

# Programozható LED-fűzőren alapuló reklámpanel - LED fűzőr vezérlése, adatok kiírása

Patka Zsolt-András | Számítástechnika BSc


2019.11.11

## 1. Bevezető


A projekt célja egy LED-fűzőr vezérlése és ennek segítségével egy reklámszöveg megjelenítése. Ehhez egy FPGA lap és egy Worldsemi WS2813 ledfűzőr lesz felhasználva.

## 2. Követelmények

### 2.1. Funkcionális követelmények

- Lehetséges legyen egy reklámszöveget kiírni a ledfűzőrek által létrehozott mátrix-ra.
- Egy sorban 100 LED található 
- Az egyes ledfűzőrek szinkronba működjenek
- A kiírás párhuzamosan történik az adott ledfűzőreken
- Másodpercenként 60 frissítés (60 Hz)
- Mozdó szöveg

### 2.2. Nem funkcionális követelmények

- Benti használatra van tervezve
- Ha az FPGA-lapnak lesz készítve külön tokozat, akkor IP31-es standardnak kell megfeleljen
- Ha az FPGA-lapnak nem lesz készítve külön tokozat, akkor nem felel meg IP standardnak (IP00 )

### 2.3. Fejlesztési követelmények

- Implementáció VHDL nyelvben
- Szimulációs állomány a rendszer tesztelésére
- Adatok kiírása lehetséges a LED fűzőrre
- Modularitás
  - Külön modul egy 24 bit-es blokk küldésére
  - Külön modul a 24 bit-es blokkokat küldő modulok használatára (paraméterezhető, bővíthető)
- Opcionális:
  - Pár betű kódolása (3-4)
  - Betűk tárolása BRAM memóriában

## 3. Rendszer-specifikáció

- WS2813 egyszálú adatátvitel protokolljának helyes használata
- Worldsemi WS2813 90 LED-es ledfűzőr van felhasználva
- Digilent Basys3 FPGA vezérli a ledfűzőrt

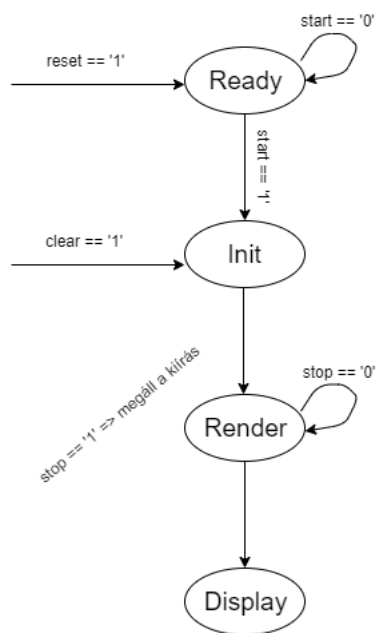


## 4. Állapotok

2019.10.14

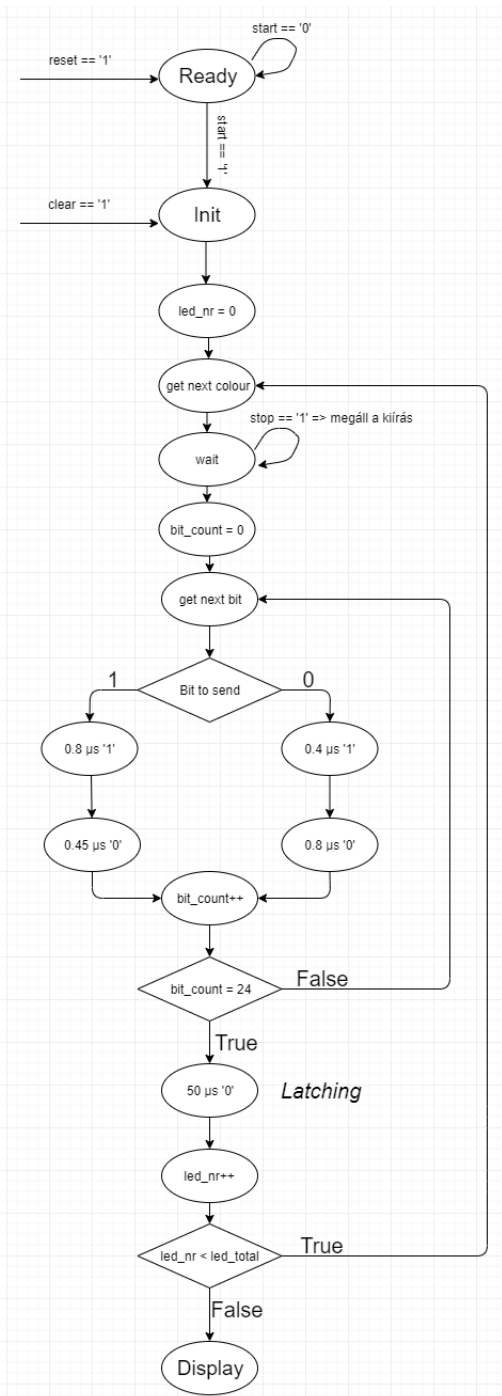
Állapotok:

- READY
  - Alap állapot
  - "reset" jel esetén ide kerül vissza az automata
- INIT
  - minden LED-et kikapcsol (0x000000-t ír)
  - "clear" jel esetén ide kerül az automata
- RENDER
  - egyenként küldi a szín információt a LED-ekre
  - annyiszor végződik el itt a művelet, ahány LED-ünk van
  - "stop" jel esetén megáll a kiírás
- DISPLAY
- megtörtént a kiírás

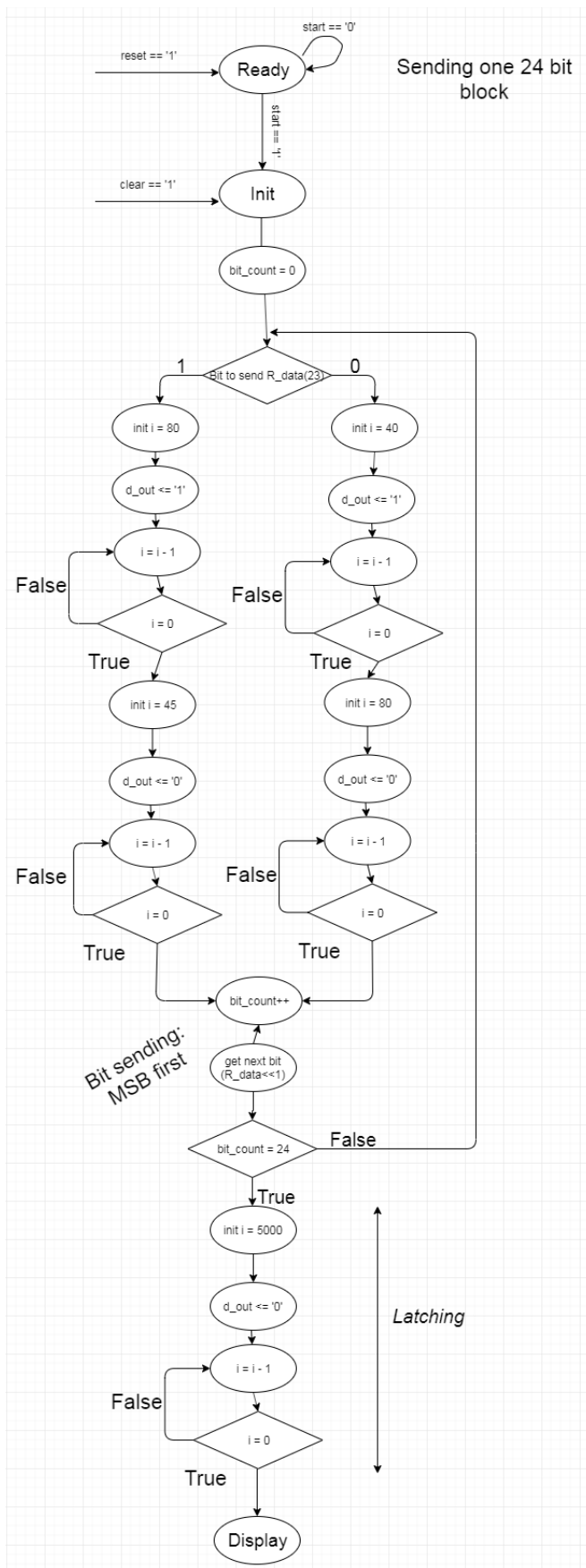


2019.10.28

Állapotdiagram átírva úgy, hogy a küldési logikát is tartalmazza:

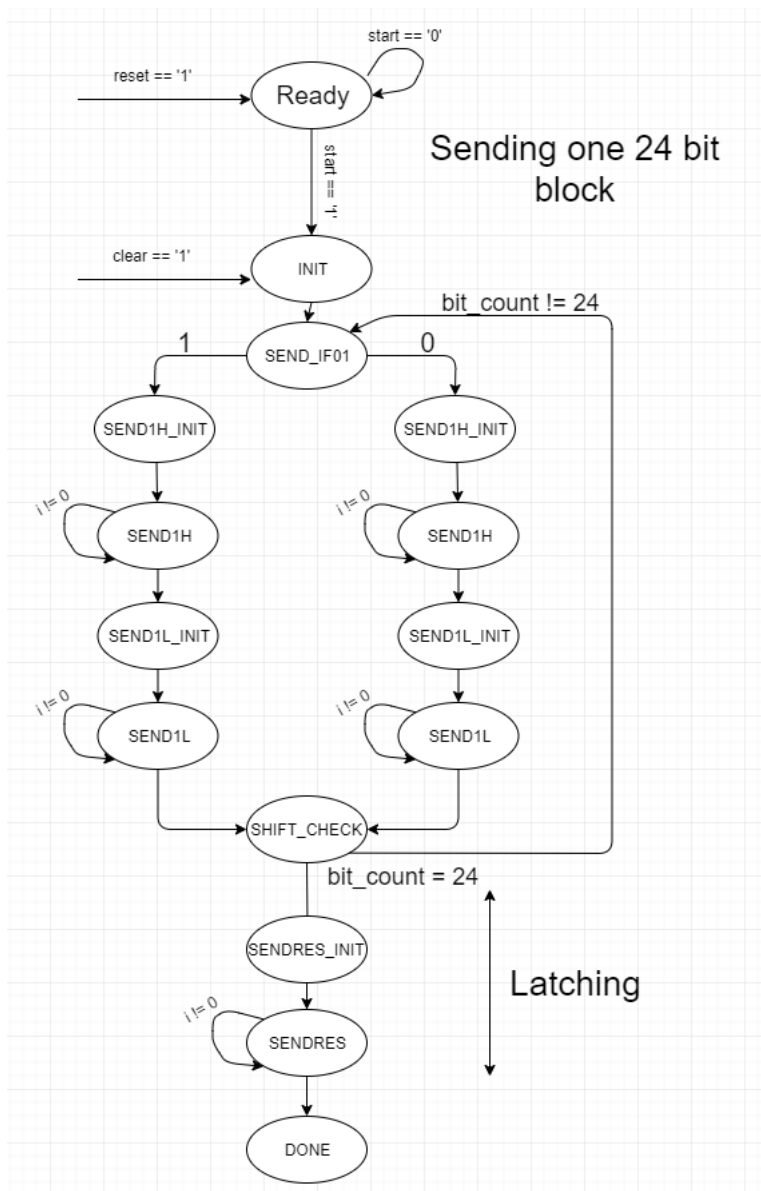


Állapotdiagram egy 100 LED-et vezérlő modulra



2019.11.11

Végleges állapotdiagram, az implementációban is használt.



TODO: Dokumentáció befejezése!

