Układy FPGA w przykładach, część 2

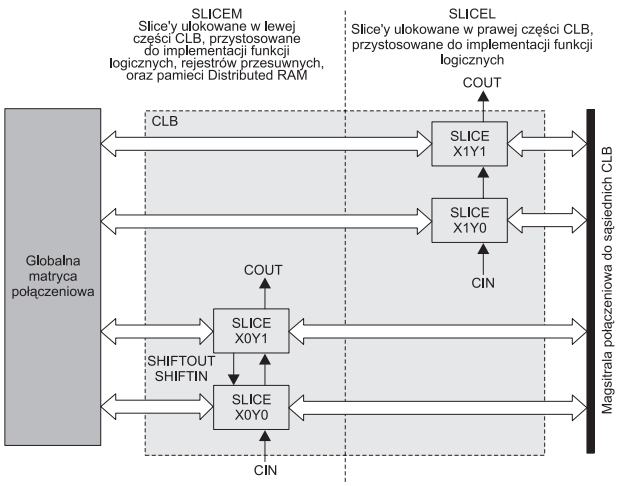
W drugiej części artykułu zajmiemy się omówieniem wyposażenia (po „mikrokontrolerowemu": peryferiów) układów FPGA z rodziny Spartan 3, co ułatwi ich wykorzystywanie w praktyce. To właśnie wewnętrzne zespoły konfigurowalnych pamięci, uniwersalne porty I/O, wbudowane syntezery sygnałów zegarowych, sprzętowe zespoły mnożące i pozostałe – mniej spektakularne elementy – tworzą potęgę możliwości współczesnych FPGA.

Już najprostsze układy FPGA z rodziny Spartan 3 oferują użytkownikom duże zasoby logiczne i bardzo bogate wyposażenie dodatkowe, charakteryzujące się dużą elastycznością i uniwersalnością. Podstawowe informacje na temat zasobów dostępnych w tych układach zebrano w **tab. 3**.

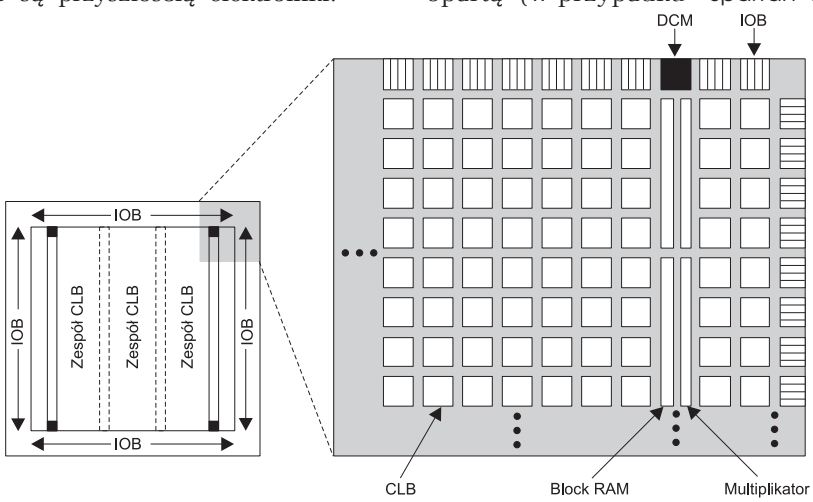
To właśnie dzięki bogatemu wyposażeniu wewnętrznemu układy FPGA są coraz częściej stosowane jako platformy *System–on–a–Chip*, w których są implementowane kompletne systemy cyfrowe łącznie z „miękkimi" rdzeniami mikroprocesorowymi. A to właśnie układy typu SoC są przyszłością elektroniki.

Budowa układów Spartan 3

Układy FPGA (*Field Programmable Gate Array*) to jeden z dwóch (drugi to układy mniejszej skali integracji – CPLD, czyli *Complex Programmable Logic Devices*), produkowanych obecnie, rodzajów układów PLD (*Programmable Logic Devices*). Układy te charakteryzują się regularną budową, opartą (w przypadku firmy Xilinx) na zespołach wielu identycznych lub bardzo do siebie podobnych bloków CLB (*Configurable Logic Block*). Schemat blo-



Rys. 8. Budowa komórki logicznej CLB w układach Spartan 3



Rys. 7. Schemat blokowy ilustrujący budowę układów Spartan 3

FPGA – co trzeba o nich wiedzieć – tip 5
Bezpieczeństwo projektów w FPGA
Układy Spartan 3 nie są wyposażone w zaawansowane mechanizmy ochrony konfiguracji, co powoduje, że projekty zagrożone przez „piratów" powinny być implementowane na bardziej zaawansowanych pod tym względem układach FPGA. Pamięć konfigurująca Flash XCF01S (i inne z serii xxS) jest zabezpieczona przed nieuprawnionym odczytem przez JTAG, ale w żaden sposób nie są chronione dane przesyłane interfejsem szeregowym wykorzystywanym do konfigurowania FPGA.

