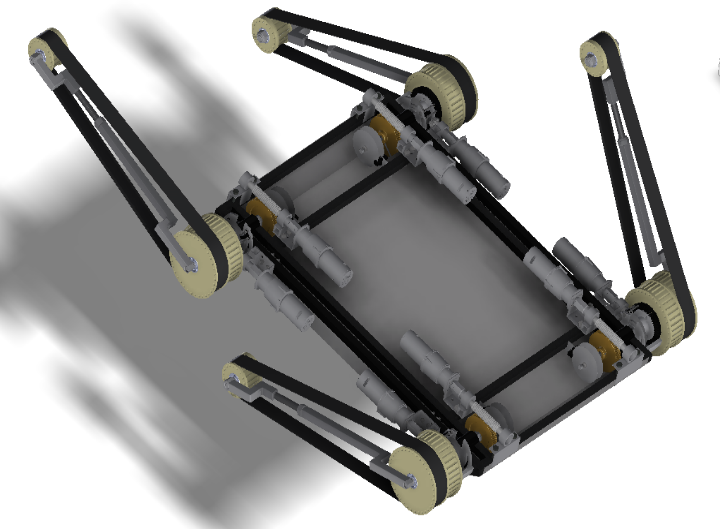
# Kivonat

A dolgozat célja mobilis tereprobot tervezése és megépítéséhez szükséges elemek tárgyalása és kivitelezése.

Egy masziv alvázhoz rögzítve van négy forgótalp, a forgótalpakot az alvázhoz képest 360 ban körbeforgathatjuk. Forgótalpakon megtalálható egy lánctalp, amelyeket egy dc motor segítségével hozzuk mozgásba Kup kerék áttételen keresztül.

A robot alapját képezi egy masszív váz, amely könnyű fémprofilokból áll össze és hegesztésekkel rögzítjük egymáshoz, az elemeket. A váz és az egész rendszer szimmetrikus két tengelyre nézve.



Kép. 2.2 Robot vázának Inventoros 3D Képe

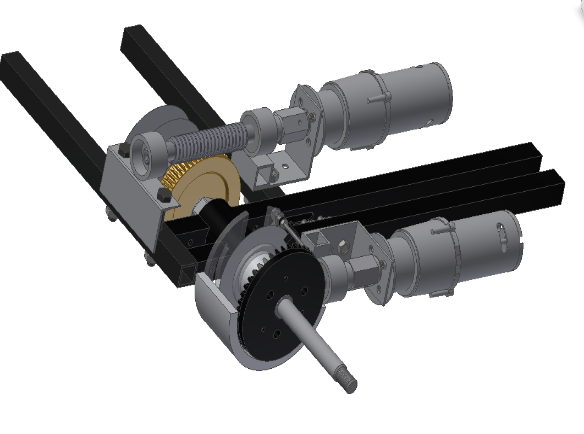
ForgóTalp1

ForgóTalp4

ForgóTalp2

KisKerék

NagyKerék



Orsó áttétel 40:1 arányban

Kupkerép áttétel 6:1 arányban

Sebesség mérő inkrementális tárcsa és szenzor

Kép. 2.2 Az alváz negyede

A rendszeren megtalálható 8 DC motor, amelyek segítségével Tudjuk a lánctalpakat mozgásba hozni, és a talpak szögét megváltoztatni, az alvázhoz képest. A motorokat H-híd segítségével vezérlem.

