BUDAPESTI MŰSZAKI ÉS GAZDASÁGTUDOMÁNYI EGYETEM
VILLAMOSMÉRNÖKI ÉS INFORMATIKAI KAR
MÉRÉSTECHNIKA ÉS INFORMÁCIÓS RENDSZEREK TANSZÉK

Digitális technika VIMIAA02 10. hét

Fehér Béla, Benesóczky Zoltán
BME MIT

Processzor belső kommunikációja

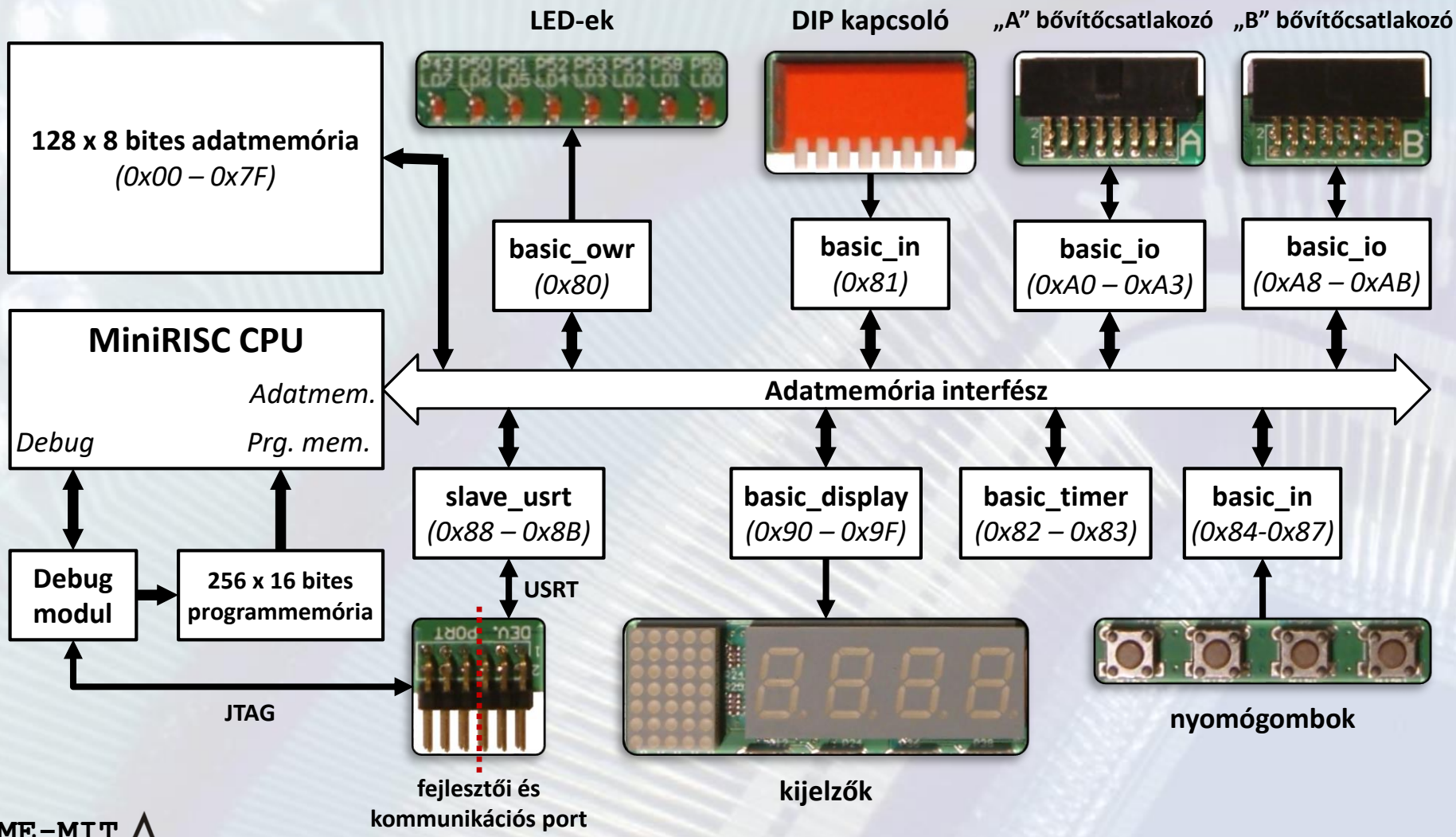
- **A mikroprocesszoros rendszerek összetett digitális rendszerek, több modulból állnak**
- **Eddig csak a központi egység felépítését, az un. CPU struktúráját és működését vizsgáltuk:**
  - Ez két fő részegységből áll: ***vezérlő*** és ***adatsruktúra*** (ALU + regiszterek) Ezek mindegyike bonyolult részrendszer, de önmagában szinte működésképtelen.
  - A vezérlő a működéséhez szükséges ***utasításokat*** a **programmemória (kódmemória) interfészen** keresztül éri el.
  - A **processzor** értelmes működésének feltétele az ***adatmemória és a perifériák*** elérése. Ezt az **adatmemória interfész** biztosítja.

Processzor belső kommunikációja

- **Adat memória szerepe**
  - ***Változók*** tárolása
    STACK memória terület (*Többnyire az adat memória egy részében van kialakítva* a szubrutinokhoz és interrupthoz szükséges STACK memória terület is. Azonban *kisebb CPU-k esetén, azt önálló hardver STACK valósítja meg*. Pl. MiniRISC.)
- A **perifériák** szerepe:
  - Lehetővé teszik, hogy a ***CPU a környezettel kommunikáljon***, onnan adatokat kapjon és oda adatokat küldjön.
    A perifériákat sokszor a processzor adat memória területére illesztik. (Mi csak ezt tárgyaljuk.) Így a periféria regisztereket ugyanazokkal az utasításokkal lehet elérni, mint a memóriát.

Processzor belső kommunikációja

(MiniRSIC egyszerűsített blokkvázlata)



Processzor belső kommunikációja

- A mikroprocesszoros (uC) rendszerek adatátviteli interfészeit is processzoros ***buszok***nak nevezzük. (Általánosan a ***busz*** valamilyen funkciót ellátó jelek halmaza.)
- A buszok teszik lehetővé az ***egységek közötti kommunikációt***.
- **A processzoros busz részei: címbusz, adatbusz, vezérlő busz**
  - **Címbusz:** a rákapcsolódó egységek regisztereinek megcímzésére (kijelölésére) szolgál.
  - **Adatbusz:** ezen küld vagy fogad adatot a CPU a megcímzett egységtől. A belső (chip-eken belüli) adatbuszból kettő van, ***bemeneti adatbusz*** és ***kimeneti adatbusz***.
  - **Vezérlő busz:** A CPU ezzel jelöli ki az **adatmozgatás irányát** (olvasás vagy írás) és vezérli ezek **időbeli lezajlását**. (A vezérlő busznak egyéb, itt nem részletezett funkciói is vannak.)

Processzor belső kommunikációja

- A mikroprocesszor rendelkezik ***belső busszal*** és rendelkezhet egyedi ***külső busszal.***
- A ***belső busz*** a belső (chip-en belüli) memóriák és perifériák elérésére szolgál.
- ***A külső busz*** *teszi lehetővé, hogy egy processzorhoz* ***külső memóriát, perifériát lehessen illeszteni.*** (A tárgyban a külső buszt nem részletezzük.)

A mikroprocesszoros busz

- **A processzoros busz részei részletesen:**
  - **Címbusz ADDR[n:0]**
  - **Adatbusz *belső busznál* külön bemeneti adat buszból (D_IN[m:0]) és külön kimeneti adatbuszból (D_OUT[m:0]) áll (m: 7, 15, 31...).**
  - **Vezérlő busz (sok egyedi jel összessége):**
    - **Rendszerjelek: CLK, RST,**
    - **Irányvezérlő jelek: READ, WRITE,**
    - **Megszakítás vezérlő jelek: IRQi,** IACK (utóbbi a MiniRISC-nél nincs).
    - Arbitrációs jelek (nem tanuljuk)
    - Átvitelvezérlő jelek (nem tanuljuk)

A mikroprocesszoros busz

- A buszra kapcsolódó egységek között megkülönböztetünk **MASTER** és **SLAVE** egységeket**.**
- **MASTER*:*** ***a buszt vezérlő egység***. (Ez vezérli pl. a többi egységbe történő adat írást, az adat olvasást stb.)
  Busz master ***elsősorban a CPU*** (processzor), de speciális periféria is lehet. A továbbiakban masterként csak a CPU-ra hivatkozunk.
- **SLAVE:** a **memória** és a **perifériák**.

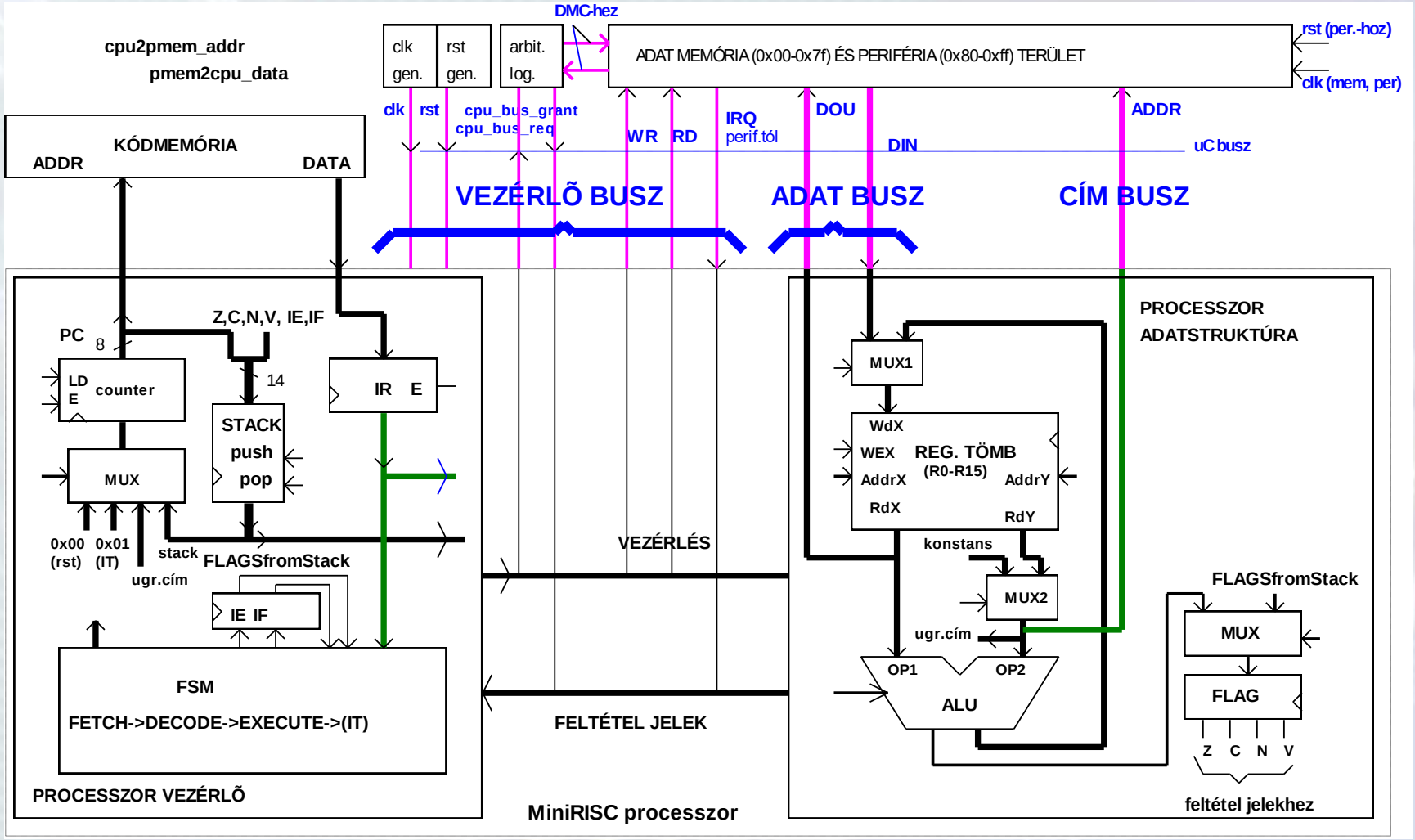
A mikroprocesszoros busz

- **MASTER:** Elsősorban a ***processzor*** (de lehet speciális periféria is). **Ez v*ezérelheti a buszt***. Kiadja a ***címet***, meghatározza az ***adatáramlás irányát*** (írás/olvasás) ***kezdeményezi az átvitelt*** és ***vezérli a működést***, ***íráskor kiadja az adatot***. A MiniRISC rendszerben a processzor (és a DMA vezérlő) lehet MASTER. (DMA-val a tárgyban nem foglalkozunk.)

