Design And Implementation of An All Digital Scanned Probe Microscope Controller

by

Avid Lemus

A thesis presented to the University of Manitoba
in partial fulfilment of the
requirements for the degree of
Master of Science
in
Electrical Engineering

Winnipeg, Manitoba, 1995

ISBN 0-612-13292-7

Name _____

Dissertation Abstracts International and *Masters Abstracts International* are arranged by broad, general subject categories. Please select the one subject which most nearly describes the content of your dissertation or thesis. Enter the corresponding four-digit code in the spaces provided.

Engineering, Electronics and Electrical

SUBJECT TERM

0544

SUBJECT CODE

UMI

Subject Categories

THE HUMANITIES AND SOCIAL SCIENCES

COMMUNICATIONS AND THE ARTS
Architecture 0729
Art History 0377
Cinema 0900
Dance 0378
Fine Arts 0357
Information Science 0723
Journalism 0391
Library Science 0399
Mass Communications 0708
Music 0413
Speech Communication 0459
Theater 0465

EDUCATION
General 0515
Administration 0514
Adult and Continuing 0516
Agricultural 0517
Art 0273
Bilingual and Multicultural 0282
Business 0688
Community College 0275
Curriculum and Instruction 0727
Early Childhood 0518
Elementary 0524
Finance 0277
Guidance and Counseling 0519
Health 0680
Higher 0745
History of 0520
Home Economics 0278
Industrial 0521
Language and Literature 0279
Mathematics 0280
Music 0522
Philosophy of 0998
Physical 0523
Psychology 0525
Reading 0535
Religious 0527
Sciences 0714
Secondary 0533
Social Sciences 0534
Sociology of 0340
Special 0529
Teacher Training 0530
Technology 0710
Tests and Measurements 0288
Vocational 0747

LANGUAGE, LITERATURE AND LINGUISTICS
Language
- General 0679
- Ancient 0289
- Linguistics 0290
- Modern 0291

Literature
- General 0401
- Classical 0294
- Comparative 0295
- Medieval 0297
- Modern 0298
- African 0316
- American 0591
- Asian 0305
- Canadian (English) 0352
- Canadian (French) 0355
- English 0593
- Germanic 0311
- Latin American 0312
- Middle Eastern 0315
- Romance 0313
- Slavic and East European 0314

PHILOSOPHY, RELIGION AND THEOLOGY
Philosophy 0422
Religion
- General 0318
- Biblical Studies 0321
- Clergy 0319
- History of 0320
- Philosophy of 0322

Theology 0469

SOCIAL SCIENCES
American Studies 0323
Anthropology
- Archaeology 0324
- Cultural 0326
- Physical 0327

Business Administration
- General 0310
- Accounting 0272
- Banking 0770
- Management 0454
- Marketing 0338

Canadian Studies 0385
Economics
- General 0501
- Agricultural 0503
- Commerce-Business 0505
- Finance 0508
- History 0509
- Labor 0510
- Theory 0511

Folklore 0358
Geography 0366
Gerontology 0351
History
- General 0578
- Ancient 0579
- Medieval 0581
- Modern 0582
- Black 0328
- African 0331
- Asia, Australia and Oceania 0332
- Canadian 0334
- European 0335
- Latin American 0336
- Middle Eastern 0333
- United States 0337

History of Science 0585
Law 0398
Political Science
- General 0615
- International Law and Relations 0616
- Public Administration 0617

Recreation 0814
Social Work 0452
Sociology
- General 0626
- Criminology and Penology ... 0627
- Demography 0938
- Ethnic and Racial Studies 0631
- Individual and Family Studies 0628
- Industrial and Labor Relations 0629
- Public and Social Welfare 0630
- Social Structure and Development 0700
- Theory and Methods 0344

Transportation 0709
Urban and Regional Planning 0999
Women's Studies 0453

THE SCIENCES AND ENGINEERING

BIOLOGICAL SCIENCES
Agriculture
- General 0473
- Agronomy 0285
- Animal Culture and Nutrition 0475
- Animal Pathology 0476
- Food Science and Technology 0359
- Forestry and Wildlife 0478
- Plant Culture 0479
- Plant Pathology 0480
- Plant Physiology 0817
- Range Management 0777
- Wood Technology 0746

Biology
- General 0306
- Anatomy 0287
- Biostatistics 0308
- Botany 0309
- Cell 0379
- Ecology 0329
- Entomology 0353
- Genetics 0369
- Limnology 0793
- Microbiology 0410
- Molecular 0307
- Neuroscience 0317
- Oceanography 0416
- Physiology 0433
- Radiation 0821
- Veterinary Science 0778
- Zoology 0472

Biophysics
- General 0786
- Medical 0760

EARTH SCIENCES
Biogeochemistry 0425
Geochemistry 0996
Geodesy 0370
Geology 0372
Geophysics 0373
Hydrology 0388
Mineralogy 0411
Paleobotany 0345
Paleoecology 0426
Paleontology 0418
Paleozoology 0985
Palynology 0427
Physical Geography 0368
Physical Oceanography 0415

HEALTH AND ENVIRONMENTAL SCIENCES
Environmental Sciences 0768
Health Sciences
- General 0566
- Audiology 0300
- Chemotherapy 0992
- Dentistry 0567
- Education 0350
- Hospital Management 0769
- Human Development 0758
- Immunology 0982
- Medicine and Surgery 0564
- Mental Health 0347
- Nursing 0569
- Nutrition 0570
- Obstetrics and Gynecology .. 0380
- Occupational Health and Therapy 0354
- Ophthalmology 0381
- Pathology 0571
- Pharmacology 0419
- Pharmacy 0572
- Physical Therapy 0382
- Public Health 0573
- Radiology 0574
- Recreation 0575
- Speech Pathology 0460
- Toxicology 0383

Home Economics 0386

PHYSICAL SCIENCES
Pure Sciences
Chemistry
- General 0485
- Agricultural 0749
- Analytical 0486
- Biochemistry 0487
- Inorganic 0488
- Nuclear 0738
- Organic 0490
- Pharmaceutical 0491
- Physical 0494
- Polymer 0495
- Radiation 0754

Mathematics 0405
Physics
- General 0605
- Acoustics 0986
- Astronomy and Astrophysics 0606
- Atmospheric Science 0608
- Atomic 0748
- Electronics and Electricity 0607
- Elementary Particles and High Energy 0798
- Fluid and Plasma 0759
- Molecular 0609
- Nuclear 0610
- Optics 0752
- Radiation 0756
- Solid State 0611

Statistics 0463

Applied Sciences
Applied Mechanics 0346
Computer Science 0984
Engineering
- General 0537
- Aerospace 0538
- Agricultural 0539
- Automotive 0540
- Biomedical 0541
- Chemical 0542
- Civil 0543
- Electronics and Electrical 0544
- Heat and Thermodynamics ... 0348
- Hydraulic 0545
- Industrial 0546
- Marine 0547
- Materials Science 0794
- Mechanical 0548
- Metallurgy 0743
- Mining 0551
- Nuclear 0552
- Packaging 0549
- Petroleum 0765
- Sanitary and Municipal 0554
- System Science 0790

Geotechnology 0428
Operations Research 0796
Plastics Technology 0795
Textile Technology 0994

PSYCHOLOGY
General 0621
Behavioral 0384
Clinical 0622
Developmental 0620
Experimental 0623
Industrial 0624
Personality 0625
Physiological 0989
Psychobiology 0349
Psychometrics 0632
Social 0451

Nom _____

Dissertation Abstracts International est organisé en catégories de sujets. Veuillez s.v.p. choisir le sujet qui décrit le mieux votre thèse et inscrivez le code numérique approprié dans l'espace réservé ci-dessous.

_____ ☐☐☐ U·M·I

SUJET — CODE DE SUJET

Catégories par sujets

HUMANITÉS ET SCIENCES SOCIALES

COMMUNICATIONS ET LES ARTS
Architecture 0729
Beaux-arts 0357
Bibliothéconomie 0399
Cinéma 0900
Communication verbale 0459
Communications 0708
Danse 0378
Histoire de l'art 0377
Journalisme 0391
Musique 0413
Sciences de l'information 0723
Théâtre 0465

ÉDUCATION
Généralités 515
Administration 0514
Art 0273
Collèges communautaires 0275
Commerce 0688
Économie domestique 0278
Éducation permanente 0516
Éducation préscolaire 0518
Éducation sanitaire 0680
Enseignement agricole 0517
Enseignement bilingue et multiculturel 0282
Enseignement industriel 0521
Enseignement primaire. 0524
Enseignement professionnel 0747
Enseignement religieux 0527
Enseignement secondaire 0533
Enseignement spécial 0529
Enseignement supérieur 0745
Évaluation 0288
Finances 0277
Formation des enseignants 0530
Histoire de l'éducation 0520
Langues et littérature 0279
Lecture 0535
Mathématiques 0280
Musique 0522
Orientation et consultation 0519
Philosophie de l'éducation 0998
Physique 0523
Programmes d'études et enseignement 0727
Psychologie 0525
Sciences 0714
Sciences sociales 0534
Sociologie de l'éducation 0340
Technologie 0710

LANGUE, LITTÉRATURE ET LINGUISTIQUE
Langues
- Généralités 0679
- Anciennes 0289
- Linguistique 0290
- Modernes 0291

Littérature
- Généralités 0401
- Anciennes 0294
- Comparée 0295
- Médiévale 0297
- Moderne 0298
- Africaine 0316
- Américaine 0591
- Anglaise 0593
- Asiatique 0305
- Canadienne (Anglaise) 0352
- Canadienne (Française) 0355
- Germanique 0311
- Latino-américaine 0312
- Moyen-orientale 0315
- Romane 0313
- Slave et est-européenne 0314

PHILOSOPHIE, RELIGION ET THÉOLOGIE
Philosophie 0422
Religion
- Généralités 0318
- Clergé 0319
- Études bibliques 0321
- Histoire des religions 0320
- Philosophie de la religion 0322

Théologie 0469

SCIENCES SOCIALES
Anthropologie
- Archéologie 0324
- Culturelle 0326
- Physique 0327

Droit 0398
Économie
- Généralités 0501
- Commerce-Affaires 0505
- Économie agricole 0503
- Économie du travail 0510
- Finances 0508
- Histoire 0509
- Théorie 0511

Études américaines 0323
Études canadiennes 0385
Études féministes 0453
Folklore 0358
Géographie 0366
Gérontologie 0351
Gestion des affaires
- Généralités 0310
- Administration 0454
- Banques 0770
- Comptabilité 0272
- Marketing 0338

Histoire
- Histoire générale 0578
- Ancienne 0579
- Médiévale 0581
- Moderne 0582
- Histoire des noirs 0328
- Africaine 0331
- Canadienne 0334
- États-Unis 0337
- Européenne 0335
- Moyen-orientale 0333
- Latino-américaine 0336
- Asie, Australie et Océanie 0332

Histoire des sciences 0585
Loisirs 0814
Planification urbaine et régionale 0999
Science politique
- Généralités 0615
- Administration publique 0617
- Droit et relations internationales 0616

Sociologie
- Généralités 0626
- Aide et bien-être social 0630
- Criminologie et établissements pénitentiaires 0627
- Démographie 0938
- Études de l' individu et de la famille 0628
- Études des relations interethniques et des relations raciales 0631
- Structure et développement social 0700
- Théorie et méthodes. 0344
- Travail et relations industrielles 0629

Transports 0709
Travail social 0452

SCIENCES ET INGÉNIERIE

SCIENCES BIOLOGIQUES
Agriculture
- Généralités 0473
- Agronomie. 0285
- Alimentation et technologie alimentaire 0359
- Culture 0479
- Élevage et alimentation 0475
- Exploitation des pâturages 0777
- Pathologie animale 0476
- Pathologie végétale 0480
- Physiologie végétale 0817
- Sylviculture et faune 0478
- Technologie du bois 0746

Biologie
- Généralités 0306
- Anatomie 0287
- Biologie (Statistiques) 0308
- Biologie moléculaire 0307
- Botanique 0309
- Cellule 0379
- Écologie 0329
- Entomologie 0353
- Génétique 0369
- Limnologie 0793
- Microbiologie 0410
- Neurologie 0317
- Océanographie 0416
- Physiologie 0433
- Radiation 0821
- Science vétérinaire 0778
- Zoologie 0472

Biophysique
- Généralités 0786
- Medicale 0760

SCIENCES DE LA TERRE
Biogéochimie 0425
Géochimie 0996
Géodésie 0370
Géographie physique 0368
Géologie 0372
Géophysique 0373
Hydrologie 0388
Minéralogie 0411
Océanographie physique 0415
Paléobotanique 0345
Paléoécologie 0426
Paléontologie 0418
Paléozoologie 0985
Palynologie 0427

SCIENCES DE LA SANTÉ ET DE L'ENVIRONNEMENT
Économie domestique 0386
Sciences de l'environnement 0768
Sciences de la santé
- Généralités 0566
- Administration des hôpitaux 0769
- Alimentation et nutrition 0570
- Audiologie 0300
- Chimiothérapie 0992
- Dentisterie 0567
- Développement humain 0758
- Enseignement 0350
- Immunologie 0982
- Loisirs 0575
- Médecine du travail et thérapie 0354
- Médecine et chirurgie 0564
- Obstétrique et gynécologie 0380
- Ophtalmologie 0381
- Orthophonie 0460
- Pathologie 0571
- Pharmacie 0572
- Pharmacologie 0419
- Physiothérapie 0382
- Radiologie 0574
- Santé mentale 0347
- Santé publique 0573
- Soins infirmiers 0569
- Toxicologie 0383

SCIENCES PHYSIQUES
Sciences Pures
Chimie
- Généralités 0485
- Biochimie 487
- Chimie agricole 0749
- Chimie analytique 0486
- Chimie minérale 0488
- Chimie nucléaire 0738
- Chimie organique 0490
- Chimie pharmaceutique 0491
- Physique 0494
- PolymÈres 0495
- Radiation 0754

Mathématiques 0405
Physique
- Généralités 0605
- Acoustique 0986
- Astronomie et astrophysique 0606
- Électronique et électricité 0607
- Fluides et plasma 0759
- Météorologie 0608
- Optique 0752
- Particules (Physique nucléaire) 0798
- Physique atomique 0748
- Physique de l'état solide 0611
- Physique moléculaire 0609
- Physique nucléaire 0610
- Radiation 0756

Statistiques 0463

Sciences Appliqués Et Technologie
Informatique 0984
Ingénierie
- Généralités 0537
- Agricole 0539
- Automobile 0540
- Biomédicale 0541
- Chaleur et thermodynamique 0348
- Conditionnement (Emballage) 0549
- Génie aérospatial 0538
- Génie chimique 0542
- Génie civil 0543
- Génie électronique et électrique 0544
- Génie industriel 0546
- Génie mécanique 0548
- Génie nucléaire 0552
- Ingénierie des systèmes 0790
- Mécanique navale 0547
- Métallurgie 0743
- Science des matériaux 0794
- Technique du pétrole 0765
- Technique minière 0551
- Techniques sanitaires et municipales 0554
- Technologie hydraulique 0545

Mécanique appliquée 0346
Géotechnologie 0428
Matières plastiques (Technologie) 0795
Recherche opérationnelle 0796
Textiles et tissus (Technologie) 0794

PSYCHOLOGIE
Généralités 0621
Personnalité 0625
Psychobiologie 0349
Psychologie clinique 0622
Psychologie du comportement 0384
Psychologie du développement 0620
Psychologie expérimentale 0623
Psychologie industrielle 0624
Psychologie physiologique 0989
Psychologie sociale 0451
Psychométrie 0632

DESIGN AND IMPLEMENTATION OF AN ALL DIGITAL SCANNED PROBE MICROSCOPE CONTROLLER

BY

AVID LEMUS

A Thesis submitted to the Faculty of Graduate Studies of the University of Manitoba in partial fulfillment of the requirements of the degree of

MASTER OF SCIENCE

I hereby declare that I am the sole author of this thesis.

I authorize the University of Manitoba to lend this thesis to other institutions or individuals for the purpose of scholarly research.

I further authorize the University of Manitoba to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Abstract

Hardware for the data conversion required by a Scanned Probe Microscope controller is designed and built. This hardware is used to implement a DSP-based generic digital controller. The generality of the system is kept whenever possible to allow for compatibility with a wide range of SPMs.

The controller is built around a Motorola 56001 Digital Signal Processor. This processor is then interfaced to 486DX2-66 IBM compatible Personal Computer, through an ISA bus prototype card. The custom built hardware includes a printed circuit board that implements all data conversion to and from the analogue and digital domains. The I/O board incorporates optically isolated 20-bit digital-to-analogue converters and 18-bit analogue-to-digital converters. Six, input and output channels with three outputs amplified to implement the control signals of a PZT positioner are implemented in the I/O board. The I/O board usefulness and generality is maintained by using an FPGA to program the interface to the Digital Signal Processor.

Software to implement a Proportional Integral (PI) controller on the DSP is written. The controller's generality is kept by allowing one or two images to be obtained at the same time. The software running on the DSP generates all control signals to control a single closed loop plus a positioner, such as the case of the raster signal for a PZT. The DSP software also handles coarse approach to the sample through a stepper motor.

Acknowledgments

I would like to thank Professor Douglas J. Thomson for his advice for and guidance throughout the work described in this thesis project. I would also like to thank Professor Bob McLeod for taking time out of his busy schedule to provide some of the parts required in the construction of the Analogue I/O Board. I remain indebted to Dr. Glenn Rosendahl for sharing his knowledge on FPGA and PCB technology with me. Ken Biegun was very helpful, and without his help this project could not have been completed. The whole technical staff must be acknowledged for going out of their way to provide me with some very rushed components.

Finally, I would like to thank my classmates who spent countless hours helping solve some of the problems I ran into along the way. I would especially like to thank David T. Shimizu and Jochonia N. Nxumalo whom I consider more than classmates, my friends.

Contents

List Of Figures

List Of Tables

CHAPTER 1 *Introduction*

1.1 Reason For Undertaking This Thesis

Scanned Probe Microscopes (SPMs) such as the Scanning Tunnelling Microscope (STM), and Scanning Force Microscope (AFM) have been around for about a decade. However, it has not been until relatively recently that digital control systems for these devices have appeared in the field. This is due to the rapid advances in VLSI technology, namely data converters such as Digital to Analog Converters (DACs) and Analog to Digital Converters (ADCs), and Digital Signal Processors (DSPs). There are a variety of mixed commercial instruments that incorporate some of the abilities of a digital system, however they are limited as well as very expensive.

Since the invention of the STM and AFM, many variations of these instruments have been introduced to the field, now known as Scanned Probe Microscopy (SPM). All these instruments require similar control signals, and hence a generic

controller that will adapt to the majority of these instruments can be constructed. The Purpose of this thesis project is to design and implement a DSP based generic digital control system for SPMs.

1.2 The Atomic Force Microscope (AFM)

As mentioned before there are various types of microscopes that work on the principle of a scanning probe. A good representative and by far the most popular is the AFM. Invented in 1985 by G. Binnig with Ch. Gerber and C.F. Quate[ref.1,2].

G. Binnig came across the idea of the AFM by drawing from previous experiences from the STM. The STM makes uses of the tunnelling current between a very sharp tip and a conductive sample, to obtain a topographical interpretation of the sample. Binnig realized that the force between the tip and the sample could be used instead of the tunnelling current. Subsequent calculations showed that it was possible to construct a cantilever with a weak enough spring constant, in order to measure the atomic force between a single atom in the tip and a single atom in the sample. The arrangement for either the AFM or STM is show in Figure 1.

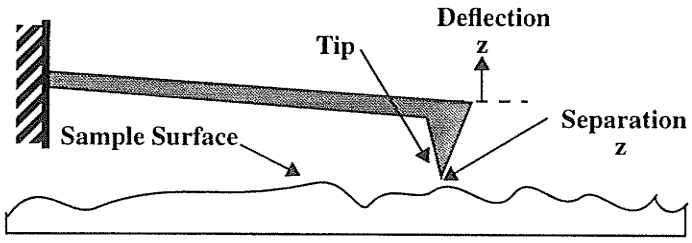


FIGURE 1. AFM tip over sample surface

Using the atomic force instead of the tunnelling current has the advantage of being able to image non-conducting as well as conducting surfaces. However, the sensitivity of the tunnelling current is higher, and thus yields more details of the surface. This is due to the fact that the tunnelling current in the STM has the following form:

$$I \propto e^{z\sqrt{\phi}} \qquad \text{(EQ 1)}$$

Where I is the tunnelling current, ϕ is a function of the tip and the surface work functions, and z is the separation between the tip and surface. In the AFM the signal of interest has the form:

$$F = Kz \qquad \text{(EQ 2)}$$

Where F is the atomic force exerted by the tip on the sample, K is the spring constant of the cantilever, and z is the deflection of the tip from is rest position. Hence when high resolution (Atomic resolution < Angstrom) is required, and a conductive sample is being analysed, the STM should be used.

In both of these instruments an image is obtained by rastering the tip over the sample, and keeping the control signal constant. In the STM this is the tunnelling current, and in the AFM the deflection of the cantilever is used. As the tip is rastered keeping the control signal constant, contours of constant contact force(AFM) or constant tunnelling current (STM) are obtained. This is done by a closed loop feedback, and is one of the functions of the control system. These equi-force or equi-current contours are then assemble into a representation of the sample

surface topography. This representation can be of several types, wire mesh, greyscale, cross-sections, or 3D assimilations. A sample of cross-sections and greyscale representations are shown on chapter 2 under section of ***Image Display System***

1.2.1 Feedback Mechanism

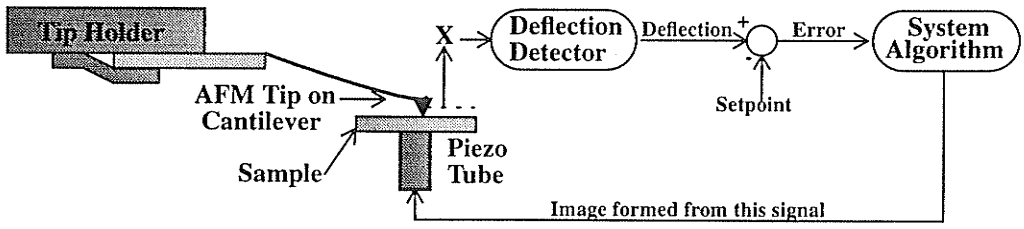


FIGURE 2. AFM Feedback Mechanism

As the sample is rastered beneath the cantilever, the cantilever deflection **x** changes due to variations in surface topography. A desired deflection $\mathbf{x_0}$ is selected as the setpoint. The tip deflection from its equilibrium point is then compared to the setpoint and the error **e** is calculated as:

$$e = x - x_o \qquad \text{(EQ 3)}$$

Where this is then used to correct the Z height of the sample. Since for **e** greater than zero too much force is being exerted on the tip by the sample, the sample is moved down, by an amount defined by the control system algorithm. This is

known as negative feedback, and the ability of the control system to compensate for this error is the control system's performance.

1.2.2 The Rastering Signals

In all SPMs the probe is rastered relative to the sample under observation. This can be accomplished by either rastering the probe over the sample, or vice versa. Our AFM head uses the later technique. In order to achieve the fine movements required by these microscopes, piezoelectric ceramics are used. These devices are widely used as electromechanical transducers to obtain minute movements and for micropositioning. Piezoelectric materials are characterized by their ability to expand or contract when under the influence of an electric field. Their movement can be approximated by a linear relation although in practice they are both nonlinear and hysteretive:

$$d = K_p v \qquad \text{(EQ 4)}$$

Where d is the distance moved by the piezoelectric material (piezo), v is the voltage applied along the axis of motion, and K is a constant characteristic of the material and its geometry. Typical values for K are in the order of 100 Angtroms/Volt for commercially available piezo positioners.

Earlier STMs and AFMs used a variety of piezoelectric actuators. Tripods, made up of three separate piezos glued together in an orthogonal fashion provided motion in three dimensions X, Y, and Z. The motion inside the plane of the sample or perpendicular to the probe to sample separation axis is usually labelled X and Y,

while Z refers to the motion along the probe-to-sample distance axis. The tripod arrangement proved to be mechanically complex, and suffered from various undesired characteristics, such as low mechanical resonance, large crosstalk between piezo rods, and small scan range [ref.3,4].

Newer SPM systems use almost exclusively the tube scanner. This device is a hollow cylinder of piezoelectric material with a single electrode on the inner wall and four symmetrical longitudinal electrodes on the outer wall as show on Figure 3.

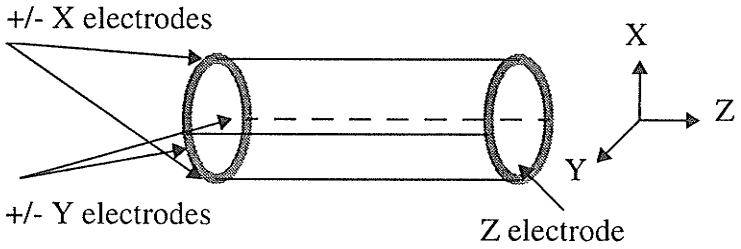


FIGURE 3. Tube scanner

These tubes are machined from a solid piece of piezoelectric material, metallized to form electrodes. The piezo tube is sectored into five electrodes to give orthogonal X, Y and Z motion. The piezo has solid electrode on the inside of the tube, which is used as the Z electrode. The outside of the tube is partitioned into four quadrants along the long axis of the tube. Two oposite quadrants are then used as the positive and negative X electrodes, and the other two are used for the Y electrodes. Movement is achieved by applying voltages to these electrodes. The

piezo tube in the AFM is the one purchased from Park Scientific Insruments[75] and has the following nominal sensitivities:

$$K_Z = 82\frac{\dot{A}}{V}$$

$$K_{X,Y} = 240\frac{\dot{A}}{V}$$

1.3 The Scanning Resistance Microscope (SRM)

The idea for the SRM was first developed by D. J. Thomson at the Electrical Engineering Department of the University of Manitoba. The SRM works very much like an AFM with a very important difference, the scanning tip is conductive and biased with a voltage as the tip is held against the surface. This is done to map out resistance of the sample and topography at the same time. The topography information is not really the important data. It is a sidefect of keeping constant pressure of the tip on the sample. This is required to avoid changes in tip to sample resistance due changes in contact area (i.e. contact area varies with applied pressure).

The feedback mechanism for the keeping a constant force on the surface is identical to the AFM. From the resistance information additional images are formed. Resistance images are very much like topography, difference being only on the physical interpretation of the data. The feedback mechanism again is identical to that of topography, with the exception of current being the control signal as opposed tip force in the topography.

The SRM can hardly be characterized by a simple current/voltage characteristic curve as this curve depends on many factors, such as pressure on the sample, tip

geometry, material characteristics for the tip, and material characteristics for the sample to name a few. A more detailed explanation of this instrument can be found in Jochonia N. Nxumalo's research papers[ref.5,6] and C. Shafai's MSc thesis [ref.7].

1.3.1 SRM Feedback Rastering Mechanism

As mentioned above the feedback mechanism for the SRM is very similar to the AFM. In the AFM the force applied to the surface by the tip is detected and the height of the sample is adjusted to compensate for any deviations from a prescribed setpoint. In the SRM two feedback loops are necessary. One keeps the force being applied to the surface constant (AFM loop), and the other keeps the current flowing from the tip to the sample constant (SRM loop).

The integral compensating function has the property of achieving zero error if we wait long enough for the feedback to settle. This is done by integrating the error, which is then applied as the compensating signal, as long as there is an error the integral keep on growing until it converges to the desired setpoint. It should be noted that this works as long as there is negative feedback or in other words the accumulating error is converging towards the desired response, if the error moves

in the opposite direction positive feedback is said to occur and the compensating signal diverges. The following block diagram depicts negative feedback.

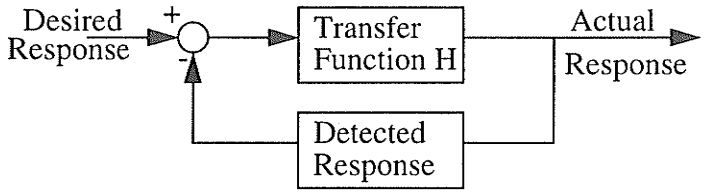


FIGURE 4. Negative Feedback Block diagram

The rastering of the tip over the sample is also done in the same manner as the AFM. The sample is mounted on a piezo tube (PZT) transducer and triangular voltage waves are applied to the +X, -X, +Y, and -Y electrodes in order to obtain sweeping in the X axis that is 512 times faster then the Y axis. An image of 256 by 256 points is obtained in the down sweep of the Y axis and a second one on its way up. Depending on the electronics, both SRM and AFM images are obtained on one pass of the y sweep, or one AFM on one sweep and one SRM on the second sweep. However due to hysteresis of the PZT positioner Corresponding images on even rows of the same image must agree with each other in X, Y direction scan, and scan frequency. The following are the parameters for the SRM PZT scanner:

$$K_Z = 133\frac{\dot{A}}{V}$$

$$K_{X,Y} = 1200\frac{\dot{A}}{V}$$

1.4 Scanning Probes

There are several different types of probes used in SPMs. These probes are optimized for the particular microscope they work on. That is to say they are characterized by the properties required by the particular type of microscope they work on. Some of these properties are: tip sharpnes, cantilever resonant frequency, and whether the tip is conducting or not. The following two subsections describe the probes used in AFMs, and those used in the Scanning Resistance Microscope (SRM).

1.4.1 AFM Scanning probes

AFM probes are usually comprised of a very sharp tip attached at the end of a very flexible beam or cantilever. This cantilever acts as a very weak spring that keeps the tip in contact with the sample surface when scanning. It is desired that this spring be very weak for a variety of reasons, one of them being that we do not want to damage the surface as we scan it. Calculations of the forces between atoms also leads to the conclusion that the cantilever must be very flexible, in order to track the surface at or approaching the atomic level. Consideration of the deflection detection method of the cantilever also requires that the probe be as pliable as possible, since more movement of the cantilever will yield a more detectable signal, and hence a better signal to noise ratio. However when we consider ambient vibrations we require that they be decoupled from the cantilever, and hence we want the cantilever to remain stationary relative to the sample. This requires a stiff spring constant, and hence a compromise is made between the resolution we require, and the noise coupled into data

The resonant frequency of the control system as a whole should be kept as high as possible to allow data acquisition at a reasonable rate. This limitation arises due to the fact that any control system will oscillate if operated at or above its resonant frequency. The resonant frequency of the cantilever when modelled as a simple mass spring system is given by:

$$f_o = \frac{1}{2\pi}\sqrt{\frac{k}{m_o}} \qquad \text{(EQ 5)}$$

where k is the spring constant of the cantilever, m_o is the effective mass loading of the spring, and π is the unitless mathematical constant 3.141592. From (eq. 5) it can be concluded that we can reduce the spring constant of the cantilever, meeting the requirement to keep the spring as flexible as possible, if at the same time we reduce the effective mass loading the spring. This ensures that the resonant frequency of the spring will be sufficiently high to allow reasonably fast data acquisition.This is one of the reasons why typical AFM tips are small, and somewhat hard to handle.

Typical cantilever lenghts are in the order of a millimetre. AFM tips today are micromachined using standard microlitographic methods [ref.8]. The most popular tips are made in batch processes from silicon nitride (Si_3N_4), making them very reliable and relatively inexpensive. Silicon nitride is unusually resistant to mechanical fatigue. Probes constructed from this material can be flexed trough an angle of 90^o to their relaxed position without breaking. Another advantage of these tips is that the back or top of the cantilever is usually made flat and can be usually coated with other material such as metals. This is useful in the deflection detection

processes, discussed on the following section, or when a conducting cantilever is required.

1.4.2 SRM Scanning Probes

Conducting AFM probes have successfully been used as SRM probes[ref.7]. Tungsten tips were initially used, but proved to be less than ideal, today diamond tips doped with Boron are used.

Tungsten tips are made by simply elctrochemicaly etching tungsten wire. One end of a tungsten wire of diameter of approximately 250 μm is placed in a bath of NaOH solution with a concentration of 4M (M = mole / l). A current is passed through the tungsten wire, into the solution, and collected by a second electrode (i.e one end of the tungsten serves as the first electrode). As the submerged part of the tungsten etches the first electrode disappears, thus ending the current etching process. Since the etching process of is anisotropic, that is different etching rates in different direction, a very sharp tip is formed at the end of the wire. These tips have typical radii of curvature of about 500 nm [ref.7] (i.e. if we approximate the apex of the tip as a sphere, we can characterize its sharpness by the radius of that sphere).

Tungsten tips work satisfactorily at large scales, however since their characteristic radii are on the order of half a micron (1 micron = 10^{-6} metres = 1 μm), the minimum feature size we can "see" in the lateral direction is roughly about the same size[ref.5,6]. For this and other reasons [ref.5,6] a boron doped diamond tip is now used. These tips are constructed by polishing a diamond, along its

crystalline planes. This forms a triangular apex with a characteristic angle of 60^o. The diamond is then doped with boron, to give it metallic like conduction properties. This diamond apex is then mounted on platinum wire of about the same diameter as the tungsten wire. A typical radius of curvature for this type of SRM tip is 100 nm.

Both the tungsten tip or diamond tip are then spring mounted in a tube a the that has been attached at the end of a V shaped molybdenum cantilever as shown in Figure 5.

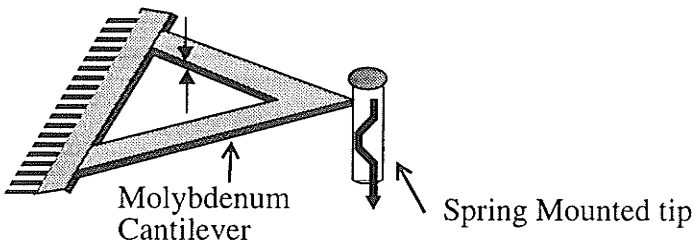


FIGURE 5. SRM Cantilever and probe

Typical parameters for such cantilevers are: spring constant K=15 N/m, and overall resonant frequency f_o = *600 Hz* (i.e. with probe attached to the end of cantilever).

1.5 Deflection Detection Schemes

There are several deflection detection schemes. The more popular two are Optical beam deflection, and Fibre Interferometry detection. Deflection detection schemes are characterized by their resolution, dynamic range, and ease of implementation. The desired characteristics in a deflection detection scheme are: very

high resolution, large dynamic range, good stability, and that it does not interfere with the operation of the cantilever. The first deflection detection scheme used by Binnig on the first AFM, used a tunnelling tip held stationary above the back of a conducting cantilever as shown in Figure 6

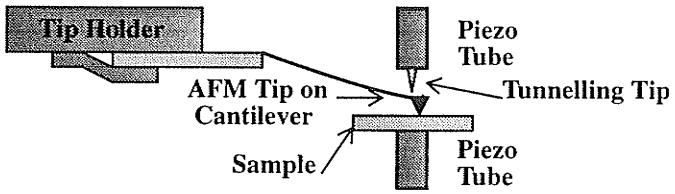


FIGURE 6. Tunnelling current monitor

As the name implies this detection method works on the same principle as the STM where the current tunnelling from the tip to the AFM cantilever is defined for the STM. Since the tunnelling current increases exponentially with decreasing separation, this scheme has the advantage of being very sensitive. Typically the current changes by a factor of 10 for every angstrom of separation between the tunnelling tip and the back of the AFM cantilever. This detection scheme is excellent as far as resolution is concerned, however it suffers from the same disadvantages as the STM. Its performance depends on the quality of the tunnelling current and suffers from the 1/f noise inherent in the current[ref. 9]. Furthermore the tunnelling tip becomes increasingly noisy as they are used. This method also relies on two tips (one tracking the surface, and the other tracking the tip on the

surface), and tunnelling current depends on the local tunnelling properties of these tips. For this reasons most AFMs today use the Optical beam deflection scheme.

The SRM uses a old technology to implement a new function. The SRM uses a capacitance sensor from an RCA^{TM} videodisc player as explained in the next subsection. The other two schemes will be explained in the following subsections.

1.5.1 Deflection Detection with the Capacitance Sensor

The SRM currently uses an RCA^{TM} Capacitance Sensor. The setup is as show in Figure 7

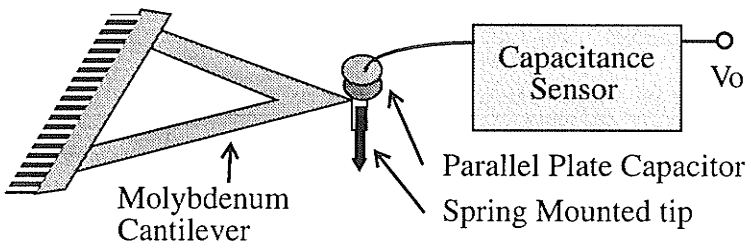


FIGURE 7. Capacitance Sensor Setup

The Capacitance Sensor is a nonlinear device, and produces a characteristic voltage Vo that is roughtly shaped like a bell with varying distance of separation of the parallel plate capacitor. During operation the capacitor is usually biased to one side of the bell curve and the control signal for the feedback is linearized about that point. The Capacitance Sensor scheme is very sensitive to movements of the scanning tip, however it is also very sensitive to extraneous movement, and the current going through the tip, couples into the sensor. The exact nature of this

coupling is not yet know [ref. 10]. These factors tend to make the capacitance sensor very sensitive and somewhat hard to use. Despite all these flaws, the SRM has successfully used this sensor for the past couple of years. One of the advantages of the Capacitance Sensor is that it does not interfere (electrically) with the tip or sample, and it does not require the tip to be conducting.

1.5.2 Optical Beam Deflection Scheme

There are mainly two types of optical deflection methods; optical beam deflection which takes advantage of sensitivity of light to it path direction, and interferometry which makes use of the interference properties of light. The latter will be discussed in the next subsection. The setup for the former scheme is as shown in Figure 8

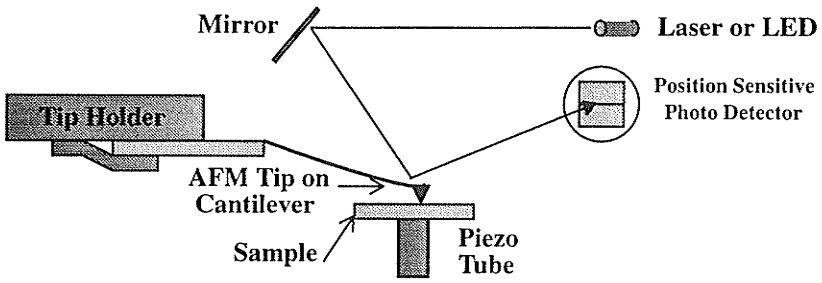


FIGURE 8. Optical Beam Deflection Setup

Both optical deflection methods are capable of resolutions up to $0.1\dot{A}$ Both of these methods are insensitive to the roughness of the cantilever, are relatively insensitive to thermal drifts by comparison with tunnelling, and do not suffer from the 1/f noise inherent in the tunnelling current. Noise due to photons bouncing of

the surface on the cantilever is for most experiments negligible and is on the order of 10^{-19} for a 1 mW of power from the light beam[ref.11].

In the above setup a laser or laser diode is shone at a mirror. This mirror is mounted on a lever that allows it to be positioned on the back of the cantilever. The reflection of the beam bouncing of the tip is then detected by split cell photodetector. Naming the upper cell A and the lower cell B we calculate the signals A+B which tells us if the beam is properly positioned to be detected, and A-B which is then a measure of the deflection of the tip. Before lowering the tip on the sample, the photodetector (Position Sensitive PhotoDetector PSDP) is positioned so that half the beam shines on each cell (i.e. A+B=maximum, A-B=0).

1.5.3 Fibre Interferometer Detection Scheme

Optical beam schemes are very reliable but they require a mirror-like surface on the back of the cantilever, and are not as accurate as the fibre interferometer setup as show below.

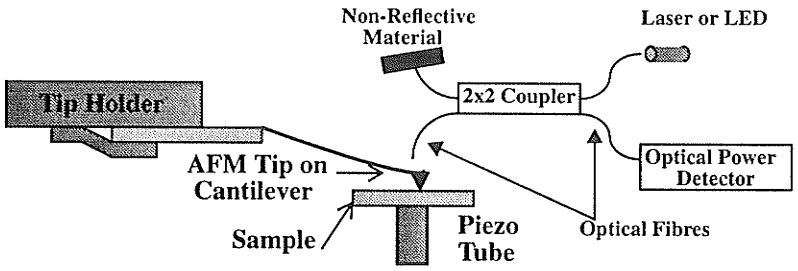


FIGURE 9. **Fibre Interferometer Detection Setup**

In this setup all we need to position on top of the cantilever is a optical fibre. This detection scheme works on the wavelike properties of light. A coherent light source is sent through a 2x2 coupler which splits the beam into 2 of equal power. At the fibre air interface approximately 4% is reflected due to the mismatch in the index of refraction of glass and air (the fibre core is glass). From the light that passes on to air some of it is reflected back into the fibre. The optical power due to the sum of these reflections are detected by the optical power detector. Since the phase of the difference between the reflections varies with the distance of the target (cantilever) from the fibre end, the characteristic function of the optical power detector will be a sum of two sinusoids, which in itself is a sinuosoid with period of $\frac{\lambda}{4}$ in the distance of the fibre to the cantilever. Where λ is the wavelength of the light used and for our fibre interferometer it is 850nm. One of the major drawbacks of this system is that the laser is very sensitive to feedback generated by the interferometer, and leads to noise in the detected signal. Despite this flaw, this system is the preferred one for piezo calibration since the periodic function of the interferometer is very stable.

