KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

# „Układy reprogramowalne i SoC”
# „Specjalizowane moduły FPGA”

Prezentacja jest współfinansowana przez
Unię Europejską w ramach
Europejskiego Funduszu Społecznego w projekcie pt.

*„Innowacyjna dydaktyka bez ograniczeń - zintegrowany rozwój Politechniki Łódzkiej - zarządzanie Uczelnią, nowoczesna oferta edukacyjna i wzmacniania zdolności do zatrudniania osób niepełnosprawnych”*

Prezentacja dystrybuowana jest bezpłatnie

# Specjalizowane moduły układów Spartan 3

- Oprócz układów CLB, pozwalających na implementację układów logicznych ogólnego przeznaczenia, rodzina układów Spartan 3 zawiera również zasoby specjalizowane:
  - Digital Clock Managers (DCMs)
  - Block RAM
  - Układy mnożące 18x18 bit
- Pozwalają one na implementację wybranych funkcji w bardziej wydajny sposób, niż za pomocą CLB

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Projekt współfinansowany przez Unię Europejską
w ramach Europejskiego Funduszu Społecznego

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

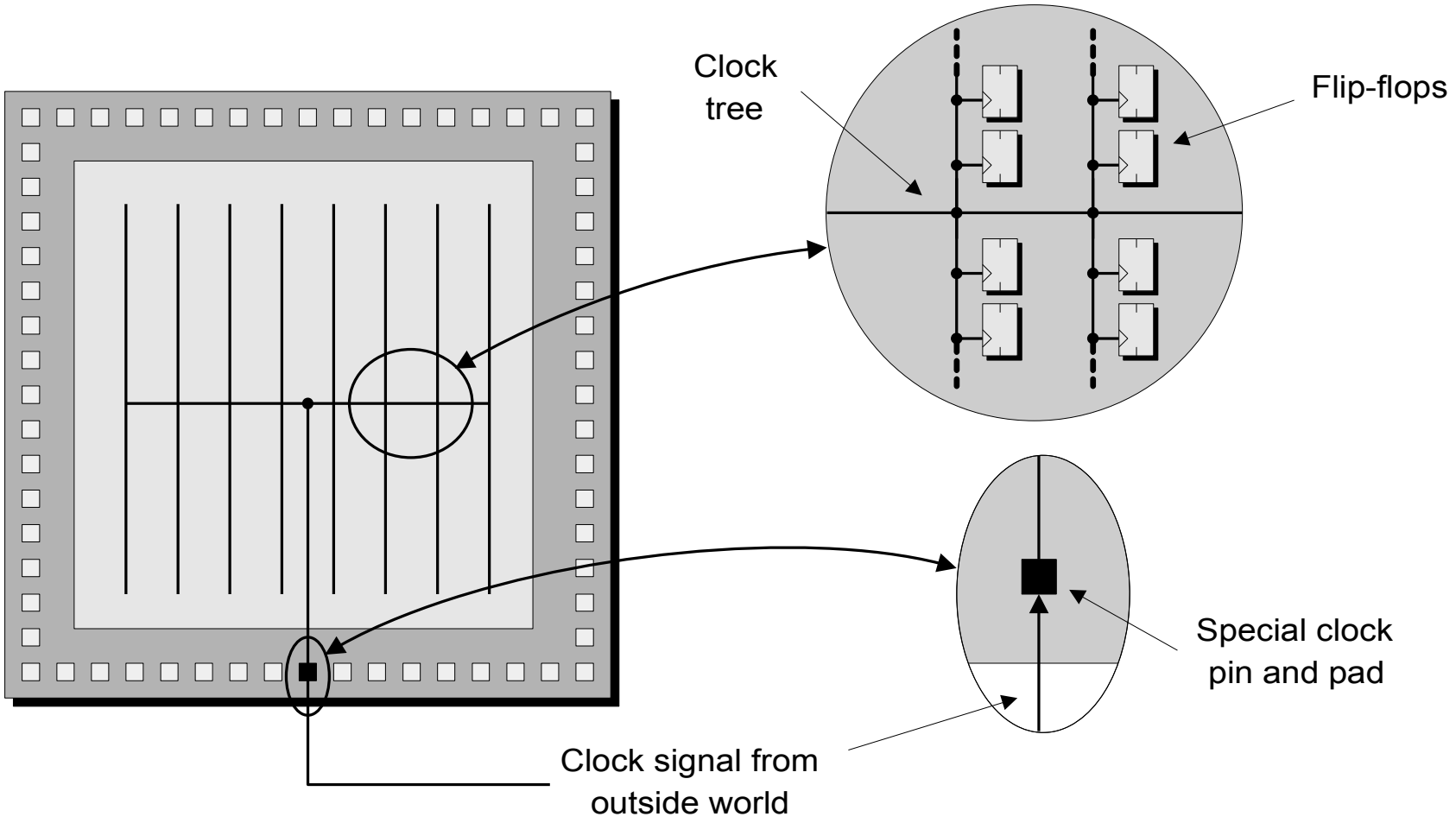
# Sygnały zegarowe i ich dystrybucja

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Projekt współfinansowany przez Unię Europejską
w ramach Europejskiego Funduszu Społecznego

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

# Drzewo zegarowe

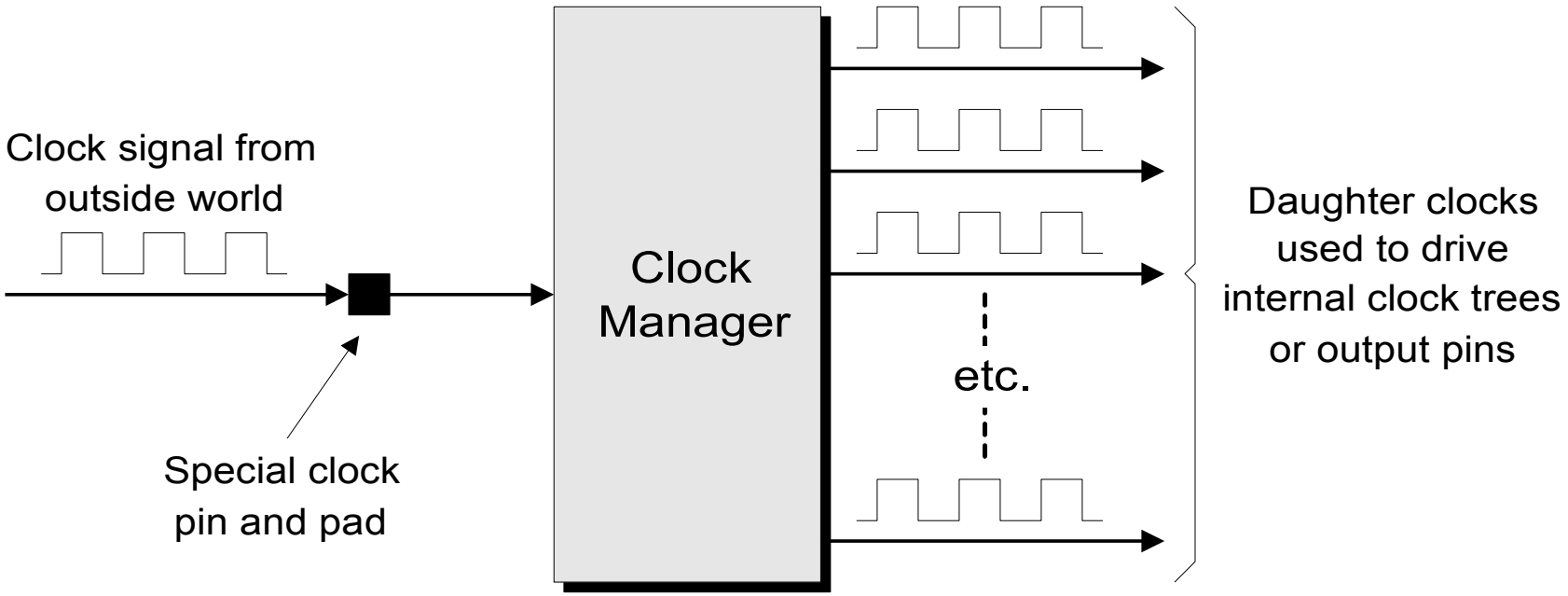



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Projekt współfinansowany przez Unię Europejską
w ramach Europejskiego Funduszu Społecznego

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

# Clock Manager

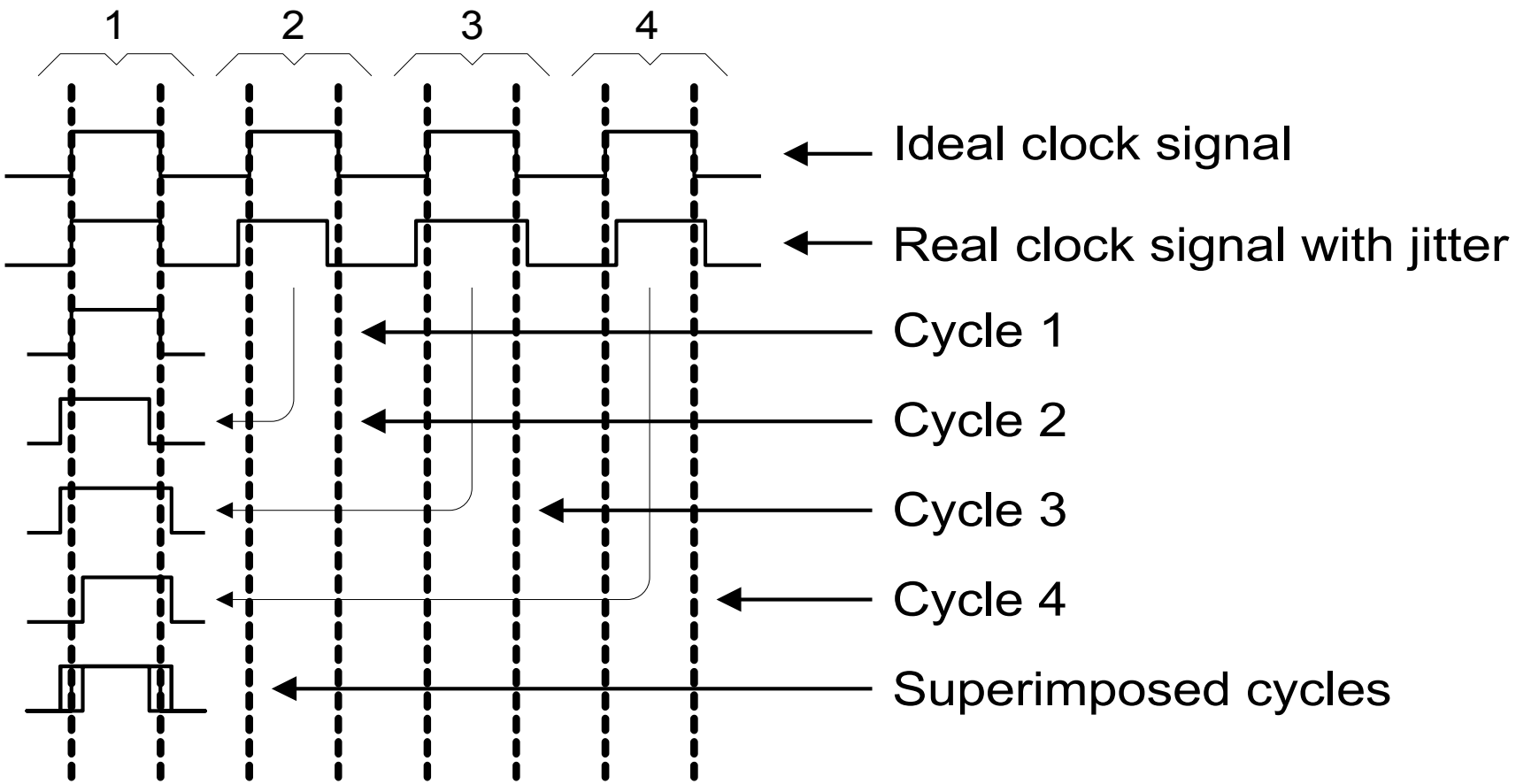



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Projekt współfinansowany przez Unię Europejską
w ramach Europejskiego Funduszu Społecznego

