Metropolia Ammattikorkeakoulu
Tietotekniikan koulutusohjelma

Eero Putkinen

Moottorikäytön rasitustestilaite

Insinöörityö 13.2.2009

Ohjaaja: tiiminvetäjä Arto Nakari
Ohjaava opettaja: yliopettaja Antti Piironen

Metropolia Ammattikorkeakoulu **Insinöörityön tiivistelmä**

Tekijä Otsikko Sivumäärä Aika	Eero Putkinen Moottorikäytön rasitustestilaite 48 sivua 13.2.2009
Koulutusohjelma	tietotekniikka
Tutkinto	insinööri (AMK)
Ohjaaja Ohjaava opettaja	tiiminvetäjä Arto Nakari yliopettaja Antti Piironen

Tämän insinöörityön aihe oli suunnitella ja toteuttaa moottorikäytön rasitustestilaite. Tavoitteena oli työkalu, jolla Kone Oyj:n luotettavuuslaboratorio pystyy seuraamaan ja ylläpitämään tuotteidensa luotettavuutta.

Moottorikäytön rasitustestilaite rakennettiin testattavan laitteen lisäksi tietokoneesta, moottorista ja kuormakoneesta. Testattavaa käyttöä ja kuormakonetta on mahdollista ohjata sarjaliikennekomennoin, joten testiympäristö pyrittiin karsimaan kaikesta muusta hissielektroniikasta, jota normaalisti tarvittaisiin. Tällä tavalla myös keskitytään ainoastaan moottorikäytön testaamiseen, eikä muu hissielektroniikka rajoita tai häiritse testiä.

Ohjelmistona käytettiin ohjelmoitavaa uCon-konsoliohjelmaa, joka ohjaa sarjaliikennekomennoilla yhtäaikaisesti sekä testattavaa käyttöä että kuormakonetta. Testilaitteelle suunniteltiin testausohjelma palautettujen käyttöjen toiminnan testaamiseen. Siinä huomioitiin hissin toiminta ja pyrittiin mahdollisimman hissimäiseen käyttäytymiseen. Lisäksi testistä pyrittiin tekemään elektroniikan testaukseen ja laadunvalvontaan sopiva.

Testilaitteen toiminta testattiin erityisellä komentoketjulla, jolla vaihdettiin eri kuorma-arvoja rasittamaan moottorikäyttöä, ajettiin lyhyt ajo ja tallennettiin moottorivirta lokitiedostoon. Mittaustulosten todettiin olevan teorian mukaisia.

Projektista syntyi testilaite, johon voidaan tarvittaessa räätälöidä erityisiä testejä moottorikäytön tietyn toiminnallisuuden tai esimerkiksi oheislaitteen testaamiseksi.

Hakusanat	moottorikäyttö, uCon, kiihdytetty testaaminen, hissi

Helsinki Metropolia University of Applied Sciences **Abstract**

Author Title Number of Pages Date	Eero Putkinen A drive stress test device 48 13 February 2009
Degree Programme	Information Technology
Degree	Bachelor of Engineering
Instructor Supervisor	Team Leader, Arto Nakari Principal Lecturer, Antti Piironen

The purpose of this thesis was to design and engineer a drive stress test device. The goal was to create a tool which Kone Ltd's reliability laboratory can use to monitor and maintain the reliability of the products.

The drive stress test device consists of the drive, a computer, a motor and a load machine. The drive under test and the load machine can be controlled by serial communication. Therefore the testing environment was stripped of all excess elevator electronics that are normally needed. This also helps to concentrate testing the drive: the elevator electronics will not restrict or disturb the test.

Programmable console software uCon was used to simultaneously control both the drive and the load machine. A test program was devised to test the returned drives. The test focused on the behaviour of an elevator. Also the test was made to be suitable for testing the drive's electronics and to be fit for quality control.

The functionality of the test device was tested with a particular test script in which different load values were used to stress the drive. A short run was made and the motor current was recorded in a log file. The measurements were found to be in accordance with the theory.

The drive stress test device can also be programmed for testing certain functionalities or for example accessory devices.

Keywords	drive, uCon, accelerated testing, elevator

Sisällys

Lyhenteet, käsitteet ja määritelmät

Ø	halkaisija
DUT	(Device Under Test) testattava laite
enkooderi	elektroninen laite pyörimisnopeuden mittaamiseen
F	voima
GPIB	(General Purpose Interface Bus) IEEE-488 standardin mukainen tiedonsiirtoväylä
HALT	(Highly Accelerated Life Test) kiihdytetty elinkaaren testaaminen
HASS	(Highly Accelerated Stress Screening) kiihdytetty rasitusselonta
jerk	Kiihtyvyyden muutos
LMU	(Load Machine Unit) kuormakone
M	momentti
r	säde
resolveri	elektroninen laite pyörimiskulman mittaamiseen
RMS	(Root Mean Square) neliöllinen keskiarvo, tehollisarvo
takometri	pyörimisnopeusanturi
UI	(User Interface) käyttöliittymä

1 Johdanto

Tämän insinöörityön tarkoituksena oli suunnitella ja toteuttaa moottorikäyttöjä testaava laite. Rikkoontuneet moottorikäytöt eivät läpäise testiä ja vikakoodit sekä mittaukset auttavat selvittämään vian ja sen syyt tarkemmassa analyysissä.

Tässä dokumentissa perehdytään moottorikäytön rasitustestilaitteen toimintaan ja esittelyyn. Dokumentista voi myös olla apua oman testiohjelman tekemiseen testilaitteen käyttämällä ohjelmistolla. Lisäksi käydään läpi hissin ominaisuuksia, jotka on pyritty huomioimaan testilaitetta tehtäessä, sekä teoriaa elektroniikan testauksesta.

Moottorikäytön rasitustestilaitetta tehtäessä syntyi uusia ideoita, ja projektin lopputulosta myös analysoidaan ja pohditaan, miten testilaitetta voidaan jatkossa kehittää. Kehittämisessä pyritään yhä enemmän ja enemmän hissimäiseen käyttäytymiseen.

Testilaitteen tilaajana on kansainvälinen hissiyhtiö Kone Oyj, jonka tuotekehityksen luotettavuuslaboratorion käyttöön laite tulee. Luotettavuuslaboratorio tutkii maailmalta viallisina palautettuja tuotteita. Tuotteet analysoidaan ja viat selvitetään. Tulosten perusteella voidaan tehdä muutos tuotteeseen tai käytettyihin komponentteihin. Jos valmistuslinjassa epäillään olevan ongelmia, voidaan ottaa yhteyttä valmistajaan ja vaikuttaa nopeasti tuotantoprosessiin.

2 Tavoitteet

Testilaitteen tulee auttaa moottorikäytön analysoimisessa ja vikojen etsinnässä. Sopiva testi estää rikkoontuneita käyttöjä läpäisemästä testiä. Vikakoodi ja tilanne, jossa testi epäonnistuu, auttavat selvittämään vian syyn.

Testilaitteen käyttö pyritään yksinkertaistamaan siten, että melkein kuka tahansa pystyy testaamaan moottorikäytön. Paras testilaite olisi sellainen, että testaajan pitää vain kytkeä testattava laite (DUT) ja painaa napista testi päälle, jonka jälkeen kaikki tapahtuu automaattisesti. Testien ohjelmoiminen kuitenkin vaatii tutustumista testilaitteen ohjelmistoon.

Testin ei tulisi olla kovin pitkä. Nopea testi pitää testilinjan liikkeessä. Mahdollisuus kytkeä DUT helposti nopeuttaa testiä.

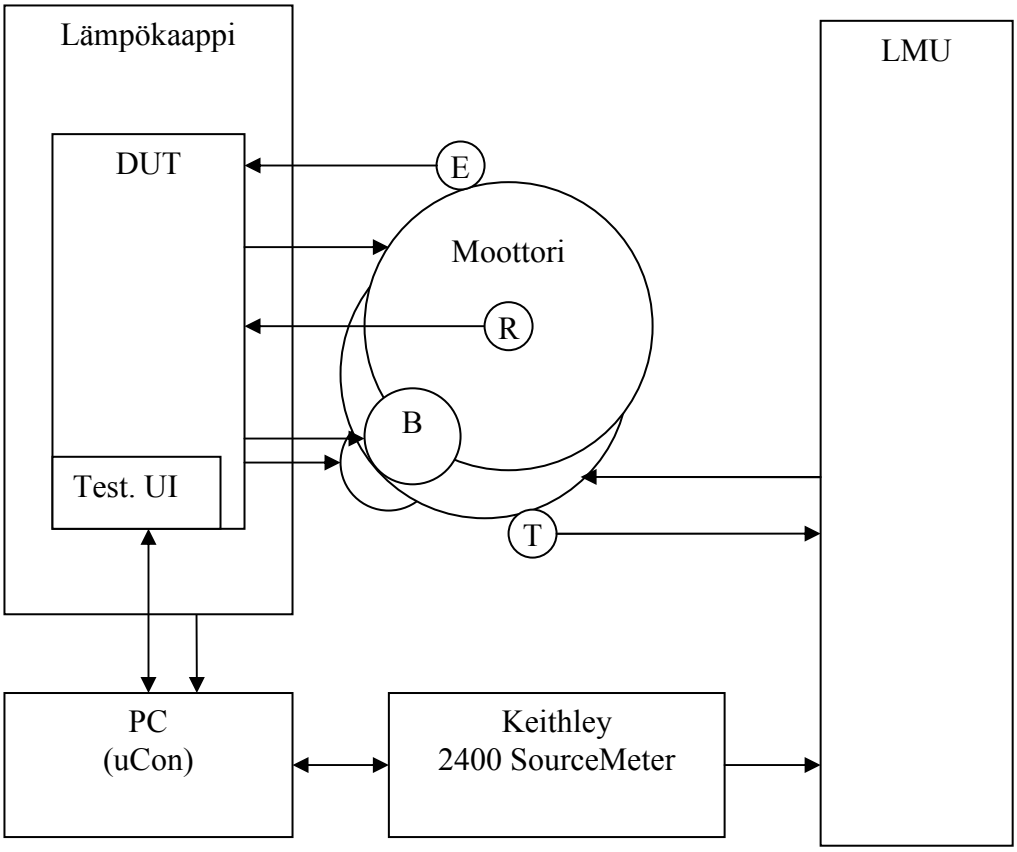
Oikea hissiympäristö ja siihen liittyvä elektroniikka pyritään karsimaan pois testitilanteesta käyttämällä DUT:n testikäyttöliittymää. Tällöin testilaite keskittyy vain moottorikäytön testaamiseen, eikä muu elektroniikka häiritse testiä. Testikäyttöliittymää voidaan ohjata sarjaliikennekomennoilla PC:ltä. Ilman hissiympäristöä testattavaa moottorikäyttöä ei voida pitää täysin toimivana, mutta testilaite seuloo suurimman osan vioista. Hissin käyttäytymistä pyritään kuitenkin simuloimaan sen mekaanisista osista koostuvien voimien osalta. Tämän simuloinnin hoitaa kuormakone (LMU), jota myös ohjataan PC:llä.

Moottorikäyttöä tulee ajaa lämpökammiossa, jotta lämpötilan muutoksista johtuvat viat löytyisivät helpommin. Moottorikäytön ympäristön suurimmaksi toimintalämpötilaksi on määritelty 40 ºC, mikä on myös lämpökaapin lämpötila. Kiihdytetyn elinkaaren testaamisen kannalta olisi hyvä yltää tarvittaessa 60 ºC:n lämpötilaan.

DUT:n testausasennolla voi olla vaikutusta muun muassa kontaktorien toimintaan. Testiasennon tulisi olla sama kuin laitteen oikea asennusasento. Käytön johdottaminen on kuitenkin helpompaa ja nopeampaa DUT:n ollessa vaaka-asennossa. Ideaalisin tilanne

onkin, että laite voidaan ensin johdottaa pöydällä, minkä jälkeen pöytä voidaan nostaa pystyasentoon testiä varten.

