MISKOLCI EGYETEM
UNIVERSITY OF MISKOLC

A Miskolci Egyetem Habilitációs Füzetei

Tudományos munkásság áttekintő összefoglalása

Rekonfigurálható Rendszerek Alkalmazásai

Írta:

Vásárhelyi József,

aki az Informatikai Tudományok tudományágban
„dr. habil.” cím elnyerésére pályázik

Miskolc
2021

Tartalom

1 Bevezetés

Egyetemi pályámat 1992-ben kezdtem. Munkám során szoros együttműködés alakult ki a hazai és nemzetközi kutatókkal. Oktatási feladataim keretében ismerkedtem meg a programozható logikákkal (ismertebb néven FPGA – Field Programmable Gate Array áramkörök). Már a kezdetekkor 1992-ben részt vettem az említett áramkörök oktatásba történő bevezetésében. Akkor (1992-ben) a magyar egyetemek közül három egyetemen köztük a Miskolci Egyetemen is bevezetésre került az FPGA. Prof. Ajtonyi István biztatására kezdtem el oktatni az az FPGA áramkörök tervezését. 2017-ben a BME Méréstechnika és Információs Rendszerek Tanszékén szerény körülmények között ünnepeltük az FPGA oktatás XXV. évfordulóját.

Amikor a tanszéket felkérték a paksi reaktorvédelmi rendszer modellezésére, akkor kezdtük el ezen áramkörök alkalmazását a kutatásban is. PhD értekezésem témája is kapcsolódik az FPGA áramkörökhöz ennek címe „**RECONFIGURABLE ARCHITECTURES FOR VECTOR CONTROL STRUCTURES OF INDUCTION MOTOR DRIVES”.** Disszertációmban azt vizsgáltam, hogy milyen mezőorientált hajtásokat lehet megvalósítani a rekonfigurálható architektúrák segítségével? Doktori cselekményemet a Magyar Állam Ösztöndíj programja támogatta.

A PhD védésem (2004) óta eltelt időszakban az alábbi kutatásokkal foglalkoztam, a kutatási eredmények elhelyezhetők a hazai és nemzetközi kutatási eredmények sorában. Ezek a tématerületek a következőek:

I. PhD védés utáni kutatásaim:
   a) Váltóáramú motorhajtások rekonfigurálhatósága (reconfigure ability), mintegy folytatva PhD kutatásaim témáját.

II. Új kutatási területek
   a) Mojette transzformációk; Lehetőségem volt nemzetközi együttműködés keretein belül a kassai egyetem egyik prominens professzorával prof. Turán Jánossal együttműködni. Így általa vezetett kutatásokba kapcsolódtam be és a képfeldolgozás, vízjelezés területén alkalmaztam az FPGA áramköröket. Az együttműködés keretében két doktorandusz hallgatóval Serfőző Péterrel és Szoboszlai Péterrel végeztem közös munkát. A közös munka eredménye lett a Mojette transzformációk hardveres megvalósítása és az MTTOOL Mojette transzformációkat vizsgáló szoftver.
   b) Robot irányítások és helyzet meghatározás -- fuzzy interpoláció FIVE módszer vizsgálata Ezen a területen közös kutatómunkát végeztem Bartók Roland, Ahmed Bouzid és Ahmad Reda PhD hallgatókkal.
   c) A másik nagy téma a Moore törvénye révén került érdeklődésem központjába, mivel a processzorok jelenlegi felépítése révén inkább utasítás központúak, mint adatfeldolgozás központúak. Ezt a témát Drótos Dániel PhD hallgató közreműködésével vizsgálom.

Habilitációs füzetem felépítése a következő: A 4. fejezetben vázlatosan ismertetem a 2004-ben történt PhD disszertációm kutatási célkitűzéseit, eredményeit. A rövid ismertetés azért szükséges, hogy a fokozat megszerzésétől eltelt időszak tudományos eredményei elválaszthatók legyenek a korábbi eredményektől.

A tézisfüzet két részre bontható: I. A PhD Védés utáni kutatásaim és II. Új kutatási területek.

Az 5. Fejezet az tartalmazza azokat a kutatási eredményeket, amelyeket közvetlenül a PhD fokozat megszerzése után közöltem. Míg a 6, és további fejezetekben bemutatom azokat az eredményeket, amelyek az új kutatási területek közé soroltam: Mojette transzformációk (6) beágyazott rendszerek alkalmazásainak párhuzamosítása (7), helyzet meghatározás re-konfigurálható platformokon (8).

2 Introduction

I started my university career in 1992 and that is when I was involved in the field of Programmable Logic Circuits (PLA) as well. In the course of my work, I developed a close cooperation with national and international research partners such as Technical University of Kosice (SK), Technical University of Cluj (RO), University of Craiova (RO), Sapientia Hungarian University of Transylvania (RO), Leuphana University of Lüneburg (D), Xilinx Inc. (USA).

Based on my teaching assignments, I became acquainted with programmable logics (better known as FPGA - Field Programmable Gate Array circuits). At the encouragement of the late Prof. István Ajtonyi, I was one of the first in Hungary who introduced FPGA circuits in education together with the colleagues from Budapest University of Technology. Then, when the department was asked to model the reactor protection system in Paks, we started using these circuits in the research. The topic of my PhD dissertation was also related to FPGA and embedded systems. In my dissertation, entitled **"CONFIGURABLE ARCHITECTURES FOR VECTOR CONTROL STRUCTURE OF INDUCTION MOTOR DRIVES,"** I examined the reconfigurable control structures of field-oriented AC drives using FPGAs. The Hungarian State Scholarship Program supported my doctoral dissertation.

In the period since 2004, I have dealt with 4 areas of application. These topics are as follows: I investigated the reconfigure ability of AC motor drives, continuing the topic of my dissertation. Within the framework of open international cooperation, I had the opportunity to work with one of the prominent professors of the University of Košice. Prof. János Turán, who did significant research in the field of telecommunications. I became involved in the research he led, and thus used FPGA circuits in the field of image processing watermarking. Within the framework of the cooperation, I worked together with two doctoral students, Péter Serfőző and Péter Szoboszlai. The joint work resulted in the hardware implementation of the Mojette transformations and the MTTOOL Mojette transform testing software. During my teaching work, I became interested in robot technology and embedded systems. I did joint research in this field with PhD students: Roland Bartók, Ahmed Bouzid and Ahmad Reda. The other topic came to the forefront of my interest through Moore's Law. The current architecture of processors is more instruction-driven than data-centric. I study this topic with the participation of Dániel Drótos PhD student. In summary, the habitation booklet summarizes my research results such as:

- Reconfigurable tandem converter field-oriented drives;
- Application of Mojette transformations in telecommunication applications and shared data storage;
- Parallelization of embedded system applications
- Reconfigurability of behaviour-based robot controls
- Positioning of slow vehicles on a reconfigurable platform;

The structure of my habitation booklet is as follows: Chapter 5 contains the research antecedents; I outline the research objectives and results of my PhD defence in 2004. A brief description is necessary to distinguish the scientific results of the period since I obtained the degree from the previous results. In the 5 and subsequent chapters, I present the research results that I achieved in the period since I obtained the degree. Thus, chapter 5 includes the results I obtained immediately after obtaining the degree, still related to the results of the dissertation. Chapter 6 presents the results related to the Mojette transformation. While the chapters 7 and 8 deal with research in the field of robotics, more specifically interrupt handling in embedded systems and positioning on re-configurable platforms.

3 Rövidítések

ACAP	Advanced Computing Adatbale Platform Adaptív számítási egység
CSI	Current Source Inverter Áram inverter
FIVE	Fuzzy Interpolation in Vague Environment Fuzzy Interpoláció homályos környezetben
FPAA	Field Programmable Analogue Array Programozható analóg mátrix
FPGA	Field Programmbable Gate Array Programozható Logikai Kapumátrix
IIoT	Industrial Internet of Things Ipari Internetre csatlakoztatható egység
IoT	Internet of Things Internetre csatlakoztatható egység
ISR	Interrupt Service Routine Megszakítás kezelő rutin
MAF	Moving Average Filter Mozgó átlagoló szűrő
NOC	Network on Chip Hálózat a chipben
PAM	Pulse Amplitude Modulation Impulzus amplitúdó moduláció
PLD	Programmable Logic Devices Programozható Logikai áramkör
PWM	Pulse Width Modulation Impulzus szélesség moduláció
PWM-VSI	Pulse Width Modulation – Voltage Source Inverter Impulzus szélesség modulált feszültség inverter
RVA	Reconfigurable Voltage Attenuator Rekonfigurálható feszültség csillapító áramkör
SFC	Static Frequecny Converter
SMAF	Simple Moving Average Filter Egyszerű mozgó átlagoló szűrő
SOC	System on Chip Rendszer a chipben
VSI	Voltage Source Inverter Feszültség inverter
XADC	Xilinx Analogue to Digital Conveter Xilnx analóg digitális átalakító

4 Előzmények

A programozható logikai kapcsoló mátrixok vagy FPGA (Field Programmable Gate Array) áramkörök a megjelenésük óta az alkalmazott kutatások középpontjába kerültek. Az áramkörök 1980-as bevezetése óta évről-évre történő fejlődésük és az alkalmazási területek bővülése új kutatási területeket nyitott meg. Az egyszerű logikai elemeket tartalmazó és olcsó PLD (Programmable Logic Device) áramkörök egyre inkább a komplex, logikai erőforrásokban bővelkedő és drága áramkörökké fejlődtek (katonai és űrkutatás alkalmazások). Ugyanakkor az alkalmazási területek, amelyek a kezdetekkor még a kétszintű logikai hálózatok megvalósítására korlátozódtak, mára a digitalizáció korszakában a telekommunikáció, valós idejű beágyazott rendszerek, mesterséges intelligencia, képfeldolgozás, működés közben megvalósuló módosítható hardver (dinamikusan történő újra konfigurálás) területeken alkalmazzák ezen áramköröket. Egy vagy több magos processzor chipre történő integrálásával ezen áramköröket rendszer a chipen SOC (System on Chip) megoldásoknak nevezzük, amelyek felépítésükben FPGA és chipre integrált processzor elemeket tartalmaznak. Megemlítendő még a NOC (Network on Chip) „hálózat a chipen” és ACAP (Adaptive Computing Acceleration Platform) – adaptív feldolgozó egység, amely még a SOC rendszernél is fejlettebb programozható logikai áramkör, és tartalmaz egy NOC elemet is (A megjelenés éve 2020).

Ezen áramkörök az alkalmazott kutatásban történő felhasználását főleg az tette lehetővé, hogy a 1998-as évektől az FPGA áramkörök dinamikusan történő újra konfigurációja valósult meg az Algotronix által fejlesztett CAL architektúra révén, amely a későbbi Xilinx FPGA áramkörök alapját is képezte. Az áramkörök fejlődése lehetővé tette olyan kutatási területek megjelenését, amelyek megoldást jelenthetnek a Reiner Hartenstein által elnevezett Neumann szindrómára. A hardver módosítása a szoftver révén, mint re-konfigurálható számítógép került meghatározásra [1]. Lehetőségem volt részt venni több Dagsthul-ban (Németország) megrendezett nemzetközi workshopok-on (Dynamically Reconfigurable Architectures – 2000, 2003, 2006, 2010), ahol aktív résztvevőként hozzájárulhattam a terület fejlődéséhez.

Prof. Reiner Hartenstein, a 2001-ben közzétett cikkében [2] a re-konfigurálható számítógép paradigmáját antigépként határozza meg – anyag-ellenanyag páros, amennyiben a Neumann modell az „anyag”, mint paradigma. Ugyanakkor a szoftver – hardver re-konfiguráció (software-to-FPGA) párost mintegy négyszeres végrehajtási gyorsításként és disszipált teljesítmény csökkenésként említi. Megjegyezzük, hogy az FPGA-k fejlődése bizonyos mértékig a Moore által megfogalmazott technológiai törvényt is „felrúgta”, olyan értelemben, hogy nem kell 18-hónaponként az áramköri elemsűrűségnek duplázódnia ahhoz, hogy a feldolgozási sebesség lényegesen növekedjen. Az FPGA esetében az órajel frekvenciája lényegesen kisebb a mikroprocesszorok órajel frekvenciájához képest, viszont az algoritmusok párhuzamosításával a művelet végrehajtás sebessége lényegesen nagyobb. Azonban az algoritmusok párhuzamosíthatóságának is vannak korlátai (lásd [S10]).

Mit is nevezünk dinamikus konfigurációnak? Az újra konfigurálhatóság az (irányító) rendszer azon képességét jelenti, amelynek segítségével módosítja belső felépítését egy új „configware” kód betöltésével. A statikus konfiguráció megkülönbözteti a konfigurációs időben (közvetlenül a tápfeszültség bekapcsolása utáni) és a futási időben történő konfigurációt. A dinamikus konfiguráció a rendszer azon képessége, amelynek segítségével képes viselkedését megváltoztatni a környezetváltozásokra válaszolva (run-time re-configuration).

4.1 Kutatás módszertan

Közel három évtizede foglalkozom a fenti szakterülettel, az általam vezetett PhD hallgatók is sikeresen alkalmazzák kutatásaikban az SOC és FPGA áramköröket.

A ***témaválasztást*** illetően a PhD disszertációmban a mezőorientált hajtások re-konfigurálhatóságát és a re-konfigurációs tranzienseket vizsgáltam. A számítástechnika

fejlődésével egyre fontosabbá vált a feladatok valódi párhuzamos végrehajtása. Folyamatosan törekedtem új mások által még nem vizsgált alkalmazások és a megvalósítások algoritmusának párhuzamosítására. PhD védésem után bekapcsolódtam az Automatizálási tanszéken folyó kutatásokba is.
A ***szakirodalom*** elemzése alapján az algoritmusok párhuzamosíthatóságának vizsgálata alapján, modellezésre az alkalmazás megvalósíthatóságát hardver leíró nyelv (VHDL) és MATLAB SIMULINK SYSTEM GENERATOR-t választottam a vizsgálati módszerek eszközéül.
A módszer **validálására** az áramköri szimulációt (MATLAB Simulink, VSIM), a HIL módszert (Hardware in the Loop azaz szimuláció és hardveres vizsgálatot), a szükséges erőforrások vizsgálatát és a végrehajtási sebesség elemzését választottam.
Jelen fejezetben a 2004-ben megvédett PhD disszertáció alapján röviden ismertetem a megelőző kutatás célkitűzéseit. A értekezésem rövid áttekintésével bemutatom azokat az eredményeket, amelyeket a fokozat megszerzési folyamatában értem el.

4.2 A megelőző kutatás célkitűzései

A mikroprocesszor nagyon gyorsan az alkalmazott kutatások középpontjába került. Hamar kiderült azonban, hogy bizonyos területeken, mint például a váltóáramú Motorok irányítása ezen belül a jelfeldolgozás nehézkes és igen időigényes számításokat igényelnek. A jelprocesszorok megjelenésével, úgy tűnt, hogy megoldódott a jelfeldolgozás. Az egyik ilyen nehezen kezelhető számításigényes feladat a váltóáramú Motorok irányítása, azaz a mezőorientált hajtások magvalósítása.

4.2.1 Szakirodalmi áttekintés

A mezőorientált hajtások szabályozási struktúráit Kelemen Á. és Imecs M. írták le, előrevetítve a jelprocesszorokkal történő megvalósítást [3]. A DSP (jelfeldolgozó processzor) segítségével először Beierke ért el eredményeket a mezőorientált hajtás fix pontos DSP-n történő implementációjával [4]. Azonban a mintavételezési idő csökkentésével, ennek következtében a szabályozási kör számítási sebesség igénye is megnőtt. A hajtásszabályozási struktúrák bonyolultságának növekedésével a kutatások az FPGA áramkörök irányába fordultak (lásd: Cirstea [5], Monamason [6], Vásárhelyi [9], [10]).
Ebben a fejezetben, a 2004-ben történt PhD védésem kutatási célkitűzéseit, illetve azok eredményét foglalom röviden össze. Az elmúlt közel tizenöt év kutatási célkitűzései, illetve az elért eredmények összefoglalása a további fejezetekben (I, II, 7 és 8) ismertetem.

4.3 Általam végzett kutatások

Kutatási eredményeimet PhD dolgozatom alapján foglalom össze.
A re-konfiguráció kutatás korai szakaszában az új konfiguráció tervezése, fordítása párhuzamosan valósul meg az adatfeldolgozással. Ez nem kivitelezhető a dinamikus rendszerek esetében (lásd mező orientált hajtásszabályozás). Ezért a különböző konfigurációkat előre kell ismernünk.
PhD kutatásaim az dinamikusan újra konfigurálható mezőorientált hajtások szabályozási struktúráit vizsgáltam. A váltóáramú motorok mezőorientált hajtásszabályozási struktúrái esetében két esetben szükséges a re-konfiguráció megvalósítása:

I. eset: re-konfiguráció a szabályozás minőségének javítására.

II: eset: re-konfiguráció az üzembiztos működés fenntartására.

Re-konfigurálhatósági korlát: Figyelembe véve, hogy a váltóáramú motorok szabályozási struktúrája folytonos működést kell, biztosítson, a re-konfigurációnak a két mintavételezés közötti időben kell megvalósulnia.

$$\Delta t_r < T_{sampling} \quad (1)$$

ahol Δt_r a re-konfiguráció időtartama, $T_{sampling}$ a mintavételezési periódus és t_r a re-konfiguráció pillanata.
Amennyiben előre ismertek a hajtás szabályozási struktúrák, úgy a különböző szabályozási stratégiákra tekinthetünk úgy is mint egy-egy véges állapotú állapotgép állapotaira.

4.4 Új tudományos eredmények

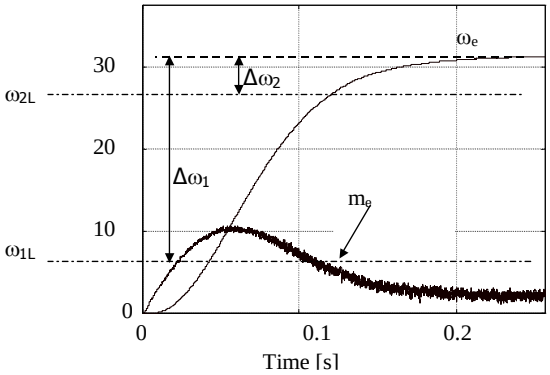
Új tudományos eredményeim a mezőorientált hajtások területén a különböző hajtás struktúrák viselkedésének vizsgálata a rekonfiguráció tükrében. Az eredményeket az alábbiakbn foglalom össze

4.4.1 I. eset: re-konfiguráció a szabályozás minőségének javítására.

Abban az esetben, ha a váltóáramú motor fordulatszámát szabályozzuk, akkor a referencia érték függvényében módosítható a szabályozási struktúra figyelembe véve a fordulatszám gyorsulásának előjelét .
Az egyes hajtásszabályozási struktúrákhoz hozzárendelünk egy-egy konfigurációt (*STATEi, STATEj*). A struktúra konfigurálási feltétele a motor fordulatszámának értéke, azaz egy viszonyítási értéknél a fordulatszám nagyobb/kisebb, mint a referencia érték. Ebben az esetben a re-konfiguráció a következő összefüggés alapján történik:

$$\begin{cases} \omega_{1L} = \omega_r^{\text{Ref}} - \text{sign}(\omega_r^{\text{Ref}})\Delta\omega_1 \\ \omega_{2L} = \omega_r^{\text{Ref}} - \text{sign}(\omega_r^{\text{Ref}})\Delta\omega_2; \end{cases} \quad (2)$$

ahol, a $\Delta\omega_1$ és $\Delta\omega_2$ pozitív küszöb érték és $\Delta\omega_1<\Delta\omega_2$, ω_{1L}, ω_{2L} pedig a referencia érték a konfiguráció váltáshoz. Induláskor a érvényes konfiguráció az első állapot (*State1),* majd amikor a Motor fordulatszáma eléri a referencia értéket ω_{2L} megtörténik a konfiguráció váltás (*State2*). A fordulatszám csökkenésekor pedig az ω_{1L} érték elérésénél a hajtás struktúra visszavált az indulási konfigurációba. re-konfiguráció hatásait az úgynevezett tandem konverteres táplálású indukciós motor hajtás szabályozási struktúrákon vizsgáltam, amelynek fordulatszám változását az 1. ábrán ábrázoltam.



1. ábra: Konfigurációs határértékek ábrázolása

Ebben az esetben a t_r re-konfiguráció kezdésének pillanata ismert és a megvalósított szabályozási struktúrák is ismertek voltak.

4.4.2 II. eset: re-konfiguráció az üzembiztos működés fenntartására.

A mező orientált hajtás üzembiztos működésének fenntartását a Maciejowski által meghatározott re-konfigurációs feltételek teljesítésére javasoltam módszert és vizsgáltam a tandem inverteres hajtások esetében a re-konfiguráció megvalósítását és annak hatásait. A Maciejowski kérdések a rendszer üzembiztos működésére és a rendszer folyamatos működésének fenntartására vonatkoznak [7]. A folyamatos és üzembiztos működés érdekében indokolt a re-konfiguráció. Ebben az esetben a re-konfiguráció t_r időpillanata nem ismert.

1. táblázat: Tandem inverter re-konfigurációs struktúrái:

CSI-táplált Motor	Tandem inverter/VSI táplált Motor		Description
(State 2)	(State 1)		Hivatkozott irodalom
Orientation Flux	Orientation Flux	PWM Method	References
Ψr	Ψr	SVM – space vector modulation	[13][14]
Ψr	Ψs	SVM	[15][16][17]
Ψr	Ψr	Áram visszacsatolt moduláció Current Feedback Modulation ("bang-bang")	[18]

Az értekezésben vizsgáltam különböző re-konfigurálható hajtás struktúrákat. A hajtás struktúrákat a forgórész (Ψr) és állórész (Ψs) fluxus és az impulzus szélesség moduláció (PWM) típusa szerint az 1. táblázat foglalja össze:

A különböző mezőorientált hajtás struktúrák vizsgálatához, létrehoztam egy FPGA elemkönyvtárat, amelynek segítségével összeállíthatók a hajtásszabályozások. Az elemkönyvtár segítségével elemeztem a re-konfiguráció következtében megjelenő tranzienseket. A tranziensek megjelenése nem függ a szabályozási struktúrától, hanem a re-konfigurációs időtől. Módszert javasoltam a tranziensek kezelésére.

A PhD cím elnyerése után is még néhány publikációban foglalkoztam a váltóáramú motor hajtások rekonfigurálhatóságának vizsgálatával. Így vizsgáltam a tandem invertert, amelyet Trzynadlowski írta le [8]. A tandem inverter tulajdonképpen egy áram inverter (CSI) és egy feszültség inverter (VSI) kombinációja. A következő fejezetben ezt a kutatási eredményt ismertetem.

I. A PhD Védés utáni kutatásaim

5 Rekonfigurálható mezőorientált hajtás

A PhD disszertációm védése után még foglalkoztam egy ideig a rekonfigurálható mezőorientált hajtások vizsgálatával. Jelen fejezet ezt a munkát mutatja be a [S1], [S2] és [S3] alapján.

5.1 Bevezetés – Szakirodalmi áttekintés

A jelen fejezetben tárgyalt tandem inverterrel a szakirodalomban 2006.ig a következő szerzők foglalkoztak: Trzynadlowski és társai [8], majd Imecs Mária és társai elemezték mélyrehatóan a tandem invertert.

Az általam végzett kutatásokban vizsgáltam a tandem inverter rekonfigurálhatóságát, a rekonfiguárció hatását a szabályzó rendszerre és a váltóáramú motorra. A rekonfigurálható tandem inverter esetében, a rekonfigurációt a hibamentes működés és a szabályzó teljesítményének javítása érdekében lehet használni.

Az újrakonfigurálás kezdetét meghatározó feltételek különbözőek, és az adott alkalmazás függvényében valósulnak meg. A váltóáramú motorok mezőorientált hajtás szabályozásának szükségessége azon gyakorlati megfigyeléseken alapul, hogy a különböző típusú mezőorientált hajtások teljesítménye eltérő. Ezek a különbségek a sebesség tartományától, a mechanikai terhelési jellemzőktől, az érzékelőktől és a tápfeszültség-átalakító típusától függenek.

