



# NanoSD IO Card

Budapest

v0.3 - 2021.06.28.

Vass Sándor

## Tartalomjegyzék

[Tartalomjegyzék](#)

[Áttekintés](#)

[Bekapcsolás előtti tudnivalók](#)

[Felhasználói útmutató](#)

[Bejelentkező képernyő - MENÜ](#)

[Navigálás](#)

[Program betöltése](#)

[Kilépés, BASIC](#)

[További lehetőségek](#)

[Rendezés](#)

[Bekapcsolás](#)

[SD kártya](#)

[CLI - Parancssori interfész](#)

[Programozási segédlet](#)

[Eszközosztály](#)

[Keretcsíkozás](#)

[NanoSD függvények](#)

[PWD](#)

[DIR\\_FIRST, DIR\\_NEXT](#)

[CHDIR](#)

[MKDIR, RMDIR](#)

[DELETE](#)

[PARAM](#)

[Alacsony szintű programozás](#)

[Köszönet](#)

## Áttekintés

Ez a doksi a Videoton TV Computer számítógéphez készített egyszerű SD olvasó interfész kártya leírását tartalmazza: felhasználói és programozói szemszögből.

A dokumentum írásakor az aktuális program verziók: **NanoSD IO Arduino fw v0.32** (Nano) és **NanoSD IO Card ROM v0.33** (EEPROM).

## Bekapcsolás előtti tudnivalók

A NanoSD IO kártya microSD memória kártyát tud olvasni. A microSD kártyán az első olyan partíciót látja a NanoSD ahol FAT16 vagy FAT32 file rendszer van. Erről a partícióról a NanoSD kártya minden file -t lát, de csak 8.3 -as filenév formátumot ismer és csak a .CAS file-okat tudja betölteni és elindítani.

**Megjegyzés:** Arduino fw v0.31 esetben csak az első partíciót tudja a NanoSD kezelni.

A NanoSD IO kártyát minden esetben kikapcsolt TVC gépbe kell berakni illetve kivenni onnan.

A NanoSD IO kártya mind a három fajta TVC-vel működik. 32kB -os TVC esetben a menü csak kb 1200 file bejegyzést tud beolvasni egyszerre. 64kB-os nagyobb testvérei pedig hozzávetőlegesen 1900 -at.

A microSD memória kártya kivehető és berakható a NanoSD IO kártyán található foglalatba a TVC ki- vagy bekapcsolása nélkül, a NanoSD IO kártya a legtöbb esetben (erről később még lesz szó) észreveszi a változást.

Amennyiben a TVC és a NanoSD kártya elveszítette a szinkront, nagyon nem akarnak együttműködni, úgy resetelni kell a TVC -t és a kártyán lévő Arduino Nano modult (a rajta található kis fehér gombbal). Először a TVC -t (hard reset), majd 1-2 másodpercen belül a Nanot. De áramtalanítani is lehet a gépet (majd újra bekapcsolni).

## Felhasználói útmutató

### Bejelentkező képernyő - MENÜ

Bekapcsoláskor alapesetben a file kiválasztó MENÜ jelenik meg a képernyőn és az SD kártya gyökérkönyvtárában található file-ok és alkönyvtárak láthatóak a file-dobozban. A

file-ok és alkönyvtárak nevei a NAME oszlopban vannak. Az alkönyvtárakat a SIZE oszlopban a <DIR> szócska jelöli, a file-ok esetében itt a file-ok méretei láthatóak - byte-ban kifejezve.

**Megjegyzés:** .CAS file-ok speciális file -ok a TVC szempontjából. Ezen file-ok esetében a file hossza az UPM header nélkül jelenik meg, azaz ha PC-ről nézzük az SD kártyát a file mérete 128 byte-al nagyobb lesz, mint amit a NanoSD jelenít meg a file-ról (itt a menüben és később a CLI-ben szintén).

Amennyiben egy file mérete nagyobb, mint 64kB, úgy a 64+kB szöveg látható a file mellett.

## Navigálás

A file-dobozban a kijelölést a beépített joystickkal (esetleg joy 1) vagy az A Y és az É ; billentyűkkel mozgathatjuk.

Föl vagy le: Joy föl/le vagy **A / Y**

Lapozás előre / vissza: joy jobb/bal vagy **; / É**

A kiválasztott könyvtárba belépni a RETURN gomb lenyomásával lehet. Ekkor betöltődik a kiválasztott könyvtár tartalma. Betöltés közben a file-dobozban alul a számláló azt mutatja, hogy mennyi könyvtárbejegyzést olvasott be a TVC a kártyáról.