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

# Clock jitter



# Digital Clock Manager

- Moduły DCM pozwalają na:
  - Eliminację przesunięć fazowych zegara (*clock skew*), wewnątrz układu FPGA lub w stosunku do elementów zewnętrznych
  - Przesunięcie fazowe zegara o ustaloną lub regulowaną część okresu
  - Generację zegara o częstotliwości pomnożonej przez iloraz dwóch liczb
  - Generację zegara o współczynniku wypełnienia 50% na podstawie zegara nie spełniającego tego warunku

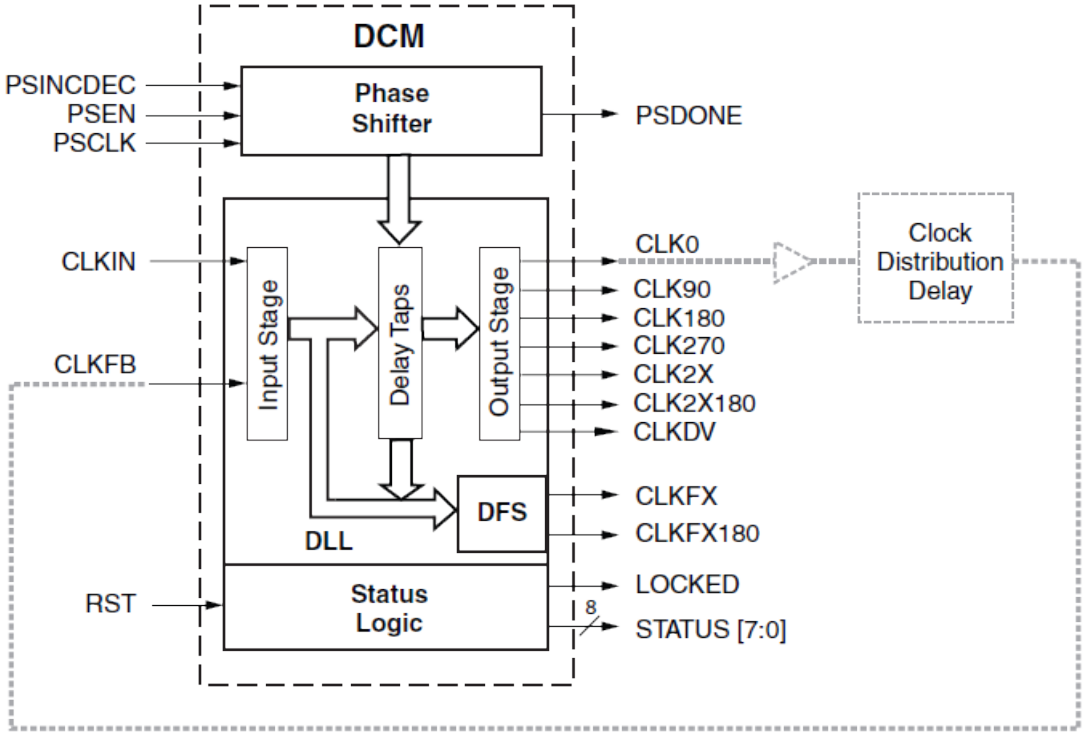
KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Projekt współfinansowany przez Unię Europejską
w ramach Europejskiego Funduszu Społecznego

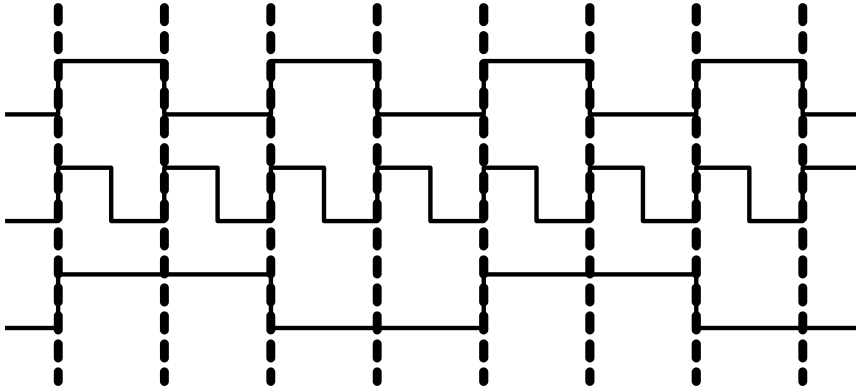
UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

# Digital Clock Manager

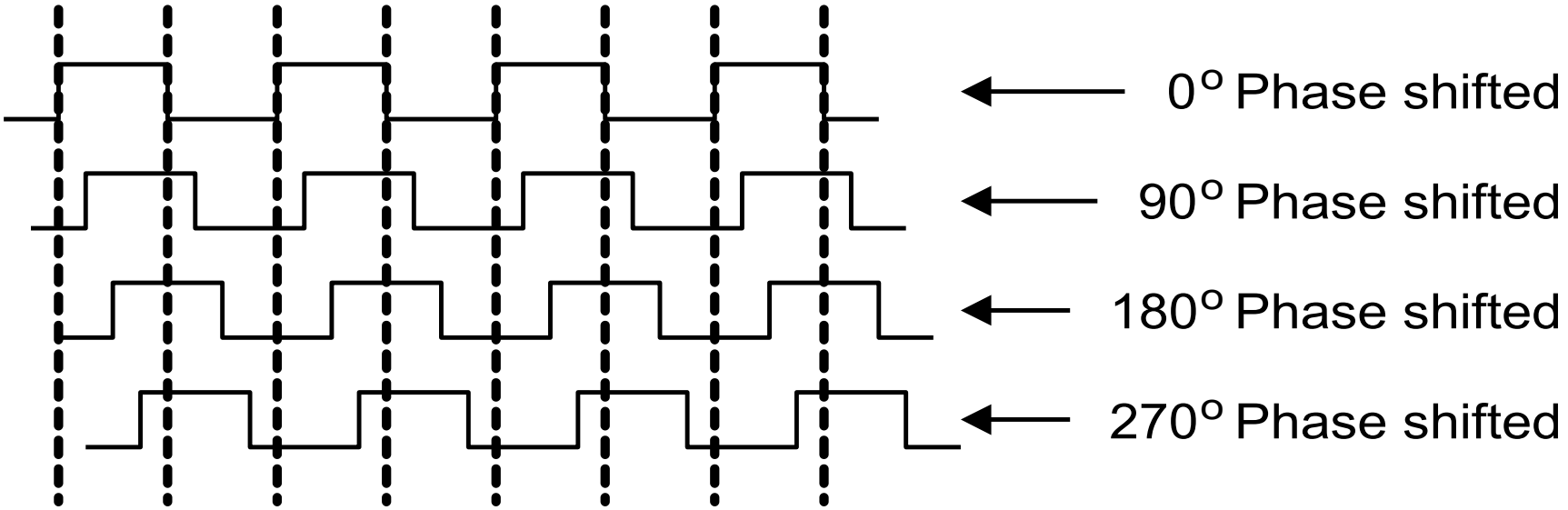
- DCM składa się z czterech komponentów:
  - Delay Locked Loop (DLL)
  - Digital Frequency Synthesizer (DFS)
  - Phase Shift (PS)
  - Status Logic