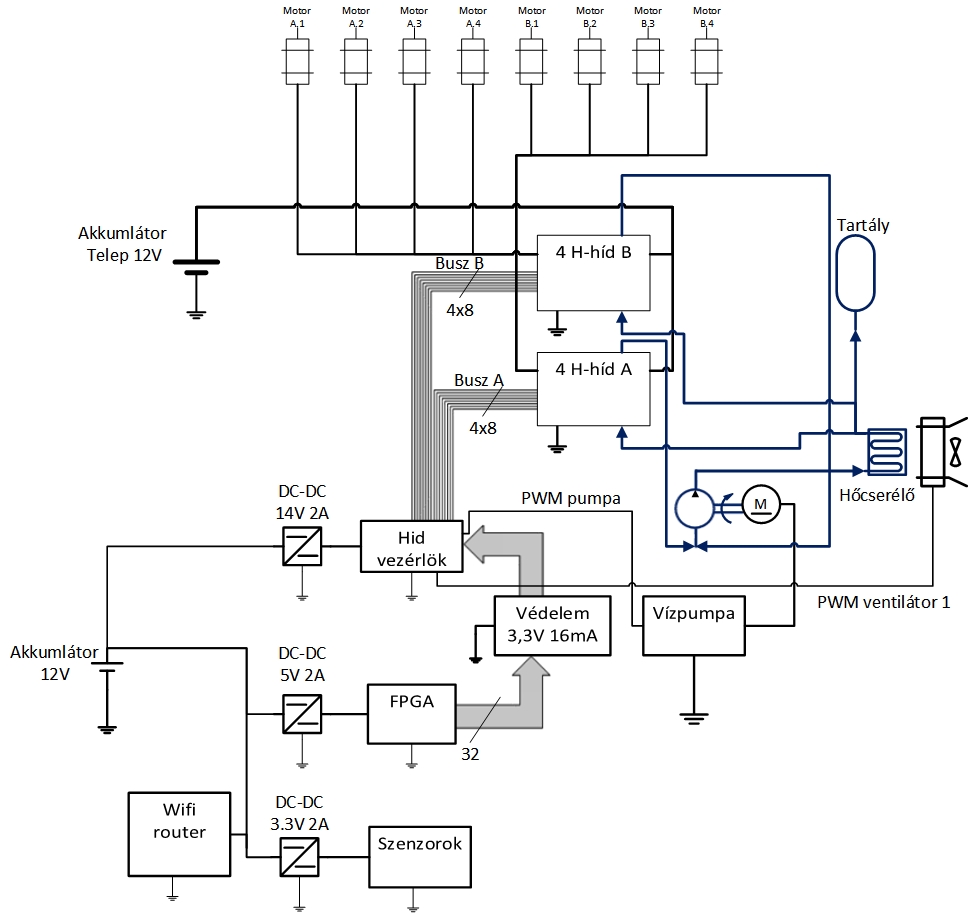
## Energiaellátás

A Kép. 3.48 látható 4H-híd B és A dobozok tartalmaznak négy H hidat, a hidak kettesével rögzítve vannak egy rézlemezre, amelyeken keresztül tudunk vizet keringetni egy réz csővezeték segítségével így hűtve a tranzisztorokat. A tranzisztorok galvanikusan levannak választva a lemeztől egy elektromos szigetelő segítségedével, de ugyanakkor a szigetelő jó hővezető is. A rendszerből a hőt vízhűtés segítségével vezetem ki, olyan megfontolásból hogy a rendszer terepen kell majd működjön és a por ne kerülhessen be a rendszerbe.

A Kép. 3.48 látható 32 bites buszvezeték, amely az FPGA rendszertől érkezik, és megtalálható benne a 8 motor hajtásához szükséges pwm beavatkozó jelek, amelyek az amplitúdója 3,3V. A busz szalagvezeték segítségével van kialakítva, és megtalálható benne egy védelem is, amely megvédi az FPGA rendszert az esetleges visszahatásoktól. A védelem 3,3V ós zenre dióda segítségével történik, melynek feladata megakadályozza a 3,3V-ál nagyobb feszültségek az FPGA rendszerbe történő továbbhaladását. A diódával sorba van egy ellenállás is, amely az áramot korlátozza mivel az FPGA a bemenetén 16mA áramot visel el. A Kép. 3.48 látható a robot energiaellátásának a terve. Az energia ellátás akkumulátorokkal fog történi, rendeltetésük szerint két csoportba oszthatók: egy 12V akkumulátor gondoskodik a rendszer digitális áramköreinek az ellátásáról. A digitális elemeket DC-DC konverteren keresztül táplálom be melyeknek a feszültsége állítható.

A másik energiaforrás egy több akkumulátorból álló telep lesz, amelyek párhuzamosan lesznek kapcsolva, és a H hidakat táplálják be energiával.

