



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

Digitális technika (VIMIAA03)

10. gyakorlat és laboratórium

Raikovich Tamás
BME MIT

GCD (Inko) számítása

- A 8. laboron a legnagyobb közös osztó számítást hardveresen valósítottuk meg
- A mai laboron ezt szoftveresen valósítjuk meg a MiniRISC processzoros rendszerre

GCD (Inko) számítása – ismétlés

- Egy számpár legnagyobb közös osztóját keressük
- $\text{GCD}(a,b)$ Euklideszi algoritmus \rightarrow maradékos osztás
 - $a = b * q_1 + r_1,$ $r_1 = a \% b$
 - $b = r_1 * q_2 + r_2,$ $r_2 = b \% r_1$
 - $r_1 = r_2 * q_3 + r_3$ $r_3 = r_1 \% r_2$
 -, ahol $|b| > r_1 > r_2 \dots >= 0$ ahol % a mod operátor
 - A GCD az utolsó nem nulla maradék
 - Jó algoritmus, viszonylag gyorsan konvergál
- **DE: A MiniRISC processzornak nincs osztó utasítása**
 - Készítsünk osztó szubrutint? Lehet, de nem könnyű.

GCD (Inko) számítása – ismétlés

- **Egyszerűsítsük az algoritmust az utasításkészletben rendelkezésre álló műveletekre**
 - $\text{GCD}(a,b) = \text{GCD}(a-b, b)$, ha $a > b$ Művelet: $a-b \rightarrow a$
 - $\text{GCD}(a,b) = \text{GCD}(a, b-a)$, ha $b > a$ Művelet: $b-a \rightarrow b$
 - Leállás adott lépés után, ha $a = b$, ez a $\text{GCD}(a,b)$
- **Ez már egyszerűen tervezhető, de több iterációt igényel, a végrehajtás hosszabb ideig tart**
 - Különösen relatív prímeknél

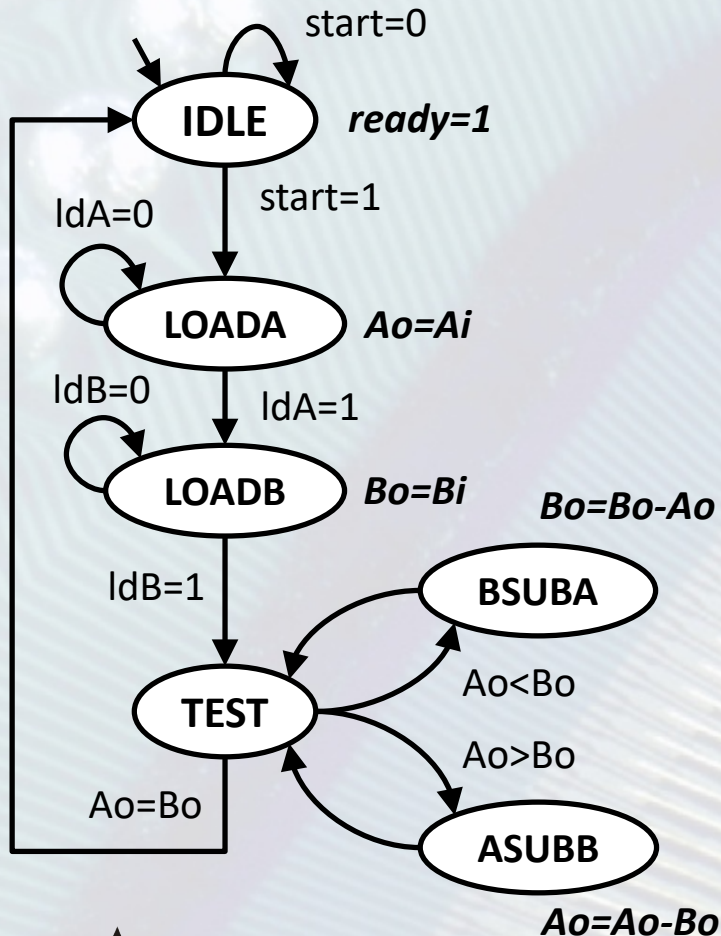
GCD (Inko) számítása – ismétlés

Több lehetséges HW adatstruktúra verzió (és ennek megfelelően algoritmus végrehajtás lehetséges)

- A gyakorlaton röviden megbeszéltük mindhárom lehetőséget
- A szoftveres megvalósításhoz a **3. verzió (két kivonó használata)** áll a legközelebb
 - Két azonos felépítésű egységet tervezünk, mindkettő képes minden ütemben kivonást végezni a saját tartalma és a másik regiszter tartalma között. Csak azt az eredményt tároljuk, ahol a nagyobb operandus volt. Amikor $a = b$, leállítás.