1.6 Scope of this Theses

The STM, AFM, and SRM are a three examples of a wide variety of scanning probe microscopes. The scope of this thesis project is to present the design considerations and implementation of all digital control system, for scanned probe microscopes, that avoid the flaws of the previous system and builds on its advantages. This control system was built to be adaptable over a wide range of parame-

ters. Doing so required significant hardware and software development, which will be discussed in the subsequent chapters. In order to fully appreciate the advantages of the new system, a brief discussion of the old system will be given in chapter 2, followed by a discussion of the new system in chapter 3. Chapter 4 presents the hardware requirements, design and implementation aspects of the system. To assess the new control system's performance as a whole, software to control both the AFM and SRM was written, and will be discussed on chapter 5. Finally chapter 6 concludes with the conclusions and recommendations on this theses project.

Before proceeding to the discussion on our existing digital AFM controller, it should be mentioned that there are at least two other SPM control systems that attempt to accomplish the same objectives as this thesis projects. One of them is that designed at the California Institute of Technology by S. M. Clark, D. R. Baselt, C. F. Spence, M.G. Youngquist and J. D. Baldeschwieler [ref.19]. In this system the approach is, to build the control electronics as a series of modules that can be interconnected to fit an SPM's particular needs. Some of the performance parameters of this system are comparable to those achieve in our SPM control system.

CHAPTER 2

AFM System Overview

2.1 Chapter Introduction

As stated before the first SPM control systems were entirely analogue. With the advances in Very Large Scale Integration (VLSI) of microprocessor systems, namely DSPs, an all digital controller with enough bandwidth became possible. The advantages of such a system are flexibility and ease of use, and more recently adaptability. Flexibility meant that parameters such as feedback loop gain, or the entire feedback loop algorithm could be replace by a simple software re-write. Ease came as an added bonus of using a DSP to control the system since a computer could be used to control the DSP and thus a simplified user interface was implemented. Adaptability as I define it in this case has to do with using the same controller for many SPM heads, since a wide variety of them require similar control signals.

The SPM instrument can be classified into three main parts:

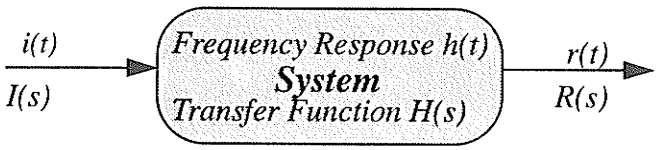
1. The SPM head which contains the sample, cantilever/tip, control signal detection...etc.
2. The control electronics which include the sample rastering, control signal acquisition, control signal feedback...etc.
3. The user interface/data display system, data storage unit

In this chapter we will look at the AFM system as a whole. This will be done in order to evaluate the all-digital control system for the AFM. From this we will extract some of the requirements for the implementation of an all-digital SPM control system.

2.1.1 The PI controller

The Proportional Integral (PI) controller works on the principle of negative feedback. It combines the fast reaction time of the proportional controller, and the zero steady stated error of the integral controller. In order to depict the PI controller in its standard form a brief explanation of the transfer function will be given. consider the following block diagram:

FIGURE 10. **Frequency-Response/Transfer-Function block diagram**



The frequency response is defined as the system output when the system input is the impulse function, or alternatively the output is defined as the convolution of the input with the system frequency response [ref.12].

$$r(t) = \int_{-\infty}^{\infty} h(t-\tau)\, i(t)\, d\tau \qquad \text{(EQ 6)}$$

Where τ is a dummy variable and cancels out when the integration is performed, t is the time parameter as input and output signals are assumed to be functions of time. Alternatively above description of the system can be written as a transfer function, where the transfer function is defined as the Laplace transform of the output over the Laplace transform of the input with the input being equal to zero for t<0. Mathematically this is a follows

$$H(s) = \frac{R(s)}{I(s)} = \frac{\mathcal{L}[r(t)]}{\mathcal{L}[i(t)]} \qquad \text{(EQ 7)}$$

$$for \quad i(t) = 0 \quad for \quad t < 0$$

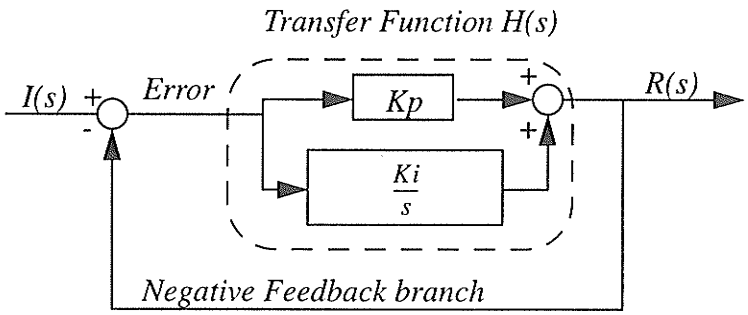
$$where \quad \mathcal{L}[\ldots] = \int_{-\infty}^{\infty} [\ldots] e^{-st} dt$$

Where in the laplace transform $\mathcal{L}[\ldots]$ for our purposes s is defined as

$$s = j\omega \qquad j = \sqrt{-1} \qquad w = 2\pi\,(f = frequency) \qquad \text{(EQ 8)}$$

In the case of a PI controller the block diagram is as shown in Figure 11

FIGURE 11. **PI Controller Block Diagram**



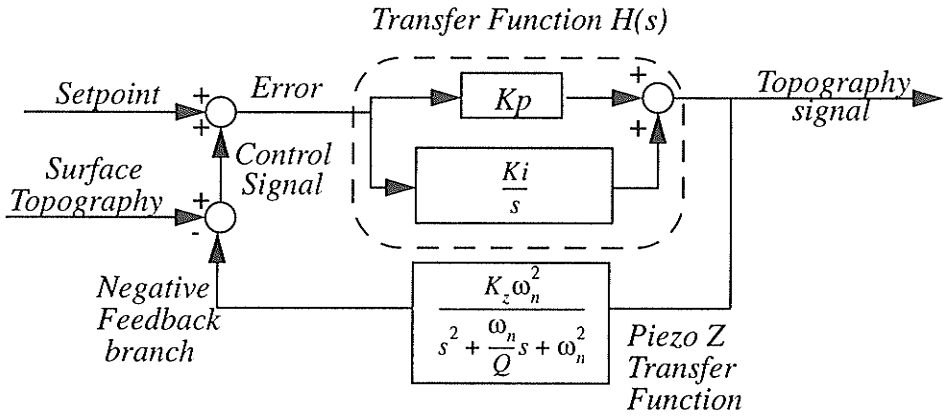
here Kp and Ki are constants and that stand for proportional gain and integral gain. In the proportional branch of the transfer function the error is multiplied by a constant and added to the result on the integral branch. The integral branch can be shown to perform the function of integration of the error, by use of the Laplace transform definition.

The integral function by itself grows unbounded with a constant input. However in the above setup it can be readily seen that this function integrates the error, which is the difference between the output and the input. As the value of the sum increases the error decreases, and hence with time the output of the sum converges to the value of the input. This is one of the desirable features of the integral controller. The Ki constant controls how fast this integral grows, and thus how fast the output tracks the input. However if we keep on increasing Ki the sum would start oscillating and eventually go unstable due to over-feedback.

2.2 The AFM Controller

The PI controller as applied to the AFM is shown on Figure 12

FIGURE 12. AFM Feedback Loop Block Diagram



In SPM systems the PI controller is almost exactly as the one above, with the input signal being a constant setpoint (i.e. in AFM we want constant force in STM constant current is desired). The controlled signal is seen as a disturbance that introduces an error in the loop, the feedback loop compensates trying to minimize this error in order to keep it at the setpoint. The output in this case is the required information, in the AFM and to a lesser extent in the STM this represents the topography of the sample.

In the above diagram the transfer function for the piezo tube along its major axis (Z) was taken from Po Kin Cheng's MSc. E.E thesis [ref.13]. The piezo is modelled as a harmonic oscillator with resonant frequency ω_n and damping factor

Q; Kz is the "DC gain", or the change in length of the piezo due to a DC voltage and has units of meters/Volt.

Technically speaking the control signal is the response of this of control system. Ideally this signal should be equal to the setpoint, in which case we would be tracking the surface perfectly. However we are not interested in this signal, but we are interested in the signal that must be applied to the piezo in order to eliminate the disturbance (surface signal), since this gives us the desired information (surface topography).

In addition to the feedback control system a stepper controller that lowers the tip into contact with the sample, and function generators for the X and Y raster scan signals are considered part of the control signal. The X and Y signals are generated by a standard function generator with an X signal that is 512 times faster then the Y signal. The piezo is assumed to be linear and thus triangular waves are used and amplified to the desired operating voltage. These and the Z voltages usually range in the hundreds of Volts for reasonable movements of the sample.

2.2.1 Image Display System

Images acquired in SPM instruments are not usually images in the sense that we could see them with our eyes, that is to say they have no optical equivalent. In earlier systems the topography information was displayed on an oscilloscope where the vertical axis was the surface height and the horizontal axis was the X raster signal. Putting a group of scan lines together one can map out the equi-

force (AFM) contours forming an image. This can be thought of as successive cross-sections of the surface. An example of a cross-section is shown on Figure 13

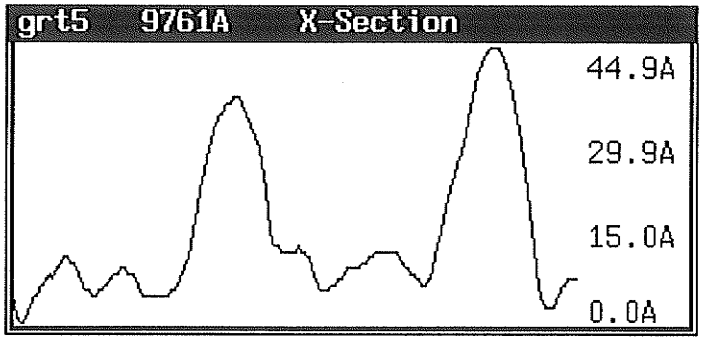


FIGURE 13. AFM Cross-Section: this cross-section corresponds to the image on the following figure.

This format is useful if we want to extract details about the surface, such as surface roughness and actual height dimensions. It also gives an idea of how well we are tracking the surface. It becomes cumbersome however when we want an overall idea of the surface. In this case a greyscale image is much more meaningful. A top down greyscale image approximates looking down on the sample, where light areas are higher and dark areas are lower. Pixels are arranged in a matrix where the

location of each pixel represents X, and Y positions, while its intensity represents the Z height. An example of a greyscale image is shown on Figure 14

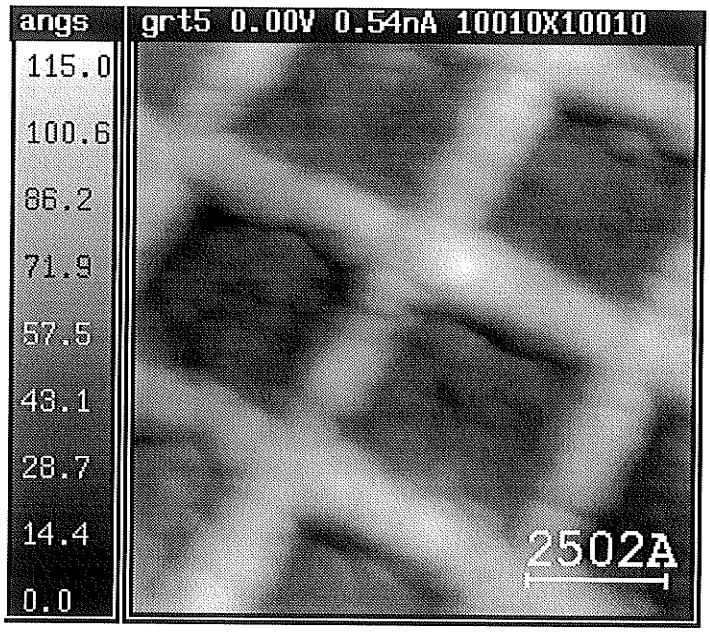


FIGURE 14. Greyscale Image, this is the surface topography of a grating sample. The scan size of the image is 1um by 1um

2.3 The All-Digital AFM System

The all-digital control system in our lab was designed by G.C. McGonigal, K. Yackobski and D.J. Thomson in 1992 [ref.11]. This control system was designed to fit the following criteria:

1. The all digital controller should at least simulate the analogue NanoScope^{TM} I controller [ref.11] used in our lab for the STM and AFM.

2. The system should be easily adapted to new experiments and to take advantages of unforseen new techniques and new equipment. For instance, although the controller is *primarily for the AFM* it should be readily adaptable to the STM with minimal of *software re-writes and hardware changes*.
3. The feedback should be controlled digitally in order to allow the following:

- Greater flexibility since modification would not have to made to the equipment (hardware) for each new experiment. New routines could be added only when they are tested and know to work eliminating down-time of the system.
- Greater flexibility in feedback mechanism, allowing the experimenter to break away from the traditional PI controller if desired, and install a new feedback algorithm through software.
- The ability to raster the tip in any direction trough software. This would be very useful in lining up samples with respect to the scan pattern in order to acquire the best image. Instead of physically rotating the sample on the stage, the user could rotate the nominal scan direction orientation.
- The ability to hold the tip at a given height above the sample as some other variable is changed. This would useful in conducting voltage spectroscopy experiments, where the tip height is held constant while a ramping voltage bias is applied to the tip. In order to carry this out with an analogue feedback mechanism, new circuitry must be added to break the feedback loop and hold the tip at a constant height.

4. The system should provide an image in real time with minimum of hardware.
5. The controller should be able to raster the tip at between 5 and 30 lines per second. Slower speeds are used in the AFM while higher rates are used in the STM.
6. The controller should have a feedback bandwidth of at least 50 kHz for efficiency, stability, and convenience [ref.3]. The high bandwidth is required for reasonable scan speeds, since by scanning the sample we are mapping its topography into the time domain, thus the faster we scan the higher the bandwidth we require for the control system.
7. The proportional and integral gain constants in the feedback loop should be easily set to prevent loop oscillation or underdamping.

In the subsections that follow the means by which most of this criteria was met as well as its implications will be discussed.

2.3.1 Digital Feedback

The current digital AFM controller emulates the analog PI feedback loop. This is done by monitoring the control signal, calculating the function defined by the PI block diagram and the converting the result to the analogue domain. The calculation of the PI algorithm can be written in continuous time as:

$$V_{PI}(t) = K_p V_{error}(t) - \int_0^t K_i V_{error}(t)\,dt \qquad \text{(EQ 9)}$$

where V_{error} is the error voltage in the AFM feedback loop and V_{PI} is the voltage output of the PI transfer function. In the digital domain this can be approximated by using the rectangle approximation to the integral function namely:

$$V_{PI}(t_n) = K_p V_{error}(t_n) + \sum_0^n K_i V_{error}(t_n)\,\Delta t \qquad t_n = n\,(T_s = \Delta t) \qquad n = 0, 1, 2\ldots \text{(EQ 10)}$$

Where we can n is the sample number and the time can be calculated from the product of the sample number with the sample period T_s. In order for the above approximation to mean anything the Nyquist criterion for the sampling theorem must be met. The Nyquist criterion says that in order to reconstruct a sampled signal, the sampling frequency must be greater than twice the frequency of the highest component in the signal [ref.14]. This can be readily seen mathematically if we consider that in order to specify a sinuosoid uniquely we need its amplitude and phase, we can solve for both if we have two or more samples of this signal (i.e. 2 equations, 2 unknowns). Redefining constants we see the simplified PI algorithm in the digital domain as:

$$V_{PI}(t_n) = K_P V_{error}(t_n) + \sum_0^n K_I V_{error}(t_n) \qquad K_I = K_i T_s \qquad \text{(EQ 11)}$$

Which implies that the digital integral gain depends on the sampling frequency while the proportional gain remains unaffected. Hence in order to calculate the analogue equivalent integral gain, K_I must be divided by the sample period.

2.3.2 Digital Rastering of the Sample

The digital rastering of the sample is generated in much the same way as the digital feedback, that is by emulating its analogue counterpart the function generator. In the digital domain a triangular waveform can be described as a counter that increases for half the period of the waveform and then counts back down to its starting point. The "voltage" applied to the piezo in the X direction can then be calculated as:

$$V_x = V_{xoffset} + K_{DAC} COUNTER_X \qquad \text{(EQ 12)}$$

similarly for the Y signal

$$V_y = V_{yoffset} + K_{DAC} COUNTER_X \qquad \text{(EQ 13)}$$

where K_{DAC} is the voltage/bit constant of the DAC used to do the conversion.

2.3.3 The All Digital AFM Hardware

The AFM has a very large dynamic range in the digital domain. Consider that in order to be able to move the sample a couple of microns in steps approaching atomic steps we need a range in the order of 10^5 (2um/0.25 angstroms). This can

only be achieved by a high resolution Digital to Analog Converter (DAC), with a resolution of 16 bits or higher. In order to make use of these DACs a custom analogue I/O board was designed and has a block diagram as shown on Figure 15

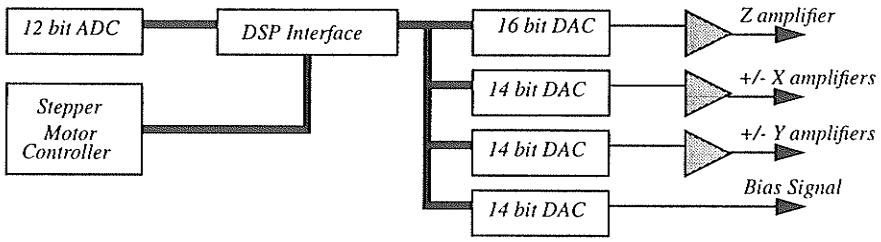


FIGURE 15. AFM Analogue I/O Board

The range for the ADC is 12 bits, with a range of +/- 5V input and digital range of 2^{12} that is it can resolve the input range in 4096 steps. The Z output DAC is amplified to a range of -250V+120V, in a digital range of 2^{16} or 65536. The X and Y DACs are both 14 bits leading to a digital range of 16384 in a voltage range of 0 to 320V. The non-symmetrical voltages for both X, and Y as well as Z are due to the fact that Z electrode on the Piezo tube cannot be less than 150V relative to the any of its outside electrodes (X,-X,Y,-Y). The Bias signal DAC is a spare in the AFM and was intended for use in biasing the tip in the STM. This PCB board was designed and constructed by Gord Mcgonigal[ref. 27].

The analogue I/O board is then interfaced to a Motorola^{TM} DSP56001 development board via a 96 pin connector. The DSP is interface to a PC via an ISA bus prototype card. The hardware for the AFM controller is a shown on Figure 16

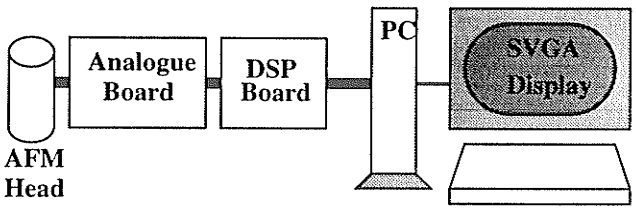


FIGURE 16. All Digital AFM Hardware Block Diagram

The relevant parameters for the Digital AFM controller are as follows:

Z distance range,

$$Z_{MAX} = \left(8.2\frac{nm}{v}\right)(120V - (-250V)) = 3.03um$$

Z resolution :

$$Z_{MIN} = \frac{3.03um}{2^{16}} = 0.46\frac{\dot{A}}{bit}$$

similarly,

$$X_{MAX} = Y_{MAX} = 7.68um \qquad X_{MAX} = Y_{MAX} = 0.47\frac{nm}{bit}$$

The system sample period is 30.5 μs, giving it a bandwidth of:

$$B = \frac{1}{2T_s} = \frac{1}{2\,(30.5\mu s)} = 16.4kHz$$

All parameters of this system are digitally controlled by the host computer. The program runs reasonably fast on 386DX IBM compatible PC, and displays all the

required data. This system has numerous advantages over an analogue system. Feedback bandwidth however fell short from the required of 50kHz. This requirement was for the STM and to date this system has only been used on the AFM, which has a piezo resonant frequency of aproximately 16kHz [ref.11], making the bandwidth suitable for the AFM.

2.4 Chapter Summary

The all digital AFM controller is a great step ahead of analogue controllers. It fulfils most of the criteria set out to accomplish. However it falls short of ideal as a generic controller that could be adapted to a wide range of SPMs. One of the main obstacles keeping it un-adaptable is the analogue I/O hardware. Other drawbacks include noise coupling from the digital signals into the analogue signals being controlled. These and other factors have dictated that a new controller that avoids this flaws must be designed and constructed. The criteria and system overview for the design of such a system is the topic of the following chapter.

CHAPTER 3

SPM System Overview

3.1 Chapter Introduction

The advantages of the all digital AFM lie in its flexibility. First, we have the flexibility to control all the feedback loop parameters, scan rate, & scan size with point and click ease through use of the mouse on the IBM PC interface. However, this flexibility ends when we want to use this controller for something like the STM and even more so, for the SRM. In the case of the STM, the controller could be adapted with some software rewrites, and minor hardware changes. With the SRM however this is not true since we are short one Analogue to Digital Converter (ADC) for the resistance channel, plus some major software rewrites. In this chapter we will set the criteria and design parameters for the a generic digital SPM controller. We will do this by building on the strengths of the AFM controller, and generalizing them for the majority of SPMs. We will also attempt to improve on

the performance parameters of the system, such as noise performance, and data sampling/quantizing dynamic range.

3.2 The SPM Instrument

Consider the previous classification of the three main parts of an SPM system:

1. The SPM head which contains the sample, cantilever/tip, control signal detection..etc.
2. The Control Electronics which include the sample rastering, control signal acquisition, control signal feedback...etc.
3. The user interface/data display system, data storage unit

We have little control over the first unit which is the SPM head. This is the part of the microscope that defines what type it is (i.e. AFM, STM, SRM,.etc.). This also defines the input and output signals of the microscope, and for most cases are entirely analogue. Indentifying the differences and similarities between this part of the SPM system dictates the viability of a generic controller.

The second item on the list can be further broken down into three parts

- The analog I/O board that translates analogue signals into the digital domain and vice-versa
- The DSP board that takes care the "number crunching", such as implementing the PI controller and generating the X,Y raster signals
- The software that calculates the actual parameters, and interfaces to the host computer

This item is the first one to be modified in order to adapt to the number of signals that we need to sample and control. The DSP itself must be taken into consideration as a more powerful DSP will allow for both faster and more accurate feedback. The software of course is re-writable, however we will like to keep rewrites to a minimum, keeping the controller's generality as long as possible.

3.2.1 Some Differences and Similarities in SPM Heads

Consider the AFM to be the basic SPM, this implies that the control functions of the AFM are a subset of those for the SPM. The basic signals in the AFM are:

1. Positioning of the sample through a PZT (inputs to piezo +/- X, +/-Y, and Z).
2. The control signal or deflection of the cantilever probe (output).
3. Stepper motor controller(signals) to load the tip on the sample (output).

In the STM control signal changes slightly in meaning, but are still related to the distance of separation of the probe and sample. Since here we don't have a probe deflection signal we shall consider the tunnelling current to take its place. The STM does however add another signal which is:

4. The tip bias voltage since in the STM the control signal is the tunnelling current (output).

The SRM brings back the same signals as the AFM. Since here the tip is also conductive we can use the same output as used in the STM for the bias voltage. The following signal still remains unaccounted for:

5. The current signal that dictates how much resistance there is between the probe and ground (i.e. tip-sample resistance).

It can be concluded from this that there seems to be an increasing need to be able to look at, and control more signals digitally. In other words bring more signals into the digital domain and control more signals from the digital domain. This is of course as we expected, since the more data converters we have available the more flexible our system will be.

3.2.2 The Control Electronics

The previous subsection suggests that the more data converters we have at our disposal, the more likely it will be that our system will adapt to a particular type of SPM. However we haven't yet taken into account dyanmic range for all these signals. The conclusion here is that a larger dynamic range will allow more accurate represention of output and input signals. Therefore this parameter should be as least as high as practical with a minimum equivalent to those of the current AFM system controller. The digital bandwidth or number of voltage levels that we can represent in the digital domain is a combination of the DSP word length and dynamic range of the data converters, the latter being the practical limiting factor. The sampling rate is also a combination of the DSP program execution speed, plus sample and update rates for the data converters. The general trend in VLSI seems to be an increase in both, with processor speed increasing at a faster rate, therefore the analogue I/O board should be designed with this aspect in mind. That is if possible a board that can interface to a wide range of DSPs is desirable.

The PC in the system serves mostly as a user interface as well as an image display system with limited image processing options such as image slope subtraction. The existing AFM system runs on a 386DX-25 PC with a video card capable of 800x600 pixels of resolution. The interface from the DSP to the PC is through a ISA (PC bus standard for 8 or 16 bit expansion of PC peripherals) 8 bit prototype card and is connected through the DSP host interface. This proved to be adequate for the AFM controller, however there are several factors that can be improved on through the purchase of a new computer. For example, the displaying

of two images would take twice as long if the same scheme could be kept, or the same speed could be kept by upgrading to a machine which is twice as fast, not taking into account other factors.

3.2.3 The PC and DSP Software

The software for the DSP was written to handle the AFM only with the hope that it would be easily modifiable for other SPMs. Some of this code is written in the C language while some of it is written in DSP56000 assembly language. Since there is no documentation for this code, and in order to modify this code one must have a good understanding of both C and DSP56000 assembly language, I opted to use DSP56000 assembly language entirely. Although this implies that in order to modify the code one must be highly familiar with the DSP system and I/O hardware, this is also the case when writing in C, but now we also require an understanding of C. Therefore no real gain is obtained by writing some of the software in C since in order to have more control on I/0 and low level registers some assembly language is still required.

The software for the PC was written for MS-DOS^{TM} using the Borland^{TM} Pascal version 6.0 compiler. In the AFM only one image is obtained at the time. The scheme adopted here was that the were two images in memory[ref.11]. One image is the previous image obtained, while the "current" image is the one in the process of being obtained. This scheme ensured that one would not have to stop scanning to save an image. A feature that is desirable considering that the piezo suffers from drifts overtime[ref.15], plus hysteresis. In this scheme drift can be considered

negligible[ref.15] within certain period of time, and the piezo although hysteretive follow approximately the same path in the X raster direction.

In the SRM for example the acquisition of two images simultaneously is desirable. However, the DOS operating system does not allow any array in memory (our images are stored as a matrix of 256 arrays of 256 16-bit elements) to be bigger that 64K. This was overcome in the AFM by using 256 arrays of 256, 16-bit elements. where each array is 512 bytes (2 bytes/element * 256 elements), for a total memory of 128kbytes (256rows * 512 bytes /row = 128 (*1024=k)bytes) per image. In order for the AFM program to run it needed 256kbytes for the images alone (always two images in memory). Trying to use this same scheme for two actual images (i.e. four in program memory) caused the program to be unstable(work sometimes) due to the fact that in the MS-DOS™ operating system one cannot directly access more that 640K of memory. This also meant that even if we could finally debug this software to run under this scheme, expanding it to acquire 3 actual images simultaneously would be impossible since the required memory (3 actual images * 2 images in memory * 128 kilobytes = 768 kilobytes) is greater that the MS-DOS™ 640K limit.

In order to avoid the MS-DOS™ 640Kbytes limit we purchased a new copy of the Borland™ Pascal version 7.0 which allows us to access up to 16 Mbytes of memory [ref.16]. This ensured that most of the code would be portable, while future expansion of the system would not be limited by the memory model.

3.3 The All Digital SPM

The all digital SPM control system was designed to meet the criteria set for the AFM, plus the following:

1. New hardware should be designed to avoid the limitations of the old hardware system. This include the following:
   - The new system should have enough bandwidth to be used with the STM. This implies that the DACs should be at least 18 bits to have a reasonable field of view while maintaining resolution.
   - The input signal bandwidth should also be increased to allow more accurate quantization. At least 14 bits of resolution were set as the lower limit, increasing the resolution by 2 bits or a factor of 4.
   - The new I/O hardware should be able to work with the current DSP56000 and adapt to others as required, with little or no hardware changes. One such DSP should be the DSP96000 also from Motorola^{TM}.
   - The hardware system noise should be brought down as much as possible, to make it negligible compared to the actual image signal noise. We don't want to add noise to the images through our control system.
   - The new analog I/O hardware board should have enough input and output channels to meet unforeseen needs. The possibility of using multiple boards in the same system should be considered.
2. New software should be designed to control not only the AFM but also the STM, and SRM without need for any modifications. This includes the ability to acquire two images at the same time, and allow for higher resolutions such as 18 bits or higher.

3.4 Chapter Summary

The requirements for hardware are extensive, as such a significant amount of time was dedicated to the design of a printed circuit board through which all the I/O requirements were met was designed and built. The design considerations, printed circuit board (PCB) layout process to the final product are discussed in the next chapter. A discussion on the new system hardware as well as its advantages over the old system and limitations is also given.

CHAPTER 4

Hardware for Digitally Controlled SPMs

4.1 Chapter Introduction

The SPM hardware system is composed of the SPM head, analogue I/O board, DSP board, and host computer. It was clear at the beginning of this thesis project that the analogue I/O board had to be replaced to allow for more I/O channels than were available in the AFM analogue I/O board. There are no off-the-shelf data acquisition boards that meet our requirements, however recent advances in digital audio technology allow for the construction of very high performance analog to digital I/O boards at a relatively low cost. The design and construction of a new data acquisition board also allowed us to use the latest technology, such as field programmable gate arrays (FPGAs) which are basically programmable hardware. The use of such devices would allow for different DSPs to be interfaced to the same I/O board, through the use of a simple hardware reconfiguration (done by software). Other factors such as the electronic noise inherent in the data acquisi-

tion electronics are also improved on in this design. The selection of the major components in the I/O board, design process, testing, trouble-shooting, and debugging of the hardware system are presented in this chapter.

4.2 General Specifications for the Analogue Input/Output Board

The block diagram for the I/O board is shown in Figure 17

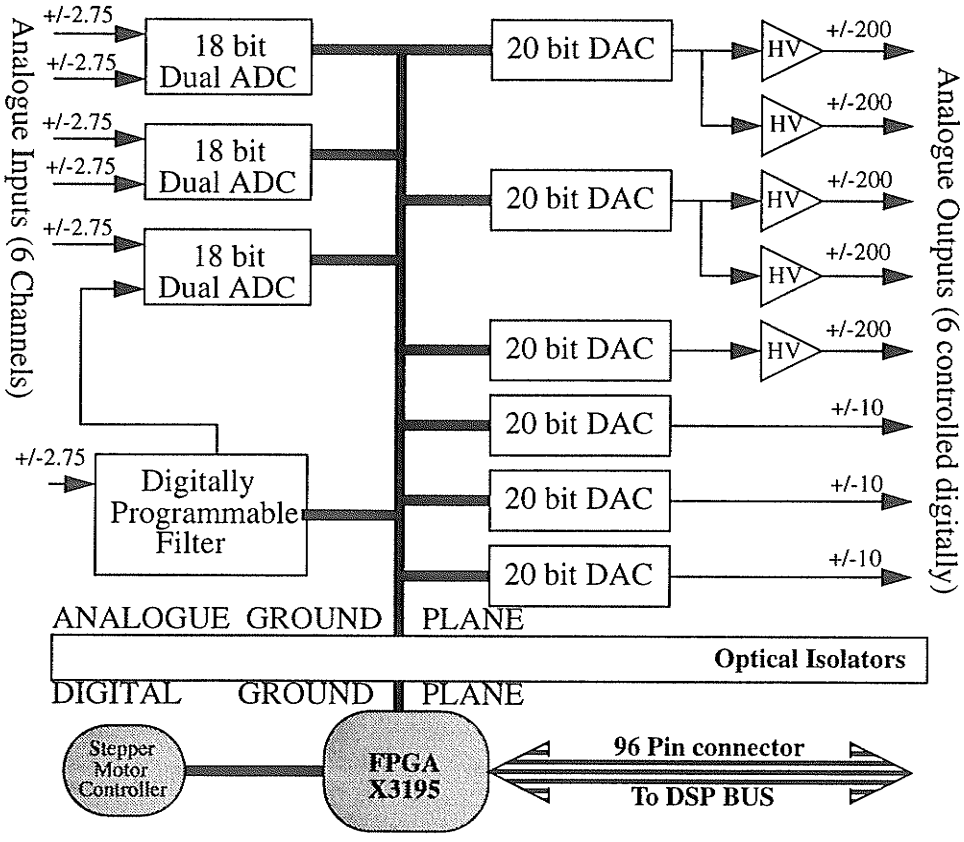


FIGURE 17. Block diagram of the analogue I/O board

The block diagram depicts the general specifications for the analogue I/O board. Three digital output channels (DACs) are required to control the sample positioner PZT. In the STM, and SRM an extra channel is required to set the bias voltage on the conducting tip. The two extra channels are left as spares for future developments and to keep flexibility of the I/O board. The PZT requires relatively high voltage signals for very small movements. The voltage range of the amplifiers for the +X, -X, +Y, -Y and Z signals is 400V. All five amplifiers are incorporated on board. In view of future developments, where the outputs may be controlled through closed loop feedback, or simply for greater flexibility, the number of input channels was made the same as the number of outputs (i.e. six). Although currently even in the SRM, where two images are acquired simoultaneously, only two input channels are required. The Field Programmable Gate Array (FPGA) was a solution to a few problems. One of them being that since the FPGA is basically programmable hardware, compatibility across DSPs could be obtained simply by implementing the DSP interface in the FPGA. One of the main sources of noise in the previous AFM I/O board is the PC. This is because the PC ground plane is connected to the I/O board ground, through which the digital switching signals on the PC couple. Isolation of this two planes was achieved trough the use of optocouplers. Here all digital signals are coupled optically into the analogue side of the board, and hence there is no electrical connection between the two planes.

4.2.1 The Digital to Analogue Converters

With the tremendous advances in digital audio technology, there is a wide range of data converters with resolutions greater than 16 bits. The two main factors governing our choice of DAC are the resolution (number of bits) and update rate. Another important factor is of course price. We found that prices for DACs range from a couple of dollars to a few thousand depending on the specifications. Our choice of DAC is the PCM63P from Burr Brown^{TM}, this DAC has the following specifications as specified by the manufacturer:

Part No.	**PCM63P**
Resolutilion	20 bits
THD+N	-96 dB
min SNR	116 dB
Settling Time	200 ns
Idle Channel Signal to Noise Ratio	120 dB typical, 116 dB min
Output Format	Current, +/- 2 mA
Input Format	Serial,2s complement, 25MHz max rate
Glitch Energy	No Glitch around Zero

TABLE 1. Specifications for the PCM63P Digital to Analogue Converter

These Specifications and more detailed parameters are found in the manufactures data books [ref.17]. A 20 bit resolution yields a quantization range of 120 dB, at an update rate of

$$\frac{25MHz}{20bits} = 1.25MHz$$

which is the maximum serial input data clock divided by the number of input bits, and the output voltage is guaranteed to settle within 200ns (maximum update rate is 800ns). This part exceeds our requirements of an update rate of 100kHz (50kHz

bandwitdth on the control feedback loop) by a factor of 12, with 20 bits of attainable accuracy with proper design layout.

4.2.2 The Analogue to Digital Converters

Since digital to analogue converters are also part of the digital audio components family, they have also benefited from the growth of this industry. Our requirements for an ADC was that of a 12 bit minimum, 14 preferably, with a bandwidth of at least 100kHz. Here we again chose a part from Burr Brown^{TM}, the ADC has the following s specifications as specified by the manufacturer:

Part No.	PCM1750P
Resolutilion	18 bits
THD+N	-88 dB without adjustments
Conversion time + Sample & Hold	4.5us min
Channel Signal to Noise Ratio	108 dB typical, 88 dB min
Input Format	Voltage, +/- 2.75 V
Output Format	Serial,2s complement, 5MHz max rate

TABLE 2. Specifications for the PCM1750P Analogue to Digital Converter

More detailed parameters are found in the manufactures data books [ref.18]. An 18 bit resolution yields a quantization range of 108 dB, at a maximum sample rate of

$$\frac{1}{4.5us} = 222kHz$$

which is simply the reciprocal of the minimum conversion plus sample and hold period. Our previous ADC had a minimum conversion time of 6.0us, making our

choice of ADC well suited for the application. The dynamic range again exceeds our requirements for speed by a factor of 2, and has 4 extra bits of resolution.

4.2.3 The High Voltage Amplifiers

The selection of the high voltage amplifiers is somewhat simpler, due to the narrow field of parts to choose from. Component level high voltage amplifiers are available from primarily two companies, Apex Microtechnology^{TM} and Burr Brown^{TM}. The three main characteristics in evaluating an op-amp's suitability for use in SPM are, (i) total output voltage swing, (ii) output noise, and (iii) current drive capacity. The current capability of the op-amp is an important consideration because of the capacitive load presented by the piezo ceramics [ref. 19]. Our choice of high voltage op-amp is the PA88 from Apex Microtechnology^{TM}. This is the same as the one used in the previous controller, and proved to be quite satisfactory. The choice for this type of parts has not changed much since the last I/O board was constructed. A few specifications for this part are given in the following table [ref.20]:

Part No.	PA88
Output Voltage Swing	440 V
Max Current Output	100 mA
Continuous Power Out	15 W
Output Noise/Closed Loop Gain	2 uVrms

TABLE 3. Specifications for the Apex PA88 High Voltage Op-Amp

A voltage swing of 440V is quite suitable for our SPMs. For the AFM for example this gives a field of view of 10.6 um (440 V* 240 Angstrom/V).

4.2.4 Digitally Programmable Analogue Filter

One of the difficulties of designing such a data acquisition board is that the proper design of the anti-aliasing filters is difficult if not impossible, due to the broad range of signals to be sampled. The spectrum of the signals is not only dependent on the type of instrument being controlled, but also on the type of sample under observation. In the AFM data acquisition board for example no anti-aliasing filter was used. This is due to the fact that the AFM cantilever response is bandlimited, and well under the Nyquist sampling requirement. For these reasons we opted to not include anti-aliasing filters in all the channels. Instead we included a digitally programmable analogue filter from Frequency Devices^{TM} on input channel six. Unlike switched capacitor filters which are inherently digital and thus still require an analogue filter, this filter is a an active analogue low-pass filter. The type of filter as well as cut-off frequency and order can be custom ordered. We chose a filter with the following specifications [ref.21]:

Part No.	**844P8L-5**
Programmable Frequency Range	200Hz - 51.2 kHz
Number of Programming Steps	256
Input Digital Word	8-bit Unsigned
Filter Type & Order	Low Pass, Linear Phase, 4th order
Typical Noise	50 uV (5 Hz to 50 kHz)

TABLE 4. Specifications for the Frequency Devices^{TM} Digitally Programmable Analogue Filter 844P8L-5

The offset voltage of this filter is 2mV and is variable with the cut-off frequency. The noise is also variable with cut-off frequency, and is below 50 uV.

4.2.5 The Field Programmable Gate Array & Opto-isolators

In order to minimize noise coupling from the digital ground plane into the analogue ground plane, optoisolators were used. Our choice of opto-isolator is the HPCL-2430, from Hewlett Packard™ [ref.22]. It has a maximum guaranteed switching specification of 20 MHz data rate and is tested in our analogue I/O board up to 25 MHz (max. data rate for the DAC).

As mentioned earlier our choice for a FPGA is due to several reasons. The reprogrammable nature of the XILINX™ FPGAs allows the logic interface to be reprogrammable, thus allowing the interface of several DSP to the same I/O chips. In fact, since the DSP96000 from Motorola™ used the same expansion port connector type (different pin layout) as the DSP56000 (a 96 pin euro-card connector) a simple rerouting of the Xilinx pins should adapt the hardware for use with this DSP. In order to minimize the electrical lines to be optically isolated, serial load and serial output DACs and ADC respectively were chosen. This implied that since the DSP is a parallel word (all bits load at the same time) processor, dedicated hardware must be designed to do the serial-to-parallel and parallel-to-serial conversion, or the task could be switched onto the DSP making the software more complex, and of course longer. The FPGA could be programmed to make the serial nature of the I/O device transparent to the DSP. Taking these into account a XC3090-50 was chosen. A more detailed explanation of the selection process, "troubleshooting", and programming of this part is given in Section 4.4, "Configuring The Field Programmable Gate Array," on page 51. Schematics for the analogue I/O board are given in Appendix A.