List. 1. Opis VHDL dwuportowej pamięci 16 x N (na bazie XAPP464)

```
--
-- Module:   XC3S_RAM16XN_D
--
-- Description: Distributed SelectRAM example
--      Dual Port 16 x N-bit
--      Use template „RAM_16D.vhd”
--      and registered outputs (optional)
--
-- Device:   Spartan-3 Family
----------------------------------------------------------------------------
library IEEE;
use IEEE.std_logic_1164.all;
--
-- pragma translate_off
library UNISIM;
use UNISIM.VCOMPONENTS.ALL;
-- pragma translate_on
--
entity XC3S_RAM16XN_D is
  generic (
     data_width : integer := 8 -- Replace by the data width
     );
  port (
     DATA_IN    : in std_logic_vector(data_width-1 downto 0);
         ADDRESS      : in std_logic_vector(3 downto 0);
         ADDRESS_DP   : in std_logic_vector(3 downto 0);
         WRITE_EN       : in std_logic;
         CLK       : in std_logic;
         O_DATA_OUT   : out std_logic_vector(data_width-1 downto 0);
         O_DATA_OUT_DP  : out std_logic_vector(data_width-1 downto 0)
        );
end XC3S_RAM16XN_D;
--
architecture XC3S_RAM16XN_D_arch of XC3S_RAM16XN_D is
--
-- Components Declarations:
--
component RAM16X1D
-- See initialization example in the reference templates
  port (
     D    : in std_logic;
         WE   : in std_logic;
         WCLK : in std_logic;
         A0   : in std_logic;
         A1   : in std_logic;
         A2   : in std_logic;
         A3   : in std_logic;
         DPRA0   : in std_logic;
         DPRA1   : in std_logic;
         DPRA2   : in std_logic;
         DPRA3   : in std_logic;
         SPO  : out std_logic;
         DPO  : out std_logic
  );
end component;
--
----------------------------------------------------------------------------
--
-- Signal Declarations:
signal DATA_OUT : std_logic_vector(data_width-1 downto 0);
signal DATA_OUT_DP : std_logic_vector(data_width-1 downto 0);
--
begin
--
-- Registered outputs / Synchronous read
REGISTERED_OUT: process (CLK)
begin
  if (CLK’event and CLK = ,1’) then
     O_DATA_OUT <= DATA_OUT;
     O_DATA_OUT_DP <= DATA_OUT_DP;
  end if;
end process REGISTERED_OUT;
--
-- Distributed SelectRAM Instantiation
RAM16X1D_X: for i in 0 to data_width-1 generate
U_RAM16X1D: RAM16X1D
  port map (
  D      => DATA_IN(i), -- insert input signal
  WE     => WRITE_EN, -- insert Write Enable signal
  WCLK   => CLK, -- insert Write Clock signal
  A0     => ADDRESS(0), -- insert Address 0 signal port SPO
  A1     => ADDRESS(1), -- insert Address 1 signal port SPO
  A2     => ADDRESS(2), -- insert Address 2 signal port SPO
  A3     => ADDRESS(3), -- insert Address 3 signal port SPO
  DPRA0  => ADDRESS_DP(0), -- insert Address 0 signal port DPO
  DPRA1  => ADDRESS_DP(1), -- insert Address 1 signal port DPO
  DPRA2  => ADDRESS_DP(2), -- insert Address 2 signal port DPO
  DPRA3  => ADDRESS_DP(3), -- insert Address 3 signal port DPO
  SPO    => DATA_OUT(i),  -- insert output signal SPO
  DPO    => DATA_OUT_DP(i) -- insert output signal DPO
  );
end generate;
--
end XC3S_RAM16XN_D_arch;
----------------------------------------------------------------------------
```