Ebben az esetben a rekonfigurációt

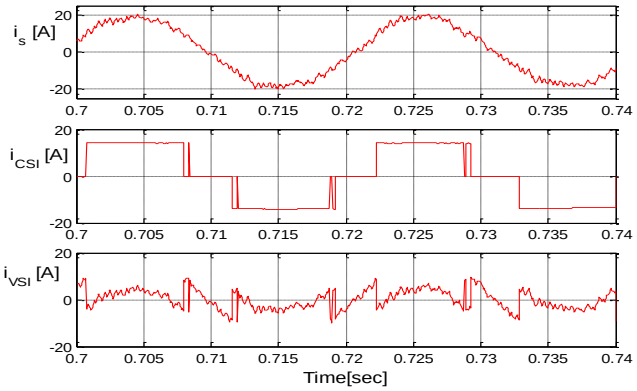
A közepes és nagy teljesítményű váltóáramú hajtások alternatív megoldása a „tandem” statikus frekvenciaváltó (SFC) táplál indukciós motor. Ez a konfiguráció egy hibrid SFC, amely egyesíti két párhuzamosan működő, különböző típusú és különböző teljesítménytartományú DC-link átalakító előnyeit. Az impulzus amplitúdó-modulációban (PAM) működő nagy teljesítményű áram-inverter (CSI) átalakítja az aktív energiát, és az impulzus szélesség-modulációban (PWM) működő kis teljesítményű feszültség inverter (VSI) biztosítja az áramellátás javításához szükséges reaktív energiát, ezzel javítva a motoráramok minőségét. Az alacsony teljesítményű másodlagos inverter (PWM-VSI) IGBT-kel valósul meg, és impulzusszélesség modulációval vezéreljük. DC kapcsolatát egy dióda-egyenirányító táplálja [12]. Ebben az egyenáramú táplálásban a VSI bemeneti feszültségét nagy értékű kondenzátor segítségével szűrjük. Nyilvánvaló, hogy a tandem konverter egy háromfázisú egyenirányító és egy VSI-alapú aktív teljesítményszűrő inverz elrendezését képviseli a motor bemeneténél. A szekunder VSI fázisában lévő áram kifejezhető az elsődleges inverter kimeneti alapárama és a négyzethullámú CSI áram ugyanazon fázisának különbségével (lásd 2. ábra):

$$i_{\mathrm{VSI},a,b,c} = i_{s,a,b,c} - i_{\mathrm{CSI},a,b,c} \tag{3}$$

A tandem másodlagos inverterének kimeneti IVSI áramerősségének négyzetes középértéke a következőképpen határozható meg:

$$I_{VSI} = \sqrt{\left(\frac{2}{6}\right) - \left(\frac{6}{\pi^2}\right)} \bullet I_{DC} \approx 0.242 \bullet I_{DC} \tag{4}$$

A tandem-átalakítóval táplált indukciós motor legjobb dinamikus viselkedése érdekében a szabályozás hagyományos vektor-vezérlő struktúrák segítségével érhető el. A tandem átalakítónak különböző vezérlési stratégiákra van szüksége a hibamentes működés és a VSI-nél alkalmazott modulációs eljárástól függően [12].

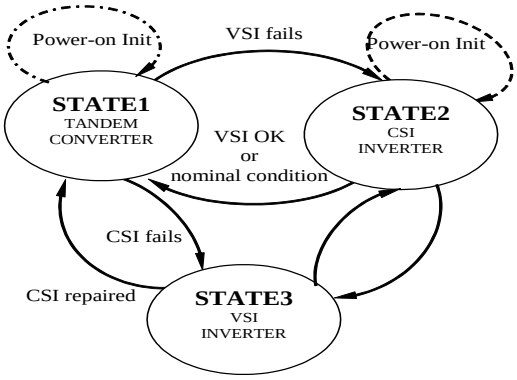


2. ábra: A tandem inverter áramainak idődiagramja

5.2 Új kutatási terület a PhD védés utáni időszakban

5.2.1 Rekonfiguráció hibamentes működés fenntartására

Egy mezőorientált szabályzó rendszer újrakonfigurálása hibamentes működésének fenntartása érdekében is megvalósítható. A rendszert (azaz a vezérlőrendszert, az inverter(ek) – a végrehajtó elemek, az érzékelők és a vezérelt motor) bármilyen állapotban működni kell, anélkül, hogy a hajtás károsodna. A tandem inverter hibamentes működésének fenntartását a 3. ábra szemlélteti. A tandem üzemmód az első állapotban valósul meg (State 1). Amennyiben valamelyik inverter (CSI, VSI) meghibásodik, úgy leállás nélkül a megfelelő inverter kikapcsolásával és a szabályozási struktúra rekonfigurációjával (Stte2 ill. State3) a hibamentes működés fenntartható.



3. ábra: A tandem inverter re-konfigurációjának ábrázolása állapot gráfon [11]

A fő szabályozási struktúra a tandem konverter (STATE1), és az újrakonfigurálás szükségességét az egyik inverter meghibásodása vagy a munkakörülmények javítása indokolja. Az újrakonfigurálást és az konfigurációs tranzienseket ehhez a konfigurációs módhoz disszertációmban ismertettem. Ezért a kutatás a tandem konverter munkakörülményeinek javítására összpontosult: A motor beindítása STATE2-ben történik, amikor a fordulatszám eléri az előírt értéket, akkor a vezérlőrendszert átállítjuk a tandem üzemmódba. A bekapcsolást követően az alapállapotba történő állítása után a motor indítása valósul meg, CSI inverteres struktúrával indítva. Az indítást és a bekapcsolás-indítást a kezdő áramok értéke szabja meg, amely nagyon magas lehet, és a vezérlési struktúrától függ (lásd: „Szimulációs eredmények”).

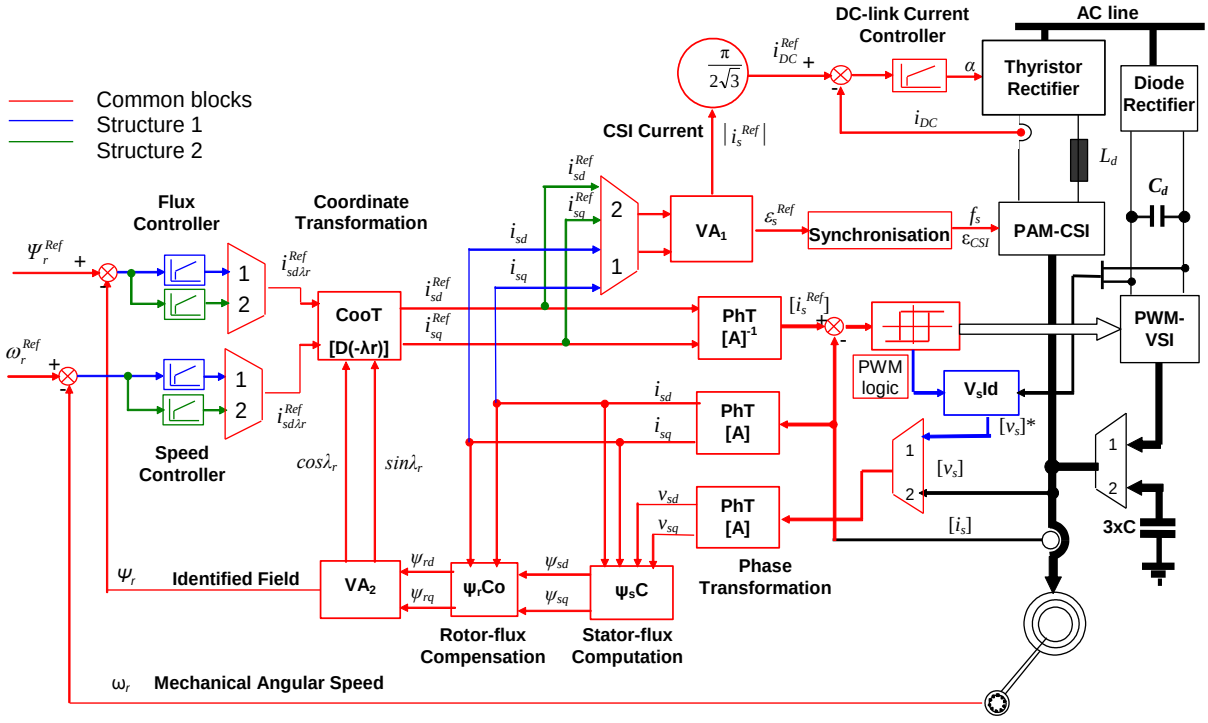
A tandem inverter érzékeny a VSI meghibásodására. Ha a VSI meghibásodik, akkor a vezérlőrendszer szerkezete elveszíti feszültségjellegét, és újra kell konfigurálni annak érdekében, hogy a motor tovább működjön és a szabályzó rendszer szerkezetét az új igényeknek megfelelően alakítsuk ki. Az új vezérlési struktúra áram karakterét CSI határozza meg.

Két konfigurációs modell létezik: részleges és teljes újrakonfigurálás. Ez azt jelenti, hogy a chip erőforrásainak részleges illetve teljes konfigurálását végezzük el. A szimulációk megvalósítása

során a teljes rekonfigurációs modellt választottam, amely átalakítja a teljes szabályozási rendszert. Az alapgondolat egyértelmű: Bekapcsolás után a váltóáramú hajtás a CSI vezérlése réven a motor áram vezérelt lesz. Amikor a motor fordulatszáma eléri az előírt értéket, a szabályzót átalakítjuk „tandem” üzemmódba.

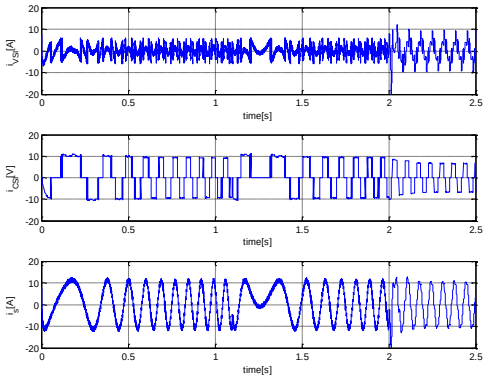
5.2.2 Szimuláció Eredmények

A szimulációkat MATLAB-Simulink környezetben végeztem, a szabályozási struktúrákhoz általam megvalósított System Generátor IP könyvtár segítségével, amely lehetővé teszi a szimulációt, a gyors prototípus készítést és az implementációt FPGA chipeken [14]. A szimuláció során használt motor adatai a következők: 5,5 kW, 50 Hz, 220 Vrms, 14 Arms és 4 póluspár. A motort CSI inverter vezérelt üzemmódban indítottam (lásd 3. ábra – State 2), amikor az a motor szögsebessége elérte ω = 94,2 rad/s, akkor megtörtént a szabályzó rendszert rekonfigurációja tandem üzemmódba (State 1 3. ábra). A rekonfiguráció időpillanata tr = 0,15 s volt.
A szimulációk során vizsgáltam a szabályozási struktúrák rekonfigurációjának hatását a motor paraméterekre. Az alábbi ábrákon, a 4. ábran bemutatott vektor-vezérlési struktúra szimulációs eredményeit mutatom be.

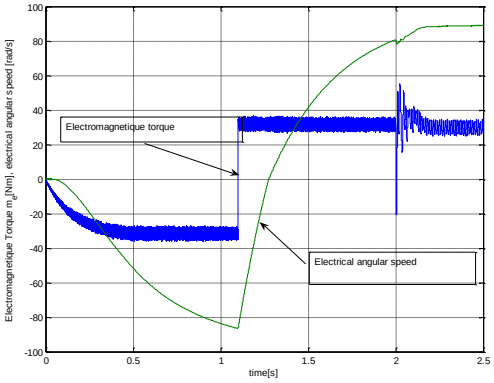


4. ábra: Rotor fluxus orientált, áramvisszacsatolt rekonfigurálható szabályzási struktúra tandem inverterrel táplált indukciós motorhoz – tömb vázlat

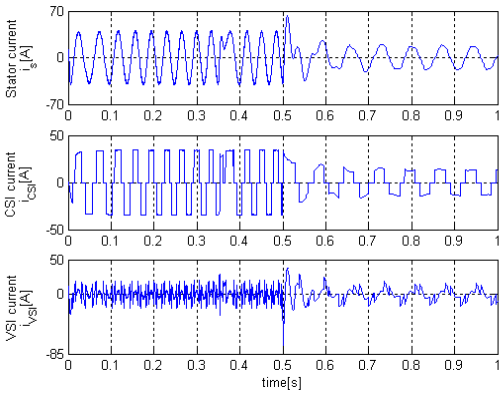
Az 5. ábra az aktuális áramerősség értékeket ábrázolja, amikor az indítás CSI-vel (State 2) történt rekonfiguráció előtt és után. A rekonfigurálás nem befolyásolta jelentősen az állórész áramerősségét. Amikor viszont tandem üzemmódról VSI üzemmódra (State 1-ről State 3.-re) történt a rekonfiguráció, akkor a konfigurációs tranziensek jelentősek (lásd 6. ábra).



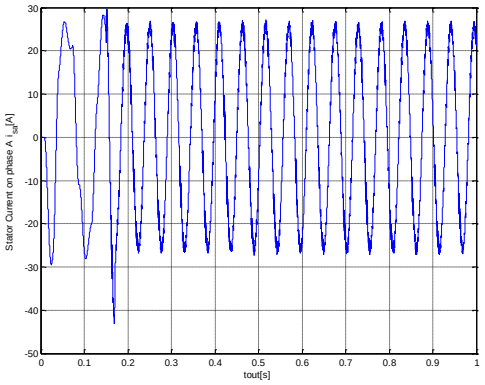
5. ábra: Áramerősség értékek ábrázolása indítás utáni rekonfigurációval



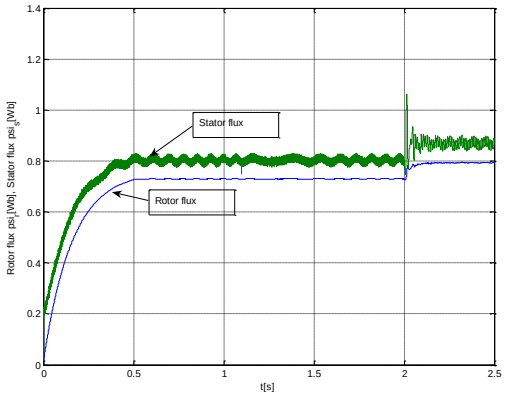
8. ábra: Elektromágneses m_e és ω_e (motor forgásirány váltással).



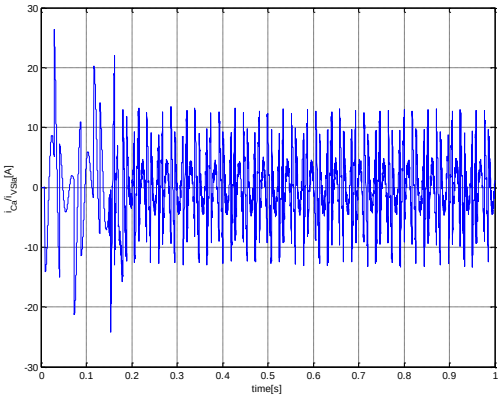
6. ábra: Áramerősség értékek tandem starttal rekonfiguráció előtt és után



9. ábra: Állórész áramerősség változása az „A”fázisban.

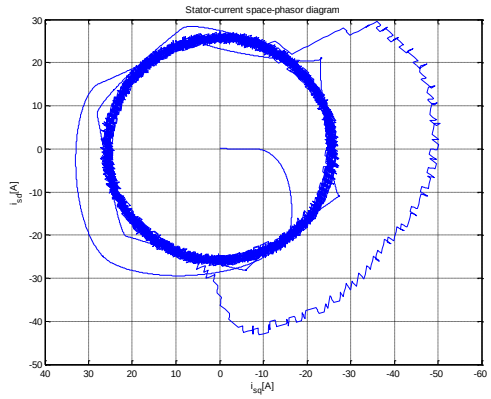


7. ábra: Rotor and Stator flux értékek rekonfiguráció előtt és után.

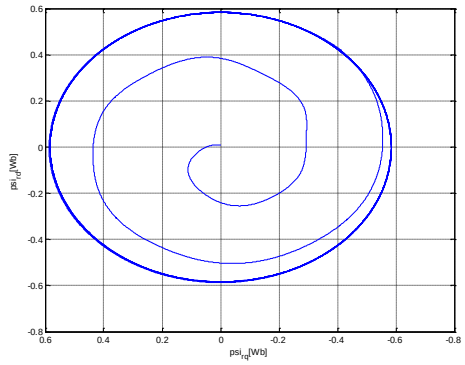


10. ábra: Áramerősség értékének változása a Kapacitás/VSI kimenetén.

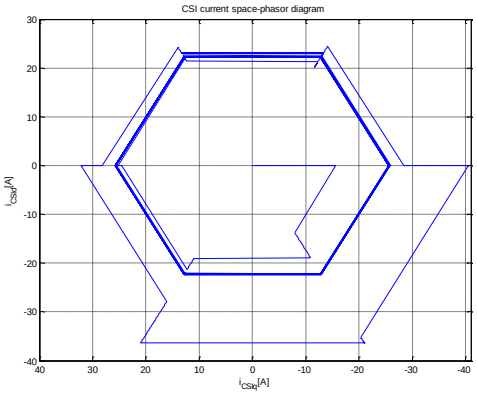
Az hajtásszabályozás minden esetben rotor- fluxus-orientált, ami a 7. ábran is megfigyelhető. A 8. ábra kezdve a szimulációs eredmények a CSI-vel vezérelt motorindítást és a rekonfiguráció utáni tandem struktúra jeleinek viselkedését ábrázoltam. A 11. ábra - 14. ábra a tér-fázis diagramokat mutatja be. Megfigyelhető, hogy az újrakonfigurálás minimális zavarokat okoz a motor feszültség és áramerősség értékeiben.. Az rekonfiguráció akkor történt, amikor a motor az indítás után állandósult állapotba került. Emiatt a motor nem szabályozott paramétereinek (vagyis az állórész-fluxus) rekonfigurációs zavarainak amplitúdója nagyobb volt.



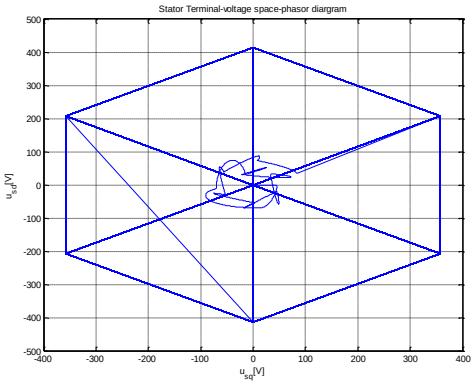
11. ábra: Motor állórész áramának tér - phasor diagramja.



13. ábra: A szabályozott forgórész fluxus tér-phasor diagramja



12. ábra: CSI kimeneti áramának tér-phasor diagramja.



14. ábra: Állórész kapocs feszültségének tér -phasor diagramja.

5.3 Tudományos eredmények a témában

A rekonfigurációt a sebesség, a nyomaték vagy más változók által előírt munkakörülmények javítására és a motor hibamentes üzemmódjának fenntartására használtam. Az egyes konfigurációkhoz hozzárendelhetünk egy diszkrét állapot gépet. A tandem inverter javítja a váltóáramú motor energiafogyasztásának hatékonyságát [15], de a hibamentes működés megőrzése és a működési paraméterek javítása érdekében először a CSI-t, majd a VSI-t kell elindítani, az optimális működés érdekében.

II. Új kutatási területek

6 Mojette transzformáció

Jelen fejezetben a 2006 – 2012 közötti időszakban végzett jelentősebb kutatási eredményeket mutatom be. A vizsgált terület a képfeldolgozásban használt Mojette transzformációk implementációi. A témában közös témavezetéssel két doktorandusz hallgató szerzett abszolutóriumot: Serfőző Péter, Szoboszlai Péter. PhD fokozatot Szoboszlai Péter szerzett. A fejezet ismerteti a Mojette transzformációk matematikai hátterét és az elért kutatási eredményeket. [S4][S5][S6]

6.1 Szakirodalmi áttekintés - Mojette transzformáció

A Mojette mint szó Franciaországból származik (Poitiers), ahol - régi franciában - fehér bab gyűjtőneveként volt ismeretes. A képfeldolgozásba Guédon vezette be a fogalmat. Mojette a transzformációnak nevezte el az általa bevezetett módszert a bab és a bin(áris) szavak analógiája alapján. A ***bin*** egy adott irányú vetületben található képpontok pixelértékeinek összegét tartalmazza [28].

Az internet elterjedésével a láthatatlan jelek (vízjelek) a képekbe történő beillesztése különböző alkalmazásoknál jelent meg, mint például: szerzői jog, az adatok épségének ellenőrzése, stb.. Vízjelek beillesztésére számos megoldást alkalmaztak az elmúlt években (Fourier, wavelet, Mojette domének stb.).

A témának széles irodalma van Hartung F a vízjel alkalmazását vizsgálta multimédiás alkalmazásokban [19]. Turán J és társai a transzformáció lehetséges alkalmazásait vizsgálta [20]. Normand N a képfeldolgozásban és az orvosi alkalmazásokban végzett kutatásokat a Mojette transzformáció felhasználásával [21]. Guedon J P és társai a képfeldolgozás és az elosztott multimédiás adatbázisok feldolgozását, és tárolását vizsgálta a Mojette transzformáció segítségével [22]. Katz M, a projekciók felhasználásával az eredeti információk (képek) visszaállításának lehetőségét kutatta [23]. További kutatások és azok eredményei a [24] - [30] publikációkban találhatók. Az idézett publikációk szerzői Autrusseau, Guédon, Turán, Szoboszlai végezték

A fent említett kutatásokban a Mojette direkt és inverz transzformációt szoftveres implementációkkal képzelték el. Ez a fajta megoldás az állóképek feldolgozásánál jól működik, de nem használható valós idejű adatfeldolgozásban, videó streamek képkockáinak feldolgozásában. Viszont valósidejű operációs rendszerekkel működő beágyazott rendszerek és célhardver használata biztosíthatja az „valósidejű” feldolgozást. A Mojette transzformáció és az inverz Mojette transzformáció hardveres megoldását azaz FGPA-án történő megvalósítását elsőként vetettem fel [S4].

6.1.1 Direkt Mojette Transzformáció

A Mojette transzformáció hasonlóan a Radon transzformációhoz kiszámolja egy adott vetület csoport összegeit egy képre, vagy egy kép adott szegmensére vonatkozóan [21]. A Mojette transzformáció (MOT) (lásd [23], [24] és [25]) kivetíti az eredeti digitális 2D képet:

$$F = \{F(i,j);\ i = 1, \ldots, N;\ j = 1, \ldots, M\} \tag{5}$$

diszkrét 1D projekciók egy K halmazára a következő képlettel:

$$M = \{M_k(l);\ k = 1, \ldots, K;\ l = 1, \ldots, l_K\} \tag{6}$$

A MOT egy egzakt diszkrét Radon transzformáció, amelyet az $S = \{(p_k, q_k),\ k = 1,\ldots, K\}$ halmazára értelmezünk:

$$M_K(l) = \text{proj}(p_k, q_k, b_l) = \sum_{(i,j)\in L} F(i,j)\delta\left(b_l - \text{iq}_k - \text{jp}_k\right) \tag{7}$$

definiálja a p_k, q_k vetületi (vetület) irányokat és δ(x) a Dirac delta függvény:

$$\begin{cases} 1 if x = 0; \\ 0 if x = 1; \end{cases} \tag{8}$$

és

$$L = \left\{(i,j); b_l - \text{iq}_k - \text{jp}_k = 0\right\} \tag{9}$$

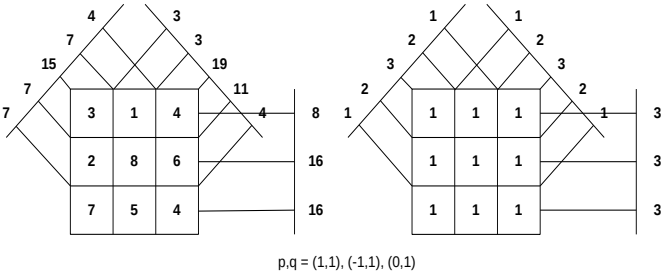
ahol b_l halmaz θ_k irányának egy digitális binje.

