

Chapter 6 Testing Using a Cantilever Beam

6.1 Introduction

In order to test the system the directional coupler was connected to a composite fibre embedded in a cantilever beam as shown in figure 22. The directional coupler allows two fibres to be coupled to the ELED transmitter, however, for this test only one was connected. The optical fibre contained a single Bragg grating fabricated at a specified point. The grating was embedded in a composite cantilever beam. This section discusses the results obtained from the test together with an explanation of why the system was unable to track the grating and the solution to this problem.

6.2 Test and results

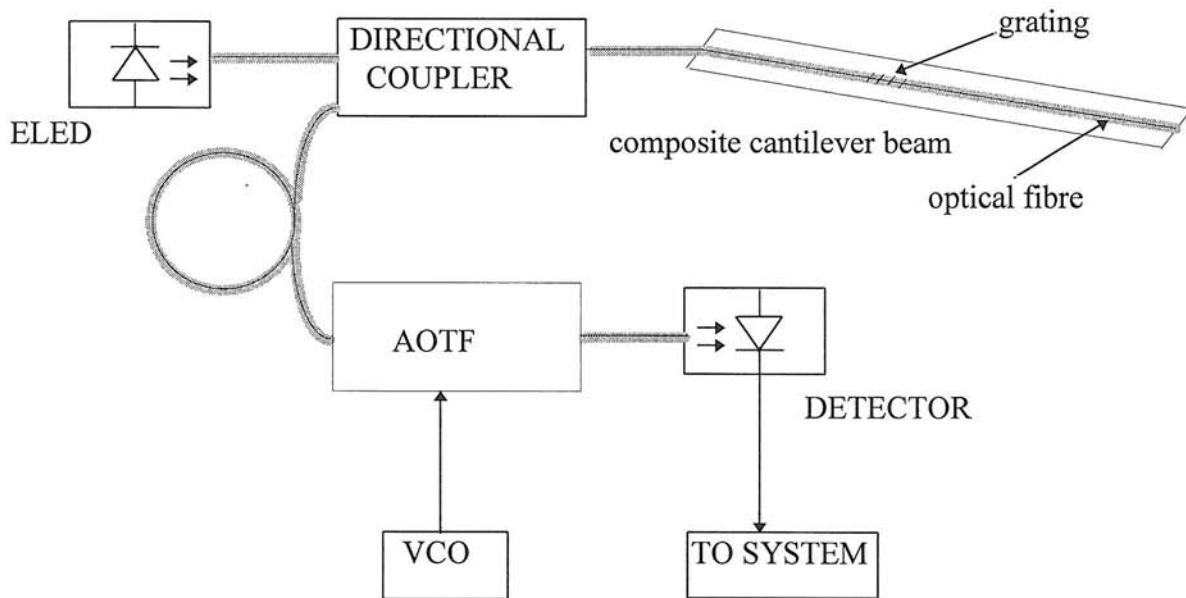
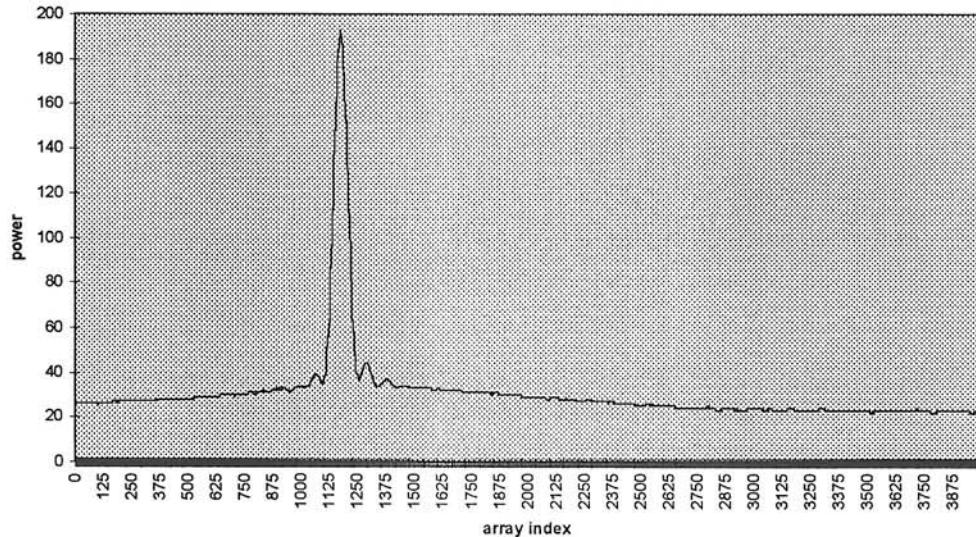


Figure 22: Experimental setup of grating interrogation system

The diagram in figure 22 shows the experimental set-up to test the system. The first stage of operation is to scan the grating's spectra within the optical fibre and determine two points on the spectra of equal optical power. The second stage is to initiate the system to lock onto the grating and track as it moves. Figure 23 shows a plot of the grating's spectra together with the two points selected by the software corresponding to two points on the grating's spectra whose optical power is equal. These are not exactly the same, but would not detract from the system's overall performance. This is because the feedback loop would correct the system for this slight as it would match one of the scenarios discussed in section 4.2 i.e. it would correspond to a condition where either the mean RF drive frequency applied by the VCO

to the AOTF is too high or too low.



point 1 = 1142 (array index)
 point 2 = 1214 (array index)

Figure 23: Plot showing the data scanned into the PC while the system is operating in scan mode, together with the two points calculated by the system of equal optical power.

Unfortunately, the second stage of operation i.e. locking onto the grating , was unable to be completed. This was due to the design of the detector circuit , whose amplifier stage did not have the required gain bandwidth product. Thus, in order to amplify the returned signal from the photodetector circuit sufficiently to allow the system to detect the grating, when the following lockin operation was tried the bandwidth of the detector had deteriorated so that instead of a square wave returning from the detector circuit as per section 4.2 a sine wave or triangular wave was observed. The solution of this problem is to redesign the photodetector circuit with an amplifier which has sufficient bandwidth to accommodate a 10KHz signal.

6.3 Conclusion

The system is in a position where it can scan a grating's peak, detect two points of equal power and initiate the feedback loop to start locking onto the grating. However, without a functioning receiver circuit the system cannot actually lock onto a grating. Appendix D contains an alternative design for the receiver circuit which has the required bandwidth for this system.

Conclusion and Further work required.

Further work required

The system requires the detector circuit to be redesigned with a larger gain bandwidth product and this was discussed in chapter six. A new receiver design has been suggested that should accomplish this. This will allow the module produced to track a grating. Following this, the software developed requires to be interfaced with the graphical user interface program developed for the previous system. However, this should only be attempted after the counter circuit has been made operational.

The future development of this system depends on the number of modules which can be linked to the computer. At present with the lines available, this is a maximum of four. An address decoder is required which will select each module, though it would be beneficial to make any future modules as a PCB (printed circuit board). In anticipation of this all the schematics produced have been drawn on a commercial drawing package⁹. The way that the lines on the digital I/O interface are used is the main limiting factor concerning the number of modules that could be linked to the PC. By allocating each board a port and a set of control lines, the complexity of interfacing was reduced at the expense of the number of modules that could be linked. Two approaches could be taken to improve the situation. The first is to reconfigure the ADC ports after a **scan mode** has been completed. These free lines could then be used to select the different modules. The second approach is to increase the complexity of the system and create a bus and bus controller circuit to control all interactions between the PC and the modules.

Another improvement that might be considered is the incorporation of a 16 bit ADC. Though these are expensive, it would increase the resolution of the system and is only required once.

Conclusion

This project has sought to extend the work already done on Bragg grating interrogation systems. The system uses an AOTF as an optical bandpass filter in the process of seeking and locking onto the gratings. The PC via the data acquisition part of the system locates the gratings and then initiates a feedback loop to lock onto and track a grating's spectra as it shifts due to a physical variable. This allows the physical variable to be measured.

The system itself although not fully functional has completed in essence the original aims of the project. This new system has increased the operating frequency at which Bragg gratings can be interrogated within optical fibres. It is also in the modular form which means that it can be easily replicated so that simultaneous tracking of many Bragg gratings is possible. The scan mode of the system is demonstrable and so the Bragg gratings within a fibre can be detected by the module as shown in chapter 6. The PC can start the lockin operation and output a square which via the VCO will toggle the AOTF to allow light of two wavelengths through corresponding to two points on the Bragg grating's spectra. Although the feedback

⁹ OrCAD version 4.10

loop has not been tested within the system it has been functionally tested and should work. However, as was stated earlier the system requires a new detector circuit.

The advantages of this type of technology over conventional sensor techniques have been discussed in chapter one. It has been predicted that intelligent materials will be incorporated into structures such as bridges, dams, buildings and aircraft within three to four decades and fibre optic sensor technology will be the leading candidate for implementation of these schemes¹⁰. Out of all the fibre optic sensors used for discrete measurements the Bragg grating sensor is one of the most versatile because of its good sensitivity to strain and simple multiplexing capability. The project harnesses these properties of Bragg gratings and sought to develop an interrogation system further which will allow the gratings to be simultaneously tracked.