A kiválasztott könyvtárat a NanoSD IO kártya memóriában kikapcsolásig, vagy az Arduino Nano reseteléséig megőrzi.

## Program betöltése

Amennyiben egy .CAS kiterjesztésű file-t választunk ki és arra ütünk RETURN-t, a kártya visszaadja a vezérlést a TVC-nek, ami folytatja az INIT rutint és a boot képernyő eltűnése után kiadja a LOAD parancsot a kiválasztott file -ra.


**Megjegyzés:** Ebbe a folyamatba bekavarhat egy másik IO kártya, ami a NanoSD kártyától balra helyezkedik el (magasabb számú slotban van, ezért annak az INIT rutinja a NanoSD kártya INIT rutinja *után* hívódik meg).

**Megjegyzés #2:** Betöltésre nem minden .CAS file indul el. Ez azon múlik, hogy a betöltött file-ban egy bizonyos AUTORUN flag be van-e állítva, vagy sem. Ha nincs, akkor a betöltés után a BASIC prompt-ot kapjuk s magunknak kell a RUN parancsot begépelni.

## Kilépés, BASIC

Amennyiben úgy gondoljuk, hogy a megjelenített MENÜben nincs semmi dolgunk az ESC gomb lenyomásával kiléphetünk belőle s ilyenkor a TVC boot folyamata folytatódik a normál INIT rutinokkal s végül eljutunk a TVC bekapcsolási képernyőjéhez.





BASICben a LOAD parancs használatával az éppen kiválasztott könyvtárból CAS formátumú file-okat be tudjuk tölteni, BASIC programunkat pedig a SAVE paranccsal ki tudjuk menteni és - egy később ismét betölthető - CAS file keletkezik belőle, a megadott névvel.

A NanoSD IO kártya a MENÜben (vagy a CLI-ben - erről később) kiválasztott könyvtárat kb egy kazettának láttatja a TVC-vel: minden kazetta művelet a kártyára irányítódik át, így működnek BASICből az OPEN #5, CLOSE, GET #5 és INPUT #5 parancsok is.

## További lehetőségek

### Rendezés

Lehetőség van arra, hogy a file-dobozban a file-ok rendezve íródjanak ki. A file-okat lehet név (angol ABC..) vagy méret szerint rendezni. Rendezett lista esetén az alkönyvtárak mindig a lista elején és név szerint rendezve látszódnak. Akár név, akár méret szerint rendezzük a file-okat, az mindig csak növekvő sorrendben történik.

Ha meguntuk a rendezettséget, akkor megtehetjük újra, hogy a file-ok az SD kártyán történő elhelyezkedésük (kb létrehozásuk időrendje) szerint, eredeti sorrendben kerüljenek kiírásra.

Sorbarendezni az **N** (név) vagy az **S** (méret) betű lenyomásával lehet, az eredeti sorrendet visszahozni pedig az **O** (nem nulla! - 'original') lenyomásával tudjuk megtenni.

A sorbarendeázés állapota megőrződik kikapcsolt TVC esetében is és globális beállítás, azaz kiválasztás után az összes könyvtárra vonatkozik. A kiválasztott rendezést egy kis kalap jelzi a NAME vagy a SIZE szavak előtt, a filedoboz fejlécében.

**Megjegyzés:** nagyobb könyvtárak esetében a rendezés elhúzódhat, több másodpercig (mérettől függően akár percekig) is eltarthat!

### Bekapcsolás

A NanoSD kártya bekapcsoláskor a MENÜ-be lép, de ezen változtathatunk. Ha a MENÜben a SPACE gombot lenyomjuk, úgy a BOOT beállítását átállíthatjuk arra, hogy a legközelebbi bekapcsoláskor (hideg resetkor) csak egy rövid szöveges üzenet jelenik meg pár másodpercre, majd a folytatódik a normális betöltés folyamata.

Amennyiben ilyenkor megnyomjuk az ESC billentyűt, a TVC belép a NanoSD menüjébe (de a BOOT beállítás ettől még nem változik). Figyeljük meg, hogy ilyenkor a NanoSD IO kártya felhívja a figyelmünket erre a lehetőségre!

Ez a beállítás is megőrződik kikapcsolások között.

## SD kártya

A NanoSD IO kártya érzékeli, ha kiveszed, vagy beraksz egy microSD kártyát a foglalatból/-ba. Kivételkor nullázza a belső változóit, berakáskor pedig inicializálja az SD kártyát és onnantól újra el lehet érni a kártyán található file-okat.