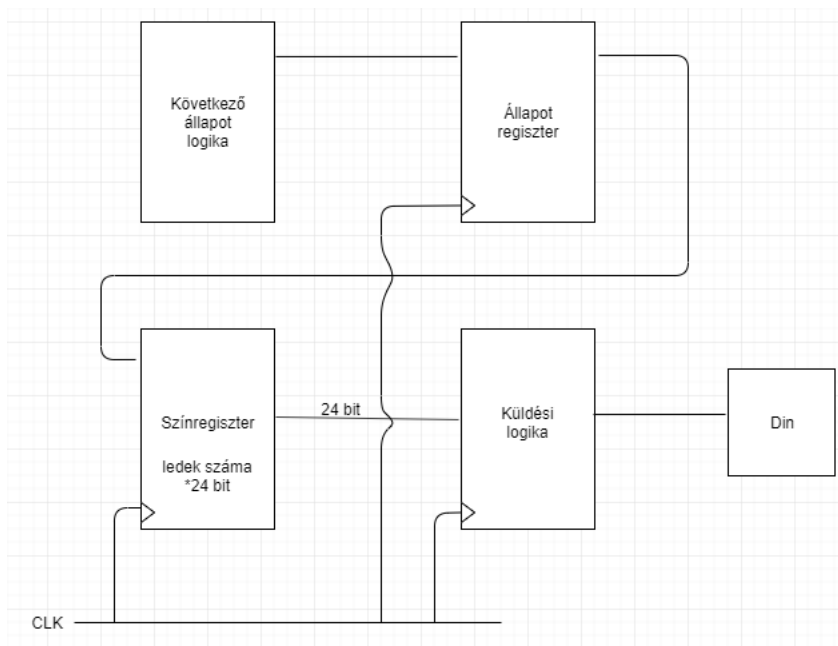
- **READY**
  - Alap állapot
  - "reset" jel esetén ide kerül vissza az automata
- **INIT**
  - inicializálja a bit-count-ot nullára
- **SENDIF\_01**
  - Megvizsgálja data[23]-as bit-et. Ha 1, akkor a következő állapot a **SEND1H\_INIT**, ha 0 akkor **SEND0H\_INIT**
- **SEND1H\_INIT**
  - Inicializálja az **i** változót a T1H értékre
  - A d\_out output-ot 1-re állítja
- **SEND1H**
  - Dekrementálja az **i** változót
  - Amikor az **i** változó nulla lesz, akkor tovább megy a **SEND1L\_INIT** állapotra
- **SEND1L\_INIT**

- Inicializálja az **i** változót a T1L értékre
- A **d\_out** output-ot 0-ra állítja
- **SEND1L**
  - Dekrementálja az **i** változót
  - Amikor az **i** változó nulla lesz, akkor tovább megy a **SHIFT\_CHECK** állapotra
- **SEND0H\_INIT**
  - Inicializálja az **i** változót a T0H értékre
  - A **d\_out** output-ot 1-re állítja
- **SEND0H**
  - Dekrementálja az **i** változót
  - Amikor az **i** változó nulla lesz, akkor tovább megy a **SEND0L\_INIT** állapotra
- **SEND0L\_INIT**
  - Inicializálja az **i** változót a T0L értékre
  - A **d\_out** output-ot 0-ra állítja
- **SEND0L**
  - Dekrementálja az **i** változót
  - Amikor az **i** változó nulla lesz, akkor tovább megy a **SHIFT\_CHECK** állapotra
- **SHIFT\_CHECK**
  - Megnézi, hogy a **bit\_count** változó 24-e, ha igen, akkor tovább megy a **SENDRES\_INIT** állapotra
  - Ha a **bit\_count** változó nem egyenlő 24-el, akkor shift-eli a **data** std logic vectort balra eggyel; vissza megy a **SEND\_IF01** állapotra
- **SENDRES\_INIT**
  - Inicializálja az **i** változót a TRES értékre
  - A **d\_out** output-ot 0-ra állítja
- **SENDRES**
  - Dekrementálja az **i** változót
  - Amikor az **i** változó nulla lesz, akkor tovább megy a **SEND\_DONE** állapotra
- **SEND\_DONE**
  - Befejeződött a 24 bit-es blokk kiírása
  - Beállítja a **done** kimenetet 1-esre

