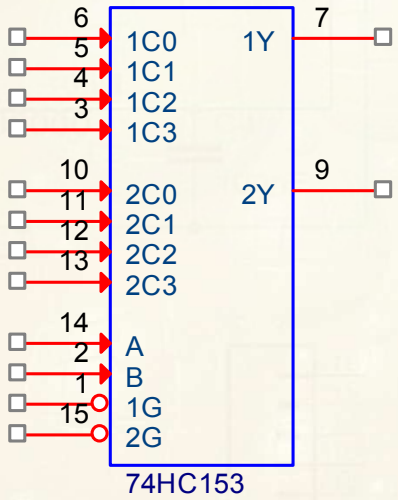
MULTIPLEXER

Multiplexer je LKO vytvořený pro realizaci elektronického přepínače logických signálů. Pro přepínač 1 ze 4 vstupů můžeme nakreslit následující značku, kde signály B a A představují číslo BA pro výběr vstupu. Zobrazena je pravdivostní tabulka a logický výraz pro výstup Y.

$$Y = \overline{G}.\left(\overline{A}.\overline{B}.I_0 + A.\overline{B}.I_1 + \overline{A}.B.I_2 + A.B.I_3\right)$$



G	B	A	Y
1	X	X	0
0	0	0	I_0
0	0	1	I_1
0	1	0	I_2
0	1	1	I_3

Multiplexer v bezhazardní provedení realizuje logickou část některých PLD obvodů.

Vyrábí v řadě 15x a 25x ve variantě

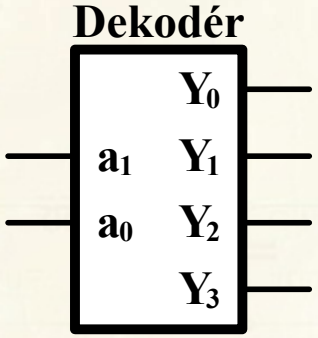
4x1ze2 (157,257),

2x1ze4 (153,253),

1x1z8 (151,251) a 1x1z16 (150,250). Řada 15x disponuje dvoustavovým výstupem a řada 25x disponuje třístavovým výstupem.

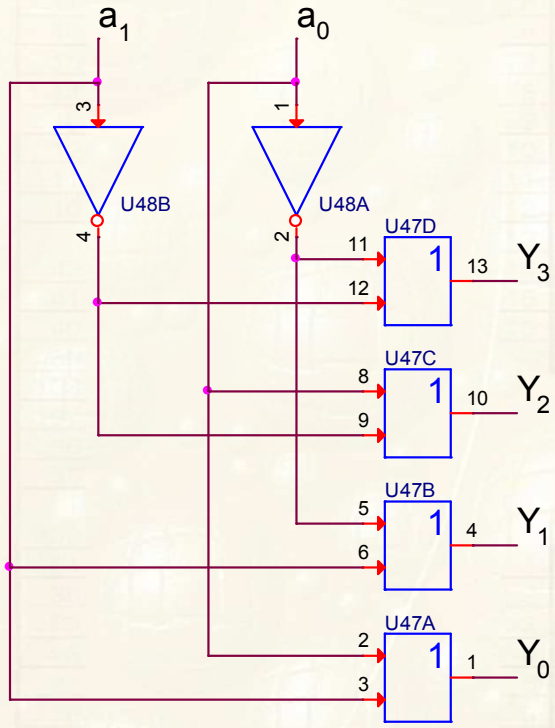
DEKODÉR

Dekodér je logický kombinační obvod, u kterého se binární číslo nebo adresa používá k aktivaci jednoho z $m=2^n$ výstupů, kde n je počet bitů vstupního čísla. Vstupní číslo obvykle určuje, který z výstupů bude aktivován. V případě dvou bitového vstupního čísla bude mít obvod 4 výstupy, jejichž stav je popsán následující pravdivostní tabulkou.



a_1	a_0	$Y_3Y_2Y_1Y_0$
0	0	1 1 1 0
0	1	1 1 0 1
1	0	1 0 1 1
1	1	0 1 1 1

Dekodér tvoří dílčí část paměťového obvodu a složitých LO. Používá se jako **adresový dekodér** v uP systémech, může šetřit počet vodičů k řízení dynamického displeje, kde jsou jednotlivé segmentovky postupně rozsvěcovány. Může indikovat stavy systému. Vyrábí se v provedení 137, 138 a 139.



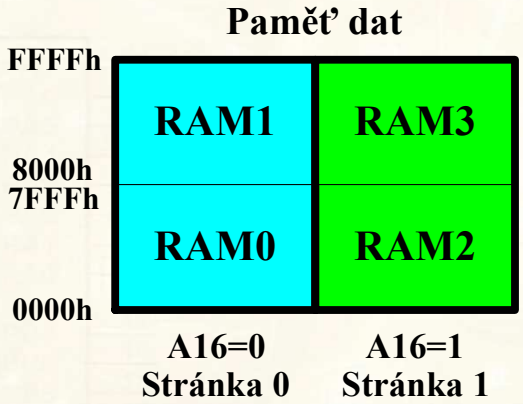
$$Y_3 = \overline{a}_1 + \overline{a}_0$$
$$Y_2 = \overline{a}_1 + a_0$$
$$Y_1 = a_1 + \overline{a}_0$$
$$Y_0 = a_1 + a_0$$

