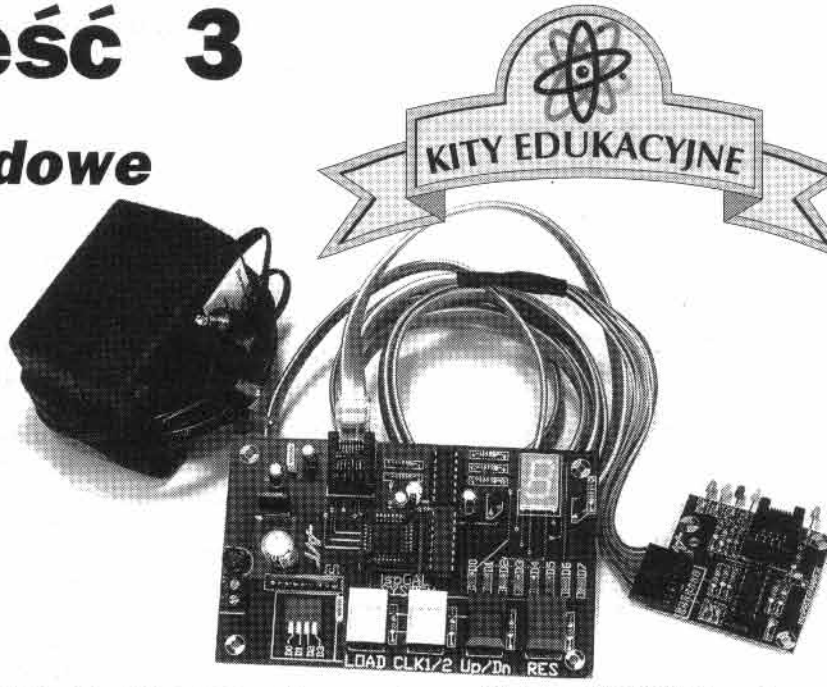
Starter Kit dla układów ispGAL, część 3

Programy przykładowe



W ostatniej części artykułu poświęconego Starter Kitowi ispGAL przedstawimy opisy programów przykładowych, które zostały wybrane spośród tych, które zamieszczono na dyskietce dołączanej do zestawu.

Obok programów źródłowych, napisanych w języku CUPL, na dyskietce znajdą się ich wersje skompilowane do postaci pliku download JEDEC, dzięki czemu można je od razu wykorzystać do testowania układu.

Programy przygotowane dla Starter Kitu można podzielić na dwie zasadnicze grupy:

- programy realizujące układy (automaty) synchroniczne,
- programy opisujące układy kombinacyjne.

Charakterystyczne dla wszystkich programów opisujących automaty synchroniczne jest wbudowanie w układ ispGAL22V10 multipleksera, dzięki któremu możliwe jest wybranie źródła synchronizującego przebiegu zegarowego. Jak wiadomo w Starter Kit wbudowano dwa generatory impulsów prostokątnych (w oparciu o układ NE556), o różnych częstotliwościach generowanych sygnałów. Dzięki zastosowaniu dwóch generatorów łatwo jest zaobserwować zmianę szybkości pracy testowanego układu. Wyboru wejścia multipleksera dokonuje się przy pomocy przycisku UP/DN, który zmienia stan logiczny na wejściu CLK_SEL US1.

Rozpoczniemy od omówienia bardzo prostego układu - dekodera dwójkowego.

```
NAME isp_d38;
REV 1.04;
DEVICE G22V10LCC

/*INPUTS*/
pin [9,7,5,6] = [I3..0];
pin 11 = LOAD;

/*OUTPUTS*/
pin 27 = SEL;
pin [25,24,23,21,20,19,18,17] = [D7..0];

/* DECLARATIONS AND INTERMEDIATE VARIABLE DEFINITIONS */
field DANA = [I3..0];
field SEGMENT = [D7..0];

/*LOGIC EQUATIONS*/
SEL = LOAD;

table DANA => SEGMENT {

/*  Wejscia      Wyjscia segmentowe              */
/*  ------------------      ------------------  */
/*  IIII         DDDDDDDD                        */
/*  3210         76543210                        */
/*                                               */
  'b'0000   =>  'b'00000001; /* 0                */
  'b'0001   =>  'b'00000010; /* 1                */
  'b'0010   =>  'b'00000100; /* 2                */
  'b'0011   =>  'b'00001000; /* 3                */
  'b'0100   =>  'b'00010000; /* 4                */
  'b'0101   =>  'b'00100000; /* 5                */
  'b'0110   =>  'b'01000000; /* 6                */
  'b'0111   =>  'b'10000000; /* 7                */
  'b'1000   =>  'b'11111110; /* 8                */
  'b'1001   =>  'b'11111101; /* 9                */
  'b'1010   =>  'b'11111011; /* A                */
  'b'1011   =>  'b'11110111; /* B                */
  'b'1100   =>  'b'11101111; /* C                */
  'b'1101   =>  'b'11011111; /* D                */
  'b'1110   =>  'b'10111111; /* E                */
  'b'1111   =>  'b'01111111; /* F                */
}
```

