

Teaching online electronics, microcontrollers and programming in Higher Education

Programing of embedded systems

3. Układy czasowo-licznikowe

Lead Partner: Warsaw University of Technology

Authors: Daniel Krol

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 Where other published and unpublished source materials have been used, these have been acknowledged.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.

I. Timer systemowy

- 1. Stwórz nowy projekt dla płyty *LPCXpresso804*, tak jak na poprzednich zajęciach i nazwij go *ZPSW_Lab04*.
- 2. Skonfiguruj trzy linie *GPIO* do sterowania diodami RGB. W tym celu, przejdź do *Config Tool -> Open Pins* a następnie z menu *Functional Group* wybierz preset *BOARD_InitLEDsPins* i aktywuj go zaznaczając ikonę flagi po prawo stronie:



- 3. Naciśnij Update Code.
- 4. Przejdź do głównego pliku projektu i zmodyfikuj kod jak poniżej:



while(1) {
}
return 0;

5. Zbuduj projekt, zaprogramuj układ i sprawdź działanie programu.

II. Funkcja "delay"

- 1. Stwórz nowy projekt dla płyty *LPCXpresso804* i nazwij go *ZPSW_Lab04_2*.
- 2. Tak jak poprzednio, skonfiguruj trzy linie GPIO do sterowania diodami RGB.
- 3. Przejdź do głównego pliku projektu i zmodyfikuj kod jak poniżej:

```
#include <stdio.h>
#include "board.h"
#include "board.h"
#include "peripherals.h"
#include "pin_mux.h"
#include "clock_config.h"
#include "LPC804.h"
#include "fsl_debug_console.h"
bool g_pinState = false;
uint32_t g_systickCounter;
void SysTick_Handler(void) {
              if (g_systickCounter) {
                           g_systickCounter--;
              }
void delay_ms(uint32_t n) {
              g_systickCounter = n;
              while (g_systickCounter)
}
 * @brief Application entry point.
int main(void) {
              /* Init board hardware. */
BOARD_InitBootPins();
              BOARD_InitBootClocks();
BOARD InitBootPeripherals();
#ifndef BOARD_INIT_DEBUG_CONSOLE_PERIPHERAL
                  Init FSL debug cons
              BOARD_InitDebugConsole();
#endif
              SysTick_Config(SystemCoreClock / 1000U); // 1 kHz
              while(1) {
                            GPI0_PinWrite(BOARD_INITLEDSPINS_LED_RED_GPI0,
                                                        BOARD_INITLEDSPINS_LED_RED_PORT,
BOARD_INITLEDSPINS_LED_RED_PORT,
g_pinState ^= true);
                            delay_ms(500);
              }
              return 0 ;
```

- 4. Zbuduj projekt, zaprogramuj układ i sprawdź działanie. Dioda LED powinna zmieniać stan 2 razy na sekundę (1 błysk co sekundę).
- 5. Przebuduj projekt w konfiguracji *Relase*, zmieniając ustawienia w rozwijanym menu przy ikonie *Build*:



- 6. Zbuduj projekt, zaprogramuj układ i sprawdź działanie. Ze względu na optymalizację kompilatora, zmienna *g_systickCounter* nie jest "odświeżana" w pętli *while*, znajdującej się wewnątrz funkcji *delay_ms*. Co za tym idzie, dioda LED przestanie błyskać.
- 7. W celu wymuszenia każdorazowego "odświeżenia" wartości zmiennej *g_systickCounter* dodaj modyfikator *volatile*:

```
#include <stdio.h>
#include "board.h"
#include "peripherals.h"
#include "jn_mux.h"
#include "clock_config.h"
#include "LPC804.h"
#include "fsl_debug_console.h"
bool g_pinState = false;
volatile uint32_t g_systickCounter;
 void SysTick_Handler(void) {
              if (g_systickCounter) {
                             g_systickCounter--;
              }
void delay_ms(uint32_t n) {
               g_systickCounter = n;
while (g_systickCounter)
}
  /*
* @brief Application entry point.
 int main(void) {
                /* Init board hardware. */
               BOARD_InitBootPins();
               BOARD_InitBootClocks();
BOARD_InitBootPeripherals();
#ifndef BOARD_INIT_DEBUG_CONSOLE_PERIPHERAL
                                debug con:
               BOARD_InitDebugConsole();
 #endif
               SysTick_Config(SystemCoreClock / 1000U); // 1 kHz
               while(1) {
                             GPI0_PinWrite(BOARD_INITLEDSPINS_LED_RED_GPI0,
                                                          BOARD_INITLEDSPINS_LED_RED_PORT,
BOARD_INITLEDSPINS_LED_RED_PORT,
g_pinState ^= true);
                             delay_ms(500);
               }
               return 0 ;
```

