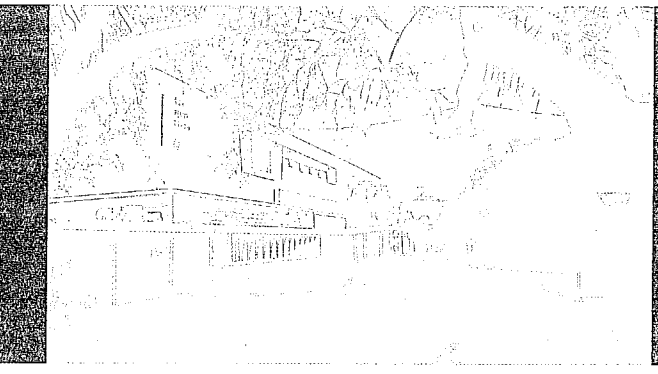
PRL-TN-2010-97

Technical Note



High voltage power supply unit for Lithium Niobate Fabry-Perot etalon in Multi Application Solar Telescope (MAST) Imager

by
S. K. Mathew & S. K. Gupta
(Udaipur Solar Observatory)

भौतिक अनुसंधान प्रयोगशाला, अहमदाबाद
Physical Research Laboratory, Ahmedabad

Published by
The Dean's office, PRL.

Contents

High voltage power supply unit for Lithium Niobate Fabry-Perot etalon in Multi Application Solar Telescope (MAST) Imager

S. K. Mathew & S. K. Gupta
(Udaipur Solar Observatory)
(shibu@prl.res.in)

Abstract

In this report, we present the details of a high voltage power supply unit, for tuning the solid state lithium niobate Fabry-Perot etalon filter. Application of high voltage across the lithium niobate crystal changes its refractive index and thus the wavelength of the transmission peak of the filter. Two high voltage power supply modules were procured from Applied Kilovolts, UK, the output of which can be changed within ±5000 volts by changing the control input of ±10 volts. The FTDI-DAC driver circuit and the associated software were developed in-house to drive the high voltage power supply unit for the application in the Multi Application Solar Telescope (MAST) imager. The details of the high voltage unit, along with the driver program are presented.

Introduction

The visible light imager instrument in the MAST telescope will use two lithium niobate Fabry - Perot etalons as narrow band filters. Lithium niobate crystal is electro-optic in nature and a tuning of the above filters to different wavelengths can be accomplished by changing the voltage across the crystal. A change in voltage varies the refractive index of the crystal and thus the wavelength of the peak transmission. In order to tune the passband of the etalon to different wavelengths, high voltage power supplies (HVPS) are required. For example, tuning the filter used in the imager, to around 1.8Å (half of the free-spectral-range, FSR, which is the normal range for which the filter has to be tuned) needs a change of 4000 volts. Often tuning of the filter in very small wavelength steps is required. This requires precise control over the applied voltage. Normally, tuning the filter to different wavelengths and image acquisition are synchronized and is controlled by a PC. In this report we give the details of the construction of a high voltage power supply unit, which drive two lithium niobate Fabry-Perot etalons, simultaneously. The high-voltage power supply modules were procured from Applied Kilovolts, UK, and its driver circuit and the associated software were developed in house. The USB based controller provides a very precise control (less than one volts) over the applied high voltage and allow a synchronization of the voltage changes with the image acquisition.

Similar power supply units are available with Commonwealth Scientific and Industrial Research Organization (CSIRO), from where we procured the Fabry-Perots. Their version of the units are slow in response to input voltage changes, and are not suitable for our application. More over two independent units are needed for controlling each of the etalons and a synchronization of the entire image acquisition programs with voltage changes is needed. This can be achieved easily in the system built by us, since all the programs that drive the power supply and the image acquisition are developed for this purpose in house. This entire power supply unit could be commissioned at a fifth of the commercial costs.

Scheme for the high voltage power supply unit

Fig. 1 provides the block diagram showing the high voltage power supply. The controller is driven by a USB port and an FTDI (Future Technology Devices Inc.), FIFO chip. A single FTDI chip provides all the signals for the two digital-to-analog converters (DAC). The voltage output of the high voltage power supply module (HVPM) can be programmed through a low voltage control input derived from the DAC. A change of ±10 volts in the control voltage translates a change in HVPM output by ±5000 volts. The DAC output voltage is programmed using a driver software written in C language.

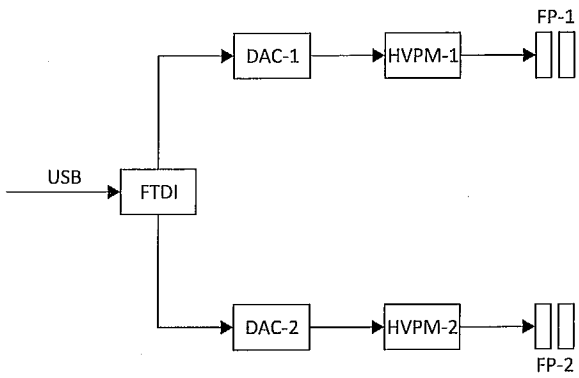


Fig. 1: Block diagram of the high voltage power supply unit for the Lithium niobate Fabry-Perot etalons. DAC and HVPM are D/A converter and high voltage power supply modules, FP is the solid state Lithium niobate Fabry-Perot etalon.

High voltage power supply modules

Two high voltage power supply modules were used to get the desired high voltage to tune the Fabry-Perot etalons. The characteristics of the power supply module are given in Table1. The control system of the power supply module is discussed in the following sections.

Table 1: *Stated specification of the HVPM from Applied Kilovolts, UK.*

Model	:	HP005ZIP025
Output voltage	:	±5000 volts
Maximum output current	:	400μA
Ripple	:	170mV (peak-to-peak)
Supply voltage	:	+24 Vdc
Input current (max)	:	1A
Operational temperature	:	0 to +45 degC
Temperature coefficient	:	25ppm/deg C
Remote programming	:	±10 Vdc, for min. to max. output

Schematic diagram of the control circuit

Appendix 1a gives the schematic diagram of the control circuit. The DAC – Computer interface is done through an FTDI chip, FT245BL, which is a USB to parallel converter. Using the available driver software for this chip, the FIFO data bus (pin no. 18 – 25, D7 – D0) is programmed, and can be used as a parallel port. Only few external components are needed for the functioning of this chip. We used the chip in USB powered configuration, and an additional EPROM, 93C46 is connected to the chip to facilitate to store and use a particular identification tag for this device. More information on this chip can be found at http://www.ftdichip.com. Four pins of the FIFO data bus was used for applying the required timing signals for each DACs. Data pins D0-D3 were used for the first DAC and D4-D7, for the second.