- **SLAVE:** A ***memória*** és a ***perifériák***. ***Figyeli a buszt***, felismeri/dekódolja ***a címet*** és a ***parancsokat*** (pl. írás/olvasás), azonosítás esetén (***ha a címe egyezik*** a master által kiadottal) ***végrehatja a kérést*** a megfelelő adat fogadásával (ha írja a MASTER) vagy kiadásával (ha olvassa a MASTER).

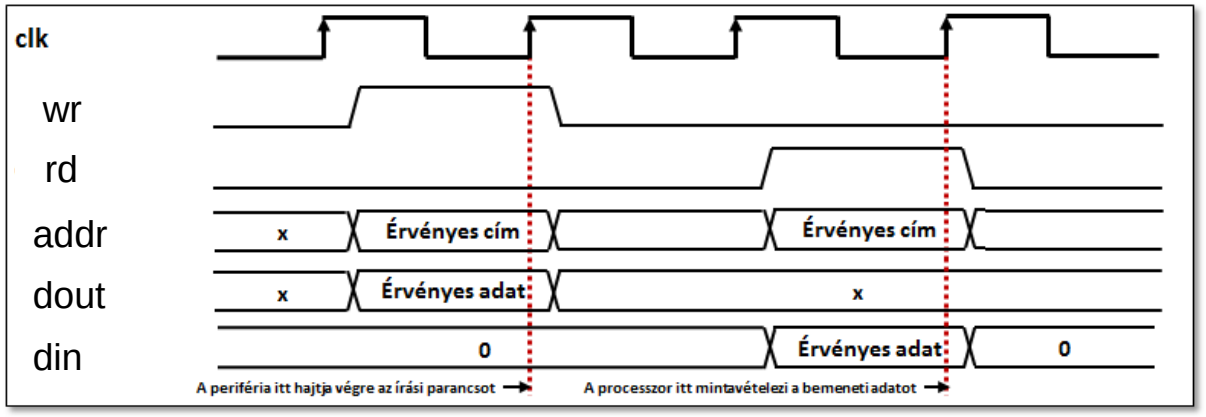
A mikroprocesszoros busz

MiniRISC busz



A busz adatátviteli ütemezése

- A buszon az adatátvitel ütemezése lehet **szinkron** és **aszinkron** (csak a szinkron ütemezéssel foglalkozunk)
- **Az adatátvitel ütemezése szinkron módon**
  - Az adatátvitelt a busz órajel ciklusai ütemezik
    - Sokfajta busz ütemezési protokoll létezik, jellemzően egy átvitel több órajel ciklus ideig tart.
      A MiniRISC busz egyetlen órajel alatt hajtja végre az átvitelt.
    - Az alábbi idődiagrammon egy írás (wr), majd egy olvasás (rd) történik (MiniRISC).

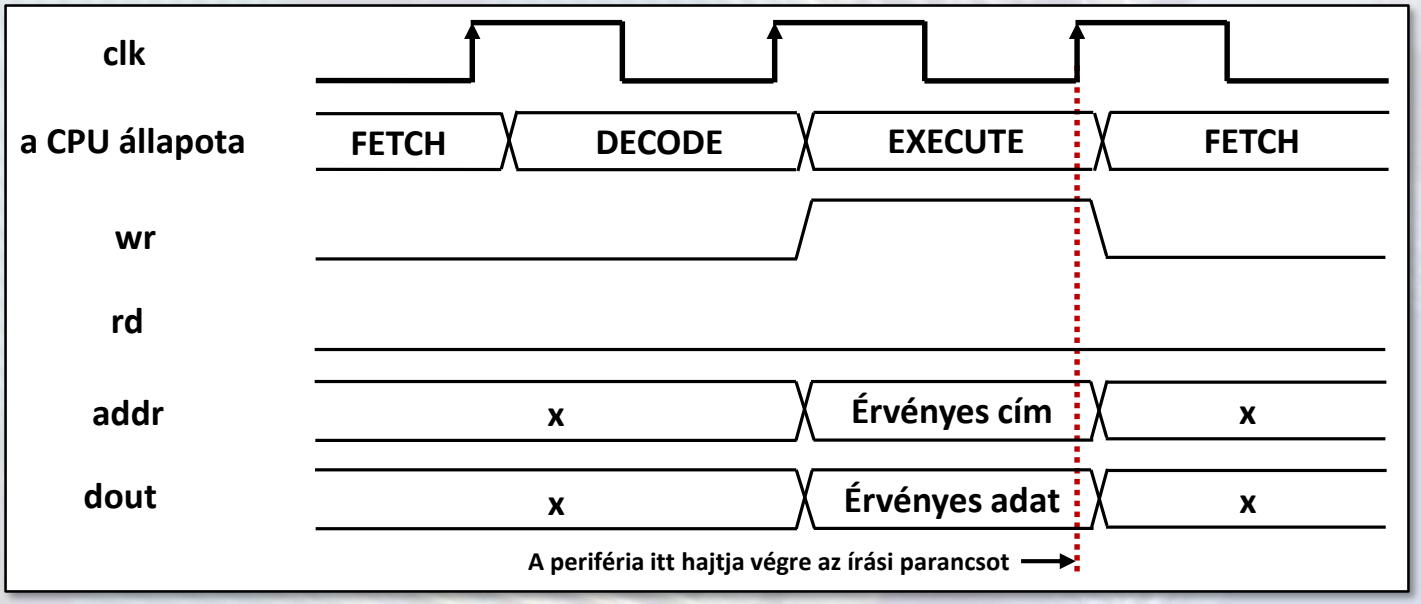


MiniRISC processzor – Interfészek

(Adatmemória interfész – Írási buszciklus)

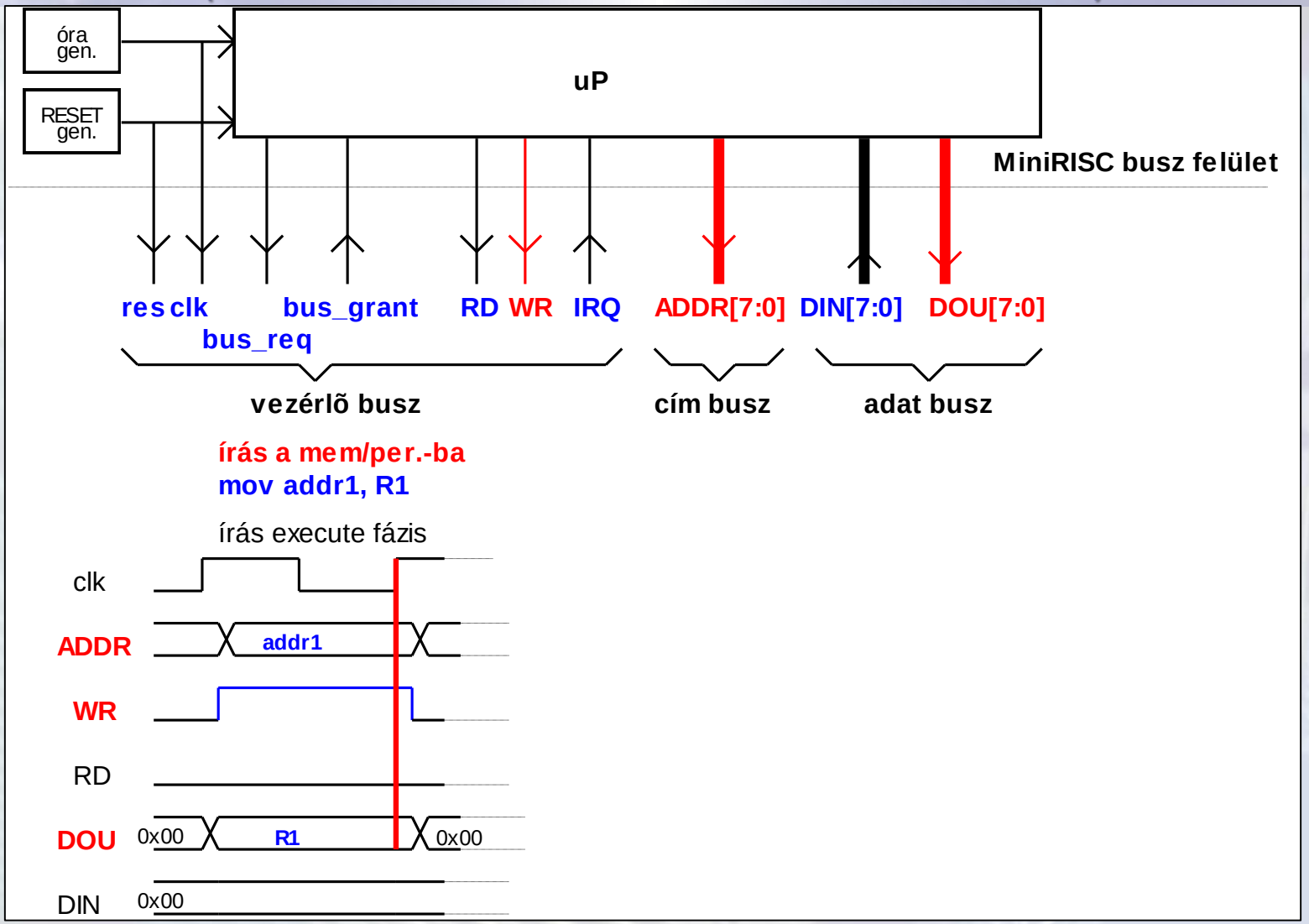
Az adatmemória interfész írási (write) ciklusa

- Az írási ciklust a végrehajtási (execute) fázisban 1 órajelciklus ideig aktív **WR** (cpu2dmem_wr) jel jelzi
- Az írási ciklus alatt a **ADDR[7:0]** (cpu2dmem_addr) címbuszon a cím stabil
- Az írási ciklus alatt a **DOU[7:0]** (cpu2dmem_data) adatbuszon az adat stabil és a **WR alatti órajel felfutó élére az adat beíródik a megcímzett periféria regiszterébe**.