A vetítő operátor összegzi azon pixel értékeket, amelynek középpontját metszi az l vetítő tengely. A különböző θk vetület szögek esetében történő mintavételezés különböző bin értékeket eredményez minden (pk, qk) projekció esetében.

Egy ϑ_i, vetület binjeinek a számát (n_i) a következő képlet adja meg:

$$n_i = (N-1)|p_i| + (M-1)|q_i| + 1 \tag{10}$$

Az 15. ábra láthatjuk a direkt MOT-ot egy 3x3-as általános képre és egy 3x3-as egységképre. A vetületek halmaza *S = {(-1, 1), (1, 1), (1, 0)}* három irányt tartalmaz és összesen tizenhárom bint ad eredményül mind az egész számokkal reprezentált képre (256 színű szürke-skálás *.pgm fájlok – Portable Grey Map), mind pedig az egységképre.

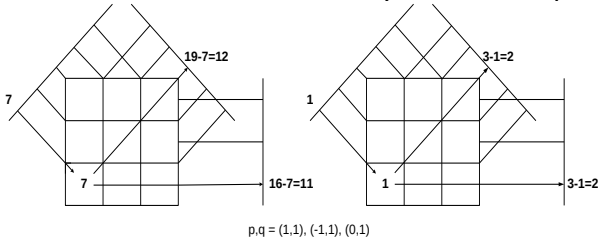


15. ábra: Direkt Mojette transzformációt bemutató példa egy valós (bal oldali kép) és egység képre (jobb oldali kép)

Egy kép MOT-ját előállíthatjuk közvetlenül a pixelek értékeinek összegzésével vagy az ún. "addition-modulus" használatával. Bináris képek (és egyéb állományok) esetén a XOR logikai művelet használható (lásd [23], [24] és [25]).

6.1.2 Inverz Mojette transzformáció

Az inverz Mojette transzformáció visszavetíti a különböző projekciók binjeinek értékét a helyreállítandó képbe. Ez azt jelenti, hogy minden iterációnál létezik egy bin, amely csupán egy pixel értéket tartalmaz. Ezáltal a kép helyreállítható. Ha a pixel értékét levonjuk a megfelelő vetületből, az iteráció folytatásával visszaállítható az eredeti kép. A rekonstrukciós folyamathoz mindig két vetület halmaz szükséges, amelyek közül az egyik a kép vetületeinek halmaza, a másik pedig egy a kép méretével megegyező méretű egységkép vetületeinek halmazát tartalmazza (minden pixel értéke 1). A . ábrán látható az iterációs folyamat első lépése.



16. ábra Inverz Mojette Transzformáció példa: a helyreállítandó kép (bal oldalon), egy pixel értékű bin keresése az egységképben (jobb oldalon)

6.1.3 A helyreállíthatóság szükséges és elégséges feltétele

A transzformációban kulcsfontosságú a vetületek redundanciája, amely az ellenőrzött vetületek, által jön létre. A vetületek száma, nagyobb vagy egyenlő, mint az elégséges vetületek száma, amelyből még visszaállítható az eredeti kép. Ehhez meg kell határozni az elégséges vetületek számát. Ezt pedig a Katz által 1979-ben megfogalmazott „Katz lémája" szerint adható meg [23], azaz egy NxM pixel méretű F kép K projekciós halmazból, a kép helyreállítható, ha:

$$N \leq P_K = \sum_{k=1}^{K} |p_k| \text{ vagy } M \leq Q_K = \sum_{k=1}^{K} |q_k| \quad (11)$$

ahol *N x M* a kép méretei, p_k az x tengely iránykoordinátája (*N*), q_k pedig a y tengelyre eső iránykoordináta (*M*). Ebből következik, hogy a kép helyreállításához annyi vetület szükséges, amennyi a projekciók iránykoordinátáinak összege valamely irány szerint vagy meghaladja a kép méretét ugyanazon irány szerint. A helyreállíthatóságnak nem szükséges feltétele, hogy ez mindkét irány szerint teljesüljön, elegendő, ha az egyik irányra teljesül a feltétel.

A MOT összetettsége a pixelek száma illetve a projekciók száma egyenes arányban áll egymással. Ahhoz, hogy ez az IMOT esetén is teljesüljön, létre kell hozni egy állományt az egységnyi pixel értékű binek címének (pozíciójának) tárolására, melyet folyamatosan frissíteni kell [24], [25].

A MOT fő tulajdonsága a redundancia, amelyet a következő képlettel számoljuk:

$$R = \frac{\sum \text{bins}}{\sum \text{pixels}} - 1 \quad (12)$$

Az IMOT sajátossága, hogy ha a binek sérülnek az átvitel során, a hiba sokkal nagyobb kiterjedésű lesz, mivel szétterjed a képen [23], [24] és [25].

6.2 Új kutatási terület - Mojette Hardveres megvalósítás és korlátai

Az eddig közzétett eredmények szoftveres megvalósításokra és azok alkalmazási lehetőségeire fókuszáltak (lásd [22], [23], [24] és [25]).

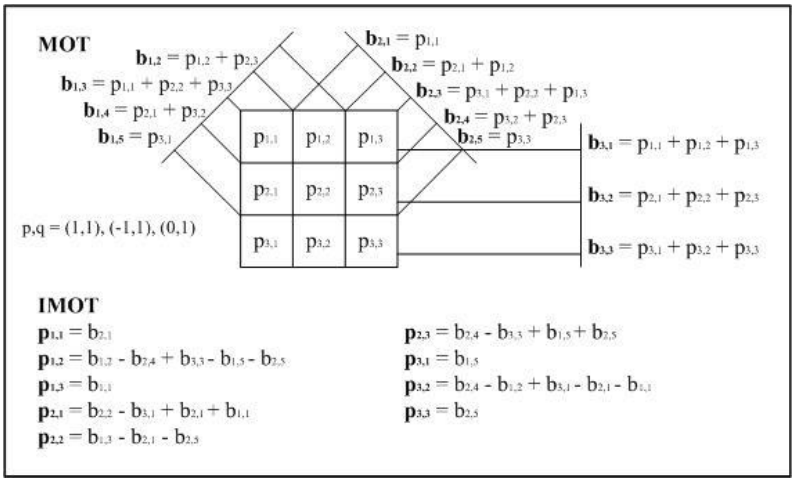
Számos publikáció bizonyította már, hogy egy célhardver sokkal gyorsabb működésre képes, mint egy általános hardveren futó szoftveres megoldás (lásd [32], [33]).

Már a kezdeti kutatási fázisban kiderült, hogy a multimédiás alkalmazások osztott adatbázisok tárolása és az adatok sérülésmentes továbbítása az új multimédiás és telekommunikációs alkalmazásokban a legnagyobb problémát.

A Mojette transzformációt közelebbről megvizsgálva, nyilvánvalóvá válik, hogy a hardveres megoldás más megvalósítási algoritmust kíván, mint az eredeti algoritmus, amely a szoftveres megközelítés eredménye. A 17. ábra a direkt és az inverz transzformációt ábrázolja részletesen konkrét pixel értékek helyett változókkal. A helyreállíthatóság feltétele magában rejti a redundanciát is.

A feltételnek eleget tevő projekciók együttes számossága minden esetben nagyobb, mint az adott kép képpontjainak száma. Amint az a 17. ábran látható a képpontok száma kilenc, míg a projekciók együttes számossága tizenhárom. Ez egy egyértelműen megoldható egyenletrendszer.

A képpontok ily módon a binek is különböző kombinációjából egymástól függetlenül kifejezhetők. Azaz az egyenletrendszer megoldható egy kombinációs hálózat segítségével. A továbbiakban több megvalósítási lehetőséget mutatunk be.



17. ábra: Teljes MOT és IMOT egy 3x3 pixel méretű kép esetén

6.2.1 "MOTIMOT" társ processzor implementáció

Az 18. ábra Mojette és Inverze Mojette transzformációt megvalósító egységet, mint társprocesszort ábrázolja. A "MOTIMOT" társ processzor elnevezés a direkt és inverz Mojette transzformáció beágyazott feldolgozó egységként megvalósított hardveres implementációját jelöli. A feladat célja a következő: az Ethernet hálózaton keresztül érkező képek, videók, vagy más forrásból érkező állományok valós időben történő feldolgozás után, a keret vagy a kép továbbítódik a címzettnek vagy egy másik kliensnek (PC, PDA, stb.).

A MOT és IMOT függvények önálló modulokként valósulnak meg, úgymint egy processzor társ-processzorai. Ahogy ez a [33]-ben is szerepel, a valós idejű képfeldolgozás nagyon nagy számítási sebesség és kis disszipált teljesítmény mellett kell megvalósuljon. Ezt a feltételt egyetlen PC vagy egy fix pontos DSP sem tudja teljesíteni. Az FPGA áramkörök alkalmazásával a gyors feldolgozás igénye teljesül. Az FPGA használata mellett szól a beágyazott rendszer chip-en belüli megvalósítása. A fejlesztést egy ML310-es kártyán valósítottuk meg [31], a Xilinx Virtex II Pro FPGA lapka erőforrásai figyelembevételével. A lapkába integrálva megtalálható egy kétmagos PowerPC 405 jelzésű RISC architektúrájú általános célú processzor (400 MHz) amely képes egy kisméretű kernel maggal megalkotott operációs rendszer futtatására (Linux). Így a beépített PPC 405-ös felügyeli a kommunikációt a külvilág felé és futtatja a társprocesszor meghajtó programját.

A kísérletek eredményét egy 256x256 pixel szürke skálás kép segítségével lehet ellenőrizni. A feldolgozási memória igény a kép méretei miatt 64KB lesz. Az egyes vetületek feldolgozása párhuzamos módon történik. A bin vektorok tárolása pedig a létrehozott úgynevezett "MOT memória állományban" történik.

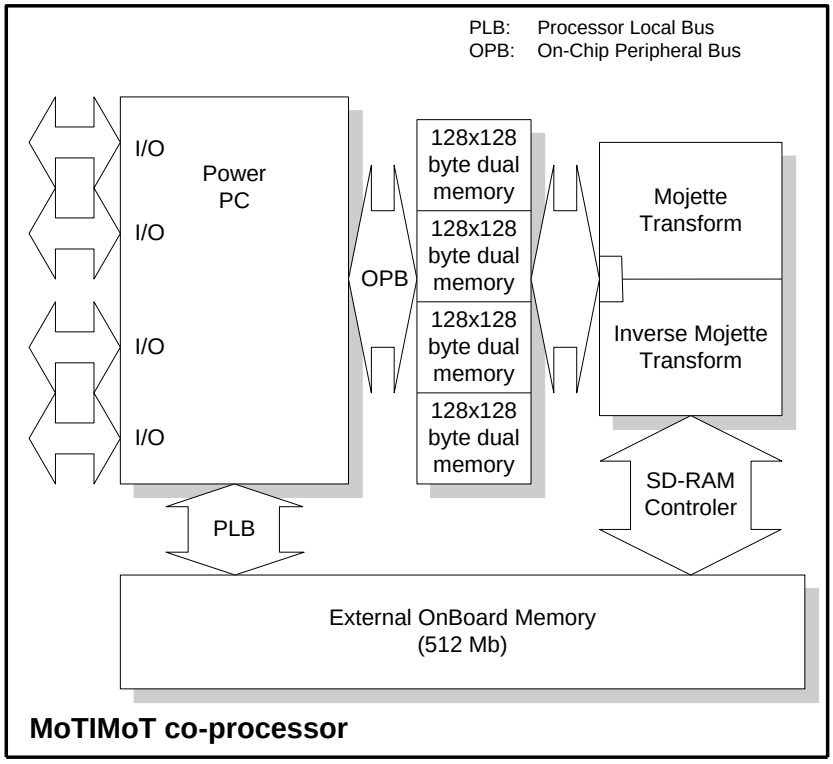
Az eredeti kép szeletekre osztásában az is motiváló tényező, hogy a MOT fájlok sérülhetnek az átvitel során. Ha a MOT fájlok megsérülnek, elegendő vetület híján az eredeti kép nem állítható helyre. Azzal, hogy a Mojette transzformációt a kép szeleteire alkalmazzuk, egyúttal csökkentjük a MOT fájlok károsodásából eredő képhibákat. Ráadásul, ha képszeletekkel dolgozunk, kisebb a feldolgozáshoz szükséges memória mérete is.

A MOT/IMOT hardver erősen függ az ML310-es kártya [31]-ban megadott struktúrájától. Ez a függőség korlátozza a képet fogadó kommunikációs csatornákat, de nem jelent megszorítást a társprocesszor megvalósításnál. A . ábrán a MOT/IMOT egyszerű blokként szerepel, mivel az adott FPGA cellatartományban csak az egyik funkció megvalósítása fér el, ami szükségessé teszi majd az FPGA parciális re-konfigurálását. Az 18. ábra csupán a képfeldolgozó rendszer főbb részeit mutatja. Ezek a kemény magos Power PC processzor, mint fő vezérlő egység, valamint a MOT és IMOT egységek/perifériák. A MOT és az IMOT társprocesszor egységek egyaránt a PLB (Processor Local Bus) sínrendszeren keresztül csatlakoznak a Power PC-hez, mivel a feladatvégzés során a fő processzor ellenőrzése alatt fut a MOT-IMOT.

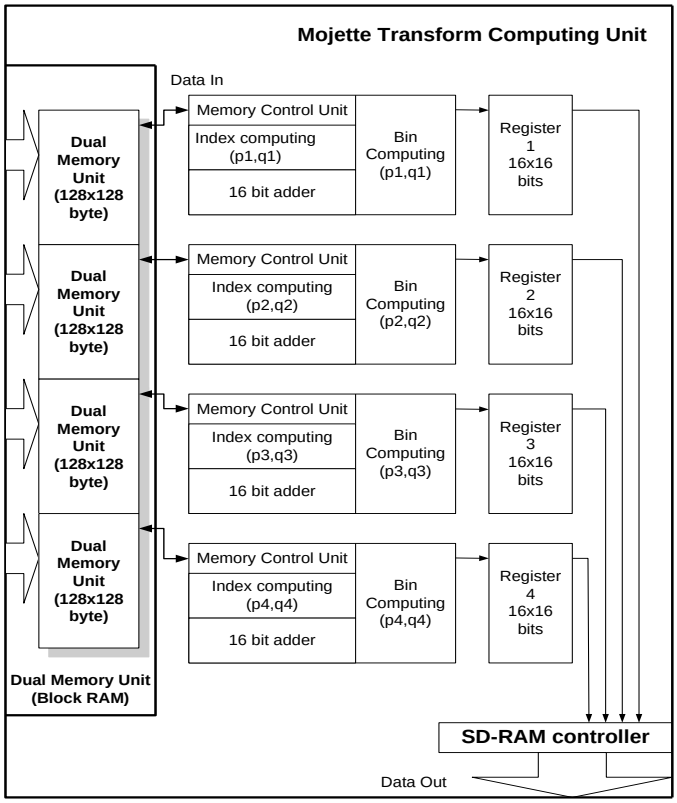
6.2.2 MOT tömbvázlata

A MOT feldolgozó egység a 19. ábrán látható. A 4x128x128 bájt méretű duális memória egység csökkenti a ki/bemeneti műveletek számát a MOT processzor és a külső memóriaegység között. A memóriába egy 128x128 pixeles képszelet kerül betöltésre 4 példányban, hogy mind a négy vetületi irányban a feldolgozás azonos időben történjék. Minden vetületi vektor az IMOT állományban egy-egy vetületi iránynak felel meg. Az algoritmus előnye más megvalósításokkal szemben a párhuzamos feldolgozás, amellyel növeljük a rendszer műveleti sebességét. Az egyes modulok fejlesztése Handel-C nyelven történt, majd a processzor PLB sínrendszeréhez való illesztés a Xilinx EDK rendszerben valósult meg. Egy vetület feldolgozása valamely irányban, azaz egy vektor létrehozásának kód részlete n látható.

A megvalósítás előnye a [26] és [27] képest a párhuzamos feldolgozás minden vetületi irány szerint, ami a számítási sebesség növekedését eredményezi alacsonyabb működési frekvenciák mellett (100 MHz). A kisebb képszeletek feldolgozása csökkenti a kép hibák megjelenését, amennyiben a binek sérülnének. Mint a fenti esetben említettem a 256x256 pixel méretű képek feldolgozása esetében a bin hibák csupán a bin oszlopában és a közvetlen környezetében jelenik meg. Még kisebb 128x128 pixel méretű képszeletek feldolgozása esetében a meghibásodott bin által okozott függőleges oszlop hiba mérete 50%-al csökken.



18. ábra: MOTIMOT processzor tömbvázlata



19. ábra: MOT processzor tömbvázlata

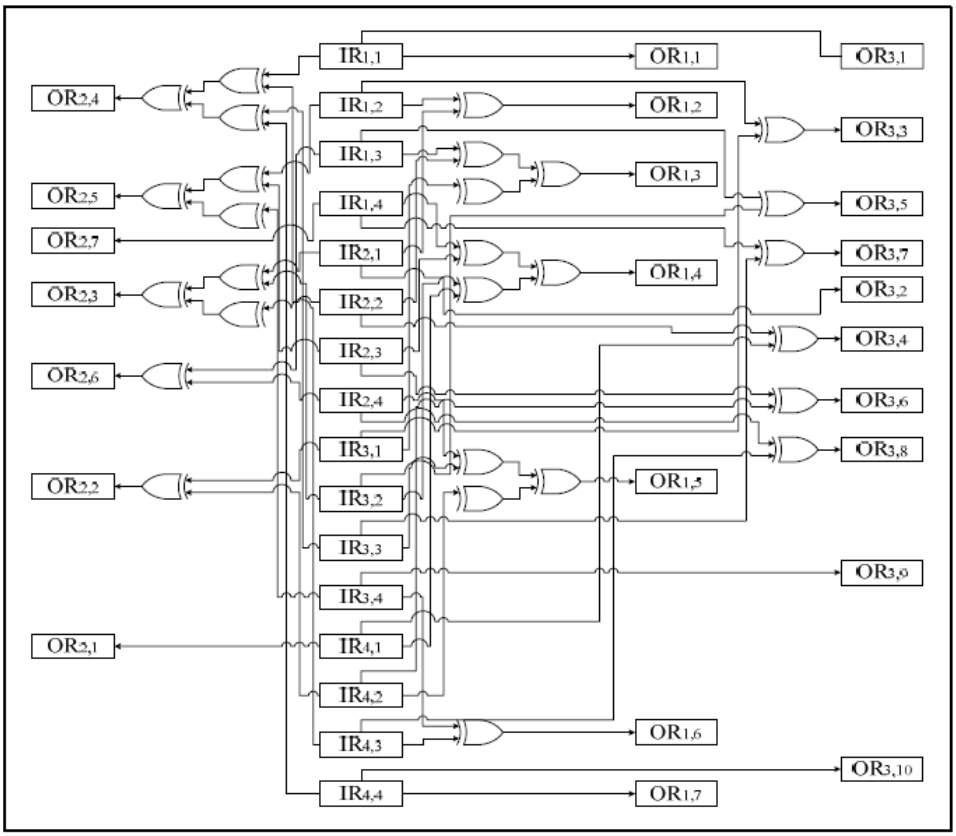
```
for (i=0; i<=im_size+3p; i++)
        {
        for (j=0; j<=q-1; j++)
                {
                if (i= =0) do
                        bin[i+j]=im[i,j];
                if (i = =1) do
                        bin[i+j]=im[i,j]+im[i-p, j+q];
                if (i = =2) do
                        bin[i+j]=im[i,j]+im[i-p, j+q]
                                        +im[i-2p,j+2q];
                if ((i = =3) && (i<=im_size) do
                        bin[i+j]=im[i,j]+im[i-p, j+q]
                        +im[i-2p,j+2q] +im[i-3p,j+3q];
                if (i= =im_size+p) do
                        bin[i+j]= im[i-p, j+q]+im[i-2p,j+2q]
                                +im[i-3p,j+3q];
                if (i= =im_size+2p) do
                        bin[i+j]= im[i-2p,j+2q] +im[i-3p,j+3q];
                if (i= =im_size+3p) do
                        bin[i+j]= im[i-3p,j+3q];

                }
}
```

20. ábra: MOT megvalósítás kódjának egy részlete [S4]

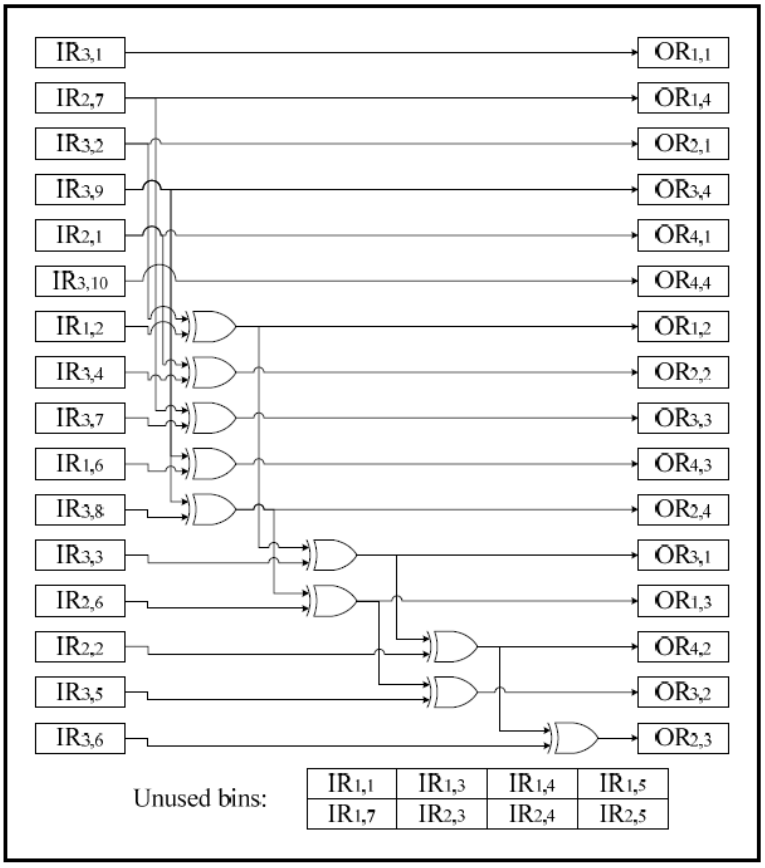
6.2.3 A Léptetőablak Módszer MOT-IMOT [S5]

A léptető ablak módszernek elnevezett eljárás lényegében nagyon egyszerű. Mivel az egy menetben feldolgozható képméretet az adott hardver erőforrásai korlátozzák, meg kell keresni azt a képméretet, ami még a korláton belülre esik. Az eredeti képet ezután fel kell osztani „feldolgozási ablak", röviden „ablak" méretű képszeletekre. A folyamat a következő: a képet beolvassuk a memóriába majd a bal felső „sarkára" illesztjük a feldolgozási ablak bal felső „sarkát"

és végrehajtjuk a transzformációt. Ezután az ablakkal egy ablaknyit oldalra lépünk és ismét végrehajtjuk a transzformációt. Amikor elértük a kép szélét egy ablaknyival lentebb lépünk és a kép bal szélétől újra végiglépkedünk a másik széléig. Feldolgozási egységnek (ablakméretnek) célszerű olyan méretű képszeletet választani, ami minél jobban kihasználja a rendelkezésre álló erőforrásokat, ugyanakkor az eredeti kép mérete a feldolgozási ablak ugyanazon irányba eső méretének egész számú többszöröse. Több FPGA egyidejű használatával a feldolgozási ablak mérete növelhető (az egyes feldolgozási egységek összege). A helyreállítás során az ablaknyi méretű kép szeleteket természetesen újra egymás mellé kell rendezni. A módszer előnye, hogy a bin-ekben keletkezett bithibák, amelyek a transzformáció jellegéből adódóan a teljes képre hatást gyakorolnának (erősen zajos kép), csupán egyablaknyi méretű szeletekre gyakorolnak hatást.
Az 21. ábra a 4x4-es képméret MOT feldolgozó egységet mutatja be. A bemeneti regiszterek ($IR_{x,y}$ ahol x = 1, …, 4; y = 1, ..., 4) tartalmazzák a kép egyes pixel értékeit, míg a kimeneti regiszterek ($OR_{i,j}$ ahol i = 1, ..., 3; j = 1, .., 10) tartalmazzák a bin értékeket. A három vetületi irány (1,1; -1,1; 1,2) szerint kapott bin értékek halmaza teljesíti a visszaállítási kritérium feltételét. A Mojette operátor a logikai antivalencia vagy XOR kapcsolat. A 4x4-es képméret természetesen nem azonos a valódi kép méretével, viszont a léptető ablak módszer megvalósításához a megfelelő ablakméretet jelenti. A regiszter mérete (byte, szó dupla szó) függ a bemeneti adatoktól. Megjegyezzük, hogy a nagyobb adatméret nagyobb regisztermeretet és nagyobb hardver erőforrásokat igényel.
Az IMOT megvalósítás logikai hálózatát a 22. ábra mutatja. A pixel értékek két módszerrel számolhatók. Első módszer minden pixel érték kiszámítható a bin értékekből. Ebben az esetben egy bin egy pixel értéket tartalmaz. A második számítási módszer esetében a pixel értékét megkapjuk a bin értékéből az előzőleg eredményként megkapott pixelek értékének felhasználásával. Ez a számolási módszer egyszerűbb hardver felépítést eredményez.
A 22. ábra a bemeneti IR regiszterek tartalmazzák a bin értékeket, a kimeneti regiszterek (OR) pedig az eredeti kép pixel értékeit. Az IMOT operátor pedig az antivalencia (XOR) függvénykapcsolat. Mindkét esetben (MOT és IMOT) a regiszterek száma azonos. Megjegyezzük, hogy az IMOT regisztereinek száma kisebb is lehet a binek redundanciája miatt.
Az egy pixel értéket tartalmazó pixel-bin megfeleltetés (a legtöbb vetítési irány tartalmaz néhány bin-pixel párt) esetében a pixel érték számítás nélkül megkapható. Az ilyen típusú értékeket nullaszintű számolási pixel értéknek nevezzük. A több pixel értéket tartalmazó binek esetében a logikai hálózat több szintű is lehet (n ≥ 2). A logikai hálózat számítási szintjei (a sorba kötött XOR kapuk száma) fogják meghatározni a további pixel értékek számolási szintjét. Az ábrán öt ilyen számítási szintet különböztetünk meg nullától négyig. A szoftveres megoldás hátránya, hogy a bemutatott léptető ablakos módszert „for” ciklusok segítségével lehet megoldani, ahol a ciklus hossza a 1-től n_{bl}-ig tart (n_{bl} az azonos szinten lévő binek). amennyiben az ablak méretét növeljük a logikai szintek száma is nő (FPGA megoldás). Felhasználva a már kiszámított pixel értékeket a számítás bonyolultsága megnövekszik és a számítási sebesség lassulni fog, amikor a nullaszintű bineket is használjuk a számításokban.



