Errata
do książki

„Mikrokontrolery AVR
Język C
Podstawy programowania”

Błędy związane z korektą tekstu, znalezione po ukazaniu się pierwszego wydania.

jest	*powinno być*
Canfiguration	Configuration
warnigi	warningi
Flasch	Flash
Fiemware	Firmware
Programers Notepad	Programmers Notepad
ememntów	elementów
presklarel	preskaler
wyrwietlacza	wyświetlacza
Zzapis	zapis
fukcję	funkcję
dziesietną	dziesiętną
elegancji sposób	elegancki sposób
onacza	oznacza
rozdzieloczością	rozdzielczością
procdury	procedury
wsytawiać	wystawiać
konieczność wprowadzenie	konieczność wprowadzenia
modłu	modułu
odrotną	odwrotną
so	do
wymedytujesz	wyedytujesz
Kolejną krok	Kolejny krok
utrudnieniemjuest	utrudnieniem jest

Strona 61 – przykład kodu:

jest	*powinno być*

`if(warunek_1)`
`{`
`  //instrukcje`
`}`
`else`
`{`
`  // instrukcje`
`}`
	`if(warunek_1)`
	`{`
	`  if(warunek_2){ //instrukcje }`
	`}`
	`else`
	`{`
	`  // instrukcje`
	`}`

Strona 25:

Jest:

„Przykładowo dla portu C mamy w procesorze ATmega8 do swobodnego wykorzystania jest tylko **5** pinów: PC0, PC1, PC2, PC3, PC4, PC5."

Powinno być:

„Przykładowo dla portu C mamy w procesorze ATmega8 do swobodnego wykorzystania jest tylko **6** pinów: PC0, PC1, PC2, PC3, PC4, PC5."

Jest:

„ATtiny2313 z obudową DIP20, posiada tylko dwa pełne 8 bitowe porty: PORTB i PORTD"

Powinno być:

ATtiny2313 z obudową DIP20, posiada tylko jeden 8 bitowy port: PORTB, 7 bitowy PORTD

Strona 65:

Jest:

```
for(i=0,k=10;i<10;i=i_1,k=k-1)
```

Powinno być:

```
for(i=0,k=10;i<10;i=i+1,k=k-1)
```

Strona 73 - tabela:

Jest:

bool logiczny logiczny 2

Powinno być:

bool logiczny logiczny **1**

Strona 74 :

Jest:

```
for(uint8_t i=0;i<10;I=I+1) instrukcja;
```

Powinno być:

```
for(uint8_t i=0;i<10;i=i+1) instrukcja;
```

Strona 76:

jest	*powinno być*

`uint8_t z=5, s=20;// deklaracja zmiennych lokalnych`
`uint8_t m=13;  // deklaracja zmiennej lokalnej`
 	
`int z=10; // deklaracja zmiennej lokalnej`
	`uint8_t z=5, s=20;// definicja zmiennych lokalnych`
	`uint8_t m=13;  // definicja zmiennej lokalnej`
	
	`int z=10; // definicja zmiennej lokalnej`

Strona 90:

Jest:

„Wystarczy rozpisać sobie działania na liczbach od 0 do 20,”

Powinno być:

„Wystarczy rozpisać sobie działania na liczbach od 0 do 19,”

Jest:

```
#define co_ile 2

for(uint8_t i=0;i<20;i=i+1)
{
      if( (i % co_ile) == 0 ) put_text("Test");
            _delay_ms(1000);
}
```

Powinno być:

```
#define co_ile 2

for(uint8_t i=0;i<20;i=i+1)
{
      if( (i % co_ile) == 0 ) put_text("Test");
            _delay_ms(1000);
}
```

Strona 95:

Jest:

<< - przesunięcie bitowe w lewo

>> - przesunięcie bitowe w prawo

& - bitowy iloczyn logiczny (AND)

| - bitowa suma logiczna (OR)

^ - bitowa różnica symetryczna (XOR)

~ - bitowa negacja

Powinno być:

<< - przesunięcie bitowe w lewo

>> - przesunięcie bitowe w prawo

& - bitowy iloczyn logiczny (AND)

| - bitowa suma logiczna (OR)

^ - bitowa różnica symetryczna (XOR)

~ - bitowa negacja

Strona 98:

Jest:

#define PD7 8

Powinno być:

#define PD7 7

Strona 102:

Jest:

W pierwszym przypadku do zmiennej a zostanie podstawiona liczba ujemna o wartości dwa, natomiast w drugim przypadku od zawartości zmiennej a zostanie odjęta liczba (stała) o artości dwa

Powinno być:

W pierwszym przypadku do zmiennej a zostanie podstawiona liczba ujemna o wartości dwa, natomiast w drugim przypadku od zawartości zmiennej a zostanie odjęta liczba (stała) o wartości dwa

Strona 105:

Jest:

„Przykładem matematycznego operatora, który ma łączność prawostronną jest potęgowanie, np. jest równe 81."