A 16-bits, high precision digital-to-analog, converter, DAC714, from Burr-Brown (now Texas Instruments) was used to convert the digital counts to analog DC voltages. The DAC714 is a monolithic circuit, including a +10V temperature compensated reference, current-to-voltage amplifier, a high-speed synchronous serial interface, and an asynchronous clear function which sets the output voltage to mid scale instantaneously. More information on this chip can be found at http://www.ti.com. The DAC can be wired to generate ±10 volts at the output, in steps of 0.305 milli volts. This is equivalent to 152 milli volts (amplification factor = 500) in the out-put of high voltage power supply module and is close to the ripple.

Fig. 2 shows the timing diagrams for the D/A converter. The DAC714 has a serial interface with two data buffers, and the 16-bits digital data is clocked through the SDI pin. $\overline{A0}$ is the enable control for the input shift registers, and in the present case, it is hard wired to the ground, enabling the continuous data flow to the shift registers. $\overline{A1}$ is the update pin for the D/A latch. $\overline{CLK}$ is used to strobe the 16 bits data in to the shift register and then the last pulse is used to update the D/A latch. This is done by pulling the $\overline{A1}$ low. Four independent lines from the FIFO data bus of the FTDI chip were used to operate each DAC.

The DAC714 accepts binary 2s complement (BTC) input codes with MSB first, which are compatible with bipolar analog output operations. For this configuration a digital input of $7FFF_H$ produces a full scale output, 8000_H produces a minus full-scale output, and 0000_H produces a bipolar-zero output. Table 2 give the pin numbers of DAC 1 and 2 and the corresponding pin numbers on the FTDI chip. The decimal data values written to the FTDI required to pull these pins up are also included in the table.

Table 2: *Pin connections for FT245 BL and DAC714*

FTDI (FT245BL pin no.)	DAC1 (DAC714 pin no.)	DAC2 (DAC714 pin no.)	Date values for FTDI pins
D0 (25)	SDI (4)		1
D1 (24)	A1 (3)		2
D2 (23)	CLK (1)		4
D3 (22)	CLR (16)		8
D4 (21)		SDI (4)	16
D5 (20)		A1 (3)	32
D6 (19)		CLK (1)	64
D7 (18)		CLR (16)	128

Driver software to generate the analog voltages

The driver software for clocking the digital data to the DAC is written in C language and is included in Appendix 2. The program also uses 'ftd2xx.lib' library and associated 'dll' files provided by Future Devices Inc. for the particular chip, FT245BL. The timing diagram shown in Fig. 2 is simulated, and converted to the desired signals through the FTDI, FIFO data bus. The program can be run from the command prompt by typing,

mastfp inpv0 inpv1 flag0 flag1

'inpv0' and 'inpv1' are the required high voltage output at the HVPM1 and HVPM2. 'flag0' and 'flag1' are provided in case if the voltage at the output need not to be changed. Setting these flags to 1 will not allow the update of the data latch of the DAC, keeping the analog output voltage from the DAC unchanged. The binary data equivalent of the desired voltage is generated from the input parameters. Since the input voltage values are given as the high voltage required for tuning the etalon, this is changed to the DAC analog range by dividing it with the amplification factor 500 (a control voltage of ±10 volts produce a dc voltage ±5000 volts at the high voltage output). The binary conversion is included in the subroutine 'clock_the_data', which also takes care of the sign of the voltage. The data is stored in a 16 elements array and is clocked sequentially to the shift register of the DAC using the same subroutine, the 'for' loop towards the end of this subroutine does the same. The main function used to write the data to the FT245BL is,

ft Status = FT_Write (ft Handle, &BTW, sizeof (BTW), & Bytes Written)

where 'fthandle' provides handle to the chip obtained from the identification tag programmed to the EPROM (in our case the tag is 'MAST_HV_FP_PS'), and BTW is the pointer to the data to be written to the FTDI, FIFO data bus. The combined value of data bit, the state of the clocking pulse, and other signals such as A1 and CLR is passed as a single value to the pointer BTW. For example, setting the following value,

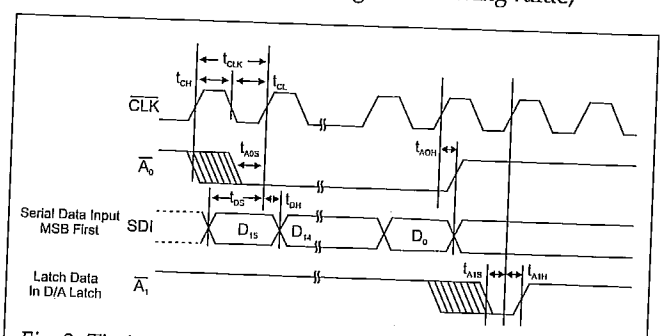


Fig. 2. Timing diagram for the DAC714. $\overline{A0}$ in our case is wired permanently to the ground. $\overline{A1}$ is pulled low to update the D/A data latch.

*BTW = SDI0*D0 [15-kk] + CLK0*1 + CLR0*1 + A10*1 + SDI1*D1 [15-kk] + CLK1*1 +CLR1*1 + A11*1*

The $\overline{CLK}$ pins of both DAC1 and DAC2 is pulled high, and also $\overline{A1}$ and $\overline{CLR}$ pins high, the serial data pin SDI is set to high or low depending upon the value present at that location D. The last step after clocking all the 16 bits data to the DAC shift register is to update the D/A data latch, which is done by pulling the A1 pin to low. The update is done depending upon the flags passed from the input parameters.

