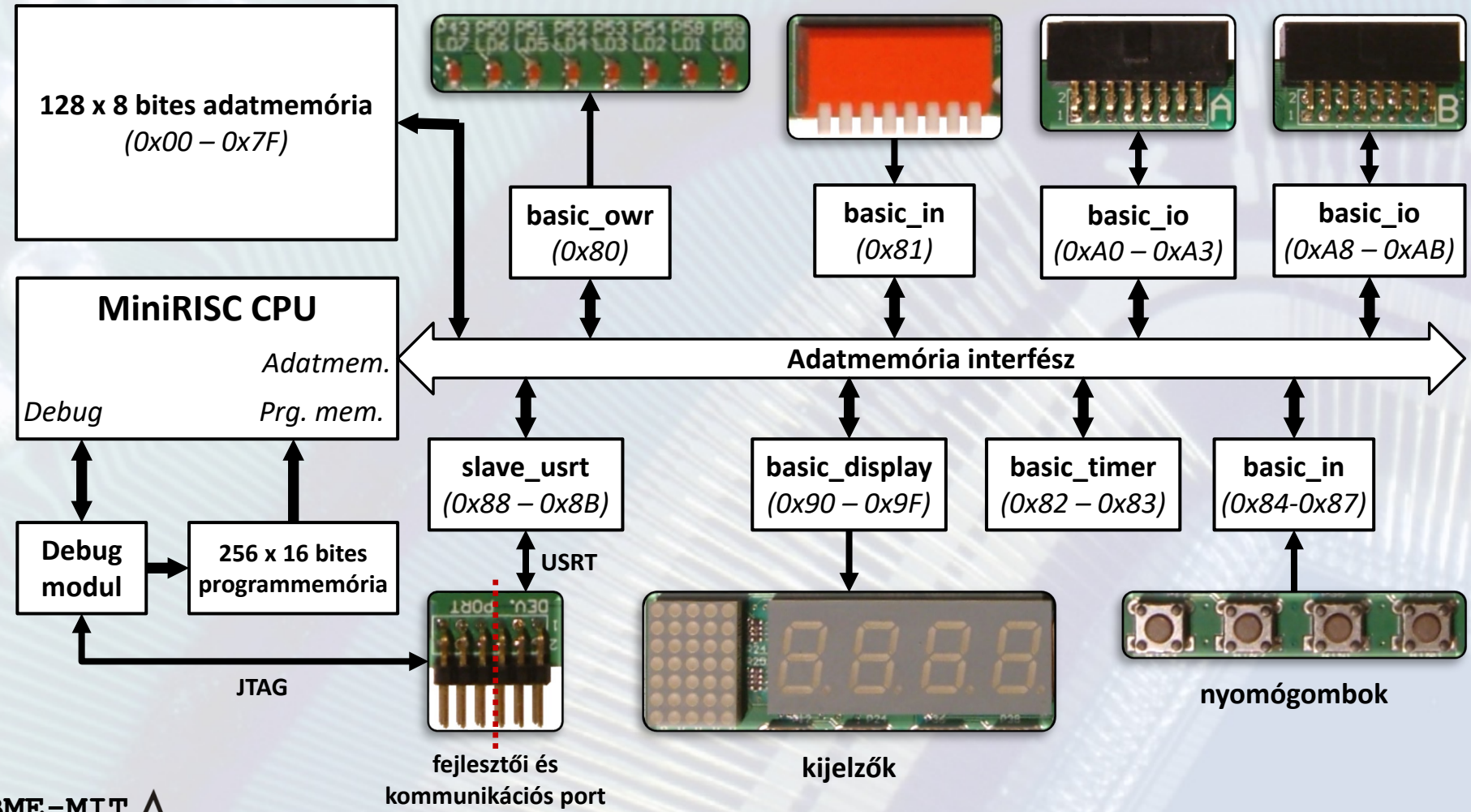
BUDAPESTI MŰSZAKI ÉS GAZDASÁGTUDOMÁNYI EGYETEM
VILLAMOSMÉRNÖKI ÉS INFORMATIKAI KAR
MÉRÉSTECHNIKA ÉS INFORMÁCIÓS RENDSZEREK TANSZÉK

Digitális technika
VIMIAA02
11. hét

Fehér Béla
BME MIT

MiniRISC mintarendszer Blokkvázlat



A mikroprocesszoros busz

- **A busz részei:**
  - **Címbusz ADDR[n:0]**
  - **Adatbusz DATA[m:0]**

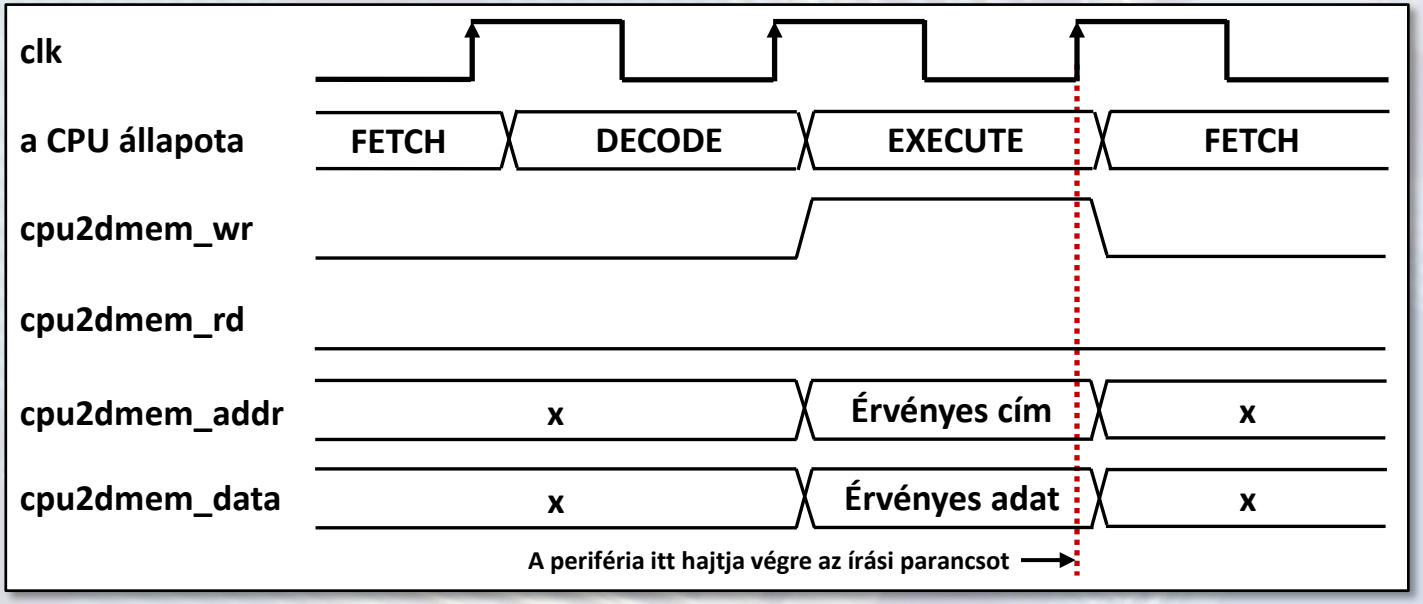
    belső busznál külön D_IN[m:0], D_OUT[m:0]
  - **Vezérlő busz (sok egyedi jel összessége):**
    - **Rendszerjelek: CLK, RST**
    - **Arbitrációs jelek: BUSREQ, BUSACK** (nem tanuljuk)
    - **Irányvezérlő jelek: READ, WRITE**
    - **Átvitelvezérlő jelek: FRAME, TS, TACK, AS, DS** (nem tanuljuk)
    - **Megszakítás vezérlő jelek: IRQi, IACK**

MiniRISC processzor – Interfészek

(Adatmemória interfész – Írási buszciklus)

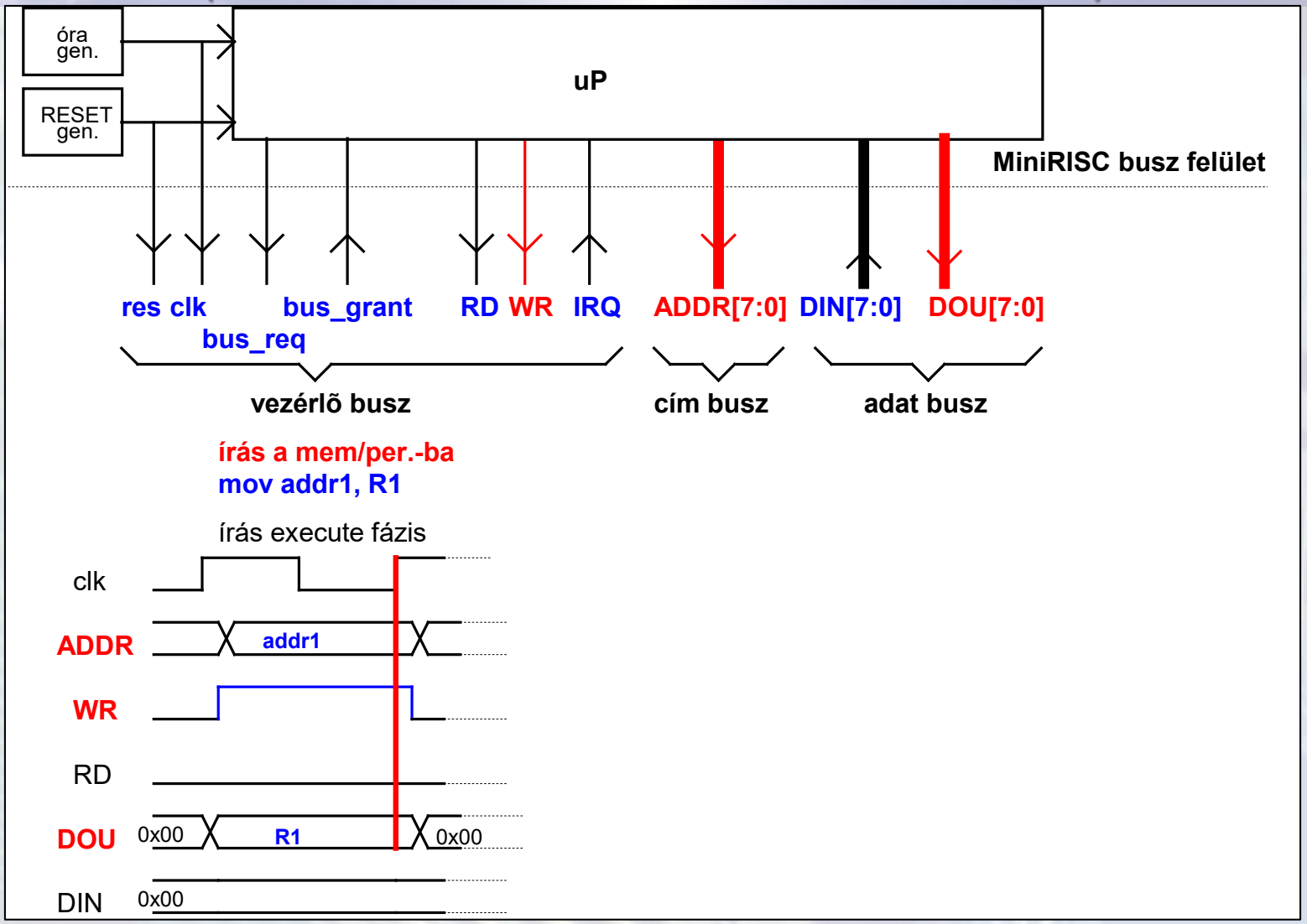
Az adatmemória interfész írási (write) ciklusa