MiniRISC processzor – Interfészek

(Adatmemória interfész – Írási buszciklus)

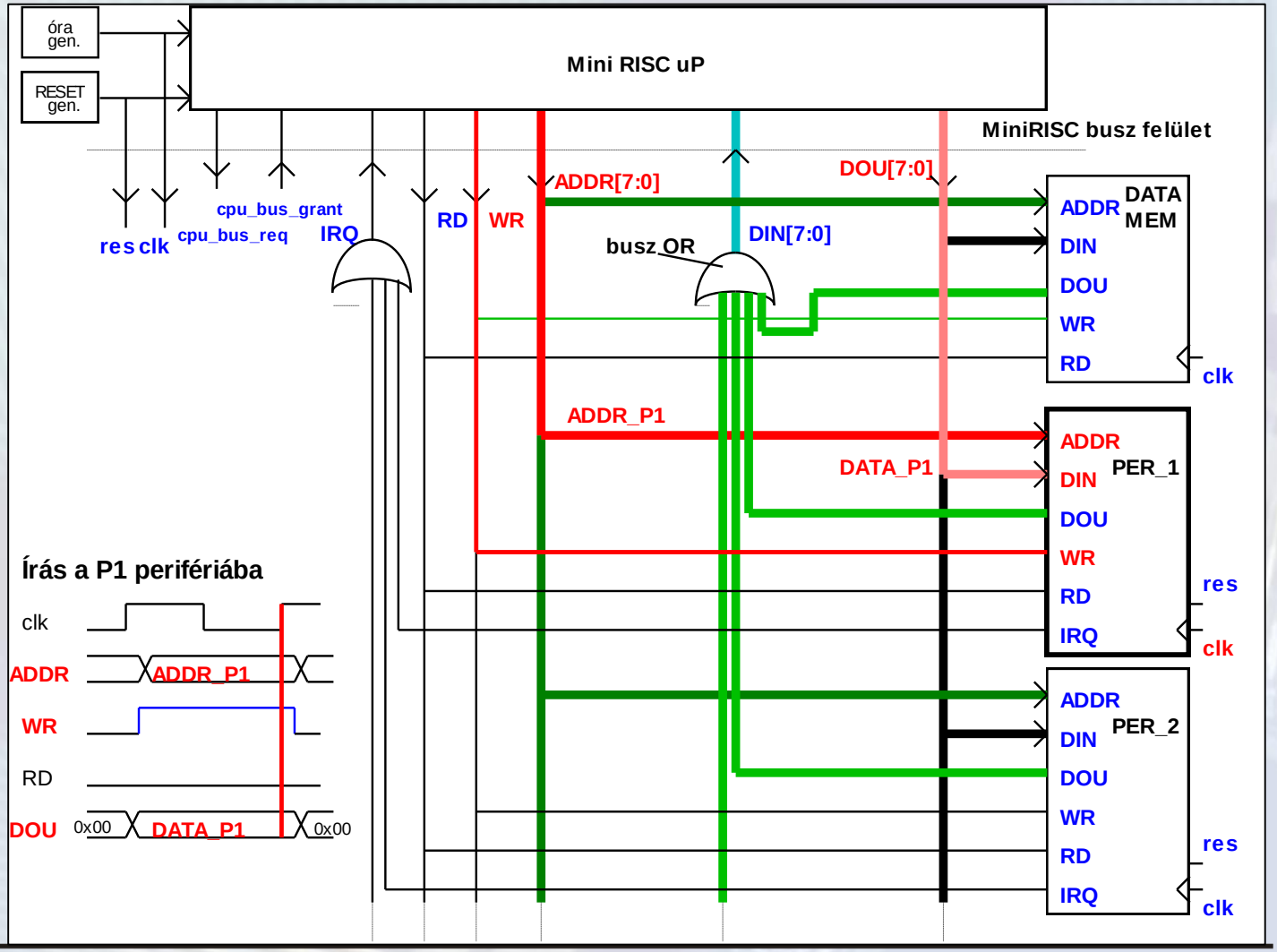


A mikroprocesszoros busz

MiniRISC: Írás a PER_1 perifériába

- Aktív jelek:
- ADDR[7:0]: **ADDR_P1**
- DOU[7:0]: **DATA_P1**
- WR: **az írás alatt 1 értékű**

Minden periféria DIN adat bemenete rákapcsolódik a uC adatbuszára, de csak a megcímzettbe (amelyikhez a kiadott cím tartozik) íródik be az adat.

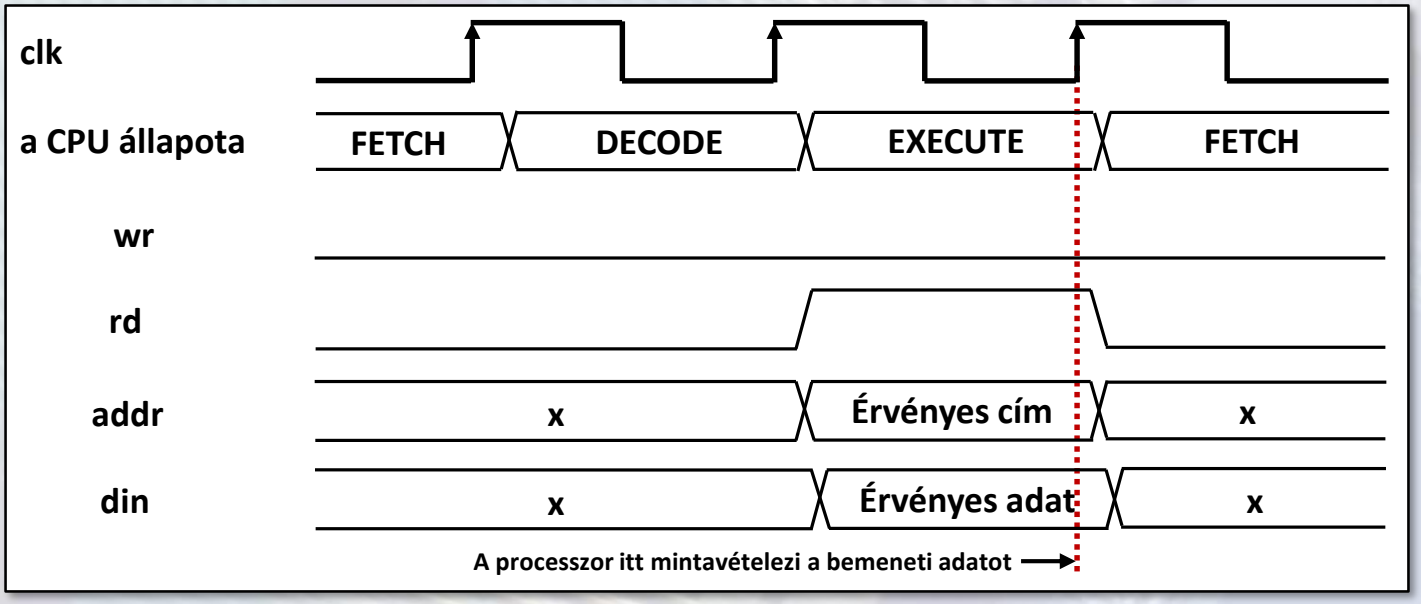


MiniRISC processzor – Interfészek

(Adatmemória interfész – Olvasási buszciklus)

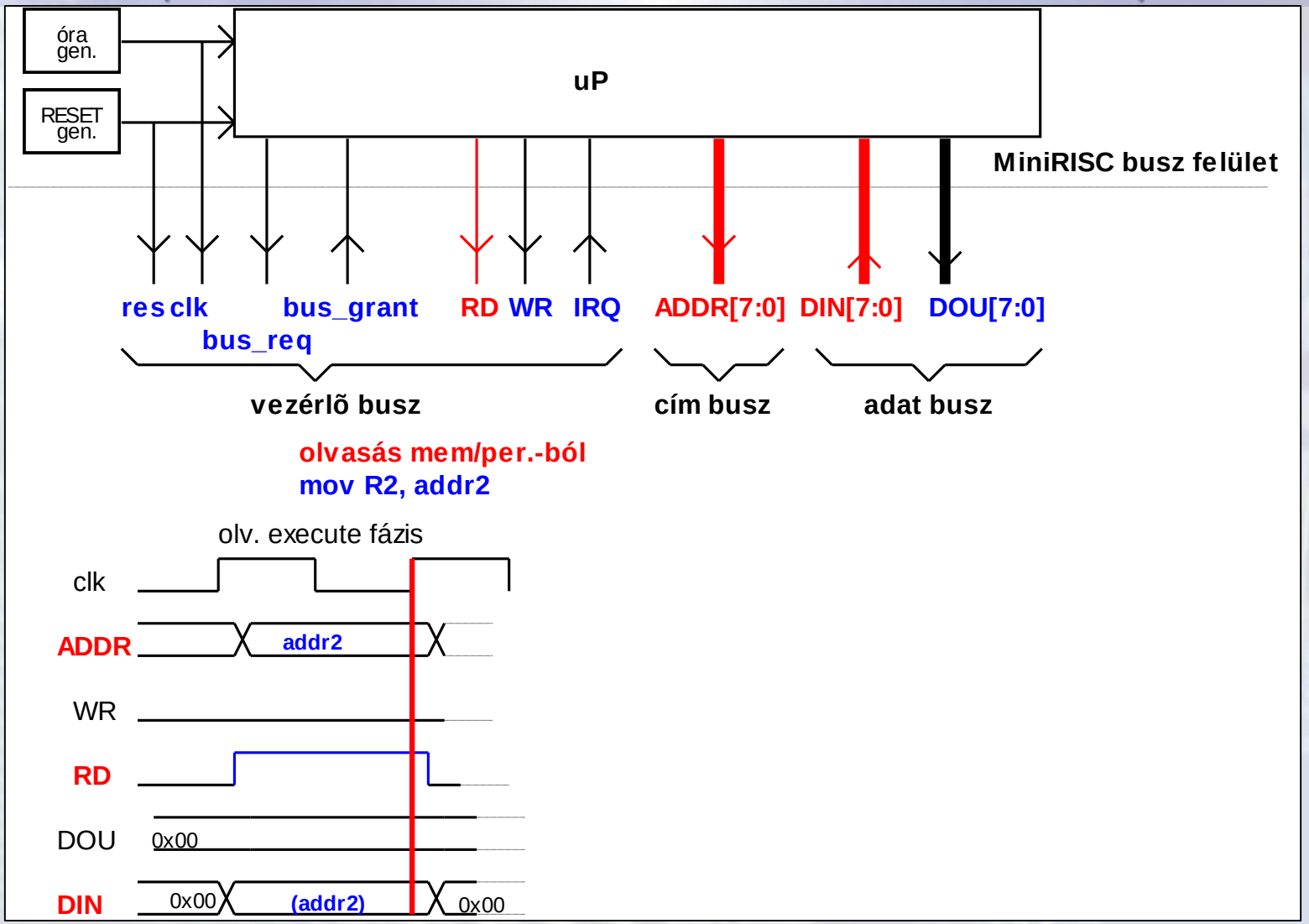
Az adatmemória interfész olvasási (read) ciklusa

- Az olvasási ciklust a végrehajtási (execute) fázisban 1 órajelciklus ideig aktív **RD** (cpu2dmem_rd) jel jelzi
- Az olvasási ciklus alatt a **ADDR[7:0]** (cpu2dmem_addr) címbuszon a cím stabil
- Az **olvasási ciklus alatt a kiválasztott periféria** a **DIN[7:0]** (dmem2cpu_data) **olvasási adatbuszra kapcsolja az adatát**, a **többi periféria** ezalatt inaktív, **nullával hajtja meg az adatkimenetét**.