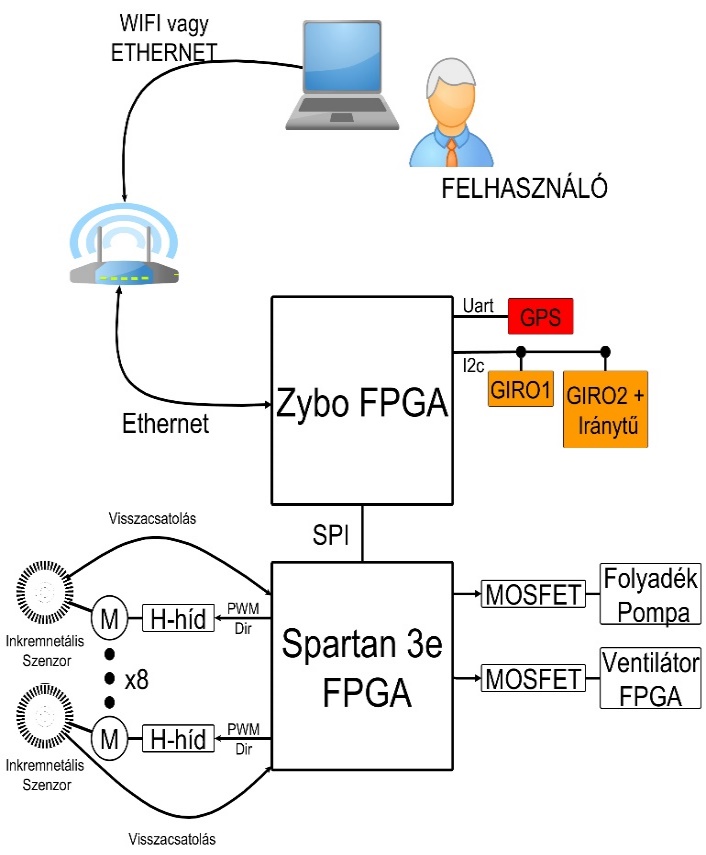
A vízpumpa és a ventilátorok motorja PWM jel segítségével van vezérelve egy N csatornás MOSFET tranzisztor segítségével.



Kép. 3.48 A robot energia ellátása valamint a hűtő rendszer elvi felépítése

# FPGA Rendszer Felépítése

A rendszeren megtalálható két FPGA fejlesztő lap, egy ZYBO amely nagyobb erőforrással rendelkezik, de kevés a kivezetéséinek a száma, és egy Spartan3e chippel rendelkező fejlesztőlap, amely kevés erőforrással bír, de 120 kivezetést tartalmaz.



Kép. 3.34 rendszer elvi felépítése

A Spartan rendszer fog foglakozni a 8 motor szabályzásával, fogadja az adatokat a ZYBO rendsertől feldolgozza és elvégzi a megfelelő beavatkozást.