- Az írási ciklust a végrehajtási (execute) fázisban 1 órajelciklus ideig aktív ***cpu2dmem_wr*** (wr) jel jelzi
- Az írási ciklus alatt a ***cpu2dmem_addr*** (A[7:0]) címbuszon a cím stabil
- Az írási ciklus alatt a ***cpu2dmem_data*** (Dou[7:0]) írási adatbuszon az adat stabil, melyet a kiválasztott periféria az órajel következő felfutó élére mintavételez



MiniRISC processzor – Interfészek

(Adatmemória interfész – Írási buszciklus)

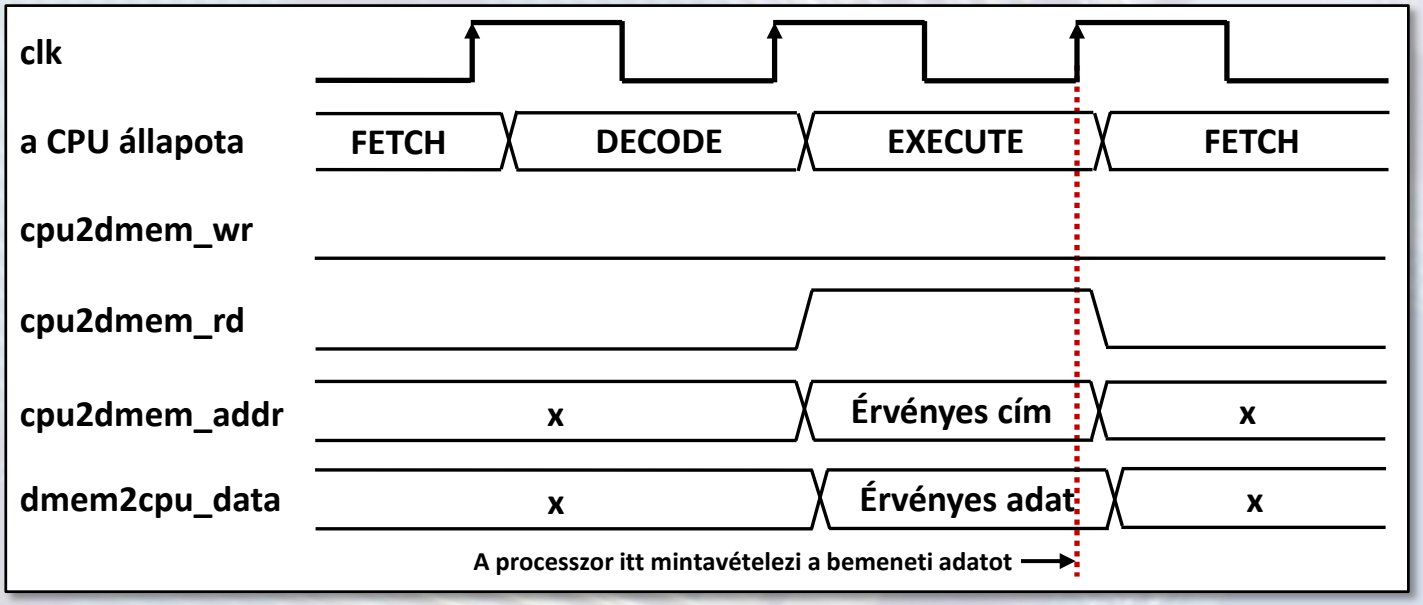


MiniRISC processzor – Interfészek

(Adatmemória interfész – Olvasási buszciklus)

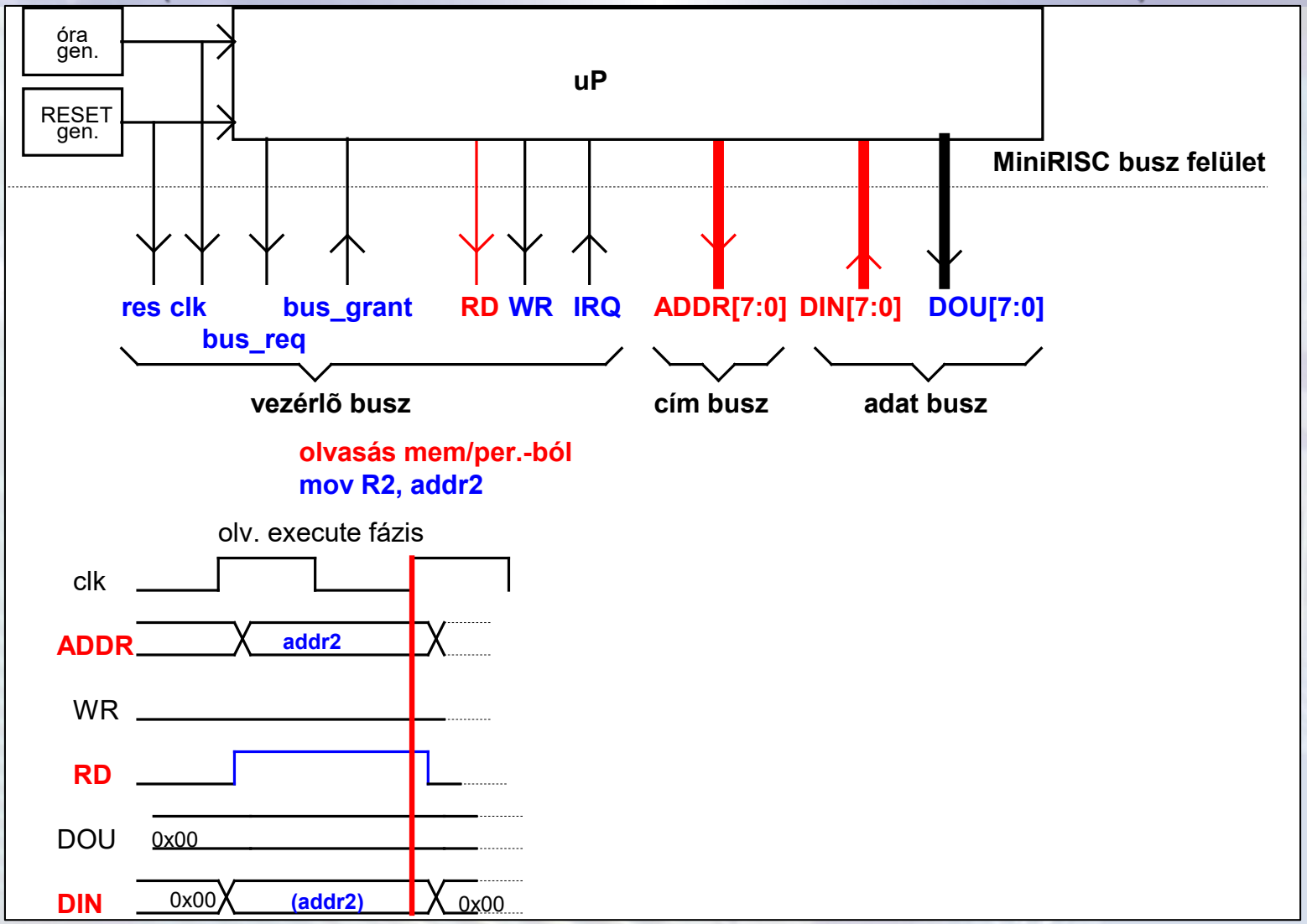
Az adatmemória interfész olvasási (read) ciklusa

- Az olvasási ciklust a végrehajtási (execute) fázisban 1 órajelciklus ideig aktív ***cpu2dmem_rd*** (rd) jel jelzi
- Az olvasási ciklus alatt a ***cpu2dmem_addr*** címbuszon a cím stabil
- Az olvasási ciklus alatt a kiválasztott periféria a ***dmem2cpu_data*** (Din) olvasási adatbuszra kapuzza az adatot, a többi periféria ezalatt inaktív nullával hajtja meg az adatkimenetét

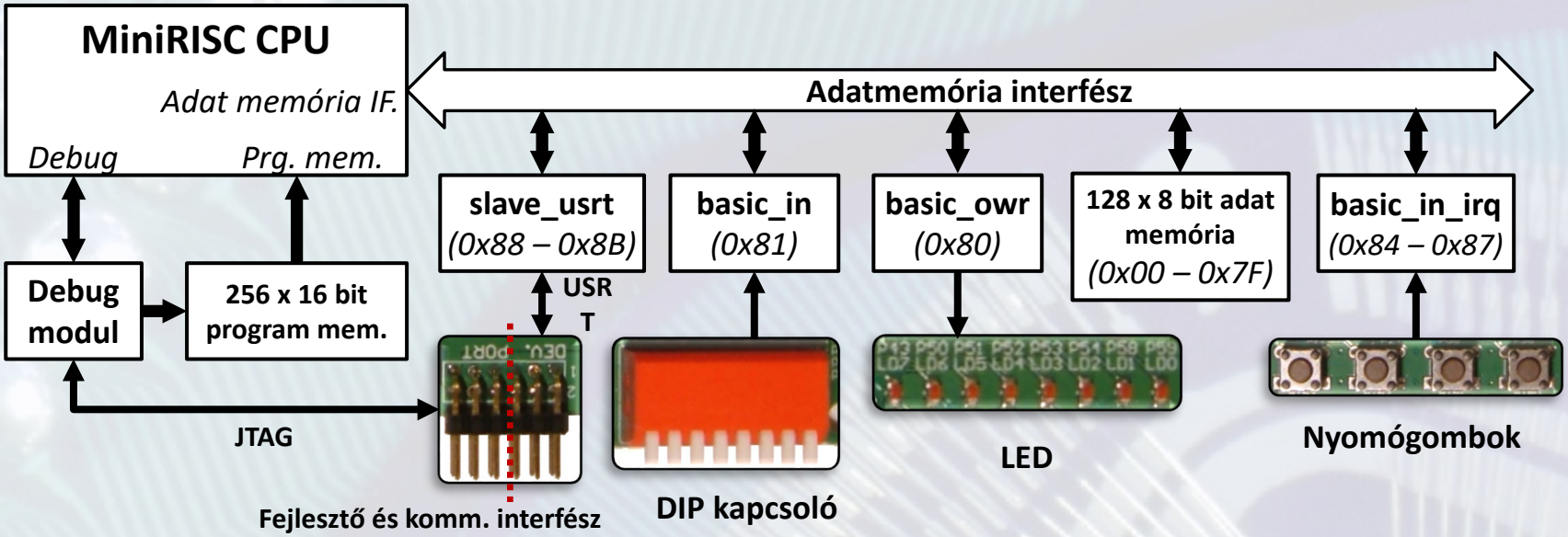


MiniRISC processzor – Interfészek

(Adatmemória interfész – Olvasási buszciklus)



