Appendix A: Raman Feasibility Calculations and Comparison with ATR

The two techniques of Raman spectroscopy and attenuated total reflection (ATR) spectroscopy are compared here for their suitability to make optical fibre remoted measurements of compounds dissolved in water. Toluene and chloroform are chosen as arbitrary but typical target analytes.

Fibre-Remoted Raman Detection

The detection of a Raman spectrum involves the detection of low levels of light, and efficient transmission and detection of the light is essential. In detecting trace quantities of compounds in water it is necessary to resolve the target spectrum from the broad features resulting from hydrogen bonding between the water molecules.

Probe considerations

The diameter D of the optical fibres used to form the entrance slit (and probe) determine the resolution of the spectrograph (and the amount of light collected). The unwanted Raman scattered light from the water will be collected as efficiently as that of the target analyte, and the water Raman spectrum is assumed here to be homogenous, constant and featureless.

When the resolution of the spectrometer is equal to the width of the lines of the target analyte (with fibres of diameter D_l), then increasing the diameter of the probe fibres increases the amount of target Raman light falling on to a tall thin detector in proportion to D. The collected background Raman light falling onto such a pixel increases with the square of the fibre width. Assuming that the shot noise in the background and target Raman light is the dominant source of noise in the measurement, the signal to noise ratio becomes essentially constant for fibre diameters greater than 8 times the limiting diameter D_l. For the case of the spectrometer constructed for this project this corresponds to fibres of diameter larger than 50 µm.

Target analyte and water scattering cross sections

Normalised scattering cross sections are tabulated below. The normalisation corrects the ν^4 dependancy of the intensity of Raman scattering, so that cross sections measured at one excitation wavelength may be compared with measurements made at another.

Table A.1

Compound	Frequency / cm^{-1}	Normalised scattering cross section, $d\sigma_j/d\Omega \cdot (\nu_0 - \nu_j)^{-4}$ / 10^{-48} $cm^6 \cdot sr^{-1}$	Measurement Wavelength / nm	Scattering cross section, σ, at 676 nm / $10^{-30} cm^2 \cdot sr^{-1}$	Source of data
Water	3,400	96	488	1.6	[1]
Water	800	5	1,064	0.2	See footnote[3]
Toluene	1,002	111.5	694	4.0	[2]
Chloroform	666	64	515	2.5	[2]

Expected signal intensities

In these calculations the tabulated differential cross sections are assumed to be constant over the solid angle viewed by the probe fibres. It is assumed that the most significant noise contribution is from the shot noise of the Raman light generated in the water, *ie* the detector noise is neglected. The experimental system under consideration is a *100 mm long perfectly guiding liquid-cored waveguide cell*, with the aqueous analyte as the guiding medium. The core diameter is 500 µm, and light is delivered to and collected from the cell by a close packed bundle of 19 125/85 µm (outer diameter/core diameter) step index optical fibres, 1 fibre used for light delivery, 18 for collection. Simple geometry then suggests that 52% of the light scattered back towards the probe, within the NA of the collection fibres, will be collected by the probe. This does not consider the small region close to the probe where less than 52% of the light is collected, as it is outside the overlap volume between the delivery and collection fibres.

A fibre numerical aperture (NA) of 0.16 is assumed, matched to the acceptance NA of the spectrograph. The numerical aperture is related to the acceptance solid angle Ω by the formula

$$\Omega = 2\pi(1 - \cos(\sin^{-1}(NA))) \qquad \textbf{(A.1)}$$

Neglecting multiple scattering events and absorption of the incident beam, the power scattered back toward the probe P_s from a beam of initial power P_0 is given by

$$P_s = P_0(1 - \exp(-\sigma n \Omega x)) \qquad \textbf{(A.2)}$$

where σ is the scattering cross section as given in the fifth column of table A.1 and x is the length of the cell. While σ is small, this may be written to a good approximation as

$$P_s = P_0 \sigma n \Omega x \qquad \textbf{(A.3)}$$

[3]Estimate based on Weber [1], and experimental data.

The power collected by the probe due to scattering from water, and 1 part per million chloroform and toluene, calculated from equation A.3 are tabulated below, when P_0=100 mW, taking into account the 52% collection efficiency of the fibre probe, and assuming a detector quantum efficiency of 0.5. Equation A.3 gives the amount of energy scattered into a single Gaussian profiled line. The quoted power in the water background (table A.2) is 6 times that given directly by equation A.3 because the water background is constant, and so considered here as a series of 6 overlapping lines.

Table A.2

Scatterer	Number of scattered photons collected per second	Associated shot noise (photons)
Water	2.8E+9	5.3E+4
1 ppm toluene	1.0E+4	100
1 ppm chloroform	6E+3	77

Measurement time required for signal to noise ratio of 3

As can be seen in table A.2, the shot noise in the water background is the dominant noise source (much larger than the shot noise in the analyte signal). The signal to noise ratios (the number of detected photons scattered by the analyte divided by the shot noise of the water scattered Raman light) for one second integration times SN_1 are tabulated in table A.3. The number of one second measurements necessary to achieve a signal to noise ratio of 3 (an arbitrary value chosen to represent unambiguous detection of a line) is also tabulated.

Table A.3

Scatterer	Signal to noise ratio (one second integration time), SN_1	Number of measurements necessary to raise signal to noise ratio to 3 (measurement time in seconds)
1 ppm toluene	0.2	225
1 ppm chloroform	0.1	900

The initial signal to noise ratio, SN_1 is simply taken as the number of photons scattered per second divided by the shot noise of the water background. The number of measurements necessary to increase the signal to noise ratio to three (*ie* the length of the measurement time τ in seconds) is found from equation A.4.

$$\tau = \left(\frac{3}{SN_1} \right)^2 \tag{A.4}$$

Attenuated Total Reflection/Evanescent Field Absorption

The technique of attenuated total reflection (ATR) spectroscopy is a standard tool for the study of the IR (from 2 to 10 µm) absorption spectra of, among other things, highly absorbing liquids.[3] Recently the technique has been extended through the use of optical fibres, where it is more usually referred to as evanescent field absorption (EFA) spectroscopy.[4]

Brief overview of the technique

When light incident from a medium of refractive index n_1 to one of lower index n_2 is reflected at an interface by total internal reflection[5] some of the energy of the reflected wave penetrates the lower index medium as an evanescent field. The evanescent filed is not a travelling wave, and does not propagate energy away from the boundary. The depth of the penetration is related to the wavelength of the light, its angle of incidence at the interface and the refractive indices of the two media by equation A.5. As α approaches the critical angle d_p can become very large with a proportionally larger fraction of energy present in the evanescent wave. If the lower index medium has any absorption bands, then energy is removed from the reflected wave at those wavelengths.

$$d_p = \frac{\lambda}{2\pi n_2(\sin^2(\alpha) - (n_2/n_1)^2)} \tag{A.5}$$

A typical ATR attachment (Infra red element (IRE)) for an IR spectrometer resembles a trapezoidal prism, with a (partially) collimated beam of light propagating obliquely down the long axis, undergoing total internal reflection (TIR) at the parallel top and bottom surfaces of the IRE. The light that passes through the IRE is analysed by a spectrometer, and the resulting spectrum contains the same information (in a slightly distorted form) as a conventional thin cell IR absorption spectrum.

The evanescent field of a length of IR transmitting fibre (with a suitably thin cladding) may also be used to perform a similar measurement, although the measurement is usually referred to as EFA spectroscopy. If the fibre used as the sensing region is multimoded, then the resulting spectrum will be further distorted from the conventional transmission spectroscopy measurement due to the uneven removal of power from each guided mode of the fibre.

To increase the sensitivity, and remove potential masking absorptions, EFA sensors for contaminants in water often consist of polymer coated tapers. The taper increases the evanescent field penetration into the polymer coating, which selectively absorbs (preconcentrates) many organic compounds, while excluding water.

Expected sensitivity (literature review)

PVC coated tapered chalcogenide fibres have demonstrated detection limits (for 12 minute measurement times) between 0.11% (chloroform) and 0.006% (nitrobenzene).[6] The diffusion into the polymer coating was reversible, reaching equilibrium in approximately 10 minutes. When the polymer coating was removed from the taper no signal was observed for 0.15% benzene in water, while the detection limit with the polymer coating was calculated to be 0.02%.

Polymer coated silver halide tapers, coupled with an FT-IR spectrometer, have also demonstrated mg/L detection limits of chlorinated hydrocarbons in water[7]. A similar sensing element used with a wavelength modulated light source (tunable diode laser) and “2f” phase sensitive detection has extended this sensitivity down to 50 µg/L of tetrachloroethylene.

Conclusions

The greatest problem faced in EFA analysis is the relatively poor transmission of optical fibres in the IR region of the spectrum. Typically silver halide fibres typically display over 1 dB/m loss, and are too fragile for use over any distance, limiting their application to short (less than a metre) sections as intrinsic fibre sensors. It may be possible to use fluoride glass fibre to optically remote an EFA measurement, but transmission is only over the mid-IR (2-5 µm), and the most useful vibrational information is usually found below 5 µm. Even so, there are other disadvantages in employing the EFA technique.

There is no possibility of making a non-contact measurement by ATR/EFA, therefore the technique is particularly susceptible to fouling. ATR/EWS is an absorption technique, and small absorptions on a large background of transmitted light must be detected. The shot noise of the transmitted background is much more likely to obscure a line than is the scattered background light of a Raman measurement.

The technology required to make optical-fibre-remote Raman measurements in the visible and NIR is readily available, especially in robust and miniaturised form, and at a cost benefit due to the large requirement from the telecommunications industry. Diode lasers capable of coupling ever increasing amounts of monochromatic light in to an optical fibre become available monthly, with powers in excess of a watt currently available. Raman offers the possibility of making non-contact measurements, so that fouling of probe optics is not a problem.

The numbers presented in this document show that Raman scattering can offer a similar level of sensitivity to EFA, but with greater potential to make the measurements optically remote from the necessary instrumentation. In this particular application it is the more promising technique for investigation. The work itself is valuable even if a prototype system capable of detecting parts per billion of a particular analyte is not built. The results will be applicable to other areas of application, and the cell arrangements and signal referencing techniques are valuable in themselves.

References to this appendix

1.Marshall, BR and Smith, CS; "Raman scattering and in ocean optical properties"; Applied Optics, 29, 1, 71. 1990.

2.Weber, A [Ed]; "Raman Spectroscopy of Gases and Liquids"; Springer-Verlag, Berlin, 1979.

3. Willis, HA, van der Maas, JH, Miller, RGJ [Eds]; "Laboratory Methods in Vibrational Spectroscopy third edition"; John Wiley and Sons, 1987.

4.Wolfbeis, OS; "Fiber Optic Chemical Sensors and Biosensors Volume 1"; CRC Press, Boca Raton, 1991.

5. Hecht, E; "Optics second edition"; Addison-Wesley Publishing Company, Reading MA, 1987.

6.Ertan-Lamontagne, MC, Lowry, SR, Seitz, WR and Tomellini, SA; "Polymer-coated, tapered cylindrical ATR elements for sensitive detection of organic solutes in water"; Applied Spectroscopy, 49, 8, 1170. 1995.

7.Krska, R, Taga, K and Kellner, R; Applied Spectroscopy, 47, 1484. 1993.

