

Teaching online electronics, microcontrollers and programming in Higher Education

#### **Programing of embedded systems**

2. Konsola debuggera i GPIO

Lead Partner: Warsaw University of Technology

Authors: Daniel Król

University of Applied Sciences in Tarnow



2. Konsola debuggera i GPIO

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 Where other published and unpublished source materials have been used, these have been acknowledged.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

2. Konsola debuggera i GPIO

#### I. Nowy projekt i konsola debuggera

1. Stwórz nowy projekt dla płyty LPCXpresso804:



2. Nazwij projekt np. *Lab01* i pozostaw domyślną konfigurację:

		SDK Wizard					SDK Wizard					
A The source from the SDK will be cop	ied into the workspace. If you want to use linked files, ple	are unzip the "SDK_2.x_LPCXpresse804" SDK.	NP 🖆						N	P 👝		
Configure the project				Advanced	project settings							
Project name: Lab01		Project name suffix:		+ C/C++ Library Se	ttings							
Use default location				Set library type (an	d hosting variant) [ Partilb (robustur	n						
Location: /Users/daniel/Documents/M	CUXpressolD6_11.4.1_6260(workspace_LPC804/Lab01		Browne									
Device Packages	Board	Project Type	Project Optiona	Bedib: Use hoat	ng point version of print?			ewservano: Use hoating ( lewiihNano: Use floating )	point version of print?			
C LPC804M101JDH24	Default board files	C Project C++ Project	SDK Debug Console Semihost O UART	0			0.					
C LPC804M101JDH20	<ul> <li>Empty board files</li> </ul>	C Static Library C++ Static Library	ChSiS-Core	C Hedinect SUK	NeedFault bandler		0 A	warress provide and to 119	n D7			
C LPC804M111JDH24			Import other files	- incruce seminosi				somet principation to the	n i			
Components		B	Components selection summary									
Add or remove SDK software component	erts											
Operating Systems Drivers CMSIS	Drivers Utilities Middleware Board Components Abst	raction Layer Software Components	Name Description Versio Info	+ MCU C Compiler								
Drivers		2 🔆 E E	> E Drivers	I service standard Consiler Atlant								
			Operating Systems     Utilities									
Name	Description	Version Info		<ul> <li>MCU Linker</li> </ul>								
adc	ADC Driver 2.5.0			Unk application t	o RAM							
Capt Clock	Clock Driver	2.1.0		+ Memory Configu	ration							
	COMMON Driver	2.3.0		Memory details								
and ac	CTimer Driver DAC Driver	2.2.1		Default LinkServer	Elash Driver					Browse		
oiag 🔂 gpio	GPID Driver	2.1.7		-								
20	I2C Driver	2.1.0		Flash	PROGRAM FLASH	Flash	Location (http://www.com/com/	5426	LPCR0x 32 rdv	- 1		
	IDCON Driver	20.1		Flash	BOOT_FLASH	Flash2	0x7180	0x80				
bc_acomp	LPC_ACOMP Driver	2.1.0		RAM	SRAM	RAM RAM2	0x1000000	Oxfe0 0x20		<u>%</u>		
🗆 💩 lpe_orc	CRC Driver	2.1.1			10 _010111	TOTAL	41101010	ULU				
- Amrt	MRT Driver	2.0.3		Add Elazh	And DAM		Solk Jain	Dalata				
D Splu	PLU Driver	2.1.8										
🛛 🍓 power_no_Rb	Power Driver	2.0.0		Import	Merge Export Gene							
2		< Back	Next > Cancel Finith	2				< Rack	Next > Cancel	Finish		
U C		< Back	Next > Cancel Firsh	G				< Back	Cancel			

2. Konsola debuggera i GPIO

#### 3. Zostanie wygenerowany szkielet kodu:



Dodaj znacznik "\r" na końcu tekstu w funkcji PRINTF.