The assembled system

Fig. 3 show the FTDI-DAC board and the assembled high voltage power pack. Component list used for the entire circuit is given in Appendix 3. The design of printed circuit board and soldering of the components are done in-house. Two high voltage, SHV series BNC connectors from *Radiall* are used to connect the high voltage output to the Fabry-Perots.

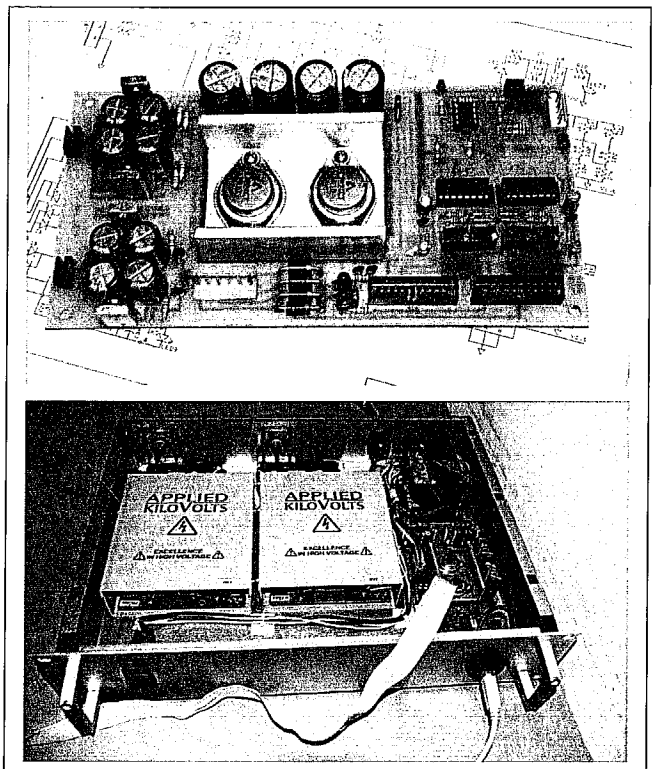


Fig. 3. The assembled FTDI-DAC board, two 16 pin ICs in the right side of the board are the DAC1 and 2. The FT245BL is a surface mount component and is soldered in the back side of the PCB. (right) Complete high voltage power supply unit under test. The high voltage from HVPMs are connected to high voltage BNC sockets fixed in the back side of the enclosure.

Tests carried out

The gain and offset adjustment for the DACs are carried out by adjusting the presets, following the procedure given in the data sheet of DAC714. The first test we carried out is to check the linearity of the DAC output analog voltage to the input data. Fig. 4 (a) shows the input (divided by 500 to remove the amplification factor) and the DAC analog voltage for the range of ±10 volts. The measurement is carried out with an HP 3486A multi-meter, which provided an accuracy of 1 micro-volt. The voltage is then increased such that a single bit in the digital input is changed, and the response is shown in Fig. 4 (b). This gives the minimum voltage step that can be achieved in the analog output voltage.

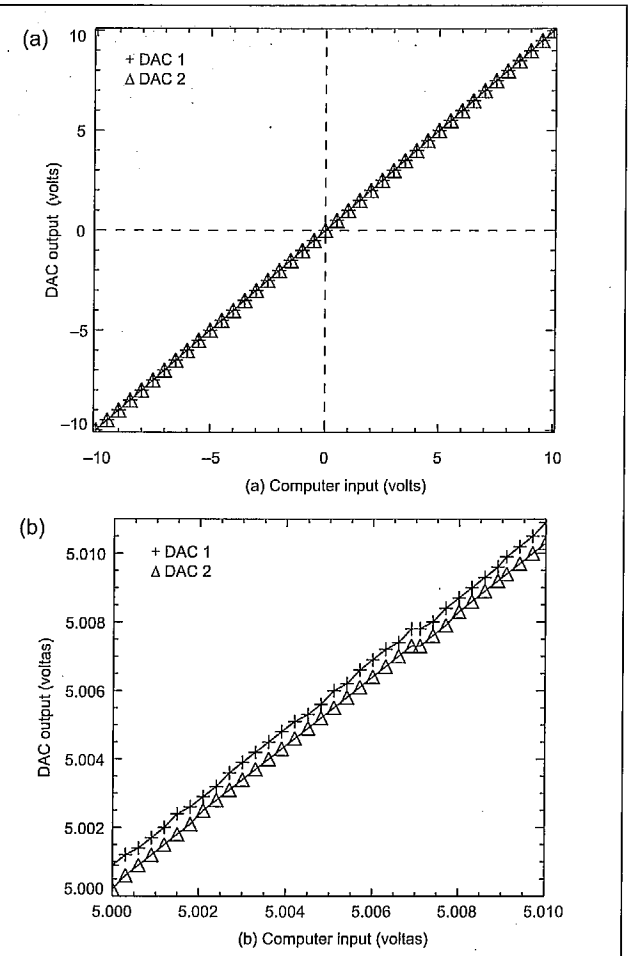


Fig. 4. Response of DAC1 and 2 for the computer digital input data. For the entire range of ±10volts. For 0.0003 volts step, this corresponds to single bit change in terms of digital input.

In the next step the analog output to the High voltage power supply units was applied. In order to measure the high voltage output from the modules with HP 3486A multi-meter, which has only a range of ±300 Vdc, a resistor network for voltage sampling was used and is shown in the inset of Fig. 5. Appropriate resistor values were computed so that the maximum current through the network (probe) does not exceed 80% of the maximum current from the module. Before using the probe for high voltage measurement, a calibration constant is computed by measuring the voltage across the sampler resistor (R1) for a set of voltages applied across the resistor network. Fig. 5 shows the calibration plot for the probe, a multiplication factor of 15.9326 is obtained for the probe. Fig. 6 shows the response of the input control voltage to the output high voltage. The measurement is carried out up to ±4600 Volts which exceeds the requirement for the tuning of the etalons. In one of the power supply module (HVPM1) we noticed a slight non-linearity beyond +4000 Volts, which is beyond the usable range for our needs.

Analog output voltages from both DAC1 and 2 show highly accurate linear behavior (with a linearity error better than ±1 LSB). We tested the DACs for the minimum voltage step possible. We conducted preliminary tests on the high voltage power supply modules and found that it works satisfactorily with the control circuit and software.