21. ábra 4x4-es méretű kép MOT megvalósításának modellje [S5]



22. ábra: IMOT 4x4-es képméret visszaállítási modellje [S5].

A tisztán kombinációs MOT-IMOT (XOR kapu) használatának hátránya, hogy a több szintű logika esetében a számítási sebességet a kapuk terjedési ideje korlátozza. A terjedési idő meghatározza a regiszterek vezérlési órajelének periódusát. Ennek alapján:

$$\sum_{i=0}^{n} t_{di} > T_{CLK} \tag{13}$$

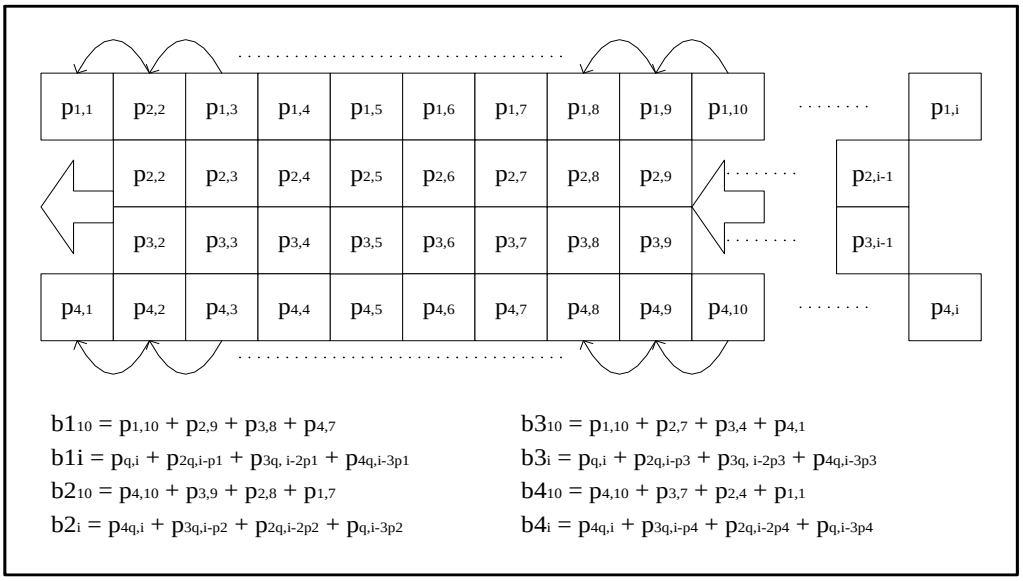
ahol t_{di} az *i.*-ik szintű kapu terjedési ideje és T_{CLK} pedig a regiszterek vezérlő órajelének periódusa. A kutatás elvégzésének idején (2009 - [S5]) az ML310 Virtex II alapú rendszer órajel sebességét is figyelembe véve (100 MHz – lásd [31]) a fentiekben bemutatott MOT-IMOT implementáció is kielégítő megoldást jelentett. Természetesen az egyes logikai fokozatok közé illesztett regiszterek segítségével a számítási sebességet növelni lehetne.

6.2.4 A Csúszó-ablak Módszer MOT-IMOT

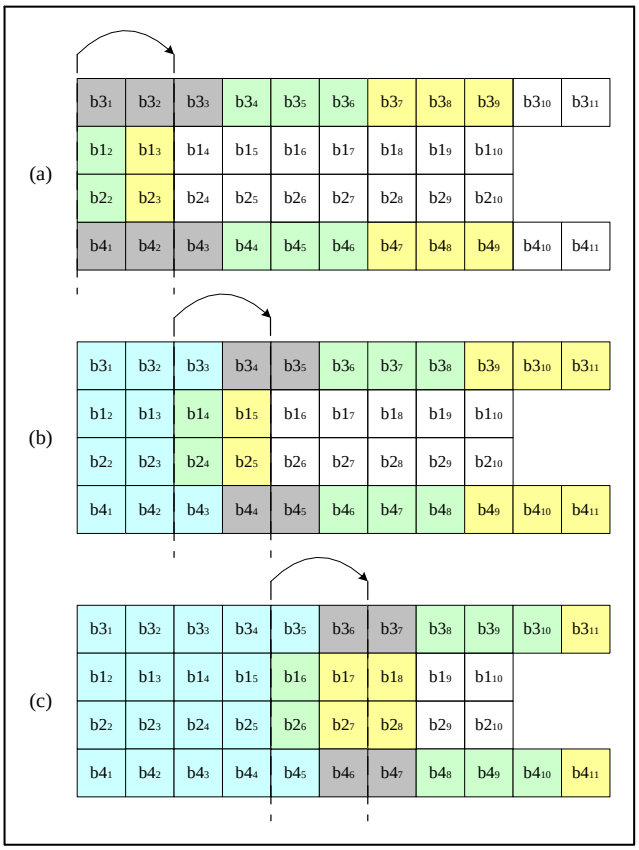
Az előző fejezetben (6.2.3) bemutatott módszerhez hasonlóan a csúszó-ablak módszer lényege is az, hogy a teljes kép helyett egy menetben csupán annak egy szeletét dolgozzuk fel. Alapvető különbség azonban a már bemutatott módszerhez képest, hogy míg a léptető ablak módszer használatakor két szomszédos ablaknak nincs közös képpontja, addig a csúszó-ablak módszernél két szomszédos ablaknak több közös képpontja is van. A csúszó ablak módszer lényege, amint azt a neve is mutatja, az ablakot (amely az aktuálisan feldolgozandó adatokat tartalmazza) nem léptetjük a képen, hanem csúsztatjuk mindig egy oszlopnyival tovább (balról jobbra). Ennek a módszernek az előnye nem a képek feldolgozása során mutatkozik meg igazán. Képek feldolgozásakor pontosan tudjuk azok méretét és színmélységét, hiszen a kép fejlécében ez az információ megtalálható. A jelenlegi digitális fényképezők azonban nem csupán állóképet képesek rögzíteni. Továbbá a Mojette transzformáció alkalmazása nem kell, hogy feltétlenül képfeldolgozást jelentsen. A feldolgozó egység számára teljesen közömbös, hogy a feldolgozandó adat mit ábrázol. Ezzel a Mojette transzformáció alkalmazását kiterjeszthetjük bármilyen adathalmazra. A csúszó ablak módszer erőssége, hogy a feldolgozandó adathalmaz méretétől függetlenül működik és különböző méretű adathalmazokhoz nem szükséges különböző méretű feldolgozási egység meghatározása. A [S5] a csúszó ablak módszer általános lépését mutatja be. A kimenetre kerülő binek értékét meghatározzuk a feldolgozási ablak adatai alapján, majd az ablakot egy oszlopnyival elcsúsztatjuk jobbra. Az első néhány lépést, amíg eljutunk az általános lépés alkalmazásának lehetőségéig célszerű szoftveres úton megvalósítani csakúgy, mint az utolsó lépéseket. Általános lépésnek tekinthető az a lépés, amelyben minden kimenő bin érték maximális számú adatérték együttest tartalmaz. Ez a tízedik lépéstől kezdve igaz.

A csúszó ablak módszer alkalmazásánál a feldolgozási ablakot feltöltjük adatokkal a kiválasztott szélességben. A kiválasztott projekciós irányok *p* koordinátái adják meg a szükséges ablak méret szélességét, míg a *„q”* koordináták a magasságát. Amennyiben nagyobb feldolgozási ablak méretet szeretnénk elérni, több FPGA egyidejű használatával időosztásos demultiplexelést alkalmazva több feldolgozási ablak tehető egymás mellé ily módon megnövelhetjük a feldolgozási egység méretét a *„q”* koordináták irányában.
Az 24. ábra az inverz transzformáció (a helyreállítás) folyamatának első három lépését mutatja be. A helyreállítás során az egy pixel értéket tartalmazó binek értéke a kimenetre kerül (bal oldali első és második oszlop), míg a többi regiszterben a kapcsolódó binek értékeinek módosítása történik. A következő lépés, az ablak csúsztatása, tulajdonképpen regiszterek közötti érték átadás, majd az új bemeneti adatok beolvasása a megfelelő regiszterekbe. A 25. ábra ábra a helyreállítási folyamat i.-dik lépését mutatja be. Az ábrán lévő egyenletek az adat vissza állítását és a megfelelő binek

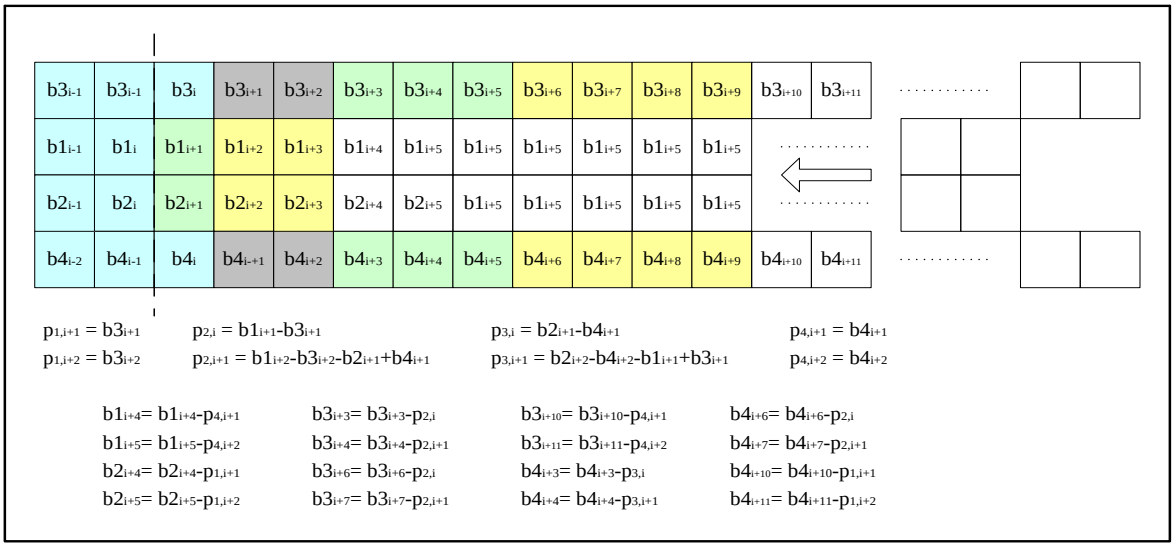
módosításának (frissítésének) lépéseit mutatja a vetítési irányok szerint. A visszaállított pixelek száma 10*q az első, 6*q az utolsó és 8*q minden egyéb ciklus esetében. Ezek szerint a általában 8*q pixel és 16*q bin számítását kell megvalósítani. Több FPGA használata esetén az eredeti információ a helyreállított részegységek megfelelő összefésülésével állítható elő.



23. ábra: MOT: Csúszó-ablak módszer



24. ábra: Csúszó ablak módszer IMOT 1-2-3. lépés



25. ábra Helyreállítási folyamat i.-ik lépés, csúszó ablakos módszer használatakor.

A csúszó ablak módszer előnye, hogy nem csak adott méretű kép feldolgozása esetén működőképes változtatás nélkül, hanem gyakorlatilag bármilyen méretű állomány feldolgozására alkalmas. További előnye, hogy az egyik iránykoordináta több eszköz párhuzamos alkalmazásával a hardver lényeges változtatása nélkül növelhető. Hátránya, hogy alacsony komplexitású hardverrel csak az általános lépések valósíthatók meg. Az általános lépések előtti és utáni lépéseknél a binek értékét célszerű szoftveres úton számolni. A módszer interneten elosztott adatbázisok (például képek, videók, stb.) esetén hasznos, amikor több vetítési irányban (három vagy négy) kell alkalmazni a transzformációt és a feldolgozandó adatmennyiség relatíve nagy (M>>N).

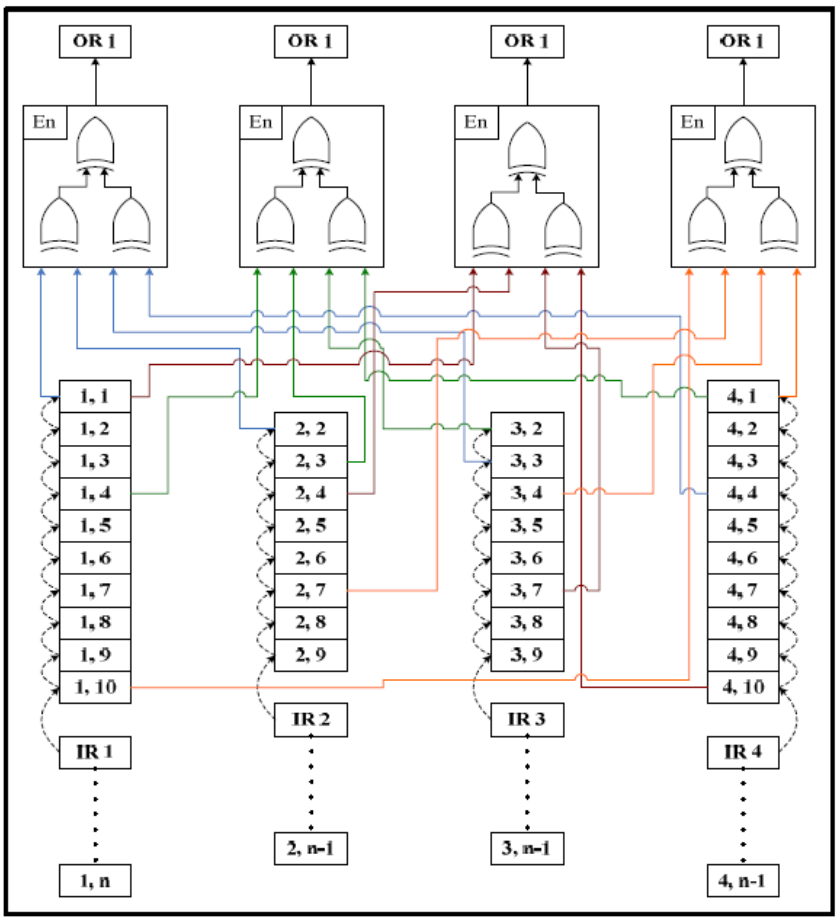
A 26**.** ábra a MOT megvalósításának kapcsolási rajzát ábrázolja, ahol p_i={1, -1, 3, -3} és q_i=1. A kép méretét a P és Q értéke adja meg. A mi esetünkben Q = 4 és P = (állomány mérete)/Q. Ebből következik, hogy a csúszó ablak mérete megválasztható, az állomány méretétől függetlenül. Azaz a csúszó ablak mérete a hardver erőforrások függvénye (logikai erőforrások, azaz cellák, memória azaz block RAM, stb.) egy adott FPGA esetében.

Amennyiben a q értékét nagyobbra választjuk (q=2) az áramkör logikai méretei, azaz a csúszó ablak mérete és így a regiszterek száma az új q értékének többszöröse lesz. Ebből pedig az következik, hogy nagyobb FPGA erőforrásokra van szükségünk vagy több FPGA áramkört kell használnunk. Egy ilyen elméleti megoldást ábrázol a 27. ábra. Minden áramkör struktúrája azonos, de a teljes rendszer virtuális ablakának mérete négyszer nagyobb, mint egy FPGAs megoldás. Az elméleti példában minden FPGA négy azonos méretű információt dolgoz fel. A négy áramkör így 16 azonos méretű adat feldolgozását végzi. A vetítési q koordináta azonos minden áramkör esetében ebből az következik, hogy az azonos irányok esetében a feldolgozott adatok összevonhatók. Négy FPGA és négy vetítési irány 16 vetítést eredményez. Az összevont elemek pedig négy bin tömbbe vonhatok össze, ahol a q koordináta (q_r = 4). Ilyen módon az elméletileg feldolgozandó adat mennyiség négyszereződik. Az IMOT rendszer pedig visszaállítja az eredeti állományt akár egy akár négy FGPA-val valósul meg az IMOT.

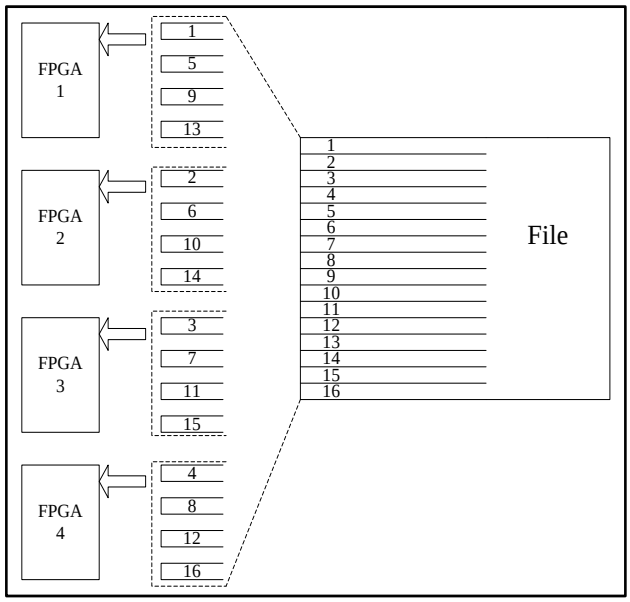
28. ábra szemlélteti a fentiekben említett összevonás folyamatát. Az egyes FPGAk négy kimenetén négy bin tömb összevonását ábrázolja az ábra. A megfelelő bineket azonos színnel jelöltük.

Az IMOT felépítése, azaz a visszaállítási folyamat megvalósítása bonyolultabb áramkört eredményez összehasonlítva a MOT áramkörével. Az egy pixelt tartalmazó binek feldolgozása egyszerű, viszont az összes többi bin esetében korrekciót kell alkalmazni a bin értékének visszaállítása során. Ez azért szükséges, mivel a visszaállítási folyamatban új egy pixeles binek keletkeznek. A korrekció pedig biztosítja a visszaállítás folyamatának folytonosságát.

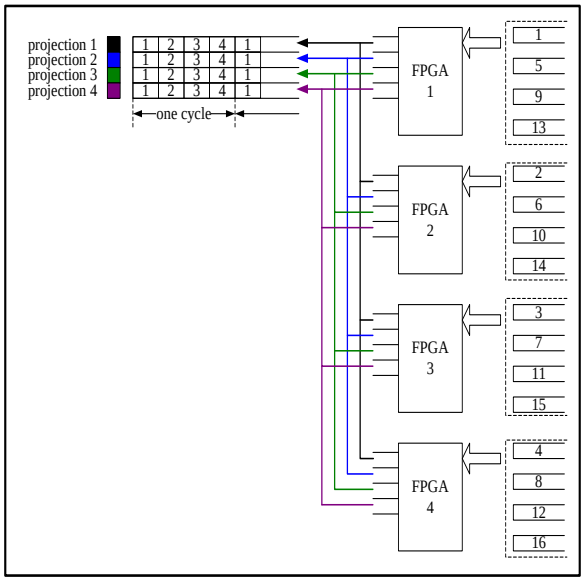
A 29. ábra az IMOT kapcsolási rajzának egy szeletét ábrázolja. Az ábrán a Pi (i=1..8) értékek a visszaállított pixel értékeket jelölik, PLi (i=1..4) a vetítési irányok, míg a felettük elhelyezkedő „téglalapok” a bineket tartalmazó regiszterek. Az ábrán a „tr” a köztes értéket tartalmazó regiszterek a bin korrekció értéket tartalmazzák. A Unit 1 kimenetei a P1 … P8 visszaállított pixel értékek, amelyeket az EN jel hatására lehet olvasni a regiszterek kimenetén. A Unit 2 a javított bin értékeket tartalmazza és ugyancsak az engedélyező jel hatására lehet olvasni. A korrekciót végző áramkört az ábrán csak egy vetítési irány szerint mutatjuk be. A többi irány szerinti korrekció azonos felépítésű. A új frissített bin értékeket tartalmazzák a tr köztes regiszterek. A második lépésben a tr regiszterek értékét visszaírjuk az eredeti bint tartalmazó regiszterbe. A harmadik lépés pedig az csúszó ablak léptetése. A regiszterek értékének léptetése két hellyel előre és az új pixel értékek olvasása/számítása.