4. Podłącz płytę LPCXpresso804 interfejsem USB do komputera:



2. Konsola debuggera i GPIO

5. Kliknij prawym przyciskiem na nazwie projektu, wybierz Properties a następnie przejdź do C/C++ Build -> Settings i wybierz Preprocessor. Sprawdź wartość stałej SDK\_DEBUGCONSOLE, która powinna być ustawiona na 1 (w przeciwnym wypadku - ustaw 1), jak na poniższym obrazku. Stała aktywuje konsolę debuggera, wykorzystującą UART. Pozostałe dwie stałe domyślnie dezaktywują obsługę liczb rzeczywistych (wykorzystywana jest ograniczona wersja biblioteki, która zajmuje mniej pamięci). Aby wyświetlać typy rzeczywiste w konsoli należy usunąć stałą CR\_INTEGER\_PRINTF oraz zmienić wartość stałej PRINTF\_FLOAT\_ENABLE na 1.



6. Zbuduj projekt klikając *Build* a następnie zaprogramuj układ klikając *GUI Flash Tool,* pozostawiając domyślne ustawienia w kolejnych oknach programatora:

•••	c workspace_LPC804 - Lab01/source/Lab01.c - MCUXpresso IDE	
🖆 • 🖩 🕲 • 💊 • 📓 🛷 🏷 🖬 💌 🕨 💷 🗠 3. (% , k) 🗮 🕱	n n n z. T. al (2 N n N 1 / N	
🔁 Project Explorer 🐹 🗱 Registers 🎋 Faults 🧏 Peripherals+ 👘 🗖	🖻 Lab01.c 🕱	
C         Cab01 < Dabup>           >         © Froject Settings           >         % Brazies           >         @Includes           >         @Includes           >         @CMSIS	2# * Copyright 2016-2021 NXP[ 30 31=0** 32 * gfile Lab01.c 33 * gbrief Application entry point. 4 */ 55 finclude "stdio.h> 36 finclude "board.h"	

7. Uruchom terminal:



2. Konsola debuggera i GPIO

8. W ustawieniach wybierz Serial Terminal i prędkość transmisji 9600. W polu serial port wybierz z listy dostępnych portów /dev/ cu.usbmodemXXXXXXXX. Uwaga, w zależności od wersji płyty ewaluacyjnej, w miejscu "X" może pojawić nie inny kod niż na poniższym obrazku:

• • •	Launch Terminal	
Choose termir Settings	al: Serial Terminal	0
Serial port:	/dev/cu.usbmodem020140202	
Baud rate:	9600	•
Data size:	8	0
Parity:	None	•
Stop bits:	1	0
Encoding:	Default (ISO-8859-1)	0
?	Cancel	

9. Naciśnij przycisk *RESET* na płycie *LPCXpresso55s69*. W oknie terminala powinien wyświetlić się tekst wysłany funkcją *PRINTF*:



10. Napisz prosty program "echo" wypisujący w konsoli odebrane znaki, poprzedzone tekstem "Znak: ". W tym celu zmodyfikuj kod w funkcji main jak poniżej:



Zbuduj projekt, zaprogramuj układ i sprawdź działanie programu w konsoli terminala.

2. Konsola debuggera i GPIO

#### II. Sterowanie diodami LED

- 1. Stwórz nowy projekt dla płyty *LPCXpresso804* i nazwij go np. *Lab01\_2*.
- Należy skonfigurować 3 linie GPIO do sterowania poszczególnymi diodami RGB diody. W tym celu kliknij prawym przyciskiem na nazwie projektu i wybierz MCUXpresso Config Tools -> Open Pins, jak na poniższym obrazku:



3. W oknie widoczne są automatycznie skonfigurowane linie TXD i RXD interfejsu UART (USARTO), wykorzystywanego przez konsolę debuggera. Z menu Functional Group wybierz preset BOARD\_InitLEDsPins, a następnie aktywuj go zaznaczając ikonę flagi po lewej stronie:

000							ć	workspace_	LPC804	Lab01_2	source,	Lab01.c	- MCUXp	oresso II	DE								
1 📬 • 🔛	🕼 🔝 🚺 Lab01	_2	🔹 🏀 🚺	💈 Update Code 🔸 🛛 Fu	nctional Grou	P BOARD_	InitLEDsPins	× 🗖	•	> 🖬 🕯	> 10 Hi	3. 8. 1	10-	<b>4</b> -]ø	1.		- 🏷 🛷 🔶	• - 🖻			Q	. 🖻 🗶	i u i O ⊞
🗄 Pins 3	8 🐼 Peripheral S	ignals				BOARD_ BOARD_	InitPins InitLEDsPins										Q Q (* )	. 📼 🗝 🗖	🔺 Overview 😂	Code Preview	Registers		- 0
990	]₩₩•	+ 🖄 Q t				BOARD_ BOARD	InitDEBUG_UARTPin InitSWD_DEBUGPin	5											> Configur	ation - General Info			
Pin	Pin name	Label	Identifier	Arduino UNO R3 Co	e SPI	G BOARD_	Initi2CPins												Y Configur	ation - HW Info			
1	PICO_18	CN6[1]/CN3[9]/PI		CN3[9] (D9)	PIO0_18[	P DOWND	Internation		J										Processo	D 1 PC804			
3	PICO_17/ADC_	<ul> <li>CN6[3]/CN5[3]/PI</li> </ul>		CN5[2] (A1) CN5[3] (A2)	PI00_17[	PI00_17	PIO0_17[.			5									Part numbe	C LPCR04M101JDH2	4		
4	PICO_13	\$1/CN8[4]/CN8[4	LED_RED;S1	CN8[4] (D4)	PIO0_13[	PI00_13	PIO0_13[.			2				0 0	ð	2			Con	<ul> <li>Cortex-MOP</li> </ul>			
5	PI00_12	\$2/CN8[6]/CN6[5	LED_GREEN;S2	CN8[6] (D2)	P100_12[	PI00_12	PIO0_12[.			5 4				QV QV	Q.	2/W			Roon	i i novembr			
6	RESETN/PICO_	5 CN6[6]/CN1[10]/.	. DEBUG_SWD_RE	CN4[3] (RST)	P100_5[]	PI00_5	PIO0_5[			100	000	20.7		0.9	0_1	00.2			SDK Version	krdk2.0			
8	SWCLK/PIOD_S	CN6[7][CN6[7][C	DEBUG_OART_TX	CN8[7] (D1); CN5	PIO0_4[]	PID0_4	PIDO_4[			2 2	2 2	Pier Pier	2 9	8 8	×.	8 8			SOR TETAD	. ABUKA_U			
9	SWDI0/P100_2	CN6[9]/CN1[2]/U	DEBUG_SWD_SW.		P100_2[]	PI00_2	PID0_2[												> Project				
10	PI00_11	CN6[10]/CN8[3]/.	. LED_BLUE	CN8[3] (D5)	PIO0_11[	PI00_11	PID0_11[.																
11	PIO0_10/ADC_	7 CN6[11]/VR1/CN.		CN5[1] (A0)	PIO0_10[	] PIO0_10	PID0_10[.												<ul> <li>Pins</li> </ul>				
12	PI00_21/ACMF	_15 CN6[12]/CN8[6]/.		CN8[5] (D3)	PIO0_21[	] PIO0_21	PIO0_21[.												$\sim$	Configures pin ro	uting, including	functional electris	cal pin
13	PICO_20	CN7[12]/CN3[10]		CN3[10] (D8)	PIO0_20[	000_20	PIO0_20[. PIO0_15[		- 40	MP	1000	CAPT	СТМ	FR0 0	1400	680				propercies, vorca	reponerrais, a	to ran-ane pri co	ingulation.