4.3 The Analogue I/O Board Printed Circuit Design

One of the objectives of designing a new data acquisition board was to improve on the noise characteristics of the previous AFM I/O board. The current I/O board uses optoisolators to physically isolate the digital and analogue planes. The particular layout of the components themselves is optimized to avoid noise coupling into the signals of interest. For example, the lines that carry the input signal to the Digital to Analogue converter are buried in a ground plane to shield it from the surrounding signals. The board was designed as a four layer board, (i)to make the physical size of the board a minimum (in fact the new I/O board has the same footprint as the old board, 9"by 12"), (ii) to maintain a "solid" ground plane (one of the layers is solid ground), and (iii) to ensure proper power supply to all integrated circuits by a placing power plane interrupted only by other power lines with different voltage supplies for analogue circuitry such as the high voltage amplifiers. All signal routes are placed on the outer layers of the board, such that any trace can be cut if the need should arise.

The design was layed out for testability, ensuring that access to signals at different stages in a circuit could be obtained. In this area the FPGA proved to be very useful, since internal or in some case external nodes in the circuit could be brought to one of the spare pins of the FPGA. There are 12 spare I/O pins on the XILINX for such purposes.

The schematic capture was done using the Orcad^{TM} Schematic Design Tools (SDT) [ref.23]. The interconnection of all components was then exported into Cadstar^{TM} Printed Circuit Board Tools [ref.24]. Importing the connectivity

information into a PCB package minimizes errors, since all we need to worry about how to layout traces on the board, and not about actual pin layout (once layout of a part is specified into the component library) of the parts. The Analogue I/O board in place with the DSP board is shown in Figure 18

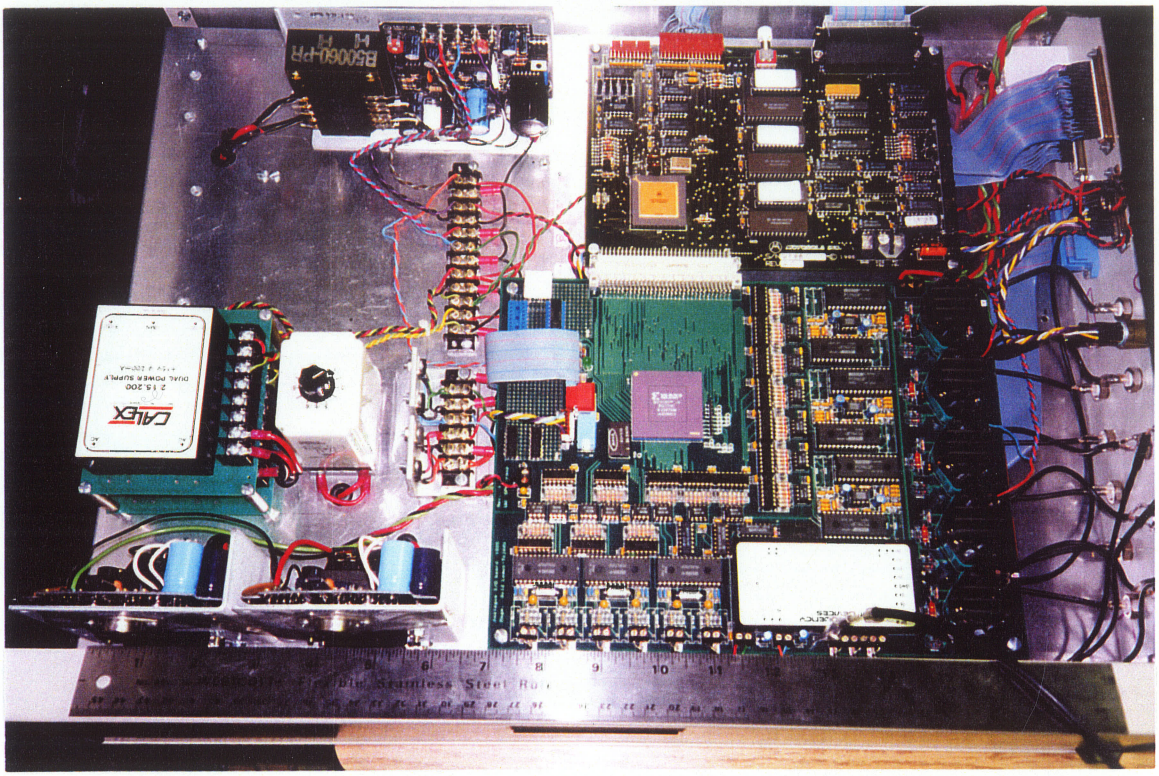


FIGURE 18. The Analogue I/O Printed Circuit Board and DSP56000 Board

All the plots required to construct the different layers of the PCB can be found in Appendix B.

4.4 Configuring The Field Programmable Gate Array

A block diagram for the FPGA configuration circuit is show in Figure 19

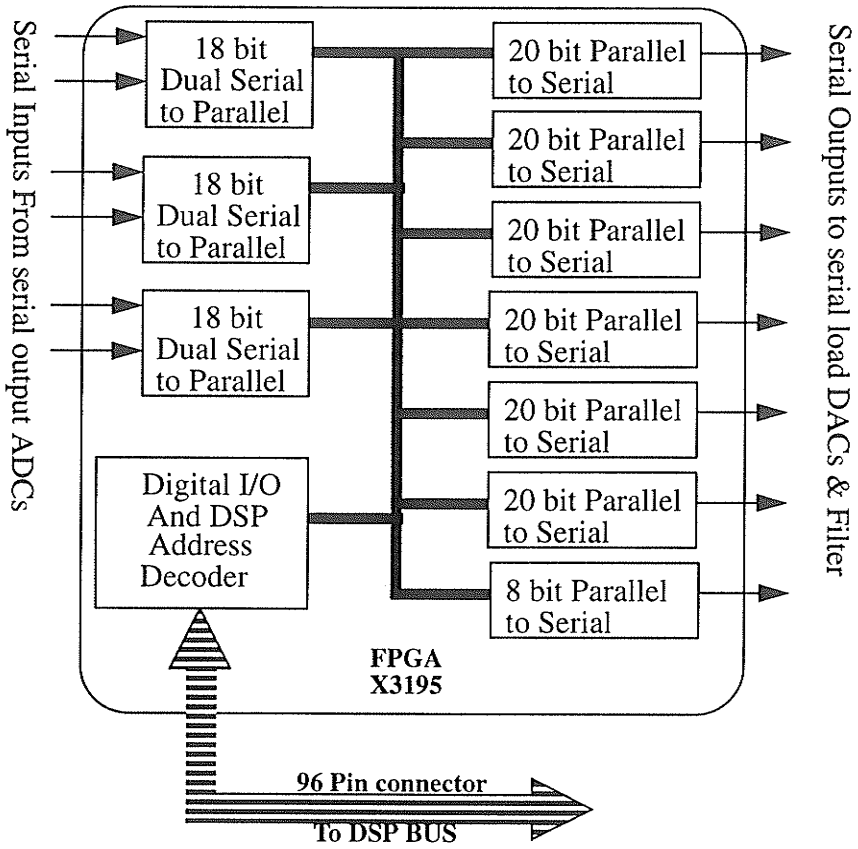


FIGURE 19. Block Diagram of XC3195 Field Programmable Gate Array Configuration

The selection of the FPGA size and speed is based on the amount of circuitry to be implemented inside it. The size of a XILINX FPGA is specified as the number

of logic blocks available for use in that part. Each one of this blocks can implement any boolean function of five variables [ref.25], and latch a maximum of two. The speed for the part is specified as the maximum toggle rate of any of its internal states. The determination of overall performance speed is not that simple, as the frequency at which we toggle a single output depends on how many devices it drives and how far apart devices are physically located (length of the electrical path).

However a rough idea can be obtained by analysing the individual digital circuits to be configured in the Xilinx FPGA. Actual size and speed cannot be determined until the digital circuit is mapped into the actual device and simulated/tested. For example for the block diagram presented for the FPGA, assuming a that:

1. one logic block is used for every bit in the 20 bit parallel to serial register
2. one logic block is used for a state machine with 20 states that shifts the bits serially to the serial load DAC (5 blocks in total since we can represent 16 states with 4 bits and 32 with 5bits)

The number of programable logic blocks on the XILINX FPGA required for one sequencer (20 bit Serial to Parallel Converter) is 25. Similarly by adding all we obtain 23 for the Parallel to Serial converters (ADC are serial output), and 11 for the digitally programmable filter (although the filter is parallel-load a serial--to-parallel was added in the analogue side of the board to make it serial-load, making the number of lines to be optically isolated 3 instead of 9). The number of blocks we require in a part to implement the sequencing alone is then:

$$6\left(25\frac{blocks}{DACsequencer}+23\frac{blocks}{ADCsequencer}\right)+9\frac{blocks}{FilterSequencer}=299LogicBlocks$$

This is of course a worst case estimate and does not include the I/O and address decoding. Assuming 2 blocks for multiplexing each bit of the six channels and 16 separate address to addresses the 13 devices in the Xilinx (6 inputs ADCs, 6 Output DACs, and one output filter sequencer), this can be implemented in 7 logic blocks (maximum of two outputs/logic block), we obtain the following figure for the blocks required to implement the I/O and address decoding in the FPGA:

$$18\left(2\frac{blocks}{ADCmultiplexer}\right)+7\frac{blocks}{I/ODecoder} = 43LogicBlocks$$

This brings the total to 342 Logic blocks. We choose to go with the XC3090-50 which due to its availability (we had a few in stock). This part contains 320 logic blocks. Although it seems like we are short by 22 blocks (342-320=22), we will not know this for sure until the actual circuits are mapped onto a specific device. The toggle rate for this device is 50 MHz leading to system speeds of about 20MHz (the DSP runs at 20 MHz) [ref.26].

4.4.1 Problems Encountered in the FPGA configuration Stage

There were no problems encountered with the configuration process itself. The design configuration process of the FPGA was carried out at the same pace at which the board was populated. As more parts were added more processes were configured in the FPGA. The first stage was configuring the FPGA itself with a test configuration (a feedthrough from one pin to another), then the clock was routed to all the opto-isolator circuits. The opto-isolator circuits were found to work satisfactorily at speed at or below 25 MHz, since the maximum serial transfer rates in the board are 25MHz this proved to be ideal.

The first sequencer implemented was that of the digitally programmable filter (DPF). This is a simple 8 state machine with a parallel-in serial-out register. Once a word is written to this register, the write process activates an 8 bit counter that enables the serial output of the register. This continues until all eight bits are shifted out. One of the problems with this scheme was that there was a delay in the clock signal as compared to the data signal, as a result the wrong data was receive by the DPF. This was easily solve by latching the data in the opposite edge of the clock used to shift it. This allowed enough time for the data to stabilize and since the data was now latched at exactly halfway of its valid period, differences in relative delays of data an clock are not important.

The following step was to get the DAC sequencer to work. This circuit is essentially the same as that for the DPF with the shift register being 20 bits and a 20 state counter respectively. Although the operation of this sequencer is identical to that of the DPF sequencer, the exact same setup proved problematic. In order to minimize switching signals in the analogue side of the board, the clock was turned off once the parallel-to-serial cycle ended. With this scheme the DAC missed the first two most significant bits. Adding an extra half a clock cycle after the download cycle has solved the problem. Which although not specified by the manufacturer lead me to conclude that the last clock edge as well as the convert signals are used to convert the digital word to analogue.

As indicated by our initial estimate, 320 logic blocks to implement the sequencers plus I/O decoders proved to be to small. Although with some optimization it can be done, the dense routing of the circuit and long lines slow the

system clock down to about 7MHz which is to slow for our application. To avoid this, the FPGA was upgraded to an XC3195 with a toggle rate of 270 MHz and 484 logic blocks. As a result there are over a hundred unused logic blocks in the FPGA for any future developments.

The Analogue I/O board sequencers run independently of the DSP clock. This lead to another complication which was not anticipated. Since the two clocks run independent of each other, the read/write cycles of the DSP could end either at a low phase or high phase of the I/O board clock. The sequencers are set to work on the falling edge of the read/write signal. In one of the two cases there would be a missing edge in sequencers output, causing the output of the DAC to be divided by two (a shift of the binary point in a binary number has the effect of dividing it by 2). The division by two effect, would appear random as the clocks came in and out of phase (since the clocks were not synchronous). Once the problem was identified, it was easily solved by latching in all data coming into the xilinx at the rising edge of the I/O board clock. This way all data cycles would end synchronous to the on board clock and always at the same clock phase.

The last major problem found was that of "spikes" in the data being read from the ADC's. These high frequency glitches are not present in the analogue signal being sampled, and as such must occur in the reading of the ADC output. Adding more supply current seems to lessen the number of glitches present in the output. As such, all signals that are grounded and set high in the 96-pin-connector through the DSP were designated ground and power respectively in the FPGA. This ensured that no power is being wasted trying to pull these pins high through pull-

up resistors. This scheme eliminated all the "spikes" in input channel 6, however some still remained in channel 5. For this reason channel four is used for the resistance information from the SRM and channel six is used for the topography signal. The rest of the channels (1,2,3) have been tested but currently are unused.

4.5 General I/O Performance Parameters of the Hardware System

The performance parameters of interest for the system are directly related to the performance of the data acquisition board. The time for the execution of a single feedback cycle is also an important parameter, however for the purposes of this thesis, this parameter will be considered as a software specification since it is highly dependent on the number of instructions the DSP performs per feedback cycle. The following table summarizes the performance parameters of the Analogue I/O board:

Part No.	SPM Analogue I/O board
DSP Interface	96 pin Programmable, Configure to Interface with DSP56000, Connector Directly compatible with DSP96000 with pin layout reprogramming.
Number of Input Channels	3 Duals for total of 6 with 18 bit resolution, Input range +/- 2.75 V
Maximum Sample Rate	156kHz which can be sped up to 222kHz
Number Of Output Channels	6 at 20 bit resolution, Output range +/-10V, 3 amplified to +/-200, 2 complemented to -/+ 200 (for +X, -X, Y, -Y and Z)
Maximum Update Rate	1MHz which can be sped up to 1.25MHz
Typical Noise in Input Channels	100 uV rms

TABLE 5. Specifications for the SPM Data Analogue I/O Board

Part No.	SPM Analogue I/O board
Typical Noise in Output Channels	900 uV rms in +/- 200 range 400 uV rms in +/- 10 range
Typical Noise in Input Channels	100 uV rms
Input Channel Useful Range	16 bits
Output Channel Useful Range	+/-200V range 19 bits +/-10V range 16 bits

TABLE 5. Specifications for the SPM Data Analogue I/O Board

The useful range for the +X channel was found as:

$$UsefulRange = log_2 \frac{400V}{900uVrms} = 18.76 \cong 19bits$$

This can be interpreted to mean that if we had an 19 bit DAC in a +/- 200 V range, the minimun step would be

$$MinimunStep = \frac{VoltageRange}{WordRange} = \frac{200V - (-200V)}{2^{19}} = 0.76mV$$

Which is on the same order as the average noise, and therefore would be the uncertainty bit. Anything bigger than one the 19-bit, will on average not be masked out by the noise floor. The useful range for the other channels was calculated in the same manner.

The performance of the noise parameters for the board were measured with all the system assembled (I/O board connected to DSP board, which is in turn connected the PC) as it would be in a typical working environment.

The SPM control system uses the same DSP56000 processor, and is interfaced in the same way to the ISA bus of a PC. The PC was upgraded to allow the aquisition of two images at the same time at reasonble scan rates. We now use a 486DX2-66 based PC as oppossed to 386DX-25. The following figure depicts the

SPM control sytem. Access to all channels is provided through coaxial connectors mounted on the front of the white case in left side of the computer monitor. A dedicated AFM connector is also supplied to allow quick setup of the AFM.

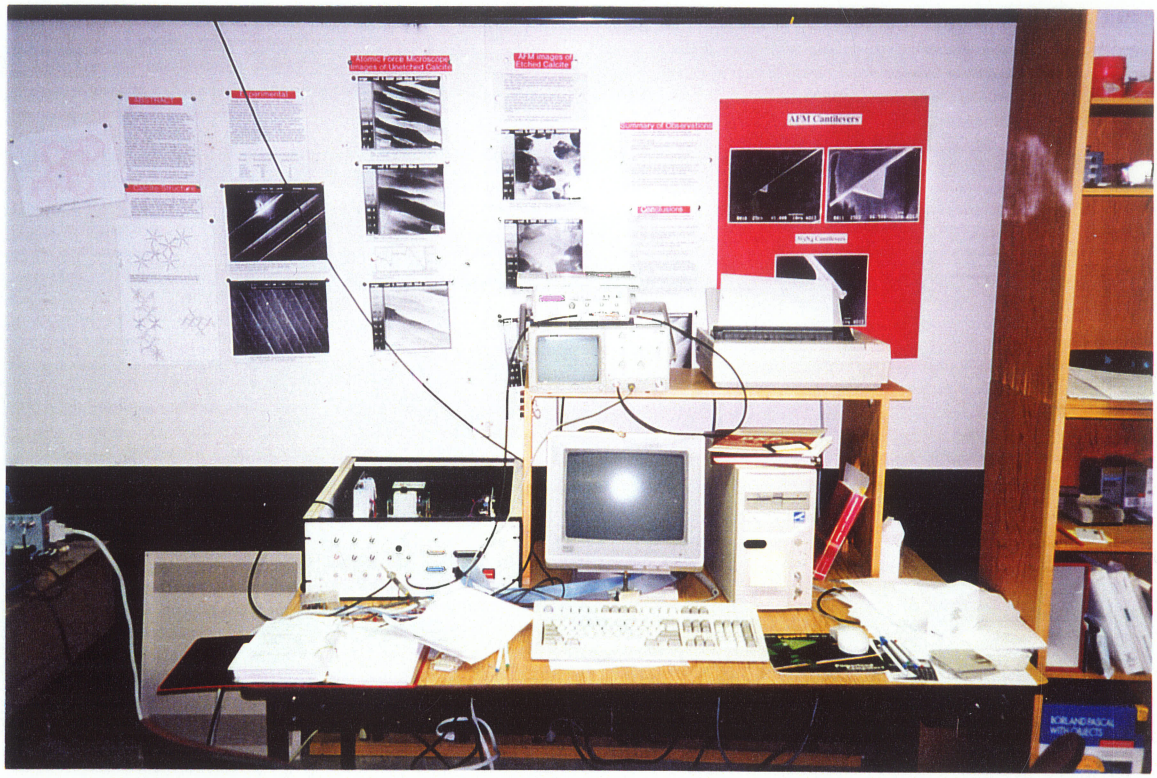


FIGURE 20. SPM Control System Hardware.

4.6 Chapter Summary

The SPM hardware system incorporates a new custom designed Analogue I/O board. The parameters of the hardware exceed in all cases the criteria we set in chapter 3 for the SPM controller. The Analogue I/O board was designed to interface to different Digital Signal Procesors, and as such the performance of the DSP

is not as important as those of the data acquisition board. The configuration for the DSP56000 board was designed using the OrcadSDT tools, and the propietary tools from Xilinx to map the schematic representation of this circuits into binary file downloaded into the FPGA. The software used to program the FPGA was written by the autor and will be briefly discussed in the following chapter. The software for the DSP and software for the PC is also discussed in the following chapter.

CHAPTER 5

Software for Digital SPM Control System

5.1 Chapter Introduction

There are three distinct portions of code used in the SPM control system. The first part is the software used to configure the FPGA. This part is divided into two parts, (i) the binary data generated by the FPGA tools to implement the device configuration, and the PC routines that interpret this data and download it into the FPGA itself. The second portion of the code is the DSP software which implements all the control functions of the microscope. The last part is the PC software which basically provides a user interface through which data from the SPM can be viewed and analysed, and parameters for the control system can be changed. This chapter gives a very brief explanation of the FPGA configuration software. The operation of the DSP software is then discussed, and finally the PC software is briefly discussed.

5.2 Field Programmable Gate Array Configuration Software

The FPGA is essentially software programmed digital hardware. This means that the actual digital hardware is defined by software. A microprocessor for example performs different task defined by software, but the hardware functions are defined upon construction of the device. In an FPGA the physical hardware is very generic as to implement boolean and memory functions only. Upon configuration of the device the actual hardware functions it performs are defined. The design process for specifying the hardware functions implemented by an FPGA vary, from using a high level Hardware Description Language (HDL) to specifying the gate level functions performed by each logic block in the FPGA. In this project the hardware functions were done at the gate level through schematic capture. OrcadSDT was used here to capture the schematic information. Through hierarchical circuits more complex functions were built. The translation of the schematically specified functions was performed by the Xilinx XDM tools. This software generates an ASCII (a standard character set) coded configuration file which is then sent to the FPGA through the PC's printer port. The C language

code that implements the translation from the ASCII coded configuration to the downloading of each bit to the FPGA is included in appendix C. A block diagram of the design process for the FPGA configuration is shown in Figure 21

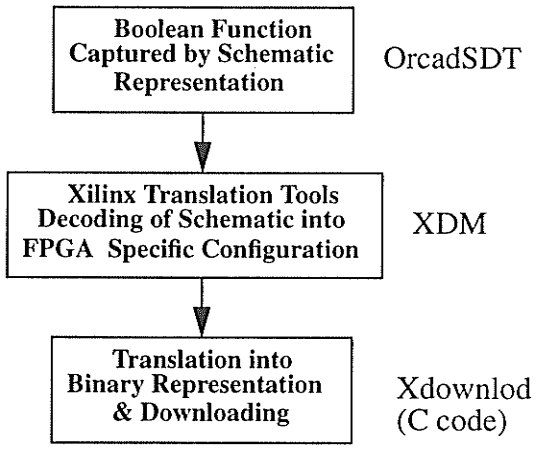


FIGURE 21. Design Process Using the Xilinx FPGA and Downloading software for the Analogue I/O board.

5.3 Digital Signal Processor Software

A DSP is a microprocessor optimized to perform mathematical calculations at high speed. Such are the functions required to emulate the effects of an analogue circuit on its input and output signals. As mentioned earlier we use a DSP56000 from Motorola. The data paths are 24 bits wide providing more precision than is needed in our control system (the DACs are 20 bits). This chip is part of a commercial development board available from motorola. The development system is

comprised of the DSP board which is interfaced to the Analogue I/O board through its 96 pin connector, the development PC interface board which sits on the ISA bus plane of the PC, and the required software to generate DSP specific code through standard C programming language. The data converters in the Analogue I/O board are all mapped into DSP memory by the FPGA. The following table gives the specific address of each device.

Device On I/O Board	DSP Address
X Channel 1 out (20bit DAC)	Y:A000$_{16}$ Write
Y Channel 2 out (20bit DAC)	Y:A001$_{16}$ Write
Z Channel 3 out (20bit DAC)	Y:A002$_{16}$ Write
Channel 4 out (20bit DAC)	Y:A003$_{16}$ Write
Channel 5 out (20bit DAC)	Y:A004$_{16}$ Write
Channel 6 out (20bit DAC)	Y:A005$_{16}$ Write
Digitally Programmable Filter	Y:A006$_{16}$ Write
Channel 1 in (18bit ADC)/ Convert 1&2 signal	Y:A000$_{16}$ Read
Channel 2 in (18bit ADC)	Y:A001$_{16}$ Read
Channel 3 in (18bit ADC)/Convert 3&4 signal	Y:A002$_{16}$ Read
Channel 4 in (18bit ADC)	Y:A003$_{16}$ Read
Channel 5 in (18bit ADC)/convert 5&6 signal	Y:A004$_{16}$ Read
Channel 6 in (18bit ADC)	Y:A005$_{16}$ Read

TABLE 6. DSP Addresses for Memory Mapped I/Os

The memory address for the input and output of channel 1 is the same, the distinction is made by the addressing mode (read or write). In order for the DAC sequencers to work properly, the data must be valid for at least 3 clock cycles (referenced to the I/O board clock, currently 20MHz).

5.3.1 Digital Feedback and Scan raster generation

The Feedback routine performs the sampling of two input signals, calculates the PI (Proportional Integral) feedback for one of the channels, and updates the feedback signal (sends calculated output to DAC). These operations are repeated 256, times the number of feedbacks between each image point, to obtain one row. Every image has 256 points across (in the case of the SRM another 256 points are acquired for the second image). Once an array of 256 points has been acquired, it is sent to the host computer (PC). Images are formed when 256 rows are acquired. The flowchart for the feedback routine is shown in Figure 22.

The generation of the feedback and the raster signals generation are done in the same subroutine. The generation of the X-raster scan is very simple. The parameters that are needed for the X-raster generation from the PC are:

1. Offset: The distance offset from the centre of the X-raster scan. This is useful for positioning the view of the microscope at the point of interest on the sample under observation. For simplicity this was not included in the flowchart. It can is incorporated simply by adding it to the Xout value before its sent out.
2. Xstepsize: The distance in bits in between every image point in the image (i.e Ximagesize/256)with a 20bit DAC there are $2^{20}/256$ image sizes or 4096
3. feedvalue: The number of feedback in between image points. From the flow chart we can see that the data sent to the Z DAC is only saved after feedvalue # of feedbacks are executed.

The equation for generating the X DAC signal is as follows

$$Xsignal = Xoffset + Round\left[\frac{(Xcounter)\ (Xstepsize)}{feedvalue}\right] \qquad \text{(EQ 14)}$$

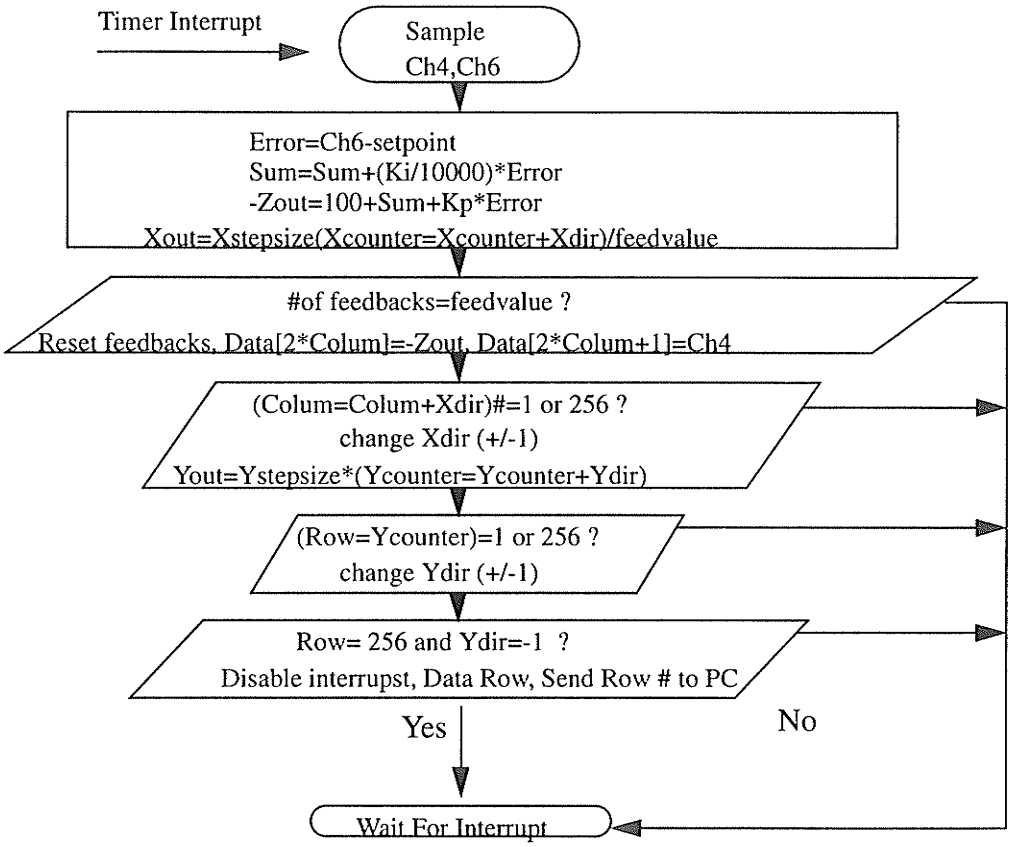


FIGURE 22. Flow-chart for feedback subroutine in DSP56000 software. This subroutine is driven by timer interrupts and its executes every 30.5us, which is equivalent to the sampling period.

It is easy to see from here that when we count up to the feedvalue before we obtain a single image point. The Xcounter is incremented once for every feedback that the DSP executes. This is compensated by dividing the xcounter by feedvalue in the generation of the X raster signal. Hence since we change the Xdir variable

every time 256 columns are counted, the size of the image remains 256*Xstepsize. However this scheme has not only the effect of allowing more feedbacks in between image points but also introduces points in between image points. This occurs due to the fact that the number Xcounter*Xstepsize/feedvalue is calculated in the following order: (i) Xcounter*Xstepsize, (ii) divide/feedvalue, and (iii) rounded off to an integer.

The Y-raster signal generation is almost identical to that of X, except that there is no analogue to the feedvalue variable and the Y signal is always made up of 256 steps. In the bigger sizes of Ystepsize, this can be a significant jump. The equation describing the Y-raster generation signal is as follows:

$$Ysignal = Yoffset + Ycounter(Ystepsize) \qquad \text{(EQ 15)}$$

The Z feedback value is also very simple. The feedback is calculated according to (eq. 10) on page 29. Where the error is simply the input value minus a digital setpoint. In terms of the actual variables used in the program the algorithm looks as show in the following expression

$$-Zout = K_P(Zin - Setpoint) + S \qquad ;Sum = Sum + K_I(Zin - Setpoint) \qquad \text{(EQ 16)}$$

The negative in front of the Zout is due to the inverting amplifier in hardware. The integral function is approximated as a running sum (i.e rectangular rule) where the mathematical steps are a little different than in the mathematical expression in (eq. 10) on page 29. If the K_I variable remains unchanged throughout the experiment,

then these two expressions are equivalent. However if the integral gain K_I changes during the experiment (acquiring of an image) then we see that this is the correct way of expressing the integral. This is because in this case the integral is no longer over one continuous function but two or even three depending upon how many times we change the integral gain during the experiment. In this case the equivalent in the analogue domain would be integrating a piecewice function.

This loop as mentioned before always executes within 30.5 us and thus the sampling frequency remains 33 kHz. However since the frequency of the input signal is directly related to the speed with which we rastered the tip over the sample, slowing the scan rate decreases the frequency of the components of the input signal. In this case this has the effect on the input signal of looking like a variable sampling rate. The scan time for one row in the forward direction in this scheme can then be calculated as:

$$ScanTime = 256 ImagePoints\,(feedvalue)\ (30.5us) \qquad \textbf{(EQ 17)}$$

Upon closer examination of the flowchart for the feedback routine one can observe that data is only sent in the forward direction of the X raster scan. This is due to the hysteresis in the properties of the piezo. If we take the data as the tip scans in both directions, the data the image will be severely distorted because points for corresponding columns in the X signal wil not correspond to the same distance in the x-distance-axes (hysteresis implies that the magnitude of voltage change required to move the sample in one direction is not the same as the magnitude of the voltage change required to bring it back to the same reference point). If the data

for only one direction is taken however points for corresponding forward voltage changes should correspond (although non-linear displacement with voltage). The scan period for one row is then twice that given in (eq. 17) on page 67. The scan time for one entire image is then

$$ImageScanTime = (2)\ (256Rows)\ (256ImagePoints)\ (feedvalue)\ (30.5us) \quad \textbf{(EQ 18)}$$

The above equation ignores the time taken to send the data to the PC since this is highly dependent on the performance of the PC speed as well as the efficiency of the code used to read the data from the DSP. It should be noted that because a fixed point DSP is used to implement all calculations, all variables must be either integer or integer fractions [ref.26]. This puts a restriction on feedvalue of it being greater than 2, since in the instead of doing the division in (eq. 14) on page 64 the reciprocal of feedvalue is precalculated and thus multiplication is used. This is because multiplication is much faster than division and we would like the feedback loop to execute as fast as possible.

5.3.2 Vectored Routines on the DSP

All subroutines on the DSP are triggered by interrupts. At loading time for the DSP software all variables are initialized and the program goes into an infinite loop. The feedback routine, as mentioned earlier executes through a timer interrupt. The timer generates an interrupt periodically and forces the feedback routine to execute, thus keeping the sampling rate constant. The rest of the routines are used to change the variables in feedback routine, with the exception of; the Pro-

grammable filter routine which simply sends and eight bit number to the address of that device, the stepper routine which takes care of stepping the motor in one direction or the other, and the meter routine which simply samples the ADC Channe l6 once and sends the value to PC. All other routines are related to the feedback routine. A table for the vectored routines is as follows, the listing for this program is included in Appendix D.

Routine name or purpose	Vector Written by PC	DSP Address
Start of DSP program		$P:0040_{16}$
Enable Timer Interrupt	86_{16}	$P:000C_{16}$
Trigger Stepper Motor	$8C_{16}$	$P:0018_{16}$
Read Channel6 for Voltage Meter	$8D_{16}$	$P:001A_{16}$
Default Time Interrupt Vector	$8E_{16}$	$P:001C_{16}$
Reserved	$8F_{16}$	$P:001E_{16}$
Default Host Transmit Interrupt Enable HTIE	90_{16}	$P:0022_{16}$
Digitally Programmable Filter	92_{16}	$P:0024_{16}$
Change Xoffset	93_{16}	$P:0026_{16}$
Change Yoffset	94_{16}	$P:0028_{16}$
Enable HTIE	95_{16}	$P:002A_{16}$
Disable HTIE	96_{16}	$P:002C_{16}$
Toggle Feedback On/Off	9716	$P:002E_{16}$
Change Xscansize	98_{16}	$P:0030_{16}$
Change Yscansize	99_{16}	$P:0032_{16}$
Change bias Voltage	$9A_{16}$	$P:0034_{16}$
Change Proportional Feedback Constant K_P	$9B_{16}$	$P:0036_{16}$

TABLE 7. Hexadecimal Values for PC vector interrupts to DSP

Routine name or purpose	Vector Written by PC	DSP Address
Change Integral Feedback Constant K_P	$9C_{16}$	P:0038_{16}
Change Scan Rate	$9D_{16}$	P:$003A_{16}$
Change Set Point	$9E_{16}$	P:$003C_{16}$

TABLE 7. Hexadecimal Values for PC vector interrupts to DSP

5.4 IBM Compatible Personal Computer User Interface

The PC implements the user interface aspect of the SPM controller. The interface was written by David T. Shimizu, and is based on the code for the AFM interface. Several limitations of the AFM interface have been removed in this version of the program. The major ones are: (i) the ability to obtain two images at the same time (as opposed to one in the earlier version), (ii) the range on the PI gains (K_I and K_P) has been increased from 1,000 (1 to1,000) to 20,000(-10,000 to 10,000), and (iii) 20-bit images are obtained as opposed to 16-bit. The numbers for the PI gains have no units, and are not to be considered loop gains. The real gain are dependent on the SPM being used, and the sampling period. The sign required for negative feedback depends on the setup of the detection on the control signal (i.e. we could change the sign on the detector for the AFM by adding a unity gain inverting amplifier), for this reason negative gains have been allowed in this program.

The ability to obtain two images at the same time was implemented in such a way such that future expansion was possible. In other words if there is a need to obtain three images at the same time this can be done without any major software

modifications. In the AFM interface this was not the case, as a result the software had to be re-written to conform to the DPMI (DOS Protected Mode) standard. DPMI, is a DOS extender that allows more than 640kbytes of memory to be accessed by the programmer (up to 16Mbytes).

Other routines to reflect the new hardware present on the Analogue I/O board were also added. One of these is used to program the Digitally Programmable Filter from Frequency Devices.

5.5 Chapter Summary

The SPM instrument includes a significant amount of software. An analysis of the impact of the software on the control system, is a research topic on its own right. In this chapter the basic workings of an approximation to a PI controller have been presented. The effects of this approximation, on the control system is beyond the scope of this thesis. The effects of this approximations to a PI controller, as well as the effects of the PI controller itself vary from instrument to instrument.

The vectored routine approach to interfacing with the PC was presented. This scheme has worked satisfactorily. The programming of the Xilinx FPGA, programming of the DSP56000 board, and starting the user interface on the IBM compatible PC have been combined into a single DOS command. This command is basically a batch file that executes all three pieces of software in the required sequence. The different parameters for different instruments are also loaded on start-up. For example the AFM, SRM, and STM have different constants for the

PZT displacement, based on this the following parameters are calculated a indicated in table 8.

Instrument	AFM	SRM
PZT Constants in Angstroms/Volt Lateral (X,Y), Vertical (Z)	240L, 82V	1200L, 133V
Field Of View (max, X,Y) Maximum Vertical Displacement (Z)	9.6 um 3.3 um	48 um 5.3 um
Minimum Step (1bit) in Angstroms Lateral(X,Y), Vertical (Z)	0.09 Angstroms L 0.03Angstroms V	0.46 Angstroms L 0.05 Angstroms V

TABLE 8. Some Important Parameters for AFM & SRM Sample Positioners

CHAPTER 6

Conclusions & Recommendations

6.1 Conclusions

The primary objective of this thesis project was to design and implement a digital control system for scanned probe microscopes. The design of such a system involved the design of custom hardware. The design constraints for the custom hardware were outlined on this thesis. Based on this criterion, an Analogue I/O board was designed and constructed. This board implements various features that are desirable in SPM control system's data acquisition boards. This custom hardware was interfaced to a DSP56000 board which includes the DSP56001 chip from Motorola^{TM}. The control system's user interface was then built on an IBM compatible PC namely a 486DX2-66 motherboard with an ATI SVGA video card (resolution of 800x600 pixels in 256 colours).

The emphasis of this thesis is on implementing the electronics, and software on which to build a control system, and not on the performance of the control system itself. As such, only an approximation to a proportional integral (PI) controller has been implemented. This was done to verify the operation of the system as a whole.

The actual quantization range on the data converters was also evaluated. This was done to avoid the any miss-understanding on the capabilities of a data converter and its useful data. For example if a data converter is specified as 20-bits, but the noise level is more than one Least Significant Bit (LSB) then its useful range is reduced and equal to the minimum voltage signal that can observed. If the noise in the 20-bit system is on the order of the 2nd LSB (i.e. bit # 2 with bit #20 being the most significant), then the useful range has been reduce to 18 bits, since the last bit is uncertain.

The DSP software was re-written to handle two images at the same time. The new software takes into account the higher resolution of the data converters. A couple of "bugs" found in the old software are worth mentioning: (i) sign extension of the digitized input signal was not implemented correctly, as a result the old control system only worked with positive control signals, (ii) the integrator was implemented as a running sum, and its limits were never checked, as a result the integrator would oscillate whenever it overflowed. Both of this have been corrected. The code still operates within 30.5 us, and can be optimized to operate even faster. Alternatively the second feedback loop to compensate for changes in resistance of the sample in the SRM can be implemented digitally in the DSP,

making the SRM controller completely digital (currently only topography change compensation is done digitally).

The new features of the system as a whole should prove very flexible for future developments. The custom designed hardware fulfilled all the objectives that we set out to accomplish. Although the bandwidth of the controller has not yet reached the 50kHz sampling rate set out in our requirements, this is not a limitation on the custom designed hardware but is more closely related to the DSP performance, an upgrade to a faster DSP such as the DSP96000 from Motorola should make this requirement a reality.

6.2 Recommendations

It was mentioned in chapter 4 that input channel 5 suffers from "spikes" that seem to appear randomly. The appearance of this glitches may be due to the way the ADC is read by the Xilinx sequencer. The reading sequence is automatically started by reading channel 5. As a result if channel 5 is read before the sequence is complete, the sequence is restarted. This does not give enough time for the sample and hold to track the signal, and erroneous results will be obtained this way. A detailed simulation of the read process would be ideal to identify any potential problems arising from this scheme.

One of the most important aspects of a control system are of course its performance or response to a desired input. The emphasis on this project has been on the implementation of the controller itself and not the optimization with respect to certain criteria. The performance of a control system is often specified as its

response to a typical input in the context of its specific purpose. For example if a control system is meant to move a motor in steps, then its step response is a typical input and the time it takes for the motor to step to the desired position would be a measure of its performance.

The performance of a digital control system is closely related to its hardware as well as the software algorithm. Since it is my belief that upgrading to a faster DSP will automatically increase the execution time of the feedback algorithm and therefore the sampling period; I suggest that upgrading to a faster DSP be considered as a means of obtaining the 50kHz sampling bandwidth.

The replacement of the DSP board would also imply modification of the interface board between the PC and DSP. Currently this is done through an 8-bit ISA (Industrial Standard Interface) bus card. It would be more desirable to redesign this card from the ground up. The ISA bus runs at around 8MHz and at 8 or 16 bits it degrades the performance of both the reading and reconstruction of the data in the PC interface. To read a single entry in a 256 array the PC must access the ISA card 3times (3 bytes, 8 bits each) the 24 bit value (DSP words are 24 bits) must then be reconstructed into a single variable (a 32 bit integer in the PC), after which it must be sign-extended to 32 bits. The redesign of this card would allow the card to be targeted to the PCI (Peripheral Component Interconnect) bus standard that allows 32 bit transfers to occur at the processor speed. In our current PC this implies that the element could be read at the speed of 33MHz (PCI speed for a 486DX2-66 PC). This value would be automatically sign extended by the

hardware, or in the case of the DSP96000 no extension would be necessary since the DSP word length is 32 bits.