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 címzése két részből áll: **BÁZIS_CÍM** + **REGISZTER_CÍM**
- **A BÁZIS_CÍM mindig egy 2^N méretű címtartomány elejére mutat.**

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 (de ennél nagyobb címen is lehet).
- **Tetszőleges 2 egység (memória vagy periféria) által lefoglalt címtartomány nem lehet átfedő!**
- Mivel a címlogika egyszerűsítésére törekszünk, **a címbiteket minden egységhez 2 részre bontjuk**.
  A **jobb oldali rész (REGISZTER_CÍM bitek)** bitszáma **az egység által lefoglalt terület (byte-ok) számától (m)** függ: $\lceil log_2(m) \rceil$
- **A BÁZIS_CÍM-ben** (kezdőcímben) **a REGISZTER_CÍM bitek mindig 0 értékűek**.
  Így a BÁZIS_CÍM + REGISZTER_CÍM összeadásakor csak a REGISZTER_CÍM bitek változnak, a többi bit nem. A többi bitet neveztük BÁZIS_CÍM biteknek. Így azt, hogy a periféria címterületéhez fordul a processzor, a BÁZIS_CÍM bitek konstanssal való összehasonlítása (konstans komparátor) tudja jelezni.

Példa: Ha az egységnek és 3 byte-nyi terület kell, akkor **2 bit, tehát 4 byte-ot fog lefoglalni**. Ha az egység kezdőcíme 0x84 (BT periféria) Akkor a cím bitek 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

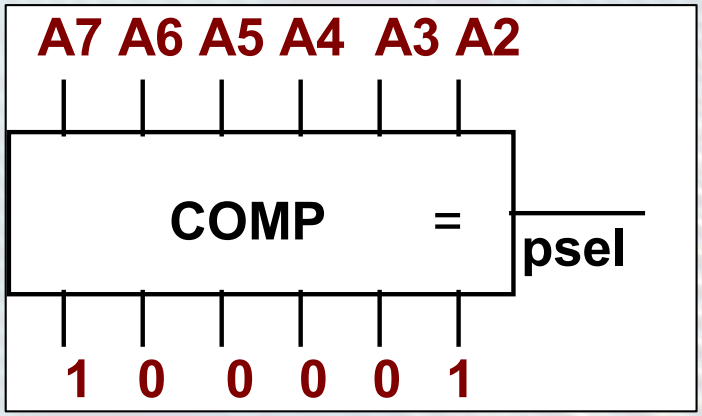
A7	A6	A5	A4	A3	A2	A1	A0		
1	0	0	0	0	1	0	0	BÁZIS_CÍM+0 (0. rekesz címe) 0x84	ez egyben a báziscím is
1	0	0	0	0	1	0	1	BÁZIS_CÍM+1 (1. rekesz címe) 0x85	
1	0	0	0	0	1	1	0	BÁZIS_CÍM+2 (2. rekesz címe) 0x86	
1	0	0	0	0	1	1	1	BÁZIS_CÍM+3 (3. rekesz címe) 0x87	(a BT-nél nem használt)

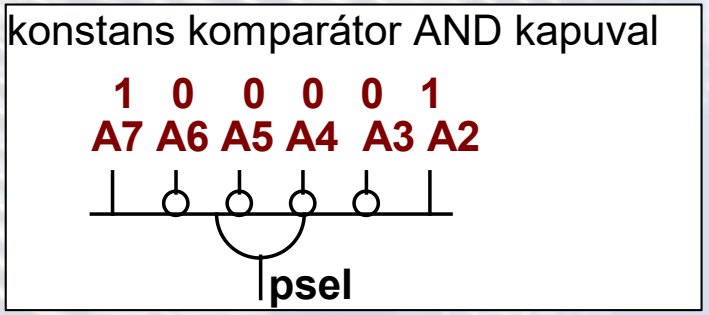
A BÁZIS_CÍM bitek (A[7:2] **a periféria bármely regiszteréhez forduláskor ugyanazok**: 100001

Ha **A[7:2] == BASE_ADDR[7:2]** akkor a periféria valamely címéhez fordul a processzor. Ezt egy **konstans komparátor** tudja jelezni a periféria select (psel) jellel:

Verilog-ban leírva: **assign psel = (BASE_ADDR[7:2]== (A[7:2);**

vagy **assign psel = (BASE_ADDR >> 2) == (A >> 2);**





MiniRISC processzor – Perifériaillesztés

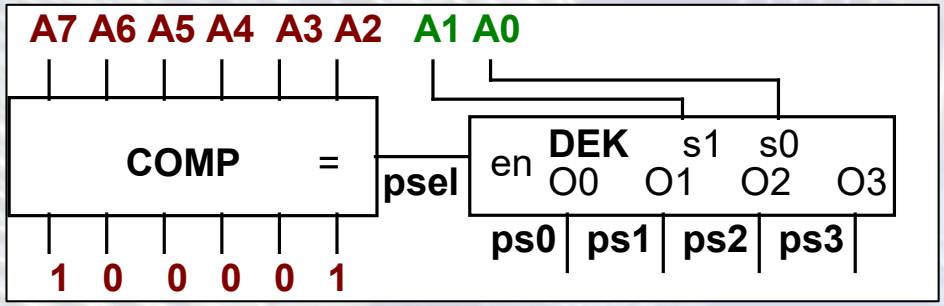
BÁZIS_CÍM bitek **REGISZTER_CÍM bitek**

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.
Ha **psel** igaz és **A[1:0] == 2'b00**, akkor a **0. rekesz van megcímezve.** Ezek egy **engedélyezhető dekóder** egyes kimenetei, psel az engedélyezés.

Verilog-ban minden rekeszhez leírva:

```
assign psel0 = psel & (A[1:0] == 2'b00);
assign psel1 = psel & (A[1:0] == 2'b01);
assign psel2 = psel & (A[1:0] == 2'b10);
assign psel3 = psel & (A[1:0] == 2'b11);
```



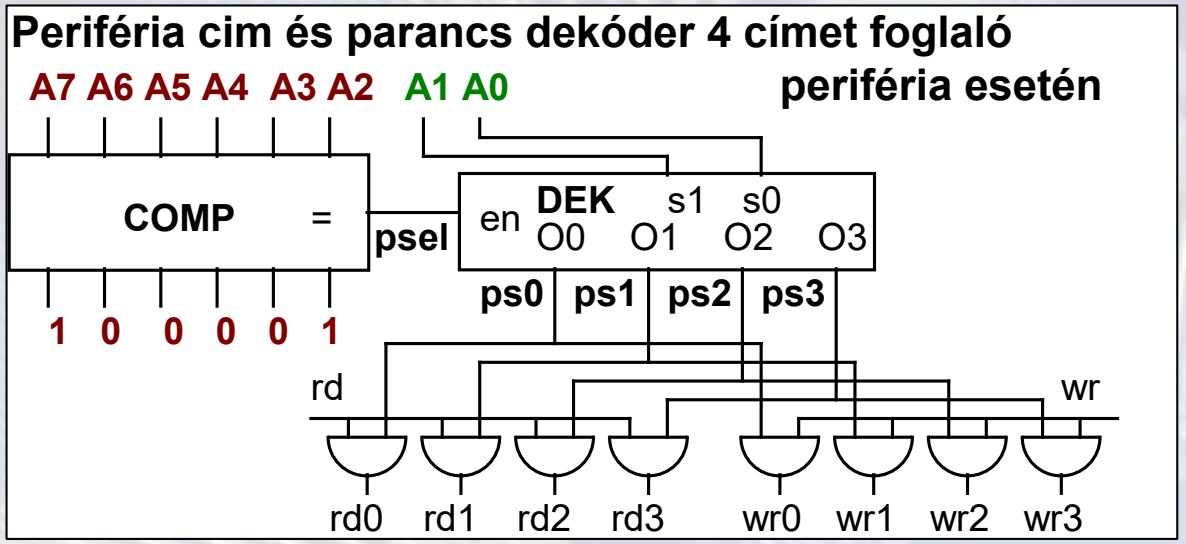
MiniRISC processzor – Perifériaillesztés

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:

```
assign wr0 = psel0 & wr;
assign rd0 = psel0 & rd;
assign wr1 = psel1 & wr;
assign rd1 = psel1 & wr;
assign wr2 = psel2 & wr;
assign rd0 = psel2 & rd;
assign wr1 = psel3 & wr;
assign rd1 = psel3 & wr;
```