15	PIC0_1/ADC_0	A CN7[10]/CN3[5]/.		CN3[5] (D13); CN.	PI00_1[]	PI00_1	PIO0_1[		120	0 1	2C1	LVLSHFT	PINT	P	าม	SP10							
16	PICO_9/ADC_4	CN7[9]/CN3[6]/R.		CN3[6] (D12)	P100_9[]	PI00_9	PIO0_9[		SU	PPLY :	OWD	SYSCON	USAR	TO U	ISART1	WKT							
17	PIC0_8/ADC_5	CN7[8]/CN3[7]/R.		CN3[7] (D11)	P100_8[]	PI00_8	PIO0_8[																
18	VDD	CN7[7]/JP5/R37/.	-																✓ Generate	d code			
20	PIO0 7/40C 1	CN7151/CN3121/1	12C SD4	CN3[2] (D14): CN	PIO0 7/ 1	PI00 7	PIO0 71												✓ Update o	ode enabled			
21	VREFP	CN7[4]/JP7/MCU																	B board	nin mux c			
22	PIO0_0/ACMP_	1 CN7[3]/CN8[8]/U.	DEBUG_UART_RX	CN8[8] (D0)	PIO0_0[]	PI00_0	PIO0_0[			LPC80	04M10	IJDH24	7 - 155	OPA 2	4 pack	cage							
23	PIO0_14/ACMF	L. CN7[2]/CN3[1]/U.	I2C_SCL	CN3[1] (D15)	PIO0_14[	] PIO0_14	PID0_14[.		_	_									board/	pin_mux.h			
										P100_16/ACMP_14/ADC_3	P100_17/ADC_9	PIO0_12	P100_4	SWDIO	Pi00_11	PIO0_10/ADC_7 PIO0_21/ACMP_JS			Punction     Punction     BOARC     BOARC     PBOARC     PBOA	al groups UnitPins UnitLEDSPins UnitDEBUG_UARTPins UnitSWD_DEBUGPins	8		
																				InitBUTTONsPins			
Routing	g Details																	=	N. Other Inc.				
Pins Si	gnals Q, type f																		• Oundr too	-			
Posting	Datails for BOARD	M 3																					
# 5	eripheral Sion	al Arrow Route	d pin/signal Label			Identifier	Direction G	PIO initial state	Mode	Invert	H	steresis	Open dra	ain PA	C Mode								
10 0	PIO PIO	), 11 -> [10] P	100_11 CN6[10]/	CN8[3]/D4/PIO0_11/M	/_PIO0_11	LED_BLUE	Output Lo	gical 1	Inactive	Disable	d En	abled	Disabled	n/4	1				A Problems 23				B 🕇 😐 🗆
5 0	PIO PIO	), 12 -> [5] Pl	00_12 S2/CN8[6	]/CN6[5]/D3/M_PIO0	12/PIO0_12	LED_GREEN	Output Lo	igical 1	Inactive	Disable	d En	abled	Disabled	n/a					type filter text				
4 0	PIO PIO	), 13 -> [4] PI	J0_13 51/CN8[4	I/CN6[4]/D2/M_PIOU	_13/PI00_13	LED_RED	Output Lo	igical 1	inactive	Disable	0 EF	abied	Disabled	n/a	,				Level	Resource		Issue	
																			🔈 Warning	USARTO		Peripheral US	ARTO is not initiali
																			🔈 Warning	12C0		Peripheral 120	0 is not initialized
Lab01_2																				I	D NXP LPC8	04* (Lab01 2)	