A MENÜben a TVC úgy szerez a változásról tudomást, hogy kb 3 másodpercenként lekérdezi a Nanotól az SD kártya állapotát. Amennyiben változás van (eddig volt kártya, most nincs vagy fordítva) úgy az új (jelenlegi) állapotnak megfelelően elrendezi a képernyőt.

**Megjegyzés:** SD kártya hiánya esetén a 'PWD failed' üzenet jelenik meg, ami azért történik, mert a TVC nem tudja lekérdezni az aktuális könyvtár tartalmát. Nem túl elegáns, tudom.

Sajnos előfordul néha, hogy a TVC nem veszi észre, hogy újra van SD kártya a foglalatban. Nem tudom pontosan az okát, de arra gyanakszom, hogy ez amiatt lehet, hogy a TVC éppen akkor kérdezi le az SD kártya állapotát a Nanotól, amikor az az SD kártyát inicializálja. Ilyenkor összeakadhat valami.

Szerencsére nálam nem túl gyakran fordul elő, a MENÜ-n kívül pedig nem tapasztaltam még ilyen hibás viselkedést (hiszen ott nincs semmi, ami folyamatosan kérdezzetné a NanoSD IO kártyát ...).

## CLI - Parancssori interfész

Nem csak a MENÜben tudunk könyvtárak között navigálni, mivel a NanoSD IO kártya lehetőséget biztosít egy ún. CLI -re (Command Line Interface), ahol különböző DOS jellegű file műveleteket tudunk végrehajtani.


CLI-be az EXT 0 (ez itt nulla!) BASIC parancssal léphetünk be. Itt rögtön egy rövid fejléc közli is velünk, hogy mi a NanoSD kártya ROM programjának a verziója és hogy a HELP parancssal tudunk további információkat szerezni a többi elérhető parancsról.

Minden sorba egy parancsot írhatunk és a beírt parancsot a RETURN gomb lenyomásával indítunk el. Scriptek, több parancs egymás utáni automatikus végrehajtására nincs lehetőségünk.

A végrehajtható parancsok listája:

DIR [/W], EXIT, HELP, MKDIR, RMDIR, DEL, CD, PWD, VER, CLS

Az EXIT, a HELP és a PWD parancsokat tudjuk rövidíteni is rendre az X, a H és a CD parancsokkal. Sőt, ha simán csak ESC-et nyomunk, akkor az pont olyan, mintha beírtuk volna azt, hogy EXIT és nyomunk utána egy RETURN -t. Bámulatos, hol tart ma a tudomány.