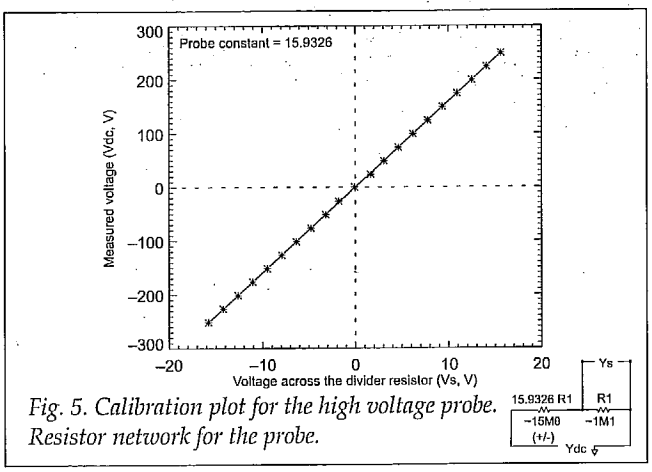


Fig. 5. Calibration plot for the high voltage probe. Resistor network for the probe.

We carried out some more tests on the HVPMs. For measuring the response time and stability, we used a different resistor network probe with 1000:1 voltage reduction.

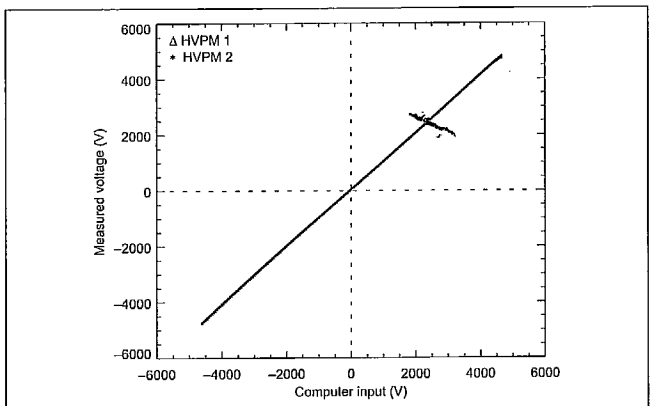


Fig. 6. Response of HVPM1 and 2 for the computerdigital input.

The stability of the HVPMs is ascertained by monitoring the high voltage outputs and the driving input DAC voltages for around two and a half hours. Fig. 7 shows the measured voltages from HVPM1 for the above mentioned period. A standard deviation (SD) of 850 mV and an end-to-end difference of 410 mV at +3000 V is observed for the above module. Small ripples in the DAC input and HV output voltages are due to the measurement noise and the above mentioned drift in the voltages are within this noise level. We repeated the same exercise for the second power supply and obtained similar results.

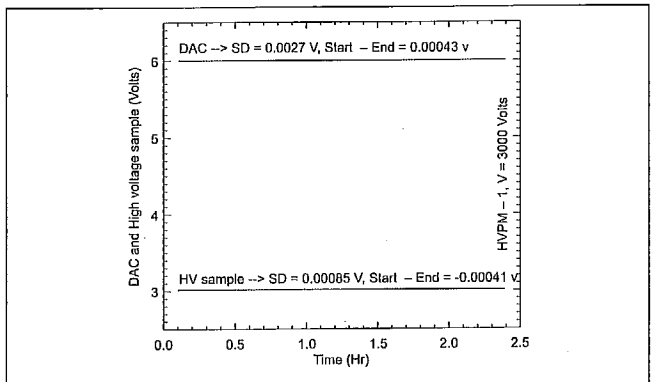


Fig. 7. Stability of one of the power supply for around 2.5 Hrs. We obtained similar results for the second power supply also.

Fig. 8 a and b show the rise and fall time for one of the power supply. There is a small ringing when the voltage is changed rapidly (with in 0.2 msec) and the settling time of 30 - 40 ms exceeds the rise and fall times (around 4 – 5 ms). As the measurement noise (found by analyzing the digitized outputs while the power supplies were switched off) was more than the expected ripple across the voltage divider, we could not accurately quantify it in the high voltage output. This will have very little effect on our measurements since the change in wavelength in the pass band due to the ripple is negligible (e.g. 450 milli-volts ripple shifts the pass band only by 0.2 mÅ)

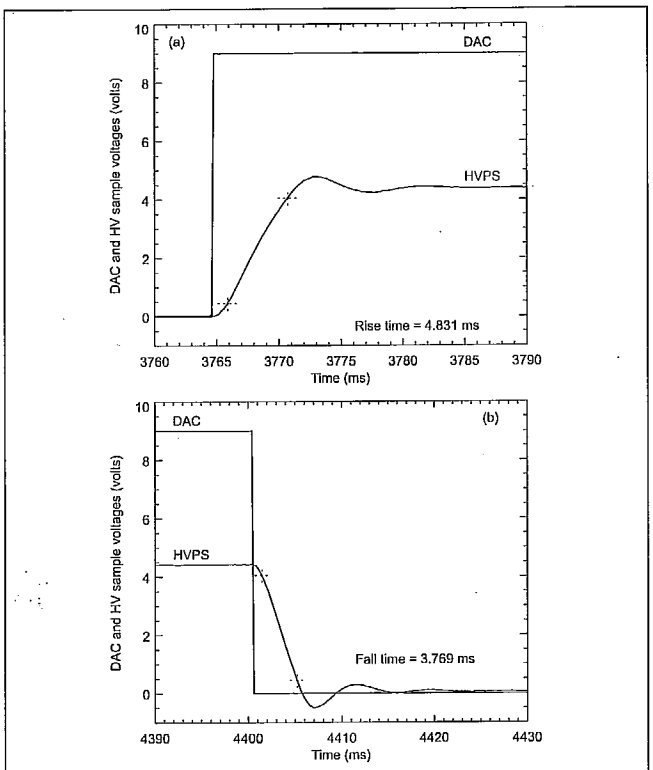


Fig. 8. Rise time (a) and fall time (b) for the HVPM1 modules. The black curve shows the DAC step voltage input and the red curve the high voltage sample. We found similar response from HVPM2.