Listing 1.

Dekoder 3->8 linii

Na **list.1** zamieszczono program opisujący działanie dekodera dwójkowego o czterech wejściach informacyjnych I3..0 oraz ośmiu wyjściach D7..0. W zależności od stanu wejść I2..0 dekoder na wyjściach zapala wybraną przy pomocy DIP-switcha diodę LED. Dzieje się tak w przypadku, gdy na wejściu I3 utrzymuje się stan „0". Jeżeli zmienimy stan tego wejścia na „1" wyjścia dekodera są negowane, dzięki czemu dioda o numerze wybranym przy pomocy I2..0 zostaje zgaszona (pozostałe świecą).

```
NAME isp_dech;
REV 1.23;
DEVICE G22V10LCC

/*INPUTS*/
pin [9,7,5,6] = [D3..0];

/*OUTPUTS*/
pin 27 = SEL;
pin [24,23,21,20,19,18,17] = ![G,F,E,D,C,B,A];

/* DECLARATIONS AND INTERMEDIATE VARIABLE DEFINITIONS */
field DANA = [D3..0];
field SEGMENT = [A,B,C,D,E,F,G,SEL];

/*LOGIC EQUATIONS*/
table DANA => SEGMENT {

/*  Wejscia      Wyjscia segmentowe              */
/*  ---------------------------                  */
/*  DDDD          ABCDEFGS                       */
/*  3210                E                        */
/*                      L                        */
 'b'0000   =>  'b'00000011; /* 0                 */
 'b'0001   =>  'b'10011111; /* 1                 */
 'b'0010   =>  'b'00100101; /* 2                 */
 'b'0011   =>  'b'00001101; /* 3                 */
 'b'0100   =>  'b'10011001; /* 4                 */
 'b'0101   =>  'b'01001001; /* 5                 */
 'b'0110   =>  'b'01000001; /* 6                 */
 'b'0111   =>  'b'00011111; /* 7                 */
 'b'1000   =>  'b'00000001; /* 8                 */
 'b'1001   =>  'b'00001001; /* 9                 */
 'b'1010   =>  'b'00010001; /* A                 */
 'b'1011   =>  'b'11000001; /* B                 */
 'b'1100   =>  'b'01100011; /* C                 */
 'b'1101   =>  'b'10000101; /* D                 */
 'b'1110   =>  'b'01100001; /* E                 */
 'b'1111   =>  'b'01110001; /* F                 */
}
```

Listing 2.

Wyjście SEL, które odpowiada za wybór wskaźnika wykorzystywanego podczas prezentacji (diody LED lub wyświetlacz) sterowane jest przez przełącznik LOAD, co pozwala zmienić sposób wyświetlania podczas pracy układu.

```
NAME isp_cndc;
REV 1.11;
DEVICE G22V10LCC;

/*INPUTS*/
pin 2 = CLK;
pin 3 = CLK1;
pin 4 = CLK2;
pin 13 = CLK_SEL;
pin 16 = RES;

/*OUTPUTS*/
pin 26 = CLK_OUT;
pin 27 = SEL;
pin [17,18,19,20,21,23,24,25] = [Q7..0];

/* LOGIC EQUATIONS */
CLK_OUT = !CLK_SEL & CLK1
        # CLK_SEL & CLK2;

SEL = 'b'0;

[Q0..7].ar = RES;
[Q0..7].sp = 'b'0;

field COUNT = [Q7..0];
$define S0 'b'00000000
$define S1 'b'00000001
$define S2 'b'00000010
$define S3 'b'00000100
$define S4 'b'00001000
$define S5 'b'00010000
$define S6 'b'00100000
$define S7 'b'01000000
$define S8 'b'10000000

sequence COUNT {
present S0 next S1;
present S1 next S2;
present S2 next S3;
present S3 next S4;
present S4 next S5;
present S5 next S6;
present S6 next S7;
present S7 next S8;
present S8 next S1;
}
```

Listing 3.

