

TDK-dolgozat

Bencs Róbert



MISKOLCI EGYETEM
GÉPÉSZMÉRNÖKI ÉS INFORMATIKAI KAR



TUDOMÁNYOS DIÁKKÖRI DOLGOZAT

**Mérési adatok automatizált feldolgozó- és dokumentum
generáló szoftverrendszere villamos hálózat
diagnosztikai alkalmazáshoz**

Bencs Róbert

I. éves mérnök informatikus MSc hallgató

Konzulens:

Váradiné Dr. Szarka Angéla

egyetemi docens

Elektrotechnikai-Elektronikai Tanszék

Miskolc, 2012.

Tartalomjegyzék

1.	Bevezetés.....	3
2.	Elméleti áttekintés	4
2.1.	A villamos energia minősége.....	4
2.1.1.	Hálózati frekvencia.....	5
2.1.2.	A tápfeszültség nagysága	5
2.1.3.	A tápfeszültség változásai	5
2.1.4.	Gyors feszültségváltozások	5
2.1.5.	Tápfeszültség-letörések.....	6
2.1.6.	A tápfeszültség kimaradásai	7
2.1.7.	Tápfeszültség-aszimmetria.....	7
2.1.8.	Felharmonikus feszültségek	7
2.2.	Számítógépes mérőrendszerek	8
3.	A fejlesztés előzményei	11
3.1.	A tanszéken korábban használt mérőrendszer	11
3.2.	Az új mérőrendszer.....	12
3.2.1.	A mérőberendezés	13
3.2.2.	A mérőprogram.....	16
3.2.3.	Adatok pontossága és mennyisége	19
3.3.	A fejlesztőkörnyezetről	19
3.3.1.	Miért esett a választás erre a fejlesztőkörnyezetre?	19
3.3.2.	Visual Studio.....	20
3.3.3.	Measurement Studio.....	21
3.3.4.	Verziókövető rendszer.....	21
3.4.	Kitűzött célok.....	22
4.	Az elkészült program bemutatása	23
4.1.	A program használatba vétele	24
4.2.	A felhasználói felület koncepciója	24
4.3.	Adatok betöltése.....	24
4.4.	Áttekintés ablak.....	25
4.5.	Egyszerű nézetek.....	26
4.5.1.	Közös jellemzők.....	26
4.5.2.	Összes adat.....	28
4.5.3.	Napi adatok.....	28
4.5.4.	Tízperces adatok.....	28
4.6.	Fogyasztás.....	29

4.7.	Tranziensek ablak	30
4.8.	Statisztika.....	31
4.9.	Flicker	32
4.10.	Torzítási teljesítmények.....	32
4.11.	Teljesítmény harmonikusok.....	33
4.12.	Jegyzőkönyv készítése.....	33
4.13.	További szolgáltatások.....	34
4.13.1.	Grafikonok testreszabása.....	34
4.13.2.	Grafikonok exportálása.....	35
4.13.3.	Konfigurációs állomány.....	35
5.	A fejlesztés során alkalmazott megoldások, a megvalósítás részletei.....	36
5.1.	Miért tart sokáig az adatok betöltése első alkalommal?	36
5.1.1.	A paraméterfájl feldolgozása	36
5.1.2.	Az adatfájlok feldolgozása	36
5.1.3.	A keletkezett új adatok mentése	36
5.1.4.	Tízperiódusokhoz tartozó időpontok kiszámítása.....	37
5.1.5.	A feldolgozási idők az adathordozók csatlakozója függvényében.....	38
5.2.	Nagyméretű adatok kezelése.....	39
5.2.1.	Logikai nézetek.....	40
5.2.2.	Aggregált értékek számítása	40
5.3.	Adatok beolvasásának optimalizálása.....	42
5.3.1.	Adatok rendszerezése.....	42
5.3.2.	Beolvasást végző függvények felépítése	42
5.3.3.	Beolvasási sor használata	43
5.4.	A jegyzőkönyvkészítés részletei.....	44
5.4.1.	Varázsló megvalósítása	45
5.4.2.	A dokumentum generálása	46
5.5.	További érdekességek.....	47
5.5.1.	Teljesítmény újraszámító program.....	47
5.5.2.	Naplózás.....	49
6.	A szoftver ipari körülmények között történő alkalmazása.....	50
6.1.	Salgótarján - Szent Lázár Megyei Kórház	50
6.2.	Debreceni Egyetem Orvos- és Egészségtudományi Centrum.....	51
7.	Összegzés.....	54
8.	Summary	55
9.	Felhasznált irodalom.....	56
10.	Mellékletek.....	57

1. BEVEZETÉS

Napjainkban életünk minden területén nélkülözhetlenné vált a villamos energia felhasználása. Jelentősége mindenki számára akkor válik tudatossá, ha ez a – minden nap magától értetődő módon – rendelkezésre álló energia valamilyen oknál fogva rövidebb, hosszabb időre megszűnik, nem is beszélve az ipari igényekről. Ezért fontossága miatt elengedhetlenné vált, hogy megfelelő mérések segítségével vizsgálni tudjuk a villamos energia minőségét, annak különböző paramétereit, valamint a felhasználás hatékonyságát.

A *Miskolci Egyetem Elektrotechnikai-Elektronikai Tanszéke* az elmúlt évek során is végzett ilyen méréseket, kutatásokat, és a felhalmozódott tapasztalatok alapján kifejlesztésre került egy új, korszerű mérőrendszer, mely segítségével minden eddiginél pontosabban tudjuk vizsgálni a villamos energiát.

Az én feladatom az új rendszer által lementett adatok megjelenítésére, kiértékelésére alkalmas, az analízist minél jobban segítő, felhasználóbarát program kifejlesztése volt. Fontos szempont volt továbbá a program bővíthetősége, hiszen a fejlesztés, vagy akár a későbbiek során új ötletek, megoldások szülehetnek a felmerülő problémákra, amelyeket bele kell építeni az alkalmazásunkba. Megvizsgáltam, hogy a kiértékelés mely részfolyamatai automatizálhatók a számítógép segítségével, hogy ezáltal az analízis minél egyszerűbbé, gyorsabbá váljon. Olyan módszereket dolgoztam ki, amelyek segítségével nagyon hosszú távú mérések kiértékelése se okozzon gondot.

A téma ismeretéhez először tanulmányoznom kellett a szakirodalmat, hogy milyen minőségi követelményeknek kell megfelelnie a villamos energiának, majd meg kellett ismernem a mérőrendszer már kész komponenseit, a fejlesztőrendszert, mielőtt a konkrét fejlesztést elkezdtem volna.

A szoftver megírásához a *C#* nyelvet és a *Visual Studio* fejlesztőkörnyezetet választottam, melyhez a *National Instruments* cég *Measurement Studio* nevű alkalmazását feltelepítve nagyszerű eszközöket kaptam a felmerülő problémák megoldásához.

Összegezve, dolgozatomban áttekintést nyújtok az elkészült mérőrendszerről, ismertetem az adatfeldolgozás menetét. Részletesen bemutatom az általam elkészített feldolgozóprogram működését, annak minden funkciójával együtt, különös tekintettel arra, hogy milyen új szolgáltatások segítik az elemzést.

2. ELMÉLETI ÁTTEKINTÉS

2.1. A villamos energia minősége

Ebben a fejezetben a villamos energia minőségének vizsgálatába nyerhetünk egy kis elméleti betekintést.

A villamos energia napjainkban talán a legfontosabb alapanyag. Ez az energia egy különös árucikk, mert nem tárolható gazdaságosan nagy mennyiségben, tehát minden pillanatban annyi villamos energiát kell termelni, amennyit a fogyasztók igényelnek. Természetesen a valóságban a villamos energia lényegesen különbözik bármilyen egyéb terméktől: a felhasználás helyétől távol állítják elő, sok hasonló generátorral együtt táplálják a hálózatba, és több transzformátoron, sok kilométernyi szabadvezetéken keresztül érkezik meg a felhasználás helyére. A felhasználási helyen szolgáltatott energia minőségének biztosítása nem könnyű feladat, nincs lehetőség a szabványnak nem megfelelő villamos energia kivonására a táphálózatból, vagy arra, hogy a fogyasztó a felhasználását megtagadja.

A fogyasztók szemszögéből a probléma még sokkal nehezebb. Rendelkezésre áll néhány korlátozott mértékű statisztika a szolgáltatott energia minőségéről, de az energiaszolgáltató (és az ipari szakmai felügyelet) által elfogadhatónak értékelt minőségi szint nagyon különbözhet a fogyasztó elvárásaitól.

A legnyilvánvalóbb energiaszolgáltatási hibák a teljes feszültség kimaradások (amelyek néhány másodperctől kezdve akár több órán át tarthatnak), illetve a feszültségletörések, amikor egy rövid időre lecsökken a feszültség. A hosszú feszültség kimaradások nyilvánvalóan gondot jelentenek valamennyi fogyasztó számára, de sok technológiai művelet nagyon érzékeny még az igen rövid kimaradásokra is.

Feltehetjük a kérdést: mit értünk tehát a villamos energia minőségén? A tökéletes energiaellátás olyan lenne, amely folyamatosan rendelkezésre áll, a feszültség és frekvencia mindig a tűrési sávon belül van, a hullámalak tökéletesen szinuszos. A felhasználói alkalmazás jellegétől, a berendezésektől és a felhasználó igényétől függ, hogy a tökéleteshez képest mekkora eltérés fogadható el.

A közcélú kiefeszültségű hálózatok feszültségjellemzőivel az MSZ EN 50160 szabvány [12] foglalkozik. A továbbiakban áttekintem a szabvány által előírt feltételeket.

2.1.1. Hálózati frekvencia

A tápfeszültség névleges frekvenciája 50 Hz legyen. Normál üzemi körülmények között az alapharmonikus frekvencia átlagértéke 10 másodpercen keresztül mérve a következő tartományban legyen: (az együttműködő hálózathoz szinkron csatlakoztatású hálózat esetében):

50 Hz $\pm$ 1%	(azaz 49,5 – 50,5 Hz)	az év 99,5%-ában,
50 Hz + 4%/-6%	(azaz 47 – 52 Hz)	az idő 100%-ában.

2.1.2. A tápfeszültség nagysága

A közcélú kisfeszültség, U_n szabványos névleges feszültsége a következő legyen négyvezetékes háromfázisú hálózatokban:

$U_n = 230$ V a fázis és a nullvezető között mérve.

2.1.3. A tápfeszültség változásai

Normál üzemi körülmények között, a feszültségkimaradásokat figyelmen kívül hagyva

- a tápfeszültség 10 perces átlagos effektív értékei 95%-ának bármely egyhetes időszakban az $U_n \pm 10\%$ tartományban kell lennie.
- a tápfeszültség minden 10 perces átlagos effektív értékének $U_n +10\%$ / -15% tartományban kell lennie.

2.1.4. Gyors feszültségváltozások

A gyors feszültségváltozások nagysága

A tápfeszültség gyors feszültségváltozásait elsősorban a fogyasztói villamos berendezések terhelésváltozásai vagy a hálózaton történő kapcsolások okozzák.

Normál üzemi körülmények között egy gyors feszültségváltozás általában nem haladja meg U_n 5%-át, de bizonyos körülmények között a változás, naponta néhányszor, rövid időre U_n 10%-át is elérheti.

Villogásmérték

Villogás vagy flicker definíciója: időben ingadozó fényességű vagy színképi eloszlású fényinger által létrehozott látásérzet-ingadozás hatása.

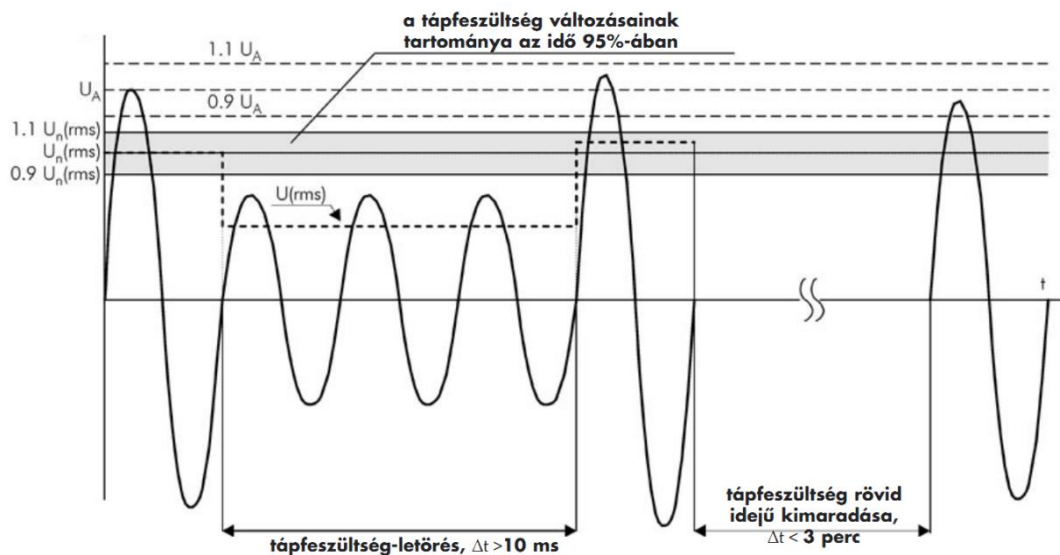
Normál üzemi körülmények között a feszültségingadozás által okozott hosszú idejű villogásmértéknek, bármely egyhetes időszakban, az idő 95%-ában $P_{lt} \leq 1$ értékűnek kell lennie.

MEGJEGYZÉS: A villogással szembeni érzékenység szubjektív és változhat a villogás észlelt okától és a fennállásának időtartamától.

2.1.5. Tápfeszültség-letörések

A tápfeszültség hirtelen csökkenése az U_n megegyezéssel feszültség 90%-a és 1%-a közötti értékre, amit egy rövid időtartam után a feszültség visszaállása követ. A feszültségletörés szokásos ideje 10 ms és 1 perc között van. A feszültségletörés nagysága a megegyezéssel feszültség és a feszültségletörés idején előforduló legkisebb feszültség effektív értékeinek különbsége. Azok a feszültségváltozások, amelyek a tápfeszültség értékét nem csökkentik az U_n megegyezéssel feszültség 90%-a alá, nem tekinthetők letöréseknek.

A feszültségletöréseket általában a fogyasztói villamos berendezésekben vagy a közcélú elosztóhálózatokban előforduló hibák okozzák. Ezek előre nem látható, véletlenszerű események. Az éves gyakoriság változik, nagymértékben függ a táphálózat típusától és a megfigyelési ponttól. Ezen kívül az éves eloszlás is nagyon szabálytalanul változhat.



1. ábra: Feszültség-letörés és a feszültség rövid idejű kimaradásának szemléltetése

U_n – a táphálózat névleges feszültsége (effektív érték),

U_A – a tápfeszültség amplitúdója,

U_{eff} – a tápfeszültség tényleges effektív értéke

2.1.6. A tápfeszültség kimaradásai

Az az eset, amikor a csatlakozási pontokon a feszültség kisebb, mint az U_n megegyezéssel feszültség 1%-a. A táplálás kimaradása a következőképpen osztályozható:

- előre tervezett, az elosztóhálózaton beütemezett munkák végrehajtása céljából, amelyről a fogyasztókat előzetesen tájékoztatták, vagy

- véletlenszerű, amelyet külső eseményekkel, villamos szerkezetek hibájával vagy zavarásával kapcsolatos tartós, vagy tranzien্স hibák okoznak, amely lehet:

- hosszú idejű kimaradás (három percnél hosszabb), amelyet tartós hiba okoz;
- rövid idejű kimaradás (három percen belül), amelyet tranzien্স hiba okoz.

2.1.7. Tápfeszültség-aszimmetria

Feszültségaszimmetria definíciója: háromfázisú hálózat olyan állapota, amelynél a fázisfeszültségek effektív értékei vagy az egymást követő fázisok közötti szögek nem egyenlők egymással.

Normál üzemi körülmények között a tápfeszültség negatív fázissorrendű összetevője 10 perces átlag effektív értékeinek 95%-a, bármely egyhetes időszakban a pozitív fázissorrendű összetevő 0-tól 2%-os tartományában legyen. Egyes területeken, ahol egyes fogyasztói villamos berendezésekhez egy vagy két fázis csatlakozik, a háromfázisú csatlakozópontoknál 3%-os aszimmetria is előfordulhat.

2.1.8. Felharmonikus feszültségek

A tápfeszültség alapharmonikus frekvenciájának egész számú többszörösével megegyező frekvenciájú szinuszos feszültség. A felharmonikus feszültségeket a következőképpen lehet számítani:

- egyenként, az U_1 alapharmonikus feszültséghez viszonyítva az U_h relatív amplitúdójukat, ahol h a harmonikus rendszám (2. ábra);

- átlagolva, például a THD (Total Harmonic Distortion) teljes harmonikus torzítási tényező segítségével a következő kifejezéssel számolva:

$$THD_U = \frac{\sqrt{\sum_{h=2}^{40} (U_h)^2}}{U_1}$$

Normál üzemi körülmények között minden egyes felharmonikus feszültség tízperces átlag effektív értékeinek 95%-a, bármely egyhetes időszakban nem lehet nagyobb az alábbi táblázatban megadott értékeknél. Az egyes felharmonikusok esetében a rezonancia nagyobb feszültségeket is okozhat.

Páratlan felharmonikusok				Páros felharmonikusok	
3-nak nem többszöröse		3-nak többszöröse			
Rendszám (h)	Relatív feszültség	Rendszám (h)	Relatív feszültség	Rendszám (h)	Relatív feszültség
5	6,0%	3	5,0%	2	2,0%
7	5,0%	9	1,5%	4	1,0%
11	3,5%	15	0,5%	6 – 24	0,5%
13	3,0%	21	0,5%		
17	2,0%				
19	1,5%				
23	1,5%				
25					

2. ábra: Harmonikus feszültségek határai

Az előzőeken túlmenően a tápfeszültség THD-értéke (beleértve az összes felharmonikust a 40-es rendszámig) nem lehet 8%-nál nagyobb.

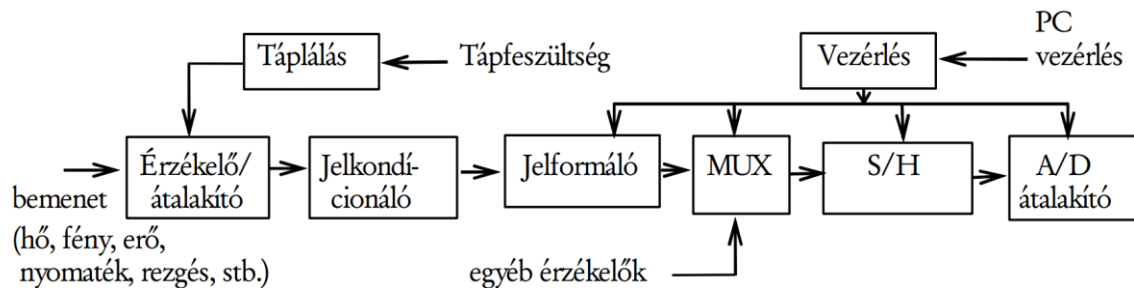
2.2. Számítógépes mérőrendszerek [3]

A számítógép a méréstechnikában csakúgy, mint a legtöbb számítástechnikai területen a rendszer olyan eszköze, amelynek a feladata az információ összegyűjtése, feldolgozása, összegzése és rendszerezése. A számítógép alkalmazása a méréstechnika területén is sokszorosára növelte a hatékonyságot. Feladata a mérőrendszerben röviden az alábbiakban foglalható össze:

- adatkezelés: gyűjtés, tömörítés, egyszerűsítés, kiértékelés, tárolás,
- műszerek, egyéb perifériák, folyamatok vezérlése,
- mérési folyamat fejlesztés,
- dokumentálás.

A számítógéppel vezérelt mérőrendszer alapfunkciója, hogy egy fizikai/kémiai mennyiséget fogadjon, azt a számítógép által használható formára (bináris) alakítsa, szükség szerint archiválja, esetleg továbbítsa. Az alábbi ábra egy mérőrendszer funkcionális

felépítését mutatja be. Minden egységnek jól meghatározható feladata van a rendszerben, jöllehet a különböző alkalmazásoknál egyik vagy másik egység hiányozhat.