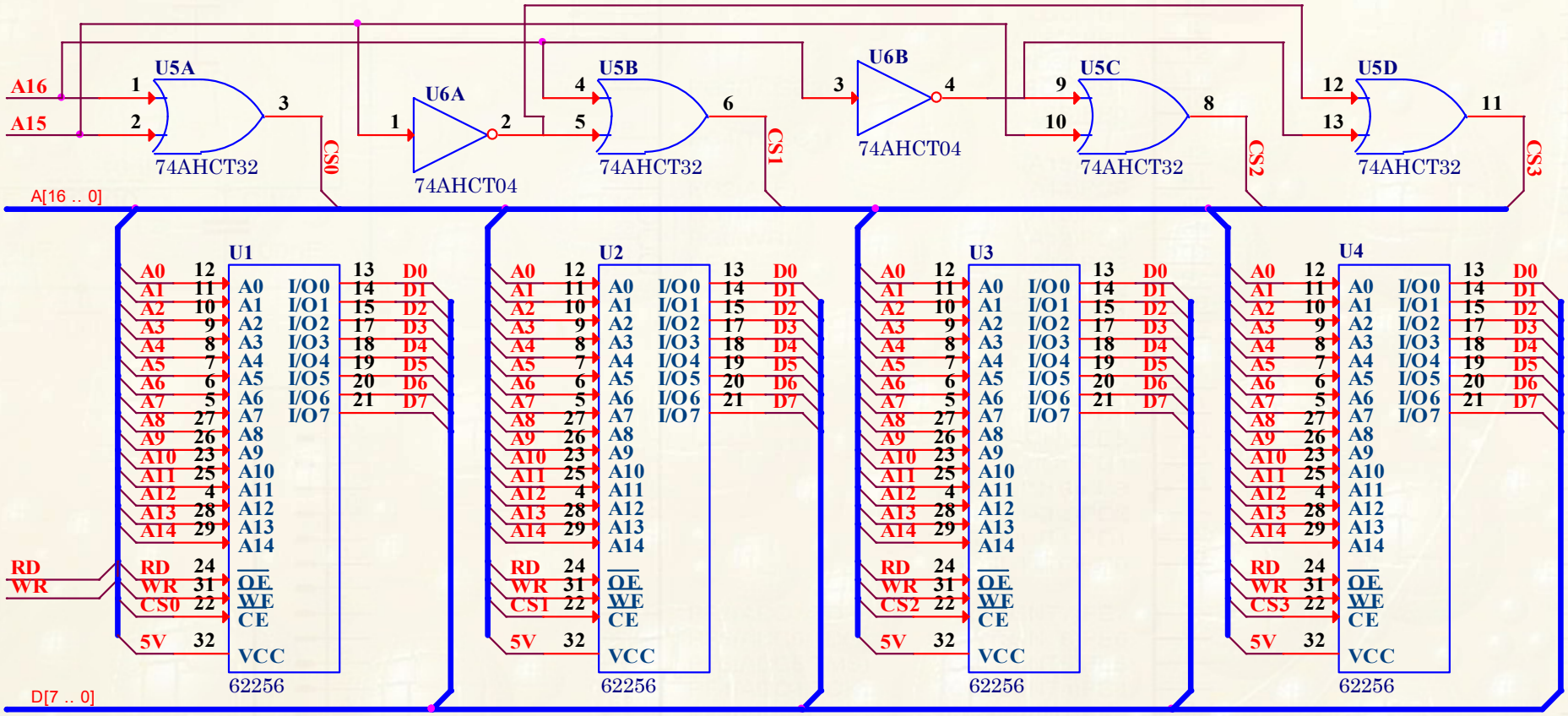
PŘÍKLAD REALIZACE 128KB PAMĚTI RAM SLOŽENÉ ZE ČTYŘ PAMĚTÍ 32KB

Pro čtyři 32kB paměti RAM musíme vytvořit aktivační signály CS0, CS1, CS2 a CS3.

$$CS0 = A15 + A16 \qquad CS1 = \overline{A15} + A16$$

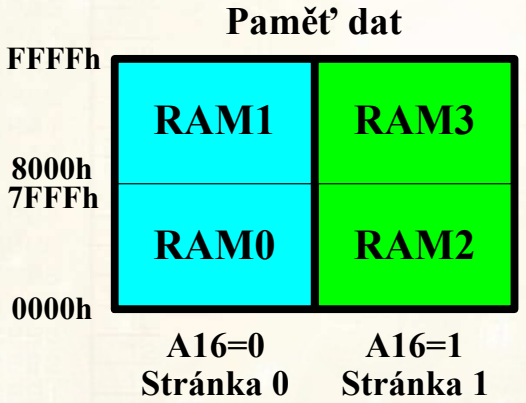
$$CS2 = A15 + \overline{A16} \qquad CS3 = \overline{A15} + \overline{A16}$$

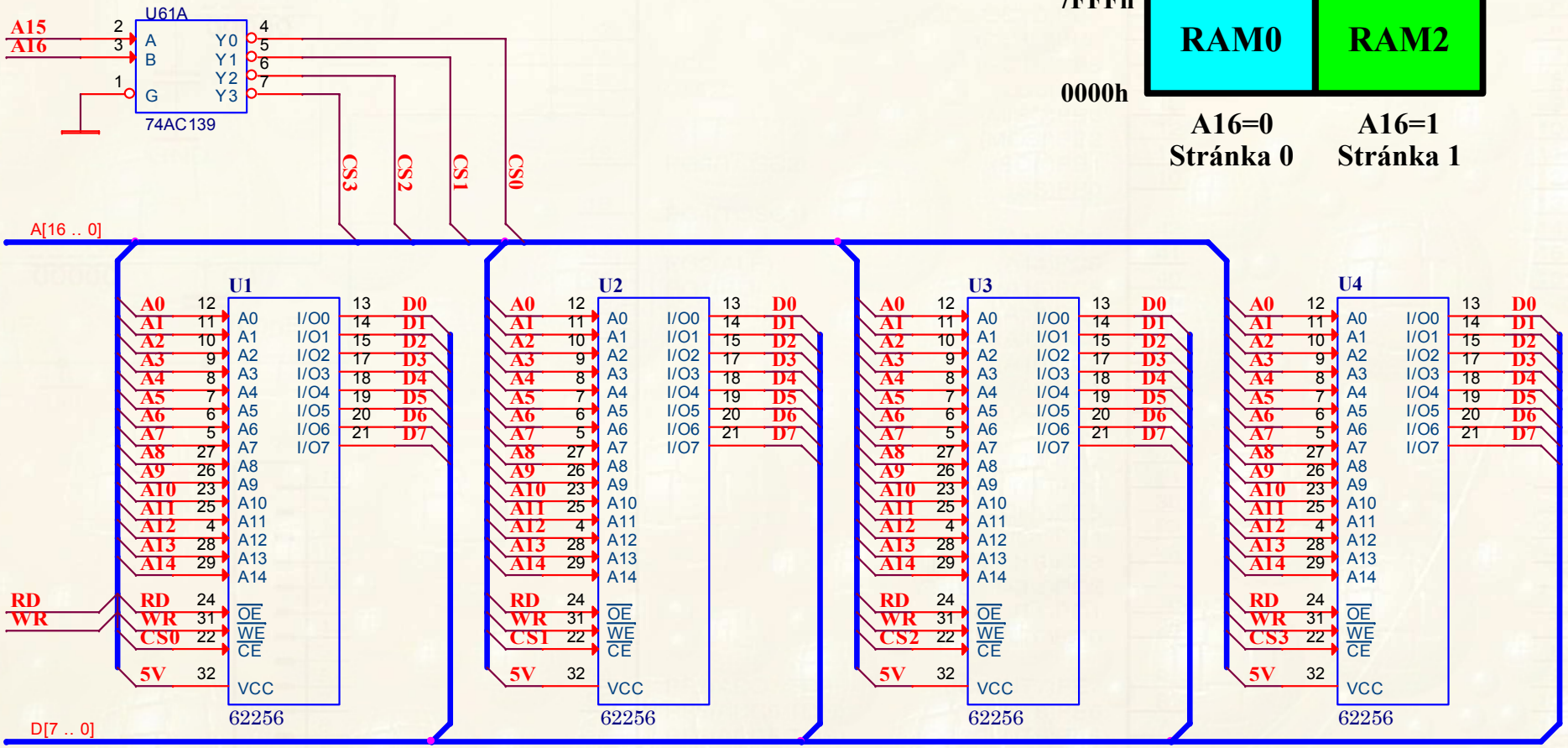




POUŽITÍ DEKODÉRU K AKTIVACI JEDNOTLIVÝCH 32KB PAMĚTÍ

Pro každou ze čtyř 32kB pamětí RAM musíme vytvořit aktivační signály CS0, CS1, CS2 a CS3 pomocí dekodéru 74139.