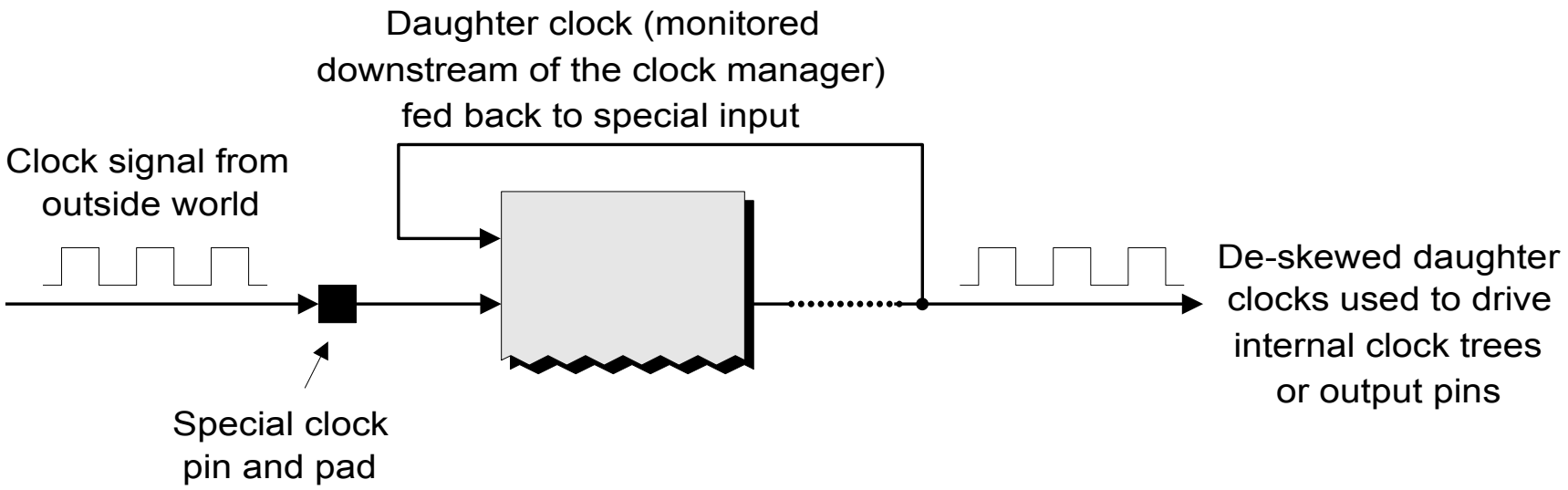
# Synteza częstotliwości

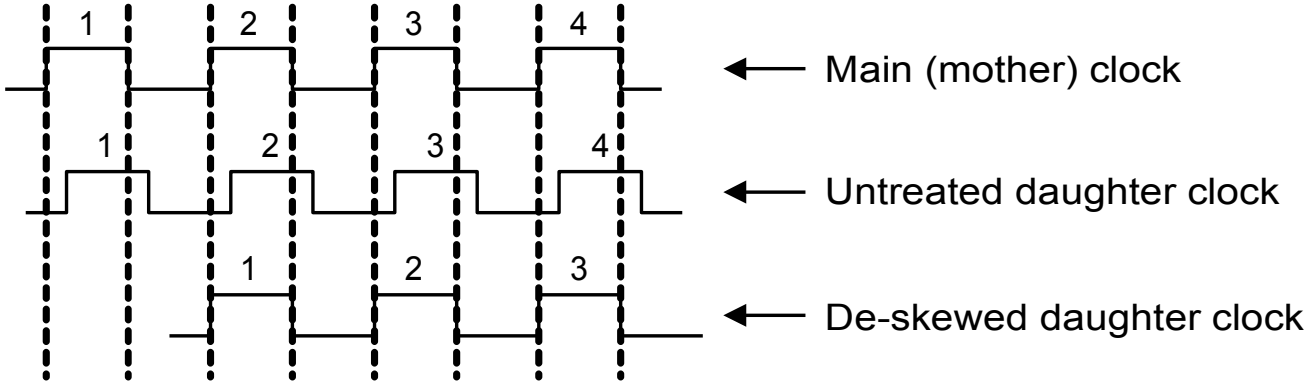


# Przesunięcie fazowe



# Usuwanie clock skew



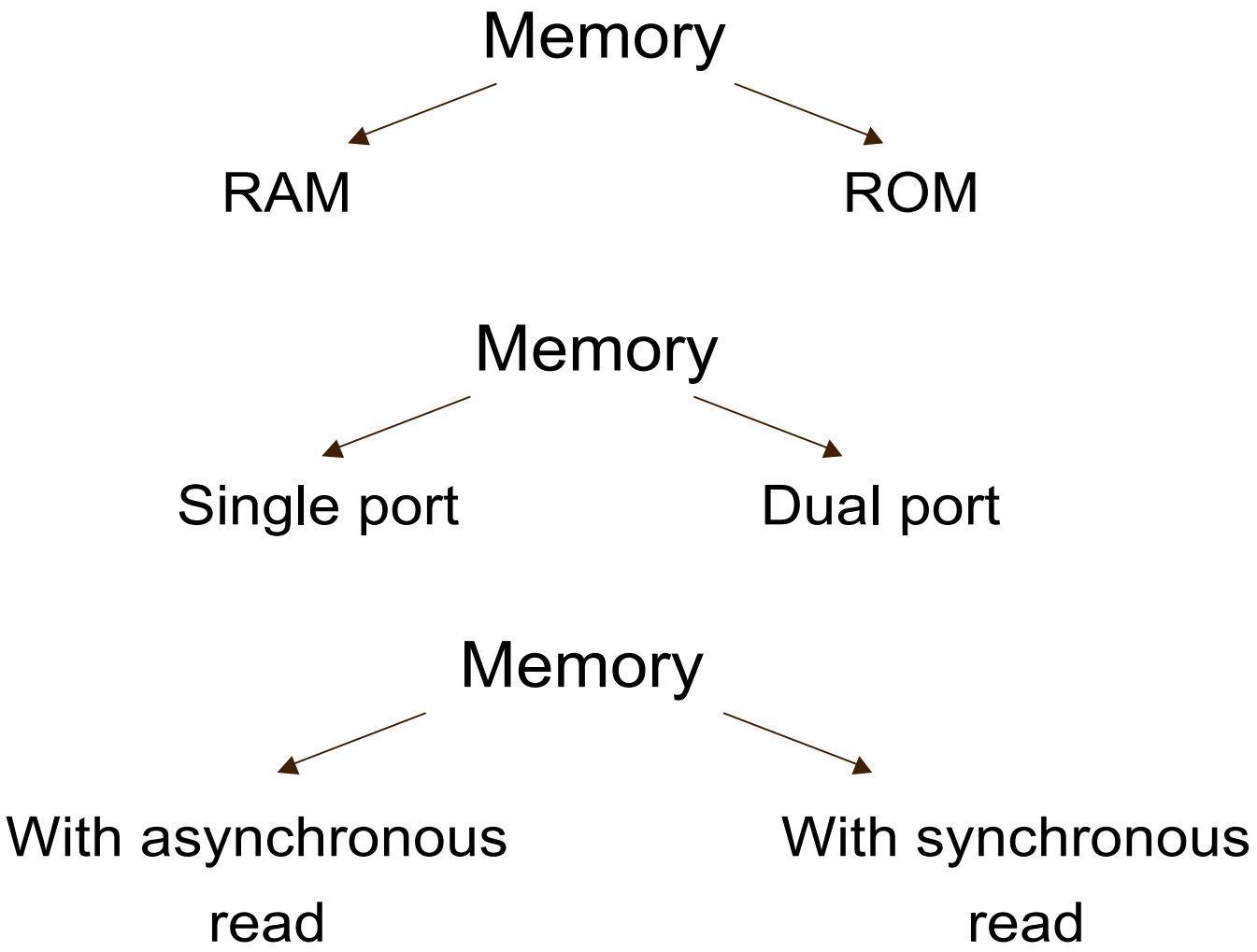
# Pamięci w FPGA

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Projekt współfinansowany przez Unię Europejską
w ramach Europejskiego Funduszu Społecznego

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

# Typy pamięci

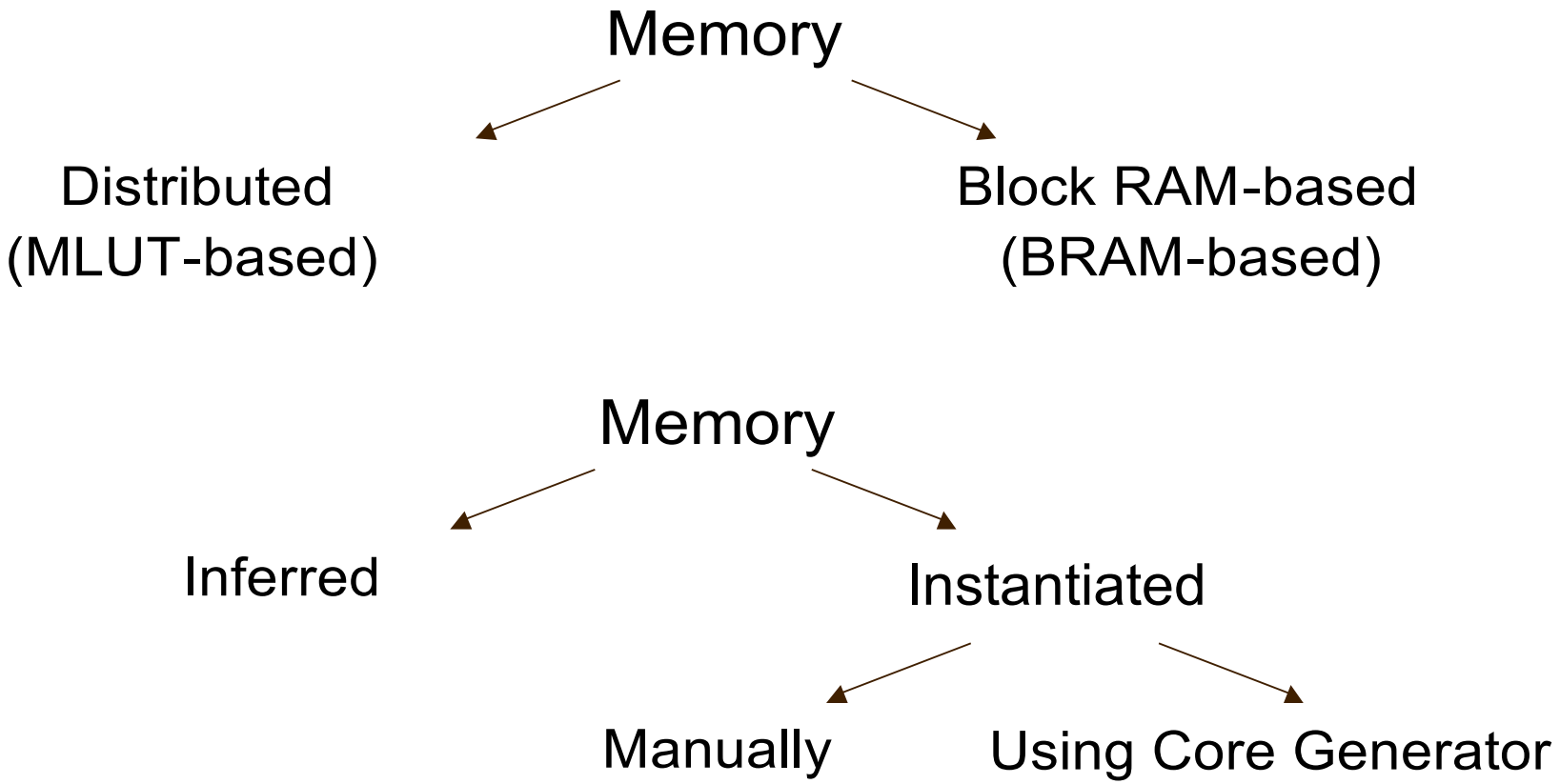



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Projekt współfinansowany przez Unię Europejską
w ramach Europejskiego Funduszu Społecznego