A ZYBO rendszer feladat

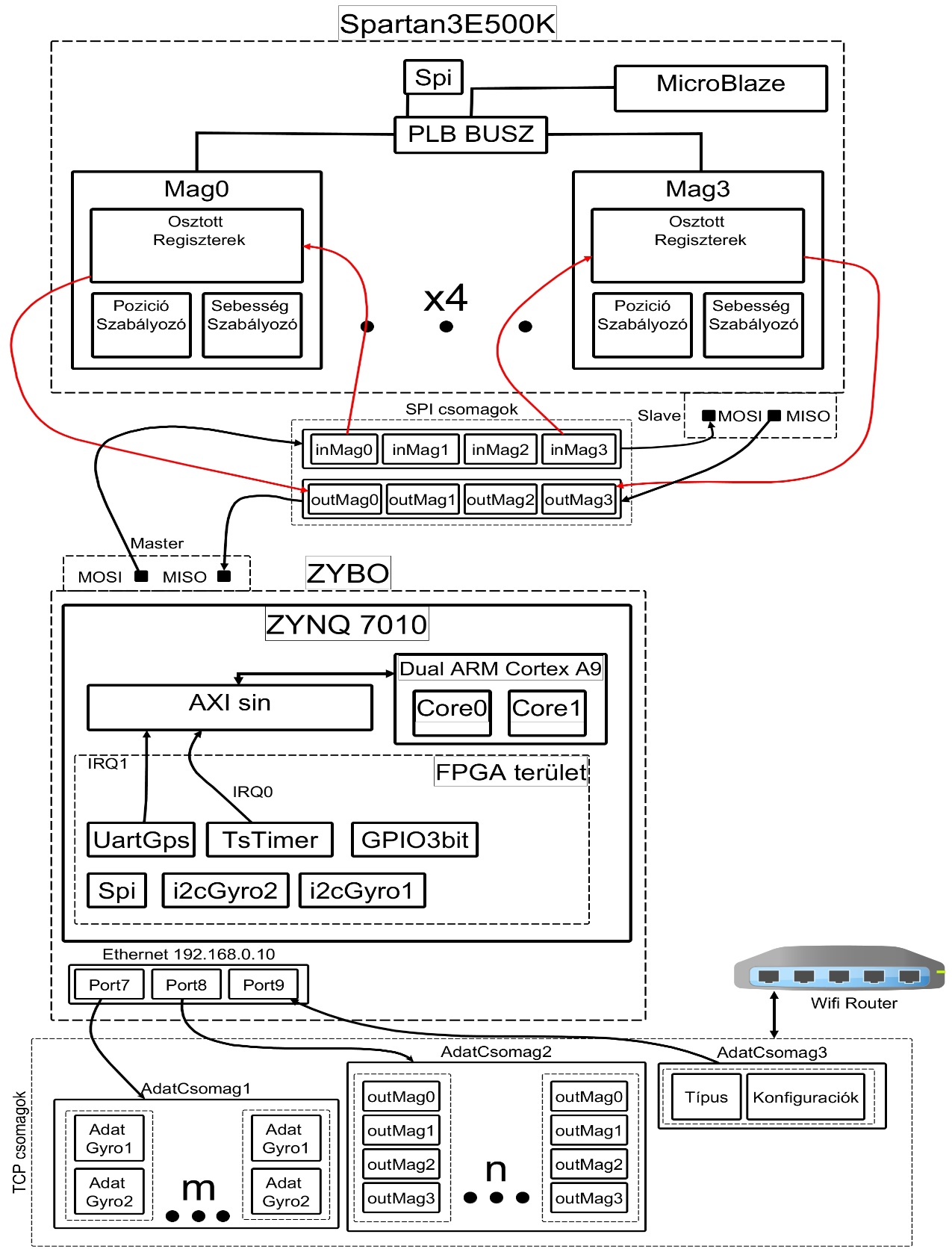
a komplex érzékelők: giroszkóp, GPS modul adatainak gyűjtése és a biztosítja a kommunikálást a rendszerrel Etherneten keresztül.

Szükséges a robot négy motor pozíciószabályzása és négy motor sebességszabályzása. A szabályzó körök kialakítására az egy hurkú kialakítás választottam, a visszacsatolást inkrementális adó segítségével valósítottam meg. A szabályozók a rendszerben hardveresen vannak megvalósítva. A PID típusú szabályzót alkalmazattam a sebesség szabályzására és egy hardveres automatát a pozíció szabályzására.

Minden szabályzót System Generátorban valósítottam meg és szimuláltam le, szoftveresen és hardveresen (COSIMULATION).

A hardveres PID szabályzót és a pozíció szabályzót, elhelyeztem egy IPmagban és a peremétereket lehetővé tettem, hogy osztott regisztereken szoftveresen eltudjuk írni és olvasni. A regiszterek tartalmát a MicroBlaze processzor kezeli. A microblaze processzor SPI kommunikációs protokollon keresztül fogadja a szabályozók előirt értékeit és beírja, a regiszterekbe valamit párhuzamosan továbbküldi a szabályzókörök adatait.

A zybo rendszerben megtalálható két beépített processzor az egyik processzor feladata az érzékelők adatainak a beolvasása, valamint három TCP szervert futasson amelyeken keresztül lekérhetjük a mért adatokat illetve parancsokat küldhetünk a rendszernek.