Tavoitteita miettiessä syntyi jo suunnitelma rasitustestilaitteen rakenteesta. Kuvassa 1 on kuvattu testilaitteen lohkokaavio. PC:llä ohjataan DUT:a ja LMU:ta samanaikaisesti. Moottoreita on kaksi, ja ne on yhdistetty toisiinsa roottoreiden akselista. DUT ohjaa toista staattoria ja jarruja. LMU taas muodostaa vastamomentin toiselle staattoreista.



Kuva 1. Moottorikäytön rasitustestilaitteen lohkokaavio. E on enkooderi, R on resolveri, B on jarru, T on takometri, LMU on kuormakone ja DUT on testattavana oleva moottorkäyttö.

3 Hissin rakenne ja toiminta

Nykyaikainen hissi on monimutkainen laitekokonaisuus. Kaikki komponentit voidaan jakaa neljään pääluokkaan:

- nostokoneisto
- sähköistys
- merkinantolaitteet
- ovet.

Nostokoneistoon lasketaan kaikki mekaaniset osat, jotka tarvitaan hissin liikuttamiseksi tasolta toiselle. Tähän kuuluvat koneisto, nostoköydet, vastapaino, tasausköydet, kori ja johteet. Nostokoneisto on rasitustestilaitteen kannalta olennaisin osa hissiä. Se sisältää kaikki liikkuvat osat, joita testilaitteen LMU:n pitäisi simuloida.

Sähköistykseen kuuluvat hissin virtalähteen sähkölaitteet, kuten moottorikäyttö. Merkinantolaitteisto koostuu matkustajalle tarkoitetusta elektroniikasta, jolla voi ohjata hissiä. Ovet estävät ihmisiltä pääsyn ulos liikkuvasta hissistä ja tasolta hissikuiluun. [1, s. 17.]

3.1 Nostokoneisto

3.1.1 Koneisto

Koneisto pyörittää vetopyörää. Se sijaitsee yleensä erillisessä konehuoneessa kuilun ylä- tai alapäässä. Nykyhissi voi olla myös konehuoneeton. Silloin moottori sijaitsee kuilun ylä- tai alapäässä seinällä. [1, s. 21.]

Jarrut kuuluvat koneistoon. Hissin jarrut on toteutettu sähkömagneetilla ja jousella. Jarru aukeaa sähköistämällä, ja jousi sulkee jarrun virran hävitessä [2, s. 14.]. Tämä tulee

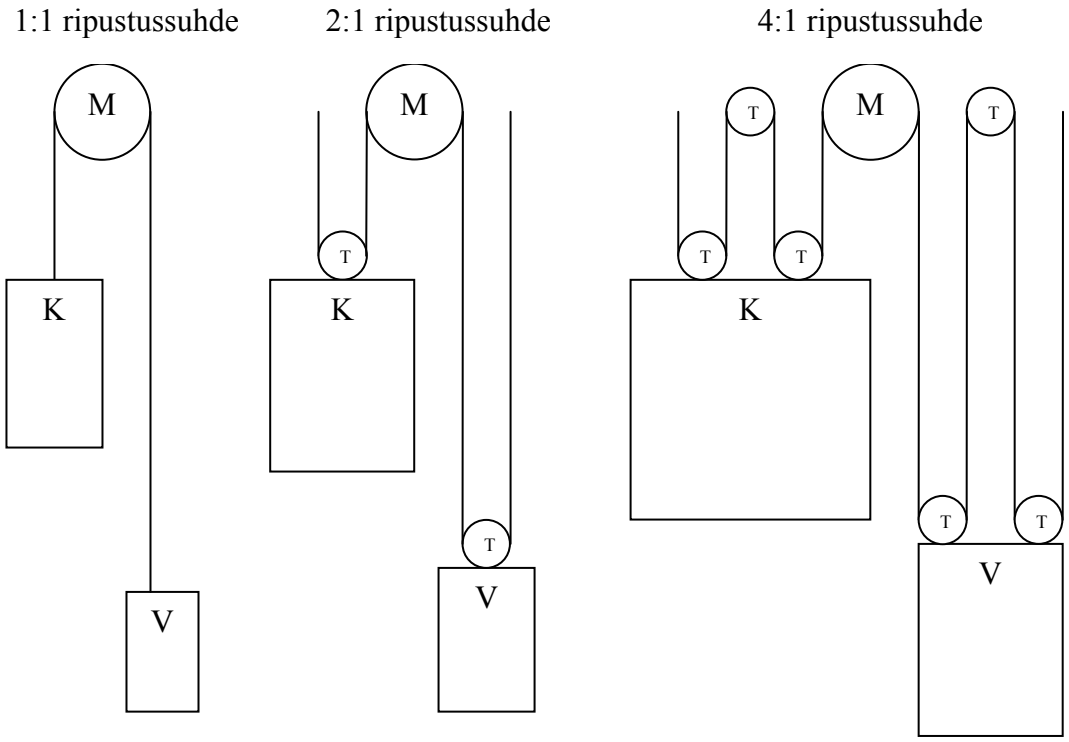
ottaa huomioon rasitustestilaitetta tehtäessä, koska DUT ohjaa moottorin jarruja. Virhetilanteessa DUT voi pudota tahdista, ja tällöin LMU pyörittää moottoria momenttiohjeella vapaasti. Täydestä vauhdista jarrujen lukitseminen kuluttaa turhaan jarruja ja aiheuttaa moottorikäytölle ylivirtatilanteen.

3.1.2 Nostoköydet

Nostoköydet liikuttavat koria. Ne on sijoitettu vetopyörän tarraavien urien yli. Nostoköydet voidaan ripustaa eri tavoilla riippuen halutusta hyötykuormasta. Eri vaihtoehtoja ripustukselle ovat 1:1, 2:1 ja 4:1.

Ripustussuhteessa 1:1 kori on köyden toisessa päässä ja vastapaino toisessa. 2:1 ripustussuhteessa köysien päät ovat kuilun katossa, ja köysi muodostaa lenkin korin ja vastapainon kohdalle. Lenkit menevät korin katolla ja vastapainossa sijaitsevien taittopyörien ympäri. Ripustus toimii kuin väkipyörä: nopeus puolittuu ja moottori joutuu pyörimään pidemmän matkan, mutta hissin nostokyky käytännössä kaksinkertaistuu.

Ripustussuhde 4:1 taas viedään vielä pidemmälle. Korin katolle ja vastapainoon ja kuilun kattoon laitetaan kaksi taittopyörää. Ripustussuhde 4:1 nelinkertaistaa ripustussuhde 1:1:n nostokyvyn ja sopii näin hyvin tavarahisseihin. Kuvassa 2 esitellään köysitysten toimintaa. [1, s 22; 3, s. 16.]



Kuva 2. Erilaisia hissien köysityksiä. M on moottorin vetopyörä, K on kori, V on vastapaino ja T on taittopyörä. [1, s 22; 3, s. 16]

Koska pelkkä kitka ei estä köysiä luistamasta vetopyörällä, luiston estämiseksi pitää käyttää esimerkiksi alileikkausta tai v-uraa. Lisäksi on erilaisia tapoja käyttää taittopyöriä hyväksi, jotta köyden tarttumispinta-ala vetopyörällä kasvaa. [1, s. 22; 2, s. 8.]

3.1.3 Vastapaino ja tasausköydet

Vastapainon tehtävä on tasoittaa korin painoa ja tuottaa riittävästi kitkaa vetopyörälle. Köysien luistamisvaara riippuu korin painojen ja vastapainon suhteesta. Vastapaino kumoaa korin painon, eikä koneisto koskaan joudu nostamaan sekä matkustajien että korin painoa. Vastapainon mitoittaminen on tärkeää. Edullisin vastapaino on korin painon lisäksi 40–50 % nimelliskuormasta. [1, s. 27; 2, s. 7.]

Hissin korkeuden kasvaessa nostoköysien paino alkaa vaikuttaa koneistoon. Esimerkiksi korin ollessa alimmalla tasolla ovat nostoköydet koko pituudeltaan lisäämässä korin puolen painoa. Tämä paino voidaan tasoittaa käyttämällä tasausköysiä. Tasausköysi kulkee korin pohjasta vastapainon pohjaan muodostaen yhtenäisen lenkin nostoköysien kanssa. Tasausköysinä käytetään yleensä ketjuja. [1, s. 24.]

3.1.4 Kori ja johteet

Kori on matkustajille hissin tutuin osa. Se kuljettaa matkustajat ja muun kuorman tasolta toiselle. Se pitää matkustajat eristettynä kuilusta ovilla ja näin suojaa matkustajaa. Koriin ei saa yleensä helposti lastattua ylikuormaa, koska ne suunnitellaan sen verran pieniksi, ettei ylikuormaa normaalikäytössä mahdu. Lisäksi vaaka pitää huolen, ettei hissi ylikuormitu.

Johteet ovat kuilussa pystysuoraan kulkevia kiskoja, joita pitkin kori kulkee. Ne on kiinnitetty tiukasti kuilun sivuseinämiin. Johteet vaikuttavat huomattavasti hissin liikkuvuuteen kuilussa. Jos johteet ovat vinossa, aiheuttaa se ajossa lisärasitusta koneistolle ja epämukavuutta koriin.

Johteet ovat myös tärkeä turvallisuustekijä. Hississä on turvalaitteena tarrain, joka ottaa kiinni johteista ja pysäyttää hissin nopeuden kasvaessa liian suureksi. [1, s. 25.]

3.2 Sähköistys

Sähköistys sisältää hissiä ohjaavan elektroniikan. Hissin ohjausjärjestelmä eli kontrolleri hallitsee käytännössä hissin toiminnan. Se ottaa vastaa signaalit merkinantolaitteilta ja ohjaa hissiä niiden mukaan sekä tarkkailee korin sijaintia, kulkusuuntaa ja ovijärjestelmää. Kontrolleri myös hallitsee käyttöjärjestelmää, joka ohjaa hissin moottoria ja syöttää sille virtaa. Testattava moottorikäyttö on osa sähköjärjestelmää. Käyttöjärjestelmä sijaitsee yleensä konehuoneessa.

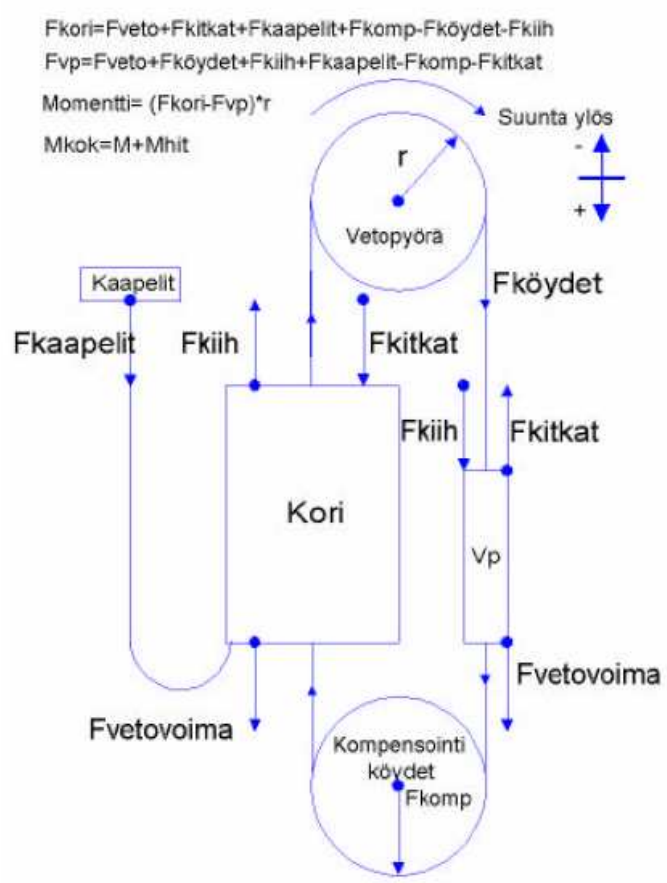
Kaapelointi siirtää informaatiota ja sähkötehoa hissin sähkölaitteille. Korin liikkuvuuden takia korin kaapelointi ei voi olla kiinteää. Korikaapeli roikkuu vapaana korin pohjassa. Hissin ollessa alimmassa kerroksessa korikaapeli roikkuu koko pituudeltaan kuilussa, kun taas hissin ollessa ylimmässä kerroksessa korikaapeli roikkuu kuilun puolivälissä.