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

# Typy pamięci



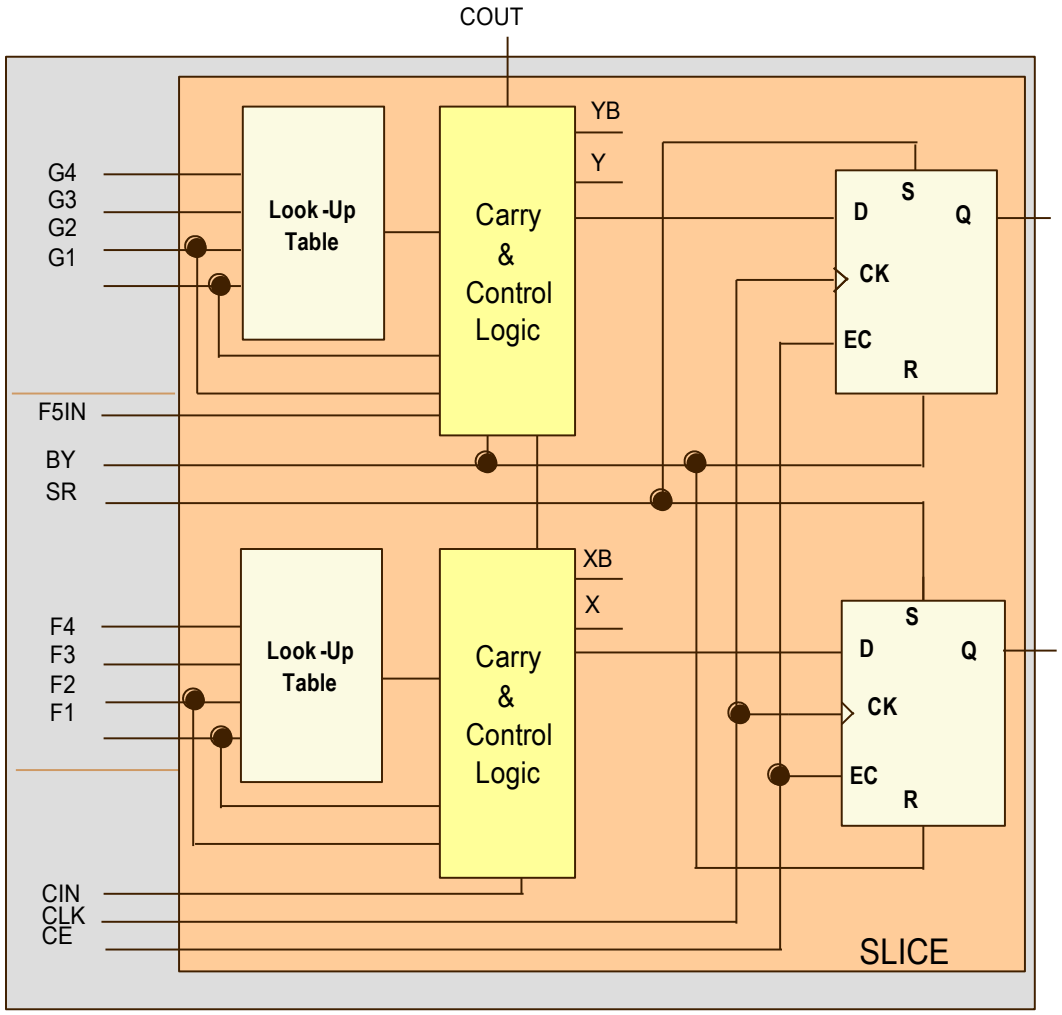
# Distributed RAM/ROM

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

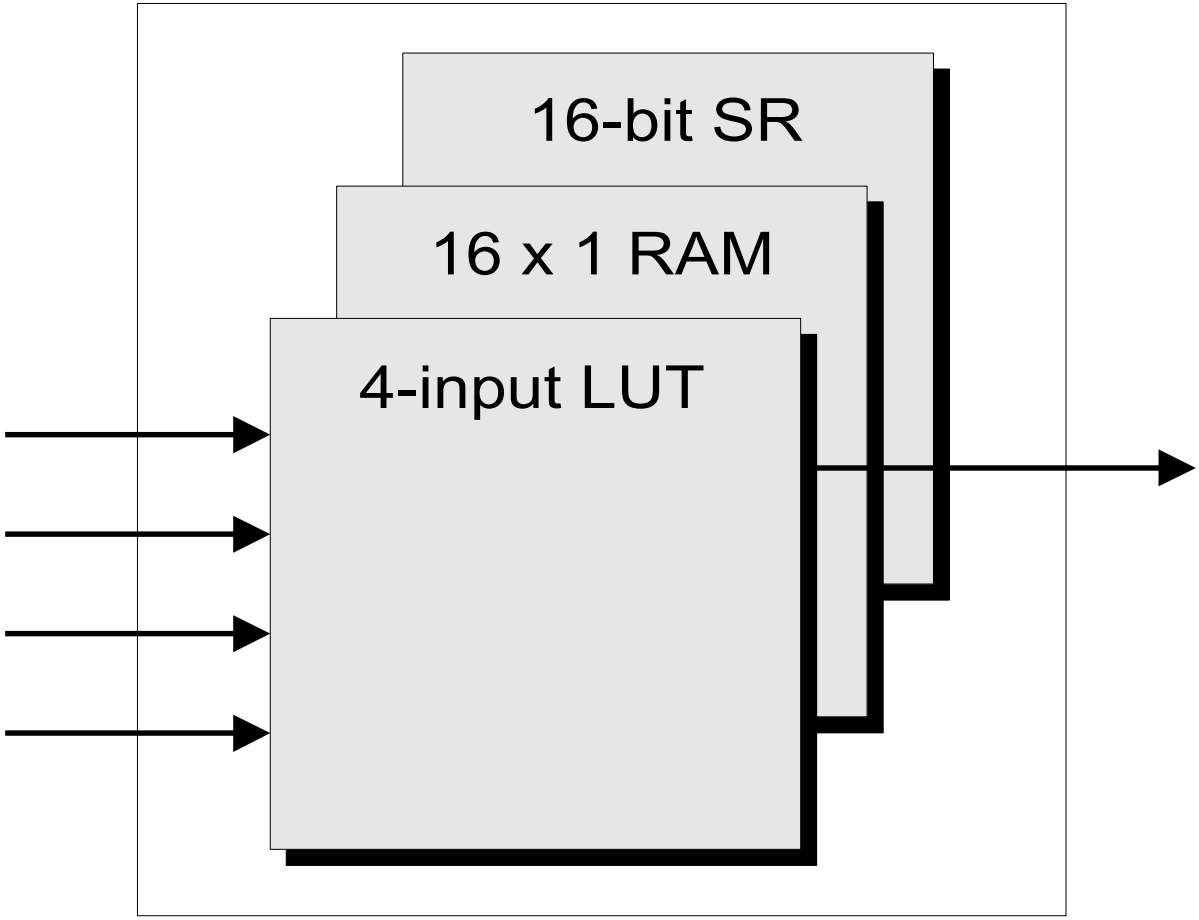
Projekt współfinansowany przez Unię Europejską
w ramach Europejskiego Funduszu Społecznego

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

# CLB Slice


COUT
YB
Y
G4
G3
G2
G1
Look -Up Table
Carry & Control Logic
S
D
Q
CK
EC
R
F5IN
BY
SR
XB
X
F4
F3
F2
F1
Look -Up Table
Carry & Control Logic
S
D
Q
CK
EC
R
CIN
CLK
CE
SLICE

# Xilinx Multipurpose LUT



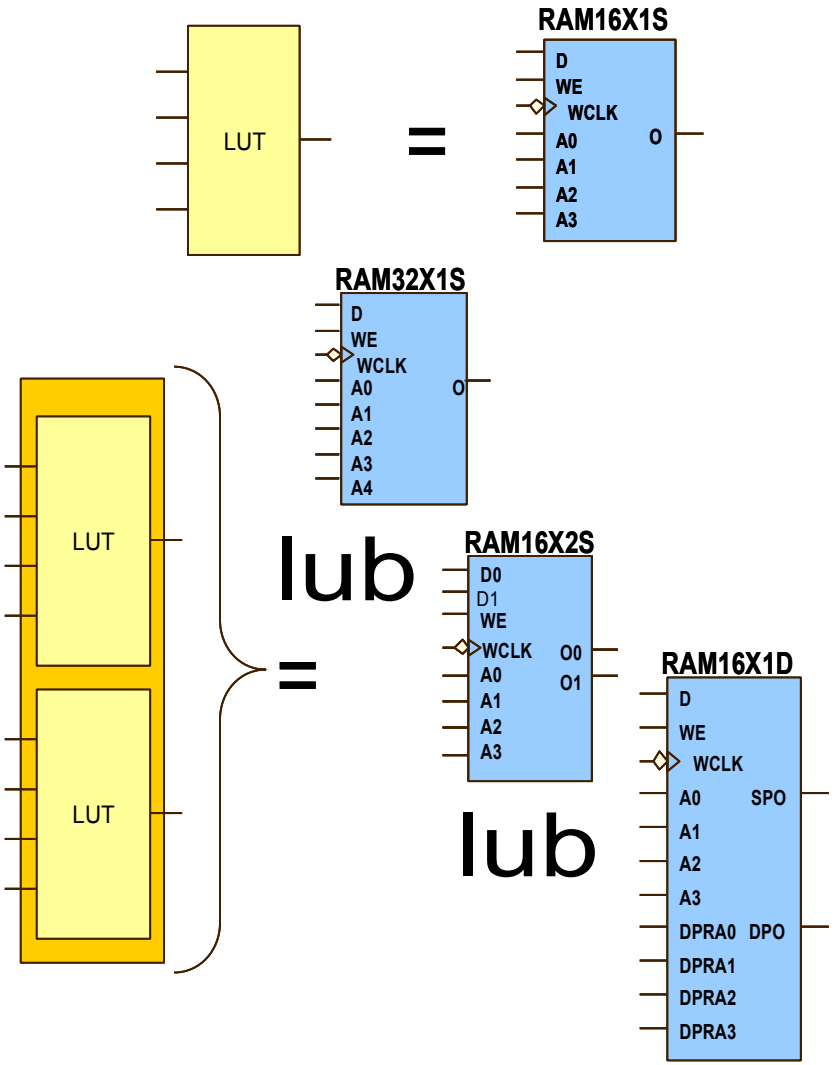

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Projekt współfinansowany przez Unię Europejską
w ramach Europejskiego Funduszu Społecznego

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

# Distributed RAM

- CLB LUT configurowalny jako Distributed RAM
  - LUT to 16x1 RAM
  - LUTy można połączyć, zwiększając obszar RAM
- Synchroniczny zapis
- Asynchroniczny odczyt
  - Używając dodatkowych przerzutników, można czytać synchronicznie
  - Bez dodatkowych zabiegów, odczyt rozproszonej pamięci RAM jest asynchroniczny
- Z dwóch LUTów może powstać:
  - 32 x 1 single-port RAM
  - 16 x 2 single-port RAM
  - 16 x 1 dual-port RAM