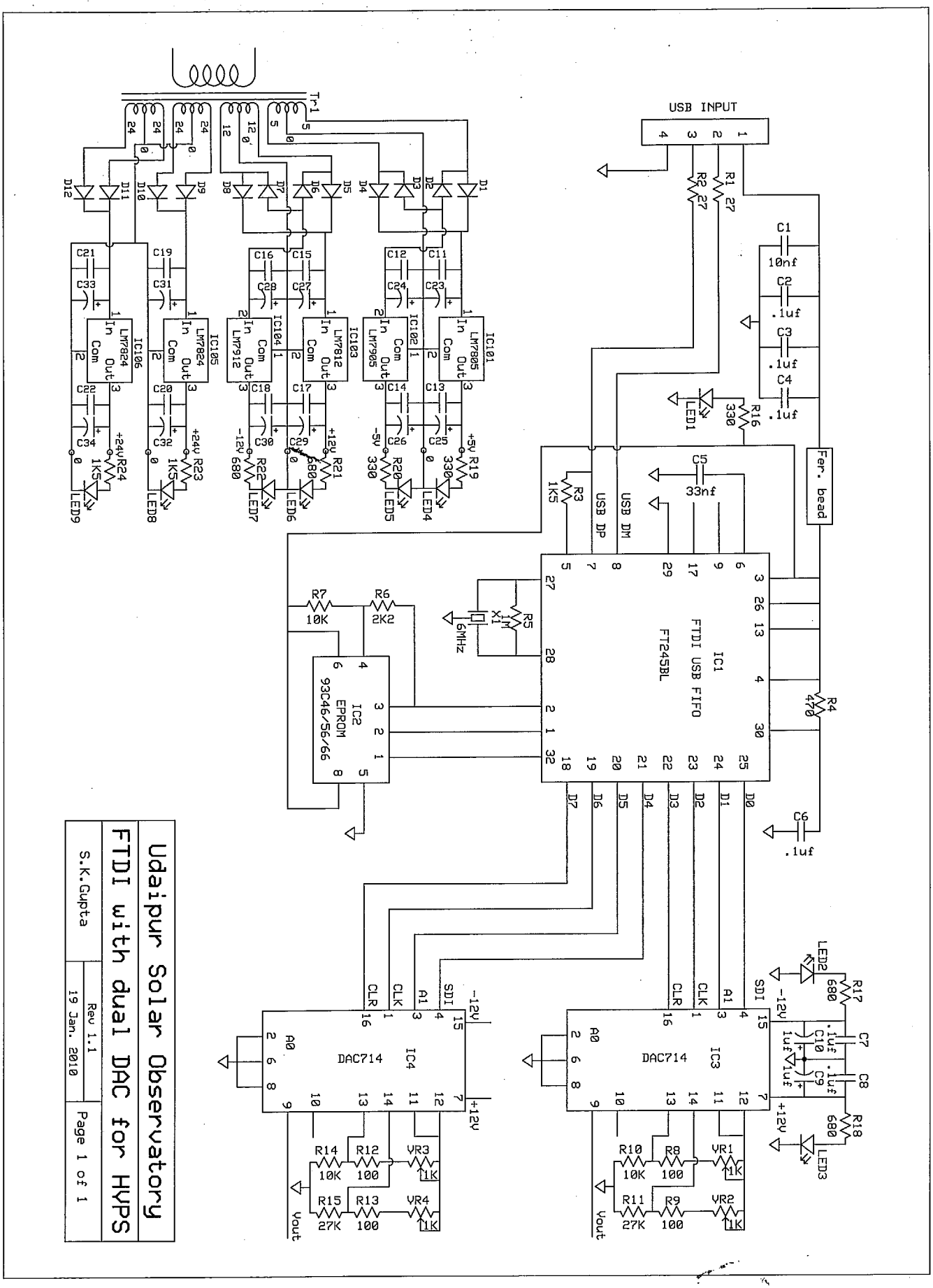
Conclusions

A high voltage power supply unit for tuning the lithium niobate Fabry-Perot etalon filter was made and preliminary tests were carried out. Two high voltage power supply modules procured from Applied Kilovolts are controlled to out-put the required high voltage dc. The control circuit is devised around an FTDI and a 16-bits DAC, which in turn can be accessed and controlled from a PC-USB port. The power supply unit can provide up to ±5000 volts independently to two etalons with a voltage resolution of less than 1 volts. The initial measurements show small ripple and very low drift (both less than 500 mV) in the output voltage, a rise time of around 40 msec, which is sufficient enough to use the instrument for the narrow-band imager in the MAST telescope.