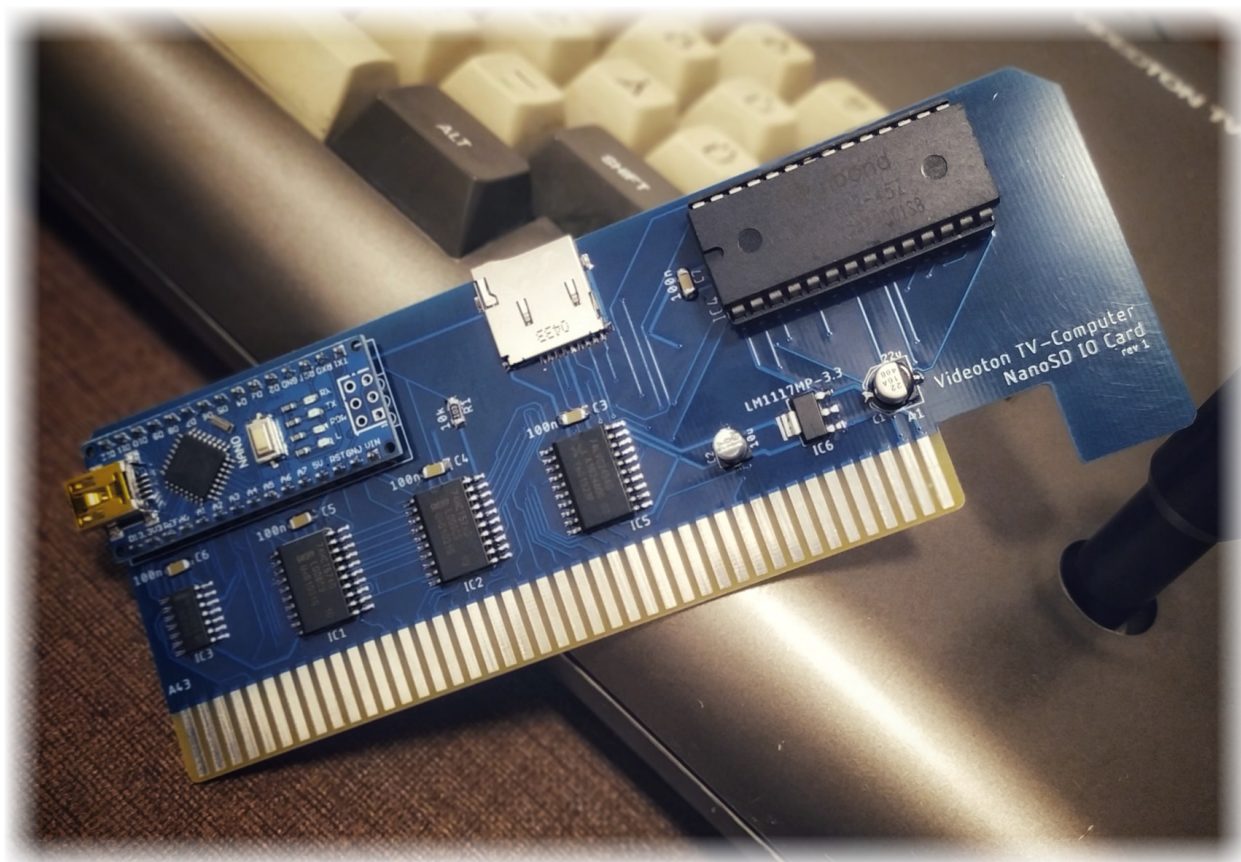


Az egyes parancsok jelentéseit a nyájas olvasó fantáziára bízám, olyan nagy meglepetéssel gondolom nem fognak szolgálni.

Viszont. A DIR és DIR /W parancsok futtatása során van még pár említésre méltó trükk:

- ha a CTRL billentyűt nyomva tartjuk, akkor lassul a könyvtár tartalmának a kiírása
- amennyiben a CTRL-ESC kombinációt lenyomjuk, úgy megszakítjuk az aktuális könyvtár listázását
- A CTRL-P kombináció hagyományosan működik (megáll a listázás)

Érdeemes a CLI -t kétszínű grafikai módban futtatni, úgy több karakter fér a képernyőre, áttekinthetőbb a képernyőkép.



## Programozási segédlet

A NanoSD IO Card egy furcsa szerzet. Kicsit diszk kezelő (a file-okat könyvtár struktúrába rendezetten név szerint tudja kezelni, betölteni - akár kimenteni is), de kicsit kazetta emulátor (BASIC-ből a kazetta kezelő parancsok elérhetőek rá). Hogyan is lehet ezt használni programból, milyen extra szolgáltatást nyújt a TVC-nek? Erről szeretnék itt írni.

**Megjegyzés:** ez a leírás a jelenlegi legfrissebb sw verzió alapján készült. Időközben kerülhetnek be újabb funkciók (jobb esetben), vagy változhattak meg jelenlegiek (rosszabb esetben).

## Eszközosztály

A NanoSD IO kártya inicializálódása során önmagát a magnó eszközosztály IO funkcióira fűzi be. Ez azt jelenti, hogy induláskor a 0B05h és a 0B0Dh címeken található mutatókat önmagára állítja. Ez a beállítás még warm resetkor sem tűnik el. Bármilyen magnóra

vonatkozó IO művelet innentől a NanoSD kártyán fog landolni. Ezek a műveletek (függvény hívások) a következők:

Gyári függvények:

- CAS\_CHIN (byte beolvasása)
- CAS\_CHOUT (byte kiírása)
- CAS\_BKIN (blokk beolvasása)
- CAS\_BKOUT (blokk kiírása)
- CAS\_OPEN (file megnyitása olvasásra)
- CAS\_CRTE (file megnyitása írásra - NanoSD: korábbi tartalom törlődik)
- CAS\_CLOSE (file lezárása)
- CAS\_VERIFY (file tartalom ellenőrzése)

NanoSD saját függvények:

- PWD (lekérdezi az aktuális könyvtárat)
- DIR\_FIRST, DIR\_NEXT (könyvtár tartalom lekérdezése)
- CHDIR (könyvtár váltás)
- MKDIR, RMDIR (könyvtár létrehozás, törlés)
- DELETE (file törlése)
- PARAM (paraméter lekérdezése, beállítása)

A gyári függvényekről sem sok szót ejtenék, azok úgy működnek, ahogyan definiálva vannak (ld. Operációs rendszer TV-Computer szakkönyv). Igyekeztem a visszatérési hibakódokat is azonosan beállítani, remélem, hogy nem sokat hibáztam...