# Block RAM/ROM

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Projekt współfinansowany przez Unię Europejską
w ramach Europejskiego Funduszu Społecznego

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

# Block RAM

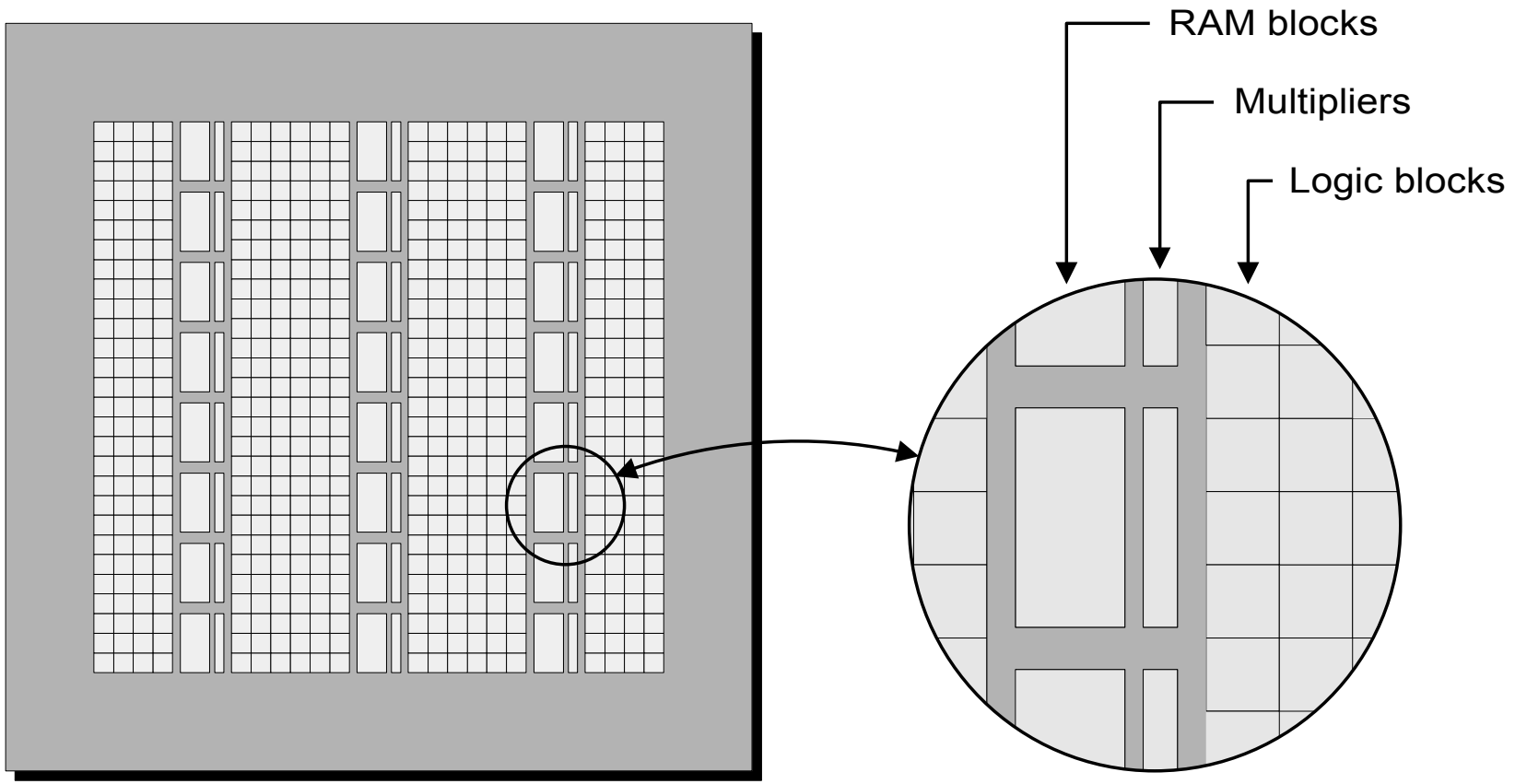
- Najefektywniejsza implementacja pamięci
  - Wykorzystuje specjalizowane bloki FPGA
- Idealne dla większości zastosowań
  - 3 do 126 bloków pamięci
    - 18 kbits = 18,432 bits per block (16 k bez bitów parzystości)
  - Większe pamięci używają wielu bloków
- Single i dual-port RAMs
- Synchroniczny zapis i odczyt (inaczej niż w pamięci rozproszonej)

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Projekt współfinansowany przez Unię Europejską
w ramach Europejskiego Funduszu Społecznego

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

# Bloki RAM i mnożarki w strukturze FPGA




KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Projekt współfinansowany przez Unię Europejską
w ramach Europejskiego Funduszu Społecznego

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

# Block RAM w rodzinie układów Spartan 3A

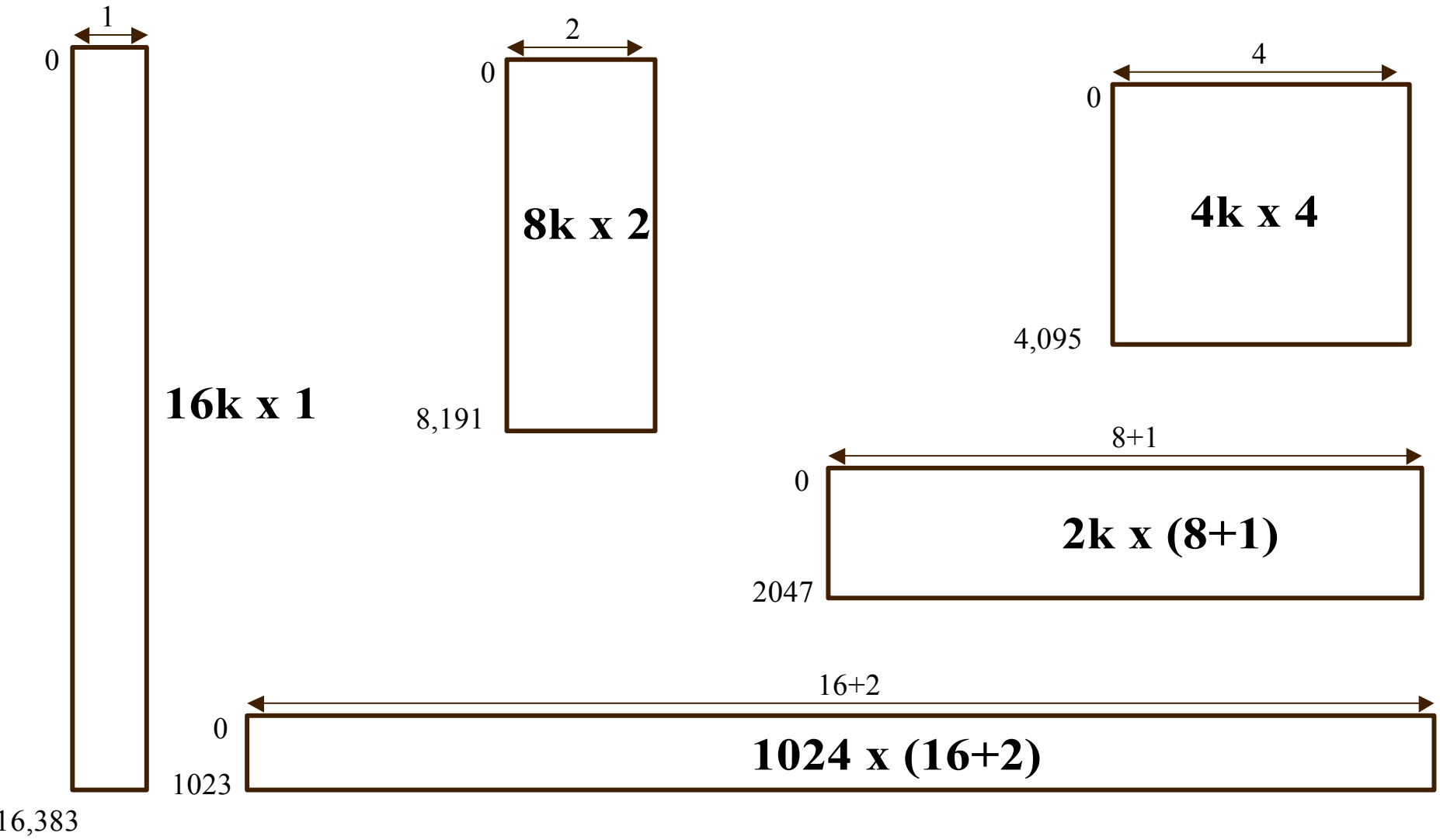
| Family | Device | RAM Columns | RAM Blocks Per Column | Total RAM Blocks | Total RAM Bits | Total RAM Kbits |
|---|---|---|---|---|---|---|
| Extended Spartan-3A FPGAs | | | | | | |
| | XC3SD1800A | 4 | 20-22 | 84 | 1,548,288 | 1,512K |
| | XC3SD3400A | 5 | 24-26 | 126 | 2,322,432 | 2,268K |
| | XC3S50A/AN | 1 | 3 | 3 | 55,296 | 54K |
| | XC3S200A/AN | 2 | 8 | 16 | 294,912 | 288K |
| | XC3S400A/AN | 2 | 10 | 20 | 368,640 | 360K |
| | XC3S700A/AN | 2 | 10 | 20 | 368,640 | 360K |
| | XC3S1400A/AN | 2 | 16 | 32 | 589,824 | 576K |

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Projekt współfinansowany przez Unię Europejską
w ramach Europejskiego Funduszu Społecznego

