**Listing 1. Plik cpx.c – Sterowanie czterech diod RGB**

/\*

KA-NUCLEO-MULTISENSOR LED mpx demo

gbm 12'2017

\*/

#include "stm32l4yy.h"

#include "ka\_nuc\_multis.h"

#define SYSCLK\_FREQ 80000000u

#define CPX\_FREQ 1600u

#define CPXROWS 4

int main**(**void**)**

**{**

RCC**->**AHB2ENR **|=** RCC\_AHB2ENR\_GPIOBEN**;**

// Clock setup: PLL - 80 MHz

RCC**->**PLLCFGR **=** RCC\_PLLCFGR\_PLLREN **|** RCC\_PLLCFGR\_PLLNV**(**40**)**

**|** RCC\_PLLCFGR\_PLLMV**(**1**)** **|** RCC\_PLLCFGR\_PLLSRC\_MSI**;**

RCC**->**CR **|=** RCC\_CR\_PLLON**;**

// set Flash speed

FLASH**->**ACR **|=** FLASH\_ACR\_PRFTEN **|** FLASH\_ACR\_LATENCY\_4WS**;**

**while** **(!(**RCC**->**CR **&** RCC\_CR\_PLLRDY**));**

RCC**->**CFGR **|=** RCC\_CFGR\_SW\_PLL**;**

SysTick\_Config**(**SYSCLK\_FREQ **/** CPX\_FREQ**);**

SCB**->**SCR **=** SCB\_SCR\_SLEEPONEXIT\_Msk**;** // sleep while not in handler

\_\_WFI**();** // go to sleep

**}**

// CPX MODER OR masks for 12 LEDs

static const uint32\_t cpxmask**[]** **=**

**{**

BF2**(**CPX\_LED3\_BIT**,** GPIO\_MODER\_OUT**)** **|** BF2**(**CPX\_LED0\_BIT**,** GPIO\_MODER\_OUT**),**// right R

BF2**(**CPX\_LED3\_BIT**,** GPIO\_MODER\_OUT**)** **|** BF2**(**CPX\_LED1\_BIT**,** GPIO\_MODER\_OUT**),**

BF2**(**CPX\_LED3\_BIT**,** GPIO\_MODER\_OUT**)** **|** BF2**(**CPX\_LED2\_BIT**,** GPIO\_MODER\_OUT**),**

BF2**(**CPX\_LED2\_BIT**,** GPIO\_MODER\_OUT**)** **|** BF2**(**CPX\_LED3\_BIT**,** GPIO\_MODER\_OUT**),**

BF2**(**CPX\_LED2\_BIT**,** GPIO\_MODER\_OUT**)** **|** BF2**(**CPX\_LED0\_BIT**,** GPIO\_MODER\_OUT**),**

BF2**(**CPX\_LED2\_BIT**,** GPIO\_MODER\_OUT**)** **|** BF2**(**CPX\_LED1\_BIT**,** GPIO\_MODER\_OUT**),**

BF2**(**CPX\_LED1\_BIT**,** GPIO\_MODER\_OUT**)** **|** BF2**(**CPX\_LED2\_BIT**,** GPIO\_MODER\_OUT**),**

BF2**(**CPX\_LED1\_BIT**,** GPIO\_MODER\_OUT**)** **|** BF2**(**CPX\_LED3\_BIT**,** GPIO\_MODER\_OUT**),**

BF2**(**CPX\_LED1\_BIT**,** GPIO\_MODER\_OUT**)** **|** BF2**(**CPX\_LED0\_BIT**,** GPIO\_MODER\_OUT**),**

BF2**(**CPX\_LED0\_BIT**,** GPIO\_MODER\_OUT**)** **|** BF2**(**CPX\_LED1\_BIT**,** GPIO\_MODER\_OUT**),**

BF2**(**CPX\_LED0\_BIT**,** GPIO\_MODER\_OUT**)** **|** BF2**(**CPX\_LED2\_BIT**,** GPIO\_MODER\_OUT**),**

BF2**(**CPX\_LED0\_BIT**,** GPIO\_MODER\_OUT**)** **|** BF2**(**CPX\_LED3\_BIT**,** GPIO\_MODER\_OUT**)** // left B

**};**

static uint32\_t cpxctrl**[**CPXROWS**];** // cpx control data

// m: bgrbgrbgrbgr, msb - left diode, lsb - right diode

static void cpx\_encode**(**uint32\_t m**)**

**{**

cpxctrl**[**0**]** **=** 0**;**

cpxctrl**[**1**]** **=** 0**;**

cpxctrl**[**2**]** **=** 0**;**

cpxctrl**[**3**]** **=** 0**;**

**for** **(**uint32\_t i **=** 0**;** i **<** 12**;** i **++)**

**{**

**if** **(**m **&** 1**)** cpxctrl**[**i **/** 3**]** **|=** cpxmask**[**i**];**

m **>>=** 1**;**