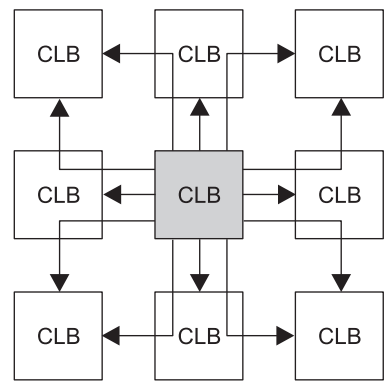
Plan kursu

1. Wprowadzenie
   - Budowa zestawu uruchomieniowego
   - Programowanie i konfiguracja układu XC3S200
   - Tryby konfiguracji układu XC3S200
   - Zasilanie układu XC3S200
   - Linie I/O w układzie XC3S200
   - JTAG jako uniwersalny interfejs do programowania i konfigurowania
2. **Budowa, cechy funkcjonalne i parametry układów FPGA z rodziny Spartan 3**
   - **CLB**
   - IOB
   - Globalne sygnały zegarowe
   - DCM
   - Sprzętowe multiplikatory
   - Pamięć BlockRAM
3. Projekty przykładowe

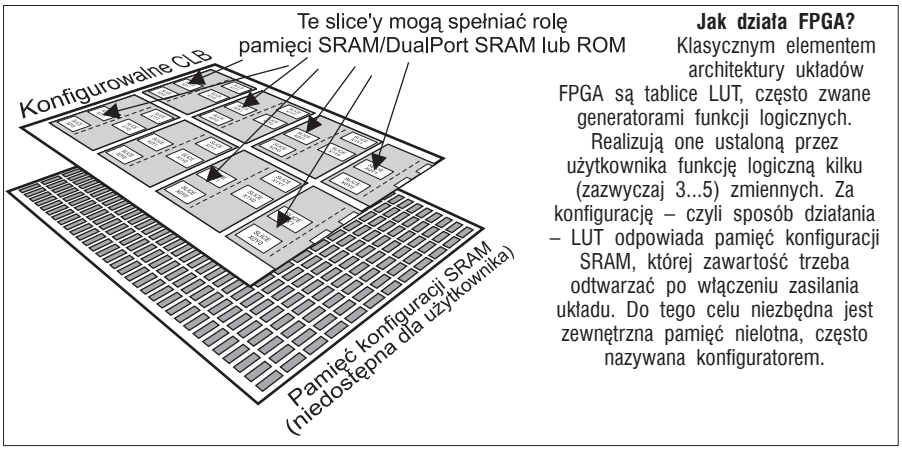
kowy układu z rodziny Spartan 3 pokazano na **rys. 7**. Na schemacie – poza CLB – widać także pamięci *BlockRAM*, sprzętowe multiplikatory, syntezery przebiegów zegarowych DCM (*Digital Clock Manager*) oraz komórki I/O o nazwie IOB (*Input–Output Block*). Kolejno je omówimy.

CLB

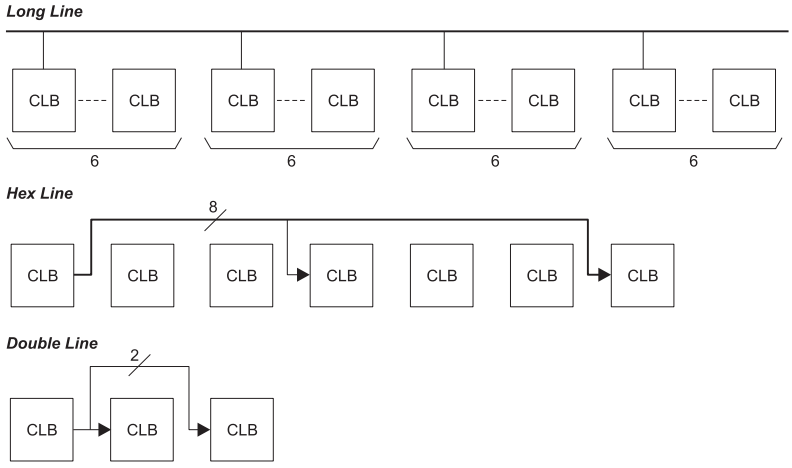
Bloki CLB (w układach Spartan 3 jest ich od 1728 do 74880 sztuk) są zbudowane z 4 bloków logicznych nazwanych przez firmę Xilinx mianem *slice*. Schemat ilustrujący rozmieszczenie *slice’ów* w CLB pokazano na **rys. 8**. Jak widać, od jednej strony *slice’y* są



Rys. 9. Każde ulokowane wewnątrz matrycy CLB może bezpośrednio komunikować się z 8 sąsiadującymi CLB



Jak działa FPGA?
Klasycznym elementem architektury układów FPGA są tablice LUT, często zwane generatorami funkcji logicznych. Realizują one ustaloną przez użytkownika funkcję logiczną kilku (zazwyczaj 3...5) zmiennych. Za konfigurację – czyli sposób działania – LUT odpowiada pamięć konfiguracji SRAM, której zawartość trzeba odtwarzać po włączeniu zasilania układu. Do tego celu niezbędna jest zewnętrzna pamięć nielotna, często nazywana konfiguratorem.