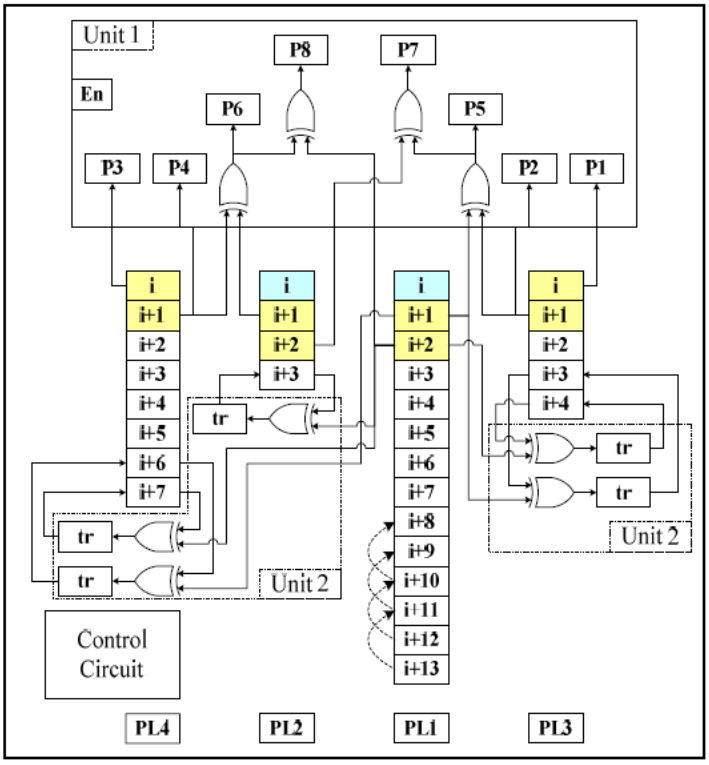
26. ábra: MOT Csúszó ablak módszer megvalósításának kapcsolási rajza



27. ábra Csúszó ablak módszer több FPGA áramkörös elméleti megoldásának tömbvázlata



28. ábra Bin vektorok összevonásának szemléltetése elméleti megoldás esetén (4 FPGA)



29. ábra: IMOT csúszó ablak módszer kapcsolási rajza

A MOT és IMOT különálló egységként valósítottam meg, mint processzor perifériákat illesztettem a PLB sínre (2004-2008 közötti technológia).

6.3 Tesztelési Eredmények

A szimulációk elvégzéséhez és a számítási sebességek összehasonlításához egy MTTOOL szoftvert hoztunk létre, amelynek segítségével a Mojette transzformációk számítási sebessége összehasonlítható. A vizsgálatokat több platformon végeztük az MTTOOL felhasználásával (direct

és inverz Mojette transzformáció). A teszteket a következő platformokon végeztük el (2005 - 2010):

- PC: Pentium P4/3,4 GHz, 1 GByte DDR DIMM RAM/400 MHz, 10/100 Mbit Ethernet kártya, 300 GByte merevlemez, Windows operációs rendszer;
- PCL: kétprocesszoros AMD Opteron(tm)/2405,482 MHz, 1024 Kbyte cache, Linux operációs rendszer (intézeti szerver)
- ML310: VIRTEX Virtex-II Pro™ XC2VP30 FPGA alaplap, két keménymagos PowerPC405 32 bites processzor, órajel: 100 MHz, lokális (FPGA) sínrendszer PLB/OPB/OCM – 100 MHz, 256 Mbyte DDR DIMM, 512 MB Compact Flash, 10/100 Gbit Ethernet, Linux operációs rendszer.

A teszteket tízszer futattuk, különböző méretű képekkel (512x512, 256x256, 128x96 pixel). A képeket felosztottuk 32x32 pixel méretű szeletekre és 3 féle tesztet végeztünk:

- 512x512 pixel méretű kép,
- 32x32 pixel méretű szeletekre bontva;
- képkeret 512x512 pixel méretű képpel,
- 32x32 pixel méretű szeletekre bontva;
- 24 képkeret 512x512 pixel méretű képpel,
- 32x32 pixel méretű szeletekre bontva

Hasonló teszteket végeztünk 16x16 pixel méretű szeletekkel. A kisméretű képekkel való tesztelést a kis felbontású LCD grafikus kijelzők miatt vettük fel a tesztelési listára.

2. táblázat: Teszteredmények a Mojette transzformációk összehasonlítására

	Képek Képkeretek száma	Min. fut. idő [s]	Max. fut. idő [s]	Átlag fut. idő [s]	max-min [%]
P IV 3000MHz 300W	1	0.4000	0.4200	0.4080	4.9020
	6	2.4800	3.0600	2.6380	21.9864
	24	11.3700	13.0000	12.3970	13.1483
AMD Opteron 2x2400MHz 500W	1	0.1040	0.1110	0.1071	6.5359
	6	0.6220	0.6550	0.6394	5.1611
	24	2.4850	2.5920	2.5590	4.1813
PowerPC 405 100MHz 100mW	1	4.9660	4.9900	4.9795	0.4820
	6	29.7300	30.3530	29.8199	2.0892
	24	118.892	121.617	119.3114	2.2839

Az 2. táblázat a minimum és a maximum teszt eredményeket tartalmazza mindhárom platformra (PC, PCL, ML310). A táblázat tartalmazza a minimális és maximális feldolgozási időt. Egy képkeret egy teljes képet, míg a 24 képkeret pedig a mozgó kép (film, videó) 1 s-os képeit tartalmazza. Az eredményekből látható, hogy a PC alapú platform feldolgozási ideje a leghosszabb (az ML310 platform 100 MHz-es órajelét figyelembe véve.). Nem szabad elfelejtenünk, hogy az operációs rendszer is elvesz időt a feldolgozásból. Az is figyelemre méltó, hogy a legkisebb disszipált teljesítményt az ML310-es rendszer adja le.

A hardver implementáció esetében az egyes kép szelet (ablak) feldolgozási idők a képszeletek perifériába történő írása és a binek olvasási idejére korlátozódik.

A feldolgozott eredeti kép (MOT) és a visszaállítás utáni (IMOT) képet ábrázolja a 30. ábra. Az ábra egy szürke árnyalatú 256x256 pixel méretű képet ábrázol (nem továbbítottuk interneten) és nem tartalmazott sérült biteket sem.

a) eredeti kép; b) visszaállított kép

30. ábra MOT/IMOT transzformációval feldolgozott kép eredménye

6.4 Új tudományos Eredmény

A Mojette transzformációk alkalmazása osztott adatbázisokban és videó streamek tárolása fontos szempont az új telekommunikációs szolgáltatások megvalósításában. Az általam javasolt MOT-IMOT hardveres megoldás a feldolgozás sebességét megnöveli. (léptető ablak, csúszó ablak, MTTool). A nagyobb méretű állományok esetében a MOT-IMOT implementáció több párhuzamosan működő FPGA áramkörrel valósítható meg.

7 Beágyazott rendszerek alkalmazásainak párhuzamosítása

Jelen fejezetben bemutatott kutatási eredményeket Bartók Roland és Drótos Dániel PhD hallgatók együtt végeztem. A kutatás célja a beágyazott rendszerekben lévő feladatok párhuzamosításának vizsgálata rekonfiguráció alkalmazásával. A fejezetben két lehetséges megoldást vizsgáltam:

- processzor megszakítás kiszolgálás több processzor alkalmazásával [S9] [S10], [S11], [S12] és [S13];
- algoritmusok párhuzamosításának vizsgálata a végrehajtási sebesség növelésére [S14], [S15], [S16] és [S17]

7.1 Szakirodalmi áttekintés - Megszakításkezelés

A számítógépes modell, amelyet Neumann János fogalmazott meg 1945-ben az "Edvacról szóló jelentés első tervezetében" [35]. A Neumann-elvek:

- Teljesen elektronikus működés (ez Neumann idejében elektroncsöves felépítést jelentett, amit később a tranzisztoros, majd az integrált áramkörös felépítés követett)
- Kettes számrendszer használata (az összes művelet, pl. összeadás, szorzás, kettes számrendszerbeli logikai műveletekre redukálható)
- Belső memória használata
- Tárolt program elve. A számításokhoz szükséges adatokat és programutasításokat a gép azonos módon, egyaránt a belső memóriában (operatív tár) tárolja.
- Soros utasítás-végrehajtás (az utasítások végrehajtása időben egymás után történjen; ennek egy alternatívája a párhuzamos utasítás-végrehajtás, amikor több utasítás egyidejűleg is végrehajtható: ezt a lehetőséget Neumann elvetette)
- Univerzális felhasználhatóság, Turing-gép (programozhatóság; a különböző feladatok programokkal legyenek megoldva, nem pedig erre a célra épített hardverrel)
- Szerkezet: öt funkcionális egység (aritmetikai egység, központi vezérlőegység, memóriák, bemeneti és kimeneti egységek)

A Neumann modell jól működik, ha egy feladatot pontosan egy processzorhoz rendelünk. Ez a helyzet általában beágyazott rendszerekben igaz. Beágyazott rendszerben a számítógépnek vagy processzornak pontosan meghatározott feladata van. Ezért a beágyazott rendszernek gyakran nincs operációs rendszere. A Neumann-modell problémái akkor jelentkeznek, amikor a feladatok száma eltér a gépben lévő processzorok számától. A szoftverfejlesztők továbbra is a Neumann modellt akarják használni, ezért az operációs rendszer elrejti a hardverréteget, és minden folyamathoz a rendelkezésre álló processzort rendeli.

Számos publikáció foglakozik a Neumann elv javításával, mi több, ha rákeresünk ezzel a témával foglalkozó publikációkra, akkor több ezer publikációt találunk a témában. Azonban itt csupán néhányat emelek ki.

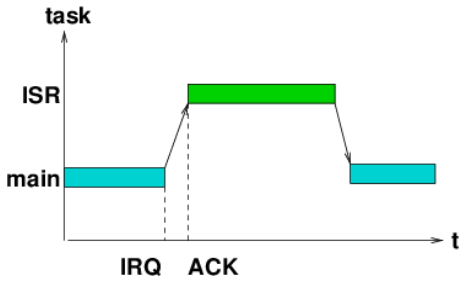
Aspray a modern számítástechnika lehetőségeit vizsgálja a Neumann elv alapján [36]. Schlansker és társai [37] egy párhuzamos feldolgozású architektúra létrehozását javasolják alkalmazás specifikus áramkörök létrehozására. A kutatás arra irányult, hogy beágyazott rendszer alkalmazásához C programozási nyelven alapuló leírásból létrehozzák a beágyazott rendszer architektúrát. Azonban a kutatók elismerik, hogy jelenleg ez a feladat még nem megoldott. Ousterhout [38] azt vizsgálta, hogy az operációs rendszerrel rendelkező gépek miért nem olyan gyorsak, mint a pusztán hardver alapú megoldások. Végh és társai pedig [S10] azt vizsgálják, hogy a feladatok párhuzamosítása milyen mélységig oldható meg és mikor érdemes ezt megtenni.

A beágyazott rendszerek párhuzamosan zajló feladatokat kell megoldaniuk, viszont a legtöbb beágyazott rendszernek nincs operációs rendszere. A több feladat megjelenése által okozott probléma azonban megjelenik egy beágyazott rendszerben is. A rendszer megszakításai okozzák ezt a gondot. A szolgáltatás megszakítása (ISR) a főprogram mellett további feladatként jelenik meg. A processzornak ezt a feladatot is végre kell hajtania [45]. A processzor nem tudja azonnal

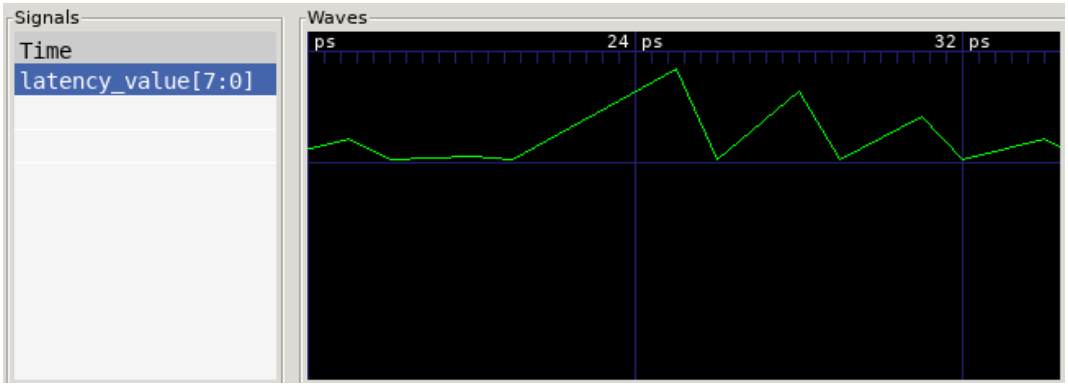
elfogadni a megszakítási kérést, ezért késés, úgynevezett késleltetés van a kérés és az elfogadás között (31. ábra). Ennek két oka van:

- a processzor a fő program kritikus területét futtatja, ha a megszakítások le vannak tiltva, vagy
- a processzor hosszabb utasítást futtat, amelyet nem lehet megszakítani.

A megszakítás elfogadásának időtartama is változhat. Ennek egyik oka, hogy az utasításátlapolásos processzoroknál ki kell üríteni a pipeline tartalmát. Ennek időtartama a a pipeline utasításainak típusától függ. A másik ok az, hogy az ISR (megszakítás kezelő program) után helyre kell állítani az előző program állapotát. A mentés időtartama attól függ, hogy az ISR hány regisztert használ. A megszakítás elfogadásának időtartama nem állandó, a késés ingadozik. A 32. ábra a megszakítási kérés teljesítésének lappangási idejét mutatja. Azaz az ISR egymás utáni kérések és első utasítás végrehajtásának időpontja között eltelt időt mutatja az ábra. Látható, hogy a kérés teljesítésének lappangási ideje nem állandó.



31. ábra: A fő program felfüggesztése és megszakítás kezelés idődiagramja



32. ábra: A megszakítás kérés lappangási idejének mérési eredménye.

Amikor egy processzor megszakítási jelet kap, és ha a megszakítás engedélyezett, akkor a processzor felfüggeszti a futási folyamatot, és megkezdi a kivétel kiszolgálását. A megszakítási szolgáltatás rutin (ISR) végrehajtása után a processzor visszatér a felfüggesztett feladathoz.
ARM processzorok esetében az ISR kivételes eseményként kezelendő, és az ARM dokumentumok a megszakítást kivételnek írják le [46]. Ezt röviden ismertetem az alábbiakban.
Ha ISR esemény következik be, és a kivétel kezelést engedélyeztük, a processzor felfüggeszti az aktuálisan végrehajtott utasítást (az úgynevezett hosszú utasításokat eldobja, és visszatérés után újraindítja a végrehajtásukat), majd a kontextus regisztereket (8 regiszter mindegyik 32 bit hosszú) menti a verem mutató által megadott veremtárba (ARM MSP - Main Stack Pointer vagy PSP - Process Stack Pointer esetén). A mentett regiszterek a következők: xPSR (Program Status Register), PC (Program Counter - tartalmazza a visszatérési címet, LR (Link Register R14) és regiszterek: R12, R3, R2, R1, R0.
Ezt követően a processzor handler üzemmódba kapcsol. Ezután a programszámlálóba (PC) kerül az ISR kezdőcím, és az LR betölti a visszatérési címet. A következő lépésben az ISR száma betöltődik az IPSR-be (Interrupt Program Status Register). Végül megkezdődik a megszakítás kezelő rutin

végrehajtása. A fentiekben leírt folyamat felemészti a processzor időt. Fontos figyelembe venni az ISR kéréstől a kivétel kezelés elindulásáig eltelt időt.
Ha a megszakítás ciklikusan ismétlődik, akkor az ISR válaszideje megnövekszik a megszakításkezelés elkezdésének lappangási idejével. Az megszakításkezelés kiszolgálásának ideje szempontjából kritikus külső események késleltethetik a kiszolgáló rutin indulását. Ilyen például az analóg-digitális átalakító mintavétele.
Mennyi idő szükséges a regiszterek mentésének elvégzéséhez („rezsiköltséghez"), és a program a PUSH/POP utasítások végrehajtásához. Ilyenkor a processzor a regiszter mentés/visszaállítás címével foglalkozik (veremtár műveletek). Ha a veremtár külső memóriában van, akkor ez a „rezsiidő" hosszabb lesz (külső memória műveletek).
Kiszámítható, egy adott processzorra, az adott órajel frekvenciával, mekkora lehet a megszakítások maximális ismétlési gyakorisága:

$$F_{MaxInt} = \frac{F_{CPU}}{C_{ISR} + C_{Overhead}}, \tag{14}$$

ahol az F_{MaxInt} a megszakítás maximális ismétlési gyakorisága, az F_{CPU} a vezérlő órajel frekvenciája, a C_{ISR} a megszakítás kezelő rutin (ISR) elvégzéséhez szükséges órajel ciklusainak száma, $C_{Overhead}$ pedig a regiszterek mentésére és visszaállítására (azaz program állapot mentésére és visszaállítására) fordított órajel ciklusok száma.
Gyakori megszakítás kérés teljesítésével, a processzor az ISR futtatását végzi. Ezért kevesebb időt fordít a fő program végrehajtására. Az ISR teljes végrehajtásához szükséges idő:

$$U_i = \frac{F_i\left(C_{ISR} + C_{Overhead}\right)}{F_{CPU}}, \tag{15}$$

ahol F_i a megszakítások gyakorisága. A fentiekből pedig az következik, hogy a fő program végrehajtása látszólag sokkal alacsonyabb órajel frekvenciával történi:

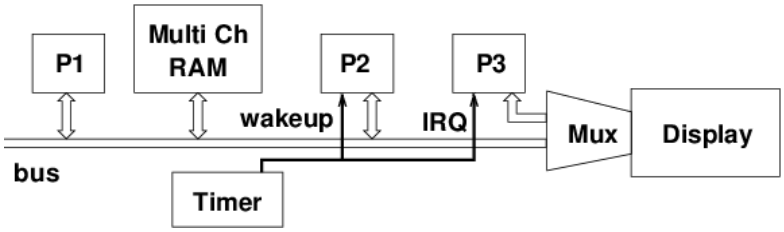
$$F_{main} = \left(1 - U_i\right)F_{CPU} \tag{16}$$

ahol F_{main} a fő program végrehajtásának látszólagos órajel frekvenciája.

7.2 Új kutatási terület: Megszakítás kiszolgálás párhuzamos feldolgozással

7.2.1 Hagyományos megoldás

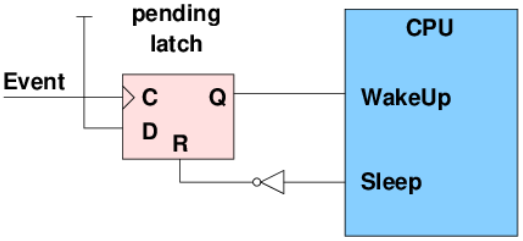
A magszakítás kezelés kiszolgálására összeállítottam egy a 33. ábra szerinti kísérleti rendszert. A kísérletben ARM lágymagos (ARM M0 processzorokat) használtam az FPGA áramkörben. A kísérletben a processzoroknak a következő feladatot kell megoldani: Az időzítő periféria megszakítást generál egy adott frekvencián (10 kHz a kísérleti rendszerben), a kivételkezelő program számolja a megszakítások számát, azaz növeli egy számláló értékét (szoftver óra). A fő program megjeleníti a számláló értékét egy kijelzőn.
A P3 processzor hagyományos módon működik. Kezeli megszakítást, azaz időzített feladatot hajt végre, majd a számláló értékét kiküldi a kijelzőre. A P3 processzoron futó hagyományos megoldás futási paramétereit hasonlítjuk össze egy a feladatot párhuzamosan (P1 és P2) végrehajtó rendszerrel. A P1 processzora fő programot végzi, míg a P2 processzor a megszakításkezelést hajtja végre. Amikor a kivételkezelés megtörtént a P2 processzor „szundi" azaz alvó állapotba kerül. A megszakítás jel hatására a P2 processzor éled és végrehajtja a kivétel kezelő programot.



33. ábra: A megszakítás kiszolgálás kísérleti rendszerének tömbvázlata

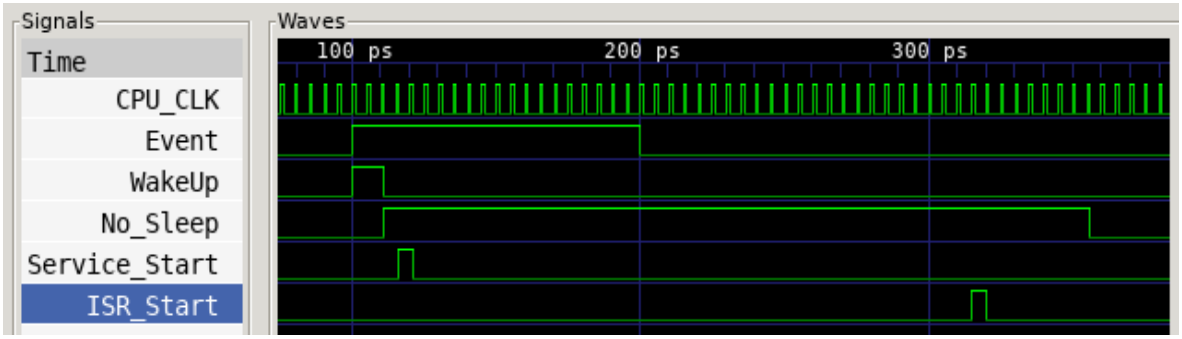
7.2.2 Párhuzamosított megoldás

A párhuzamos megoldás magában foglalja a P1 és P2 processzorokat. A P1 processzor feladata a fő program futtatása. Ez a fő program ugyanazt a feladatot látja el, mint a P3 processzor fő programja. A P1 processzor fő programját nem használjuk meg ISR kiszolgálásra. Ezt a feladatot a P2 processzor végzi. A P2 processzor párhuzamosan működik a P1-gyel, így a fő program futtatását nem kell felfüggeszteni az ISR futása közben. Ez gyorsabb végrehajtást eredményez.
A megszakítási kérelmeket a P2 processzor kezeli. Ennek a processzornak a működését úgy terveztük meg, hogy elkerülje a szokásos megszakítás kezelési folyamat végrehajtását. Ezért az időzítő perifériás jelet nem az ISR jel megszakítási kérelem bemenetére csatlakoztattuk. A processzor felfüggesztett állapotban van (alvó/szundi üzemmódban). Az időzítő által generált periféra jel csatlakozik a processzor „ébresztését” indító (WakeUp) bemenetre. Amikor a jel indítja a processzort, akkor megtörténik ISR kiszolgálása, de ezúttal a P2 fő programjaként szerepel. Mivel az ébredés az ISR jel felfutó élére történik (a processzor felfüggesztése a program elején van), azonnal megkezdődik az „ISR” kiszolgálása. Nincs szükség a megszakítás elfogadás folyamatára. Nincs szükség a fő program állapotának mentésére, mert a processzornak az ISR a fő programja. Ezért a $C_{Overhead}$ = 0 az (14) és (15)-ös egyenletekben.
Az „ISR” kiszolgálás végén a program visszaugrik a program elejére, ahol egy utasítás felfüggeszti a processzort, így szundi üzemmódban várja a következő ISR jelet. A periféria megszakítás jelét a processzor megszakítás kezelő áramköre tárolja a szolgáltatás megkezdéséig. Mivel a P2 processzorban nincs megszakítás kezelő áramkör, a jel tárolásához külső tárolót (D-retesz) telepítettünk (lásd: 34. ábra).



34. ábra: D-retesz tömbvázlata

Amikor a P2 processzor kilép a felfüggesztett állapotból, a retesz törlődik, készen áll a következő jel vételére. A P2 processzor a folyamat eredményét a MultiChRam többcsatornás (összesen 4 csatorna) memóriába írja be. Az eredményt a P1 processzor pedig olvassa a MultiChRam memóriából és a kiírja a kijelzőre. A párhuzamos megoldás szimulációjának eredményét összehasonlítva a hagyományos megoldással az 35. ábra mutatja. Az esemény (Event) jel a periféria megszakítási kérését jelzi, ez az impulzus az időzítő perifériáról érkezik 10 kHz-es frekvenciával.



35. ábra: Párhuzamos ISR kiszolgálás és hagyományos ISR szimuláció eredménye

A WakeUp jel ébreszti a P2 processzort (ez a külső retesz kimenete). A No_sleep jelet a processzor generálja, és jelzi, hogy folytatja a program végrehajtást. Ez a jel törli a külső reteszt. A Service_Start jelet a P2 processzor „ISR kiszolgálás" első utasításával generálja, amely jelzi, hogy a szolgáltatás elindult.
Az Event jel hagyományos módon megszakítja a P3 processzor programot. A P3 processzor ISR funkciójának első utasítása generálja az ISR_Start jelet. Ezen indikátorjel segítségével összehasonlíthatjuk a két ISR kezdőpontját, az egyiket, amelyet a P2-n hajtanak végre, és a másikat, amelyet a P3-on hajtanak végre. Látható, hogy az kivétel kezelési folyamat miatt (regiszterek mentése, stb.) a P3 processzor később kezdi meg az ISR folyamatot, mint a P2 processzor.