Korikaapeleiden paino on vaikea tasoittaa. Tästä aiheutuu, että raskain tilanne hissille syntyy, kun täysi kori joudutaan siirtämään toiseksi ylimmästä kerroksesta ylimpään kerrokseen. Korin, kuorman ja tasausköysien lisäksi on nostettavana korikaapeli. [1, s. 35–39.]

3.3 Nostokoneiston mallinnus ja momenttikäyrä

Käytännössä siis hissin on raskainta liikkua kun kori on täynnä tai tyhjillään. Tämä johtuu siitä, että kun kori on tyhjä, menee moottorin voima koria raskaamman vastapainon liikuttamiseen, ja kun kori ollessa täynnä se on vastapainoa raskaampi. Eniten virtaa siis kuluu lähdöissä ja pysähdyksissä. Raskas kori vaatii paljon voimaa lähteäkseen alhaalta ylöspäin ja pysähtyäkseen ylhäältä alaspäin tullessa. Silloin raskas suunta on ylöspäin ja kevyt suunta alaspäin. Kevyt kori taas vaatii paljon voimaa lähteäkseen ylhäältä alaspäin ja pysähtyäkseen alhaalta ylöspäin mennessä. Silloin raskas suunta on alaspäin ja kevyt suunta ylöspäin. [3, s. 21.]

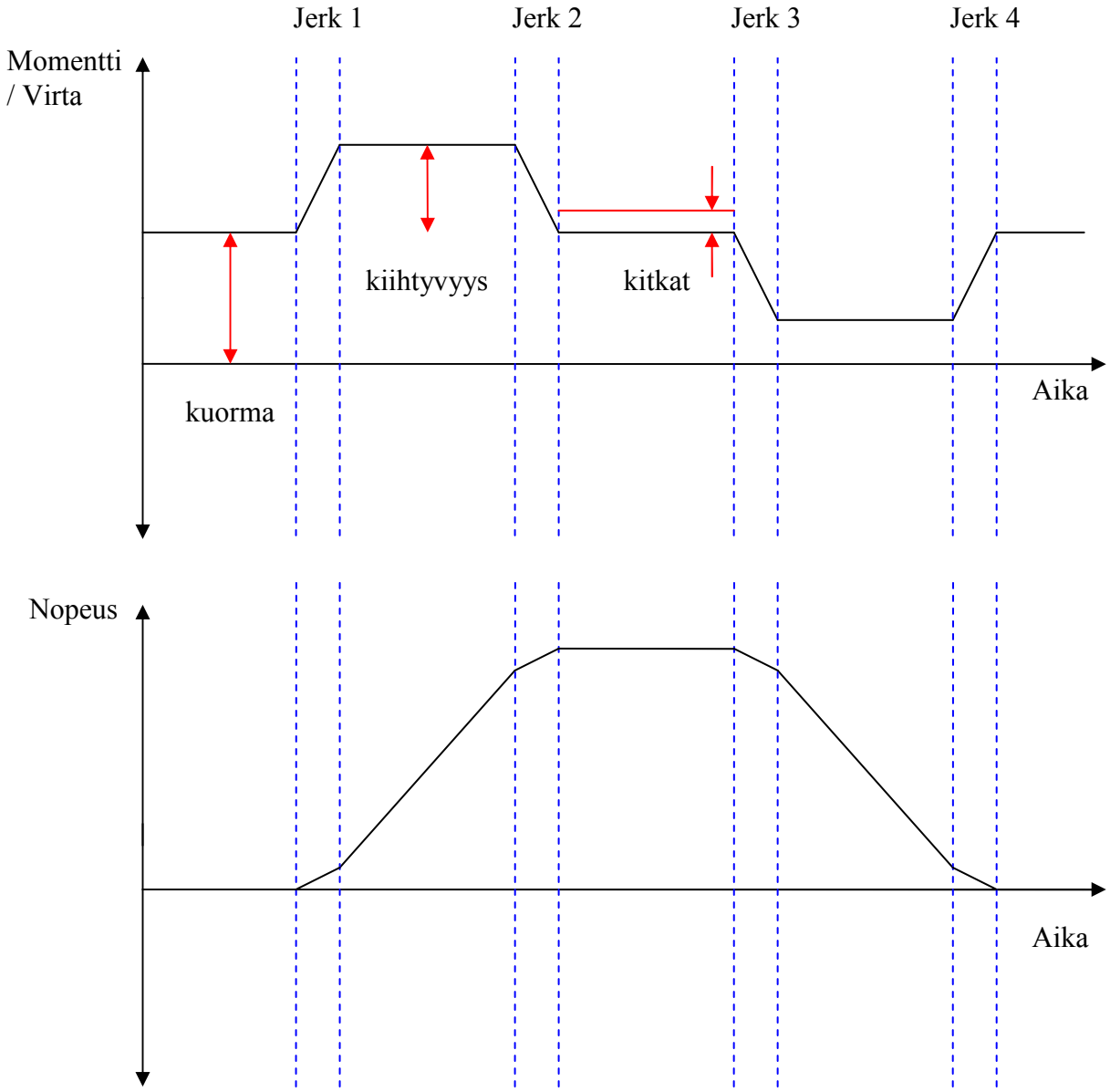
Nostokoneisto sisältää siis kaikki voimat, jotka moottorikäytön rasitustestilaitteen LMU simuloi. Kuvassa 3 nähdään nostokoneiston voimat ja se, kuinka ne vaikuttavat ja muodostavat momentin.



Kuva 3. Käytännöllinen hissimalli ja voimavaikutukset. Nuoli kertoo voiman suunnan [3, s. 20]

Punnitusjärjestelmä tarkkailee korin kuormaa. Kun paino saavuttaa 80 % nimellispainosta, katsotaan korin olevan täynnä, eikä hissi ota lisää matkustajia kyytiin. Kun kuormaa on 110 %, hissi ei lähde liikkeelle ennen kuin kuormaa on vähennetty. Rasitustestin raskain käytetty kuorma on siis 110 % kuormaa ja riippuen siitä mihin suuntaan hissi liikkuu, valitaan aina raskas suunta.

Moottorikäytön syöttämä virta moottoriin muodostaa vetopyörään momentin. Kuvassa 4 on kuvattu, miltä hissin momenttikäyrä ja virran kulutus näyttävät hissin liikkuessa.



Kuva 4. Hissin momentti- ja nopeuskäyrä

Kun momentti integroidaan, saadaan hissin nopeus. Kuorman muuttuminen näkyy suoraan momenttikäyrän nousemisena tai laskemisena y-akselilla. Kitkat näkyvät momenttikäyrässä hissin liikkuessa.

4 Luotettavuus ja elektroniikan testaus

4.1 Luotettavuus

Paineet globaaleilla markkinoilla saavat yritykset panostamaan laatuun entistä enemmän. Kuluttajat pyrkivät yleensä valitsemaan tuotteen, joka toimii parhaiten ja maksaa vähiten.

Luotettavuus on todennäköisyys, jolla tuote hajoaa tai sen toiminta heikkenee tietyn käyttöajan kuluessa. Luotettavuus on ajan funktio. Rajattu käyttöaika määritellään tuotteelle esimerkiksi takuuajaksi tai elinajaksi. [4, s. 10, 237.]

4.2 Kiihdytetty testaaminen

Tuotteen täydellisen luotettavuuden selvittäminen vaatii testejä, joiden tekeminen voi kestää vuosia riippuen tuotteen suunnitellusta eliniästä. Testaamista onkin pyritty nopeuttamaan eri tavoilla. Pääasiassa pyritään lisäämään laitteen vikaantumisen todennäköisyyttä kiihdytetyssä testissä.

Ylirasittaminen (overstressing) on yleisin kiihdytetty testimenetelmä. Siinä pyritään testaamaan DUT normaalikäyttöä ylittävillä rasitteilla. Nämä rasitteet syntyvät yleensä ympäristöstä tai sähköisistä, mekaanisista tai kemiallisista virikkeistä. Näitä ovat esimerkiksi lämpötila ja sen muutokset, kosteus, säteily, yli- ja alijännite, virta ja tärinä.

Laadullinen testi on yleensä määrätty tuotteen suunnittelu- ja kehitysvaiheessa. Tuote testataan useaan kertaan ja toistuvat vikaantumiset pyritään korjaamaan mahdollisimman nopeasti. Testit selvittävät laitteen keston esimerkiksi lämpötilan vaihteluissa tai värinätestissä.

Tuotteen heikot lenkit pyritään poistamaan. Parannukset tuotteeseen parantavat täten myös laatua jatkossa. Tällaista menetelmää kutsutaan kiihdytetyksi elinkaaren testaamiseksi (HALT). HALT-testin edut ovat testin lyhyt kesto sekä suuri määrä hyödyllistä informaatiota, joka saadaan tuotteen suunnittelijalle sen parantamiseksi. [4, s. 237–238; 5, s. 123–125.]

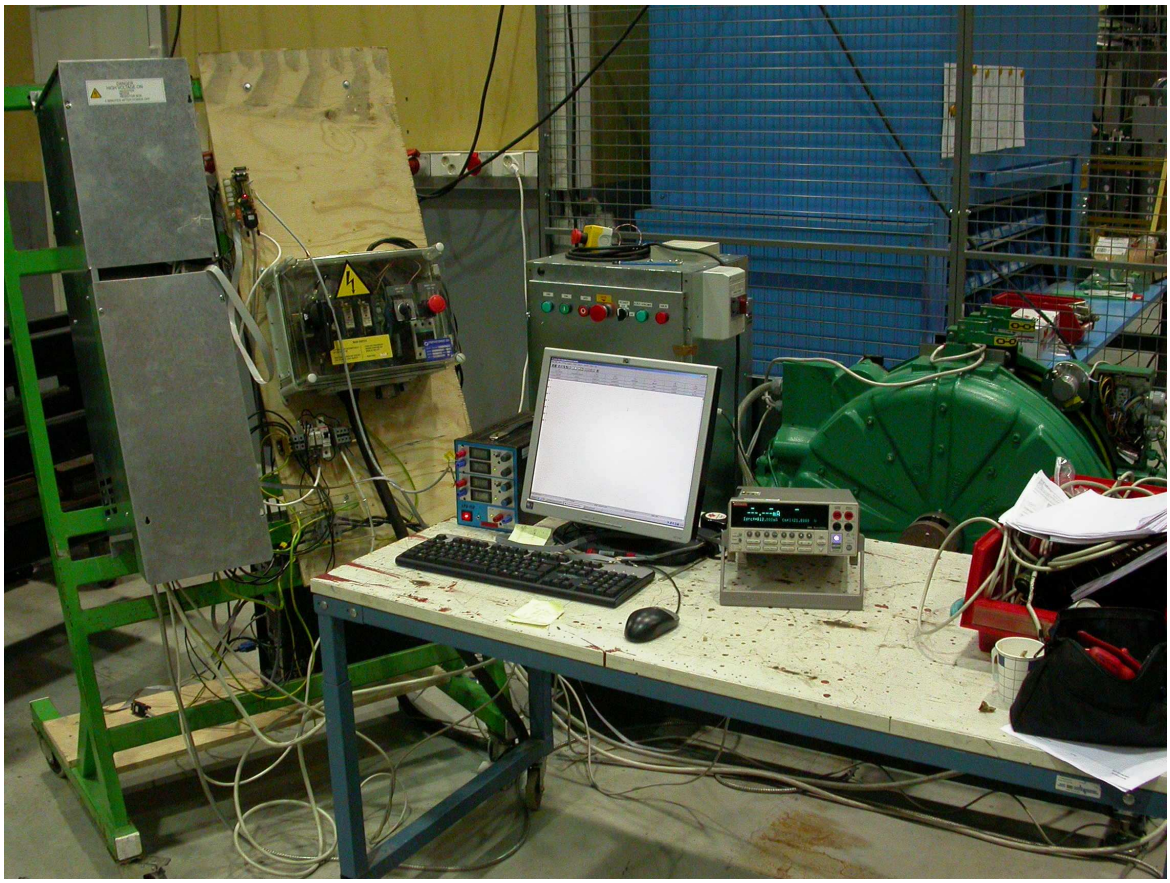
Tuotannon jälkeistä laadunvarmentamista varten on kehitetty voimakkaasti kiihdytetty rasitusseulonta (HASS). Tämä testi voidaan tehdä esimerkiksi jokaiselle valmistetulle tuotteelle tuotantolinjan lopuksi. Kuten nimi vihjaa, testi on sen verran rasittava, että tuotannon vialliset tuotteet seuloutuvat pois eivätkä pääse kentälle. HASS-testi ei kuitenkaan saa rikkoa tai alentaa tuotteen suorituskykyä. [6, s. 97–99.]