The need to speed up the interface arises from the display performance of the SPM image display system. Currently at slow scan rates (2 to 10 Hz row scan period) the display keeps up with the data being received. At higher speeds however the images being displayed start missing rows. Although upon retrieval of the image it appears to be correct, the display system does not reflect the data we are acquiring. Code optimization is one solution, but the eliminating of the ISA bottleneck would prove to be a long term solution when more then two images are to be acquired simultaneously.

The generation of the X and Y raster signals has remained linear throughout both the AFM controller and now the SPM controller. However the fact cannot be ignored that the PZT is a non-linear device. A solution to this problem would be close loop control of the PZT in the lateral directions. Although this is desirable, its implementation would be to complicated for a small gain in overall system performance. Instead the generation of non-linear raster signal to linearise the response of the PZT should be considered. This could be implemented in software or in hardware through the use of "smart DACs". These "smart DACs" could be implemented in the FPGA by eliminating channels not being used by the current instrument (the AFM uses only three DACs, and one ADC out of six of each). These "smart DACs" would be given the scan rate and size, and would generate either a linear raster signal or a non-linear one through the use of pre-calculated look-up tables.

This is essentially moving the workload from the DSP software into the hardware. This is already accomplished in the current I/O board. The serial writing and reading of the DACs is all done in hardware (in the FPGA). Sign extension of the 18-bit ADCs is also done in hardware, saving the DSP about 5 instruction cycles per every channel being read.

Finally the DSP software should incorporate routines specifically designed for the type of microscope being controlled. The feedback routine that currently runs on the DSP samples two channels and controls one. The data from these two channels is then sent to the PC. In the SRM both of this channels are used (one for topography and the other for resistance), but in the AFM the other data is just disposed-of. Instead of a physical feedback routine a pointer that could be changed from the PC by the user could be used. This pointer would point to the appropriate feedback routine for the specific instrument being used. In this manner the DSP software would not have to be modified, and different feedback schemes could be introduced simply by adding a new routine, and a specific address that points to it, in the DSP software.

References

[1] G. Binnig, C.F. Quate and Ch. Gerber, "Atomic force microscope," Phys. Rev. Lett. 56, 930 (1986)

[2] G. Binnig, Ch. Gerber, E. Stoll, T.R. Albrecht and C.F. Quate, "Atomic resolution with atomic force microscope," Europhys. Lett. 3, 1281 (1987)

[3] R. G. Carr, "Finite element analysis of PZT tube scanner motion for scanning tunneling microscopy," J. Microscopy 152, pt.2, 379 (1988)

[4] Y. Kuk and P.J. Silverman, "Scanning tunneling microscope instrumentation," Rev. Sci. Instrum. 60, 165 (1989)

[5] J. N. Nxumalo, D. Shimizu, D. J. Thomsom, "Cross Sectional Imaging of Semiconductor Device Structures by Scanning Resistance Microscopy, " J. Vac. Sci. Technol. (1995)

[6] J. N. Nxumalo, D. Shimizu, D. J. Thomsom, "High Resolution Imaging of MOSFET Cross-Sections by SRM, " accepted for publication in J. IEEE Electron Device Letters. (1995)

[7] C. Shafai, "Two Dimensional Delineation of Semiconductor Doping by Scanning Conductance Microscopy," M.Sc. Thesis (1992)

[8] T. R. Albrecht, S. Akamine, T.E. Carter and C. F. Quate, "Microfabrication of cantilever styli for the atomic force microscope," J. Vac. Sci. Technol. A 8, 3386 (1990)

[9] R. Möller, A Esslinger and B. Koslowski, "Thermal noise in vacuum scanning tunneling microscopy at zero bias voltage," J. Vac. Sci. Technol. A 8, 590 (1990)

[10] J. N. Nxumalo, "Analysis of Semiconductor Device Cross-sections By Scanning Resist-

ance Microscopy", PhD Thesis (1995)

[11] K. Yackoboski, "Design and Implementation of an All-Digital Atomic Force Microscope Controller," M.Sc. Thesis, (1992)

[12] R. C. Dorf, "Feedback Control Systems" Addison-Wesley , (1988)

[13] P. K. Cheng, "New Image Acquisition Algorithms for Atomic Force Microscope," MSc Thesis (1994)

[14] S. Onyshko, "Digital Control Systems Course (24.771), Lecture Notes", (1993)

[15] A. Lemus, and H. Nguyen, "The Angstrommeter," BSc Thesis, (1993)

[16] Borland International, "Borland Pascal with Objects 7.0", Language Guide, Chapters 17 and 18, (1992)

[17] BurrBrown, "Integrated Circuits Data Book Supplement Volume 33b," pages 6.2-39 to 6.2-48, (1990)

[18] BurrBrown, "Integrated Circuits Data Book Supplement Volume 33c," pages 6.2-39 to 6.2-48, (1990)

[19] S. M. Clark D. R. Baselt, C. F. Spence, M. G. Yougquist, and J. D. Baldeschweiler, "Hardware For Digitally Controlled Scanned Probe Microscopes," Rev. Sci. Instrum. 63, 4296 (1992)

[20] Apex Microtechnology Corporation "Data Book", pages E137 to E140, (1993)

[21] Frequency Devices, "844 Series Data Book," pages 1 to 6, (1994)

[22] HewlettPackard, "Optoelectronics Designers Catalog", pages 6-46 to 6-52, (1992)

[23] Orcad, "Schematic Design Tools User's Guide," (1992)

[24] Racal-Redac, "Cadstar PCB Design Software User's Guide", (1993)

[25] Xilinx, "The Programmable Logic Data Book," (1994)

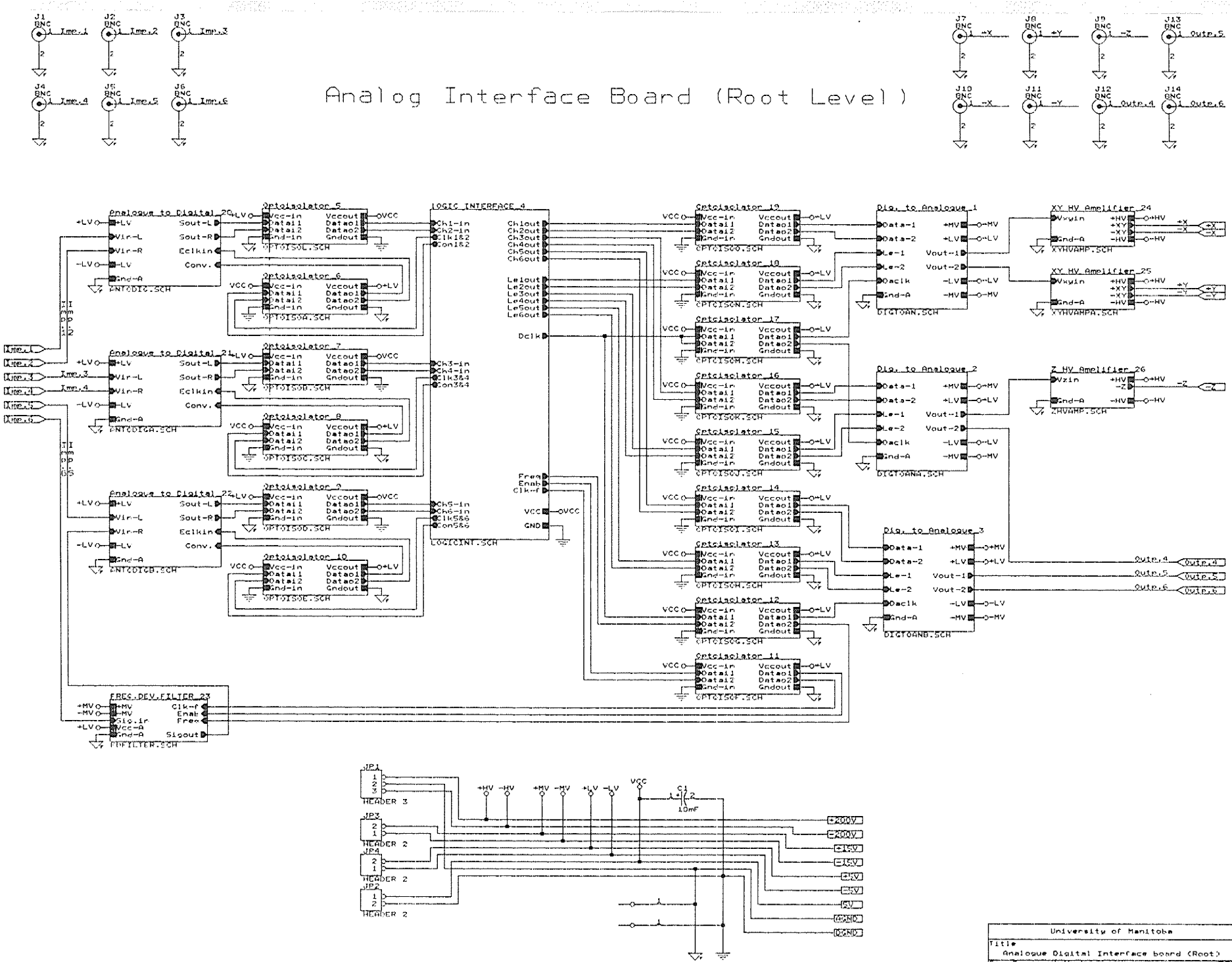
[26] Motorola, "DSP56000/DSP56001 Digital Signal Processor User's Manual," (1990)

[27] G. C. McGonigal, K. Yackobski, P. K. Cheng, and D. J. Thomsom, "Digital Signal Processor Based AFM Controller, Design and Issues," (1992)

Appendix A

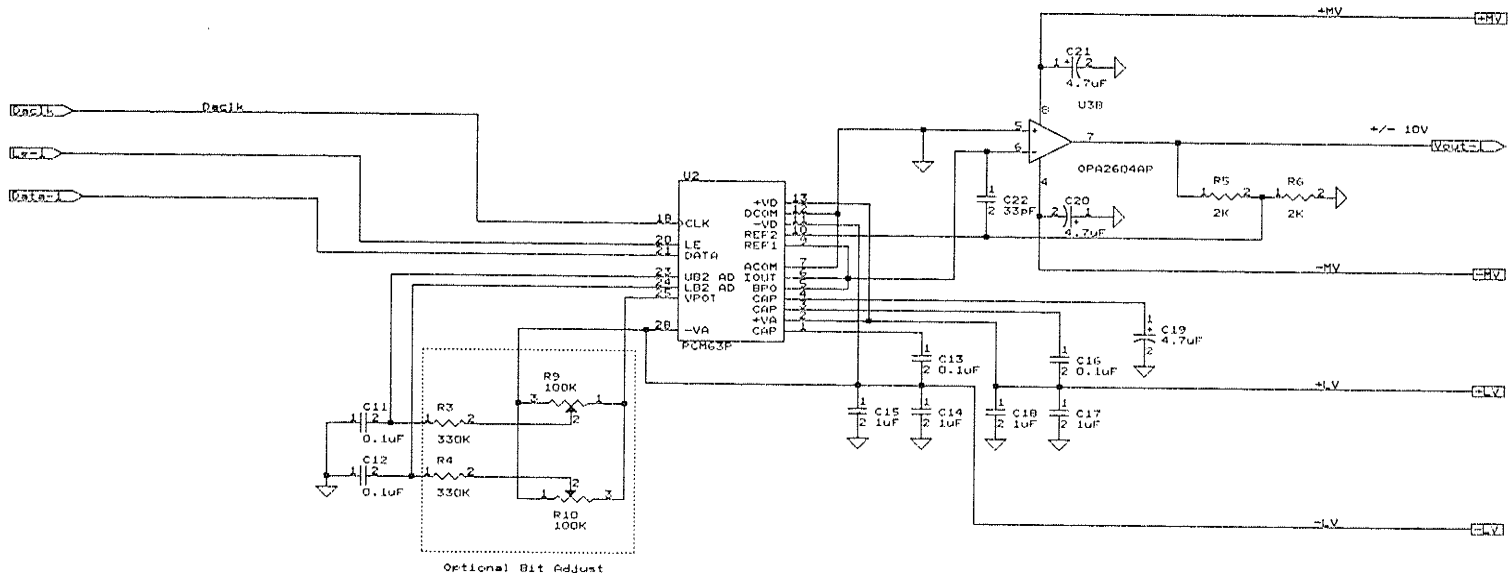
Schematics for the Analogue I/O Board

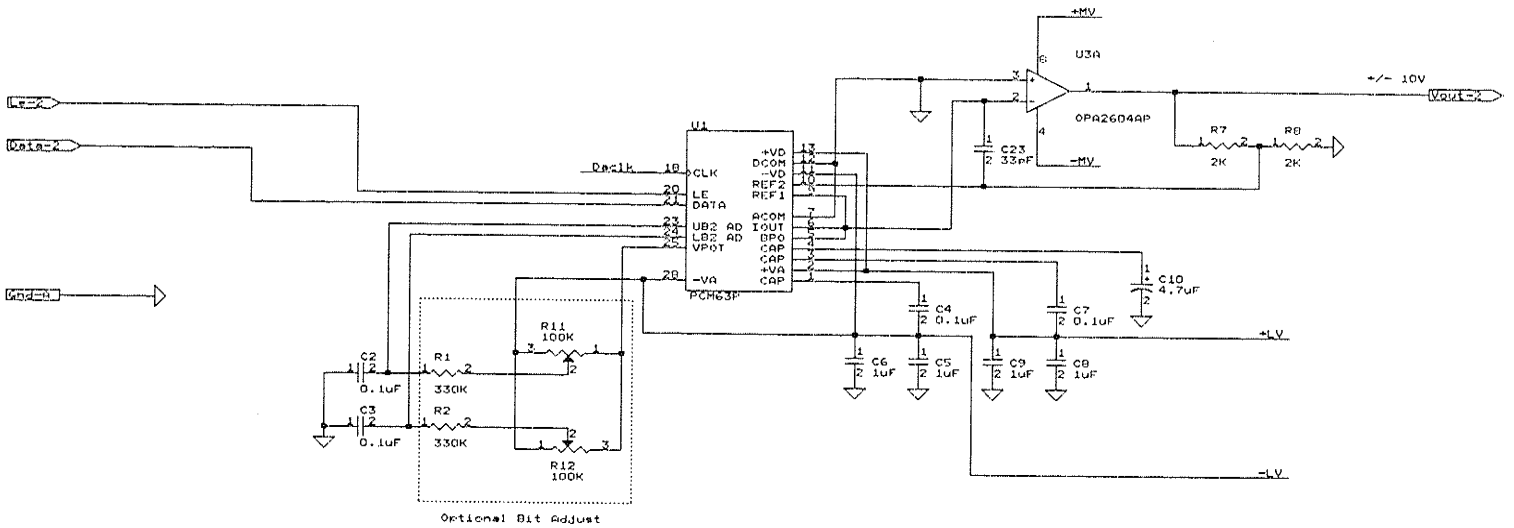
Analog Interface Board (Root Level)



University of Manitoba		
Title: Analogue Digital Interface board (Root)		
Size: E	Document Number: 2 (File = SSRM.SCH)	REV: A
Date: February 2, 1995	Sheet 1 of 2	

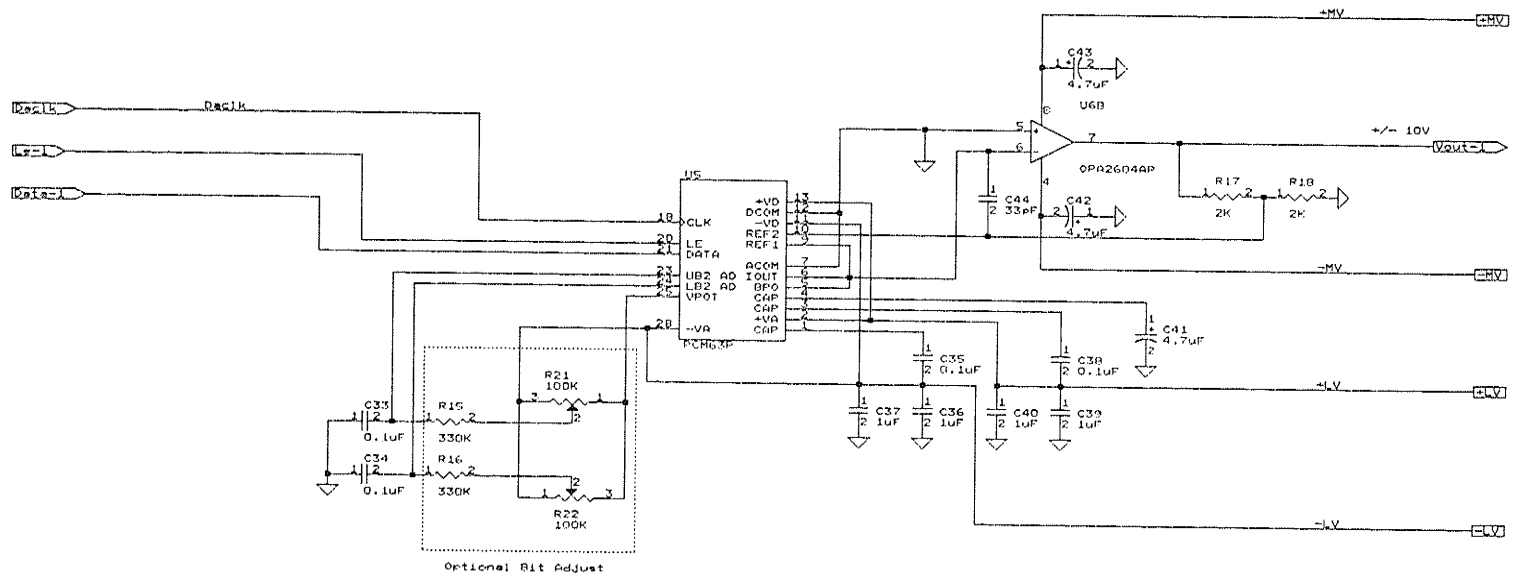
Digital to Analogue Converter

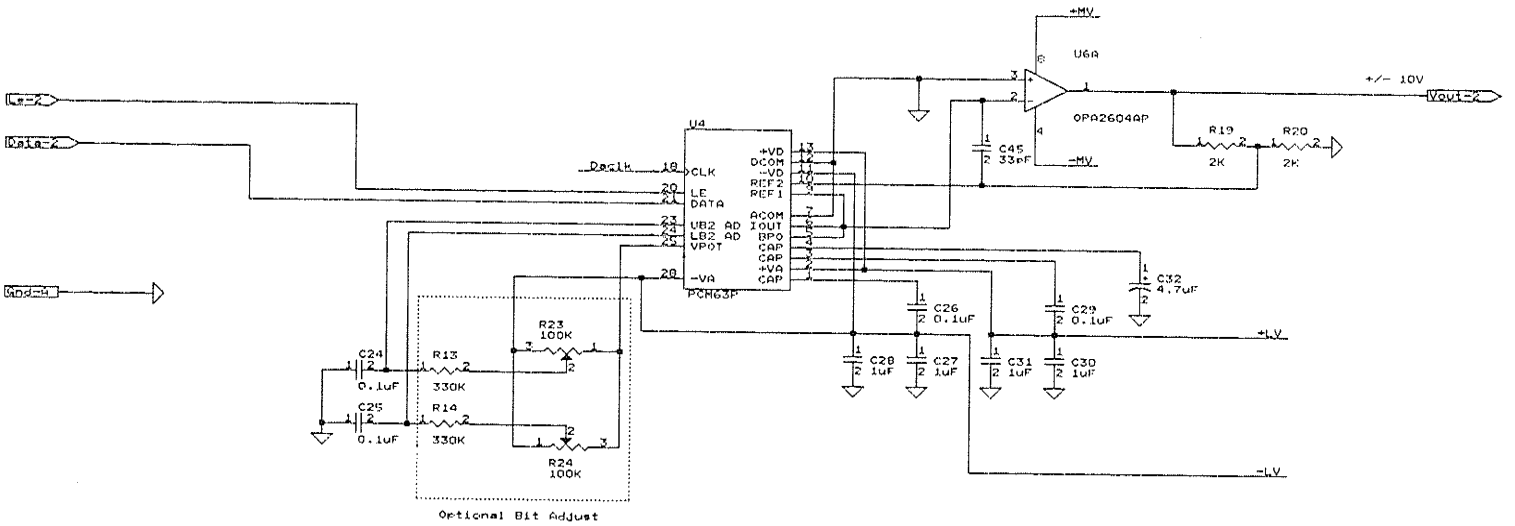




University of Manitoba		
Title: Digital to Analogue Converter		
Size: E	Document Number: 2 (File = DIGTOAN.SCH)	REV: A
Date: February 2, 1995	Sheet: 1 of 2	

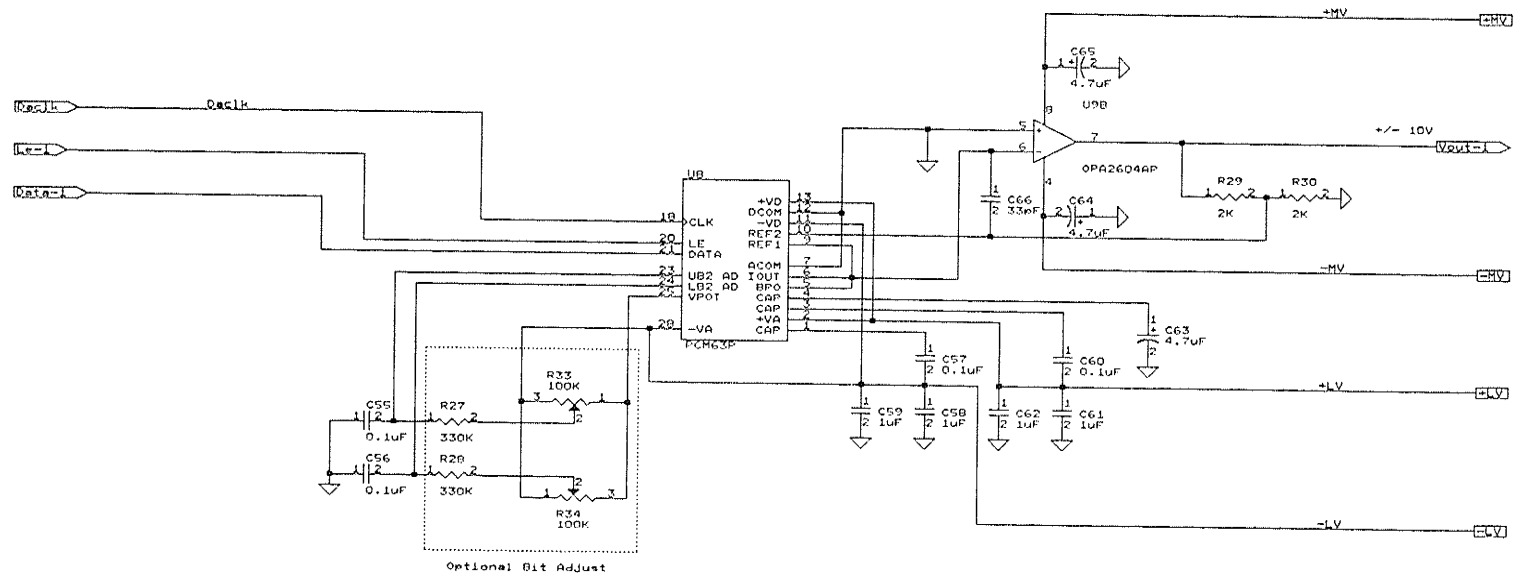
Digital to Analogue Converter

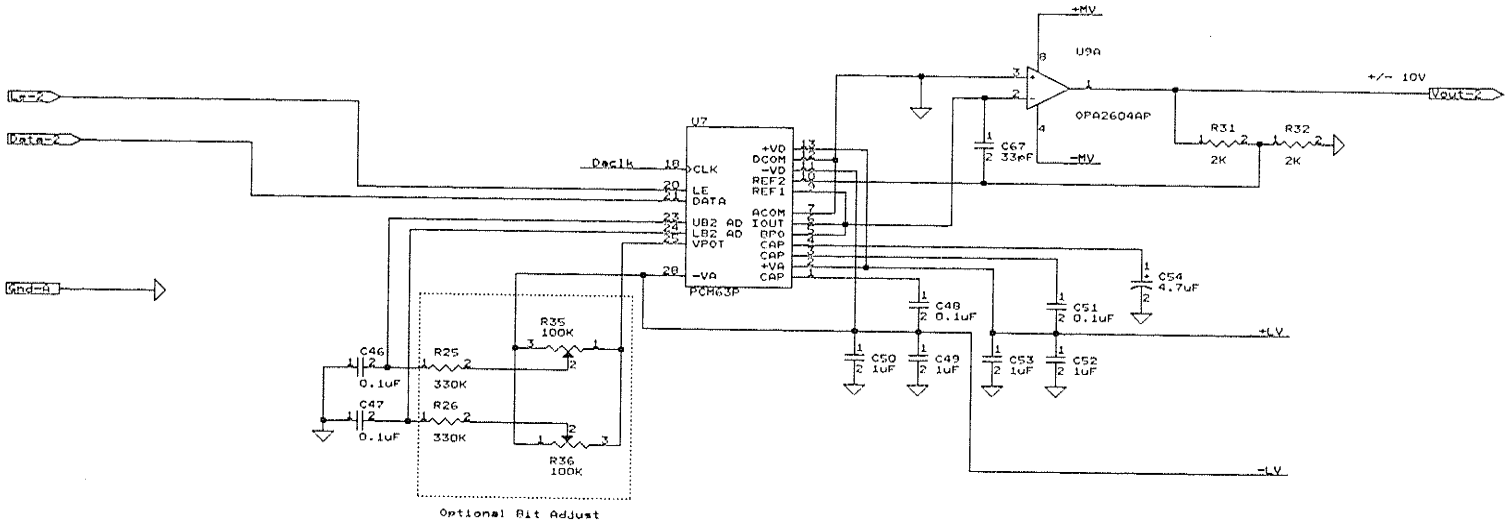




University of Manitoba		
Title: Digital to Analogue Converter		
Size: C	Document Number: 2 (File = DIGTOANA.SCH)	REV: A
Date: February 2, 1995	Sheet 3 of 27	

Digital to Analogue Converter

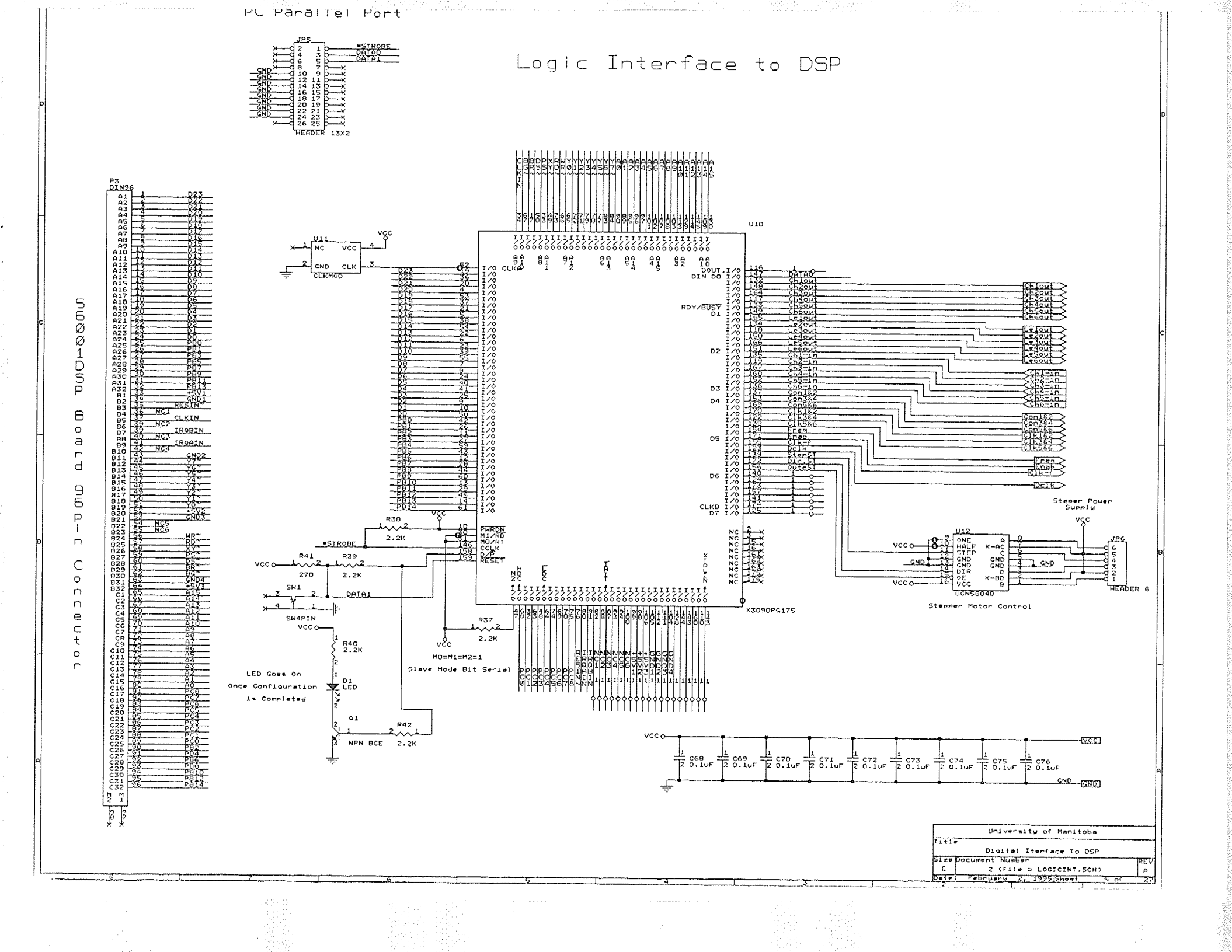




University of Manitoba		
Title: Digital to Analogue Converter		
Size: E	Document Number: 2 (File = DIGTOANB.SCH)	REV: A
Date: February 2, 1995	Sheet 4 of 27	

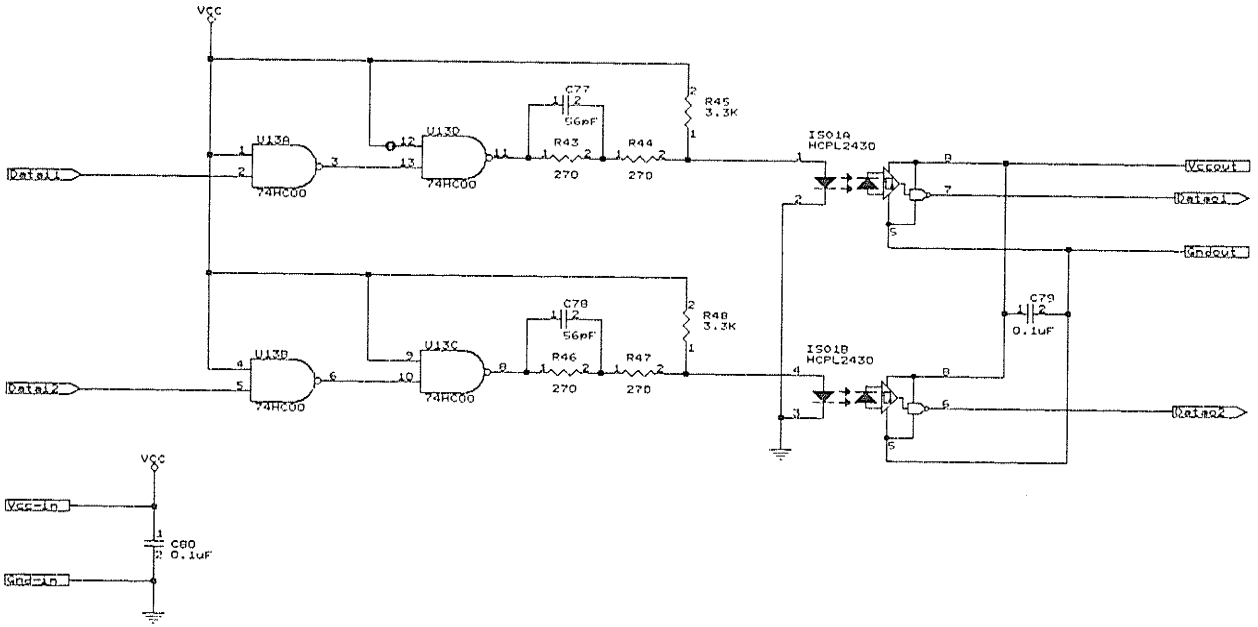
PC Parallel Port

Logic Interface to DSP



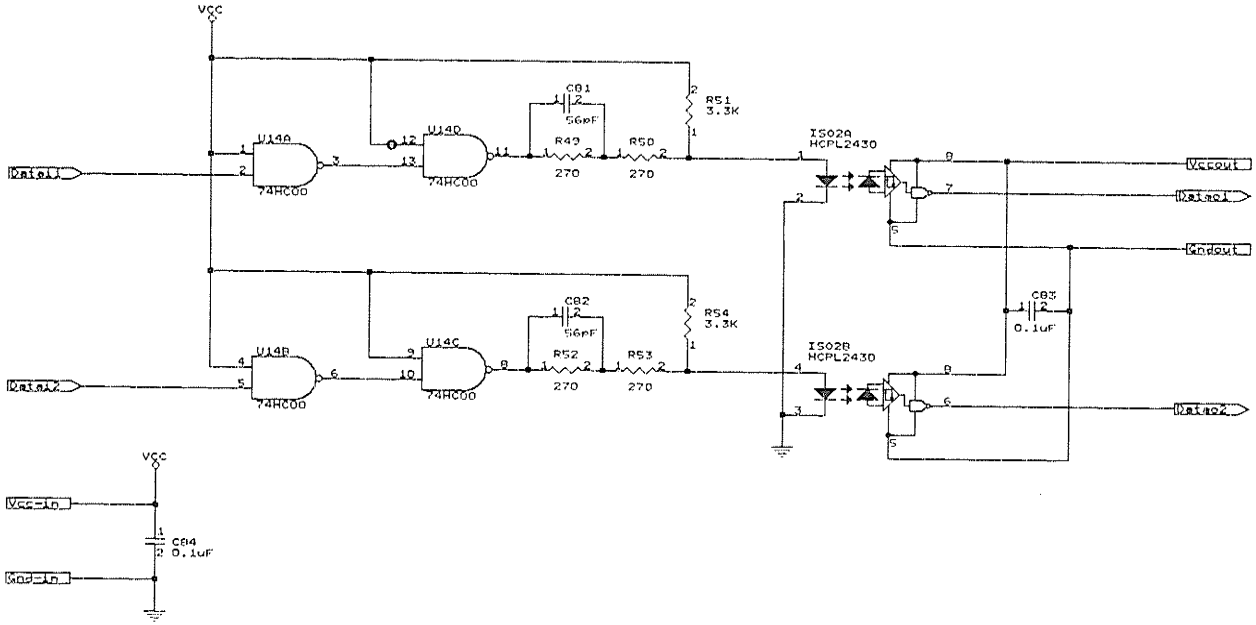
University of Manitoba		
Title: Digital Iterface To DSP		
Size: E	Document Number: 2 (File = LOGICINT.SCH)	REV: A
Date: February 27, 1995	Sheet 5 of 7	

Optoisolator



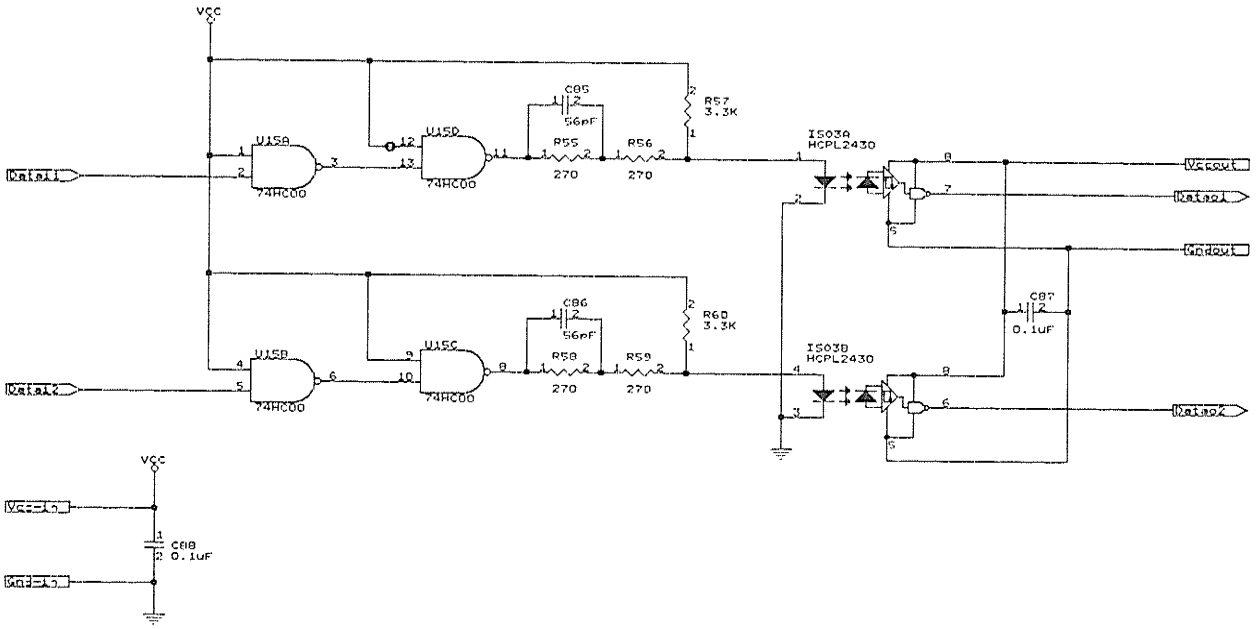
University of Manitoba		
Title: Optoisolator Interface		
Size: C	Document Number: 2 (File = OPTOISOL.SCH)	REV: A
Date: February 27, 1995	Sheet 4 of 27	

Optoisolator



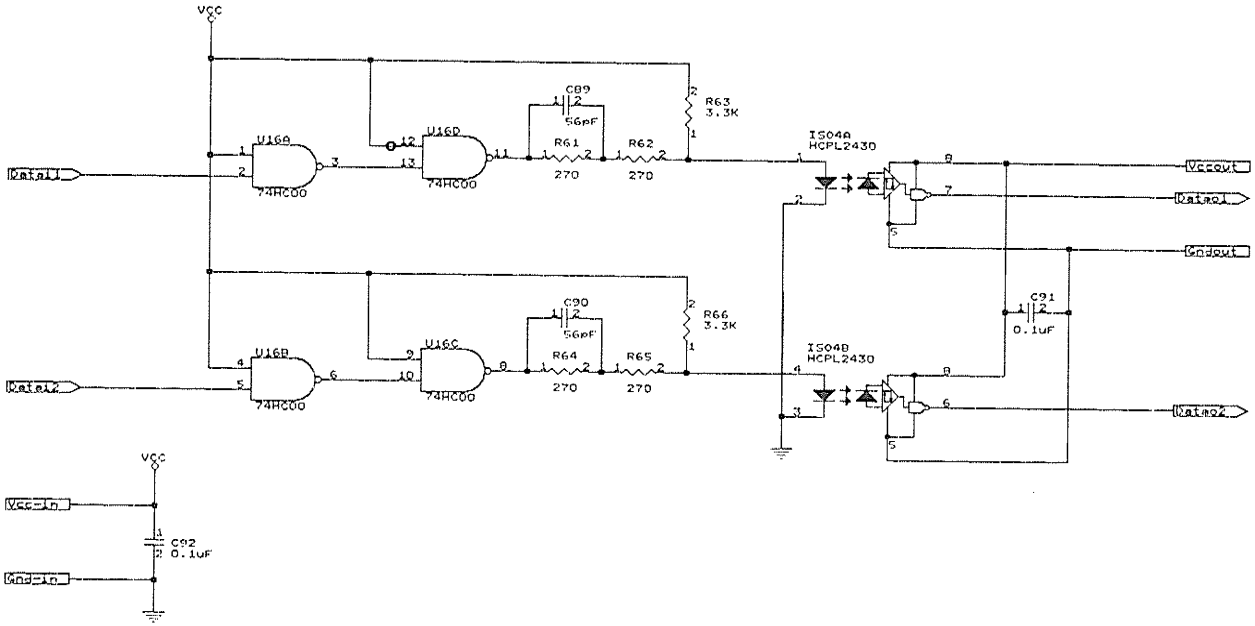
University of Manitoba		
Title: Optoisolator Interface		
Size: E	Document Number: 2 (File = OPTOISOA.SCH)	REV: A
Date: February 2, 1995	Sheet 7 of 27	

