Beágyazott Elektronikai Rendszerek
Terv

Okos öntözőrendszer

Nagy Levente
levente.nagy960115@gmail.com
2020 január 8.

SAPIENTIA
ERDÉLYI MAGYAR
TUDOMÁNYEGYETEM
Villamosmérnöki Tanszék
Számítógépes Irányítási Rendszerek

Témavezető:
Dr. Brassai Sándor Tihamér, docens

Tartalom

1 Bevezetés

Napjainban a dolgok internete (Internet of things) egyre elterjedtebb. Már nem csak okos házak, okos irodák, de okos kertek illetve okos gazdaságok is léteznek. Az ilyen okos rendszerek lényege a mindennapi élet, munka, megkönnyítése illetve az erőforrások optimális kihasználása.
Ebben az projektben egy intelligens öntözőrendszert fogunk bemutatni. Az okos öntözőrendszer biztosítani tudja a növények, adott esetben a termés, kellő növekedését és az öntözővíz hatékony felhasználását. Az általunk bemutatott öntözőrendszer három paramétert felhasználva határozza meg hogy mikor kell a növényt öntözni. A paraméterek a következők: a talaj nedvességtartalma, a levegő hőmérséklete és a fény intenzitása.

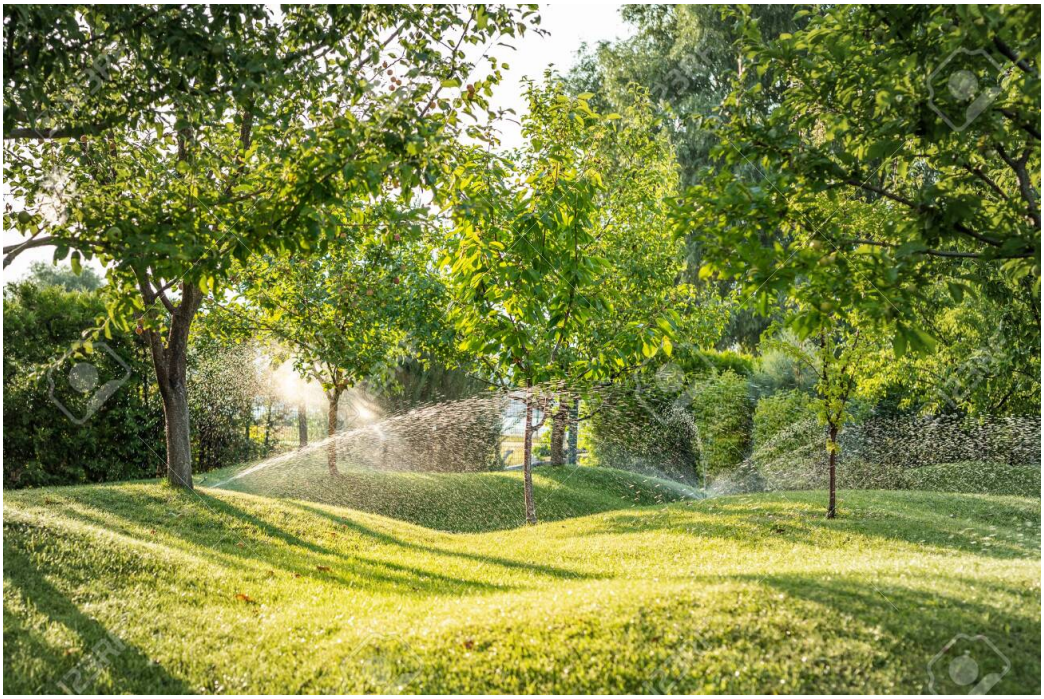

Figure 1: Automata öntözőrendszer

2 Tervezés

Az öntözőrendszert egy véges állapotú automatát segítségével irányítjuk. Az állapotgép a következő bementek alapján fogja kiszámolni a vezérlőjelet:

-*Nedvességszint*, digitális bemeneti buszon kapott adat 3 biten. Jelöljük M-el a talaj nedvesség szintjét, $M \in [0, 7]$.

-*Hőmérséklet*, digitális bemenet egy 1 bites jel. Jelöljük T-vel, $T \in [0, 1]$, ha $T = 0$ akkor a hőmérséklet alacsony, ha $T = 1$ a hőmérséklet magas.

-*Fényintenzitás*, digitális bemenet egy biten. Jelöljük L-el, $L \in [0, 1]$, ha $L = 0$ akkor a napfény értéke alacsony, azaz alkalmas az öntözősre, ha $L = 1$ a napfény túl erős.