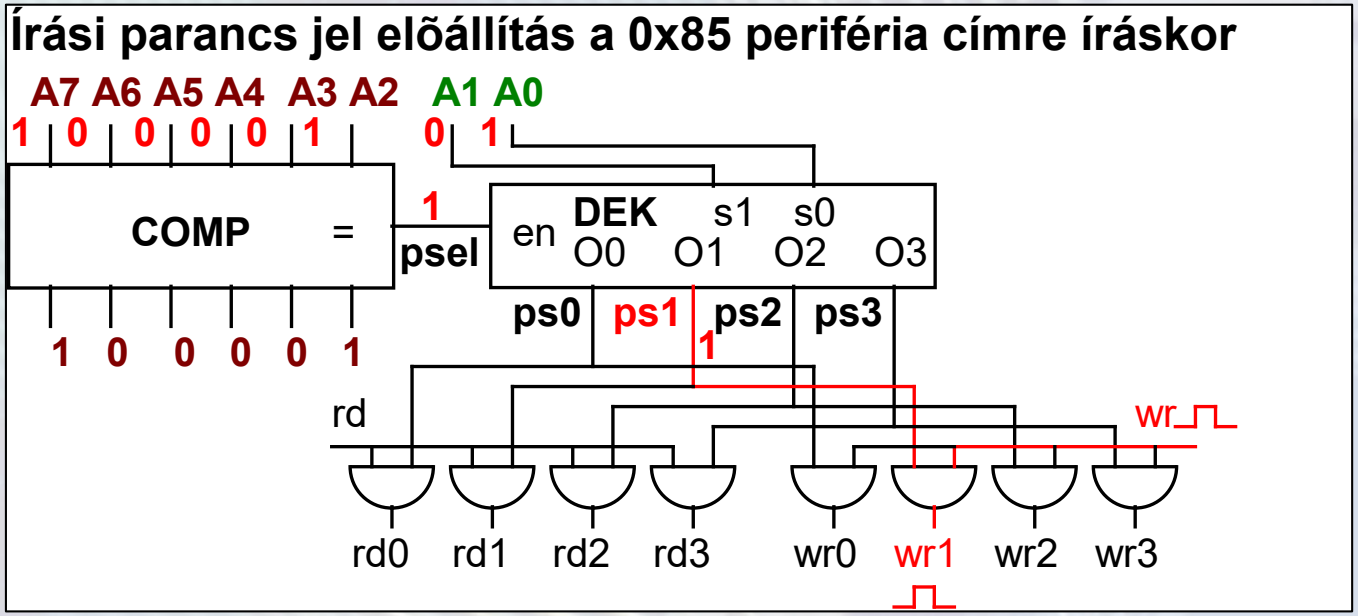


MiniRISC processzor – Perifériaillesztés

- BÁZIS_CÍM bitek REGISZTER_CÍM bitek

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

Példa írásra: mov 0x85, r0

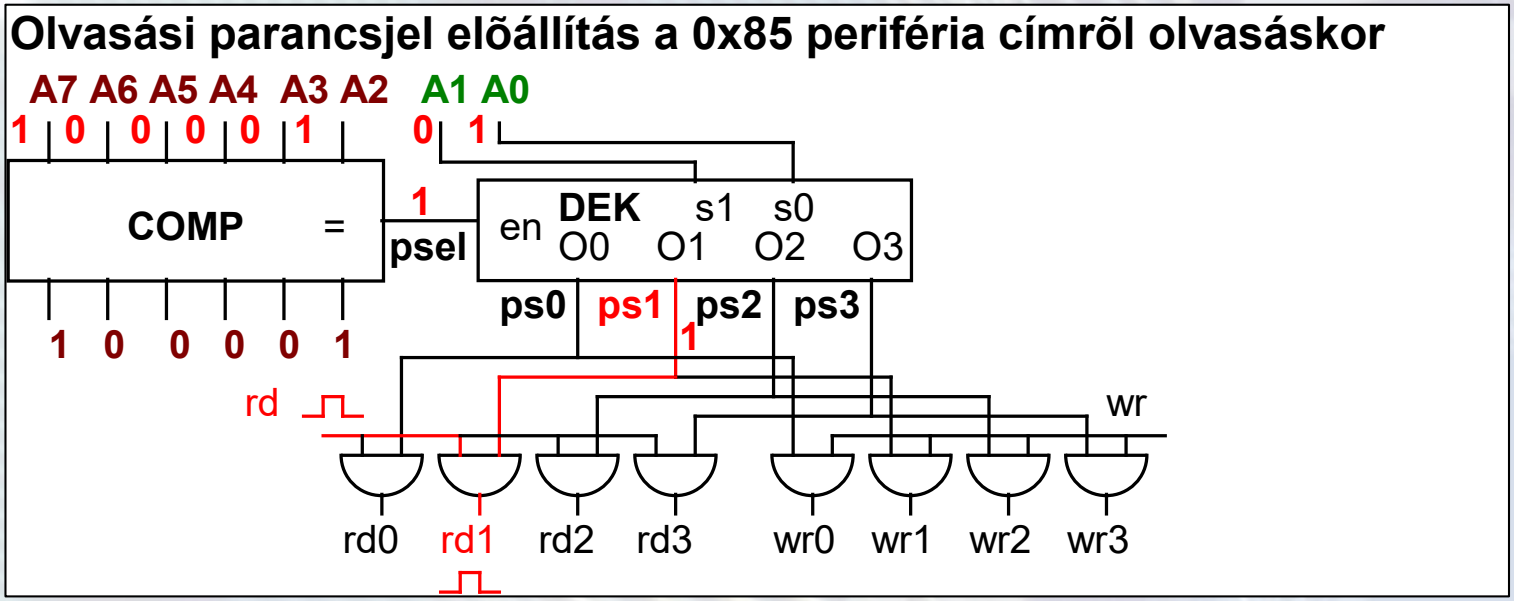


MiniRISC processzor – Perifériaillesztés

- BÁZIS_CÍM bitek REGISZTER_CÍM bitek

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

Példa olvasásra: mov r0, 0x85



MiniRISC processzor – Memóriaillesztés

(Memória címtartomány dekódolása)

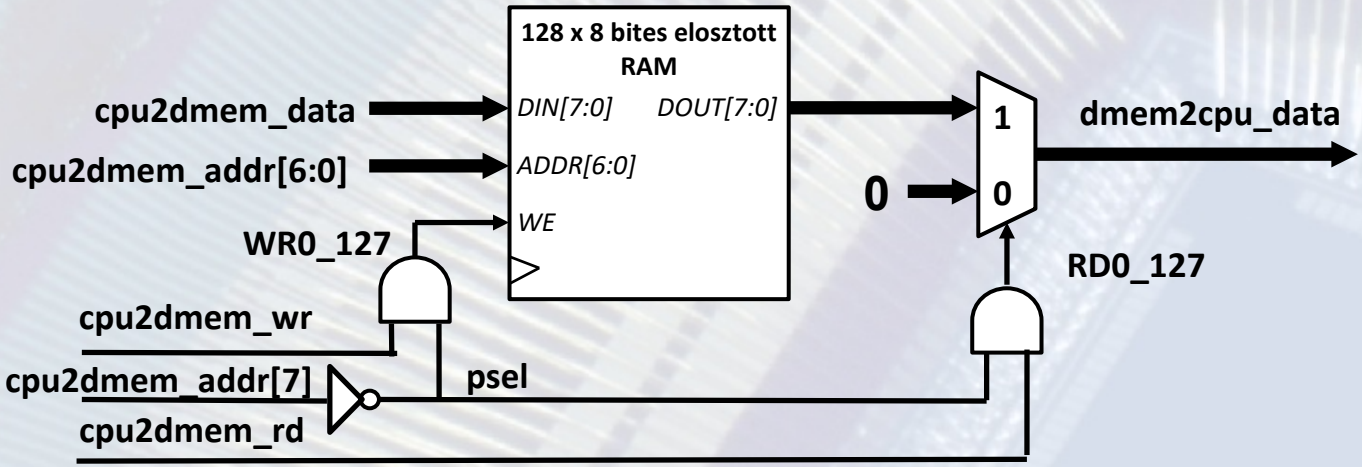
Feladat: 128 x 8 bites memória illesztése a processzoros rendszerhez

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

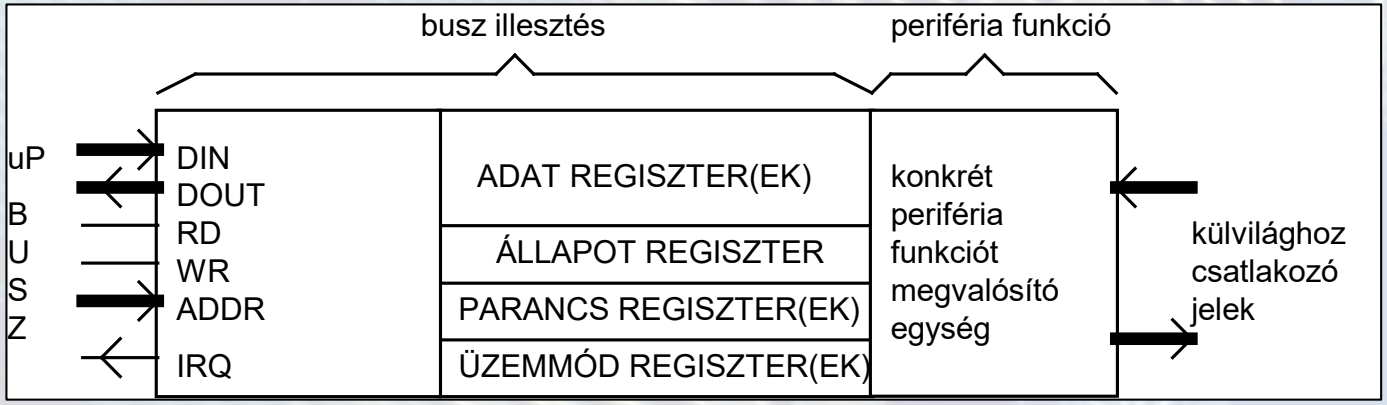
(Megvalósítás Verilog nyelven)