MiniRISC processzor – Interfészek

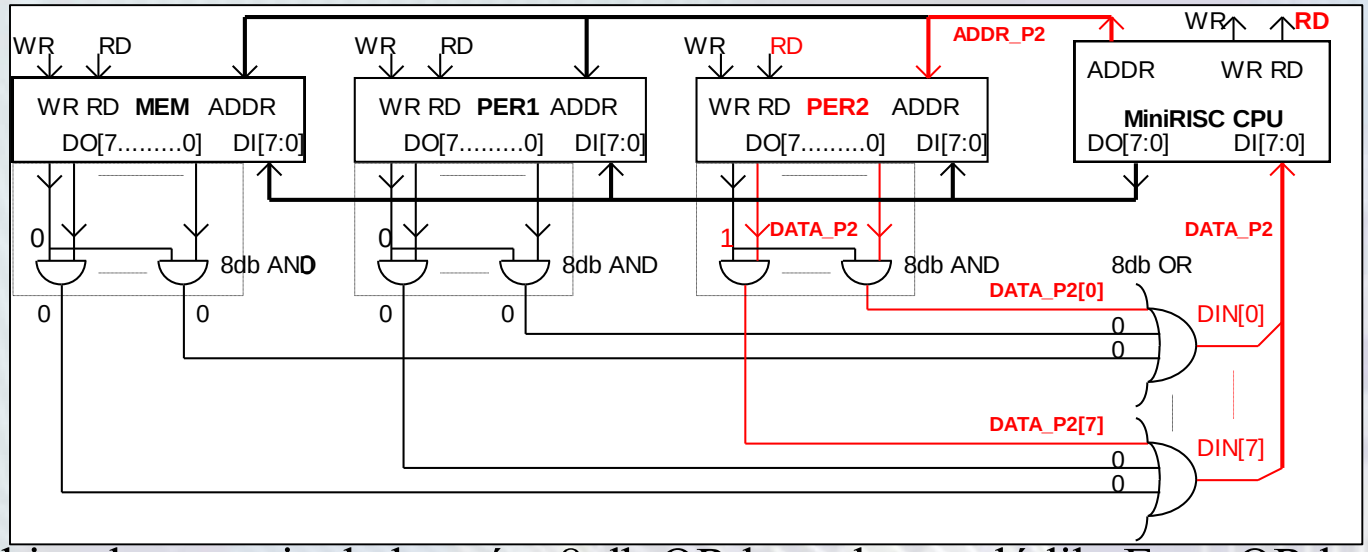
(Adatmemória interfész – Olvasási buszciklus)



A mikroprocesszoros busz

MiniRISC: Olvasás a PER_2 perifériából

- Aktív jelek:
- ADDR[7:0]: **ADDR_P2**
- DIN[7:0]: **DATA_P2**
- RD: **az olvasás alatt 1 értékű**

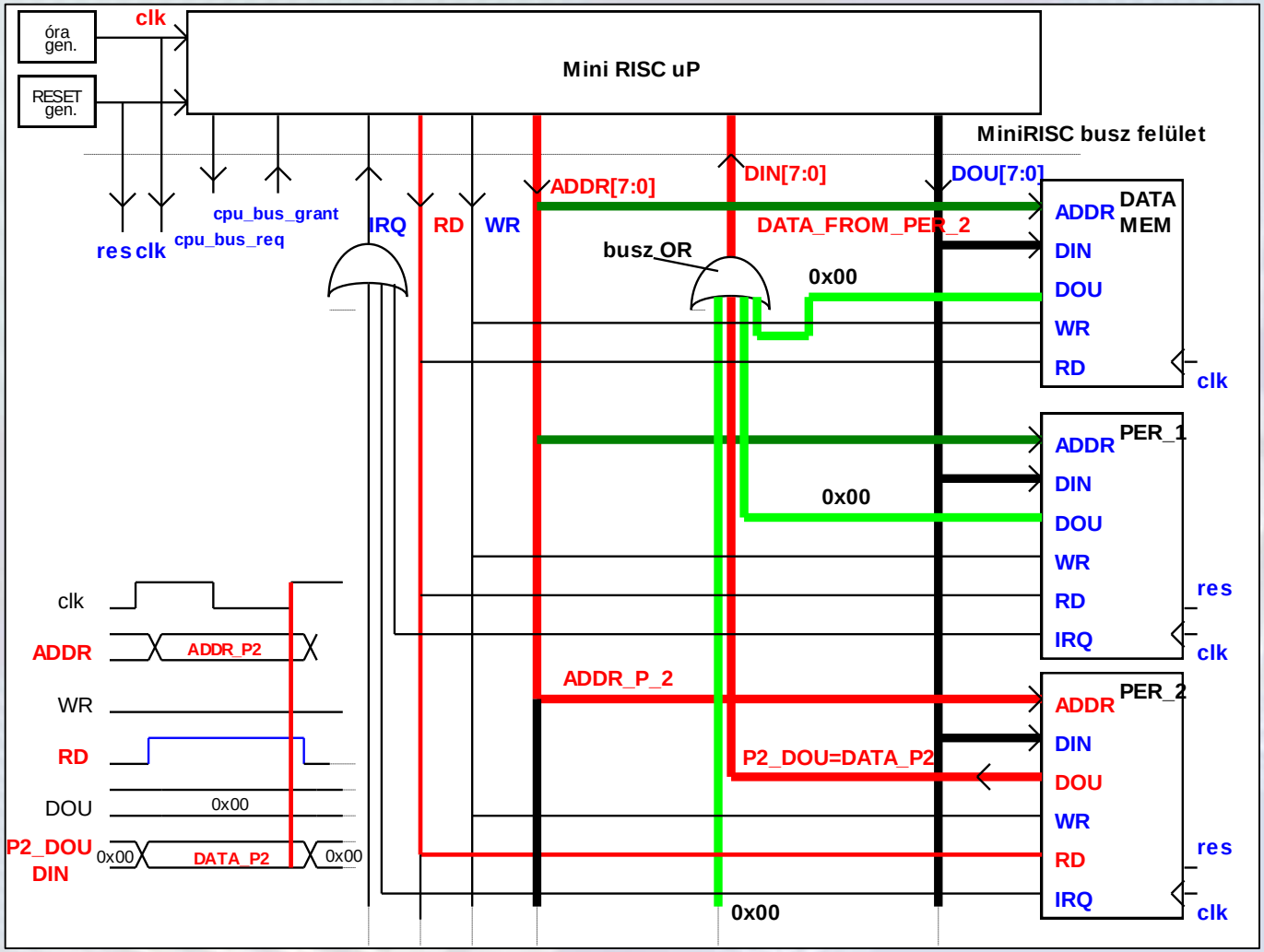


A processzor 8 bites bemeneti adatbuszára 8 db OR kapu kapcsolódik. Ezen OR kapuk bemeneteire kapcsolódnak a perifériák adat kimenetének bitjei (az i-edik OR kapura az i-edik adat bit). **A rajz szerinti példában (pirossal jelölve) a megcímzett periféria** (most PER2) **az adatával hajtja meg** (DATA_P2), **a többi pedig 0-val**. Így **a processzor DIN bementére a megcímzett periféria adata kerül**, amit eltárol az utasításnak megfelelő regiszterébe. A perifériák AND kapui és a processzor adatbuszához rendelt OR kapuk együtt egy (elosztott) multiplexert alkotnak. Az AND kapuk a perifériákhoz, az OR kapuk a DIN bemeneti adat buszhoz tartoznak.

A mikroprocesszoros busz

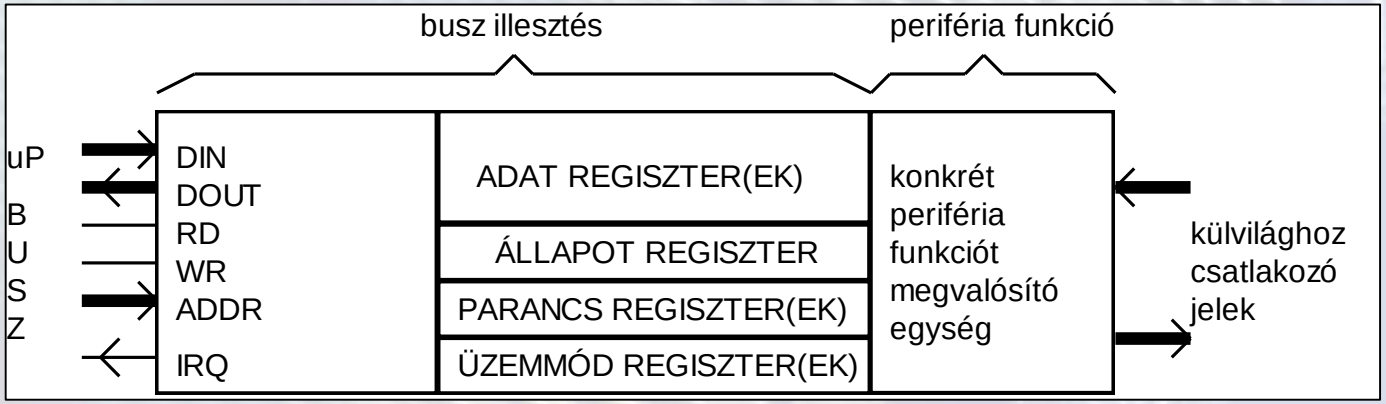
MiniRISC: Olvasás a PER_2 perifériából

- Az előbbi ábra más rajtechnikai jelöléssel:
- Itt az adatbusz-hoz tartozó 8 db OR kaput egyet-len „busz OR” kapuval jelöltük. A perifériák AND kapu nem látszanak, azok a periféria részei.
- Az aktív jeleket pirossal jelöltük.

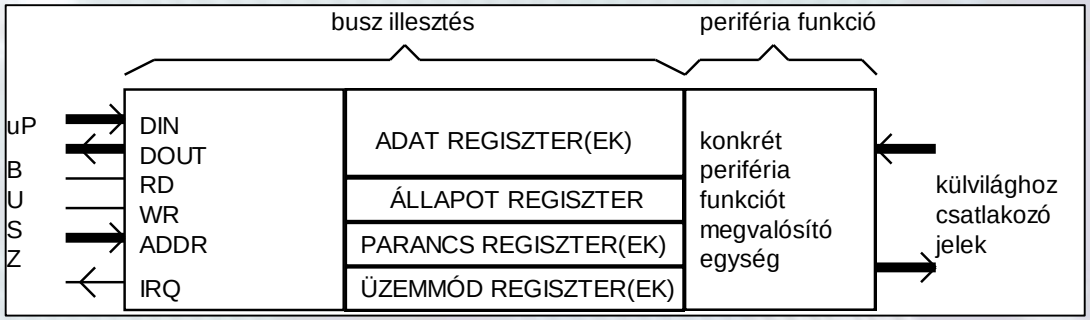


MiniRISC processzor – Perifériaillesztés

- **A processzor a perifériákat a külvilággal való kommunikációra használja**.
- Az információ a periféria **programozói felületén (írható és/vagy olvasható regiszterek)** keresztül küldhető ki, olvasható be.
- A programozói felület különféle információkhoz rendelt írható és/vagy olvasható regisztereket tartalmaz.