Appendix B: C Source Code for the CCD Interface

1. Software Overview

All operation of the CCD is controlled via the software window shown in figure B.1. This controls the CCD integration time set, the number of averages in each measurement, any system correction used (*eg* for various optical filters at the spectrograph input), the LabPC+ preamp gain, and any offset in the reference cell signal. Other comments to be stored with the data can also be entered from the front panel, as can the name of the file in which measured data is to be recorded.

The results of the most recent acquisition are displayed as a line graph on the front panel, unless acquisition is in progress, in which case the display is live, displaying the current intensity profile across the CCD. Each acquisition may comprise up to 25 individual readings. The user may step through each of these readings using the displayed spectrum control. By double clicking the control the program will step through each set automatically.

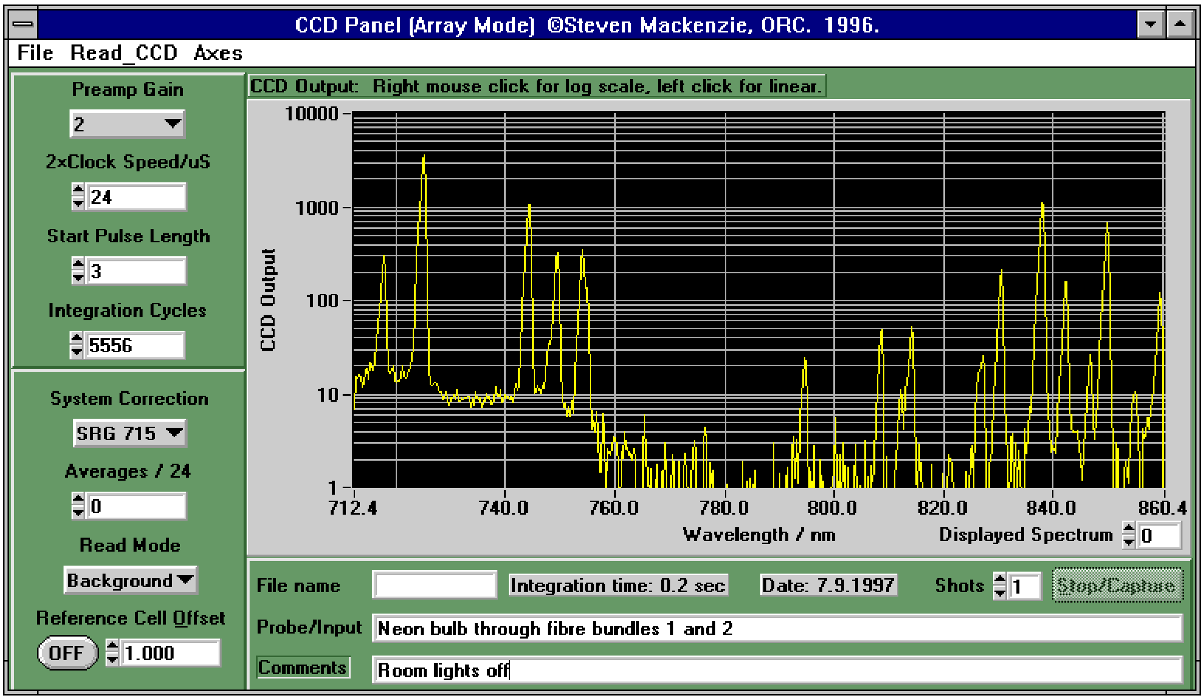


Figure B.1 The interface to the CCD Array readout program. Data is acquired and saved using the menu bars at the top of the screen. System parameters are entered directly into the boxes in the window. All parameters, the date, and any comments typed into the appropriate boxes are saved with the data. Here the intensity is shown plotted against a log scale to accentuate the lower peaks and the readout noise.

Data is filed by using the File data option from the File menu, and the intensity profiles for each of the recorded spectra is saved in columns as ASCII text, along with other information, as shown in figure B.2.

```
Hamamatsu C5809 CCD Data. Date: 27.7.1997
Read mode:    1
LabPC+ ADC gain:    1
Integration Time: 2.000160e-01 seconds
No. of averages:    0
System correction:    3
Probe/input arrangement: fibre 2, 4.5mL/L of 600mg/L PB -+4%, 14.9mW
Comments: 5×24 readings in background, laser = 134.2mA
     Wavelength/nm  Wavenumber shift   Normalised data
        860.660000       3209.005178      -3134.819149    -200555.657909    -179129.950854
        860.370400       3205.094235          1.083672          1.083672          5.727958
        860.080800       3201.180658         -0.308891         -0.308891          7.799513
        859.791200       3197.264445          0.142926          1.286336          7.003382
```

Figure B.2 The first 14 rows of a typical data file from the program ccdarray.exe. Three columns of data have been recorded in this example; each data set comprises 512 points, corresponding to the 512 CCD columns.

The first line of the data file is a line of text which identifies the source of the file, and the date on which it was recorded.