Optoisolator



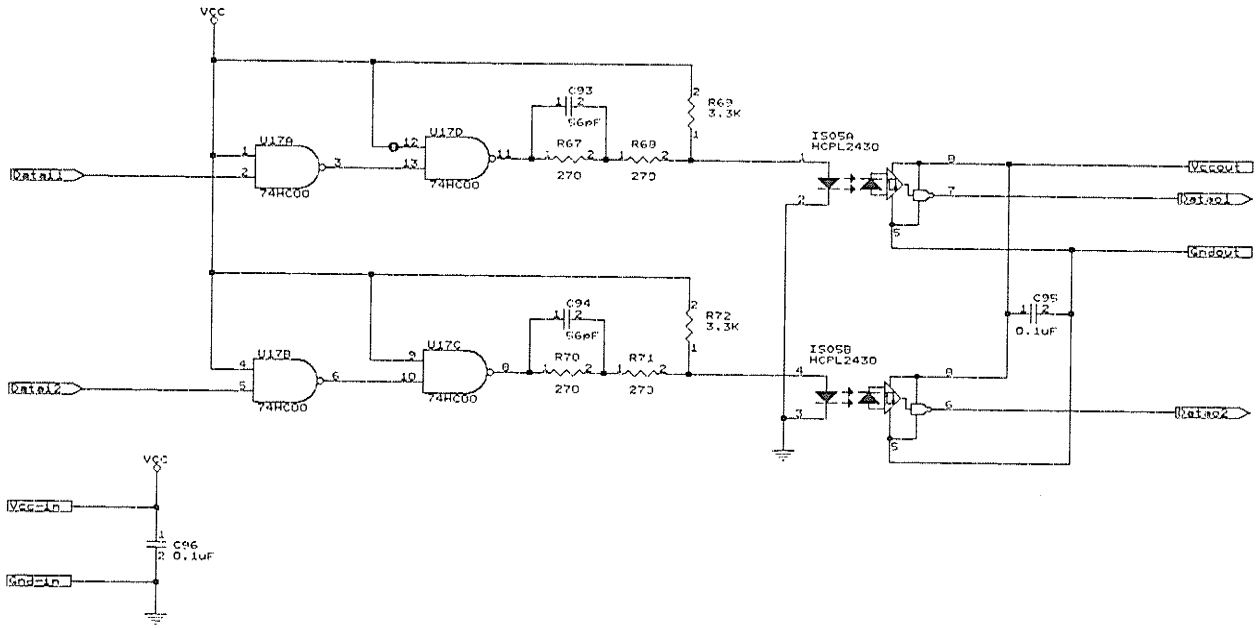
University of Manitoba		
Title: Optoisolator Interface		
Size: E	Document Number: 2 (File = OPTOISOB.SCH)	REV: A
Date: February 2, 1995	Sheet	of

Optoisolator



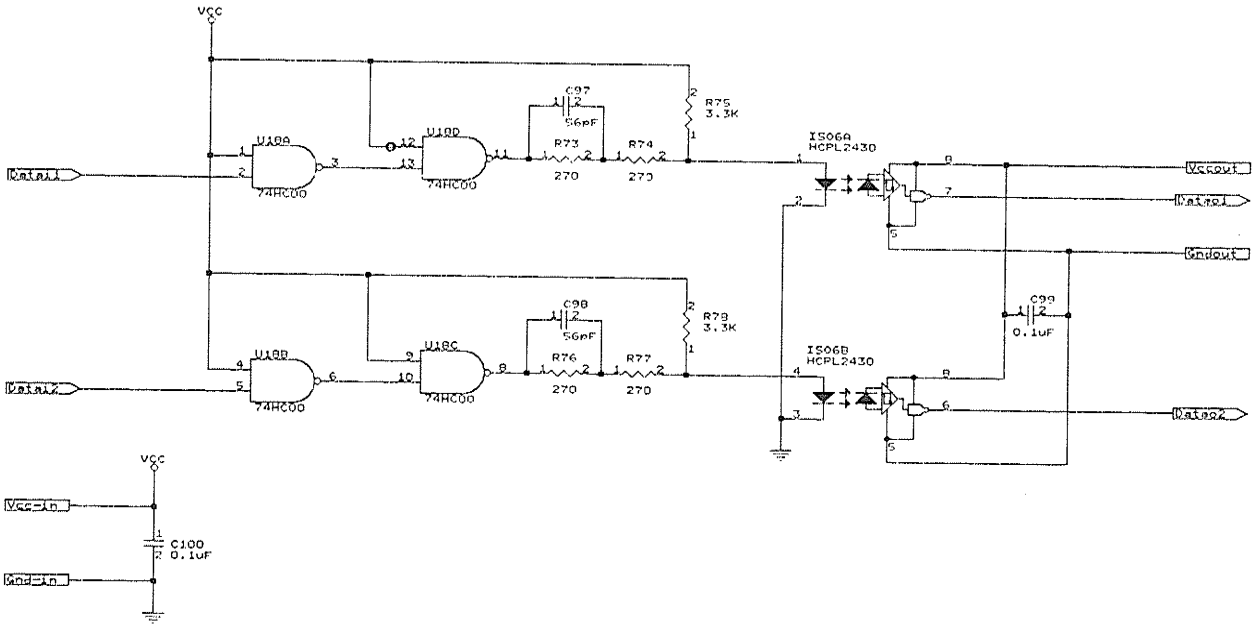
University of Manitoba		
Title: Optoisolator Interface		
Size: C	Document Number: 2 (File = OPTOISOC.SCH)	REV: A
Date: February 2, 1995	Sheet: of	

Optoisolator



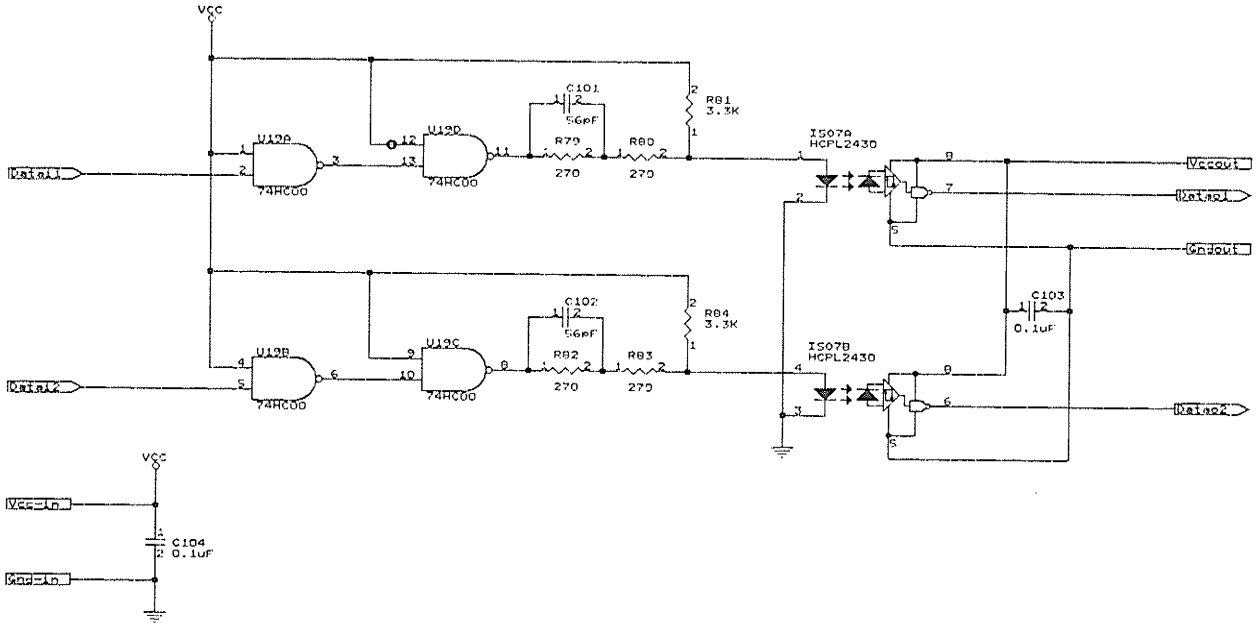
University of Manitoba		
Title: Optoisolator Interface		
Size: C	Document Number: 2 (File = OPTOISOD.SCH)	REV: A
Date: February 8, 1995	Sheet 10 of 27	

Optoisolator



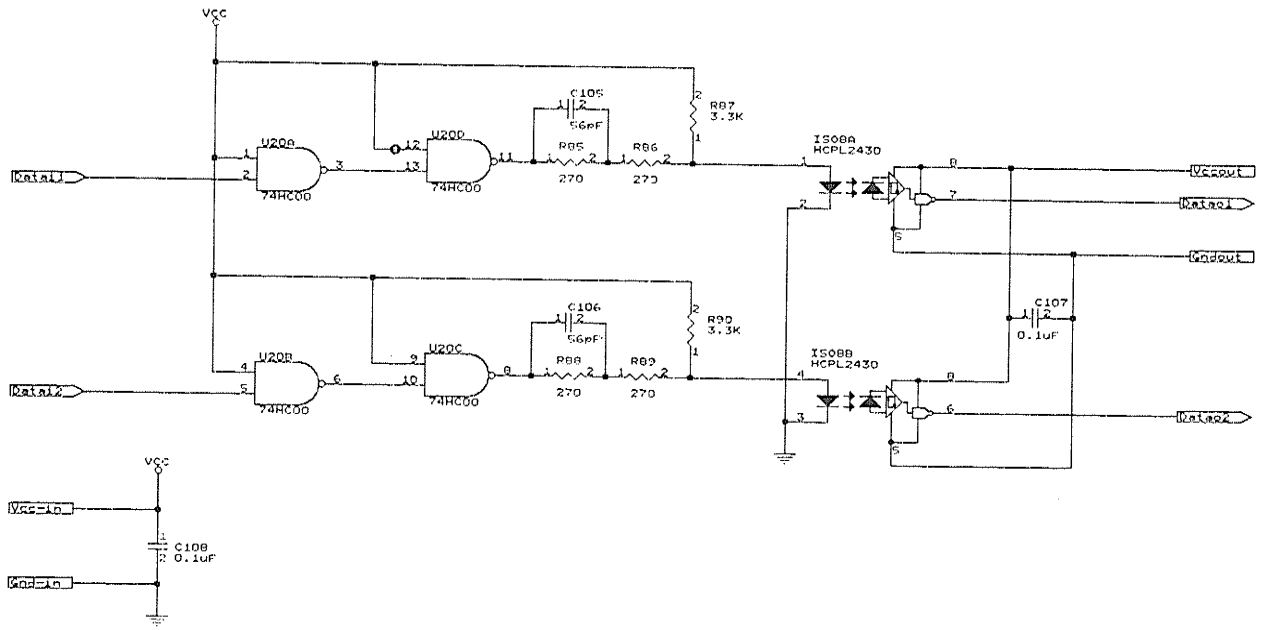
University of Manitoba		
Title: Optoisolator Interface		
Size: B	Document Number: 2 (File = OPTOISOE.SCH)	REV: A
Date: February 27, 1995	Sheet 1 of 2	

Optoisolator



University of Manitoba		
Title: Optoisolator Interface		
Size: C	Document Number: 2 (File = OPTOISOF.SCH)	REV: A
Date: February 2, 1995	Sheet 12 of 27	

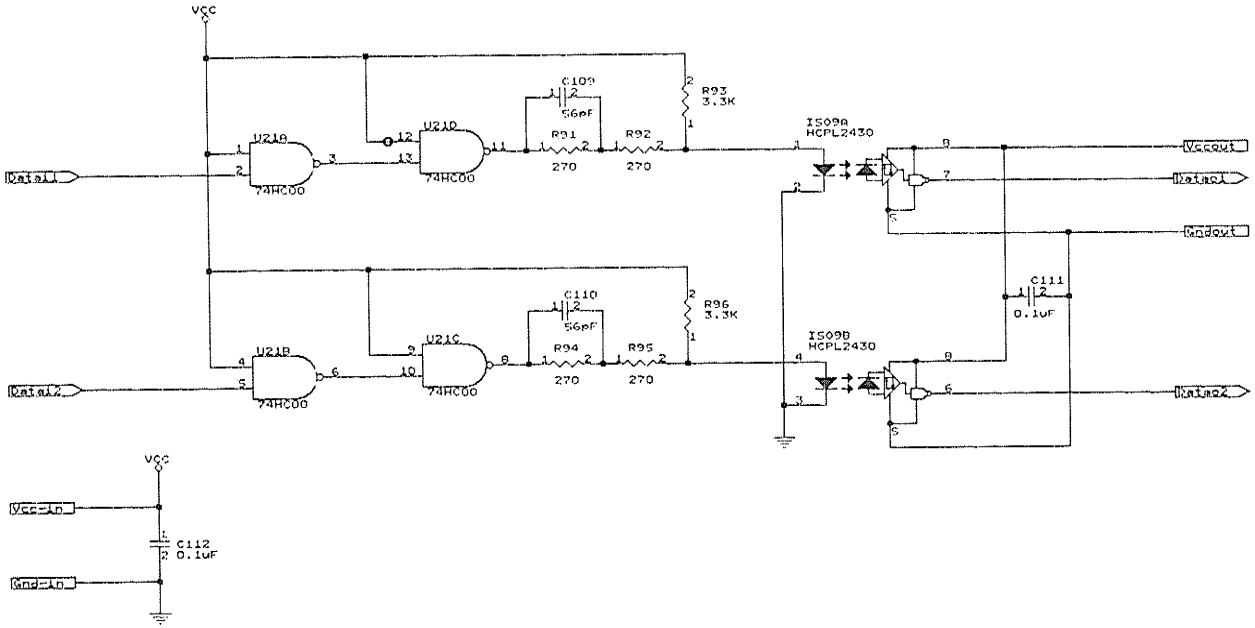
Optoisolator



University of Manitoba		
Title: Optoisolator Interface		
Size: C	Document Number: 2 (File = OPTOISOG.SCH)	REV: A
Date: February 2, 1996	Sheet 13 of 27	

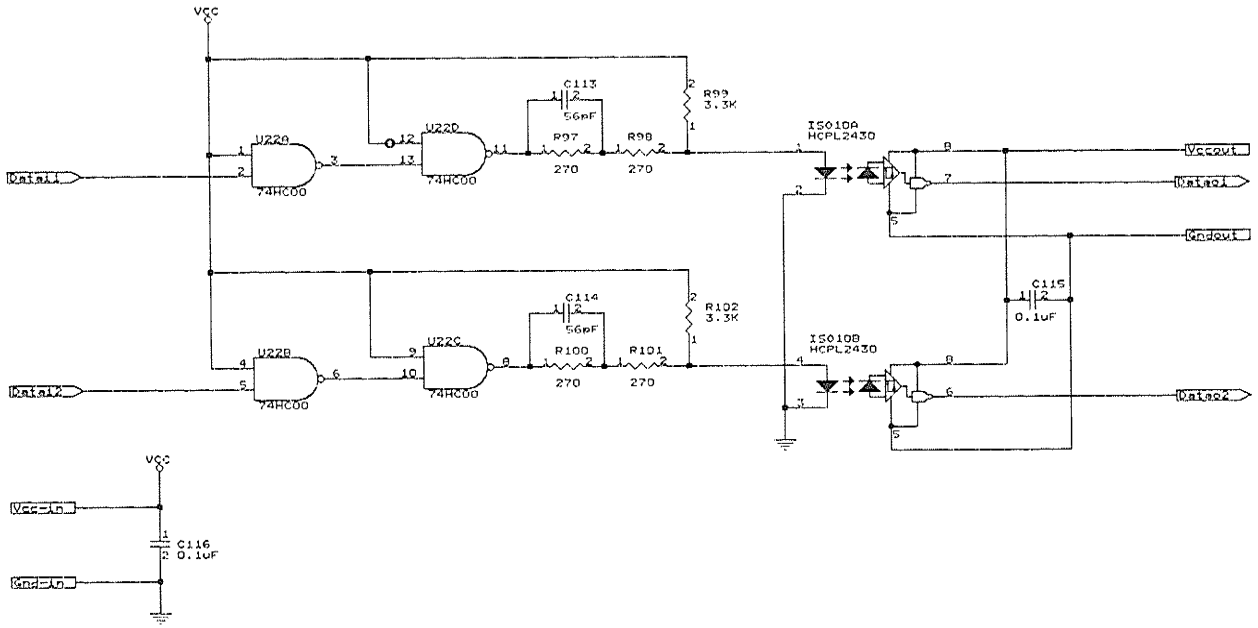
```
        org   p:$003A           ; where to put this interrupt vector
        jsr Fchangefeedvalue+4  ;jump to read and set scan rate from PC
        nop                     ; it's a 2-instruction interrupt
        endsec
%/
/* ------------------------------------------------------------------- */

/* host command vector interrupt                                      */
/* change current set point on DSP                                    */
/%
        section setpoint
        org   p:$003C           ; where to put this interrupt vector
        jsr Fsetpoint+4         ; jump to read in setpoint from PC
        nop                     ; it's a 2-instruction interrupt
        endsec
%/

/* ---------------------------- END CODE ------------------------------*/
```

Optoisolator



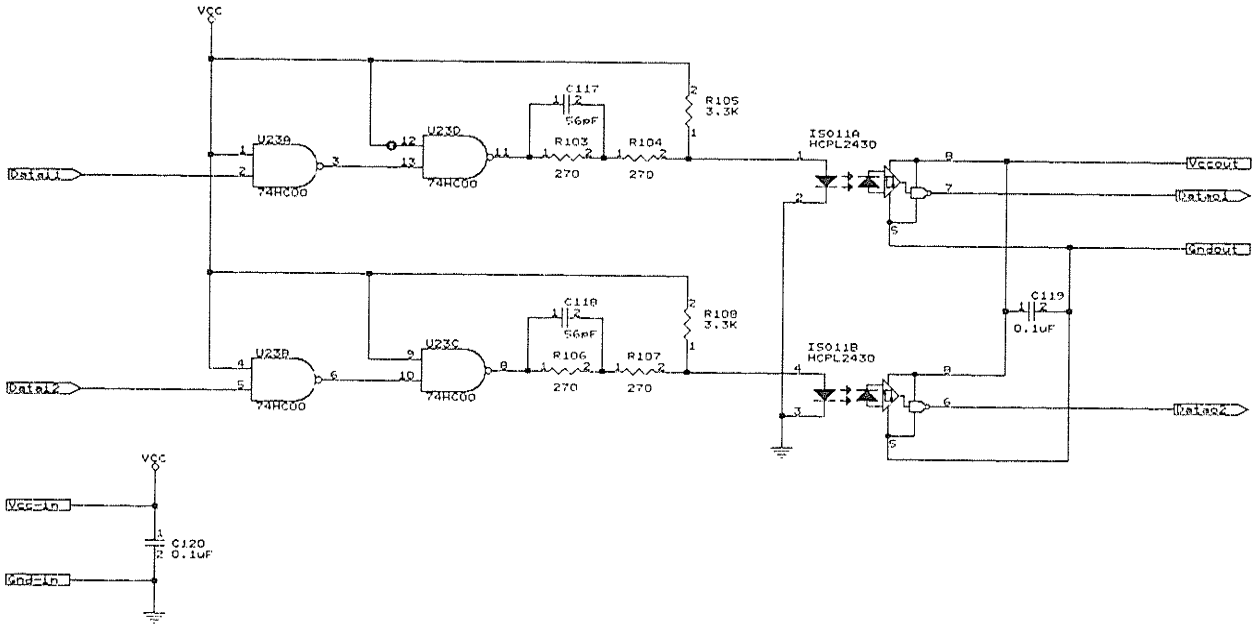
University of Manitoba		
Title: Optoisolator Interface		
Size: E	Document Number: 2 (File = OPTOISOH.SCH)	Rev: A
Date: February 2, 1995	Sheet 14 of 21	

Optoisolator



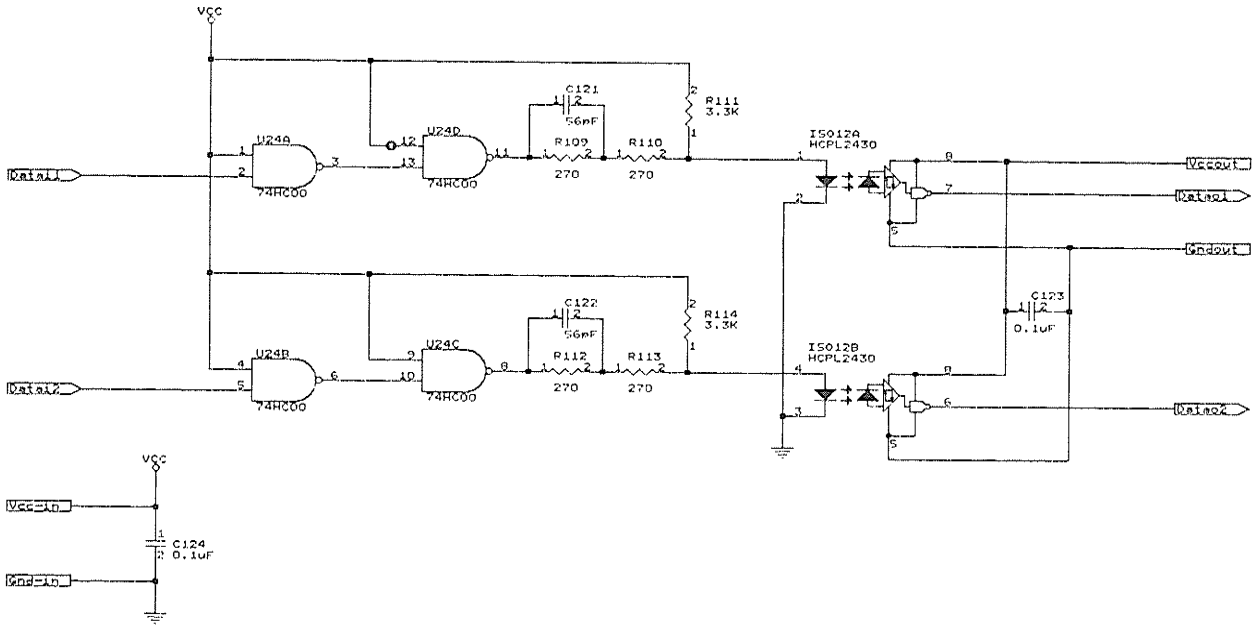
University of Manitoba		
Title: Optoisolator Interface		
Size: C	Document Number: 2 (File = OPTOISOI.SCH)	REV: A
Date: February 6, 1995	Sheet 15 of 27	

Optoisolator



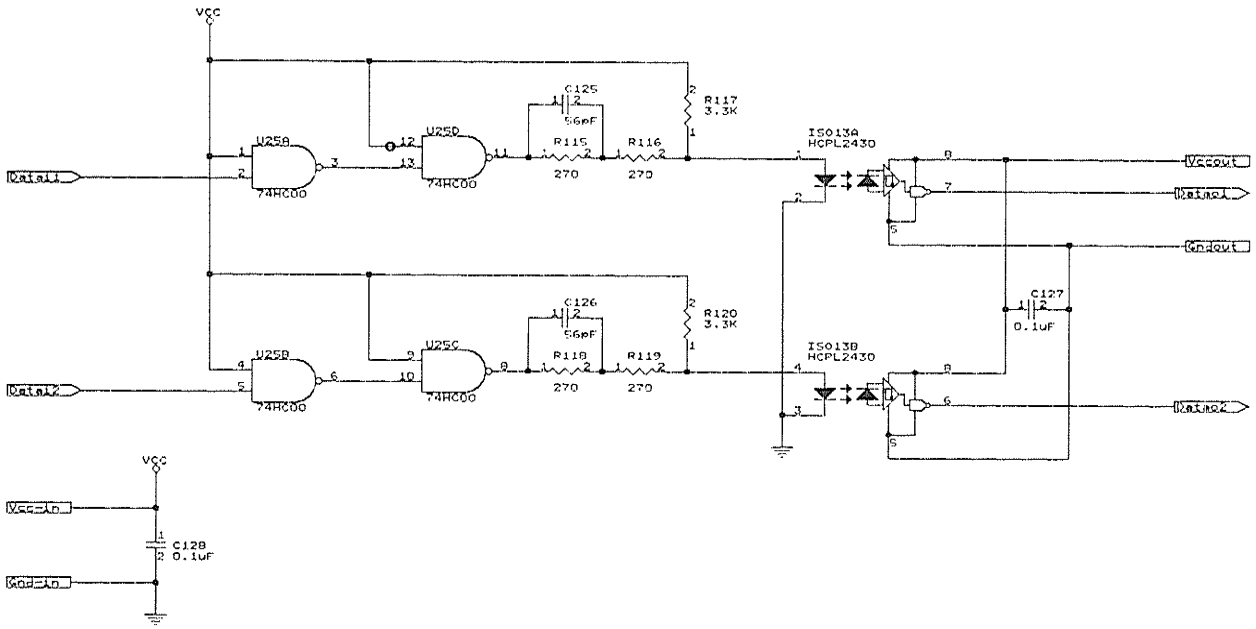
University of Manitoba		
Title: Optoisolator Interface		
Size: C	Document Number: 2 (File = OPTOISOJ.SCH)	REV: A
Date: February 2, 1995	Sheet 16 of 27	

Optoisolator



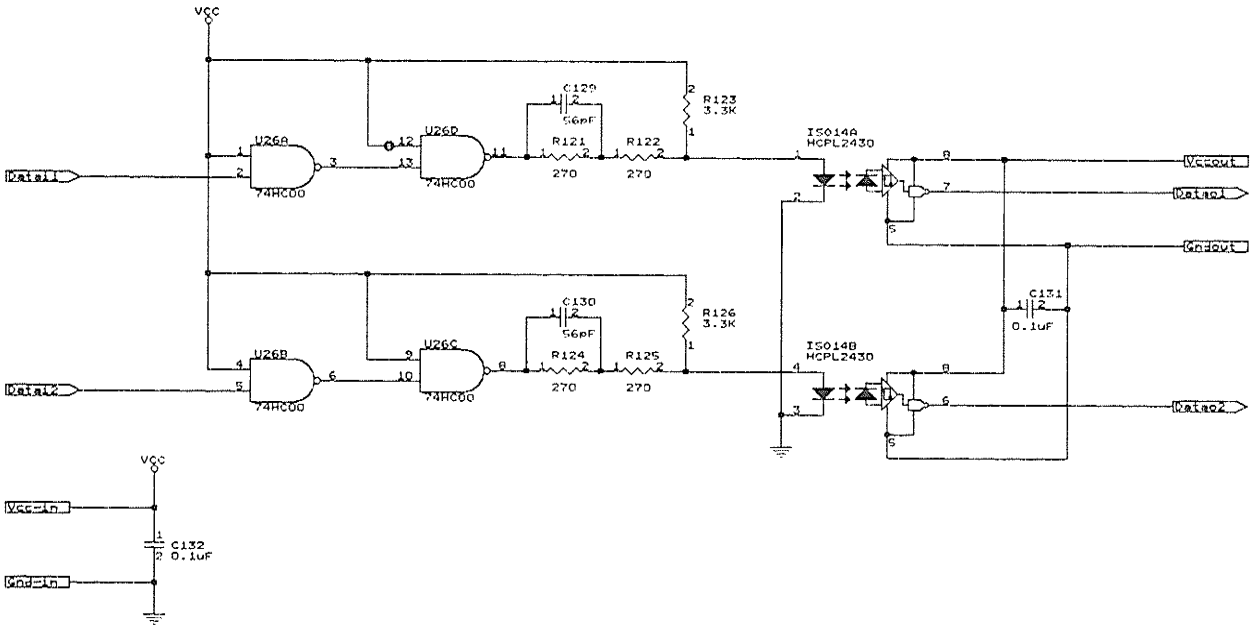
University of Manitoba		
Title: Optoisolator Interface		
Size: E	Document Number: 2 (File = OPTOISOK.SCH)	REV: A
Date: February 2, 1995	Sheet 17 of 27	

Optoisolator



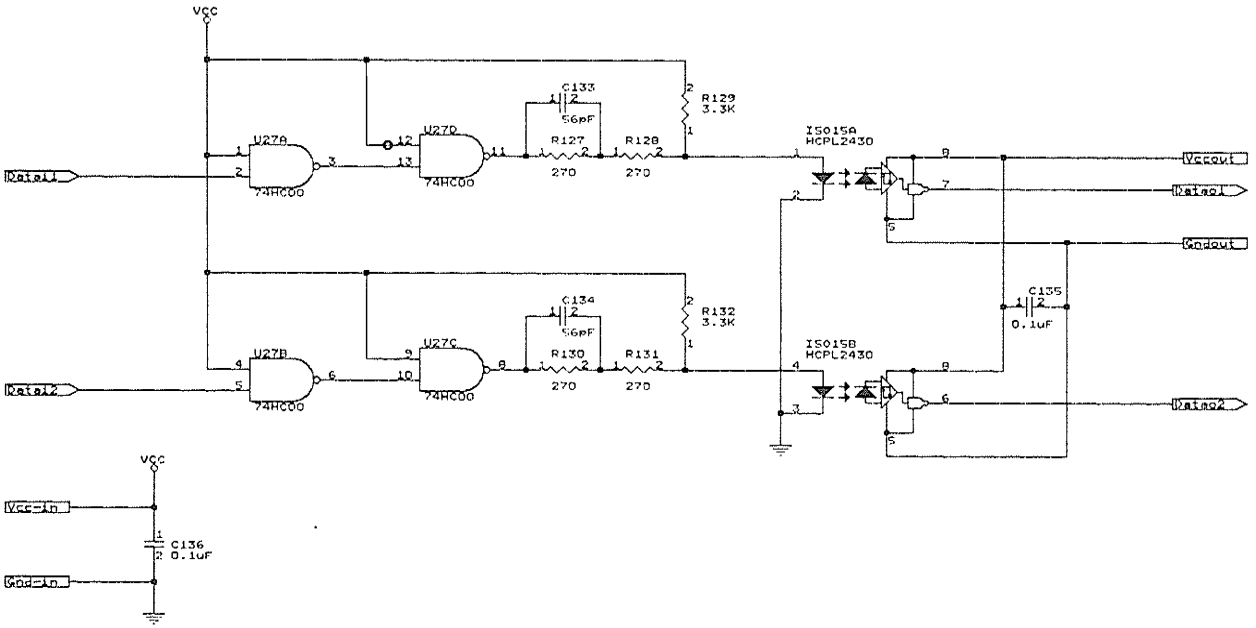
University of Manitoba		
Title: Optoisolator Interface		
Size: E	Document Number: 2 (File = OPTOISOM.SCH)	REV: A
Date: February 2, 1995	Sheet 10 of 27	

Optoisolator



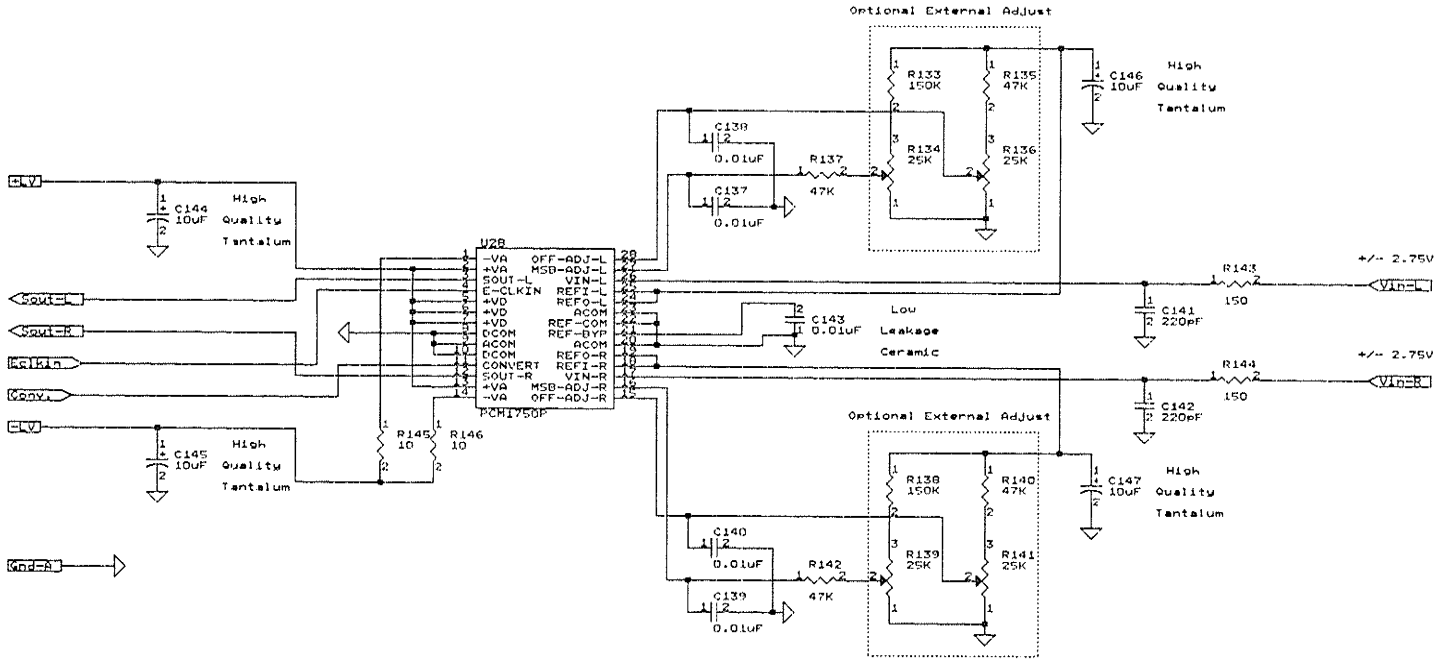
University of Manitoba		
Title: Optoisolator Interface		
Size: E	Document Number: 2 (File = OPTOISON.SCH)	REV: A
Date: February 2, 1995	Sheet 10 of 27	

Optoisolator



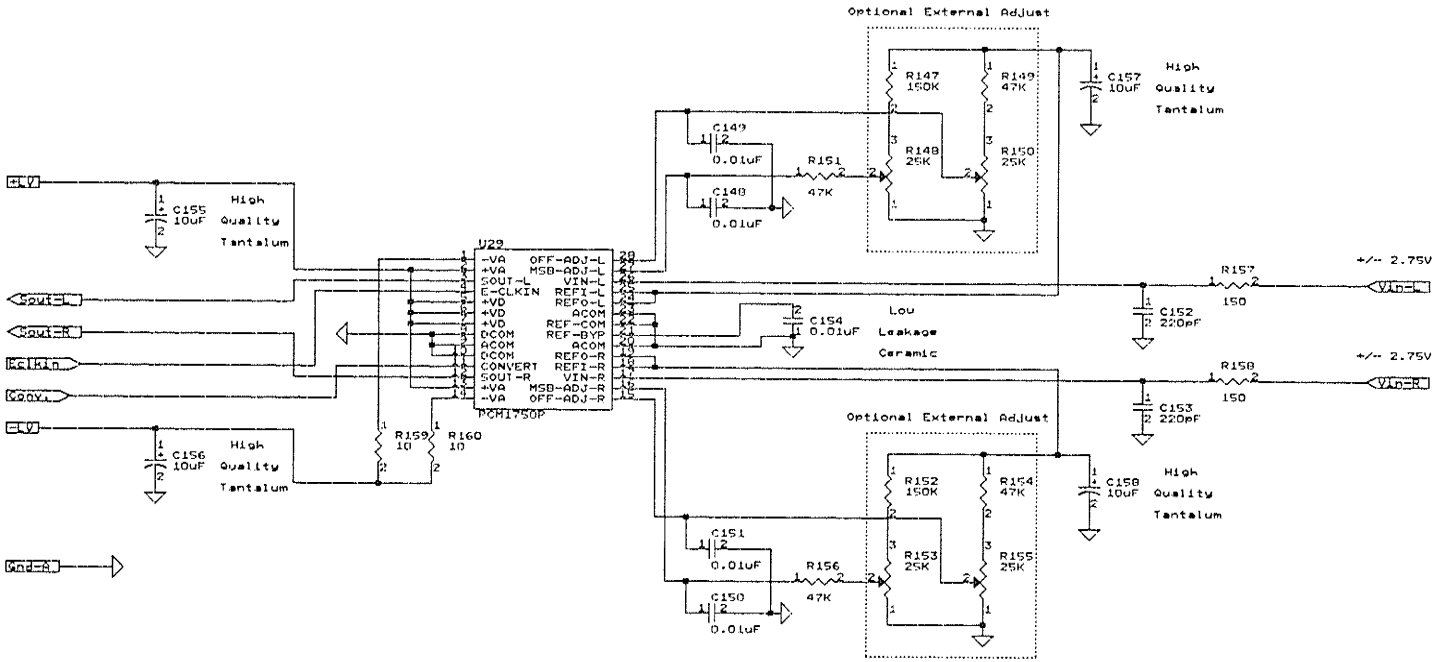
University of Manitoba		
Title: Optoisolator Interface		
Size: E	Document Number: 2 (File = OPTOIS00.SCH)	REV: A
Date: February 2, 1995	Sheet 20 of 27	

Analogue to Digital Converter



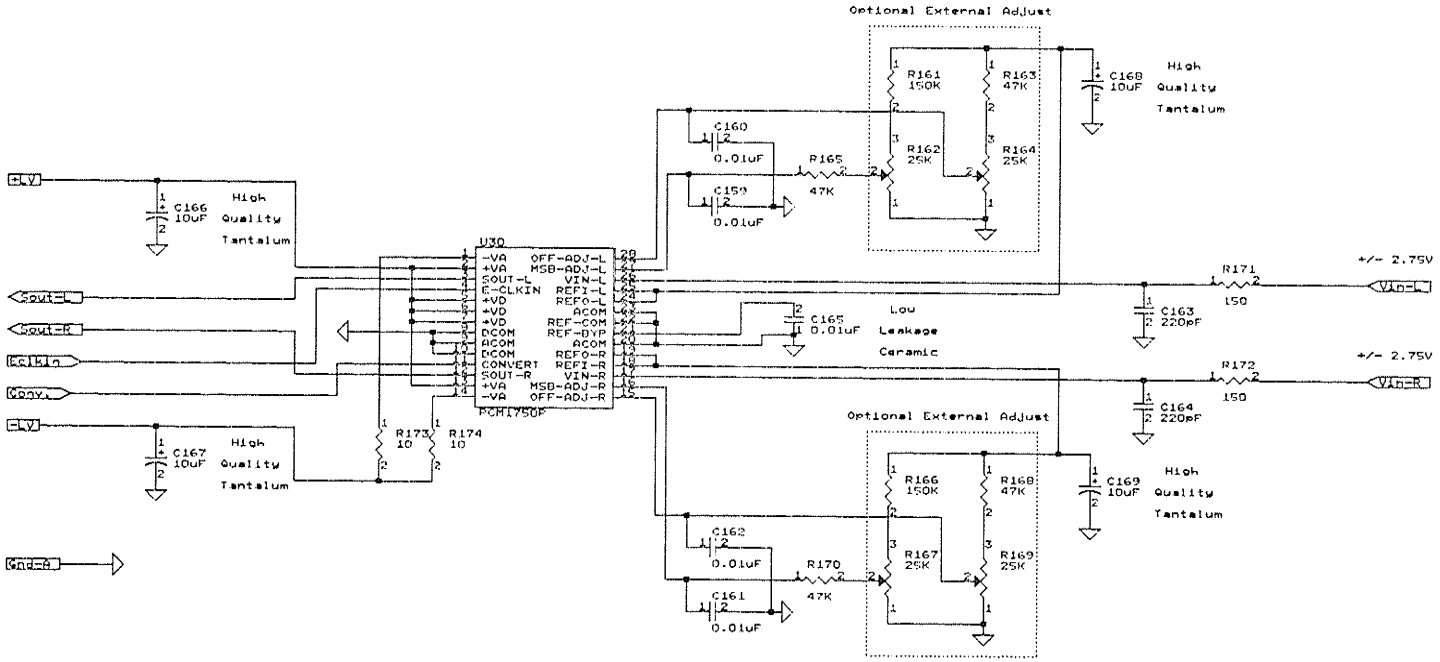
University Of Manitoba		
Title: Analogue to Digital Converter		
Size: C	Document Number: 2 (File = ANTODIG.SCH)	REV: A
Date: February 27, 1995	Sheet: 21 of 27	