MiniRISC processzor – Perifériaillesztés



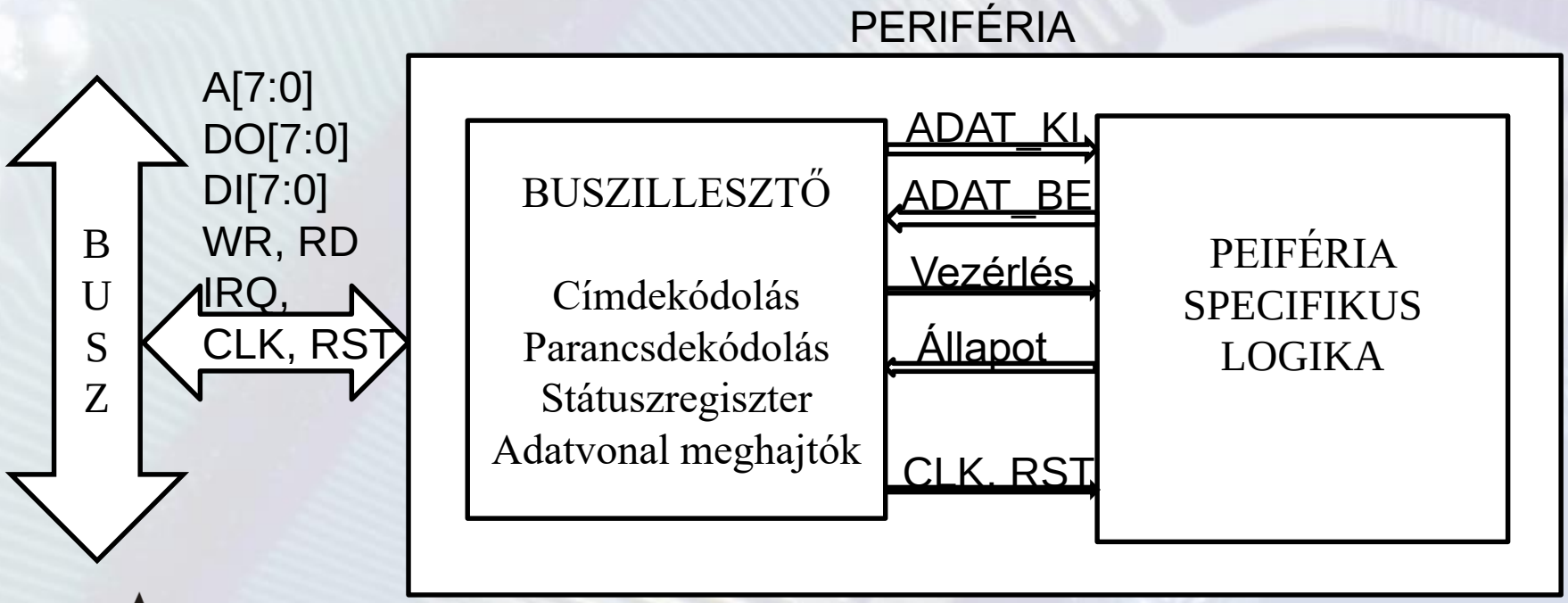
- **Üzemmód regiszter:** A ***periféria működési módjának beállítására*** szolgál. Egy-egy periféria egy adott feladatcsoport megoldását teszi lehetővé *konfigurálható üzemmódok* segítségével (pl. egy timer egységgel időzítési, számlálási, impulzus generálási feladatok oldhatók meg).
- **Parancs regiszter:** A periféria valamely ***működését lehet vele kezdeményezni***. Pl. A/D konverzió indítása.
- **Állapot (státus) regiszter:** A periféria ***állapotáról ad információkat***. Az állapotregiszter bitjei ún. megszakítást is kérhetnek, ha az engedélyezett.
- Sokszor az üzemmód és parancs reigsztert összevonják és **vezérlő regiszternek** nevezik.
- **Adat regiszter:** ***Ide kell beírni itt lehet kiolvasni az adatot*** (ugyanarról a címről nem feltétlenül ugyanazt, amit beírtunk, lásd később USRT).

MiniRISC processzor – Perifériaillesztés

- **A perifériaillesztési feladat lépései**
  - A periféria típusa alapján az igények felmérése
    - Regiszter szám, igény és használati mód meghatározása (írható, olvasható)
    - Pl. regiszterek: Adat be-, adat ki, parancs, státusz, üzemmód, stb.

MiniRISC processzor – Perifériaillesztés

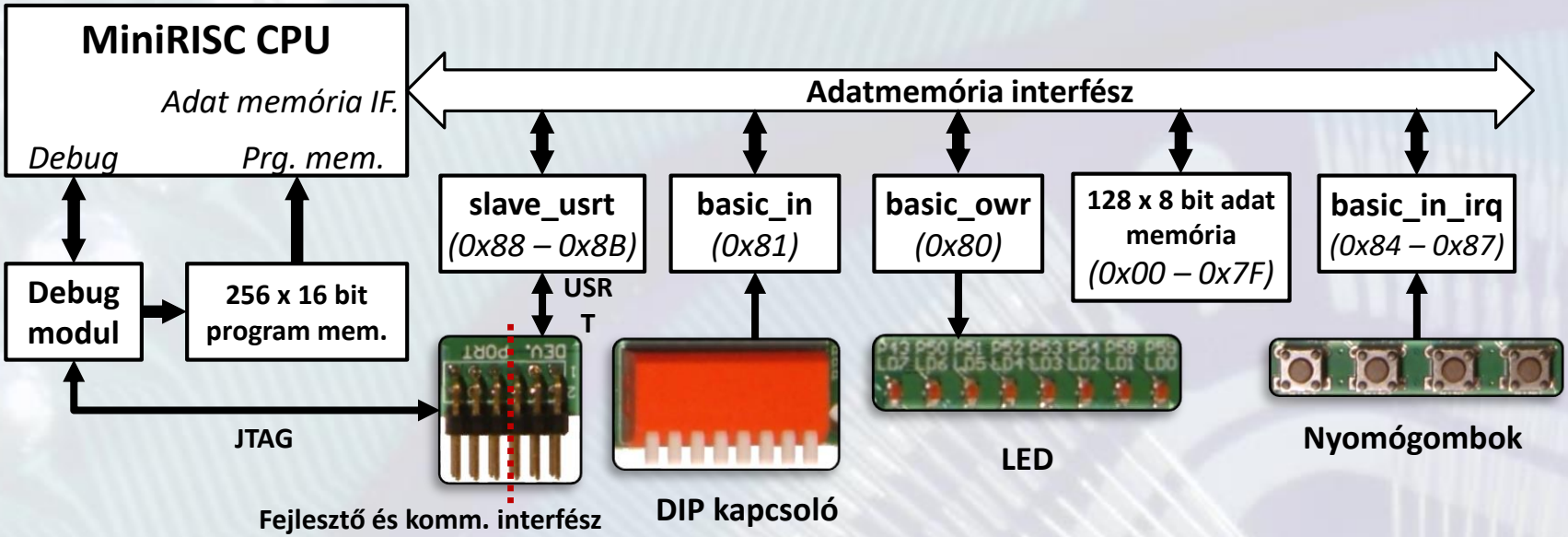
- **A perifériaillesztés általános tulajdonságai**
  - Minden egység az alábbi rajzon látható a részlegységeket tartalmazza: busz illesztő logika, egyedi periféria specifikus logika.



MiniRISC processzor – Perifériaillesztés

- **A perifériaillesztés általános tulajdonságai**
  - A BUSZ tulajdonságai rendszerfüggőek
    - Mi a **MiniRISC** egyszerű belső buszával tervezünk
    - Kommunikáció: 8 CÍM, 8 ADAT_KI, 8 ADAT_BE vonal,
    - Vezérlőjelek: CLK, RST, WR, RD, IRQ,
  - A *buszillesztő logika felépítése* tehát nem annyira a periféria, mint inkább *a busz tulajdonságai által meghatározott*.

Az egyszerűsített MiniRISC rendszer



Címtartomány	Méret	Periféria	Funkció
0x00 – 0x7F	128 byte	adatmemória	128 x 8 bit memória
0x80	1 byte	basic_owr	LED periféria
0x81	1 byte	basic_in	DIP kapcsoló periféria
0x84 – 0x87	4 byte	basic_in_irq	Nyomógomb periféria
0x88 – 0x8B	4 byte	slave_usrt	Soros USRT kommunikáció
0x90 – 0xFF	112 byte		További perifériák

MiniRISC processzor – Címtérkép

- **A MiniRISC processzor teljes címtartománya: 0x00 – 0xFF**
  - 0x00 – 0x7F RAM memória
  - 0x80 – 0xFF Perifériák
- **Szabályok:**
  - Minden eszköz címének bitjeit két részre osztjuk áll: **BÁZIS_CÍM bitek REGISZTER_CÍM bitek**
    A[N-1]...A[b] A[b-1]...A[0]
  - **A regiszter címbitekkel különböztetjük meg a periféria belső regisztereit** egymástól. Ezért **a szükséges regiszter címbitek száma k db regiszter esetén**: $\lceil \log_2(k) \rceil$ (Pl. k= 3 esetén 2.)
  - **A maradék felső címbitek egy periféria bármely regiszterének megcímzésekor ugyanolyan értékűek.**
  - **A maradék felső címbitek és a regiszter címbitek 0-val helyettesítve adják a periféria BÁZISCÍM-ét**. (A priféria által lefoglalt címtartomány kezdő címe.) **(**A regiszter címbiteket 1-el helyettesítve pedig a periféria által lefoglalt legnagyobb címet.)
  - **Egy periféria címtartománya nem fedheti át más perifériák címtartományát.**
  - A fentiekből következik, hogy **egy periféria mindig 2^n címet foglal le** (1, 2, 4). (Így adódhatnak a periféria által lefoglalt, de nem használt periféria címek, Pl. a BT periféria 0x87 címe.)