2. Software Source Code

This is the source code for the program CCDARRAY. It works in conjunction with a National Instruments LabPC+ data acquisition card, interface circuitry described in chapter 3, and the binary file ccdarray.uir, produced with the National Instruments CVI compiler. The ccdarray.uir file can be loaded into CVI, and contains the information used to draw the program window and menu bars. See chapter 3 for further details of the software and its operation.

```
/*
*** Program to control and read the Hamamatsu C5809 CCD array, via a National Instruments
*** LabPC+ card and the buffer, interface, and power supply box labelled “CCD Power Supply”.
*/

#include <userint.h>
#include <ansi_c.h>
#include <dataacq.h>
#include <analysis.h>
#include <formatio.h>
#include <utility.h>
#include "ccdarray.h"

#define NO_OF_COUNTS 512
#define NO_OF_POINTS 512
#define OFF 0
#define ON 1
#define NO 0
#define YES 1

/*
*** Several lines of code to pause the program, to study the state of certain variables,
*** can be included here.
*/
#define DEBUGGING NO

/* Macros to convert the CCD column number (an integer q) to wavelength in nm */
```

```
/* or frequency shift in 1/cm (float). This formula is from the wavelength  */
/* calibration of 22.1.96. Used in file_data and plot_output. */
#define DIODE_WAVELENGTH 674.4
#define MEAS_WAVELENGTH(q) (((q) * -0.2896) + 860.66)
#define FREQ_SHIFT(q) (1.0e7 * (1 / DIODE_WAVELENGTH - 1 / MEAS_WAVELENGTH(q)))
/* Defined constants corresponing to the read mode control. */
#define NORMAL 0
#define BACKGROUND_SUBTRACT 1
#define REFERENCE_CELL_SUBTRACT 2
/* Defined constants corresponding to the system correction control */
#define NONE 0
#define NO_FILTER 1
#define SRG695 2
#define SRG715 3
/* Defined constants corresponding to the x-axis control */
#define COLUMN_NO 0
#define WAVELENGTH 1
#define FREQUENCY_SHIFT 2
/* Defined constants corresponding to the y axis control */
#define LINEAR 0
#define LOG 1
/* Number of spectra stored (ccdarray.c verion of the program only) */
#define STORED_SPECTRA 25
#define TEMP_STORED_SPECTRA 32

unsigned long points;
unsigned int integration_cycles, mode, filter, correction = OFF;
float output_array[STORED_SPECTRA][NO_OF_POINTS], filter_data[NO_OF_POINTS],
        background[NO_OF_POINTS], running_average[NO_OF_POINTS];
int ccd_panel, x_choose_panel, y_choose_panel, ccd_menu, month, day, year, x_axis_style,
        y_axis_style;
short clock_speed, finished=0, pulse_length, gain;
unsigned short count_check, array_plot, averages=0, shots=1, spectrum,
        temp_read_array[TEMP_STORED_SPECTRA][NO_OF_POINTS];
double offset=1.0, y_max=4100.00, y_min=0.00;
char *date="Date: dd.mm.yyyy", *int_time="Integration time: 0.054   sec";


int graph(int panel, int control, int event, void *callbackData, int eventData1, int eventData2)
{
switch (event)
        {
        case EVENT_LEFT_CLICK:
        SetCtrlAttribute (ccd_panel, CCD_PANEL_GRAPH, ATTR_YMAP_MODE,
                                        VAL_LINEAR);
        SetAxisRange (ccd_panel, CCD_PANEL_GRAPH, VAL_NO_CHANGE, 0.0, 1.0,
                                VAL_MANUAL, y_min, y_max);
        break;

        case EVENT_RIGHT_CLICK:
        SetAxisRange (ccd_panel, CCD_PANEL_GRAPH, VAL_NO_CHANGE, 0.0, 1.0,
                                VAL_MANUAL, y_min, y_max);
        SetCtrlAttribute (ccd_panel, CCD_PANEL_GRAPH, ATTR_YMAP_MODE,
                                        VAL_LOG);
        break;
        }
return 0;
}

void quit(int menubar, int menuItem, void *callbackData, int panel)
{
        QuitUserInterface(0);
}

void print_screen(int menubar, int menuItem, void *callbackData, int panel)
{
        SetPrintAttribute (ATTR_ORIENTATION, VAL_LANDSCAPE); /* sets printer to landscape */
/*
*** The print panel dialogue box recommends the following lines to ensure the
*** printout fits on the page (which it doesn't if the lines are omitted).
*/
        SetPrintAttribute (ATTR_PAPER_WIDTH, VAL_INTEGRAL_SCALE);
        SetPrintAttribute (ATTR_PAPER_HEIGHT, VAL_USE_PRINTER_DEFAULT);
        PrintPanel (ccd_panel, "", 1, VAL_FULL_PANEL, 0);
}

int checkstop(void)
{
int clicked, panel;
GetUserEvent (0, &panel, &clicked); /*  Read User Event Queue.  */
 #if DEBUGGING
 /*
 *** This if statement is for debugging only. Allows the watch window to be
```

```
 *** read after the program locks.
 */
 if (points==(NO_OF_COUNTS - 1))
        Delay (0.5);
 #endif
if(clicked==CCD_PANEL_STOP && panel==ccd_panel)
        return 0;                /*  Return 0 if stop has been clicked. */
return 1;  /*  Return 1 otherwise. */
}

void disable_controls(void) /* Disables all controls except STOP. */
{
/*
*** Disable the individual panel controls
*/
SetInputMode (ccd_panel, CCD_PANEL_CLOCK_SPEED, 0);
SetInputMode (ccd_panel, CCD_PANEL_PULSE_LENGTH, 0);
SetInputMode (ccd_panel, CCD_PANEL_INTEGRATION_CYCLES, 0);
SetInputMode (ccd_panel, CCD_PANEL_GAIN, 0);
SetInputMode (ccd_panel, CCD_PANEL_MODE, 0);
SetInputMode (ccd_panel, CCD_PANEL_NO_AVERAGES, 0);
SetInputMode (ccd_panel, CCD_PANEL_CORRECTION, 0);
SetInputMode (ccd_panel, CCD_PANEL_OFFSET, 0);
SetInputMode (ccd_panel, CCD_PANEL_PROBE, 0);
SetInputMode (ccd_panel, CCD_PANEL_COMMENTS, 0);
SetInputMode (ccd_panel, CCD_PANEL_FILE_NAME, 0);
SetInputMode (ccd_panel, CCD_PANEL_SYSTEM_CORRECTION, 0);
SetInputMode (ccd_panel, CCD_PANEL_SHOTS, 0);
SetInputMode (ccd_panel, CCD_PANEL_DISPLAYED_SPECTRUM, 0);
SetInputMode (ccd_panel, CCD_PANEL_STOP, 1);
/*
*** Disable the menu bar. The -1 indicates everything on ccd_menu, the 0
*** indicates that input is disabled.
*/
SetInputMode (ccd_menu, -1, 0);
}

void read_controls(void)  /* Reads clock_speed, etc, from the front panel. */
{
GetCtrlVal (ccd_panel, CCD_PANEL_CLOCK_SPEED, &clock_speed);
GetCtrlVal (ccd_panel, CCD_PANEL_PULSE_LENGTH, &pulse_length);
GetCtrlVal (ccd_panel, CCD_PANEL_INTEGRATION_CYCLES, &integration_cycles);
GetCtrlVal (ccd_panel, CCD_PANEL_GAIN, &gain);
GetCtrlVal (ccd_panel, CCD_PANEL_MODE, &mode);
GetCtrlVal (ccd_panel, CCD_PANEL_CORRECTION, &correction);
GetCtrlVal (ccd_panel, CCD_PANEL_OFFSET, &offset);
GetCtrlVal (ccd_panel, CCD_PANEL_SHOTS, &shots);
}

void enable_controls(void) /* Enables all controls except STOP. */
{
SetInputMode (ccd_panel, CCD_PANEL_CLOCK_SPEED, 1);
SetInputMode (ccd_panel, CCD_PANEL_PULSE_LENGTH, 1);
SetInputMode (ccd_panel, CCD_PANEL_INTEGRATION_CYCLES, 1);
SetInputMode (ccd_panel, CCD_PANEL_GAIN, 1);
SetInputMode (ccd_panel, CCD_PANEL_MODE, 1);
SetInputMode (ccd_panel, CCD_PANEL_NO_AVERAGES, 1);
SetInputMode (ccd_panel, CCD_PANEL_CORRECTION, 1);
SetInputMode (ccd_panel, CCD_PANEL_OFFSET, 1);
SetInputMode (ccd_panel, CCD_PANEL_PROBE, 1);
SetInputMode (ccd_panel, CCD_PANEL_COMMENTS, 1);
SetInputMode (ccd_panel, CCD_PANEL_FILE_NAME, 1);
SetInputMode (ccd_panel, CCD_PANEL_SYSTEM_CORRECTION, 1);
SetInputMode (ccd_panel, CCD_PANEL_SHOTS, 1);
SetInputMode (ccd_panel, CCD_PANEL_DISPLAYED_SPECTRUM, 1);
SetInputMode (ccd_panel, CCD_PANEL_STOP, 0);
SetInputMode (ccd_menu, -1, 1);/* The -1 implies all the menu, the 1 implies enabled. */
}

int mode_av_txt (int panel, int control, int event,
                void *callbackData, int eventData1, int eventData2)
/*
*** This function ensures that the text on the averages label is consistant with the
*** read mode.
*/
{
        unsigned short temp_mode;
        switch (event) {
                case EVENT_VAL_CHANGED:
                        GetCtrlVal (ccd_panel, CCD_PANEL_MODE, &temp_mode);
                        if(temp_mode==REFERENCE_CELL_SUBTRACT)
                                SetCtrlAttribute (ccd_panel, CCD_PANEL_NO_AVERAGES,ATTR_LABEL_TEXT,
                                                                    "No ofAverages / 14");
```

```
                        else
                                SetCtrlAttribute (ccd_panel, CCD_PANEL_NO_AVERAGES,ATTR_LABEL_TEXT,
                                                                        "Averages / 24");
                        break;
        }
        return 0;
}

void store_background(int menubar, int menuItem, void *callbackData, int panel)
{
register unsigned short p;
for(p=0;p<NO_OF_POINTS;p++)
         background[p] = output_array[shots - 1][p];
}

void display_integration_time()
{
sprintf(int_time, "Integration time: %.4g sec", 5e-7*clock_speed*pulse_length*
                                                                integration_cycles);
SetCtrlVal (ccd_panel, CCD_PANEL_INTEGRATION_TIME_DISP, int_time);
}

void set_axes()
{
SetAxisRange (ccd_panel, CCD_PANEL_GRAPH, VAL_NO_CHANGE, 0.0, 1.0,
                         VAL_MANUAL, y_min, y_max);
SetCtrlAttribute (ccd_panel, CCD_PANEL_GRAPH, ATTR_YMAP_MODE,
                                y_axis_style);
switch(x_axis_style)
        {
        case(COLUMN_NO):
        SetAxisRange (ccd_panel, CCD_PANEL_GRAPH, VAL_MANUAL, 0.0,
                                NO_OF_POINTS, VAL_NO_CHANGE, 0, 1);
        SetCtrlAttribute (ccd_panel, CCD_PANEL_GRAPH, ATTR_XNAME,
                                        "CCD Column Number");
        break;

        case(WAVELENGTH):
        SetAxisRange (ccd_panel, CCD_PANEL_GRAPH, VAL_AUTOSCALE, 710.0, 860.0,
                                VAL_NO_CHANGE, 0, 1);
        SetCtrlAttribute (ccd_panel, CCD_PANEL_GRAPH, ATTR_XNAME,
                                        "Wavelength / nm");
        break;

        case(FREQUENCY_SHIFT):
        SetAxisRange (ccd_panel, CCD_PANEL_GRAPH, VAL_AUTOSCALE, 700.0,
                                3500.0, VAL_NO_CHANGE, 0, 1);
        SetCtrlAttribute (ccd_panel, CCD_PANEL_GRAPH, ATTR_XNAME,
                                        "Frequency Shift . cm");
        break;
        }
}

void system_correction (void)
{
FILE *file_handle;
file_handle = fopen ("c:\\cvi\\ccd\\ccdaryfc.fdt", "r");
if (file_handle == NULL)
        {
        MessagePopup ("Read File Error",
                                "The filter data file ccdaryfc.fdt is missing.");
        SetCtrlVal (ccd_panel, CCD_PANEL_SYSTEM_CORRECTION, NONE);
        }
else
        {
        int q=0;
        GetCtrlVal (ccd_panel, CCD_PANEL_SYSTEM_CORRECTION, &filter);
        while(q<NO_OF_POINTS)
        {
        float temp_filter_data[4];
        if (fscanf(file_handle, "%f\t%f\t%f\t%f\n",
                &temp_filter_data[NONE],&temp_filter_data[NO_FILTER],
                &temp_filter_data[SRG695],&temp_filter_data[SRG715]) != 4)
                        {
                        MessagePopup ("Read File Error",
                                        "The filter data file ccdfcors.fdt is corrupt:\n\
Data is in the wrong format.");
                        SetCtrlVal (ccd_panel, CCD_PANEL_SYSTEM_CORRECTION, NONE);
                        break;
                        };
                filter_data[q]=temp_filter_data[filter];
                q++;
                }
```

```
	if(feof(file_handle))
		MessagePopup ("Read File Error",
					"The filter data file ccdfcors.fdt is corrupt:\n\
Not enough data in file.");
	fclose(file_handle);
	}
}

void plot_output(unsigned short int i)
{
float temp_array[NO_OF_POINTS], temp_x_array[NO_OF_POINTS];
unsigned short int q;

DeleteGraphPlot (ccd_panel, CCD_PANEL_GRAPH, array_plot, 0);
/*
***The ith array in output_array data is scaled before it is plotted
*/
for (q=0; q<NO_OF_POINTS; q++)
	temp_array[q] = output_array[i][q] * filter_data[q];
/*
***Plot the data, give the plot the handle `array_plot'
*/
switch(x_axis_style)
	{
	case(COLUMN_NO):
	array_plot = PlotY (ccd_panel, CCD_PANEL_GRAPH, temp_array,
				NO_OF_POINTS, VAL_FLOAT, VAL_THIN_LINE,
				VAL_EMPTY_SQUARE, VAL_SOLID, 1, VAL_YELLOW);
	break;

	case(WAVELENGTH):
	for(q=0; q<NO_OF_POINTS; q++)
		temp_x_array[q]=MEAS_WAVELENGTH(q+1);
	array_plot = PlotXY (ccd_panel, CCD_PANEL_GRAPH, temp_x_array,
					temp_array, NO_OF_POINTS, VAL_FLOAT, VAL_FLOAT,
					VAL_THIN_LINE, VAL_NO_POINT, VAL_SOLID, 1,
					VAL_YELLOW);
	break;

	case(FREQUENCY_SHIFT):
	for(q=0; q<NO_OF_POINTS; q++)
		temp_x_array[q]=FREQ_SHIFT(q+1);
	array_plot = PlotXY (ccd_panel, CCD_PANEL_GRAPH, temp_x_array,
					temp_array, NO_OF_POINTS, VAL_FLOAT, VAL_FLOAT,
					VAL_THIN_LINE, VAL_NO_POINT, VAL_SOLID, 1,
					VAL_YELLOW);
	break;
	}
}

void choose_x_axis (int menuBar, int menuItem, void *callbackData,
		int panel)
/*
***The choose_x_axis function displays a child panel with one control, a ring containing
***a list of options for the x axis. The control is linked to the call back function
***xstyle, which reads the control value and closes the panel when a commit event recorded.
*/
{
SetInputMode (ccd_menu, -1, 0);/* The -1 implies all the menu, the 0 implies disabled. */
x_choose_panel = LoadPanel (ccd_panel, "ccd.uir", CHOOSE_X);
SetCtrlVal (x_choose_panel, CHOOSE_X_XSTYLE, x_axis_style);
DisplayPanel (x_choose_panel);
}

int xstyle (int panel, int control, int event,
		void *callbackData, int eventData1, int eventData2)
{
	switch (event) {
		case EVENT_COMMIT:
			GetCtrlVal (x_choose_panel, CHOOSE_X_XSTYLE, &x_axis_style);
			DiscardPanel (x_choose_panel);
			SetInputMode (ccd_menu, -1, 1);
					/* The -1 implies all the menu, the 1 implies enabled. */
			set_axes();
			plot_output(shots - 1);
			break;
	}
	return 0;
}

void choose_y_axis (int menuBar, int menuItem, void *callbackData,
		int panel)
{
```

```
SetInputMode (ccd_menu, -1, 0);/* The -1 implies all the menu, the 0 implies disabled. */
y_choose_panel = LoadPanel (ccd_panel, "ccd.uir", CHOOSE_Y);
DisplayPanel (y_choose_panel);
SetCtrlVal (y_choose_panel, CHOOSE_Y_YSTYLE, y_axis_style);
SetCtrlVal (y_choose_panel, CHOOSE_Y_MAX, y_max);
SetCtrlVal (y_choose_panel, CHOOSE_Y_MIN, y_min);
}

int ystyle (int panel, int control, int event,
		void *callbackData, int eventData1, int eventData2)
{
	switch (event) {
		case EVENT_COMMIT:
			GetCtrlVal (y_choose_panel, CHOOSE_Y_YSTYLE, &y_axis_style);
			GetCtrlVal (y_choose_panel, CHOOSE_Y_MAX, &y_max);
			GetCtrlVal (y_choose_panel, CHOOSE_Y_MIN, &y_min);
/*
***Don't allow y_min = 0 if the y_style is log.
*/
			if((y_axis_style == 1) && (y_min == 0.00))
				SetCtrlVal (y_choose_panel, CHOOSE_Y_MIN, 1.00);
	}
	return 0;
}

int done (int panel, int control, int event,
		void *callbackData, int eventData1, int eventData2)
/*
*** This function closes the y-axis style chose box
*/
{
	ystyle (panel, control, event,
		callbackData, eventData1, eventData2);
	switch (event) {
		case EVENT_COMMIT:
			switch (y_axis_style) {
				case 0:
				y_axis_style = VAL_LINEAR;
				break;

				case 1:
				y_axis_style = VAL_LOG;
				break;
				}
			DiscardPanel (y_choose_panel);
			SetInputMode (ccd_menu, -1, 1);
					/* The -1 implies all the menu, the 1 implies enabled. */
			set_axes();
			plot_output(shots - 1);
			break;
	}
	return 0;
}

int sys_correction (int panel, int control, int event,
		void *callbackData, int eventData1, int eventData2)
{
	switch (event) {
		case EVENT_VAL_CHANGED:
			system_correction();
			plot_output(shots - 1);
			break;
	}
	return 0;
}

int scroll (int panel, int control, int event,
		void *callbackData, int eventData1, int eventData2)
/*
*** Call-back function to Displayed Spectrum control, scrolls
*** through aquired spectra on the graph display.
*/
{
	switch (event)
		{
		unsigned short int viewed_spectrum;
		case EVENT_VAL_CHANGED:
			GetCtrlVal (ccd_panel, CCD_PANEL_DISPLAYED_SPECTRUM, &viewed_spectrum);
			plot_output(viewed_spectrum);
			break;

		case EVENT_LEFT_DOUBLE_CLICK:
			disable_controls();
```

```
                        for(viewed_spectrum=0;viewed_spectrum<shots;viewed_spectrum++)
                                {
                                SetCtrlVal      (ccd_panel,      CCD_PANEL_DISPLAYED_SPECTRUM ,
viewed_spectrum);
                                plot_output(viewed_spectrum);
                                Delay(0.5);
                                if(!checkstop())
                                        {
                                        enable_controls();
                                        break;
                                        }
                                }
                        enable_controls();
                        break;
                }
        return 0;
}

void file_data(int menubar, int menuItem, void *callbackData, int panel)
{
        register short q=0,p=0;
        FILE *file_handle;
        char comments[256], probe[256], file_name[13], full_file_name[MAX_PATHNAME_LEN + 1];

/*
***Read file_name, probe type and comments from the front panel.
*/
        GetCtrlVal (ccd_panel, CCD_PANEL_FILE_NAME, file_name);
        GetCtrlVal (ccd_panel, CCD_PANEL_PROBE, probe);
        GetCtrlVal (ccd_panel, CCD_PANEL_COMMENTS, comments);
/*
***If the FileSelectPopup() function returns a +ve value (new or existing file selected)
***then open the file and write the data to it, else return.
*/
        switch(
        FileSelectPopup ("\\cvi\\ccd\\data", file_name, "*.dat",
                                        "Name of File to Save", VAL_OK_BUTTON, 0, 0, 1, 1,
                                        full_file_name))
                {

                /*