3. ábra: Számítógéppel vezérelt mérőrendszer funkcionális felépítése

Érzékelő/átalakító:

Fizikai/kémiai mennyiséget érzékel és azt valamilyen ismert törvényszerűség szerint átalakítja az adatgyűjtés számára alkalmas, általában feszültségjel formára. Egyes érzékelőknek külső energiaforrásra is szükségük lehet a működésükhöz. Bizonyos érzékelők az érzékelésen túl a mérőkör leválasztását is megoldják. Ez különösen a villamos mennyiségek mérésénél lehet lényeges, mert ha például egy esetleges meghibásodás miatt túl nagy mérendő mennyiség jelenne meg az érzékelő bemenetén, akkor védelem nélkül a számítógép bemenetére olyan szintű feszültség juthat, amely a berendezés meghibásodásához vezetne. Ezért a mérőkört mindig megfelelő módon célszerű leválasztani a mérendő körrel.

Jelkondicionáló:

Az átalakító kimeneti jele a számítógép analóg bementére kapcsolható olyan feszültség jel, melyet sok esetben a megfelelő pontosságú, minőségű átalakítás elérése érdekében „kezelní”, kondicionálni kell. A jelkondicionálás legtipikusabb példája a szűrés és erősítés. A mért jelek általában zajosak és mindig tartalmazznak olyan egyéb, nem hasznos összetevőt, amit a feldolgozás előtt szükséges kiszűrni. Gyakran fordul elő, hogy a jel nagysága nem elegendő a digitalizáláshoz. Ha például a jel a maximális bemeneti tartományhoz képest túl kicsi, akkor digitalizálásnál az A/D átalakító csak az alsó néhány bitet használja fel, ami az átalakítás pontosságát nagymértékben csökkenti. A leggyakrabban alkalmazott analóg jelkondicionálók: erősítő, osztó, zajsűrő, antialiasing szűrő.

Jelformáló:

Vannak olyan esetek, amikor a mért jelen a szükséges átalakításokat érdemes még a „számítógépen kívül” elvégezni. Ez főleg akkor hasznos, ha processzoridőt kívánunk megtakarítani, amivel a rendszer sebessége jelentősen növelhető. Processzoridőt takaríthatunk meg például egy integráló áramkör beiktatásával, vagy jelformáló lehet egy egychipes számítógép, amely segítségével a processzor terhelése jelentősen csökkenthető.

Mintavevő/tartó (Sample and hold - S/H):

A rendszernek ez az egysége végzi az adatgyűjtő berendezés analóg bemeneti csatornájára kapcsolt jel mintavételezését. A mintavételezett analóg feszültségértéket gyakran „várákoztatni” kell amíg az feldolgozásra kerül, ezért szükség van egy olyan áramkörre, amely a jelet a szükséges ideig „tartja”.

A mintavevő tartó áramkörök a funkcióiknak megfelelően két kiemelten fontos paraméterrel rendelkeznek: az egyik a mintavételezési idő, a másik a tartási drift. A mintavételezési idő alatt azt az időt értjük, ami egy minta beolvasás vezérlőjelének kiadásától az áramkör jelkövető állapotának beállásáig eltelik. A tartási drift azt mutatja meg, hogy egységnyi idő alatt mennyit esik a tartó áramkör kimenetén a feszültség.

MUX (Multiplexer):

Ha több bemenő jelet akarunk egyidőben a számítógépbe vinni, akkor vagy több A/D átalakítót kell alkalmazni, vagy a jeleket valamilyen sorrendben, csatornánként kell behívni és átalakítani. Mindkét megoldást alkalmazzák, többségében azonban az A/D átalakítók magas ára miatt olyan sorrend kapcsolókat, ún. multiplexereket alkalmaznak, amelyek több csatornát képesek fogadni, a jeleket sorba rendezni, vezérelni és továbbítani az A/D átalakító bementére.

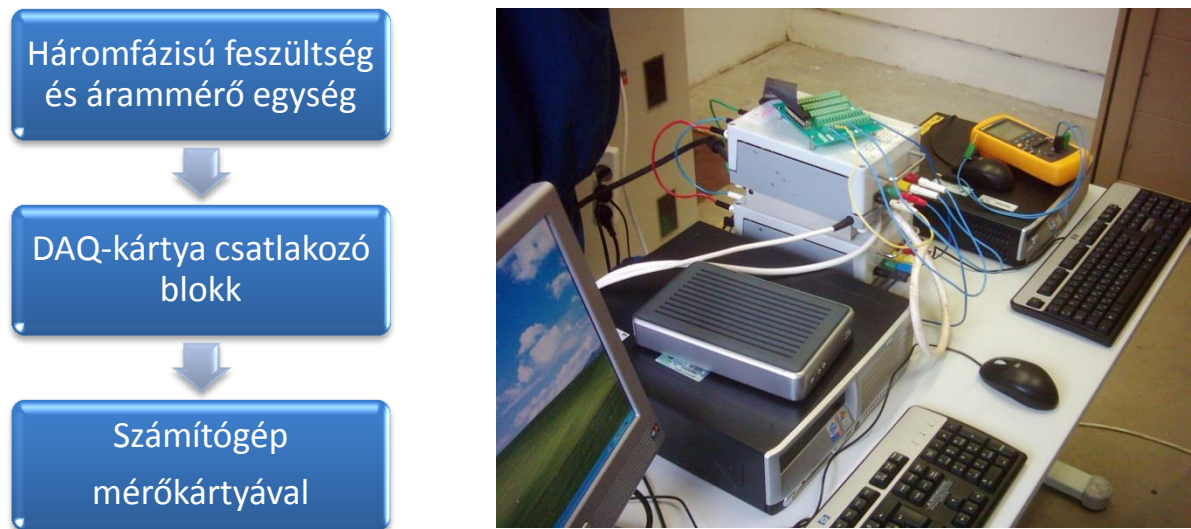
A/D átalakító:

A mintavételezett analóg feszültségértéket digitális (bináris) formára alakítja át. Működési elvük, felbontásuk, sebességük, stb. szerint nagyon sokféle A/D átalakító van forgalomban, ezek között a legelterjedtebb az ún. fokozatos közelítéses (successive approximation) elven működő átalakítók, melyek mind sebességüket, pontosságukat, mind az árukat figyelembe véve gazdaságos megoldást kínálnak a legtöbb mérési feladathoz.

3. A FEJLESZTÉS ELŐZMÉNYEI

3.1. A tanszéken korábban használt mérőrendszer

Az elmúlt időszakban a tanszék saját fejlesztésű számítógépes mérőrendszerei egy számítógépbe helyezhető, 16 bites mérőkártyát, és a hozzá kapcsolt, egyenként három fázis mérésre alkalmas feszültség- és árammérő egységeket integráltak. A rendszer blokkvázlata, illetve egy összeállított mérés látható a 4. ábrán.



4. ábra: Hagyományos mérés összeállításának blokkvázlata és felszerelése

Ezeknél a méréseknél lényeges volt a mérési körülmények és paraméterek pontos dokumentálása, mivel a rendszer nem volt képes a berendezések automatikus felismerésére, a méréshatárok és kalibrációs adatok betöltésére. A dokumentációnak tartalmaznia kellett, hogy melyik mérődobozt alkalmazzuk, milyen méréshatárban használjuk, melyik csatornára milyen jelet és milyen sorrendben kapcsolunk, illetve azt, hogy melyik dobozra milyen primer oldali feszültséget vagy áramot csatlakoztatunk. Ez igen bonyolulttá tette a manuális adminisztrációt, és jelentősen növelte a hibázás lehetőségét.

A mérések során a rendszer csak adatrögzítést végzett, a mért adatok feldolgozása, a vizsgált paraméterek kiszámítása a mérés után történt egy külön alkalmazás segítségével. Ez a program gyakran több napig futott (a mérés hosszúságától, illetve a használt csatornák számától függően), és a mérési adatok kiértékelése csak ezután kezdődhetett el.

3.2. Az új mérőrendszer

A Miskolci Egyetem Elektrotechnikai-Elektronikai Tanszéke az elmúlt évek, évtizedek során nagy tapasztalatra tett szert villamos hálózatok analizálása terén. A technika fejlődésével párhuzamosan mindig újabb és fejlettebb mérőrendszerek kerültek kialakításra.

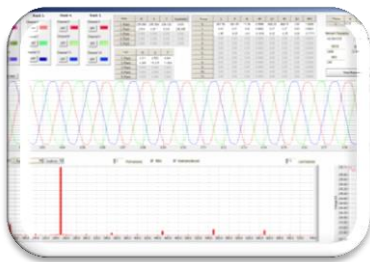
Nemrégiben is kifejlesztésre került egy teljesen új, korszerű mérőrendszer, melynek segítségével minden eddiginél pontosabban tudjuk vizsgálni a villamos energiát. Egy olyan komplex mérőrendszer került kialakításra, melynek képességei, rugalmassága messze túlmutat a konkurensekén. (A hasonló mérőrendszerekről egy nagyon jó kis összefoglalót láthatunk a [16] hivatkozott dolgozat 2. fejezetében.) A munka során a *National Instruments* különböző hardver és szoftvertermékeit használtuk.

Az új mérőrendszert három részre bonthatjuk (5. ábra):

- mérőberendezés (hardver)
- mérőprogram
- feldolgozószoftver



Mérőberendezés



Mérőprogram



Feldolgozószoftver

5. ábra: Az új mérőrendszer részei

Az én feladatom a feldolgozószoftver megtervezése illetve elkészítése volt, de ehhez alaposan tanulmányoznom kellett a rendszer többi részét is, hiszen ezek alapvetően meghatározzák a feldolgozószoftver működését. A rendszer további részei az elmúlt időszakban készültek el a tanszéken. Az alábbiakban tehát a rendszer már meglevő részeinek ismertetése következik, kiemelve azokat a tulajdonságokat, amelyek az én munkám során fontosak voltak.

3.2.1. A mérőberendezés [16]

A tanszéken három műszer került kialakításra, ezek különböző méretűek, illetve a legnagyobb dobozba (6. ábra) beépítésre került egy NI PCIe-6251 típusú multifunkciós mérésadatgyűjtő kártya és egy komplett számítógép is, míg a kisebbek használatához szükség van külön PC-re is.

A rugalmasságot szem előtt tartva a műszerek moduláris felépítésűek: fiókok kerültek kialakításra (7. ábra), melyek a tényleges mérési feladatok igényei szerint cserélhetők. A műszer első felében található a feszültség- illetve árammérő fiókok, melyeket tetszőleges sorrendben és összeállításban helyezhetünk a mérőműszerbe. Ezen modulok végzik a mért jelek átalakítását mérhető tartományú feszültségjellé.

Az 5 cserélhető modul különböző méréshatárokból készült, melyek a következők:

- Feszültségmérő: 5 V, 200 V, 500 V méréshatár
- Árammérő: 1.5 A és 5 A méréshatár

A mérőrendszer egy szabványos méretű műszerházba került beépítésre. Ez biztosítja a blokkszerűen elhelyezkedő fiórendszer rögzítését és lehetőséget nyújt a táp-, biztonsági-, fő- és segédberendezések elhelyezésére.



6. ábra: Mérőműszer integrált számítógéppel

Mivel a DAQ-kártya maximálisan $\pm 5V$ feszültség fogadására alkalmas (analóg bemeneti csatornák), ezért áram- és feszültségváltók alkalmazására volt szükség, melyek a cserélhető fiókokba lettek beépítve.

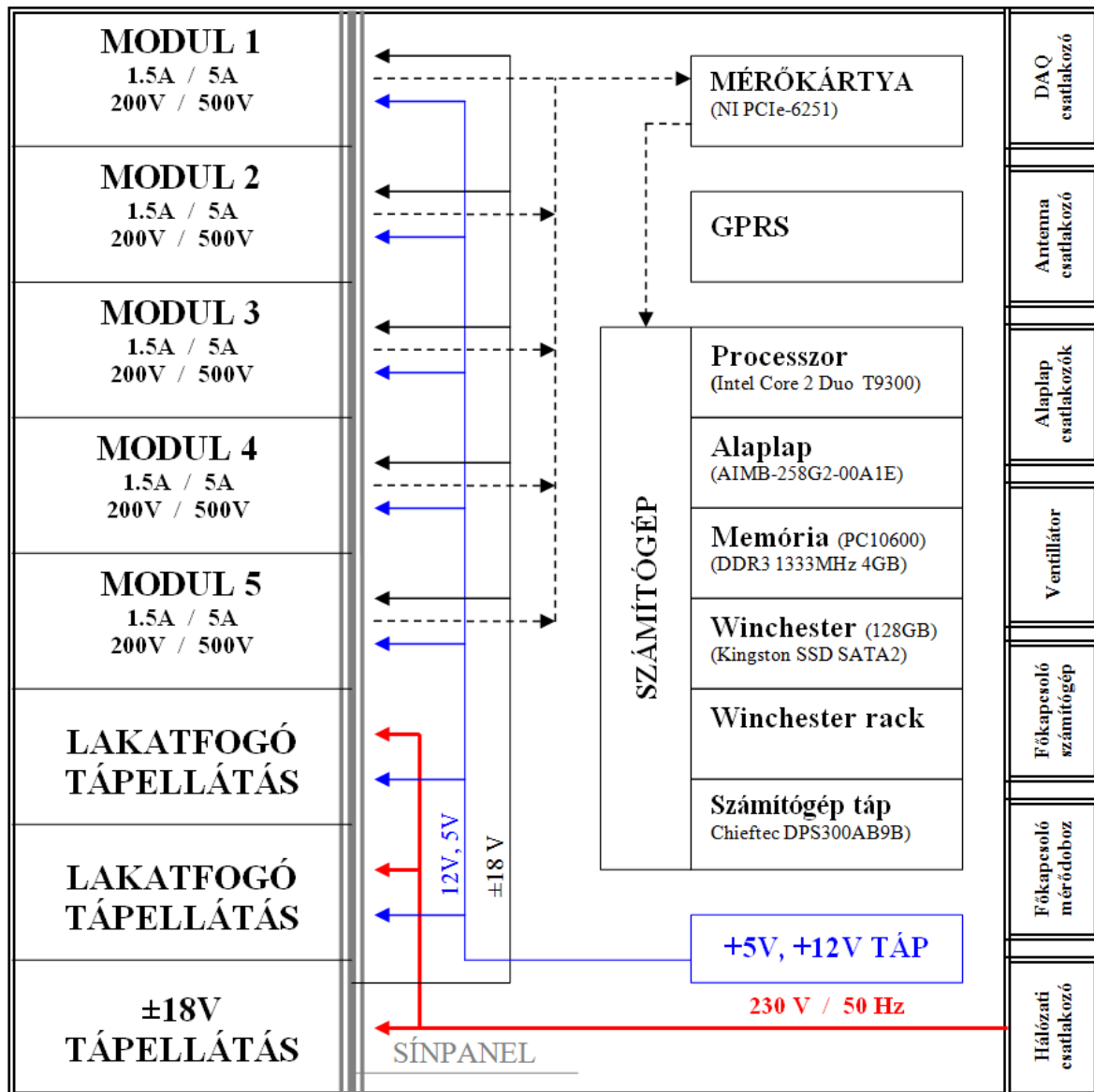
A fiókokban a LEM cég *Hall-effektuson* alapuló érzékelői találhatóak. Az ilyen típusú érzékelők nagy előnye, hogy a mérendő körről galvanikus leválasztást is biztosítanak.



7. ábra: Feszültségmérő fiók

A rendelkezésre álló mérőkártyák maximum 16 csatorna mérésére alkalmasak, és mivel mindig háromfázisú rendszereket vizsgálunk, így a műszerek segítségével maximum $5 * 3$ fázis áram és/vagy feszültség mérhető.

A műszerdoboz hátsó felében található egy GPRS adatkommunikációt végrehajtó modul, amellyel bármikor távolról leellenőrizhető, hogy a mérés megfelelően zajlik-e. Meghatározott mérési hibák esetén automatikus hibaüzenetet küld SMS-ben a mérést végző személynek, így azonnal információt lehet szerezni a hibajelenségről, és időben megoldható a probléma. Ezenkívül előre meghatározott időközönként egy FTP szerverre mérési adatokat tölt fel GPRS adatkommunikációval, hogy a mérési eredmények is figyelemmel kísérhetők legyenek. A GPRS modul és a megfelelően felprogramozott funkciók biztosítják a berendezés biztonságosságát a mérések folyamán és helyettesítik a felügyelő személy feladatait.

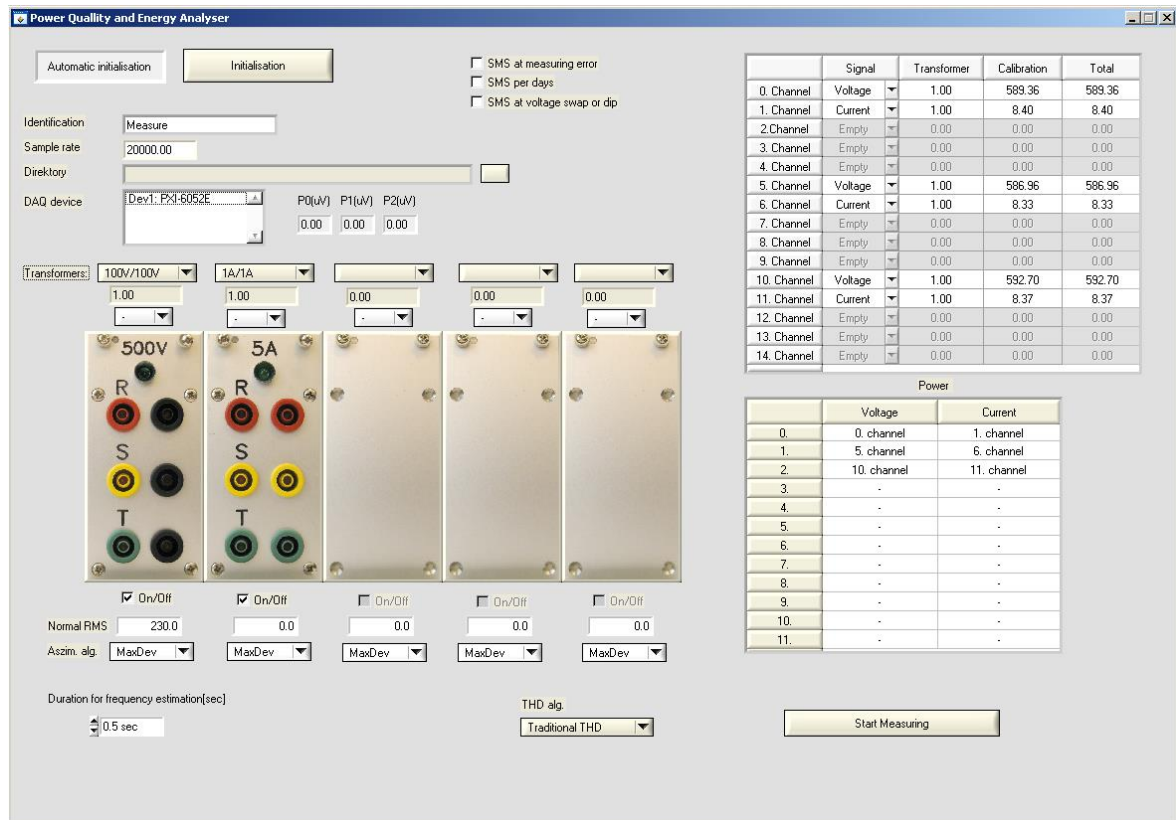


8. ábra: A műszer blokkvázlata

A kifejlesztésre került berendezés lényegesen szélesebb körben alkalmazható, mint a kereskedelemben kapható komplett hálózatanalizáló berendezések, mindemellett lényegesen kisebb költséggel készült, mint a konkurens berendezések. A kereskedelemben kapható analízátorok többsége nagyon drága és többségük még az érvényes szabványok előírásait sem elégítik ki, vagy ha kielégítik, akkor igen korlátozott számú jel mérhető velük (általában 1 háromfázisú rendszer, azaz összesen 6 jel). Ezzel szemben a saját fejlesztésű műszer nagyobb mintavételi sebességgel, pontosabb szinkronizálási eljárásokkal, szélesebb körű jelfeldolgozást tesz lehetővé összesen 15 jel mérésével.

3.2.2. A mérőprogram

A mérőprogram a *National Instruments LabWindows/CVI* fejlesztőkörnyezetében készült. Az alábbiakban a mérőprogram legfontosabb tulajdonságaival, illetve újdonságaival ismerkedhetünk meg. A program kezelőfelülete látható a mérés elindítása előtt a 9. ábrán, mérés közben pedig a 11. ábrán.