Moottorikäytön rasitustestilaite on tarkoitus suunnitella toimimaan HASS-testiä rasittavammin. Testilaitteella voidaan kuitenkin tehdä sekä HALT- että HASS-testi. Testin rasitustasoa voidaan säätää tarvittaessa millaiseksi vain.

5 Testilaitteen esittely

5.1 Yleiskuvaus

Moottorikäytön rasitustestilaite koostuu moottorista, LMU:sta, tietokoneesta, Keithleyn Sourcemeteristä ja DUT:sta. Kuvassa 5 nähdään ensimmäinen kehitysversio moottorikäytön rasitustestilaitteesta.



Kuva 5. Moottorikäytön rasitustestilaite.

5.2 Moottori

Testilaitteen moottoriksi on valittu kaksi MX10-moottoria. Moottorit ovat toisistaan kiinni akselistaan. Toista staattoria ajetaan DUT:lla ja toista LMU:lla. Jarrupyörän ke-

hälle on asennettu enkooderi ja akselille resolveri moottorin liikkeen määrittelemiseksi DUT:lle. DUT ohjaa moottorin jarruja.

DUT on suunniteltu käytettäväksi eri moottorilla, mutta se pystyy ajamaan myös MX10-moottoria.

Taulukoissa 1 ja 2 nähdään moottorin ja enkooderin tekniset tiedot.

Taulukko 1. MX10 -moottorin tekniset tiedot

Koneisto	MX 10
Teho	5.6 kW
Nimellisjännite	280 V
Nimellisvirta	16 A
Pyörimisnopeus	79.6 rpm
Jarrupyörän halkaisija	720 mm
Vetopyörän halkaisija	480 mm
Suurin kuorma	1000 kg
Ripustussuhde	2:1
Suurin korin nopeus	1.6 m/s

Taulukko 2. Enkooderin tekniset tiedot

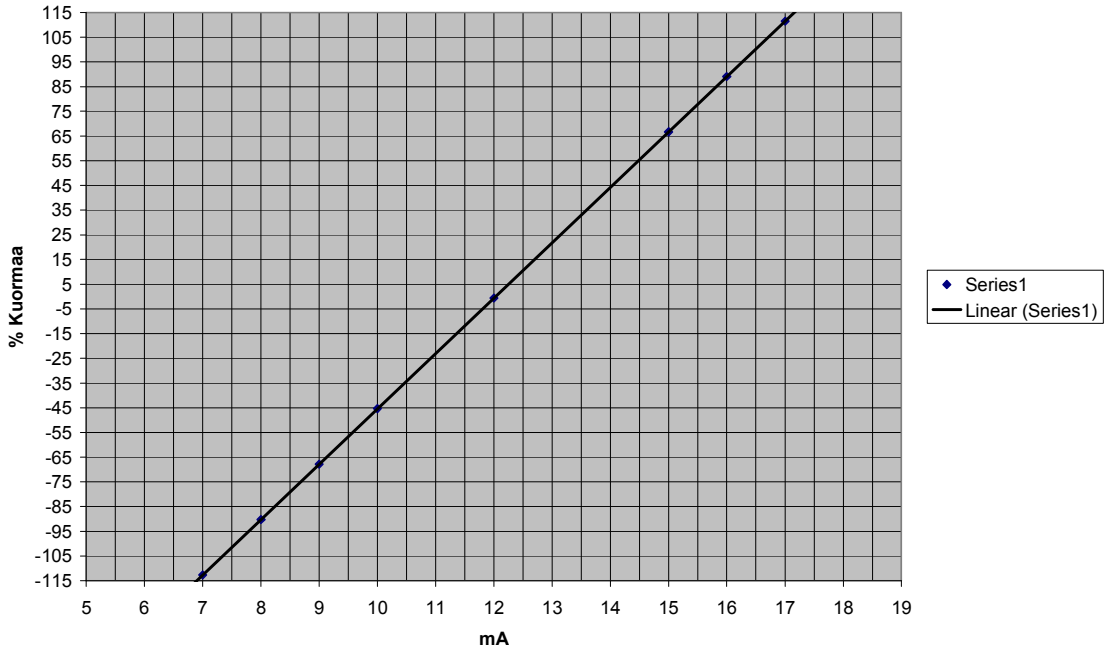
Pulssiluku	1024 pulssia / kierros
Kumipyörän halkaisija	37.3 mm

5.3 Kuormakone

Kuormakone ohjaa moottorin toista staattoria. LMU on valittu siten, että se sopii tyypillään käytettäväksi yhdessä moottorin kanssa. LMU synnyttää moottorille tiettyä kuormaa vastaavan vastamomentin ohjauksen mukaan.

LMU:n ohjaus tapahtuu virtaohjeella 4…20 mA, joka on skaalattu vastaamaan lineaarisesti -115…+115 %:n kuormaa. On huomioitava, että hissiä ei oikeasti ole olemassa ja että liikkumissuunta ylös- ja alaspäin on määrätty vain moottorin pyörimissuunnalla. Kuorman etumerkki kertoo suunnan, johon on raskasta lähteä. Luvun 3.3 mukaan on skaalattu, että negatiivinen kuorma tarkoittaa lähdön olevan raskasta alaspäin. Positiivinen kuoma kertoo lähdön olevan raskasta ylöspäin mentäessä.

LMU:lle kokeiltiin antaa eri virtaohjeita SourceMeter-mittarilla. Tuloksista saatiin kuvaaja (Kuva 6), josta näkee virran vastaavuuden kuorman kanssa. Kuvaa voidaan käyttää hyväksi ohjelmoitaessa virtaohjetta Keithley 2400 SourceMeterille.



Kuva 6. Virtaohjeen ja kuormaohjeen vastaavuus

Tilanteessa, jossa kuormaa ei tarvita ja hissi on pysähdyksissä, tulisi kuormaohjeen olla 0 % eli virtaohje 12 mA. Jos virtaohjeen laittaa pois päältä, säätyy kuorma alle -115 %:n ja LMU menee virhetilaan. Testin ohjelmassa pitää siis huomioida, että LMU:lla tulee jatkuvasti olla jokin virtaohje kuormakoneen ollessa päällä.

Kuormakoneen tarvitseman virtaohjeen voi toteuttaa monella eri tavalla. Suurin haaste on virtaohjeen synkronoiminen ajo-ohjeen kanssa.

Nopeuden derivoiminen valmiilla laitteella

Kuormaohje on aikaisemmin tehty säätövastuksella, jolla säädetään kuorman suunta ja taso. Sen jälkeen erikseen suunniteltu piirikortti generoi kiihtyvyyden perusteella hitausmomentin. Tämä on kuitenkin aiheuttanut ongelmia, kuten että momenttiohje tulee viiveellä lähtötilanteessa ja että häiriöt heijastuvat momenttiohjeeseen. Lisäksi kuorman on voinut säätää vain toiseen suuntaan kerrallaan. Derivointilaitetta voidaan käyttää tarvittaessa, mutta se rajoittaa raskaan suunnan ajamisen mahdolliseksi vain jompaankumpaan suuntaan.

Kuormaohjeen saaminen sulautetulta kortilta

Kuormakoneen ohjaamiseen voitaisiin käyttää laitetta, joka generoisi kuormaohjeen takaisinkytkentäarvona DUT:n saamien ohjeiden mukaan. Lisäksi laiteessa olisi optinen väylä, joka sopisi hyvin LMU:n kanssa ja olisi näin häiriötön. DUT:n ja laitteen synkronoiminen on kuitenkin vaikeaa, koska molemmat laitteet tarvitsevat ohjeen samanaikaisesti. Laite myös vaatii erityisen ohjelman sulautetulle kortille. Sulautettua korttia voidaan käyttää DUT:n ohjaamiseen, mutta se vaatii DUT:n purkamista ja sulautetun kortin lisäämistä moottorikäyttöön. Tämä estää DUT:n testaamisen kentältä tulleena koskemattomana pakettina ja kortin asentaminen lisää huomattavasti testausaikaa.

Labview

Labview on graafinen ohjelmointiympäristö, jonka on kehittänyt National Instruments. Ohjelmointi mahdollistaa erilaisten mittausten helpon lisäämisen.

Labview-ohjelmistolla voidaan ohjata DUT:ta sarjaliikennekomennoin ja saada virtaohje kuormakoneelle ohjelmiston kanssa yhteensopivalta laitteelta. Lisäksi voidaan laskea hitausmomentti takaisinkytkennällä ja lisätä se kuormaohjeeseen. Käytettäväksi laitteistoksi soveltuu esimerkiksi National Instrumentin kehittelemä cDAQ-9172-mittauslaitteisto, johon on kytketty sopiva moduuli virtaohjeen generoimiseksi. Moduu-

leita voidaan myös lisätä, kun joudutaan esimerkiksi tekemään uusia mittauksia. Sopiva moduuli virtaohjeen generoimiseksi on NI 9265, joka on neljän kanavan 0…24 mA virtalähde, joka toimii päivitysnopeudella 100 Ks/s. [7.]

Kuitenkin rasitustestilaitteen ensimmäisen kehitysversion ohjaamisessa päädyttiin käyttämään uCon-ohjelmistoa sen yksinkertaisuuden takia. uCon-ohjelmistosta kerrotaan enemmän luvussa 5.5.

Keithley 2400 SourceMeter -mittari

Moottorikäytön rasitustestilaitteen kuormanohjaukseen valittiin Keithley 2400 SourceMeter, jolla voidaan toteuttaa tarvittava kuormaohje virtaohjeena sarjaliikennekäskyillä.

Keithley 2400 SourceMeter -mittari on ohjelmoitava mittari ja DC-virtalähde. SourceMeteriä voidaan ohjata GPIP-portin tai sarjaliikenneportin kautta. Koska DUT:a ohjataan sarjaliikennekomennoin, voidaan SourceMeteriä ohjata samalla ohjelmalla vaihtamalla välillä COM-porttia.

SourceMeterin komentamiseen käytetään samaa ohjelmaa kuin DUT:n ohjaamiseen. Tämä pitää testilaitteen ohjaamisen yksinkertaisena, kun useampia erityisiä ohjelmia ei tarvita.

SourceMeterin ohjauskomennot ovat yksinkertaisia. Taulukon 3 komennoilla saadaan kuormataulukon (kuva 6) mukaan 0 %:n kuormaohje LMU:lle. Tämän jälkeen kuormaohjetta voidaan vaihtaa :SOURCE:CURRENT -komennolla. [8.]