                *** Check that it is ok to overwrite the file, if it already exists.
                */
                case VAL_EXISTING_FILE_SELECTED:
                if(ConfirmPopup ("Warning!", "Overwrite existing file?")==NO)
                        break;
                /*
                *** If it is ok to overwrite, the program drops through to the next block.
                */

                case VAL_NEW_FILE_SELECTED:
                file_handle = fopen (full_file_name, "w");
                /*
                ***Write the comments and information at the top of the file. If the
                ***read mode is anything other than BACKGROUND_SUBTRACT, averages is muliplied
                ***by 24 (see average_24() function).
                */
                fprintf(file_handle,"Hamamatsu C5809 CCD Data. %s\n\
Read mode:%5d\nLabPC+ ADC gain:%5d\nIntegration Time: %e seconds\nNo. of averages\
:%5d\nSystem correction:%5d\nProbe/input arrangement: %s\nComments: %s\n\
%18s%18s%18s\n",date, mode,gain,((1.0/2e6)*clock_speed * pulse_length * integration_cycles),
(mode==REFERENCE_CELL_SUBTRACT)?averages:(averages*24), filter, probe, comments,
"Wavelength/nm", "Wavenumber shift", "Normalised data");

                /*
                ***Write the data to the file. NB The column number is converted to wavelength
                ***and the data is corrected for the system response.
                */
                for(q=0;q<NO_OF_POINTS;q++)
                        {
                        fprintf(file_handle,"%18f%18f", MEAS_WAVELENGTH(q) ,
                                        FREQ_SHIFT(q));
                        for(p=0;p<shots;p++)
                                fprintf(file_handle,"%18f",output_array[p][q] * filter_data[q]);
                        fprintf(file_handle,"\n");
                        }
                fclose(file_handle);
/*
***Write the file_name (the part of the full_file_name after the final '\') to the
***File Name string control on the front panel. The strrchr() function returns a pointer
***to the final '\', and the pointer is incremented by one.
*/
                SetCtrlVal (ccd_panel, CCD_PANEL_FILE_NAME, strrchr (full_file_name, '\\' ) + 1);
```

```
                break;
                }
}

int monitor_ADC()
{
while(!finished && checkstop())    /* Waits in loop until DAQ_Check() sets  */
        DAQ_Check (1, &finished, &points); /* finished = 1 or checkstop() returns 0. */
/*
*** Occaisionally, during long sequences of scans, the program gets stuck in the loop
*** above. The variable 'finished' sticks at 0 (ie not finished) and the variable
*** 'points' sticks at NO_OF_POINTS - 1.
*/

if(!finished)
        {
        DAQ_Clear (1);         /* Cancels DAQ operation if not already finished. */
        return 0;
        }
else
        {
        finished=0;
        return 1;              /* Returns the number of completed scans (ie, 1). */
        }
}


int read_once(unsigned short temp_read_once_array[])
/*
*** Returns 0 for no reading. Returns 1, and plots raw data if one reading is successful.
*/
{
register short i;
/*
*** nb: temp_read_once_array in the DAQ_Start function is offset by one (the first
*** conversion pulse is ignored by the Lab_PC+ card), NO_OF_COUNTS decreased by 1 to compensate.
*/
DAQ_Start (1, 4, gain, temp_read_once_array + 1, NO_OF_COUNTS - 1, 0, 0);
while(!finished && checkstop())                      /* Waits in loop until DAQ_Check() sets  */
        DAQ_Check (1, &finished, &points);   /* finished = 1 or checkstop() returns 0. */
/*
*** Occaisionally, during long sequences of scans, the program gets stuck in the loop
*** above. The variable 'finished' sticks at 0 (ie not finished) and the variable
*** 'points' sticks at NO_OF_POINTS - 1. (NB, this was when the loop was part of
*** the monitor_ADC() funcion, might be different now.)
*/

if(!finished)
        {
        DAQ_Clear (1);         /* Cancels DAQ operation if not already finished. */
        return 0;
        }
else
        {
        finished=0;
        for(i=1;i<NO_OF_POINTS;i++)
                {
                output_array[spectrum][i] = temp_read_once_array[i];
                }
        return 1;              /* Returns the number of completed scans (ie, 1). */
        }
}

int average_24(unsigned short average_24_array[][NO_OF_POINTS])
/*
*** This function averages 24 consecutive readings. (Averages a multiple of 4
*** because clock noise has a 1, 2, 3, 4 componant.) The first spectrum is ignored
*** as the first point is not recorded by the LabPC+ card.
*/
{
unsigned short p;
unsigned short q;

DAQ_Start (1, 4, gain, *(average_24_array) + 1, (25 * NO_OF_COUNTS) - 1, 0, 0);
if(!monitor_ADC())
        return 0;

for(p=0;p<NO_OF_POINTS;p++)
                {
                output_array[spectrum][p] = 0.0;
                }

for(p=0;p<NO_OF_POINTS;p++)
```

```
        {
        for(q=1;q<25;q++)
                {
                output_array[spectrum][p] += average_24_array[q][p] / 24.0;
                }
        }
return 1;
}


int referenced_read(unsigned short ref_read_array[][NO_OF_POINTS])
/*
*** Returns 1 and plots data if one referenced reading is successful, returns 0 otherwise.
*** A referenced reading is the reading from the reference cell (orange port of DiCon
*** optical fibre switch) subtracted from the sample cell reading (green port). The
*** referenced reading is stored in output_array, the same array as specified for the
*** raw data in the DAQ_Start function.
*/
{
        unsigned short p, q;

/*
*** Take C0 low for 10mS: both JK-clear gates, and timer B2 gate are taken low,
*** This forces both JKs into the low state, and restarts timer B2 so that the
*** DiCon fibre switch always starts with the same port on.
*** 32 consecutive CCD frames are read; the switch changes port after the second reading.
*** Sub-arrays 0 and 2 (readings 1 and 3) are discarded as the switch position is ambiguous.
*/
        DIG_Out_Port (1, 2, 0);
        Delay (0.01);           /* Delay in seconds. */
        DAQ_Start (1, 4, gain, *(ref_read_array) + 1, (32 * NO_OF_COUNTS) - 1, 0, 0);
        DIG_Out_Port (1, 2, 1);
        if(!monitor_ADC())
                return 0;

/*
*** Clear the arrays ref_read_array[0][] and r_r_a[16][]. Average [1][] to [14][] and put
*** in [0][], and average [17][] to [30][] and put in [16][].
*/
        for (q=0;q<512;q++)
                {
                ref_read_array[0][q]=0;
                ref_read_array[16][q]=0;
                }
        for (p=1;p<15;p++)
                {
                for (q=0;q<512;q++)
                        {
                        ref_read_array[0][q] += ref_read_array[p][q];
                        ref_read_array[16][q] += ref_read_array[p+16][q];
                        }
                }

/*
*** If reference cell correction is ON (selected from the front panel)
*** then the reference cell reading is scaled by the factor 'offset'
*** (also selected from the front panel).
*/
switch(correction)
        {
        case ON:
                for(q=0;q<NO_OF_POINTS;q++)
                        output_array[spectrum][q] = ref_read_array[0][q]/14.0 -
                        offset * ref_read_array[16][q]/14.0;
        break;
        case OFF:
                for(q=0;q<NO_OF_POINTS;q++)
                        output_array[spectrum][q] = ref_read_array[0][q]/14.0 -
                        ref_read_array[16][q]/14.0;
        break;
        }
return 1;
}

void read(int menubar, int menuItem, void *callbackData, int panel)
{
        averages=0;
        finished=0;

        disable_controls();
        read_controls();
        display_integration_time();
```

```
        ICTR_Setup (1, 0, 3, clock_speed, 1);  /* Sets LabPC timers from CLOCK_SPEED, etc */
        ICTR_Setup (1, 1, 3, pulse_length, 1);
        ICTR_Setup (1, 2, 2, integration_cycles, 1);

/*
*** This block takes a reading, which is then discarded due to ambiguous
*** integration time. (Point [0][0] is not used, as the LabPC+ does not measure
*** the first point. (See DAQ_CONFIG in the NI_DAQ reference manual.)
*/
        DAQ_Start (1, 4, gain, &temp_read_array[0][1], NO_OF_COUNTS, 0, 0);
        if(!monitor_ADC())
                enable_controls();

        switch(mode)
        {
        register unsigned short p=0;

        case REFERENCE_CELL_SUBTRACT:
                switch(menuItem)
                {

                case CCD_MENU_READ_SEQUENTIAL:
                for (spectrum=0;spectrum<shots;spectrum++) /* bigarrayoutput for loop */
                        {
                        referenced_read(temp_read_array);
                        /* *** Plots ouptut at the end of this function *** */
                        }
                break;

                case CCD_MENU_READ_MANUAL:
                for (spectrum=0;spectrum<shots;spectrum++) /* bigarrayoutput for loop */
                        {
                        SetCtrlVal (ccd_panel, CCD_PANEL_DISPLAYED_SPECTRUM, spectrum);
                        while(referenced_read(temp_read_array))/*  referenced_read  returns  1,
indicating scan completed. */
                                {
                                /* *** Sits in this loop until stop is pressed. *** */
                                plot_output(spectrum);
                                }
                        }
                DAQ_Clear(1);
                break;

                case CCD_MENU_READ_AUTO:
                for (spectrum=0;spectrum<shots;spectrum++) /* bigarrayoutput for loop */
                        {
                        register unsigned short n;

                        GetCtrlVal (ccd_panel, CCD_PANEL_NO_AVERAGES, &averages);
                        for(n=1; referenced_read(temp_read_array) && (n <= averages); n++)
                                {
                                for(p=0;p<NO_OF_POINTS;p++)
                                        running_average[p] = ((n-1.0)/n)*running_average[p] +
                                                      (1.0/n) * output_array[spectrum][p];
                                plot_output(spectrum);
                                SetCtrlVal (ccd_panel, CCD_PANEL_DISPLAYED_SPECTRUM, spectrum);
                                }
                        for(p=0;p<NO_OF_POINTS;p++)
                                output_array[spectrum][p]=running_average[p];
                        }
                break;
                }
         break;

        default: /* For modes other than reference cell subtract. */
                switch(menuItem)
                {
                case CCD_MENU_READ_SEQUENTIAL:
/*
*** This DAQ_Start command will read 'shots' * NUMBER_OF_POINTS datapoints from the CCD, ie
*** 'shots' complete spectra. The very first data point is skipped, as the LabPC+ card will not
*** measure it. (See NI-DAQ ref manual description of the DAQ_Config function.)
*/
                DAQ_Start (1, 4, gain, *(temp_read_array)+1, shots*NO_OF_COUNTS - 1 , 0, 0);
                while(!finished && checkstop())    /* Waits in loop until DAQ_Check() sets  */
                        DAQ_Check (1, &finished, &points); /* finished = 1 or checkstop() returns
0. */
/*
*** Occasionally, during long sequences of scans, the program gets stuck in the loop
*** above. The variable 'finished' sticks at 0 (ie not finished) and the variable
*** 'points' sticks at NO_OF_POINTS - 1. (NB, this was when the loop was part of
*** the monitor_ADC() funcion, might be different now.)
*/
```

```
                if(!finished)
                        {
                        DAQ_Clear (1);         /* Cancels DAQ operation if not already finished.
*/
                        }
                else
                        {
                        finished=0;
                        }
                if(mode==BACKGROUND_SUBTRACT)
                        {
                        for(p=1;p<shots*NO_OF_POINTS;p++)
                                {
/*
*** output_array is cast into a single dimensional array so that only one counter is needed.
*/
                                (*output_array)[p] = (*temp_read_array)[p] - background[(unsigned
short)fmod(p,NO_OF_POINTS)];
                                }
                        }
                else
                        {
                        for(p=1;p<shots*NO_OF_POINTS;p++)
                                {
/*
*** output_array is cast into a single dimensional array so that only one counter is needed.
*/
                                (*output_array)[p] = (*temp_read_array)[p];
                                }
                        }
                break;

                case CCD_MENU_READ_MANUAL:
                for (spectrum=0;spectrum<shots;spectrum++) /* bigarrayoutput for loop */
                {
                SetCtrlVal (ccd_panel, CCD_PANEL_DISPLAYED_SPECTRUM, spectrum);
                while(read_once(*temp_read_array))
                /* ie, if read_1 returns 1, indicating scan completed. */
                /* if monitor_ADC() returns zero, loop terminates.   */
                        {
                        if(mode==BACKGROUND_SUBTRACT)
                                {
                                for(p=0;p<NO_OF_POINTS;p++)
                                        output_array[spectrum][p] -= background[p];
                                }
                        plot_output(spectrum);
                        }
                }/* End of bigarrayoutput for loop */
                break;

                case CCD_MENU_READ_AUTO:
                for (spectrum=0;spectrum<shots;spectrum++) /* bigarrayoutput for loop */
                {
                register unsigned short n;
                GetCtrlVal (ccd_panel, CCD_PANEL_NO_AVERAGES, &averages);
                for(n=1; average_24(temp_read_array) && (n <= averages); n++)
                {
                if(mode==BACKGROUND_SUBTRACT)
                        {
                        for(p=0;p<NO_OF_POINTS;p++)
                                output_array[spectrum][p] -= background[p];
                        }
                for(p=0;p<NO_OF_POINTS;p++)
                        running_average[p] = ((n-1.0)/n) * running_average[p] +
                                                        (1.0/n) * output_array[spectrum][p];
                }
                for(p=0;p<NO_OF_POINTS;p++)
                        output_array[spectrum][p]=running_average[p];
                plot_output(spectrum);
                SetCtrlVal (ccd_panel, CCD_PANEL_DISPLAYED_SPECTRUM, spectrum);
                }/* End of bigarrayoutput for loop */
                break;
                }
        break;
        }
        enable_controls();
        SetCtrlVal (ccd_panel, CCD_PANEL_NO_AVERAGES, averages);
        SetCtrlVal (ccd_panel, CCD_PANEL_DISPLAYED_SPECTRUM, shots - 1);
        plot_output(shots - 1);
        Beep ();
}
```

```
main()
{
/*
*** Set up the display panel and menu. ***
*/
ccd_panel = LoadPanel (0, "ccdarray.uir", CCD_PANEL);
ccd_menu = LoadMenuBar (CCD_PANEL, "ccdarray.uir", CCD_MENU);
DisplayPanel(ccd_panel);
SetPanelAttribute (ccd_panel, ATTR_CLOSE_CTRL, CCD_MENU_FILE_QUIT);

/*
*** Read the default parameters, display the integration time in seconds, and load the
*** filter correction matrix.
*/
read_controls();
display_integration_time();
system_correction();

/*
***Read the system date for inclusion in the data file, and display it on the front panel.
*/
GetSystemDate (&month, &day, &year);
sprintf(date, "Date: %d.%d.%d", day, month, year);
SetCtrlVal (ccd_panel, CCD_PANEL_DATE, date);

/*
***Plot the (currently) empty array 'output_array' so that first call to
***'DeleteGraphPlot()' (in 'plot_output()') does not return an error.
*/
array_plot = PlotY (ccd_panel, CCD_PANEL_GRAPH, &output_array[0][0],
                                    NO_OF_POINTS, VAL_FLOAT, VAL_THIN_LINE,
                                    VAL_EMPTY_SQUARE, VAL_SOLID, 1, VAL_YELLOW);

/*
*** Line 0 of Port C controls the gate of LabPC card counter B2,
*** and the clear line of the JK flip flop in the power supply box.
*/
DIG_Prt_Config (1, 2, 0, 1);
DIG_Out_Port (1, 2, 1);
/*
***While C0 is high, gate B2 is open.
*/
/*
*** This part starts the counters ticking.     ***
*/
ICTR_Setup (1, 0, 3, clock_speed, 1);
ICTR_Setup (1, 1, 3, pulse_length, 1);
ICTR_Setup (1, 2, 2, integration_cycles, 1);
/*
*** They will carry on until the computer is switched ***
*** off or they are told to do something else.     ***
*/

DAQ_Config (1, 1, 3);
/*
*** External start scan triggering on Pin 38 (positive edge).
*** External clock control of sample-interval AND scan interval
*** timing (ie mode 3) selected. Mode 1 should work (?) but then
*** the counter B1 is required by the LabPC card for scan interval
*** timing. In this project all three counters are required to
*** generate the timing control signals, and scan interval timing
*** is not required.
*/
/*
*** All further actions are prompted by input from   ***
*** the front panel.                 ***
*/
RunUserInterface();
}
```