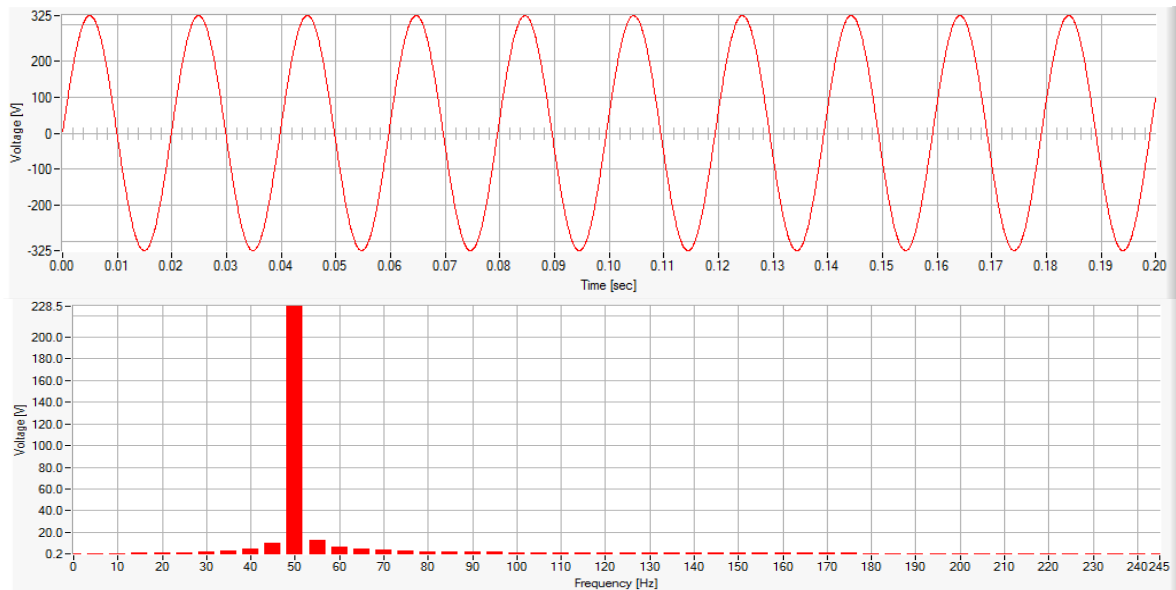
9. ábra: A mérőprogram a mérés elindítása előtt - inicializálás

Mintaszámmal történő szinkronizálás [2]

A pontos és szabvány szerint történő villamos hálózati vizsgálat megköveteli, hogy az egyes villamos hálózat minőségét leíró paraméterek minden egész 10 hálózati periódusra legyenek meghatározva. A hálózati frekvencia ingadozása miatt azonban a 10 periódusok ideje nem állandó, tehát valamilyen módszerrel garantálnunk kell azt, hogy a mintavételezés során mindig 10 periódus legyen analizálva, különben az amplitúdó-frekvencia spektrumon ún. sátras spektrumkép alakul ki, ami hibás THD számítást eredményez (10. ábra).

Megállapítható tehát, hogy valamilyen módon szinkronizálni kell a mérést a hálózati frekvenciához. Általában ezt egy PLL áramkör beépítésével oldják meg, ami a kimenetén egy olyan órajelet állít elő, amelynek frekvenciája egész számú többszöröse a hálózati

frekvenciának. A mintavételezés sebességét ez az órajel határozza meg, ami így biztosítja, hogy a 10 periódusokból mindig egy meghatározott számú minta legyen véve annak a rovására, hogy a minták között eltelt idő nem azonos.



10. ábra: Sátras spektrumkép az amplitúdó-frekvencia spektrumon

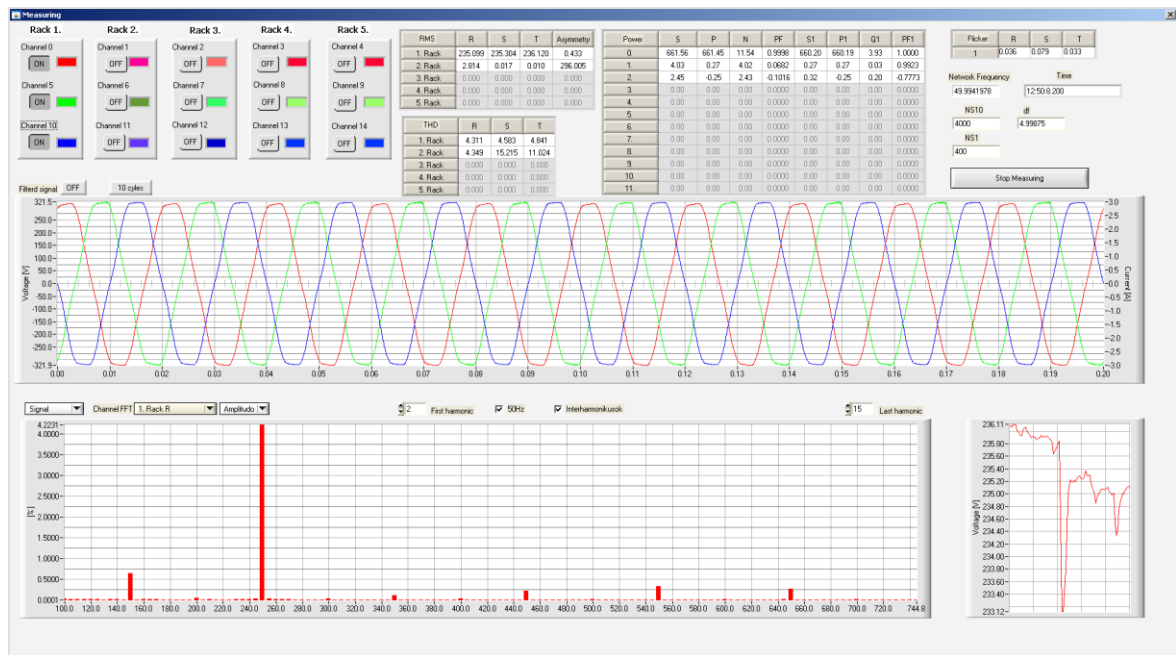
Egy másik, új módszer a *mintaszámmal történő szinkronizálás*. Ebben az esetben nem a minták közötti idő változtatásával állunk rá a 10 periódusra, hanem a minták számának a változtatásával. Kellően magas mintavételezési frekvencia lehetőséget nyújt arra, hogy a minták számának mérés közben való növelésével vagy csökkentésével mindig 10 periódus legyen analizálva a szabványok által előírt pontossággal.

Ha ismerjük a hálózati frekvencia (f_h) pontos értékét, akkor a mintavételezési frekvencia segítségével (f_{mv}) meghatározható, hogy hány mintát ($N_{10\text{periódus}}$) kell venni ahhoz, hogy biztosan 10 periódus legyen analizálva.

$$N_{10\text{periódus}} = \left\lfloor \frac{10f_{mv}}{f_h} + 0.5 \right\rfloor$$

Paraméterek számítása mérés közben

A program másik nagy újdonsága az, hogy minden vizsgált paraméter (pl. RMS, THD) kiszámítása már a mérés alatt, valós időben megtörténik. Ez eddig úgy zajlott, hogy a mérés után kellett kiszámolni minden adatot, és ez a folyamat gyakran napokig is eltartott. Az új mérőprogram segítségével tehát jelentős időmegtakarítás érhető el a paraméterek online kiszámításával, viszont az így keletkezett adatok mentését is meg kell oldani.



11. ábra: A mérőprogram mérés közben

Az adatok mentése

Minden egyes mérés megkezdésekor le kell tárolni bizonyos pluszinformációkat, hogy a későbbiek során értelmezni tudjuk a lementett adatokat. Ilyen kísérő információ például, hogy milyen fiókokkal, milyen mintavételezési frekvenciával mértünk, és még sok más egyéb adat is lementésre kerül. Ezt a konfigurációs fájlt paraméterfájlnak nevezzük (kiterjesztése **.par*), pontos felépítése megtalálható a mellékletek között.

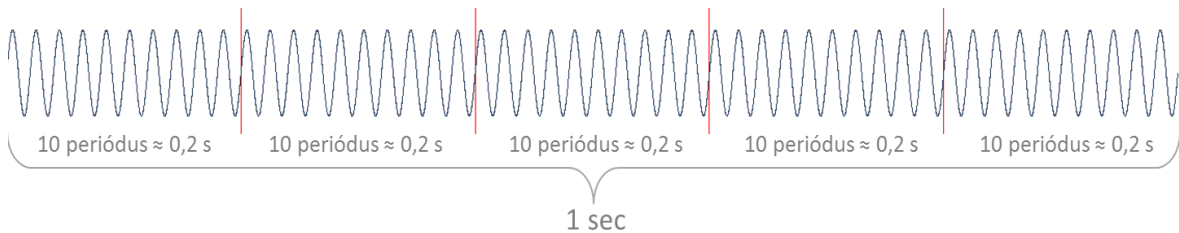
Az adatok mentése a mérés közben történik, mindig tízpercenként. Az egyes tízperces időszakok adatai mindig egy-egy új fájlba kerülnek, az ún. adatfájlokba (kiterjesztésük **.dat*). Az összes csatorna jelalakjainak mentése folyamatosan történik, az összes kiszámított paraméter mentése pedig a tízperces periódusok végén. Az adatfájlok felépítése is megtekinthető a mellékletekben.

Előfordulhat az is, hogy a mérés valamilyen okból kifolyólag leáll, például a mérő számítógép tápellátása megszűnik. A rendszer ilyen helyzetek kezelésére is fel van készítve. Amikor újraindul a számítógép, a mérőprogram a paraméterfájlban elmentett információk alapján folytatja a mérést, ilyenkor új adatfájlt kezd. A feldolgozónak tehát arra is fel kell készülnie, ha hiányzik néhány adatfájl.

3.2.3. Adatok pontossága és mennyisége

Fontos arról külön is beszélnünk, hogy az új rendszer használatával milyen pontosságot érhetünk el.

Minden egyes vizsgált paraméter (például RMS, THD) meghatározása tíz hálózati periódusonként történik. Mivel a hálózat frekvenciája 50 Hz, ez azt jelenti, hogy másodpercenként 5 érték keletkezik minden egyes paraméterből (12. ábra). Ebből egy kis számolással megkapjuk, hogy egy 30 napos mérés során $5 * 60 * 60 * 24 * 30 \cong 13$ millió értékünk lesz minden paraméterből, csatornánként!



12. ábra: Minden paraméter kiszámítása tíz periódusonként, azaz másodpercenként 5 érték keletkezik

Az előbbi példánál maradva, ha 30 napig 20 KHz-es mintavételezési frekvenciát használva mérünk 6 csatornán, akkor több mint 500 GB területet igényelnek a lementett adatok, ez a mai nagy kapacitású merevlemezek mellett is hatalmas tárterületet.

Megállapíthatjuk tehát, hogy az új rendszer segítségével minden eddignél nagyobb pontosságot érhetünk el, azonban az ennek következtében keletkezett hatalmas adatmennyiségek kezelése még a mai korszerű számítógépek segítségével sem egyszerű feladat.

3.3. A fejlesztőkörnyezetről

A fejlesztés *Visual Studioban* zajlott, C# nyelven. A munkát nagymértékben segítette a *National Instruments Measurement Studio* nevű szoftvere. A feldolgozóprogram készítéséhez konkrétan a *Visual Studio 2010*-et, a *Measurement Studio 2010*-et és a *.NET* keretrendszer 4.0-ás verzióját használtam (13. ábra).

3.3.1. Miért esett a választás erre a fejlesztőkörnyezetre?

A projekt szorosan kapcsolódik a mérés technikához, ezért az első pillanattól kezdve fontos szempont volt, hogy olyan környezetet válasszak a fejlesztéshez, amely eredményes lehetőséget biztosít ilyen témájú alkalmazások elkészítéséhez.

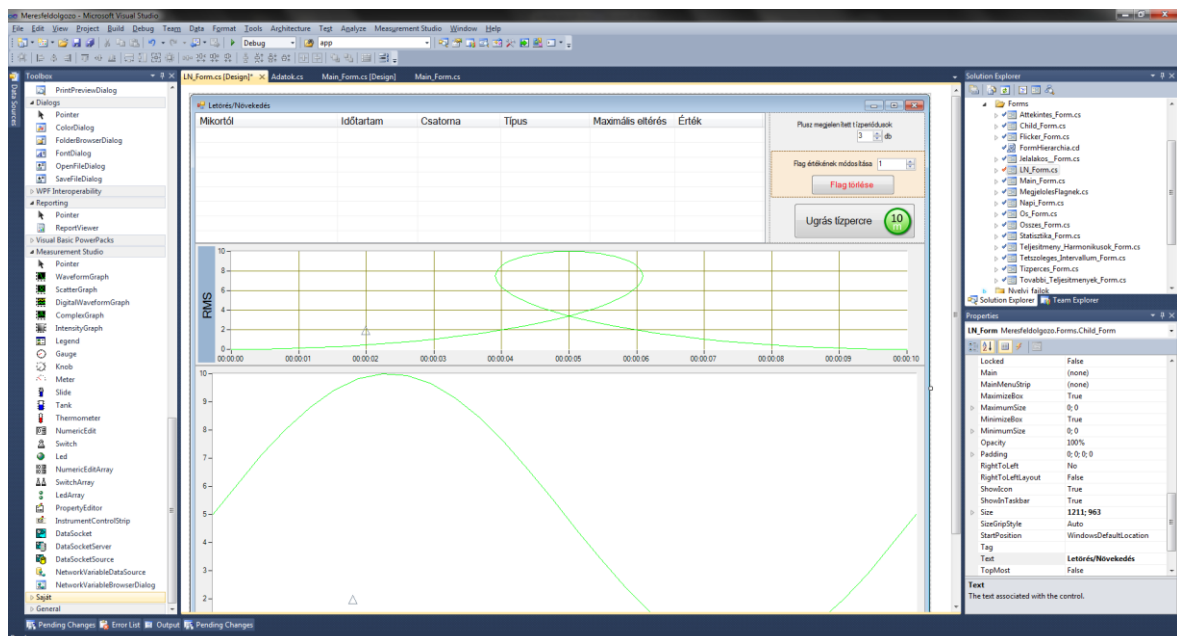
A tanszéken korábban mérés technikai alkalmazások készítése általában C nyelven, CVI környezetben zajlott. Azonban ettől sokkal modernebb és kényelmesebb lehetőséget nyújt az általam választott környezet a fejlesztésre, és különösen a felhasználói felület (user interface) kialakítására. Például a CVI-ban nem érhető el kódkiegészítési (autocompletion) és újratervezési (refactoring) funkciók sem.

Valamint a mai trendeknek megfelelően ki szerettem volna használni az *Objektum Orientált Programozás* hatékony módszereit is, amelyre az alap C nyelv nem nyújt lehetőséget.

Összegezve elmondható, hogy a CVI által nyújtott lehetőségeket kicsit elmaradottnak éreztem, ezért egy sokkal modernebb megoldásokat kínáló alternatíva mellett döntöttem.

3.3.2. Visual Studio

A *Visual Studio* a *Microsoft* terméke, az egyik legnépszerűbb fejlesztői környezet Windows-ra. Több programozási nyelvet is támogat, én ezek közül a C#-ot használtam a szoftver megírásához. A C# egy objektumorientált nyelv, a .NET keretrendszerhez fejlesztették ki, ezért annak minden szolgáltatása, funkciója használható. A .NET keretrendszer támogatottsága kitűnő, az interneten rengeteg hasznos leírás, ill. nyílt forráskódú kiegészítő található, melyek nagy segítséget nyújtottak a fejlesztés során.



13. ábra: A fejlesztőkörnyezet

3.3.3. Measurement Studio

A *Measurement Studio* feltelepítésével rengeteg hasznos eszközzel, osztállyal és kontrollal egészül ki a fejlesztőkörnyezet. Ezek nagyon hatásos segítséget nyújtanak a méréstechnikai alkalmazások elkészítéséhez. A *Measurement Studio* gazdag szolgáltatásai nagymértékben segítették a feldolgozószoftver létrejöttét.

3.3.4. Verziókövető rendszer

Manapság már minden komolyabb szoftverfejlesztés során követelmény valamilyen verziókövető rendszer alkalmazása. A projekt komplexitása megkövetelte, hogy valamilyen, a fejlesztés menetét segítő rendszert én is igénybe vegyek. A választásom az *SVN* rendszerre esett, könnyű kezelhetősége és robusztussága miatt.

Szerverként a tanszék saját *SVN* szerverét, kliensként pedig a nagyon népszerű *TortoiseSVN*-t (<http://tortoisesvn.net>) használtam.

3.4. Kitűzött célok

A fejlesztés előzményeinek ismeretében lássuk tehát, hogy milyen általános célokat tűztünk ki, milyen követelményeket támasztottunk a készítendő programmal szemben.

3.4.1. Az új mérőrendszer adatainak megjelenítése

A program legalapvetőbb és egyben legfontosabb feladata az, hogy képes legyen az összes lementett adat megjelenítésére. Ez nagyon fontos, hiszen az adatok kiértékelése mindig a mérés után, offline módon történik.

3.4.2. Nagymennyiségű adatok kezelése

Az előbbieken már láthattuk, hogy milyen hatalmas adatmennyiségek keletkezhetnek egy-egy mérés során. Még a mai korszerű számítógépek segítségével sem egyszerű ilyen méretű adathalmazok kezelése, ezért valamilyen hatékony módszert, módszereket kellett kidolgozni erre, hogy az alkalmazás használható lehessen.

3.4.3. Rengeteg szolgáltatás

A feldolgozószoftvernek illetve a mérőrendszer egészének olyan komplex szolgáltatáscsomagot kell nyújtani, amely segítségével a tanszéki mérések hatékonyan és egyszerűen elvégezhetőek. A konkurensok képességein és használhatóságán túlmutató rendszer megalkotása volt a fejlesztési munkák célja.

3.4.4. Analízis folyamatának egyszerűsítése

A feldolgozót úgy kell megvalósítani, hogy az analízis folyamata minél egyszerűbb legyen. Ehhez meg kellett vizsgálni, hogy az analízis mely részfolyamatai automatizálhatóak a számítógép segítségével. Tehát fontos cél volt az elemző munka segítése, gyorsítása.

3.4.5. Felhasználóbarát felület

Egyszerű, átlátható felületet kellett kialakítani, a felhasználó munkájának segítése érdekében. Törekedni kellett arra, hogy az egész program felépítése, minden gomb, kontroll funkciója magától értetődő legyen, az alkalmazás használatához ne legyen szükség komolyabb ismeretekre.

4. AZ ELKÉSZÜLT PROGRAM BEMUTATÁSA

Ebben a fejezetben a kész program felépítését, szolgáltatásait mutatom be a felhasználó szemszögéből.

Összegzésként álljon itt egy rövid lista, amelyben az elkészült alkalmazás legfőbb funkcióba nyerhetünk betekintést:

- Grafikon ábrázolhatjuk az RMS, THD, aszimmetria, hálózati frekvencia, $\cos \varphi$ és különböző teljesítményértékeket bármely időtartományban.
- Bármely időpillanatra megjeleníthetők a feszültség, az áram és a teljesítmény jelalakok, illetve ezek amplitúdó-frekvencia spektruma, amely segítségével leolvashatjuk az egyes felharmonikusokhoz tartozó relatív amplitúdó értékeket.
- A program tud villamosenergia-fogyasztást is számolni a kijelölt időintervallumra.
- A feszültség- és áramharmonikusok által okozott torzítási teljesítmények is elemezhetők.
- Megjeleníthetők a harmonikusok hatásos teljesítményei, egészen 2000 Hz-ig, így láthatjuk ezek időben történő változását is.
- Statisztikai analízist végezhetünk mindegyik paraméterre.
- A mérés során történt különböző tranziens jelenségek (például feszültség letörések, növekedések) ki vannak gyűjtve egy külön táblázatba, illetve megtekinthetők az egyes eseményekhez tartozó RMS értékek, valamint jelalakok.
- Minden egyes feszültség-csatornához tartozó flicker értékek grafikonon ábrázolhatók.
- A mérések eredményeit összefoglaló jegyzőkönyvek (*.docx) elkészítéséhez hatékony gépi támogatást nyújt a program, így ezek előállításának folyamata csak minimális emberi interakciót követel.