Címtartomány	A7	A6	A5	A4	A3	A2	A1	A0	Eszköz
0x00	0	0	0	0	0	0	0	0	RAM
.....		x	x	x	x	x	x	x	
0x7F		1	1	1	1	1	1	1	
0x80	1	0	0	0	0	0	0	0	LD
0x81	1	0	0	0	0	0	0	1	SW
0x82	1	0	0	0	0	0	1	0	TIMER
0x83								1	
0x84	1	0	0	0	0	1	0	0	BT_sel
0x85							0	1	
0x86							1	0	
0x87							1	1	
0x88	1	0	0	0	1	0	0	0	USRT
0x89							0	1	
0x8A							1	0	
0x8B							1	1	
0x8C	1	0	0	0	1	1	0	0	DMA
0x8D							0	1	
0x8E							1	0	
0x8F							1	1	
0x90	1	0	0	1	0	0	0	0	DISP
.....					0	x	x	x	
0x98					1	0	0	0	
0xA0	1	0	1	0	0	0	0	0	GPIO_A
0xA1							0	1	
0xA2							1	0	
0xA4	1	0	1	0	0	1	0	0	GPIO_C
0xA5							0	1	
0xA6							1	0	
0xA8	1	0	1	0	1	0	0	0	GPIO_B
0xA9							0	1	
0xAA							1	0	
0xAC	1	0	1	0	1	1	0	0	GPIO_D
0xAD							0	1	
0xAE							1	0	
0xB0	1	0	1	1	0	0	0	0	VGA
.....						0	x	x	
0xB5						1	0	1	
0xB8	1	0	1	1	0	0	0	0	PS/2
0xB9								1	
0xBA									Tartalék
.....									
0xFF									

MiniRISC processzor – Perifériaillesztés

- **Az illesztendő egység báziscíme (kezdőcíme) tipikusan az előző egység által lefoglalt terület utolsó címe +1 (vagy ennél nagyobb).**

Példa: Ha az egységnek és 3 byte-nyi terület kell, akkor a regiszter címbitek száma **2, tehát 4 byte-ot fog lefoglalni**. Ha az egység báziscíme (kezdőcíme) 0x84 (BT periféria) Akkor az egyes regiszterek cím bitjei a következőképpen alakulnak:

BÁZIS_CÍM bitek						REGISZTER_CÍM bitek			
A7	A6	A5	A4	A3	A2	A1	A0		
1	0	0	0	0	1	0	0	**BÁZIS_CÍM+0 (0. rekesz címe) 0x84**	ez egyben a báziscím is
1	0	0	0	0	1	0	1	**BÁZIS_CÍM+1 (1. rekesz címe) 0x85**	
1	0	0	0	0	1	1	0	**BÁZIS_CÍM+2 (2. rekesz címe) 0x86**	
1	0	0	0	0	1	1	1	**BÁZIS_CÍM+3 (3. rekesz címe) 0x87**	a BT-nél nem használ

MiniRISC processzor – Perifériaillesztés

BÁZIS_CÍM bitek REGISZTER_CÍM bitek

A7 A6 A5 A4 A3 A2 A1 A0

1 0 0 0 0 1 0 0 **BÁZIS_CÍM+0 (0. rekesz címe) 0x84** ez egyben a báziscím is

1 0 0 0 0 1 0 1 **BÁZIS_CÍM+1 (1. rekesz címe) 0x85**

1 0 0 0 0 1 1 0 **BÁZIS_CÍM+2 (2. rekesz címe) 0x86**

1 0 0 0 0 1 1 1 **BÁZIS_CÍM+3 (3. rekesz címe) 0x87** (a BT-nél nem használt)

A cím BÁZIS_CÍM bitek része (A[7:2] **a periféria bármely regiszteréhez forduláskor ugyanaz**: 100001

Ezért ha a periféria cím báziscím bitje megegyeznek a processzor által kiadott címmel (A[7:2] == BASE_ADDR[7:2]), akkor a periféria valamely címéhez fordul a processzor.

Ezt egy **konstans komparátor** tudja jelezni, ez lesz a periféria select (**psel**) jel (*cím dekódolás*):

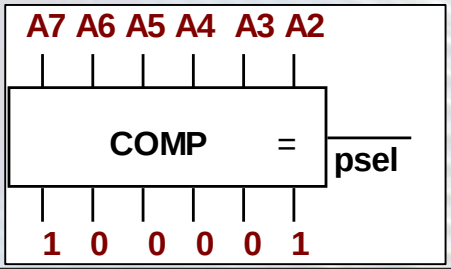
Verilog-ban leírva:

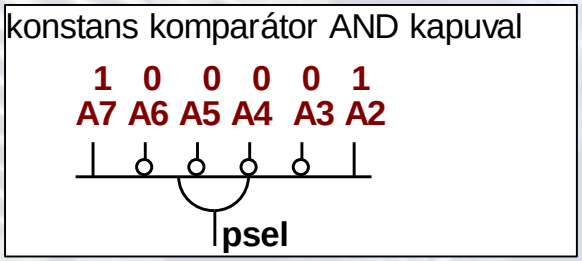
```
assign BASE_ADDR = 8'h84;
assign psel = (BASE_ADDR[7:2]== (A[7:2]);
```

vagy

```
assign psel = (BASE_ADDR >> 2) == (A >> 2);
```

Az (A >>2)-ben a 2 a regiszter címbitek száma.





MiniRISC processzor – Perifériaillesztés

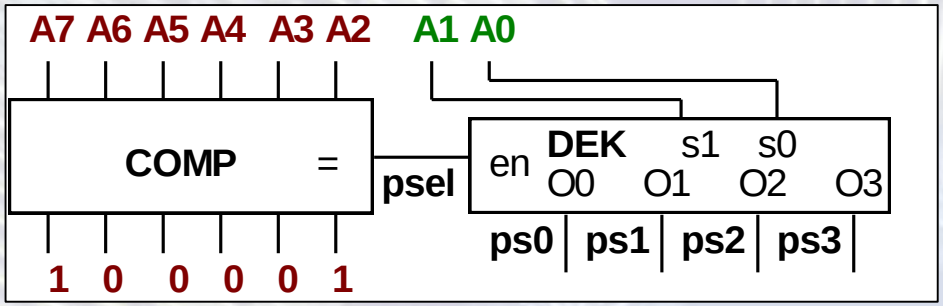
BÁZIS_CÍM bitek REGISZTER_CÍM bitek

A7	A6	A5	A4	A3	A2	A1	A0		
1	0	0	0	0	1	0	0	**BÁZIS_CÍM+0 (0. rekesz címe) 0x84**	egyben a báziscím is
1	0	0	0	0	1	0	1	**BÁZIS_CÍM+1 (1. rekesz címe) 0x85**	
1	0	0	0	0	1	1	0	**BÁZIS_CÍM+2 (2. rekesz címe) 0x86**	
1	0	0	0	0	1	1	1	**BÁZIS_CÍM+3 (3. rekesz címe) 0x87**	(a BT-nél nem használt)

A periférián belüli rekeszek címét a REGISZTER_CÍM bitek határozzák meg (itt A[1:0]). Ha **psel** igaz és **A[1:0] == 2'b00**, akkor a **0. rekesz van megcímezve.** Azt, **hogy a periféria mely regiszterét címzi a processzor egy a psel-le engedélyezett dekóder egyes kimenetei jelzik**. (Pl.0x85 cím esetén a dekóder ps1 kimenet 1, a többi kimenete 0.)

Verilog-ban leírva:

```
assign ps0 = psel & (A[1:0] == 2'b00);
assign ps1 = psel & (A[1:0] == 2'b01);
assign ps2 = psel & (A[1:0] == 2'b10);
assign ps3 = psel & (A[1:0] == 2'b11);
```

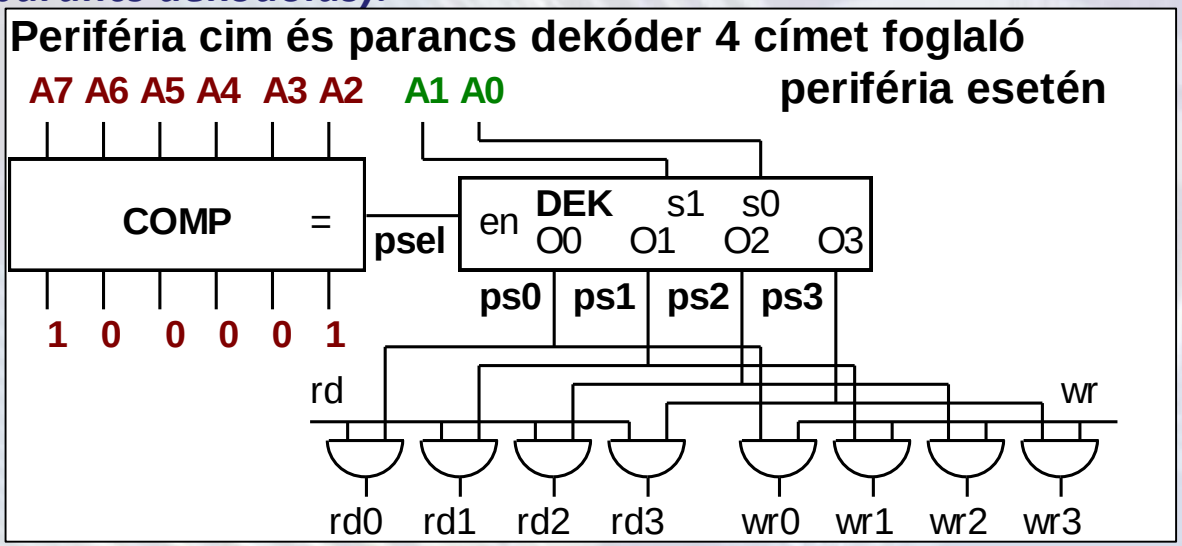


MiniRISC processzor – Perifériaillesztés

BÁZIS_CÍM bitek					REGISZTER_CÍM bitek				
A7	A6	A5	A4	A3	A2	A1	A0		
1	0	0	0	0	1	0	0	BÁZIS_CÍM+0 (0. rekesz címe) 0x84	ez egyben a báziscím is
1	0	0	0	0	1	0	1	BÁZIS_CÍM+1 (1. rekesz címe) 0x85	
1	0	0	0	0	1	1	0	BÁZIS_CÍM+2 (2. rekesz címe) 0x86	
1	0	0	0	0	1	1	1	BÁZIS_CÍM+3 (3. rekesz címe) 0x87	(a BT-nél nem használt)

A megcimzett regiszternek szóló **írás (wr), olvasás (rd) parancsát** ezekből egy **AND kapuval** lehet előállítani ***(parancs dekódolás):***

```
assign wr0 = ps0 & wr;
assign rd0 = ps0 & rd;
assign wr1 = ps1 & wr;
assign rd1 = ps1 & wr;
assign wr2 = ps2 & wr;
assign rd2 = ps2 & rd;
assign wr3 = ps3 & wr;
assign rd3 = ps3 & wr;
```

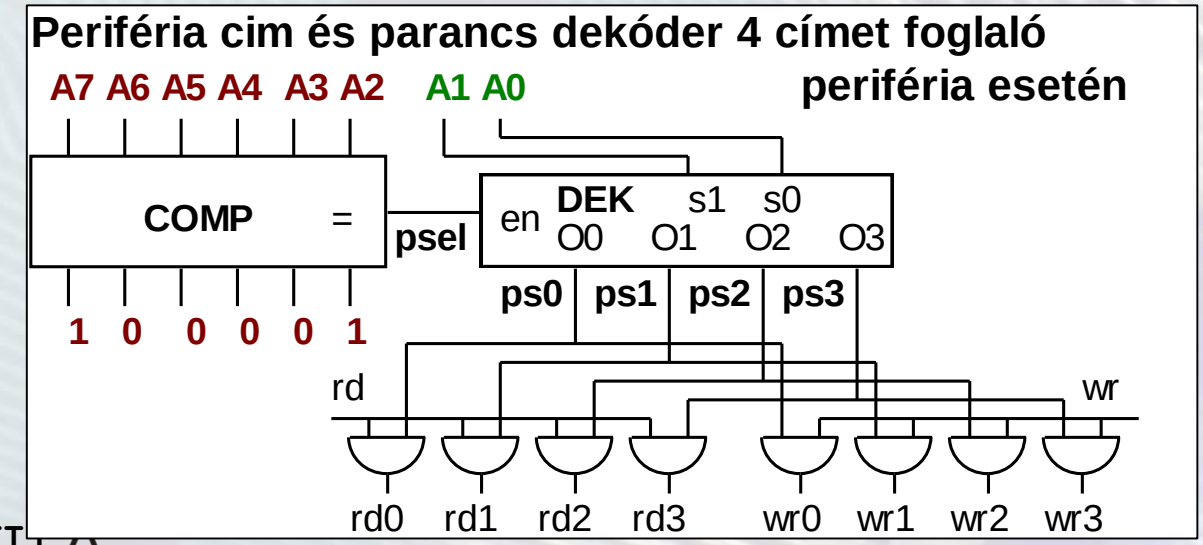


Pl. Ha a processzor a 0x85 címre ír, akkor a parancs dekóder wr1 kimenete lesz 1 értékű, az összes többi kimenete 0. Ha a 0x86 címről olvas, akkor csak az rd2 kimene lesz 1.

