MŰEGYETEM 1782

SZAKDOLGOZAT-FELADAT

Marton Dániel (S4IBL3)
szigorló villamosmérnök hallgató részére

Impulzusszélesség-moduláció alapú motorvezérlés FPGA-val

A modern gépjárművek biztonságtechnikai és kényelmi funkcióinak megvalósításában, környezetvédelmi jellemzőinek javításában stb. egyre jelentősebb szerepet kapnak a számítástechnikai megoldások. Az egyik kritikus új funkció az elektronikus kormányrendszer. Ez egy – általában háromfázisú szinkron – motoron keresztül ad rásegítő nyomatékot a kormánygépre vagy a kormányoszlopra. A motor pontos, dinamikus szabályozása komplex, nagy számításigényű feladat, amelyet általában nagyteljesítményű mikrokontrollerek segítségével végeznek.

Érdekes kutatás-fejlesztési feladat a számítások FPGA-n történő implementálása, melynek eredményeként a mikrokontroller terhelése csökkenthető, a valós idejű kód nagy része a programozható áramkörön (későbbiekben sorozatgyártásban ASIC-en vagy ASSP-n) futtatható. A jelölt a megvalósíthatóság vizsgálatába kapcsolódik be, a motorszabályozási hurok egy részét – a számított motoráram alapján az impulzusszélesség-vezérelt kimeneti jelek előállítását – valósítja meg. A hallgató feladata magában foglalja az alábbiakat:

- A mezőorientált motorvezérlés megismerése
- A használt PWM minták megismerése
- A kiszámított motor áramnak megfelelő PWM minták kiszámítása FPGA-n
- A PWM minták fázisonkénti kivezérlése az FPGA kimenetén
- A szükséges FPGA tesztkörnyezet létrehozása
- A feldolgozás redundánssá tételéhez szükséges lépések (PWM minta visszamérése) megvalósíthatóságának vizsgálata

A feladat megvalósításához szükséges eszközöket a ThyssenKrupp Presta Hungary Kft. biztosítja.

Tanszéki konzulens: Dr. Sujbert László docens
Külső konzulens: Dr. Balogh András (ThyssenKrupp Presta Hungary Kft.)

Budapest, 2014. szeptember 18.

……
Dr. Jobbágy Ákos egyetemi tanár
tanszékvezető

Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Méréstechnika és Információs Rendszerek Tanszék

1117 Budapest, Magyar Tudósok krt. 2. I. ép. I.E.444.
Telefon: 463-2057, Fax: 463-4112
http://www.mit.bme.hu • e-mail: mitadm@mit.bme.hu

Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Méréstechnika és Információs Rendszerek Tanszék

Impulzusszélesség-moduláció alapú motorvezérlés FPGA-val

Szakdolgozat

Készítette
Marton Dániel István

Konzulens
dr. Balogh András
dr. Sujbert László

2014. december 17.

Tartalomjegyzék

HALLGATÓI NYILATKOZAT

Alulírott *Marton Dániel István*, szigorló hallgató kijelentem, hogy ezt a szakdolgozatot meg nem engedett segítség nélkül, saját magam készítettem, csak a megadott forrásokat (szakirodalom, eszközök stb.) használtam fel. Minden olyan részt, melyet szó szerint, vagy azonos értelemben, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

Hozzájárulok, hogy a jelen munkám alapadatait (szerző(k), cím, angol és magyar nyelvű tartalmi kivonat, készítés éve, konzulens(ek) neve) a BME VIK nyilvánosan hozzáférhető elektronikus formában, a munka teljes szövegét pedig az egyetem belső hálózatán keresztül (vagy autentikált felhasználók számára) közzétegye. Kijelentem, hogy a benyújtott munka és annak elektronikus verziója megegyezik. Dékáni engedéllyel titkosított diplomatervek esetén a dolgozat szövege csak 3 év eltelte után válik hozzáférhetővé.

Budapest, 2014. december 17.

Marton Dániel István
hallgató

Kivonat

Jelen szakdolgozat egy *Xilinx Spartan-3E* FPGA-val implementált 3 fázisú szinkronmotort meghajtó inverteráramkör vezérlőjeleit előállító modul leírását tartalmazza. Igyekszik először egy rövid általános elméleti áttekintést nyújtani a váltóáramú motorokról, vezérlésükről, meghajtásukra használt energiaátakítókról és mindezek alkalmazásakor előkerülő jelentősebb megfontolásokról. Ezek után a térvektoros leírás matematikai eszköztárát tekinti át, majd egy térvektorokon alapuló zárthurkú szabályzási kört mutat be, illusztrálva benne a korábban bemutatott elemek szerepét. Végezetül a szabályzókör kimeneti modulját implementálja a fentebb említett FPGA-val, almodulonként funkcionális bontásban, kitérve az implementáció során felmerülő kérdésekre.

Bár mikrokontroller alapú rendszerekben ez szabályzási feladat már kiterjedten implementált, az FPGA-k terjedése és elérhetősége miatt indokolt ezen platformon is a rendszer megvalósíthatóságának vizsgálata. Ennek oka egyrészről az FPGA-k determinisztikusabb működése és az emiatti nagyobb megbízhatósága, másrészről a szabályozási kör mikrokontrollerének terhelése ezáltal csökkenthető, valósidejű kódjának megbízhatósága javítható.

Az FPGA-val való implementáció vizsgálata továbbá a későbbi esetlegesen ASIC-ben történő implementációt is segíti, mind megvalósíthatóság és erőforrásigény vizsgálatával, mind egy kezdeti HDL nyelvű hardveres leírás szolgáltatásával.

Abstract

This thesis contains the description of an inverter controler designed to drive a 3-phase motor, implemented on a *Xilinx Spartan-3E* FPGA. First it aims to provide an overview of AC motors, their control and the power converters used to drive them, while examining the most frequent problems one has to face during the application of these units. After this, the mathematical toolbox used in space vector control is explained. The subsequent chapter uses these aforementioned elements to overview a general space vector based closed-loop control circuit. Last it implements the output module of one such circuit with the aforementioned FPGA, providing a hierarchical and functional description of the submodules, while examining the problems, which emerged during the implementation.

Although such control systems for this tasks are already extensively implemented on microcontroler based systems, due to the spread and availability of FPGA-s its justifiable to assess the feasibility of the system on this platform. The reason for this lies partially in the more deterministic operation of FPGA-s (thus better reliabilty), and partially in the aim to reduce the load of the microcontroler in the control-loop (thus allowing tighter real-time code).

Furthermore by assessing the implementation on FPGA, it provides an overview for a later ASIC based implementation, both in feasibility and resource estimation, and by providing an early HDL based description.

Bevezető

A villamos motorok viszonylagosan régi találmányok, már 1855-ben megalkotta Jedlik Ányos a 'villámdelejes forgony'-át. Bár sokféle villanymotor létezik, a felfedezésük és működési elvük leírása jórészt a múlt században megtörént. Ennek ellenére a villanymotorok folyamatosan fejlődnek, javulnak egyik részről új konstrukciós megoldásoknak köszönhetően, másrészt a vezérlésük fejlődésével. Mind a szabályozáselmélet fejlődése, mind az energiaátalakítók fejlődése ezt segítette. A kapcsolóüzemű átalakítóknak hála, a tipikusan jobb paraméterekkel rendelkező váltakozó áramú motorok tömegével kerültek egyenáramú környezetben is használatra. A processzorok megjelenésével és elterjedésével nagy számításigényű szabályozási és jelfeldolgozási feladatok is olcsón elvégezhetővé váltak. Az FPGA-k napjainkbani terjedése ezen a mikrokontrollereken meglévő algoritmusok párhuzamosítását és ezzel további gyorsítását teszik lehetővé. Nem mellékesen ASIC-ek tervezéséhez és funkcionális teszteléséhez az FPGA-k kiválóan használhatók. A valósidejű működéshez való közeledés a zárthurkú szabályzási kör nagyobb sávszélességét is eredményezi. Továbbá az ASIC-ek, illetve FPGA-k determinisztikusabb működésüknek hála biztonságkritikus rendszerekben előnyt élveznek. A fentieket szem előtt tartva bizonyos rendszerek, például egy szervórendszer meghajtásának vezérlésére (ami biztonságkritikus, gyors jelfeldolgozást igényel, nagy dinamikatartományon) egy FGPA-val implementált vezérlő jó választás lehet. Bár az áruk csökken, de még e szempontból nem érték el mikrokontrollerekkel való egyenrangú félként való versenyzés határát. Mindenesetre megfontolásra érdemes, hogy az FPGA flexibilitása miatt az eddigi külső ASIC-ek FPGA-ban történő megvalósításával az elemszám csökkenthetővé válhat egy piaci termék esetén.

1. fejezet

Motorok

1.1. Motorok típusai, csoportosítása

Azokat a villamos gépeket nevezzük motoroknak, melyek egyen- vagy váltakozó áramú villamos energiát alakítanak át mozgási energiává. [18]

A villamos forgógépek nyomatékát mágneses mező és áramot vivő vezeték hozzák létre. A mágneses mezőt a gép álló- vagy forgórésze létesítheti, az áramot vivő vezeték - ennek megfelelően - lehet a forgórészen vagy az állórészen. Állandó nagyságú forgatónyomaték eléréséhez a vezetőben az áramirányt változtatni kell, attól függően, hogy pillanatnyilag milyen polaritású mágneses térben helyezkedik el. Ez megoldható mechanikus megoldásokkal (mechanikus kommutáció, pl. klasszikus egyenáramú motorok), váltakozó áram használatával (pl. állandó mágnessel vagy egyenárammal gerjesztett szinkrongépek). Váltakozó áramú gerjesztést viszont előállíthatunk egyenáramból átalakítók segítségével megfelelő vezérléssel (2 fejezet). Ezeket a megoldásokat tekinthetjük elektromos kommutációnak.[17]

A mechanikus kommutációval szemben ennek az előnye meglehetősen nyilvánvaló, hiszen a félvezető technológiának hála nincs szükség mozgó/kopó alkatrészekre (kefékre), így robusztusabb, megbízhatóbb kivitel valósítható meg, valamint a kommutációs sebesség jelentősen megnövelhető [16]. Továbbá a legtöbb esetben a váltakozó áramú motorok egyéb jellemzői is kedvezőbbek (nincsen szikrázás, emiatti feszültségesés és elektromágneses interferencia, kedvezőbb súly/teljesítményarány, hatásfok stb...).[12] Nem mellékesen az autókban általában nem áll rendelkezésre váltóáramú tápellátás, de az alkalmazás a váltakozó áramú motorok megbízhatóságát és hatékonyságát követeli meg.[5]

Tekintsük át a főbb váltakozó áramú motorokat ennek megfelelően[12]:

<table>
<tr><td colspan="6">AC motorok</td></tr>
<tr><td colspan="2" rowspan="2">Aszinkron</td><td colspan="4">Szinkron</td></tr>
<tr><td colspan="3">Permanens mágneses</td><td>Tekercselt</td></tr>
<tr><td colspan="2" rowspan="2"></td><td colspan="2">PMSM</td><td rowspan="2">BLDC</td><td rowspan="2"></td></tr>
<tr><td>Kalickás</td><td>Csúszógyűrűs</td><td>SPM</td><td>IPM</td></tr>
</table>

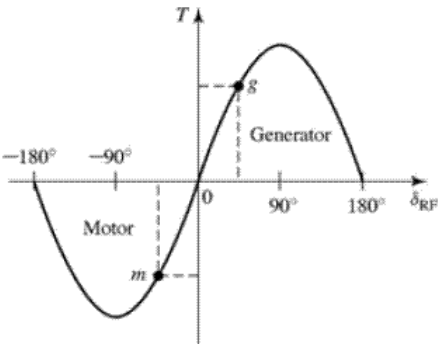
1.2. Szinkron motorok

A szinkron motorok alapvető működési elve, hogy a forgórésszel állandó mágneses mezőt létesítünk (főmező), majd az állórész tekercseire váltakozó feszültséget kapcsolva forgó mágneses mezőt hozunk létre. Az ellentétes pólusok vonzása miatt így a rotor a forgó mágneses mezővel fog szinkronban forogni. Ennek következménye, hogy a szinkron motorok fordulatszáma a terheléstől függetlenül állandó, és csak a pólusszámtól és a forgó mező frekvenciájától függ.[18]

$$n = \frac{60f}{p} \tag{1.1}$$

Ahol n a fordulatszám, p a motor póluspárjainak száma.