Taulukko 3. Esimerkki Keithleyn SourceMeter -mittarin komentamisesta

Sarjaliikennekomento	Toiminto
*RST	Alustetaan laite
:ROUTE:TERMINALS REAR	Valitaan takana olevat s/t portit käyttöön
:SOURCE:FUNCTION CURRENT	Valitaan syöttötoiminto ja virran syöttö
:SOURCE:CURRENT 0.012	Asetetaan virtasyötön arvoksi 12 mA
:OUTP:STAT 1	Laitetaan ulostulo päälle

Jatkossa kuormaohjeeseen tullaan lisäämään hissin hitausmomentti. Tämä onnistuu joko askeltamalla virtaohjetta tai käyttämällä sweep-toimintoa, jolla saadaan liukuva virtaohje alkuarvosta loppuarvoon.

5.4 Testikäyttöliittymä ja parametrien mitoittaminen

DUT:n ohjaamiseen käytetään erityistä testikäyttöliittymää. Testikäyttöliittymä mahdollistaa käytöllä ajamisen välittämättä hissiympäristöstä, eikä se tarvitse hissille muuten välttämättömiä signaaleita tai elektroniikkaa. Moottorikäytön testaaminen siis onnistuu ilman, että hissiympäristö häiritsee tai rajoittamatta testiä. Tällöin keskitytään vain itse moottorikäytön testaamiseen.

Testikäyttöliittymään kytkeydytään sarjaliikennettä pitkin. Testikäyttöliittymän avaamisen jälkeen sitä voidaan käyttää yksinkertaisilla aakkosnumeerisilla käskyillä. Testikäyttöliittymän kautta voidaan myös lukea ajon aikana arvoja, kuten esimerkiksi virta, nopeus ja laitteen lämpötila, jotka ovat hyödyllisiä tietoja DUT:n analysoinnissa.

Testikäyttöliittymä käyttää RAM-muistia, joten sen käyttäminen ei muuta vanhoja parametreja muistista. Aikaisemmin käytetyt parametrit ovat siis luettavissa ja voivat auttaa analysointiprosessissa.

Jotta moottoria voi ajaa, täytyy testikäyttöliittymän kautta asettaa moottorille sopivat parametrit, mitoittaa enkooderin pulssitaajuus suhteessa moottorin kierrokseen sekä hakea resolverin asennuskulma suhteessa moottorin asentoon.

Ensin pitää selvittää, montako pulssia enkooderi antaa yhdelle moottorin kierrokselle kaavalla 1:

$$Pulceno_{encoder} \times \left(\frac{Ø_{jarrupyörä}}{Ø_{encoder}} \right) = Pulceno_{motor}, \quad (1)$$

$Pulceno_{encoder}$ on enkooderin pulssiluku

$Ø_{jarrupyörä}$ on jarrupyörän halkaisija

$Ø_{encoder}$ on enkooderin kumipyörän halkaisija.

Kaavan 1 ja luvun 5.1 tiedoilla voidaan laskea yhden moottorikierroksen vastaavan 19 766 enkooderin pulssia ($Pulceno_{motor}$).

Lisäksi pitää selvittää, kuinka monta pulssia saadaan enkooderilta, kun hissin kori liikkuu 10 cm:n matkan. Luvun 3.1.2 mukaan vetopyörän köydet liikkuvat 20 cm, kun hissi liikkuu 10 cm, köysityksen ollessa 2:1. Tällöin saadaan kaava 2:

$$Pulceno_{motor} \times \left(\frac{0.2[m]}{Ø_{jarrupyörä}[m] \times \pi} \right) = Pulcenohissi_{10cm}, \quad (2)$$

$Pulceno_{motor}$ on moottorikierroksen pulssiluku enkooderilla

$0.2[m]$ on vetopyörän kulkema matka, kun hissi liikkuu 0,1 m

$Ø_{jarrupyörä}[m]$ on jarrupyörän halkaisija metreissä.

Kaavalla 2 saadaan laskettua hissin kulkeman 10 cm:n matkalle 2 621,5 pulssia enkooderilla ($Pulcenohissi_{10cm}$).

5.5 uCon-konsoliohjelmisto

UCon on osa MicroMonitor-ohjelmistoa. Se toimii hyvänä korvaajana Windowsin HyperTerminal-ohjelmistolle ja toimii Windows-käyttöjärjestelmässä.

MicroMonitor on tarkoitettu sulautettujen järjestelmien ohjaamiseen ja kehittämiseen. Ohjelmiston on kehittänyt Ed Sutter. MicroMonitor on myös OSI-hyväksytty avoimen lähdekoodin ohjelmisto. [9.]

5.5.1 Ohjelmoitavuus

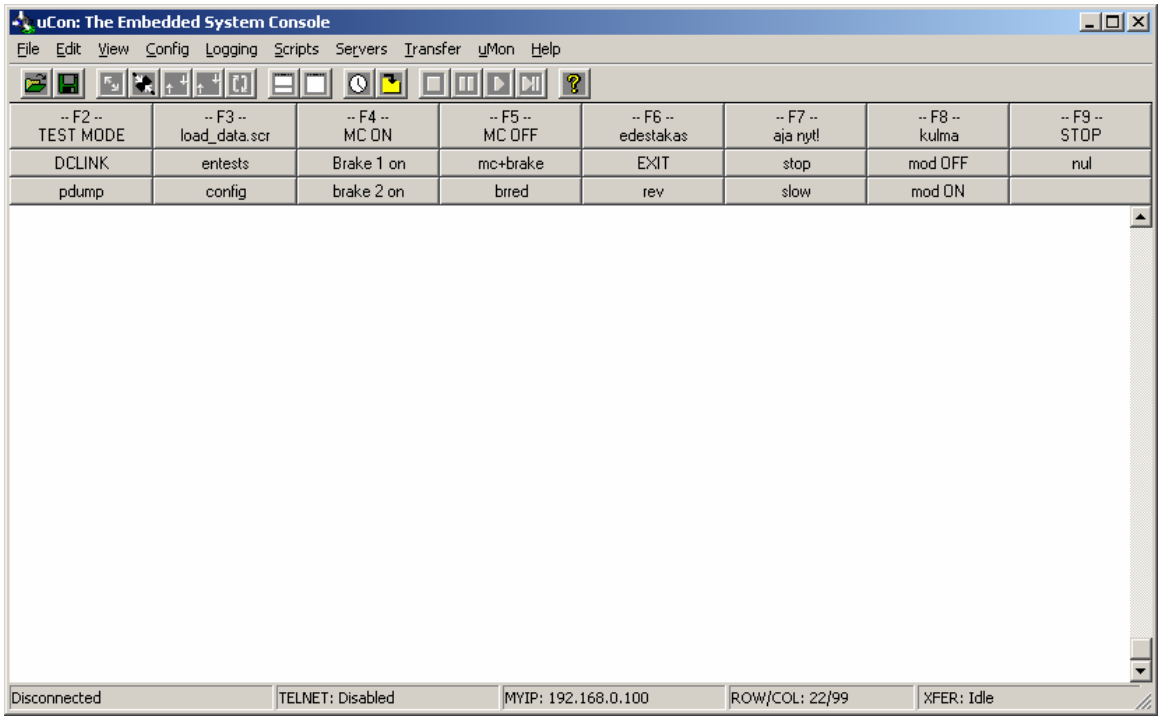
Ucon sopii testilaiteelle hyvin sen helpon ohjelmoitavuuden vuoksi: se on ohjelmoitavissa omalla ohjelmointikielellään. Liitteessä 1 on lista uConin ohjelmointikielen käskyistä. Käskyistä voi tehdä komentosarjoja, joilla saa luotua testilaitteelle haluamansa testin. Ohjelmointikieli mahdollistaa kaikki ohjelmoinnin perustoiminnat, kuten silmukat, ehdollistamiset, vertailut, aliohjelmat, hypyt ja tiedostojen muokkauksen.

Tiedoston muokkaus mahdollistaa testitulosten tallentamisen raporttiin. Raportti koostuu lähinnä vain testikäyttöliittymän mittaustuloksista ja mahdollisista vikakoodeista. Raporttiin tallentuvat myös aika, jolloin DUT on testattu, ja testin kesto. Jatkossa raporttia voidaan käyttää laadunvalvonnassa tilastojen tekemiseen.

Hallittavaa laitetta voidaan myös vaihtaa kesken komentosarjan. CONNECT-komennolla voidaan vaihtaa sarjaliikenneporttia, johon halutaan lähettää komentoja. Tätä käytetään hyväksi, kun hallitaan kuormakonetta. Kun kuorma halutaan käynnistää tai muuttaa sitä, vaihdetaan sarjaliikenneyhteys Keithleyn SourceMeter -mittarille ja annetaan kuormaohje. Sen jälkeen vaihdetaan takaisin DUT:n ohjaukseen ja jatketaan testiä.

5.5.2 Käyttöliittymä

Käyttöliittymä saadaan kommunikoimaan DIALOG-komennolla käyttäjän kanssa. Se saadaan tarjoamaan ohjeita ja kysymään esimerkiksi, minkä testin käyttäjä haluaa tehdä. Myös erilaiset äänimerkit ovat mahdollisia. Tehdyt komentosarjat voidaan ohjelmoida käynnistymään käyttöliittymän napista painamalla. Kuvassa 7 nähdään uCon-konsoliohjelman käyttöliittymä toimintanappeineen.



Kuva 7. uCon-konsolin käyttöliittymä

5.6 Testin suunnittelu

Rasitustestilaitteelle voidaan tehdä monenlaisia testejä. Esimerkiksi voidaan keskittyä vain kontaktorien tai jarrujen testaamiseen. Mahdollisuus ohjelmoida omia erityisiä testejä sopii hyvin luotettavuuslaboratorion tarpeisiin.

Kentältä palautetuille moottorikäytöille pitää kuitenkin suunnitella testi, joka testaa tuotteen toiminnallisuuden. Testin pitäisi olla sopivan raskas, jotta mahdollinen vika ilmenisi. Luvussa 4.2 esitetty tuotannon jälkeinen HASS-testi ei välttämättä seulo pois kaikkia vikoja, joten rasitustestilaitteen tekemän testin pitäisi olla HASS-testiä hieman rasittavampi. Liitteessä 2 on kommentoitu esimerkki testiohjelmasta.

5.7 Lämpökaappi

Tätä raporttia kirjoittaessa lämpökaappia ei ollut vielä rakennettu rasitustestilaitteelle. Testilaitteen melko helppo liikuteltavuus mahdollistaa kuitenkin sen, että DUT on helppo sijoittaa esimerkiksi erilliseen sääkaappiin testin ajaksi. Sääkaapissa voi muun muassa muuttaa kosteusarvoja, ja lämpötilaa voi hallita syklisesti välillä -10... +60 ºC.

Rasitustestilaitteen kiinteässä lämpökaapissa tulee olemaan automaattinen ohjaus, joka on ohjelmoitu pitämään lämpötila kahden pisteen välissä. Lämpötilan saavuttaessa ylemmän pisteen lämmitys lopetetaan, ja lämpötilan laskiessa alemmalle pisteelle lämmitystä jatketaan. Näin saadaan lämpötila pysymään suurin piirtein halutussa lämpötilassa. Lämpökaapin lämpötilaa tulisi pystyä säätämään 40... 60 ºC:n lämpöiseksi.

Lämpökaapin seinien tulisi olla läpinäkyviä, jotta käyttöä voi testin aikana halutessaan tarkkailla. Lämpökaappi myös suojaa käyttäjää, ettei hän pääse vahingossa koskemaan käyttöä virran ollessa kytkettynä.