**}**

**}**

void SysTick\_Handler**(**void**)**

**{**

static uint8\_t phase**;**

CPX\_PORT**->**MODER **&=** CPX\_MODER\_OFF**;**

CPX\_PORT**->**BSRR **=** CPX\_BSRR\_OFF **|** CPX\_LED3\_MSK **>>** phase**;**

CPX\_PORT**->**MODER **|=** cpxctrl**[**phase**];**

**if** **(++**phase **==** CPXROWS**)** phase **=** 0**;**

static uint16\_t tdiv**;**

**if** **(++**tdiv **==** CPX\_FREQ**)**

**{**

tdiv **=** 0**;**

// prepare image

static uint8\_t seq**;**

**if** **(++**seq **==** CPXROWS**)** seq **=** 0**;**

cpx\_encode**(**042104210 **>>** **(**seq **\*** 3**));**

**}**

**}**

**Listing 3. Obsługa wyświetlacza multipleksowanego przy użyciu DMA**

/\*

STM32L4 KAMduino LoL shield demo

gbm, 11'2016

\*/

#include "stm32l4yy.h"

#include "stm32nucleo64.h"

#include "stm32nucleo64\_ard.h"

#define SYSCLK\_FREQ 80000000u

#define SYSTICK\_FREQ 4000u

#define IMGROWS 9

#define IMGCOLS 14

#define CPXPHASES 12

#define PATTERN 0x03030303 // bar pattern to animate

#define GPIOA\_MODER\_CPXOFF GPIOA\_MODER\_SWDA

#define GPIOB\_MODER\_CPXOFF 0xffffffff

#define GPIOC\_MODER\_CPXOFF 0xffffffff

#define GPIOA\_BRR\_CPXOFF AR\_D\_GPIOA\_MASK

#define GPIOB\_BRR\_CPXOFF AR\_D\_GPIOB\_MASK

#define GPIOC\_BRR\_CPXOFF AR\_D\_GPIOC\_MASK

enum gpio\_ **{**PA**,** PB**,** PC**};** // gpio port indices

GPIO\_TypeDef **\*** const portmap**[]** **=** **{**GPIOA**,** GPIOB**,** GPIOC**};**

// CPX pin indices (NOT pin numbers)

enum dind\_ **{**D2**,** D3**,** D4**,** D5**,** D6**,** D7**,** D8**,** D9**,** D10**,** D11**,** D12**,** D13**};**

// pin map - gpio ports and bits for Arduino conn D2..D13 pins

const struct **{**

uint8\_t port**:**4**,** bit**:**4**;**

**}** ard\_pinmap**[]** **=** **{**

**{**PA**,** AR\_D2\_BIT**},**

**{**PB**,** AR\_D3\_BIT**},**

**{**PB**,** AR\_D4\_BIT**},**

**{**PB**,** AR\_D5\_BIT**},**

**{**PB**,** AR\_D6\_BIT**},**

**{**PA**,** AR\_D7\_BIT**},**

**{**PA**,** AR\_D8\_BIT**},**

**{**PC**,** AR\_D9\_BIT**},**

**{**PB**,** AR\_D10\_BIT**},**

**{**PA**,** AR\_D11\_BIT**},**

**{**PA**,** AR\_D12\_BIT**},**

**{**PA**,** AR\_D13\_BIT**}**

**};**

// anode, cathode pin for each LOL LED

static const struct **{**

uint8\_t a**:**4**,** c**:**4**;**

**}** lolmap**[**IMGROWS**][**IMGCOLS**]** **=** **{**

**{{**D13**,** D5**},** **{**D13**,** D6**},** **{**D13**,** D7**},** **{**D13**,** D8**},** **{**D13**,** D9**},** **{**D13**,** D10**},** **{**D13**,** D11**},**

**{**D13**,** D12**},** **{**D13**,** D4**},** **{**D4**,** D13**},** **{**D13**,** D3**},** **{**D3**,** D13**},** **{**D13**,** D2**},** **{**D2**,** D13**}},**

**{{**D12**,** D5**},** **{**D12**,** D6**},** **{**D12**,** D7**},** **{**D12**,** D8**},** **{**D12**,** D9**},** **{**D12**,** D10**},** **{**D12**,** D11**},**