Terhelés nagyságától függően a forgórész mágneses mezőjének szöge δ szöggel elmarad a forgó mágneses mezőtől. Ezt hívjuk a motor *terhelési szögének* és a nyomaték, illetve ennek megfelelően a motorból kivehető maximális teljesítmény függ tőle, 1.1. ábrának megfelelően.[18]



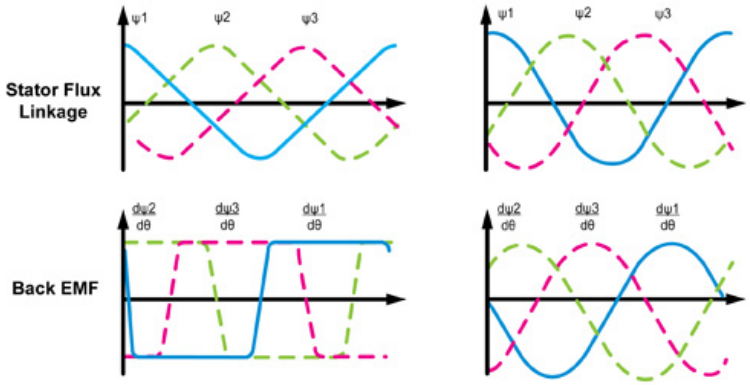
1.1. ábra. *Szinkronmotor nyomatéka a terhelési szög függvényében.*

Láthatóan a maximális nyomatékot 90°-nál érjük el, ezt hívjuk *billenőteljesítmények*, illetve *billenőnyomatéknak*. Ezt meghaladva a hajtónyomaték csökken, a motor kiesik a szinkronizmusból.[18]

A szinkron motorokat csoportosíthatjuk a főmező létrehozásának módja szerint is. Ez lehet egyenárammal hajtott elektromágnes, vagy permanens mágnes (*PMSM*, permanent magnet synchronous motor). Permanens mágneses esetben a mágnes elhelyezhető a rotor

belsejében (*IPM*, interior permanent magnet), illetve felszínén (*SPM*, surface permanent magnet). [6][3]

Érdemes még megemlíteni a *PMSM* és *BLDC* (brushless DC, kefe nélküli egyenáramú) motorok közötti különbséget. A *BLDC* megtévesztő egyenáramú neve ellenére valójában egy elektromosan kommutált szinkron motor. Működési elvét tekintve megegyezik a *PMSM* motorokkal, de míg a *BLDC* tekercselése egyenletes, addig a *PMSM* szinuszosan osztott. Emiatt a forgórész által a tekercselésben indukált áram (*back-EMF*) az elsőnél trapéz alakú, a másodiknál viszont egyenletes szinusz (1.1. ábra). Emiatt meghajtani az elsőt trapézjellel (megfelelően kapcsolt/kommutált DC), míg a másodikat szinusszal kell (egyenletes nyomatékhoz). Mindenesetre a tekercselés miatt a nyomatékhullámosság a *BLDC*-nél mindenképp nagyobb lesz. A *BLDC*-k jelentősége a PMSM-el szemben elsősorban mind a hardveres (elég Hall-szenzor használata, nem kell pontos pozícióismeret) mind a szoftveres (egyszerűbb, kisebb számításigényű vezérlés) egyszerűségében rejlik, de mindkét típus használható térvektor alapú vezérléssel is (2 fejezet). [16]



1.2. ábra. *PMSM-BLDC különbsége.*

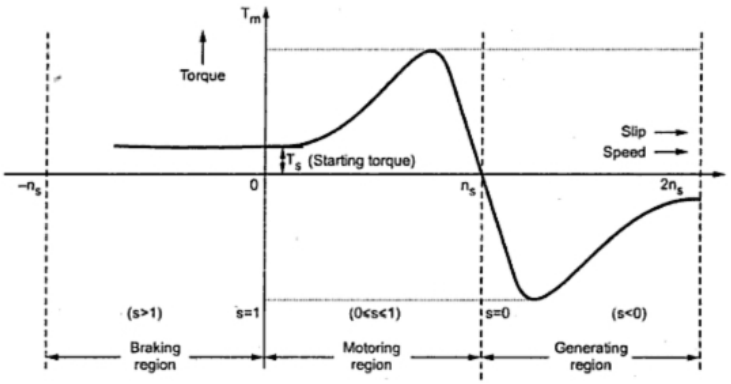
1.3. Aszinkron motorok

Az aszinkron motor állórészén is többfázisú tekercselés található, amely forgó mágneses mezőt hoz létre. A forgó mágneses tér erővonalai metszik a forgórész tekercselését, és abban feszültséget indukálnak. Mivel a tekercselés zárt, vagy rövidre zárt áramkört alkot, az abban indukálódott feszültség hatására a körben áram folyik. *Lenz-törvény*e értelmében az így indukált áram akadályozni igyekszik az őt létrehozó indukáló folyamatot, ezért a forgórész elfordul, így igyekezvén megakadályozni az erővonalmetszést, és vele az indukciót. Természetesen a forgórész soha nem érheti el az állórész forgó mágneses mezőjének értékét, mivel akkor megszűnne az erővonalmetszés. Ezt az elcsúszást nevezik *slip*nek.[18]

$$s = \frac{n_0 - n}{n_0} \tag{1.2}$$

Ahol s a slip, n_0 a forgó mező fordulatszáma, n pedig a forgórész fordulatszáma.

Mivel a fordulatszám és a slip között lineáris kapcsolat van, ezért könnyen ábrázolhatjuk a nyomatékot a slip/fordulat függvényében (1.3. ábra).[14]



1.3. ábra. *Nyomaték a slip/fordulatszám függvényében.*

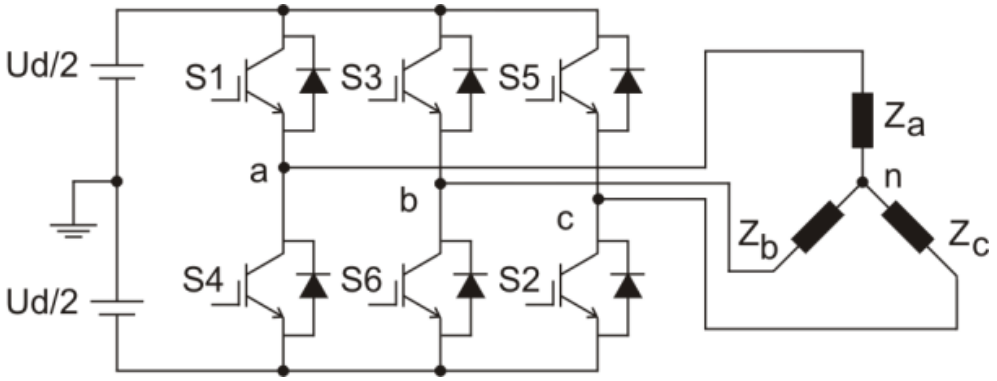
Kialakítás szempontjából 2 fő típusuk van:

- *Csúszógyűrűs:* A csúszógyűrűs aszinkron motor forgórészén a tekercselés végpontjai ki vannak vezetve egy-egy csúszógyűrűre. A forgórészek kivezetéseit közvetlenül, vagy ellenálláson keresztül rövidre zárják, ezzel az adott üzemállapotnak megfelelően tudják a fordulatszám-nyomaték jelleggörbét szabályozni.[17]
- *Kalickás:* Más néven rövidrezárt forgórészű. A forgórészen helyezzünk el vezető rudakat (réz, alumínium) és zárjuk a végeiket rövidre. Az így kialakított szerkezet mókusketrecre emlékeztet (innen az angol neve is: *squirrel-cage*) és zárt áramkörként működve áram indukálódhat benne.[17]

2. fejezet

Inverter és PWM

Az előző fejezetben ismertetett váltakozó áramú jel előállításához leggyakrabban kapcsolóüzemű DC-AC átalakítókat, invertereket használnak. 3 fázis esetén jellemző a 3 félhidas topológia. Ez a fajta inverter legegyszerűbb esetben 6 kapcsolható tranzisztorból, egy közös DC tápból áll (2.1. ábra). Az tranzisztorokat kapcsolóüzemben működtetjük az előállítani kívánt kimeneti jelalaknak megfelelően (nem kapcsolóüzemben túl nagy hőveszteség lépne fel a tranzisztorokon). A tranzisztorok általában szigetelt gate-ű bipolárisak (*IGBT*), vagy *teljesítmény MOS-FET*-ek. *IGBT*-k elsősorban a teljesítményelektronikában kerülnek alkalmazásra, míg a *teljesítmény MOS-FET*-ek akkor, ha kisebb teljesítményre, kisebb feszültségen van szükség különösképp, ha nagy kapcsolási frekvenciával. [12][15][12]

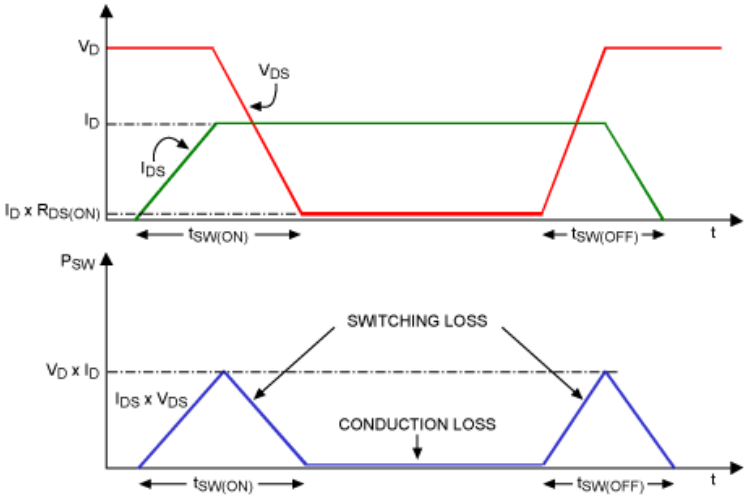


2.1. ábra. *3 fázisú 3 félhidas inverter.*

Sajnos a valós tranzisztorok nem ideális kapcsolóként működnek. Az alábbiakban ebből fakadó hatásokat részletezek.

2.1. Kapcsolási veszteségek

Bekapcsoláskor a kollektor-emitter (drain-source) feszültség csak teljes kollektoráram (draináram) felépülése után csökken le. Kikapcsoláskor ugyanez játszódik le a fordított irányban (a kollektoráram csak a kollektor-emitter feszültség felépülése után szűnik meg) (2.2. ábra). Emiatt kapcsolási veszteségek keletkeznek, ami hőként jelenik meg és kezelni kell az átmenet hőmérsékletének a megszabott határok között tartásához (*junction temperature*). Fontos megjegyezni, hogy az így keletkező hő egyenesen arányos a kapcsolási frekvenciával. Megfontolás tárgya lehet a kapcsolási frekvencia megválasztásánál a vezetési és kapcsolási veszteségek aránya. [12]



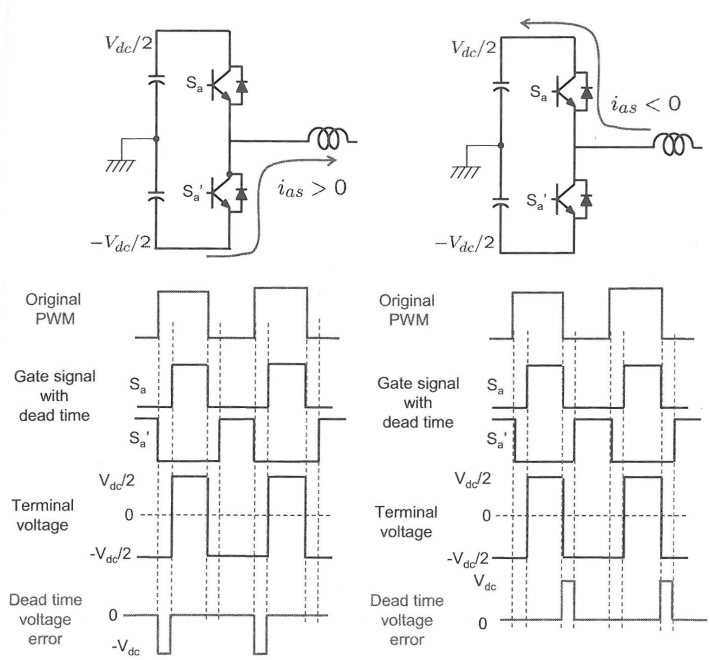
2.2. ábra. *Kapcsolási veszteségek szemléltetése.*

2.2. Holtidő és torzítása

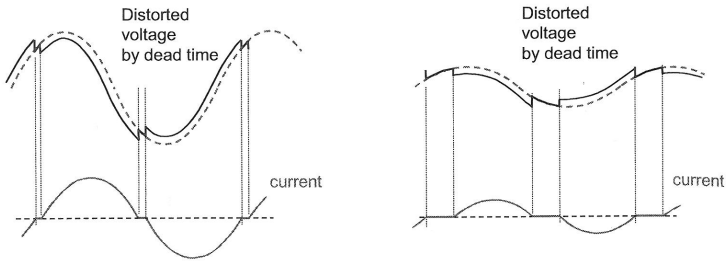
A 2.2. ábrának megfelelően látható, hogy a tranzisztor árama nem pillanatszerűen szűnik meg. Emiatt ha a vele egy ágban szereplő komplementer párját egy időben kapcsolnánk be akkor az adott ágon rövidzár keletkezne (*shoot-through* jelenség) Ezt a bekapcsolás késleltetésével lehet megakadályozni. [6][3]

Legtöbbször induktív terhelés (mint pl. motor) esetén diódával védjük a tranzisztort a feszültségugrás miatti árammal szemben. Emiatt holtidő esetén is a kimeneti kapcsokon a pozitív vagy negatív tápsínnek megfelelő feszültséget észlelünk (2.3. ábra).