Dekoder szesnastkowy

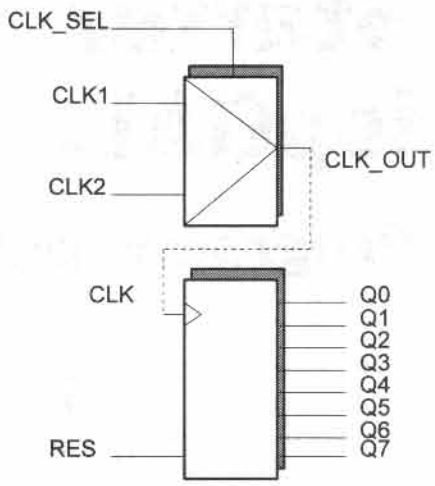
Nieco inaczej działa program przedstawiony na **list.2**. Wejścia oznaczone D3..0 są wejściami układu dekodera. W zależności od stanu tych wejść na wyjściach segmentowych A..G pojawiają się kombinacje stanów, które powodują zapalenie lub zgaszenie segmentów. W tabeli prawdy na list.1 przyjęto, że segment jest zapalony gdy zadeklarujemy „0". Ponieważ zastosowano w układzie wyświetlacz ze wspólną katodą zapalenie segmentu następuje po zasileniu elektrody segmentu stanem „1". Z tego właśnie powodu przy deklaracji sygnałów wyjściowych przed nawiasem z wypisanymi segmentami A..G postawiono znak logicznej negacji „!". Pole wyjściowe (SEGMENT) jest 8-bitowe, co może wydawać się dziwne - przecież wyświetlacz LED ma siedem segmentów! Otóż w polu SEGMENT zadeklarowano jako ósmy bit wyjście SEL, który jest odpowiedzialny za wybór wskaźnika (wyświetlacz LED lub diody). Jak łatwo zauważyć w zadeklarowanej tabeli wyjście SEL znajduje się w stanie „1", powodując ciągłe świecenie wyświetlacza.

Dekoder skonstruowano w taki sposób, że po zadaniu przy pomocy DIP-switcha liczby z zakresu 0..15 (zakodowanej dwójkowo) na wskaźniku zaświeca się znak odpowiadający tej liczbie w postaci standardowych kodów 0..F.

Licznik Johnsona

Kolejnym przykładem jest bardzo prosty, 8-bitowy licznik pierścieniowy z asynchronicznym kasowaniem. **Listing 3** przedstawia program opisujący działanie tego układu.

Wyjście CLK_OUT jest wyjściem dwuwejściowego multipleksera, dzięki któremu możliwy jest wybór częstotliwości taktowania licznika. Wyboru częstotliwości dokonuje się poprzez zmianę stanu logicznego na wejściu CLK_SEL. Na **rys.1** przedstawiono schemat blokowy układu skonfi-



Rys. 1. Schemat blokowy licznika z list. 3.

gurowanego zgodnie z programem przedstawionym na list.3. Na rysunku pominięto wyjście SEL.

Licznik pracuje w zamkniętym cyklu, sygnalizując w kodzie mod.10 ilość zliczonych impul-

```
NAME isp_cntw;
REV 1.02;
DEVICE G22V10LCC;

/*INPUTS*/
pin 2 = CLK;
pin 3 = CLK1;
pin 4 = CLK2;
pin 12 = UD;
pin 13 = CLK_SEL;
pin 16 = !RES;

/*OUTPUTS*/
pin 26 = CLK_OUT;
pin 27 = SEL;
pin [24,23,21,20,19,18,17] = [G,F,E,D,C,B,A];
pin 25 = DP;

/* LOGIC EQUATIONS */

CLK_OUT = !CLK_SEL & CLK1
        # CLK_SEL & CLK2;
DP = CLK;
SEL = 'b'1;
[G,F,E,D,C,B,A].ar = RES;
[G,F,E,D,C,B,A].sp = 'b'0;
field COUNT = [G,F,E,D,C,B,A];

$define S0 'b'0111111
$define S1 'b'0000110
$define S2 'b'1011011
$define S3 'b'1001111
$define S4 'b'1100110
$define S5 'b'1101101
$define S6 'b'1111101
$define S7 'b'0000111
$define S8 'b'1111111
$define S9 'b'1101111

sequence COUNT {
present S0 if UD next S1;
           if !UD next S9;
present S1 if UD next S2;
           if !UD next S0;
present S2 if UD next S3;
           if !UD next S1;
present S3 if UD next S4;
           if !UD next S2;
present S4 if UD next S5;
           if !UD next S3;
present S5 if UD next S6;
           if !UD next S4;
present S6 if UD next S7;
           if !UD next S5;
present S7 if UD next S8;
           if !UD next S6;
present S8 if UD next S9;
           if !UD next S7;
present S9 if UD next S0;
           if !UD next S8;
}
```