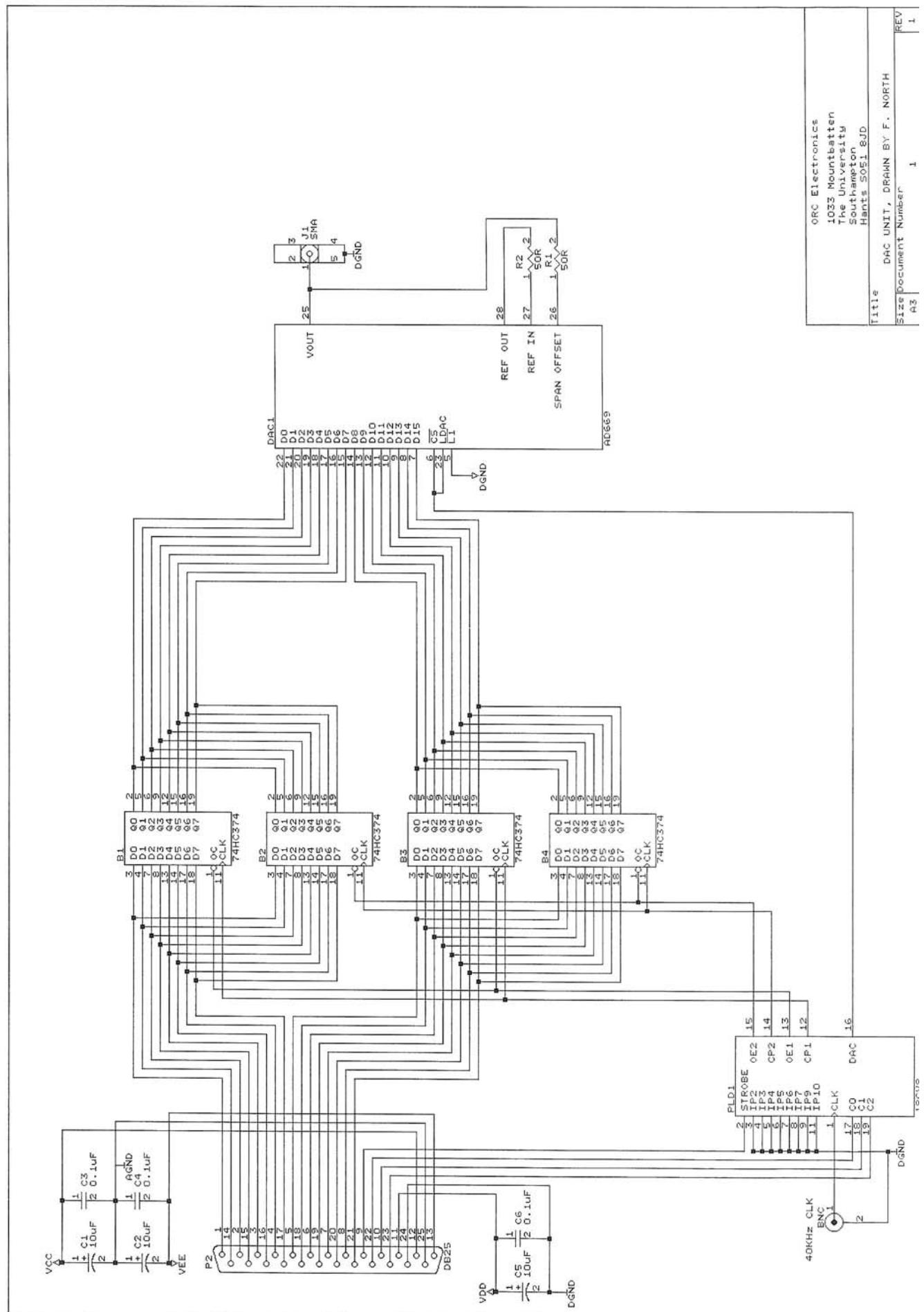
¹⁰ Fibre-optic sensors and smart structures: Developments and prospects by D. Uttamchandani

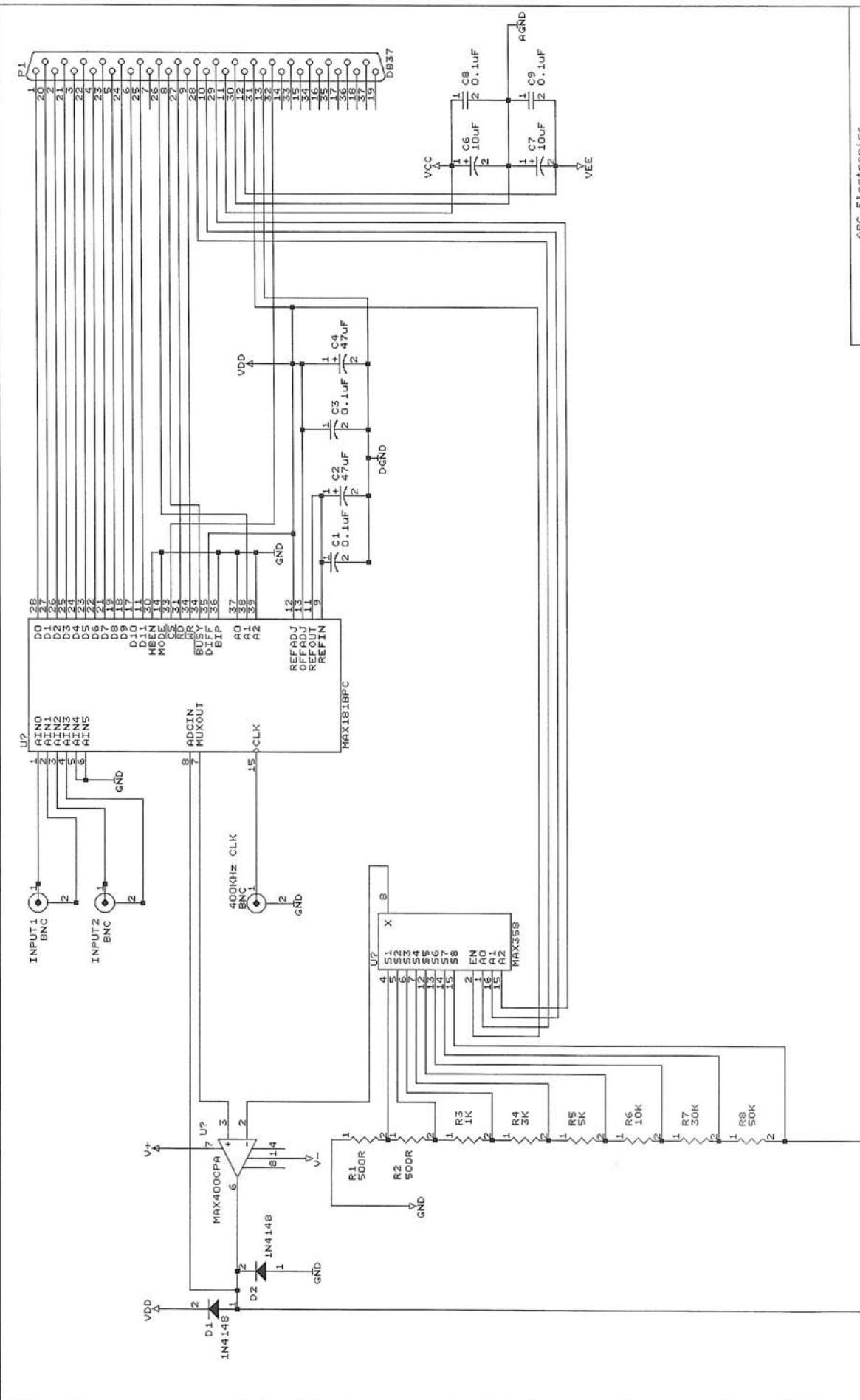
References

1. Clayton G.B., 1982, "Data Converters", Macmillan, London.
2. Segio Franco, 1989, "Design with Operational Amplifiers and Analog Integrated Circuits", McGraw Hill, New York.
3. Geiger H, 1995, Chapter 6, PhD Thesis "Quasi-Distributed Optical Fibre Strain Sensors", Optical Fibre Group, Department of Electronics & Computer Science, University of Southampton.
4. Geiger H, Xu M.G., Eaton N.C, Dakin J.P., 1995, "Electronic tracking system for multiplexed fibre grating sensors", Electronics Letters, Vol 31 No. 17.
5. Goutzoulis A.P., Kludzin V.V., 1994, "Principles of Acousto-Optics" from "Design and Fabrication Of Acousto-Optic Devices", Dekker, New York
6. Grabel A, Millman J, 1987, "Microelectronics" 2nd Edition, McGraw Hill, New York.
7. Horowitz P, Hill W, "The Art of Electronics" 2nd Edition, Cambridge University Press.
8. Xu M.G., Geiger H. , J.P.Dakin, 1995, "Interrogation Of Fibre Optic sensors using acousto-optic tunable filter", Electronics Letters, Vol 31 No. 17.
9. Uttamchandani D., Oct. 1994 "Fibre -optic sensors and smart structures: Developments and prospects", Electronics and Communication Engineering journal.
10. National Instruments, PC-DIO-96 User Manual, September 1990 Edition, Part Number 320289-01