Analogue to Digital Converter



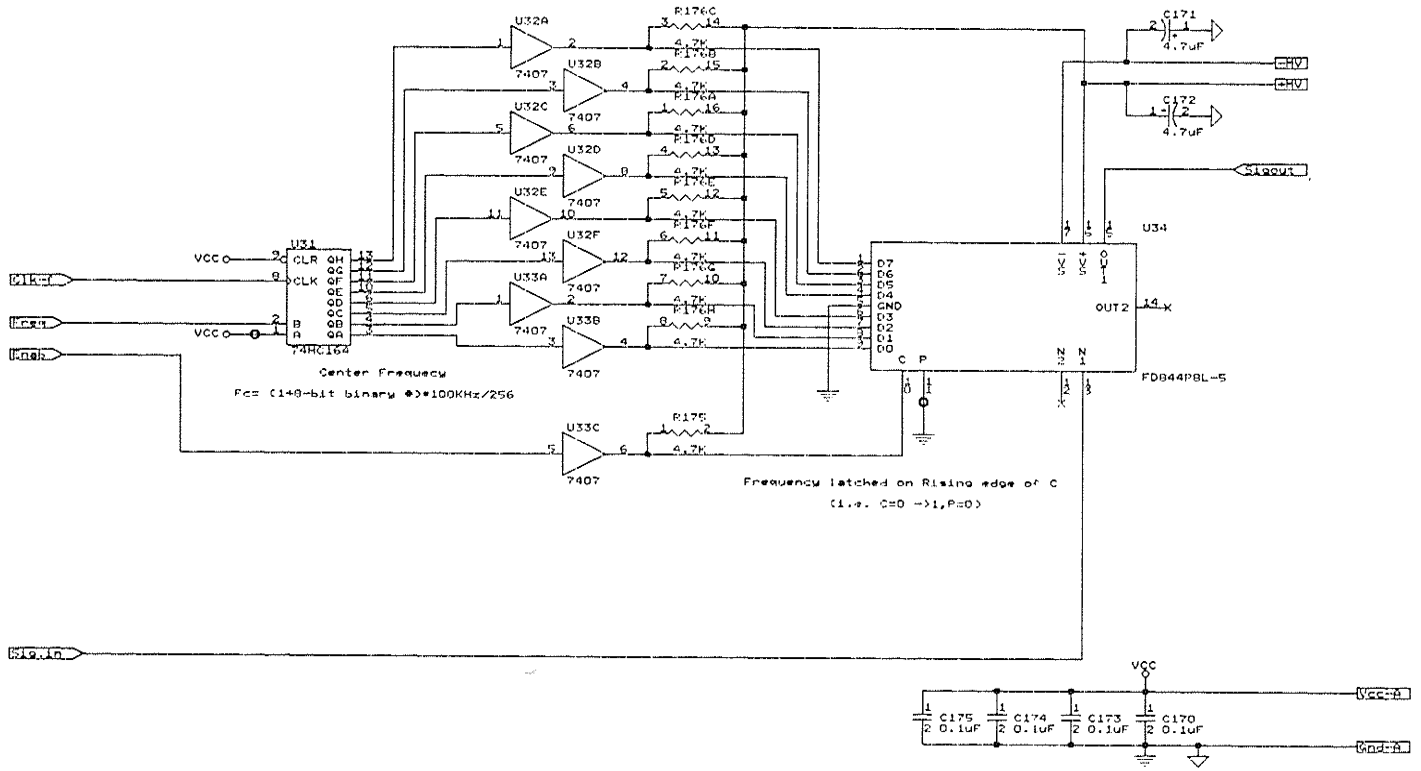
University Of Manitoba		
Title: Analogue to Digital Converter		
Size: E	Document Number: 2 (File = ANTODIGA.SCH)	REV: A
Date: February 2, 1995	Sheet 22 of 27	

Analogue to Digital Converter



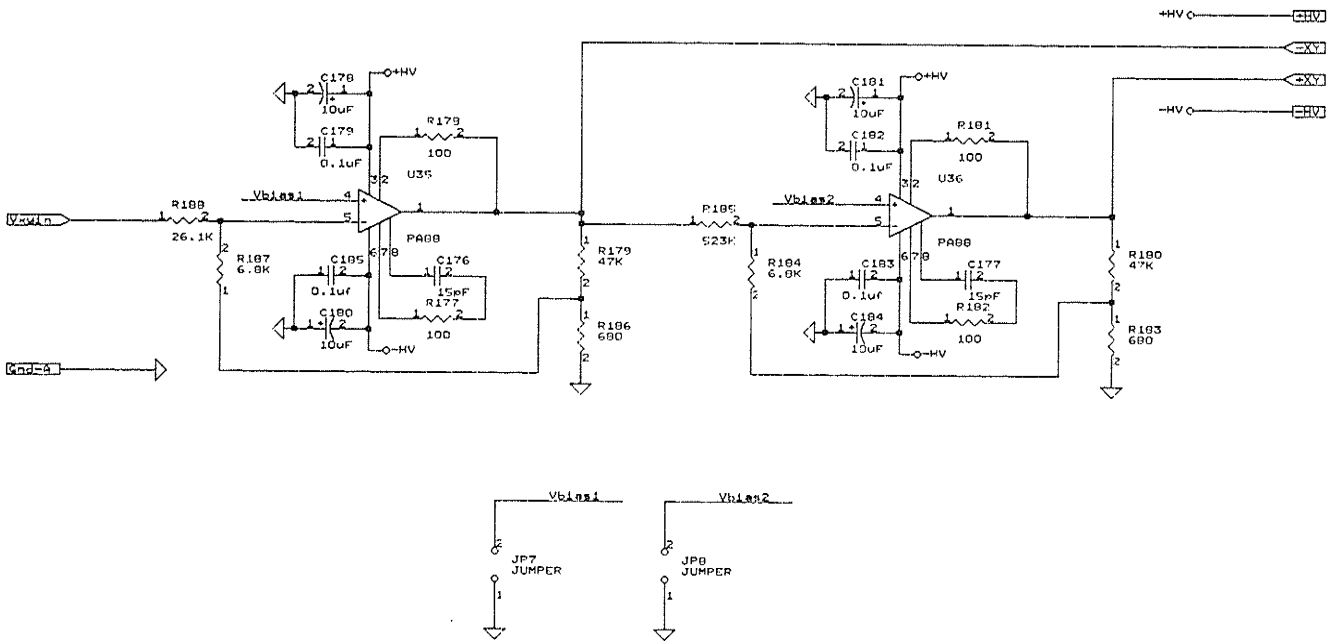
University Of Manitoba		
Title: Analogue to Digital Converter		
Size: E	Document Number: 2 (File = ANTODIGB.SCH)	REV: A
Date: February 2, 1995	Sheet 23 of 27	

Digitally Programmable Filter



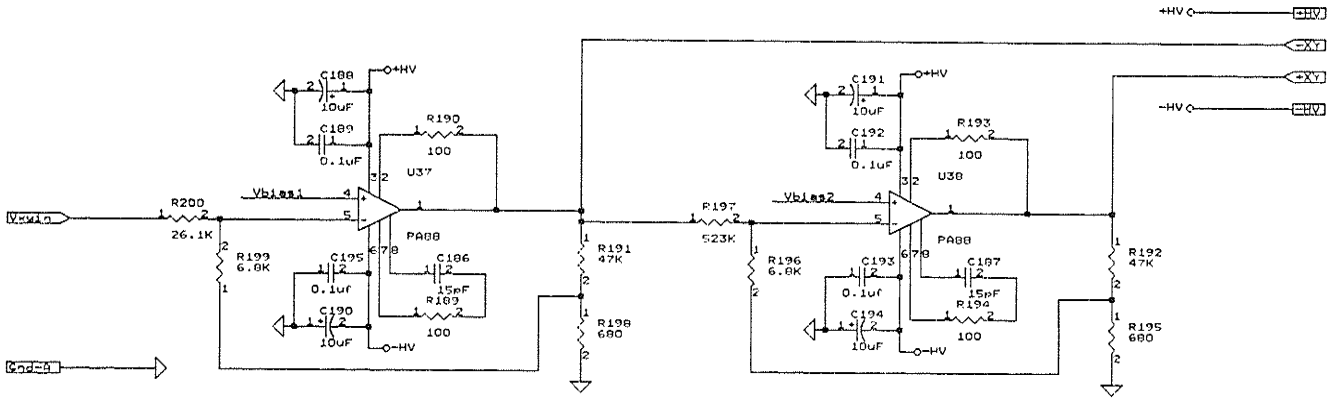
University of Manitoba		
Title: Digitally Programable Filter		
Size: E	Document Number: 2 (File = FDFILTER.SCH)	REV: A
Date: February 2, 1995	Sheet 24 of 27	

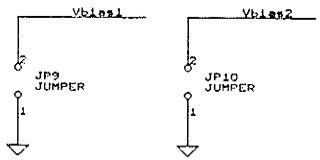
X or Y High Voltage Amplifiers



University of Manitoba		
Title: X & Y High Voltage Amplifier		
Size: E	Document Number: 2 (File = XYHVAMP.SCH)	REV: A
Date: February 2, 1995	Sheet 25 of 27	

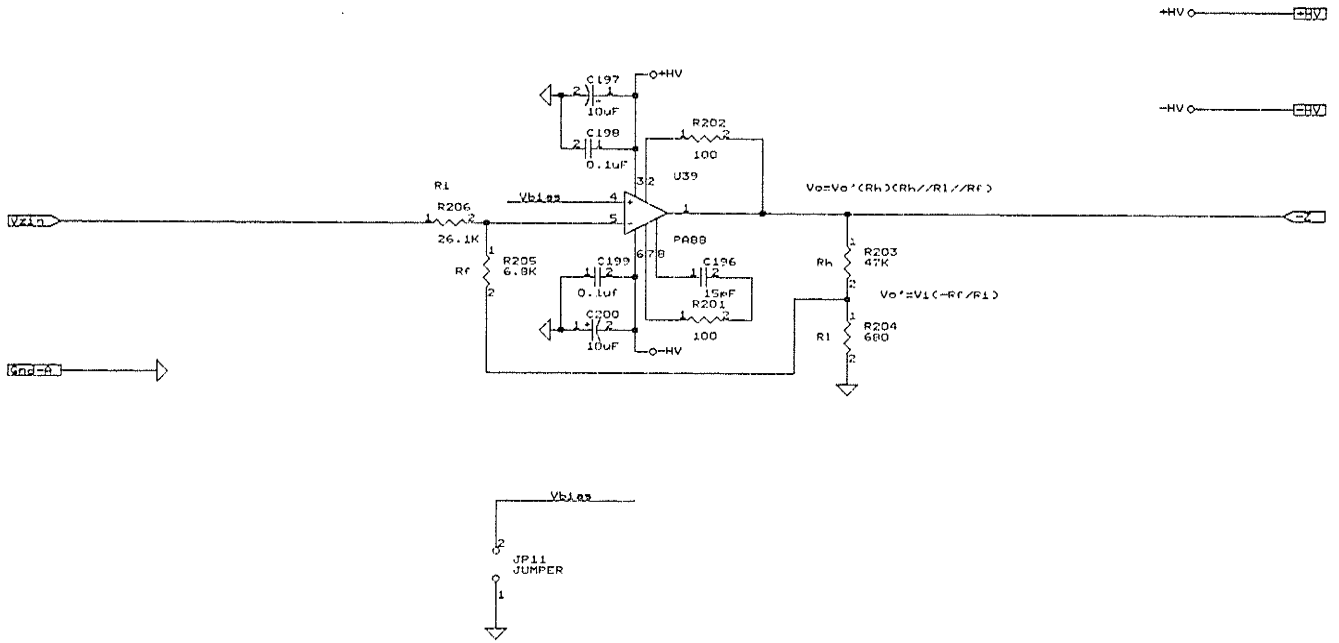
X or Y High Voltage Amplifiers





University of Manitoba		
Title: X & Y High Voltage Amplifier		
Size: E	Document Number: 2 (File = XYHVAMPA.SCH)	REV: A
Date: February 2, 1995	Sheet 26 of 27	

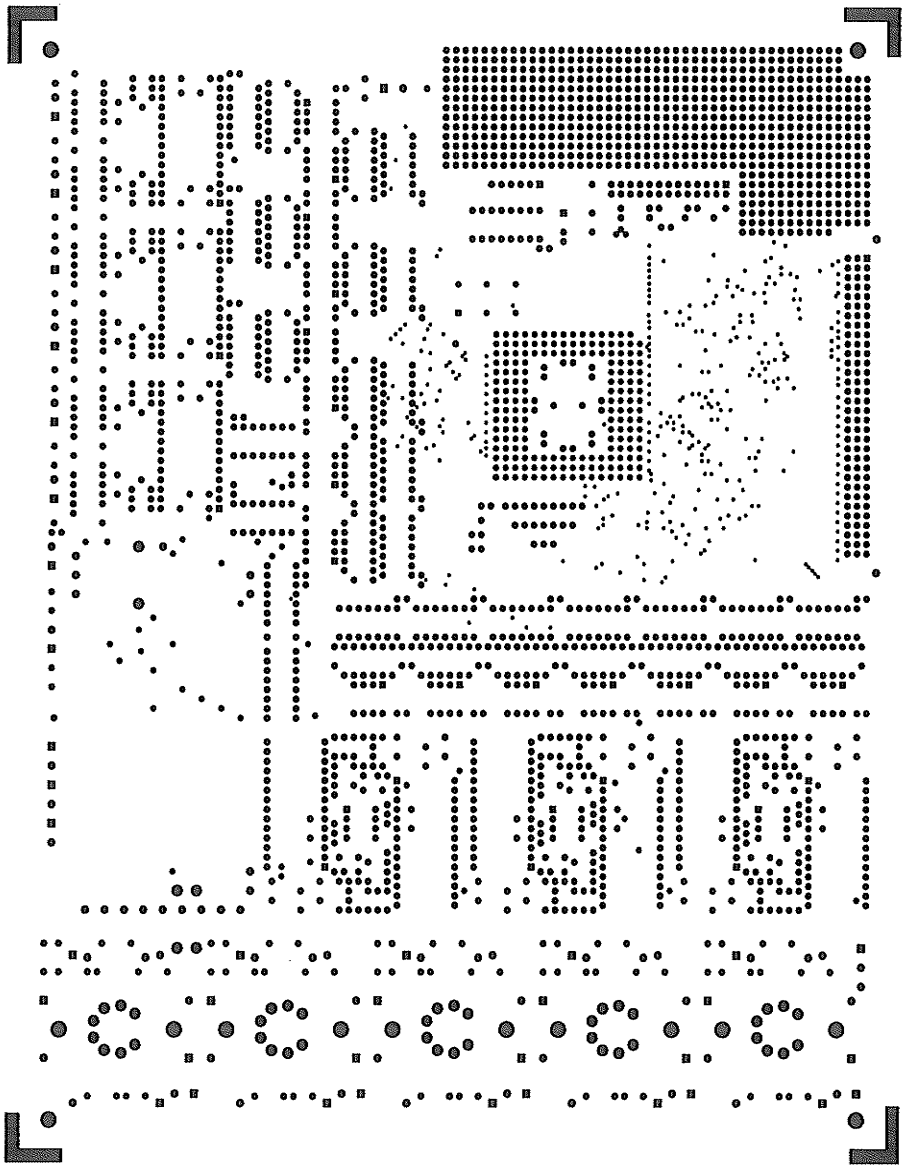
Z High Voltage Amplifier



University of Manitoba		
Title: Z High Voltage Amplifier		
Size: E	Document Number: 2 (File = ZHVAMP.SCH)	REV: A
Date: February 2, 1995	Sheet 27 of 27	

Appendix B

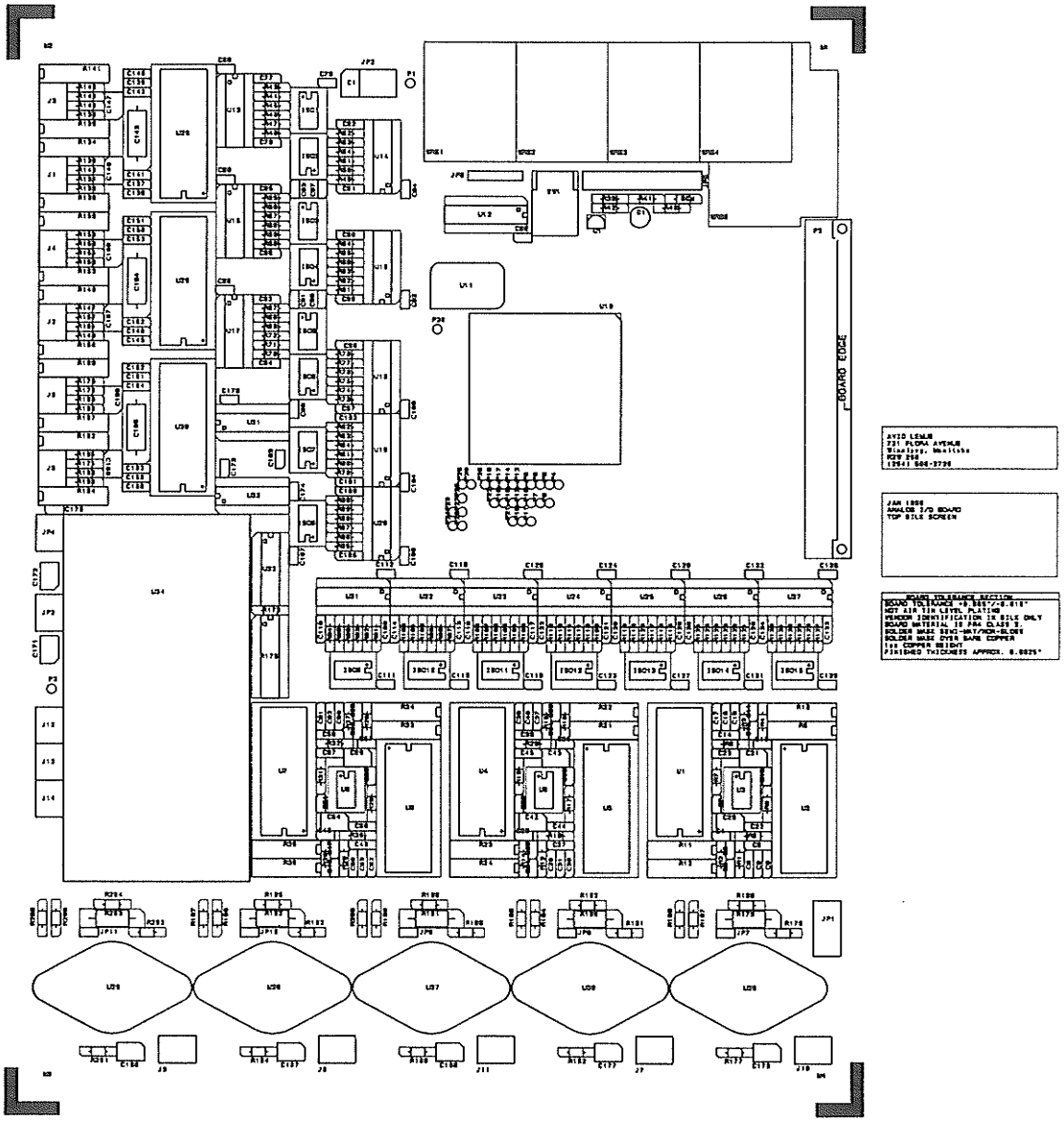
Mask Plots for PCB Construction of the Analogue I/O Board

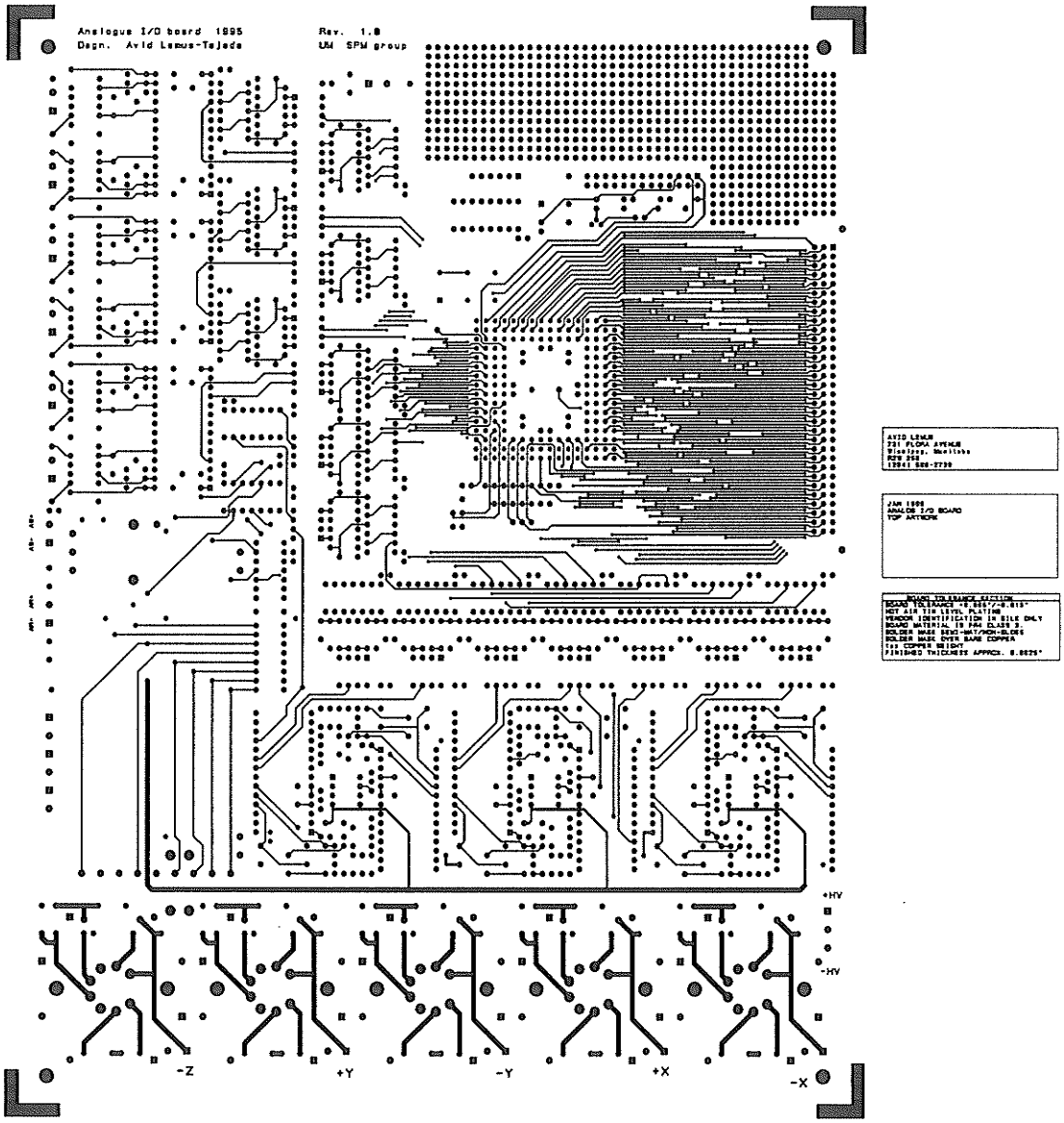


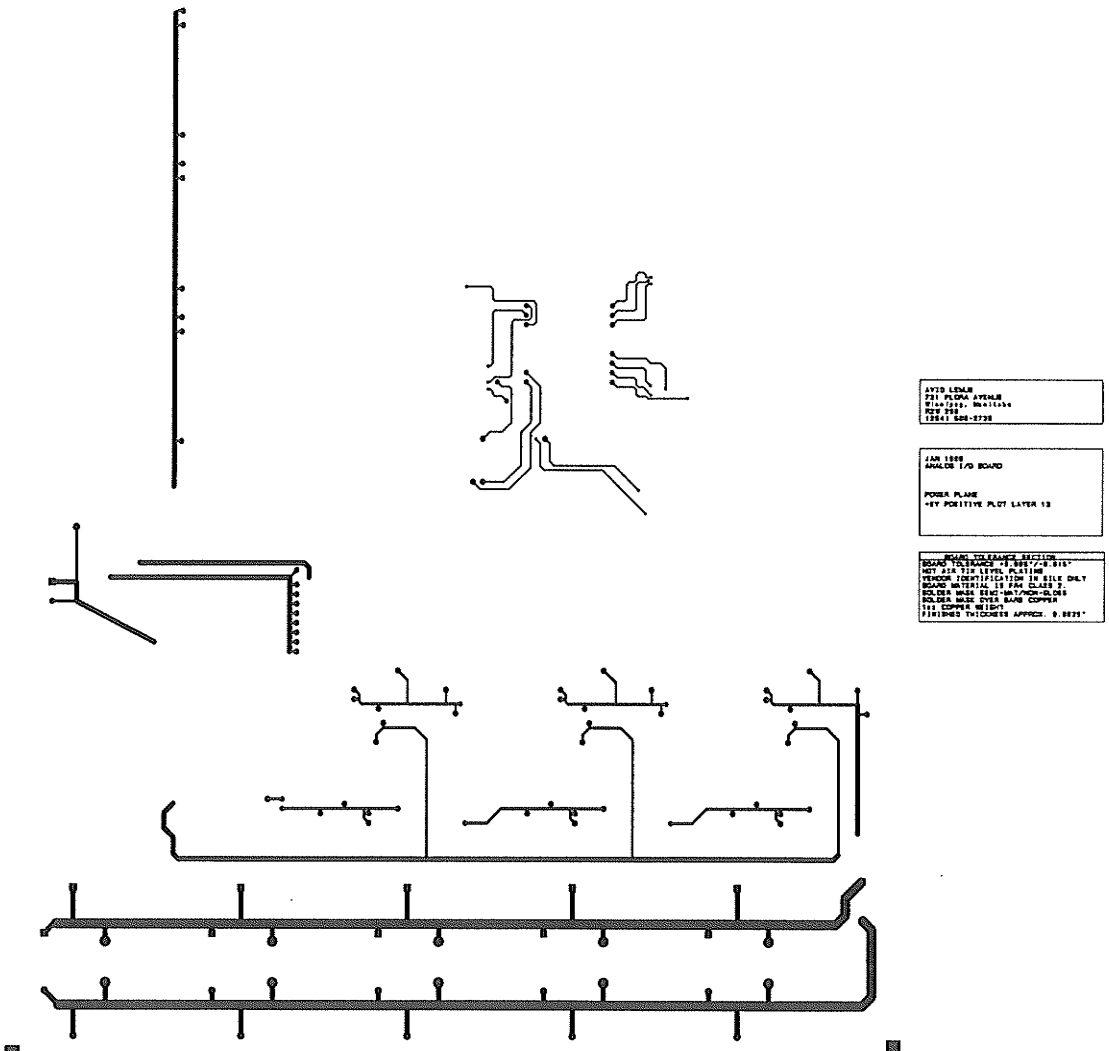
AVID LEMUR
731 FLORA AVENUE
Winnipeg, Manitoba
R2K 2S8
(204) 666-2730

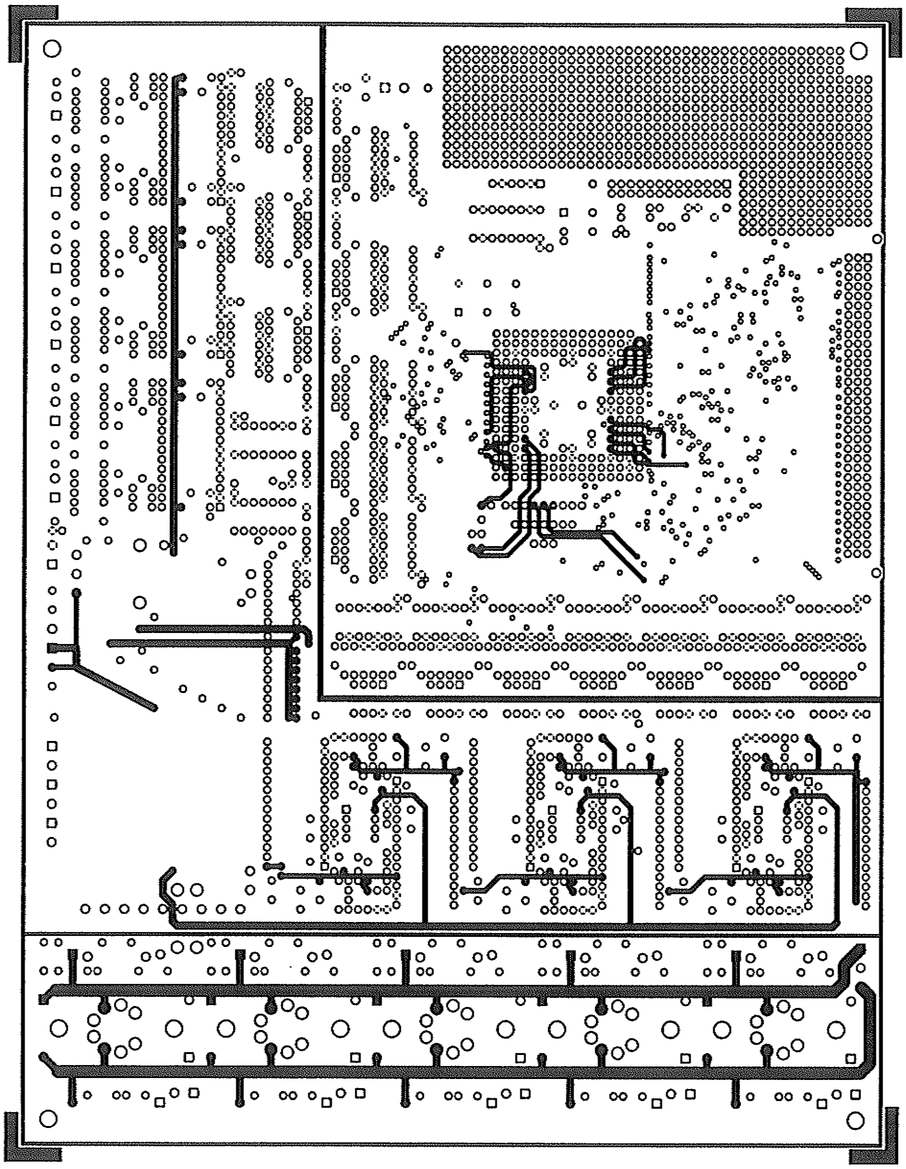
JAN 1996
ANALOG I/O BOARD
TOP SOLDER RESIST

BOARD TOLERANCE_SECTION
BOARD TOLERANCE +0.001"/-0.010"
HOT AIR TIN LEVEL PLATING
VENDOR IDENTIFICATION ON SILK ONLY
BOARD MATERIAL IS FR4 CLASS 2.
SOLDER MASK SEMI-MAT/MOD-SLOBS
SOLDER MASK OVER BARE COPPER
1oz COPPER WEIGHT
FINISHED THICKNESS APPROX. 0.0025"


BOARD EDGE


Analogue I/O board 1995
Rev. 1.0
-Z
+Y
-Y
+X
-X
+HV
-HV


AVID LEAD
721 FLORA AVENUE
Winnipeg, Manitoba
R2K 2R6
(204) 668-5738
JAN 1998
ANALOG I/O BOARD
POWER PLANE
+5V POSITIVE PLOT LAYER 13
BOARD TOLERANCE SPECIFICATION
BOARD TOLERANCE +0.005"/-0.015"
HOT AIR TIN LEVEL PLATING
TENDOR IDENTIFICATION IN SILK ONLY
BOARD MATERIAL IS FR4 CLASS 2
SOLDER MASK 5INC-WHITE/NON-GLOSS
SOLDER MASK OVER BARE COPPER
1oz COPPER WEIGHT
FINISHED THICKNESS APPROX. 0.0075"



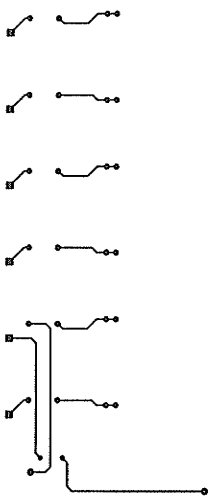
[illegible]
[illegible] FLORA AVENUE
Winnipeg, Manitoba
[illegible]
[illegible]

JAN [illegible]
ANALOG I/O BOARD

POWER PLANE

+5V NEGATIVE PLOT LAYER 13

BOARD TOLERANCE SECTION
BOARD TOLERANCE [illegible]
HOT AIR TIN LEVEL PLATING
VENDOR IDENTIFICATION IN SILK ONLY
BOARD MATERIAL IS FR4 CLASS 3.
SOLDER MASK SEMI-MAT/NON-GLOSS
SOLDER MASK OVER BARE COPPER
1oz COPPER WEIGHT
FINISHED THICKNESS APPROX. [illegible]



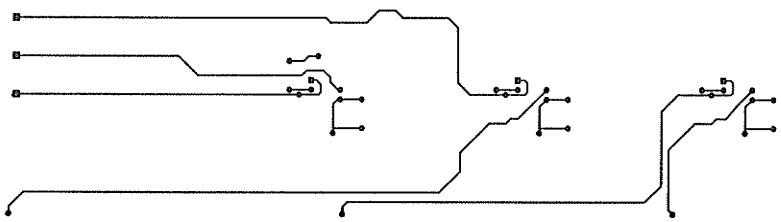
AVID LEMLM
721 FLORA AVENUE
Winnipeg, Manitoba
R2R 2S0
(204) 668-2798

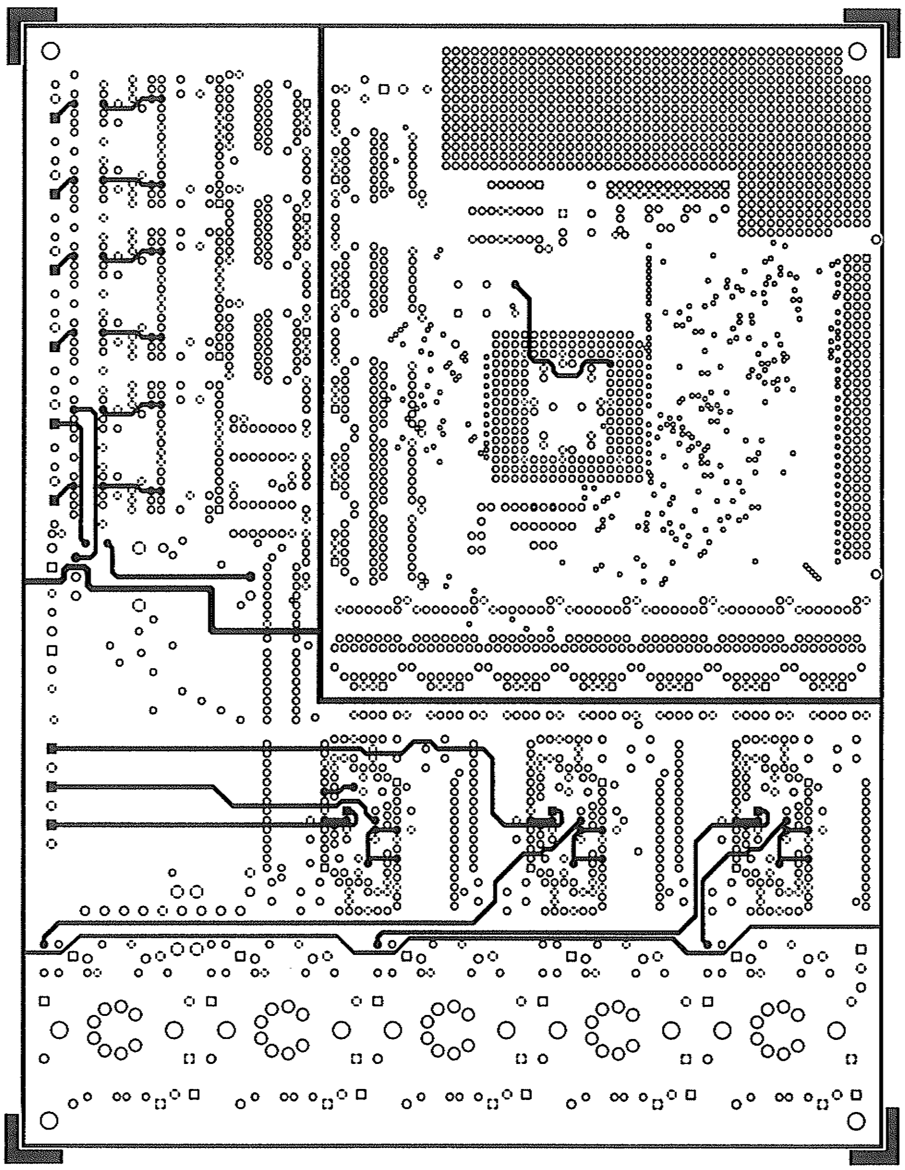
JAN 1998
ANALOG I/O BOARD

POWER PLANE

GND POSITIVE PLOT LAYER 4

BOARD TOLERANCE +0.005"/-0.010"
HOT AIR TIN LEVEL PLATING
VENDOR IDENTIFICATION IN SILK ONLY
BOARD MATERIAL IS FR4 CLASS 2
SOLDER MASK EXCEPT/NON-GLOSS
SOLDER MASK OVER BARE COPPER
1oz COPPER WEIGHT
FINISHED THICKNESS APPROX. 0.062"





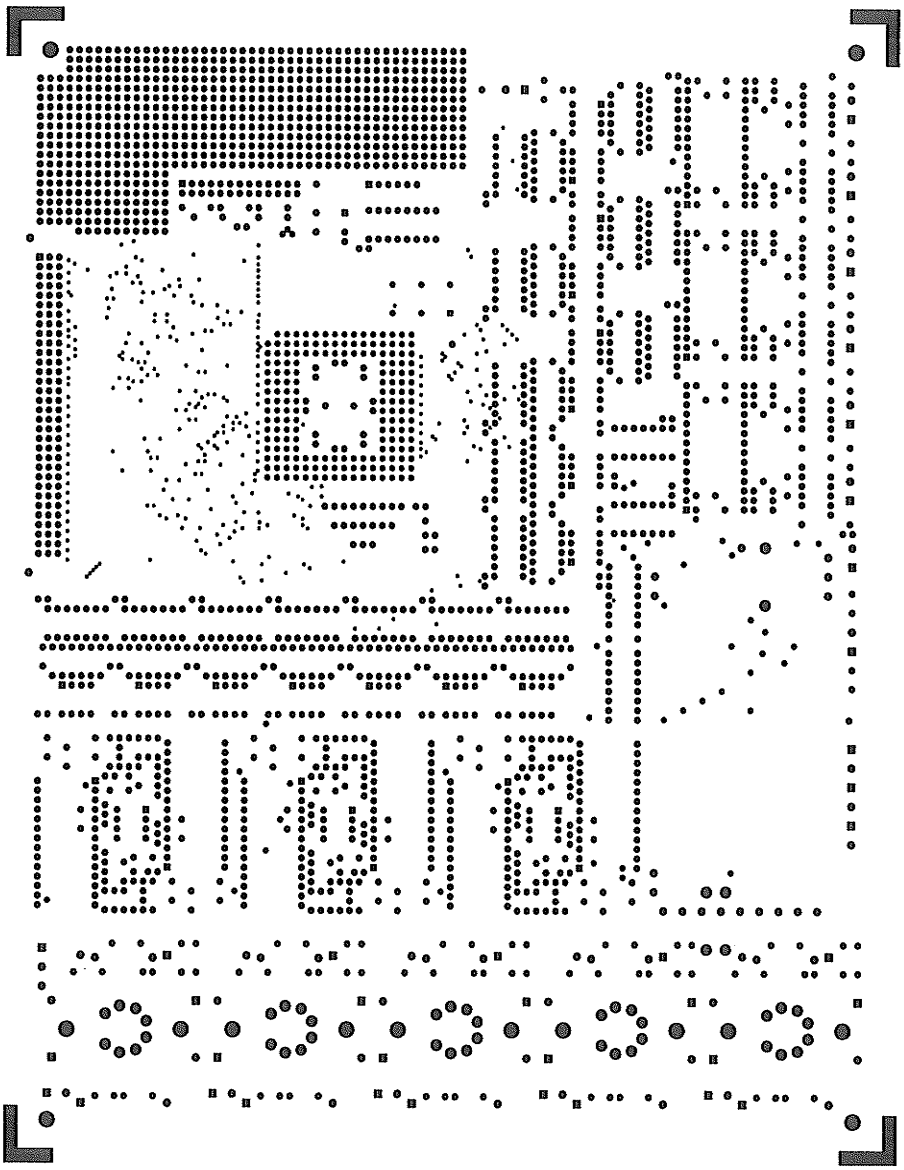
AVID LEWIS
721 FLORA AVENUE
Winnipeg, Manitoba
R2W 2S4
(204) 586-2731