## 5. Modulok

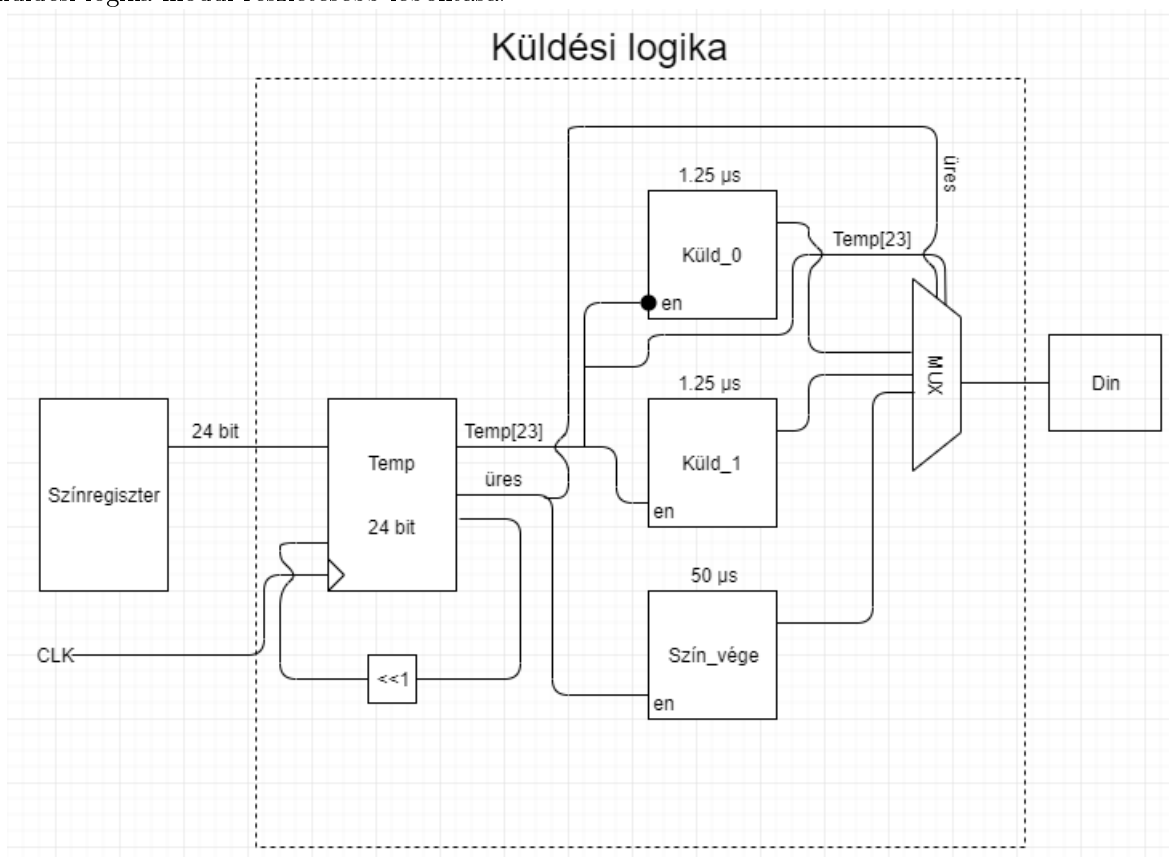
2019.10.14

- Következő állapot regiszter *Next State Register*
- Állapot regiszter *State Register*
- Szín regiszter *Colour Register*
- Küldési logika regiszter *Transmission Logic Register*