Feladat: 128 x 8 bites memória illesztése a processzoros rendszerhez

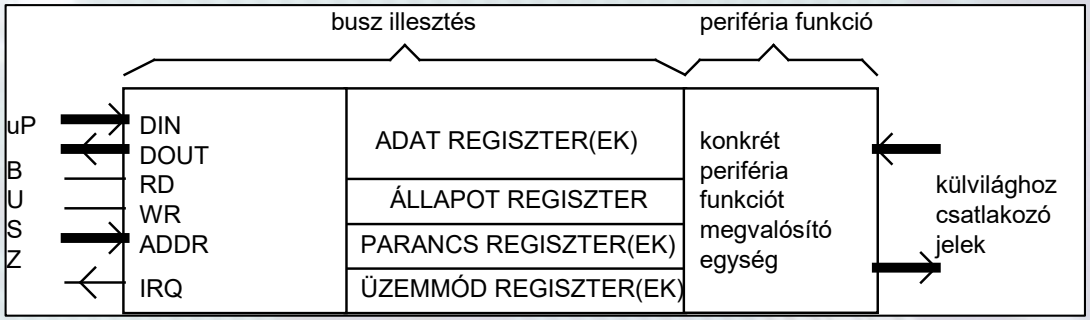
```
wire        memsel   = ~cpu2dmem_addr[7];     //A memória kiválasztó jele.
wire [6:0] mem_addr = cpu2dmem_addr[6:0];     //A memória címbusza.
wire        mem_wr   = memsel & cpu2dmem_wr;  //A memória írás engedélyező jele.
wire        mem_rd   = memsel & cpu2dmem_rd;  //A memória olvasás engedélyező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

MiniRISC processzor – Perifériaillesztés

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



MiniRISC processzor – Perifériaillesztés



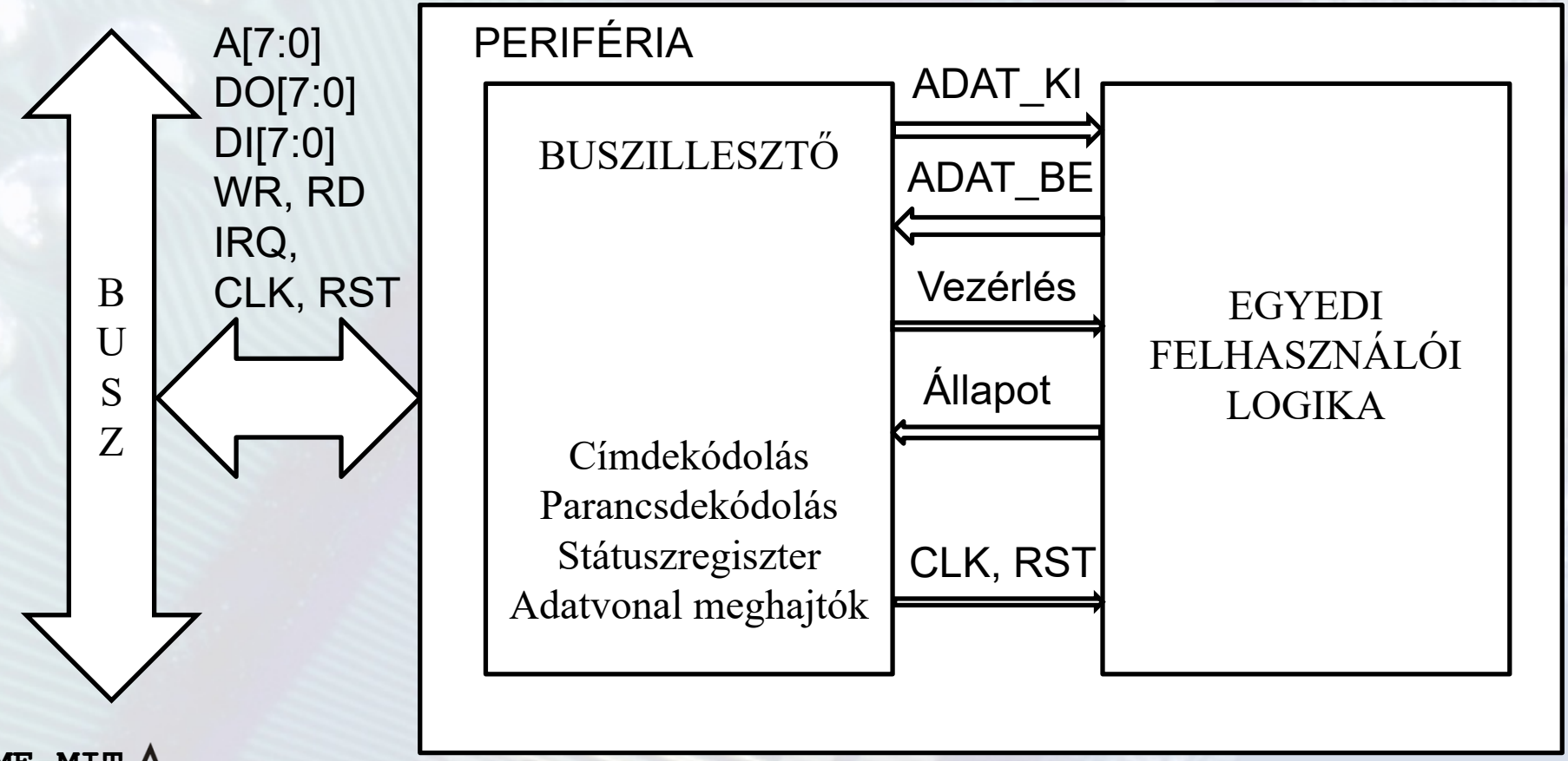
- **Üzemmód regiszter:** A periféria működési módjának beállítására szolgál. Egy-egy periféria egy adott feladatcsoport megoldását teszi lehetővé konfigurálható üzemmódok segítségével (pl. egy timer egységgel időzítési, számlálási, impulzus generálási feladatok oldhatók meg).
- **Parancs regiszter:** 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 (nem feltétlenül ugyanazt, amit beírtunk).

MiniRISC processzor – Perifériaillesztés

- **A perifériaillesztési feladat lépései**
- **A periféria típusa alapján az igények felmérése**
  - Regiszter szám, igény és használati mód meghatározása (írható, olvasható)
    - Adat be-/ki, parancs, státusz, üzemmód, stb. regiszterek
- **A báziscím kijelölése, a címtartomány használatának megtervezése**
  - Mindig teljes 2^N méretű címtartomány jelölünk ki, azaz 1-2-4-8-16 méretű regisztercím tartományt foglalunk le
  - Az egyes regiszter címek így a (báziscímhez képest relatív) 0,1,2,3..címeken lesznek!

MiniRISC processzor – Perifériaillesztés

- **A perifériaillesztés általános tulajdonságai**
  - Minden egység ezeket a részleteket tartalmazza



MiniRISC processzor – Perifériaillesztés

- **A perifériaillesztés általános tulajdonságai**
- **A BUSZ tulajdonságai rendszerfüggőek**
  - A jelek száma, használati módja ezért egyedi lehet
  - Mi a **MiniRISC** egyszerű belső buszával tervezünk
  - Kommunikáció: 8 CÍM, 8 ADAT_KI, 8 ADAT_BE vonal,
  - Vezérlőjelek: CLK, RST, WR, RD, IRQ,
    - Busz master egységeknél: REQ, GRANT (nem tanuljuk)
- A **buszillesztő logika felépítése** tehát nem annyira a periféria, mint **inkább a busz tulajdonságai által meghatározott**.

MiniRISC processzor – Perifériaillesztés

- **A perifériaillesztés általános tulajdonságai**
- **A buszillesztő logika és a felhasználói logika interfésze**
  - 8 **ADAT_KI**, 8 **ADAT_BE**, az adatátviteli vonalak
  - **Vezérlő jelek** a felhasználói logika működtetésére, a regiszterek beírására, kiválasztására
  - **Állapotjelek**, melyek a periféria működéséről tájékoztatnak (az állapotregiszterből kiolvashatók)
- Az egyes részáramköröket a továbbiakban elemezzük

MiniRISC processzor – Perifériaillesztés

- **A címdekódolás kialakítása (BÁZIS_CÍM felismerése)**
  - ***psel = ((cpu2dmem_addr >> N) == (BASEADDR >> N))***
  - A címtartomány mérete 2^N byte (1,2,4,8,16..)
- **Parancsdekódolás:**
  - Írás engedélyező jelek
  - ***xxx_wr = psel & cpu2dmem_wr & (cpu2dmem_addr[N-1:0] == REGADDR)***
  - Az egyes perifériaregiszterek beíró/betöltő jelei (**WR0, WR1, WR2**…)
  - Olvasás engedélyező jelek
  - ***xxx_rd = psel & cpu2dmem_rd & (cpu2dmem_addr[N-1:0] == REGADDR)***
  - Az egyes perifériaregiszterek olvasás engedélyező jelei (**RD0, RD1, RD2**…)
  - A jeleket a kimeneti elosztott MUX (AND-OR hálózat) vezérlésére, engedélyezésére használjuk: egy időben csak egy kimeneten lehet érvényes adat, a többi periféria kimenetének értéke inaktív nulla lesz
  - Használni kell még, ha az olvasás állapotváltozást okoz (pl. FIFO)

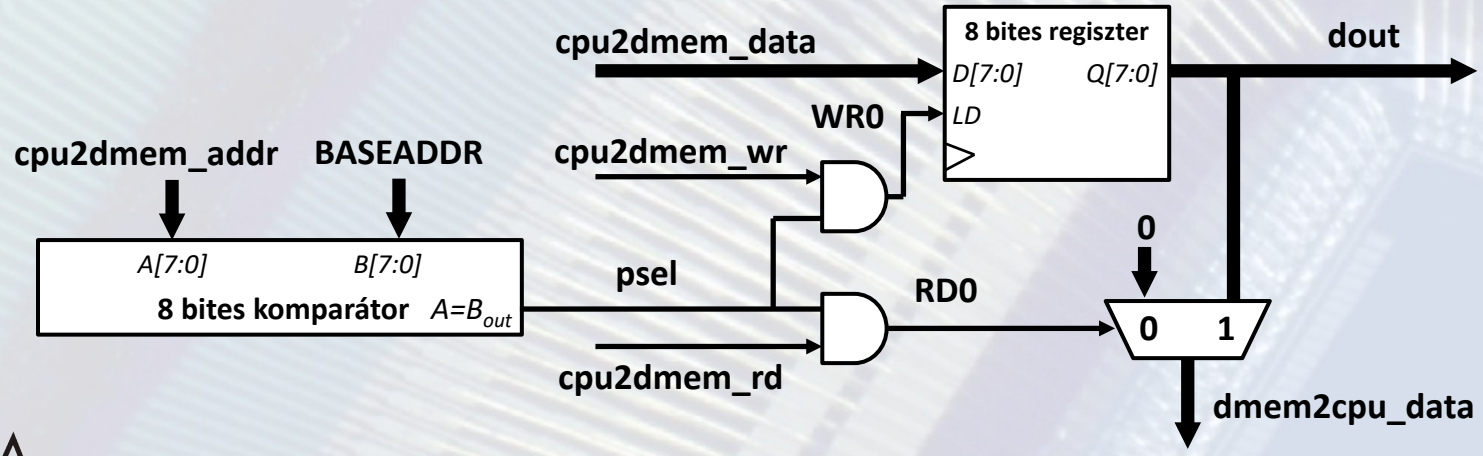
MiniRISC processzor – Perifériaillesztés

(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**
- **Adatregiszter: BASEADDR + 0x00, 8 bites, írható/olvasható**
  - Az OUT_i bit hajtja meg az i-edik LED-et

7. bit	6. bit	5. bit	4. bit	3. bit	2. bit	1. bit	0. bit
OUT7	OUT6	OUT5	OUT4	OUT3	OUT2	OUT1	OUT0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W



MiniRISC processzor – Perifériaillesztés

(1. példa – Megvalósítás Verilog nyelven)

Feladat: 8 darab LED illesztése a processzoros rendszerhez

```
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] cpu2dmem_addr,
   input  wire       cpu2dmem_wr,
   input  wire       cpu2dmem_rd,
   input  wire [7:0] cpu2dmem_data,
   output reg  [7:0] dmem2cpu_data,