JAN 199[illegible]
ANALOG I/O BOARD

POWER PLANE

END NEGATIVE PLOT LAYER 4

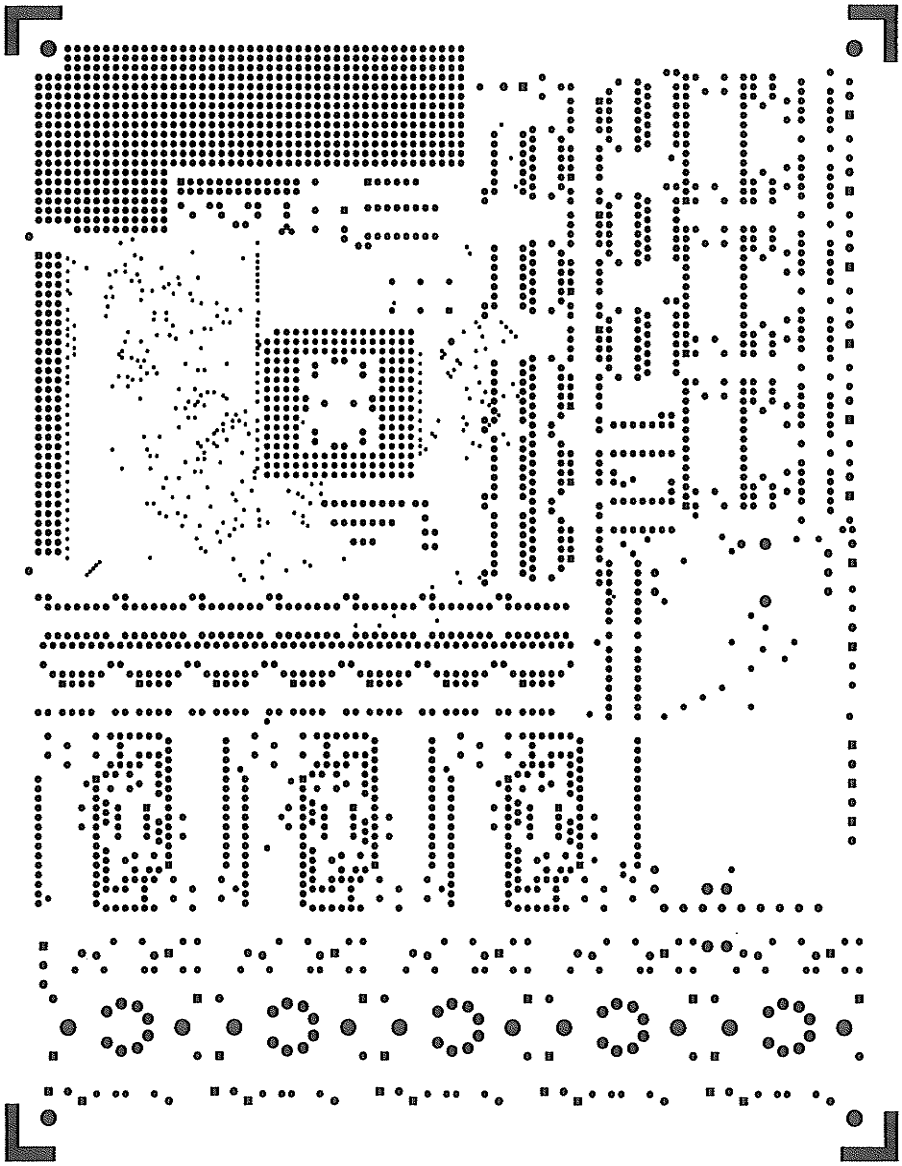
BOARD TOLERANCE SECTION
BOARD TOLERANCE +0.005"/-0.010"
HOT AIR TIN LEVEL PLATING
VENDOR IDENTIFICATION IN SILK ONLY
BOARD MATERIAL IS FR4 CLASS 3.
SOLDER MASK SEMI-MAT/NON-GLOSS
SOLDER MASK OVER BARE COPPER
1oz COPPER WEIGHT
FINISHED THICKNESS APPROX. 0.062"



AVID LEMUR
701 FLORA AVENUE
Stratford, Manitoba
R0C 2Z0
(204) 646-3799

JAN 1995
ANALOG I/O BOARD
SOT SOLDER RESIST

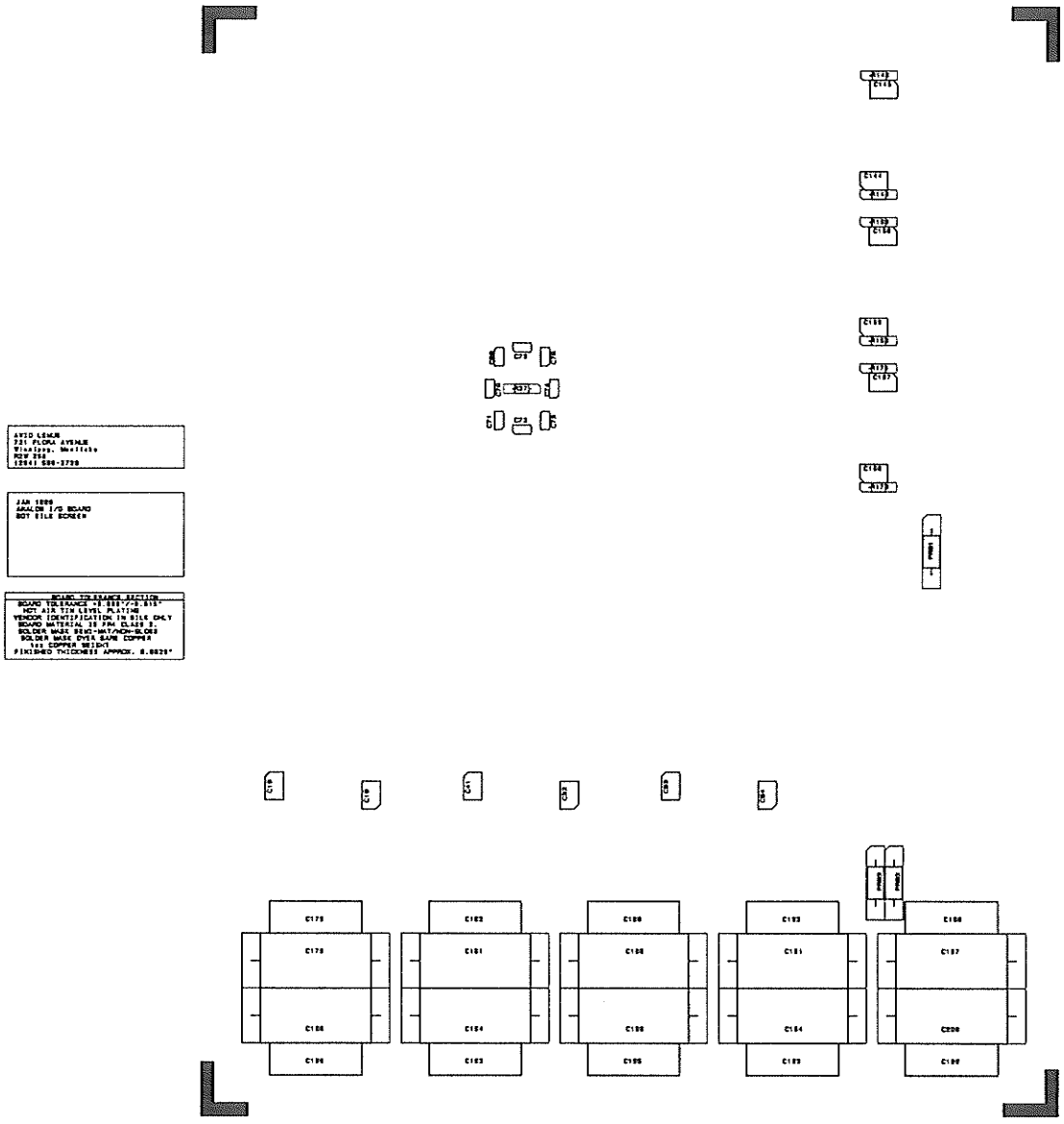
BOARD TOLERANCE SECTION
BOARD TOLERANCE +0.005"/-0.010"
HOT AIR TIN LEVEL PLATING
VENDOR IDENTIFICATION IN FILE ONLY
BOARD MATERIAL IS FR4 CLASS 2
SOLDER MASK EXIC-MAT/VCM-8L02
SOLDER MASK OVER BARE COPPER
1oz COPPER WEIGHT
FINISHED THICKNESS APPROX. 0.0625"

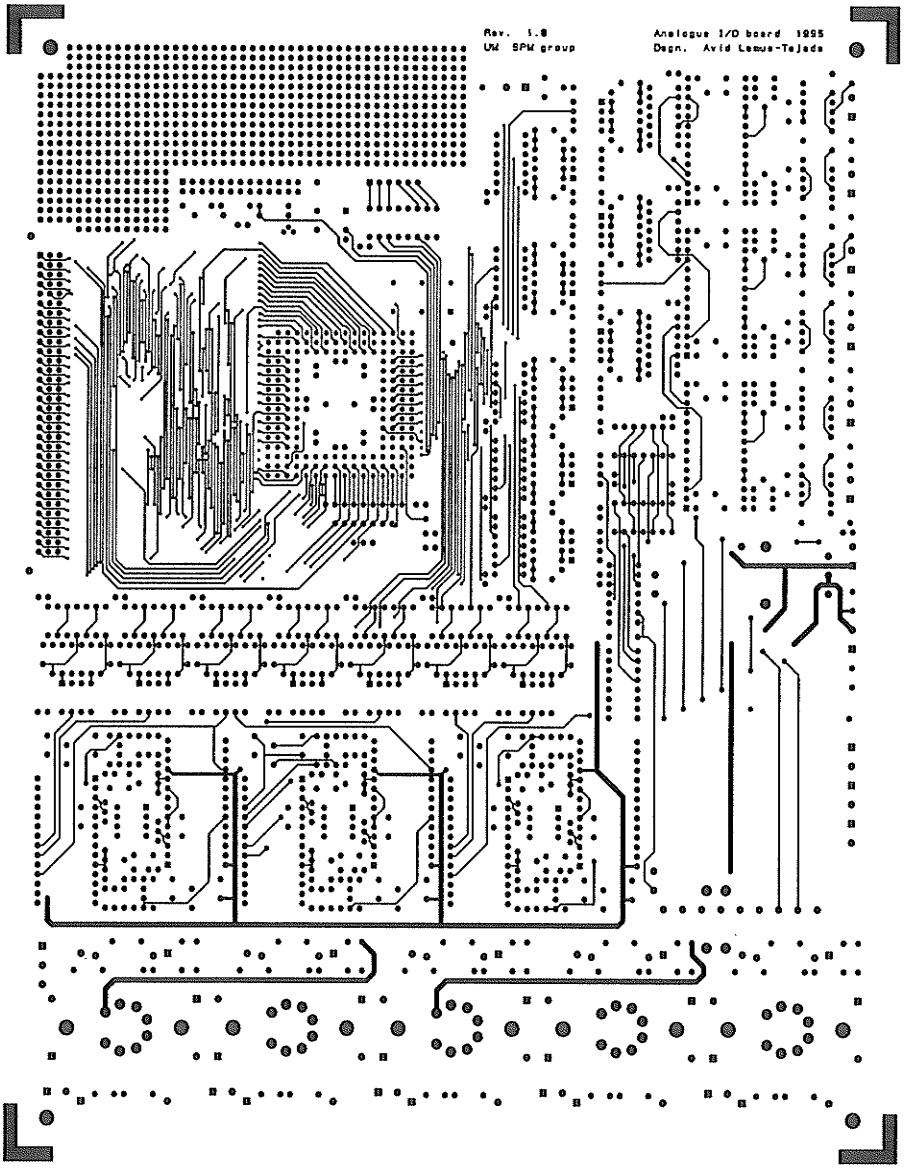


AVID LEHER
731 FLORA AVENUE
Kinston, Montana
406 756
(914) 566-3726

JAN 1998
ANALOG I/O BOARD
BOT SOLDER RESIST

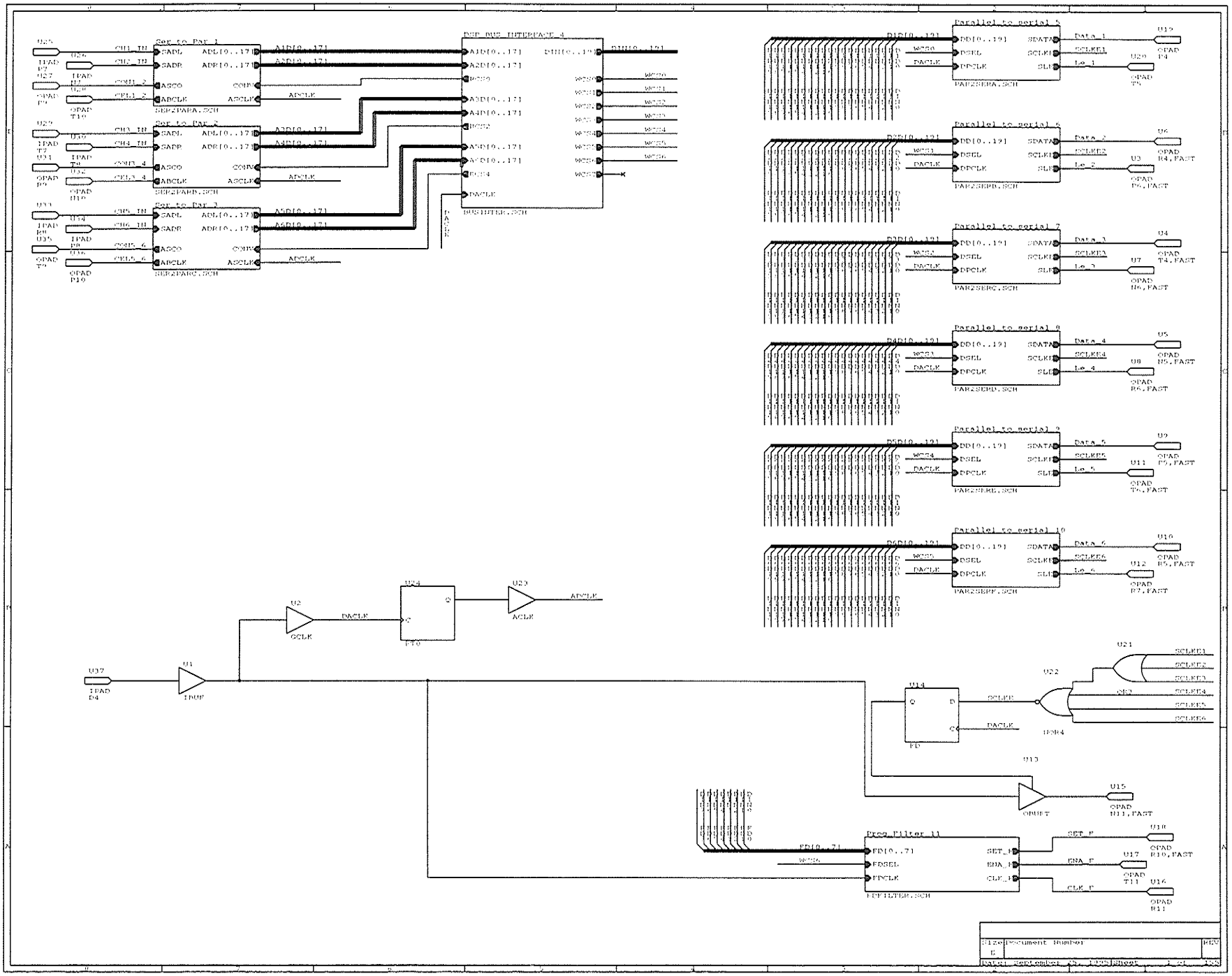
BOARD TOLERANCE EXCEPTION
BOARD TOLERANCE +0.005"/-0.010"
HOT AIR TIN LEVEL PLATING
VENDOR IDENTIFICATION IN SILK ONLY
BOARD MATERIAL IS FR4 CLASS 7
SOLDER MASK SEMI-MATHON-SLOSH
SOLDER MASK OVER BARE COPPER
1oz COPPER WEIGHT
FINISHED THICKNESS APPROX. 0.0070"

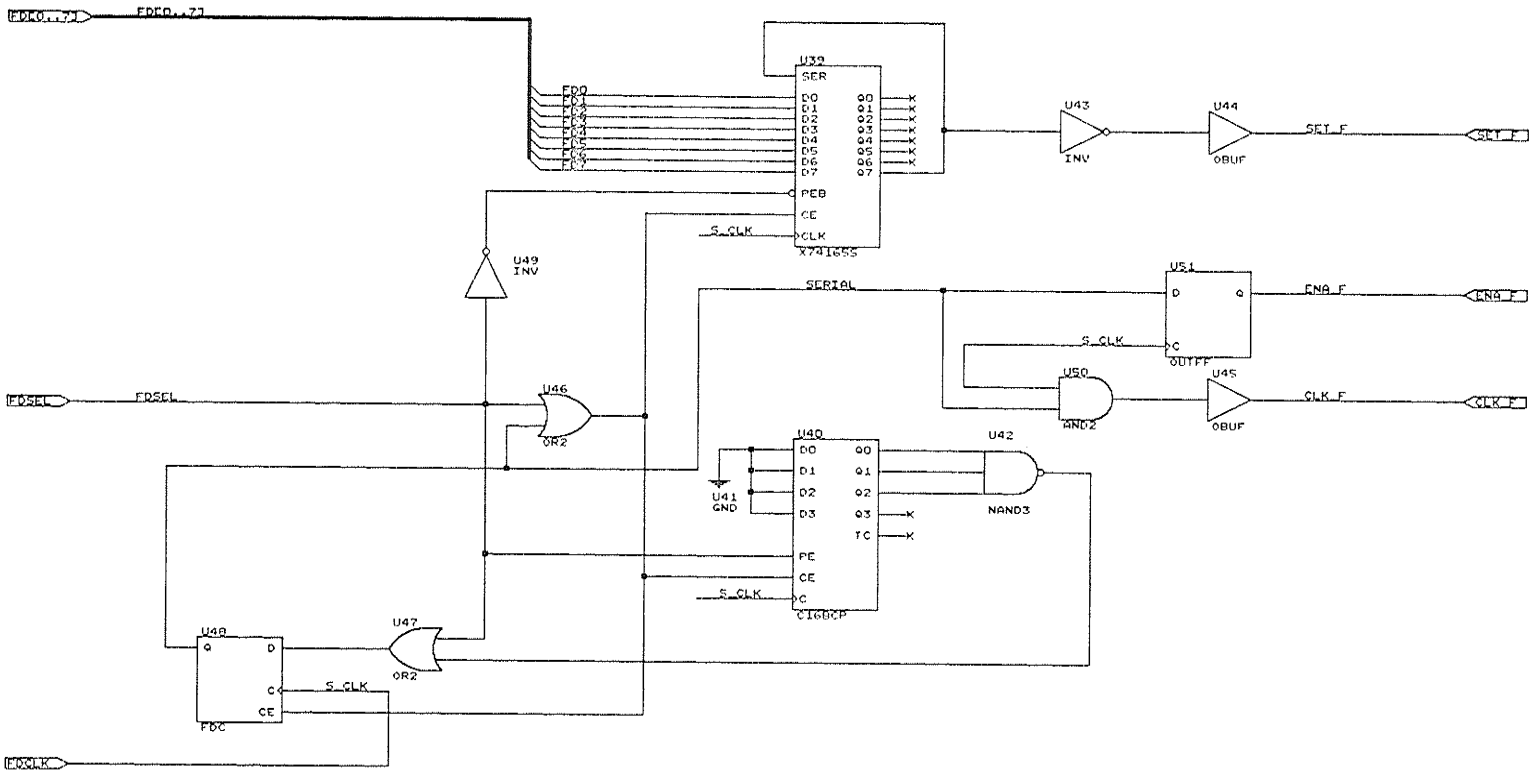



Rev. 1.0
UW SPM group
Analogue I/O board 1995
Dsgn. Avid Lemus-Tejada

Appendix C

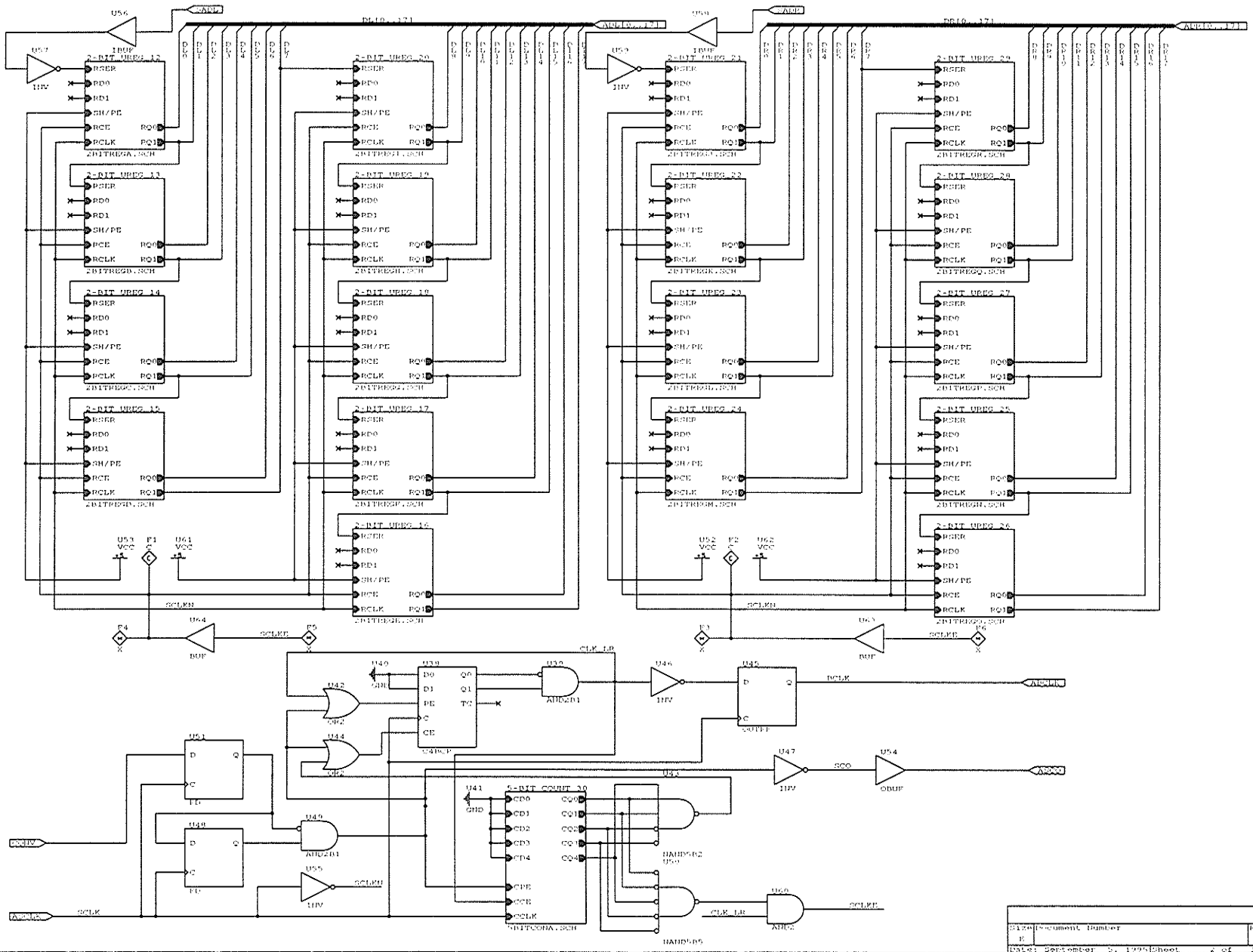
Schematic Representation Of Xilinx XC3195 Field Programmable Gate Array Configuration

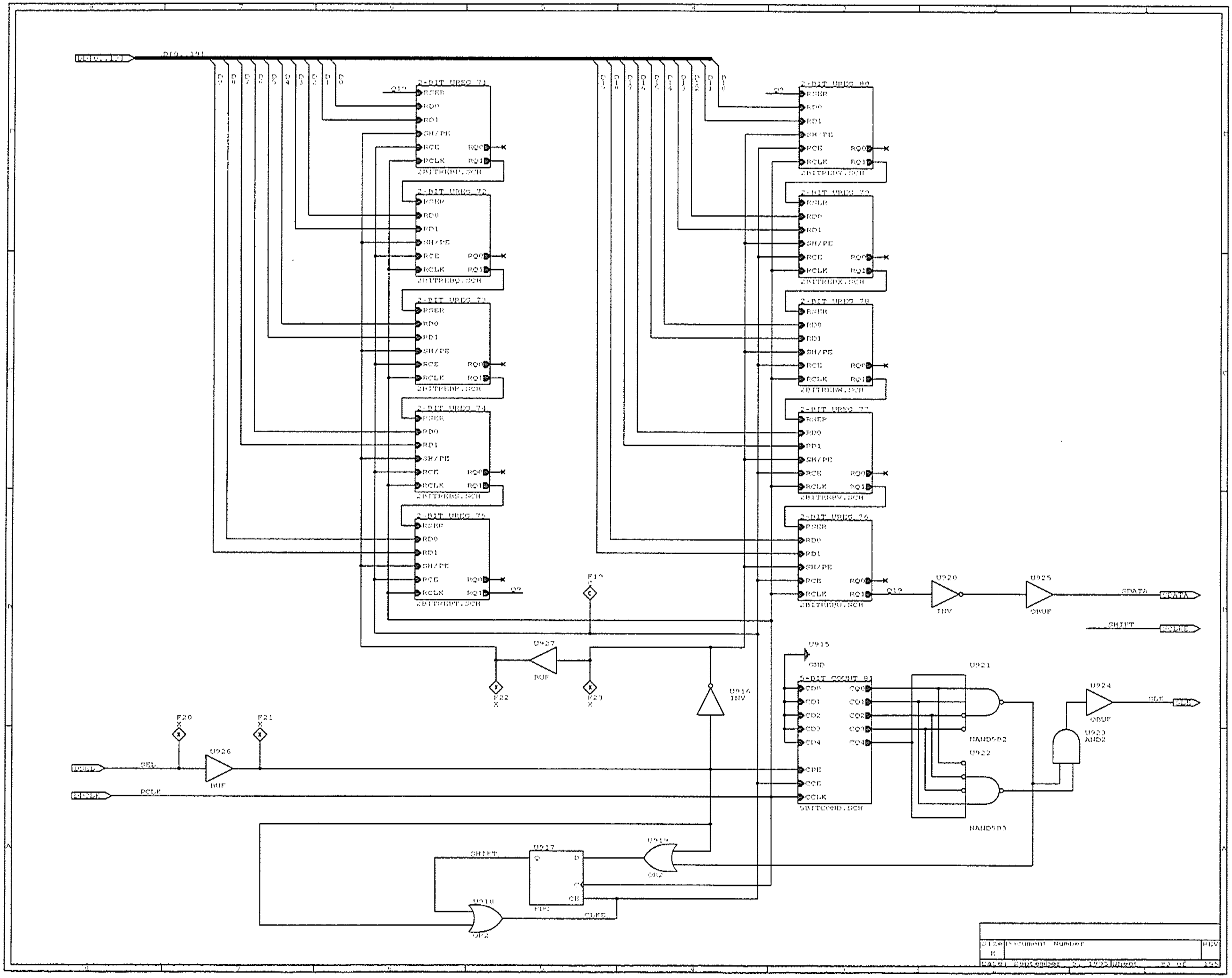

SER2PARA.SCH
SER2PARB.SCH
SER2PARC.SCH
DSP_DUO_INTERFACE_4
DUO2PFB6.SCH
PAR2SERA.SCH
PAR2SERB.SCH
PAR2SERC.SCH
PAR2SERD.SCH
PAR2SERE.SCH
PAR2SERF.SCH
PFILTER.SCH
ADCLK
DACLK
GCLK
ACLK
OBUF
IBUF
FD
OBUFT
IPAD
OPAD



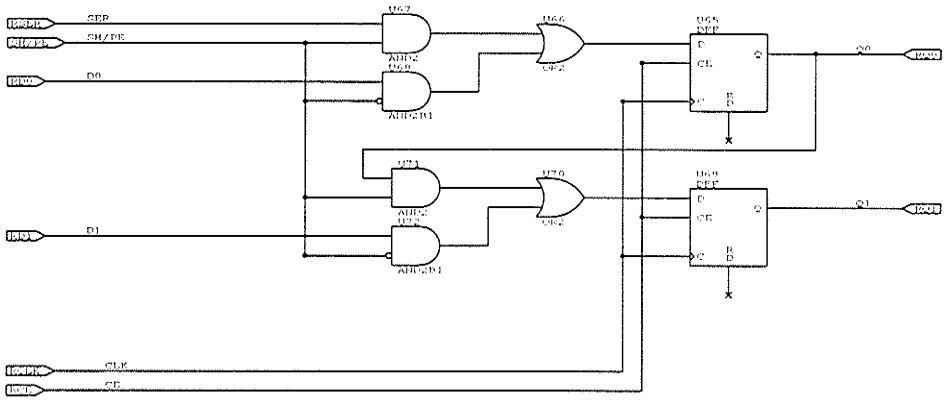
Size	Document Number	REV
C		
Date: June 29, 1995	Sheet 2 of 15	

Dual Serial to Parallel Converter

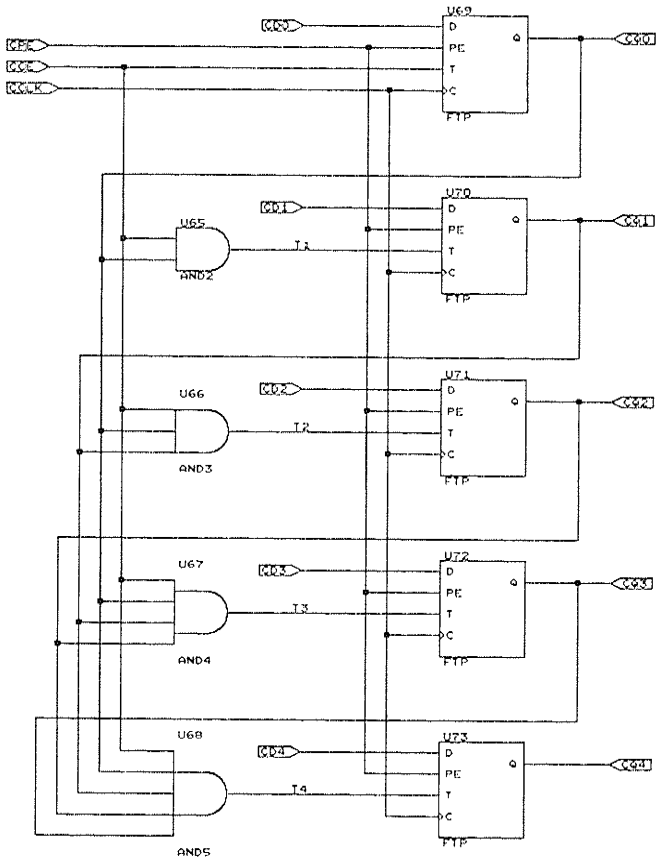



D[0..19]
2-BIT UREG 71
2-BIT UREG 72
2-BIT UREG 73
2-BIT UREG 74
2-BIT UREG 75
2-BIT UREG 76
2-BIT UREG 77
2-BIT UREG 78
2-BIT UREG 79
2-BIT UREG 80
RSER
RD0
RD1
SH/PE
RCE
RCLK
RQ0
RQ1
U920
INV
U925
OBUF
SDATA
SHIFT
U915
GND
U921
NAND5B2
U922
NAND5B3
U923
AND2
U924
OBUF
SLE
U926
BUF
U927
BUF
U916
INV
U917
FDC
U918
OR2
U919
OR2
SEL
PCLK
CLKE
CD0
CD1
CD2
CD3
CD4
CQ0
CQ1
CQ2
CQ3
CQ4
CPE
CCE
CCLK
Size
E
Document Number
REV
Date: September 5, 1995 Sheet
of

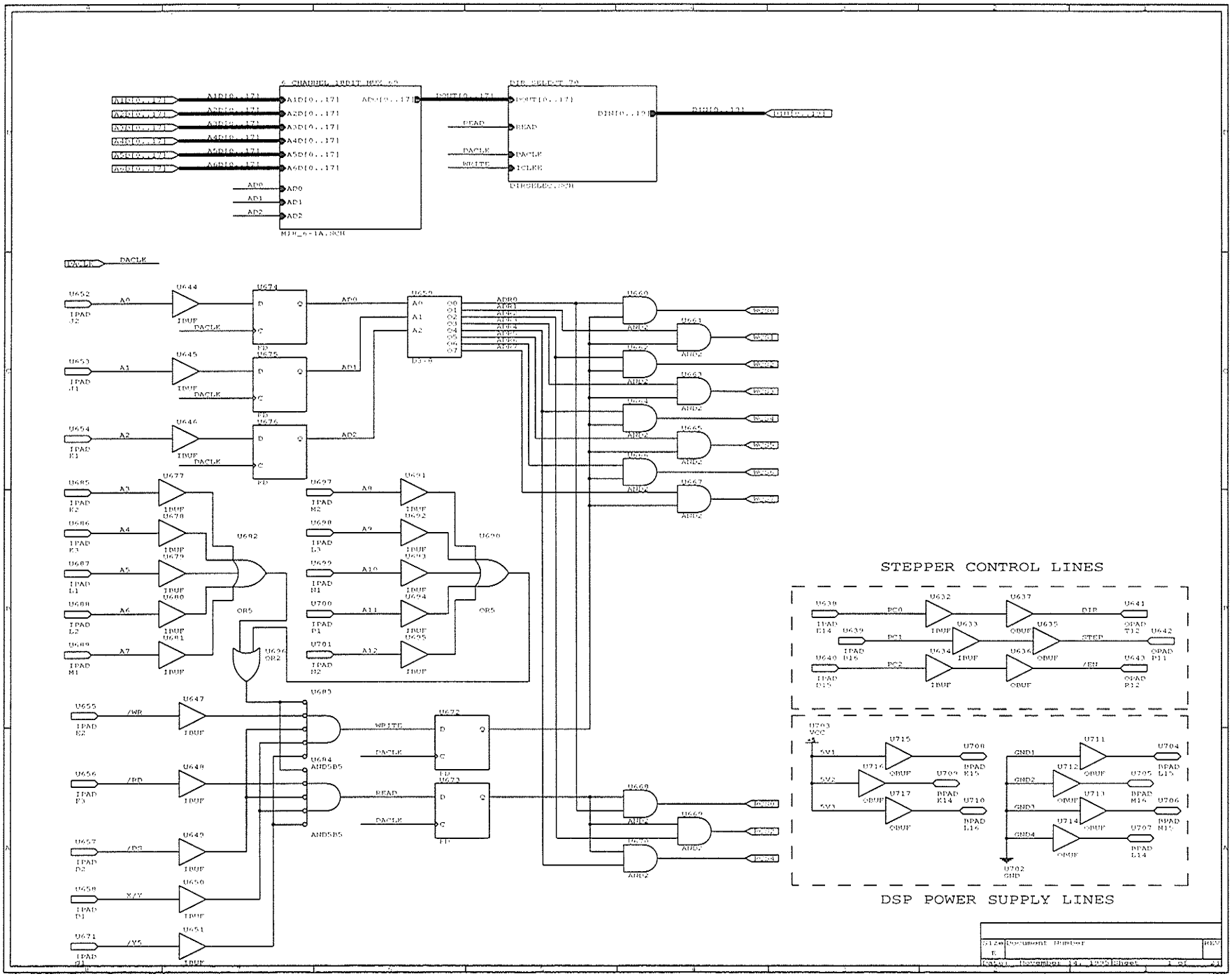
2-bit Par/Ser-In Par/Ser-Out,CE,/PE



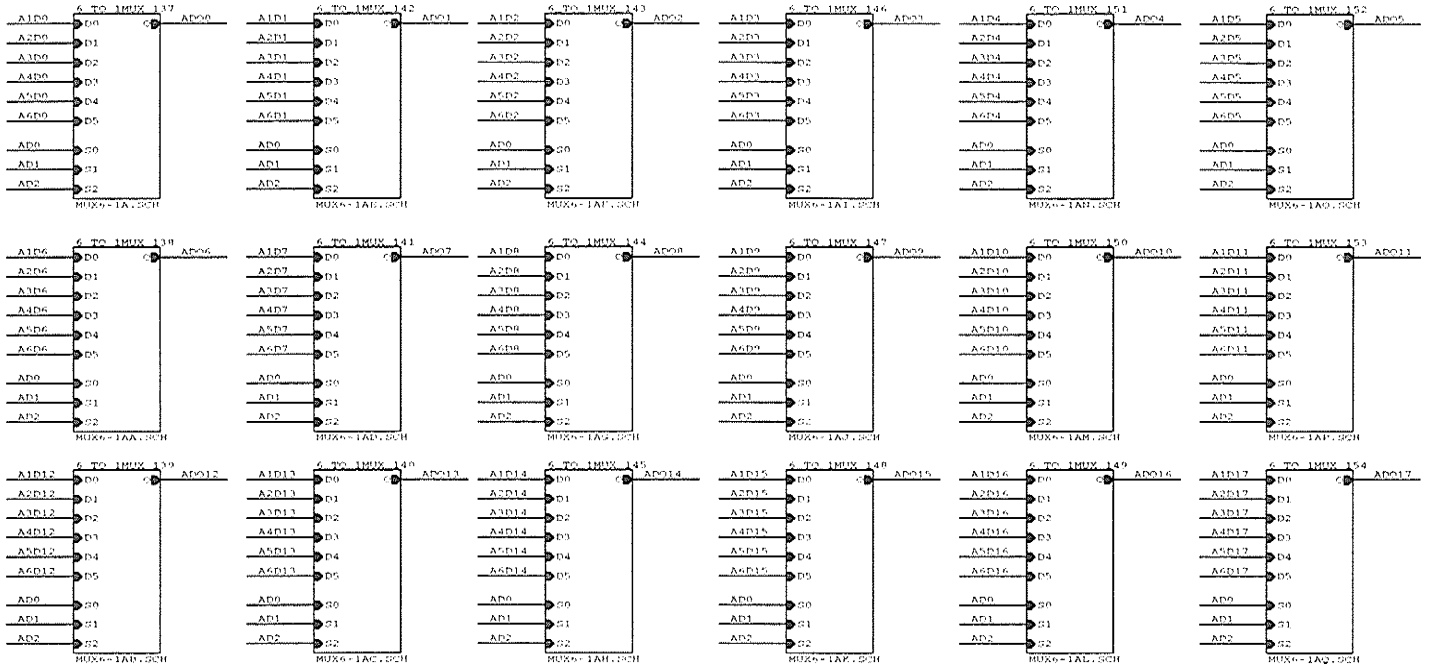
5-bit presetable counter

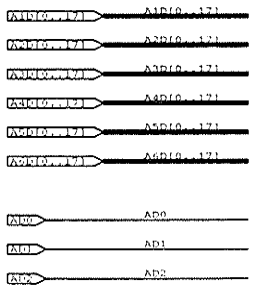


Size	Document Number	REV
C		
Date: December 12, 1995	Sheet 4 of 165	

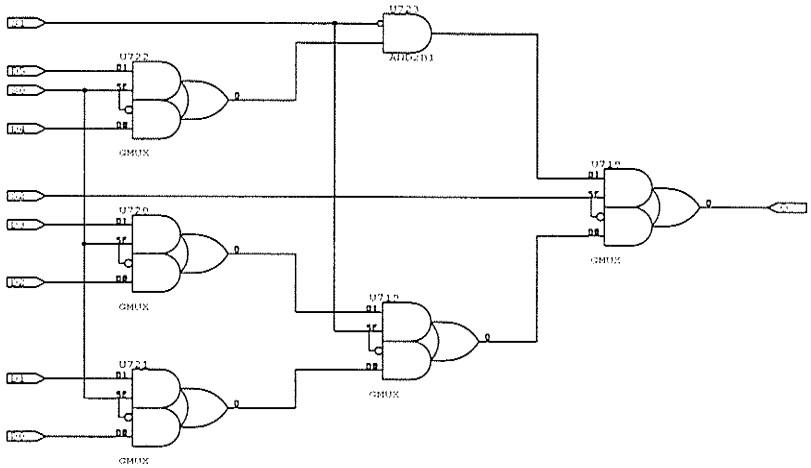

STEPPER CONTROL LINES
DSP POWER SUPPLY LINES

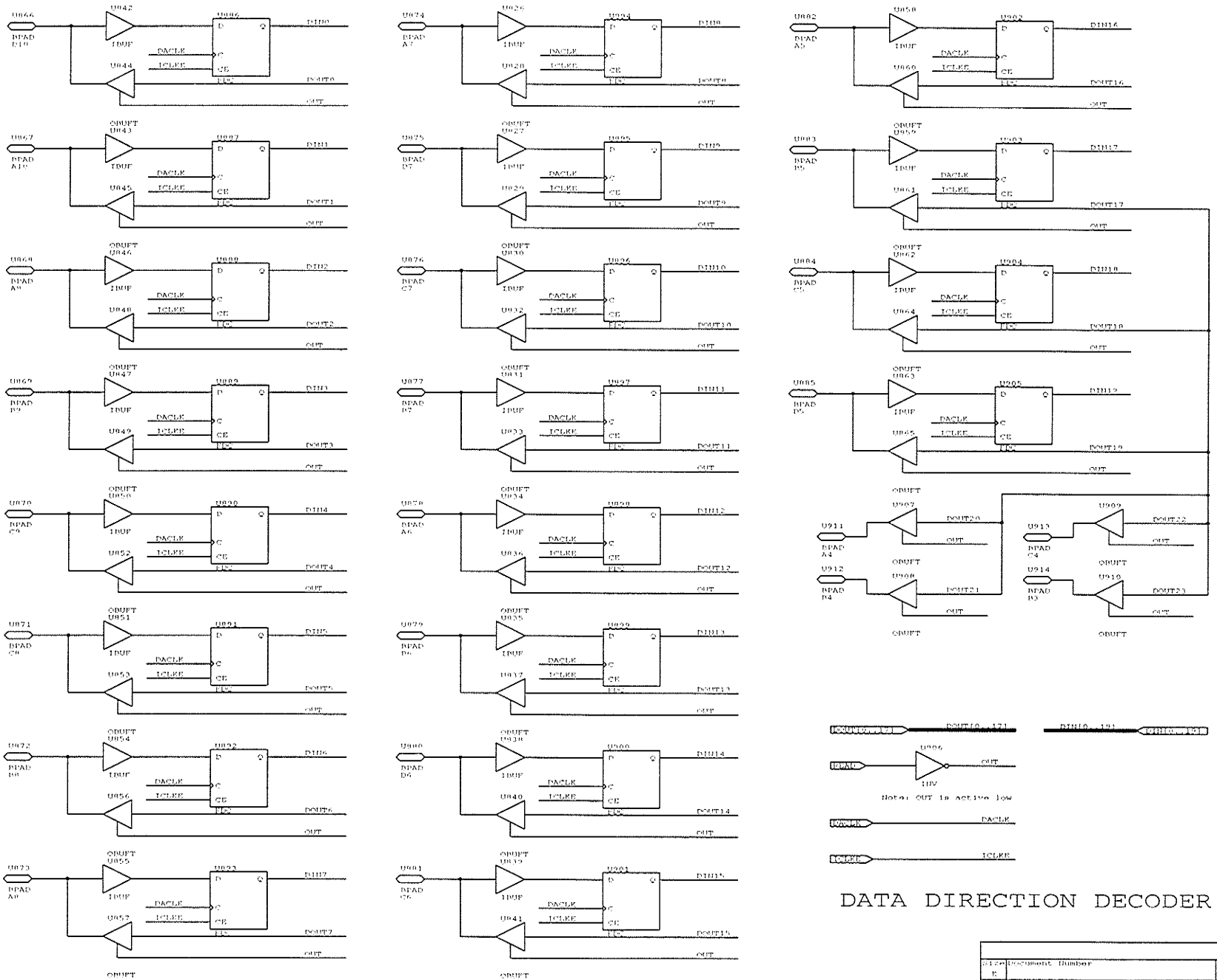
18 BIT 6 CHANNEL MULTIPLEXER





6 TO 1 MULTIPLEXER




Sign Extension
DATA DIRECTION DECODER
Note: OUT is active low
DACLK
ICLKE
Size
Document Number
REV
Date: Wednesday, July 12, 1995
Sheet 21 of 21