Listing 4.

sów zegarowych. Stanem aktywnym każdego z wyjść jest „1", której pojawienie powoduje zapalenie odpowiedniej diody LED. W programie zadeklarowano, że wyjście SEL ma mieć na stałe przyporządkowane „0" logiczne, co powoduje świecenie diod LED.

Asynchroniczne kasowanie jest możliwe dzięki wykorzystaniu wejść RESET makrokomórek i podłączenie ich do przycisku RESET, co uzyskano poprzez deklarację z rozszerzeniem *.ar (list.3).

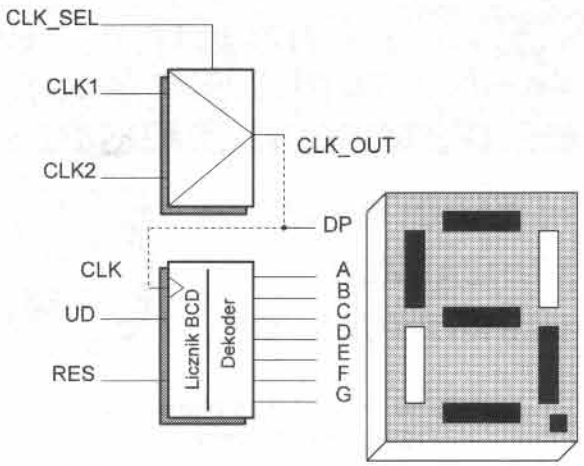
Program opisuje automat synchroniczny przy pomocy deklaracji przejść pomiędzy poszczególnymi stanami, przy czym stan S0 wykorzystywany jest w celu umożliwienia wystartowania układu (czyli po włączeniu zasilania lub ręcznym wyzerowaniu). Po jednokrotnym przejściu stanu S0 licznik pracuje w cyklu S1..8.

Licznik BCD - dekoder dziesiętny

Nieco bardziej zaawansowane rozwiązanie przedstawia **list.4**. W układ programowalny „wbudujemy" tym razem licznik dziesiętny z dekoderem sterującym wyświetlacz 7-segmentowy. Funkcjonalność tego licznika podnosi wejście UD (z ang. Up-Down), przy pomocy którego możliwa jest zmiana kierunku zliczania. Liczenie „w górę" polega na zwiększaniu wartości liczby po każdym kolejnym takcie zegara (0, 1, 2....9), natomiast liczenie „w dół" oznacza zmniejszanie wartości wskazywanej liczby po kolejnym takcie zegara.

Licznik wyposażony jest w wejście asynchronicznego zerowania RES, co pozwala na ustalenie stanu początkowego w dowolnie wybranym momencie zliczania.

Wyjścia A..G sterują poszczególnymi segmentami wyświetlacza 7-segmentowego (włączonego dzięki logicznej „1" na wyjściu SEL), a dodatkowo wyjście DP podłączono do wyjścia multipleksera selekcji sygnału zegarowego. Dzięki zastosowaniu tego rozwiązania migająca kropka wyświetlacza sygnalizuje obecność lub brak sygnału zegarowego.



Rys. 2. Schemat blokowy licznika - dekodera z list. 4.

Uwagi końcowe

Wszystkie programy dołączane do zestawu kompilowano do postaci JEDEC przy pomocy programu CUPL. Podczas kompilacji nie są wymagane żadne dodatkowe parametry co oznacza, że każdy z przedstawionych programów „mieści" się w układzie ispGAL22V10 bez konieczności dodatkowej minimalizacji.

W artykule przedstawiliśmy skrótowo tylko cztery programy przykładowe. Nie są to wszystkie programy, jakie przygotowaliśmy dla zestawu Starter Kit. Kompletny zestaw oprogramowania wchodzi w skład kitu.

Piotr Zbysiński, AVT