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

//A periféria kiválasztó jele.
wire psel = (cpu2dmem_addr == BASEADDR);
```

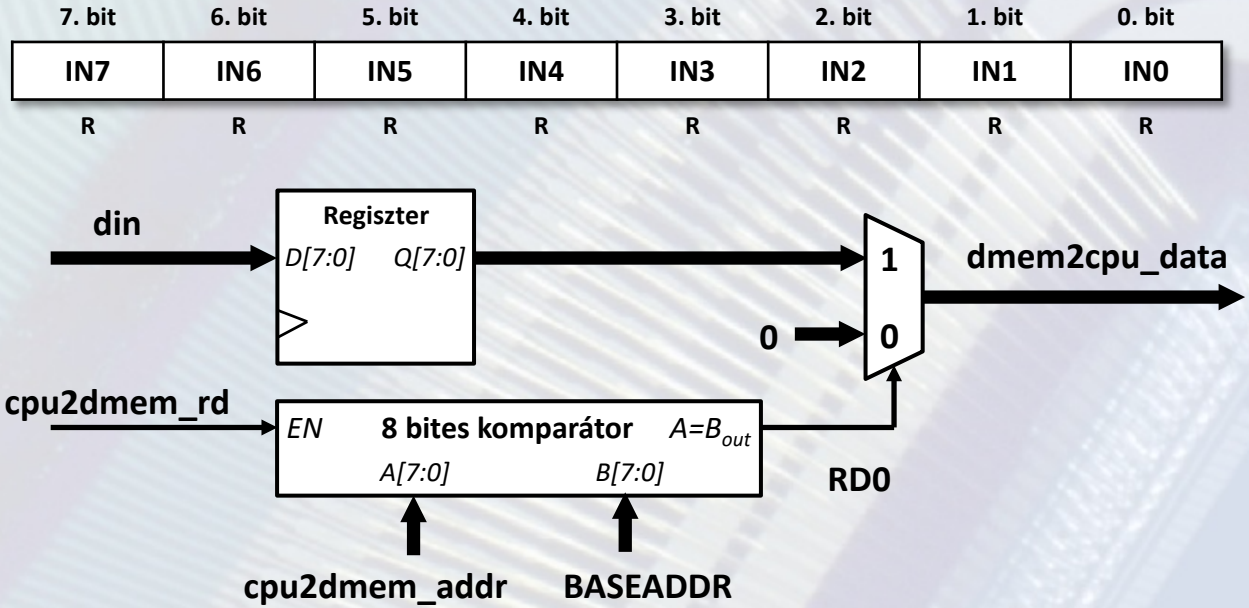
```
//Az adatreg. írás engedélyező jele.
wire dreg_wr = psel & cpu2dmem_wr;

//Az adatreg. olvasás engedélyező jele.
wire dreg_rd = psel & cpu2dmem_rd;

//Adatregiszter.
always @(posedge clk)
   if (rst)
      dout <= 8'd0;
   else
      if (dreg_wr)
         dout <= cpu2dmem_data;

//Az olvasási adatbusz meghajtása.
always @(*)
   if (dreg_rd)
      dmem2cpu_data <= dout;
   else
      dmem2cpu_data <= 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



MiniRISC processzor – Perifériaillesztés

(2. példa – Megvalósítás Verilog nyelven)

Feladat: 8 darab kapcsoló 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] cpu2dmem_addr,
   input  wire       cpu2dmem_rd,
   output reg  [7:0] dmem2cpu_data,

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

//A periféria kiválasztó jele.
wire psel = (cpu2dmem_addr == BASEADDR);
```

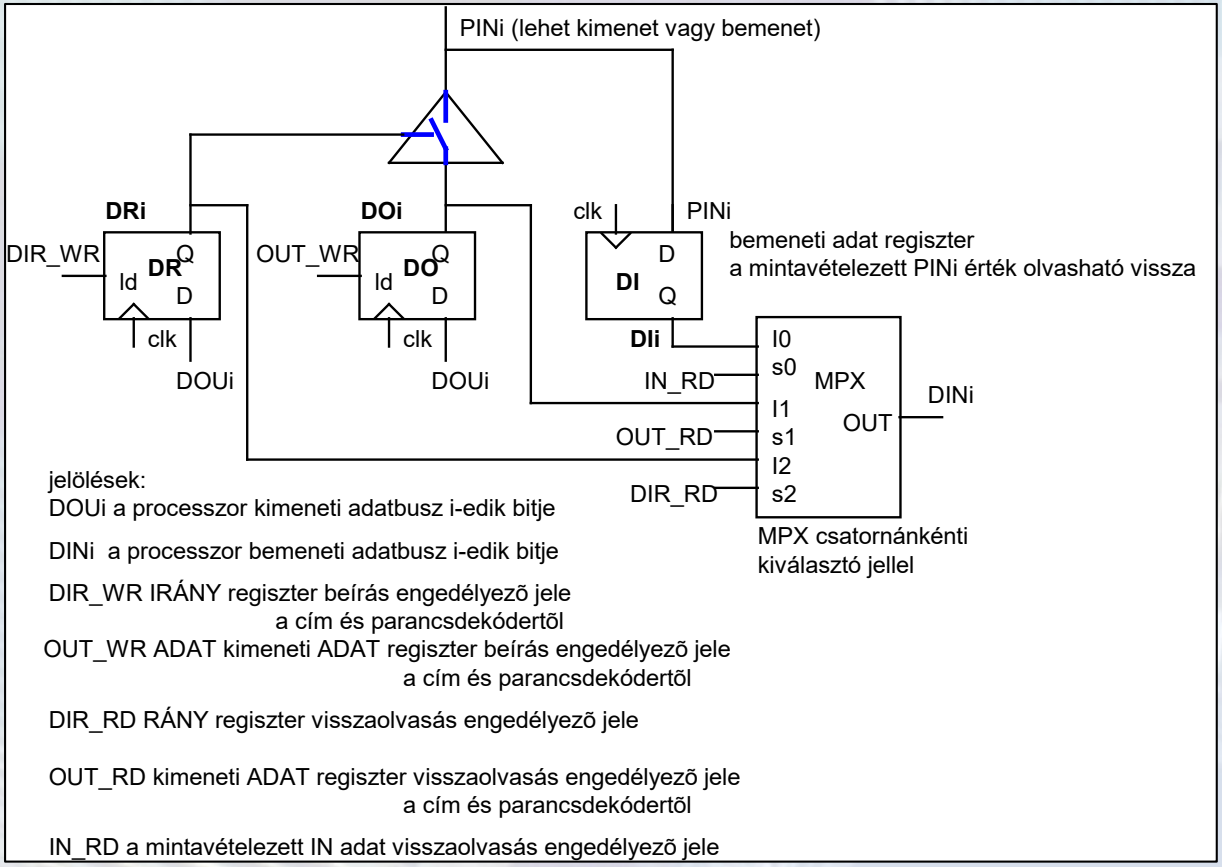
```
//Az adatreg. olvasás engedélyező jele.
wire in_reg_rd = psel & cpu2dmem_rd;

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

always @(posedge clk)
   if (rst)
      in_reg <= 8'd0;
   else
      in_reg <= din;

//Az olvasási adatbusz meghajtása.
always @(*)
   if (in_reg_rd)
      dmem2cpu_data <= in_reg;
   else
      dmem2cpu_data <= 8'd0;