Kép. 3.37 Kommunikációs csomagok és az FPGA áramkörökbe programozott modulok elvi felépítése

# Pozíció Szabályzása

A mechanikai rendszer kialakításából adódóan, ha a hajtó motor leáll és a hajtott tengely terhelés alatt marad, a hajtott tengely a súrlódások miatt nem tud visszafele hajtani, ezért elegendő, ha a megfelelő időpillanatban a hajtó motort leálltjuk. Mivel a Dc motor polaritás váltásakor a motor forgási iránya is megváltozik elegendő, ha a maximális vagy minimális szabályozó jellel avatkozunk be a rendszerbe.

Az elkészített szabályozót a következő egyenletek írják le:

Elmondható a kimeneti szabályozó jel függ a hiba értékétől. A mechanikai rendszerben kotyogás van, és az ebből származó zajokat szeretnénk kiszűrni úgy, hogy ha a mechanizmus a megfelelő pozícióban van, akkor egy tartományban a szabályozót érzéketlené tesszük a bemenetre mindaddig, amíg a hiba ki nem lép a sávból vagy a referencia jel meg nem változik.

A szabályozó felépítése:

A bementek: „RefVal” –előírt pozíció impulzusban mérve, 16 bites előjeles érték, „AktVal” – aktuálisan mért pozíció impulzusban mérve, 16 bites előjeles, „U” –kimente 17 bites előjeles.

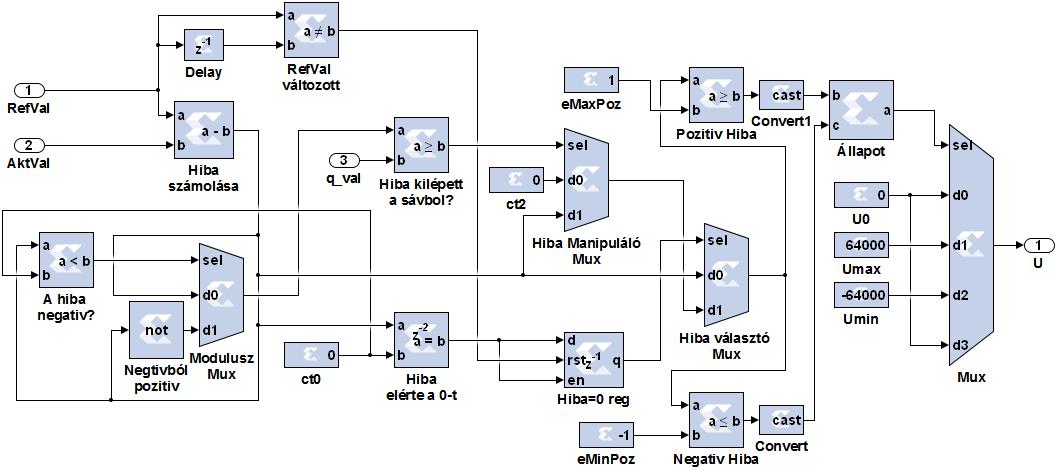
A „*Hiba Számolása*” modul végzi a hiba kiszámolását az aktuális és az előirt pozícióból. A hiba lehet negatív is ezért „*A hiba negatív?”* Komparátor segítségével eldöntjük, hogy negatív vagy pozitív a hiba, azután a „*ModuluszMux*” segítségével kiválasztjuk a magát a számolt hibát, ha az pozitív, vagy a hiba tagadottját, ha az negatív így megközelítve a moduluszát.

Ha a hiba elérte a 0-t akkor a „Hiba=0 reg” értéke 1 lesz mindaddig amíg az előírt pozíció meg nem változik.

A „Hiba kilépett a sávból” modul megvizsgálja, hogy a hiba moduluszakisebb mint a sáv értéke ha igen akkor a „Hiba Manipuláló Mux” segítségével a továbbiakban a hiba 0 lesz.

A „Hiba választó Mux” a „Hiba=0 reg” irányítására válasza ki manipulált hibát vagy számolt hibát, amelye továbbmegy a háromállású szabályzóba.