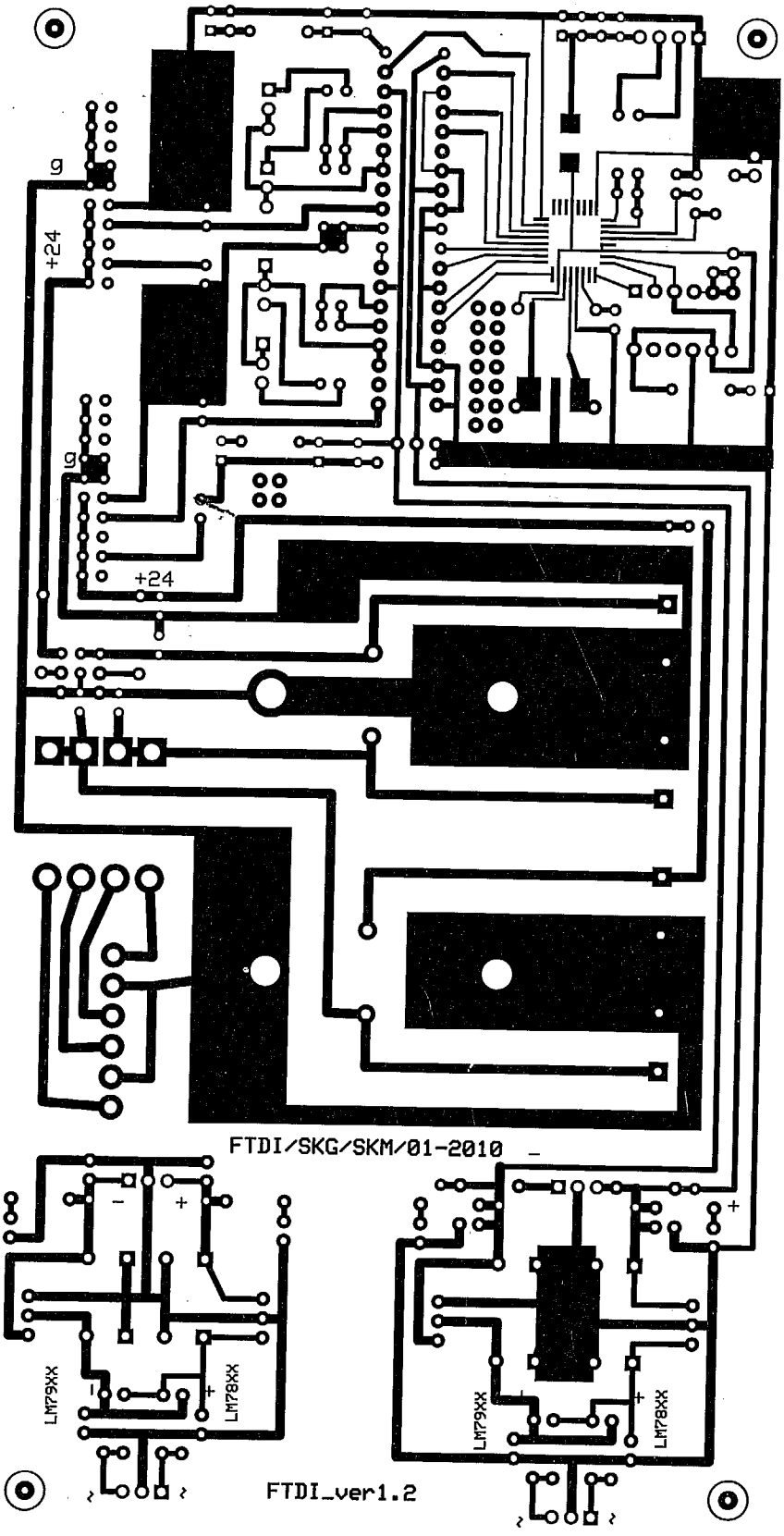
Acknowledgement

We acknowledge the contribution of Mr. Mukesh Sardava for all the mechanical fixture

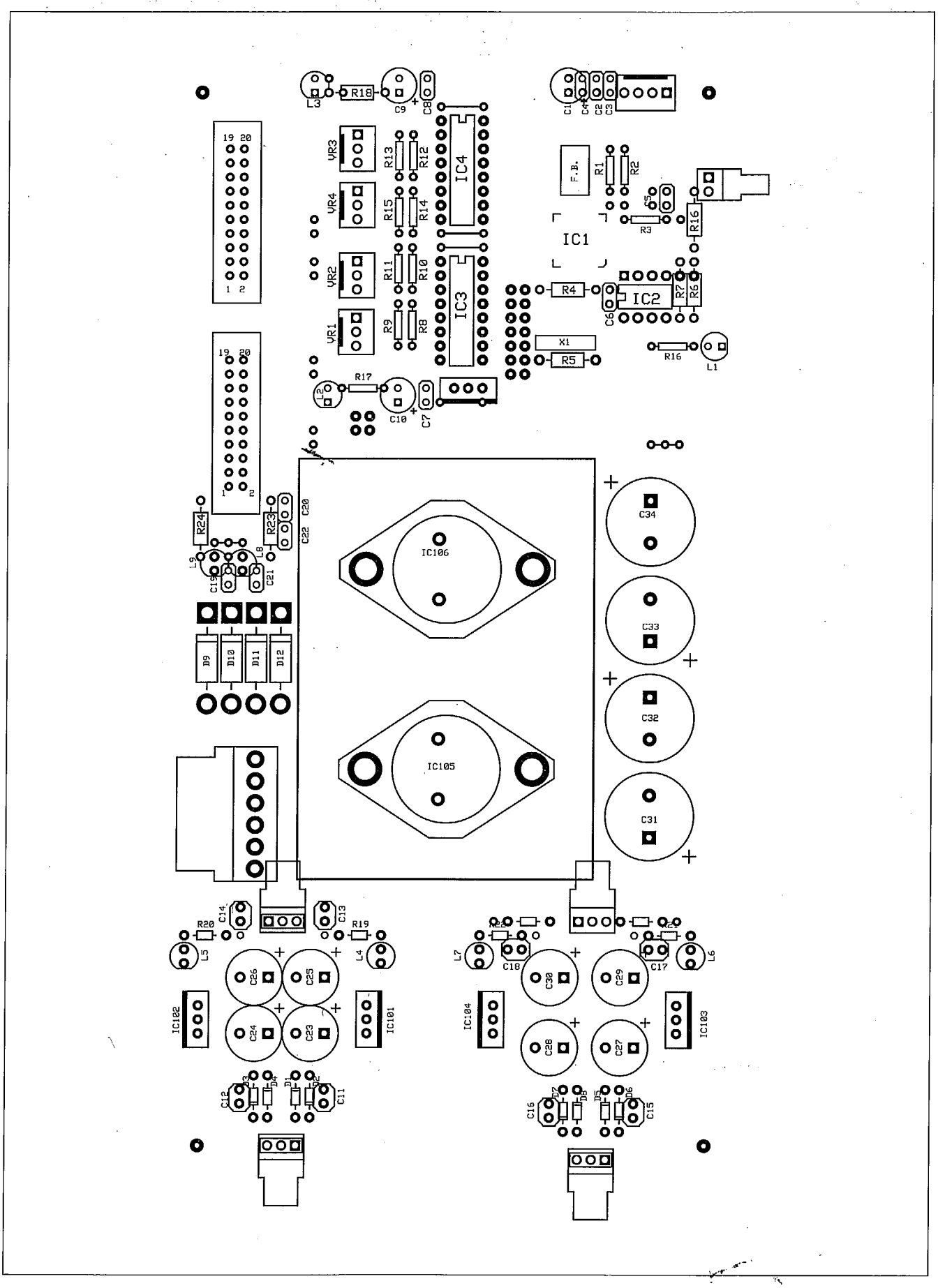
Appendix – 1a. Schematic of control circuit