Kis áramszinteknél a holtidő alatt a 2.4. ábrának megfelelően torzul a jel. Ezen torzítás kiküszöbölésére léteznek különböző megoldások, egyesek előzetes (a priori) ismereteket használnak a vezérlőjel beállítására, mások egyszerűen megmérik a félhíd kimenetét és ezzel szerzett információkkal korrigálják a vezérlőjelet.[12]



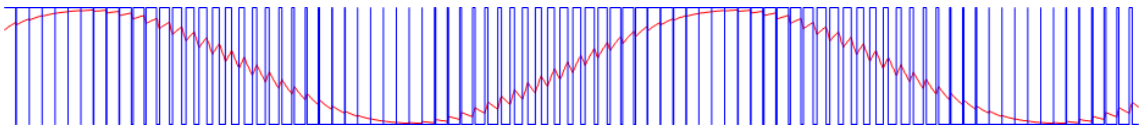
2.3. ábra. *Holtidő miatti hiba.*



2.4. ábra. *Holtidő okozta torzítás.*

2.3. Pulzusszélesség modulált jelek

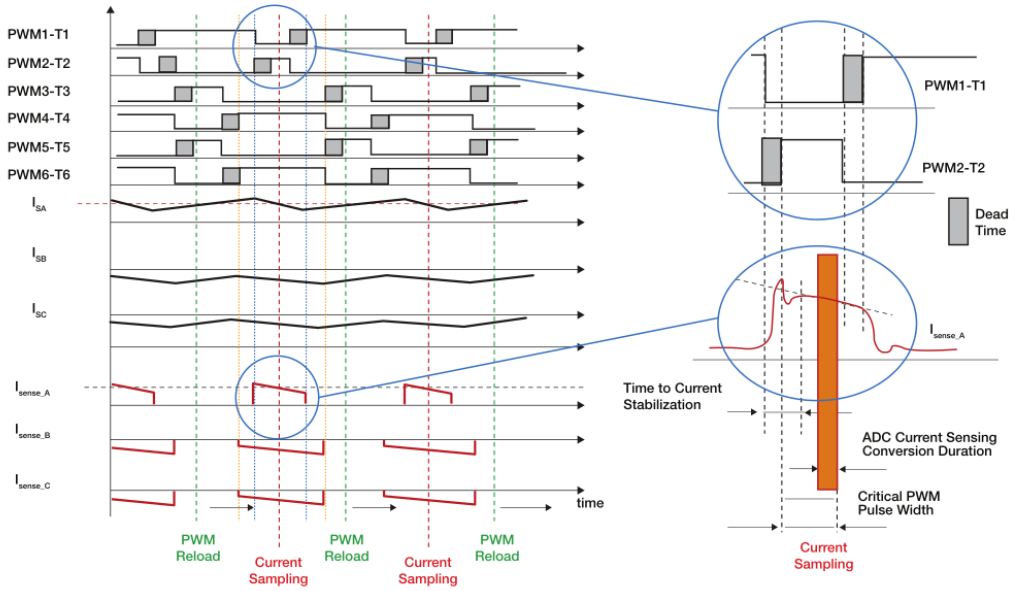
Pulzusszélességmodulált (PWM) jelek olyan kétállapotú periodikus jelek, ahol az információtartalmat a pulzus hossza hordozza. Ilyen jelekkel hajtva az inverter tranzisztorait a jel kitöltési tényezőjével beállíthatjuk a nyitás/zárási időket illetve időbeli elhelyezkedésüket (a pulzus egy perióduson belüli elhelyezésével). A nagy induktivitású terhelés miatt ezek szűrődnek így a PWM frekvenciát kellően nagyra választva tetszőleges áramot állíthatunk be a pólusokon, ezzel megszabva motortekercs mágnesező áramát. Például szinusztábla felhasználásával szinuszosat (2.5. ábra)[6]



2.5. ábra. *PWM jelekkel előállított szinusz.*

2.4. PWM vezérelt áramjelek mintavételezése

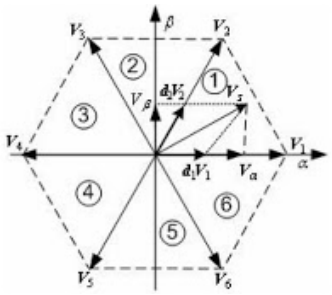
Amennyiben PWM vezérelt induktív terhelésen szűrődő áramjeleket mintavételezünk, érdemes a kialakuló jelformákat áttekinteni (2.6. ábra). Látható hogy az átlagérték méréséhez a meghajtási szakasz közepén kell mintát venni. Amennyiben ezek a szakaszok nem szimmetrikusak minden periódusban, márpedig nem lesznek azok, hiszen ők hordozzák a modulációs tartalmat (kivéve center aligned pulzusok használata esetén), akkor a kapott mintáink nem ekvidisztánsak és ezt kezelnünk kell (jellemzően szoftveresen)[3][6].



2.6. ábra. *PWM vezérelt áramjelek mintavételezése.*

2.5. Kapcsolóállások és elektromos vektorok

A hat tranzisztort összesen $2^6 = 64$ állásba kapcsolhatjuk. Ezek nagy része hibás (rövidre zárja az ágat) vagy felesleges. Mindenesetre 8 állás figyelemre érdemes, mert ha a kimeneteket 3 fázisú tekercsek pólusainak feleltetjük meg akkor vonali értékeket tekintve hat db egymástól 60°-ra elhelyezkedő vektort és két különböző nullvektort kapunk. Ezek a 2.1. ábrán láthatók.[12]



2.7. ábra. *Elektromos alapvektorok.*

V_i	A	B	C	V_{An}	V_{Bn}	V_{Cn}
V_0	0	0	0	0	0	0
V_1	1	0	0	$2V_{DC}/3$	$-V_{DC}/3$	$-V_{DC}/3$
V_2	1	1	0	$V_{DC}/3$	$V_{DC}/3$	$-2V_{DC}/3$
V_3	0	1	0	$-V_{DC}/3$	$2V_{DC}/3$	$-V_{DC}/3$
V_4	0	1	1	$-2V_{DC}/3$	$V_{DC}/3$	$V_{DC}/3$
V_5	0	0	1	$-V_{DC}/3$	$-V_{DC}/3$	$2V_{DC}/3$
V_6	1	0	1	$V_{DC}/3$	$-2V_{DC}/3$	$V_{DC}/3$
V_7	1	1	1	0	0	0

2.8. ábra. *Alapvektorok kapcsolóállás függvényében.*

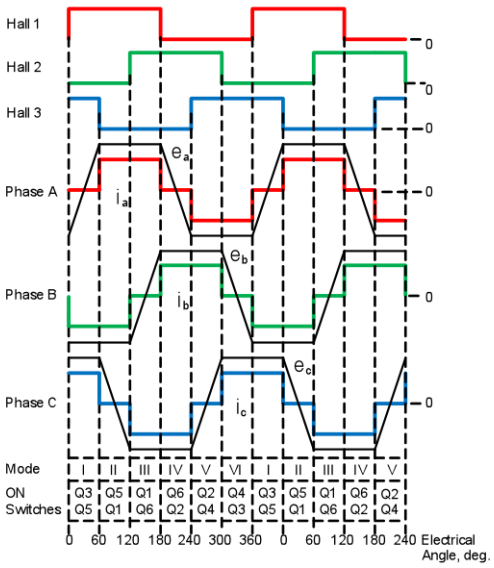
3. fejezet

Főbb kommutációs sémák és PWM mintáik

Az előző fejezetek alapján tekintsük át egy PWM vezérelt inverterrel elektromosan megvalósítható kommutációs lehetőségeket egy motoron.

3.1. Trapézkommutáció

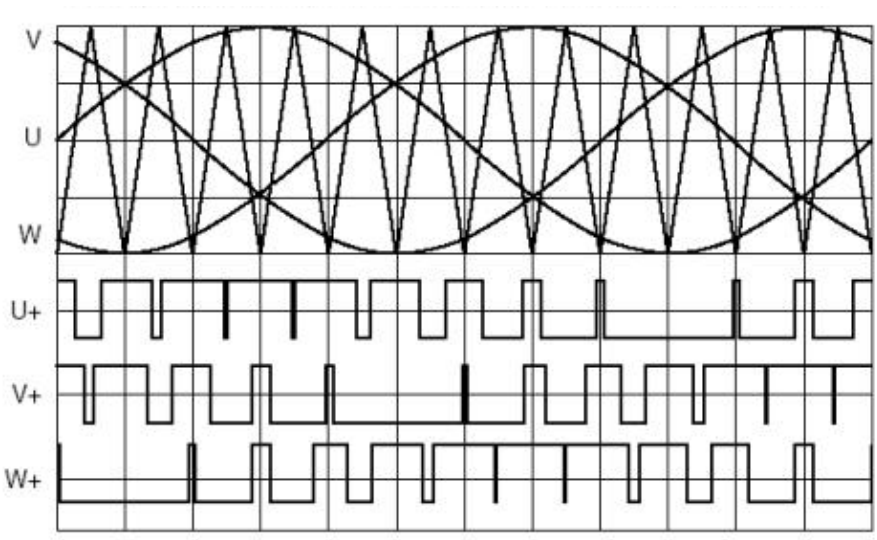
Trapézkommutáció során két tranzisztort kapcsolunk mindig be a rotorpozíció függvényében. Fontos, hogy nem szükséges a pontos pozícióismeret, elég tudni melyik két alapvektor között tartózkodunk. Ez az implementációt olcsóvá teszi, hiszen ennek érzékelése három 120°-al eltolt Hall szenzorral megoldható. További előnye az egyszerű, kis számításigényű algoritmus. Hátránya viszont a jelentős nyomatékhullámosság. Tipikusan *BLDC* motoroknál használható jól (hiszen ezek back-EMF jele is trapéz).[16][12] (3.1. ábra)



3.1. ábra. *Trapézkommutáció.*

3.2. Szinuszos kommutáció

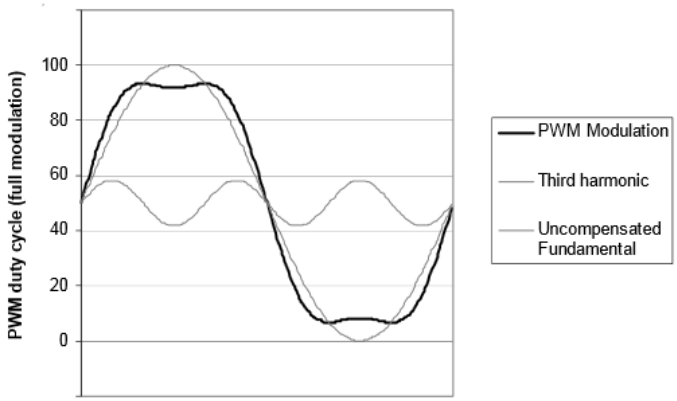
Szinuszos tekercselésű *PMSM* motoroknál érdemesebb trapéz helyett szinuszos jelalakot használni, ehhez viszont már a hat aktív vektor mindegyikét alkalmazni kell, továbbá nem elegendő a rotorhelyzetének hozzávetőleges ismerete, hanem pontos pozícióismeret szükséges (pl. inkrementális adó). Emiatt az implementáció bonyolódik és a költsége nő, viszont kiküszöböli a nyomaték ingadozást. Sajnos ez a módszer nagy nyomaték létrehozásához energetikailag nem hatékony. (3.2. ábra)[16][12]



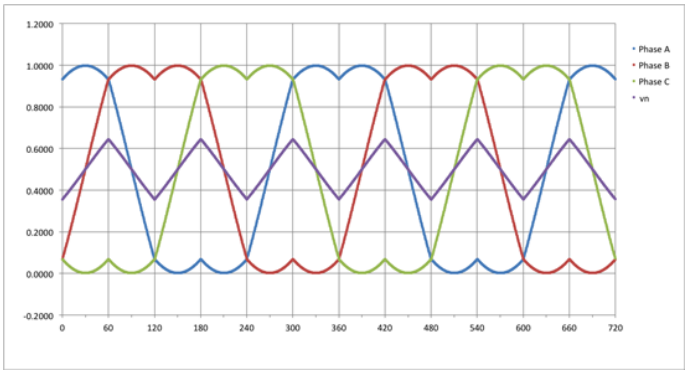
3.2. ábra. *Szinuszos kommutáció.*

3.2.1. Harmadik harmonikussal való moduláció

A szinuszos kommutáció hatékonyságának javítására szolgál a szinuszjel harmadik harmonikusával történő modulálása. Ez a 3 fázis esetén a 120°-os fáziskülönbségek miatt a fázisok között mérve kiejtik egymást, továbbra is szinuszos gerjesztést adva. Másképp megfogalmazva felfogható a csillagponti potenciál eltolásának, a jobb buszfeszültség kihasználása érdekében.[6]



3.3. ábra. *Harmadik harmonikussal moduláció.*



3.4. ábra. *Csillagponti feszültség változása.*

3.3. Mezőorientált vezérlés

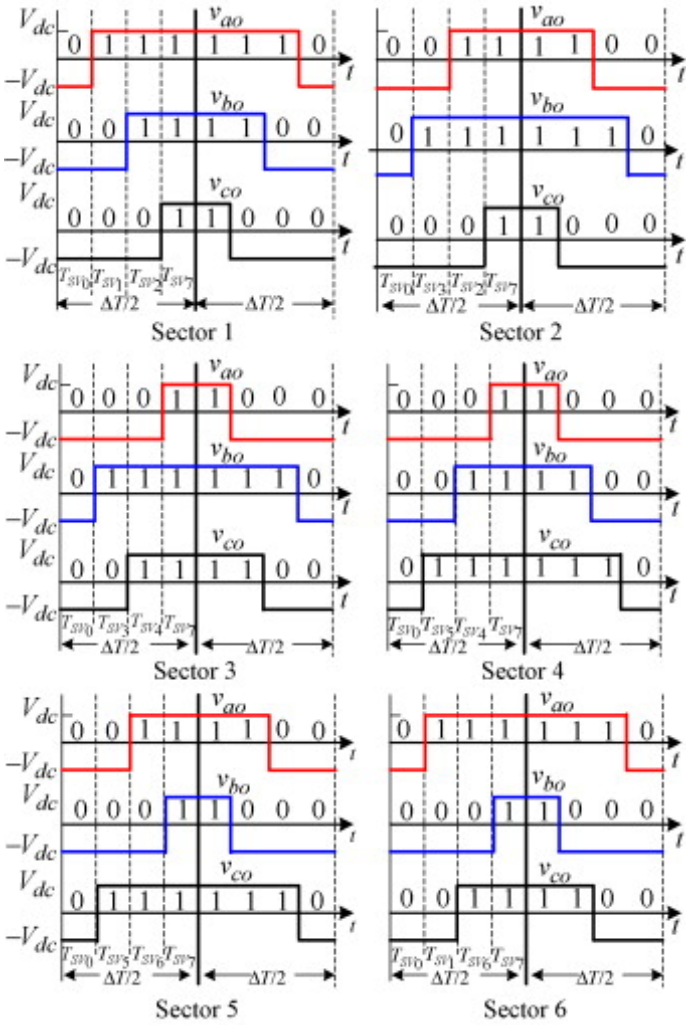
Mivel előállíthatunk tetszőleges nagyságú alapvektort, ezek lineáris kombinációjával bármely tetszőleges térvektort alkothatunk. Mezőorientált vezérlés esetén az aktuális rotorpozícióhoz tartozó optimális vektort kívánjuk kivezérelni[11]. Ezt a vektort egy ortogonális forgó koordináta-rendszerben két komponensből építjük fel (nyomatéki és mezőgyengítő, *field-weakening*). A háromfázisú rendszer leképzésének módjáról a 4 fejezetben lesz szó.