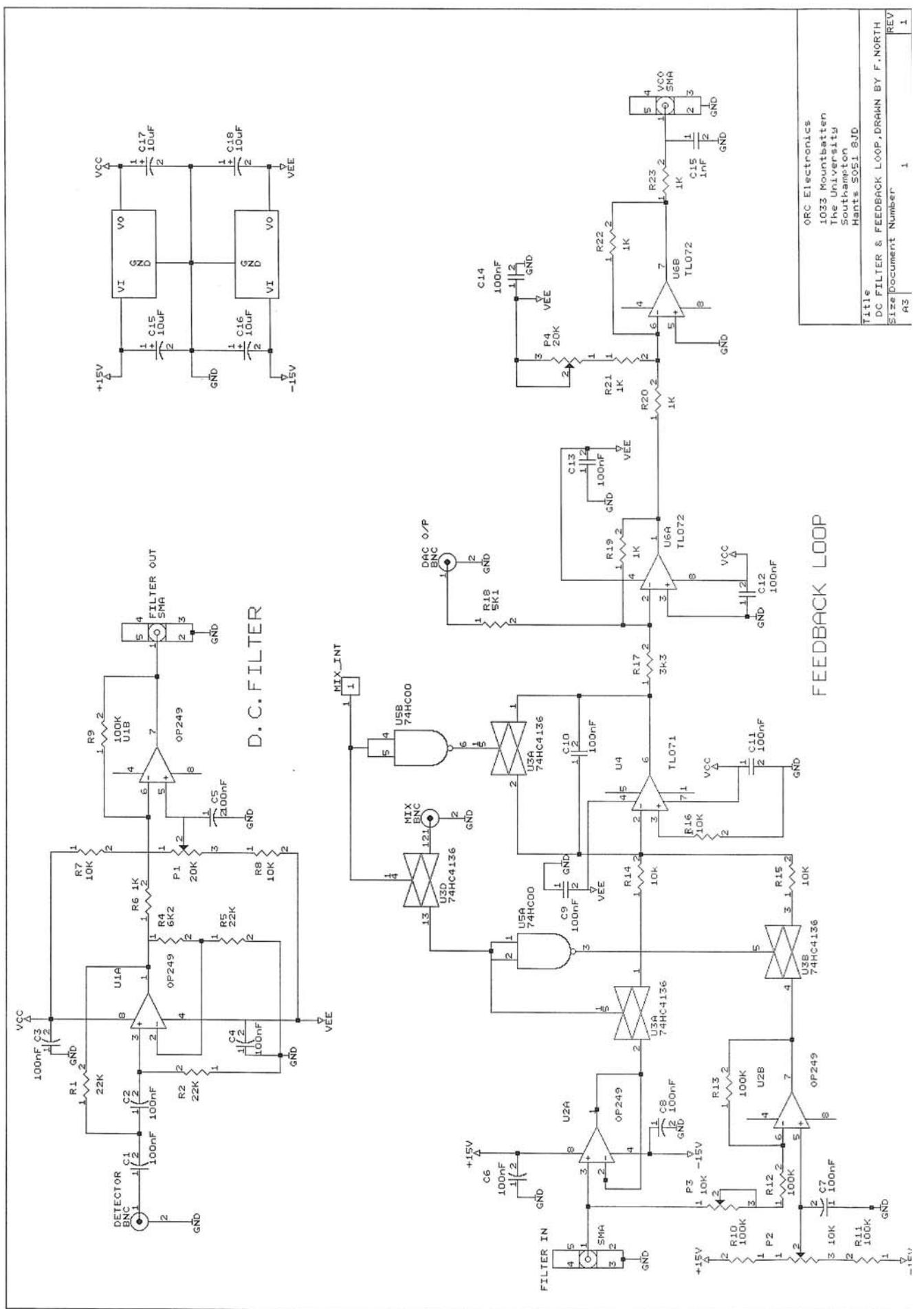
Appendix A: Schematic Diagrams

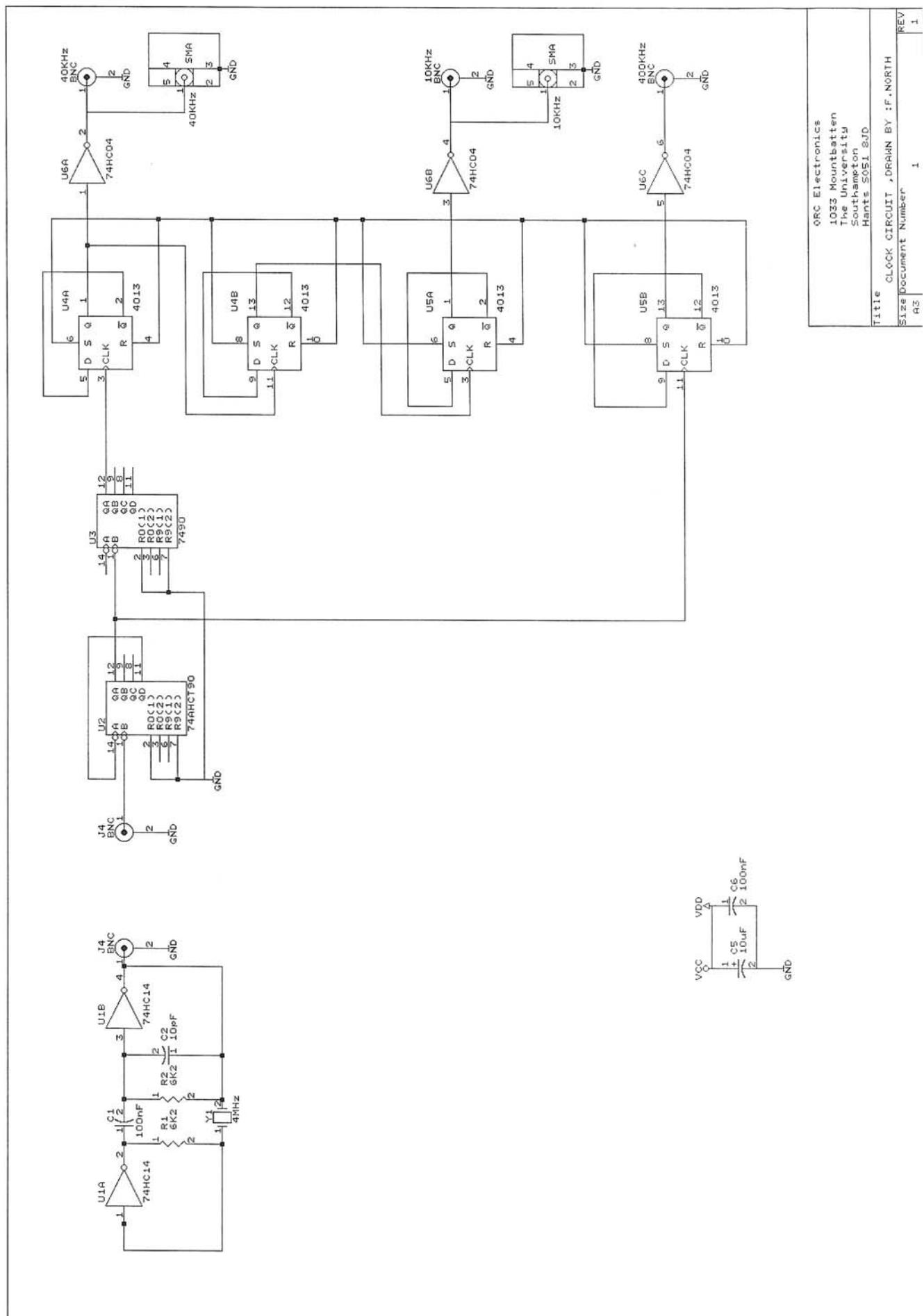
DAC Board Schematic	pg 1 of 4
ADC Board Schematic	pg 2 of 4
Feedback loop Schematic	pg 3 of 4
Clock Circuit and Distribution circuit schematic	pg 4 of 4





ORC Electronics	1033 Mountbatten
	The University
	Southampton
	Hants SO51 8JD
Title	ADC SCHEMATIC, DRAWN BY F.NORTH
Size Document Number	1
A3	REV 1





Appendix B: PLD Program listing

```

device DAC_controller (p18cv8) "Control logic for DAC and buffers"

pin
clk=1          (clock)
strobe=2        (input combinatorial) "strobe signal for data transfer"

c[2:0]=19:17   (output reg active_high feed_reg)
CP1=12         (output reg active_high feed_reg) "clock pulse for buffer1"
OE1=13         (output reg active_high feed_reg) "output enable for buffer1"
CP2=14         (output reg active_high feed_reg) "clock pulse for buffer2"
OE2=15         (output reg active_high feed_reg) "output enable for buffer2"
DAC=16         (output reg active_high feed_reg) "Signal to load DAC"

;

define  wait      = #b000,
loadbuffer1    = #b001,                      "definitions of every state with aliases"
loadbuffer2    = #b010,
cycle1         = #b011,
DACload1       = #b111,
cycle2         = #b101,
DACload2= #b100,
ill1           = #b110

;

begin
enable (CP1,OE1,CP2,OE2,DAC,c[2:0]);          "all outputs enabled permanently through tri-state output"
case (c[2:0])
begin
    wait)
    begin
        CP1=0;
        OE1=1;
        CP2=0;
        OE2=1;
        DAC=0;
        if(/strobe) then
            begin
                c[2:0]=loadbuffer1;
            end;
        else
            begin
                c[2:0]=wait;
            end;
    end;

    loadbuffer1)
    begin
        CP1=1;
        OE1=1;
        CP2=0;
        OE2=1;
        DAC=0;
        if(strobe) then
            begin
                c[2:0]=loadbuffer2;
            end;
        else
            begin
                c[2:0]=loadbuffer1;
            end;
    end;
end;
loadbuffer2)
begin
    CP1=0;
    OE1=1;
    CP2=1;

```

```

OE2=1;
DAC=0;
if(/strobe)
  then
  begin
    c[2:0]=cycle1;
  end;
else
  begin
    c[2:0]=loadbuffer2;
  end;
end;
cycle1)
begin
  CP1=0;
  OE1=0;
  CP2=0;
  OE2=1;
  DAC=0;
  c[2:0]=DACload1;
end;
DACload1)
begin
  CP1=0;
  OE1=0;
  CP2=0;
  OE2=1;
  DAC=1;
  c[2:0]=cycle2;
end;
cycle2)
begin
  CP1=0;
  OE1=1;
  CP2=0;
  OE2=0;
  DAC=0;
  c[2:0]=DACload2;
end;
DACload2)
begin
  CP1=0;
  OE1=1;
  CP2=0;
  OE2=0;
  DAC=1;
  if(strobe) then
  begin
    begin
      c[2:0]=wait;
    end
    else
    begin
      c[2:0]=cycle1;
    end;
  end;
end;
ill1)
begin
  CP1=0;
  OE1=1;
  CP2=0;
  OE2=1;
  DAC=0;
  c[2:0]=wait;
end;
end;
end;