Appendix – 1b. PCB layout, bottom layer



Appendix – 1c. PCB layout, silk screen



Appendix 2. Control Software

```
// Program for controlling the DAC for high voltage power supply.
// USB - FIFO device FT245BL is used in Bit-Bang mode,
//Shibu K. Mathew (shibu@prl.res.in), USO/PRL
#include <stdio.h>
#include <windows.h>
#include <time.h>
#include <winioctl.h>
#include "ftd2xx.h"
#include <stdlib.h>
#include <math.h>

// Clocking and DAC constants

#define amp_factor 500.0
#define t_rate 0.01
#define limit 5000.0
#define t1 1.e-8
#define t2 2.*1.e-8
#define t3 1.e-8
#define clrpw 2.*1.e-8

// Pin configurations
//First DAC

#define SDI0 1
#define A10 2
#define CLK0 4
#define CLR0 8

//Second DAC

#define SDI1 16
#define A11 32
#define CLK1 64
#define CLR1 128

void delay(double);
void initialize(int flag0, int flag1);
void stop(void);
void clock_the_data(float inp0 ,float inp1, int flag0 , int flag1 );
FT_HANDLE ftHandle;
FT_STATUS ftStatus;

int main (int argc, char *argv[])
{
unsigned char* DD;
unsigned char* DD0;
unsigned char* DD1;
int flag0=1,flag1=1;

unsigned int res,i,j;
float digi_in,rem,sum,inp0,inp1;

// Things for delay function

float remin;
LARGE_INTEGER timestart,timeend,systemfrequency,tts,tte;
double millisecondsleft,msl;

// End of delay function variables char des[15]="MAST-FP-HV-PS";
//Description written to the EPROM connected to FT245 chip
UCHAR Mask = 255;
 UCHAR Mode = 1;
 ftStatus= FT_OpenEx (des, FT_OPEN_BY_DESCRIPTION, &
 ftHandle);
 if (ftStatus != FT_OK)
 {
 printf("No device found --> %s successfully\n",des);
 }
 ftStatus = FT_SetBaudRate(ftHandle, 115200);
 if (ftStatus != FT_OK)
 {
 printf("Problem with baud rate :)\n");
 }
 ftStatus = FT_SetDataCharacteristics(ftHandle, FT_BITS_8,
 FT_STOP_BITS_1, FT_PARITY_NONE);
 ftStatus = FT_SetBitMode(ftHandle, Mask, Mode);

 if (argc != 4)
 {
                        inp0=0.0;
                        inp1=0.0;
 }
                        if (argc >= 4)
                        {
                        sscanf(argv[1],"%f",&inp0);
                        sscanf(argv[2],"%f",&inp1);
                        sscanf(argv[3],"%d",&flag0);
                        sscanf(argv[4],"%d",&flag1);
                        }
 QueryPerformanceFrequency(&systemfrequency);
 if ((inp0 < -1.*limit)||(inp1 < -1.*limit))
 {
 printf("The voltage is too high for the FP\n");
                        inp0=0.0;
                        inp1=0.0;
                        exit(1);
 }
 if ((inp0 > limit)||(inp1 > limit))
 {
 printf("The voltage is too high for the FP\n");
                        inp0=0.0;
                        inp1=0.0;
                        exit(1);
 }
 //initialize(flag0,flag1);
 clock_the_data(inp0,inp1,flag0,flag1);
 delay(t_rate);
 return 0;
 }
 // Delay Functions starts here. After having trouble with clock()
 //function (in most of the computers this function gives only 10
 // msec time resolution), the delay function is implemented using
 // high-resolution performance counter, type declarations are in
 // windows.h, this function uses the system's high resolution
 // counter as the time base --> Shibu, 13.06.2004. This counter
 // is system dependent, but I found this function gives time
 // accuracy better than one microsec

 void delay(double seconds)
 {
 LARGE_INTEGER timestart,timeend,systemfrequency;
 double millisecondsleft;
 QueryPerformanceFrequency(&systemfrequency);
```

```
QueryPerformanceCounter(&timestart);
QueryPerformanceCounter(&timeend);
__int64 timetaken= (timeend.QuadPart-timestart.QuadPart);
millisecondsleft=(timetaken)*1000./systemfrequency.QuadPart;
while (millisecondsleft < (double) seconds*1e3 )
{
QueryPerformanceCounter(&timeend);
__int64 timetaken= (timeend.QuadPart-timestart.QuadPart);
millisecondsleft=(timetaken)*1000./systemfrequency.QuadPart;
}
}
// end of Delay

// Begin initialization
void initialize(int flag0, int flag1)
{
DWORD BytesWritten;
UCHAR BTW=0;
BTW=flag0*CLK0*1+flag0*A10*1+flag0*CLR0*1+flag1*CLK
1*1+flag1*A11*1+flag1*CLR1*1;
ftStatus = FT_Write (ftHandle, &BTW, sizeof(BTW), &
BytesWritten);
delay(t2*2.0);
BTW=flag0*CLK0*1+flag0*A10*1+flag0*CLR0*0+flag1*CLK
1*1+flag1*A11*1+flag1*CLR1*0;
ftStatus = FT_Write (ftHandle, &BTW, sizeof (BTW), &
BytesWritten);
delay(clrpw);

BTW=flag0*CLK0*1+flag0*A10*1+flag0*CLR0*1+flag1*CLK
1*1+flag1*A11*1+flag1*CLR1*1;
ftStatus = FT_Write (ftHandle, &BTW, sizeof (BTW), &
BytesWritten);
delay(t2);
}

// End of initialization
// Stop function

void stop(void)
{
DWORD BytesWritten;
char des[15]="MAST-FP-HV-PS";
UCHAR BTW=0;
BTW=0;      // All to zero
ftStatus = FT_Write (ftHandle, &BTW, sizeof (BTW), &
BytesWritten);
FT_Close(ftHandle);
}
// End of initialization
// Begin clocking
void clock_the_data(float inp0, float inp1, int flag0, int flag1)
{
int i,kk;
DWORD BytesWritten;
UCHAR BTW=0;
unsigned char D0[16];
unsigned char D1[16];
float dac_vol0=inp0/amp_factor;
float dac_vol1=inp1/amp_factor;
int counter=0,j;
float con=32767./10.0,res,rem,sum,digi_in;

// For DAC1
// For positive voltage
digi_in=dac_vol0*con;
if (dac_vol0 >= 0.)
{for (i=0;i<=15;i++)
{res=pow(2,15-i);
rem=digi_in-float(res);
if (rem < 0)
D0[15-i]=0;
else
{digi_in=rem;
D0[15-i]=1;}}}

// For negative voltage

if (dac_vol0 < 0.)
{digi_in=-1.*digi_in;
for (i=0;i<=15;i++)
{res=pow(2,15-i);
rem=digi_in-float(res);
if (rem < 0)
D0[15-i]=0;
else
{digi_in=rem;
D0[15-i]=1;}}

for (i=0;i<=15;i++)
{if (D0[i] <= 0)
D0[i]=1;
else
D0[i]=0;}
sum=0.0;

for (i=0;i<=15;i++)
{sum=sum+D0[i]*pow(2,i);}
sum=sum+1.0;
for (i=0;i<=15;i++)
{res=pow(2,15-i);
rem=sum-float(res);
if (rem < 0)
D0[15-i]=0;
else
{sum=rem;
D0[15-i]=1;}}}

// For DAC2
// For positive voltage
digi_in=dac_vol1*con;
if (dac_vol1 >= 0.){
for (i=0;i<=15;i++)
{res=pow(2,15-i);
rem=digi_in-float(res);
if (rem < 0)
D1[15-i]=0;
else
{digi_in=rem;
D1[15-i]=1;}}}

// For negative voltage
if (dac_vol1 < 0.)
{digi_in=-1.*digi_in;
for (i=0;i<=15;i++)
```

```
{res=pow(2,15-i);
rem=digi_in-float(res);
if (rem < 0)
D1[15-i]=0;
else
{digi_in=rem;
D1[15-i]=1;}}

for (i=0;i<=15;i++)
{if (D1[i] <= 0)
D1[i]=1;
else
D1[i]=0;}
sum=0.0;

for (i=0;i<=15;i++)
{sum=sum+D1[i]*pow(2,i);}
sum=sum+1.0;
for (i=0;i<=15;i++)
{res=pow(2,15-i);
rem=sum-float(res);
if (rem < 0)
D1[15-i]=0;
else
{sum=rem;
D1[15-i]=1;}}}
for (kk=0;kk<=15;kk++)
{
BTW=SDI0*D0[15-kk]+CLK0*1+CLR0*1+A10*1+SDI1*D1[15-kk]+CLK1*1+CLR1*1+A11*1;
ftStatus = FT_Write (ftHandle, & BTW, sizeof (BTW), & BytesWritten);
delay(t1);
BTW=SDI0*D0[15-kk]+CLK0*0+CLR0*1+A10*1+SDI1*D1[15-kk]+CLK1*0+CLR1*1+A11*1;
ftStatus = FT_Write (ftHandle, &BTW, sizeof (BTW), & BytesWritten);
delay(t2);
BTW=SDI0*D0[15-kk]+CLK0*1+CLR0*1+A10*1+SDI1*D1[15-kk]+CLK1*1+CLR1*1+A11*1;
ftStatus = FT_Write (ftHandle, & BTW, sizeof (BTW), & BytesWritten);
delay(t3);
}
delay(t1);
BTW=SDI0*D0[0]+CLK0*0+CLR0*1+A10*1+SDI1*D1[0]+CLK1*0+CLR1*1+A11*1;
ftStatus = FT_Write (ftHandle, & BTW, sizeof (BTW), & BytesWritten);
delay(t1);
BTW=SDI0*D0[0]+CLK0*0+CLR0*1+flag0*A10+SDI1*D1[0]+CLK1*0+CLR1*1+flag1*A11;
ftStatus = FT_Write(ftHandle,&BTW,sizeof(BTW),&BytesWritten);
delay(t1);
BTW=SDI0*D0[0]+CLK0*1+CLR0*1+flag0*A10+SDI1*D1[0]+CLK1*1+CLR1*1+flag1*A11;
ftStatus = FT_Write (ftHandle, & BTW, sizeof (BTW), & BytesWritten);
delay(t1);
BTW=SDI0*D0[0]+CLK0*1+CLR0*1+A10*1+SDI1*D1[0]+CLK1*1+CLR1*1+A11*1;
ftStatus = FT_Write (ftHandle, & BTW, sizeof (BTW), & BytesWritten);
delay(t1);
}
// end of clocking data
```