endmodule
```

GPIO Periféria

- **A GPIO általános célú be-/kimeneti periféria**
- **A mikroprocesszorok gyakran rendelkeznek tetszőlegesen használható GPIO interfészekkel**
- **Ez a funkció többnyire „másodlagos” kialakítású**
  - A mikroprocesszor valamelyik speciális interfészének kivezetését rendelték az adott lábhoz
  - Ha azt nem használjuk, akkor lehet GPIO bites/bájtos interfészként is használni a (maradék) lábat/lábakat

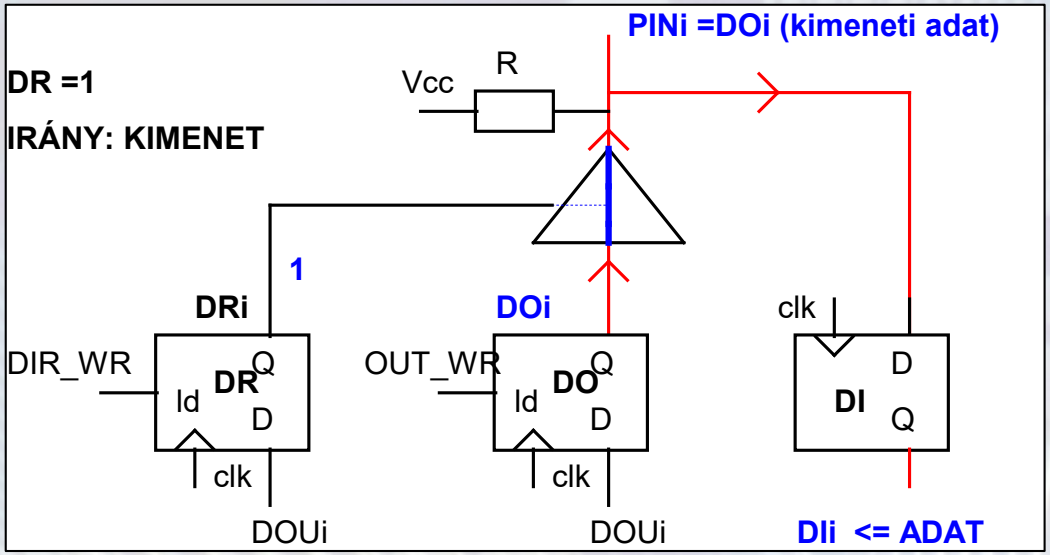
GPIO Periféria

- **A GPIO periféria a PIN-ekkel csatlakozik a külvilághoz. Minden PIN-hez 3 regiszter egy-egy bitje tartozik.**
- **Adat kimeneti regiszter: DO**. Ide kell írni a PIN-en megjelenítendő adatot.
- **Adat bemeneti regiszter**: DI. Itt lehet beolvasni a PIN-en levő adatot.
- **Irány regiszter: DR**. Ide 1-et kell írni, ha a DO regiszterben levő értéket a PIN-re akarjuk kapcsolni, 0-t, ha a PIN-re kívülről kapcsolódik adat és azt akarjuk beolvasni. A három állpotú meghajtó engedélyezését vezérli.



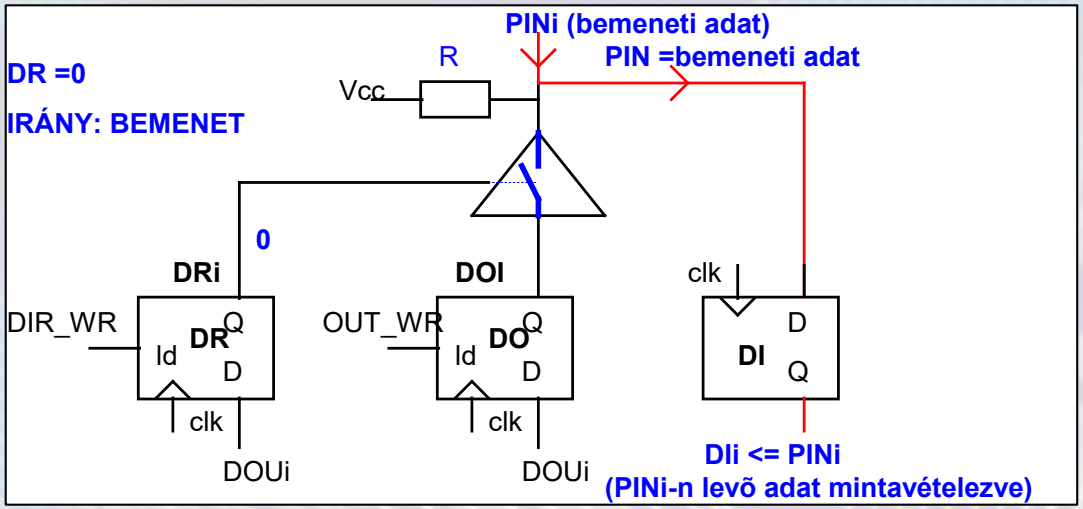
GPIO Periféria

- **A GPIO működés kimeneti módba állítva:**
- A Dri-be 1-et írva a 3 állapotú meg-hajtó a PINi-re kap-csolja a kimeneti adatregisztert (Doi-t), tehát a PIN-en ennek logikai értéke jele-nik meg és Dli-ből (bemeneti adatregiszter) ez olvasható viszsza.



GPIO Periféria

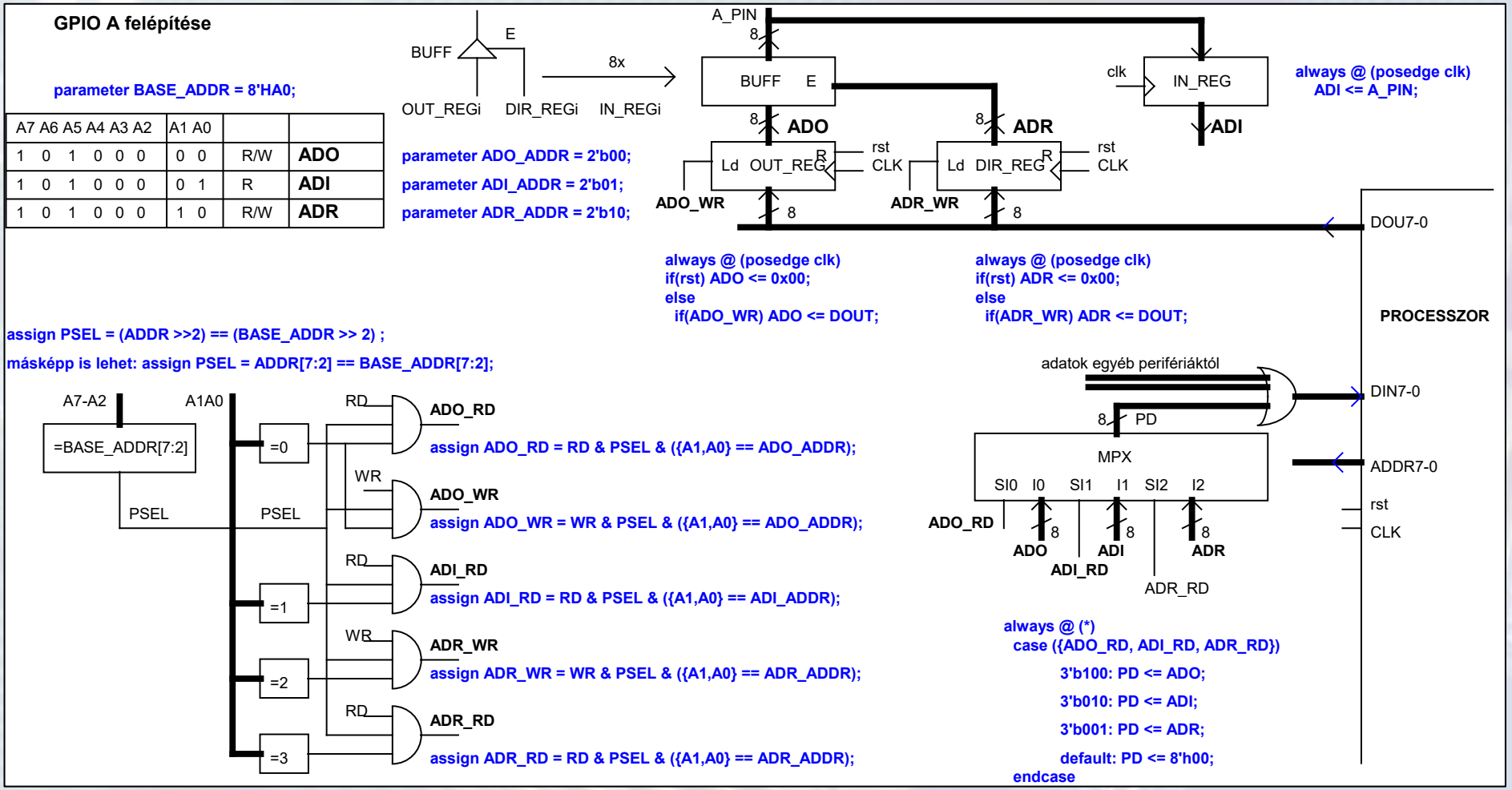
- **A GPIO működés bemeneti módba állítva:**
- A Dri-be 0-át írva a 3 állapotú meghajtó kikapcsolt állapotba kerül. Ilyenkor szabad a PINi-t kívülről meghajtani valamilyen logikai értékkel.



Ekkor a DIi-ből (bemeneti adatregiszter) ez a logikai érték olvas-ható visz-sza. A Mini RISC GPIO PIN-jeire egy ellenállással a tápra vannak húzva. Ezért bemeneti módban, ha semmi nem hajtja meg a bemeneti adatbit 1 értékű.

GPIO Periféria

- **A Mini RISC GPIO A illesztése:**



GPIO Periféria

- **A GPIO periféria használata:**
  - Párhuzamos módban, együtt vezérelve a 8 bitet
  - Függetlenül, bitenként megadva és használva az egyes jeleket bemenetként/kimenetként

GPIO Periféria

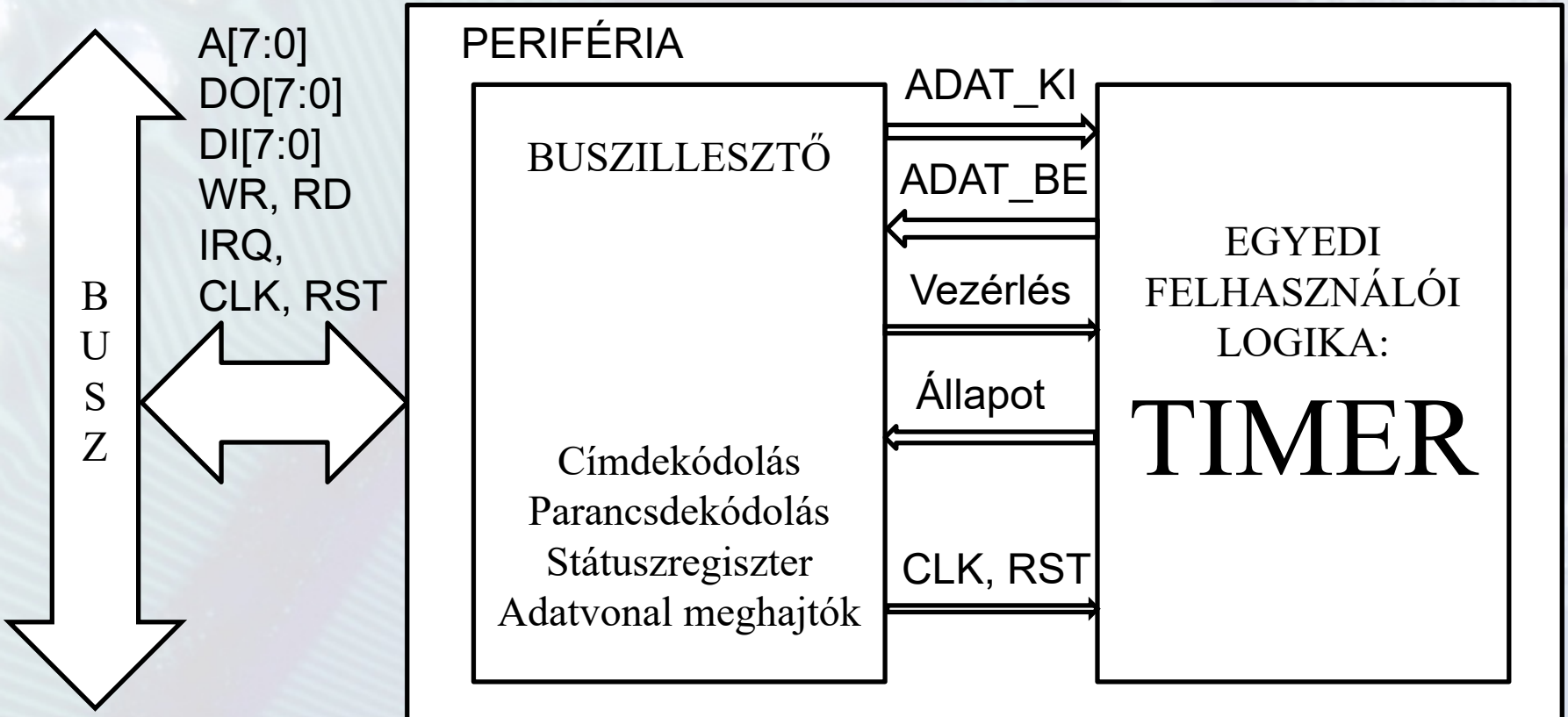
- **A GPIO periféria utasítás szinten programozható**
  - A MiniRISC Load/Store működési elve miatt ezért 2 utasításonként adható ki új érték/változtatható a GPIO láb értéke (2*187,5ns = 375ns)