8. Zbuduj projekt, zaprogramuj układ i sprawdź działanie. Dioda led powinna zmieniać stan 2 razy na sekundę (1 błysk co sekundę), jak miało to miejsce w trybie *Debug*.

III. Układ CTIMER - tryb Match

- 1. Stwórz nowy projekt dla płyty *LPCXpresso804* i nazwij go *ZPSW_Lab04_3*.
- 2. Dodaj sterownik *ctimer*:

ect name:	ZPSW_Lab04_3		Project name suffix: Search		
Use defa	ult location				
ation: /	Users/daniel/Documents/MCL	JXpressolDE_11.5.0_7232/workspace_LPC55s69/ZPS	W_Lab04_3		Browse
vice Pac	kages	Board	Project Type	Project Options	
LPC804	M101JDH24	Default board files	O Device t	SDK Debug Console O Semihost	UART
LPC804	M101JDH20	Empty board files	C Statia Library C Statia Library C Statia Library	CMSIS-Core	_
LPC804	M111JDH24		C Static Library C++ Static Library	Copy sources	
				Import other files	
mponen	its		Ē	Components selection summary	F
d or rem	ove SDK software component	15			-
perating !	Systems Drivers CMSIS D	rivers Utilities Middleware Board Components Ab	ostraction Laver) Software Components		
				Name Description	Versio Info
Drivers				> E Drivers	
type to	filter			> E Operating Systems	
Name		Description	Version Info	> Utilities	
			Version mile		
✓ 4	clock	Clock Driver	2.3.1		
	clock common	Clock Driver COMMON Driver	2.3.1 2.3.1		
	clock common	Clock Driver COMMON Driver CTimer Driver	2.3.1 2.3.1 2.2.1		
	clock common ctimer dac	Clock Driver COMMON Driver CTimer Driver DAC Driver	2.3.1 2.3.1 2.2.1 2.0.2		
	clock common ctimer dac gpio	Clock Driver COMMON Driver CTimer Driver DAC Driver GPIO Driver	2.3.1 2.3.1 2.3.1 2.0.2 2.0.2 2.1.7		
	clock common ctimer dac gpio i2c	Clock Driver COMMON Driver CTimer Driver DAC Driver GPIO Driver 12D Driver	2.3.1 2.3.1 2.2.1 2.0.2 2.1.7 2.1.0		
	clock common dac gpio j222	Clock Driver COMMON Driver Crimer Driver DAC Driver GPID Driver I2C Driver IAD Driver	2.3.1 2.3.1 2.2.2 2.0.2 2.1.7 2.1.0 2.0.6		
	clock common dac gpio i2c iap iccon	Clock Driver COMMON Driver DRC Driver GPIO Driver I2C Driver IAD Driver IOCON Driver	2.3.1 2.3.1 2.2.1 2.0.2 2.1.7 2.1.0 2.0.6 2.0.1		
	clock common dac gpio 22c iap icon bocon	Clock Driver COMMON Driver DAC Driver GPIQ Driver IZD Driver IAP Driver IGCON Driver LPC_ACOMP Driver	2.3.1 2.2.1 2.0.2 2.1.7 2.1.0 2.0.6 2.0.1 2.1.0		