Powinno być:

„Przykładem matematycznego operatora, który ma łączność prawostronną jest potęgowanie, np. $3^{\wedge 2 \wedge 2}$ jest równe 81."

Strona 106:

Jest:

Jednym słowem możemy w programie dokonać zamiany typu char na ty int,

Powinno być:

Jednym słowem możemy w programie dokonać zamiany typu char na typ int,

Strona 106:

Jest:

```
m = char(b);
```

Powinno być:

```
m = (char)b;
```

Strona 108:

jest	powinno być

`#include <avr/io.h>  //import pliku nagłówkowego`
 	
`uint8_t zapal_bit(uint8_t nr);`
`  //deklaracja funkcji`
 	
`//*************************************`
`int main(void)`
`  //funkcja główna programu`
`{`
`  DDRB=0xFF;`
`  //ustawienie całego portu B jako wyjście`
 	
`  PORTB=waga_bitu(5);`
`  //wywołanie funkcji i wynik na port B`
`}`
`//*************************************`
 	
`uint8_t zapal_bit (uint8_t nr)`
`  //definicja funkcji`
`{`
`  uint8_t temp;`
`  temp=(1<<nr);`
 	
`  return temp;`
`  // koniec funkcji, zwrot wyniku`
`}`
	`#include <avr/io.h>  //import pliku nagłówkowego`
	
	`uint8_t zapal_bit(uint8_t nr);`
	`  //deklaracja funkcji`
	
	`//*************************************`
	`int main(void)  //funkcja główna programu`
	`{`
	`  DDRB=0xFF;`
	`  //ustawienie całego portu B jako wyjście`
	
	`  PORTB=zapal_bit(5);`
	`  //wywołanie funkcji i wynik na port B`
	`}`
	`//*************************************`
	
	`uint8_t zapal_bit (uint8_t nr)`
	`  //definicja funkcji`
	`{`
	`  uint8_t temp;`
	`  temp=(1<<nr);`
	
	`  return temp;`
	`  // koniec funkcji, zwrot wyniku`
	`}`

Strona 114:

Jest:

Zatem w momencie, gdy wywołujemy różnego rodzaju funkcje, podajemy już argumenty formalne albo inaczej mówiąc parametry aktualne.

Powinno być:

Zatem w momencie, gdy wywołujemy różnego rodzaju funkcje, podajemy już argumenty aktualne albo inaczej mówiąc parametry aktualne.

Strona 115:

Jest:

To bardzo cenna informacja dla ciebie i jeden z pierwszych korków na drodze do ujarzmienia stosu.

Powinno być:

To bardzo cenna informacja dla ciebie i jeden z pierwszych kroków na drodze do ujarzmienia stosu.

Strona 117:

Jest:

Zaoszczędzimy na stosie na pewno, zaoszczędzimy na czasie wykonywania takiego kodu, bo przecież nie będzie musiał być wykonywany skok do funkcji, nie będą musiałby być tworzone kopie

Powinno być:

Zaoszczędzimy na stosie na pewno, zaoszczędzimy na czasie wykonywania takiego kodu, bo przecież nie będzie musiał być wykonywany skok do funkcji, nie będą musiały być tworzone kopie

Strona 118:

Jest:

Jedne z pinów służyłyby do sterowania tranzystorów, inne bramek logicznych a jeszcze inne byłby wejściami

Powinno być:

Jedne z pinów służyłyby do sterowania tranzystorów, inne bramek logicznych a jeszcze inne byłyby wejściami

Strona 122:

Jest:

Dlatego, że tym razem sama deklaracja mu nie wystarczy, on musi znać już całą definicję funkcji, aby ocenić, czy uda mi się ją wbudować w to miejsce i wykonać to, jeśli uzna, że tak.

Powinno być:

Dlatego, że tym razem sama deklaracja mu nie wystarczy, on musi znać już całą definicję funkcji, aby ocenić, czy uda mu się ją wbudować w to miejsce i wykonać to, jeśli uzna, że tak.