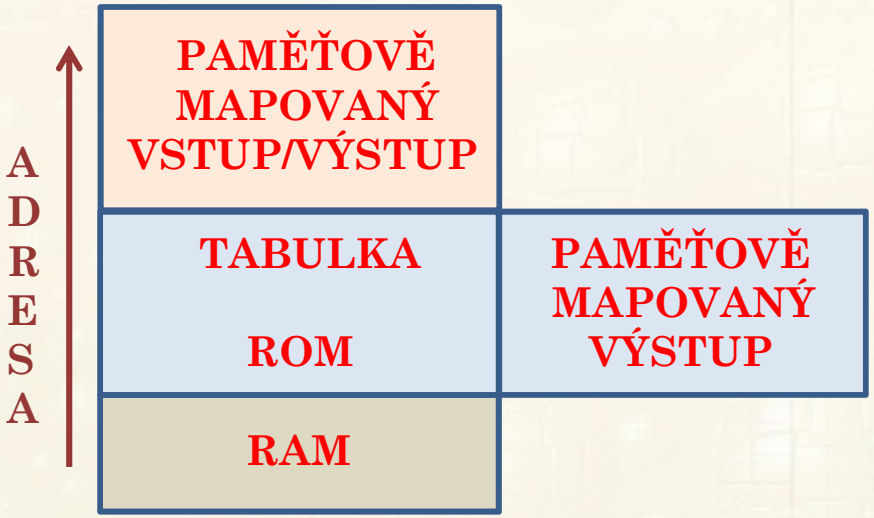




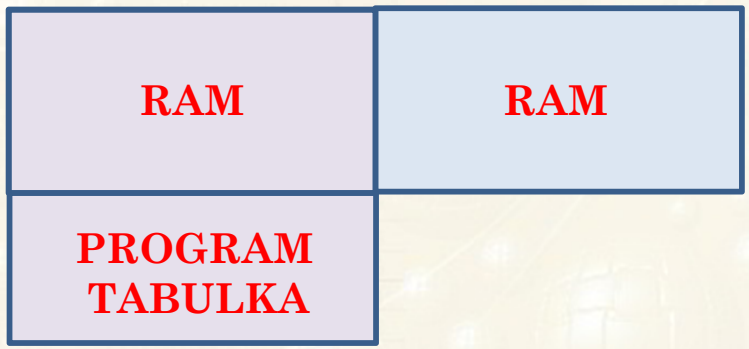
OBECNÉ PRINCIPY PŘIPOJOVÁNÍ PAMĚTÍ A PERIFERIÍ KE SPOLEČNÉ SBĚRNICI

PŘEKRÝVÁNÍ adresových prostorů - Ačkoliv by dvě periferie nebo paměti neměly ležet ve stejném prostoru, může tomu tak být za těchto předpokladů (První tři výrobce uP, čtvrtou výrobce i návrhář)

V DATOVÉM PROSTORU

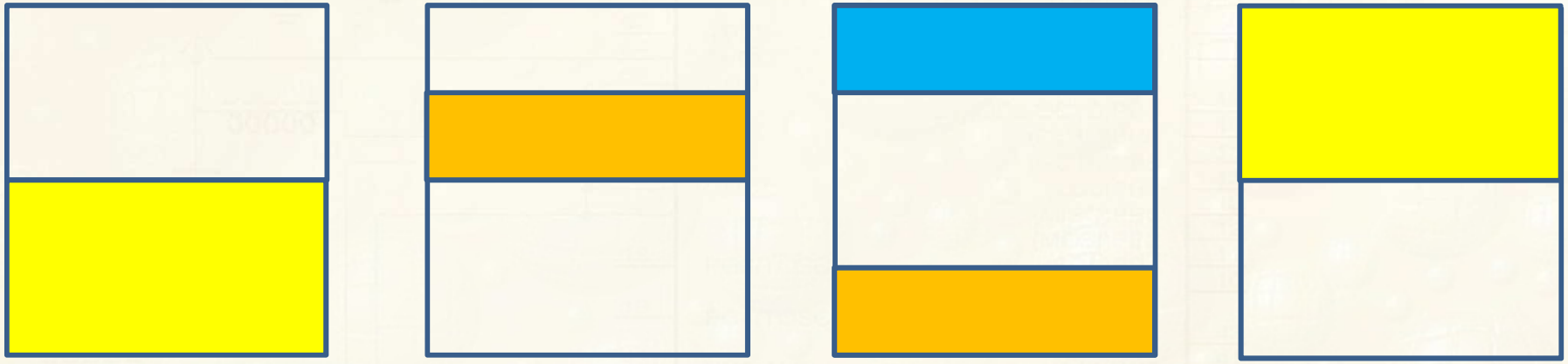


V PROGRAMOVÉM A DATOVÉM PROSTORU



STANDARDNÍ UMÍSTĚNÍ PAMĚTI V PROSTORU 64KB – tj. 0x0000-0xFFFF

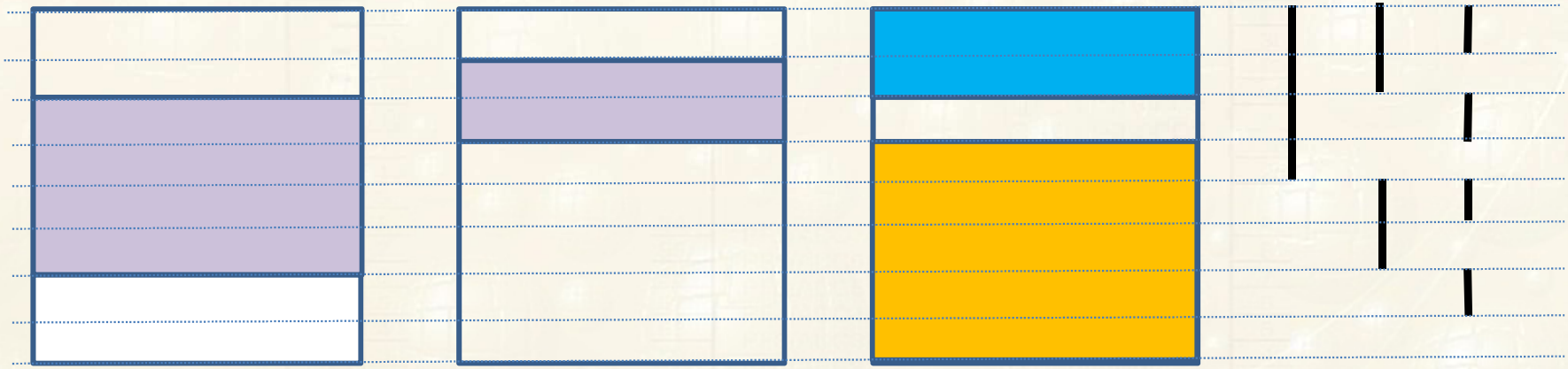
Počáteční adresa umístění paměti modulo kapacita paměti = 0