	clock common dac gain loc loc loc loc loc_scomp loc_scomp loc_scomp	Clock Driver COMMON Driver DRC Driver GPID Driver I2D Driver I2D Driver IAP Driver IOCON Driver LPC_ACOMP Driver CRC Driver	2.3.1 2.3.1 2.2.1 2.0.2 2.1.7 2.1.0 2.0.6 2.0.1 2.1.0 2.1.1		
	clock common ctimer dac gpio j22 iap locon loc_acomp ipc_acomp mrt	Clock Driver COMMON Driver DRC Driver GPIO Driver I2C Driver I2C Driver IDCON Driver UCCN Driver LPC_ACOMP Driver CRC Driver MRT Driver	2.3.1 2.2.1 2.0.2 2.1.7 2.1.0 2.0.6 2.0.1 2.1.0 2.1.0 2.1.1 2.0.4		
	clock common dac gpio gl2c iap joc_acomp joc_acomp mrt pint	Clock Driver COMMON Driver DAC Driver GPIO Driver I2D Driver IAD Driver IOCON Driver LPC_ACOMP Driver CRC Driver MRT Driver PINT Driver	2.3.1 2.3.1 2.2.1 2.0.2 2.1.7 2.1.0 2.0.6 2.0.1 2.1.1 2.1.1 2.1.1 2.0.4 2.1.9		
	i dock common cinner dec prio prio prio procent p	Clock Driver COMMON Driver DAC Driver GPIO Driver I2D Driver I2D Driver I0CON Driver UPC_ACOMP Driver CRC Driver MRT Driver PINT Driver PINT Driver PUL Driver	2.3.1 2.3.1 2.0.2 2.1.7 2.1.0 2.0.6 2.0.1 2.1.0 2.1.0 2.1.1 2.1.1 2.0.4 2.1.9 2.2.1		
	clock common ctimer papio papio papo	Clock Driver COMMON Driver Offiner Driver GPIO Driver I2D Driver IAD Driver IOCON Driver UCCON Driver UCCON Driver CRC Driver MRT Driver PINT Driver PLU Driver PLU Driver PUD Driver	2.3.1 2.3.1 2.2.1 2.0.2 2.1.7 2.1.0 2.0.6 2.0.1 2.1.0 2.0.6 2.0.1 2.1.0 2.1.1 2.0.4 2.1.1 2.0.4 2.1.9 2.2.1 2.0.0		
	common ctimer dec galo galo b2c bap bcc.com bcc.comp bcc.	Clock Driver COMMON Driver CTImer Driver GPID Driver I2C Driver I2C Driver I2C Driver I2C Driver I2C, ACOMP Driver CRC Driver MRT Driver PINT Driver PLU Driver POWED Driver Reset Driver	2.3.1 2.3.1 2.2.1 2.0.2 2.1.7 2.1.0 2.0.6 2.0.1 2.1.0 2.1.1 2.0.4 2.1.1 2.0.4 2.1.9 2.2.1 2.0.0 2.1.1 2.0.1 2.1.1 2.0.4 2.1.1 2.0.4 2.1.1 2.0.4 2.1.1 2.0.4 2.1.1 2.0.4 2.1.1 2.0.4 2.1.1 2.0.4 2.1.1 2.0.4 2.1.1 2.0.4 2.1.1 2.0.4 2.1.1 2.0.4 2.1.1 2.0.4 2.1.1 2.0.4 2.1.1 2.0.4 2.1.1 2.0.4 2.1.1 2.0.4 2.1.1 2.0.4 2.0.1 2.0.1 2.1.1 2.0.4 2.0.1 2.1.1 2.0.4 2.1.1 2.0.2 2.1.1 2.0.2 2.1.1 2.1.1 2.0.4 2.1.1 2.1.2 2.2 2		