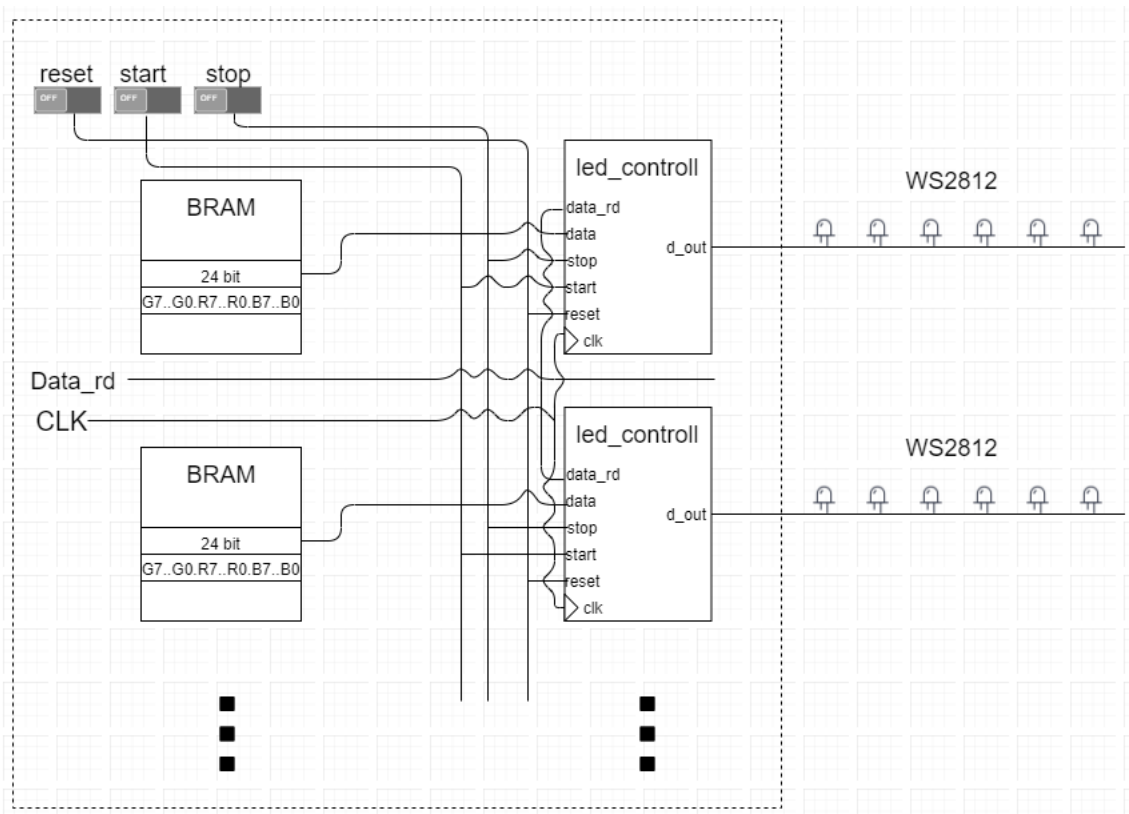
### 5.1. Küldési logika regiszter

A küldési logika modul részletesebb lebontása:



**2019.10.28**

Minden **led\_controll** modulhoz tartozik egy BRAM blokk és minden ilyen modul egy ledfűzért vezérel meg. Öt ilyen blokk megvezérel öt ledfűzért, ezáltal létrehozva a ledmátrixot. Az órajel, start, reset, stop és data-rd jelek közősek minden modulnak.



## 6. WS2813 egyszálú adatátvitel protokoll leírása

A LED-eket vezérlő áramkörök egymás után vannak bekötve úgy, hogy az egyik áramkörnek az adatkimenete a következő áramkörnek az adatbemenetét képi. Egyszálú az adatátvitel, fontos a protokoll betartása, ahhoz, hogy adatokat tudjunk megjeleníteni a LED-fűzéken.

Amikor egy áramkör megkap egy 24 bit-es kódot, akkor ezt addig tárolja amíg más kódot nem kap, vagy a tápforrást el nem veszti.