A fejezet többi részében a program használatának részletes leírása következik, a megvalósítás részleteire pedig a következő fejezetben térek ki.

4.1. A program használatba vétele

Először is lássuk, hogy mit is kell tennünk a program használatba vételéhez! Mindössze az alábbi szoftverek megléte szükséges:

- A .NET keretrendszer 4.0-ás verziója:
<http://www.microsoft.com/hu-hu/download/details.aspx?id=17718>
- Microsoft Visual C++ 2010 Redistributable Package:
<http://www.microsoft.com/download/en/details.aspx?id=5555>

Miután megbizonyosodtunk, hogy rendelkezünk a fentiekkel, a programunk már használatra kész, az összes többi szükséges függőség megtalálható a programunk mellett.

4.2. A felhasználói felület koncepciója

A program grafikus felületének felépítése alapvetően az *MDI (Multiple Document Interface, magyarul több dokumentumos felület)* szemléletmódot követi. Ez konkrétan azt jelenti, hogy a fő ablakon belül több ablak látható (14. ábra), így lehetőség nyílik arra, hogy ugyanazon adathalmazt elemezzük többféle nézőpontból, szemszögből.



14. ábra: Felhasználói felület - "több dokumentumos felület"

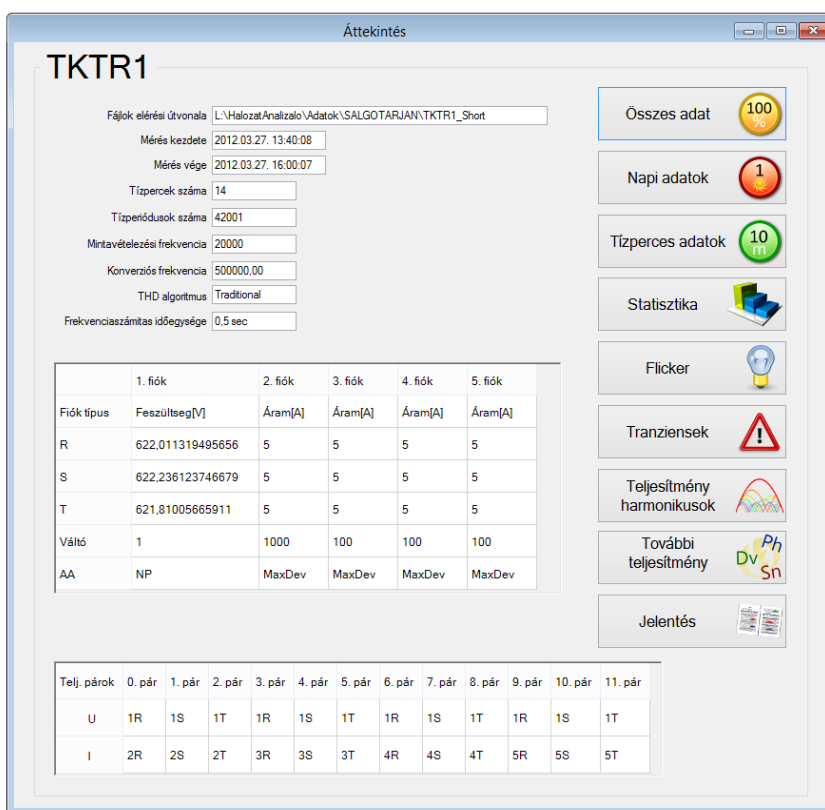
4.3. Adatok betöltése

A program elindítását követően meg kell nyitnunk egy mappát, amelyben mérési adatok találhatóak. Ezt a **Fájl > Betöltés** menüpont segítségével tehetjük meg.

Az adatok első betöltése egy kicsit több időt vesz igénybe, amíg bizonyos inicializációs műveletek végrehajtnak (ezek részletes ismertetése a következő fejezetben található). A folyamat előrehaladásáról az ablak bal alsó sarkában látható folyamatjelző (progress bar) tájékoztat. A második, és minden további megnyitáskor az adatok azonnal betöltődnek.

4.4. Áttekintés ablak

Az adatok betöltése után egy áttekintő ablakot kapunk a mérésről, amiben a mérés legfontosabb paramétereit találhatók meg (15. ábra).



15. ábra: Áttekintés ablak

Innen megtudhatjuk többek között, hogy mikortól meddig mértünk, milyen mintavételezési frekvenciával, milyen fiókokat használtunk, melyik feszültség és áramjelekből számítottunk teljesítményeket, stb.

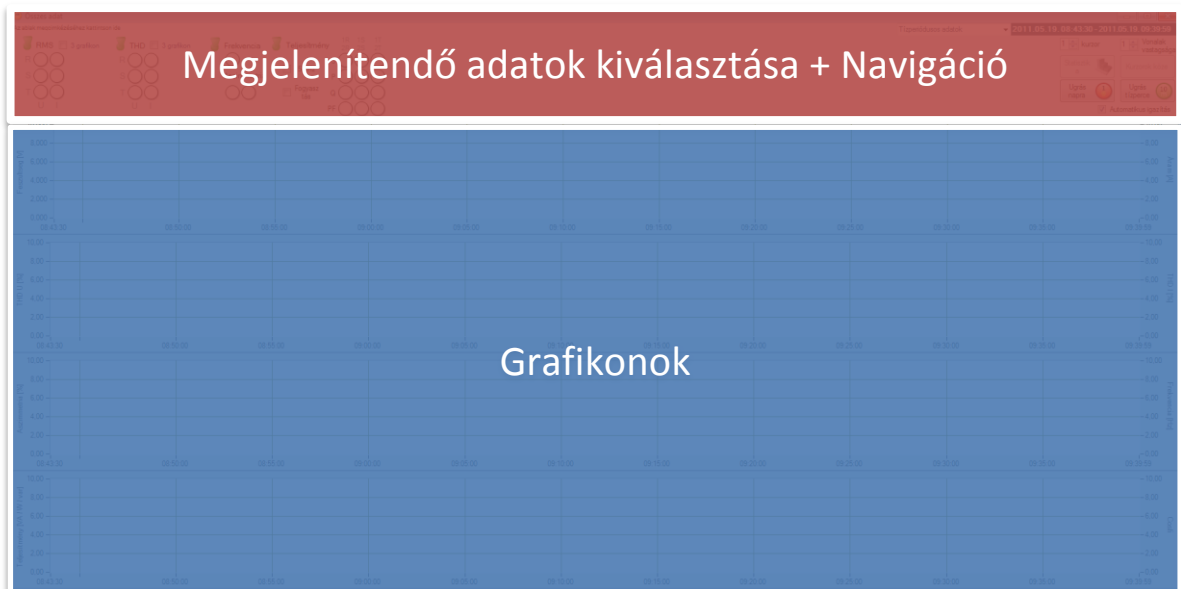
Az összes többi nézetnek is ez az ablak a kiindulópontja. Egyszerre bármennyi ablakunk lehet nyitva, ennek csak az általunk használt számítógép teljesítménye szabhat határt.

4.5. Egyszerű nézetek

Mint már láthattuk, hatalmas adatmennyiségek keletkeznek egy-egy mérés során. Ennek kezelésére ki kellett dolgozni valamilyen hatékony módszert, hogy a programunk működése optimális legyen. Az egyik ilyen módszer az, hogy azt az időintervallumot, ameddig a mérés zajlott, virtuálisan kisebb intervallumra osztottuk. Így alakultak ki az *Összes adat*, *Napi adatok* és a *Tízperces adatok* nézetek.

4.5.1. Közös jellemzők

Mind a három nézet felépítése ugyanazokra az alapokra épül. A képernyő felső kis részén kiválaszthatjuk, hogy milyen adatok legyenek a képernyőn és különböző navigációs gombokat láthatunk, a többi részen pedig az adatok tényleges megjelenítését szolgáló grafikonok találhatók (16. ábra).



16. ábra: Ablakok általános felépítése

Ezekben a nézetekben a következő paraméterek tekinthetők meg:

- RMS értékek minden csatornára (1. grafikon)
- THD értékek minden csatornára (2. grafikon)
- Hálózati frekvencia és aszimmetria (3. grafikon)
- Látszólagos, hatásos, meddő teljesítmények és a teljesítménytényező (4. grafikon)

Az ablak felső részében lehetőségünk van az egyes grafikonok megjelenítésére és eltüntetésére, az aktuális igényeknek megfelelően.

Kiválaszthatjuk, hogy milyen vastag vonalakkal rajzoljunk, illetve hogy hány kurzor legyen a grafikonokon. A kurzorok együtt mozognak az egyes grafikonokon, ezáltal nagyon egyszerű egy bizonyos időpillanathoz tartozó különböző értékek leolvasása.

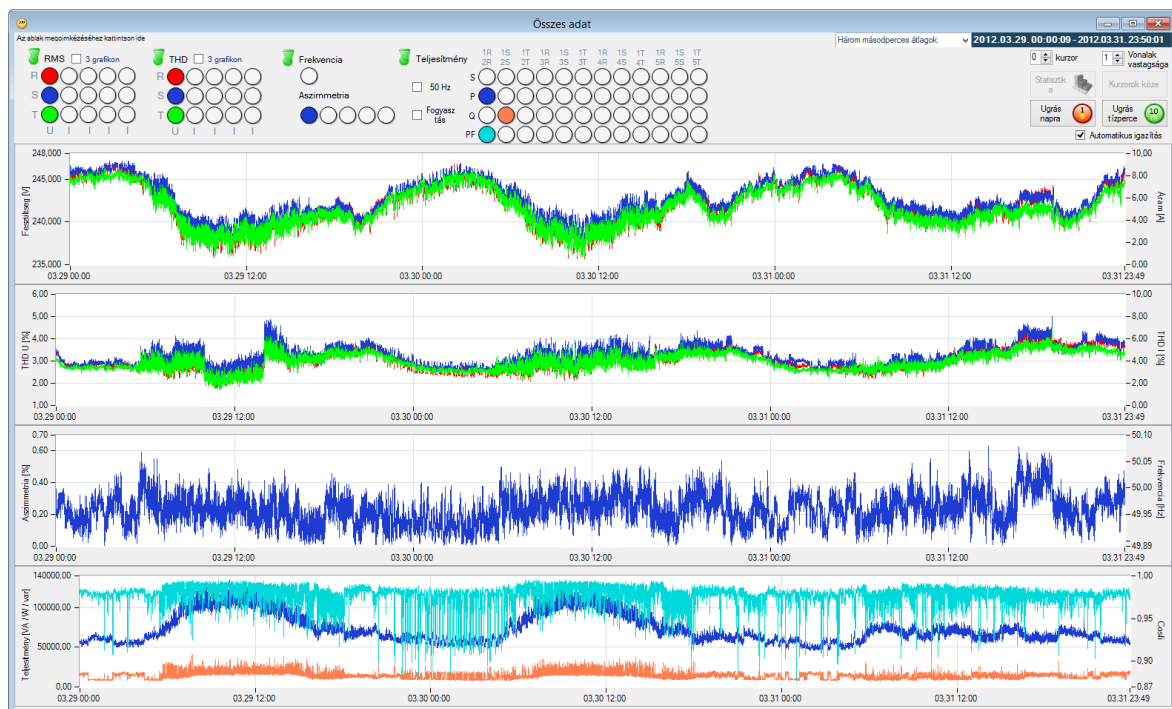
Két kurzor segítségével kijelölhetünk bármilyen tetszőleges intervallumot. Lehetőségünk van ezen intervallum adatainak megnyitására egy külön ablakban (ekkor ugyanazokat a paramétereket láthatjuk az új ablakban).

A kurzorok léptethetőek a billentyűzet segítségével is (17. ábra).

	Balra	Jobbra
1. kurzor	←	→
2. kurzor	A	D

17. ábra: Kurzorok mozgatásának billentyűparancsai

Az RMS és THD értékek megjelenítésekor kiválaszthatunk egy „3 grafikon” nevű opciót is, amelyet bekapcsolva az egyes fázisokhoz tartozó értékek külön grafikonokon jelennek meg. Ez azért hasznos, mert mindig háromfázisú rendszereket vizsgálunk, és az opció használatával könnyedén összehasonlíthatjuk az egyes fázisokhoz tartozó értékeket.



18. ábra: Összes adat nézet

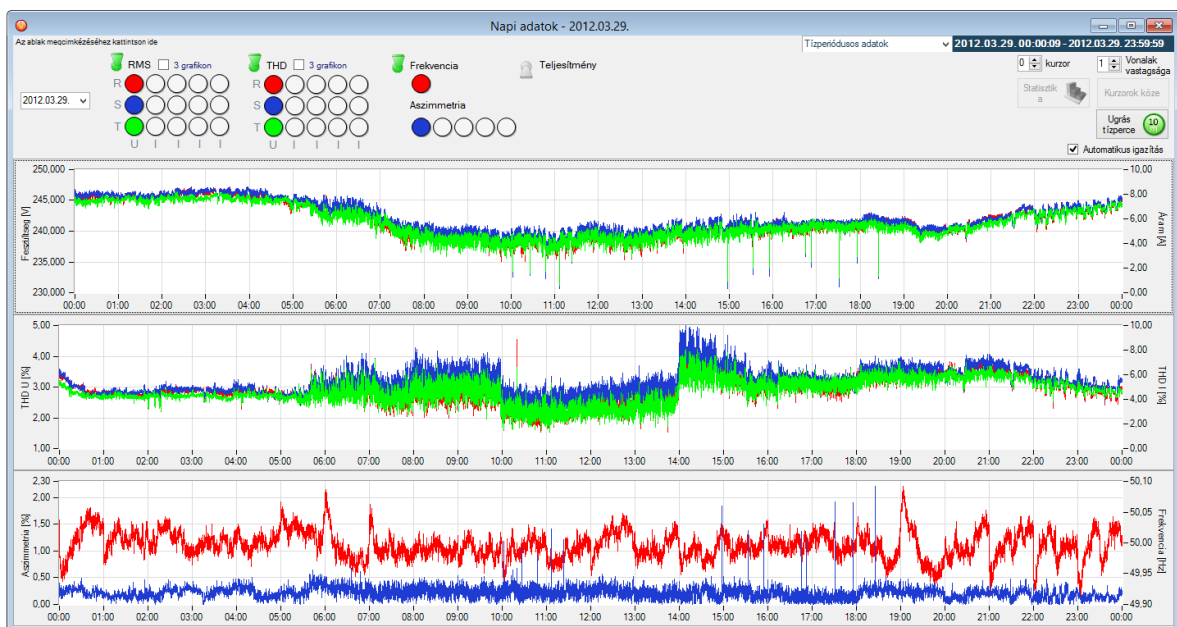
4.5.2. Összes adat

Ebben a nézetben a teljes mérés adatai láthatóak (18. ábra). Tehát hosszú mérések esetén rengeteg a megjelenítendő adatértékek száma, ezért a hatalmas memóriaigény miatt az Összes adat nézetből csak egy darab lehet nyitva az alkalmazáson belül.

Ha valamelyik részt meg szeretnénk vizsgálni részletesebben is, a kurzorok által kijelölt időponthoz tartozó *Napi adatok* vagy *Tízperces adatok* nézetre ugorhatunk.

4.5.3. Napi adatok

Napi adatok nézetben a mérés egyes napjai láthatóak, 0-24 óráig (19. ábra). Ez a nézet kiegészül a bal felső sarokban egy legördíthető listával, amely segítségével válthatunk az egyes napok között.



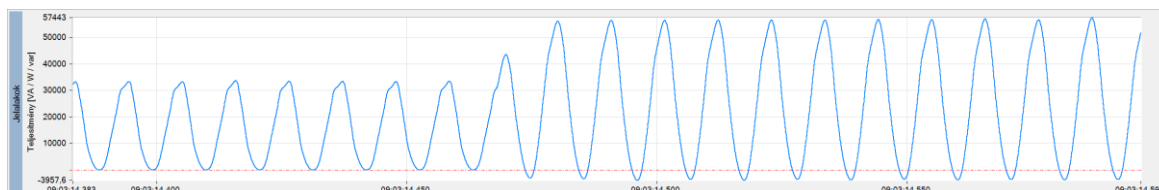
19. ábra: Napi adatok nézet

4.5.4. Tízperces adatok

Ebben a nézetben az egyes tízperces fájlok adatai jeleníthetők meg (20. ábra). Itt már lehetőségünk van a feszültség, áram, és teljesítmény jelalakok (21. ábra) megjelenítésére is, ill. ezekre FFT analízist is végezhetünk. Az FFT grafikonon a jel felharmonikus összetevőit elemezhetjük.



20. ábra: Tízperces nézet, jelalak és FFT grafikonokkal



21. ábra: Teljesítmény jelalakok

4.6. Fogyasztás

Bármelyik nézetben megtekinthetők az egyes teljesítményekből számolt fogyasztások a kijelölt időintervallum alatt az alábbi táblázatos formában (22. ábra).

1R-2R	1S-2S	1T-2T	3R-4R	3S-4S	3T-4T
163,36 kVAh	255,55 kVAh	190,88 kVAh	158,85 kVAh	250,07 kVAh	189,99 kVAh
138,65 kWh	204,67 kWh	175,44 kWh	136,98 kWh	201,47 kWh	175,27 kWh
86,15 kvarh	152,42 kvarh	74,51 kvarh	80,17 kvarh	147,58 kvarh	72,57 kvarh

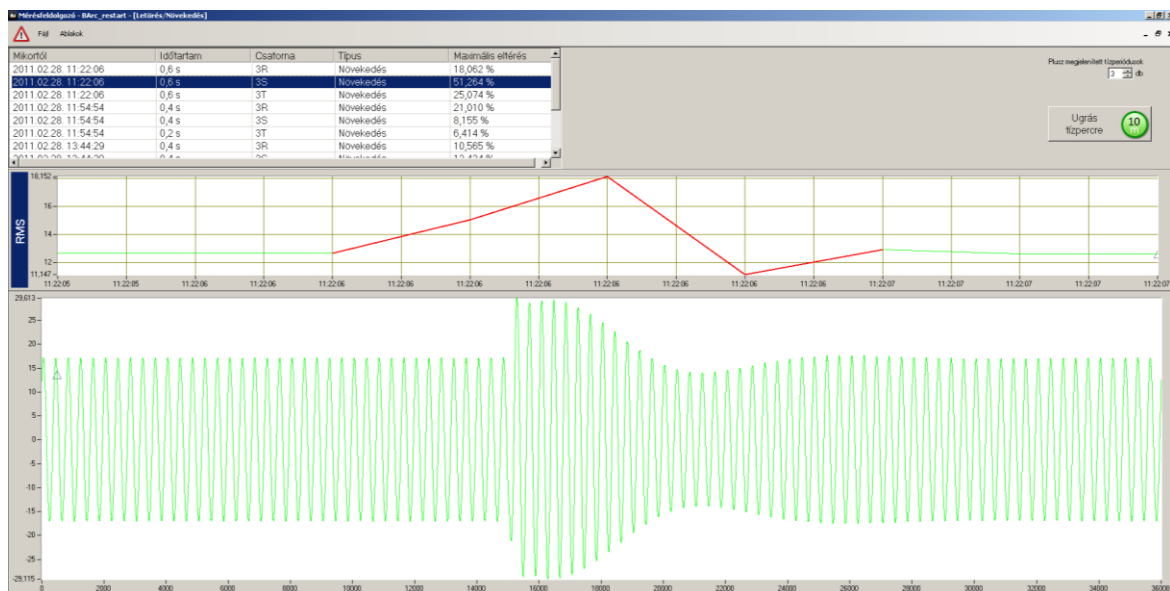
22. ábra: Fogyasztás adatok táblázatos formátumban

4.7. Tranziensek ablak

Ha a mérés során történt valamilyen tranziens jelenség (például feszültségletörés, emelkedés), ezek kigyűjtve megtekinthetők a *Tranziensek* ablakban (23. ábra), táblázatos formában.