A NanoSD függvényekről azt kell tudni, hogy mindegyik blokkoló fgv, azaz akkor tér vissza, ha van eredmény (ilyen, vagy olyan). Az összes fgv az IO ROM-ban helyezkedik el, azaz U3 RAM területeket nem tudnak használni a függvények.

## Keretcsíkozás

Töltéskor alapesetben a képernyőkeret csíkozódik. Egy byte betöltése során 2-vel nő, a puffer feltöltésekor pedig 2-vel csökken a keretszín.

**Megjegyzés:** az Arduino Nano egy 64 byte-os puffert tart fent magában s file betöltéskor azokat a byte-okat egyenként adja át a TVC -nek. A puffer feltöltése általában tényleges IO művelettel jár a Nano és az SD kártya között, szóval az szemmel láthatóan tovább tart - vastagabb a csík -, mint 1 byte transzferálása a TVC-re. Általában, mivel az Arduino SD library viszont egy 256byte-os buffert tart fent magában (csak arra kintről nincs semilyen rálátás) s amikor ebből a bufferből tölti fel a Nano a 64 byte-os buffert, akkor keskenyebb az a csík (de ekkor sem 1 pixel magas).



Ez a csíkozás kikapcsolható a v0.33 -as EEPROM verziótól kezdve, mégpedig úgy, hogy a BORDER (0B4Fh) keretszín regiszter 0. bitjét 1-re állítjuk. Látható, hogy ez a keret színét nem befolyásolja (a keret színét a b1, b3 b5 és b7 bitek állapota határozza meg). Ha valaki programból akar utántölteni, ott esetleg kikapcsolható a keretszínezés.

De LOAD parancs előtt beírva a

```
POKE 2895, PEEK(2895) OR 1
```

parancsot, ugyanúgy kiiktathatjuk a betöltés közbeni keretcsíkozás (s visszakapcsolható ha AND 254 -et használunk).

## NanoSD függvények

### PWD

Lekérdezi az aktuális munkakönyvtárat

Kód:	56h, D6h	
Input	DE	egy legalább 65 byte-os területre mutató pointer, ahova a fgv bemásolhatja a jelenlegi könyvtárat, C stringként, azaz 00h-val lezárva.
Output:	A	00h: OK EDh: Error

### DIR\_FIRST, DIR\_NEXT

Az aktuális könyvtár tartalmának lekérdezése. Ha egy könyvtár tartalmát ki akarjuk listázni, akkor az első entry lekérdezéséhez a DIR\_FIRST fgv -t kell hívunk, majd a DIR\_NEXT -et az összes többihez. Input paramétereik és visszatérési értékeik megegyeznek.

A DIR\_NEXT fgv-t addig kell hívni, míg az A visszatérési értéke 01h lesz vagy az utolsó DIR\_NEXT fgv-t az 58h kóddal hívtuk meg (a D8h helyett)

#### DIR\_FIRST:

Kód:	57h, D7h	
Input	DE	egy 17 byte hosszú területre mutató pointer, ahol a DIR ENTRY ki lesz töltve
Output:	A	00h: OK, DIR ENTRY kitöltve 01h: vége a listának, nincs dir entry 80h+: Error

**DIR\_NEXT:**

Kód:	D8h	Amennyiben 58h kóddal hívjuk meg úgy lezárjuk a directory lista letöltést. Ilyenkor nincs input paraméter és a visszatérési érték mindig 00h lesz
Input	DE	egy 17 byte hosszú területre mutató pointer, ahol a DIR ENTRY ki lesz töltve
Output:	A	00h: OK, DIR ENTRY kitöltve 01h: vége a listának, nincs dir entry 80h+: Error

**DIR ENTRY** struktúra:

Összesen és fixen 17 byte hosszú.

0-11: filenév 8.3 formátumban, jobbról SPACE (20h) karakterekkel kiegészítve. (12 byte)

12: file típusa. 00h esetben directory, 01h esetben file (1 byte)

13-16: file hossza, LSB formában. Directory esetén 00000000h (4 byte)

**CHDIR**

Az aktuális könyvtár váltása. A paraméter lehet abszolút (független a jelenlegi könyvtártól), ilyenkor egy / jellel kezdődik a paraméter, s lehet relatív (a jelenlegi munkakönyvtárhoz viszonyított), ilyenkor . / -val, vagy csak simán egy létező könyvtárnévvel kezdődik a paraméter. A paraméteren belül a / . / előfordulhat, azt a NanoSD feloldja. A paraméternek nem kell / -re végződnie, de nem rossz gyakorlat..