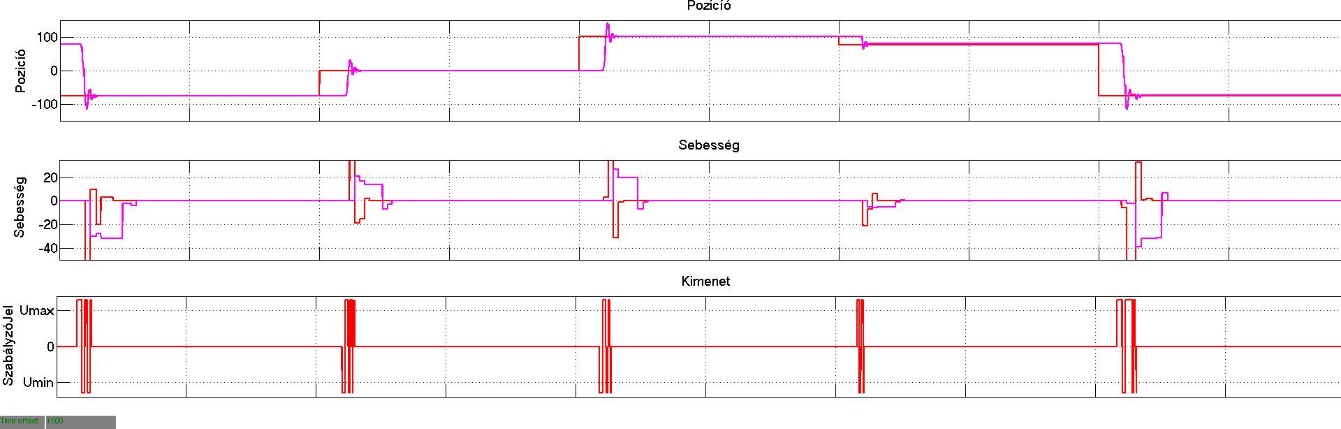
Az „Umin”, „Umax”, „U0” 17 bites előjeles regiszterek segítségével kiválaszthatjuk a szabályzó maximális és minimális beavatkozó jelének értékét. A „Pozitív Hiba” „Negatív Hiba” eldöntik, hogy a hiba mely tartományba van. Három tartományt különböztetünk meg: negatív pozitív, és 0 hibát. A „Mux” kiválassza az aktuális állapotnak megfelelő vezérlő jelet.



Kép. 3.9 A Pozíció szabályozó System generátoros felépítése

DC motor pozíció szabályzása mérőstandon

Mivel a mérőstand nem rendelkezik, csiga áttelel ezért a szabályozó enyhén oszcillál, de ez majd az áttétel jelenlétében nem fog fennállni.



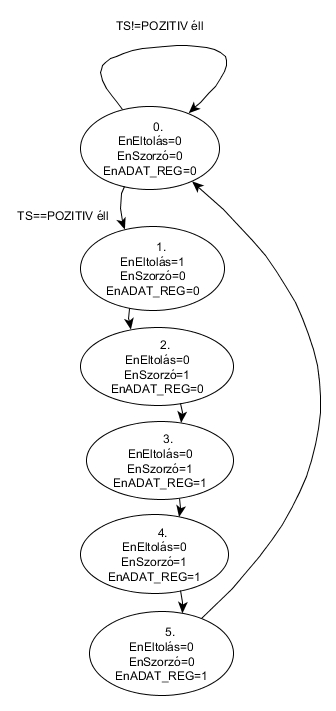
Kép. 3.14 Pozíció szabályzása a Kép. 3.9 látható kialakításban.

# **Diszkrét Hardveres PID szabályozó**

Napjainkban az egyik leghasználtabb szabályozótípus a PID, amely rekurzív egyenlete a következő:

[2]

Az általam elkészített PID szabályozó hardveresen van megvalósítva FPGA áramkörben. A fent látható összefüggések alapján egy adat utas automatát terveztem, amelyet majd System Generátorban építtettem meg. A PID szabályozó paramétereit, a Q paraméterek segítségével adhatjuk meg, amelyek függenek az ismert paraméterektől: - deriválási idő, -integrálási idő, mintavételezési periódus, valamintproporcionális erősítés.Az automata öt állapotot tartalmaz. Minden mintavételre, az automata végigpörög az állapotokon és majd visszatér a kiinduló állapotba. Az állapotokban végzet műveletet az FPGA fejlesztő lap órajelének a frekvenciájára hajtjuk végre, minden állapoton egy órajel periódus alatt lép át.