Ezen vezérlés hatékony, a buszfeszültséget jobban használja ki, mindemellett kiváló nyomaték és sebességkontrollt is biztosít. Hátránya a bonyolultabb, számításigényesebb algoritmus, valamint nem csak a pontos rotorpozíció ismerete szükséges a leképezésekhez, hanem a fázisáramok ismerete is.[16]

3.3.1. Mezőorientált vezérlés PWM mintái

Felmerül a kérdés, hogy az alapvektorokat milyen sorrendben vezéreljük ki a kimenetekre. Ennek jó megválasztásával a felharmonikus tartalmat (elektromágneses kompatibilitás) és a kapcsolási veszteségeket is csökkenthetjük. A legegyszerűbb módszer sorban kivezérelni az alapvektorokat és a maradékidőt zérusvektorral kitölteni. Ezen a talán legtöbb szakirodalom által bemutatott PWM minta ([12][3]) jelentősen javít. Mivel pozitív és negatív zérusvektorral is rendelkezünk ezért létrehozhatunk szimmetrikus (center aligned) PWM jeleket a kimeneten (3.5. ábra), a negatív zérus, aktív vektorok, pozitív zérus, illetve ennek a sorrendnek a megfordításával.[12][10]

Az efféle meghajtás előnye hogy megvalósítása egyszerű (egy fel-le számláló egyetlen értékére elég komparálni, fázisonként), valamint az áram átlagérték mintavételezése ekvidisztánsan történhet, (mindig a pozitív zérusú kivezérlés közepén, a számláló irányváltásakor).[13]



3.5. ábra. *Szimmetrikus PWM jelekkel SVPWM minta.*

Szakdolgozatomban nem ezek a mintákat alkalmaztam, hanem a [9]-nek megfelelőeket. Az implementáció viszont könnyen kibővíthető a mintagenerálás modularitása miatt egy ennek megfelelő mintával.

A [13]-nak megfelelően viszont szisztematikusan feloszthatjuk az alkalmazott mintákat. Az ezekre alkalmazott megnevezések nem egységesek ezért a citált forrást tekintem a továbbiakban hivatkozási alapnak.

3.3.2. Folytonos minták / Continuous reference function

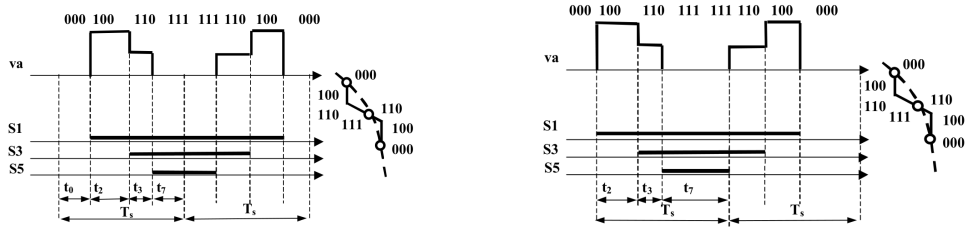
Folytonosnak tekinthetők, mert a kivezérelt minta minden kivezérlési periódusra alakilag egyezik, mindössze a szakaszok kitöltése változik a kivezérelendő vektor függvényében.

Direkt-Inverz / Direct-Inverse SVM

Az inverterkapcsolások számának csökkentése érdekében válasszunk olyan kapcsolási sorrendet mellyel egy perióduson belül új alapvektorra váltáskor csak egy tranzisztorpár kap-

csolása szükséges. Ezután a megmaradó szabad választási lehetőségünk a két különböző zérusvektorunk arányának és alkalmazásának megválasztásában van (hiszen a teljes zérusokkal kitöltendő idő adott). Két szélsőséges eset:

- DIH: Mindkét nullvektort használjuk és minden kivezérlési periódusban egyenlően osztjuk meg őket.
- DIO: Egy kivezérlési periódusban csak egyféle nullvektort használunk. A használt nullvektor periódusonként alternál.



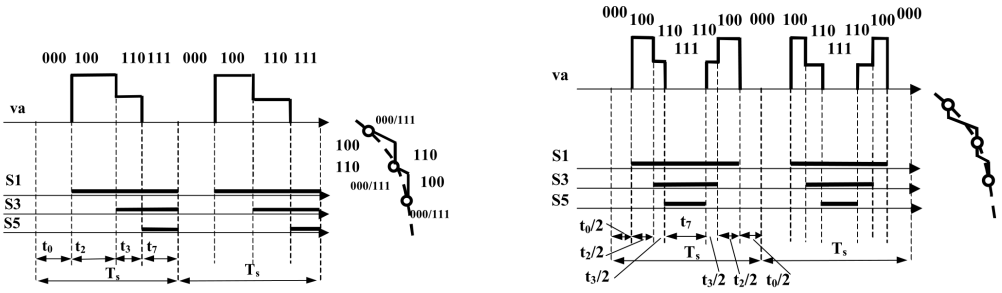
3.6. ábra. *DIH és DIO minta.*

Szimpla direkt / Simple Direct SVM

Egy egyszerű minta, minden periódus elején a tranzisztorokat kikapcsoljuk (negatív zérus), majd sorba kapcsoljuk vissza őket a pozitív zérus eléréséig.

Szimmetrikusan generált / Symetrically Generated SVM

Ez a legtöbb szakirodalom és 3.3.1 által ismertetett minta. A korábbi direkt-inverz alapján működik az alternáló direkt és inverz részek egy kivezérlési periódusba helyezésével. Bár komplikáltabb, ennek ellenére jelentős előnye hogy a legtöbb kereskedelmi forgalomban kapható szimmetrikus jelet generáló PWM IC-vel könnyen megvalósítható.

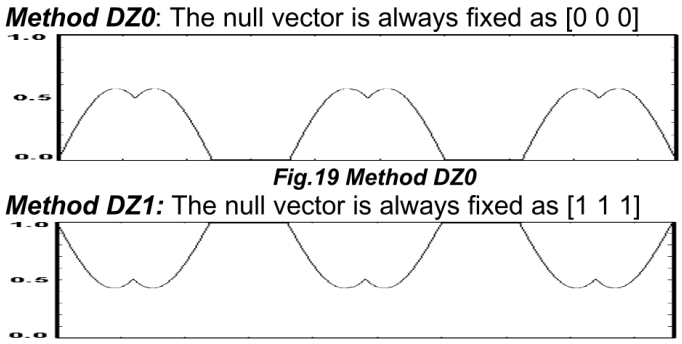


3.7. ábra. *Szimpla direkt és Szimmetrikusan generált minta.*

3.3.3. Nem-folytonos minták / Discontinuous reference function

Bármely két szomszédos alapvektor közötti váltáshoz egy tranzisztorpár kapcsolása szükséges. Ezt kihasználhatjuk a kapcsolások számának csökkentésére azáltal, hogy egy zérusvektort alkalmazunk minden egyes szektorra. Ez kétféle megvalósítást tesz lehetővé:

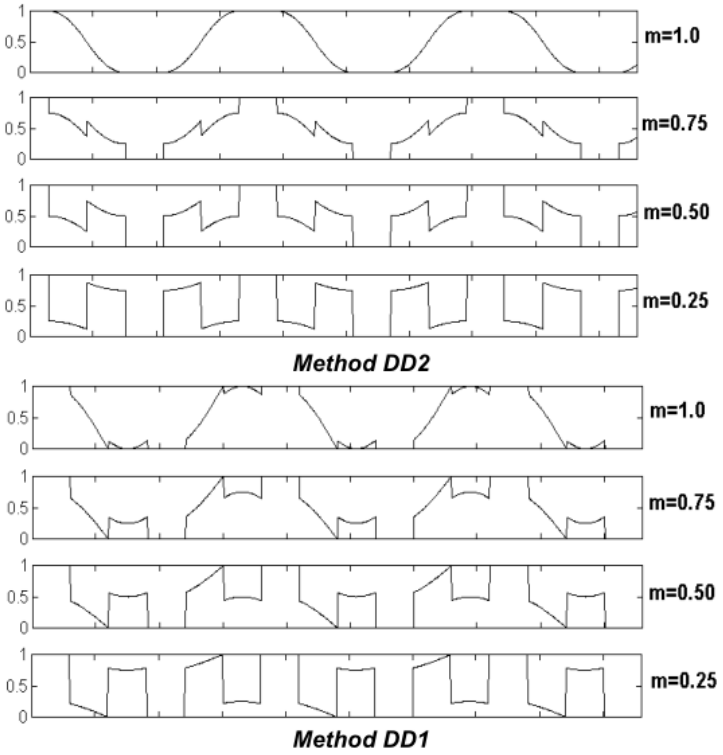
- DZ0: Negatív nullvektor használata.
- DZ1: Pozitív nullvektor használata.



3.8. ábra. *DZ0 és DZ1.*

Ezen megoldásnál a folytonos terhelés problémákat okozhat, hiszen bizonyos tranzisztorok folyamatosan bekapcsolt állapotban vannak. Mivel a kapcsolási veszteség arányos a tranzisztoron folyó árammal ezért kerülendő a legnagyobb áramot szállító tranzisztor kapcsolása.

- DD1: A pozitív zérusvektort használjuk 1,3,5-ös szektorokhoz, míg a negatívat 2,4,6-hoz. A fázisfeszültség maximuma utáni 60°-fokban nem kapcsoljuk a hozzátatozó tranzisztort. Mivel az áram késik a feszültséghez képest ezért a maximuma ebbe a tartományba fog esni. Ez a módszer a kapcsolási veszteségeket csökkenti.
- DD2: A 60°-os kapcsolásmentes állapotot a feszültség maximuma környékére választjuk. Ez a módszer a nem motormeghajtó átalakítókra jellemző, ahol az áram maximuma a feszültség maximumához közel esik (teljesítménytényezőjük közelít a egységnyihez).
- DD3: A mért áramcsúcs környékén nincs kapcsolás ±30°-ban. A módszer megfelelő működéséhez szükséges hogy a feszültségcsúcs is ebbe a tartományba essen.



3.9. ábra. *DD1 és DD2 különböző modulációs mélységekkel.*

3.4. Kommutációs módok összefoglalása

Kommutációs mód	Sebesség szabályzás	Nyomatékszabályzás		Szükséges visszacsatolás	Algoritmus komplexitás
		Kis sebesség	Nagy sebesség		
Trapéz	jó	nyomatékhullámosság	hatékony	durva pozícióismeret	egyszerű
Szinuszos	jó	hatékony	nem hatékony	pontos pozícióismeret	közepes
Mezővezérelt	jó	kiváló	kiváló	pontos pozícióismeret fázisáramok	bonyolult

4. fejezet

Matematikai eszköztár

Mielőtt egy általános térvektor alapú motorvezérlés felépítését megvizsgáljuk, érdemes két matematikai transzformációval és inverzeikkel megismerkedni. Ezek a Clarke és a Park transzformációk. Előfordulnak $\alpha\beta$- és dq-transzformáció néven is.