Appendix D

Listing For Xilinx Configuration Software And DSP Xhandler & Subroutines

```
//*************************************************************************
/
//
//              Software to Download a *.mcs file to xilinx trough // port
//              Date:       20/06/95
//              Author:     Avid Lemus
//
//*************************************************************************
/


#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <dos.h>



// Function prototypes

void download_xilinx(void);
int convert_to_binary(FILE *file_in);
int trans_ihex_line (unsigned char *line_in);
int translate (unsigned char c);
int bit_shift(int port,unsigned char byte);

// Global variables

unsigned char binary_file[0x1000];
int binary_file_length = 0;
int par_port1 = 0x378;
int par_strobe = 0x37a;


int bit_shift(unsigned char byte)      //bit shift serially to Xilinx
{
        int bit_count,bit;

        for(bit_count=0;bit_count<=7;bit_count++)
        {
                bit = ((byte >> bit_count) & 0x01)|0x02;
                outportb(par_port1,bit);                //latch bit data
                outportb(par_strobe,0x01);              //cclk low
                outportb(par_strobe,0x00);              //cclk hi

        }

        return (0);
}

int translate (unsigned char c)                 //translating ascii to hex fo
rmat
{
        if ((c >= 48) && (c <= 57))
        {
                c -= 48;
        }
        else
        {
```

```
                c -= 55;
        }
        return (c);
}

int trans_ihex_line ( unsigned char *line_in)           //translating ascii t
o hex line
{
        unsigned char i, c, byte_count;
        unsigned char binary_file[0x80];
        int binary_file_length = 0;

        int count;

        if((line_in[0]==':')&&(line_in[7]=='0')&&(line_in[8]=='0'))
        {
                byte_count=((translate(line_in[1]))<<4)|(translate(line_in[2]
));
                for (i = 0;i < 2 * (byte_count);i += 2)
                {
                        binary_file[binary_file_length]=((translate(line_in[i
+9]))<<4)|(translate(line_in[i+10]));
                        binary_file_length ++;
                }
        }
        for (count = 0; count < binary_file_length; count++)
        {
                bit_shift(binary_file[count]);
        }

        return (0);
}

int convert_to_binary(FILE *file_in)
{
        unsigned char line_in[0x80];
        int string_length = 0x80;

        fgets(line_in,string_length,file_in);
        while (!feof(file_in))
        {
                trans_ihex_line(line_in);
                fgets(line_in,string_length,file_in);
        }
        return (0);
}

void download_xilinx(void)
{

        FILE *file_in;
        unsigned char name[80];
        printf("Enter the Xilinx download bit file. \n\r");
        gets(name);


        file_in = fopen( name, "rb" );

        if ( file_in == NULL )
        {
```

```
                printf("\nFile not found.");

        }

        else
        {
                convert_to_binary(file_in);    //ready to convert bit file
        }
        fclose(file_in);
}

main(void)
{       outport(par_port1,0x00);                //reset xilinx
        delay(100);
        outport(par_port1,0xFF);
        outportb(par_strobe,0x00);              //set cclk hi
        download_xilinx();
        outportb(par_strobe,0x00);              //set cclk hi
        return 0;
}
```

```
/* XHANDLER.C, OCT. 95                                                    */
/* revised by AVID LEMUS                                                  */
/* z-volt(In6) and In5 are sent to the PC                                 */
/* Reads Vectored interupts from the HOST (PC) and multiplies the         */
/* Value by 2 to get the address of the service routine to which the      */
/* the program counter goes.  a maximun of 32 such interupts are          */
/* available (ie. 5bits)                                                  */
/* When Doing feedback.  Every time it finishes a row it should send      */
/* it to the PC via an INT 7 RREQ.                                        */
/* Now driven by timer interrupts. This program sends the line number     */
/* before the data.                                                       */

/* DSP constants CONTROL REGISTERS and such                               */

#define PBC x:$ffe0      /* Port B Bus Control register                   */
#define HCR x:$ffe8      /* host control register                         */
#define HSR x:$ffe9      /* host status register                          */
#define HRX x:$ffeb      /* host receive register                         */
#define HTX x:$ffeb      /* host transmit register                        */
#define IPR x:$ffff      /* interrupt priority register                   */
#define SCR x:$fff0      /* Serial Comms. Interface control register      */
#define SCCR x:$fff2     /* SCI clock control register                    */
#define PCC x:$ffe1      /* Port C Control Register                       */
#define PCDDR x:$ffe3    /* Port C Data Direction Register                */
#define PCD x:$ffe5      /* Port C Data Register                          */
#define BCR x:$fffe      /* Bus Control Register                          */

/* Memory mapped I/O constants...(i.e. Analog I/O board )                 */

#define CDA1 y:$a000     /* X D/A, 20 bits, +/- 200 V                     */
#define CDA2 y:$a001     /* Y D/A, 20 bits, +/- 200 V                     */
#define CDA3 y:$a002     /* Z D/A, 20 bits, +/- 200 V                     */
#define CDA4 y:$a003     /* V D/A, 20 bits, +/- 10 V                      */
#define CDA5 y:$a004     /* V D/A, 20 bits, +/- 10 V                      */
#define CDA6 y:$a005     /* V D/A, 20 bits, +/- 10 V                      */
#define FDPF y:$a006     /*Dig. Prog. Filter Cut_off_f=value(51kHz/256))*/

#define CAD1 y:$a000    /* In 1 A/D, 18 bits, Convert 1&2                 */
#define CAD2 y:$a001    /* In 2 A/D, 18 bits                              */
#define CAD3 y:$a002    /* In 3 A/D, 18 bits, Convert 3&4                 */
#define CAD4 y:$a003    /* In 4 A/D, 18 bits                              */
#define CAD5 y:$a004    /* In 5 A/D, 18 bits, Convert 5&6                 */
#define CAD6 y:$a005    /* In 6 A/D, 18 bits                              */

/* Some Variables used in this program                                    */

int darray[512];                          /* new image data               */
int i;                                    /* counter                      */
int datacounter;                          /* index to data array          */
int xoffset, yoffset;                     /* start pt of a rast. line*/
int xsignedoffset, ysignedoffset;         /* signed offset                */
int xstepsize, ystepsize;                 /* spacing betw. image pts */
int xincfactor;                           /* scanning left or right? */
int yincfactor;                           /* scanning up or down?    */
int pconst, iconst;                       /* Intergral & Proportional*/
int setpt, voltage;                       /* feedback stuff               */
int deltaz, error;                        /* feedback stuff               */
int sum, oldsum;                          /* running integral vars.  */
int zin, rin;                             /* digitzd inputs               */
int xout, yout, zout;                     /* new piezo values             */
```

```
int meter;                                  /* value at ADC for meter  */
int stepdir;                                /* direct. in which to step*/
int stop;                                   /* PC stop/start of AFM    */
int feedcounter;                            /* count how many times feedback
 has */
                                            /* been updated at this point */
int xfrac;
int feedvalue;
int xinc, yinc;

main()
{

        /%
        movep #1,PBC            ; enable host port, no interrupts
        movep #0,HCR            ; clear host status

        movep #0,PCC            ; initialize Port C for parallel I/O
        movep #$7,PCDDR         ; set bits PC0, PC1, PC2 as outputs
        movep #0,PCD            ; write 0's to Port C output pins

        movep #$070f,BCR        ; Initialize Bus Control Register setting

                                ; number of wait states to 0,4,0,F on X,Y,P,
                                ; and I/O memory respectively

        movep #$c800,IPR        ; set priorities in IPR
                                ; turn on timer interrupt with priority 2 (hi
ghest)
                                ; turn on host interrupt with priority 1 (mid
dle)

        move  #0,SR             ; unmask interrupts, accept any priority

        movep #4,HCR            ; enable host command vector only

        movep #$0000,SCR        ; disable timer interrupt to start

        movep #$008,SCCR        ; set timer interrupt rate at 10*256 per seco
nd
                                ; (arbitrarily chosen).
                                ; Interrupts / second =
                                ; fosc / (64 * (7(SCP) + 1) * (CD + 1))
                                ; Note that this is the same equation
                                ; as for SCI async baud rate.
                                ; For 1 ms, SCP = 0,
                                ; CD = 0001 0011 1111
                                ; For 8 Hz, SCP = 1,
                                ; CD = 1111 1111 1111
        %/
        /* Initialize Image Data to Zero               */
        for (i=511; i>=0; i--) darray[i] = 0;

        /* Initialize the rest of the variables        */
        datacounter = 0;                    /* array counter           */

        error = 0; setpt = 0;               /* feedback variables      */
        sum = 0; oldsum = 0;                /* start integral at 0     */
        zin = 0; rin = 0;                   /* sampled variables       */
        zout = 0; xout = 0; yout = 0;       /* piezo voltages          */
```

```
        xstepsize = 1; ystepsize = 1;          /* initial size 256 ls bits*/
        xoffset = 0; yoffset = 0;              /* start at center of scan */
        xsignedoffset = -127;
        ysignedoffset = -127;                  /* make x,y simm. about 0  */
        xincfactor = 1; yincfactor = -1;       /* scan directions         */
        xinc = 0; yinc = 0;
        feedcounter = 0;
        feedvalue = 1;
        meter = 0;

        /* Initialize xfrac to scan at 2 Hz                */
        /%
        move #0.03125,y1
        move y1,y:Yxfrac
        %/

        /* initialize the DAC's to the centre of their  */
        /* range to start                               */
        /%
        move #$000000,x0
        move x0,CDA1            ; relax X piezo
        move x0,CDA2            ; relax Y piezo
        move x0,CDA4            ; relax Out 4 DAC
        move x0,CDA5            ; relax Out 5 DAC
        move x0,CDA6            ; relax Out 6 DAC
        move #$03FFFF,x0
        move x0,FDPF            ;Initialize Digitally Prog. Filter fo 51.2Khz
        move x0,CDA3            ;Initialize Z to -100 volts
        %/

        for (;;)                /* infinite loop */
        {
        }

}       /* end program */

/****** Interrupt Driven Routines are Grouped in the Following Files ******/

#include "feedback.c"
#include "interupt.c"
#include "utils.c"
#include "scansize.c"
#include "vector.c"
```

```
feedback()
{
/* subroutine to update feedback */
/* Called by SCI timer interrupt vector at p:$001c */
/* Called by interrupt to p:$001c */
/%
        move    CAD5,x1         ; start conversion on channels 5 & 6
        move    CAD3,x1         ; start conversion on channels 3 & 4

        rep     #64             ; wait 6.4 us
        nop
        move    CAD6,a          ;read the digitized value from PSPD
        move    CAD4,b
        move    a1,Y:Yzin       ;store it
        move    b1,Y:Yrin       ;store it

                ; CALCULATING THE ERROR
        move    Y:Yzin,x0       ; read zin into data acc. x0
        move    Y:Ysetpt,b      ; read setpt into acc. b1
        sub     x0,b            ; setpt-zin go into acc. b1
        move    b1,Y:Yerror     ; error = (setpt - zin)

                ; UPDATE THE RUNNING SUM (INTEGRAL)
        move    b1,x0           ; load error into x0 ready for multiplication
        move    Y:Yiconst,x1    ; load the integral gain
        move    Y:Ysum,a1       ; load sum into acc. a
        move    a1,Y:Yoldsum    ; save old sum in case of saturation
        macr    x0,x1,a         ; sum=sum+iconst*error
        move    a1,Y:Ysum       ; save the updated sum

                ; CALCULATE THE P I FEEDBACK (Z VOLTAGE)
        move    Y:Ypconst,x1    ; pconst -> acc. x1
        macr    x0,x1,a         ; zout= (sum + iconst*error) + pconst*error
        move    a1,Y:Yzout      ; save updated zout
        move    a1,a            ; sign extend a1 to 56 bits move 0 into a0

        ; CHECK FOR SATURATION LIMITS +/- 2^18 (HALF DAC RANGE)
        move    #>$3ffff,x0     ; 3ffff -> acc. x0
        cmp     x0,a            ; is zout >= 3ffff ?
        jlt     L12             ; if not skip next 2 instructions
        move    x0,Y:Yzout      ; else zout=7ffff i.e saturated
        move    Y:Yoldsum,x1    ; if saturated don't integrate any more
        move    x1,Y:Ysum       ; keep the old integral value
        jmp     L11             ; don't check for negative saturation
]L12
        move    #>$fc0000,x0    ; -3ffff -> x0
        cmp     x0,a            ; is zout =< -3ffff ?
        jgt     L11             ; if not skip next instruction
        move    x0,Y:Yzout      ; else zout=-7ffff ie saturated in - directio
n
        move    Y:Yoldsum,x1    ; same as in positive saturation since we
        move    x1,Y:Ysum       ; want to stop integrating if either happens
]L11

                ; PUT OUTPUT IN PROPER FORMAT FOR 0 -> 2^19 RANGE (-Z)

        move    Y:Yzout,a       ; load zout into a1
        move    #>$40000,x0      ; add offset = 2^18 (-100V)
        add     x0,a            ; add the offset
        move    a1,CDA3         ; send it to Z DAC
```

```
        ; CALCULATING X RASTER SIGNAL
move    y:Yxinc,x0      ; load xinc=> counter 0->255*feedvalue
move    Y:Yxfrac,x1     ; load xfrac=1/feedvalue
mpyr    x0,x1,a         ; calculate xinc*xfrac
move    a1,x0           ; we are interested in the integer part
move    y:Yxstepsize,x1 ; load xstepsize
mpy     x0,x1,a         ; calculate xstepsize*xinc*xfrac
asr     a               ; correct sign extension... DSP specific
move    a0,a            ; get ready to add offset  a0->a1
move y:Yxsignedoffset,x1  ; load xsignedoffset
add     x1,a            ; a1=X=xsignedoffset+xstepsize*xinc*xfrac
move    a1,CDA1         ; output the new X value

        ; INCREMENTING X RASTER COUNTER
move    y:Yxinc,x0      ; load xinc into x0
move    y:Yxincfactor,a ; load xincfactor...ie scan direction
add     x0,a            ; and x0 with a1 ...Result in a1
move    a1,y:Yxinc      ; xinc = xinc + xincfactor

        ; INCREMENTING THE # OF FEEDBACKS COUNTER
move    #>$1,x0         ; load 1 into x0
move y:Yfeedcounter,a   ; load the feedcounter into a1
add x0,a y:Yfeedvalue,x1 ; add the above 2 result in a1
move a1,y:Yfeedcounter  ; feedcounter = feedcounter + 1

        ; JUMP OUT UNTIL ENOUGH FEEDBACKS ARE DONE
cmp     x1,a            ; compare feedcounter with feedvalue
jlt     L100            ; jump out if feedvalue - feedcounter < 0

        ; ENOUGH FEEDBACKS...RESET FEEDBACK COUNTER
move    #>$0,a          ; reset a
move a1,y:Yfeedcounter  ; load 0 into feedcounter

        ; SAVE Z VOLTAGE IN DATA ARRAY
move    Y:Yzout,a       ; load data in acc. a
move Y:Ydatacounter,b   ; load datacounter in acc. b
move    #Ydarray,x0     ; load darray-pointer
add     x0,b            ; offset darray-pointer by datacounter
move    b,r5            ; move the datapointer in r5
move    a1,Y:(r5)       ; darray[datacounter] = zout

        ; INCREMENT DATA COUNTER
move Y:Yxincfactor,x0   ; load xincfactor (i.e scan direction)
move Y:Ydatacounter,a   ; load datacounter in a1
add  x0,a               ; add xincfactor to datacounter, result in a1
move a1,Y:Ydatacounter  ; datacounter = datacounter + xincfactor

        ; SAVE R VOLTAGE IN DATA ARRAY (RESISTANCE FOR SRM)
move    Y:Yrin,a        ; load data in acc. a
move Y:Ydatacounter,b   ; load datacounter in acc. b
move    #Ydarray,x0     ; load darray-pointer
add     x0,b            ; offset darray-pointer by datacounter
move    b,r5            ; move the datapointer into r5
move    a1,Y:(r5)       ; darray[datacounter] = rin

        ; INCREMENT DATA COUNTER
move Y:Yxincfactor,x0   ; load xincfactor (i.e scan direction)
move Y:Ydatacounter,a   ; load datacounter in a1
add  x0,a               ; add xincfactor to datacounter, result in a1
```

```
        move a1,Y:Ydatacounter  ; datacounter = datacounter + xincfactor

                ; CHECK TO SEE IF WE HAVE REACHED THE BEGINNING OF A ROW
                ; IF YES CHANGE X SCAN DIRECTION, RESTART DATA COUNTER
                ; AND EXIT FEED BACK ROUTINE
        move    #>$0,x1         ; load 0 into acc. x1
        move Y:Ydatacounter,a   ; load datacounter into a1
        cmp     x1,a            ; calculate datacounter - 0
        jge     L22             ; if datacounter >= 0 then exit this check

        move Y:Yxincfactor,a    ; load xincfactor
        neg     a               ; change scandirection
        move a1,Y:Yxincfactor   ; xincfactor=-xincfactor

        move    #>$0,a          ; load 0 into a
        move a1,Y:Ydatacounter  ; datacounter = 0  (RESET datacounter)

        jmp  L100               ; EXIT FEEDBACK ROUTINE

]L22
                ; CHECK SEE IF WE HAVE REACHED THE END OF A ROW
                ; IF NO EXIT FEEDBACK ROUTINE, IF YES CHANGE X SCAN DIRECTION
                ; DISABLE INTERRUPT TIMER

        move    #>$1ff,x0       ; load 511 into x0 ( 2 images of 256 pixels)
        move Y:Ydatacounter,a   ; load datacounter in a1
        cmp     x0,a            ; calculate datacounter - 511
        jle     L100            ; if datacounter-511 <= 0 then EXIT FEEDBACK

        movep   #$0000,x:$fff0  ; disable timer interrupt TMIE

        move Y:Yxincfactor,a    ; load xincfactor
        neg     a               ; change scandirection
        move a1,Y:Yxincfactor   ; xincfactor=-xincfactor

        move    #>$1ff,a        ; load 511 into a
        move a1,Y:Ydatacounter  ; datacounter = 511  (SET datacounter)


                ; CHECK IF WE HAVE REACHED THE TOP OR BOTTOM OF THE IMAGE
                ; AND CHANGE Y SCAN DIRECTION
        move Y:Yyinc,a          ; load linecounter into a1
        move    #>$ff,x0        ; load 255 into acc. x0
        cmp     x0,a            ; calculate linecounter-255
        jeq     L26             ; if linecounter=255 CHANGE Y SCAN DIRECTION
        cmp     x1,a            ; value in x1=0 calculate a1-0
        jne     L27             ; if linecounter is not = 0 EXIT CHECK
]L26
        move Y:Yyincfactor,a    ; load yincfactor (Y SCAN DIRECTION)
        neg     a               ; change y scan direction
        move a1,Y:Yyincfactor   ; yincfactor = -yincfactor
]L27

                ; CALCULATING Y RASTER SIGNAL
        move    y:Yyinc,x0      ; load yinc=>row counter
        move Y:Yystepsize,x1    ; load ystepsize=>defines y scan size
        mpy     x0,x1,a         ; calculate yinc*ystepsize
        asr     a               ; correct for signed arguments (DSP SPECIFIC)
        move    a0,a            ; get ready to add offset
        move y:Yysignedoffset,x1 ; load offset+signcorrection
```

```
        add     x1,a            ; calculate ysignedoffset+yinc*ystepsize
        move    a1,CDA2         ; output the new Y value

                ; INCREMENTING Y RASTER COUNTER (ROW/LINE COUNTER)
        ;move    y:Yyinc,x0      ; load yinc into x0
        move    y:Yyincfactor,a ; load yincfactor...ie y scan direction
        add     x0,a            ; add x0 (= yinc) to a1 ...Result in a1
        move    a1,y:Yyinc      ; yinc = yinc + yincfactor

                ; START DATA TRANSFER TO PC
                ; SET DATA POINTER TO BEGINNING, ENABLE HOST INTERRUPTS OF
                ; DSP AND VICE-VERSA...EVERYTIME HTX LOW BYTE IS READ DSP
                ; IS INTERRUPTED AND INT. SERVICE ROUTINE UPDATES THE DATA
        move    #Ydarray,r3     ; point data to start
        movep   #6,HCR          ; enable HCIE and HTIE
        movep   Y:Yyinc,x:$ffeb   ; start transfer,interrupt PC

]L100
        rti                     ; return from subroutine
%/
}
/* ------------------------------------------------------------------- */


setpoint()
{
/* read in new setpoint from PC                                        */
/* called by CV to p:$003c                                             */
/%
        movep   HRX,y1          ; read in new set point from PC
        move    y1,+setpt       ; store it in variable memory
        rti                     ; return from subroutine
%/
}
/* ------------------------------------------------------------------- */


changebias()
{
/* changes bias voltage on channel 4                                   */
/* called by CV to p:$0034                                             */
/%
        movep   HRX,y0          ; read in bias voltage
        move    y0,Y:Yvoltage   ; set bias voltage
        move    y0,CDA4         ; set bias voltage
        rti                     ; return from subroutine
%/
}
/* ------------------------------------------------------------------- */

changeifeedback()
{
/* Reads the I-feedback value from PC divides it by 10000              */
/* & updates it                                                        */
/* called by CV to p:$0038                                             */
/%
        movep   HRX,x1          ; read in I-feedback constant
        move    #0.0001,y1      ; load 1/10000 into y1
        mpy     x1,y1,a         ; x1*y1 go into a1.a0 (integer.fraction)
        asr     a               ; correct for signed arguments (DSP SPECIFIC)
```

```
        move    a0,+iconst      ; set iconst=PCvalue/10000 (store frac. part)
        rti                     ; return from subroutine
%/
}
/* ------------------------------------------------------------------- */

changepfeedback()
{
/* Reads the P-feedback value from PC divides it by 10000       */
/* & updates it                                                 */
/* called by CV to p:$0036                                      */
/%
        movep   HRX,x1          ; read in P-feedback constant
        move    #0.0001,y1      ; load 1/10000 into y1
        mpy     x1,y1,a         ; x1*y1 go into a1.a0 (integer.fraction)
        asr     a               ; correct for signed arguments (DSP SPECIFIC)
        move    a0,+pconst      ; set pconst=PCvalue/10000 (store frac. part)
        rti                     ; return from subroutine
%/
}
/* ------------------------------------------------------------------- */
```

```
enabletimerint()
{
/* enables timer interrupts. ie scanning                          */
/* called by CV to p:$000c                                        */
/%
        movep #$2000,SCR        ; enable timer interrupt
        rti                     ; return from subroutine
%/
}
/* ---------------------------------------------------------------- */

disableint()
{
/* change host control register on DSP (different interrupts)     */
/* called by a CV to p:$002c                                      */
/%
        movep #4,HCR            ; enable host command vector only
        rti                     ; return from subroutine
%/
}
/* ---------------------------------------------------------------- */

enableint()
{
/* change HCR on DSP enabling HCV & HTIE interrups                */
/* called by CV to p:$002a */
/%
        movep #6,HCR            ; enable host command vector & HTIE inter.
        rti                     ; return from subroutine
%/
}
/* ---------------------------------------------------------------- */

startstop()
{
/* start/stop the AFM  - if stop is TRUE, stop it else start it */
/* called by CV to p:$002E                                        */
/%
        movep HRX,Y:Ystop       ; read flag to start/stop AFM
        move Y:Ystop,a

        tst  a                  ; if stop..stop scaning
        jeq  L54                ; else start..
        movep #$0000,SCR        ; disable timer interrupt TMIE
        jmp  L53                ; end stop

]L54
        movep #$2000,SCR        ; enable timer interrupt TMIE, ie start
]L53
        rti                     ; return from subroutine
%/
}
/* ---------------------------------------------------------------- */
```

```
stepper()
{
/* Single Step the stepper motor in either direction                */
/* Called by CV to p:$0018                                          */

/%
        movep HRX,a             ; read in step direction
        move #>25000,x0         ; load number of clock cycles/2 to wait
                                ; between state changes of step bit ie 2.5ms

        tst  a                  ; is it going down?
        jeq  L344               ; else step up..

        move #>$3,a             ; set PC2,PC1,PC0 to 011, 0=/en,1=step,1=dir
        movep a1,PCD
        nop                     ; send step command to stepper controller
        nop                     ; and wait two cycles
        rep x0                  ; wait x0 times 2 clock cycles
        nop

        move #>$1,a             ; set PC2,PC1,PC0 to 001, 0=/en,0=step,1=dir
        movep a1,PCD
        nop                     ; send step command to stepper controller
        nop                     ; toggling step bit only
        rep x0                  ; wait x0 times 2 clock cycles
        nop

        jmp  L343               ; stepping down done exit

]L344
        move #>$2,a             ; set PC2,PC1,PC0 to 010, 0=/en,1=step,0=dir
        movep a1,PCD
        nop                     ; send step command to stepper controller
        nop                     ; and wait two cycles
        rep x0                  ; wait x0 times 2 clock cycles
        nop

        move #>$0,a             ; set PC2,PC1,PC0 to 000, 0=/en,0=step,0=dir
        movep a1,PCD
        nop                     ; send step command to stepper controller
        nop                     ; toggling step bit only
        rep x0                  ; wait x0 times 2 clock cycles
        nop

]L343
        move #>$4,x0            ; load motor-off bit-mask
        or   x0,a               ; or mask with step out to preserve direction
        movep a1,PCD
        nop                     ; sent stepout variable to stepper motor
        nop                     ; turning it off

        rti                     ; return from subroutine
%/
}
/* ---------------------------------------------------------------- */

sendmeter()
{
/* Send photodiode meter value to PC                                */
/* value is 18 bits sign extended to 24...2s complement             */
```

```
/* Called by Command Vector to p:$001a...PC call # $1D          */

/%
        move CAD5,x0            ; start conversion Chnls 5&6

        rep #64                 ; wait 64x2 clock cycles
        nop

        move CAD6,x0            ; read the value in from the ADC
        move x0,Y:Ymeter        ; store it in meter variable

        movep #4,HCR            ; enable host command vector only, disable HT
IE
        movep Y:Ymeter,HTX      ; write value to host data register
        movep #6,HCR            ; enable host command vector & HTIE interrupt
s

        rti                     ; return from subroutine
%/
}

/* ------------------------------------------------------------------- */
changecutoff()
{
/* Send 8 bit number to digitally programmable filter           */
/* the cutoff=(value of binary number/256)*51.2KHz              */
/* Called by Command Vector to p:$0024...PC call # $92          */

/%
        movep HRX,a             ; read cutoff number from PC
        move a1,FDPF            ; send it to xilinx

        rti                     ; return from subroutine
%/
}

/* ------------------------------------------------------------------- */
```

```
changeyrange()
{
/* updates yrange variable & calculates yfrac=yrange/yincrange  */
/* effectively changing Y step size. Called by CV to p:$0032    */
/%
        movep HRX,Y:Yystepsize    ; Read in New ystepsize

        move Y:Yystepsize,x0      ; load ystepsize into a
        move #>128,y0             ; load number of steps/2 in 1 image
        mpy -x0,y0,a              ; calculate -range/2
        asr a                     ; required for signed int * signed int,
                                  ; see "Fractional and Integer Arithmetic
                                  ; Using the DSP56000 Family of General-
                                  ; Purpose Digital Signal Processors"
        move a0,a                 ; result in a0 move it up to a1
        move Y:Yyoffset,x0        ; load yoffset into x0
        add x0,a                  ; calculate corrected offset for
        move a1,Y:Yysignedoffset  ; signed output (a1 -> a1+x0)

        move #>$07ffff,x0         ; this translates into sending minimum
        move x0,CDA3              ; voltage to piezo in order to lift the
                                  ; tip. i.e if the sample is mounted on
                                  ; the piezo, we shrink the piezo as much
                                  ; as possible.

        rti                      ; return from subroutine
%/
}
/* ------------------------------------------------------------------- */

changeyoff()
{
/* changes the Y offset to a value dictated by PC                      */
/* called by CV to p:$0028                                             */
/%
        movep HRX,Y:Yyoffset      ; read in Y offset value

        move Y:Yystepsize,x0      ; load ystepsize into a
        move #>128,y0             ; load number of steps/2 in 1 image
        mpy -x0,y0,a              ; calculate -range/2
        asr a                     ; get read of "sign bit..DSP SPECIFIC"

                                  ; depends on type of operands
        move a0,a                 ; result in a0, move it to a1
        move Y:Yyoffset,x0        ; load yoffset into x0
        add x0,a                  ; calculate corrected offset for
        move a1,Y:Yysignedoffset  ; signed output ( a1->a1+x0 )

        move #>$07ffff,x0         ; this translates into sending minimum
        move x0,CDA3              ; voltage to piezo in order to lift the
                                  ; tip. i.e if the sample is mounted on
                                  ; the piezo, we shrink the piezo as much
                                  ; as possible.

        rti                       ; return from subroutine
%/
}
/* ------------------------------------------------------------------- */

changexrange()
```

```
{
/* updates xrange variable & calculates xfrac=xrange/xincrange  */
/* effectively changing X step size. Called by CV to p:$0030    */
/%
        movep HRX,Y:Yxstepsize    ; Read in New ystepsize

        move Y:Yxstepsize,x0      ; load ystepsize into a
        move #>128,y0             ; load number of steps/2 in 1 image
        mpy -x0,y0,a              ; calculate -range/2
        asr a                     ; required for signed int * signed int
                                  ; see "Fractional and Integer Arithmetic
                                  ; Using the DSP56000 Family of General-
                                  ; Purpose Digital Signal Processors"
        move a0,a                 ; result in a0 move it up to a1
        move Y:Yxoffset,x0        ; load yoffset into x0
        add x0,a                  ; calculate corrected offset for
        move a1,Y:Yxsignedoffset  ; signed output (a1 -> a1+x0)

        move #>$07ffff,x0         ; this translates into sending minimum
        move x0,CDA3              ; voltage to piezo in order to lift the
                                  ; tip. i.e if the sample is mounted on
                                  ; the piezo, we shrink the piezo as much
                                  ; as possible.

        rti                       ; return from subroutine
%/
}
/* ------------------------------------------------------------------- */

changexoff()
{
/* Changes the X offset to a value dictated by PC                      */
/* called by CV to p:$0026                                             */
/%
        movep HRX,Y:Yxoffset      ; read in X offset value

        move Y:Yxstepsize,x0      ; load xstepsize into a
        move #>128,y0             ; load number of steps/2 in 1 image
        mpy -x0,y0,a              ; calculate -range/2
        move a0,a

        move Y:Yxoffset,x0        ; load xoffset into x0
        add x0,a                  ; calculate corrected offset for
        move a1,Y:Yxsignedoffset  ; signed output

        move #>$07ffff,x0         ; this translates into sending minimum
        move x0,CDA3              ; voltage to piezo in order to lift the
                                  ; tip. i.e if the sample is mounted on
                                  ; the piezo, we shrink the piezo as much
                                  ; as possible.

        rti                       ; return from subroutine
%/
}
/* ------------------------------------------------------------------- */

changefeedvalue()
{
/* calculates xfrac=1/feedvalue and updates feedvalue, which in turn   */
/* which defines the scan rate since # feedbacks/row=feedvalue*256     */
```

```
/* TIME FOR ONE FEEDBACK = 30.5 uS, defined by the interrupt rate timer */
/* called by CV to p:$003a */
/%
        movep HRX,Y:Yfeedvalue    ; read in new feedvalue
        move #>0,y1               ; load 0 into Y1
        move y1,y:Yxinc           ; reset xinc i.e start scanning row
        move y1,y:Ydatacounter    ; reset datacounter i.e. no data
        move y1,y:Yfeedcounter    ; reset feedcounter => we're starting
        move #>1,y1               ; set Y1 to 1
        move y1,y:Yxincfactor     ; scan direction is "+" increasing X

        move Y:Yfeedvalue,x0
        move #>1,a
        abs a
        nop
        rep  #$18                 ; repeat 24 times for 24 bit precision
        div  x0,a                 ; calculate 1/feedvalue
        move a0,Y:Yxfrac          ; save result in xfrac

        move #>$07ffff,x0         ; this translates into sending minimum
        move x0,CDA3              ; voltage to piezo in order to lift the
                                  ; tip. i.e if the sample is mounted on
                                  ; the piezo, we shrink the piezo as much
                                  ; as possible.

        rti                       ; return from subroutine
%/
}

/* ------------------------------------------------- */
```

```
/* host command vector interrupt                                    */
/* Enable timer interrupts                                          */
/%
        section enabletimerinterrupt
        org   p:$000c           ; where to put this interrupt vector
        jsr   Fenabletimerint+4 ; jump to routine to enable timer interrupts
        nop                     ; it's a 2-instruction interrupt
        endsec
%/
/* ---------------------------------------------------------------- */

/* host command vector interrupt                                    */
/* Call routine to step the motor in one direction or the other */
/%
        section step
        org   p:$0018           ; where to put this interrupt vector
        jsr   Fstepper+4        ; jump to routine to step the motor
        nop                     ; it's a 2-instruction interrupt
        endsec
%/
/* ---------------------------------------------------------------- */

/* host command vector interrupt                                    */
/* Call routine to send a photodiode meter value to the PC      */
/%
        section meter
        org   p:$001A           ; where to put this interrupt vector
        jsr   Fsendmeter+4      ; jump to routine to output a meter value
        nop                     ; it's a 2-instruction interrupt
        endsec
%/
/* ---------------------------------------------------------------- */

/* SCI timer interrupt vector                                       */
/* Call routine to update Z feedback, X,Y rastering & init. data trans. */
/%
        section update
        org   p:$001c           ; where to put this interrupt vector
        jsr   Ffeedback+4       ; jump to routine to update feedback
        nop                     ; it's a 2-instruction interrupt
        endsec
%/
/* ---------------------------------------------------------------- */

/* default host command vector interrupt                            */
/* send data from DSP to PC                                         */
/%
        section senddatum
        org   p:$0022           ; where to put this interrupt vector
        movep     y:(r3)+,HTX   ; move data at pointer to output
        nop                     ; it's a 2-instruction interrupt
        endsec
%/
/* ---------------------------------------------------------------- */

/* host command vector interrupt                                    */
/* Pointer to routine to change Dig. Prog. Filter cutoff freq.  */
/%
        section filtercutoff
        org   p:$0024           ; where to put this interrupt vector
```

```
        jsr Fchangecutoff+4     ; jump into sub. to change filter cutoff
        nop                     ; it's a 2-instruction interrupt
        endsec
%/
/* ------------------------------------------------------------------- */

/* host command vector interrupt                                       */
/* change X scan offset                                                */
/%
        section xoffset
        org   p:$0026           ; where to put this interrupt vector
        jsr Fchangexoff+4       ; jump to subroutine to read X offset value
        nop                     ; it's a 2-instruction interrupt
        endsec
%/
/* ------------------------------------------------------------------- */

/* host command vector interrupt                                        */
/* change Y scan offset                                                 */
/%
        section Yoffset
        org   p:$0028           ; where to put this interrupt vector
        jsr Fchangeyoff+4       ; jump to subroutine to read X offset value
        nop                     ; it's a 2-instruction interrupt
        endsec
%/
/* ------------------------------------------------------------------- */

/* host command vector interrupt                                       */
/* change host control register on DSP (enable HTIE interrupts) */
/%
        section enable
        org   p:$002A           ; where to put this interrupt vector
        jsr Fenableint+4        ; jump to subrout. to enable HTIE interrupts
        nop                     ; it's a 2-instruction interrupt
        endsec
%/
/* ------------------------------------------------------------------- */

/* host command vector interrupt                                       */
/* change host control register on DSP (disable HTIE interrupts)*/
/%
        section disable
        org   p:$002C           ; where to put this interrupt vector
        jsr Fdisableint+4       ; jump to subrout. to disable HTIE interrupts
        nop                     ; it's a 2-instruction interrupt
        endsec
%/
/* ------------------------------------------------------------------- */

/* host command vector interrupt                                       */
/* toggle Z feedback on/off                                            */
/%
        section togglefeedback
        org   p:$002E           ; where to put this interrupt vector
        jsr  Fstartstop+4       ; jump to subroutine to stop/start AFM
        nop                     ; it's a 2-instruction interrupt
        endsec
%/
/* ------------------------------------------------------------------- */
```

```
/* host command vector interrupt                                        */
/* change X Range                                                       */
/%
        section xrange
        org   p:$0030           ; where to put this interrupt vector
        jsr Fchangexrange+4     ; jump to subrout. to read X range from PC
        nop                     ; it's a 2-instruction interrupt
        endsec
%/
/* -------------------------------------------------------------------- */

/* host command vector interrupt                                        */
/* change Y Range                                                       */
/%
        section yrange
        org   p:$0032           ; where to put this interrupt vector
        jsr Fchangeyrange+4     ; jump to subrout. to read Y Range from PC
        nop                     ; it's a 2-instruction interrupt
        endsec
%/
/* -------------------------------------------------------------------- */

/* host command vector interrupt                                        */
/* change bias voltage  - unused now, saved for STM                     */
/%
        section biasvoltage
        org   p:$0034           ; where to put this interrupt vector
        jsr Fchangebias+4       ; jump to subrout. to read bias Volt. from PC
        nop                     ; it's a 2-instruction interrupt
        endsec
%/
/* -------------------------------------------------------------------- */

/* host command vector interrupt                                        */
/* change proportional feedback constant                                */
/%
        section pfeed
        org   p:$0036           ; where to put this interrupt vector
        jsr Fchangepfeedback+4  ;jump to sub. to read P-feed cons. from PC
        nop                     ; it's a 2-instruction interrupt
        endsec
%/
/* -------------------------------------------------------------------- */

/* host command vector interrupt                                        */
/* change integral feedback constant                                    */
/%
        section ifeed
        org   p:$0038           ; where to put this interrupt vector
        jsr Fchangeifeedback+4  ;jump to sub. to read I-feed cons. from PC
        nop                     ; it's a 2-instruction interrupt
        endsec
%/
/* -------------------------------------------------------------------- */

/* host command vector interrupt                                        */
/* change feedvalue on DSP                                              */
/%
        section feedvalue
```