Kód:	59h, D9h	
Input	DE	C string-re mutató pointer, ahol a létrehozandó könyvtár neve található (abszolút vagy relatív eléréssel)
Output:	A	00h: OK F1h: Nem lehet beváltani a megadott könyvtárba F2h: Idő-túllépés (nincs kártya?)

## MKDIR, RMDIR

Könyvtár létrehozása, illetve törlése. Vegyük figyelembe, hogy ezek a függvényhívások esetenként hosszú ideig (másodpercekig) is tarthatnak.

### MKDIR

Kód:	5Ah, DAh	
Input	DE	C string-re mutató pointer, ahol a létrehozandó könyvtár neve található (abszolút vagy relatív eléréssel)
Output:	A	00h: OK F1h: Nem lehet létrehozni a könyvtárat F2h: Idő-túllépés (nincs kártya?)

### RMDIR

Kód:	5Bh, DBh	
Input	DE	C string-re mutató pointer, ahol a törlendő könyvtár neve található (abszolút vagy relatív eléréssel)
Output:	A	00h: OK F1h: Nem sikerült letörölni a könyvtárat (nem üres, nem létezik?) F2h: Idő-túllépés (nincs kártya?)

## DELETE

File törlése az SD kártyáról.

Kód:	5Ch, DCh	
Input	DE	C string-re mutató pointer, ami a letörlendő file nevét tartalmazza (abszolút vagy relatív eléréssel)

Output:	A	00h: OK F1h: Nem sikerült letörölni a file -t (nem létezik?) F2h: Idő-túllépés (nincs kártya?)
---------	---	--

## PARAM

Lehetőség van 4 előre definiált paraméter lekérdezésére és ezek közül kettő beállítására is. A paraméterekre kódjukkal hivatkozunk.

Paraméterek és értékeinek típusa:

Kód	Paraméter leírása	Típusa	Lehetséges értékek
0	Boot menü állapota, azaz induláskor legyen-e a MENÜ megjelenítve vagy sem.	BYTE (0)	0 - nincs MENÜ 1 - van MENÜ
1	Rendezés állapota, azaz milyen rendezettséggel jelenjenek meg a file-ok a MENÜ file-dobozában.	BYTE (0)	0 - Nincs rendezés 1 - ABC szerint 2 - Fileméret szerint
2	Van-e inicializált (felismert) SD kártya a foglalatban. Ez a paraméter csak lekérdezhető.	BYTE (0)	0 - Nincs 1 - Van
3	A NanoSD kártyán található Arduino Nano -ban található program (Arduino FW) verziója. Ez a paraméter csak lekérdezhető.	STRING (2)	C típusú (00h -val terminált) string, amire DE mutat

Kód:	DEh	paraméter lekérdezése
Input	C	A lekérdezendő paraméter kódját (0, 1, 2, 3) tartalmazza
	DE	Egy előre lefoglalt területre mutató pointert tartalmaz, ahova a lekérdezendő string értéke másolódik be. Nem String típusú paraméterek esetében nincs jelentősége.
Output:	A	00h: OK. 01h: Nincs ilyen paraméter 02h: Bármilyen más hiba esetén
	DE	Csak sikeres lekérdezés esetén értelmezett.


		BYTE típusú változó esetén: E tartalmazza a lekérdezett értéket WORD típusú változó esetén: DE tartalmazza a lekérdezett értéket (jelenleg nincs ilyen paraméter) STRING típusú változó esetén: DE nem változik a bemeneti paraméterhez képest, a lekérdezett paraméter értékére mutat
--	--	--

Kód:	5Eh	paraméter beállítása (jelenleg csak a 0 és az 1-es paramétereknél - és csak BYTE típusra - lehetséges)
Input	C	A beállítandó paraméter kódját (0, 1) tartalmazza
	DE	<ul style="list-style-type: none"> <li>- E a byte típusú paraméterek esetében az új értéket tartalmazza</li> <li>- DE a word típusú paraméterek esetében az új értéket tartalmazza (fogja tartalmazni, ha le lesz implementálva)</li> <li>- DE a string típusú paraméterek esetén egy pointert tartalmaz a null-terminált sztringre (fog tartalmazni, ha le lesz implementálva)</li> </ul>
Output:	A	00h: OK. 02h: Nem sikerült beállítani a paramétert (nem jó ID, stb..)
	DE	Csak sikeres lekérdezés esetén értelmezett. BYTE típusú változó esetén: E tartalmazza a lekérdezett értéket WORD típusú változó esetén: DE tartalmazza a lekérdezett értéket (jelenleg nincs ilyen paraméter) STRING típusú változó esetén: DE nem változik a bemeneti paraméterhez képest, a lekérdezett paraméter értékére mutat

