**Listing 1. Inicjalizacja licznika TMR0**

void TMR0\_Initialize**(**void**)** **{**

/\* Set TMR0 to the options selected in the User Interface

PSA assigned; PS 1:16; TMRSE Increment\_hi\_lo; mask the nWPUEN and INTEDG bits \*/

OPTION\_REG **=** **(**OPTION\_REG **&** 0xC0**)** **|** 0xD3 **&** 0x3F**;**

// TMR0 6;

TMR0 **=** 0x06**;**

// Load the TMR value to reload variable

timer0ReloadVal **=** 6**;**

// Clear Interrupt flag before enabling the interrupt

INTCONbits**.**TMR0IF **=** 0**;**

// Enabling TMR0 interrupt

INTCONbits**.**TMR0IE **=** 1**;**

**}**

**Listing 2. Funkcja użytkownika wywoływana co 1msek z obsługi przerwania**

#define timeout bits.timeout

extern unsigned int Delay**;**

void TMR0\_CallBack**(**void**)** **{**

/\* Add your custom callback code here

this code executes every 1 TMR0 periods \*/

**if(!**timeout**)** **{**

**--**Delay**;**

**if(**Delay**==**0**)** timeout**=**1**;**

**}**

//obsluga enkodera

encoder**();**

CheckRtc**();**//sprawdzanie czy wyświetlić czas

**}**

**Listing 3. Inicjalizacja interfejsu I2C**

void I2C\_Initialize**(**void**)** **{**

i2c\_object**.**pTrHead **=** i2c\_tr\_queue**;**

i2c\_object**.**pTrTail **=** i2c\_tr\_queue**;**

i2c\_object**.**trStatus**.**s**.**empty **=** **true;**

i2c\_object**.**trStatus**.**s**.**full **=** **false;**

i2c\_object**.**i2cErrors **=** 0**;**

// BF RCinprocess\_TXcomplete; UA dontupdate; SMP Standard Speed; P stopbit\_notdetected

//S startbit\_notdetected; R\_nW write\_noTX; CKE Idle to Active; D\_nA lastbyte\_address;

SSPSTAT **=** 0x80**;**

// SSPM0 FOSC/4\_SSPxADD; SSPEN enabled; WCOL no\_collision; SSPOV no\_overflow;

//CKP Idle:Low, Active:High;

SSPCON1 **=** 0x28**;**

// BOEN disabled; AHEN disabled; SBCDE disabled; SDAHT 300ns; DHEN disabled;

//ACKTIM ackseq; PCIE disabled; SCIE disabled;

SSPCON3 **=** 0x08**;**

// Baud Rate Generator Value: SSPADD 39;

SSPADD **=** 0x27**;**

/\* Byte sent or received \*/

// clear the master interrupt flag

PIR1bits**.**SSPIF **=** 0**;**

// enable the master interrupt

PIE1bits**.**SSPIE **=** 1**;**

**}**

**Listing 4. Inicjalizacja ECCP1 i funkcja EPWM\_LoadDutyValue**

void EPWM1\_Initialize**(**void**)** **{**

/\* Set the PWM to the options selected in MPLAB® Code Configurator

P1M single; CCP1M P1AP1Chi\_P1BP1Dhi; DC1B 16; \*/

CCP1CON **=** 0x1C**;**

// CCP1AS0 disabled; PSS1AC0 P1AP1C\_0; CCP1ASE operating; PSS1BD0 P1BP1D\_0;

ECCP1AS **=** 0x00**;**

// P1DC0 0; P1RSEN automatic\_restart;

PWM1CON **=** 0x80**;**

// STR1A P1A\_to\_CCP1M; STR1C P1C\_to\_port; STR1B P1B\_to\_port;

// STR1D P1D\_to\_port; STR1SYNC start\_at\_begin;

PSTR1CON **=** 0x01**;**

// CCPR1L 120;

CCPR1L **=** 0x78**;**

// CCPR1H 0x0;

CCPR1H **=** 0x00**;**

// Selecting Timer2

CCPTMRS0bits**.**C1TSEL **=** 0x0**;**

**}**

void EPWM1\_LoadDutyValue**(**uint16\_t dutyValue**)** **{**

// Writing to 8 MSBs of pwm duty cycle in CCPRL register

CCPR1L **=** **((**dutyValue **&** 0x03FC**)** **>>** 2**);**

// Writing to 2 LSBs of pwm duty cycle in CCPCON register

CCP1CON **=** **(**CCP1CON **&** 0xCF**)** **|** **((**dutyValue **&** 0x0003**)** **<<** 4**);**

**}**

**Listing 5. Inicjalizacja TMR2**

void TMR2\_Initialize**(**void**)** **{**

/\* Set TMR2 to the options selected in the User Interface

TMR2ON off; T2CKPS 1:1; T2OUTPS 1:1; \*/

T2CON **=** 0x00**;**

// PR2 240;

PR2 **=** 0xF0**;**

// TMR2 0x0;

TMR2 **=** 0x00**;**

// Clearing IF flag.