2.1 Véges állapotú gép

Egy véges állapotú gép (Finite State Machine, FSM) vezérli az öntözést, az analóg-digitális átalakító bemeneteitől kapott jel alapján számolja ki a vezőrlő jelet. A tervezett automata két kimeneti állapottal rendelkezik:

- ST0 állapot, a nem öntözési állapot;
- ST1-es állapot, az öntözési állapot.

Ha a talaj nedvesség tartalma normál körülmények között 3-nál kevesebb vagy azzal egyenlő, az automata a ST0 állapotból az ST1-es állapotba megy át, és jelet ad az öntözőrendszer számára az öntözés megkezdéséhez.
Az vezérlő addig továbbítja az öntözési jelet, amíg a nedvességszint eléri a 7-et, vagy amíg a rendszer elveszíti az ideális öntözési körülményeket. Ideális körülmények között a hőmérséklet és a fényszint alacsony jelet küld, ha egy meghatározott beállított küszöbértéket meghaladnak, az automata magas jelet fog kapni, amely nem ideális az öntözéshez. Nem ideális körülmények között, ahol a hőmérséklet vagy a fény szintje túl magas, az automata csak akkor fog az ST0 állapotból az ST1-es állapotra váltani, és megkezdi az öntözést, ha a nedvességszint kisebb vagy egyenlő mint 1, ilyen körülmények között egész addig fog öntözni az automata amíg a talaj nedvesség tartalma eléri a 3-as szintet. Ha a körülmények ideálissá válnak, amikor a nedvességszint még nem érte el a 3-as szintet, az automata az ST1-es állapotban marad, és folytassa az öntözést, amíg el nem éri a 7-es nedvességszintet, ha elérte természtesen visszavált az ST0 állapotba és leállítja az öntözést.