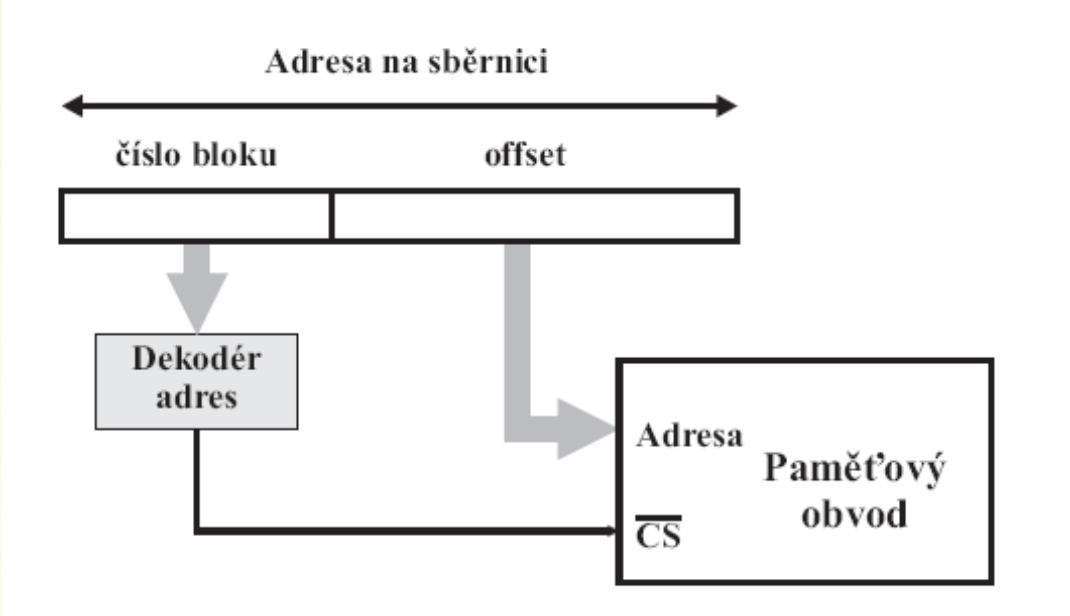
ADRESA 0xYYYY = A15,A14,A13,A12 A11,A10,A9,A8 A7,A6,A5,A4 A3,A2,A1,A0 binárně