5.8 Turvallisuus ja suojaukset

Turvallisuus on pyritty huomioimaan testilaitetta suunniteltaessa. Kytkennät ja maadoitukset on varmistettu, ja laitteet toimivat vikavirtasuojatussa ympäristössä.

Lämpökaappi estää käyttäjää vahingossa koskemasta käyttöä virran ollessa kytkettnä. Lämpökaapin voi myös virittää kytkimellä, ettei testiä saa toimimaan kaapin ollessa auki.

Virhetilanteessa ohjelma pyrkii pysäyttämään moottorin hallitusti. Tämä on parempi vaihtoehto kuin jarruttaa äkillisesti täydestä vauhdista, mikä kuluttaisi turhaan jarruja ja aiheuttaisi DUT:lle ylivirtatilanteen. Hätäpysäytys-nappi löytyy kuitenkin testilaitteen pysäyttämiseksi.

Mekaanisesti liikkuvat osat on pyritty suojaamaan siten, ettei testilaitteen käyttäjä pääse takertumaan mihinkään. Moottori on suojattu metalliverkolla, joka estää pääsemästä siihen käsiksi.

Testilaitteen suojaukset on pyritty tekemään huolellisesti, koska ne voivat vaikuttaa esimerkiksi kuorman virtaohjeen laatuun. Lisäksi testilaitetta käytetään laboratorio-oloissa, joiden tulisi olla mahdollisimman häiriöttömät muillekin testilaitteille. Moottorikaapelit, resolverikaapeli sekä enkooderikaapeli on maadoitettu molemmista päistä. Virtaohje annetaan vaipalla suojattua kaapelia pitkin. Moottorin häkki toimii Faradayn häkin tapaan suojaten ympäristöä sähkömagneettiselta säteilyltä. Myös kuormakone ja sen kaikki elektroniikka on suljettu metallikaappiin. DUT voidaan testata sen kuoret normaalisti kiinnitettyinä.

6 Rasitustestilaitteen kehittäminen jatkossa

Hitausmomentin vaikutusten lisääminen kuormaan

Hitausmomentin vaikutus kuormassa tullaan jatkossa lisäämään LMU:n kuormaohjeeseen. Koska testin kiihdytykset ja hidastukset on tarkoin määritetty, voidaan hitausmomentin vaikutus laskea ja lisätä kuormaohjeeseen.

Kuilun ominaisuuksien lisääminen momenttiohjeeseen

Kuilun ominaisuudet, kuten johteiden tiukkuus, voivat aiheuttaa hissille lisäkuormaa. Eri ominaisuuksien ohjelmoiminen kuormaohjeeseen voisi hyödyntää käytön testaamista. Käytännössä tämä tarkoittaisi momentin lisäämistä hieman hissin liikkuessa simuloidun tiukan kohdan yli.

Moottorityypin vaihtaminen

Käytöllä voidaan ajaa eri moottorityyppejä. On mahdollista, että DUT:n vika esiintyy vain tietyllä moottorityypillä. Jos testilaitteeseen voidaan vaihtaa eri moottorityyppejä, voitaisiin valita moottori, jolla vikaantunutta DUT:ta on käytetty kentällä, ja testata se samalla kokoonpanolla. Testiohjelma kuitenkin pitää muuttaa sopivaksi kaikille moottoreille parametrien osalta.

Hissisimulaattori

Hissisimulaattori on sähköisesti hissin kaltainen kokonaisuus, joka koostuu kaikesta hississä tarvittavasta elektroniikasta ja signaaleista, kuten kontrollerista, kuilun tasosignaaleista ja ovikäyttöjärjestelmästä. Mekaanisesti liikkuvat laitteet, kuten ovet, kori ja kuilu, on karsittu ja korvattu niitä simuloivilla aputoimilaitteilla.

Lisäämällä testilaitteeseen hissisimulaattori voitaisiin DUT testata perinpohjaisesti samalla laitteella. Testi kattaisi DUT:n toiminnallisuuden ja toimivuuden hissielektroniikan kanssa.

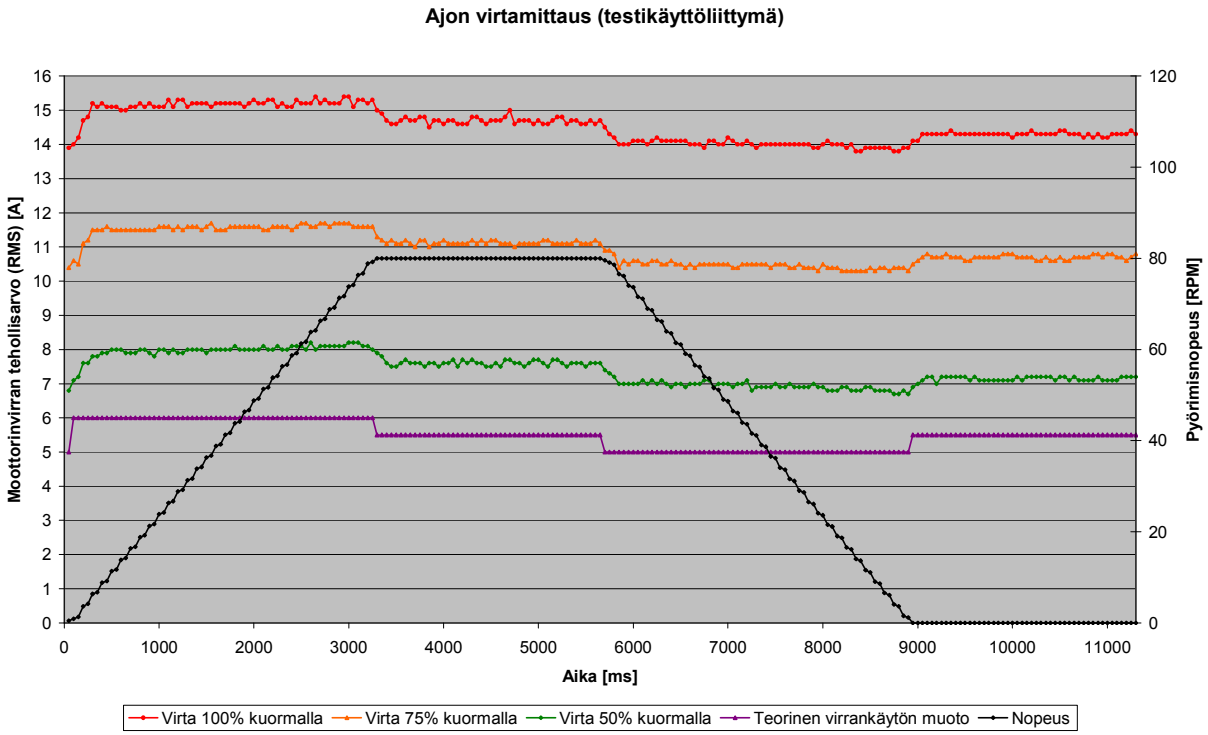
Testi voitaisiin jakaa kahteen osaan. Ensin suoritettaisiin hissisimulaattoritesti ja sitten rasitustesti. Jos laite läpäisee molemmat testit, se todennäköisesti toimisi myös hyvin kentällä.

Täydellinen testilaite analysoisi vielä vikakoodit ja virtamittaukset selvittäen laitteen vian, pitäisi tietokantaa kaikista testatuista laitteista ja tulostaisi valmiit raportit.

7 Mittaukset, testilaitteen toiminta ja päätelmät

7.1 Mittaukset

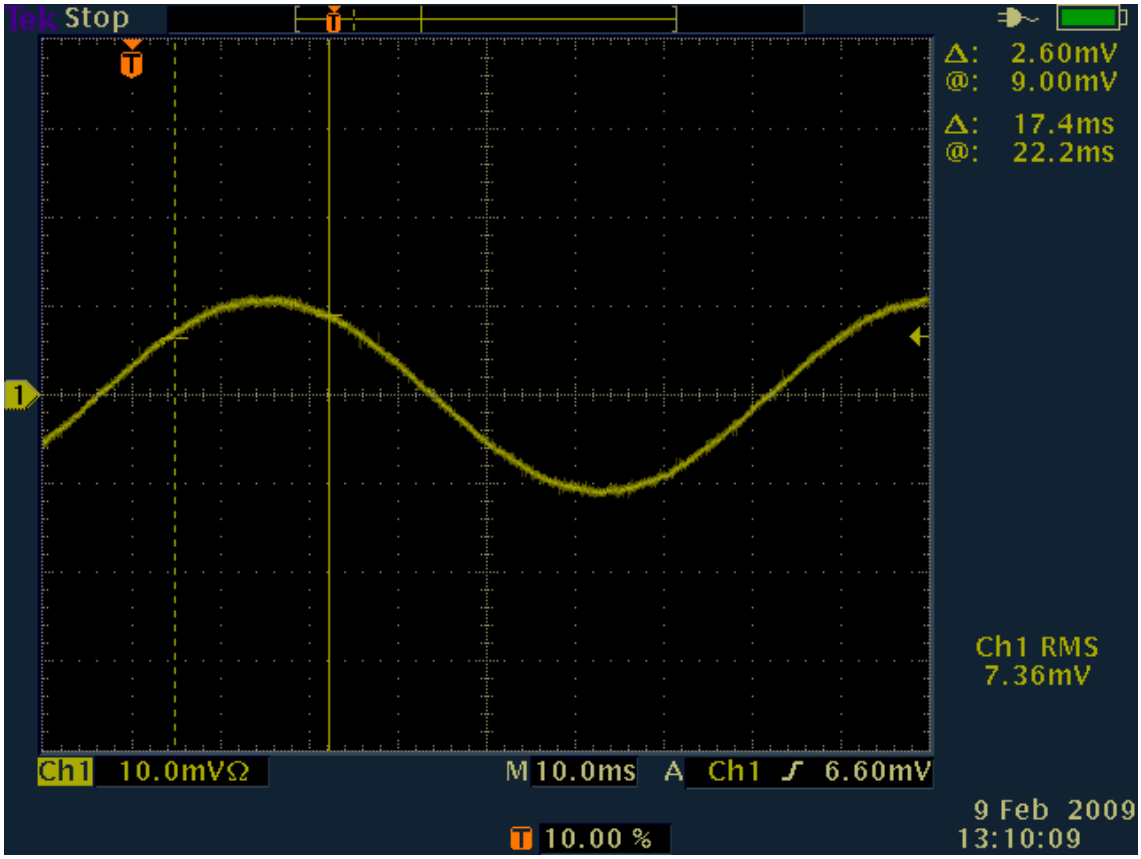
Testilaitteen toiminnan varmistamiseksi tehtiin ohjelma, jolla moottori kiihdytettiin huippuvauhtiin 80 kierrosta minuutissa, ajoi vähän aikaa vakionopeutta ja sitten hidasti kunnes pysähtyi. Kiihdytykselle ja hidastukselle määrättiin kolme sekuntia aikaa. Ajo ja mittaukset suoritettiin kuorman ollessa 100 %, 75 % ja 50 %. Ajon aikana testikäyttöliittymä ohjelmoitiin syöttämään kierrosnopeus ja moottorivirran arvot lokitiedostoon viidenkymmenen millisekunnin välein. Tuloksista saatiin kuvaaja (kuva 8), josta nähdään ajon aikaisen virrankäytön tehollisarvo.



Kuva 8. Virranmittaus testikäyttöliittymällä eri kuorma-arvoilla

Saadut tulokset ovat luvussa 3.3 esitetyn teorian mukaisia. Kiihtyessään raskaaseen suuntaan hissi vaatii enemmän virtaa kuin hidastaessaan. Hissin liikkuessa kitka näkyy virrankulutuksen korkeampana tasona verrattuna paikallaan pysyvän hissin virrankulutukseen.