MiniRISC processzor – Perifériaillesztés

A Verilog leírásból ps0, ps1 stb. Elhagyható (a kifejezést közvetlenül beírjuk wri/rdi-be):

```
assign wr0 = psel & (A[1:0] == 2'b00) & wr;
assign rd0 = psel & (A[1:0] == 2'b00) & rd;
assign wr1 = psel & (A[1:0] == 2'b01) & wr;
assign rd1 = psel & (A[1:0] == 2'b01) & rd;
assign wr2 = psel & (A[1:0] == 2'b10) & wr;
assign rd2 = psel & (A[1:0] == 2'b10) & rd;
assign wr3 = psel & (A[1:0] == 2'b11) & wr;
assign rd3 = psel & (A[1:0] == 2'b11) & rd;
```

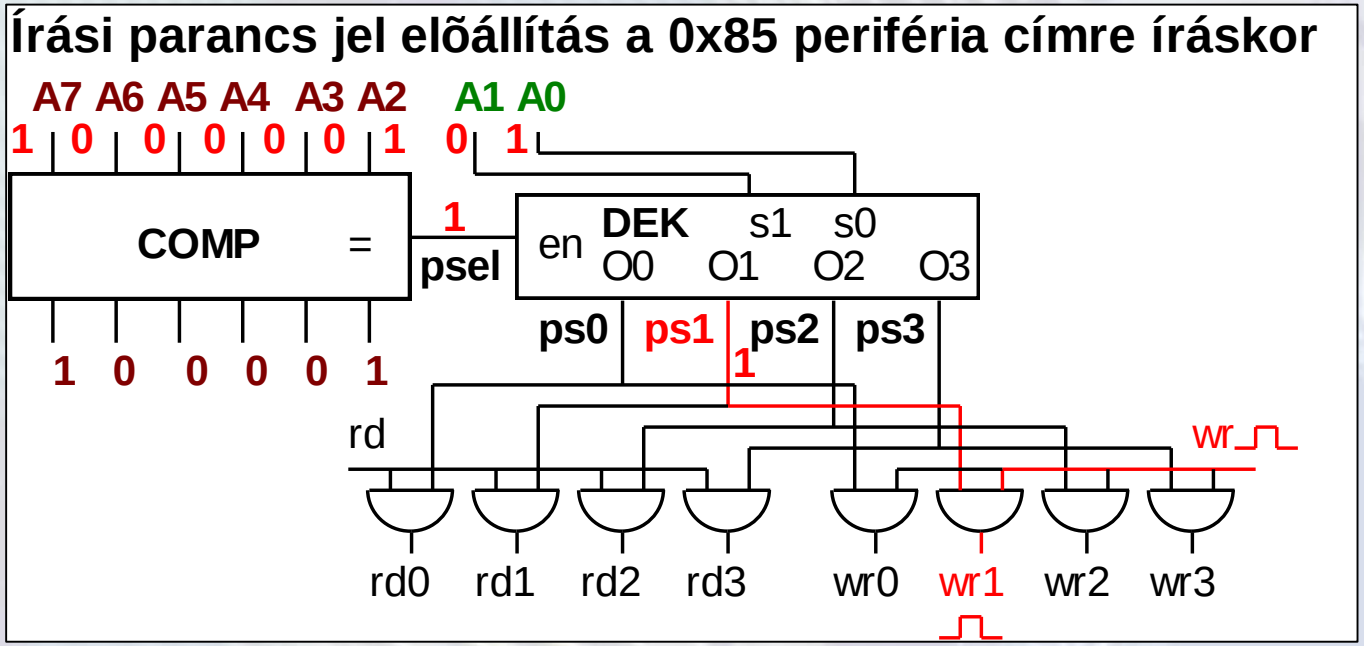


MiniRISC processzor – Perifériaillesztés

- BÁZIS_CÍM bitek REGISZTER_CÍM bitek

A7	A6	A5	A4	A3	A2	A1	A0	
1	0	0	0	0	1	0	0	BÁZIS_CÍM+0 (0. rekesz címe) 0x84
1	0	0	0	0	1	0	1	BÁZIS_CÍM+1 (1. rekesz címe) 0x85, WR
1	0	0	0	0	1	1	0	BÁZIS_CÍM+2 (2. rekesz címe) 0x86
1	0	0	0	0	1	1	1	BÁZIS_CÍM+3 (3. rekesz címe) nem használt

Példa írásra: mov 0x85, r0



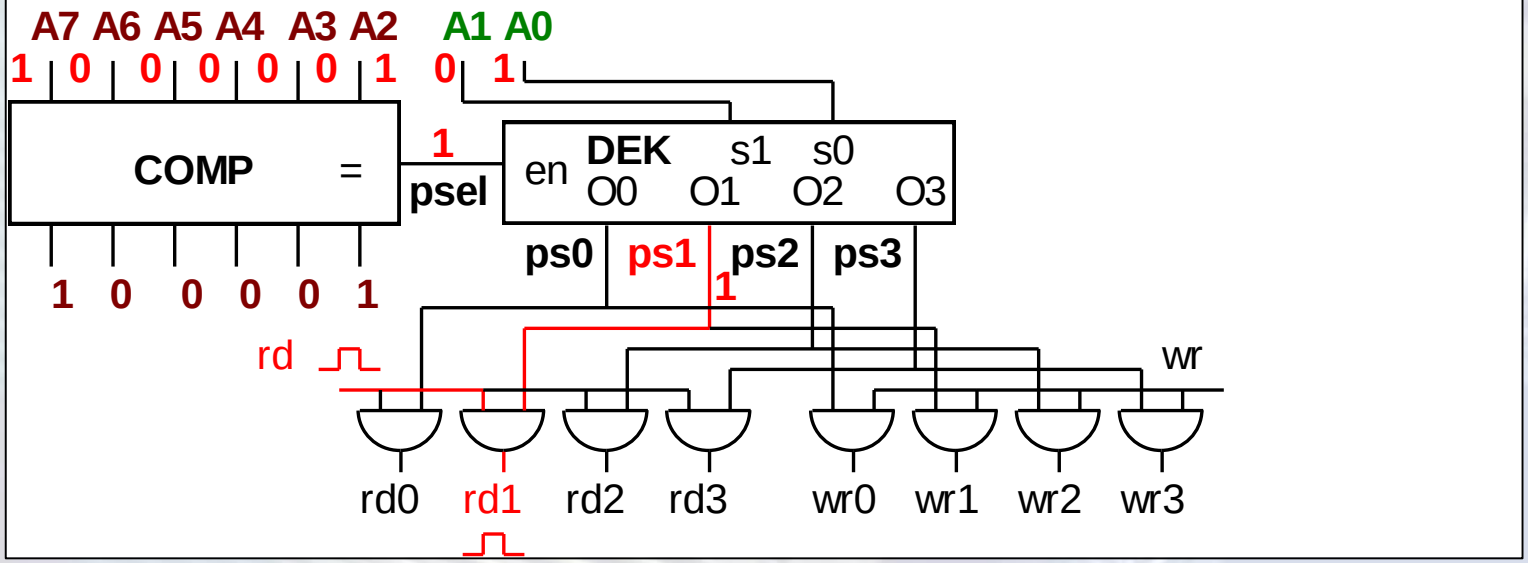
MiniRISC processzor – Perifériaillesztés

- BÁZIS_CÍM bitek REGISZTER_CÍM bitek

A7	A6	A5	A4	A3	A2	A1	A0	
1	0	0	0	0	1	0	0	BÁZIS_CÍM+0 (0. rekesz címe) 0x84
1	0	0	0	0	1	0	1	BÁZIS_CÍM+1 (1. rekesz címe) 0x85, RD
1	0	0	0	0	1	1	0	BÁZIS_CÍM+2 (2. rekesz címe) 0x86
1	0	0	0	0	1	1	1	BÁZIS_CÍM+3 (3. rekesz címe) nem használt

Példa olvasásra: mov r0, 0x85

Olvasási parancsjel elõállítás a 0x85 periféria címrõl olvasáskor



MiniRISC processzor – Perifériaillesztés

- **Parancsdekódolás:**
  - Írás engedélyező jelek

    wri = psel & wr & (A[b-1:0] == REGADDR).

  – Elsősorban az egyes periféria regiszterek betöltés engedélyező (ld) jeleiként használjuk a **wr0, wr1, stb**… jeleket (***b***: a legkisebb báziscím bit sorszáma. ***REGADDR*** a regiszter sorszáma.)

MiniRISC processzor – Perifériaillesztés

- **Parancsdekódolás:**
  - Olvasás engedélyező jelek

    rdi = psel & rd & (A[b-1:0] == REGADDR)

    – Az egyes perifériaregiszterek olvasás engedélyező jeleit **rd0, rd1...** használjuk a periféria megfelelő adatának a bemeneti adatbuszra kapcsolására. Ha nem olvassák a perifériát, akkor 8'h00-át kapcsol a bemeneti adatbuszra. Busz multiplexerrel valósítjuk meg.