PIR1bits**.**TMR2IF **=** 0**;**

// Start TMR2

TMR2\_StartTimer**();**

**}**

**Listing 6. Funkcja SYSTEM\_Initialize i OSCILLATOR\_Initialize**

void SYSTEM\_Initialize**(**void**)** **{**

OSCILLATOR\_Initialize**();**

PIN\_MANAGER\_Initialize**();**

TMR0\_Initialize**();**

I2C\_Initialize**();**

TMR2\_Initialize**();**

EPWM1\_Initialize**();**

**}**

void OSCILLATOR\_Initialize**(**void**)** **{**

// SPLLEN disabled; SCS INTOSC; IRCF 16MHz\_HF;

OSCCON **=** 0x7A**;**

// OSTS intosc; HFIOFR disabled; HFIOFS not0.5percent\_acc; PLLR disabled; T1OSCR disabled; MFIOFR disabled; HFIOFL not2percent\_acc; LFIOFR disabled;

OSCSTAT **=** 0x00**;**

// TUN 0x0;

OSCTUNE **=** 0x00**;**

// Set the secondary oscillator

**}**

**Listing 7. Inicjalizacja nastaw sterownika**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Inicjalizacja działania przedwzmacniacza po włączeniu zasilania

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void BOARDInit**(**void**)**

**{**

unsigned char vol**,** vols**;**

//sekwencja zalaczenia zasilania

STBY**=**0**;**asm**(**"nop"**);**asm**(**"nop"**);**asm**(**"nop"**);**asm**(**"nop"**);** //moduł analogowy wyłączony

MUTE**=**1**;** //uklad MUTE dziala

Delay\_ms**(**100**);**

STBY**=**1**;** //uklad analogowy włączany

Delay\_ms**(**500**);**

MUTE**=**0**;** //wyłącz MUTE

SCK**=**0**;**SDO**=**0**;**CS**=**1**;**//wartości początkowe SPI dla PGA2311

OPTION\_REGbits**.**nWPUEN**=**0**;**

PEIE**=**1**;** //odblokowanie przerwań od peryferii

ei**();** //odblokowanie wszystkich przerwań

Delay\_ms**(**500**);**

LcdInit**();** //inicjalizacja wyświetlacza

InitRtc**();** //inicjalizacja RTC

//ta sekwencja wykonuje sie po pierwszym zaprogramowaniu pamięci FLASH

**if(**DATAEE\_ReadByte**(**ADR\_INIT**)!=**0x29**){**

InitCzas**();**

DATAEE\_WriteByte**(**ADR\_VOL**,**20**);**//wartość tłumienia/wzmocnienia

DATAEE\_WriteByte**(**ADR\_FUN**,**0**);**//numer głównej funkcji

DATAEE\_WriteByte**(**ADR\_FUN1**,**0**);**//numer głównej funkcji

DATAEE\_WriteByte**(**ADR\_CWYS**,**0**);**//nie wyświetlaj czasu na ekranie

DATAEE\_WriteByte**(**ADR\_JW**,**5**);**//jasność wyświetlacza

DATAEE\_WriteByte**(**ADR\_INP**,**0**);**//pierwsze wejście aktywne

DATAEE\_WriteByte**(**ADR\_RC5\_ZN**,**0**);**//znacznik zaprogramowania kodów - nie zaprogramowane

DATAEE\_WriteByte**(**ADR\_INIT**,** 0x29**);**

**}**

EPWM1\_LoadDutyValue**(**DATAEE\_ReadByte**(**ADR\_JW**)\***100**);**//odtwórz jasność wyświetlacza

**if(**DATAEE\_ReadByte**(**ADR\_RC5\_ZN**)==**1**)** GetRC5Code**();** //jezeli kody sa w pamięci eeprom, to odczytaj i wpisz do RAM

vol**=**DATAEE\_ReadByte**(**ADR\_VOL**);**

vols**=(**vol**\***2**-**1**);**

SendPGA**(**vols**,**vols**);** //odtwórz poziom sygnału

SetIn**(**DATAEE\_ReadByte**(**ADR\_INP**));** //ustawienie aktywnego wejścia

**}**

**Listing 8. Inicjalizacja sterownika ST7066U**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Inicjalizacja sterownika LCD