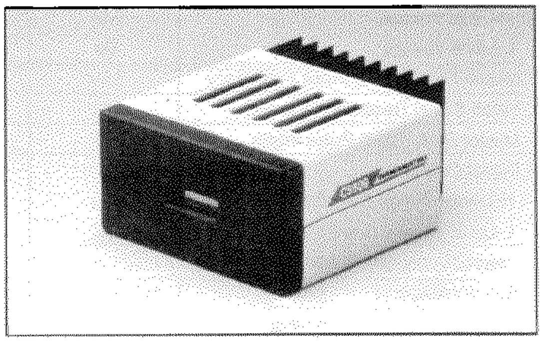
Appendix C: Data Sheets

HAMAMATSU

TENTATIVE DATA
Mar. 1994

CCD MULTICHANNEL DETECTOR HEAD C5809 SERIES

A CCD multichannel detector head that assures high sensitivity and large active area by utilizing the binning operation of an FFT-CCD image sensor



FEATURES

- Designed for use with a thermoelectrically-cooled FFT-CCD image sensor *1
  Binning operation assures use of a large active area*2
- Built-in driver/amplifier and temperature control circuits
- Highly stable temperature control ensures a constant cooling temperature of Ts=0±0.05℃ (at Ta=10 to 35℃)
- Operates from simple signal inputs
- High sensitivity and wide dynamic range
- Various models are available according to image sensor type

APPLICATIONS

- Fluorescence spectroscopy
- Raman spectroscopy
- Other low-light-level detection

The C5809 series is a family of high sensitivity multichannel detector heads specifically developed for spectrophotometry at very low light levels where conventional image sensors cannot measure any signal. The C5809 is designed to accommodate an FFT-CCD image sensor that provides significantly low noise when compared with other image sensors, making it ideally suited for fluorescence spectroscopy, Raman spectroscopy and other low-light-level detection.
The C5809 incorporates FFT-CCD image sensor, a low-noise driver/amplifier circuit and a temperature control circuit that enable stable operation of a thermoelectrically-cooled FFT-CCD image sensor by input of simple external signals. The image sensor can be cooled to the preset temperature (Ts=0℃) as soon as the power is turned on. Should the cooler fail and cause the circuitry to overheat, the built-in protection circuit automatically turns the power off. Despite its compact size, the case configuration is designed for good heat dissipation, and threaded mounting holes are also provided on the front panel for connections to another device such as a monochromator.

■SELECTION GUIDE

The C5809 series consists of the following models depending on the CCD image sensor used.

Type No.	CCD Image Sensor			
	Type No.	Number of Pixels Pixels (H)×Pixels (V)	Pixel Size μm(H)×μm(V)	Effective Active Area mm(H)×mm(V)
C5809-0906	S5469-0906	512 × 64	24×24	12.28×1.54
C5809-0907	S5469-0907	512 ×128		12.28×3.07
C5809-1006	S5469-1006	1024 × 64		24.57×1.54
C5809-1007	S5469-1007	1024 ×128		24.57×3.07

*1: The FFT-CCD (full frame transfer CCD) has charge transfer sections that are also used as light receiving areas, being different from interline transfer CCD (IT-CCD) commonly used in video cameras. Compared to the IT-CCD, the FFT-CCD offers advantages of low dark current, a 100% open area ratio and low image lag.

*2: The FFT-CCD was originally designed as a 2-dimensional image sensor. However, it can be operated like a linear image sensor having a large active area by transferring all the pixel signals in the vertical direction to the horizontal register (this is referred to as line binning).

