**Listing 1. Funkcja wysyłająca jeden bajt przez SPI**

uint8\_t SPI\_Exchange8bit**(**uint8\_t data**)**

**{**

 // Clear the Write Collision flag, to allow writing

 SSP1CON1bits**.**WCOL **=** 0**;**

 SSPBUF **=** data**;**

 **while(**SSP1STATbits**.**BF **==** SPI\_RX\_IN\_PROGRESS**)**

 **{**

 **}**

 **return** **(**SSPBUF**);**

**}**

**Listing 2. Wysłanie 3 bajtów koloru**

void SendColor**(**uint24\_t color**)**

**{**

 SPI\_Exchange8bit**(**color**>>**16**);**

 SPI\_Exchange8bit**(**color**>>**8**);**

 SPI\_Exchange8bit**(**color**);**

**}**

**Listing 3. Zmodyfikowana procedura wysłania 3 bajtów koloru**

void SenColor**(**uint8\_t G**,**uint8\_t R**,** uint8\_t B**)**

**{**

 SPI\_Exchange8bit**(**G**);**

 SPI\_Exchange8bit**(**R**);**

 SPI\_Exchange8bit**(**B**);**

**}**

**Listing 4. Nieskończona pętla zaświecająca cyklicznie 24 diody**

color**=**0x550000**;**

**while(**1**)**

**{**

 g**=**color**>>**16**;**

 r**=**color**>>**8**;**

 b**=**color**;**

 **for(**i**=**0**;**i**<**8**;**i**++)**

 **{**

 SPI\_Exchange8bit**(**g**);**

 SPI\_Exchange8bit**(**r**);**

 SPI\_Exchange8bit**(**b**);**

 SPI\_Exchange8bit**(**b**);**

 SPI\_Exchange8bit**(**g**);**

 SPI\_Exchange8bit**(**r**);**

 SPI\_Exchange8bit**(**r**);**

 SPI\_Exchange8bit**(**b**);**

 SPI\_Exchange8bit**(**g**);**

 SPI\_Exchange8bit**(**0**);**

 **}**

 \_\_delay\_ms**(**100**);**

 color**=(**color**>>**8**);**

 **if(**color**==**0**)**

 **{**

 color**=**0x550000**;**

 **}**

**}**

**Listing 5. Funkcja obsługi przerwań**

void interrupt INTERRUPT\_InterruptManager **(**void**)**

**{**

 //interrupt handler

 **if(**INTCONbits**.**PEIE **==** 1 **&&** PIE1bits**.**SSP1IE **==** 1 **&&** PIR1bits**.**SSP1IF **==** 1**)**

 **{**

 SPI\_ISR**();**

 **}**

 **else**

 **{**

 //Unhandled Interrupt

 **}**

**}**

**Listing 6. Procedura obsługi przerwania od SPI**

void SPI\_ISR**(**void**)**

**{**

 // clear the SPI interrupt flag

 PIR1bits**.**SSP1IF**=**0**;**

 WS2812\_Send**();**

**}**

**Listing 7. Wysłanie bufora buffer przez SPI**

volatile uint8\_t point**=**0**;** //licznik adresujący elementy w buforze

uint8\_t buffer**[**72**]={**0x55**,**0**,**0**,**0**,**0x55**,**0**,**0**,**0**,**0x55**,**//bufor z predefiniowanymi wartościami

0x55**,**0**,**0**,**0**,**0x55**,**0**,**0**,**0**,**0x55**,**

0x55**,**0**,**0**,**0**,**0x55**,**0**,**0**,**0**,**0x55**,**

0x55**,**0**,**0**,**0**,**0x55**,**0**,**0**,**0**,**0x55**,**

0x55**,**0**,**0**,**0**,**0x55**,**0**,**0**,**0**,**0x55**,**

0x55**,**0**,**0**,**0**,**0x55**,**0**,**0**,**0**,**0x55**,**

0x55**,**0**,**0**,**0**,**0x55**,**0**,**0**,**0**,**0x55**,**

0x55**,**0**,**0**,**0**,**0x55**,**0**,**0**,**0**,**0x55**,};**

void WS2812\_Send**(**void**){**

 **if(**point**>=**MAX\_BUF\_SPI**)**

 **{**

 PIE1bits**.**SSP1IE**=**0**;**

 point**=**0**;**

 **return;**

 **}**

 SSPBUF**=**buffer**[**point**++];**

**}**

**Listing 8. Inicjowanie wysyłania zawartości bufora buffer**

SPBUF **=** 0**;**

\_\_delay\_ms**(**1**);**

PIE1bits**.**SSP1IE**=**1**;**