NESTANDARDNÍ UMÍSTĚNÍ PAMĚTI

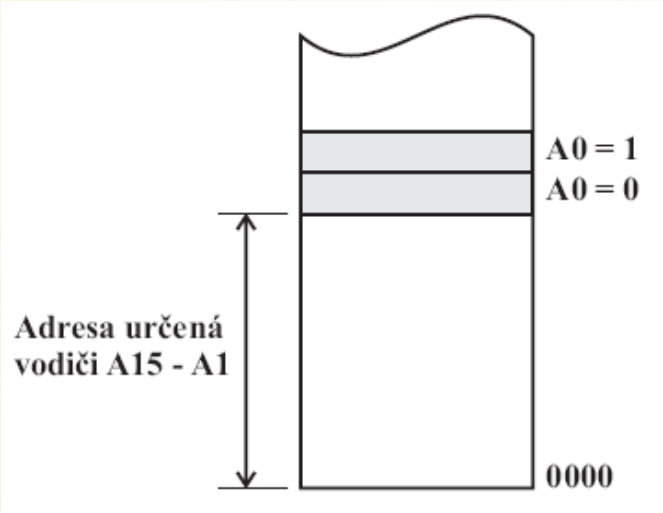
NEVYUŽITÍ CELÉ KAPACITY PAMĚTI



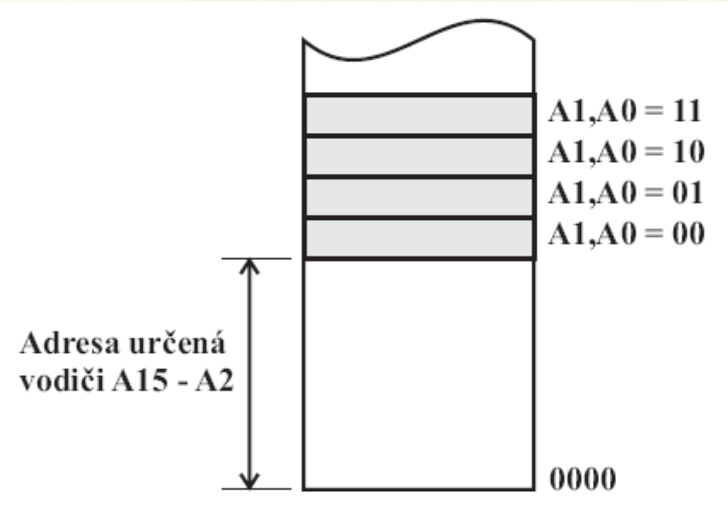
OBECNÉ PRINCIPY PŘIPOJOVÁNÍ PAMĚTÍ A PERIFERIÍ KE SPOLEČNÉ SBĚRNICI



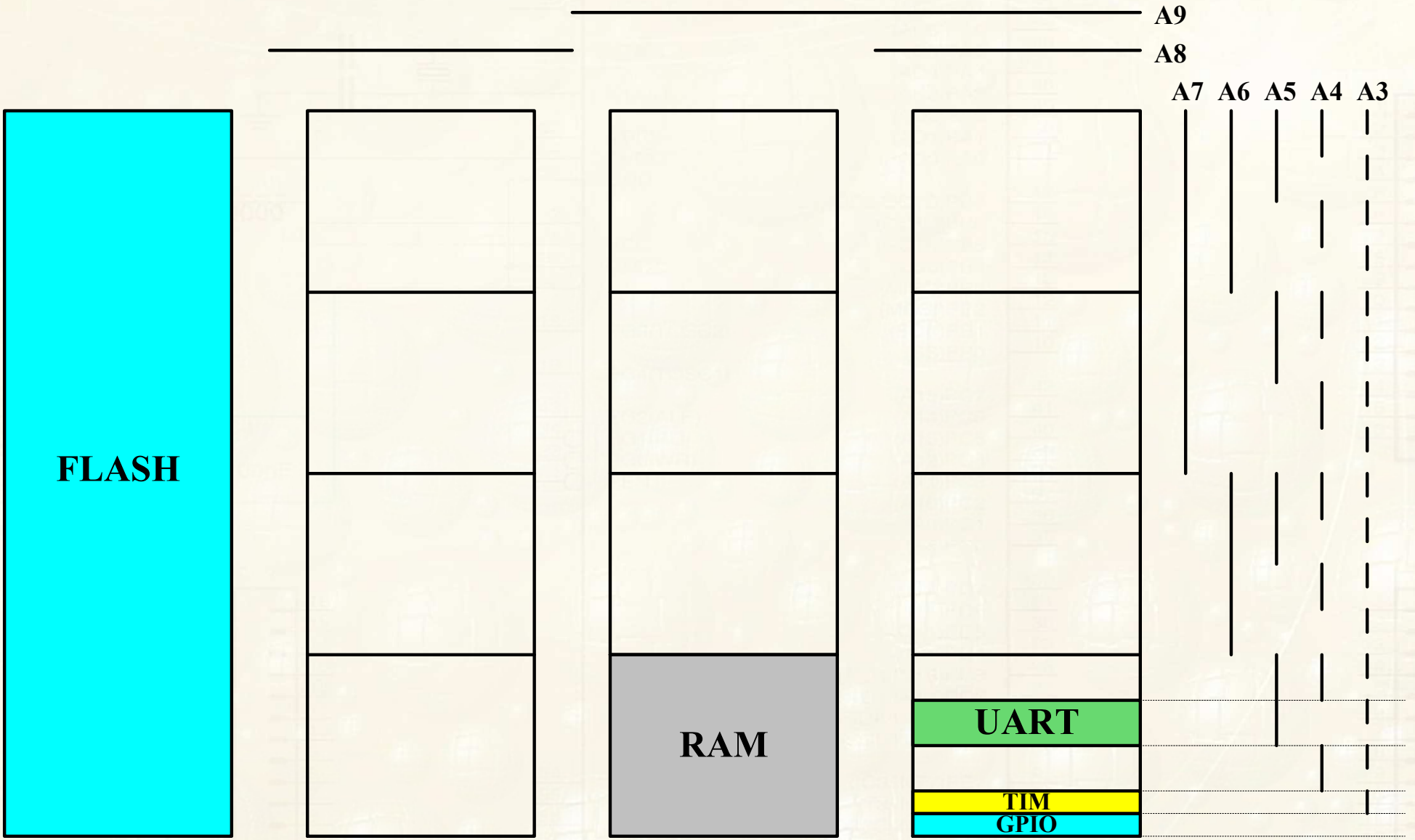
Zarovnání adres na slovo



na dvojnásobné slovo



REALIZACE ADRESOVÉHO DEKODÉRU V FPGA2



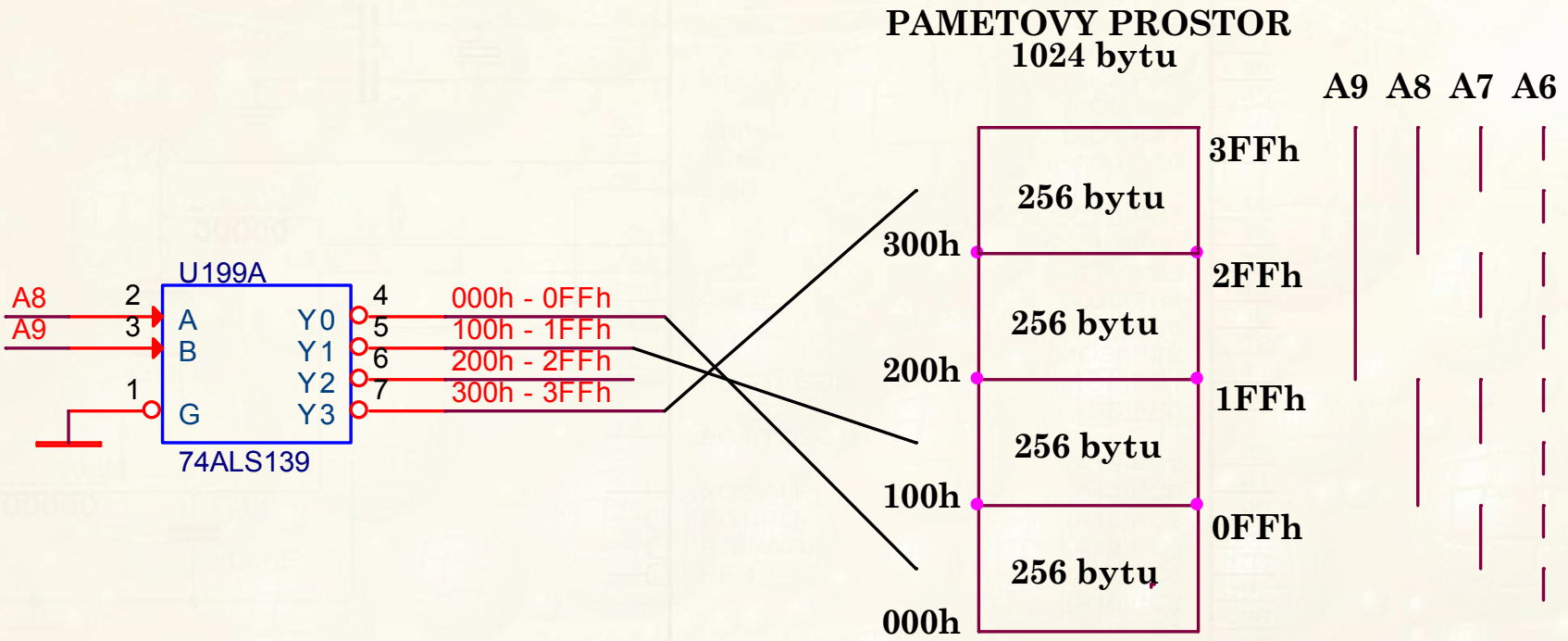
REALIZACE ADRESOVÉHO DEKODÉRU V FPGA2

- Paměť FLASH i všechny periferie jsou umístěny standardně
- Adresové vodiče, které se v požadovaném prostoru mění, slouží k dosažení příslušných adres v prostoru. **Označeny**
- Adresové vodiče, které se v požadovaném prostoru nemění, identifikují požadovaný adresový prostor a jsou proměnnými obvodu vytvářejícího aktivační signál periferie. **Označeny**