7.2.3 Négycsatornás RAM memória

A fent említett alkalmazási példában a fő program és az ISR két külön folyamatnak tekinthető. A legtöbb esetben a folyamatoknak kommunikálniuk kell egymással. Párhuzamos megoldás esetén nehezebb megoldani a folyamatok kölcsönös kizárási feltételét, mivel a két folyamat különálló processzorokon fut. Ezért terveztünk egy többcsatornás memóriát, amelynek a segítségével a processzorok kölcsönösen átadhatják az adatokat egymásnak.
Mivel az általunk tervezett memória 4 csatornás, ezért a P1 és P2 processzorok egy – egy memória csatornát használnak fel a négyből. A négycsatornás memóriát a további kutatások érdekében fejlesztettük ki.
Az FPGA technológia ismeri a kétcsatornás teljes hozzáférésű memóriákat, hiszen az úgynevezett BlockRAM memóriák két csatornásak (A és B), a memória írása/olvasása történhet szinkron módon egyszerre. Mindenik memória port rendelkezik saját vezérlő-, cím- és adat-sínnel. Tudomásunk van egy a Cypress által megépített teljesen párhuzamos két csatornás duális RAM típusú memóriáról. Azonban az általuk tervezett párhuzamos hozzáférésű memória négy csatornás, ezért egyszerre négy processzor is használhatja adatok tárolására. A memória könnyen illeszthető az ARM processzorok sínrendszerére.
A csatornákon a processzorok egyszerre olvashatják és írhatják a memória tartalmát. Az írási ütközések kezelésére külön vezérlőt hoztunk létre. Erre azért van szükség, mert két vagy több processzor egyszerre végezhet írási műveleteket ugyanarra a memória területre. Az ütközés kezelő áramkör egyszerű kialakítású, nagyobb prioritást biztosít a nagyobb indexszámmal rendelkező csatorna írásának. Ha a processzorok különböző memóriacímeket írnak, nincs ütközés, mivel az írási műveletek egyszerre és külön memória területen zajlanak, és nem zavarják egymást.
A Peterson algoritmus lehetővé teszi a nagyobb adatmennyiség írását olvasását. Ezt az algoritmust azonban nem kell használnunk, amikor az adatmennyiség kicsi, például nem nagyobb, mint a processzor regiszter mérete, mert akkor a processzor atomi hozzáféréssel olvassa/írja az tárat. Az atomi hozzáférés azt jelenti, hogy elegendő az egy utasítást használni az adatok olvasására/írására, így a másik folyamat nem képes megszakítani a műveletet.

7.3 A FIVE módszer párhuzamosíthatóságának vizsgálata

Napjainkban elterjedtek a több processzormagot tartalmazó kisméretű és alacsony energia felvételű, de jelentős számítási teljesítményt nyújtó rendszer a chip-en (System on Chip – SoC, vagy Programmable System on Chip – PSoC) megoldások. A több processzormag együttes munkájával elérhető, hogy bonyolult számítási feladat is végrehajtható legyen akár valós idejű rendszerek esetén is. Ennek köszönhetően építhetők olyan kisméretű mobil robotok, amelyek irányítása fuzzy viselkedés-leírással valósulhat meg. A fuzzy viselkedés-leírás alapja a fuzzy interpolációs eljárás, a FIVE (Fuzzy Interpolation in Vague Environment) módszer. A fuzzy interpoláció jól használható a beágyazott rendszerek számítási teljesítményének vizsgálatában is.[41]

7.4 Szakirodalmi áttekintés - Fuzzy Interpoláció FIVE módszer

Az etorobotika területén a magyar kutatások világelső helyezettek. Miklósi Ádám és társai jelentős kutatásokat végeztek a témában – a 2021-es és 2020-as adat [39], [40]. Miklósi Ádám szerint a jövő robotjait a kutyák viselkedésének alapján kell megvalósítani. Az említett szerző kutatásaiban a BME etorobotika és a Miskolci Egyetem fuzzy rendszerek kutatócsoportja is részt vett. Korondi Péter és társai ([42], [43]), Kovács Szilveszter és társai [41], [44], Bartók Roland [S22] által végzett kutatások a fuzzy interpoláció alkalmazását vizsgálják az etorobotika területén.

Egy mobil robot feladata lehet viszonylag egyszerű vagy pedig meglehetősen összetett. Ilyen lehet akár egy vonalkövető robot, amely elkerüli az akadályokat, önállóan parkol, feltölti az akkumulátorát. Vagy lehet esetleg egy olyan robot, amely valamilyen etológiai viselkedésmodell alapján utánozza egy állat viselkedését, lásd etorobotika. Utóbbi alkalmazások esetén már nem létezik ismert és kezelhető bonyolultságú matematikai modell a feladat megoldására, ezért érdemes soft compting módszereket használni. Egyik lehetőségként alkalmazható a fuzzy logika. Ekkor a rendszer matematikai modellezése helyett a rendszer viselkedését szabályokkal lehet közelíteni az elvárt viselkedéshez.

A fuzzy irányítás két részre osztható, egyik a fuzzy halmazok, másik a halmazokból alkotott szabályok. Fuzzy implikációval összekapcsolva a halmazokat, fuzzy szabályok jönnek létre. Klasszikus fuzzy esetén minden lehetséges megfigyeléshez tartozik szabálypont, a szabálybázis teljesnek tekinthető. Ez feladattól függően nagyon sok szabályt jelenthet ezzel együtt jelentős számítási teljesítményigény is szükséges. Ritka a szabálybázisnak nevezzük azt, amikor nem minden megfigyeléshez tartozik szabálypont. Ekkor azonban lehet olyan megfigyelés, amelynél nem megjósolható a rendszer működése vagy hibás működést eredményezhet.

Ilyen esetekre jelent megoldást a fuzzy szabály interpolációs módszerek (FRI – Fuzzy Rule Interpolation) alkalmazása. A módszer a megadott szabályok alapján számítja ki a hiányzó szabálypontokhoz tartozó döntést. A módszer előnye, hogy a ritka szabálybázisban elegendő megadni a legfontosabb szabályokat a rendszer működéséhez. A kis szabályszámnak köszönhetően csökken a számítási igény, így gyors szabályzó rendszerek hozhatók létre [41].

Egyik ilyen interpolációs eljárás a FIVE (Fuzzy Interpolation in Vague Environment), amely egy gyors eljárás, a szabályok hasonlóságán alapul. A szabályok hasonlóságát a szabályok egymástól való távolsága határozza meg, a közeli szabályok hasonlósága magasabb, mint az egymástól távol eső szabályok hasonlósága. A távolságon a szabálypontok Euklidészi távolsága értendő. A FIVE módszer közvetlenül felhasználja a megfigyelések (érzékelők, állapotok adatai) értékeit, illetve a kimeneti értékek is közvetlenül felhasználhatók a rendszer működtetéséhez, tehát nem szükséges a klasszikus fuzzyban ismert fuzzyfikálás és defuzzyfikálás művelete. A FIVE módszer alapja a Shepard-interpoláció, melynek összefüggése a következő:

$$S_0(f,x,y)=\begin{cases} f_k & \text{if } (x,y)=(x_k,y_k) \text{ néhány } k \text{ esetén}, \\ \left(\dfrac{\sum_{k=0}^{n} f(x_k,y_k)}{d_k^{\lambda}}\right) \Big/ \left(\sum_{k=0}^{n}\dfrac{1}{d_k^{\lambda}}\right) & \text{egyébként}, \end{cases} \quad (17)$$

ahol x_k, y_k, (k ϵ [0, n]) megfigyelések, f ϵ $R^2 \rightarrow R$, λ>0 és d_k az Euklideszi távolság [41]
A FIVE módszer gyors és eredményesen alkalmazható interpolációs módszer beágyazott rendszereken is. Az eljárás egyes elemeinek egymástól független számíthatósága miatt eredményesen párhuzamosítható. Ezért a módszer alkalmas számítási sebesség vizsgálatokra.

7.4.1 Amdahl képlet

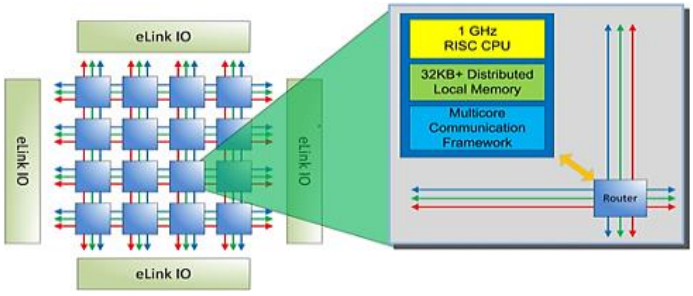
G. M. Amdahl 1967-ben jelentetett meg egy összefüggést, amely modellezi, hogy mennyivel gyorsabb a párhuzamosított algoritmus a számításhoz felhasznált processzormagok függvényében. Ez egy elméleti felső korlát a számítás elérhető gyorsítására, de az eléréséhez szükséges a programkód alapos ismerete szükséges hozzá [45]. A képlet nem tartalmazza a különféle egyéb működés közbeni műveletek időigényét (memória elérés, várakozás a merevlemezre, stb.), és nem adja meg, hogy az aktuális párhuzamosított megvalósítás mennyire hatékony [S8].
Az Amdahl formulához a párhuzamosított algoritmus számítási idejét két részre kell osztani:a párhuzamos futásban eltöltött időre és a soros futásban eltöltött időre. Az algoritmus soros futásban eltöltött ideje azon időszakaszokból adódik össze, amely tartalmazza a feladatkiosztás idejét és az egyéb várakozási időket, például valamely hardverelemre várakozás vagy késleltetési időket, stb. [45]. Az Amdahl formula a következő:

$$S(P)=\frac{1}{1-t_P+\frac{t_P}{P}}, t_p=\frac{T_p}{T} \quad (18)$$

ahol, *P* a magok számát jelenti, A T_P a párhuzamos számításban eltöltött időt, *T* pedig a teljes számítás idejét jelöli. A t_P a párhuzamos számításban töltött idő hányadát jelöli. Az $1-t_P$ a soros szakaszban eltöltött időt jelöli. *S* a számítási sebességnövekedés értéke.
Az Amdahl törvény rámutat az egyprocesszoros szemlélet korlátaira, ha azt kiterjesztik több processzorral történő számításra [S8].

7.4.2 A teszteléshez használt beágyazott rendszer

A tesztelésekhez a „szuperszámítógépnek” is nevezett Parallella nevű hitelkártya méretű számítógépet használtuk. Felépítésében egy 16 magos Epiphany III áramkör ás egy Xilinx Zynq (667 MHz, két magos ARM) SOC rendszer található.



36. ábra: Adapteva Epiphany tömbvázlata

A rendszer 1 GB DDR3-as memóriát használ, amelyből 32 MB használható osztott memóriaként a Zynq és az Epiphany közötti kétirányú adatcserére. Az Epiphany különlegessége, hogy a processzormagok (eCores) az eMesh nevű nagy sebességű kétdimenziós duplex router-hálózaton

keresztül kommunikálnak. Egy-egy routerhez a legközelebbi négy mag közvetlenül kapcsolódik [47].

7.4.3 Vizsgálati módszerek

Ebben a fejezetben két vizsgálatot mutatok be. Egyik a Shepard interpoláció kétdimenziós megvalósítása. A Shepard interpoláció FIVE módszer alapja és egyben önmagában a legköltségesebb eleme számítási igény vizsgálatának szempontjából. Ebben az esetben a teljes számításra fordított idő kerül mérésre.
A kétdimenziós Shepard interpoláció formája a következő:

$$f(x) = \frac{\sum_i w_i(x) y_i}{\sum_i w_i(x)}, \quad w_i(x) = \left(\frac{1}{|x - x_i|}\right)^p, \quad p = 2 \tag{19}$$

ahol w_i a súly, x_i, y_i, az ismert pontok, valós számok. Az x az a pont, ahol az y számított, valós számok. A p egy kitevő, jelen esetben 2.
A tesztben a fuzzy interpoláció többszöri futásának ideje kerül rögzítésre. A teszteket különböző felépítésű rendszereken futtatuk:

- Parallella, csak az Epiphany magokon, minden mag egyszerre számol;
- Parallella, csak a Zynq, operációs rendszer futtatása közben;
- ZedBoard, Zynq (800 MHz), operációs rendszer nélkül;
- Intel Core i5 4200H (3,3 GHz és 0,7 GHz, energiatakarékos mód), operációs rendszerrel

A második vizsgálatban a rendszer nagy számú szabálybázist tartalmaz, így az egyes processzorok számára is több szabálybázis teljes számítása a feladat. A teszt rámutat arra az esetre, amikor a több processzor esetén romlik a párhuzamosítás jósága a feladatok kiosztása és az adatmozgatásokra használt idők miatt. Ekkor mért adatként szerepel a memória elérés ideje (írás illetve olvasás), a számításra fordított idő, és a teljes számításra fordított idő. Az Amdahl formulában a T_P paraméter az alábbiak szerint épül fel:

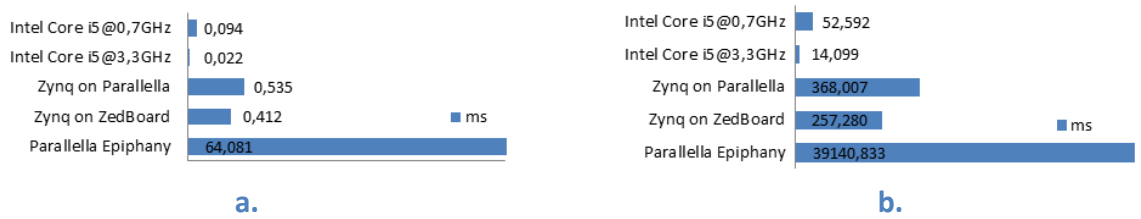
$$T_p = \frac{\sum_{magok\ darabszáma}(T_{számítás} + T_{MOlvasás} + T_{Mírás})}{magok\ darabszáma} \tag{20}$$

ahol $T_{számítás}$ a hasznos számítással töltött idő, $T_{MOlvasás}$ a memória olvasással töltött idő, $T_{Mírás}$ memória írásával töltött idő. A memória írás és olvasás idejét az Epiphany mag szempontjából mértük. T_P párhuzamos szakaszban töltött idő [S16].

7.5 Új kutatási terület: Fuzzy interpoláció vizsgálata

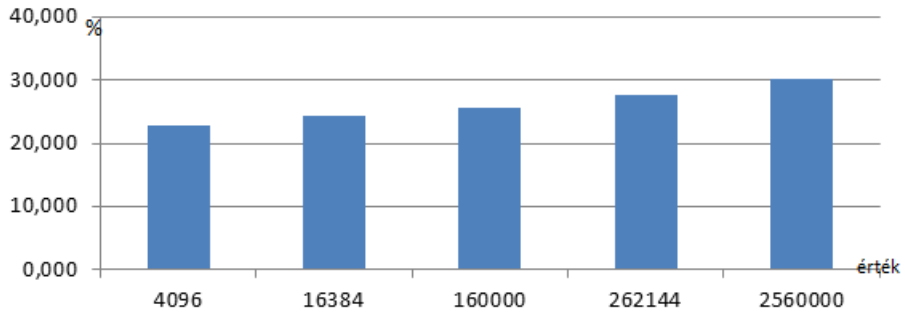
7.5.1 Első teszt

Elsőként a Shepard interpolációval végrehajtott tesztek eredményeit ismertetem. A számítások 4096, 16 384, 160 000, 262 144 és 2,56 millió értékpárra futottak le, a kapott értékek a teljes számításhoz szükséges mért idők, melyek közül a legkevesebb és a legtöbb tesztadatra elvégzett számítás grafikonjai láthatók az alábbi ábrán (37. ábra).



37. ábra: Teszt eredmények 4096 értékre (a), 2,56 millió értékre (b)

A Shepard interpoláció tartalmaz lebegőpontos számításokat, amelyek hosszabb ideig tartanak az ARM magon illetve az Epiphany processzoron. Megjegyzem, hogy az Adapteva adatlapja szerint az Epiphany lebegőpontos számítási teljesítménye még elmarad az asztali számítógépek processzoraitól.
A teszt eredményei megmutatták azt is, hogy az operációs rendszer működése, a feladatok ütemezése és egyéb feladatok mennyivel lassítják a számítások végrehajtását. A Parallella és a ZedBoard ugyan azt a Zynq SoC-t tartalmazza, így lehetőség volt arra, hogy operációs rendszer futtatása közben és operációs rendszer nélkül is lefussanak a számítások. Amennyiben az operációs rendszer nélküli számításokat egységnek tekintjük (100%), akkor az alábbi ábrán (38. ábra) ábrázolt eredmények azt mutatják, hogy az egységhez képest mennyivel hosszabb ideig tartottak a számítások (%).

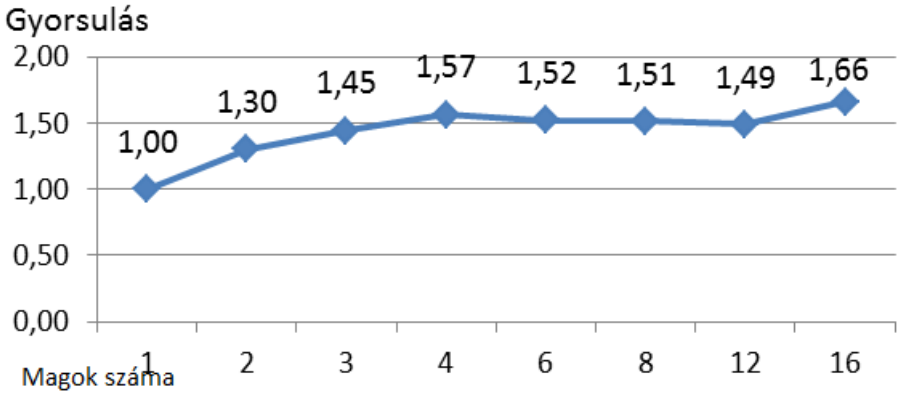


38. ábra: Az operációs rendszer mellett végzett teszt eredmény

Az operációs rendszer használata 20-30%-al növeli meg a számításhoz szükséges időt, mivel az operációs rendszer a saját feladatai ellátására is fordít processzor időt. Ez a különbség hosszabb számítások esetén nagyobb, mivel a hosszabb ideig tartó számítások alatt az operációs rendszer is többször veszi függeszti fel az éppen futó folyamatot.

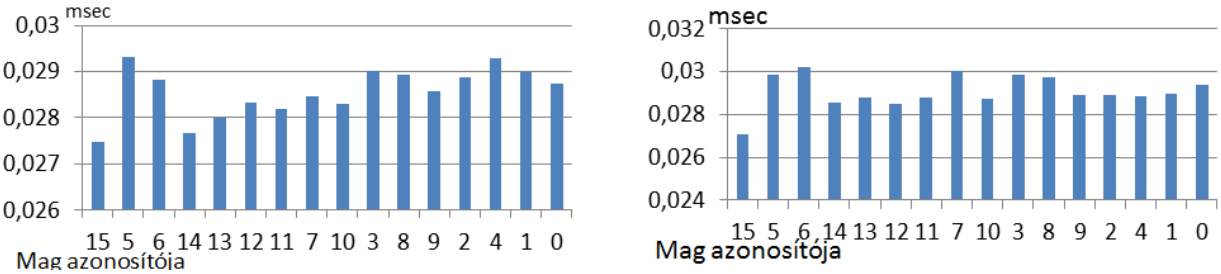
7.5.2 Második teszt

A második teszt során 1-16 darab Epiphany mag számította ugyan annak a szabálybázisnak az eredményét. A következő grafikonon az Amdahl-törvény alapján számított gyorsulás értéke került ábrázolásra (39. ábra).



39. ábra: Gyorsulás a mért adatok alapján Amdahl törvénye alapján

A gyorsulás a 16 mag együttes munkája ellenére is csak másfélszeres, aminek az oka, hogy a legrosszabb eset szerepelt a tesztben. Ebben az esetben a feladatok kiosztásának ideje és az adatmozgatások ideje közel összemérhető a számítás idejével. Tehát ilyen esetben nem érhető el jelentősen rövidebb számítási idő. Ezért ilyen esetekben át kell szervezni a program párhuzamosítását, hogy hosszabb idő jusson a számítások elvégzésére.



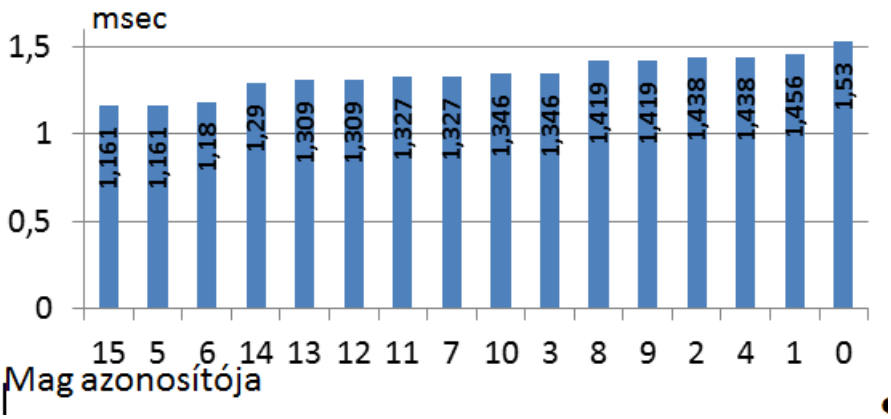
a. b.

40. ábra: a. A memória olvasás ideje (adatok gyűjtése), b. A memória írás ideje (eredmények közlése)

Az Epiphany magok a rendszer osztott memóriáján keresztül kétirányú kommunikációt végeztek a Zynq-kel. Tehát versenyhelyzet alakult ki a Zynq és az Epiphany magok illetve maguk az Epiphany magok között is. Ezt szemlélteti az alábbi két grafikon.

A grafikonon a magok abban a sorrendben kerültek ábrázolásra, ahogy elkészültek a számításokkal. A függőleges tengelyen a memória hozzáférés ideje jelenik meg. Az eredményekből valószínűsíthető, hogy a magasabb azonosítóval ellátott magok fizikailag közelebb helyezkednek el a memóriához. A 15-ös sorszámú mag végzett a leghamarabb a számítással és viszonylag rövid idő alatt hozzá is fért a memóriához. A többi mag esetén versenyhelyzet alakult ki és már jóval hosszabb időre volt szükség az adatok írásához, olvasásához.

A következő ábrán (41. ábra) a számítás befejezéséhez szükséges időket mutatja a grafikon, amelyen abban a sorrendben jelennek meg a magok, ahogyan végeztek a számításokkal és a Zynq olvasni tudta az eredményeket, illetve az eredmények beírását jelző memóriaterületen a megfelelő jelzést észlelte a Zynq ARM. A kész jelzés az Epiphany magok belső memóriájába került.



41. ábra: Az egyes magoknak a feladatra fordított ideje

A fuzzy interpoláció számításának gyorsítását párhuzamosítással 1–16 magon mértük a Parallellával végzett mérések során. A Zynq kettős ARM magja volt a felügyeleti processzor, és az Epiphany magok végezték a számítási feladatokat. A vizsgálatok során a fuzzy interpolációban 16 szabályalappal dolgoztunk, szabályalaponként 2 szabállyal.
A számítási feladat elvégzése összemérhető volt az operációs rendszer feladatainak hosszával (az operációs rendszer ütemezője és a memória elérési ideje). A gyorsulást Amdahl törvényével számoltuk a mért soros és párhuzamos számítási idők alapján.