Käyttöliittymästä saatu nimellisvirta tarkistettiin vielä mittaamalla oskilloskoopilla ja virtakahvalla syöttövirta moottorikaapelista. Mittaukset tehtiin moottorin pyöriessä tasaista nopeutta 80.0 kierrosta minuutissa 100 %:n kuormalla (kuva 9)



Kuva 9. Virran tehollisarvon (RMS) mittaus tasaisella nopeudella 80 rpm ja 100 %:n kuormalla.

Virtakahva oli kiinnitetty vahvistimeen, jonka asetusten mukaan oskilloskoopin näyttäessä 10 millivolttia virta on 20 ampeeria. Tästä saadaan kaava 3:

$$mittausRMS[mV] \times \frac{20}{10} = TehollisvirtaRMS[A], \qquad (3)$$

Kaavan mukaan virran tehollisarvo (RMS) kuvassa 8 on 7,36 x 2 = 14,72 A. Käyttöliittymän kautta saadut tulokset ovat siis todellisia.

7.2 Päätelmät

Projektin tavoitteet täyttyivät, ja työn tuloksena syntyi hyödyllinen moottorikäytön rasitustestilaite. Testilaitteen mittaukset osoittautuivat teorian mukaisiksi, ja testilaite käyttäytyi toivotusti. Se toimii hyvin luotettavuuslaboratorion laadunvalvonnan työkaluna.

DUT ja kuormakone ovat moottorikäytön rasitustestilaitteella vapaasti hallittavissa. Testilaitteella voidaan tehdä lähes millaisia testejä tahansa, ja siten se sopii paremmin laadunvalvonnan käyttöön kuin tuotannossa käytetty tuotantotestilaite. Testilaitteesta voi olla myös apua moottorikäytön oheis- tai lisälaitteiden sekä moottorin testaamisessa.

Jos testikäyttöliittymä pidetään samankaltaisena ja uusien moottorikäyttöjen kanssa yhteensopivana, testilaite on helposti muutettavissa myös niiden testaamiseen soveltuvaksi.

Hissisimulaattorin puuttumista voidaan pitää sekä heikkoutena että vahvuutena. Vahvuus piilee siinä, että testilaitteella voidaan tehdä ajoja, joita oikealla hissillä ei pystytä tekemään. Testilaitetta kuitenkin pyritään kehittämään jatkossa vastaamaan enemmän oikean hissin käyttäytymistä.

Lähteet

1 Kone Elevators Training Center, Hissitekniikan perusteet. Hyvinkää: Kone Oyj, 2001.

2 Virkkala, Vilkko. Hissitekniikan perusasioita 2. painos. Helsinki: 1973.

3 Honkanen, Urho. Kuormitussimulaattori hissikäyttöjen testaamiseen. Insinöörityö. Helsingin ammattikorkeakoulu, 2003.

4 Yang, Guangbin. Life cycle reliability engineering. John Wiley & Sons, Inc.: 2007.

5 Porter, Alex. Accelerated testing and validation. Newnes: 2004.

6 Levin & Kalal. Improving Product Reliability. Wiley: 2003.

7 National Instruments, NI 9265 overview (WWW-dokumentti) <http://sine.ni.com/psp/app/doc/p/id/psp-172/lang/en> Päivitetty 10.02.2009. Luettu 10.02.2009.

8 Keithley 2400 SourceMeter manual (WWW-dokumentti) <http://www.keithley.com/data?asset=883>. Päivitetty 12.2000. Luettu 21.10.2008.

9 MicroMonitor. (WWW-dokumentti) Micrcross. <http://microcross.com/html/micromonitor.html>. Päivitetty 8.9.2008. Luettu 21.10.2008.

SCRIPTING
uCon supports the ability to run scripts. This means that you can create command files that allow you to programmatically automate interaction with the device uCon is connected to. The Scripts menu item supports the ability to create, edit, run, halt, pause, resume and even single-step a script. The script can be run full speed or at a debug rate, where each non-comment line in the script must be approved by the user through a simple dialog box interaction. A script can be run from the menu item Scripts->Run or through a function key or button whose "script" checkbox has been set.
QUICK JUMPS:
BEEP BMPVIEW CD CLS
COMPORT CONNECT DIALOG ECHO
EXIT FILE FKEY FSEND
GOSUB GOTO IF LOG
MONCMD NEWMON RECV REGEX
RETURN RUN SEND SET
SLEEP SYSTEM TCP TFTP
TIME TITLE XMODEM
CREATING A SCRIPT:
A script is simply a file, so any editor that allows you to create a non-formatted text file (like notepad, or vi or emacs) can be used to create the script. Within uCon the Scripts->Edit and Scripts->New menu items will open up an editor on a current or new script respectively. The default editor is Notepad; however, this can be changed through the dialog box opened by the menu item Config->Miscellaneous by modifying the content of the "Editor" item within that dialog. A script is simply a set of commands that build up some type of interaction with the target connection (COM port, telnet, ssh, etc...) The most typically used command is likely to be "SEND" because this is the command that sends a string to the target. For more information on each of the script commands, refer to the following list. For a few quick examples refer to the examples below.
RUNNING A SCRIPT:
A script can be run through the Scripts->Run dialog box, or through a function key or button configured to access a script. In either case, when the filename is specified, it can be a full path or just a filename. If a full path is specified, the uCon will look for the script in the specified directory path. If only a filename is specified, then uCon will look for that file under the scripts directory below the directory under which uCon was installed. Typically this is X:/program files/ucon (with 'X' being a drive specification).
As of release 5.2 if uCon (Jan 2006), a script can also be paused. This simply allows the user to tell the script runner to stop prior to starting the next command in the script. Then, when in the paused state, the script can either be resumed or stepped. Resume simply allows the script to continue and step tells the script running to execute the next command in the script and return to a paused state.
TRACING A RUNNING SCRIPT:
uCon's system trace facility allows the user to monitor the execution of the script. Click the Config->SystemTrace menu item to open the system trace dialog box, and check the SCRIPTS check box. Then, if the trace window is not visible (refer to main window for view of trace window), click the Toggle button in that dialog box to enable it.
COMMANDS:
Following is the list of commands currently available for scripts:
· BEEP {frequency (hz)} {duration (msec)}
Send audible tone.
· BMPVIEW [-h:l:m:R:rT:t:w:] {imagefile name} {viewer tag}
Display raw bitmap image file.
· CLS [-t]
Clear console screen.
· COMPORT ['command']
Run some kind of command or configuration change on the COM port.
· CONNECT {connection_type} {connection-specific arguments}

Change the current backend connection.
· DIALOG [-h:mt:w:x:y:] {dialog type} {user_message_string}
Interact with the user through some type of dialog box.
· ECHO {any string}
Echo the specified string to the console.
· EXIT [UCON]
Terminate the script.
· FKEY [-bcs] {1-16} [{label} {text}]
Establish a function key (or button) setting.
· FILE {sub-command} {filename} [args] ...
Interface to a file.
· FSEND [-d] {filename}
Transfer file to target with no protocol.
· GOSUB {tag}
Call some tagged subroutine within the script.
· GOTO {tag}
Branch to some tagged location in a script file.
· IF {var1} {test} {var2} {ACTION} [else ACTION]
Conditional test.
· LOG {on|off|append} [file to log to]
Enabled and/or disable logging.
· MONCMD [-p:vw:] {target_ip_address} {"string_to_target"}
Send a command to a target running MicroMonitor..
· NEWMON [-A:B:b:p:vw:] {target_ip_address} {binary_image_file}
Update the boot image of a target running MicroMonitor.
· RECV [-txX] [arg1]
Wait for some sequence of characters.
· REGEX [-bcin] {regular expression} {test string}
Run regular expression processing on a specified string.
· RETURN
Return from a subroutine (something called by GOSUB).
· RUN {scriptname}
Run a script from within a script.
· SEND [-cnp:rt:Tx] ["string_to_target"]
Pass the specified string to the target and wait for a prompt.
· SET [-e] {VAR} {VAL | =EXPRESSION}
Establish the content of a shell variable.
· SLEEP {msecs}
Delay in milliseconds.
· SYSTEM [-pw] {"cmdline"}
Process a command line as a local, PC-resident command.
· TCP [-ch:p:r]
Interact with other TCP based servers, particularly a remote uCon session's Telnet server.
· TIME [-ef:s]
Print the current time of day or compute elapsed time in milliseconds.
· TITLE [-t] [string]
Update the content of uCon's title bar.
· TFTP [-r:v] {server_ip} {get | put} {srcfile} {destfile} {netascii | octet}
Transfer a file with the PC being the client.
· XMODEM [-dnp:t:] {send | send1k | recv | recvcrc} {xmodem cmdline} {filename}
Startup an xmodem file transfer through a script.
SHELL VARIABLES:
Shell variables are used in uCon's scripts the same way shell variables would be used in any other shell. A variable is established either intrinsically (see below) or with the SET command, then the content of that variable can be accessed later by prefixing the name of the variable with a dollar sign ($).
Some variable names are dedicated to certain commands...
- FILE is used by the FILE command

- PROMPT is used by the SEND command
- TIMEOUT is used by SEND and RECV to indicate a timeout state.
- DIALOG is used by the DIALOG command
- TIME is used by the TIME command
- DHCP_CLIENT_IP is set by the DHCP server to the most recent "your_ip" value configured by the server.
- BAUDRATE and COMPORT are set by the COMPORT command.
- REGEX, RM_N, RS_N & RE_N (where 1<=N<=??) are populated by the REGEX command.
- TFTPSRVR_PORT is used by the TFTP server in uCon to override the default port 69.
- FKEY is used by the FKEY command.

. . .

Some variable names are initialized at uCon startup...
- MYIP is loaded with the "dot" formatted IP address of the PC running uCon.
- INSTALLDIR is loaded with the directory into which uCon was installed.
- LINES is loaded (and reloaded when size changes) with the size (height in lines) of the console window.

Note that it is likely that the value of INSTALLDIR will contain the string "Program Files"; hence, when using $INSTALLDIR as an argument to a command, you will probably need to wrap it with double quotes so that it is treated as a single argument.
Most of the examples below use shell variables in some way, particularly, EXAMPLE_4 shows how shell variables can be nested to allow you to treat a set of variables as a table.
Note that the current set of established shell variables can be dumped to the console using the Scripts->Environment menu item.
SYMBOLS:
An extension to shell variables is the script runner's ability to process symbols. Similar to the way a '$' is used to indicate a shell variable, a '%' (percent sign) is used to indicate a symbol. Symbols are simply strings found in a symbol file (named by the content of the SYMFILE shell variable). The format of the content of the symbol file is simply...
symbol relacement-text
symbol1 relacement-text1
symbol2 relacement-text2
With this file pointed to by the SYMFILE shell variable, a script can include "%symbol" and it will be replaced with "replacement-text" at runtime. Note this level of command line processing is only performed if the SYMFILE shell variable is set and points to a valid file.
EXAMPLES:
In each of the following examples, the content of the script is in blue, fixed-width font and the output of the script is green, fixed-width font to distinguish it from the descriptive text. Also refer to each of the various command help text for additional examples applicable to that command.
EXAMPLE_1:
This first example demonstrates how a script is used to automate a login process using the SEND command and its ability to wait for a specified string (or prompt). We assume that the target connected to the COM port is waiting for a login or user name, and after that username is entered the user must enter a password and if the password is correct, a system prompt will be presented. The prompt used for the password is "Password:" and the system prompt generated after the password has been entered is "CLI:".

```
SET PROMPT Password:
SEND "els" # Send "els" and wait for the string
"Password": to
# be sent by the target system.
SET PROMPT CLI:
SEND "els_pswd" # With the PROMPT shellvar changed, this
SEND command
# waits for the string "CLI:" to complete.
```

EXAMPLE_2:
The "programmability" mentioned earlier is provided by the "IF" command and the script's ability to deal with subroutines and branches. Here is a simple example that will count (and

echo that count) up to five...

```
SET COUNT 0
# TOP_OF_LOOP:
GOSUB SHOW_COUNT
SET COUNT=$COUNT+1
IF $COUNT le 5 GOTO TOP_OF_LOOP
EXIT
# SHOW_COUNT:
ECHO Hello, the count is now $COUNT
RETURN
```

The '#' is used in the script in two ways:

· it indicates that all text after (and including) the '#" is not to be processed as a command

· it is used as GOTO and GOSUB labels.