Jest:

Miałoby prawo to działać, jednak wyobraź sobie, co by się stało, gdyby zaszła konieczność drobnej zmiany której z nich

Powinno być:

Miałoby prawo to działać, jednak wyobraź sobie, co by się stało, gdyby zaszła konieczność drobnej zmiany którejś z nich

Strona 124:

Jest:

że wraz końcem funkcji taka zmienna łącznie z jej lokalizacją w pamięci RAM znika bezpowrotnie.

Powinno być:

że wraz z końcem funkcji taka zmienna łącznie z jej lokalizacją w pamięci RAM znika bezpowrotnie.

Strona 131:

Jest:

```
// deklaracje funkcji udostępnianych na zewn.
voidl cd_init(void);.
```

Powinno być:

```
// deklaracje funkcji udostępnianych na zewn.
  void lcd_init(void);
```

Strona 136:

Jest:

```
TOSTRING (MCU)
. . .
TOSTRING(a)   STRINGX(a)
STRINGX(a) #a
```

Powinno być:

```
#define TOSTRING (MCU)
. . .
#define TOSTRING(a)   STRINGX(a)
#define STRINGX(a) #a
```

Strona 159:

Jest:

„a=fun(&a); *// wywołanie funkcji i przypisanie rezultatu do a*

Powinno być:

„fun(&a); *// wywołanie funkcji i przypisanie rezultatu do a*

Strona 160:

Jest:

```
fun1(tab);          // przekazanie adresu zerowego elementu do fun1
fun1(&tab[3]);      // przekazanie adresu trzeciego elementu do fun1
fun1(&(tab+5));     // przekazanie adresu piątego elementu do fun1

fun2(tab);          // przekazanie adresu zerowego elementu do fun2
fun2(&tab[3]);      // przekazanie adresu trzeciego elementu do fun2
fun2(&(tab+5));     // przekazanie adresu piątego elementu do fun2
```

Powinno być:

```
fun1(tab);          // przekazanie adresu zerowego elementu do fun1
fun1(&tab[3]);      // przekazanie adresu czwartego elementu do fun1
fun1(&(tab+5));     // przekazanie adresu szóstego elementu do fun1

fun2(tab);          // przekazanie adresu zerowego elementu do fun2
fun2(&tab[3]);      // przekazanie adresu czwartego elementu do fun2
fun2(&(tab+5));     // przekazanie adresu szóstego elementu do fun2
```

Strona 161:

Jest:

„int tab[] = {100, 333, 999, 2500, 45000}”

Powinno być:

„int tab[] = {100, 333, 999, 2500, 4500}”

Strona 163:

Jest:

„uint9_t *wsk;”

Powinno być:

„uint8_t *wsk;”

Strona 178:

Jest:

„Kod programu przedstawiony jest w ramce nr XX.”

Powinno być:

„Kod programu przedstawiony jest poniżej.”

Strona 184:

Jest:

„teraz trzeba przymierzyć do oprogramowania”

Powinno być:

„teraz trzeba przymierzyć się do oprogramowania”

Strona 184:

Jest:

„Chcemy tak ustawić jeden z dostępnych Timerów, aby generował nam przerwania co 200Hz”

Powinno być:

„Chcemy tak ustawić jeden z dostępnych Timerów, aby generował nam przerwania co 5ms (z częstotliwością 200Hz).”

Strona 190:

Jest:

```
„// mających się wyświetlać na wyrw. LED"
```

Powinno być:

```
„// mających się wyświetlać na wyśw. LED"
```

Strona 191:

Jest:

```
ANODY |= CA1 | CA2 | CA3  | CA4;
```

Powinno być:

```
ANODY_PORT |= CA1 | CA2 | CA3  | CA4;
```

Strona 192:

Jest:

```
#define     LED_DATA      PORTC  // port z podłączonymi segmentami
// port z podłączonymi anodami- 4 bity najmłodsze
#define     ANODY_PORT    PORTA

#define     CA1    (1<<PA0)      // CA1 oznacza bit nr nr 0 portu A
#define     CA2    (1<<PA1)      // CA2 oznacza bit nr nr 1 portu A
#define     CA3    (1<<PA2)      // CA3 oznacza bit nr nr 2 portu A
#define     CA4    (1<<PA3)      // CA4 oznacza bit nr nr 3 portu A"
```