```

Appendix C: Program listing

DAC.C

```
#include <stdio.h>
#include <conio.h>
#include <dos.h>
#include "regdefs.h" /*register names*/
#include "nidaq.h" /*data aquisition function prototypes*/
#include <time.h>
#include "bragg2.h" //library with functions
/*function for reading what state the PLD is in on DAC
board*/
void DACPLDREAD(int *state_ack)
{
    int state;
    int i;
    /*read PLD state*/
    for(i=0;i<=2;i++)
    {
        DIG_In_Line(DIOBOARD,PORTAPC,i,&state);
        //printf("c%d=%d",i,state);
        state_ack[i]=state;
    }
}
*****
*Function to return DACD PLD to wait state no matter
which state it starts up*
*in
*inputs:none
*outputs:return value to say that the PLD is in wait state
*****
void DACPLDSETUP(void)
{
    int state_ack[3];
    int mem_ack[3];
    int i;
    /*passes pointer of array to DACPLDREAD to
    fill in array*/
    DACPLDREAD(state_ack);

    if( (state_ack[0]==mem_ack[0]!=0)&&
        (state_ack[1]==mem_ack[1]!=0)&&
        (state_ack[2]==mem_ack[2]!=0))
    {
        printf("stuck");
        for(i=0;i<=2;i++)
            printf("c%d=%d",i,state_ack[i]);
        getch();
    }

    mem_ack[0]=state_ack[0];
    mem_ack[1]=state_ack[1];
    mem_ack[2]=state_ack[2];

    /*while PLD is not in wait state*/
    while(!((state_ack[0]==0)&&(state_ack[1]==0)
    &&(state_ack[2]==0)))
    {
        /*read PLD state*/
        DACPLDREAD(state_ack);
        if((state_ack[0]==1)&&(state_ack[1]==0)&&
        (state_ack[2]==0))
        {
            DIG_Out_Line(DIOBOARD,PORTBPA,STR
OBE,1);/*strobe high*/
            if((state_ack[0]==0)&&(state_ack[1]==1)&&(state_ack[2]
]==0))
                DIG_Out_Line(DIOBOARD,PORTBPA,STROBE,0);/*st
robe low*/
            else
                DIG_Out_Line(DIOBOARD,PORTBPA,STROBE,1);
            /*strobe high*/
        }
    }
}
*****
*Function:To output data to DAC Buffers by controlling
PLD
*      and continually output number given to it
*      i.e. loads in same data to each buffer
*inputs:integer value fro scan
*outputs:none
*****
void DATA_OUT_SCAN(unsigned int number)
{
    int i=0;
    int state_ack[3];
    int check=FALSE;
    unsigned int out_buffer[2];

    DACPLDSETUP();

    /*split up word into two bytes*/
    if(number>255){ out_buffer[0]=255-number;
    out_buffer[1]=number>>8;
    }
    else
    {
        out_buffer[0]=number;
        out_buffer[1]=0;
    }
    while(check==FALSE)
    {
        /*set up data at ports data at port*/
        DIG_Out_Port(DIOBOARD,PORTAPB,out_b
uffer[0]);
        DIG_Out_Port(DIOBOARD,PORTAPB,out_b
uffer[1]);
        /*pulse strobe to move to next state*/
        DIG_Out_Line(DIOBOARD,PORTBPA,7,0);
        /*strobe low*/
        /*see whether PLD has moved to next state by
        checking acknowledge*/
        /*read PLD state*/
        DACPLDREAD(state_ack);
        if((state_ack[0]==1)&&(state_ack[1]==0)&&
        (state_ack[2]==0))
        {
            check=TRUE;
            i=i+1;
            /*set up data at ports data at port*/
            DIG_Out_Port(DIOBOARD,PORTAPB,out_b
uffer[0]);
            DIG_Out_Port(DIOBOARD,PORTAPB,out_b
uffer[1]);
            /*pulse strobe to move to next state*/
        }
    }
}
```

```

        DIG_Out_Line(DIOBOARD,PORTBPA,STR
        OBE,1); /*strobe high*/

    }

}

check=FALSE;
while(check==FALSE)
{
/*see whether PLD has moved to next state by
checking acknowledge*/
/*read PLD state*/
DACPLDREAD(state_ack);
if((state_ack[0]==0)&&(state_ack[1]==1)&&(state_ack[2]==0))
{
    check=TRUE;
    DIG_Out_Line(DIOBOARD,PORTBPA,STR
    OBE,0);/*strobe low*/
}
}

/*do not cycle*/
DIG_Out_Line(DIOBOARD,PORTBPA,STR
OBE,1);/*strobe high*/
}

*****
*Function:To output data to DAC Buffers by controlling
PLD
*inputs:pointer to array full of lockin values
*
*outputs:none
*****
void DATA_OUT_LOCK(long int *buffer)
{
    int i=0;
    int state_ack[3];
    int check=FALSE;

    DACPLDSETUP();
    while(check==FALSE)
    {
/*set up data at ports data at port*/
printf("\nbuff[%d]=%ld,buff[%d]=%ld",i,buff
r[i],i+1,buffer[i+1]);
    DIG_Out_Port(DIOBOARD,PORTAPA,buffer
[i]);
    DIG_Out_Port(DIOBOARD,PORTAPP,buffer
[i+1]);
/*pulse strobe to move to next state*/
    DIG_Out_Line(DIOBOARD,PORTBPA,7,0);
/*strobe low*/
/*see whether PLD has moved to next state by
checking acknowledge*/
/*read PLD state*/
DACPLDREAD(state_ack);
if((state_ack[0]==1)&&(state_ack[1]==0)&&(state_ack[2]==0))
{
    check=TRUE;
    i=i+2;
}

/*set up data at ports data at port*/
    DIG_Out_Port(DIOBOARD,PORTAPA,buffer
[i]);
    DIG_Out_Port(DIOBOARD,PORTAPP,buffer
[i+1]);
/*pulse strobe to move to next state*/
    DIG_Out_Line(DIOBOARD,PORTBPA,STR
OBE,1); /*strobe high*/
}
}

}
}

check=FALSE;
while(check==FALSE)
{
/*see whether PLD has moved to next state by
checking acknowledge*/
/*read PLD state*/
DACPLDREAD(state_ack);
if((state_ack[0]==0)&&(state_ack[1]==1)&&(state_ack[2]==0))
{
    check=TRUE;
    DIG_Out_Line(DIOBOARD,PORTBPA,STR
OBE,0);/*strobe low*/
}
}

/*cycle*/
DIG_Out_Line(DIOBOARD,PORTBPA,STR
OBE,0);/*strobe low*/
}

*****
Function:IOsetup
Configures ports for block output of data
Inputs:none
Outputs:none

NB: mode=0 Port is configured for no handshaking
mode=1 Port is configured for handshaking
dir =0 Port is configured as an input port
dir =1 Port is configured as an output port
dir =2 Port is configured as an input
Function format:(board,port,mode,dir)
*****
*****void iOSetup(void)
{
int statusAPAconfig,statusAPBconfig,statusAPCconfig,
statusBPAconfig,statusBPBconfig,statusBPCconfig,
statusCPAconfig,statusCPBconfig,statusCPCconfig,
statusDPAconfig,statusDPBconfig,statusDPCconfig;
int MODE=0;
statusAPAconfig=DIG_Prt_Config(DIOBOARD,PORTA
PA,MODE,1);/*output port*/
statusAPBconfig=DIG_Prt_Config(DIOBOARD,PORTA
PB,MODE,1);/*output port*/
statusAPCconfig=DIG_Prt_Config(DIOBOARD,PORTA
PC,MODE,0);/*input port*/
statusBPAconfig=DIG_Prt_Config(DIOBOARD,PORTB
PA,MODE,1);/*output port*/
statusBPBconfig=DIG_Prt_Config(DIOBOARD,PORTB
PB,MODE,0);/*input port*/

DIG_Prt_Config(DIOBOARD,PORTBPC,MODE,0);/*co
nfig ports*/
/*BPC split into 2 nibbles*/
statusBPCconfig=REG_Level_Write(DIOBOARD,PCDI
096_PPIB_CNFG_REG,255,131,NULL);
/*Due to space constraints and ADC only using 12 bits for
outputting going to use this command to make upper
nibble of port BPC a bitwise
output port,lower nibble input port
REG_Level_Write(device number,registerIndex,bits
affected,bitsettings,registerValue)
device number=DIOBOARD
registerindex for BPC=PCDIO96_PPIB_CNFG_REG-see
regdef.h
bitsaffected-A mask 255 selects all bits
bitsetting=see pg 4-11 PC DIO-6 user manual register
value=NULL*/
/*make sure gain is one*/
}

```

```

GAIN(1,0);

DIG_Out_Line(DIOBOARD,PORTBPA,STROBE,1);
/*strobe high*/
statusCPAconfig=DIG_Prt_Config(DIOBOARD,PORTC
PA,MODE,0);/*input port*/
statusCPBconfig=DIG_Prt_Config(DIOBOARD,PORTC
PB,MODE,0);/*input port*/
statusCPCconfig=DIG_Prt_Config(DIOBOARD,PORTC
PC,MODE,0);/*input port*/
statusDPAconfig=DIG_Prt_Config(DIOBOARD,PORTD
PA,MODE,1);/*output port*/
statusDPBconfig=DIG_Prt_Config(DIOBOARD,PORTD
PB,MODE,1);/*output port*/
statusDPCconfig=DIG_Prt_Config(DIOBOARD,PORTD
PC,MODE,0);/*input port*/



if(
!((statusAPAconfig==0)&&(statusAPBconfig==0)&&(sta
tusAPCconfig==0)&&
(statusBPAconfig==0)&&(statusBPBconfig==0)&&(statu
sBPCconfig==0)&&
(statusCPAconfig==0)&&(statusCPBconfig==0)&&(statu
sCPCconfig==0)&&
(statusDPAconfig==0)&&(statusDPBconfig==0)&&(statu
sDPCconfig==0) ) )
{
    printf("io setup error");
    printf("\nstatusAPAconfig=%d",statusAPAconfig);
    printf("\nstatusAPBconfig=%d",statusAPBconfig);
    printf("\nstatusAPCconfig=%d",statusAPCconfig);
    printf("\nstatusBPAconfig=%d",statusBPAconfig);
    printf("\nstatusBPBconfig=%d",statusBPBconfig);
    printf("\nstatusBPCconfig=%d",statusBPCconfig);
    printf("\nstatusCPAconfig=%d",statusCPAconfig);
    printf("\nstatusCPBconfig=%d",statusCPBconfig);
    printf("\nstatusCPCconfig=%d",statusCPCconfig);
    printf("\nstatusDPAconfig=%d",statusDPAconfig);
    printf("\nstatusDPBconfig=%d",statusDPBconfig);
    printf("\nstatusDPCconfig=%d",statusDPCconfig);
}
}

```

SCAN.C

```

#include <stdio.h>
#include <conio.h>
#include <dos.h>
#include "regdefs.h" /*register names*/
#include "nidaq.h"  /*data aquisition function
prototypes*/

#define COARSE 1
#define FINE 0
#define in_MAX 4096
#define CHANNEL0 0
#define VCOCOARSESTART 0
#define VCOCOARSESTOP 4095 /* highest VCO
coarse input value, max 4095 */
#define VCOCOARSEPEAKLENGTH 50
#define RANGE 10
#define VCOFINEMAX 4095
#define VCOFMININ 0
#define VCOFINEMEAN (VCOFINEMAX-
VCOFMININ)/2 /* mean VCO fine tune input voltage
*/
#define VCOFINEPOINTS 200 /* no of
points on fine tune */
#define COARSE_FINE_RATIO 10 /* ratio of sensitivity
between coarse and fine input*/
#define SLOPE 2 /* rise to detect slope */