. . .

Notice in example #1 that the '#' does not have to be at the beginning of a line, it can be at any point in the line to let the command parser know that no further processing of that line should be done. If used as a label for GOTO or GOSUB, then it must be the first character of a new line and the label is the first block of non-whitespace after the initial '#' character.

This example also demonstrates the expression evaluation capability of the SET command.

The output of this script would be...

```
Hello, the count is now 0
Hello, the count is now 1
Hello, the count is now 2
Hello, the count is now 3
Hello, the count is now 4
Hello, the count is now 5
```

EXAMPLE_3:

This script demonstrates the use of the DIALOG command. Assume we want to interact with the user to query for some number between 0 and 4.. This script will query the user, and respond if the input is out of range...

```
# TOP:
DIALOG Q_AND_A "Please enter a number between 1 and 4:"
IF $DIALOG lt 1 GOTO ERROR
IF $DIALOG gt 4 GOTO ERROR
DIALOG OK "Thanks!"
EXIT
# ERROR:
DIALOG YES_NO "Out of range, want to try again?"
IF $DIALOG seq YES GOTO TOP
DIALOG OK "Fine, just click OK to exit."
EXIT
```

EXAMPLE_4:

This script demonstrates the use of nested shell variables. It allows you to establish a set of shell variables and then use another variable as an index into the set (as if it was a table in 'C').

```
SET VAR1 Hello,
SET VAR2 my
SET VAR3 name
SET VAR4 is
SET VAR5 Fred
SET IDX 1
# NEXT:
ECHO ${VAR${IDX}}
SET IDX=$IDX+1
IF $IDX lt 6 GOTO NEXT
```

In the above script, the variables VAR[1-5] are created, and can be thought of as a 5-element table of words. The loop below it uses $IDX to step through the table. The output of this

script would be...

```
Hello,
my
name
is
Fred
```

EXAMPLE_5:

This script demonstrates the use of the FILE command and its ability to query file information and content for use in a script. The script assumes that the target is running MicroMonitor, and that there are a few files in the monitor's file system. In the nutshell, this script dumps the output of the MicroMonitor "tfs ls" command to a file (C:/tmp/log). The presence of that file is verified by the first FILE command, then the content of that file is is queried by the second FILE command and acted upon using IF.

```
SET LOGFILE C:/tmp/log
SET PROMPT "uMON>"
LOG on $LOGFILE
SEND "tfs ls"
LOG off
FILE fstat $LOGFILE
IF $FSTAT sne FILE GOTO ERROR_1
SET LINE 3
SET COUNT 1
ECHO
# DUMP_FILES:
FILE token $LOGFILE 1 $LINE
IF $TOKEN seq \$TOKEN GOTO DUMP_DONE
ECHO " "FILE$COUNT: $TOKEN
SET LINE=$LINE+1
SET COUNT=$COUNT+1
GOTO DUMP_FILES
# DUMP_DONE:
ECHO OK, all finished!
EXIT
# ERROR_1:
ECHO Log file $LOGFILE not found
EXIT
```

The ouput of the above script depends on the set of files currently in TFS; however, here is an example...

```
uMON>tfs ls
Name Size Location Flags Info
app_01 196752 0xffe2412c Ec
app_go 68844 0xffe0005c bEc
monrc 354 0xffe10eac e
symtbl 77918 0xffe5421c
try 158 0xffe10dac e
Total: 5 items listed (344026 bytes).
uMON>
FILE1: app_01
FILE2: app_go
FILE3: monrc
FILE4: symtbl
FILE5: try
OK, all finished!
```

Note1: Some of the output is generated by the target and some is generated by the ECHO command in the script.

Note2: The "token" subcommand of FILE is useful for parsing whitespace delimited strings within a file. If you need to parse something a bit more complicated, refer to the REGEX command.

EXAMPLE_6:
This example shows you how to make some noise when uCon starts up. We will build a script called startup_sound, and to make the script execute when uCon starts up we add that script name to the "StartScr" text box in the Config->Miscellaneous dialog. We also have to make a few assumptions about the environment for this...
The tool sndrec32 is located under C:\windows\system32 and some system .wav files are found under C:\windows\media (refer to the script below, and make appropriate changes if needed).

```
SET SOUND C:\windows\system32\sndrec32
SET MEDIA C:\windows\media\tada.wav
SYSTEM -pw "${SOUND} /embedding /play /close \"${MEDIA}\""
```

As far as scripting goes, this is a simple example; however, it does demonstrate the use of the SYSTEM command, and provides a cute way to generate whatever sound is in a .wav file on your system.
Just in keeping with the examples above, now lets do something that uses the "FILE" command to step through all of the .wav files found in the C:\windows\media directory...

```
SET SOUND C:\windows\system32\sndrec32
SET MEDIA C:\windows\media\*.wav
FILE first $MEDIA
# NEXTFILE:
FILE next
if "$FILE" seq \$FILE EXIT
SYSTEM -pw "${SOUND} /embedding /play /close
\"C:\windows\media\\${FILE}\""
GOTO NEXTFILE
```

The output of the above script is audible. You'll hear the playing of each .wav file in the media directory. The important point here is to note that the "FILE first" and "FILE next" commands provide the ability to walk through a directory of files with a name filter (the above case uses *.wav).

```
GOSUB DUT

SLEEP 100

#Muuttujat
SET TMP
SET VIRHE

#Pääohjelma

#Enter testmode
send -n -r "testmode"
sleep 5000

send -n -r ""
sleep 200

send -n -r ""
sleep 200

GOSUB VIKA_TESTI_K

#parametrit
send -n -r "param 2.5 1.6 15.0 0.0 5.0 87.0 19766 1"
sleep 100


# kulman haku
send -n -r "kulma 1.0"
sleep 100

# pääkontaktori päälle
send -n -r "mc 1 1"
sleep 100

# modulaatio päälle ja 5s jälkeen pois
send -n -r "mod 1"
sleep 4500
send -n -r "mod 0"
sleep 200

# mc pois
send -n -r "mc 0"
sleep 100


#log päälle
LOG on TMP

#kysyy kulmaa
send -n -r "getangle"
sleep 100

#log pois päältä
LOG off
```

```
#Ottaa kulman logista, kolmas token, eka rivi
FILE token TMP 3 1

#Asettaa tuloksen muuttujaan kulma
SET KULMA=$FILE

#Jos kulmassa virhe niin siirry virhetilaan
IF $KULMA eq 0 GOTO KULMA_ERROR

#uudet parametrit kulman kanssa
send -n -r "param 2.5 1.6 15.0 $KULMA.0 5.0 87.0 19766 1"
sleep 300

GOSUB KUORMA_NO

#ohjeita käyttäjälle
DIALOG OK "Laita kuormakone päälle ja paina sitten ok jatkaaksesi testiä."

GOSUB DUT

#nollaa laskuri
SET COUNT 0

# aseta skaalaukset
send -n -r "scale 7.0 1.0 1.0"
sleep 500


# kontaktori päälle
send -n -r "mc 1"
sleep 300



# modulaatio päälle
send -n -r ""
sleep 200
send -n -r "mod 1"
sleep 200
send -n -r "mod 1"
sleep 200

send -n -r ""
sleep 200


# UUDESTAAN:

send -n -r "scale 0.0 0.0"
SLEEP 100

# Kuorma paalle 80%
GOSUB PLUS80

SLEEP 1000

# jarru auki
```

```
send -n -r "br1 1"
sleep 100

# jarru auki
send -n -r "br2 1"
sleep 200

GOSUB VIKA_TESTI

# aja 1000rpm 2s ramppi
send -n -r "drive 1000.0 2.0"
sleep 5000

send -n -r "drive 200.0 0.5"
sleep 1000

send -n -r "drive 0.0 0.5"
sleep 700

GOSUB VIKA_TESTI

# jarru kiinni
send -n -r "br1 0"
sleep 100

# jarru kiinni
send -n -r "br2 0"
sleep 100

GOSUB VIKA_TESTI

# kuorma päälle -80%
GOSUB MINUS80


sleep 1000

# jarru auki
send -n -r "br1 1"
sleep 100

# jarru auki
send -n -r "br2 1"
sleep 100

# aja 1000rpm 2s ramppi
send -n -r "drive -1000.0 2.0"
sleep 5000

send -n -r "drive -200.0 0.5"
sleep 1000

send -n -r "drive 0.0 0.5"
sleep 700

GOSUB VIKA_TESTI

# jarru kiinni
```

```
send -n -r "br1 0"
sleep 100

# jarru kiinni
send -n -r "br2 0"
sleep 200

# incrementoi laskuria
SET COUNT=$COUNT+1
sleep 100

# ajelee siis kolmesti ylös ja alas
IF $COUNT le 3 GOTO UUDESTAAN
sleep 100

GOTO LOPPU



#Aliohjelmat

# VIKA_TESTI:

LOG on VIRHE
sleep 100

#kysyy vointia
send -n -r ""

sleep 100
LOG off
sleep 100

#tarkistaa onko saatu virhekoodi
FILE strstr VIRHE M

ECHO $FILE

#jos on hypätään virhetilaan
IF $FILE sne YES GOTO CODE_ERROR

#jos ei palataan pääohjelmaan
RETURN



# CODE_ERROR:

#Ilmoitetaan käyttäjälle virhe
ECHO Got error code
send -n -r ""
send -n -r ""

GOSUB KEITHLEY

sleep 500
# asetetaan kuorma nollaan
send -n -r ":SOURCE:CURRENT 0.012"
```

```
GOTO PYS



#yhdistä Kiethley Sourcemeter laitteeseen
# KEITH:

CONNECT CLOSE

CONNECT comport 2 57600 8 NONE 1 NONE

SLEEP 100

RETURN


# yhdistä DUT
# DUT:

CONNECT CLOSE

CONNECT comport 1 19200 8 NONE 1 NONE

SLEEP 100

RETURN


#asetakuorma 80%
# PLUS80

GOSUB KEITH

send -n -r ":SOURCE:CURRENT 0.0157"
SLEEP 300

GOSUB DUT

RETURN


# MINUS80

GOSUB KEITH

send -n -r ":SOURCE:CURRENT 0.0086"
SLEEP 300

GOSUB DUT

RETURN


# pysäytä DUT
```

```
# PYS:

GOSUB DUT

# jarru kiinni
send -n -r "br1 0"
sleep 100

# jarru kiinni
send -n -r "br2 0"
sleep 200

GOTO LOPPU


#testaa tuleeko vika kulmanhakuprosessissa
# VIKA_TESTI_K:

LOG on VIRHE
sleep 200
send -n -r ""
sleep 200
LOG off
sleep 200
FILE strstr VIRHE OK

ECHO $FILE

sleep 200

IF $FILE sne YES GOTO CODE_ERROR

RETURN


# CODE_ERROR:

ECHO Got error code

send -n -r ""
send -n -r ""

GOTO LOPPU



# KULMA_ERROR:

ECHO Error getting resolver angle
GOTO LOPPU


#Keithleyn asetusten alustaminen
# KUORMA_NO:

GOSUB KEITH

send -n -r "*RST"
```

```
send -n -r ":ROUTE:TERMINALS REAR"

send -n -r ":SOURCE:FUNCTION CURRENT"

send -n -r ":SOURCE:CURRENT 0.012"

send -n -r ":OUTP:STAT 1"

GOSUB DUT

RETURN


# LOPPU:

GOSUB KEITH

# lopettaessa asetetaan aina kuorma nollaksi
send -n -r ":SOURCE:CURRENT 0.012"

GOSUB DUT

EXIT
```