Appendix 3. Component list

Semiconductors:		
IC1	IC FT245BL	1
IC2,	IC 93C46	1
IC3,IC4	IC DAC714	2
IC101	IC LM7805	1
IC102	IC LM7905	1
IC103	IC LM7812	1
IC104	IC LM7912	1
IC105 & 106	IC LM7824K	2
X1	Ceramic Resonator 6MHz	1
FB	Ferrite Bead	1
D1-D8	Diode IN4007	8
D9-D12	Diode IN5408	4
Capacitors:		
C1	.01 mfd	1
C2-C4,C6-C8,C11-C22	0.1 mfd/50V	18
C5	.033 mfd	1
C9,C10	1 mfd/16V	2
C23-C30	2200 mfd/25V	8
C31-C34	2200 mfd/50V	4
VR1-VR4	Trimpot 1 K	4
Resistances:		
R1,R2	27 W	2
R3	1K5W	1
R4	470W	1
R5	1MW	1
R6	2K2W	1
R7,R10,R14	10KW	3
R8,R9,R12,R13	100W	4
R11,R15	27KW	2
R16,R19,R20	330W	3
R17,R18,R21,R22	680W	4
R23,R24	1K5W	5
Misc:		
IC DIP socket	8 pin	1
IC DIP socket	16 pin	2
Ribbon connector	20 pin	2
connector(0.156" pitch)	6 pin	1
connector(0.1" pitch)	3 pin	5
connector(0.1" pitch)	5 pin	1
Heat sink	TO-3	2
Heat sink	TO-220	4
Mica washer	TO-3	2
Mica washer	TO-220	4
LED	RED/GREEN	9