```

```

*****
***** Function name:GAIN
***** controls gain stage input to ADC
***** inputs:gain select values G0,G1,G2
***** outputs:none
*****
***** void GAIN(int gain,int channel)
{
    int channel_in;
    if(channel==1) channel_in=128;
    if(channel==0) channel_in=0;

    /*G0 G1 G2*/
    //if(gain==100) outp(x_PORTBPC,(0+channel_in)); /* 0
    0 0*/
    if(gain==100) outp(x_PORTBPC,(16+channel_in)); /* 1
    0 0*/
    if(gain==50) outp(x_PORTBPC,(32+channel_in)); /* 0
    1 0*/
    if(gain==20) outp(x_PORTBPC,(48+channel_in)); /* 1
    1 0*/
    if(gain==10) outp(x_PORTBPC,(64+channel_in)); /* 0
    0 1*/
    if(gain==5) outp(x_PORTBPC,(80+channel_in)); /* 1
    0 1*/
    if(gain==2) outp(x_PORTBPC,(96+channel_in)); /* 0
    1 1*/
    if(gain==1) outp(x_PORTBPC,(112+channel_in));/* 1
    1 1*/
}

*****
***** Function name:scan
***** controls ADC to scan in data from ADC
***** inputs:out array -filled with values 0-2power 16 to be
outputted by the DAC
***** as voltages in the range 0-10V
***** in array -hold sample values
***** length -constant that is the difference
***** channel_in-channel to be sampled from

outputs:none
*****
***** void scan(int *in,int *out,int length,int channel_in,int
scan_type,int gain)
{
    int j=0;//count variable
    int *state_ready=0;//check variables for ADC
    int check=0;
    int scan_buffer[2];//temporary buffer storage
    /*select gain*/
    GAIN(gain,channel_in);
    /*Initialise Inputs to ADC*/
    DIG_Out_Line(DIOBOARD,PORTBPA,RD,1);
    DIG_Out_Line(DIOBOARD,PORTBPA,CS,1);

    for(j=0;j<=length;j++)
    {
        /*output a voltage*/
        if(scan_type==COARSE)
            DATA_OUT_SCAN((out[j]<<4));
        else
            DATA_OUT_SCAN(out[j]);
        /*Read & CS go low to start data conversion*/
        DIG_Out_Line(DIOBOARD,PORTBPA,CS,0)
        ;
        DIG_Out_Line(DIOBOARD,PORTBPA,RD,0
    );
}
```

```

        }
        printf("min=%d,max=%d",*min,max);
        printf("range=%d",range);

        for (i=0;i<length-1;)
        {
        /* look for local & sharp rising edge */
        for (;in[i+1] - in[i] < SLOPE && i<length-1;i++)
        /* find peak */
        for (;in[i+1] >= in[i] && i<length-1;i++)
        /*changed crange division*/
                if (in[i]>*min +
range/RANGE && i < length -1)
                {
                        peak[k]=i;
                        oldI=i;
                /* find peak stop and check for new maximum */
                for (;in[i+1] > (in[peak[k]]+*min)/2;i++)
                (in[i] > in[peak[k]])
                peak[k]=i;
                }
                peakStp[k]=i;
                i=oldI;
                /* find peak start */

for (;in[i-1] > (in[peak[k]]+*min)/2;i--);
/* new peak ? */
if (k==0 || i > peakStp[k-1]
{ peakStrt[k]=i;
    i=peakStp[k];
    k++;
}
else
    i=peakStp[k];
}
}
return k;
}
*****
Function name:coarseScan
tunes VCO from 2-10V,stores the filter output in array
in[],and locates the peaks.
inputs:out array -filled with values 0-2power 16 to be
outputted by the DACas voltages in the range 0-10V
in array -hold sample values
*min-starting minimum value
coarsePk -stores the voltage of the peak in the in array
pkStrt -stores the voltage of the peak in the in array
pkstp -stores the voltage of the peak in the in array
length -constant that is the difference
channel_in-channel to be sampled from
outputs:number of peaks
*****
int coarseScan(int *in,int *out,int *min,int *coarsePk,int
*pkStrt,int *pkStp,int channel,int gain)
{
int i; //count variable
int vcoCoarse;//variable for filling out array
int noOfPk;
const int length = VCOCOARSESTOP-
VCOCOARSESTART+1;
char *coarse_scan= "coarse scan error";

        /*fill out array with values to be outputted*/
        for(i=0,vcoCoarse=VCOCOARSESTART;i<length;
        gth;vcoCoarse++,i++)
        out[i]=vcoCoarse;
        /*output voltage from VCO and read in in array with field
lengthfrom channel 0*/
        scan(in,out,length,channel,COARSE,gain);
}

```

```

        movWinAv(in,length,10);
        noOfPk=peakFind(in,length,min,coarsePk,pk
Str      t,pkStp);
        printf("nno_of peaks%d",noOfPk);
        /*going through out array and finding
        the voltages corresponding to the
        peak coarse,start and stop values*/
        /*printing positional values*/
        for(i=0;i<noOfPk;i++)
printf("\n
NoOfpeaks=%d,\ncoarsepeak=%d,\npkstrt=%d,\npeakstp
=%d",noOfPk,coarsePk[i],pkstrt[i],pkstp[i]);
        for (i=0;i<noOfPk;i++)
{
/*change index into binary output values */
        coarsePk[i] = out[coarsePk[i]];
        pkstrt[i] = out[pkstrt[i]];
        pkstp[i] = out[pkstp[i]];
/* compute approximate fine half height values
*/
        pkstrt[i]= VCOFINEMEAN - (coarsePk[i]-
        pkstrt[i])*COARSE_FINE_RATIO;
        pkstp[i] = VCOFINEMEAN + (pkstp[i] -
        coarsePk[i])*COARSE_FINE_RATIO;

/*check to see if values are sensible*/
        if (pkstp[i] > VCOFINEMAX)pkstp[i] =
VCOFINEMAX;
        if (pkstrt[i] < VCOFINEMIN || pkstrt[i] >
pkstp[i])pkstrt=VCOFINEMIN;
}

return (noOfPk);
}

LOCKIN.C
#include "nidaq.h" /* data acquistition function
prototypes */
#include "regdefs.h" /* register names */
#include <stdio.h>
#include <dos.h>
#include <conio.h>
#include "bragg2.h"

long int pow(long int x, long int y)
{
int i;
long int z=1;

for(i=0;i<=(y-1);i++) z=z*x;
printf("z=%lu",z);
return (z);
}

/*converts 16bit word into two bytes*/
void eight_bit(unsigned int number,long int *word)
{
int v=0;
long int n=0;
long int num=0;
num=number;
/*find biggest subactor of power 2*/
        for(v=0;num>=0;v=v+1)
{
        n=pow(2,v);
        num=num-n;
}
v=v-1;
n=0;
num=number;
/*decrement subtractor by factor of 2*/
/*until differenec is no less than 256*/
        for(;num>=256;v--)
{
        n=pow(2,v);
        num=num-n;
        if(num<0) num=num+n;
}
*word=num;
*(word+1)=((number-num)>>8);
}

*****
Function:lockIn sets up a square wave
inputs corsePk-coarse pk value pertaining to peak in
question
*finePk-array containg peak start peak stop values
pertaining to peak in question.
outputs:none
called in:graphics.c case(BRAGG_MODE),case 2
*****
void lockIn(long int *finePk)
{
    long int buffer[4];//tremporay buffer to pass 8 bit words
    int i;
    printf("\nuptog=%ld,lowtog=%ld",finePk[1],finePk[0]);
    /*spit up word into two bytes to be outputted by two
    ports*/
    eight_bit(finePk[0],&buffer[0]);
    eight_bit(finePk[1],&buffer[2]);

    for(i=0;i<=3;i++) printf("\nbbuffer[%u]=%ld",i,buffer[i]);
    /* things to do if switched to lock*/
    /*1)stop integration and mixing*/
    DIG_Out_Line(DIOBOARD,PORTBPA,INT_MIX,0);
    /*2)output square wave voltage corresponding to peak*/
    DATA_OUT_LOCK(buffer);
    /*3)start integration and mixing*/
    //DIG_Out_Line(DIOBOARD,PORTBPA,INT_MIX,1);
}

```

Appendix D: Transimpedance Amplifier

The detector circuit in figure 3 Chapter 2 consists of a transimpedance amplifier. This is shown in figure 1. This circuit has a d.c. gain of 47×10^7 with $R_f = 47 \times 10^7$ and a bandwidth of 677Hz, with C_f the parasitic capacitance assumed to be 1pF given by Equation 1.

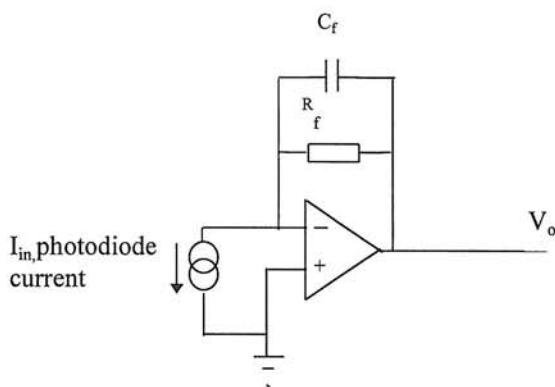


Figure1: Transimpedance amplifier

$$\frac{V_o}{I_{in}} = \frac{-R_f}{1 + j\omega R_f C_f} \quad \text{Equation 1}$$

where the corner frequency

$$f_c = \frac{1}{2\pi R_f C_f}$$

Figure 2 shows an alternative topology¹ which has nearly the same gain as the previous circuit , but greater bandwidth.

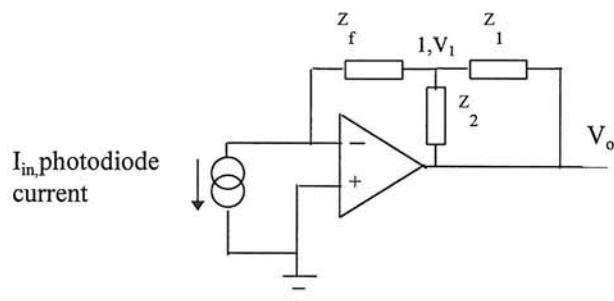


Figure2: Transimpedance amplifier

Z_f , Z_1 and Z_2 represent resistors with their parasitic capacitances (i.e. $Z_f = R_f || C_f$). Applying Kirchoff's current law to node 1 we get a gain expression as per Equation 2.

$$\frac{V_o}{I_{in}} = Z_f \left(\frac{Z_2}{Z_f} + \frac{Z_2}{Z_1} + 1 \right) \quad \text{Equation 2}$$

¹ Design with Operational Amplifiers and Analog Integrated circuits S. Franco pg 59

Substituting the Z values in equation 2 for their parallel combinations of resistors and parasitic capacitances we get equation 3.

$$\frac{V_o}{I_{in}} = \frac{R_2 R_f (C_1 + C_2 + C_f) \left[s + \frac{R_1 R_f + R_2 R_f + R_1 R_2}{R_1 R_2 R_f (C_1 + C_2 + C_f)} \right]}{(1 + s R_2 C_2)(1 + s R_f C_f)} \text{ Equation 3}$$

$$\text{a zero at } \omega_{zero} = \frac{R_1 R_f + R_2 R_f + R_1 R_2}{R_1 R_2 R_f (C_1 + C_2 + C_f)}$$

$$\text{and poles at } \omega_2 = \frac{1}{R_2 C_2} \text{ and } \omega_f = \frac{1}{R_f C_f}$$

values of $R_f = 1.6M\Omega$, $R_2 = 160K\Omega$ and $R_1 = 800\Omega$, assuming $C_1 = C_2 = C_f = 1pF$

gives

$$f_{zero} = 66\text{MHz}, f_f = 100\text{KHz} \text{ and } f_2 = 1\text{MHz}.$$

$$\text{d.c. gain} = 321.6 \times 10^6$$

which gives a response approximated by figure 3.