4.1. Clarke transzformáció

Nevét *Edith Clarke*-ról kapta. 3 fázisú rendszerek analízisét megkönnyítő transzformáció. Teljes alakja:

$$i_{\alpha\beta\gamma}(t) = T i_{abc}(t) = \frac{2}{3}\begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{bmatrix}\begin{pmatrix} i_a(t) \\ i_b(t) \\ i_c(t) \end{pmatrix} \tag{4.1}$$

Kiegyensúlyozott rendszerek (ahol $i_a(t) + i_b(t) + i_c(t) = 0$) esetén $i_\gamma(t) = 0$. Mivel a 3 fázisú motorjaink ilyenek (Kirchhoff törvény), így egyszerűbb alakba írható a transzformáció és az inverze[13]:

$$i_{\alpha\beta}(t) = \frac{2}{3}\begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix}\begin{pmatrix} i_a(t) \\ i_b(t) \\ i_c(t) \end{pmatrix} \tag{4.2}$$

$$i_{abc}(t) = \frac{3}{2}\begin{bmatrix} \frac{2}{3} & 0 \\ -\frac{1}{3} & \frac{\sqrt{3}}{3} \\ -\frac{1}{3} & -\frac{\sqrt{3}}{3} \end{bmatrix}\begin{pmatrix} i_\alpha(t) \\ i_\beta(t) \end{pmatrix} \tag{4.3}$$

Jelen alkalmazásban a jelentősége hogy 3 időfüggő vektorral leírt 3 fázisú rendszert a pillanatnyi értékeik alapján 2 vektorral leírhatunk [13].

4.2. Koordináta rendszer forgatás

Az alábbi összefüggés alapján térhetünk át kétdimenziós esetben egy, az eredeti koordináta-rendszerrel megegyező origójú, de θ-val elforgatott koordináta-rendszerbe[2].

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{4.4}$$

Szerepe abban rejlik hogy a forgó rotorunkhoz szeretnénk a koordináta-rendszerünket rögzíteni és ezt a fenti összefüggéssel tehetjük meg.

4.3. Park transzformáció

Nevét *Robert H. Park*-ról kapta. Más néven $dq0$ vagy dq (direkt-kvadratúra transzformáció). Gyakorlatilag a *Clarke*-transzformációra elvégzett vektoriális forgatás. Segítségével egy forgó koordináta-rendszerbe képezhetjük le a 3 gerjesztő vektorunkat 2 vektorrá [1].

$$i_{dq0} = K i_{abc} = \frac{2}{3} \begin{bmatrix} \cos\theta & \cos(\theta - \frac{2\pi}{3}) & \cos(\theta + \frac{2\pi}{3}) \\ -\sin\theta & -\sin(\theta - \frac{2\pi}{3}) & -\sin(\theta + \frac{2\pi}{3}) \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{bmatrix} \begin{pmatrix} i_a \\ i_b \\ i_c \end{pmatrix} \tag{4.5}$$

$$i_{abc} = K^{-1} i_{abc} = \frac{2}{3} \begin{bmatrix} \cos\theta & -\sin\theta & \frac{\sqrt{2}}{2} \\ \cos(\theta + \frac{2\pi}{3}) & -\sin(\theta + \frac{2\pi}{3}) & \frac{\sqrt{2}}{2} \\ \cos(\theta + \frac{2\pi}{3}) & -\sin(\theta + \frac{2\pi}{3}) & \frac{\sqrt{2}}{2} \end{bmatrix} \begin{pmatrix} i_d \\ i_q \\ i_0 \end{pmatrix} \tag{4.6}$$

Mivel a *Clarke*-transzformációból származik, így szabályos 3 fázisú rendszerben a képlet és inverze is egyszerűsödik. Gyakorlatban sokszor külön valósítják meg a Clarke-transzformációra végzett forgatással.

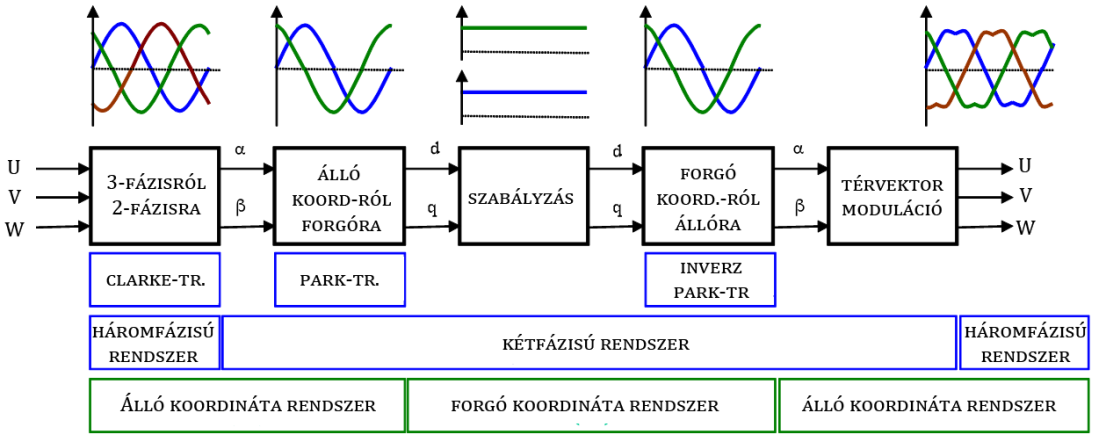
4.4. Jelentőségük

A rotorszög ismeretében fenti leképzésekkel az állandó nagyságú forgó vektorhoz szükséges 3 szinuszos gerjesztésünket, 2 ortogonális konstans vektorrá képezhetjük le. Ezáltal külön szabályozhatók, akár két egyszerű PI szabályzóval is. A két komponensnek ráadásul fizikai jelentése is van, i_q a nyomatékképző, i_d pedig a mezőgyengítő, így ezek szabályzása is egyszerűbben, külön kezelhető. Gyakorlati jelentősége abban is látszik a fenti transzformációknak és szétválasztásuknak, hogy sok szakirodalom magára a vektoriális forgatásra utal Park-transzformációként.

5. fejezet

Mezőorientált motorvezérlés

A váltóáramú motorok nyomatéka a mágnesező áram és a fluxus vektoriális szorzatával arányos. Ennek megfelelően ortogonális vektorok esetén maximalizálható. A szabályos 3 fázisú rendszernek két szabadsági foka van. Ezt a két szabadsági fokot az áram és a fluxus szabályozására használjuk fel. Ehhez szét kell választani őket, ami egy álló koordináta-rendszerben nem nyilvánvaló, de egy a rendszerrel szinkron forgóban egyértelműek (4 fejezet).[12]



5.1. ábra. *Térvektor alapú vezérlés transzformációi.*

Az előző fejezetek ismeretében egy nyomaték szabályzási kör összeállíthatóvá válik. Nézzük át a szabályozás lépéseit:

5.1. Nyomatékszabályzási kör

- *Gerjesztő áramok mérése:* Fázisonkénti áramok ismeretére van szükségünk a transzformációkhoz, de kettő gerjesztő áram mérése elég, hiszen a harmadik ezekből számítható, mert az összegüknek nullát kell adnia. (*Kirchoff-tv*)[6]
- *Mérjük meg a rotormező szögét:* Rotormezőszög ismeretében a kívánt kivezérelendő vektor szögét megállapíthatjuk (szinkronmotor esetében ez +90° a 1.1. ábra értelmében)

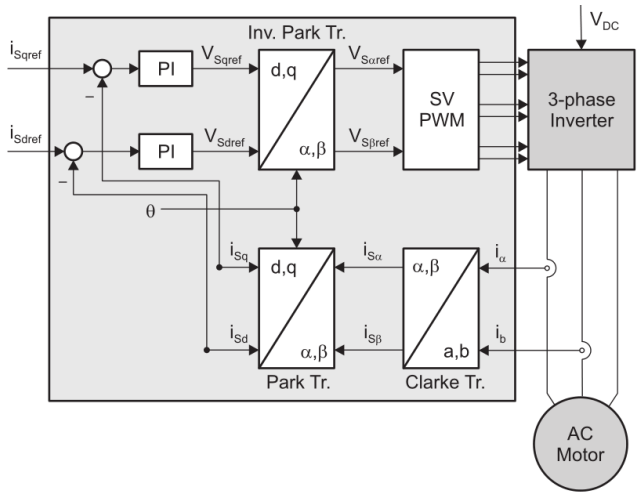
– *Transzformáció:* A transzformációkkal térjünk át *dq* térbe.

– *Hibajel generálása:* Hasonlítsuk össze a *dq* tér vektorait (nyomaték és mezőgyengítés) a kívántakkal, generáljuk a szabályzók hibajelét ebből.

– *Szabályzás:* Erősítsük a hibajeleket a beállított szabályzóparaméterekkel (PI / PID).

– *Visszatérés 3 fázisra:* Az inverz transzformációkkal térjünk vissza 3 fázisra.

– *Kivezérlés:* A kiszámított korrekciós feszültségeket moduláljuk a kimenetre.

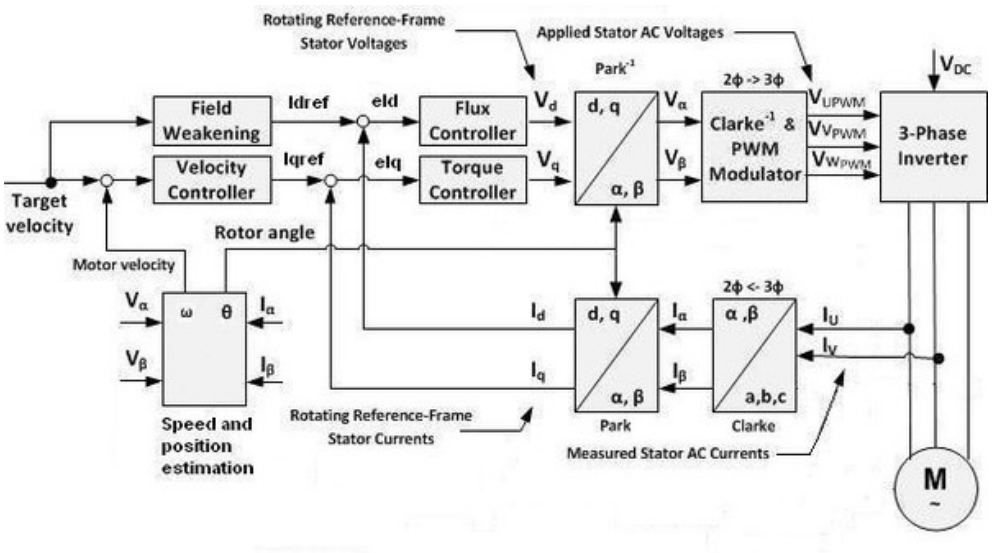


5.2. ábra. *Térvektor alapú nyomatékszabályzás.*

A rotormező szögének mérésében eltérés van szinkron, illetve aszinkron motor esetében. Szinkron motornál a rotor sebessége a mezejének sebességével egyezik meg, így pozíciószenzorral vagy rotorsebesség integráljával megállapítható. Aszinkron motornál viszont ez nem igaz. A mező szögének meghatározása kétfelé bontható: direkt és indirekt orientálásra. Direkt orientáció esetén igyekszünk a rotormező sebességét direkt módon mérni (pl.: légrésben elhelyezett szenzorokkal). Indirekt orientációnál nem mérjük a mező szögét, lévén nehézkes, helyette inkább a rotor szögéből és slipjéből számítjuk.[7]

5.2. Sebességszabályzási kör

Mivel a szögsebesség a szögpozíció első deriváltja, ezért a nyomtékszabályzási kör könnyen sebességszabályzásivá is bővíthető kaszkádosítással. (5.3. ábra) [7][8]



5.3. ábra. *Térvektor alapú kaszkádos sebesség és nyomatékszabályzás.*

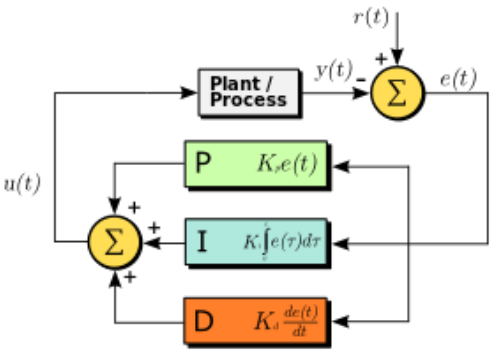
5.3. PI vagy PID szabályzó?