7.6 Új tudományos eredmények

A fejezetben ismertetetett a többprocesszoros környezetben megvalósított újfajta kivétel kezelési megoldás felgyorsítja a párhuzamos feladatokat ellátó beágyazott rendszereket. Az egyes megszakításkezelések egy-egy lágymagos processzoron való futtatással felgyorsítják a végrehajtási sebességet. Egy új négycsatornás szinkron írható-olvasható memóriát hoztunk létre a processzorok közötti adatcsere érdekében [S12].
Az algoritmusok párhuzamosításának lehetőségét vizsgáltam a robotirányítások fuzzy logikával történő megvalósítására. Ezen belül a viselkedésirányításban használt fuzzy interpoláció párhuzamosításának lehetőségét javasoltam SOC rendszereken [S16].
Az etorobotikában használt fuzzy interpolációs módszer, mint az állatok (kutyák) viselkedésének leírása a szabálybázis elemeinek növekedésével egyre bonyolultabb rendszer alakítható ki, ezért új eredményként a rekonfigurálható fuzzy állapotgépet javasolom a viselkedés leírás megvalósítására.

8 Nagy pontosságú hibrid adatgyűjtő rendszer

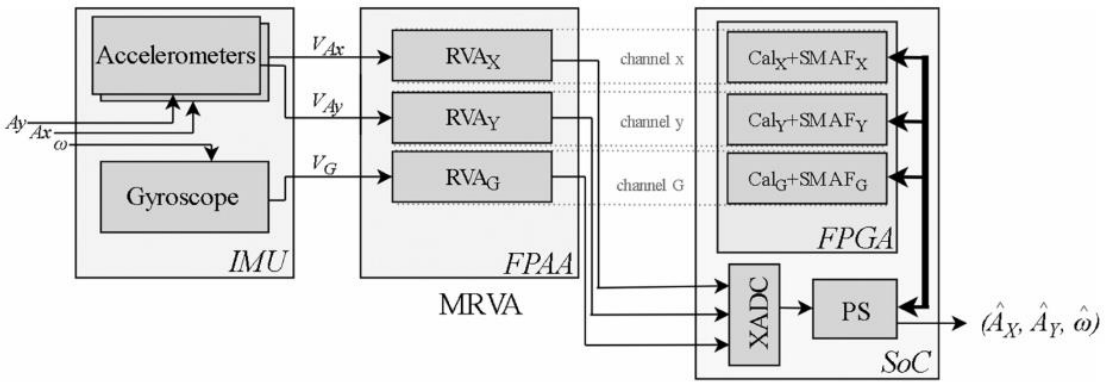
Jelen fejezet kutatási eredményeit közösen Ahmed Bouzid PhD hallgatóval végeztem, a járművek és mobil robotok kis sebességgel történő mozgását és helymeghatározását vizsgáltam. A fejezet az [S17], [S20] és [S21] publikációk alapján készült.

8.1 Szakirodalmi áttekintés

A kutatás célja a tehetetlenségi (inerciális) mérések pontosságának javítására irányul. A tehetetlenségi mérés az inerciális érzékelő által szolgáltatott jelek segítségével a robot mozgására vonatkozó információkat gyűjtünk. Az érzékelők a mindennapi használatban lévő eszközeinkben is jelen vannak (pl. okos telefon). A rezgéselemzésben használt gyorsulásmérők [50], [51] első használata a helyzet meghatározásban jelen kutatás része volt. Jelen fejezetben feldolgozott fizikai mennyiségeket olcsó érzékelő használatával mértük. Ennek a hátránya az érzékelők által keltett zaj, ami torzítja a mérések pontosságát. A sorozatos integrálások következtében összegződött mérési hibák a sebesség és helyzet mérésének következtében keletkeztek. Mivel a zaj szűrése egy redundáns folyamat, ezért egyszerűbb átruházni a feladatot egy hardveres gyorsítóra, amely szisztematikusan elvégzi a feladatokat. Az alacsony költségű érzékelők használata előnyt jelenthet a megvalósított rendszer és / vagy a tömeggyártás költségei szempontjából; ugyanakkor nagy számítási erőforrásokat igényel, mivel a zajszűrés hardverigényes folyamat.
Liu és társai [58] által javasolt kalibrálási módszer becsli a gyorsulás mérési hibáit. Ding pedig olyan zajszűrő algoritmust javasolt [59], amely a jel modulusának maximumát használja. Más megoldások a zajszűrésre egy mikrovezérlő és egy FPAA használatát javasolták [60].
Az FPAA által használt analóg jelet még digitalizálni kell ráadásul ezen áramkörök erőforrásai nagyon korlátozottak. Az FPGA és SOC áramkörök használata és a beépített analóg-digitális átalakító (XADC) lehetővé teszi az FPAA és SOC áramkörök ötvözését az érzékelőktől érkező jelek feldolgozására. Az XADC hátránya, hogy nem képes 1 V-nál nagyobb amplitúdójú jelek feldolgozására. Ezért egy olyan megoldást javasoltam, amely az FPAA-SOC rendszer illesztésével megnöveli a mérési pontosságot és az 1 V-os a mérési tartományt kiterjeszti egészen az FPAA tápfeszültségének értékéig (5 V).

8.2 Új kutatási terület: Hibrid analóg-digitális rekonfigurálható rendszer

Az 42. ábra által ábrázolt beágyazott rendszer tömbvázlata a jelek feldolgozásának menetét is mutatja. A hibrid (FPAA+SOC) rendszer ARM alapú processzoraival, analóg és digitális rekonfigurálhatóságával biztosítja a rendszer rugalmasságát. Mivel az érzékelők által szolgáltatott jelek nagyobbak az 1 V-os XADC mérési tartománynál, ezért a rekonfigurálható feszültség illesztő áramkörrel illesztjük az integrált XADC-hez.



42. ábra: Beágyazott adatgyűjtő/feldolgozó rendszer tömbvázlata

Az érzékelők feszültség illesztése a következő módszerrel történik: Az FPAA tápfeszültsége Vdd=5 V. A klasszikus mérési quantum (vagy lépés feszültség V_{LSB})számítása nem járható út:

$$q = \frac{V_{ref}}{2^N} \ (vagy \ \frac{V_{ref}}{2^N - 1}); \tag{21}$$

ahol V_{ref} a referencia feszültség, N pedig a mennyiség ábrázolásában használt bitek száma. Esetünkben a megfelelő lépés feszültség a következő képlet adja:

$$q = \frac{V_{refXADC}}{2^{NxXADC}}; \tag{22}$$

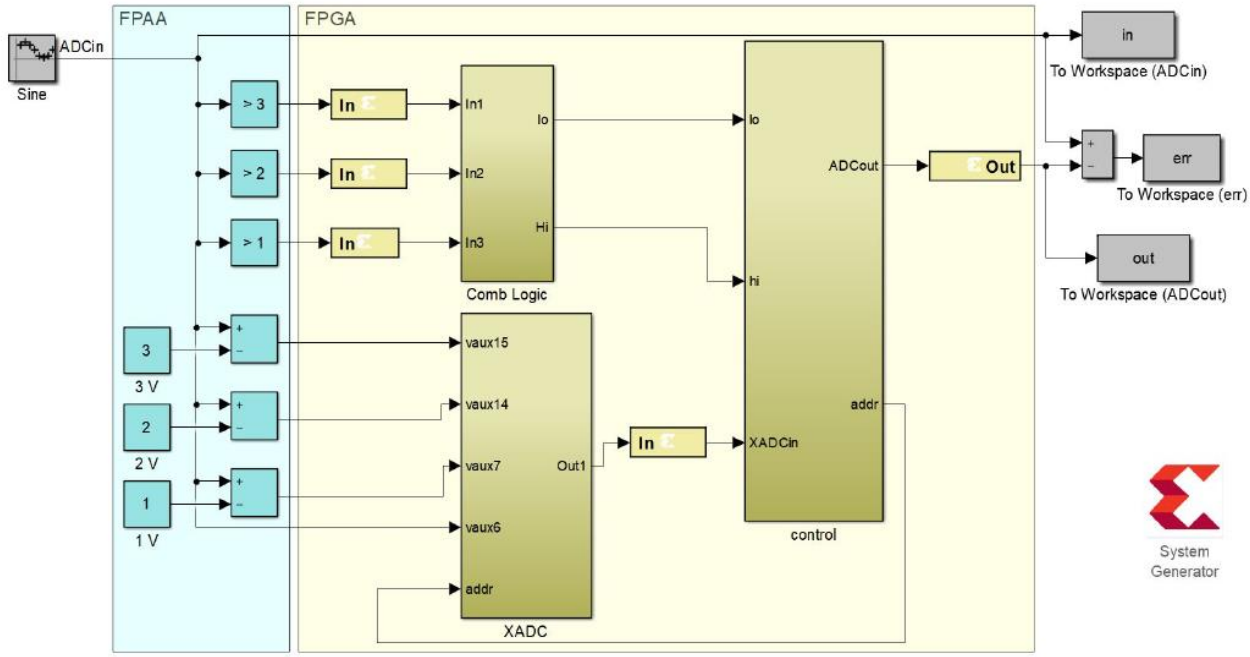
ahol az eredeti V_{ref} feszültséget csökkentjük az $V_{refXADC}$ méretére.
A mérési tartományt pedig felosztjuk 1 V –os mérési intervallumokra. Az XADC 12-biten ábrázolja az átalakítás eredményét, ezért a lépés feszültség értéke 244 µV lesz.

$$q = \frac{1\,V}{2^{12}} = 244\ \mu V; \tag{23}$$

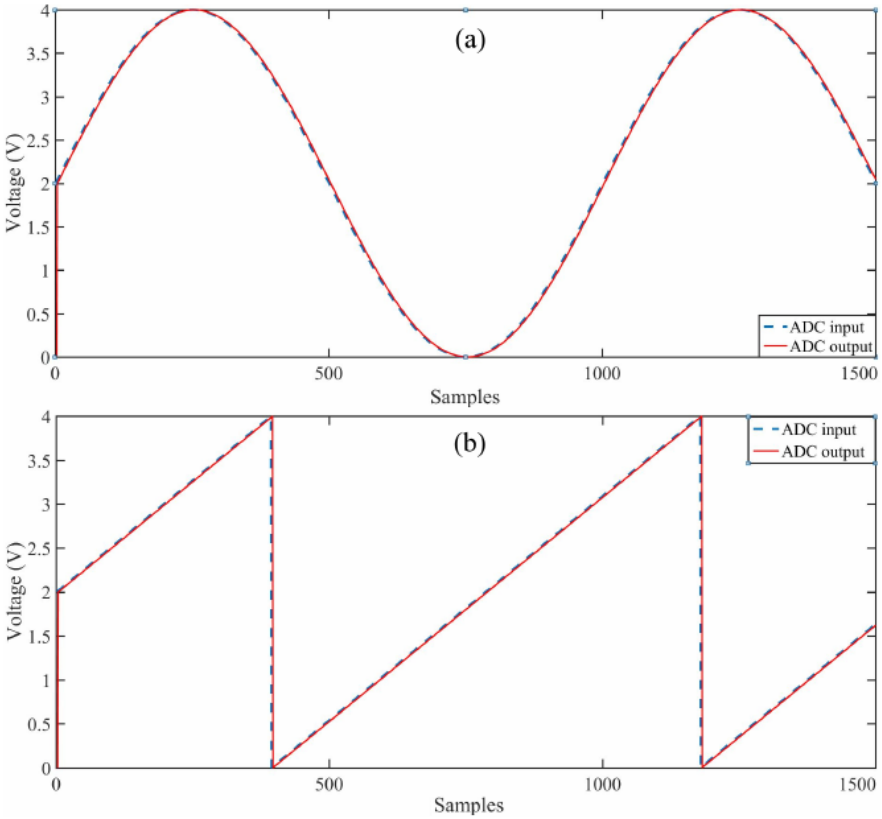
A mérési intervallum felosztása 1 V-os tartományokra az FPAA-ban (RVA tömb) történik:

$$XADC_{i\_in} = \begin{cases} ADCin \ if \ ADCin \ \leq 1\,V; \\ ADCin - 1\,V \ if \ 1\,V < ADCin \leq 2\,V; \\ ADCin - 2\,V \ if \ 2\,V < ADCin \leq 2\,V; \\ ADCin - 1\,V \ if \ 3\,V < ADCin \leq 4\,V; \end{cases} \tag{24}$$

ahol $XADC_{i\_in}$ az analóg-digitális átalakító i.-ik bemenetét jelenti, ADC_{in} pedig az érzékelő által szolgáltatott feszültség.
A 43. ábra mutatja az új átalakító megvalósítását. Az XADC csatornák kiválasztása a (>3, >2, >1) komparátorok kódolásával történik, míg XADC csatorna választás az átalakító multiplexerével valósul meg. A szimulációs eredményeket a 44. ábra mutatja.



43. ábra: XSG/Simulink modellje az RVA-XADC illesztésnek.



44. ábra: Az új ADC vizsgálatának szimulációs eredményei szinuszos (a) fűrészfogú (b) jel esetében

8.2.1 Az Érzékelők kalibrálása és a zaj szűrése

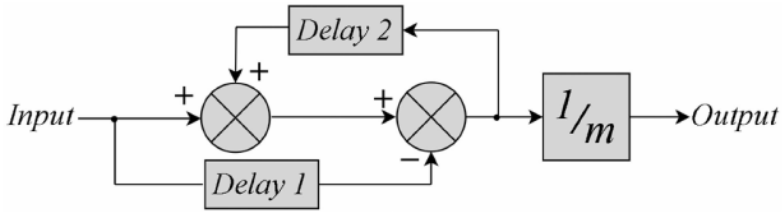
Ipari környezetben erős a zaj jelenléte, különösen az ipari környezet elektromágneses szennyezése torzítja az érzékelők adatainak értelmezését (lásd még IOT és IIOT). Ebben a kutatásban használt szenzorok a legolcsóbb típusúak, ezért a zajjal szembeni immunitásuk gyenge. Az érzékelőket kalibrálni kell a kísérlet elkezdésekor. Ezt a lépést még a kísérlet elején végezzük el, amikor az álaluk mért adat még statikus. Ebben az állapotban az érzékelők ideális esetben nulla körüli feszültséget szolgáltatnak, de a gyakorlatban ez sosincs így.
A kalibrálási eljárás a mi esetünkben 1000 minta (250 ms) megfigyeléséből áll, amikor az érzékelők nyugalmi állapotban vannak, a mérések eredményét átlagoljuk és kivonjuk az érzékelők által adott pillanatnyi értékből.

8.2.2 Simított átlagolású szűrés (SMAF - Simple Moving Average Filter)

Az SMAF egy előre meghatározott számú minta átlagolásából áll. A minták száma m. Az átlagot a következő egyenlettel számoljuk:

$$y(j) = \sum_{i=0}^{m-1} x(i+j); \tag{25}$$

Ahol x(j) és y(j) a szűrő bemenete illetve kimenete az adott j idő pillanatban.



45. ábra: XSG-vel megvalósított SMAF tömbvázlata

Valós idejű számítások esetében m mintavételezési idő késleltetést okoz a (25). egyenletben magadott szűrő. Ezért valós idejű szűrés megvalósítása érdekében az alábbi képlettel számoljuk ki az átlagot:

$$\begin{cases} y(j) = \frac{1}{m}\{\left[\sum_{i=0}^{m-1} x(i)\right] - x(j-m), j \geq 0; \\ x(i) = 0, j < 0; \end{cases} \quad (26)$$

A 45. ábra ennek a megvalósításnak a tömbvázlatát mutatja, amelyet a Xilinx System Gerentárorral valósítottunk meg. A Delay 1 és Delay 2 a megfelelő késleltetéseknek felel meg (m és 1). A megvalósított algoritmus a (26) egyenlet alapján a következő egyenlet alapján történt:

$$y(j) = \begin{cases} y(j-1) + \frac{1}{m}[x(i) - X(j-m)], j \geq 0; \\ x(j), j < 0 \end{cases} \quad (27)$$

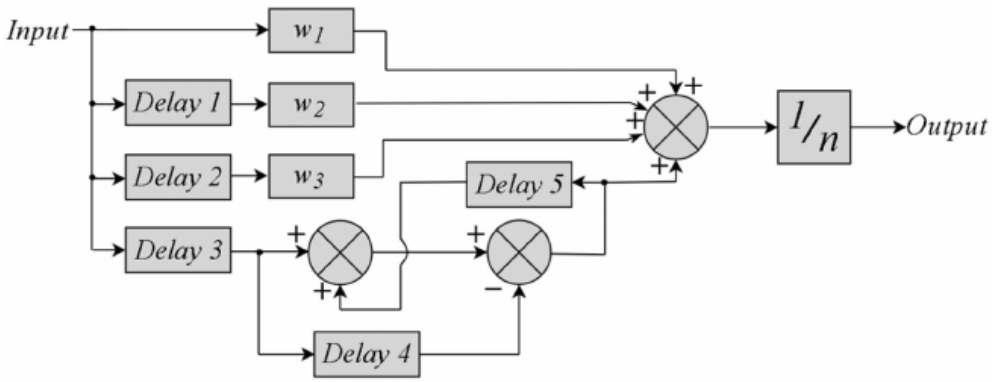
A szűrő egy órajel alatt végzi el a műveleteket, viszont a pipline struktúra miatt a szűrő holtideje $\frac{m+1}{S_r}$ órajel ciklus, a folyamat elején ezért ennyi mintát figyelmen kívül kell hagyni.

8.2.3 Súlyozott MAF szűrő (WMAF)

A WMAF általános egyenlete felírható úgy, mint:

$$\begin{cases} y(j) = \frac{1}{m}\left[\sum_{i=0}^{j} w(i)x(i) - w(j-m)x(j-m)\right] i \geq 0; \\ x(i) = w(i) = 0, \quad i < 0; \end{cases} \quad (28)$$

Ahol w ∈ W, W a súlyozott átlagok halmaza.

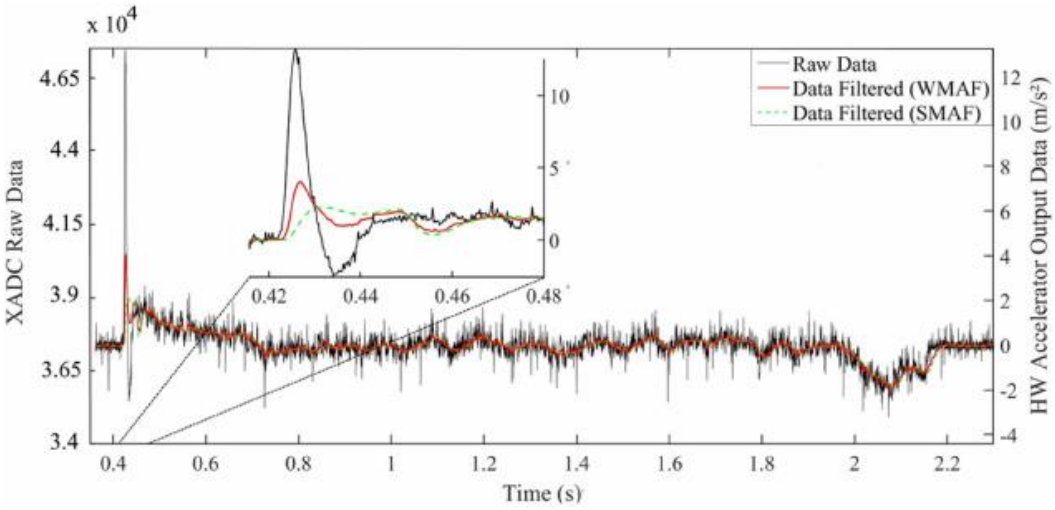


46. ábra : XSG-vel megvalósított WMAF tömbvázlata

A 3 mintavételezett jel átlagolására megvalósított WMAF a 46. ábran látható, ahol w_i jelenti az mennyiség átlagát, n = m + w_1 + w_2 + w_3 – 3 a minták száma, a késleltetés pedig 1, 2, 3, m-3 mintavételezési periódus.

8.3 Új tudományos eredmények

A kísérletek célja az SMAF és a WMAF hatásának tesztelése volt, a mozgásban lévő robot által szolgáltatott valós gyorsulásmérési adatokra. A járműre szerelt egytengelyes gyorsulásmérőből gyűjtött adatokat, miközben a robot egy 400 mm-es pályán egyenes vonalú mozgása mellett mértünk. A mérés kezdete t=0,42 s és 2,18 s-nál volt. A SMAF (m = 100) és a WMAF (m = 100, w1 = 20, w2 = 10, w3 = 5) által szolgáltatott eredmények a 47. ábran láthatók.



47. ábra Adatgyűjtés és mért eredmények feldolgozásának idődiagramja

A szűrők alul áteresztő szűrőként működnek, mivel a kiugró amplitúdójú impulzus jeleket megszűrik. Néhány ilyen impulzus hasznos információt is tartalmaz. Az ábrán jól látható, hogy a szűrt jel simább és követi az eredeti jel tendenciáját. Viszont a negatív amplitúdójú impulzust (0,42 s) a rendszer szűrte és amplitúdójának mértékét csökkentette.

A fejezet bemutatta hibrid újrakonfigurálható adatgyűjtő rendszer elemeit. A hardveres kalibrálás és két különböző szűrő megvalósítását javasoltuk és teszteltük zajos analóg inerciális érzékelők használatával. Megvalósítottuk egy MRVA egységet, amely a feszültségeket az XADC által megengedett tartományon belül szolgáltatja. A kísérleti eredmények elemzéséből kiderül, hogy a jelentős zaj az MRVA második csatornáján van. A megvalósított rendszer mérési pontossága 99,92% -ot az első csatorna (X tengelyes gyorsulásmérő), a második (Y tengelyes gyorsulásmérő) pontossága 99,72% és a harmadik (Giroszkóp) csatorna esetében és 99,97%-os pontosságot értünk el.

9 Köszönetnyilvánítás

A dolgozatban ismertetett kutató munka az EFOP-3.6.1-16-2016-00011 jelű „Fiatalodó és Megújuló Egyetem – Innovatív Tudásváros – a Miskolci Egyetem intelligens szakosodást szolgáló intézményi fejlesztése” projekt részeként – a Széchenyi 2020 keretében – az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósul meg.

10 Felhasznált Irodalom

[1] N. TREDENNICK: The Case for Reconfigurable Computing; Microprocessor Report, Vol. 10 No. 10, 5 August 1996, pp 25–27.

[2] HARTENSTEIN, R. 2001. A decade of reconfigurable computing: a visionary retrospective. In *Proceedings of the Conference on Design, Automation and Test in Europe (DATE 2001)* (Munich, Germany). W. Nebel and A. Jerraya, Eds. Design, Automation, and Test in Europe. IEEE Press, Piscataway, NJ, 642–649.

[3] KELEMEN A, IMECS M., *Vector Control of AC Drives*, OMIKK Publisher Budapest, ISBN 963 593 140 9, Budapest, Hungary, 1992

[4] BEIERKE S., *Rapid Implementation of a Field-Oriented Control Method for Fixed-Point DSP Controlled Asynchronous Servo Drives,* European Power Electronics Chapter Symposium on Electric Drive Design and Applications, organized by EPFL Lausanne, Switzerland; 19-20 Oct. 1994, pp. 361-365.

[5] CIRSTEA M., AOUNIS A., DINU A., McCORMICK M.: A VHDL Approach To Induction Motor Modelling, Control 2000 Conference CD-ROM and Book of Abstracts Control 2000, Oxford, England, 2000. pp. 90.

[6] BELMIMOUN M. H., MONMASSON E., SAMBUIS E., *Modularity in Code Development for DTSFC Algorithms Implementation on a Fixed-Point DSP,* PCIM 2002 Power Electronics Intelligent MOTion Power Quality, May 12-16, 2002 Nuremberg, Germany, pp.129-135

[7] MACIEJOWSKI J. M., Re-configurable Control Using Constrained Optimization, Proceeding of European Control Conference ECC'97, Bruxelles, Belgium, 1997, pp. 107-130.

[8] TRZYNADLOWSKI A. M; BLAABJERG F.; PEDERSEN J. K., PATRICIU Niculina, The Tandem Inverter: Combining the Advantages of Voltage-Source and Current-Source Inverters, Proceedings of Applied Power Electronics Conference, APEC'98, Anaheim, USA, 1998, pp. 315-320.

[9] IMECS M., BIKFALVI P, NEDEVSCHI S., VÁSÁRHELYI J.: *Implementation of a Configurable Controller for an AC Drive Control a Case Study*, Proceedings of IEEE Symposium on FCCM 2000, Napa, California, USA, 2000 pp. 323-324.