GCD (Inko) számítása – ismétlés

Magasszintű állapotgép



Interfész és belső regiszterek

- Bemenetek: start, IdA, IdB, Ai, Bi
- Kimenetek: ready, Ao, Bo
- Regiszterek: Ao, Bo

HW megvalósítás (8. labor)

- Adatstruktúra
- Vezérlő állapotgép

SW megvalósítás (most)

- Rajzoljuk fel a folyamatábrát
- Hogyan valósíthatók meg a magasszintű aritmetikai műveletek és állapotátmeneti feltételek?

GCD (Inko) számítása

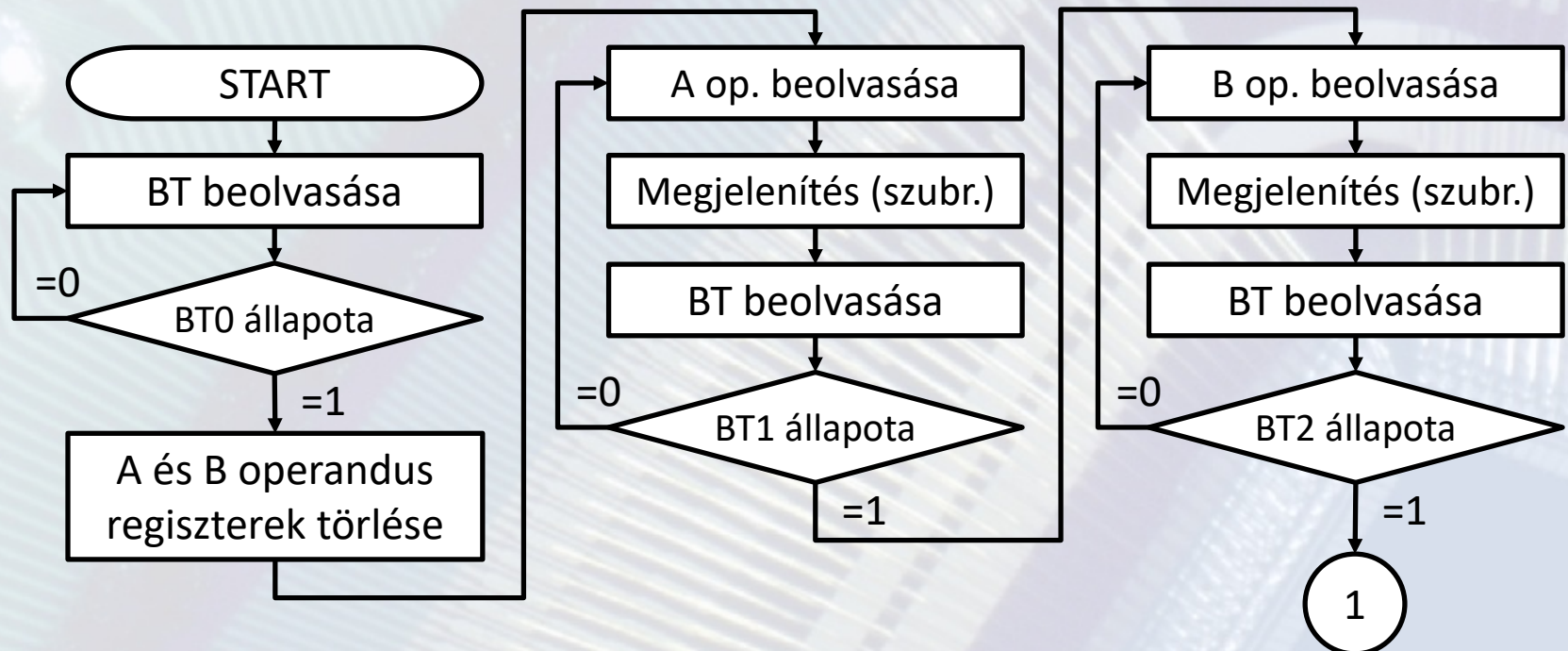
A GCD számító szoftver működésének specifikációja

1. Indítás a BT0 gomb lenyomásának hatására
2. Az A operandus betöltése a BT1 gomb lenyomására
3. A B operandus betöltése a BT2 gomb lenyomására
 - Az operandusokat a DIP kapcsolón adjuk meg
 - Az operandusok betöltése alatt a kapcsolók állapota jelenjen meg a hétszegmenses kijelzőn
4. A legnagyobb közös osztó számítás elvégzése
 - A részeredmények és az LNKO megjelenítése a hétszegmenses kijelzőn
 - A MiniRISC processzor gyors, ezért a részeredmények megtekintéséhez például töréspontot kell elhelyezni a megfelelő forráskód sorhoz