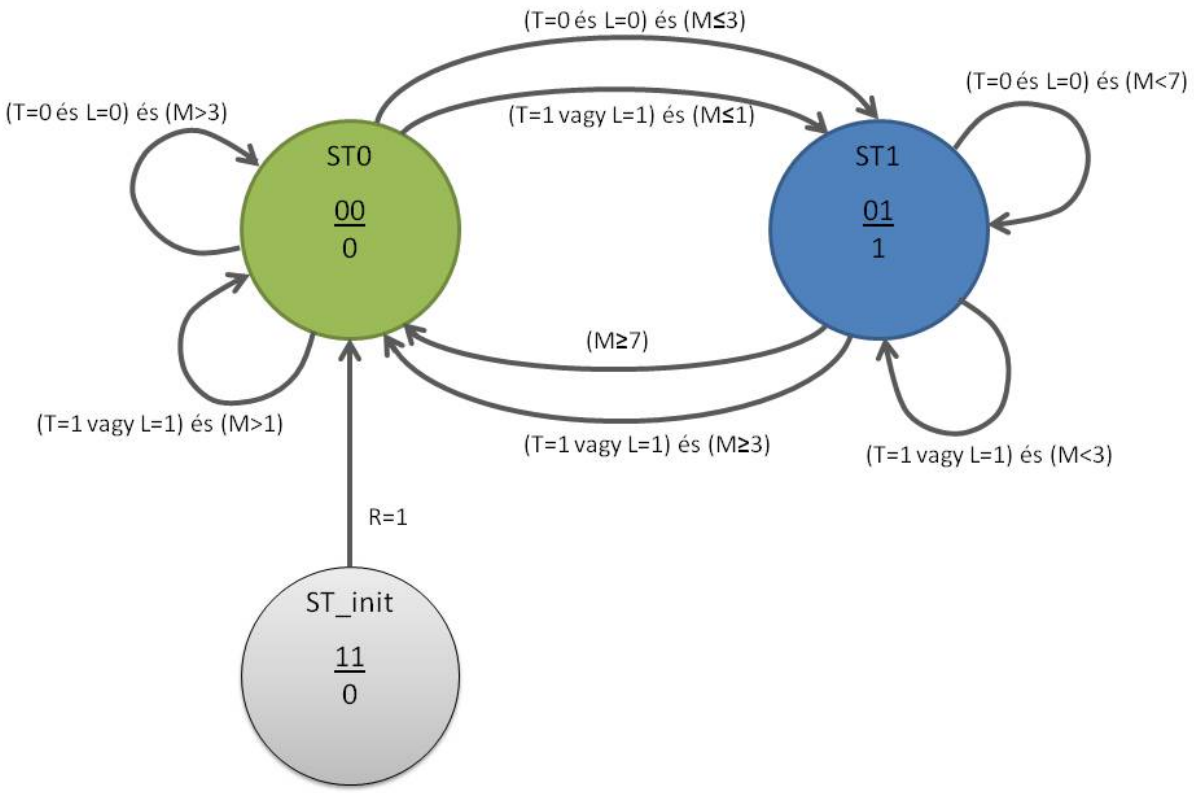

Figure 2: Véges állapotú gép

3 Implementáció

A projekt során Xilinx Vivado 17.4-es környezetet használtunk. A rendszert Digilent Zybo Z7 FPGA alapú fejlesztőlaphoz konfiguráltuk. A Zybo Z7 fejlesztőlap Zynq 7000-es processzort használ (Z-7010).

Az implementáció során először elkészítettük a rendszerünk állapotgépét VHDL modulban, majd azt teszteltük.

A következő lépésben kigeneráltuk az IP magot.

Majd az IP mag segítségével elkészítettük a hardver dizájnt.

3.1 Szimuláció

A tervezett VHDL modult a Vivado beépített szimulátorával teszteltük a következő esetekre.

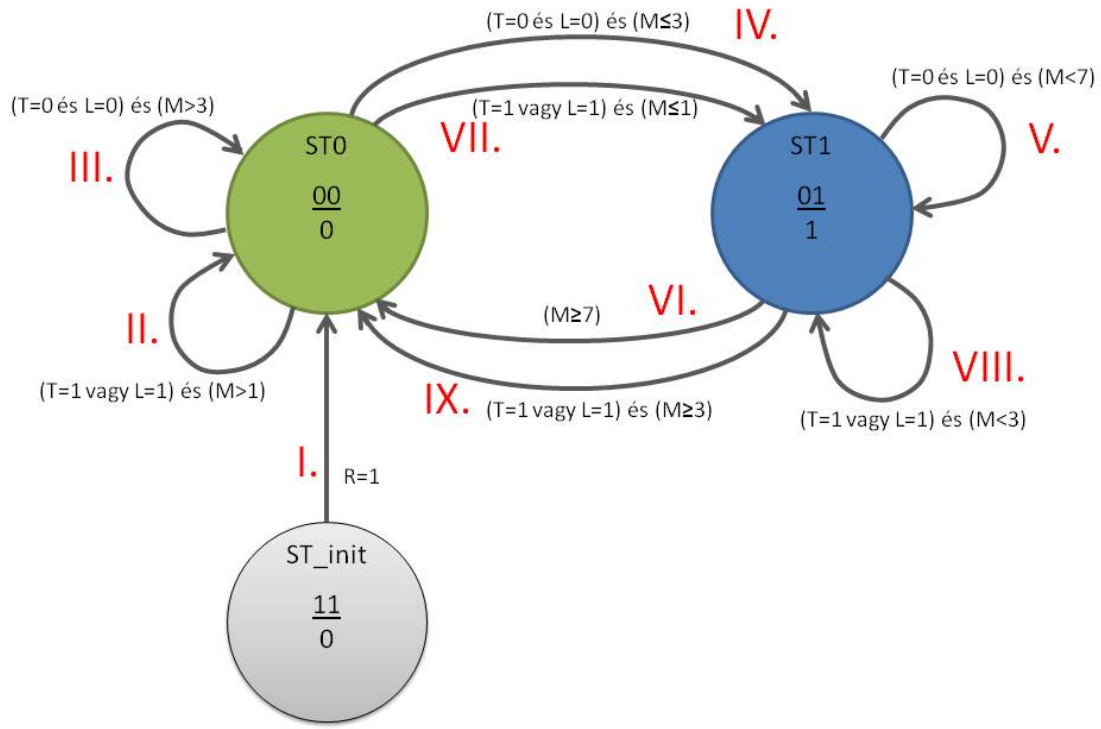

Figure 3: A véges állapotú gép és a teszt esetek.