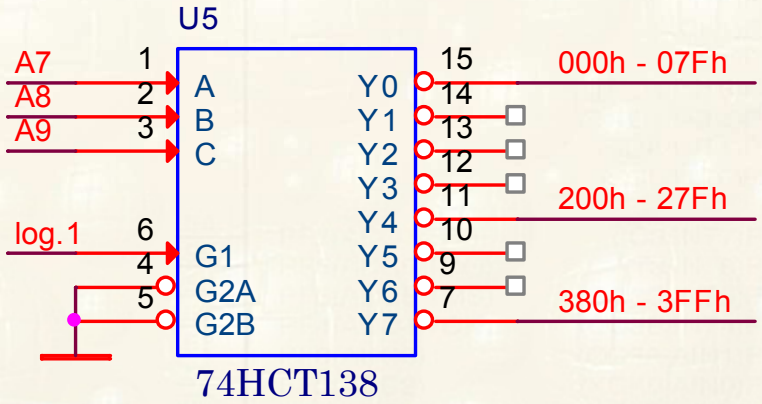
- ADRESA se skládá 0x A9 A8 A7 A6 A5 A4 A3 A2 A1 A0
- FLASH 0x000÷0x0FF = **00 0000 0000 ÷ 00 1111 1111**
- RAM 0x200÷0x23F = **10 0000 0000 ÷ 10 0011 1111**
- GPIO 0x300÷0x307 = **11 0000 0000 ÷ 11 0000 0111**
- TIM 0x308÷0x30F = **11 0000 1000 ÷ 11 0000 1111**
- UART 0x320÷0x32F = **11 0010 0000 ÷ 11 0010 1111**