**{**D12**,** D13**},** **{**D12**,** D4**},** **{**D4**,** D12**},** **{**D12**,** D3**},** **{**D3**,** D12**},** **{**D12**,** D2**},** **{**D2**,** D12**}},**

**{{**D11**,** D5**},** **{**D11**,** D6**},** **{**D11**,** D7**},** **{**D11**,** D8**},** **{**D11**,** D9**},** **{**D11**,** D10**},** **{**D11**,** D12**},**

**{**D11**,** D13**},** **{**D11**,** D4**},** **{**D4**,** D11**},** **{**D11**,** D3**},** **{**D3**,** D11**},** **{**D11**,** D2**},** **{**D2**,** D11**}},**

**{{**D10**,** D5**},** **{**D10**,** D6**},** **{**D10**,** D7**},** **{**D10**,** D8**},** **{**D10**,** D9**},** **{**D10**,** D11**},** **{**D10**,** D12**},**

**{**D10**,** D13**},** **{**D10**,** D4**},** **{**D4**,** D10**},** **{**D10**,** D3**},** **{**D3**,** D10**},** **{**D10**,** D2**},** **{**D2**,** D10**}},**

**{{**D9**,** D5**},** **{**D9**,** D6**},** **{**D9**,** D7**},** **{**D9**,** D8**},** **{**D9**,** D10**},** **{**D9**,** D11**},** **{**D9**,** D12**},**

**{**D9**,** D13**},** **{**D9**,** D4**},** **{**D4**,** D9**},** **{**D9**,** D3**},** **{**D3**,** D9**},** **{**D9**,** D2**},** **{**D2**,** D9**}},**

**{{**D8**,** D5**},** **{**D8**,** D6**},** **{**D8**,** D7**},** **{**D8**,** D9**},** **{**D8**,** D10**},** **{**D8**,** D11**},** **{**D8**,** D12**},**

**{**D8**,** D13**},** **{**D8**,** D4**},** **{**D4**,** D8**},** **{**D8**,** D3**},** **{**D3**,** D8**},** **{**D8**,** D2**},** **{**D2**,** D8**}},**

**{{**D7**,** D5**},** **{**D7**,** D6**},** **{**D7**,** D8**},** **{**D7**,** D9**},** **{**D7**,** D10**},** **{**D7**,** D11**},** **{**D7**,** D12**},**

**{**D7**,** D13**},** **{**D7**,** D4**},** **{**D4**,** D7**},** **{**D7**,** D3**},** **{**D3**,** D7**},** **{**D7**,** D2**},** **{**D2**,** D7**}},**

**{{**D6**,** D5**},** **{**D6**,** D7**},** **{**D6**,** D8**},** **{**D6**,** D9**},** **{**D6**,** D10**},** **{**D6**,** D11**},** **{**D6**,** D12**},**

**{**D6**,** D13**},** **{**D6**,** D4**},** **{**D4**,** D6**},** **{**D6**,** D3**},** **{**D3**,** D6**},** **{**D6**,** D2**},** **{**D2**,** D6**}},**

**{{**D5**,** D6**},** **{**D5**,** D7**},** **{**D5**,** D8**},** **{**D5**,** D9**},** **{**D5**,** D10**},** **{**D5**,** D11**},** **{**D5**,** D12**},**

**{**D5**,** D13**},** **{**D5**,** D4**},** **{**D4**,** D5**},** **{**D5**,** D3**},** **{**D3**,** D5**},** **{**D5**,** D2**},** **{**D2**,** D5**}}**

**};**

int main**(**void**)**

**{**

// PLL - 40 MHz

RCC**->**PLLCFGR **=** RCC\_PLLCFGR\_PLLREN **|** RCC\_PLLCFGR\_PLLNV**(**40**)** **|** RCC\_PLLCFGR\_PLLMV**(**1**)** **|** RCC\_PLLCFGR\_PLLSRC\_MSI**;**

