include "fsl_debug_console.h"
#include "fsl_debug_console.h"
#include "oled.h"
static adc_result_info_t gAdcResultInfoStruct;
adc_result_info_t *volatile gAdcResultInfoPtr = &gAdcResultInfoStruct;
char sbuff[32];
volatile uint16_t adcValue = 0;
float data=0;
void Gauge(uint8_t x0, uint8_t y0, uint8_t radius, float v) {
    float k= (v*270) - 135; // degrees
    float p, q=(2*PI*k)/360.0;
```

```
uint8 t radius0 = radius * 0.9;
             for(int i=-135; i<=135;i+=15) {</pre>
                         p=(2*PI*i)/360.0;
OLED_Draw_Line(x0 + radius0*sinf(p), y0 - radius0*cosf(p), x0 + radius*sinf(p), y0 - radius*cosf(p));
             OLED_Draw_Line(x0, y0 , x0 + radius*sinf(q), y0 - radius*cosf(q));
ļ
/* ADC_SEQA_IRQn interrupt handler */
void ADC_ADC_SEQ_A_IRQHANDLER(void) {
            /* Get status flags */
if (kADC_ConvSeqAInterruptFlag & ADC_GetStatusFlags(ADC_PERIPHERAL))) {
                         /* Place your interrupt code here */
ADC_GetChannelConversionResult(ADC_PERIPHERAL, 0, gAdcResultInfoPtr);
                         adcValue = gAdcResultInfoStruct.result;
/* Clear status flags */
ADC_ClearStatusFlags(ADC_PERIPHERAL, kADC_ConvSeqAInterruptFlag);
            }
 * @brief Application entry point.
int main(void) {
            /* Power on ADC. */
POWER_DisablePD(kPDRUNCFG_PD_ADC0);
            /* Init board hardware.
BOARD InitBootPins();
             BOARD_InitBootClocks();
BOARD_InitBootPeripherals();
#ifndef BOARD_INIT_DEBUG_CONSOLE_PERIPHERAL
                Init
                       FSL debug cons
            BOARD_InitDebugConsole();
#endif
             /* Initialize OLED *
            OLED_Init(I2C0_PERIPHERAL);
            while(1) {
                         OLED_Clear_Screen(0);
                          data=adcValue/4095.0;
                         Gauge(64, 32, 32, data);
sprintf(sbuff, "%3d%%", (uint8_t)(data*100));
OLED_Puts(50, 7, sbuff);
                         OLED Refresh Gram():
             return 0 ;
```

2. Zbuduj projekt w trybie *Release*, zaprogramuj układ i sprawdź działanie przykładu.

IV. Zadania

- 1. Zmodyfikuj wygląd wskaźnika analogowego według własnego uznania.
- 2. Napisz funkcję rysującą *n*-ostatnich próbek w postaci wykresu słupkowego. Wykres ma przesuwać się po ekranie wyświetlacza (poziomu lub pionowo).