REALIZACE ADRESOVÉHO DEKODÉRU V FPGA2

- Rozdělení prostoru dekodérem 74139

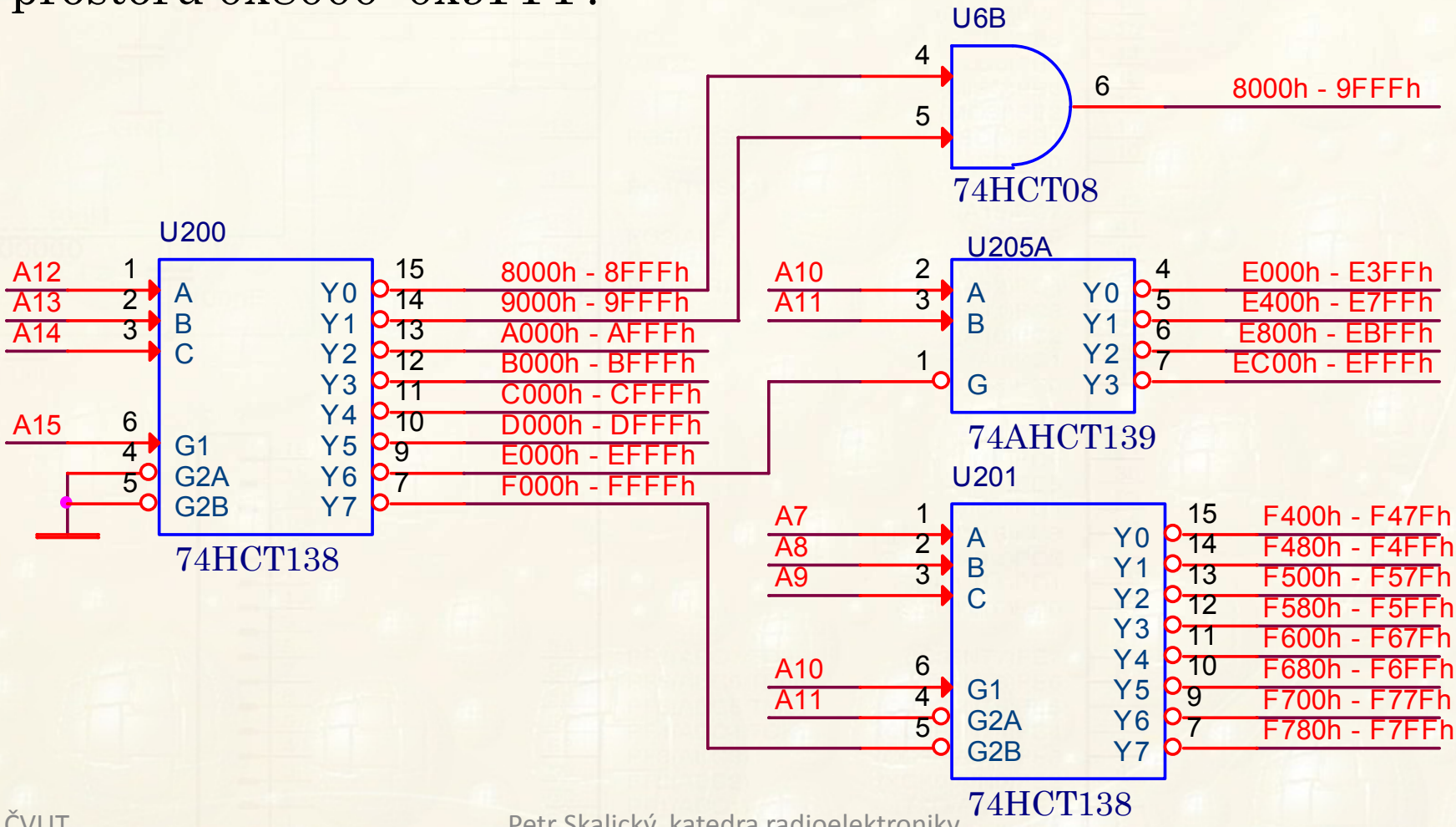


- Rozdělení prostoru dekodérem 74138



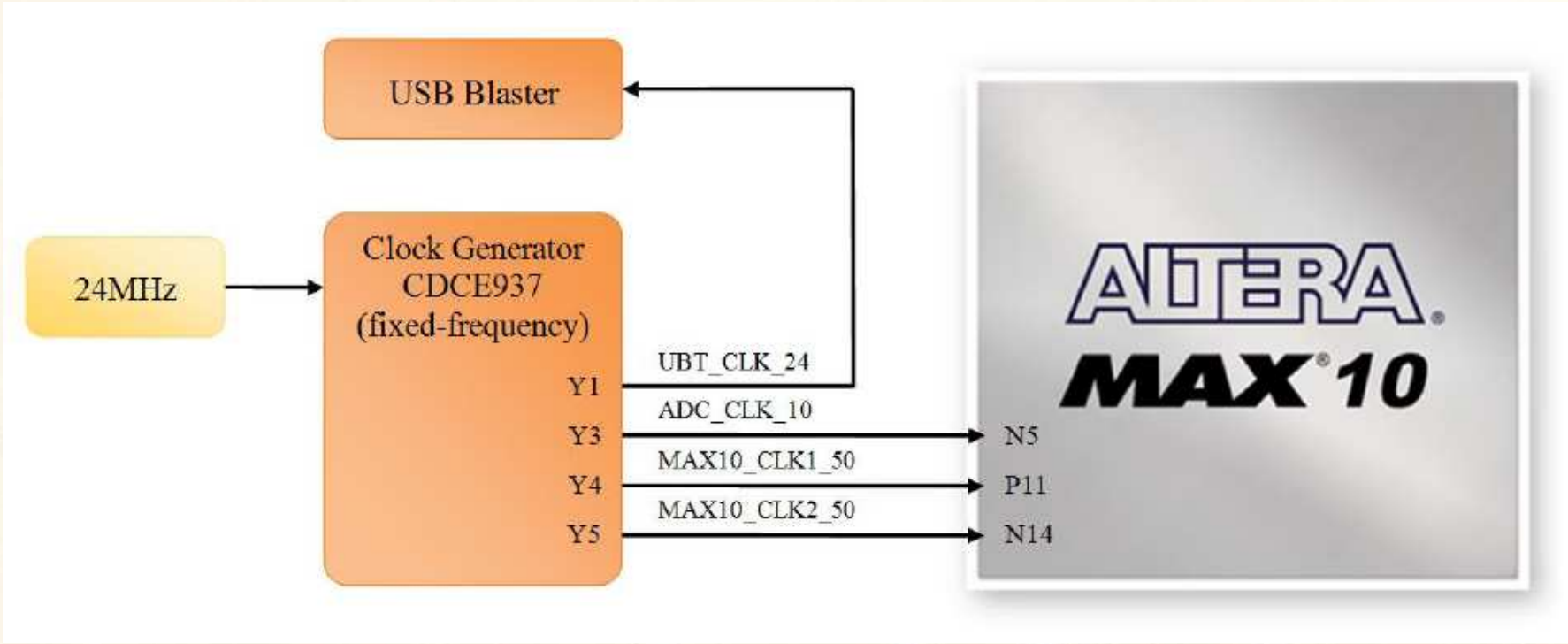
REALIZACE ADRESOVÉHO DEKODÉRU V FPGA2

- Kaskádní řešení – dekodér U200 rozděluje 32kB prostor na 8 4kB prostorů. U205A realizuje čtyři 1kB bloky v prostoru od adresy 0xE000÷0xEFFF, obvod U201 vytváří 8 bloků o kapacitě 128 bytů v prostoru adres 0xF400÷0xF7FF a obvod U6B vytváří 8kB blok v prostoru 0x8000÷0x9FFF.



VÝVODY JEDNOTLIVÝCH PERIFERIÍ NA DESCE DE10-LITE

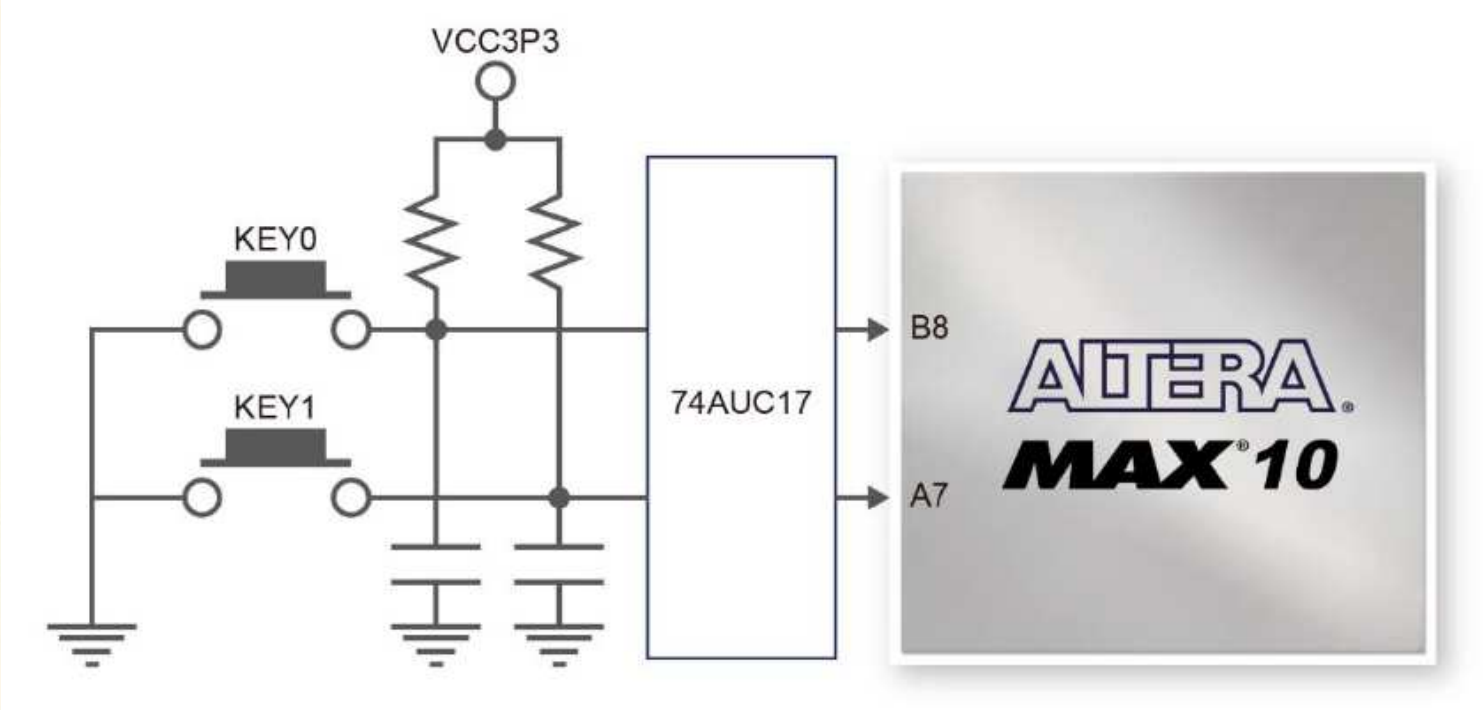
- Hodinový signál

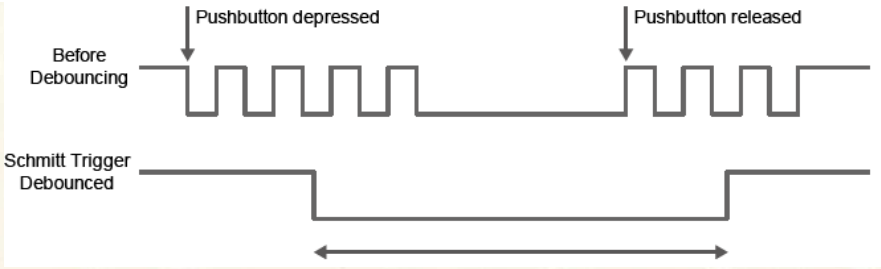


Signal Name	FPGA Pin No.	Description	I/O Standard
ADC_CLK_10	PIN_N5	10 MHz clock input for ADC (Bank 3B)	3.3-V LVTTL
MAX10_CLK1_50	PIN_P11	50 MHz clock input(Bank 3B)	3.3-V LVTTL
MAX10_CLK2_50	PIN_N14	50 MHz clock input(Bank 3B)	3.3-V LVTTL

VÝVODY JEDNOTLIVÝCH PERIFERIÍ NA DESCE DE10-LITE

- Tlačítka





Signal Name	FPGA Pin No.	Description	I/O Standard
KEY0	PIN_B8	Push-button[0]	3.3 V SCHMITT TRIGGER"
KEY1	PIN_A7	Push-button[1]	3.3 V SCHMITT TRIGGER"

