Listing 1. Sterowanie sprzętowym interfejsem SPI

void ConfigSpi**(**uint8\_t config**)** // 1- konfiguracja dla DDS, 2 - konfiguracja dla DAC

**{**

**switch(**config**)**

**{**

**case** 1**:**

SPCR **|=(**1**<<**CPOL**);**

SPCR **&=** **~(**1**<<**CPHA**);**

**break;**

**case** 2**:**

SPCR **&=~(**1**<<**CPOL**);**

SPCR **&=** **~(**1**<<**CPHA**);**

**break;**

\_delay\_us**(**20**);**

**}**

**}**

Listing 2. Wysłanie bajtu i słowa za pomocą SPI

void SendSpi**(** uint8\_t bajt **)**

**{**

SPDR **=** bajt**;**

**while(!(**SPSR **&** **(**1**<<**SPIF**)));**

**}**

void SendSpi16**(**uint16\_t word**)**

**{**

SPDR**=(**word**>>**8**);**

**while(!(**SPSR **&** **(**1**<<**SPIF**)));**

SPDR**=(**uint8\_t**)**word**;**

**while(!(**SPSR **&** **(**1**<<**SPIF**)));**

**}**

Listing 3. Funkcja wybierająca układ

void Select\_config**(**unsigned char select**)**

**{**

**if(**select**==**'a'**||**select**==**'b'**||**select**==**'c'**||**select**==**'k'**||**select**==**'l'**)**

**{**

**switch(**select**)**

**{**

**case** 'k'**:**

PORTA **|=** **(**1**<<**PA5**)** **|** **(**1**<<**PA6**)** **|** **(**1**<<**PA7**);**

PORTB **|=** **(**1**<<**PB0**)** **|** **(**1**<<**PB1**)** **|** **(**1**<<**PB2**);**

**break;**

**case** 'a'**:**

PORTA **&=** **~(**1**<<**PA5**);**

**break;**

**case** 'b'**:**

PORTA **&=** **~(**1**<<**PA6**);**

**break;**

**case** 'c'**:**

PORTA **&=** **~(**1**<<**PA7**);**

**break;**

**case** 'l'**:**

PORTA **&=** **~((**1**<<**PA5**)** **|** **(**1**<<**PA6**)** **|** **(**1**<<**PA7**));**

**break;**

**}**

**}**

**else** PORTB**&=~(**select**);**

\_delay\_us**(**20**);**

**}**

Listing 4. Ustawianie aplitudy sygnału wyjściowego

void SetAmplitude**(**uint8\_t ksztalt**,** unsigned char k**,** unsigned long int wartosc**)**

**{**

uint16\_t tempamp**=**0**;**

unsigned long int tempwartosc**=**0**;**

ConfigSpi**(**2**);**

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

tempamp**=(**1**<<**14**);** //Vref jest buforowane

tempamp**|=(**1**<<**13**);** //Vref nie jest mnozozne razy dwa

tempamp**|=(**1**<<**12**);** //wyjscie Vout zalaczone

tempwartosc**=**wartosc**\***819**;**

tempwartosc**=**tempwartosc**/**1000**;**

**if(**ksztalt**==**3**)**

**{**

tempwartosc**=**tempwartosc**\***wspolczynnikprostokat**;**

tempwartosc**=**tempwartosc**/**100**;**

**}**

tempamp**|=(**uint16\_t**)**tempwartosc**;** //przypisanie wartosci

Select\_config**(**k**);**

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

SendSpi16**(**tempamp**);**

Select\_config**(**'k'**);**

**}**

Listing 5. Struktura z parametrami DDS

**typedef** struct

**{**

uint8\_t sleep1**;**

uint8\_t sleep12**;**

unsigned char b13**;**

uint8\_t hlb**;**

uint8\_t chosefreq**;**

uint8\_t chosephase**;**

uint8\_t opbit**;**

uint8\_t div2**;**

uint8\_t mode**;**

uint8\_t reset**;**

**}** optionsdds**;**

Listing 6. Komendy generatora DDS

void CommandDDS**(**optionsdds **\***command**,** unsigned char k**)**

**{**

uint16\_t temp**=**0**;**

Select\_config**(**k**);**

**if(**command**->**b13**)** temp**|=**1**<<**13**;** **else** temp**&=~(**1**<<**13**);**

**if(**command**->**hlb**)** temp**|=**1**<<**12**;** **else** temp**&=~(**1**<<**12**);**

**if(**command**->**chosefreq**)** temp**|=**1**<<**11**;** **else** temp**&=~(**1**<<**11**);**

**if(**command**->**chosephase**)** temp**|=**1**<<**10**;** **else** temp**&=~(**1**<<**10**);**

**if(**command**->**reset**)** temp**|=**1**<<**8**;** **else** temp**&=~(**1**<<**8**);**

**if(**command**->**sleep1**)** temp**|=**1**<<**7**;** **else** temp**&=~(**1**<<**7**);**

**if(**command**->**sleep12**)** temp**|=**1**<<**6**;** **else** temp**&=~(**1**<<**6**);**

**if(**command**->**opbit**)** temp**|=**1**<<**5**;** **else** temp**&=~(**1**<<**5**);**

**if(**command**->**div2**)** temp**|=**1**<<**3**;** **else** temp**&=~(**1**<<**3**);**

**if(**command**->**mode**)** temp**|=**1**<<**1**;** **else** temp**&=~(**1**<<**1**);**

SendSpi16**(**temp**);**

Select\_config**(**'k'**);**

**}**

Listing 7. Wpisanie częstotliwości do rejestru FREQ0

void FreqDDS**(**uint8\_t rejestr**,** unsigned char k**,** unsigned long int wartosc**,** unsigned char przedrostek**)**

**{**

uint16\_t temp**=**0**;**

unsigned long int temp2**=**0**;**

Select\_config**(**k**);**

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

temp2**=**wartosc**\***1074**;**

temp2**=**temp2**/**100**;**

**if(**przedrostek**==**'M'**)** temp2**=(**temp2**<<**20**);** **else** **if(**przedrostek**==**'k'**)** temp2**=(**temp2**<<**10**);**

**if(!**rejestr**)** temp**|=(**1**<<**14**);** **else** temp**|=(**1**<<**15**);**

temp**|=(**uint16\_t**)(**temp2 **&** 0x3FFF**);**

SendSpi16**(**temp**);**

temp**&=(**0xC000**);**

temp**|=(**uint16\_t**)((**temp2**>>**14**)** **&** 0x3FFF**);**

SendSpi16**(**temp**);**

Select\_config**(**'k'**);**

**}**

Listing 8. Ustawienie akumulatora fazy DDS

//k - wybor do ktorego DDS ma byc wyslana dana, wartosfazy - wartisc fazy poczatkowej

void PhaseDDS**(**unsigned char k**,** uint16\_t wartoscfazy**,** uint8\_t rejest**)**

**{**

uint16\_t temp**;**

temp**=**wartoscfazy**\***117**;**

temp**=**temp**/**10**;**

temp**=(**0xfff**)** **&** temp**;**

**if(!**rejest**)** temp**|=(**0x0C**<<**12**);** **else** temp**|=(**0x0E**<<**12**);**

Select\_config**(**k**);**

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

SendSpi16**(**temp**);**

Select\_config**(**'k'**);**

**}**