CCD MULTICHANNEL DETECTOR HEAD C5809 SERIES

■MAXIMUM RATINGS

Parameters	Symbols	Min.	Typ.	Max.	Units
Supply Voltage (For Digital Circuitry)	$+V_D$	−0.5		+7	V
Supply Voltage (For Analog Circuitry)	$\pm V_{A1}$			±18	V
	$+V_{A2}$			+30	V
Digital Input Voltage				V_D	V

■ELECTRICAL SPECIFICATIONS (Ta=25℃, V_D=+5V, V_{A1}=±15V, V_{A2}=+24V, unless otherwise specified)

Parameter	Symbols	Min.	Typ.	Max.	Units
Digital Input					
H-Level Voltage	V_{IH}	+2.0		$+V_D$	V
L-Level Voltage	V_{IL}	−0.5		+0.8	V
Master Clock (CLK) Pulse Frequency	f_{CLK}			200	kHz
Video Signal Readout Frequency	f_V			$f_{CLK}/4$	Hz
Master Start (Start) Pulse Width	tst	$1/f_{CLK}$			s
Digital Output					
H-Level Voltage (Io=−6mA)	V_{IH}	+2.0			V
L-Level Voltage (Io=+6mA)	V_{IL}			+0.8	V
Power Supply Conditions					
Rated Voltage: Digital	$+V_D$	+4.75	+5.0	+5.25	V
Analog	$\pm V_{A1}$	±14.5	±15.0	±15.5	V
	$+V_{A2}$	+23.5	+24.0	+24.5	V
Current Consumption: + 5Vdc *3				+2.0	A
+15Vdc				+100	mA
−15Vdc				−100	mA
+24Vdc				+10	mA
Operating Temperature	Topr	+10		+35	℃
Storage Temperature	Tstg	0		+50	℃

*3: Including the current consumption of the Peltier element incorporated in the CCD image sensor (S5469 series).

HAMAMATSU

■ ELECTRICAL AND OPTICAL SPECIFICATIONS (Ta=25℃, Ts=0℃, VD=+5V, VA1=±15V, VA2=+24V, unless otherwise specified)

Parameters*4	Symbols	Min.	Typ.	Max.	Units
Full Well Capacity *5	Fw		600k		e-
Conversion Gain *6	Sv		8.0		μV/e-
Dark Signal *7	DS		200	600	e-/pixel/s
Readout Noise	Nr	15	20		e-rms
Dynamic Range	DR		30k		
Photo Response Non-Uniformity *8	PRNU		-	±10	%
Spectral Response Range	λ		400 to 1100		nm

*4: Common to all models.
*5: Horizontal register value.
*6: Including the circuit gain.
*7: at MPP mode. Vertical register value. The actual value equals the sum of the pixels in the vertical direction because of the binning operation.
*8: Measured at 50% of the saturated output charge.

■ OTHER SPECIFICATIONS (For Temperature Controller) (Ta=25℃, VD=+5V, VA1=±15V, VA2=+24V, unless otherwise specified)

Parameters*9	Symbols	Min.	Typ.	Max.	Units
Cooling Temperature	Ts	−1	0	+1	℃
Temperature Control Range	ΔTs	−0.05		+0.05	℃
Power Consumption of Peltier Element	Pp			7	W
Cool Down Time to Preset Temperaure	to			5	min
Setting Temperature for Overheat Protection	To	+40			℃

*9: Other functions include error display, automatic power off, and detection of electrical opens and shorts by the thermosensor.

Figure 1: Spectral Response

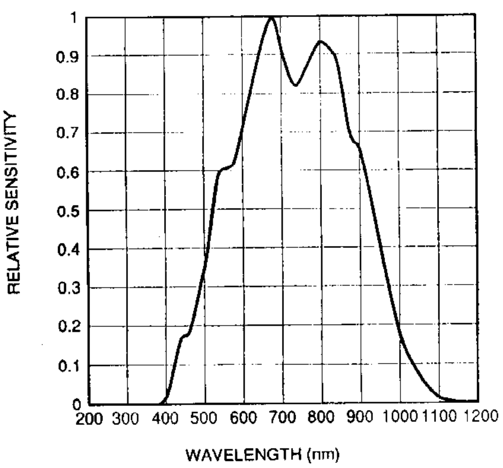
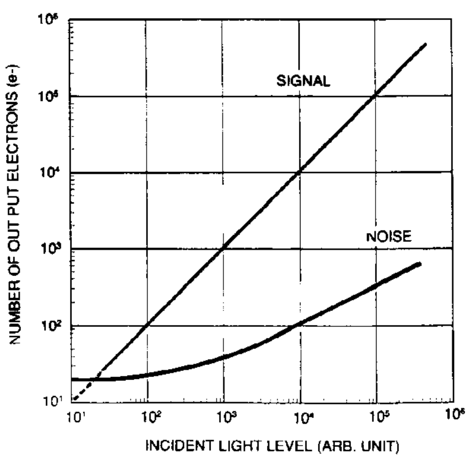


Figure 2: Input/Output Characteristics



CCD MULTICHANNEL DETECTOR HEAD C5809 SERIES

Figure 3: Block Diagram of S5469 Series FFD-CCD Image Sensor

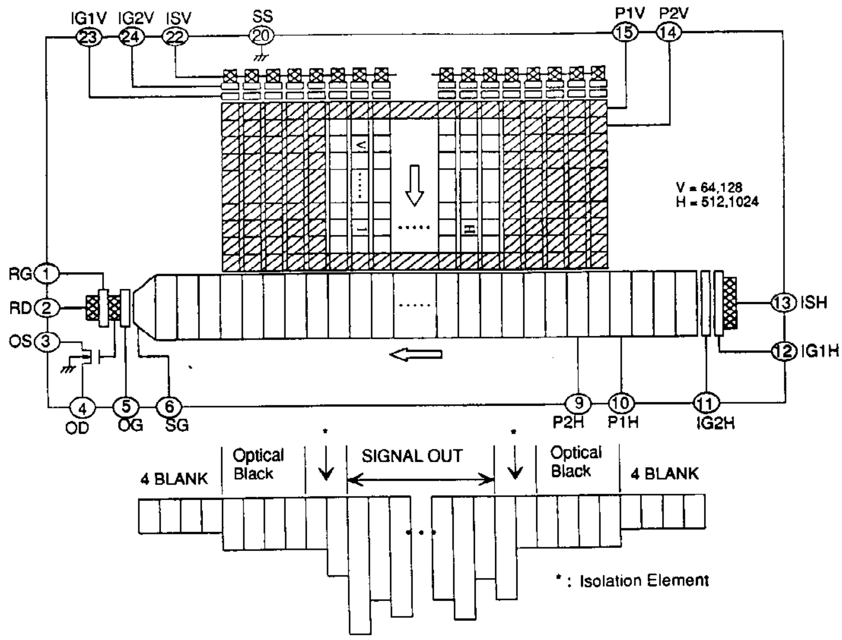
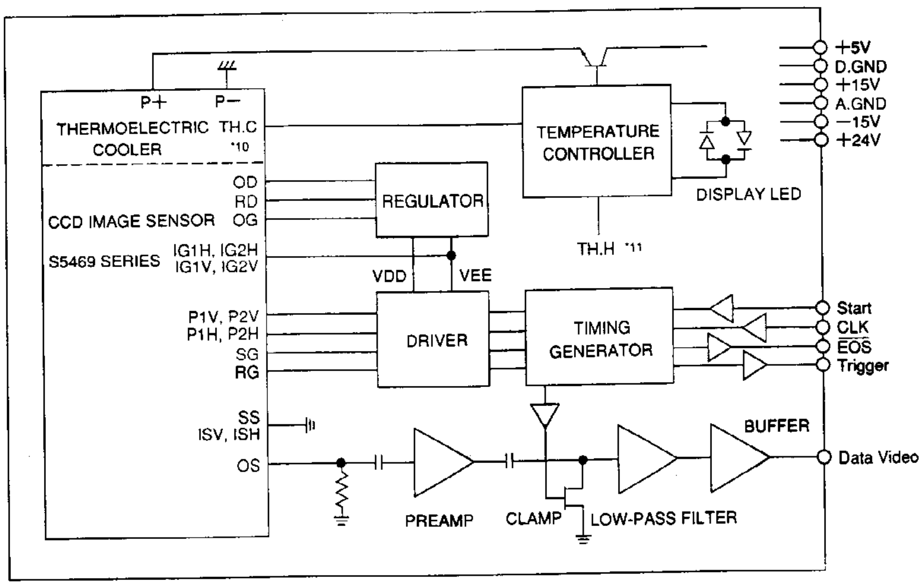


Figure 4: Block Diagram of C5809 Series

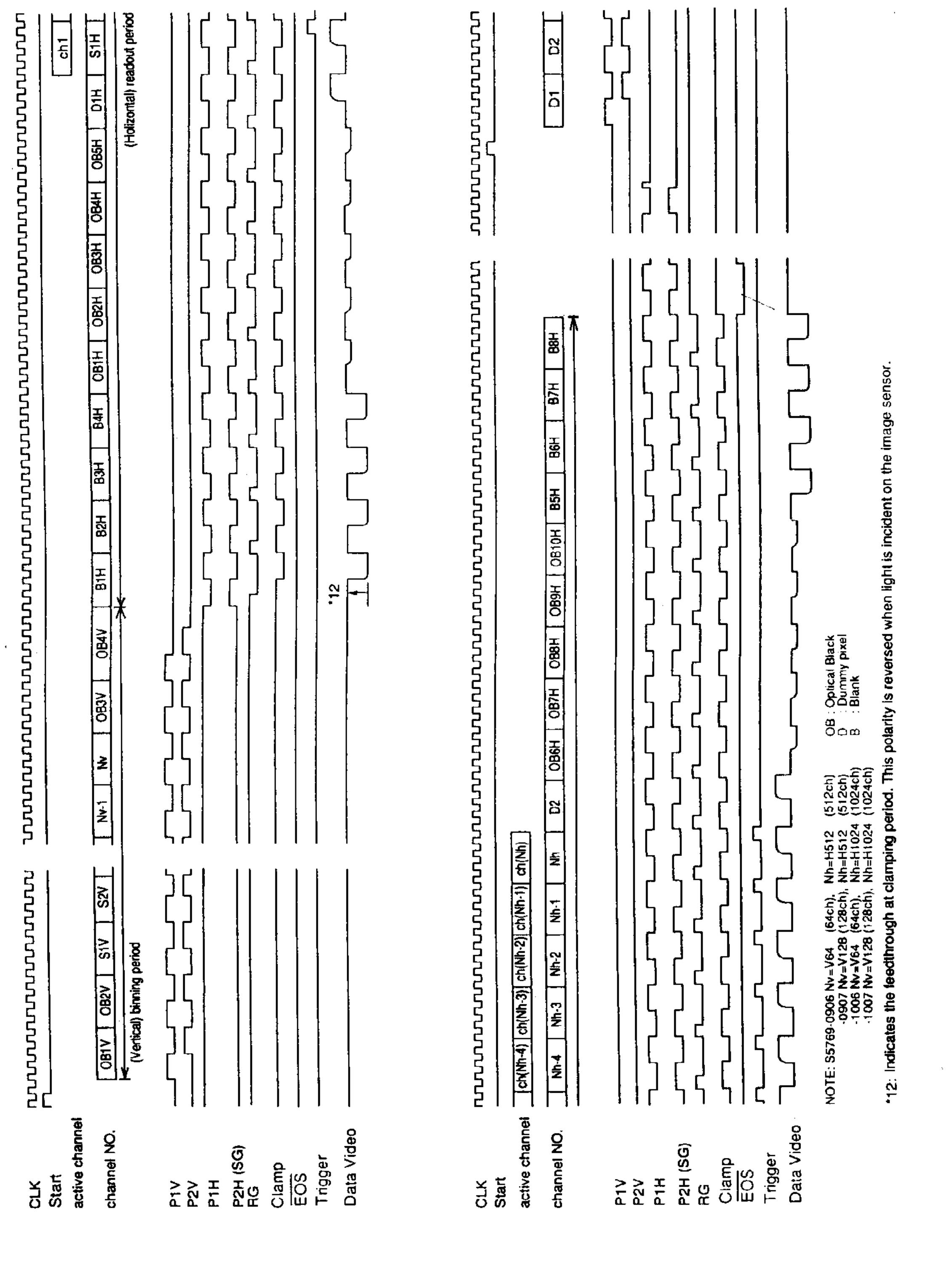


*10: Thermistor incorporated in the image sensor. Used for temperature monitoring of the image sensor.

