## Faktoriális generátor program MiniRISC-re, érték a memóriában BCD-ben, amíg > GOOGOL! (70!)

A feladat megvalósításában a lehető legegyszerűbb módszert választottam: f(0) = 1, és minden f(n) = f(n-1) \* n.

Mivel az utasításkészlet nem tartalmaz szorzást, főleg nem ekkorát, más módszert kellett kitalálnom. A megoldás, amit kitaláltam, abból áll, hogy egy háromszorosan egymásba ágyazott ciklusban összeadok, aminek a vége egy nagyszámokkal történő szorzás lesz. Ehhez először lemásolom az aktuális értéket a memória második feléből az első felébe, majd ezt az értéket adom hozzá az eredeti helyen lévő adatokhoz n-szer minden ciklusban. Ez egy nagyon nem optimális megoldás, sokféleképpen lehetne optimalizálni – ebből az én megoldásom csak azt tartalmazza, hogy minden n>10 számra először 4 bittel balra shifteli az egész értéket, amivel egy helyiértékkel nőtt minden BCD számjegy, tehát egy 10-zel való szorzást valósítottam meg.

Ez a megoldás maximum 84! < 10^(0x40\*2) < 85! számokat képes megjeleníteni, tehát max 84!-ig mehetünk.

A programom moduláris, azaz a moduljait egyesével is lehet használni – ilyen pl. az a függvény, ami lenullázza a legelején a teljes memóriát (ezt nem tehetjük alapból fel, hogy 0), vagy ilyen a BCD összeadóm. A program fő váza a következő modell szerint működik:

|  |  |
| --- | --- |
| C:\Users\bajcz\Downloads\1.png | A teljes program vázának részletezése ASM modellel túl hosszú lenne, ezért csak szóban fejtem ki (illetve a programban szinte minden sor végén van comment, hogy épp mint tettem az adott sorban): Beállítás  * Felhasznált regiszterek lenullázása (XOR önmagával) * Memóriaterület lenullázása * Kezdőérték (0! = 1) memóriába írása  Szorzás  * Adott értéket a memória felső részébe átmásoljuk * Elimináljuk az első tíz összeadást (ha tudjuk) * Hozzáadjuk a lemásolt értéket az eredetihez n-szer   + *addbcd* segítségével az egész alsó 0x40 byte-ot adjuk a teljes felső 0x40 byte-hoz  LED  * Megszámoljuk a tízeseket * Levonjuk a tízesek számát (\*10) az egyesekből * Össze VAGY-oljuk a két értéket * Kiírjuk ezt a LED-ekre  Végérték  * Definiálva van (alapból 84, ez a legnagyobb érték)  LED  * Megszámoljuk a tízeseket * Levonjuk a tízesek számát (\*10) az egyesekből * Össze VAGY-oljuk a két értéket * Kiírjuk ezt a LED-ekre, ekkor olvassuk ki a memória felső 40 byte-ját  End  * Tartjuk az értéket |

A kód maga (.lst file):