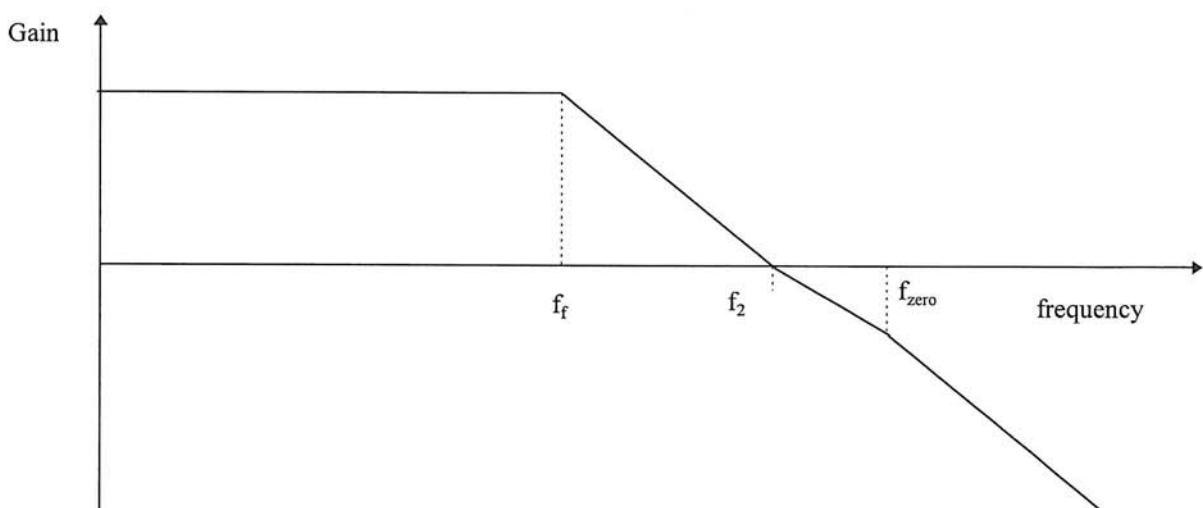


Figure 3: Approximate frequency response of circuit in figure 2.

Appendix E: Cost breakdown of project

	COST £
1)CABLING	
NI INSTRUMENTS NB5 CONNECTOR	33-00
IDC MALE 50 WAY SOCKET	9-33
2xIDC MALE 50 WAY PLUG(PCB MOUNT @2-18)	4-36
2x37 WAY IDC D-TYPE(@£4-36)	8-72
2x25 WAY IDC D-TYPE(@£3-62)	7-24
2x37 D-TYPE (@£1.42)	2-84
2x25 D-TYPE (@£0.80)	1-60
2FT 25 WAY RIBBON CABLE	1-22
2FT 27 WAY RIBBON CABLE	1-82
11xBNC BOX MOUNT(@0-81)	8-91
11xBNC(@0-78)	8-48
2 xSMA BULKHEAD(@11-16)	22-32
4 xSMA PLUG(@5-38)	21-52
SUBTOTAL	131-36
2)BOXES	
2xDIECAST BOX(@5-39)	10-78
2xDIECAST BOX(@3-61)	7-22
SUBTOTAL	18.00
3)SEMICONDUCTOR CHIPS	
2xAD669 DAC(@25-40)	50-80
MAX181 ADC	25-04
MAX358MUX	6-31
MAX400 OPAMP	6-94
OP249 OPAMP	1-23
TLO71 OPAMP(@0-60)	0-60
TLO72 OPAMP(@0-80)	1-60
MM74HC4136	0-59
2x74LS90(@0-51)	1-02
2xHEF4013(@0-54)	1-08
74HC04	0-55
74HC14	0-72
SUBTOTAL	96-48
4)MISCELLANEOUS	
4MHz CRYSTAL	0-50
10XM3 SPACERS (@0-67)	6-70
SUBTOTAL	7-20
TOTAL	253-04

Appendix F: Wiring Schedule

ADC WIRING SCHEDULE

50 way IDC From DIO-96 digital I/O card.	CONVERSION BOX 37 WAY D TYPE	ADC BOARD 37 WAY D TYPE	NAME OF WIRE
32	19	1	D0
30	37	20	D1
28	18	2	D2
26	36	21	D3
24	17	3	D4
22	35	22	D5
20	16	4	D6
18	34	23	D7
16	15	5	D8
14	33	24	D9
12	14	6	D10
10	32	25	D11
42	31	26	A1 INPUT SELECT
9	12	8	/BUSY
44	30	27	RD
7	11	9	/WR -OUTPUT READY
8	29	28	G0 GAIN SELECT
6	10	10	G1
4	28	29	G2
	9	11	+15V
	27	30	0V
	8	12	-15V
	26	31	+5V
	7	3	0V
46	25	32	CHIP SELECT

DAC BOARD WIRING SCHEDUEL

50 WAY IDC from DIO-96 digital I/O card	CONVERSION BOX	DAC BOARD	NAME
47	13	1	D0
45	25	14	D1
43	12	2	D2
41	24	15	D3
39	11	3	D4
37	23	16	D5
35	10	4	D6
33	22	17	D7
31	9	5	D8
29	21	18	D9
27	8	6	D10
25	20	19	D11
23	7	7	D12
21	19	20	D13
19	6	8	D14
17	18	21	D15
15	5	9	C0

13	17	22	C1
11	4	10	C2
34	16	23	STROBE
	3	11	+5V
	15	24	0V
	2	12	+15V
	14	25	0V
	1	13	-15V

MISCELLANEOUS

50 WAY IDC from DIO-96 digital I/O card	DESTINATION	NAME
1	COUNTER	CC0
2	COUNTER	CC1
3	COUNTER	RCO
36	COUNTER	DATA_READY
38	FREE	
40	FEEDBACK LOOP	MIX_INT
42	FREE	
48	COUNTER	/OE

Appendix G: DATA SHEETS

Evaluation Kit Manual
Follows Data Sheet

MAXIM

Complete, 8-Channel, 12-Bit Data-Acquisition Systems

MAX180/MAX181

General Description

The MAX180/MAX181 are complete 12-bit Data Acquisition System (DAS) which combine 8/6-channel input multiplexer, high bandwidth Track-and-Hold (T/H), low-drift zener reference, and flexible microprocessor (μ P) interface with high conversion speed and low power consumption. The MAX180/MAX181 can be configured by a μ P for unipolar or bipolar conversions and single-ended or differential inputs. Both devices sample and digitize at 100kHz throughput rate and feature a fast 8- or 16-bit μ P interface.

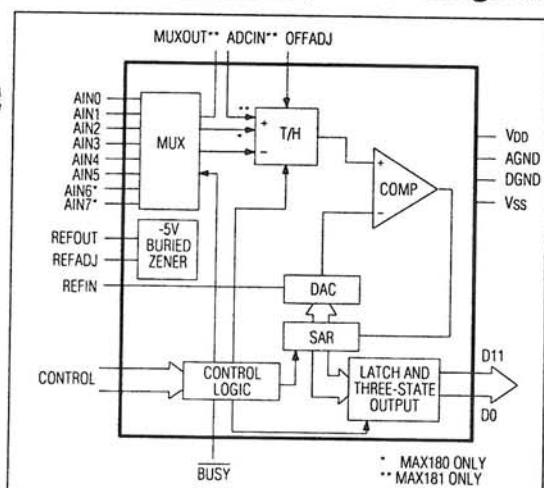
The MAX180 has 8 analog input channels, while the MAX181 has 6. The multiplexer output of the MAX180 is fed directly into the Analog-to-Digital Converter (ADC) input. The MAX181 brings out both the multiplexer output and ADC input to separate pins, allowing a programmable gain amplifier to be inserted between the MUX and the ADC.

The systems allow the user to choose between an internal or an external reference. Furthermore, the internal reference value and the offset can be adjusted, allowing the overall system gain and offset errors to be nulled. The multiplexer has high impedance inputs, simplifying analog drive requirements.

Applications

- High-Speed Servo Loops
- Digital-Signal Processing
- High-Accuracy Process Control
- Automatic Testing Systems

Block Diagram



Features

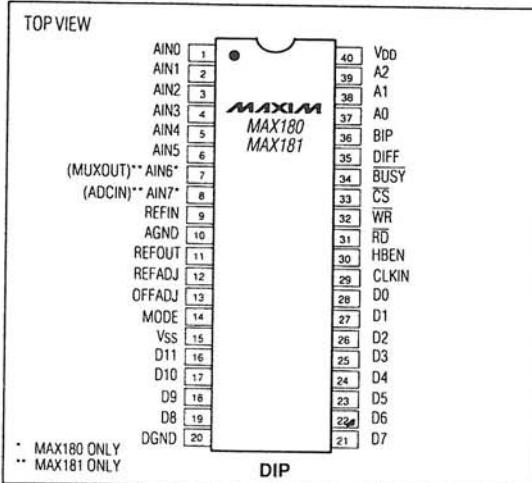
- ◆ 12-Bit Resolution, $\pm 1/2$ LSB Linearity
- ◆ 8-Channel Multiplexed Inputs (MAX180)
- ◆ Single-Ended 1-of-6 Multiplexer (MAX181)
- ◆ Built-In Track-and-Hold
- ◆ 100kHz Sampling Rate
- ◆ DC and Dynamically Tested
- ◆ Internal 25ppm/ $^{\circ}$ C Voltage Reference
- ◆ Each Channel Configurable for Unipolar (0V to +5V) or Bipolar (-2.5V to +2.5V) Input Range
- ◆ Each Channel Configurable for Single-Ended or Differential Inputs
- ◆ Fast 8-/16-Bit μ P Interface
- ◆ +5V and -12V to -15V Supply Operation
- ◆ 110mW Power Consumption

Ordering Information

PART	TEMP. RANGE	PIN-PACKAGE	ERROR (LSBs)
MAX180ACPL	0°C to +70°C	40 Plastic DIP	$\pm 1/2$
MAX180BCPL	0°C to +70°C	40 Plastic DIP	± 1
MAX180CCPL	0°C to +70°C	40 Plastic DIP	± 1
MAX180ACQH	0°C to +70°C	44 PLCC	$\pm 1/2$
MAX180BCQH	0°C to +70°C	44 PLCC	± 1

Ordering information continued on last page.