Rys. 10. Wymianę sygnałów w układach Spartan 3 zapewniają rozbudowane zasoby połączeniowe o różnych cechach

dołączone do magistral zapewniających komunikację w obrębie całego układu FPGA (tzw. połączenia globalne o różnym zasięgu), od drugiej strony – do magistral zapewniających komunikację lokalną z sąsiednimi CLB. Każdy *slice* ma własny adres w obrębie CLB (np. X1Y0), który projektant może wykorzystać wraz z numerem CLB

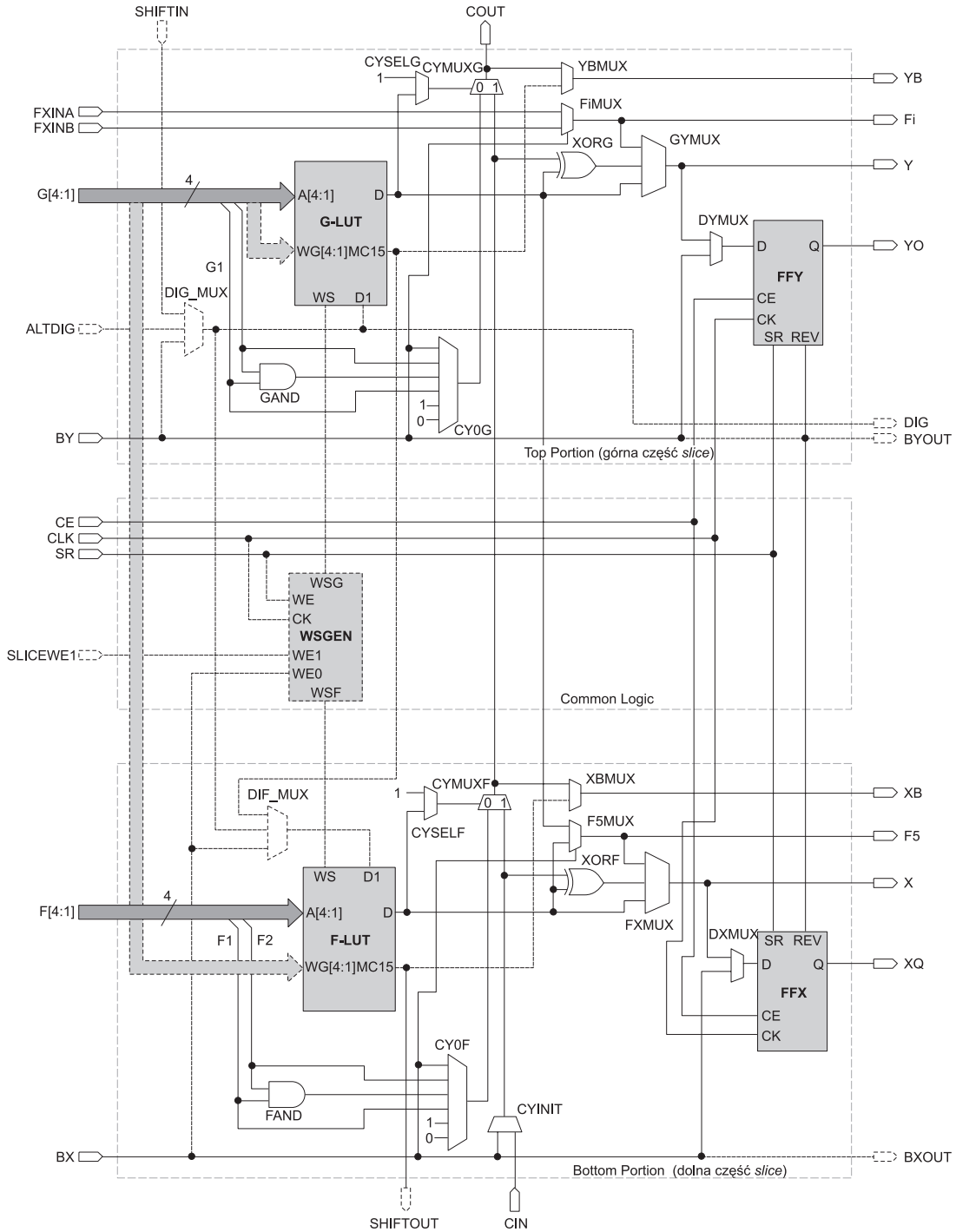
FPGA – co trzeba o nich wiedzieć – tip 6
Alternatywne możliwości CLB
Bloki CLB, będące podstawowym konfigurowalnym elementem logicznym w układach Spartan 3, mogą spełniać także dodatkowe funkcje: rejestrów przesuwnych o regulowanej długości oraz pamięci ROM, SRAM i DualPort SRAM. Pojemności tej pamięci w układach Spartan 3 mieszczą się w przedziale 12…520 kb.