3. Przejdź do *Config Tools -> Peripherals* i dodaj konfigurację sterownika układów *CTIMER*:

000				c workspace	_LPC55s69 - ZPS	W_Lab04_3/source/ZPSW_Lab0	04_3.c - MCUXpresso IDE		
📑 🔹 🐘 📓 ZPSW_Lab04_3	• 希 🔺	Update Code 🔹 Fu	nctional Group	BOARD_InitPeripherals	• 🖻 📑 🤞	/ 🖓 i 💼 📢 💷 iii) 🛍 🔍 i	8 ik 🚳 💁 💁 🖉 • 刘 ·	2) + 5) + 🗫 🛷 🗇 + 🗇 🗾	Q 💷 🗶 🖷 🗘 🗑 🖽
Components 🕄 🦞 Peripherals	- 0	A Overview 🕃 🖻	Code Preview						
type filter text	0	> Configuration	- General Info						
		 Configuration 	- HW Info						
Peripheral drivers (Device specific)	0	Processor: U	PC804						
		Part number:	PC804M101JDH2	14					
Custom initialization	0	Core: C	Cortex-MOP						
		Board: L	PCXpresso804						
		SDK Version: k	sdk2_0						
		> Project							
		 Peripherals 							
			Configures the	initializ 😐 💿	Sel	ect configuration component			
		(🖈)		Palast which someone	ante alta uld ha affarad	Process in the basel shade services			
				Select which compone	and should be offered	Present in the tool-chain project			
		\sim		type filter text					
		✓ Generated cos	de	Configuration component	ent Component descri	otion	Category		
		✓ Update code e	enabled	CTIMER	Standard counter/t	imer	Peripheral drivers (Device s		
		board/peript	herals.c	USART	Universal Synchron	ious/Asynchronous Receiver/Transmitte	r (USART) Peripheral drivers (Device s		
		board/peript	herals.h						
		the Exception of some							
		 Functional gro 	oups						
		BOARD_InitPeriphera							
		 Other tools 							
				Open SDK compor	ients manager	Cancel	OK		
			(
			('UI						
		()							
		🚵 Problems 🕄							B 🍸 🖶 🗖
		type filter text							
		Level ~ Re	source	Issue		Origin	Target	Туре	
		S Warning US	SARTO	Peripheral U	SARTO is not initialized	Pins:BOARD_InitDEBUG_UARTPins	Peripherals: BOARD_InitPeripherals	Validation	
		ta warning 120		Peripheral 12	too is not initialized	PHIS.BUARD_INITIZCHINS	Peripridrais: BUAKD_InitPeripherals	vanuation	
ZPSW_Lab04_3									U NXP LPC804* (ZPSW_Lab04_3)

4. Skonfiguruj układ CTIMER0:

			o workspace_LPC55s69 - ZPSW_Lab04_3/source/ZPSW_Lab04_3.c - MCUXpresso IDE				
📑 • 🔛 🌚 🗟 ZPSW_Lab04_3	• 🔺 🔺	Update Code • Functional Group * BOARD_InitF	eripherals 💽 🗖 🗐 🧳 🖓 💼 📢 😡 🐘 🐘 🐘 🖏 🗞 🔅 – 🂁 🔗 - 🎾	御・御・�� �・	<u>ା ଅ</u> ସ୍ଥା 🗙 🛎 ୩	90	
🎕 Components 😫 🦞 Peripherals	- 0	1 CTIMERO 🕸			A Overview 🔀 🔁 Code Preview		