    – Használni kell még, ha az olvasás állapotváltozást okoz (pl. FIFO olvasása).
- ***A további mintapéldáknál és a vizsga ZH feladatoknál egyszerűsítés miatt feltételezzük, hogy egyedül a megtervezendő periféria van a rendszerben. Ezért a bemeneti adatbuszon (din) nincs szükség a 8 db OR kapukra, így olvasáskor a periféra közvetlenül a din-re kapcsolhatja az adatát.***

MiniRISC processzor – Perifériaillesztés

(1. példa – Specifikáció)

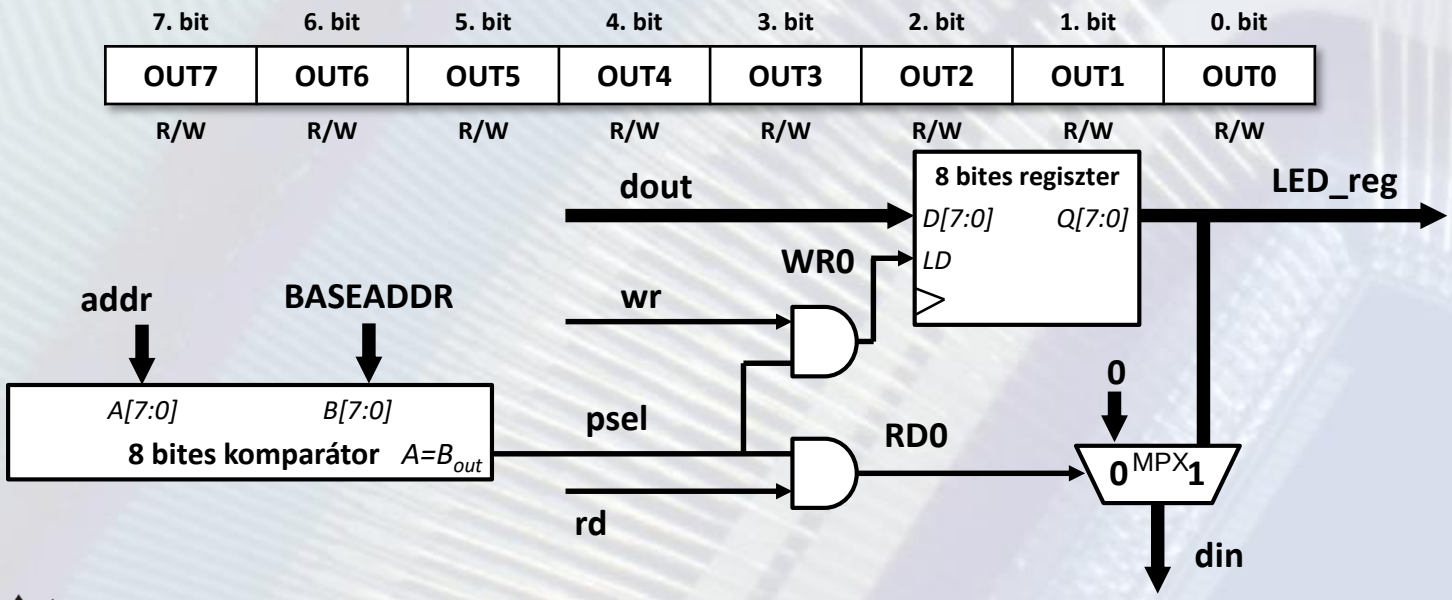
<u>Feladat:</u> 8 darab LED illesztése a processzoros rendszerhez. Kimeneti periféria, de az aktuális állapot legyen visszaolvasható.

Egyszerű, egy 8 bites írható és olvasható regiszter szükséges.

A periféria egyetlen címet foglal le (nincs regiszter címbitje)

Adatregiszter: BASEADDR + 0x00, 8 bites, írható/olvasható

- Az OUT_i bit hajtja meg az i-edik LED-et



MiniRISC processzor – Perifériaillesztés

(1. példa – Megvalósítás Verilog nyelven)

Feladat: 8 darab LED illesztése a processzoros rendszerhez. A periféria csak 1 címet foglal el, így nincs szükség külön regiszter címbitekre.

```
module basic_owr #(
   //A periféria báziscíme.
   parameter BASEADDR = 8'hff
) (
   //Órajel és reset.
   input  wire       clk,
   input  wire       rst,

   //Adatmemória interfész.
   input  wire [7:0] addr,
   input  wire       wr,
   input  wire       rd,
   input  wire [7:0] dout,
   output reg  [7:0] din,

   //Kimenő adat.
   output reg  [7:0] LED_reg
);

//A periféria kiválasztó jele.
wire psel = (addr == BASEADDR);
```

```
//Az adatreg. írás engedélyező jele.
wire WR0 = psel & wr;

//Az adatreg. olvasás engedélyező jele.
wire RD0 = psel & rd;

//Adatregiszter.
always @(posedge clk)
   if (rst)
      LED_reg <= 8'd0;
   else
      if (WR0)
         LED_reg <= dout;

//Az olvasási adatbusz meghajtása.
always @(*)
   if (RD0)
      din <= LED_reg;
   else
      din <= 8'd0;

endmodule
```

MiniRISC processzor – Perifériaillesztés

(2. példa – Specifikáció)

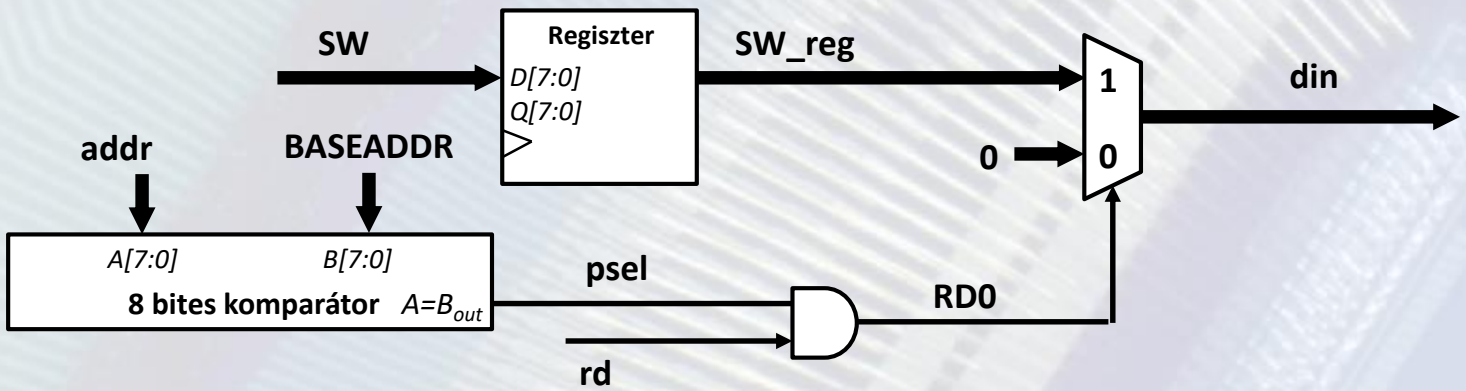
Feladat: 8 darab kapcsoló illesztése a processzoros rendszerhez

Egyszerű, egy 8 bites csak olvasható regiszter szükséges, amely folyamatosan mintavételezi a 8 kapcsoló állapotát

Adatregiszter: BASEADDR + 0x00, 8 bites, csak olvasható

- Az IN_i bit az i-edik kapcsolón beállított értéket veszi fel

7. bit	6. bit	5. bit	4. bit	3. bit	2. bit	1. bit	0. bit
IN7	IN6	IN5	IN4	IN3	IN2	IN1	IN0
R	R	R	R	R	R	R	R



MiniRISC processzor – Perifériaillesztés

(2. példa – Megvalósítás Verilog nyelven)

Feladat: 8 darab kapcsoló (SW) illesztése a processzoros rendszerhez

```
module basic_in #(
   //A periféria báziscíme.
   parameter BASEADDR = 8'hff
) (
   //Órajel és reset.
   input  wire       clk,
   input  wire       rst,

   //Adatmemória interfész.
   input  wire [7:0] addr,
   input  wire       rd,
   output reg  [7:0] din,

   //Bejövő adat.
   input  wire [7:0] SW;
);

//A periféria kiválasztó jele.
wire psel = (addr == BASEADDR);
```

```
//Az adatreg. olvasás engedélyező jele.
wire RD0 = psel & rd;

//Adatregiszter.
reg [7:0] SW_reg;

always @(posedge clk)
   if (rst)
      SW_reg <= 8'd0;
   else
      SW_reg <= SW;

//Az olvasási adatbusz meghajtása.
always @(*)
   if (RD0)
      din <= SW_reg;
   else
      din <= 8'd0;

endmodule
```

MiniRISC processzor – Memóriaillesztés

(Memória címtartomány dekódolása)

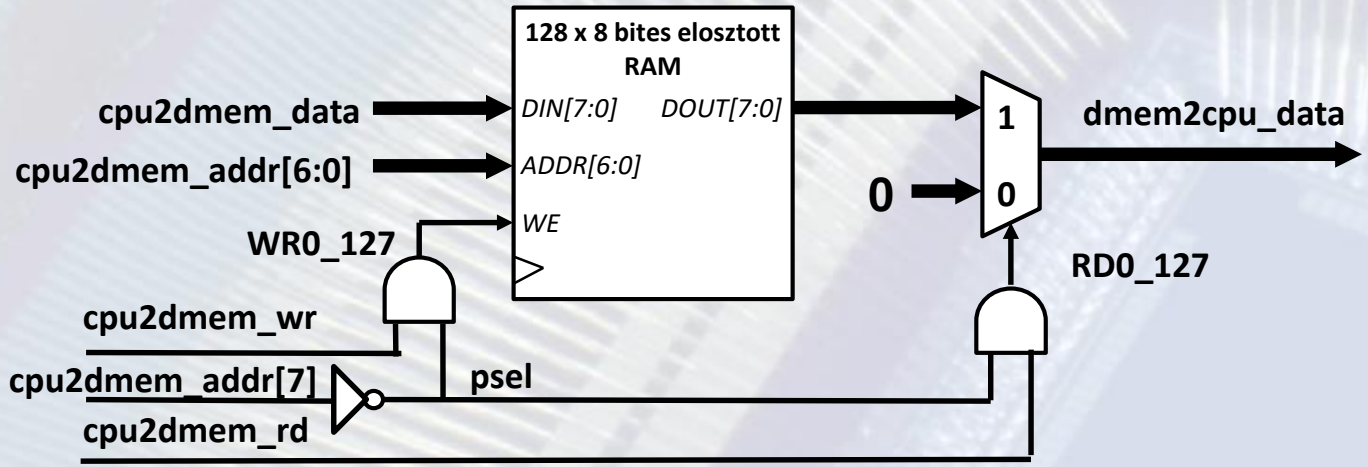
Feladat: 128 x 8 bites memória illesztése a processzoros rendszerhez

A memóriának 7 címbitje van (2^7 = 128), a címbusz 8 bites

- Kijelölt címtartomány: **00000000 (0x00) – 01111111 (0x7F)**

A címdekóder logika megvalósítása

- A címbusz alsó 7 bitje közvetlenül kapcsolódik a memóriához
- A legfelső címbitet (A[7]) használjuk a báziscím dekódoláshoz
  - Ha ez 0, akkor a memória van kiválasztva, ha nem, akkor nincs memória művelet (psel = (A[7]==0) vagy psel = ~A[7])



MiniRISC processzor – Memóriaillesztés

(Megvalósítás Verilog nyelven)

Feladat: 128 x 8 bites memória illesztése a processzoros rendszerhez

```
wire        memsel    = ~cpu2dmem_addr[7];      //A memória kiválasztó jele.
wire [6:0]  mem_addr  = cpu2dmem_addr[6:0];     //A memória címbusza.
wire        mem_wr    = memsel & cpu2dmem_wr;   //A memória írás engedélyező jele.
wire        mem_rd    = memsel & cpu2dmem_rd;   //A memória olvasás engedélyező jele.

//128 x 8 bites elosztott RAM deklarálása.
(* ram_style = "distributed" *)
reg [7:0] mem [127:0];

//Az elosztott RAM írási portja.
//Az elosztott RAM írása szinkron művelet.
always @(posedge clk)
   if (mem_wr)
      mem[mem_addr] <= cpu2dmem_data;

//Az elosztott RAM olvasási portja és az olvasási adatbusz meghajtása
//(ezt kell VAGY kapuval rákötni a processzor olvasási adatbuszára).
//Az elosztott RAM olvasása aszinkron művelet.
wire [7:0] mem2cpu_data = (mem_rd) ? mem[mem_addr] : 8'd0;
```

Digitális technika
10. EA vége