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

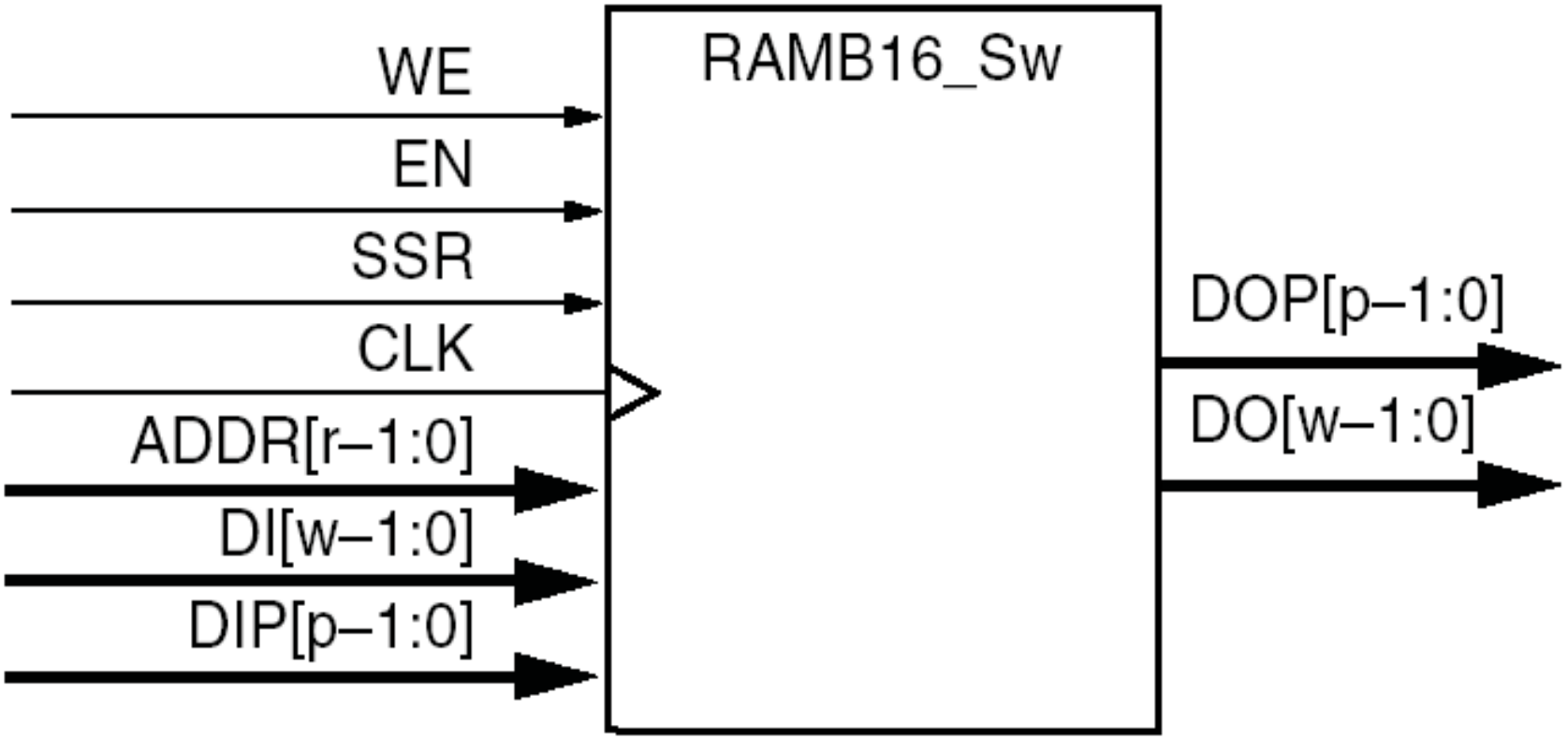
# Konfiguracje pamięci Block RAM



# Konfiguracje pamięci Block RAM

| DI/DO Bus Width (w – p bits) | DIP/DOP Bus Width (p bits) | Total Data Path Width (w bits) | ADDR Bus Width (r bits) | No. of Addressable Locations (n) | Block RAM Capacity (bits) |
|---|---|---|---|---|---|
| 1 | 0 | 1 | 14 | 16,384 | 16,384 |
| 2 | 0 | 2 | 13 | 8,192 | 16,384 |
| 4 | 0 | 4 | 12 | 4,096 | 16,384 |
| 8 | 1 | 9 | 11 | 2,048 | 18,432 |
| 16 | 2 | 18 | 10 | 1,024 | 18,432 |
| 32 | 4 | 36 | 9 | 512 | 18,432 |

# Single-Port Block RAM



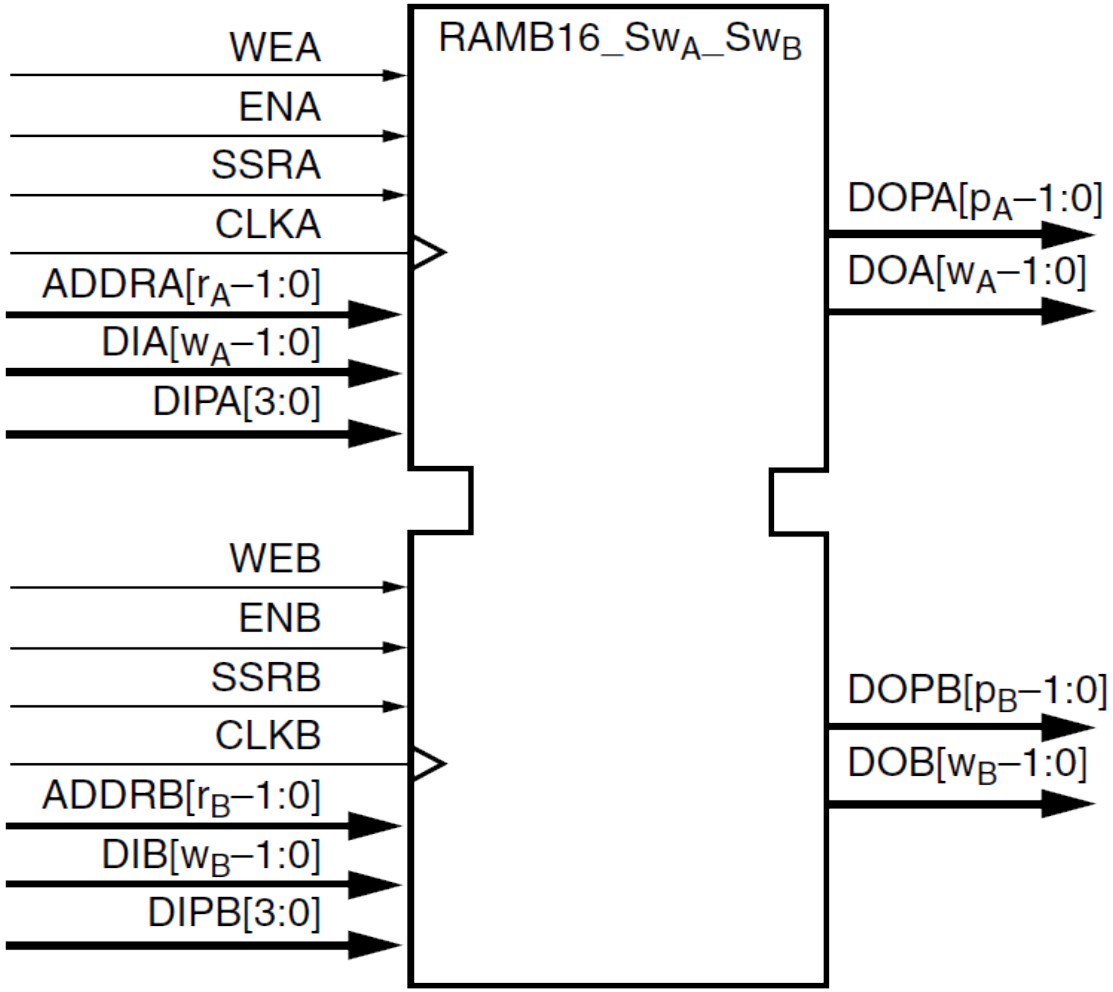

**(b) Single-Port**

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Projekt współfinansowany przez Unię Europejską
w ramach Europejskiego Funduszu Społecznego

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

# Dual-Port Block RAM




**(a) Dual-Port**

# Inference vs. Instatiation

- Istnieją dwie metody umieszczenia pamięci RAM:
  - Wywnioskowane przez syntezer (inferred)
    - Przenośny kod
  - Jawnie umieszczone przez projektanta (instantiated)
    - Najbardziej efektywne wykorzystanie zasobów w danej technologii
    - Wspiera wszystkie rodzaje pamięci RAM

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Projekt współfinansowany przez Unię Europejską
w ramach Europejskiego Funduszu Społecznego

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

# Distributed RAM

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;
USE ieee.std_logic_unsigned.all;

entity raminfr is
  generic ( bits : integer := 32;
            -- number of bits per RAM word
            addr_bits : integer := 8);
            -- 2^addr_bits = number of words in RAM
  port (clk : in std_logic;
        we  : in std_logic;
        a   : in std_logic_vector(addr_bits-1 downto 0);
        di  : in std_logic_vector(bits-1 downto 0);
        do  : out std_logic_vector(bits-1 downto 0));
end raminfr;

architecture behavioral of raminfr is
  type ram_type is array (2**addr_bits-1 downto 0)
       of std_logic_vector (bits-1 downto 0);
  signal RAM : ram_type;
begin
  process (clk)
  begin
    if (clk'event and clk = '1') then
      if (we = '1') then
        RAM(conv_integer(unsigned(a))) <= di;
      end if;
    end if;
  end process;
  do <= RAM(conv_integer(unsigned(a)));
end behavioral;
```

```
Logic Utilization:
  Number of 4 input LUTs:                656 out of  11,776    5%
Logic Distribution:
  Number of occupied Slices:             332 out of   5,888    5%
    Number of Slices containing only related logic:     332 out of     332 100%
    Number of Slices containing unrelated logic:          0 out of     332   0%
      *See NOTES below for an explanation of the effects of unrelated logic.
  Total Number of 4 input LUTs:          656 out of  11,776    5%
    Number used as logic:                144
    Number used for 32x1 RAMs:           512
      (Two LUTs used per 32x1 RAM)
  Number of bonded IOBs:                  74 out of     372   19%
  Number of BUFGMUXs:                      1 out of      24    4%
```

# Block RAM - wersja 1

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;
USE ieee.std_logic_unsigned.all;

entity raminfr is
  generic ( bits : integer := 32;
            -- number of bits per RAM word
           addr_bits : integer := 8);
            -- 2^addr_bits = number of words in RAM
  port (clk : in std_logic;
        we  : in std_logic;
        a   : in std_logic_vector(addr_bits-1 downto 0);
        di  : in std_logic_vector(bits-1 downto 0);
        do  : out std_logic_vector(bits-1 downto 0));
end raminfr;

architecture behavioral of raminfr is
  type ram_type is array (2**addr_bits-1 downto 0)
      of std_logic_vector (bits-1 downto 0);
  signal RAM : ram_type;
begin
  process (clk)
  begin
    if (clk'event and clk = '1') then
      if (we = '1') then
        RAM(conv_integer(unsigned(a))) <= di;
      end if;
    do <= RAM(conv_integer(unsigned(a)));
    end if;
  end process;

end behavioral;
```

```
Logic Utilization:
Logic Distribution:
    Number of Slices containing only related logic:       0 out of       0   0%
    Number of Slices containing unrelated logic:          0 out of       0   0%
      *See NOTES below for an explanation of the effects of unrelated logic.
  Number of bonded IOBs:                 74 out of     372   19%
  Number of BUFGMUXs:                     1 out of      24    4%
  Number of RAMB16BWEs:                   1 out of      20    5%
```