Ennek választása leginkább az alkalmazási céltól függ. PI méretezése és beállítása egyszerűbb, valamint természetéből fakadóan állandósult hiba mentes és stabil, viszont a beállási idő alatt integrált hibákat túllövéssel adja le. (Ezt a hatást az integrátor jelének maximumának megszabásával lehet csökkenteni). Differenciális tag hozzáadásával csillapíthatjuk a túllövést, a hosszabb beállási időért cserébe. PID szabályzó differenciális tagjának nullára választásával könnyen PI szabályzót nyerhetünk. [6]

PI és valós PID szabályzó átviteli függvénye:

$$W_{PI}(s) = A_p(1 + \frac{1}{sT_i}) \tag{5.1}$$

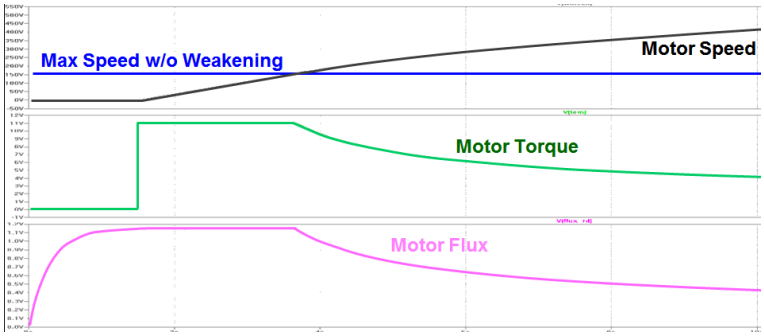
$$W_{PID}(s) = A_p(1 + \frac{1}{sT_i} + \frac{sT_d}{sT_c + 1}) \tag{5.2}$$



5.4. ábra. *PID szabályzó blokkvázlata.*

5.4. Mezőgyengítés

Nyomaték (i_q) növelésével növelhetjük a motor sebességét. A rotor forgásának következtében az állórész tekercseiben indukálódó feszültség nő ($\frac{d\Phi}{dt}$, ahol Φ állandó és dt csökken). A maximális kapocsfeszültség (tápfeszültség) elérése után tovább nem nőhet, itt van a motornak a nominális sebessége. A Φ tagot (légrés fluxusát) csökkenthetjük, hiszen mi állítjuk be (i_d-vel), ezzel az indukált feszültség is csökken, így kisebb érhető elő, azaz a sebesség növelhető, a nyomaték kárára. Ezt hívják mezőgyengítésnek. (5.5. ábra). Megjegyzendő, hogy a rotor által a státorban indukált áram nem káros, hiszen korlátozza a motor tekercseit terhelő áramot.[6][4]



5.5. ábra. *Mezőgyengítés hatása.*

6. fejezet

A megvalósítandó feladat

Feladatom egy a szabályzó kimeneti jeleiből a háromfázisú inverter vezérlőjeleit előállító modul megvalósítása volt FPGA-n. Ehhez egy Xilinx *Spartan3E-100* FPGA-t kaptam *BASYS2* próbapanelen. Maga az FPGA a családjának legkisebb tagja (mindössze 960 slice-al és 4 dedikált szorzóval), ez némi problémákat okozott, de hatékonyabb implementálásra serkentett.

A feladathoz a specifikációt a ThyssenKrupp féle Component Design for Inverter utasítás szolgáltatta[9]. Emiatt a fontosabb megkötések a következők voltak:

- A kivezérelendő vektor nagyságától függően különböző mintákat (3ASx, 6AS, 3AS0) kell alkalmazni, teljesítményoptimalizálás miatt.
- A mintaváltásnak hiszterézissel kell rendelkeznie.
- Bármely minta esetén meg lehessen mérni mindhárom fázis áramát (V_{min} paraméternyi ideig egy kivezérlési periódus alatt legyen minden fázis meghajtva).
- Holtidő megvalósítása.
- Hibajelzés megvalósítása, valótlan bemeneti paraméterek esetén.

A kivezérelendő minták pontos leírását a [9] utasítás tartalmazta. Ezek a 2 fejezet-ben ismertetett minták variánsai.

6.1. 6AS

A hat aktív szektoros minta, a szinuszos mintán alapszik. Egy periódusa alatt mind a hat alapvektort kivezéreljük, zérusvektor esetén egyenlő arányban. Fázisvektoroknak megfelelően a hozzájuk tartozó szakaszt nyújtjuk, a többit arányosan csökkentjük. Mindhárom, az inverz Clarke transzformációval megkapott fázisvektort használjuk. Ezt a mintát használjuk akkor ha kicsi (< 0.36) a kivezérelendő vektor. Mivel ez a megoldás önmagában biztosítja minden fázison az áram megmérhetőségét, ezért nincs szükség V_{min}-nek megfelelő kompenzálásra. Az nullvektor nélküli meghajtás miatt nagy vektorokra nem hatékony. Ezt a mintát használjuk kis vektor esetén (< 0.36), ha a meghajtóáramkör a vezérlő által teszteletlen.

6.2. 3ASx

Három aktív szektoros minta, az α, β két legközelebbi alapvektorra történő leképezésén alapszik. V_{min}-nek paraméterezett ideig minden fázis meg van hajtva. A teljes kört 30°-onként 12 szektorra bontjuk. A kivezérelt mintasorrend az aktuális szektortól függ. Egymást követő két periódusban a mintasorrend változik. Ez felfogható úgy is, hogy páros és páratlan periódusban más mintákat vezérlünk ki. Nagy vektor esetén (> 0.36) alkalmazott minta.

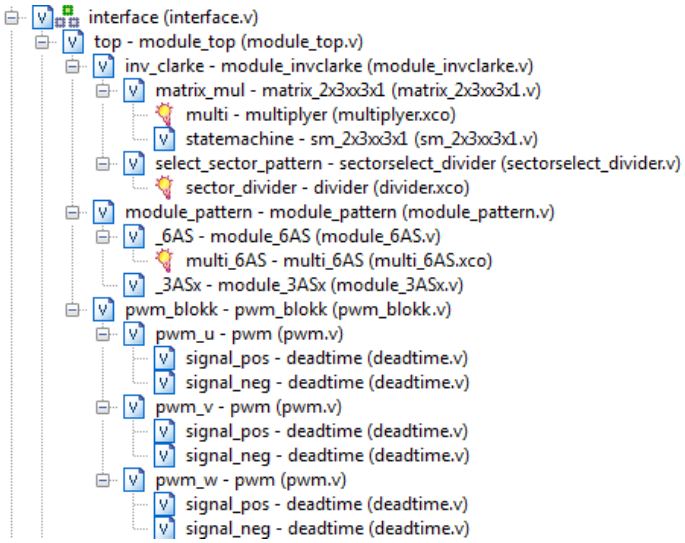
6.3. 3AS0

Három aktív szektoros minta, az α, β két legközelebbi alapvektorra történő leképezésén alapszik. V_{min}-nel paraméterezett ideig minden fázist meghajtunk, nullvektor esetén is, hogy az üzemkészséget ellenőrizni lehessen. Egyébként kihasználjuk a pozitív zérusú kivezérlést a folytonosabb PWM jelek biztosításához (ezzel is csökkentve a kapcsolási frekvenciát és emiatt a veszteségeket). A teljes kört 30°-onként 12 szektorra bontjuk. A kivezérelt mintasorrend az aktuális szektortól függ ([9] szerint). Ezt a módszert használjuk kis vektor esetén (< 0.36), ha a vezérlő tesztelte a meghajtóáramkört.

7. fejezet

Megvalósított modul

Ezen fejezetben a fentieknek megfelelő modult és almoduljaikat vettem sorba funkcionális működésük szerint. A modul hierarchikus felépítse a 7.1. ábrán látható.



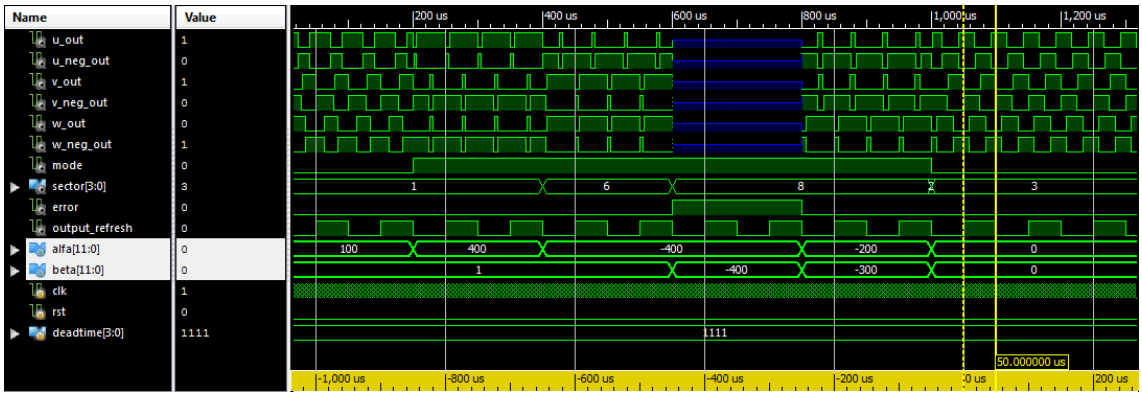
7.1. ábra. *Hierarchikus felépítés.*

A top modul feletti modul az interface, ő mindössze a kimenti és bemeneti jelek fogadásáért és kijuttatásáért felel. Bemeneti paramétereket (α, β, holtidő) a kapcsolókkal állíthatjuk be binárisan (*sw7*=MSB illetve előjelbit, *sw0*=LSB), majd a gombokkal frissíthetjük a modul paramétereit (*btn0*=, *btn1*=β, *btn2*=holtidő, *btn3*=reset). Ellenőrzéshez néhány a normál működés során nem szükséges jelet is kivezettem (*led3:0*=aktuális szektorérték binárisan, *led5*=3ASx (magas), vagy 6AS (alacsony), *JD header C12*=kimeneti frissítés jele, páros (magas) illetve páratlan (alacsony) periódusú kivezérlés vizsgálatához (gyakorlatilag egy 10kHZ-es jel). A három fázis kimeneti jele a *JA, JB, JC header*-ökről vehető le. Hibás állapot esetén ők nagyimpedanciás állapotba mennek, ezt a *led7* jelzi.

7.1. inverter_control

Gyakorlatilag a top modul, 12 biten, 2-es komplemensben, fixpontosan ábrázolt (2QN formátum: 1 bit előjel, 2 egészrész, 9 törtrész) α, β-ból előállítja a fázisoknak megfelelő PWM jeleket, 50 MHz-es órajellel, 4 bites deadtime-al paraméterezhető holtidővel (max. 320 nsec). Három funkcionálisan elkülöníthető almodulra tagolható, melyek az inverz Clarke transzformációért, az ebből történő mintagenerálásért és ennek a kivezérléséért felelnek.

```
module module_top(
        input [11:0] alfa,
        input [11:0] beta,
        output u_out,
        output v_out,
        output w_out,
        output u_neg_out,
        output v_neg_out,
        output w_neg_out,
        input clk,
        input rst,
        input [3:0] deadtime,
        output mode,
        output [3:0] sector,
        output error,
        output output_refresh
);
```



7.2. ábra. *inverter_control modul szimulációja.*

7.2. pwm_blokk

A kimeneti jeleket előállító *pwm* modulokat fogja össze és szinkronizálja.

A számított kivezérelendő jeleket leíró értékeket kapja meg bemenetnek, kimeneti jelei maguk a tranzisztorok holtidős vezérlőjelei és azok negáltjai. 20 kHz-es frekvenciával vezérli ki a kimeneteket. Bemeneti hibajelre a hozzá tartozó kivezérlési periódusra a kimeneteket nagyimpedanciás állapotba teszi.

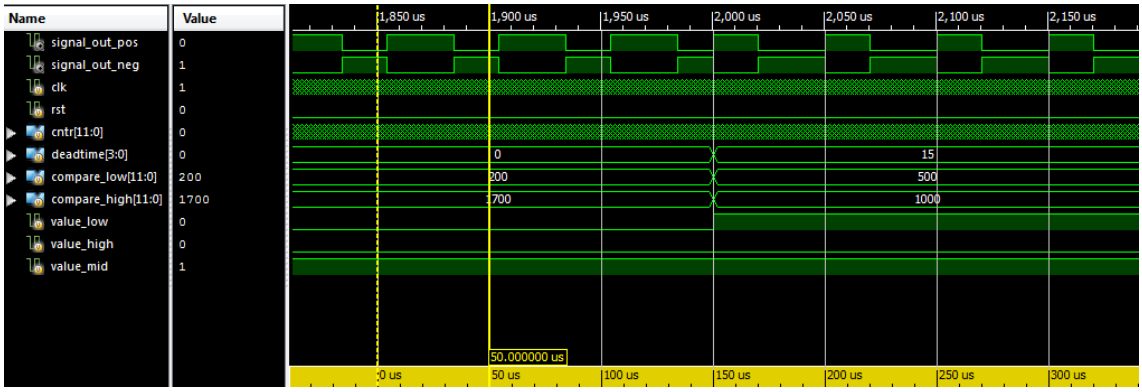
A modul 3 PWM jelet előállító almodult tartalmaz, ők állítják be fázisonként és órajelciklusonként a bemeneti paramétereik alapján, számlálóval történő komparálással a kimeneteket. A 20 kHz-es jelet egy órajelciklusonként léptetett 12 bites, 2499-ig számláló számláló állítja elő, ennek értékét mindegyik pwm almodul megkapja ezzel is biztosítva a szinkronitást. A kimeneti PWM jeleinek leírását ennek megfelelően fázisonként két 12 bites komparálási érték írja le. A hozzájuk tartozó *high, low, mid* értékek a különböző szakaszokon előírt kimeneti értékeket tartalmazzák.

```
module pwm_blokk(
                input u_value_high,
                input u_value_low,
                input u_value_mid,
                input v_value_high,
                input v_value_low,
                input v_value_mid,
                input w_value_high,
                input w_value_low,
                input w_value_mid,
                input [11:0] u_cmp_high,
                input [11:0] u_cmp_low,
                input [11:0] v_cmp_high,
                input [11:0] v_cmp_low,
                input [11:0] w_cmp_high,
                input [11:0] w_cmp_low,
                input clk,
                input rst,
                input [3:0] deadtime,
                output reg output_refresh,
                output u_out,
                output u_neg_out,
                output v_out,
                output v_neg_out,
                output w_out,
                output w_neg_out,
                input error_in
);
```

7.2.1. pwm

Fázisonkénti kimeneti jelek előállítása közös számlálóra történő komparálással. Almodulja a *deadtime* modul. Kimeneti jeleit rajta keresztül vezeti. Minden *pwm* modulnak két *deadtime* modulja van.

```
module pwm(
                input clk,
                input rst,
                input [11:0] cntr,
                input [3:0] deadtime,
                input [11:0] compare_low,
                input [11:0] compare_high,
                input value_low,
                input value_high,
                input value_mid,
                output signal_out_pos,
                output signal_out_neg
);
```