Powinno być:

```
#define     LED_DATA      PORTC  // port z podłączonymi segmentami
#define     LED_DATA_DIR  DDRC
#define     ANODY_PORT    PORTA
#define     ANODY_DIR     DDRA

#define     CA1    (1<<PA0)      // CA1 oznacza bit nr 0 portu A
#define     CA2    (1<<PA1)      // CA2 oznacza bit nr 1 portu A
#define     CA3    (1<<PA2)      // CA3 oznacza bit nr 2 portu A
#define     CA4    (1<<PA3)      // CA4 oznacza bit nr 3 portu A"
```

Strona 192:

Jest:

```
void d_led_init(void)
```

Powinno być:

```
void d_led_init(void);
```

Strona 214:

Jest:

```
_delay_ms(15);
PORT(LCD_RSPORT) &= ~(1<<LCD_RS);
PORT(LCD_RWPORT) &= ~(1<<LCD_RW);
```

Powinno być:

```
_delay_ms(15);
PORT(LCD_RSPORT) &= ~(1<<LCD_RS);
#if USE_RW == 1
     PORT(LCD_RWPORT) &= ~(1<<LCD_RW);
#endif
```

Strona 238:

Jest:

```
ADMUX |= (ADMUX & 0xF8) | kanal; <---------- poważny błąd
```

Powinno być:

```
ADMUX = (ADMUX & 0xF8) | kanal;
```

Jest:

```
while( ADCSR & ADSC );
```

Powinno być:

```
while( ADCSR & (1<<ADSC) );
```

Jest:

```
ADMUX |= (1<<REFS0)|(1<<REFS2);
```

Powinno być:

```
ADMUX |= (1<<REFS0)|(1<<REFS1);
```

Strona 252: poprawki jednostek w 3 tabelkach

ATmega32			
prąd	bocznik	dzielnik	
A	V	V	ADC
0,01	0,001	0,0005	0
0,05	0,005	0,0025	1
0,1	0,010	0,0050	1
0,15	0,015	0,0075	2
0,2	0,020	0,0100	2
0,4	0,040	0,0200	4
0,7	0,070	0,0350	7
1	0,100	0,0500	10
3	0,300	0,1500	30
4,1	0,410	0,2050	41
5	0,500	0,2500	50
dzielnik			
22000	22000		
wzmocnienie			
1			

ATmega32			
prąd	bocznik	dzielnik	
A	V	V	ADC
0,01	0,001	0,0005	1
0,05	0,005	0,0025	5
0,1	0,010	0,0050	10
0,15	0,015	0,0075	15
0,2	0,020	0,0100	20
0,4	0,040	0,0200	40
0,7	0,070	0,0350	70
1	0,100	0,0500	100
3	0,300	0,1500	300
4,1	0,410	0,2050	410
5	0,500	0,2500	500
dzielnik			
22000	22000		
wzmocnienie			
10			

ATmega32			
prąd	bocznik	dzielnik	
A	V	V	ADC
0,01	0,001	0,0005	20
0,05	0,005	0,0025	100
0,1	0,010	0,0050	200
0,15	0,015	0,0075	300
0,2	0,020	0,0100	400
0,4	0,040	0,0200	800
0,7	0,070	0,0350	1400
1	0,100	0,0500	2000
3	0,300	0,1500	6000
4,1	0,410	0,2050	8200
5	0,500	0,2500	10000
dzielnik			
22000	22000		
wzmocnienie			
200			

Strona 253: poprawki jednostek w 2 tabelkach oraz wartości ostatniej kolumny w pierwszej tabeli

ATtiny26			
prąd	bocznik	dzielnik	
A	V	V	ADC
0,01	0,001	0,0005	0
0,05	0,005	0,0025	1
0,1	0,010	0,0050	2
0,15	0,015	0,0075	3
0,2	0,020	0,0100	4
0,4	0,040	0,0200	8
0,7	0,070	0,0350	14
1	0,100	0,0500	20
3	0,300	0,1500	60
4,1	0,410	0,2050	82
5	0,500	0,2500	100
dzielnik			
22000	22000		
wzmocnienie			
1			

ATtiny26			
prąd	bocznik	dzielnik	
A	V	V	ADC
0,01	0,001	0,0002	2
0,05	0,005	0,0012	10
0,1	0,010	0,0024	20
0,15	0,015	0,0037	29
0,2	0,020	0,0049	39
0,4	0,040	0,0098	78
0,7	0,070	0,0171	137
1	0,100	0,0244	196
3	0,300	0,0733	587
4,1	0,410	0,1002	802
5	0,500	0,1222	978
dzielnik			
68000	22000		
wzmocnienie			
20			