RCC**->**CR **|=** RCC\_CR\_PLLON**;** //

RCC**->**AHB2ENR **=** RCC\_AHB2ENR\_GPIOAEN **|** RCC\_AHB2ENR\_GPIOBEN **|** RCC\_AHB2ENR\_GPIOCEN**;**

FLASH**->**ACR **|=** FLASH\_ACR\_PRFTEN **|** FLASH\_ACR\_LATENCY\_4WS**;**

GPIOA**->**PUPDR **=** GPIOA\_PUPDR\_SWD**;** // leave only SWD pulls

GPIOB**->**PUPDR **=** 0**;** // turn off default pulls

**while** **(!(**RCC**->**CR **&** RCC\_CR\_PLLRDY**));**

RCC**->**CFGR **|=** RCC\_CFGR\_SW\_PLL**;**

**while** **((**RCC**->**CFGR **&** RCC\_CFGR\_SWS**)** **!=** RCC\_CFGR\_SWS\_PLL**);**

SysTick\_Config**(**SYSCLK\_FREQ **/** SYSTICK\_FREQ**);**

SCB**->**SCR **=** SCB\_SCR\_SLEEPONEXIT\_Msk**;**

\_\_WFI**();** // sleep

**}**

void SysTick\_Handler**(**void**)**

**{**

static uint16\_t lol\_img**[**IMGROWS**];** // image to display - 9 rows x 14 cols

static \_Bool lol\_upd**;** // update request flag

static uint8\_t cpxphase**;** // current cpx phase

static uint32\_t moder\_val**[**CPXPHASES**][**3**];**

GPIOA**->**MODER **=** GPIOA\_MODER\_CPXOFF**;** // all LEDs off

GPIOB**->**MODER **=** GPIOB\_MODER\_CPXOFF**;** // all LEDs off

GPIOC**->**MODER **=** GPIOC\_MODER\_CPXOFF**;** // all LEDs off

GPIOA**->**BRR **=** GPIOA\_BRR\_CPXOFF**;**

GPIOB**->**BRR **=** GPIOB\_BRR\_CPXOFF**;**

GPIOC**->**BRR **=** GPIOC\_BRR\_CPXOFF**;**

GPIOA**->**MODER **=** moder\_val**[**cpxphase**][**PA**];**

GPIOB**->**MODER **=** moder\_val**[**cpxphase**][**PB**];**

GPIOC**->**MODER **=** moder\_val**[**cpxphase**][**PC**];**

portmap**[**ard\_pinmap**[**cpxphase**].**port**]->**BSRR **=** 1u **<<** ard\_pinmap**[**cpxphase**].**bit**;**

**if** **(++** cpxphase **==** CPXPHASES**)**

**{**

cpxphase **=** 0**;**

**if** **(**lol\_upd**)**

**{**

lol\_upd **=** 0**;**

// convert bitmap to MODER content

// anode activation

**for** **(**uint32\_t ph **=** 0**;** ph **<** CPXPHASES**;** ph **++)**

**{**

moder\_val**[**ph**][**PA**]** **=** GPIOA\_MODER\_CPXOFF**;**

moder\_val**[**ph**][**PB**]** **=** GPIOB\_MODER\_CPXOFF**;**

moder\_val**[**ph**][**PC**]** **=** GPIOC\_MODER\_CPXOFF**;**

moder\_val**[**ph**][**ard\_pinmap**[**ph**].**port**]** **&=** BF2A**(**ard\_pinmap**[**ph**].**bit**,** GPIO\_MODER\_OUT**);**

**}**

// cathode activation

**for** **(**uint32\_t r **=** 0**;** r **<** IMGROWS**;** r **++)**

**for** **(**uint32\_t c **=** 0**;** c **<** IMGCOLS**;** c **++)**

**if** **(**lol\_img**[**r**]** **>>** c **&** 1**)**

**{**

uint32\_t cathode **=** lolmap**[**r**][**c**].**c**;**

moder\_val**[**lolmap**[**r**][**c**].**a**][**ard\_pinmap**[**cathode**].**port**]** **&=** BF2A**(**ard\_pinmap**[**cathode**].**bit**,** GPIO\_MODER\_OUT**);**

**}**

**}**

**}**

static uint16\_t tdiv**;**

**if** **(++** tdiv **==** SYSTICK\_FREQ **/** 10**)**

**{**

tdiv **=** 0**;**

// animate arrow pattern

static uint8\_t s**;**

lol\_img**[**8**]** **=** lol\_img**[**0**]** **=** PATTERN **>>** s**;**

lol\_img**[**7**]** **=** lol\_img**[**1**]** **=** PATTERN **>>** **(**s **+** 1**);**

lol\_img**[**6**]** **=** lol\_img**[**2**]** **=** PATTERN **>>** **(**s **+** 2**);**

lol\_img**[**5**]** **=** lol\_img**[**3**]** **=** PATTERN **>>** **(**s **+** 3**);**

lol\_img**[**4**]** **=** PATTERN **>>** **(**s **+** 4**);**

**if** **(++**s **==** 8**)** s **=** 0**;**

lol\_upd **=** 1**;**

**}**

**}**