w przypadku konieczności ręcznego (rzadko się to obecnie zdarza) rozmieszczania bloków funkcjonalnych w obrębie FPGA. *Slice* pogrupowano je po dwa (w kolumny) z wydzielonymi szybkimi łańcuchami propagacji sygnału CARRY, dzięki czemu możliwe są implementacje szybko działających bloków logicznych wykorzystujących kaskadowe przeniesienia (liczniki, arytmometry itp.). Jak wspomniano, każdy CLB ulokowany wewnątrz grupy ma możliwość bezpośredniej komunikacji z sąsiadującymi CLB, jest ich zazwyczaj 8 (**rys. 9**). Wymiana danych z dalej położonymi CLB odbywa się za pomocą dodatkowych zasobów połączeniowych (**rys. 10**):

- linii długich (*Long Lines*, dołączonych do – kolejno – co szóstego CLB), które są najszybszym traktem komunikacyjnym wewnątrz FPGA, często wykorzystywanym zamiennie z globalnymi liniami zegarowymi,
- linii 8–krotnych (*Hex Lines*), które rozprowadzają sygnały na mniejsze odległości, oferując większe (niż *Long Lines*) moż-

Tab. 3. Zestawienie najważniejszych parametrów układów z rodziny Spartan–3

Parametr	Typ układu							
	XC3S50	XC3S200	XC3S400	XC3S1000	XC3S1500	XC3S2000	XC3S4000	XC3S5000
Liczba bramek przeliczeniowych	50000	200000	400000	1000000	1500000	2000000	4000000	5000000
Liczba komórek logicznych	1728	4320	8064	17280	29952	46080	62208	74880
Sprzętowe multiplikatory	4	12	16	24	32	40	96	104
Pojemność pamięci Block RAM	72 kb	216 kb	288 kb	432 kb	576 kb	720 kb	1728 kb	1872 kb
Pojemność pamięci rozproszonej Distributed RAM	12 kb	30 kb	56 kb	120 kb	208 kb	320 kb	432 kb	520 kb
Liczba DCM	2	4	4	4	4	4	4	4
Maksymalna liczba różnicowych linii I/O	56	76	116	175	221	270	312	344
Maksymalna liczba asymetrycznych linii I/O	124	173	264	391	487	565	712	784