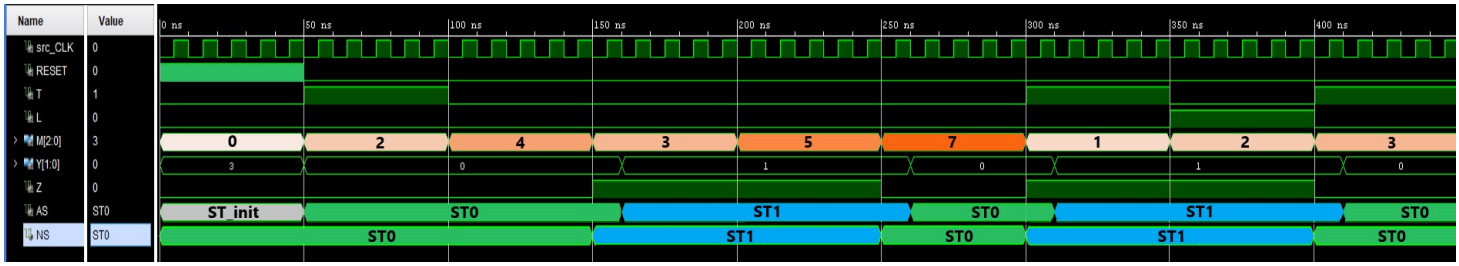

Figure 4: A szimuláció eredménye

A hardver dizájnba beillesztettük a tervezett IP magot illetve a Zynq Processzort, továbbá hozzáadtuk a processzor reszet modult illetve az AXI interkonnekt modult.

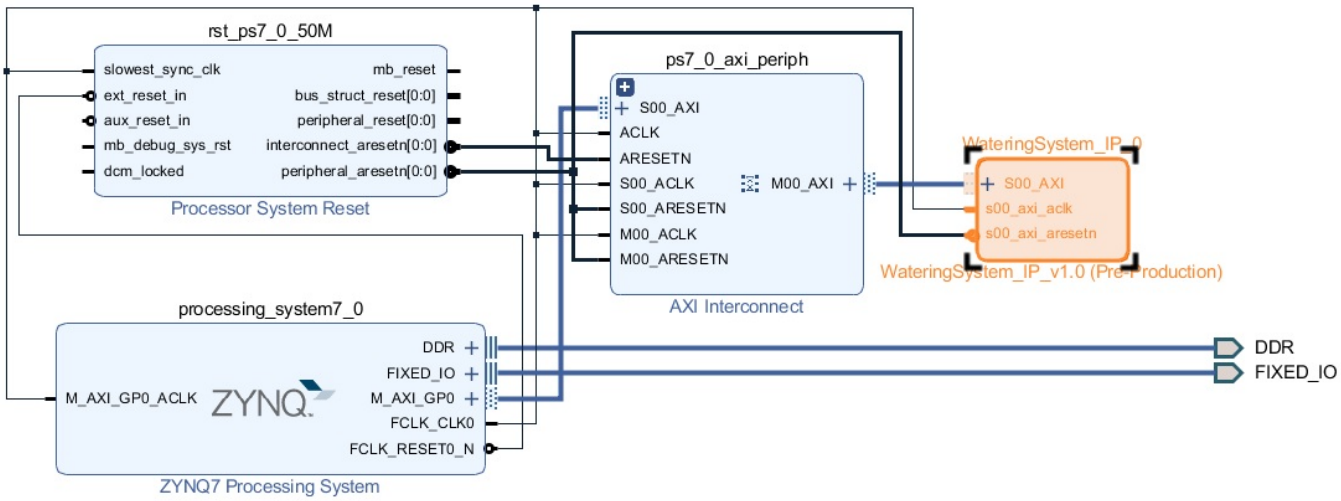

Figure 5: Hardver dizájn

4 Eredmények

```
****** Xilinx System Debugger (XSDB) v2017.4
  **** Build date : Dec 15 2017-21:08:33
    ** Copyright 1986-2017 Xilinx, Inc. All Rights Reserved.


xsdb% connect
tcfchan#0
xsdb% target 2
xsdb% fpga design_1_wrapper.bit
100%    1MB   1.8MB/s  00:01
xsdb% mwr 0x43C00000 0x00000001
xsdb% mrd 0x43C00004
43C00004:   00000003

xsdb% mwr 0x43C00000 0x00000012
xsdb% mrd 0x43C00004
43C00004:   00000000

xsdb% mwr 0x43C00000 0x00000020
xsdb% mrd 0x43C00004
43C00004:   00000000

xsdb% mwr 0x43C00000 0x00000018
xsdb% mrd 0x43C00004
43C00004:   00000005

xsdb% exit
exit
```

Figure 6: Regiszter irás-olvasás

5 Következtetések

5.1 További fejlesztések

References