Ebben a táblázatban a jelenségek legfontosabb jellemzői láthatóak:

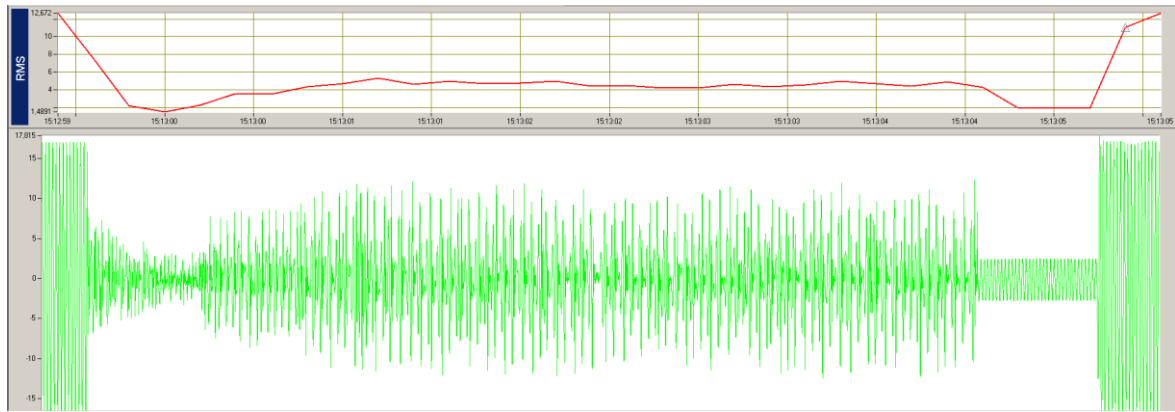
- Kezdőidőpont
- Időtartam
- Melyik csatornán történt
- Jelleg (növekedés vagy letörés)
- Maximális eltérés a névleges effektív értéktől



23. ábra: Tranziensek - Feszültségnövekedés

Ha a lista egy elemére kattintunk, megjelenik a kiválasztott jelenséghez tartozó RMS értékek illetve jelalakok a grafikonokon. Mindig a teljes tranziens jelenik meg.

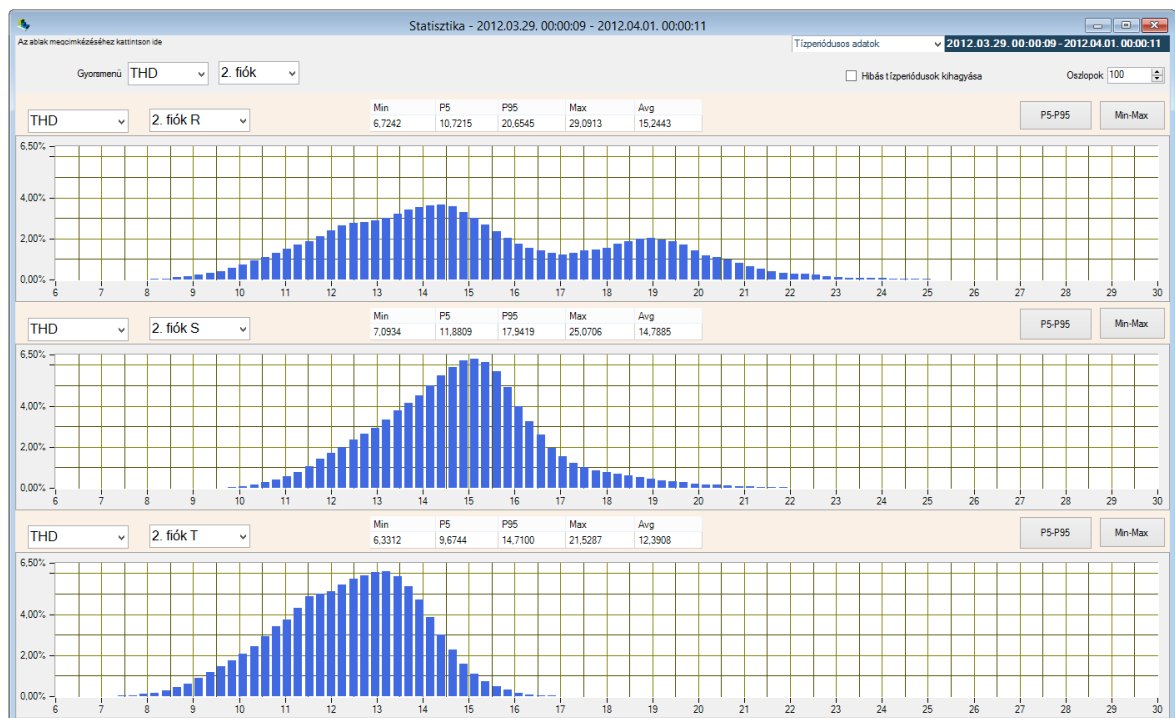
Lehetőségünk van továbbá egy tetszőlegesen kijelölt intervallumot megjelölni tranziensnek, illetve törölhetjük is az egyes eseményeket is.



24. ábra: Tranziensek - Feszültségletörés

4.8. Statisztika

A statisztika funkció segítségével bármely vizsgált paraméterről készíthetünk statisztikai analízist, bármely időtartományban (25. ábra). Ha az *Áttekintés* ablakból nyitjuk meg a *Statisztika* funkciót, akkor a teljes mérés adatait elemezhetjük. *Napi adatok* és *Tízperces adatok* nézetekből használható az *Ugrás statisztikára* nevű funkció is, mely segítségével az éppen ott megjelenített intervallum adatait vizsgálhatjuk statisztikailag.

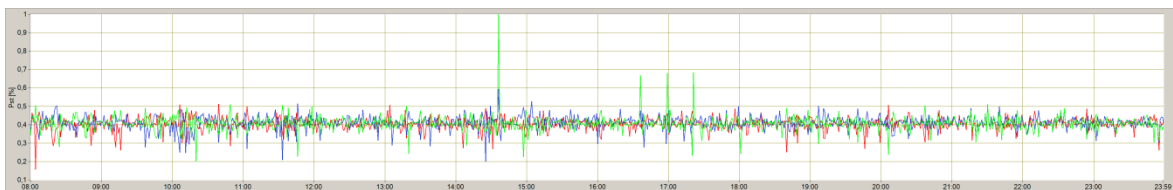


25. ábra: Statisztika ablak

4.9. Flicker

A villogás (flicker) a feszültség-ingadozás következménye, amelyet a villamos energia termelése, átvitele vagy elosztása közben fellépő zavarok idézhetnek elő, de a legjellemzőbb ok a nagy változó terhelések alkalmazása, amelyeknek gyorsan ingadozik a hatásos és meddő teljesítményigényük. Pszichológia hatása jelentős lehet, a villogó fényforrások rossz közérzetet, így a munkaminőség területén negatív hatásokat eredményezhetnek.

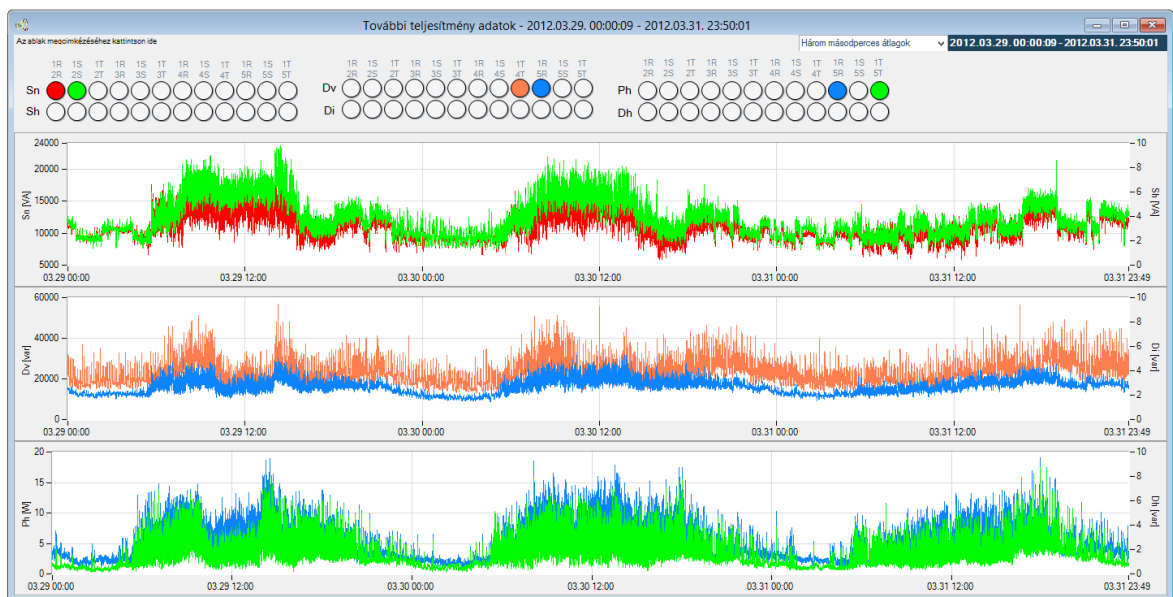
A mérés alatt kiszámolt flicker értékek is megjeleníthetők a programban (26. ábra).



26. ábra: Flicker grafikon

4.10. Torzítási teljesítmények

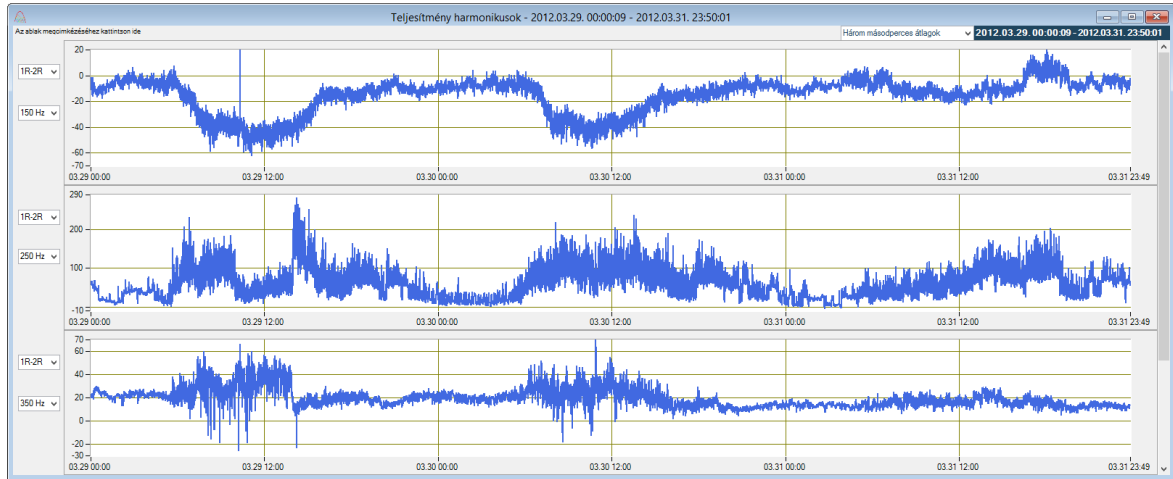
A teljesítményeket egy teljesen új nézőpontból is vizsgáljuk. Az elterjedt szemléletmód nem számol a villamos hálózaton megjelenő feszültség és áram felharmonikusokkal, ezért csak látszólagos, hatásos, és meddőteljesítményeket különböztet meg. A teljesítmények harmonikustartalmát is figyelembe véve azonban meghatározhatók a harmonikusokra vonatkozó hatásos és meddőteljesítmények is, az alapharmonikus teljesítmények, valamint a torzítási teljesítmények is. A szoftver lehetőséget nyújt ezen paraméterek elemzésére is (27. ábra).



27. ábra: Torzítási teljesítmények

4.11. Teljesítmény harmonikusok

Ebben az ablakban megtekinthetők bármely teljesítményre az egyes harmonikusok hatásos teljesítményei, egészen 2000 Hz-ig (28. ábra).



28. ábra: Teljesítmény harmonikusok

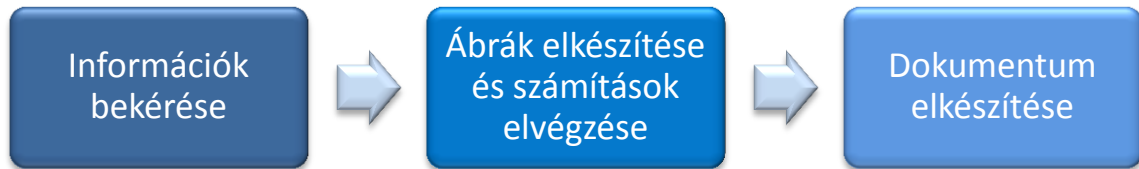
A harmonikus hatásos teljesítmények, valamint a torzítási teljesítmények változásának vizsgálata új lehetőséget nyit arra, hogy a zavarterjedési irányok meghatározhatóak legyenek.

4.12. Jegyzőkönyv készítése

Az egyik legfontosabb szolgáltatás a jegyzőkönyvkészítés. Segítségével a mérési eredményeket összefoglaló jelentés néhány gombnyomás után, teljesen automatikusan generálódik, nagyban megkönnyítve ezzel a felhasználó munkáját. (A jegyzőkönyvek elkészítése eddig manuálisan történt, a *PrintScreen* billentyű és valamilyen képszerkesztő segítségével, ez akár több napi munkát is jelenthetett.)

A program a jelentéskészítést több logikai lépésben végzi el (29. ábra). Az első lépésben egy varázsló segítségével bekérjük a felhasználóktól a szükséges információkat, mint például a melléklet címe, a csatornák megnevezései, mely napok adatait tartalmazza a dokumentum. A korábban megadott adatokból a program automatikusan ajánlatot tesz az egyes megnevezésekre, így általános esetben például a fiókok neveinek megadása után a csatornák és teljesítmények megnevezéseinek nem is kell változtatni.

Miután minden szükséges információt megadtunk, hátra is dőlhetünk, hiszen a jegyzőkönyv elkészítése ezután automatikusan zajlik.



29. ábra: A jegyzőkönyvkészítés logikai lépései

Második lépésben a program automatikusan beolvassa a szükséges adatokat, ábrázolja ezeket grafikonon, ill. kiszámolja a szükséges statisztikai értékeket. Az összes grafikon lementi képfájlként a háttértárolóra *png* formátumban (ha esetleg szükség lenne külön a grafikonokra is). A képek felbontása állítható, a kívánt minőség elérése érdekében. Túl nagy felbontást sem érdemes viszont beállítanunk, mert ilyenkor akár több száz MB is lehet az elkészített állomány mérete, melynek kezelése nehézkes lehet.

Harmadik lépésben a konkrét dokumentum elkészítése történik, az előző lépésekben lementett adatok és ábrák alapján. A jegyzőkönyv *docx* formátumban készül el. Az elkészített jegyzőkönyvek formátumára megtekinthető egy minta a mellékletek között.

A dokumentum elkészítése gombnyomásra, de nem pillanatok alatt történik. A jelentéskészítő rutin lefutásához szükséges idő azonban nagymértékben függ attól, hogy hány csatorna adatait kell feldolgozni, valamint attól, hogy milyen hosszú mérési időszakokat jelöltünk ki. Természetesen fontos a programot futtató számítógép teljesítménye is. Például egy 15 csatornás mérésből készített 7 nap adatait tartalmazó dokumentum körülbelül 5 perc alatt készült el egy viszonylag gyorsnak mondható 4 magos számítógépen.

4.13. További szolgáltatások

4.13.1. Grafikonok testreszabása

A felhasználó igényei szerint testre szabhatja minden egyes grafikon megjelenését. Ki kell jelölni a grafikon, és az alábbi billentyűkombinációk segítségével módosíthatjuk a kinézetet. A magasság a *Ctrl* és a „+” billentyűk lenyomásával növelhető, valamint a *Ctrl* és a „-” billentyűk lenyomásával csökkenthető.

A programban általában a grafikonok jobb és bal oldalához is skálázunk (bal oldalra legtöbbször a feszültségértékeket, jobb oldalra pedig az áramértékeket). Sokszor zavaró hatást kelthet a rácsvonalak sokasága. Ezért a billentyűzet segítségével állítható, hogy a bal illetve jobb oldalhoz tartozó segédvonalak megjelenjenek-e (*I* – bal oldali, *O* – jobb oldali vonalak be- és kikapcsolása).

4.13.2. Grafikonok exportálása

A program bármelyik grafikonját ha kijelöljük, a „Ctrl + S” billentyűkombinációval lehetőségünk van az adott grafikont képként elmenteni. A felugró párbeszédablakban kell beállítanunk a mentés formátumát az alábbiak közül: *.bmp, *.gif, *.jpg, *.png.

4.13.3. Konfigurációs állomány

A program könyvtárában található egy konfigurációs állomány, neve **Konfig.xml**. A felhasználóhoz kötődő beállításokat itt tároljuk. A fájl egyszerű szövegszerkesztővel (pl. *Jegyzettömb*) szerkeszthető.

A felhasználói felület nyelvének beállítása

A fájlban a <Nyelv> tag értékét kell a kívánt nyelvre állítani. Egyelőre a magyar és angol nyelvek támogatottak, ezek használatához a „hu-HU” illetve az „en-US” értékeket kell beírunk.

A grafikonok beállításai

A konfigurációs állomány segítségével beállíthatóak továbbá a programban használt grafikonokhoz kapcsolódó alábbi opciók is:

<Plot_szinek>: A lenti példa szerint, RGB komponensek segítségével állíthatjuk be a használni kívánt színeket.

<Grafikon_magassaga>: Beállíthatjuk, hogy a megjelenő grafikonok alapértelmezés szerint milyen magasak legyenek (a szélesség mindig igazodik az ablak aktuális méretéhez).

<ExportaltGrafikonSzelessege>: A 4.13.2 pontban ismertetett grafikonok exportálása funkció használatakor innen, a konfigurációs állományból olvassa be a program, hogy az exportált kép milyen méretű legyen.

Konfigurációs fájl - részlet

```
<Konfig
  <Nyelv>hu-HU</Nyelv>
  <Grafikon_magassaga>190</Grafikon_magassaga>
  <Plot_szinek>
    <SColor R="255" G="0" B="0" />
    <SColor R="30" G="60" B="210" />
    <SColor R="0" G="250" B="0" />
    <SColor R="0" G="217" B="217" />
  </Plot_szinek>
  <ExportaltGrafikonSzelessege>400</ExportaltGrafikonSzelessege>
</Konfig>
```

5. A FEJLESZTÉS SORÁN ALKALMAZOTT MEGOLDÁSOK, A MEGVALÓSÍTÁS RÉSZLETEI

Miután a program használatát, kezelését a felhasználó szemszögéből megismertük, ebben a fejezetben a szoftver megvalósításának legfőbb részleteiről lesz szó a tervező, a programozó szemszögéből.

5.1. Miért tart sokáig az adatok betöltése első alkalommal?

A mérési adatok első betöltésekor több inicializációs művelet végrehajtására is szükség van. Ezek lefuttatására csak az első alkalommal van szükség, ekkor maximum néhány perc alatt lezajlanak, és eredményük külön elmentésre kerül, hogy a további megnyitások várakozás nélkül megtörténjenek.

Ebben a részben ismertetésre kerülnek a lezajló műveletek, nem feltétlen időrendi, hanem logikai sorrendben.

Általánosságban megállapíthatjuk, hogy bár az adatok első betöltésekor lehet, hogy várnunk kell néhány percre, de ez a kis idő bőven megéri, hiszen az egyszeri várakozás után a program további működése drámaian felgyorsul.

5.1.1. A paraméterfájl feldolgozása

Mindenekelőtt a paraméterfájl információt kell feldolgoznunk, hiszen nélküle nem értelmezhetőek az adatfájlok sem. Innen olvassuk ki például, hogy hány darab és milyen típusú fiókokat, milyen mintavételezési frekvenciát használtunk, a teljesítményeket milyen feszültség és áramjelből számoljuk, stb. Ezeket az információkat (és még sok mást is) az általam megírt *Adatok* nevű osztály adattagjaiban tároljuk el.

5.1.2. Az adatfájlok feldolgozása

A paraméterfájl után az adatfájlokat dolgozzuk fel. Ez egy komplex művelet, több külön részre bontható, egyes részeit a fejezet későbbi pontjaiban tárgyalom.

5.1.3. A keletkezett új adatok mentése

Az első megnyitáskor keletkezett összes adat tárolására létrehozunk a mérési adatokat tartalmazó mappán belül egy almappát (neve általában *X1*). Mindeközben az eredeti paraméterfájl és az adatfájlok változatlanok maradnak.