Rys. 11. Budowa *slice'a*

List. 2. Opis VHDL dwuportowej pamięci 16 x N (na bazie XAPP464)

```
--
-- Module:   XC3S_RAM16XN_S
--
-- Description: Distributed SelectRAM example
--      Single Port 16 x N-bit
--      Use template „RAM_16S.vhd"
--      and registered outputs (optional)
--
-- Device:   Spartan-3 Family
--------------------------------------------------------------------------------
library IEEE;
use IEEE.std_logic_1164.all;
--
-- pragma translate_off
library UNISIM;
use UNISIM.VCOMPONENTS.ALL;
-- pragma translate_on
--
entity XC3S_RAM16XN_S is
  generic (
     data_width : integer := 8 -- Replace by the data width
     );
  port (
     DATA_IN    : in std_logic_vector(data_width-1 downto 0);
         ADDRESS      : in std_logic_vector(3 downto 0);
         WRITE_EN       : in std_logic;
         CLK       : in std_logic;
         O_DATA_OUT   : out std_logic_vector(data_width-1 downto 0)
        );
end XC3S_RAM16XN_S;
--
architecture XC3S_RAM16XN_S_arch of XC3S_RAM16XN_S is
--
-- Components Declarations:
--
component RAM16X1S
-- See initialization example in the reference templates
  port (
     D     : in std_logic;
         WE    : in std_logic;
         WCLK  : in std_logic;
         A0    : in std_logic;
         A1    : in std_logic;
         A2    : in std_logic;
         A3    : in std_logic;
         O     : out std_logic
  );
end component;
--
--------------------------------------------------------------------------------
--
-- Signal Declarations:
signal DATA_OUT : std_logic_vector(data_width-1 downto 0);
--
begin
--
-- Registered outputs / Synchronous read
REGISTERED_OUT: process (CLK)
begin
  if (CLK'event and CLK = ,1') then
     O_DATA_OUT <= DATA_OUT;
  end if;
end process REGISTERED_OUT;
--
-- Distributed SelectRAM Instantiation
RAM16X1S_X: for i in 0 to data_width-1 generate
U_RAM16X1S: RAM16X1S
  port map (
  D       => DATA_IN(i), -- insert input signal
  WE      => WRITE_EN, -- insert Write Enable signal
  WCLK    => CLK, -- insert Write Clock signal
  A0      => ADDRESS(0), -- insert Address 0 signal
  A1      => ADDRESS(1), -- insert Address 1 signal
  A2      => ADDRESS(2), -- insert Address 2 signal
  A3      => ADDRESS(3), -- insert Address 3 signal
  O       => DATA_OUT(i)  -- insert output signal
  );
end generate;
--
end XC3S_RAM16XN_S_arch;
--------------------------------------------------------------------------------
```

liwości połączeniowe i są dołączone do co trzeciego CLB,

– linii podwójnych (*Double Lines*), które zapewniają bezpośrednią komunikację pomiędzy pozostałymi CLB.

W *slice'ach* tworzących CLB ulokowano zasoby logiczne, których nawet pobieżna analiza (choćby na schemacie pokazanym na **rys. 11**) pokazuje ogrom możliwości i elastyczność tych komórek, których – ogólnie rzecz ujmując – budowa jest taka sama jak w pierwszych układach FPGA. Każdy *slice* wyposażono w dwie konfigurowalne tablice LUT (F–LUT i G–LUT), na wejścia których są podawane 4 sygnały (zmienne). Tablice te spełniają rolę konfigurowalnych, kombinacyjnych funktorów logicznych (często są nazywane generatorami funkcji), które umożliwiają wykonanie dowolnej funkcji logicznej do 4 zmiennych wejściowych. Na wyjściu LUT ulokowano przerzutnik, którego sposób działania (czyli jego typ) można także skonfigurować. Na schemacie pokazanym na rys. 11 zilustrowano budowę *slice'y* X0Y1 i X0Y0 (rys. 8), które wyposażono w sprzętowe rozszerzenia (zaznaczone na rys. 11 linią przerywaną) pozwalające skonfigurować je jako rejestry przesuwne lub zespoły rozproszonej pamięci (tzw. *Distributed RAM*). *Slice'y* X1Y0 i X1Y1 mają nieco prostszą budowę (bez fragmentów oznaczonych liniami przerywanymi na rys. 11), co ogranicza ich funkcjonalność do znanej z klasycznych wersji FPGA. O ile – w większości przypadków – możliwość wygodnej implementacji rejestrów przesuwających nie budzi specjalnych emocji, to możliwość uzyskania dodatkowych zasobów pamięciowych w LUT bywa atutem nie do pogardzenia. W każdym CLB można zaimplementować pamięć ROM o pojemności do 128x1 bitów, pamięć SRAM o pojemności do 64x1 bitów (co oznacza, że możliwe są także warianty 2x32x1 lub 4x16x1) lub pamięć DualPortRAM o pojemności 2x16x1 bit. Pamięci te – dzięki rejestrom na wyjściach CLB – można wyposażyć w mechanizmy synchronizacji odczytu danych.

Na **list. 1** przedstawiono opis w języku VHDL dwuportowej pamięci SRAM implementowanej w zasobach *Distributed RAM*, a na **list. 2** opis pamięci jednoportowej (obydwie o organizacji 16xN, w obydwu zastosowano synchroniczny odczyt danych). Prezentowane przykłady pochodzą z przykładów przygotowanych przez inżynierów firmy Xilinx do noty aplikacyjnej XAPP464 (publikujemy na CD–EP11/2006B).

Jacek Majewski
jacek.majewski@pwr.wroc.pl
Piotr Zbysiński, EP
piotr.zbysinski@ep.com.pl