# Block RAM - wersja 2

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;
USE ieee.std_logic_unsigned.all;

entity raminfr is
  generic ( bits : integer := 32;
              -- number of bits per RAM word
            addr_bits : integer := 8);
              -- 2^addr_bits = number of words in RAM
  port (clk : in std_logic;
        we  : in std_logic;
        a   : in std_logic_vector(addr_bits-1 downto 0);
        di  : in std_logic_vector(bits-1 downto 0);
        do  : out std_logic_vector(bits-1 downto 0));
end raminfr;

architecture behavioral of raminfr is
  type ram_type is array (2**addr_bits-1 downto 0) of std_logic_vector (bits-1 downto 0);
  signal RAM : ram_type;
  signal read_a : std_logic_vector(addr_bits-1 downto 0);

begin
  process (clk)
  begin
    if (clk'event and clk = '1') then
      if (we = '1') then
        RAM(conv_integer(unsigned(a))) <= di;
      end if;
    read_a <= a;
    end if;
  end process;
  do <= RAM(conv_integer(unsigned(read_a)));
end behavioral;
```

```
Logic Utilization:
Logic Distribution:
    Number of Slices containing only related logic:       0 out of       0   0%
    Number of Slices containing unrelated logic:          0 out of       0   0%
      *See NOTES below for an explanation of the effects of unrelated logic.
  Number of bonded IOBs:                 74 out of     372   19%
  Number of BUFGMUXs:                     1 out of      24    4%
  Number of RAMB16BWEs:                   1 out of      20    5%
```

# Dual Port RAM - distributed

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use ieee.std_logic_arith.all;

entity raminfr is
 generic ( bits : integer := 32;
           -- number of bits per RAM word
           addr_bits : integer := 8);
           -- 2^addr_bits = number of words in RAM
  port (clk  : in std_logic;
        we   : in std_logic;
        a    : in std_logic_vector(addr_bits-1 downto 0);
        dpra : in std_logic_vector(addr_bits-1 downto 0);
        di   : in std_logic_vector(bits-1 downto 0);
        spo  : out std_logic_vector(bits-1 downto 0);
        dpo  : out std_logic_vector(bits-1 downto 0));
end raminfr;

architecture syn of raminfr is
  type ram_type is array (2**addr_bits-1 downto 0) of std_logic_vector (bits-1 downto 0);
  signal RAM : ram_type;
begin
  process (clk)
  begin
    if (clk'event and clk = '1') then
      if (we = '1') then
        RAM(conv_integer(unsigned(a))) <= di;
      end if;
    end if;
  end process;
  spo <= RAM(conv_integer(unsigned(a)));
  dpo <= RAM(conv_integer(unsigned(dpra)));
 end syn;
```

```
Logic Utilization:
  Number of 4 input LUTs:            1,584 out of  11,776   13%
Logic Distribution:
  Number of occupied Slices:           800 out of   5,888   13%
    Number of Slices containing only related logic:     800 out of     800 100%
    Number of Slices containing unrelated logic:          0 out of     800   0%
      *See NOTES below for an explanation of the effects of unrelated logic.
  Total Number of 4 input LUTs:      1,584 out of  11,776   13%
    Number used as logic:              560
    Number used for Dual Port RAMs:  1,024
      (Two LUTs used per Dual Port RAM)
  Number of bonded IOBs:               114 out of     372   30%
  Number of BUFGMUXs:                    1 out of      24    4%
```

# Dual Port RAM - Block - wersja 1

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use ieee.std_logic_arith.all;

entity raminfr is
 generic ( bits : integer := 32;
             -- number of bits per RAM word
           addr_bits : integer := 8);
             -- 2^addr_bits = number of words in RAM
  port (clk  : in std_logic;
        we   : in std_logic;
        a    : in std_logic_vector(addr_bits-1 downto 0);
        dpra : in std_logic_vector(addr_bits-1 downto 0);
        di   : in std_logic_vector(bits-1 downto 0);
        spo  : out std_logic_vector(bits-1 downto 0);
        dpo  : out std_logic_vector(bits-1 downto 0));
end raminfr;

architecture syn of raminfr is
  type ram_type is array (2**addr_bits-1 downto 0) of std_logic_vector (bits-1 downto 0);
  signal RAM : ram_type;
begin
  process (clk)
  begin
    if (clk'event and clk = '1') then
      if (we = '1') then
        RAM(conv_integer(unsigned(a))) <= di;
      end if;
      spo <= RAM(conv_integer(unsigned(a)));
      dpo <= RAM(conv_integer(unsigned(dpra)));
    end if;
  end process;
 end syn;
```

```
Logic Utilization:
Logic Distribution:
    Number of Slices containing only related logic:       0 out of       0   0%
    Number of Slices containing unrelated logic:          0 out of       0   0%
      *See NOTES below for an explanation of the effects of unrelated logic.
  Number of bonded IOBs:                114 out of     372   30%
  Number of BUFGMUXs:                     1 out of      24    4%
  Number of RAMB16BWEs:                   1 out of      20    5%
```

# Dual Port RAM - Block - wersja 2

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use ieee.std_logic_arith.all;

entity raminfr is
 generic ( bits : integer := 32;
             -- number of bits per RAM word
           addr_bits : integer := 8);
            -- 2^addr_bits = number of words in RAM
  port (clk  : in std_logic;
        we   : in std_logic;
        a    : in std_logic_vector(addr_bits-1 downto 0);
        dpra : in std_logic_vector(addr_bits-1 downto 0);
        di   : in std_logic_vector(bits-1 downto 0);
        spo  : out std_logic_vector(bits-1 downto 0);
        dpo  : out std_logic_vector(bits-1 downto 0));
end raminfr;

architecture syn of raminfr is
  type ram_type is array (2**addr_bits-1 downto 0) of std_logic_vector (bits-1 downto 0);
  signal RAM : ram_type;
  signal read_a, read_dpra: std_logic_vector(addr_bits-1 downto 0);
begin
  process (clk)
  begin
    if (clk'event and clk = '1') then
      if (we = '1') then
        RAM(conv_integer(unsigned(a))) <= di;
      end if;
      read_a <= a;
      read_dpra <= dpra;
    end if;
  end process;
  spo <= RAM(conv_integer(unsigned(read_a)));
  dpo <= RAM(conv_integer(unsigned(read_dpra)));
 end syn;
```

```
Logic Utilization:
Logic Distribution:
    Number of Slices containing only related logic:       0 out of       0   0%
    Number of Slices containing unrelated logic:          0 out of       0   0%
      *See NOTES below for an explanation of the effects of unrelated logic.
  Number of bonded IOBs:                 114 out of     372   30%
  Number of BUFGMUXs:                      1 out of      24    4%
  Number of RAMB16BWEs:                    1 out of      20    5%
```

# ROM - distributed

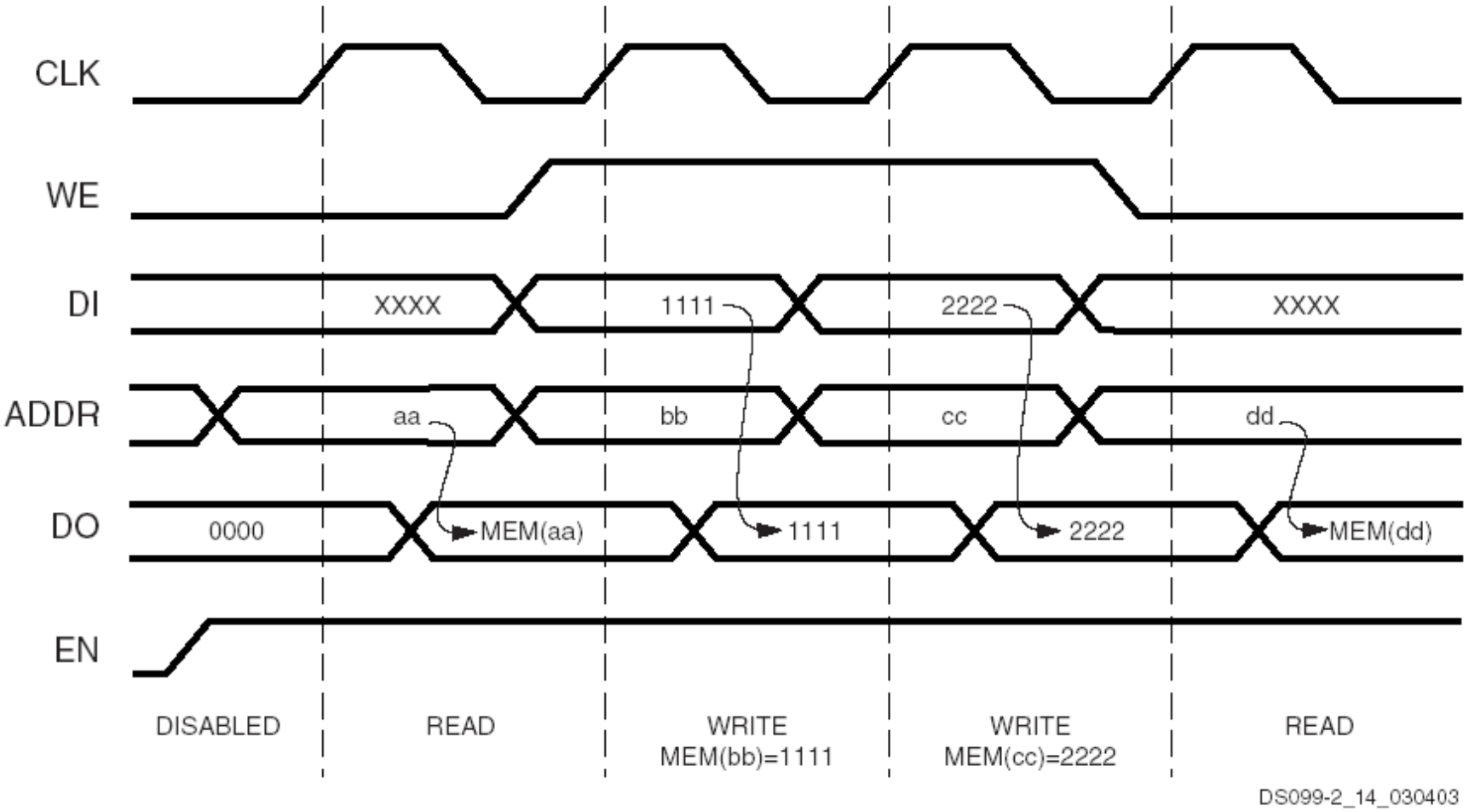
```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;
USE ieee.std_logic_unsigned.all;

entity rominfr is
  generic ( bits : integer := 10;
             -- number of bits per ROM word
            addr_bits : integer := 3);
             -- 2^addr_bits = number of words in ROM
  port (a   : in std_logic_vector(addr_bits-1 downto 0);
        do  : out std_logic_vector(bits-1 downto 0));
end rominfr;

architecture behavioral of rominfr is
  type rom_type is array (2**addr_bits-1 downto 0)
    of std_logic_vector (bits-1 downto 0);
  constant ROM : rom_type :=
  ("0000110001",
   "0100110100",
   "0100110110",
   "0110110000",
   "0000111100",
   "0111110101",
   "0100110100",
   "1111100111");
begin
  do <= ROM(conv_integer(unsigned(a)));
end behavioral;
```

```
Logic Utilization:
  Number of 4 input LUTs:                 9 out of  11,776    1%
Logic Distribution:
  Number of occupied Slices:              5 out of   5,888    1%
    Number of Slices containing only related logic:     5 out of       5 100%
    Number of Slices containing unrelated logic:        0 out of       5   0%
      *See NOTES below for an explanation of the effects of unrelated logic.
  Total Number of 4 input LUTs:           9 out of  11,776    1%
  Number of bonded IOBs:                 13 out of     372    3%
```