2. Konsola debuggera i GPIO

4. Aktywne presety można sprawdzić klikając ikonkę *Functional group properties*. W otwartym oknie widoczna jest lista presetów. Można je edytować oraz dodawać własne:



5. Wybierz *Update Code* (powyższy rysunek) w celu wygenerowania kodu na podstawie wprowadzonej konfiguracji. Kod zostanie dodany do plików oznaczonych ikoną "*change*":



Przez klikniecie w *change* można zobaczyć jakie zmiany zostaną wprowadzone w poszczególnych plikach z kodem źródłowym.

2. Konsola debuggera i GPIO

- 6. Zaakceptuj zmiany przyciskiem OK.
- 7. Stałe opisujące poszczególne linie, sterujące diodą *RGB*, zostały wygenerowane w pliku *board/pin\_mux.h*:

	尾 Lab01.c	🚡 pin_mux.h 🛿				- 8			
	28 /*! 29 * @br 30 * 31 */ 32 <b>void B</b>	ief Configures OARD_InitPins(	pin routing and	optionally pin electric	al features. ex-MOP */				
	34 #define IOCNN_PIO_MYS_EN 0x20u /#:⊲0prief Enable hysteresis */ 35 #define IOCNN_PIO_MYD 10x00u /*!⊲0prief Input not invert */ 37 #define IOCNN_PIO_MODE_INACT 0x00u /*!⊲0prief Disables Open-drain function */ 38								
	39⊜ /*! @name PI00_11 (number 10), CN6[10]/CN8[3]/D4/PI00_11/M_PI00_11 40 @{ */ 41								
	42 /* Sym 43 #defin 44 #defin 45 #defin 46 #defin 47 #defin 48	bols to be use e BOARD_INITLE e BOARD_INITLE e BOARD_INITLE e BOARD_INITLE e BOARD_INITLE	d with GPIO drive DSPINS_LED_BLUE_( DSPINS_LED_BLUE_( DSPINS_LED_BLUE_1 DSPINS_LED_BLUE_1 DSPINS_LED_BLUE_1	PT */ SPIO_GPIO SPIO_PIN_MASK (1U << 110 PORT 0U PIN 11U PIN_MASK (1U << 11U)	<pre>/*!&lt;@brief GPIO peripheral base )/*!&lt;@brief GPIO pin mask */ /*!&lt;@brief PORT device index: 0 /*!&lt;@brief PORT pin number */ /*!&lt;@brief PORT pin mask */ /* @) */</pre>	pointer */ */			
49 500 /*! @name PIO0_12 (number 5), 52/CN8[6]/CN6[5]/D3/M_PIO0_12/PIO0_12 51 @{ */									
	53 /* Sym 54 #defin 55 #defin 56 #defin 57 #defin 58 #defin 59	bols to be use e BOARD_INITLE e BOARD_INITLE e BOARD_INITLE e BOARD_INITLE e BOARD_INITLE	d with GPIO drive DSPINS_LED_GREEN DSPINS_LED_GREEN DSPINS_LED_GREEN DSPINS_LED_GREEN DSPINS_LED_GREEN	er */ GPIO GPIO GPIO_PIN_MASK (1U << 12 PORT 0U PIN 12U PIN_MASK (1U << 12U)	/*!<@brief GPIO peripheral base U) /#!<@brief GPIO pin mask */ /*!<@brief PORT device index: 0 /*!<@brief PORT pin number */ /*!@brief PORT pin mask */ /* @} */	pointer */ */			
	61⊖/*!@n 62 @{*	ame PIO0_13 (n /	umber 4), S1/CN8	<pre>[4]/CN6[4]/D2/M_PI00_13/</pre>	PI00_13				
	64 /* Sym 65 #defin 66 #defin 67 #defin 68 #defin 69 #defin 70 71	bols to be use e BOARD_INITLE e BOARD_INITLE e BOARD_INITLE e BOARD_INITLE e BOARD_INITLE	d with GPI0 driv DSPINS_LED_RED_G DSPINS_LED_RED_G DSPINS_LED_RED_P DSPINS_LED_RED_P DSPINS_LED_RED_P DSPINS_LED_RED_P	er */ PIO GPIO PIO_PIN_MASK (1U << 13U) RT AU IN 13U IN_MASK (1U << 13U)	/*I<@brief GPIO peripheral base p /*I<@brief GPIO pin mask */ /*I<@brief PORT device index: 0 * /*I<@brief PORT pin number */ /*I<@brief PORT pin mask */ /* @} */	⊃inter */ /			
	72 /*! 73 *@br 74 * 75 */	ief Configures	pin routing and	optionally pin electric	al features.				
	76 void B	OARD_InitLEDsP	<pre>ins(void); /* Fun</pre>	nction assigned for the	Cortex-M0P */				

8. Zmodyfikuj kod w funkcji *main*, tak aby wysłanie znaku "a" powodowało zaświecenie diody *LED* na czerwono. Z kolei, wysłanie znaku "z" powinno zgasić składową czerwoną:



2. Konsola debuggera i GPIO

Zbuduj projekt, zaprogramuj układ i sprawdź działanie programu.