[10] VÁSÁRHELYI J.: *Run-Time Reconfiguration of AC Drive Controllers,* Dagstuhl Seminar 0261 on Dynamically Reconfigurable Architectures, June 25-31, 2000, Dagstuhl, Germany

[11] VÁSÁRHELYI J.: Reconfigurable Architectures for Vector Control Structures of Induction Motor Drives, PhD értekezés, 180 p. Megjelenés/Fokozatszerzés éve: 2004

[12] M. IMECS, A. M. TRZYNADLOWSKI, I. I. INCZE AND C. SZABO, "Vector control schemes for Tandem-converter fed induction motor drives," in IEEE Transactions on Power Electronics, vol. 20, no. 2, March 2005, doi: 10.1109/TPEL.2004.842952. pp. 493-501

[13] IMECS M., INCZE I. I., VÁSÁRHELYI J., SZABÓ Cs.: Tandem Converter Fed Induction Motor Drive Controlled With Re-Configurable Vector Control System, PCIM 2001 Power Electronics Intelligent MOTion Power Quality, June 19-21, 2001 Nuremberg, Germany, pp. 341-346.

[14] VÁSÁRHELYI J., IMECS M., INCZE J.J., SZABÓ Cs: Module Library for Rapid Prototyping and Hardware Implementation of Vector Control Systems, INES 2002 IEEE International Conference on Intelligent Engineering Systems, May 26-28, 2002, Opatija, Croatia, ISBN 953-6071-17-7, pp. 447-452.

[15] IMECS M., VÁSÁRHELYI J., INCZE J. J., SZABÓ Cs., Vector Control Of Tandem Converter Fed Induction Motor Drive Using Configurable System On A Chip, INES 2001 IEEE International Conference on Intelligent Engineering Systems, September 16-18, 2001, Helsinki-Stockholm-Helsinki, Finland-Sweden, pp. 489-495.

[16] VÁSÁRHELYI J., IMECS M., INCZE J. J., SZABÓ Cs.: Reconfiguration Generated Perturbations In The Vector Controlled AC Drives, Power Electronics Intelligent MOTion Power Quality PCIM 2002, May 12-16, 2002 Nuremberg, Germany, pp. 219-225.

[17] VÁSÁRHELYI J., IMECS M., SZABÓ CS, INCZE J.J., ÁDÁM T., Implementation Characteristics of Library Modules for Vector Control System for Tandem Converter Fed AC Drive, Transactions on Automatic Control and Computer Science, IV. International Conference on Technical Informatics CONTI'2002, Oct. 17-19, Timisoara, Romania, ISSN 1224-600X, pp. 13-18.

[18] IMECS M., INCZE J. J, SZABÓ Cs., VÁSÁRHELYI J. Simple Approach For Induction Motor Control using Reconfigurable Hardware, Proceedings of the National Conference on Automation CNAE2002, ISBN 973-8352-83-5, Galati, Romania, pp. 158-164.

[19] HARTUNG, F., KUTTER, M.: *Multimedia Watermarking Techniques*, Proc. IEEE, vol. 87, No7, 1999.

[20] TURÁN, J.: Fast Translation Invariant Transforms and Their Applications, ELFA, Kosice, 1999.

[21] NORMAND, N., GUEDON, J. P.: *La transformee Mojette: une representation recordante pour l'image*, Comptes Rendus Academie des Sciences de Paris, Theoretical Comp. SCI. Section, 1998, pp. 124 – 127.

[22] GUEDON, J. P., PARREIN, B., NORMAND, N.: *Internet Distributed Image Databases*. Int. Comp. Aided Eng., Vol. 8, 2001, pp. 205 – 214.

[23] KATZ, M.: Questions of uniqueness and resolution in reconstruction from projections. Springer Verlag, Berlin, 1977. pp.

[24] AUTRUSSEAU, F., GUEDON, J. P.: *Image Watermarking for Copyright Protection and Data Hiding via the Mojette Transform*, Proceedings of SPIE, VOL. 4675, 2002, pp. 378 – 386.

[25] TURÁN, J., OVSENIK, L., BENCA, M., TURÁN, J. JR: *Implementation of CT and IHT Processors for invariant Object Recognition System*, Radioengineering, Vol. 13, No 4 2004. dec. page 65-71

[26] AUTRUSSEAU F.: *Modélisation psychovisuelle pour le tatouage des images*, Ph. D. Dissertation, Nantes, France, 2002.

[27] SZOBOSZLAI P.: *Digital Watermarking with Mojette and Wawelet transforms*, Diploma work, University of Miskolc, Department of Automation, 2006, pp. 58.

[28] GUÉDON J-P., NORMAND N.**:** *The Mojette Transform: The First Ten Years,* Proceedings of DGCI 2005, LNCS 3429, 2005, pp. 79-91.

[29] GUÉDON J-P., Normand N.**:** Spline Mojette Transform Application in tomography and communication, In EUSIPCO, sep. 2002.

[30] PARREIN B., NORMAND N., GUÉDON J-P.**:** *Multimedia Forward Error Correcting Codes For Wireless Lan,* Annals of Telecommunications (3-4) 448-463 March-April, 2003.

[31] Xilinx, *ML310 User Guide*, pp73, http:\\xilinx.com

[32] BOBDA C., HUEBNER M., NIYONKURU A, BLODGET B., MAJER M., AHMADINIA A.,: *Designing Partial and Dynamically Reconfigurable Applications on Xilinx Virtex-II FPGAs using Handel-C,* "http://www.celoxica.com/cup/registered_users/community/default.asp", pp. 16.

[33] AMILCAR DO CARMO L., HEITHECKER S., ROLF E., *FlexFilm An Image Processor for Digital Film Processing,* Dagstuhl Seminar on Dynamically Reconfigurable Architectures, 2-7.04.2006. http://www.dagstuhl.de/06141/Materials/

[34] HEITHECKER S., ROLF E., *FlexFilm: A Quality of Service Capable Scheduling SDRAM Controller***,** Dagstuhl Seminar on Dynamically Reconfigurable Architectures, 2-7.04.2006. http://www.dagstuhl.de/06141/Materials/

[35] JOHN VON NEUMANN: First Draft of a Report on the EDVAC, Moore School of Electrical Engineering, University of Pennsylvania, 1945

[36] W ASPRAY: John von Neumann and the Origins of Modern Computing, MIT Press, Cambridge, p. 34—48, (1990)

[37] M. S. SCHLANSKER AND B. R. RAU, "EPIC: Explicitly Parallel Instruction Computing," in Computer, vol. 33, no. 2, pp. 37-45, Feb. 2000, doi: 10.1109/2.820037. (2000)

[38] JK OUSTERHOUT: Why Aren't Operating Systems Getting Faster As Fast As Hardware?, USENIX Summer Conference, (1990)

[39] MIKLÓSI ÁDÁM; ABDAI JUDIT; TEMESI ANDREA: Searching where the treasure is: on the emergence of human companion animal partnership (HCAP), ANIMAL COGNITION, DOI WoS PubMed, Folyóiratcikk/Szakcikk (2021)

[40] ABDAI JUDIT ; FERDINANDY, BENCE; LENGYEL, ATTILA; MIKLÓSI, ÁDÁM: Animacy perception in dogs (Canis familiaris) and humans (Homo sapiens): Comparison may be perturbed by inherent differences in looking patterns. JOURNAL OF COMPARATIVE PSYCHOLOGY (2020)

[41] KOVÁCS SZILVESZTER: Extending the Fuzzy Rule Interpolation "FIVE" by Fuzzy Observation", Computational Intelligence, Theory and Applications: 9th International Conference on Dortmund Fuzzy Days. 802 p., Dortmund, Germany, 2006.09.18-2006.09.20. Berlin; Heidelberg: Springer-Verlag, ISBN:978-3-540-34780-4, pp. 485-497, 2006.

[42] KORONDI PÉTER; KORCSOK, BEÁTA; KOVÁCS SZILVESZTER: Niitsuma Mihoko: Etho-robotics: What kind of behaviour can we learn from the animals? IFAC PAPERSONLINE 48: 19 pp. 244-255. , 12 p. (2015)

[43] KORONDI PÉTER; BJØRN SOLVANG; BARANYI PÉTER: Cognitive Robotics and Telemanipulation, In: anon (szerk.) 15th International Conference on Electrical Drives and Power Electronics (EDPE 2009), Dubrovnik, Horvátország : KoREMA Croatian Society for Communications, Computing, Electronics, Measurement and Control, (2009) Paper: TPL-001, 8 p (2009).

[44] D. VINCZE, Sz. KOVÁCS: Performance Optimization of the Fuzzy Rule Interpolation Method 'FIVE', Journal of Advanced Computational Intelligence and Intelligent Informatics (JACIII), Vol.15 No.3, Special issue on Fuzzy Rule Interpolation, 2011, Fuji Technology Press, Tokyo, Japan, pp. 313-320.

[45] GM AMDAHL: Validity of the Single Processor Approach to Achieving Large-Scale Computing Capabilities. In AFIPS Conference Proceedings, Vol. 30. DOI: http://dx.doi.org/10.1145/1465482.1465560, pp. 483-485. 1967

[46] A. N. SLOSS, D. SYMESS, C. WRIGHT: ARM System Developer's Guide Designing and Optimizing System Software, Elsevier, Morgan Kauffmann Publishers, ISBN: 1-55860-874-5, 2004, pp. 702

[47] A. Olofsson, T. Nordström and Z. Ul-Abdin, "Kickstarting high-performance energy-efficient manycore architectures with Epiphany" 2014 48th Asilomar Conference on Signals, Systems and Computers, Pacific Grove, CA, ISBN: 978-1-4799-8297-4, 2014, pp. 1719-1726.

[48] C. POLLOCK, L. K. BARRETT, P. G. DEL CORRO, A. STANGE, T. G. BIFANO, AND D. J. BISHOP, "PWM as a Low-Cost Method for the Analog Control of MEMS Devices," *Journal of Microelectromechanical Systems*, Vol. 28, No. 2, pp. 245-253, April 2019.

[49] A. POULOSE, O. S. EYOBU AND D. S. HAN, "An Indoor Position-Estimation Algorithm Using Smartphone IMU Sensor Data," *IEEE Access,* Vol. 7, pp. 11165-11177, Jan. 2019.

[50] M. I. M. ISMAIL, R. A. DZIYAUDDIN, N. A. A. SALLEH, F. MUHAMMAD-SUKKI, N. A. B., M. A. M. IZHAR AND L. A. LATIFF, "A Review of Vibration Detection Methods Using Accelerometer Sensors for Water Pipeline Leakage, " *IEEE Access*, Vol. 7, pp. 51965-51981, Jan. 2019.

[51] M. I. M. ISMAIL, R. A. DZIYAUDDIN, N. A. M. SALLEH, R. AHMAD, M. H. AZMI AND H. M. KAIDI, "Analysis and Procedures for Water Pipeline Leakage Using Three-Axis Accelerometer Sensors: ADXL335 and MMA73," *IEEE Access,* Vol. 6, pp. 71249-71261, Oct. 2018.

[52] C. CAI, X. MA, M. HU, Y. YANG, Z. LI AND J. LIU, "SAP A Novel Stationary Peers Assisted Indoor Positioning System," *IEEE Access,* Vol. 6, pp. 76475-76489, Nov. 2018.

[53] C. DONG, Y. YE, X. LIU, Y. YANG AND W. GUO "The Sensitivity Design of Piezoresistive Acceleration Sensor in Industrial IoT," *IEEE Access,* Vol. 7, pp. 16952-16963, Jan. 2019.

[54] L. B. PUPO, "Characterization of Errors and Noises in MEMS Inertial Sensors Using Allan Variance Method," Master of Science Dissertation, Dept. de Teoria del Senyal i Comunicacions, Univ. Politechnica de Catalunya Barcelonatech, , Barcelona, 2016.

[55] Y. L. HSU AND J. S. WANG "Random Drift Modeling and Compensation for MEMS Based Gyroscopes and its Application in Handwriting Trajectory," *IEEE Access,* vol. 7., pp. 17551-17560, Jan. 2019.

[56] R. H. ROGNE, T. H. BRYNE, T. I. FOSSEN AND T. A. JOHANSEN, "MEMS-based Inertial Navigation on Dynamically Positioned Ships: Dead Reckoning," in *Proc. 2016 10th IFAC Conference on Control Applications in Marine Systems (CAMS)*, pp. 139-146.

[57] T. YAMAGUCHI, K. ISHIHARA AND Y. SHINODA, "Field-Programmable Gate Array-based Multichannel Measurement System for Interrogating Fiber Bragg Grating Sensors," *IEEE Sensors Journal*, pp. 6163 - 6172, Vol. 19, Issue. 15, Apr. 2019.

[58] X. LIU, S. WANG, X. GUO, W. YANG AND G. XU, "A Method for Gravitational Apparent Acceleration Identification and Accelerometer Bias Estimation," *IEEE Access,* vol. 7, pp. 38115- 38122, Mar. 2019.

[59] W. DING AND Z. LI, "Research on adaptive modulus maxima selection of wavelet modulus maxima denoising," *Journal of Engineering*, Vol. 2019 Issue. 13, pp. 175-180, Jan. 2019

[60] T. YIN, X. CHENG, F. XIN, Q. WU, F. WANG AND H. YANG, "A Smart Sensory Platform Based on Field Programmable Analog Array," in *Proc. 2015 IEEE Sensors,* pp. 1-4, Jan. 2016

[61] M. TURQUETI, J. SANIIE AND E. ORUKLU, "MEMS Acoustic Array Embedded in an FPGA Based Data Acquisition and Signal Processing System," in *Proc. 2010 53rd IEEE International Midwest Symposium on Circuits and Systems*, pp. 1161-1164.

[62] V. V. AVRUTOV, A. N. SAPEGIN, Z. S. STEFANISHIN AND V. V. TSISARZH, "Calibration of An Inertial Measurement Unit," *International Applied Mechanics*, Vol. 53, No. 2, pp. 228-236, Mar., 2017

[63] L. WANG, F. WANG, "Intelligent Calibration Method of low cost MEMS Inertial Measurement Unit for an FPGA-based Navigation System," *International Journal of Intelligent Engineering and Systems*, Vol.4, No.2, pp. 32-41, 2011

[64] A. KALUPAHANA, N. HEMADASA, N. WIJERATHNE, A. RANASINGHE AND A. PASQUAL, "FPAA and FPGA Based Universal Sensor Node Design," in *Proc. 2017 Eleventh International Conference on Sensing Technology (ICST)*, pp. 1-6, (2011)

11 A Tézisfüzet témájában megjelent közlemények

[S1] VÁSÁRHELYI JÓZSEF; SZABÓ CSABA; INCZE IOAN I; IMECS MÁRIA: FPGA implementation of vector control of tandem converter fed induction machine In: BMF (szerk.) 6th International Symposium of Hungarian Researchers on Computational Intelligence: proceedings : [Magyar Kutatók 6. Nemzetközi Szimpóziuma] Konferencia helye, ideje: Budapest, Magyarország 2005.11.18. - 2005.11.19. Budapest: Budapesti Műszaki Főiskola, Magyar Fuzzy Társaság, pp 215-229 (2005)

[S2] VÁSÁRHELYI JÓZSEF; IMECS MÁRIA; SZABÓ CSABA; INCZE IOAN I: Improved starting of the induction motor fed by a run-time reconfigurable frequency converter, In: IEEE (szerk.), INES 2006 10th International Conference on Intelligent Engineering Systems, Konferencia helye, ideje: London, Egyesült Királyság / Anglia 2006.06.26. - 2006.06.28. (IEEE), New York (NY): IEEE Press, pp 74-79 (2006)

[S3] VÁSÁRHELYI J.; IMECS MÁRIA; SZABÓ CSABA; INCZE JOAN JOV: *Run-Time Reconfiguration of Tandem Inverter for Induction Motor Drives,* In:IEEE (szerk.) Power Electronics and Motion Control Conference, 2006., EPE-PEMC 06. 12th International Conference, Konferencia helye, ideje: Portoroz, Szlovénia 2006.08.30. - 2006.09.01. New York (NY): IEEE Press, Paper t5-607, 2006, pp 408-414.

[S4] VÁSÁRHELYI J., SERFŐZŐ P.: *Analysis of Mojette Transform Implementation on Reconfigurable Hardware*, In: P. M. Athanas; J. Becker; G. Brebner; J. Teich (szerk.), Dynamically Reconfigurable Architectures: Seminar 06141, Konferencia helye, ideje: Schloss Dagstuhl, Németország 2006.04.02. - 2006.04.07., Schloss Dagstuhl, 2006, pp 1-6.

[S5] SZOBOSZLAI P.; VÁSÁRHELYI J; TURÁN J.; SERFŐZŐ P.: *MTTool Software Tool and Low Complexity Hardware for Mojette Transformation* STUDIES IN COMPUTATIONAL INTELLIGENCE (1860-949X 1860-9503): 313 pp 15-26 (2010)

[S6] VÁSÁRHELYI J.; TURÁN J.; SZOBOSZLAI P.: „Inverse Mojette Transform Implementation and the MTTool" CARPATHIAN JOURNAL OF ELECTRONIC AND COMPUTER ENGINEERING (1844-9689): 4 1 pp 127-132 (2011).

[S7] TURÁN J.; SZOBOSZLAI P.; VÁSÁRHELYI J.: „Mojette Transform Software – Hardware Implementations and its Applications, INFOCOMMUNICATIONS JOURNAL (2061-2079 2061-2125): 3/1 pp 39-47 (2011).

[S8] RIMA B.; VÁSÁRHELYI J.; VÉGH J.; TURÁN J.: „Mojette Transform implemented in Labview", In: Petráš Ivo; Podlubny Igor; Kačur Ján; Farana Radim (szerk.): 15th International Carpathian Control Conference, ICCC 2014, Velké Karlovice, Csehország 2014.05.28. - 2014.05.30., New York (NY): IEEE, pp 481-484 (2014)

[S9] J. VÉGH, P. MOLNÁR, J. VÁSÁRHELYI: A figure of merit for describing the performance of scaling of parallelization, arXiv:1606.02686v3 [cs.PF] , 2016, pp. 14.

[S10] J. VÉGH, J. VÁSÁRHELYI, D. DRÓTOS, "Can parallelization save the (computing) world?," *2018 19th International Carpathian Control Conference (ICCC)*, Szilvasvarad, doi: 10.1109/CarpathianCC.2018.8399587. (2018)

[S11] JÁNOS VÉGH; JÓZSEF VÁSÁRHELYI: Can Broken Multicore Hardware be Mended, GLOBAL JOURNAL OF RESEARCHES IN ENGINEERING (0975-5861 2249-4596): 18 1-A Paper 1778. (2018), ISSN: 2249-4596, pp. 15-21 (2018)

[S12] DRÓTOS DÁNIEL; VÁSÁRHELYI JÓZSEF: *Interrupt driven parallel processing,* In: Kot A.; Nawrocka A. (szerk.), 2019 20th International Carpathian Control Conference : ICCC,

Konferencia helye, ideje: Wieliczka, Lengyelország, Krakow, Lengyelország 2019.05.26. - 2019.05.29., New York (NY): IEEE, pp 70-73 Paper 8765909. (2019)

[S13] DRÓTOS DÁNIEL; VÁSÁRHELYI JÓZSEF; Végh János: Szoft-processzorok alkalmazása FPGA-n multiprocesszoros rendszerek vizsgálatához, Using Soft-processors on FPGA for Examininn Multiprocessor Systems. In: Bíró-Károly Ágoston; Sebestyén-Pál György; Szabó Lóránd (szerk.) ENELKO 2018 XIX. Nemzetközi Energetika-Elektrotechnika Konferencia SzámOkt 2018 XXVIII. Nemzetközi Számítástechnika és Oktatás Konferencia, Konferencia helye, ideje:: Tusnádfürdő, Románia 2018.10.11. - 2018.10.14., Erdélyi Magyar Műszaki Tudományos Társaság (EMT), pp 163-168 (2018)

[S14] BARTÓK R.; VÁSÁRHELYI J.: „Examining Cache Handling of the FIVE method on Multicore Systems" in: Szakál Anikó, 2019 IEEE 17th World Symposium on Applied Machine Intelligence and Informatics (Sami 2019) Herlany, Szlovákia 2019.01.24. - 2019.01.26. (Kassai Műszaki Egyetem, Óbudai Egyetem), Herlany: IEEE, Paper 24_sami2019, pp 141-146. (2019).

[S15] BARTÓK R.; VÁSÁRHELYI J.: „Fuzzy Rule Interpolation Based Object Tracking and Navigation for Social Robot" LECTURE NOTES IN MECHANICAL ENGINEERING (2195-4356 2195-4364), 2018 pp 370-375 (2018).

[S16] BARTÓK R.; VÁSÁRHELYI J.: „A Fuzzy Rule Interpolation Base Algorithm Implementation on Different Platforms", In: Ivo Petras; Igor Podlubny; Jan Kacur; Vásárhelyi József (eds): Proceedings of the 16th International Carpathian Control Conference, Szilvásvárad, Magyarország 2015.05.27. - 2015.05.30., Miskolc: IEEE IAS/IES/PELS, pp 37-40 (2015).

[S17] BARTÓK ROLND; VÁSÁRHELYI JÓZSEF: Two Methods for Autonomous Robot Obstacle Sensing and Application Programming Interface for Fuzzy Rule Interpolation, In: Dan Popescu; Dorin Şendrescu; Monica Roman; Elvira Popescu; Lucian Bărbulescu (szerk.), 2017 18th International Carpathian Control Conference (ICCC), Konferencia helye, ideje: Sinaia, Románia 2017.05.28. - 2017.05.31. (IEEE Computational Intelligence Society), Craiova: IEEE, Paper 7970376. (2017), ISBN: 9781509058259, pp 87-92 (2017)

[S18] AHMED BOUZID; VÁSÁRHELYI JÓZSEF: Implementation of Dead Reckoning Solution on Zynq Target, In: Dan Popescu; Dorin Şendrescu; Monica Roman; Elvira Popescu; Lucian Bărbulescu (szerk.), 2017 18th International Carpathian Control Conference (ICCC), Konferencia helye, ideje: Sinaia, Románia 2017.05.28. - 2017.05.31. (IEEE Computational Intelligence Society), Craiova: IEEE, Paper 7970376. (2017), ISBN: 9781509058259, pp 134-139 (2017)

[S19] BOUZID A., VÁSÁRHELYI J.: Implementation and Experimentation of an Embedded Data Acquisition/Preprocessing System based on a Hybrid Reconfigurable Hardware Accelerator for Inertial Measurements, IEEE TRANSACTIONS ON INDUSTRY APPLICATIONS (0093-9994): 56 2 pp 2012-2019 Paper 2019-IACC-0538. (2020)

[S20] BOUZID A., VÁSÁRHELYI J.: High Resolution Large Scale ADC. Case study of an N bit per Volt ADC Implemented using FPAA and FPGA Applied for Precision Altimetery, In: Petras I.; Kacur J. (szerk.), Proceedings of the 2020 21st International Carpathian Control Conference (ICCC), Konferencia helye, ideje: Magas-Tátra, Szlovákia 2020.10.27. - 2020.10.29. (Slovak Society for Applied Cybernetics and Informatics (SSAKI), Piscataway (NJ): IEEE, pp 1-6 Paper 9257269. (2020)

[S21] REDA AHMAD; BOUZID AHMED; VÁSÁRHELYI JÓZSEF: Model Predictive Control for Automated Vehicle Steering, ACTA POLYTECHNICA HUNGARICA (1785-8860): 17 7 pp 163-182 (2020).

[S22] BARTÓK ROLAND; L KISS MÁRTON; VÁSÁRHELYI JÓZSEF, KOVÁCS SZILVESZTER; AHMED BOUZID: Embedded behavioral model implementation, In: Ivo Petráš ; Igor Podlubny;

Ján Kačur (szerk.), Proceedings of the 2016 17th International Carpathian Control Conference (ICCC), Konferencia helye, ideje: Tatranska Lomnica, Szlovákia 2016.05.29. - 2016.06.01., s.l.: IEEE, ISBN: 9781467386050, pp 35-40 (2016)