VÝVODY JEDNOTLIVÝCH PERIFERIÍ NA DESCE DE10-LITE

- Diody LED

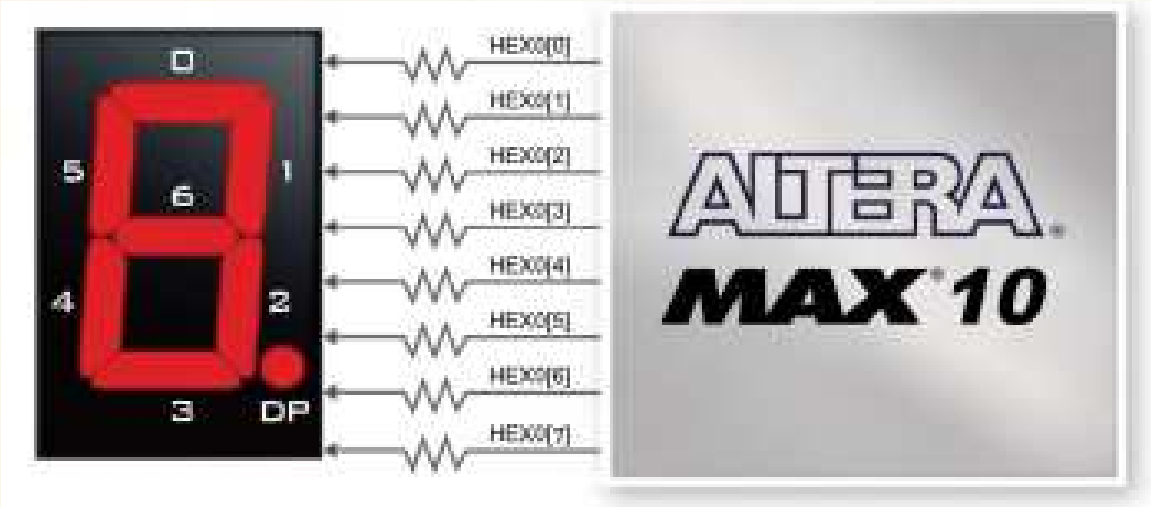
Signal Name	FPGA Pin No.
LEDR0	PIN_A8
LEDR1	PIN_A9
LEDR2	PIN_A10
LEDR3	PIN_B10
LEDR4	PIN_D13
LEDR5	PIN_C13
LEDR6	PIN_E14
LEDR7	PIN_D14
LEDR8	PIN_A11
LEDR9	PIN_B11

- Přepínače

Signal Name	FPGA Pin No.
SW0	PIN_C10
SW1	PIN_C11
SW2	PIN_D12
SW3	PIN_C12
SW4	PIN_A12
SW5	PIN_B12
SW6	PIN_A13
SW7	PIN_A14
SW8	PIN_B14
SW9	PIN_F15

VÝVODY JEDNOTLIVÝCH PERIFERIÍ NA DESCE DE10-LITE

- Sedmisegmentové zobrazovače LED



Signal Name	FPGA Pin No.
HEX00	PIN_C14
HEX01	PIN_E15
HEX02	PIN_C15
HEX03	PIN_C16
HEX04	PIN_E16
HEX05	PIN_D17
HEX06	PIN_C17
HEX07	PIN_D15

- LED1

HEX10	PIN_C18
HEX11	PIN_D18
HEX12	PIN_E18
HEX13	PIN_B16
HEX14	PIN_A17
HEX15	PIN_A18
HEX16	PIN_B17
HEX17	PIN_A16

LED2

HEX20	PIN_B20
HEX21	PIN_A20
HEX22	PIN_B19
HEX23	PIN_A21
HEX24	PIN_B21
HEX25	PIN_C22
HEX26	PIN_B22
HEX27	PIN_A19

VÝVODY JEDNOTLIVÝCH PERIFERIÍ NA DESCE DE10-LITE

- LED3

HEX30	PIN_F21
HEX31	PIN_E22
HEX32	PIN_E21
HEX33	PIN_C19
HEX34	PIN_C20
HEX35	PIN_D19
HEX36	PIN_E17
HEX37	PIN_D22

LED4

HEX40	PIN_F18
HEX41	PIN_E20
HEX42	PIN_E19
HEX43	PIN_J18
HEX44	PIN_H19
HEX45	PIN_F19
HEX46	PIN_F20
HEX47	PIN_F17

- LED5

HEX50	PIN_J20
HEX51	PIN_K20
HEX52	PIN_L18
HEX53	PIN_N18
HEX54	PIN_M20
HEX55	PIN_N19
HEX56	PIN_N20
HEX57	PIN_L19

DÚ - PŘÍPRAVA NA REALIZACI LKO NA DESCE DE10

Navrhněte v prostředí Quartus II a ověřte simulací návrh převodníku BCD na 7 segmentový displej pro toto uspořádání segmentů.

Segmentovky jsou se společnou anodou (log.0=segment svítí).
Logickou funkci pro jednotlivé segmenty zrealizujte následující metodou:

a) Formou využívající pouze obvody NOR se dvěma vstupy
b) Formou využívající pouze obvody NAND se dvěma vstupy
c) AND-OR-INVERT s obvodem 74F64
d) Pamětí s kapacitou 16x1 bit v zápise AHDL (návod v závěru)
e) Reed-Mullerovou formou logické funkce
f) Multiplexerem se třemi adresovacími vstupy
g) Pomocí dekodéru 74138 a hradla AND

REALIZACE PAMĚTI ROM (LKO) V JAZYCE AHDL

```
SUBDESIGN ROM_AHDL
(   A[3..0]       : INPUT;
    Y             : OUTPUT; )
BEGIN
        TABLE
        A[3..0]  =>       Y;
        H"0"     =>       0;
        H"1"     =>       1;
        H"2"     =>       0;
Atd.
        H"9"     =>       0;
        H"A"     =>       1;
        H"B"     =>       1;
        H"C"     =>       1;
        H"D"     =>       1;
        H"E"     =>       1;
        H"F"     =>       1;
        END TABLE;
 END;
```

OVĚŘENÍ LKO V PROSTŘEDÍ QUARTUS II NA DESCE DE10_LITE

- Návod na stránkách předmětu - *Návody a užitečné informace - Quartus Prime Lite - Vytvoření projektu*
- Zrealizujte návrh projektu, jeho překlad, vytvoření Univerzitní simulace, grafické interpretace vstupních signálů pro simulaci.
- Ověřený návrh zrealizujte na vývojové desce DE10-Lite. V návodu na stránkách předmětu - *Návody a užitečné informace - Quartus Prime Lite – Instalace* najdete instalaci ovladače **usb-blaster.**
- Jednotlivým vývodům navrženého obvodu přiřaďte v položce Assigment - Pin Planner odpovídající vývody Přepínačů a diody LED. viz str.12÷14.
- Projekt přeložte a nahrajte – Programming