Kép. 3.2 Állapot automata, amely leírja a Diszkrét PID szabály ózót

# Megvalósítás System Generátorban

Az adatút kiválasztására egy 2bit-es számlálót alkalmazunk (ADAT UT) amely, órajelre számol, ha az enable(en) bemenetén logikai 1 érték található, 2biten a számláló maximum 4 értéket vehet,ezért a számlálót úgy állítjuk, be hogy a maximális értéke 2 lehessen így 0,1,2 értékeket veheti fel. Az adat utakat két 16bites multiplexerrel MUXQ és MUXE válaszuk ki.

Bemeneti paraméterek a 16bit előjeles egész értékek, 16bit előjeles egész érték,-bool típusú.

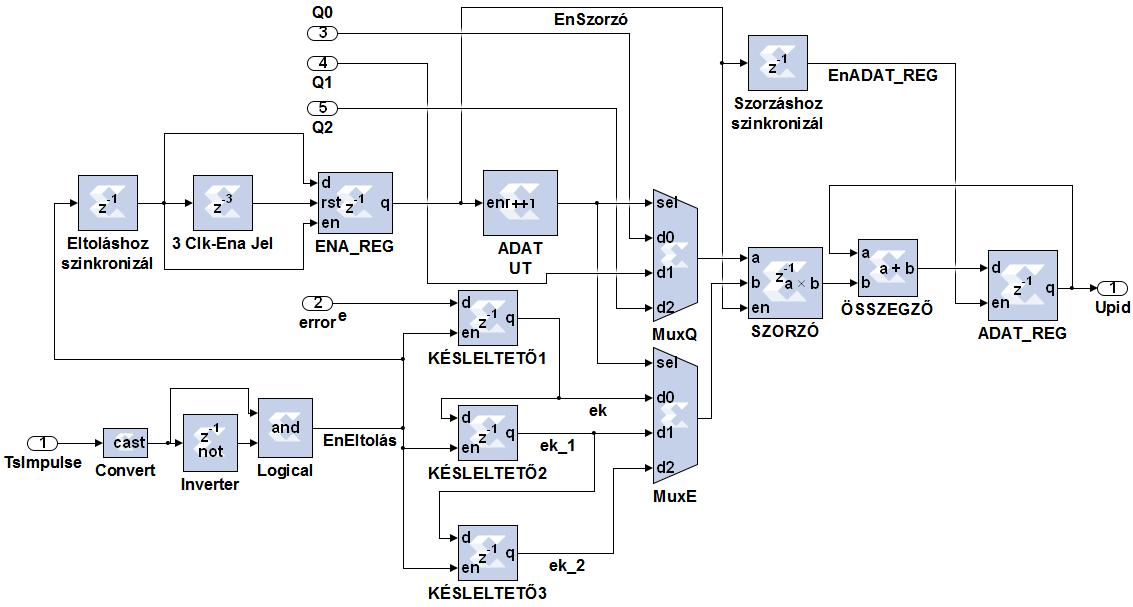
Kimenetek: 17bit előjeles egész érték.A MUXQ a Q paraméterek kiválasztásáért felelős, valamint a MUXE az időben késleltetett bemeneti értékek kiválasztásáért felelős. A Kép. 3.3a„*SZORZÓ*” modul a két szelekciós multiplexertől kapott értéket összeszorozza, aztán hozzáadja az „*ADAT\_REG*” regiszter értékéhez.

Minden modulértéke szaturálódik abban az esetben, ha túlcsordulna akár negatív vagy pozitív irányba,így elkerülhetjük azt is, hogy az integráló tag változatlan hiba bemenete esetén túlcsorduljon és felborítaná a rendszer működését.

A Kép. 3.3 látható KÉSLELTETŐ regiszterek állítják elő múltbeli hiba értékeit, úgy hogy a három regiszter egymás után van láncolva és a felfutó élére akövetkező regiszterbe csúszik át az érték. A KÉSLELTETŐ1 regiszterbe kerül mindig az aktuális mintavételezett hiba értéke.