type filter text	0 1	Standard counter/timer (Peripheral driv	ers (Device specific))	🐪 🜑	> Configuration - General Info		
		Name CTINEDO		Curtom name	V Configuration - HW Info		
Peripheral drivers (Device specific)	0	CIMENO			Processor: LPC804		
t CTIMER0		Mode Input Capture/Match			Part number: LPC804M101JDH24		
		✓ [≜] Timer/counter general configuration		Preset Custom 🔻	Core: Cortex-MOP		
Custom initialization	0	 Timer counter configuration 			Board: LPCXpresso804		
		Timer mode Timer (bi	is clock source)	۳	and reader. Rading of		
		Bus clock source System of	lock - BOARD_BootClockFRO18M: 9 MHz, BOARD_BootClockFRO24M: 12 MHz, BOARD_BootClockFRO30M: 15 MHz	۳	> Project		
		Clock source frequency 9 MHz (E	OARD_BootClockFRO18M)	۳	V Peripherals		
		Timer input frequency/prescaler 1			Configures the initialization of the SDK peripheral drivers.		
		Calculated prescaler 1			()		
		Calculated timer input frequency 9 MHz; 1	11.111 ns				
		Photo Alexandre De la					
		Start umer in initialization code			V Generated code		
		✓ ⁴ Match channels + ×			Violate code enabled		
		[Market al			board/peripherals.c		
		Match_0			board/nerinherals.h		
		Mainterio Ma					
		Chappel frequency/period/offset	ich channel o	•	 Functional groups 		
		Channel neriori (ticks) 45	10000		🗦 BOARD_InitPeripherals 🌇		
		Calculated match frequency/period/offset 2 F	iz; 500 ms				
		Enable counter reset on match			 Other tools 		
		Enable counter stop on match					
		Output control No	action is taken	•			
		Initial output value Low	1	•			
		Enable match interrupt request					
		Capture channels + ×					
		Add item by clicking on plus button					
		 Interrupt and callback settings 			Mar 11 - 77		
		Configuration enabled (a match/capture	(pwm channel interrupt is enabled)		Problems X B	A =	
		✓ Interrupt					
		Interrupt CTIMER0_IRC	2n	¥	Level v Resource Issue		
		Enable priority initialization			Warning CTIMER0.ctimer_match.0 Match output of the m Warning USAPT0 Derinheral USAPT0 is	match C	
		Priority 0			S Warning 12C0 Peripheral 05Ak 10 is	st initiali	
		Callback mode Single callback		٣			
		Shared callback function cbTimer					
		11					
WARNINGS: ZPSW_Lab04_3: Match output of the m	match 0 channel of pr	aripheral CTIMERO is not routed. Match output of the m	atch 0 channel of peripheral CTIMER0 is not routed. Match output of the match 0 channel of peripheral CTIMER0 is n	ot routed.	U NXP LPC804* (ZPSW_Lab04_3)		