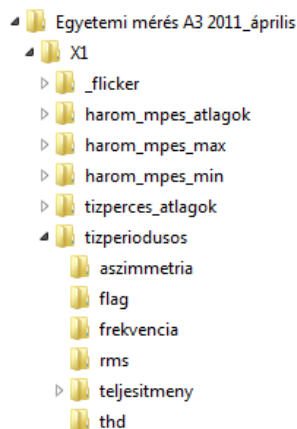
A paraméterfájl adatait és az egyéb metaadatokat tartalmazó **Adatok** objektumot is ebbe a mappába mentjük el, `adatok.xml` néven. Az objektum elmentéséhez a .NET keretrendszer *XML* szerializációs szolgáltatását használom. A mentés menete nagyon egyszerű, mindössze az alábbi néhány sor szükséges.

```
XmlSerializer serializer = new XmlSerializer(typeof(Adatok));  
using (StreamWriter sw = new StreamWriter(new FileStream(path, FileMode.Create)))  
{  
    serializer.Serialize(sw, this);  
}
```

Később, a deszerializálás (visszatöltés) is hasonlóan egyszerű.

```
XmlSerializer serializer = new XmlSerializer(typeof(Adatok));  
using (StreamReader sr = new StreamReader(new FileStream(path, FileMode.Open)))  
{  
    Tarolo.Adatok = (Adatok)serializer.Deserialize(sr);  
}
```

Az **Adatok** osztály tartalmán kívül az összes további feldolgozott adat a feldolgozási mappába kerül az alábbi rendszerezett formában (30. ábra).



30. ábra: Feldolgozott adatok mappaszerkezete

5.1.4. Tízperiódusokhoz tartozó időpontok kiszámítása

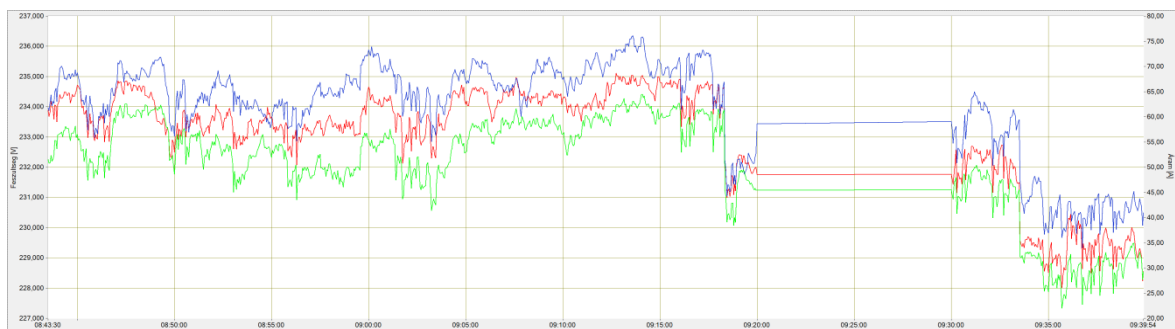
Mivel a hálózati frekvencia sajnos nem állandó, ezért a vizsgált tízperiódusok hossza sem állandó (nem pontosan 0,2 sec), ezért ahhoz, hogy minden egyes értékhez pontos időt tudjunk rendelni, meg kell határoznunk az összes tízperiódus hosszát. Abból, hogy hány mintából ($N_{10periódus}$) áll egy-egy tízperiódus, egyszerűen meghatározhatjuk, hogy pontosan mennyi ideig ($t_{10periódus}$) is tartott az adott tízperiódus:

$$t_{10periódus} = \frac{N_{10periódus}}{f_{mv}}$$

Minden adatfájl elején pedig megtalálható, hogy mikor kezdődött az adott tízperces intervallum. Tehát ehhez az időponthoz hozzáadva a tízperiódusok hosszát, megkapjuk, hogy mikor kezdődtek az egyes tízperiódusok.

Ezek az időpontok is lementésre kerülnek, minden egyes időpontot egy `double` szám reprezentál.

A pontos időadatok meghatározása mellett a megoldás másik előnye, hogy segítségével minden paramétert ábrázolhatunk `ScatterGraph` grafikonokon, ahol az ábrázoláshoz a `PlotXY(double[] xdata, double[] ydata)` függvényt használjuk. Az X tengely adatai (`xdata`) lesznek a lementett időpontok, az Y tengely adatai (`ydata`) pedig a vizsgált értékek. Így megoldottuk azt is, hogyha hiányzó adatfájljaink vannak, a grafikon képen az időadatok helyesek maradnak (31. ábra). Hiányzó adatfájlok akkor fordulhatnak elő, ha a mérés bármilyen okból leáll, például megszűnik a mérőműszer tápellátása. Ilyenkor sajnos adatvesztés történik, de a számítógép újraindítása után a mérés újraindul.



31. ábra: Hiányzó adatfájlok esetén is helyes időtartományt használunk

Egyik mérésünk alatt volt éppen a tavaszi óraállítás, amikor is 2-ről 3-ra kell átállítani az órákat, ám ezzel sem volt probléma, hiszen a számítógép automatikusan átállt a nyári időszámításra, és a feldolgozás során is időhelyesek maradtunk, a „kiesett” óra helyén a fenti ábrához hasonló szakadást láthattunk.

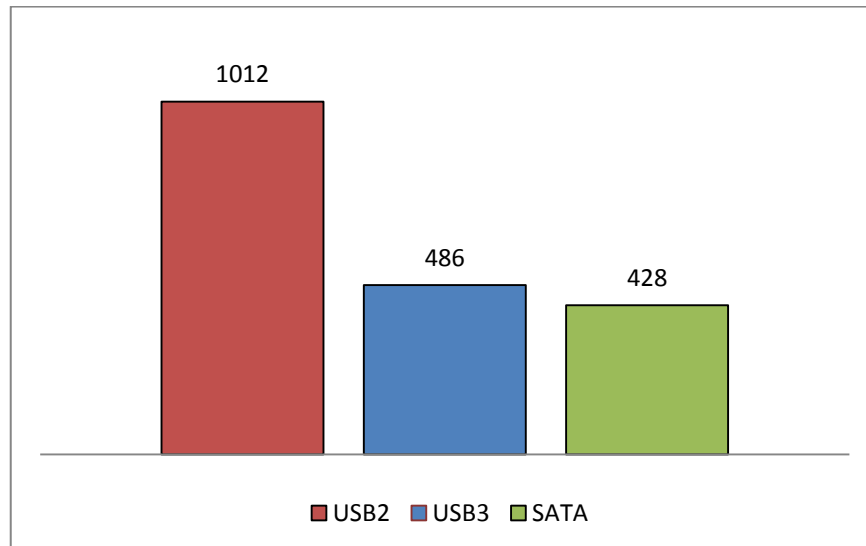
5.1.5. A feldolgozási idők az adathordozók csatlakozója függvényében

A mérési adatokat sokszor hordozható merevlemezeken tároljuk (természetesen van biztonsági másolat is), így a feldolgozás is általában ezekről történik. Teszteltem azt is, hogy a winchesterek csatlakozási módja mennyire befolyásolja a feldolgozás sebességét.

A teszteket egy 4 magos *AMD Phenom II 965* (3.4 GHz), 12 GB memóriával felszerelt gépen végeztem, így a számítógép teljesítménye nem lehetett a szűk keresztmetszet. Ugyanazt a winchestert (*Samsung 750 GB, 7200 rpm*) használtam mindhárom esetben, előbb

USB2-es, majd USB3-as és SATAII-es porton keresztül. A teszteléshez egy 8 napos, 9 csatorna és 6 teljesítmény adatait tartalmazó adatsort választottam.

Az alábbi ábrán láthatóak a teszt eredményei. Az értékek másodpercben szerepelnek (tehát a kisebb érték a gyorsabb).



32. ábra: Az adatok első feldolgozásának ideje (másodpercben) a csatolók függvényében

Az eredményekből megállapítható, hogy a használt interfész típusa nagymértékben meghatározza a feldolgozási sebességet. Az USB2 mára már kicsit elavult, ez meg is látszik az eredményeken, az USB3 viszont már majdnem eléri a belső SATA csatoló sebességét. Ezért célszerű lenne a jövőben USB3-mas külső winchestert és ilyen porttal rendelkező számítógépet használni az elemzéshez.

5.2. Nagyméretű adatok kezelése

Ahogy már az előző fejezetekben is láthattuk, a mérések során hatalmas adatmennyiségek keletkezhetnek. Ennek kezelésére valamilyen hatékony módszert, módszereket kellett kidolgozni.

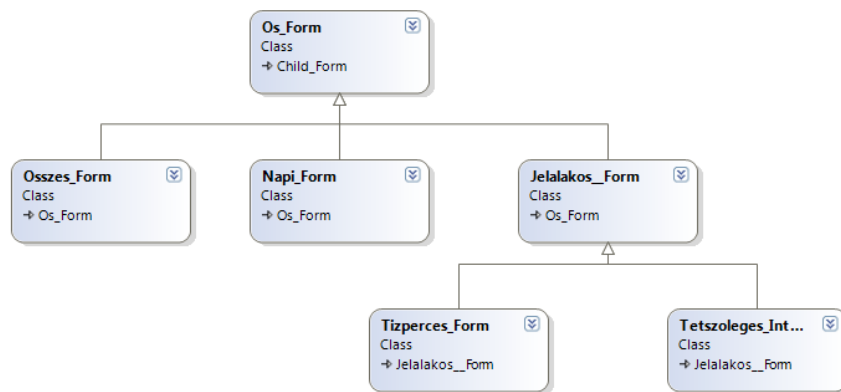
Alapvetően két ilyen módszert használunk:

- Logikai nézetek kialakítása
- Aggregált értékek számítása

5.2.1. Logikai nézetek

Az egyik ilyen módszer az, hogy azt az időintervallumot, ameddig a mérés zajlott, virtuálisan kisebb intervallumra osztottuk. Az előző fejezetben láttuk, hogy az így kapott nézetek kinézete csak nagyon kis mértékben különbözik egymástól. Ezért ezt úgy oldottam meg, hogy az *Objektum Orientált Programozás* öröklődés tulajdonságát felhasználva készítettem egy őszosztályt, és ebből származtattam a speciális osztályokat.

Az őszosztályban (neve **Os_Form**) található meg az összes olyan komponens illetve függvény, amelyre minden nézetben szükség van, a gyermekosztályokban így csak a speciális funkciókat megvalósító elemeket kellett implementálni. Az osztályhierarchiát az alábbi *UML* osztálydiagram szemlélteti.



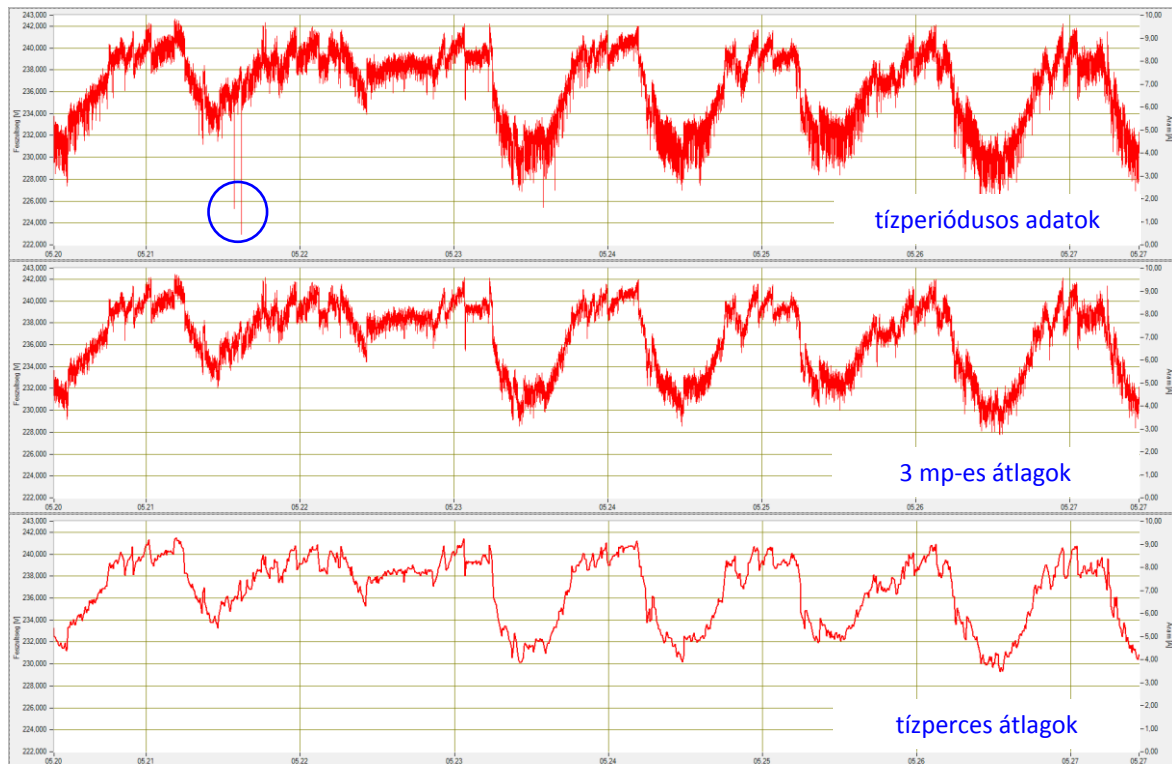
33. ábra: Osztályhierarchia

5.2.2. Aggregált értékek számítása

A másik módszer a nagyméretű adathalmazok kezelésére az, hogy kiszámítunk különböző átlag- ill. szélsőértékeket minden egyes paraméterből. Ennek segítségével a vizsgált adathalmaz mérete a töredékére csökkenthető. Az aggregált értékek kiszámítása három másodperces, illetve tízperces időintervallumokra történik meg. Bizonyos szabványok is előírják ezen értékek vizsgálatát.

Miért van szükség a minimum és a maximum értékek meghatározására is?

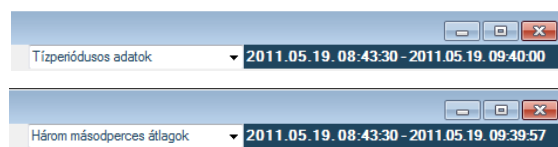
Ha az átlagolni kívánt értékek között van néhány nagyon kiugró érték, az átlagban ezek nem mutatkoznak meg, a villamos energia elemzése során pedig ezen kiugró értékek vizsgálata nagyon fontos. Az alábbi ábrán három grafikon látható, mindhárom grafikonon ugyanazon egyhetes mérés adatai látszanak. Az első grafikonon az eredeti, tízperiódusos értékek láthatóak, a másodikon az ezekből kapott 3 másodperces átlagértékek, a harmadikon pedig a tízperces átlagolás után kapott értékek láthatóak.



34. ábra: Az átlagolás hatása elnyomja a kiugró értékeket

Az első grafikonon látható néhány kiugró érték, viszont a második és harmadik grafikonon megfigyelhető, hogy ezek a kiugrások „elvesztek” az átlagolás hatása miatt. Ezért fontos külön meghatározni az átlagértékek mellett a minimum és maximumértékeket is.

Minden ablak (ahol ennek értelme van) jobb felső sarkában megtalálható, hogy éppen milyen adatokat látunk, és milyen időintervallumban. Itt tudunk váltani is, hogy milyen adatokat szeretnénk látni: tízperiódusos, három másodperces vagy tízperces aggregált értékek.



35. ábra: Az ablakok jobb felső sarkában látható információs sáv

Az aggregált értékek kiszámítása az adatok első megnyitásakor történik, az adatok rendszerezését követően.

5.3. Adatok beolvasásának optimalizálása

5.3.1. Adatok rendszerezése

Minden tízperces intervallum összes adatát egy-egy adatfájl tárolja (az adatfájlok pontos felépítése megtekinthető a mellékletek között). Ha viszont például egy hét adatait szeretnénk megtekinteni, a felépítésből adódóan körülbelül 1000 adatfájlból kellene kiolvasni a logikailag összetartozó információkat. A program kezdetleges verziójában ez így működött, de nagyon komoly teljesítményproblémák voltak ezzel a megközelítéssel (1 hét adatainak a beolvasása több percre is eltarthatott), pedig a beolvasandó adat mennyisége relatív kicsi (a példában kb. 20-30 MB) a merevlemezek folyamatos olvasási sebességéhez képest. A problémát az okozta, hogy az igényelt adatok nem szekvenciálisan helyezkedtek el (azaz nagyon szét voltak szórva) a lemezen, a merevlemezek véletlen elérési ideje pedig igen nagy. Erre logikusan az a megoldás, hogy az összetartozó adatokat egymás mellett kellene tárolni.

Azt a kompromisszumos módszert választottam, hogy minden mérés esetén, az adatok első megnyitásakor a feldolgozóprogram végigmegy az összes adatfájlon, és az összetartozó információkat egymás mellé másolja. Fontos megjegyezni, hogy az adatfájlokat közben nem módosítjuk. Igaz, hogy így ezek az adatok kétszer is megtalálhatóak lesznek a lemezen, de minimális plusz helyet igényelnek. Ez azért van így, mert a mérési adatok közül a lementett jelalakok foglalják le a tárterület nagy részét, az ezekből kiszámított paraméterek a teljes méret csak kb. 1 százalékát teszik ki, így minimális tárterület növekedésért cserébe a feldolgozó működése hihetetlen mértékben felgyorsul.

A hatékony működés érdekében az adatfájlból egyszerre olvasom be a szükséges adatokat ömlesztve, majd egy `MemoryStream` objektum segítségével navigálok a beolvasott dolgok között.

5.3.2. Beolvasást végző függvények felépítése

A program működése során, amikor konkrétan adatok betöltésére van szükség, az első perctől világos volt számomra, hogy a beolvasást más szálon (`Thread`-en) kell végrehajtani, hiszen nem megengedhető, hogy a felhasználói felület (GUI) akár több másodpercig működésképtelen legyen.

A `WinForms` technológiában csak arról a szálról módosíthatjuk a GUI elemeit, amelyik szál létrehozta azokat. Ezért nyilvánvaló volt, hogy a külön szál által beolvasott adatokat át kell adni a fő szálnak, és így kell őket kirajzolni a GUI-ra.

Az alábbi példán keresztül láthatjuk, hogy hogyan is épül fel minden beolvasást, majd kirajzolást végző függvény. Először is létrehozunk 2 tömböt, amelyek a grafikonok X és Y tengelyének adatait jelképezik. Utána beolvassuk mind az X, mind az Y adatokat. Lock-olásra azért van szükség, hogy ne változtathassa meg más szál a beolvasni kívánt intervallumot, mert így nem összetartozó adatokat ábrázolnánk. Mivel ezek a függvények nem a fő szálon futnak le, a tényleges kirajzolást szükséges „visszaadni” a fő szálnak a `BeginInvoke` függvény segítségével.

Az alábbiakban egy beolvasást végző függvény látható.

```
protected void rms_getdata(int csatorna, int grafikon)
{
    double[] xdata;
    double[] ydata;

    lock (szakaszJelzo.Szakasz)
    {
        xdata = szakaszJelzo.Szakasz.getIdopont();
        ydata = szakaszJelzo.Szakasz.getRMS(csatorna, flagekSzamitanak);
    }

    this.BeginInvoke(kirajzolas, new object[] {
        rms_grafikonok[grafikon], csatorna, xdata, ydata });
}
```

5.3.3. Beolvasási sor használata

Tehát világos, hogy az adatok beolvasását végző függvényeket „át kell adni” egy másik, háttérben futó szálnak. Több megoldást kipróbáltam, és egyértelműen az alábbi megközelítés adta a legjobb, leggyorsabb eredményt.

Létrehoztam egy statikus osztályt (`WRKR`), amely tartalmaz egy sort (queue-t), amelyben a futtatni kívánt függvényeket (pontosabban a `System.Action` típusú delegátumokat) gyűjtöm. Létrehoztam egy új, a háttérben futó szálát, amely egy végtelen ciklust futtat. Ebben a ciklusban mindig ellenőrzöm, hogy van-e a várakozási sorban végrehajtandó elem. Ha van, végrehajtom, ha nincs, a szál futását nagyon kis időre felfüggesztem (a `Thread.Sleep(25)` parancs segítségével). A `Sleep` parancs kiadása nagyon fontos üresjáratban, mert ha nem tesszük, akkor az közel 100%-os CPU használatot okoz úgy, hogy közben nem is végzünk semmilyen lényegi feladatot!

Minden beolvasást végző függvényt tehát a `WRKR` osztálynak adok át futtatásra, az alábbi módon: `WRKR.Add(StartWork);`

Viszont mivel a `System.Action` típusú delegátumoknak nincs paramétere, ezért a paraméterekkel rendelkező függvények meghívása ilyen egyszerűen nem lehetséges.