GCD (Inko) számítása

Indítás és az operandusok betöltése

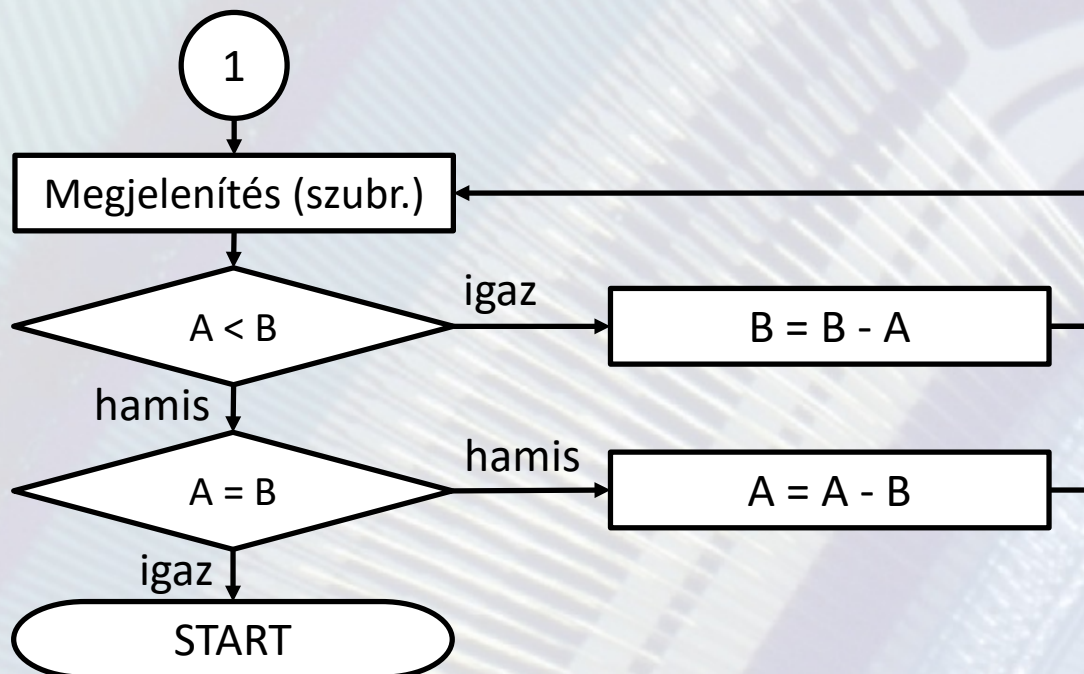
- Hogyan tudjuk vizsgálni a nyomógombok állapotát?
- Milyen feltételes ugró utasítást kell használnunk?



GCD (Inko) számítása

Az LNKO számítás elvégzése

- Hogyan tudjuk megállapítani A és B relációját?
- Milyen feltétel bitek vizsgálata szükséges ehhez?



GCD (Inko) számítása

2 digitos megjelenítés a hétszegmenses kijelzőn

- Hogyan használható a kijelző a MiniRISC processzoros rendszerben?
- Készítsünk szubrutint, amely megjelenít egy 8 bites számot hexadecimálisan a hétszegmenses kijelzőn.
Paraméterek:
 - A megjelenítendő érték
 - Az egyesek helyiétékéhez tartozó digit regiszter címe
- Hogy valósítható meg hatékonyan a hétszegmenses dekóder szoftveresen?
- Milyen címzést kell használnunk?

GCD (Inko) számítása

Néhány bemeneti adat a várt eredménnyel

- Milyen formátumban kell megadni a bemeneti adatokat?

Ai operandus	Bi operandus	Inko(Ai, Bi)
35 = 5·7 (0x23, 0010_0011)	63=3 ² ·7 (0x3f, 0011_1111)	7 (0x07, 0000_0111)
66 = 2·3·11 (0x42, 0100_0010)	84 = 2 ² ·3·7 (0x54, 0101_0100)	6 (0x06, 0000_0110)
84 = 2 ² ·3·7 (0x54, 0101_0100)	70 = 2·5·7 (0x46, 0100_0110)	14 (0x0e, 0000_1110)
65 = 5·13 (0x41, 0100_0001)	21 = 3·7 (0x15, 0001_0101)	1 (0x01, 0000_0001)

10-es számrendszer használata (opcionális)

- **A bemenetek bináris (hexadecimális) számrendszerben történő megadása nem túl kényelmes**
- **Tudnánk-e használni a decimális számrendszert is?**
 - Igen, BCD kódolású adatokkal
 - A komparálás működik ilyen adatok esetén is
 - A kivonáshoz viszont BCD kivonó szükséges
 - Helyettesítsük a bináris kivonást (SUB utasítás) az előre megírt BCD kivonó szubrutinnal