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void LcdInit**(**void**){**

Delay\_ms**(**100**)** **;**

RS**=**0**;**

asm**(**"nop"**);**asm**(**"nop"**);**asm**(**"nop"**);**asm**(**"nop"**);**

asm**(**"nop"**);**asm**(**"nop"**);**asm**(**"nop"**);**asm**(**"nop"**);**

EN**=**0**;**

asm**(**"nop"**);**asm**(**"nop"**);**asm**(**"nop"**);**asm**(**"nop"**);**

asm**(**"nop"**);**asm**(**"nop"**);**asm**(**"nop"**);**asm**(**"nop"**);**

WrRiInit**(**2**);**

Delay\_ms**(**5**);**

WrRiInit**(**2**);**

Delay\_ms**(**5**);**

WrRiInit**(**8**);**

Delay\_ms**(**5**);**

WriteRi**(**0x0f**);**

Delay\_ms**(**5**);**

WriteRi**(**8**);**

Delay\_ms**(**5**);**

WriteRi**(**1**);**

Delay\_ms**(**5**);**

WriteRi**(**6**);**

Delay\_ms**(**5**);**

WriteRi**(**0x0c**);**

Delay\_ms**(**5**);**

**}**

**Listing 9. Inicjalizacja DS1307**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Wykrywanie zbocza opadającego sygnału RTC

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void CheckRtc**(**void**){**

**if(**rtc**==**0**){**

**if(**rtc0**==**1**&&** RTC**==**1**)**

**{**

rtc0**=**0**;**rtc1**=**1**;**

**}**

**if(**rtc1**==**1**&&**RTC**==**0**)**

**{**

rtc**=**1**;**rtc1**=**0**;**rtc**=**1**;**

**}**

**}**

**}**

**Listing 10. Testowanie stanu linii RTC – wywoływana co 1msek w obsłudze przerwania od TMR0 (listing2)**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Inicjalizacja zegara RTC

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void InitRtc**(**void**){**

KonfRtc**(**0x10**);**

**}**

**Listing 11. Pętla główna programu**

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

kod**=**GetEncoder**();**//testuj czy obrót ośki, przyciśnięcie oski, stan niski na IR