```
mov r0, #konst
mov ADO, r0
```

- **A GPIO lábakon tetszés szerinti protokoll/időzítési szekvencia lejátszható (maximum a fenti sebességgel)**
  - Ezt hívjuk „**bit-banging**” módnak, azaz a processzor a programból, a megfelelő időzítéssel, dinamikusan változtatja a lábakon megjelenő kimeneti értékeket, illetve mintavételezi és beolvassa a bemeneti értékeket.
  - Az időzítés nem mindig pontos, kisebb ingadozás, „jitter” felléphet

MiniRISC processzor – Perifériaillesztés

Időzítő periféria (TIMER) a processzoros időzítések támogatására



MiniRISC processzor – Perifériaillesztés

- **Probléma:** A processzorral programozottan tudunk időzítést beállítani (tervezett hosszúságú ciklussal), de akkor a processzor nem tud hasznos munkát végezni
- A idő múlását számolgató utasításciklusok alatt végezhetne hasznos feladatokat is a processzor
- Erre használható a **TIMER** (időzítő) periféria
- Az időzítő perifériák a leggyakoribb perifériák a processzoros rendszerekben
- Az időzítő perifériák legfontosabb funkcionális eleme a **számláló**.

MiniRISC processzor – Perifériaillesztés

- **A MinRISC processzor TIMER perifériájának paraméterei:**
  - **8 bites** időzítő **számláló TM**, amivel 1 - 256 ütemezési ciklus hosszú időzítés állítható be. A számláló egy tölthető, engedélyezhető, **lefelé számláló**, ami a végértéknél, azaz a nulla érték elérésekor jelez. (Időzítés letelt, **TOUT**).
  - Az időzítő rendelkezik egy **programozható előosztóval PS**, ami a TR számlálót a beállított számú órajel leteltekor lépteti. A PS a rendszer órajelet programozhatóan leosztja 1-16-64-256-1024-4096-16384-65536 értékkel, azaz a PS érték 2^0-tól 2^{16}-ig skálázható
  - A relatív időzítés pontossága mindig < 0,5% (1/256)!
  - A beállítható teljes időzítési tartomány

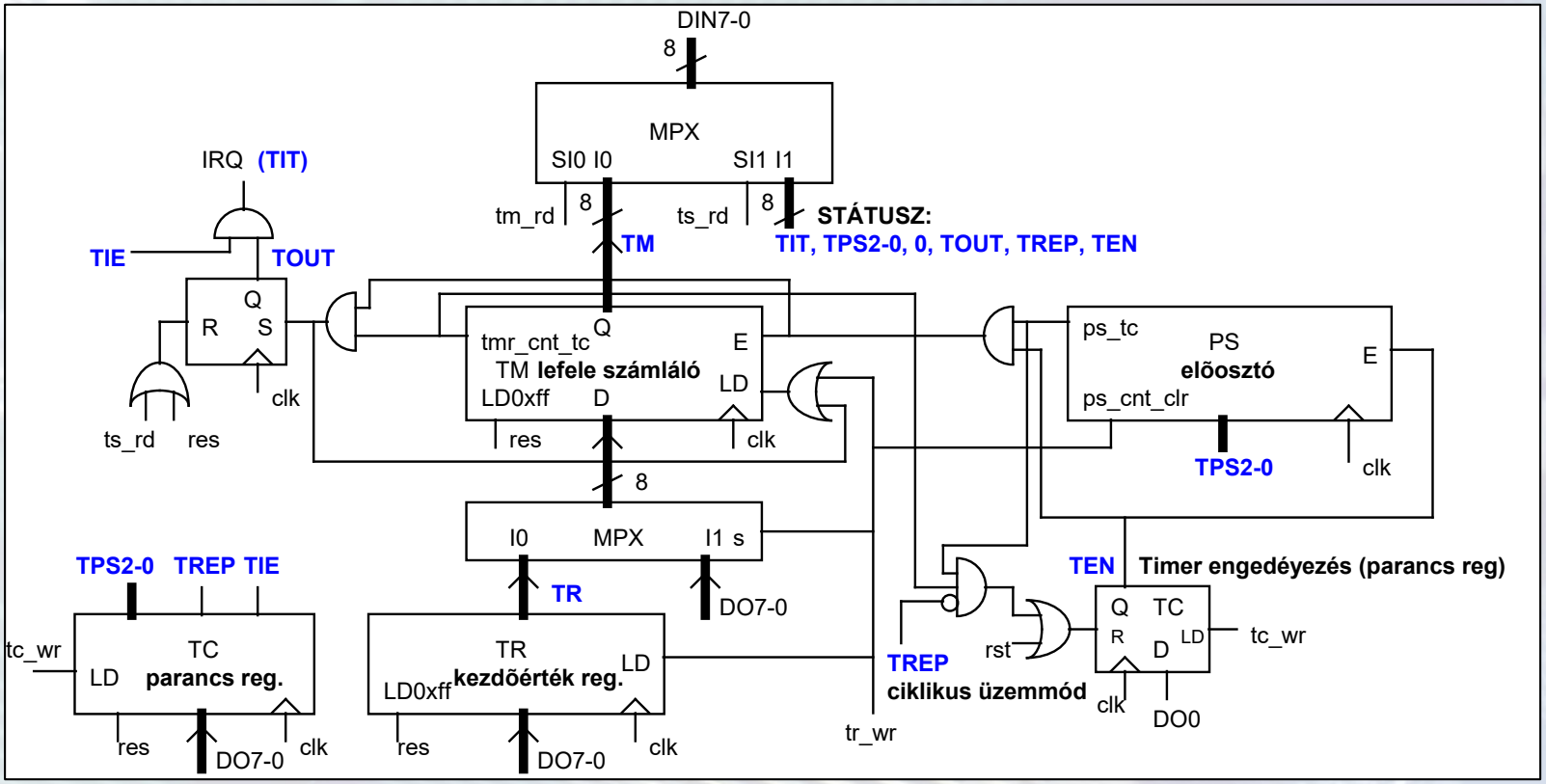
$$T = (TR+1)*PS*T_{clk} \quad (62{,}5ns – 1{,}048s)$$

MiniRISC processzor – Perifériaillesztés

- **A TIMER periféria üzemmódjai:**
  - Egyszeres lefutási mód, az időzítés végén jelzéssel és leállással
  - Periodikus (újra indulásos) mód, az időzítés végén jelez és automatikus újraindul.
  - Az időzítés végét jelző TOUT jelzés állapota lekérdezéssel kiolvasható, vagy ha engedélyezett, akkor lehetőség van megszakításkérésre is.
  - Az időzítő regiszter TR aktuális értéke időzítés közben bármikor visszaolvasható

MiniRISC processzor – Perifériaillesztés

- **A TIMER periféria felépítése**



A cím és parancs dekódert nem rajzoltuk fel.

MiniRISC processzor – Perifériaillesztés

A TIMER egység programozói felülete

- **TR (0x82): kezdőállapot regiszter (csak írható)**
  - Az ide írt érték az időzítés kezdetekor beíródik a TM számlálóba ami innen számol lefele 0-ig.
- **TM (0x82): számláló regszter**
  - A lefele számláló értéke innen olvasható ki (csak olvasható)
- **TC (0x83) Parancsregiszter (írható), TS(0x83) státusregiszter (olvasható):**

	D7	D6	D5	D4	D3	D2	D1	D0
parancs regiszter TC:	TIE	TPS2	TPS1	TPS0	-		TREP	TEN
státus regszter TS:	TIT	TPS2	TPS1	TPS0	0	TOUT	TREP	TEN

- **TIE (**interrupt engedéylezés), **TPS[2:0] (**előoszás megadása), **TREP** (0: egyszeri jelzés 1: ciklikus jelzés)**, TEN: (**működés engedélyezés**)**

TPS2-0:	**000**	**001**	**010**	**011**	**100**	**101**	**110**	**111**
előosztás:	1	16	64	256	1024	4096	16384	65536
Periódusidő (frekvencia) ha fclk: 16MHz	1/16 us (16MHz)	1 us (1MHz)	4 us (250kHz)	16 us (62500Hz)	64 us (15625Hz)	256 us (3906.25Hz)	1024 us (976.5625Hz)	4096 us (244.141 Hz)

- A státusregiszter bitek részben a parancsregiszter bitjei, új bit a **TOUT**, ami az időzítés lejártának állapotát jelzi és a **TIT** amely az interrupt kérést jelzi. **A státusregiszter olvasása törli a TOUT jelzést.**

MiniRISC processzor – Perifériaillesztés

- **A Timer egység programozói felületete**
- **4 interfész regiszter, ebből kettő-kettő azonos címen**
- **A Báziscím 0x82, a regisztercímek 0x82 és 0x83.**

Időzítő	TR	kezdőállapot regiszter	0x82	WR	0xFF	Az időzítő számláló kezdeti értéke					
	TM	számláló regiszter	0x82	RD	0xFF	Az időzítő számláló aktuális értéke					
	TC	parancs regiszter	0x83	WR	0x00	TIE	TPS[2:0] - előosztás	-	-	TREP	TEN
	TS	státusz regiszter	0x83	RD	0x00	TIT	TPS[2:0] - előosztás	0	TOUT	TREP	TEN

MiniRISC processzor – Perifériaillesztés

- **A TIMER periféria használata**
- **Példa: 0.5sec-os időzítés felprogramozása**
  - Üzemmód: nincs megszakítás, előosztás 65536, ismétléses üzemmód, működés engedélyezés
    TC: TIE, TPS[2:0],0,0,TREP, TEN
    0 111 0 0 1 1
  - 1/2= 1/16e6 *65536*(N+1) -> N = 16e6/(2*65536)-1 = 121

```
                        ; Az időzítő beállítása kb. 0,5 s periódusidőre (2 Hz).
                        ; 16*10^6 Hz / 65536 / 122 = 2 Hz
    mov     r2, #121    ; A számláló regiszter értéke 121 (0x79).
    mov     TM, r2      ; Beírás a TM időzítő regiszterbe.
    mov     r2, #0x73   ; Az időzítő konfigurálása: 65536-os előosztás,
    mov     TC, r2      ; ismétléses mód, időzítő engedélyezve (0111_0011).
    mov     r2, TS      ; Az időzítő esetleges TOUT jelzésének törlése.
```

A TOUT státuszbit tesztelése:

TS: TIE, TPS[2:0],0,**TOUT**,TREP, TEN
0 000 0 **1** 0 0

Kimaszkoljuk (bitenkénti ÉS kapcsolatba hozzuk) kiolvasott TOUT értékét a 0b00000100 = 0x04 maszkkal
Ha az eredmény nem 0, akkor lejárt az időzítés, egyébként maradunk a ciklusban. Az TS olvasása egyben a TOUT jelzést is törli.

```
tm_loop:                ; Várakozunk, amíg az időzítő nem jelez.
    mov     r2, TS      ; Beolvassuk az időzítő státusz regisztert
    tst     r2, #0x04   ; a TOUT bit vizsgálatához.
    jz      tm_loop     ; Várakozás, ha a TOUT bit még mindig 0.
```

Megjegyzés: A tm_loop-ban bármi „hasznosat” is csinálhatnánk, a processzor ráér, van szabad utasítás végrehajtási ideje.

Digitális technika
11. EA vége