## Alacsony szintű programozás

A NanoSD IO kártya egy egyszerű szerzet (és furcsa is, ahogy azt már korábban említettem). Van rajta egy nem túl erős mikrokontroller (no, nem a TVC-hez képest értendően), van rajta 2db regiszter IC (egy 'bemeneti' és egy 'kimeneti'), egy feszültség-illesztő IC az SD kártya miatt és egy quad logikai NOR kapu. Ez utóbbi dolga, hogy gyorsan és egyszerűen előállítsa a Nano-nak azt a jelet, amikor a TVC a kártyára OUT vagy IN hívást intéz: ennek hatására kerül bele a regiszter IC-kbe a megfelelő adat.





A Nano SD kártya nem elég erős ahhoz, hogy a címeket is feldolgozza, ezért az adott TVC kártyafoglalathoz tartozó 16 széles címtartomány bármelyikére írhatunk-olvashatunk, azt a NanoSD IO kártya minden esetben érzékelni fogja.

**Megjegyzés:** Érdemes a kártya legelső címét használni (10h, 20h, 30h és 40h), hogy egy esetleges későbbi fejlesztés során ne legyenek fennakadások...

A kártyával való kommunikációt mindig a TVC kezdeményezi egy OUT paranccsal, majd a kiválasztott függvénytől függően folytatódhat a párbeszéd a TVC és a NanoSD IO kártya között.

Vájt szemű (kód-)olvasók megfigyelhették a forrásban, hogy minden OUT parancs után található egy IN, amelynek értékét nem használjuk fel. Ez azért van, mert az IN utasítás hatására az egyik (a 'kimeneti') regiszter tartalma jelenik meg az adatbuszon és ugyanezen utasítás hatására kerül bele (kicsivel az utasítás végrehajtása után) a valódi érték, amit a NanoSD küld a TVC-nek. A lényeg, hogy az első IN parancs eredményét még nem tudjuk felhasználni, az még biztosan nem a mi OUT parancsunk hatására keletkezett.

Ebben a fejezetben nem szándékom részletesen ismertetni a NanoSD IO kártya által nyújtott alacsony szintű függvényeket. Azok teljes működésének és helyes paraméterezéséhez mindenképp a Nano forráskódjának alapos ismerete (ahhoz pedig a kód tanulmányozása) szükséges. Nem kívánom ettől az élvezettől megfosztani a kedves olvasót.

A jelenleg támogatott alacsony szintű függvények egyébként a következők:

**SDISK\_NANO\_CMD\_OPEN\_RO (0x00)**

File megnyitása olvasásra. CAS file esetben az UPM header -t eltávolítjuk és a GETDATA csak a 'hasznos' adatot adja vissza!

**SDISK\_NANO\_CMD\_CLOSE\_RO (0x01)**

Olasásra megnyitott file lezárása

**SDISK\_NANO\_CMD\_CHDIR (0x02)**

Aktuális könyvtár váltása. Tudni kell, hogy a felhasznált SD library nem állapottartó, azaz az aktuális munkakönyvtárat a NanoSD firmware tartja számon.

**SDISK\_NANO\_CMD\_LIST (0x03)**

Könyvtár listázás. Ez a parancs csak elkezdi, a folyamatos listázáshoz a megfelelő ACK beküldése is szükséges.

**SDISK\_NANO\_CMD\_ACKLIST (0x04)**



Könyvtár listázás során a bejegyzést beolvasta a TVC, jöhet a következő bejegyzés.

**SDISK\_NANO\_CMD\_GETDATA (0x05)**

Korábban olvasásra megnyitott file-ból adat beolvasás.

**SDISK\_NANO\_CMD\_GETCDIR (0x07)**

Aktuális munkakönyvtár lekérdezése

**SDISK\_NANO\_CMD\_BANKTO0 (0x08)**

Memória bank váltása a 0. bankra (NEM HASZNÁLT, NEM TESZTELTI). A v0.32 -es ROM verzió elfér 8kB-os bankban, ezért nincs szükség BANK váltásra. Ez az utasítás az A13 -as címvonal értékét változtatja az EEPROM -on.

**SDISK\_NANO\_CMD\_BANKTO1 (0x09)**

Memória bank váltása az 1. bankra (NEM HASZNÁLT, NEM TESZTELTI)

**SDISK\_NANO\_CMD\_CREATE (0x0A)**

File létrehozása írásra.