Pin Configurations



Maxim Integrated Products 1

Call toll free 1-800-998-8800 for free samples or literature.

Complete, 8-Channel, 12-Bit Data-Acquisition Systems

MAX180/MAX181

ABSOLUTE MAXIMUM RATINGS

VDD to DGND	-0.3V, +7V
VSS to DGND	-0.3V, -17V
AGND to DGND	-0.3V, VDD + 0.3V
AIN_-, MUXOUT, ADCIN, REFADJ,	
OFFADJ to REFIN	-0.3V, VDD + 0.3V
REFIN to DGND	+0.3V, VSS - 0.3V
CS, WR, RD, CLK, A2-A0,	
BIP_DIFF, HBEN to DGND	-0.3V, VDD + 0.3V
BUSY, D0-D11 to DGND	-0.3V, VDD + 0.3V

Continuous Power Dissipation (any package)	
to +70°C	1000mW
derates above +70°C by	10mW/°C
Operating Temperature Ranges:	
MAX18_C	0°C to +70°C
MAX18_E	-40°C to +85°C
MAX18_MJL	-55°C to +125°C
Storage Temperature Range	-65°C to +160°C
Lead Temperature (soldering, 10 sec)	+300°C

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

ELECTRICAL CHARACTERISTICS

(VDD = +5V ±5%, VSS = -12V ±5% or -15V ±5%, REFIN = -5V, Internal Reference Mode, Bipolar Mode, Slow-Memory Mode (see text), fCLK = 1.6MHz external, MAX180/MAX181 all grades, TA = TMIN to TMAX, unless otherwise noted.) (Note 1)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
ACCURACY (Note 2)						
Resolution	N		12			Bits
Integral Nonlinearity Error	INL	MAX18_A		±1/2		LSB
		MAX18_B/C		±1		
Differential Nonlinearity Error	DNL	Guaranteed monotonic over temperature		±1		LSB
Unipolar Offset Error (Note 3)				±1	±4	LSB
Bipolar Offset Error (Note 3)				±1	±6	LSB
Unipolar Gain Error				±2	±10	LSB
Bipolar Gain Error				±2	±15	LSB
Gain-Error Tempco (Note 4)				±5		ppm/°C
Channel-to-Channel Matching				±1/4		LSB
DYNAMIC PERFORMANCE (Note 2)						
Signal-to-Noise + Distortion Ratio	SINAD	10kHz input signal, 100kHz sampling rate, bipolar mode, TA = +25°C	70			dB
Total Harmonic Distortion (up to the 5th harmonic)	THD	10kHz input signal, 100kHz sampling rate, bipolar mode, TA = +25°C		-80		dB
Spurious-Free Dynamic Range	SFDR	10kHz input signal, 100kHz sampling rate, bipolar mode, TA = +25°C	80			dB
Full-Power Sampling Bandwidth		In track mode, under-sampled waveform	6			MHz
Track-and-Hold Acquisition Time (Note 5)	tACQ		1.875			μs
Conversion Time	tCONV	Asynchronous hold mode	7.500	8.125		μs
		Note 5 ROM, Slow-Memory, and I/O Port Modes; 15-16 clock cycles	9.375	10.000		
ANALOG INPUT						
Voltage Range		AIN_-, MUXOUT, and ADCIN	REFIN	VDD		V
Unipolar, Single-Ended Range		AIN_- to AGND	0	5.0		
Unipolar, Differential Range		AIN_+ to AIN_-	0	5.0		
Bipolar, Single-Ended Range		AIN_- to AGND	-2.5	2.5		
Bipolar, Differential Range		AIN_+ to AIN_-	-2.5	2.5		

Complete, 8-Channel, 12-Bit Data-Acquisition Systems

MAX180/MAX181

ELECTRICAL CHARACTERISTICS (continued)

($V_{DD} = +5V \pm 5\%$, $V_{SS} = -12V \pm 5\%$ or $-15V \pm 5\%$, $REFIN = -5V$, Internal Reference Mode, Bipolar Mode, Slow-Memory Mode (see text), $f_{CLK} = 1.6MHz$ external, MAX180/MAX181 all grades, $T_A = T_{MIN}$ to T_{MAX} , unless otherwise noted.) (Note 1)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
ANALOG INPUT (continued)						
Input Current		AIN ₋ , ADCIN,	MAX180 MAX181		±1.0 ±0.1	µA
Mux-On Resistance	R _{ON}	A _{IN} ₋ = 2.5V, I _{MUXOUT} = 1.25mA, MAX181		2	kΩ	
Mux-On Leakage Current	I _{ON}	A _{IN} ₋ = MUXOUT = ±5V, MAX181		±100	nA	
Mux-Off Leakage Current	I _{IN (OFF)} I _{OUT (OFF)}	A _{IN} ₋ = ±5V, V _{OUT} = ±5V, MAX181 A _{IN} ₋ = ±5V, V _{OUT} = ±5V, MAX181		±100 ±100	nA	
Input Capacitance (Note 5)	C _{IN}	A _{IN} ₋ , ADCIN MUXOUT	25 35 35 45	35 45	pF	
REFERENCE INPUT						
Input Range (Note 5)			-4.92	-5.00	-5.08	V
Input Current				-2	mA	
Input Resistance			2.5			kΩ
REFERENCE OUTPUT						
V _{REF} Output Voltage		T _A = +25°C	-4.98	-5.00	-5.02	V
V _{REF} Output Tempco (Note 6)		MAX18_A/B		25	ppm/°C	
		MAX18_C		45		
V _{REF} Load Regulation (Note 7)		I _{OUT} = 0mA to 5mA, T _A = +25°C	0.2	1.0	mV/mA	
REFADJ, OFFADJ						
Input Current		V _{REFADJ} , V _{OFFADJ} = V _{DD} to REFIN		±1	µA	
Disable Threshold			4.5			V
REFADJ Adjustment Range		REFIN < REFADJ < AGND	±60	±80		mV
OFFADJ Adjustment Range		REFIN < OFFADJ < AGND	±15	±25		LSB
LOGIC INPUTS						
Input Low Voltage	V _{IL}	MODE CS, RD, WR, CLK, A2-A0, DIFF, BIP, HBEN	0.5 0.8			V
Input High Voltage	V _{IH}	MODE CS, RD, WR, CLK, A2-A0, DIFF, BIP, HBEN	4.5 2.4			V
Input Mid-Level Voltage	V _{IMID}	MODE	1.5	3.5		V
Input Floating Voltage	V _{FILT}	MODE	2.5			V
Input Current	I _{IN}	MODE CS, RD, WR, CLK, A2-A0, DIFF, BIP, HBEN	T _A = +25°C T _A = T _{MIN} to T _{MAX} T _A = +25°C T _A = T _{MIN} to T _{MAX}	±50 ±50 ±1 ±10	±100 ±100 ±1 ±10	µA
Input Capacitance (Note 5)	C _{IN}			15	pF	
LOGIC OUTPUTS						
Output Low Voltage	V _{OL}	D11-D0, BUSY, RDY, I _{SINK} = 1.6mA		0.4		V
Output High Voltage	V _{OH}	D11-D0, BUSY, RDY, I _{SOURCE} = 360µA	4.0			V
Floating State Leakage Current	I _{LKG}	D11-D0, V _{OUT} = 0V to V _{DD}		±10	µA	
Floating State Output Capacitance (Note 5)	C _{OUT}			15	pF	



Monolithic 16-Bit DACPORT

AD669

FEATURES

- Complete 16-Bit D/A Function
- On-Chip Output Amplifier
- High Stability Buried Zener Reference
- Monolithic BiMOS II Construction
- ± 1 LSB Integral Linearity Error
- 15-Bit Monotonic over Temperature
- Microprocessor Compatible
- 16-Bit Parallel Input
- Double-Buffered Latches
- Fast 40 ns Write Pulse
- Unipolar or Bipolar Output
- Low Glitch: 15 nV-s
- Low THD+N: 0.009%
- MIL-STD-883 Compliant Versions Available

PRODUCT DESCRIPTION

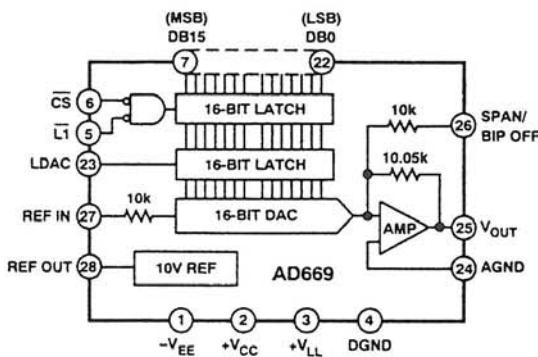
The AD669 DACPORT™ is a complete 16-bit monolithic D/A converter with an on-board reference and output amplifier. It is manufactured on Analog Devices' BiMOS II process. This process allows the fabrication of low power CMOS logic functions on the same chip as high precision bipolar linear circuitry. The AD669 chip includes current switches, decoding logic, an output amplifier, a buried Zener reference and double-buffered latches. The AD669's architecture insures 15-bit monotonicity over temperature. Integral nonlinearity is maintained at $\pm 0.003\%$, while differential nonlinearity is $\pm 0.003\%$ max. The on-chip output amplifier provides a voltage output settling time of 10 μ s to within 1/2 LSB for a full-scale step.

Data is loaded into the AD669 in a parallel 16-bit format. The double-buffered latch structure eliminates data skew errors and provides for simultaneous updating of DACs in a multi-DAC system. Three TTL/LSTTL/5 V CMOS compatible signals control the latches: CS, L1 and LDAC.

The output range of the AD669 is pin programmable and can be set to provide a unipolar output range of 0 V to +10 V or a bipolar output range of -10 V to +10 V.