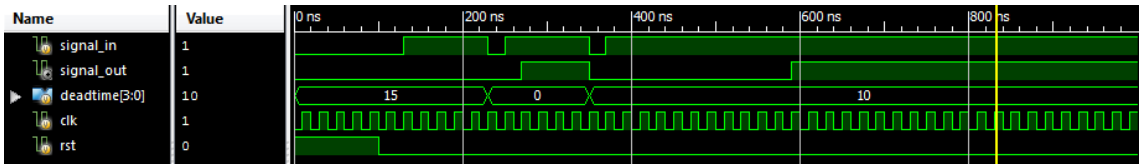


7.3. ábra. *pwm modul szimulációja.*

7.2.2. deadtime

Holtidőt beillesztő modul. Szigorúan szinkron módon felfutó élt detektál korábbi és aktuális érték komparálásával. Felfutó él esetén számlálót indít és letiltja a kimeneti jelet a 4 biten megadott holtidő eléréséig. Kimenete regiszterezett így közvetlenül kiköthető a hierarchikusan felette elhelyezett modulokon keresztül (ennek további hatása hogy minimálisan egy órajelciklusnyi holtidő beillesztődik).

```
module deadtime(
        input [3:0] deadtime,
        input signal_in,
        output signal_out,
        input clk,
        input rst
);
```



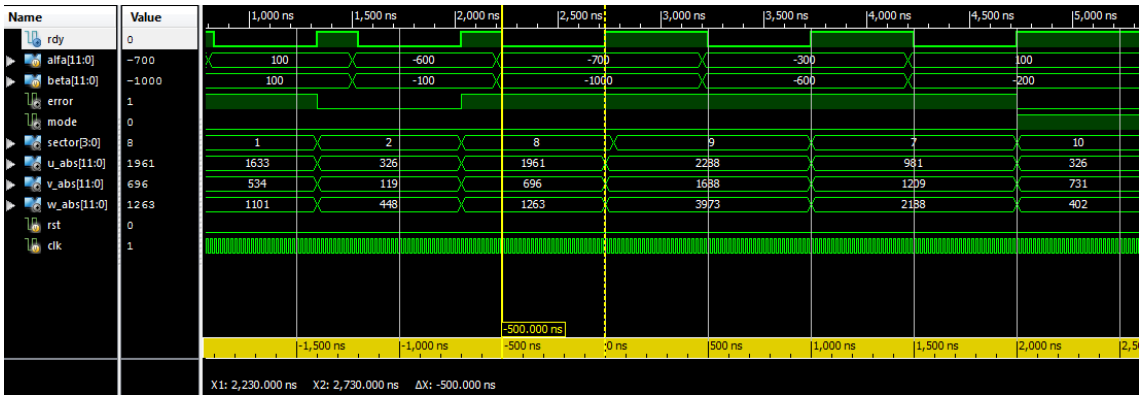
7.4. ábra. *deadtime modul szimulációja.*

7.3. inv_clarke

Bemenetei és kimenetei regiszterezettek, minden órajelciklusra mintavételezi bemeneteit. Amennyiben az előző mintával nem egyezik meg a bemenet a belső modulok számára *newdata* jelet generál. Nemcsak a Clarke transzformációért felel hanem, a szektordiszkriminációért és a bemeneti adatok helyességének ellenőrzéséért is. Ellenkező esetben hibajelet generál, amit a további modulok felé továbbít. A kimeneti regiszterezés szerepe, hogy a különböző sebességű párhuzamosan működő almodulok által szolgáltatott jelek szinkronban kerüljenek a kimenetére, konzisztensek legyenek (ne fordulhasson elő, hogy egy gyorsabb szektordiszkrimináció nem a hozzá tartozó inverz Clarke transzformált értékekkel együtt látszódik). Fontos megemlíteni hogy a kimeneti fázisértékek skálázottak a pwm komparátorhoz és abszolút-értékben értendők, lévén a [9] általi algoritmusok ilyen értékeket vesznek alapul (szemben az előjeles inverz Clarke transzformáció eredményével). Előjelbit elhagyásával és a Clarke transzformációba integrált skálázással továbbá LUT-okat spórolhatunk, ami egy ilyen kis FPGA-esetén előnyös. Almoduljai által párhuzamosan állapítja meg az aktuális szektort (*sector[3:0]*, 1-12 közötti érték), az alkalmazandó mintát a vektor hosszából (*mode*, 1'b0=6AS, 1'b1=3ASx), és végzi el az *inverz Clarke*-transzformációt mátrixszorzással.

A modul késleltetése 25 órajelciklus (50 MHz-en ez 500ns).

```
module module_invclarke(
        input signed [11:0] alfa,
        input signed [11:0] beta,
        input rst,
        input clk,
        output reg rdy,
        output reg [11:0] u_abs,
        output reg [11:0] v_abs,
        output reg [11:0] w_abs,
        output reg mode,
        output reg [3:0] sector,
        output reg error
);
```



7.5. ábra. *inv_ clarke modul szimulációja.*

7.3.1. matrix_2x3xx3x1

Egy általános $2x3$ és $1x3$-as mátrixok szorzására szolgáló modul. A kisméretű FPGA miatt a párhuzamosan végzett *inverz Clarke* transzformáció megvalósítása nehéz (6 szorzás és 3 összeadás). Emiatt egy dedikált szorzó felhasználásával, annak eredményének regiszterezésével egy állapotgéppel vezérelt időmultiplexált mátrixszorzót hoztam létre. A dedikált szorzó a *Xilinx IP Core* segítségével lett példányosítva. A szorzó kimenete csonkolva van a további *LUT* takarékosság jegyben. Továbbá mivel a kimeneti számlálók vezérlőjelének skálázása szorzást igényelne (lévén 50 MHZ-es órajelnél ekkora szószélességen nem lehetett kettő hatványán úgy 20kHz-re skálázni hogy shifteléssel beállítható legyen), ezért a skálázást is ez a szorzó modul végzi (a $2x3$-as *Clarke* tagokat tartalmazó mátrix elemeinek skálázásával).

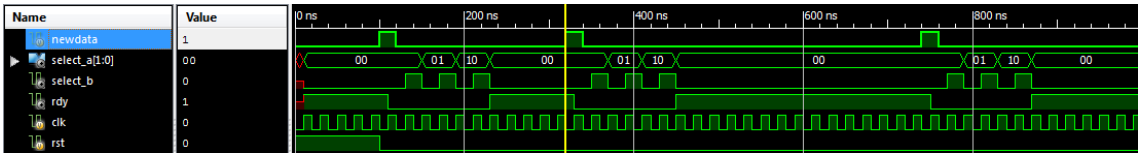
A szorzó vezérlését és az eredmények mentését egy állapotgép vezérli. Ennek kimeneti jelei alapján történik a mátrixelemek szorzó bemenetére történő multiplexálása és a kimeneti eredmények regiszterbe való mentése.

```
module matrix_2x3xx3x1(
        input signed [12:0] a11,
        input signed [12:0] a12,
        input signed [12:0] a21,
        input signed [12:0] a22,
        input signed [12:0] a31,
        input signed [12:0] a32,
        input signed [11:0] b1,
        input signed [11:0] b2,
        input clk,
        input rst,
        input newdata,
        output [11:0] out1_abs,
        output [11:0] out2_abs,
        output [11:0] out3_abs,
        output reg rdy
);
```

7.3.2. sm_2x3xx3x1

A mátrixszorzó bemenetére és a részeredményeket mentő regiszterek multiplexelését vezérlő jeleket előállító állapotgép, ami ennek megfelelően az egész *Clarke*-transzformáció elvégzését jelző *rdy* jelet is szolgáltatja.

```
module sm_2x3xx3x1(
    output reg [1:0] select_a,
    output reg select_b,
    input newdata,
    output reg rdy,
    input clk,
    input rst
                );
```



7.6. ábra. *állapotgép modul szimulációja.*

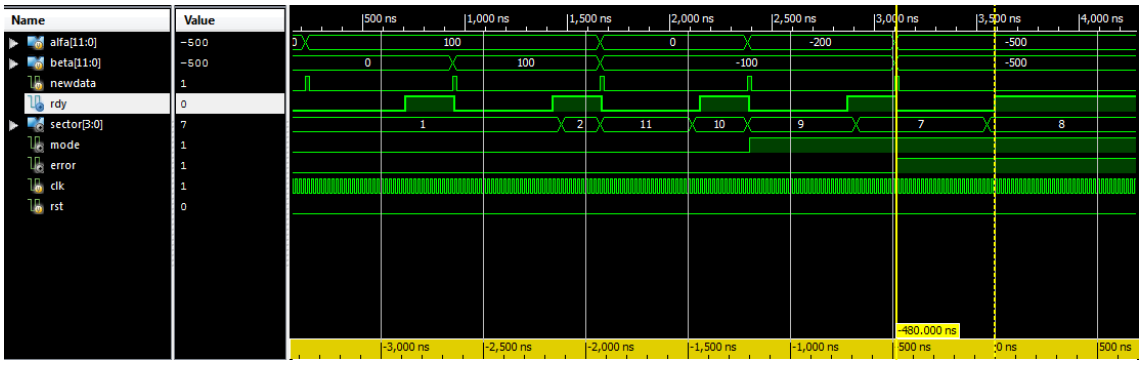
7.3.3. sectorselect_divider

Az α, β bemeneti értékeknek megfelelő szektor eldöntésért felelős modul. Mivel a pontos szöghelyzet meghatározásához arctan modul implementálása lenne szükséges, de ennek erőforrásfelhasználása az *FPGA*-hoz képest nagy (kb. 200 slice a 960-ból) ezért más megoldásra volt szükség. Mivel a pontos szögérték nem szükséges (hiszen az inverz *Park*-transzformáció elvégzése nem a modul feladata), és a 12 szektor 3-asával illeszkedik a koordinátarenszer 4 kvadránsába ezért előjelbitek alapján a kvadránst meghatározhatjuk, majd egy osztómodullal a 30° és 60°-ra komparálással a szektor kinyerhető. Az osztó *IP Core*-al van példányosítva és erőforrástakarékosság miatt lassúra választva (25 órajelciklusnyi késleltetés). Bár ezzel ő szabja meg a teljes *inv_clarke* modul sebességét, ez nem zavaró hiszen leggyorsabb esetben is 20 kHz-enként vezérelhetünk ki új mintát.

A modul további feladatai közé tartozik a szektordiszkriminációval párhuzamosan a bemeneti értékek ellenőrzése (nem kaptunk-e egységnél nagyobb vektort), valamint az eredő vektor hosszának megállapítása és ennek függvényében a mintaválasztás (*mode*). Az eredő vektor a specifikációban megadott maximális hosszának túllépése szintén hibát eredményez. A mintaválasztás hiszterézissel rendelkezik, a korábban kivezérelt minta alapján dönt az átmeneti tartományban.

Az eredő vektor hosszának megállapítása Pitagorasz tétel alapján történik α, β alapján. Az erőforrásigény csökkentéséhez itt is időmultiplexált dedikált szorzóval van megvalósítva és a továbbiakban a négyzetek összege alapján kezelve.

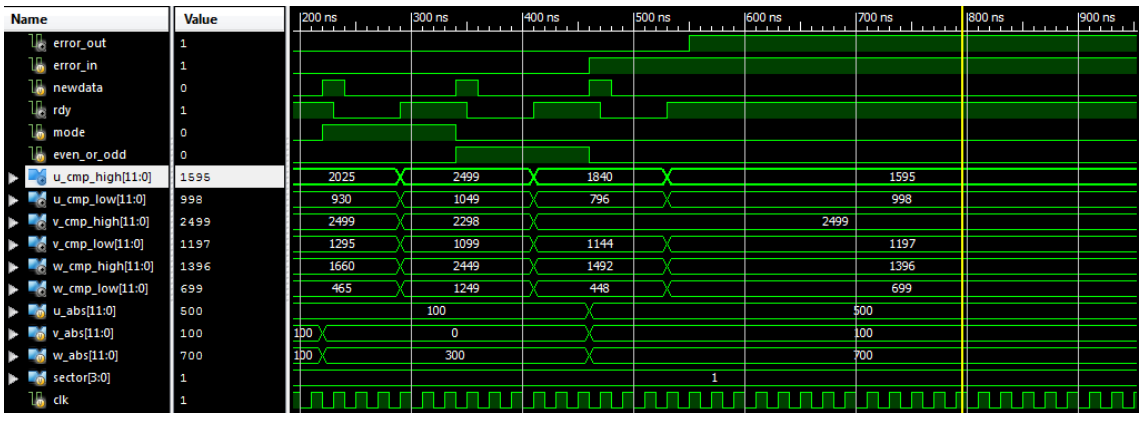
```
sectorselect_divider(
        input signed [11:0] alfa,
        input signed[11:0] beta,
        output reg [3:0] sector,
        input clk,
        input rst,
        output reg mode, //0=3AS0vagy6AS 1=3ASX
        input newdata,
        output reg error,
        output reg rdy
);
```



7.7. ábra. *sectorselect_ divider modul szimulációja.*

7.4. module_pattern

Az inverz *Clarke* transzformáció eredményéből a *mode* és az aktuális *sector* alapján a kivezérelendő mintának megfelelő pwm-számláló komparálási értékeinek, és a hozzájuk tartozó kivezérelendő értékek meghatározásáért felel, amit a *mode* és az *even_ or_ odd* jel alapján a kimenetre multiplexál. Az *error* jelet ezzel szinkronban terjeszti a konzisztencia biztosítás miatt. Az *even_ or_ odd* bemenete a kivezérlési periódus alapján a 3ASx-esetén kimultiplexált mintát határozza meg. Ezt a pwm modul biztosítja a kivezérlési periódus végének jelzésével (*output_ refresh* jel).