**if(**kod**==**KOD\_IR**)**

**{**

kod**=**GetRc5**();**//odbór kodu IR

Delay\_ms**(**100**);**

**}**

//wciśnięcie ośki enkodera

**if(**kod**==**KOD\_IMP\_ST**)**

**{**

**while(**ST**==**0**);**

MenuFun**();** //wejście do menu funkcyjnego

sdata**=**1**;**

**}**

//regulacja głośniej

**if(**kod**==**KOD\_IMP\_UP**||**kod**==**RC5\_UP**)**

**{**

**++**vol**;**

**if(**vol**==**127**)** vol**=**126**;**

PosLcd**(**3**,**1**);**

DispdB**(**vol**);**

vols**=(**vol**\***2**-**1**);**

SendPGA**(**vols**,**vols**);**

PosLcd**(**11**,**1**);**

**if(**vol**>**95**)** DispLinVol**(**8**);**

**else** DispLinVol**(**vol**/**11**);**

DATAEE\_WriteByte**(**ADR\_VOL**,** vol**);**

**}**

//regulacja ciszej

**if(**kod**==**KOD\_IMP\_DWN**||**kod**==**RC5\_DOWN**)**

**{**

**--**vol**;**

**if(**vol**==**0**)** vol**=**1**;**

PosLcd**(**3**,**1**);**

DispdB**(**vol**);**

vols**=(**vol**\***2**-**1**);**

SendPGA**(**vols**,**vols**);**

PosLcd**(**11**,**1**);**

**if(**vol**>**95**)** DispLinVol**(**8**);**

**else** DispLinVol**(**vol**/**11**);**

DATAEE\_WriteByte**(**ADR\_VOL**,** vol**);**

**}**

//sterowanie wejściami impulsatora

**if(**kod**==**RC5\_IN1**)**

**{**

PosLcd**(**3**,**3**);**

DispIn**(**0**);**

SetIn**(**0**);**

DATAEE\_WriteByte**(**ADR\_INP**,**0**);**

**}**

**if(**kod**==**RC5\_IN2**)**

**{**

PosLcd**(**3**,**3**);**

DispIn**(**1**);**

SetIn**(**1**);**

DATAEE\_WriteByte**(**ADR\_INP**,**1**);**

**}**

**if(**kod**==**RC5\_IN3**)**

**{**

PosLcd**(**3**,**3**);**

DispIn**(**2**);**

SetIn**(**2**);**

DATAEE\_WriteByte**(**ADR\_INP**,**2**);**

//komenda MUTE

**if(**kod**==**RC5\_MUTE**)** MuteVol**();**

//komenda włącz/wyłącz

**if(**kod**==**RC5\_POWER**)** Power**();**

**}**

DispCzas**(**1**,**4**);**

**}**

**Listing 12. Wysłanie 16 bitów do PGA2311**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Wysłanie 16 bitów do PGA2311

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void SendPGA**(**unsigned char volL**,** unsigned char volR**)**

**{**

uint16\_t vol**=**0**;**

unsigned char i**;**

vol**=**0**;**

vol**=**volR**;**

vol**=(**vol**<<**8**)|**volL**;**

CS**=**0**;**

del**();**

**for(**i**=**0**;**i**<**16**;**i**++)**

**{**

SCK**=**0**;**

del**();**

**if((**vol**&**0x8000**)==**0**)** SDO**=**0**;**

**else** SDO**=**1**;**

vol**=**vol**<<**1**;**

del**();**

SCK**=**1**;**

del**();**

**}**

SCK**=**0**;**

del**();**

SDO**=**0**;**

del**();**

CS**=**1**;**

del**();**

**}**

**Listing 13. Funkcja menu funkcyjnego**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Główne menu funkcyjne

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void MenuFun**(**void**){**

unsigned char fun**;**

unsigned char kod**;**

LcdClear**();** //wyczyść wyświetlacz

fun**=**DATAEE\_ReadByte**(**ADR\_FUN**);**

PosLcd**(**4**,**1**);** PosLcd**(**4**,**1**);**

DispLcd**(**"Zmien wejscie"**);**

PosLcd**(**4**,**2**);**

DispLcd**(**"Ustawienia"**);**

PosLcd**(**4**,**3**);**

DispLcd**(**"Kody RC5"**);**

PosLcd**(**4**,**4**);**

DispLcd**(**"Wyswietlacz"**);**

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

MarkFun**(**fun**);**

kod**=**GetEncoder**();**

**if(**kod**==**KOD\_IMP\_ST**)**

**{**

**while(**ST**==**0**);**

DATAEE\_WriteByte**(**ADR\_FUN**,** fun**);** //ostatnio wybrana funkcja

**break;**

**}** //przycisniecie oski - koniec ustawiania

**if(**kod**==**KOD\_IMP\_UP**)** //zwiększ wartość

**{**

**++**fun**;**

**if(**fun**==**4**)** fun**=**0**;** //ClearEnc();//wyczyść kody enkodera

**}**

**if(**kod**==**KOD\_IMP\_DWN**)** //zwiększ wartość

**{**

**--**fun**;**

**if(**fun**==**0xff**)** fun**=**3**;**

**}**

**}**

LcdClear**();** //wyczyść wyświetlacz

**if(**fun**==**0**)** UstawWejscie**();**

**if(**fun**==**1**)** MenuFun1**();**

**if(**fun**==**2**)** SetCodeRC5**();**

**if(**fun**==**3**)** UstawWysw**();**

MainScreen**();**

**}**

**Listing 14. Funkcja uczenia się kodów RC5**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Uczenie kodów RC5

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void SetCodeRC5**(**void**)**

**{**

int kod**;**

LcdClear**();**

PosLcd**(**1**,**1**);**

DispLcd**(**"kody RC5"**);**

PosLcd**(**1**,**2**);**

DispLcd**(**"DOWN ---- "**);**

**while(**IR**==**1**);** //czekaj na poczatek kodu

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

kod**=**GetRc5**();**

**if(**kod**!=**0xffff**)** **break;**

**}**

//RC5\_DOWN=kod;

DispHex**((**kod**>>**8**),**6**,**2**);** //kod rc5 w postaci hex

DispHex**(**kod**,**8**,**2**);**

DATAEE\_WriteByte**(**ADR\_RC5\_DWN**,**kod**>>**8**);**

DATAEE\_WriteByte**(**ADR\_RC5\_DWN**+**1**,**kod**);**

Delay\_ms**(**200**);**

**while(**ST**==**1**);** //czekaj na przyci?ni?cie oski

PosLcd**(**1**,**2**);**

DispLcd**(**" UP ---- "**);**

**while(**IR**==**1**);** //czekaj na poczatek kodu

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

kod**=**GetRc5**();**

**if(**kod**!=**0xffff**)**

**break;**

**}**

//RC5\_UP=kod;