De a lambda szintaxis használatával névtelen (anonymous) metódusokat is átadhatunk, amelyek belsejébe már akármilyen kifejezést írhatunk, így általánosan használható a megoldás.

```
WRKR.Add( () => rms_getdata(e.n, f) );
```

Részlet modell megvalósításának forráskódjából:

```
private static ConcurrentQueue<Action> sor;
...
while (ready)
{
    if (sor.IsEmpty)
    {
        Thread.Sleep(25);
    }
    else
    {
        Action fv;

        if (sor.TryDequeue(out fv))
        {
            fv.Invoke();
        }
    }
}
```

A megoldás előnyei:

- A tényleges, fájl szintű olvasási műveleteket a **BinaryReader** osztály függvényeivel végzem. Azonban, ez alaptól nem szálbiztos, tehát ha több szálat használnék, a szálak között valamilyen külön szinkronizációs mechanizmus lenne szükséges. De ha minden olvasási műveletet a fent említett módon végzek, nem jöhet létre versenyhelyzet a szálak között, hiszen minden művelet egyesével, egymás után hajtódik végre.
- A beolvasni kívánt adatok, az adatok rendszerezését követően alapvetően egymás után, szekvenciálisan helyezkednek el. Ezért nem érdemes több különböző szálon beolvasást végző metódusokat futtatni, mert a szálak versengése miatt nem optimális eredmény születne.

5.4. A jegyzőkönyvkészítés részletei

Mint azt a 4.12. pontban is láthattuk, a jegyzőkönyvkészítést három logikai lépésre bonthatjuk: a szükséges adatok bekérésére, az adatok beolvasására és feldolgozására, valamint a dokumentum generálására. A modularitás jegyében az egyes lépéseket külön-külön komponensek valósítják meg.

5.4.2. A dokumentum generálása

A dokumentum generálás megtervezésekor figyelembe kellett vennem azt, hogy a program által készített dokumentum további módosítása bizonyos esetekben még szükséges lehet. Ezért döntöttem a manapság legnépszerűbb szövegszerkesztő, a *Word* formátumának használata mellett.

A jegyzőkönyv konkrétan *docx* formátumban készül el, ehhez az ingyenes és nyílt forráskódú *DocX .NET Library*-t (<http://docx.codeplex.com/>) hívtam segítségül.

Egyszerűen használható, valamint nagy előnye, hogy működéséhez nem szükséges, hogy a gépre fel legyen telepítve a *Word* szövegszerkesztő.

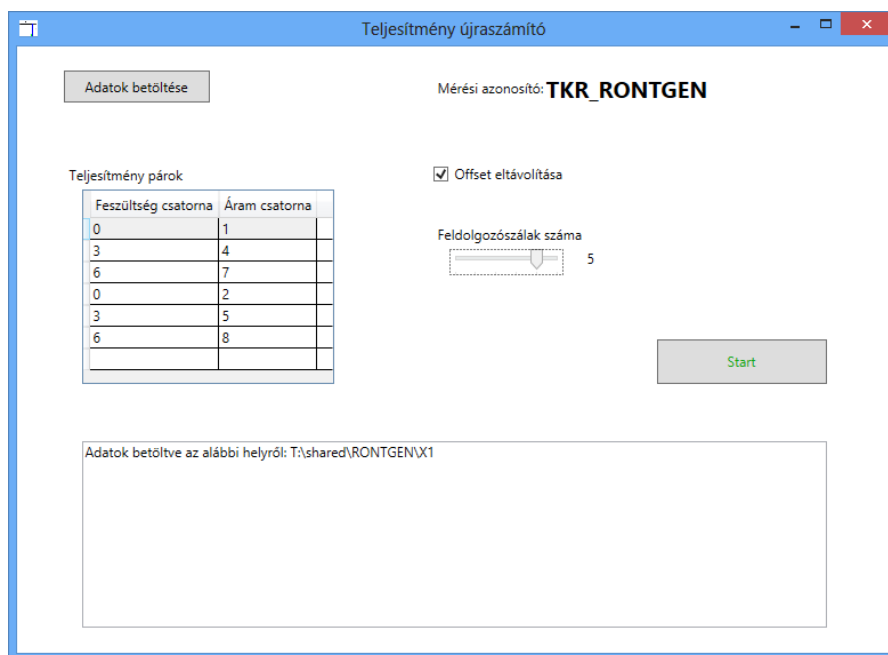
A *docx* formátum *XML* alapú, ezért bizonyos mértékben az emberi szemmel is olvasható. Azért csak bizonyos mértékben, mert igen bonyolult fájlok is előfordulhatnak. A fejlesztés során egyszer mégis kénytelen voltam manuálisan szerkeszteni a fájlt: a grafikonok beszúrásakor, ha nagyobb felbontású képeket szúrtam be, azok lelógtak a lapról, bárhogyan is állítgattam a beszúrás paramétereit. Ezért megfigyeltem, hogy a *Word* hogyan szúr be egy nagyfelbontású képet, és az *XML* kódot megfejtve találtam rá a megfelelő paraméterekre. Az *extent* és az *ext* elemek *cx* attribútumait a kívánt szélességre, a *cy* attribútumot pedig a kívánt magasságra kellett beállítani. Érdekesség, hogy az *Office Open XML* formátum szerint a képek méretét az *EMU* (*English Metric Unit*) mérték szerint kell megadni (1 cm = 360000 EMU).

5.5. További érdekességek

5.5.1. Teljesítmény újraszámító program

Szükségessé vált, hogy néhány paraméter értékét újra kiszámítsuk a lementett jelalakokból, erre pedig egy külön kis alkalmazást írtam, mivel ez nem kapcsolódik szorosan a feldolgozó feladataihoz.

Ennek az az érdekessége, hogy a kiszámítást végző szálak (Thread) számát be tudjuk állítani az indításkor (lásd 37. ábra), így több szál használatával a töredékére csökkenthető a feldolgozási idő.



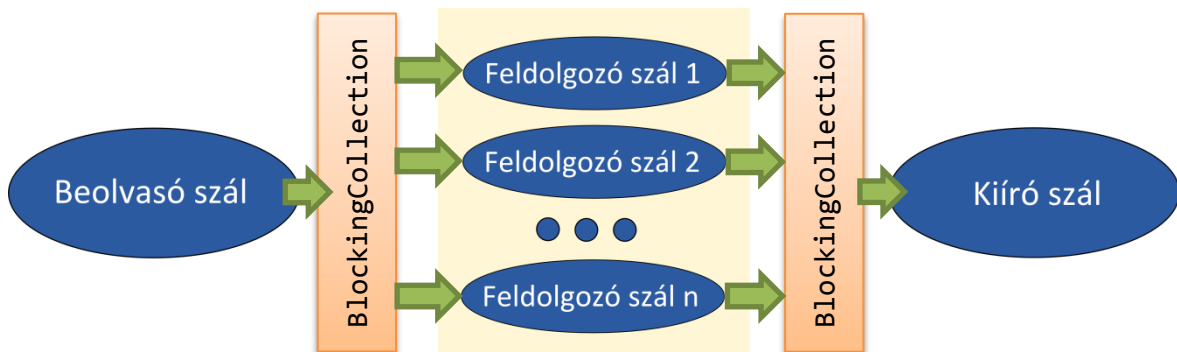
37. ábra: A teljesítmény újraszámító program felülete

Általában nehéz olyan programokat írni, amelyek több szálát használnak, sokszor gondot jelenthet például a szálak közti kommunikáció, szinkronizáció. Ehhez a speciális feladathoz azonban sikerült egy egyszerű és jól működő többszálú modellt megvalósítanom.

A megvalósításhoz a keretrendszer `BlockingCollection` nevű gyűjteményét használtam. Ez a jól ismert termelő-fogyasztó (producer-consumer) probléma megoldására ad egy egyszerű, szálbiztos megoldást. Több szál is adhat hozzá elemeket (az `Add` módszer segítségével) egyidejűleg, ha pedig elértük a tároló kapacitását, akkor blokkolja a hozzáadó szálakat, amíg valaki ki nem vesz egy elemet. Hasonlóan, több szál is vehet ki elemeket (a `Take` módszer segítségével), de ha közben kiürül a tároló, akkor addig blokkolja a szálakat, amíg nem lesz elem benne.

Az elemek hozzáadását a `CompleteAdding` metódus segítségével zárhatjuk le, a fogyasztó szálak pedig az `IsCompleted` tulajdonságot figyelik. Ez akkor válik igazzá, ha befejeztük az elemek hozzáadását és üres a gyűjtemény.

A kidolgozott modell lényege, hogy több adatfeldolgozó szál segítségével gyorsítsuk a feldolgozási időt (38. ábra). Természetesen a több szál használata csak a többmagos (vagy esetleg több processzoros) számítógépeken jelent gyorsulást.



38. ábra: Többszálú feldolgozás modellje

A modell működése nagyvonalakban:

- Létrehozok 1 darab szálát, amely beolvassa az adatokat, és átadja őket az `itemsToProcess` nevű `BlockingCollection` objektumnak.
- Létrehozok több darab (a felhasználó által megadott számú) szálát, amelyek az `itemsToProcess` objektumból veszik ki a feldolgozandó adatokat. A feldolgozott adatokat átadják az `itemsToWriteOut` nevű, szintén `BlockingCollection` típusú objektumnak.
- Létrehozok 1 darab szálát, amely az adatok lemezre írásáért felel. Ez a kiírandó adatokat az `itemsToWriteOut` nevű objektumból veszi.

A beolvasás során továbbá minden beolvasott adatelemnek sorszámot adtam, hogy a kiírás is a megfelelő sorrendben történjen, hiszen több feldolgozószál esetén előfordulhat az, hogy a később beolvasott adatokat hamarabb feldolgozzuk, és így a kiírás sorrendje is felborulna.

5.5.2. Naplózás

A feldolgozó működéséről XML formátumú naplófájlt készítek. Az XML formátum előnye, hogy nagyon egyszerűen feldolgozható, például az *XPath* kifejezések segítségével. A 39. ábrán látható egy példa a naplófájl felépítésére.

Naplózom többek között a következőket:

- *Rendszerinformációkat*: például milyen verziójú *Windows* fut a gépen, 32 vagy 64 bites-e a rendszer. Ezek az esetleg előforduló hibák felderítésében lehetnek segítségünkre.
- *Kivételeket*: ha a program futása során valamilyen kivétel keletkezik, az nem megfelelő működésre utal. A hibakereséshez elengedhetetlen, hogy pontos információk álljanak rendelkezésünkre ezekről.
- *Statisztikai információkat*: például mennyi ideig tartott a feldolgozás vagy éppen a jegyzőkönyvek elkészítése.

```
<logFile>
  <systemInfo computerName="KW" OS="Microsoft Windows NT 6.1.7601 Service Pack 1"
    is64bitOs="True" is64bitProcess="False" processorCount="4" ramSize="12287"
    programDirectory="t:\shared\Feldolgozo\Meresfeldolgozo-120803" date="2012.08.04. 10:56:59" />

  <adatokFeldolgozasa meresiAzonosito="TKR_RONTGEN" tizpercekSzama="11"
    csatornakSzama="9" teljesitmenyekSzama="3" path="T:\shared\rontgen.eredeti">
    <masolas start="2012.08.04. 11:05:32" stop="2012.08.04. 11:05:33" elteltIdo="00:00:00.7225917" />
    <atlagSzamitas start="2012.08.04. 11:05:33" stop="2012.08.04. 11:05:33" elteltIdo="00:00:00.3735474" />
  </adatokFeldolgozasa>

  <exception type="System.ObjectDisposedException" date="2012.08.04. 11:19:00">
    <message>Cannot access a disposed object. Object name: 'Icon'.</message>
    <stackTrace>...</stackTrace>
  </exception>
</logFile>
```

39. ábra: Példa a naplófájl felépítésére

6. A SZOFTVER IPARI KÖRÜLMÉNYEK KÖZÖTT TÖRTÉNŐ ALKALMAZÁSA

A szoftvert ipari méréseink kiértékelése során is használtuk. Az alábbiakban két ilyen mérés tapasztalatairól lesz szó.

6.1. Salgótarján - Szent Lázár Megyei Kórház

2011 tavaszán a salgótarjáni kórház villamos hálózatának energiahatékonysági és minőségi vizsgálatára kapott megbízást a tanszék. A mérés egyidőben három helyszínen zajlott, két transzformátorállomáson és az egyik épület elosztóhelyiségében, nyolc napon keresztül. [14]

6.1.1. Feszültség effektív értéke

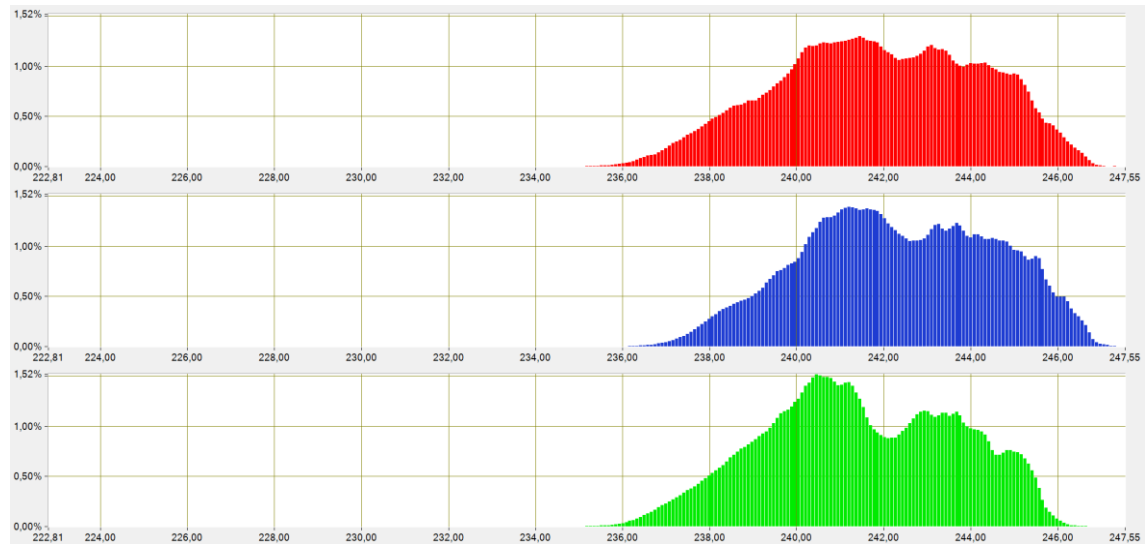
A megfelelő (230V) feszültség szint biztosítja a fogyasztók optimális működését és élettartamát. A szükségesnél magasabb feszültség szint csökkentheti a fogyasztók élettartamát és a fogyasztók egy csoportjánál (ohmikus fogyasztók) a feszültség emelkedésével négyzetes arányban nő a teljesítményfelvétel, ami jelentős többletköltséget okoz. A szabvány a feszültség határait $230 \pm 10\%$ mértékben korlátozza, az áramszolgáltatók a lakossági hálózaton $\pm 7\%$ -on belül vállalják a hálózati feszültség ingadozását, de energiahatékonysági megfontolásból érdemes arra törekedni, hogy csúcsidőben 230 V közeli feszültség szinten működjön a hálózat, ennek érdekében, ahol szükséges, érdemes a transzformátor szekunder oldali feszültség szintjét állítani.

A mért értékek alapján megállapítható, hogy a feszültség szintek a mért időszak 95%-ában 236 V fölött voltak, a minimum értékek csak bizonyos tranziens jelenségek eredményei. Az eloszlás ábrák jól mutatják, hogy a feszültség értéke általában legalább 240 V. Az energiaköltségek minimalizálása érdekében javasolt a transzformátorok szekunder oldali feszültségének csökkentése 6-8 voltal.

Szemléltetésként a TK transzformátorállomás adatai láthatóak az alábbi ábrákon.

	Min	P5	P95	Max	Átlag
R fázis [V]	230.44	237.98	245.45	247.50	241.88
S fázis [V]	230.71	238.68	245.76	247.55	242.32
T fázis [V]	222.81	237.82	245.05	246.91	241.47

40. ábra: A TK tr. állomás szekunder oldali feszültségének statisztikai adatai az R, S, T fázisokon



41. ábra: A TK tr. állomás szekunder oldali feszültségének eloszlásai az R, S, T fázisokon

6.1.2. Röntgengépek okozta tranziensek

Az alábbiakban az egyes röntgengépek által okozott áram tranziens jelenségekről látható néhány érdekes ábra.



42. ábra: Az egyes röntgen berendezések által okozott tranziens jelenségek

6.2. Debreceni Egyetem Orvos- és Egészségtudományi Centrum

Az elmúlt nyáron a Debreceni Egyetem Orvos- és Egészségtudományi Centrum villamos hálózatának vizsgálatára kaptunk megbízást. Összesen 9 transzformátorállomáson mértünk, a transzformátorok szekunder oldalának, valamint néhány leágazásnak feszültség és áramértékeit.

A jelentés mellékletei az új dokumentumgeneráló modul segítségével készültek. A mellékleteket tartalmazó dokumentum 146 oldalas lett és körülbelül 700 ábrát tartalmaz.

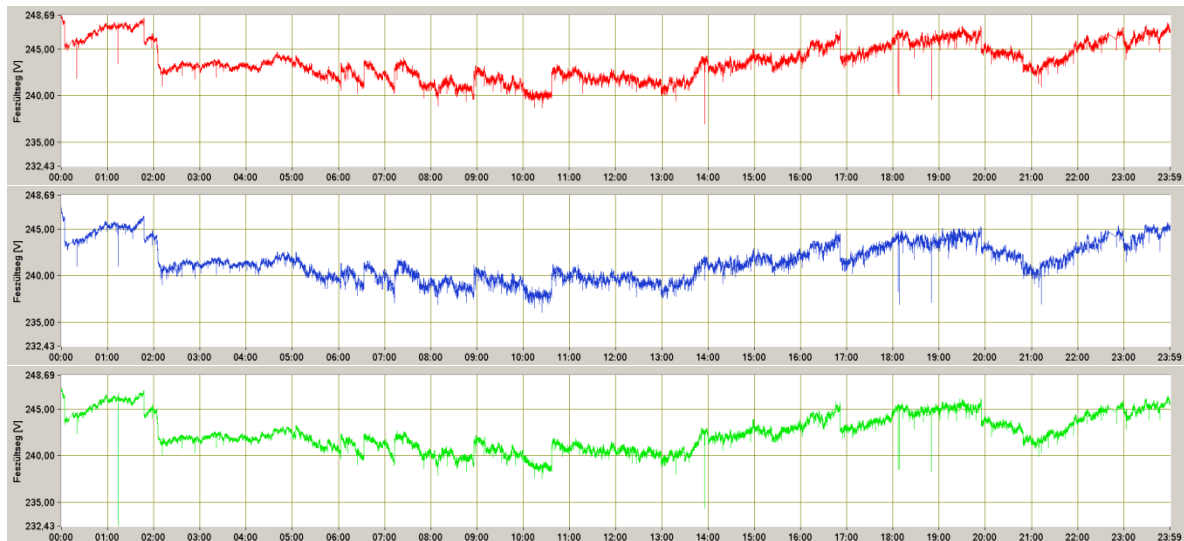
Egy ilyen hatalmas méretű dokumentum kézzel történő elkészítése óriási munka lett volna. Nyugodtan mondhatjuk, hogy az elkészült alkalmazás napokat spórolt meg az életünkből. Az alábbiakban a mérési eredmények néhány érdekes megállapítását közöljük. [7]

6.2.1. Feszültség effektív értéke

A mérési eredményekből megállapítható, hogy több transzformátorállomáson indokolatlanul magas a feszültségszint, a feszültség csökkentése javasolt. Példaként az alábbi ábrákon a C_I transzformátorállomáson mért adatok láthatóak. Az ábrákból leolvasható, hogy mindhárom fázison a minimumértékek 230 volt felettiek, az átlagértékek pedig meghaladják a 240 voltot, ezért itt is ajánlott a szekunder oldali feszültség csökkentése.