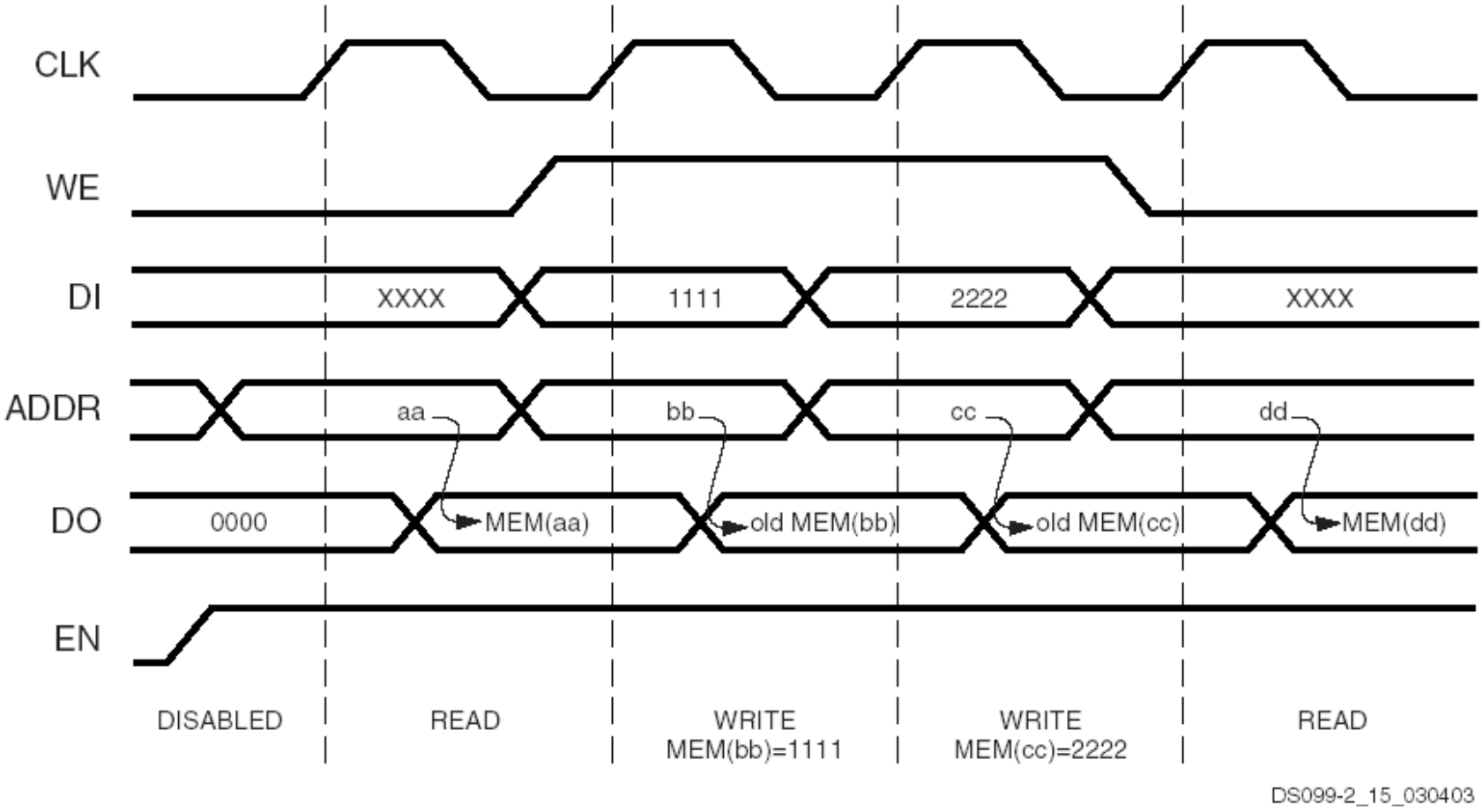
# Distributed RAM - instantiation

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;

library UNISIM;
use UNISIM.vcomponents.all;

entity RAM_16X8_DISTRIBUTED is
   port(
         CLK : in STD_LOGIC;
         WE : in STD_LOGIC;
         ADDR : in STD_LOGIC_VECTOR(3 downto 0);
         DATA_IN : in STD_LOGIC_VECTOR(7 downto 0);
         DATA_OUT : out STD_LOGIC_VECTOR(7 downto 0)
       );
end RAM_16X8_DISTRIBUTED;

architecture RAM_16X8_DISTRIBUTED_STRUCTURAL of RAM_16X8_DISTRIBUTED is
begin

  GENERATE_MEMORY:
  for I in 0 to 7 generate
RAM_16x1_S_1: ram16x1s
    generic map (INIT => X"0000")
    port map
       (O => DATA_OUT(I),
        A0 => ADDR(0),
        A1 => ADDR(1),
        A2 => ADDR(2),
        A3 => ADDR(3),
        D => DATA_IN(I),
        WCLK => CLK,
        WE => WE
       );
  end generate;

end RAM_16X8_DISTRIBUTED_STRUCTURAL;
```

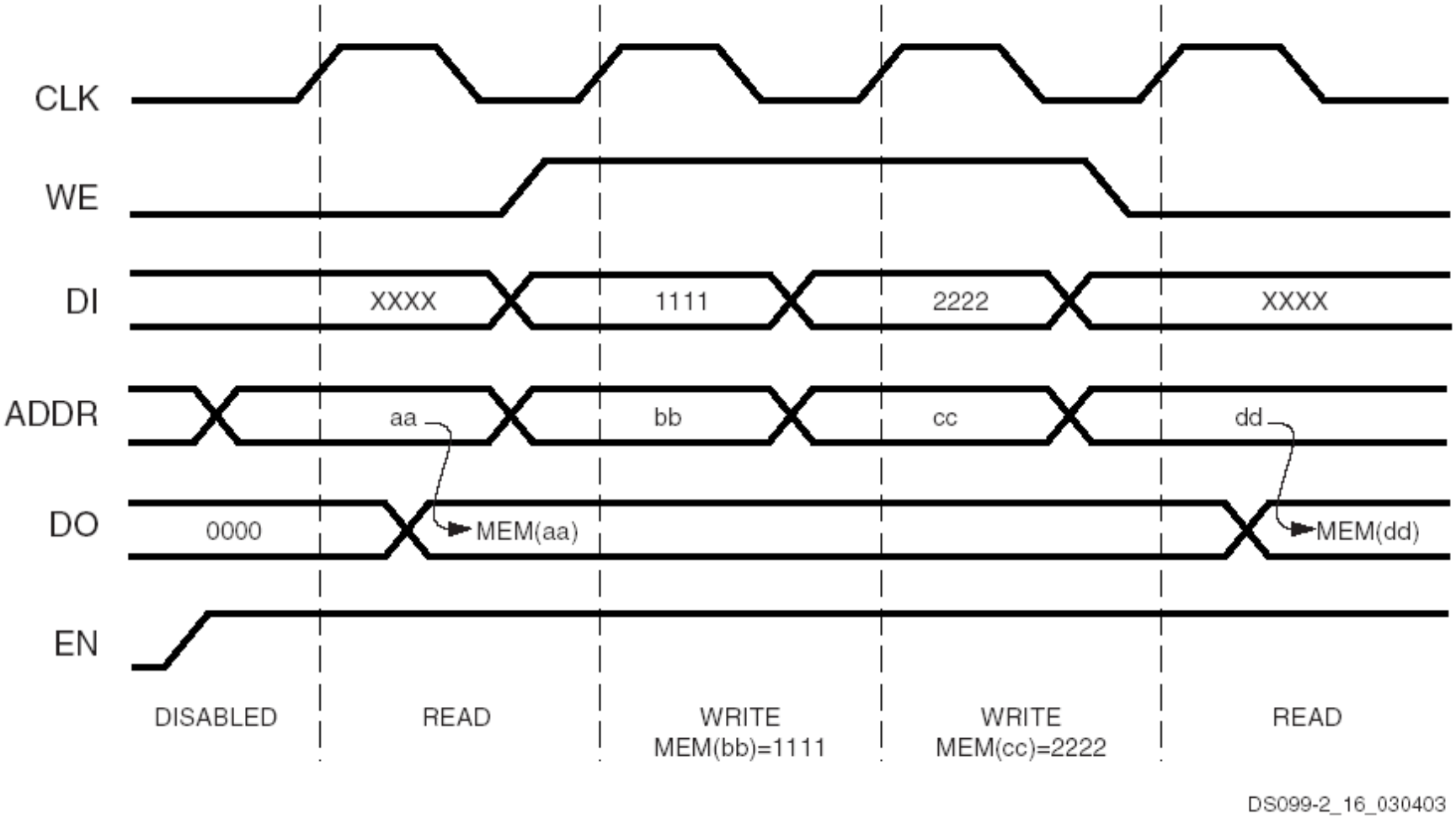
# Block RAM Waveforms – WRITE_FIRST



# Block RAM Waveforms – READ_FIRST



# Block RAM Waveforms – NO_CHANGE



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

# „Układy reprogramowalne i SoC”
# „Specjalizowane moduły FPGA”

Prezentacja jest współfinansowana przez
Unię Europejską w ramach
Europejskiego Funduszu Społecznego w projekcie pt.

*„Innowacyjna dydaktyka bez ograniczeń - zintegrowany rozwój Politechniki Łódzkiej - zarządzanie Uczelnią, nowoczesna oferta edukacyjna i wzmacniania zdolności do zatrudniania osób niepełnosprawnych”*

Prezentacja dystrybuowana jest bezpłatnie