Emberi késleltetés mérése

Hüttl Frigyes, Híz Dániel

**Specifikáció**:

Az MCU-n futó program 10-től elkezd visszaszámolni másodpercenként, és az elején az aktuális értéket megjeleníti az Giant Gecko LCD kijelzőjén. A felhasználó feladata, hogy az LCD elsötétülése után jelezze a program számára, szerinte mikor érte el a visszaszámlálás a 0 időpontot. Az eltérést a program eltárolja mind negatív mind pozitív irányba is Ezeket az adatokat ,Linuxos PC-vel összekötve a fejlesztőkártyát, a PC grafikusan is meg tudja jeleníteni.

**Megvalósítás:**

A feladatot FreeRTOS beágyazott operációs rendszerrel valósítottuk meg. Két taszk kommunikál egymással egy globális változón keresztül. Az egyik task a LCD a globális változó kiíratására szolgál, míg a másik a számlálást valósítja meg. A Coun-nak elnevezett taskban a számlálást valósítjuk meg, ha a számláló aktuális érétke kisebb mint 4 , akkor átírjuk a text nevezetű globális változót „WAIT”-re, ha ennél nagyobb a számláló aktuális értéke akkor csökkentjük eggyel és és átalakítjuk úgy, hogy amikor eltároljuk a char típusú text karaktertömbbe akkor is megőrizze az értékét, ehhez a változó egyes számjegyeit tesszük bele. Ezután a taskot késleltetjük 1 másodpercig .

Az LCDCount task a Giant Gecko LCD kijelzőjére írja ki az text változót. Ha megkapja a semphoret akkor ez SegmentLCDWrite függvény meghívásával.

A feladat mikrokontrolleres részéhez még szükségünk volt két darab interruptra. Az egyik a PB0 jelzésű nyomógomb esteén generál egy megszakítást, a másik pedig a TIMRE0 túlcsordulás estén növeli egy változó értékét. A PB0-as gomb által generált megszakítást a GPIO-hoz tartozó páratlan megszakításkezelőfüggvénnyel kezeltük le, ahol megszakítás esetén a program kiszámolja, hogy hányszor lépett a számláló és ez mennyivel tér el az ideális 136720 másodperctől amely a pontos 0-át jelenti esetünkben. Ennek az hibának a kiszámítása: a TIMER0 aktuális értéke + a sec változóban tárolt érték\*13672. A soros porton ezt a különbségi értéket küldjük ki. A számláló megszakítás kezeéőjében, pedig mindig a sec számláló értékét növeltük eggyel.

A feladatmegvalósításához, hozzátartozott még egy linuxos PC-n futó program megírása is.  
A mi feladatunkhoz a programnak a soros porton érkező adatokat kellett, gnuplot segítségével megjeleníteni.

A soros port kezelést és konfiguálást egy külön függvényben végezzük el. A függvényben először konfiguráltuk a soros portot 115200 baudrates kommunikációra és 8N1 (8 adatbit, nincs paritás. 1 stop bit) formátumú üzenetek fogadására után. Az adatokat /dev/ttyACM0-án tudjuk elérni ezért ezt megnyitottuk csak olvashatóként és a beérkező adatokat pedig a data.txt fájlba mentettük le. Ezután az üzenetek olvasás történik, majd kiírása mind a soros sportra, mind a konzol ablakba. A gnuplot hazsnálatához egy csövön keresztül küldjük ki a megjelenítést végző parancsot. A pipe létrehozását a main függvényünkben valósítottuk meg, ahol a gnuplot megnyitása és kezelése is történik a csövön keresztül érkező adatokkal.