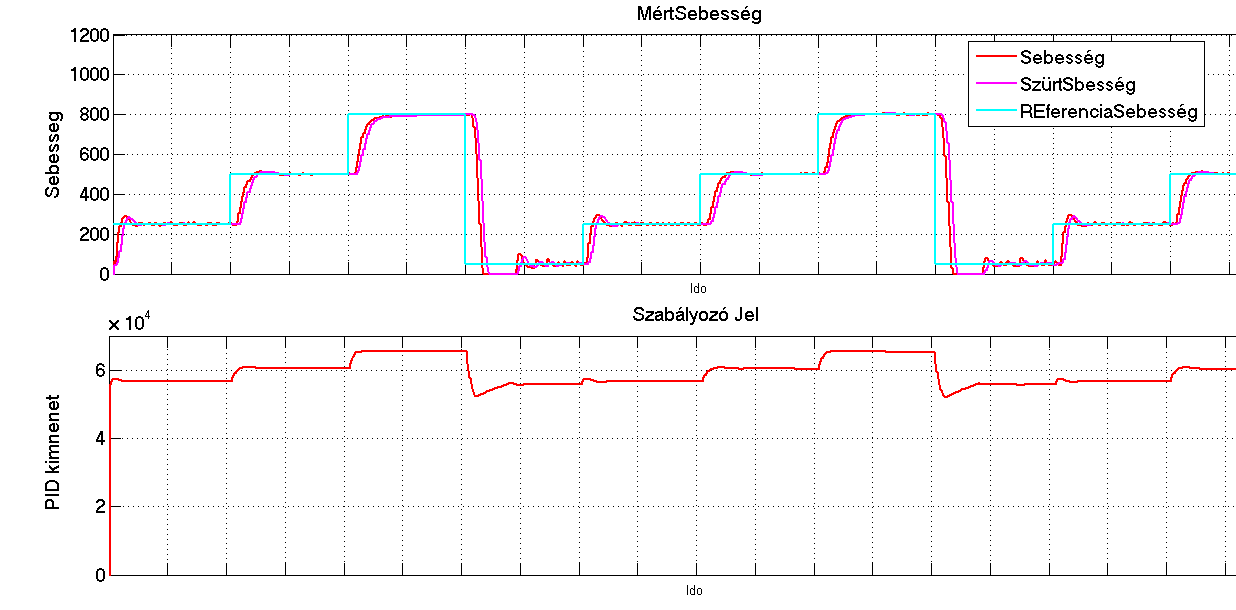
Az „*ADAT UT”* számláló csak a regiszterek elcsúsztatása után indul el, amelyet „*Eltoláshoz Szinkronizál*” késleltető elem valósít meg. Az „ENA\_REG” és a „3 Clk-Ena Jel” egyűt valósítják meg a három órajelig tartó logikai engedélyező jelet, amely a számlálót indítja el.

Kép. 3.3 A PID felépítése System Generatorban



Az él detektáló elemet egy tagadó kapu „*Inverter*”valamint egy „és”kapu (Logical) biztosítják, mégpedig úgy, hogy figyeljük egyazon jel előbbi periódusban az értéket és összehasonlítva a két értéket tudjuk detektálni a jel váltózását. A „*Szorzáshoz Szinkronizál*” modul segítségével tudjuk engedélyezni az „*ADAT\_REG*” bemenetét a „*Szorzó*” modul „*EnSzorzó*” engedélyező jeléhez képest késleltetve 1 órajelet. Erre azért van szükség, mert a szorzás eredménye 1 órajelet késik az elindítást követően és az eredményt szeretnénk eltárolni.

# Dc motor sebesség szabályzása mérőstandon



Kép. 3.13 Sebesség szabályozás PID szabályzóval.

**Megvalósítások**

* Megépített mechanikai rendszer
* Összeállított FPGA rendszer
  + Két FPGA közti kommunikáció SPI
  + Hardveres elemek megvalósítása System Generatorban
  + Több megszakításos szoftveres és hardveres rendszer
  + C programok: MicroBlaze és ARM processzorokra
* 8 dc motor szabályzó (PID, pozíció)
* Megépített mérőstand a szabályozók és az érzékelők tesztelésére
* Szenzor adatok beolvasása: gyroszkop1 és gzroszkop2, inkrementális érzékelők
* Ethernet kommunikáció + kliens GUI (Matlabban)
* Teljesítmény elektronika megépítése: hídvezérlők, H-hidak, vízhűtés