*11: Thermistor mounted on the heatsink fins. Used for temperature monitoring of the heat radiating side.

HAMAMATSU

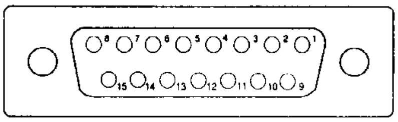
Figure 5: Timing Chart



CCD MULTICHANNEL DETECTOR HEAD C5809 SERIES

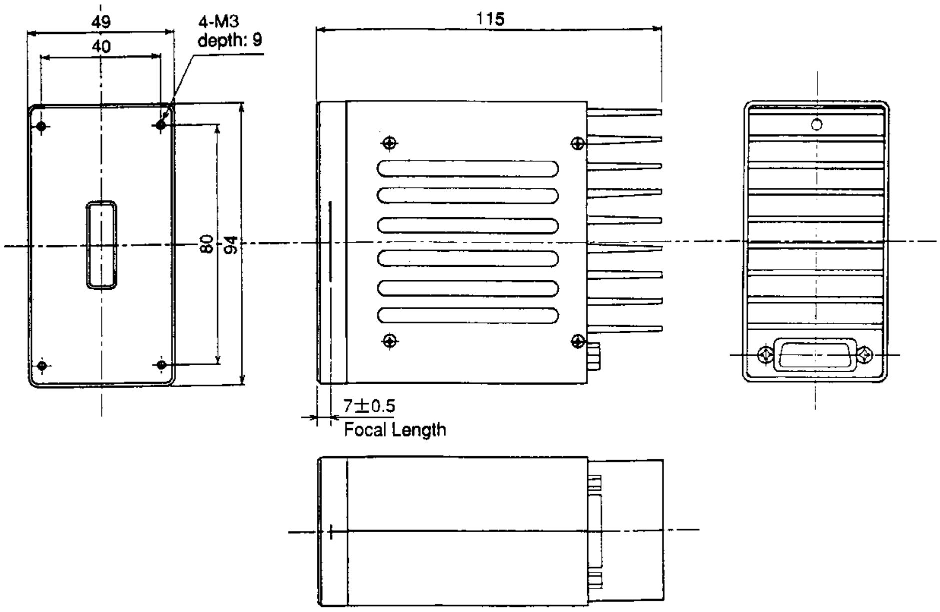
■I/O CONNECTOR PIN DESCRIPTION

Pin No.	Terminal Name	Description
1	NC	No connection.
2	Data Video	Analog video output signal. Positive polarity.
3	+VA1 (+15V)	Power supply for analog circuitry.
4	−VA1 (−15V)	Power supply for analog circuitry.
5	+VD (+5V, P+)	Power supply for digital circuitry. For the thermoelectric cooler in the CCD image sensor.
6	Start	Digital input signal to initialize the circuit. H-CMOS compatible. Positive logic. The start pulse interval determines the signal storage time of the sensor.
7	CLK	Digital input signal to specify the circuit operation. H-CMOS compatible. Operates at the rising edge.
8	$\overline{EOS}$	Digital output signal to indicate the end of scan of the CCD image sensor. H-CMOS compatible. Negative logic.
9	A. GND	Analog ground.
10	A.GND	Analog ground.
11	+VA2 (+24V)	Power supply for analog circuitry.
12	D.GND (P−)	Digital ground. Power supply return of the thermoelectric cooler mounted in the CCD image sensor.
13	D.GND	Digital ground.
14	D.GND	Digital ground.
15	Trigger	Digital output signal for A/D conversion. H-CMOS compatible. Positive logic.



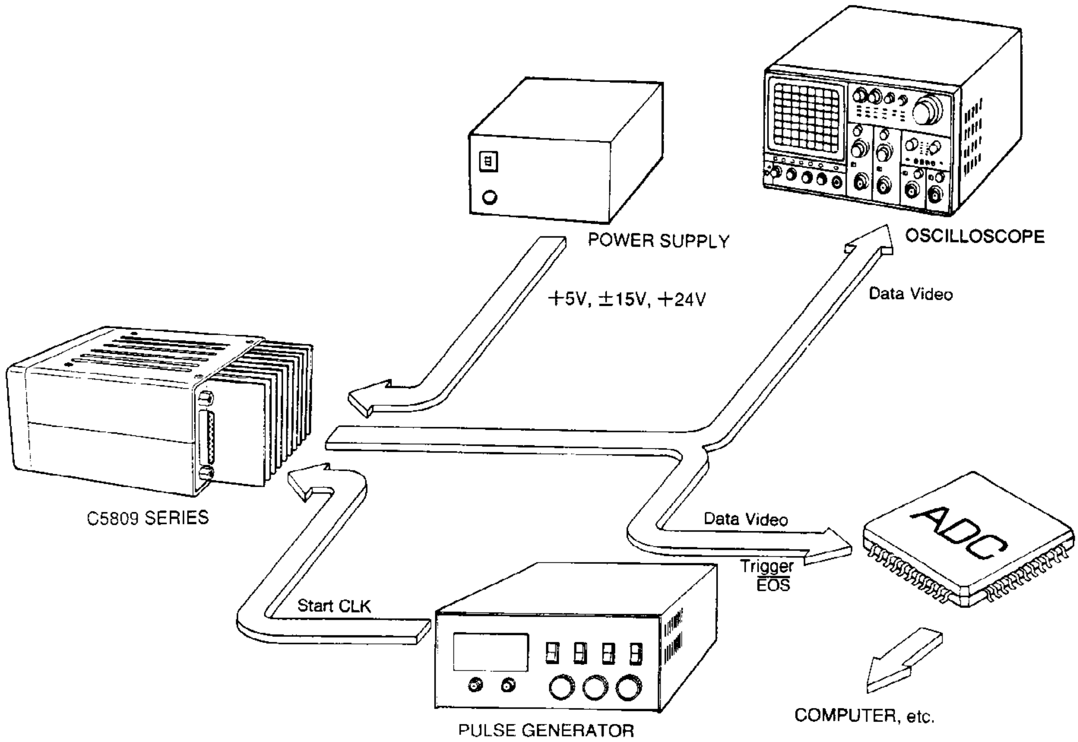
15-pin D-sub Connector

Figure 6: Dimensional Outlines (Unit: mm)



HAMAMATSU

Figure 7: Connection Example



■ Handling Precautions

The C5809 series is a precision device, so sufficient care should be taken in its handling.

- Never disassemble or modify the device as this may cause an operating failure.
- Protect the device from shocks such as drops or impacts as these may cause breakage.
- Avoid storing the device in high temperature and high humidity locations for long periods of time.
- Air vents are provided on the case. Blocking these vents during operation may cause overheating, and therefore this must be avoided.
- Take sufficient care when making connections to other equipment.
- Never exceed the maximum ratings during operation.

Observe the following precautions to obtain the fullest performance of the device.

- Take sufficient care to protect the device against external electrostatic or magnetic effects. Provide an appropriate shield; for example, use of a shield cable is recommended.
- Use of a power supply that exhibits minimum ripple and noise is recommended.
- For high-precision measurements, take action to ensure no extraneous light enters the device.

HAMAMATSU

TENTATIVE DATA

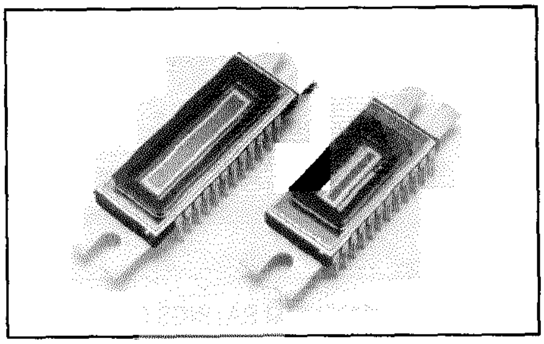
Apr. 1994

FFT-CCD Image Sensors for Scientific Applications S5469 Series

Wide Dynamic Range, for Low Light Level Detection

FEATURES

- 512(H)×64(V) to 1024(H)×128(V) Pixel Format
- Pixel Size 24 μm×24 μm
- Line, Pixel Binning
- 100% Fill Factor
- Wide Dynamic Range
- Low Dark Signal
- Low Readout Noise
- MPP Operation



APPLICATIONS

- Fluorescence Spectrometer
- Raman Spectrophotometer
- Optical and Spectrophotometric analyzer
- For Low Light Level Detection Requiring

The S5469 series is a family of FFT-CCD (appendix 1) image sensors specifically designed for low-light-level detection in scientific applications. By utilizing the binning operation (appendix 2), the S5469 can be used as a linear image sensor having a long aperture in the direction of the device length. This makes the S5469 series ideally suited for use in spectrophotometry. The binning operation offers significant improvement in signal-to-noise ratio and signal processing speed compared to conventional methods by which signals are digitally added by an external circuit. The S5469 series also features low noise and low dark signal (MPP mode operation; appendix 3). This enables low-light-level detection and long integration time, thus achieving a wide dynamic range.

The S5469 series has an effective pixel size of 24 μm×24 μm and is available in image areas ranging from 12.28(H)×1.54(V)mm^2 (512×64 pixels) up to a large image area of 24.57(H)×3.07(V)mm^2 (1024×128 pixels).

A one-stage Peltier element is built into the same package for thermoelectric cooling. At room temperature operation the device can be cooled down to 0℃ without using any other cooling technique. In addition, since both the CCD chip and Peltier element are hermetically sealed, no dry air is required thus allowing easy handling.

■S5469 Series Selection Guide

Sensors	Effective Pixels pixels(H)×pixels(V)	Image Area mm(H)×mm(V)
S5469-0906	512 × 64	12.28×1.54
S5469-0907	512 × 128	12.28×3.07
S5469-1006	1024 × 64	24.57×1.54
S5469-1007	1024 × 128	24.57×3.07

FFT-CCD Image Sensors for Scientific Applications S5469 Series

■S5469 Series

Pixel size	: 24 μm(H)×24 μm(V) (aspect ratio 1:1)
Effective Pixels	: Selection Guide
Vertical Clock Phase	: 2 phase
Horizontal Clock Phase	: 2 phase
Output	: 1 Stage MOSFET source follower
Package	: 24 pin ceramic DIP with internal cooler DIMENSIONAL OUTLINES
Window	: Sapphire

The S5469 Series dose not have gate protection circuit.
It requires extreme care during handling to avoid static damage.