- 5. Naciśnij Update Code.
- 6. Przejdź do głównego pliku projektu i zmodyfikuj kod jak poniżej:



- 7. Zbuduj projekt, zaprogramuj układ.
- 8. Uruchom terminal i sprawdź działanie programu w konsoli debuggera.

9. Przejdź do Config Tools -> Peripherals i aktywuj sprzętowe wyjście bloku Match:

		Durate order i Durational Orace			As the state of the			0 : 📫 😪 🗰 nn 👌 r	
Components \$2 Peripherals		CTIMERO 23	www_merengherals			A Overview XX	Code Preview		
type filter text	0 1	Standard counter/timer (PM	ipheral drivers (Device specific)]			> Configur	ation - General Info		
					Configuration - HW Info				
Peripheral drivers (Device specific)	0	Name CTIMERO			Custom name	Processo	r: LPC804		
0.0784/500		Mode Input Capture/Match	Peripheral CTIMER0		¥	Part number: LPC804M101JDH24			
CTIMERU		v ⁴ Timer/counter general configuration	'n		Preset Custom V	Cor	e: Cortex-MOP		
Custom initialization	0	Y Timer counter configuration					Board: LPCXpresso804		
		Timer mode	Timer (but clock source)			SDK Versio	n: ksdk2_0		
		Bus clock source	Sustem clock - ROARD RestClockED01914 0 8	MUX BOARD BootClockED024M 12 MUX BOARD BootClockED020M 15 MUX		> Project			
		Clock source frequency	0 Mila (BOARD Best Clask CD019M)	anz, sono_sonoicer rozam. 12 anz, sono_sonoicer rozam. 10 anz					
		Timer input frequency/orenaler	a white (BORRO_BODICIOCAT ROTOM)			 Peripher 	Configures the initialization	an of the PDV parishers delvers	
		Calculated prescaler	1				Computes the initializatio	an on one oper periprise all uniteds.	
		Calculated timer input frequency	9 MHz; 111.111 ns			(Ψ			
		Timer counter period	500 ms						
		Start timer in initialization code							
		a data da servicio de la construcción de la	~	✓ Generated code					
		Match channels +	×			🗹 Update c	ode enabled		
		Match_0				board/	peripherals.c		
		Channel ID	Match_0			board/	peripherals.h		
		Match channel :	Match channel 0		•	 Function 	al groups		
		Channel frequency/period/offset	4500000			B POART	LinitDariobarais		
		Channel period [ticks] 4600000					D BOOKD_INIT/GRIDINALIS T		
		Calculated match frequency/period/o	ffset 2 Hz; 500 ms			V Other to	dis		
		Enable counter reset on match	<u> </u>			\sim			
		Output control	Fonela bit/output						
		Initial output value	Low.						
		Enable match interrupt request	Low 🛛						
						C 🏠			
		M. Conturn abannala							
		• capture chamiers							
		Add item by clicking on plus button	item by cicking on plus button Interrupt and callback settings						
		 Interrupt and callback settings 						в 7	
		Configuration enabled (a ma	(a match/capture/pwm channel interrupt is enabled)			forme filter text		- 1	
		✓ Interrupt				Land	D	tere of	
		Interrupt C	IMER0_IRQn		•	Level Division	CTIMERO otimos motob 0	ISSUE	
		Enable priority initialization				S Warning	USARTO	Peripheral USARTO is not in	
		Priority 0				😘 Warning	12C0	Peripheral I2C0 is not initial	
		Callback mode Single	callback		٣				
		Shared callback function cbTim	or						