**SDISK\_NANO\_CMD\_PUTDATA (0x0B)**

Írásra megnyitott file-ba adatot küld.

**SDISK\_NANO\_CMD\_CLOSEWRITE (0x0C)**

Írásra megnyitott file lezárása.

**SDISK\_NANO\_CMD\_GETPARAMETER (0x0D)**

Egy rendszer-paraméter lekérdezése.

**SDISK\_NANO\_CMD\_SETPARAMETER (0x0E)**

Egy rendszer-paraméter beállítása.

**SDISK\_NANO\_CMD\_MKDIR (0x0F)**

Könyvtár létrehozása.

**SDISK\_NANO\_CMD\_RMDIR (0x10)**

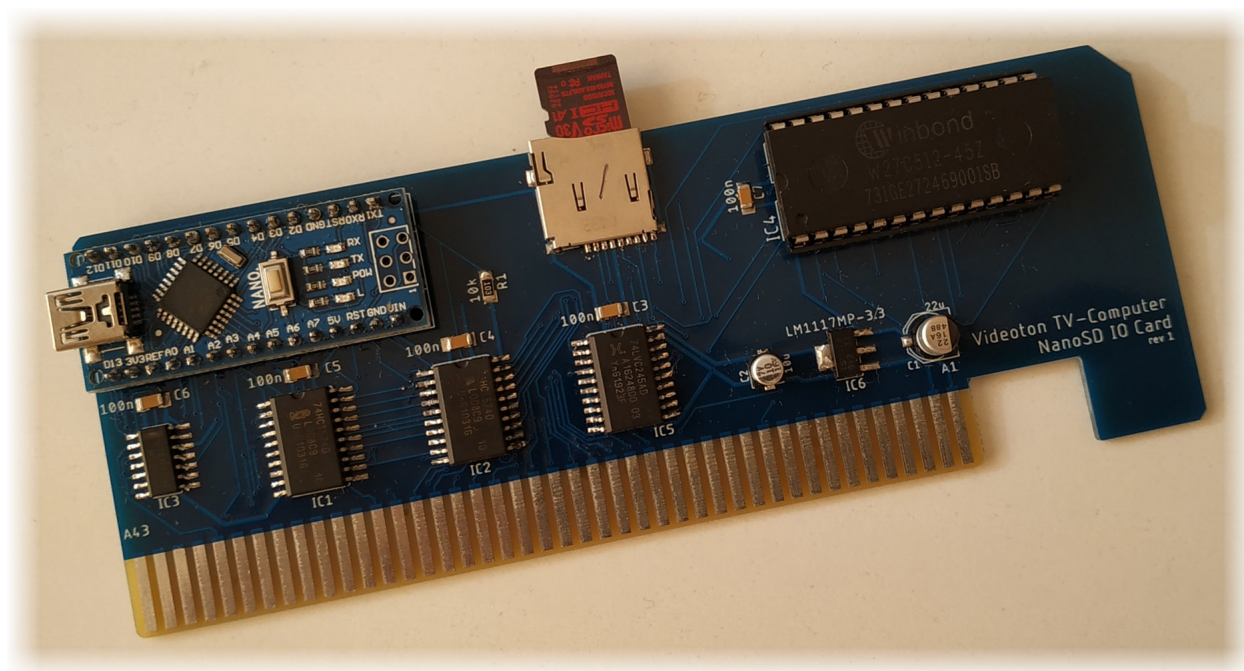
Könyvtár törlése.

**SDISK\_NANO\_CMD\_DELETE (0x11)**

File törlése.



**Megjegyzés:** A NanoSD forrásprogramjának a kommentjei nem feltétlenül a legutóbbi állapotot tükrözik. Mindenképp javaslom a forrásprogram tanulmányozását és a kommentek csak amolyan útmutatóként való felhasználását...



## Köszönet

Hálás köszönet illeti türelmes feleségemet, aki elviseli, hogy a családom mellett ilyesmivel töltsem a drága időmet.

Köszönöm Árvai Lászlónak, Hanák Miklósnak és Major Tamásnak (ABC rendben!) a hasznos tanácsokat, inspirációt, ötleteket amiket a kártya fejlesztése során kaptam tőlük.

És nem utolsó sorban köszönöm a rendes képeket Lakatos Attila és Sáránszki Mihály (ugyancsak ABC rendben) TVC Facebook csoporttársaknak, amiket a doksiban felhasználhatok jóvoltukból!

Jól néz ki ez a kártya, ugye? :)