	Min	P5	P95	Max	Átlag
C_I transzformátor szekunder oldali feszültség R fázis [V]	236.95	240.81	247.07	248.69	243.74
C_I transzformátor szekunder oldali feszültség S fázis [V]	236.07	238.60	245.05	247.33	241.55
C_I transzformátor szekunder oldali feszültség T fázis [V]	232.43	239.71	245.69	247.31	242.49

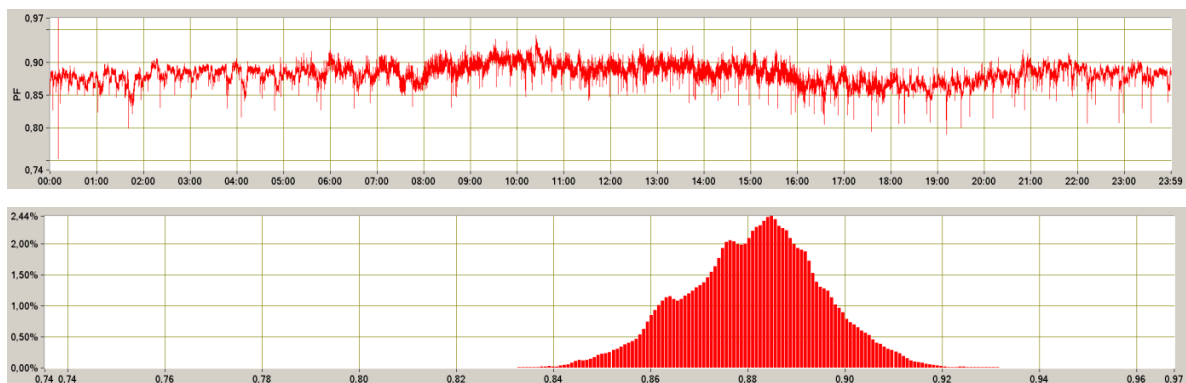
43. ábra: C_I tr. szekunder oldali feszültség effektív statisztikai adatai rendre az R, S és T fázison



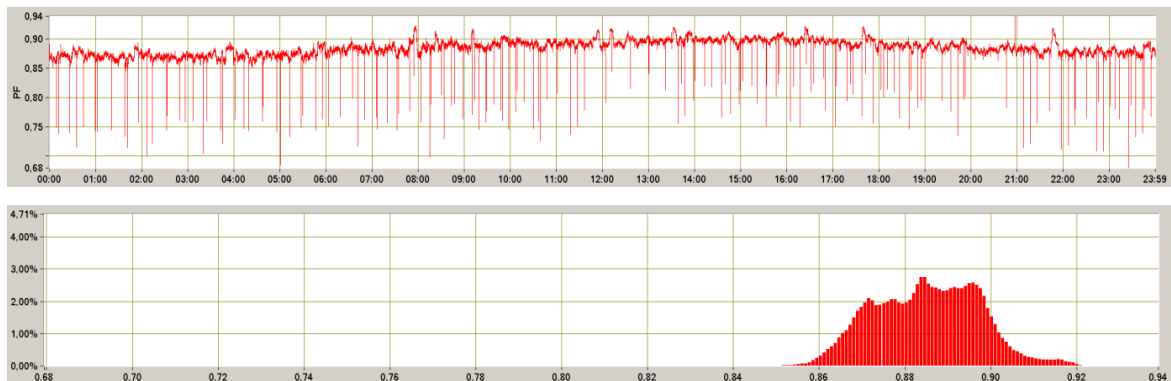
44. ábra: C_I tr. szekunder oldali feszültség effektív értékének idődiagramjai rendre az R, S és T fázison

6.2.2. Meddőenergia

A meddőenergia kétféle módon okoz többlet energiaköltséget. Az egyik közvetlenül jelenik meg meddődíj formájában a villamos energia számlán, a másik „láthatatlan” módon terheli a számlát, mivel a belső hálózaton terjedő meddőáram a kábeleken „lengve” hálózati veszteség formájában növeli a hatásos energiafogyasztást. Ennek mértéke függ a meddőáram mértékétől, a kábelek hosszától, keresztmetszetétől, anyagától. Tapasztalataink azt mutatják, hogy a 0.97 értékű átlagos teljesítménytényezővel működő hálózatok esetében ez a veszteség elhanyagolható, 0.93 alatti értékeknél azonban érdemes számolni ezzel.



45. ábra: A teljesítménytényező idő- és statisztikadiagramja az „A” tr. szekunder oldal R fázisán



46. ábra: A teljesítménytényező idő- és statisztikadiagramja az „Élettan” leágazás R fázisán

Jól látható a fenti ábrákból, hogy a teljesítménytényező értéke általában 0.90 érték alatt van, ezért javasolt meddőkompenzátor telepítése ill. felülvizsgálata.

7. ÖSSZEGZÉS

A Miskolci Egyetem Elektrotechnikai- Elektronikai Tanszékén kifejlesztésre került egy új, korszerű mérőrendszer, amely a villamos energia minden eddiginél pontosabb vizsgálatát teszi lehetővé. Az új rendszer az eddigiektől eltérően a mintaszám szinkronizáláson alapszik. Az én feladatom a mérőrendszer által lementett adatok megjelenítésére, kiértékelésére szolgáló komplex feldolgozószoftver készítése volt.

Az alkalmazást C# nyelven, *Visual Studio 2010* fejlesztőrendszerben és a *Measurement Studio 2010* segítségével készítettem el.

A program elkészítése során törekedtem arra, hogy az analízis folyamata minél egyszerűbb, gyorsabb legyen. Megvizsgáltam, hogy a kiértékelés mely részfolyamatai automatizálhatók a számítógép segítségével, így jött létre például a jegyzőkönyvkészítő szolgáltatás, amely automatikus feldolgozó és dokumentumgeneráló működésével nagyban megkönnyíti a felhasználó munkáját.

Továbbá olyan módszereket dolgoztam ki, melyek segítségével a nagyon hosszú távú mérések kiértékelése sem okozhat gondot (akár több hónapos mérések is előfordulhatnak).

Dolgozatomban bemutattam a program részletes használatát a felhasználó szemszögéből, azaz a munka során milyen lehetőségeket nyújt a szoftver, illetve mely funkciók teszik könnyebbé, gyorsabbá az analízis folyamatát. A következő fejezetben az alkalmazás pontos működésébe, valamint a fejlesztési munka részleteibe is betekintést nyerhettünk.

Természetesen, mint minden alkotást, az elkészített szoftvert is lehet még tovább tökéletesíteni, bővíteni. A fejlesztés során is kiemelt szempontként kezeltem, hogy a későbbiek során is egyszerűen lehessen új szolgáltatásokat integrálni a programba. Jelenleg is folynak kutatások a mérőrendszer továbbfejlesztésével kapcsolatban.

Az elkészített program mára a mérőrendszer szerves részét alkotja, hiszen az adatok elemzése az esetek döntő többségében offline módon zajlik. A mérőrendszert és a feldolgozószoftvert a laboratóriumi tesztek mellett ipari körülmények között is használtuk.

8. SUMMARY

Electrical energy is a significant energy source for industrial-commercial application and also for home users; therefore the quality is a very important attribute of the system. So it became essential, that we could examine the quality of electrical energy by means of several parameters.

In the Department of Electrical and Electronic Engineering of University of Miskolc has been examining and measuring the quality of electrical energy for decades. A new measurement system was developed by the department, which allows us to examine the electrical networks more accurate than ever. The new measurement process required an advanced new software to analyse the results of the measurement.

My task was to create this application which can display and evaluate the saved power parameters. The most important thing was during the development to make the evaluating process as easy and fast as it could be. Especially the new report and document generator function helps the user to speed up the analysing process. The software also had to be expandable, so it could meet the promiscuous new requirements. The program was developed in C# language, in Visual Studio, with the powerful help of the special tools of Measurement Studio.

Advantage of the new measurements system is the calculations of the power parameters are executed during the measuring therefore there is no need to wait hours or days to calculate all different parameters. Consequently immediately after the measurements examining of results can be started. Although most of the cases it is not necessary to use raw measured data, they are saved as well therefore other parameters which were not calculated during measuring can be evaluated in later time.

Since measurements last days a large number of data are sampled, calculated and also stored on the hard drive. Handling of these data is not an easy task. During a 7 days measurements 3 million value will belong to every parameters in each channel. This is a lot of data which can be displayed, examined and evaluated only with today's powerful hardware and computers.

Consequently, the aim of my work is the introducing of the measurements system with the different states of data processing and showing new methods and functions which help and increase the efficiency of data analysing in this power quality field using Measurement Studio.

9. FELHASZNÁLT IRODALOM

- [1] Bátorfi Richárd: Electrical Power Quality and Efficiency Diagnostic System, IMEKO XIX World Congress, September 6-11, 2009, Lisbon, Portugal, pp. 749-752
- [2] Bátorfi Richárd: Hálózati frekvencia ingadozásának vizsgálata mintaszámmal történő szinkronizálás alkalmazásához, XII. Nemzetközi Energetika-Elektrotechnika Konferencia, 2011. október 6-9., Kolozsvár
- [3] Bátorfi Richárd - Hegedűs János - Unhauzer Attila - Váradiné dr. Szarka Angéla: Méréstechnika, Miskolc, Miskolci Egyetem, 2007. 227 p.
<http://www.gepesz.uni-miskolc.hu/hefop/letolt.php?dwn=1merestechika>
- [4] Bencs Róbert: Komplex mérésfeldolgozó alkalmazás fejlesztése Measurement Studio-ban villamos hálózatok elemzéséhez, Szakdolgozat, 2011
- [5] Bencs Róbert: Mérési adatok automatizált feldolgozó- és dokumentum generáló szoftverrendszere villamos hálózat diagnosztikai alkalmazáshoz, XIII. Nemzetközi Energetika-Elektrotechnika Konferencia, 2012. október 11-14., Gyulafehérvár
- [6] David Chapman: Villamosenergia-minőség – Alkalmazási segédlet:
<http://www.rezinfo.hu/villamosenergia-minoseg>
- [7] A DEOEC fázisjavító berendezés telepítését megelőző villamos hálózat energiahatékonysági és minőségi vizsgálata, Szakértői jelentés, Uni-Energy Egyetemi Kutató és Fejlesztő Közhasznú Nonprofit Kft., 2012
- [8] IEC 61000-4-7:2003 Electromagnetic compatibility (EMC) Part 4-7
- [9] IEC 61000-4-30:2003 Electromagnetic compatibility (EMC) Part 4-30: Testing and measurement techniques – Power quality measurement methods
- [10] IEEE Trial-Use Standard 1459-2000, Definitions for the Measurement of Electric Power Quantities Under Sinusoidal, Nonsinusoidal, Balanced, or Unbalanced Conditions
- [11] MSDN Online Library: <http://msdn.microsoft.com/library/default.aspx>
- [12] MSZ EN 50160 szabvány: A közcélú elosztóhálózatokon szolgáltatott villamos energia feszültségjellemzői
- [13] National Instruments Online Forum: <http://forums.ni.com/>
- [14] A salgótarjáni Szent Lázár Megyei Kórház villamos hálózatának vizsgálata, Szakértői jelentés, Uni-Energy Egyetemi Kutató és Fejlesztő Közhasznú Nonprofit Kft., 2012
- [15] Unhauzer Attila: Development of digital flicker meter and specification of disturbance propagation direction, XIX. IMEKO World Congress (Fundamental and Applied Metrology), September 6-11, 2009, Lisbon, Portugal, ISBN: 978-963-88410-0-1, pp. 753-756.
- [16] Unhauzer Edina: Villamos hálózat analízáló műszer mérőrendszerének tervezése, Szakdolgozat, 2011
- [17] Váradiné Szarka Angéla, Measuring harmonic distortion in electrical power networks – New approach, Measurement 43, 2010, 1628-1635

10. MELLÉKLETEK

10.1. A paraméterfájl felépítése

2 Byte (1 unsigned short)	• Verziószám
30 Byte (30 char)	• Mérési azonosító
5 Byte (5 char)	• Fiók típusai (5 fiók)
5 Byte (5 char)	• Fiók be/ki kapcsolt állapotok (5 fiók)
5 Byte (5 char)	• Fiók prefixumok (5 fiók) -1 milli, 0 semmi, +1 kiló, +2 mega
24 Byte (3 double)	• Fiók 1 szorzó (3 fázis)
24 Byte (3 double)	• Fiók 2 szorzó (3 fázis)
24 Byte (3 double)	• Fiók 3 szorzó (3 fázis)
24 Byte (3 double)	• Fiók 4 szorzó (3 fázis)
24 Byte (3 double)	• Fiók 5 szorzó (3 fázis)
40 Byte (5 double)	• Váltó szorzók (5 fiók)
40 Byte (5 double)	• Normál RMS értékek (5 fiók)
1 Byte (1 char)	• Teljesítmények száma (teljes_szama)
24 Byte (24 char)	• Teljesítmények párosítása ($2 * MAX_TELJESITMENY(=12)$)
8 Byte (1 double)	• Valós mintavételezési frekvencia [Hz]
8 Byte (1 double)	• Konverziós frekvencia [Hz]
1 Byte (1 char)	• THD algoritmus
5 Byte (5 char)	• Aszimmetria algoritmus (5 fiók)
8 Byte (1 double)	• Frekvenciaszámítás időegysége [sec]
40 Byte (5 double)	• Kártya polinom (5db)
50 Byte (50 char)	• Mérőkártya típusa

10.2. Az adatfájlok felépítése

x Byte (struct tm)	• Időpont 10 perces kezdet
2 Byte (1 unsigned short)	• 10 periódusok száma a 10 perc alatt (tiz_periodusok_szama)
4 Byte (1 int)	• Összes mintaszám a 10 perc alatt csatornánként (osszmintaszam)
CS * MSZ * 2 Byte (short)	• Minták (csatornaszám * osszmintaszam)
4 Byte (1 unsigned int)	• Minták száma a 10 periódusokban (tiz_periodusok_szama)
PSZ * 8 Byte(double)	• Hálózati Frekvencia értékek(tiz_periodusok_szama)
CS * PSZ * 1 Byte(char)	• Flag hiba jelzés (csatornaszám * tiz_periodusok_szama)
CS * PSZ * 8 Byte(double)	• RMS értékek (csatornaszám * tiz_periodusok_szama)
CS * PSZ * 8 Byte(double)	• THD értékek (csatornaszám * tiz_periodusok_szama)
5 * PSZ * 8 Byte (double)	• Aszimmetria értékek (FIOKOK_SZAMA * tiz_periodusok_szama)
4 * TSZ * PSZ * 8 Byte (double)	• Teljesítményértékek(S, P, Q, cosfi) ($4 * teljes_szama * tiz_periodusok_szama$)
10 * TSZ * PSZ * 8 Byte (double)	• Teljesítményértékek(S1,P1,Q1,PF1,SN,PH,DU,DI,SH,DH) ($10 * teljes_szama * tiz_per_szama$)
40 * TSZ * PSZ * 8 Byte (double)	• Harmonikus teljesítmények ($40 * teljes_szama * tiz_periodusok_szama$)
FESZ_CS * 60 * 8 Byte (double)	• Flicker értékek (Pst)

10.3. Részletek egy, a jegyzőkönyvkészítő szolgáltatás által generált dokumentumból

Uni-Energy Közhaszú Nonprofit Kft.

I. melléklet - A transzformátor

I. melléklet - A transzformátor

I.1. 2012.07.17. 03:10:10 - 2012.07.17. 03:40:10

I.1.1. Feszültség effektív értékek elemzése

	Min	P5	P95	Max	Átlag
A transzformátor szekunder oldali feszültség R fázis [V]	240.63	241.41	241.99	242.16	241.73
A transzformátor szekunder oldali feszültség S fázis [V]	241.24	242.27	242.96	243.19	242.67
A transzformátor szekunder oldali feszültség T fázis [V]	239.76	240.48	241.24	241.44	240.92



I.1. ábra: A transzformátor szekunder oldali feszültség R fázis effektív értékek idődiagramja



I.2. ábra: A transzformátor szekunder oldali feszültség R fázis effektív értékek statisztikai eloszlása



I.3. ábra: A transzformátor szekunder oldali feszültség S fázis effektív értékek idődiagramja



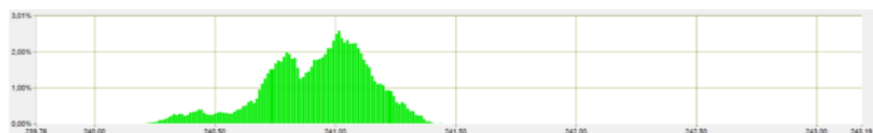
I.4. ábra: A transzformátor szekunder oldali feszültség S fázis effektív értékek statisztikai eloszlása



I.5. ábra: A transzformátor szekunder oldali feszültség T fázis effektív értékek idődiagramja

Uni-Energy Közhazsnú Nonprofit Kft.

I. melléklet - A transzformátor



I.6. ábra: A transzformátor szekunder oldali feszültség T fázis effektív értékének statisztikai eloszlása

I.1.2. Feszültség torzítás (THD) elemzése

	Min	P5	P95	Max	Átlag
A transzformátor szekunder oldali feszültség R fázis [%]	1.29	1.37	1.54	1.60	1.45
A transzformátor szekunder oldali feszültség S fázis [%]	1.22	1.30	1.48	1.64	1.38
A transzformátor szekunder oldali feszültség T fázis [%]	1.14	1.19	1.34	1.50	1.26



I.7. ábra: A transzformátor szekunder oldali feszültség R fázis torzítás (THD) értékének idődiagramja



I.8. ábra: A transzformátor szekunder oldali feszültség R fázis torzítás (THD) értékének statisztikai eloszlása



I.9. ábra: A transzformátor szekunder oldali feszültség S fázis torzítás (THD) értékének idődiagramja



I.10. ábra: A transzformátor szekunder oldali feszültség S fázis torzítás (THD) értékének statisztikai eloszlása

Uni-Energy Közhassznú Nonprofit Kft.

I. melléklet - A transzformátor



I.31. ábra: Nullvezető áram torzítás (THD) értékek idődiagramja



I.32. ábra: Nullvezető áram torzítás (THD) értékek statisztikai eloszlása

I.1.9. Teljesítményfelvétel elemzése

	Min	P5	P95	Max	Átlag
P A transzformátor szekunder oldali R fázis [W]	33860.82	34420.20	39716.94	43224.60	36618.21
P A transzformátor szekunder oldali S fázis [W]	31917.54	32767.29	37267.63	41513.81	34368.23
P A transzformátor szekunder oldali T fázis [W]	33013.07	33317.27	39126.42	43550.09	35437.82
Q A transzformátor szekunder oldali R fázis [var]	17860.97	18007.62	22148.36	29960.32	19232.70
Q A transzformátor szekunder oldali S fázis [var]	17254.16	17667.42	22014.34	30153.72	18895.47
Q A transzformátor szekunder oldali T fázis [var]	19020.54	19450.26	23433.55	30393.81	20573.05
PF A transzformátor szekunder oldali R fázis	0.8211	0.8719	0.8954	0.9032	0.8855
PF A transzformátor szekunder oldali S fázis	0.8064	0.8593	0.8882	0.9009	0.8764
PF A transzformátor szekunder oldali T fázis	0.8012	0.8510	0.8762	0.8978	0.8648



I.33. ábra: A transzformátor szekunder oldali R fázis hatásos teljesítmény idődiagramja



Uni-Energy Közhasznú Nonprofit Kft.

I. melléklet - A transzformátor

I.34. ábra: A transzformátor szekunder oldali R fázis hatásos teljesítmény statisztikai eloszlása



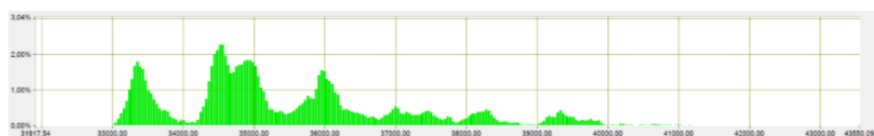
I.35. ábra: A transzformátor szekunder oldali S fázis hatásos teljesítmény idődiagramja



I.36. ábra: A transzformátor szekunder oldali S fázis hatásos teljesítmény statisztikai eloszlása



I.37. ábra: A transzformátor szekunder oldali T fázis hatásos teljesítmény idődiagramja



I.38. ábra: A transzformátor szekunder oldali T fázis hatásos teljesítmény statisztikai eloszlása



I.39. ábra: A transzformátor szekunder oldali R fázis meddő teljesítmény idődiagramja



I.40. ábra: A transzformátor szekunder oldali R fázis meddő teljesítmény statisztikai eloszlása



Köszönetnyilvánítás

A dolgozat a TÁMOP-4.2.1.B- 10/2/KONV-2010-0001 jelű projekt részeként – az Új Magyarország Fejlesztési Terv keretében – az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósult meg.