10. Przejdź do *MCUXpresso Config Tools-> Pins* i w *Functional group properties* a następnie stwórz własny preset o nazwie *BOARD_InitCTIMERsPins*:

000				c wo	rkspace_LPC55s69 -	ZPSW_Lab04_3/source	/ZPSW_Lab04_3.c - MCUXpresso IDE				
•	🚡 🔜 🗄 ZPSW_Lat	04_3 💌 🐔 🔒	Update Code 🔹 Fun	tional Group BOARD_InitPins	. 🗎 🗧) ؇ 🖓 🖬 🖬 🕅 🦷	3. 78 A (6 - 🂁 🥔 🖌 🖇	- § - ♥ ♥ ♥ - ⇒ - 🛃		c	እ 📧 🗶 🖲 ጣ 🕴 🔿 🖩
Pins 🕄	🗄 Pins 🗱 🐼 Peripheral Signals					pansion Header		ର୍ ପ 🕁 🖻 🗖 🗖	A Overview 23	Code Preview 🔝 Registers	- 0
880		🕈 💈 🔯 🔍 type filter text							> Configura	tion - General Info	
Pin	Pin name	Label Identifier	Arduino UNO R3 Cor	SPI GPIO USART					V Configura	tion - HW Info	
1	PIO0_18	CN6[1]/CN3[9]/PIO0	CN3[9] (D9)	SPI0:SCK[] GPI0:PI00,18 USARTO					Processor	LPC804	
2	PIO0_16/ACMP_14/A	CN6[2]/CN5[2]/U4[1 CN6[3]/CN5[3]/PIOC	CN5[2] (A1) CN5[3] (A2)	SPI0:SCK[] GPI0:PI00,16 USARTO			-		Part number	LPC804M101JDH24	
4	PIO0_13/ADC_10	\$1/CN8[4]/CN6[4]/D LED_RED;\$1	CN8[4] (D4)	• • •		Functional group pr	operties		Core	Cortex-MOP	
5	PIO0_12	S2/CN8[6]/CN6[5]/E LED_GREEN;S2	CN8[6] (D2)		Name:	BOARD_InitCTIMERsPin	8		Board	LPCXpresso804	
6	RESETN/PIO0_5 PIO0_4/ADC_11	CN6[6]/CN1[10]/S3/ DEBUG_SWD_RES CN6[7]/CN8[7]/CN5 DEBUG_UART_TX	E' CN4[3] (RST) CN8[7] (D1): CN5[5]	Functional groups	3 ^ V	Called by default initi	alization function		SDK Version	ksdk2_0	
8	SWCLK/PIO0_3	CN6[8]/CN1[4]/U1[1 DEBUG_SWD_SWE	X	BOARD_InitPins		Set custom #define p	refix		> Project		
9	SWDIO/PIO0_2	CN6[9]/CN1[2]/U1[1 DEBUG_SWD_SWD	N	BOARD_InitDEBUG_UARTPins	Prefix:						
10	PIO0_11/ADC_6/WK	CN6[10]/CN8[3]/D4/ LED_BLUE	CN8[3] (D5)	BOARD_InitSWD_DEBUGPins		Clock gate enable			 Pins 		
12	PIO0_21/ACMP_15	CN6[12]/CN8[5]/PIO	CN8[5] (D3)	BOARD_Initi2CPins		Full pins initialization				Configures pin routing, includin	g functional electrical pin
13	PI00_20	CN7[12]/CN3[10]/PIC	CN3[10] (D8)	BOARD_InitCTIMERsPins		De-initialization funct	ion			properties, ronage;power rana;	and ren-eme pin comparation.
14	PIO0_15/ADC_8	CN7[11]/CN3[8]/PIO	CN3[8] (D10)		Descriptio	n: Configures pin routing	ind optionally pin electrical features.				
16	PIO0_9/ADC_4	CN7[9]/CN3[6]/R13/	CN3[6] (D12)							🖉 💽 🏦	
17	PIO0_8/ADC_5	CN7[8]/CN3[7]/R20/	CN3[7] (D11)								
√ 18	VDD	CN7[7]/JP5/R37/VDI							Generate	1 code	
20	PIOD 7/ADC 1/ACM	GND FCN7[5]/CN3[2]/JP23[2C_SDA	CN3[2] (D14): CN5[4						V Update co	de enabled	
21	VREFP	CN7[4]/JP7/MCU_VF							board/p	in_mux.c	
22	PIO0_0/ACMP_I1	CN7[3]/CN8[8]/JP2/ DEBUG_UART_RX	CN8[8] (D0)						board/p	in_mux.h	
23	PIOD_19/DACOUT	CN7[2]/CN3[1]/DP4/12C_SCL CN7[1]/CN8[2]/PIO0	CN8[2] (D6)						 Functional 	aroups	
									- Tunction	i groups	
									BOARD	Interns 🔠	
									P BOARD	InitLEDsPins	
									BOARD	InitDEBUG_UARTPins 🏦	
									P BOARD	InitSWD_DEBUGPins	
									E ROARD	Initi2CDing	
							Cance	н ок	P BOARD	InitBUTTONsPins	
									BOARD	InitCTIMERsPins	
-									 Other too 	s	
Routing	Details							⊒ • 8			
Pins Sig	nals Q type filter	text							🐴 Problems 🔀		в 🍸 🖶 🖬
Routing 0	Details for BOARD_Initi	Pins 0 💽 🐼 🔨 🗸									
	Peripheral S	ignal Arrow Routed pin/sig Lab	el Identifier	Direction GPIO initial sta Mo	de Invert	Hysteresis Open drain	DAC Mode		Level	Resource	Issue
									🔈 Warning	CTIMER0.ctimer_match.0	Match output of the match 0 ch
									S Warning	USARTO	Peripheral USART0 is not initiali
									Va warning	1200	rempheral (200 is not initialized
70000 1 - 1 4										1.46	
ZPSW_Labo	4_3									U NXP LPC8	04* (ZPSW_Lab04_3)