#### III. Obsługa przycisków

- 1. Stwórz nowy projekt dla płyty LPCXpresso804 i nazwij go np. Lab01\_3.
- Otwórz ponownie narzędzie do konfiguracji wyprowadzeń: MCUXpresso Config Tools -> Open Pins. Ponieważ na płytce prototypowej diody LED i przyciski współdzieli wyprowadzenie mikrokontrolera, odznacz preset BOARD\_InitLEDsPins i zaznacz preset BOARD\_InitBUTTONsPins a następnie kliknij Update Code.
- 3. Zmodyfikuj kod w funkcji main, tak aby wciśnięcie przycisku S1 wypisywało w konsoli komunikat "S1" natomiast wciśnięcie przycisku S2 ma wypisać komunikat "S2":

```
#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"
 /* TODO: insert other include files here. */
 /* TODO: insert other definitions and declarations here. */
 * @brief Application entry point.
 int main(void) {
              /* Init board hardware. */
BOARD_InitBootPins();
             BOARD_InitBootClocks();
BOARD_InitBootPeripherals();
 #ifndef BOARD_INIT_DEBUG_CONSOLE_PERIPHERAL
                           SL debug cons
             BOARD_InitDebugConsole();
 #endif
             PRINTF("Start\r\n");
             while(1) {
if(GPI0_PinRead(BOARD_INITBUTTONSPINS_S1_GPI0, BOARD_INITBUTTONSPINS_S1_PORT,
BOARD_INITBUTTONSPINS_S1_PIN) == 0)
                           PRINTF("S1\r\n");
                       if(GPI0_PinRead(BOARD_INITBUTTONSPINS_S2_GPI0, BOARD_INITBUTTONSPINS_S2_PORT,
 BOARD_INITBUTTONSPINS_S2_PIN) == 0)
                           PRINTF("S2\r\n");
             }
              return 0 ;
```

Zbuduj projekt, zaprogramuj układ i sprawdź działanie programu.

4. Zmodyfikuj kod w funkcji main, tak aby wciśnięcie przycisku *S1* oraz S2 wypisywało w konsoli odpowiedni komunikat tylko raz (detekcja zbocza opadającego):



2. Konsola debuggera i GPIO

```
bool sw1=false, tm1=false;
           bool sw2=false, tm2=false;
           /* Init board hardware. */
           BOARD_InitBootPins();
BOARD_InitBootClocks();
BOARD_InitBootPeripherals();
#ifndef BOARD_INIT_DEBUG_CONSOLE_PERIPHERAL
                         debug cons
           BOARD_InitDebugConsole();
#endif
           PRINTF("Start\r\n");
           while(1) {
                       tml=sw1;
sw1 = GPI0_PinRead(BOARD_INITBUTTONSPINS_S1_GPI0, BOARD_INITBUTTONSPINS_S1_PORT,
______
BOARD_INITBUTTONSPINS_S1_PIN);
tm2=sw2;
sw2 = GPI0_PinRead(BOARD_INITBUTTONSPINS_S2_GPI0, BOARD_INITBUTTONSPINS_S2_PORT,
BOARD_INITBUTTONSPINS_S2_PIN);
                       if(sw1 < tm1) {
                          PRINTF("S1\r\n");
                       }
                       if(sw2 < tm2) {
                          PRINTF("S2\r\n");
                       }
           return 0 :
```

Zbuduj projekt, zaprogramuj układ i sprawdź działanie programu.

#### **IV. Exercises**

- 1. Zmodyfikuj program sterowania LED tak, aby możliwe było sterowanie trzema diodami RGB. Wysyłając znak:
  - a: Red-On
  - z: Red-Off
  - s: Green-On
  - x: Green-Off
  - d: Blue-On
  - c: Blue-Off
- 2. Napisz ten sam program, używając instrukcji switch-case.
- 3. Zmodyfikuj program obsługi przycisków, aby można było wykryć zwolnienie przycisku.