### 6.1. A 24 bit-es kód

A 24 bit-es kód a következőképpen kell kinézzen:

8 bit GREEN | 8 bit RED | 8 bit BLUE

Az adatátvitel a következő sorrendben kell történnjen:

1. GREEN
2. RED
3. BLUE

### 6.2. Bit-ek küldési sorrendje

**Az egyes byte-ok küldését úgy kell elvégezni, hogy az MSB-vel kell kezdeni és haladni az LSB fele.**

24 bit-es kód részletesebb felbontása:

- *G7 G6 G5 G4 G3 G2 G1 G0 | R7 R6 R5 R4 R3 R2 R1 R0 | B7 B6 B5 B4 B3 B2 B1 B0*

A küldés a következő sorrendben kell elvégeződjön:

- **G7 G6 G5 G4 G3 G2 G1 G0 | R7 R6 R5 R4 R3 R2 R1 R0 | B7 B6 B5 B4 B3 B2 B1 B0**



### 6.3. Időzítések

Minden 24 bit-es adatátvitel után kell legalább 50  $\mu$ s-ot várakozni, alacsony feszültségen. Ez jelzi azt, hogy egy 24 bit-es blokk továbbítása megtörtént.

Az egyes bit-ek átvitele a következőképp történik:

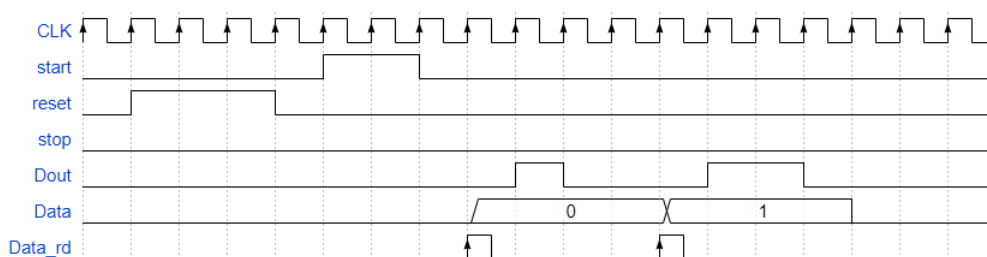
- Logikai 1-es
  - 0.8  $\mu$ s-ot magas feszültségen
  - 0.45  $\mu$ s-ot alacson feszültségen
- Logikai 0-ás
  - 0.4  $\mu$ s-ot magas feszültségen
  - 0.85  $\mu$ s-ot alacson feszültségen
- 24 bit-es adatblokk küldése után:
  - > 50  $\mu$ s-ot alacsony feszültségen

A bit-ek továbbításánál egy +/- 150 ns-os eltérés megengedett.

A várakozási értékeket nem az adatlapból, hanem az alábbi útmutatóból vettem. Az útmutató szerint az adatlapban levő értékek rosszul vannak kiszámolva.

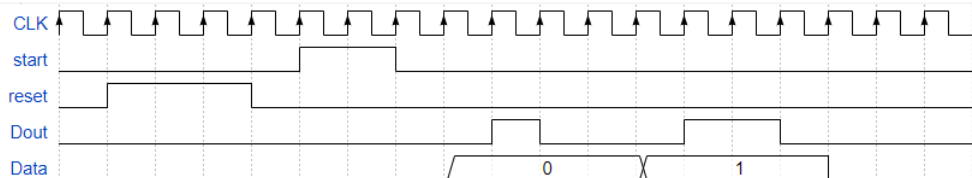
Egyelőre megpróbálok az útmutatóban megadott értékekkel dolgozni. Ha ez nem megfelelő működéshez vezet, akkor veszem az adatlapban levő értékeket.

### 7. Idődiagram



- CLK: 100 MHz-es órajel
- start: jel a folyamat elindításához
- reset: jel a folyamat resetálásához
- stop: jel a kiírás megállításához. Csak két 24 bit-es blokk kiírása közben tudja megállítani a kiírást
- Dout: Egyszálú adatsín a LED-ekre.
- Data: Kiírandó adat, Data\_rd felmenő órajelére olvassa be az adatot.
- Data\_rd: Aktiváló bit az adat beolvasására

**2019.11.11**



- CLK: 100 MHz-es órajel
- start: jel a folyamat elindításához
- reset: jel a folyamat resetálásához
- Dout: Egyszálú adatsín a LED-ekre.
- Data: Kiírandó adat