The AD669 is available in seven grades: AN and BN versions are specified from -40°C to +85°C and are packaged in a 28-pin DIP. The AR and BR versions are specified for -40°C to -15°C operation and are packaged in a 28-pin SOIC. The AQ and BQ versions are also specified from -40°C to +85°C, while the SQ version is specified from -55°C to +125°C. The AQ, BQ and SQ versions are all packaged in a hermetic 28-pin ceramic package. The AD669 is also available compliant to MIL-STD-883. Refer to the AD669/883B data sheet for specifications and test conditions.

FUNCTIONAL BLOCK DIAGRAM



PRODUCT HIGHLIGHTS

1. The AD669 is a complete voltage output 16-bit DAC with voltage reference and digital latches on a single IC chip.
2. The internal buried Zener reference is laser trimmed to 10,000 volts with a $\pm 0.2\%$ maximum error. The reference voltage is also available for external applications.
3. The AD669 is both dc and ac specified. DC specs include ± 1 LSB INL error and ± 1 LSB DNL error. AC specs include 0.009% THD+N and 0.0063% SNR. The ac specifications make the AD669 suitable for signal generation applications.
4. The double-buffered latches on the AD669 eliminate data skew errors while allowing simultaneous updating of DACs in multi-DAC systems.
5. The output range is a pin-programmable unipolar 0 V to +10 V or bipolar -10 V to +10 V output. No external components are necessary to set the desired output range.
6. The AD669 is available in versions compliant with MIL-STD-883. Refer to the Analog Devices Military Products Databook or current AD669/883B data sheet for detailed specifications.

DACPORT is a trademark of Analog Devices, Inc.

AD669—SPECIFICATIONS

($T_A = +25^\circ\text{C}$, $V_{CC} = +15\text{ V}$, $V_{EE} = -15\text{ V}$, $V_{LL} = +5\text{ V}$ unless otherwise stated)

Model	AD669AN/AR			AD669AQ/SQ			AD669BN/BQ/BR			Units
	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
RESOLUTION	16			16			16			Bits
DIGITAL INPUTS (T_{MIN} to T_{MAX})										
V_{IH} (Logic "1")	2.0	5.5	*	*	*	*	*	*	*	Volts
V_{IL} (Logic "0")	0	0.8	*	*	*	*	*	*	*	Volts
I_{IH} ($V_{IH} = 5.5\text{ V}$)			± 10							μA
I_{IL} ($V_{IL} = 0\text{ V}$)			± 10							μA
TRANSFER FUNCTION CHARACTERISTICS¹										
Integral Nonlinearity			± 2			*			± 1	LSB
T_{MIN} to T_{MAX}			± 4			*			± 2	LSB
Differential Nonlinearity			± 2			*			± 1	LSB
T_{MIN} to T_{MAX}			± 4			*			± 2	LSB
Monotonicity Over Temperature	14			14			15			Bits
Gain Error ^{2, 3}			± 0.15			± 0.10			± 0.10	% of FSR
Gain Drift ² (T_{MIN} to T_{MAX})			25			15			15	$\text{ppm}/^\circ\text{C}$
Unipolar Offset			± 5			± 5			± 2.5	mV
Unipolar Offset Drift (T_{MIN} to T_{MAX})			5			3			3	$\text{ppm}/^\circ\text{C}$
Bipolar Zero Error			± 15			± 15			± 10	mV
Bipolar Zero Error Drift (T_{MIN} to T_{MAX})			12			10			5	$\text{ppm}/^\circ\text{C}$
REFERENCE INPUT										
Input Resistance	7	10	13	*	*	*	*	*	*	$\text{k}\Omega$
Bipolar Offset Input Resistance	7	10	13	*	*	*	*	*	*	$\text{k}\Omega$
REFERENCE OUTPUT										
Voltage	9.98	10.00	10.02	*	*	*	*	*	*	Volts
Drift			25			15			15	$\text{ppm}/^\circ\text{C}$
External Current ³	2	4		*	*	*	*	*	*	mA
Capacitive Load										pF
Short Circuit Current			25			*				mA
OUTPUT CHARACTERISTICS										
Output Voltage Range										
Unipolar Configuration	0		+10	*		*	*		*	Volts
Bipolar Configuration	-10		+10	*		*	*		*	Volts
Output Current	5			*			*		*	mA
Capacitive Load										pF
Short Circuit Current			25			*				mA
POWER SUPPLIES										
Voltage										
V_{CC}^4		+13.5	+16.5	*		*	*		*	Volts
V_{EE}^4		-13.5	-16.5	*		*	*		*	Volts
V_{LL}^4		+4.5	+5.5	*		*	*		*	Volts
Current (No Load)										
I_{CC}		+12	+18	*		*	*		*	mA
I_{EE}		-12	-18	*		*	*		*	mA
I_{LL}										
@ $V_{IH}, V_{IL} = 5, 0\text{ V}$		0.3	2	*		*	*		*	mA
@ $V_{IH}, V_{IL} = 2.4, 0.4\text{ V}$		3	7.5	*		*	*		*	mA
Power Supply Sensitivity		1	3	*		*	*		*	$\text{ppm}/\%$
Power Dissipation (Static, No Load)		365	625	*						mW
TEMPERATURE RANGE										
Specified Performance (A, B)	-40		+85	-40		+85	-40		+85	$^\circ\text{C}$
Specified Performance (S)				-55		+125				$^\circ\text{C}$

NOTES

¹For 16-bit resolution, 1 LSB = 0.0015% of FSR = 15 ppm of FSR. For 15-bit resolution, 1 LSB = 0.003% of FSR = 30 ppm of FSR. For 14-bit resolution, 1 LSB = 0.006% of FSR = 60 ppm of FSR. FSR stands for Full-Scale Range and is 10 V for a 0 to +10 V span and 20 V for a -10 V to +10 V span.

²Gain error and gain drift measured using the internal reference. Gain drift is primarily reference related. See the Using the AD669 with the AD688 Reference section for further information.

³External current is defined as the current available in addition to that supplied to REF IN and SPAN/BIPOLAR OFFSET on the AD669.

⁴Operation on $\pm 12\text{ V}$ supplies is possible using an external reference like the AD586 and reducing the output range. Refer to the Internal/External Reference Use section.

⁵Measured with fixed $50\ \Omega$ resistors. Eliminating these resistors increases the gain error by 0.25% of FSR (Unipolar mode) or 0.50% of FSR (Bipolar mode).

Refer to the Analog Circuit Connections section.

*Same as AD669AN/AR specification.

Specifications subject to change without notice.

Specifications in boldface are tested on all production units at final electrical test. Results from those tests are used to calculate outgoing quality levels. All

and max specifications are guaranteed. Those shown in boldface are tested on all production units.

PERFORMANCE CHARACTERISTICS

(With the exception of Total Harmonic Distortion + Noise and Signal-to-Noise
≤ TA ≤ T_{MAX}, V_{CC} = +15 V, V_{EE} = -15 V, V_{LL} = +5 V except where stated.)

Parameter	Limit	Units	Test Conditions/Comments
Output Settling Time Time to ±0.0008% FS with 2 kΩ, 1000 pF Load)	13 8 10 6 8 2.5	μs max μs typ μs typ μs typ μs typ μs typ	20 V Step, TA = +25°C 20 V Step, TA = +25°C 20 V Step, T _{MIN} ≤ TA ≤ T _{MAX} 10 V Step, TA = +25°C 10 V Step T _{MIN} ≤ TA ≤ T _{MAX} 1 LSB Step, T _{MIN} ≤ TA ≤ T _{MAX}
Total Harmonic Distortion + Noise A, B, S Grade	0.009	% max	0 dB, 1001 Hz; Sample Rate = 100 kHz; TA = +25°C
A, B, S Grade	0.056	% max	-20 dB, 1001 Hz; Sample Rate = 100 kHz; TA = +25°C
A, B, S Grade	5.6	% max	-60 dB, 1001 Hz; Sample Rate = 100 kHz; TA = +25°C
Signal-to-Noise Ratio	0.0063	% max	A-Weight Filter; TA = +25°C
Digital-to-Analog Glitch Impulse	15	nV·s typ	DAC Alternately Loaded with 8000H and 7FFFH
Digital Feedthrough	2	nV·s typ	DAC Alternately Loaded with 0000H and FFFFH; CS High
Output Noise Voltage Density (1 kHz - 1 MHz)	120	nV/√Hz typ	Measured at V _{OUT} , 20 V Span; Excludes Reference
Reference Noise	125	nV/√Hz typ	Measured at REF OUT

Specifications subject to change without notice.
 Specifications in boldface are tested on all production units at final electrical test. Results from those tests are used to calculate outgoing quality levels. All min and max specifications are guaranteed. Those shown in boldface are tested on all production units.

TIMING CHARACTERISTICS

V_{CC} = +15 V, V_{EE} = -15 V, V_{LL} = +5 V, V_{HI} = 2.4 V, V_{LO} = 0.4 V

Parameter	Limit +25°C	Limit -40°C to +85°C	Limit -55°C to +125°C	Units
Figure 1a)				
CS	40	50	55	ns min
L1	40	50	55	ns min
t _{OS}	30	35	40	ns min
t _{OH}	10	10	15	ns min
t _{LH}	90	110	120	ns min
t _{LW}	40	45	45	ns min
Figure 1b)				
t _{LOW}	130	150	165	ns min
t _{HIGH}	40	45	45	ns min
t _{DS}	120	140	150	ns min
t _{DH}	10	10	15	ns min

Specifications subject to change without notice.

Specifications in boldface are tested on all production units at final electrical test. Results from those tests are used to calculate outgoing quality levels. All min and max specifications are guaranteed. Those shown in boldface are tested on all production units.

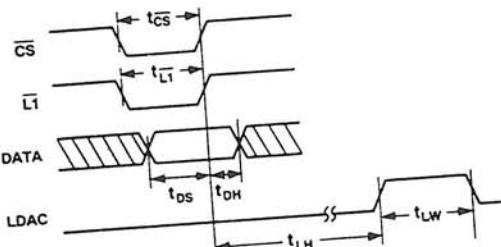


Figure 1a. AD669 Level Triggered Timing Diagram

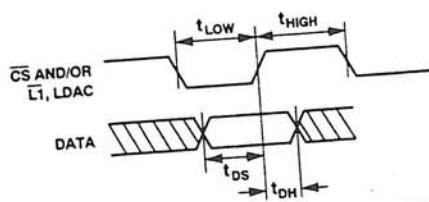


Figure 1b. AD669 Edge Triggered Timing Diagram
 TIE CS AND/OR L1 TO GROUND OR TOGETHER WITH LDAC

(1.5 × 10⁻⁹)