DispHex**((**kod**>>**8**),**6**,**2**);** //kod rc5 w postaci hex

DispHex**(**kod**,**8**,**2**);**

DATAEE\_WriteByte**(**ADR\_RC5\_UP**,**kod**>>**8**);**

DATAEE\_WriteByte**(**ADR\_RC5\_UP**+**1**,**kod**);**

**while(**ST**==**1**);** //czekaj na przycisniecie oski

PosLcd**(**1**,**2**);**

DispLcd**(**"FUN ---- "**);**

**while(**IR**==**1**);** //czekaj na poczatek kodu

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

kod**=**GetRc5**();**

**if(**kod**!=**0xffff**)**

**break;**

**}**

//RC5\_FUN=kod;

DispHex**((**kod**>>**8**),**6**,**2**);** //kod rc5 w postaci hex

DispHex**(**kod**,**8**,**2**);**

DATAEE\_WriteByte**(**ADR\_RC5\_FUN**,**kod**>>**8**);**

DATAEE\_WriteByte**(**ADR\_RC5\_FUN**+**1**,**kod**);**

**while(**ST**==**1**);** //czekaj na przycisniecie oski

PosLcd**(**1**,**2**);**

DispLcd**(**"MUT ---- "**);**

**while(**IR**==**1**);** //czekaj na poczatek kodu

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

kod**=**GetRc5**();**

**if(**kod**!=**0xffff**)** **break;**

**}**

//RC5\_MUT=kod;

DispHex**((**kod**>>**8**),**6**,**2**);** //kod rc5 w postaci hex

DispHex**(**kod**,**8**,**2**);**

DATAEE\_WriteByte**(**ADR\_RC5\_MUT**,**kod**>>**8**);**

DATAEE\_WriteByte**(**ADR\_RC5\_MUT**+**1**,**kod**);**

**while(**ST**==**1**);** //czekaj na przycisniecie oski

PosLcd**(**1**,**2**);**

DispLcd**(**"IN1 ---- "**);**

**while(**IR**==**1**);** //czekaj na poczatek kodu

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

kod**=**GetRc5**();**

**if(**kod**!=**0xffff**)** **break;**

**}**

//RC5\_MUT=kod;

DispHex**((**kod**>>**8**),**6**,**2**);** //kod rc5 w postaci hex

DispHex**(**kod**,**8**,**2**);**

DATAEE\_WriteByte**(**ADR\_RC5\_IN1**,**kod**>>**8**);**

DATAEE\_WriteByte**(**ADR\_RC5\_IN1**+**1**,**kod**);**

**while(**ST**==**1**);** //czekaj na przycisniecie oski

PosLcd**(**1**,**2**);**

DispLcd**(**"IN2 ---- "**);**

**while(**IR**==**1**);** //czekaj na poczatek kodu

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

kod**=**GetRc5**();**

**if(**kod**!=**0xffff**)** **break;**

**}**

//RC5\_MUT=kod;

DispHex**((**kod**>>**8**),**6**,**2**);** //kod rc5 w postaci hex

DispHex**(**kod**,**8**,**2**);**

DATAEE\_WriteByte**(**ADR\_RC5\_IN2**,**kod**>>**8**);**

DATAEE\_WriteByte**(**ADR\_RC5\_IN2**+**1**,**kod**);**

**while(**ST**==**1**);** //czekaj na przycisniecie oski

PosLcd**(**1**,**2**);**

DispLcd**(**"IN3 ---- "**);**

**while(**IR**==**1**);** //czekaj na poczatek kodu

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

kod**=**GetRc5**();**

**if(**kod**!=**0xffff**)** **break;**

**}**

//RC5\_MUT=kod;

DispHex**((**kod**>>**8**),**6**,**2**);** //kod rc5 w postaci hex

DispHex**(**kod**,**8**,**2**);**

DATAEE\_WriteByte**(**ADR\_RC5\_IN3**,**kod**>>**8**);**

DATAEE\_WriteByte**(**ADR\_RC5\_IN3**+**1**,**kod**);**

**while(**ST**==**1**);** //czekaj na przycisniecie oski

Delay\_ms**(**20**);**

**while(**ST**==**0**);** //czekaj na puszczenie oski

DATAEE\_WriteByte**(**ADR\_RC5\_ZN**,**1**);** //znacznik zaprogramowania kodow

GetRC5Code**();**

**}**