Strona 275 oraz 276: → **a także w kodzie źródłowym na płycie DVD**

Jest:

```
#define UART_DE_ODBIERANIE  UART_DE_PORT |= UART_DE_PIN
#define UART_DE_NADAWANIE  UART_DE_PORT &= ~UART_DE_PIN
```

Powinno być:

```
#define UART_DE_ODBIERANIE  UART_DE_PORT &= ~UART_DE_PIN
#define UART_DE_NADAWANIE  UART_DE_PORT |= UART_DE_PIN
```

Strona 279:

Jest:

> "Dzięki temu, że pin A0 zegarka podłączony jest do GND, adres Slave układu to: 0xA**2** szesnastkowo *(16**2** dziesiętnie)"*

Powinno być:

> "Dzięki temu, że pin A0 zegarka podłączony jest do GND, adres Slave układu to: 0xA**0** szesnastkowo *(16**0** dziesiętnie)"*. 0xA2 gdy pin A0 zwarty do VCC.

UWAGA! Istotny błąd!

Jest:

```
void TWI_stop(void) {
       TWCR = (1<<TWINT)|(1<<TWEN)|(1<<TWSTO);
       while ( !(TWCR&(1<<TWSTO)));

}
```

Powinno być:

```
void TWI_stop(void) {
       TWCR = (1<<TWINT)|(1<<TWEN)|(1<<TWSTO);
       while ( (TWCR&(1<<TWSTO))); // nie może być negacji

}
```

Strona 280:

Jest:

> 2. **TWI_stop()** – Początek analogicznie jak wyżej, tzn odblokowanie przerwania i TWI, a także ustawienie bitu TWSTO, co ma spowodować wygenerowanie przez część sprzętową prawidłowej sekwencji I2C STOP. W pętli while() oczekujemy na **ustawienie** bitu TWSTO, co będzie oznaczało prawidłowe zakończenie tejże operacji.

Powinno być:

> 2. **TWI_stop()** – Początek analogicznie jak wyżej, tzn odblokowanie przerwania i TWI, a także ustawienie bitu TWSTO, co ma spowodować wygenerowanie przez część sprzętową prawidłowej sekwencji I2C STOP. W pętli while() oczekujemy na **wyzerowanie** bitu TWSTO, co będzie oznaczało prawidłowe zakończenie tejże operacji.

Strona 292:

Jest:

> Natomiast gdy układ jest typu Slave/podrzędny, to wtedy linia ta jest Wyjściem zgodnie z drugim członem nazwy „Slave output"

Powinno być:

> „Natomiast gdy układ jest typu Slave/podrzędny, to wtedy linia ta (MOSI) jest Wejściem zgodnie z drugim członem nazwy „Slave Input"

Strona 303:

Jest:

```
        /* dokonujemy odczytu temperatury z pierwszego czujnika o ile został wykryty */
if(  DS18X20_OK  ==  DS18X20_read_meas(gSensorIDs[1],  &subzero,  &cel,  &cel_fract_bits)  )
display_temp(9);
```

Powinno być:

```
        /* dokonujemy odczytu temperatury z drugiego czujnika o ile został wykryty */
if( DS18X20_OK == DS18X20_read_meas(gSensorIDs[1], &subzero, &cel, &cel_fract_bits) )
display_temp(9);
```

Strona 323:

Jest:

PORTD &= !(T1|T2|T3|T4); /* *wyłączamy wszystkie tranzystory* */

Powinno być:

PORTD &= ~(T1|T2|T3|T4); /* *wyłączamy wszystkie tranzystory* */

Strona 422:

Jest:

„Kod potrzebny do utworzenia podstawowego serwera, będzie w odpowiedzi na zapytanie z przeglądarki wyświetli nieskomplikowanę stronę, jest niesamowicie prosty i krótki z udziałem najnowszej wersji implementującej stos TCP ze strony tuxgraphics.org.”

Powinno być:

„Kod potrzebny do utworzenia podstawowego serwera, który będzie w odpowiedzi na zapytanie z przeglądarki wyświetlał nieskomplikowaną stronę, jest niesamowicie prosty i krótki, z udziałem najnowszej wersji implementującej stos TCP ze strony tuxgraphics.org.”

Strona 422:

Jest:

„jeśli ztwój komputer”

Powinno być:

„jeśli twój komputer”