- 11. Na rysunku przedstawiającym mikrokontroler kliknij na CTIMER.
- 12. W otwartych oknach dialogowych wybierz odpowiednio *MATCH,0* a następnie *PIO0_11*:



- 13. Naciśnij *Done* w poszczególnych oknach dialogowych a następnie *Update Code*.
- 14. Zbuduj projekt, zaprogramuj układ i sprawdź działanie. Dioda *LED* (niebieska) powinna zmieniać stan 2 razy na sekundę (1 błysk co sekundę).

IV. Układ CTIMER - tryb PWM

1. Przejdź do *Peripherals* i zmień konfigurację *CTIMER0* na *PWM* i ustaw wartości jak poniżej:

	o workspace_LPC55s69 - ZPSW_Lab04_4/source/ZPSW_Lab04_4.c - MCUXpresso IDE		
📑 • 🔛 🌚 😸 ZPSW_Lab04_4 💌 🌴	🔒 📴 Update Code 📲 Functional Group 🛛 🛚 BOARD_InitPeripherals 💎 🌹 🗒 🔗 🗇 👔 🏚 🖗 🐘 🔯 🐘 🔞 🐘 🕲 👘 🚷 🗞 🔅 🖉 📲 🥔 🏈 📲 🍠	8 - 8 - % 4 0 - 4	A 18 X 10 Y 0 H
Components 🕄 🦞 Peripherals 🗢	CTIMER0 23	- 8	🔺 Overview 😫 🔂 Code Preview 🗢 🗆
Sype filter text	Standard counter/timer (Peripheral drivers (Device specific))	🚡 🜑	> Configuration - General Info
Peripheral drivers (Device specific)	Name CTIMERO	Custom name	V Configuration - HW Info
Perpiteral drivers (Device specific)	Mode PWM	¥	Processor: LPC804
CTIMERO	✓ Timer/counter general configuration	Preset Custom	Core: Cortex-M0P
Custom initialization	Y Timer counter configuration		Board: LPCXpresso804
	Timer mode Timer (hus clock source)		SDK Version: ksdk2_0
	Bus clock source System clock - BOARD_BootClockFR018M: 9 MHz, BOARD_BootClockFR024M: 12 MHz, BOARD_BootClockFR030M: 15 MHz		> Project
	Clock source frequency 9 MHz (BOARD_BootClockFR018M)	•	V Peripherals
	Timer input frequency/prescaler 1		Configures the initialization of the SDK peripheral drivers.
	Calculated prescaler 1		(t)
	Calculated timer input frequency 9 MHz; 111.111 ns		
	Start timer in initialization code ✓ PWM channels configuration		
	PWM period channel PWM channel 3		✓ Generated code
	PWM frequency/period 9000		Update code enabled
	PWM period (ticks) 9000		 Deardyperprise as: c
	Calculated www.induencyperiod INTC: Ims Enable PWM period interrupt request Q		Board/peripherals.n
			 Functional groups
	V PWM channels + X		BOARD_InitPeripherals
	PWM_0		✓ Other tools
	Channel ID PWM_0		
	PWM channel PWM channel 0	•	
	PWM duty 0		
	PVMA duty period (ticks) 0 Calculated World duty period 0		
	Enable PWM duty interrupt request		
	✓ Interrupt and callback settings		
	Configuration enabled (a match/capture/pwm channel interrupt is enabled)		
	▼ Interrupt		🏠 Problems 😫 🛛 🔒 🖬
	Interrupt CTMERQ_IRQn	•	
	Priority D		Level v Resource Issue
			Sa Warning USART0 Peripheral USART0 is not initialized
	Canada allask function akTimor		S Warning 1200 Penpheral 1200 is not initialized
	Oracido carindora Lanciana		
7PSW Lab04_4			
(:	: C NXP LPC804 (ZPSW_Laboa_4)

2. Naciśnij Update Code i zmodyfikuj kod programu:

```
#include <stdio.h>
#include "board.h"
#include "portpherals.h"
#include "pin_mux.h"
#include "clock_config.h"
#include "LPC804.h"
#include "fsl_debug_console.h"
volatile uint8_t pwmDuty0=0;
void cbTimer(uint32_t flags) {
               CTIMER_UpdatePwmDutycycle(CTIMER0_PERIPHERAL,
CTIMER0_PWM_PERIOD_CH,
CTIMER0_PWM_0_CHANNEL,
100-pwmDuty0); // LED is active low
}
 * @brief Application entry point.
int main(void) {
               /* Init board hardware. */
BOARD_InitBootPins();
               BOARD_InitBootClocks();
BOARD_InitBootPeripherals();
#ifndef BOARD_INIT_DEBUG_CONSOLE_PERIPHERAL
                              SL debug con
               BOARD_InitDebugConsole();
#endif
               PRINTF("Start\r\n");
               char c;
               while(1) {
                              c=GETCHAR();
                              switch(c) {
```

case 'a':	$if(n_{m}) = 100$
	I (pwmbutyo < 100) (
	pwmDuty0++; }
	<pre>PRINTF("PWM0: %d\r\n", pwmDuty0); break;</pre>
case 'z':	
	if (pwmDuty0 > 0) {
	pwmDuty0;
	<pre>PRINTF("PWM0: %d\r\n", pwmDuty0); break;</pre>
}	
}	
return 0 ;	

- 3. Zbuduj projekt i zaprogramuj układ.
- 4. Uruchom terminal i sprawdź sterowanie jasnością niebieskiej diody LED za pomocą klawiatury.

```
a: Blue PWM ++
z: Blue PWM --
```

V. Zadania

- 1. Skonfiguruj dodatkowe kanały PWM (PWM_1 i PWM_2) dla CTIMER0.
- 2. Podłącz ich wyjścia odpowiednio do PIO0_12 (Green LED) oraz PIO0_13 (Red LED).
- 3. Napisz program sterujący niezależnie jasnością każdej z 3 diod LED za pomocą terminala. Wysyłane znaki:

```
a: Blue PWM ++
z: Blue PWM --
s: Green PWM ++
x: Green PWM --
d: Red PWM ++
c: Red PWM --
```