7.8. ábra. *mintakezelő modul szimulációja.*

```
        module module_pattern(
        input [11:0] u_abs,
        input [11:0] v_abs,
        input [11:0] w_abs,
        input [3:0] sector,
        input mode,
        input error_in,
        output reg error_out,
        input even_or_odd,
        input newdata,
        input clk,
        output reg rdy,
        output reg [11:0] u_cmp_high,
        output reg [11:0] u_cmp_low,
        output reg [11:0] v_cmp_high,
        output reg [11:0] v_cmp_low,
        output reg [11:0] w_cmp_high,
        output reg [11:0] w_cmp_low,
        output reg u_value_high,
        output reg u_value_mid,
        output reg u_value_low,
        output reg v_value_high,
        output reg v_value_mid,
        output reg v_value_low,
        output reg w_value_high,
        output reg w_value_mid,
        output reg w_value_low
);
```

7.4.1. modul_6AS

A 6AS mintát megvalósító modul. Mivel ez a minta zérusvektort a 6 alapvektor egyenletes kivezérlésével éri el, ezért a teljes periódus a *Clarke* transzformáció eredményeivel csökkentett fennmaradó részét egyenletesen kell szétosztani az alapvektorok között. Ez egy 6-os osztást eredményez, ami nem 2 hatvány lévén egy szorzót használ fel. Mivel volt szabad dedikált szorzó, emiatt egy konstans szorzótényezőjű (1/6)-al implementáltam és példányosítottam. A keletkező számítási hibákat kiküszöbölendően a V fázis felső komparálási értéke a számláló maximumával van egyeztetve ([9] alapján), így nem jelenhet meg tüske új kivezérlési periódus kezdetekor a műveletvégzés során keletkező numerikus hibák miatt.

```
module module_6AS(
        input signed [11:0] u_abs,
        input signed [11:0] v_abs,
        input signed [11:0] w_abs,
        input [3:0] sector,
        input clk,
        output [11:0] u_cmp_low,
        output [11:0] u_cmp_high,
        output u_value_low,
        output u_value_mid,
        output u_value_high,
        output [11:0] v_cmp_low,
        output [11:0] v_cmp_high,
        output v_value_low,
        output v_value_mid,
        output v_value_high,
        output [11:0] w_cmp_low,
        output [11:0] w_cmp_high,
        output w_value_low,
        output w_value_mid,
        output w_value_high
);
```

7.4.2. modul_3ASx

A 3ASx mintát megvalósító modul. Kiszámolja páros és páratlan kivezérlési periódusra is a komparálási értékeket, majd az *even_or_odd*, illetve a *sector* jel alapján multiplexálja kimenetére. Bár erőforrástakarékosabb lenne egyszerre csak az aktuálisra meghatározni, de átlagértéket akarunk mintaként venni az ADC-n, emiatt triggerelni a PWM-jel közepére kell. Mivel a páros-páratlan kivezérlésű minták a meghajtott fázist tekintve folytonosak, emiatt mindkettő ismerete szükséges a triggerjel időpontjának számításához. A V_{min}-el megszabható az a minimális idő ameddig minden fázist meg kívánunk hajtani ([9] általi kitétel). Továbbá mivel a 3ASx számítási algoritmusa a 2 legközelebbi alapvektorra való leképzést használja és a harmadikat ő maga állítja be, ezért a bementi 3 vektorról át kell 2 vektorra térni. Mivel ezek 120°-os szöget zárnak be, ezért a legkisebb megtalálásával egyszerű összeadással és kivonással megoldható, így nem szükséges külön transzformációt használni a normál *Clarke* helyett.

```
module module_3ASx(
        input [11:0] u_abs,
        input [11:0] v_abs,
        input [11:0] w_abs,
        input [3:0] sector,
        input clk,
        input rst,
        input even_or_odd,
        input [10:0] Vmin,
        output reg [11:0] u_cmp_low,
        output reg [11:0] u_cmp_high,
        output reg u_value_low,
        output reg u_value_mid,
        output reg u_value_high,
        output reg [11:0] v_cmp_low,
        output reg [11:0] v_cmp_high,
        output reg v_value_low,
        output reg v_value_mid,
        output reg v_value_high,
        output reg [11:0] w_cmp_low,
        output reg [11:0] w_cmp_high,
        output reg w_value_low,
        output reg w_value_mid,
        output reg w_value_high
);
```

7.4.3. modul_3AS0

A 3AS0 mintát megvalósító modul. A *sector* és a kisebbik kompenzált aktív vektor V_{min}-el való komparálása alapján multiplexálja a 3AS0 mintákat a kimeneteire. A V_{min}-el megszabható az a minimális idő ameddig minden fázist meg kívánunk hajtani ([9] általi kitétel). Továbbá mivel a 3AS0 számítási algoritmusa a 2 legközelebbi alapvektorra való leképzést használja és a harmadikat ő maga állítja be, ezért a bementi 3 vektorról át kell 2 vektorra térni. Mivel ezek 120°-os szöget zárnak be, ezért a legkisebb megtalálásával egyszerű összeadással és kivonással megoldható, így nem szükséges külön transzformációt használni a normál *Clarke* helyett.

```
module module_3ASx(
        input [11:0] u_abs,
        input [11:0] v_abs,
        input [11:0] w_abs,
        input [3:0] sector,
        input clk,
        input rst,
        input [10:0] Vmin,
        output reg [11:0] u_cmp_low,
        output reg [11:0] u_cmp_high,
        output reg u_value_low,
        output reg u_value_mid,
        output reg u_value_high,
        output reg [11:0] v_cmp_low,
        output reg [11:0] v_cmp_high,
        output reg v_value_low,
        output reg v_value_mid,
        output reg v_value_high,
        output reg [11:0] w_cmp_low,
        output reg [11:0] w_cmp_high,
        output reg w_value_low,
        output reg w_value_mid,
        output reg w_value_high
);
```

8. fejezet

Konklúzió

A megvalósítás során igyekeztem szem előtt tartani, hogy a modul egy nagyobb rendszer részét képezi. Lévén a *Clarke* és *Park* transzformációk és inverzeik lényegében mátrixszorzások, ezért először egy általános mátrixok méretével paraméterezhető szorzót valósítottam meg. Ezt az órajelhez képesti lassú kivezérlési frekvencia miatt időosztásban lehetett volna üzemeltetni állapotgéppel a teljes szabályozási kör transzformációinak elvégzéséhez. Miután az FPGA mérete miatt később erőforráshiányt tapasztaltam, ezt elvetettem és csak egy specifikus a korábban ismertetett transzformációs modult implementáltam.

Egy teljes szabályzási körben trigonometrikus modul lenne szükséges a forgó-álló koordináta-rendszerekbe való áttéréshez, viszont ez azzal is járna, hogy a szektordiszkriminációs modul elhagyhatóvá válik (bár ennek erőforrásigénye még így is nagyobb).

Szerettem volna továbbá a kivezérlési periódust szabadon megválaszthatóvá, a feldolgozási adatszélességtől függetlenné tenni. Ezt implementáltam külön órajel segítségével, melynek beállításával a kivezérlési periódus megszabható lett volna. Ekkor nem lenne szükséges a skálázás és az inverz *Clarke* transzformációs mátrix elemeinek összemosása. Viszont így 2 külön órajeltartományt kellet volna kezelni. A *pwm_blokk* járt volna a megválasztható órajelről, míg a többi modul a rendszerórajelről. Bár a rendszer működött egy szintetizált külön órajellel (5-ös szorzású, 3-as osztású, 30 MHz-es jellel, 19.54 kHz-es kimeneti frissítéssel), mivel nincs kellő tapasztalatom külön órajeltartományok kezelésében és az idő fogytán volt, emiatt végül a szigorúan szinkron, közös rendszerórajeles megvalósítás mellett döntöttem.

A kimenet viszonylagosan jó felbontással állítható (20ns), ezért ha nem szükséges ekkora precizitás akkor az erőforrástakarékosságot szem előtt tartva az architektúra leskálázható (viszont ez nem hatékony addig amíg a skálázást és a Clarke transzformációt együtt valósítjuk meg). A skálázás elkülöníthető lenne egy külön konstansszorzó számára, viszont ekkor ezt is időmultiplexálva kéne üzemeltetni, mely tovább feldolgozási késleltetéseket okozna.

A jó felbontás miatt, ahol a kerekítés nem volt külön erőforrás felhasználása nélkül megoldható egyszerű csonkolást használtam, hiszen az így bevitt numerikus hibák hatása kevésbé zavaró (szemben egy kisebb felbontású esettel).

Erőforráshiány miatt nem sikerült a 3AS0 modult beültetni a 6AS-el együtt, de működőképes és szabadon kicserélhető vele a 6AS (elvégre a feladatuk megegyezik, kis modulációs mélységű kivezérlés). Ugyanezen okból tettem le UART küldő-fogadó modul megvalósításáról (bemeneti paraméterek és lekérdezés lett volna a feladata).

Az ADC triggerelés szintén erőforrás hiányában nem lett megvalósítva, de a rendelkezésre álló minták szakaszainak hossza (3AS0, 6AS, 3ASx) alapján a PWM jelváltásokhoz komparálási értékeinél alkalmazott módon számolható és a szektorérték alapján a PWM modulnak átadható azért, hogy a számláló rájuk történő komparálásával ADC triggerpulzust adjon az árammméréshez.

Köszönetnyilvánítás

Ezúton is szeretném megköszönni Dr. Balogh Andrásnak (és rajta keresztül a ThyssenKrupp Prestának) és Dr. Sujbert Lászlónak (valamint rajta keresztül BME-MIT tanszék oktatóinak és dolgozóinak) a szakmai segítségüket, nem csak ezen téma kapcsán, hanem visszamenőleg is mind az önálló labororatórium, mind az elmúlt évek egyetemi tárgyai és feladatai során a nyújtott tudásért és a biztosított eszközökért. Mivel tisztában vagyok, hogy néha nem egyszerű velem együttműködni, ezért feltétlen köszönetet érdemel a nem szakmai szempontokat nézve a felém tanúsított türelmük és hozzáállásuk.

Irodalomjegyzék

[1] Christopher Laughman; Steven B. Leeb; Leslie K. Norford; Steven R. Shaw; Peter R. Armstrong. *A PARK TRANSFORM-BASED METHOD FOR CONDITION MONITORING OF THREE-PHASE ELECTROMECHANICAL SYSTEMS.*

[2] Analog Devices. *Reference Frame Conversions, AN21990-11*, 2002.

[3] Freescale. *Beyond Bits: Motor Control Edition.*

[4] A. LAGRIOUI H. MAHMOUDI. Flux-weakening control of permanent magnet synchronous machines. *Journal of Theoretical and Applied Information Technology*, 34(2):110–117, December 2011.

[5] Austin Hughes. *Electric Motors and Drives: Fundamentals, Types and Applications, Third edition.* Elsevier Ltd., 2006.

[6] Texas Instruments. *TI Motor Control Compendium*, 2010.

[7] Texas Instruments. *Sensored Field Oriented Control of 3-Phase Induction Motor, SPRABP8*, 2013.

[8] Texas Instruments. *Sensored Field Oriented Control of 3-Phase Permanent Magnet Synchronous Motors, SPRABQ2*, 2013.

[9] ThyssenKrupp Presta Hungary Kft. *Componenet Design for Inverter.*

[10] D.G. Holmes T.A. Lipo. *Pulse Width Modulation for Power Convertes*, 2003.

[11] Ned Mohan. *Advanced Electric Drives.* MNPERE, 2001.

[12] Kwang Hee Nam. *AC Motor Control and Electric Vehicle Applicatons.* CRC Press, 2010.

[13] Dorin O. Neacsu. Space vector modulation an introduction, 2001.

[14] J. Pollefliet. *Electric Power Control. Volume 2: Electric Motor Control.* Academia Press, 2012.

[15] Philips Semiconductors. *Power Semiconductor Applications, Chapter 3: Motor Control.*

[16] Ph.D. Shiyoung Lee. *A COMPARISON STUDY OF THE COMMUTATION METHODS FOR THE THREE-PHASE PERMANENT MAGNET BRUSHLESS DC MOTOR*, 1998.

[17] Dr. Halász Sándor. *Villamos hajtások.* Tankönyvkiadó, 1975.

[18] Malya János Dr Nagy Lorant Farkas Andras Kapolyi Zoltan. *Villamos gepek elmelete és gyakorlata.* Műszaki Kiadó, 2013.

Függelék

F.1. Mellékelt fájlok

1. Adatlapok: A felhasznált panel kapcsolási rajza, felhasznált *IP-Core* elemek adatlapjai, az *FPGA* adatlapja.
2. VerilogKodok: A végső modulok verilog kódjai.
3. Bitfile: A letöltendő bitfájl.
4. Reportok: Szintézis, időzítés, routolási reportok.
5. TeljesProjekt: A teljes ISE projekt, különböző testbenchek stb...