■ABSOLUTE MAXIMUM RATINGS (Ta=25℃)

Parameter	Symbol	Min.	Max.	Unit
Storage Temperature	Tstg	−50	50	℃
Operating Temperature	Topr	−50	30	℃
OD Voltage	VOD	−0.5	+25	V
RD Voltage	VRD	−0.5	+18	V
ISV Voltage	VISV	−0.5	+15	V
ISH Voltage	VISH	−0.5	+15	V
IGV Voltage	VIG1V, VIG2V	−10	+15	V
IGH Voltage	VIG1H, VIG2H	−10	+15	V
SG Voltage	VSG	−10	+15	V
OG Voltage	VOG	−10	+15	V
RG Voltage	VRG	−10	+15	V
All Vertical Clock		−10	+15	V
All Horizontal Clock		−10	+15	V

All voltage are respect to terminal SS.

HAMAMATSU

■DC AND CLOCK CHARACTERISTICS (Ta=25℃)

Parameter		Symbol	Min.	Typ.	Max.	Unit
Output Transistor Drain Voltage		VOD	18	20	22	V
Reset Drain Voltage MPP mode		VRD	11.5	12	12.5	V
Output Gate Voltage MPP mode		VOG	1	3	5	V
Substrate Voltage		VSS		0		V
Test Point (Input Source)		VISV, VISH		VRD		V
Test Point (Vertical Input Gate) MPP mode		VIG1V,VIG2V	−8	0		V
Test Point (Horizontal Input Gate) MPP mode		VIG1H,VIG2H	−8	0		V
CCD Vertical Shift Register Clock Voltage MPP mode	HIGH	VP※VH	4	6	8	V
	LOW	VP※VL	−9	−8	−7	
CCD Horizontal Shift Register Clock Voltage MPP mode	HIGH	VP※VH	4	6	8	V
	LOW	VP※VL	−9	−8	−7	
Summing Gate Voltage MPP mode	HIGH	VSGH	4	6	8	V
	LOW	VSGL	−9	−8	−7	
Reset Gate Voltage MPP mode	HIGH	VRGH	4	6	8	V
	LOW	VRGL	−9	−8	−7	

※ indicate 1 or 2

■CHIP THERMISTOR

Parameter	Value
Resistance (at 25℃)	10kΩ
B-constant (at 25℃)	3450k
Operating Temperature	−40 ~ +100℃

REGISTANCE vs. TEMPERATURE CHARACTERISTIC

Temperature (℃)	Registance (kΩ)
−20	78.4
−10	46.7
0	28.1
10	18.2
20	12.2
25	10.0
30	8.3
40	5.7

FFT-CCD Image Sensors for Scientific Applications S5469 Series

■PELTIER ELEMENT TYPE 1. (T-06E 144P-RNO)

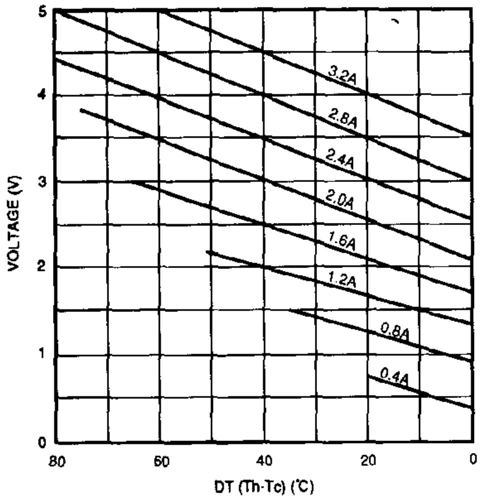
Peltier element type 1 is built into the binning type CCD "S5469-1006", "S5469-1007" for thermoelectric cooling.

Parameter	Value
Internal Resistant (at 25℃)	1.25 Ω
Max. Current (Tc-Th 20℃)	3.6A
Max. Voltage (Tc-Th 80℃)	6.2V
Max. Heat Absorption (Tc-Th 20℃)	7.5W

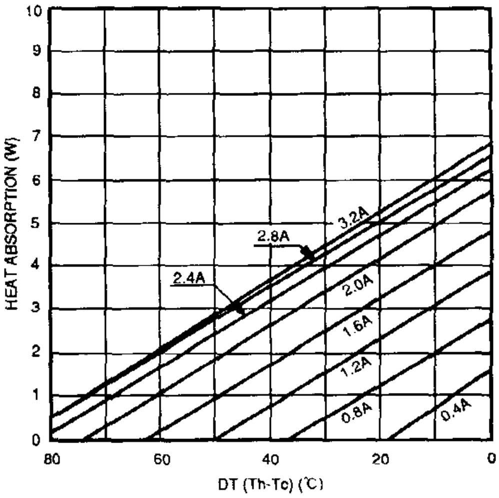
PERFORMANCE DIAGRAM OF PELTIER ELEMENT TYPE 1.

(Tc=0℃)

■Voltage - DT

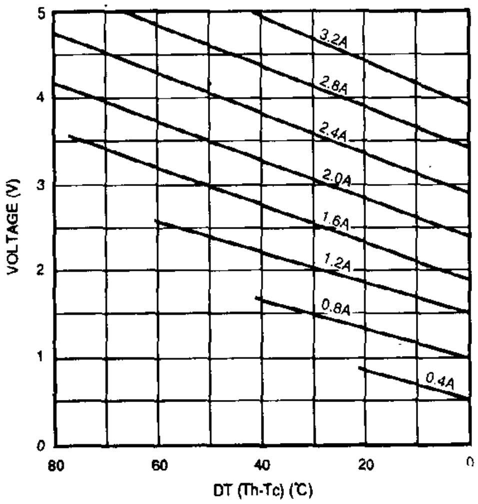


■Heat Absorption - DT

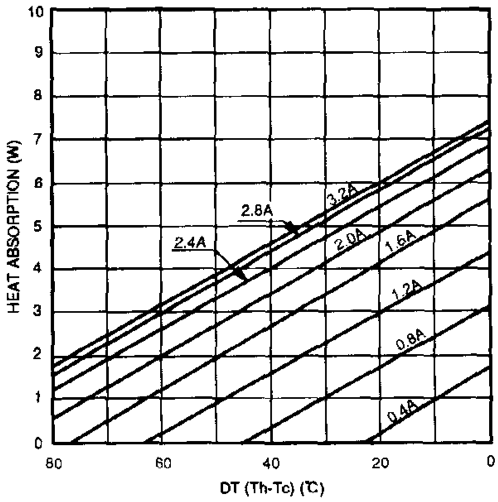


(Tc=20℃)

■Voltage - DT



■Heat Absorption - DT



HAMAMATSU

■PELTIER ELEMENT TYPE 2. (T-06E 108P-RNO)

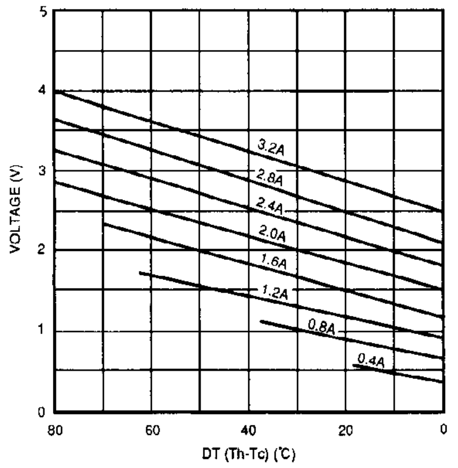
Peltier element type 2 is built into the binning type CCD "S5469-0906", "S5469-0907" for thermoelectric cooling.

Parameter	Value
Internal Resistant (at 25℃)	0.983Ω
Max. Current (Tc-Th 20℃)	3.6A
Max. Voltage (Tc-Th 80℃)	4.7V
Max. Heat Absorption (Tc-Th 20℃)	5.7W

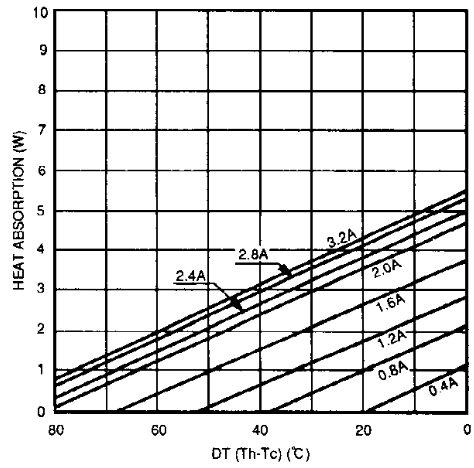
PERFORMANCE DIAGRAM OF PELTIER ELEMENT TYPE 2.

(Tc=0℃)

■Voltage - DT

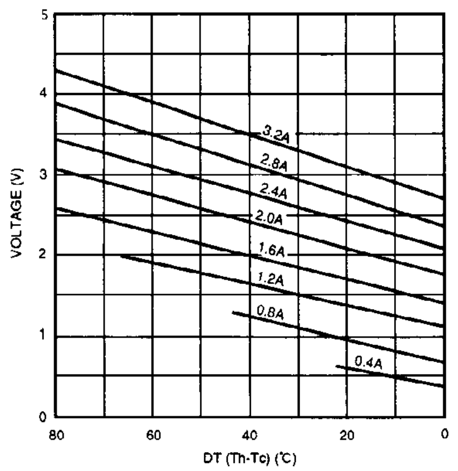


■Heat Absorption - DT

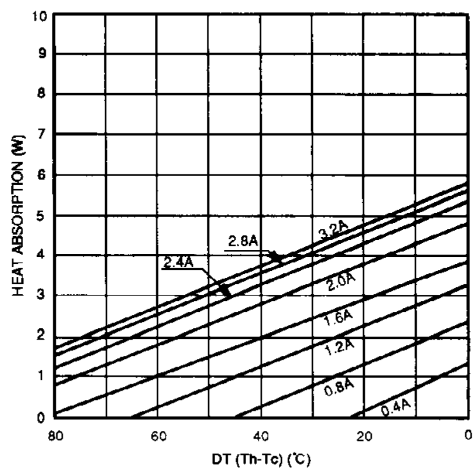


(Tc=20℃)

■Voltage - DT



■Heat Absorption - DT



FFT-CCD Image Sensors for Scientific Applications S5469 Series

■Electrical Characteristics (Ta=25℃)

Parameter	Symbol	Min.	Typ.	Max.	Unit
Signal Output Frequency	fc		1		MHz
Reset Clock Frequency	frg		1		MHz
CCD Vertical Shift Register Capacitance*1	CP1V, CP2V				pF
S5469-1006			2000		
S5469-1007			3200		
S5469-0906			1000		
S5469-0907			1600		
CCD Horizontal Shift Register Capacitance	CP1H, CP2H				pF
S5469-1006, 1007			300		
S5469-0906, 0907			150		
Summing Gate Capacitance	CSG		7		pF
Reset Gate Capacitance	CRG		7		pF
Charge Transfer Efficiency*1	CTE		0.99995		
DC Output Level*3	Vout	12	15	18	V
Output Impedance*3	Zo		3K		Ω
Power Consumption*2, 3	P		15		mW

*1 : Measured at half of full well capacity. CTE is defined per pixel.
*2 : Power consumption at on chip amplifier.
*3 : This value depends on load resistance.

■Electric-Optical Characteristics (−40℃ if not Remark)

Parameter	Symbol	Min.	Typ.	Max.	Unit
Saturation Output Voltage	Vsat		Fw×Sv		V
Full Well Capacity*1	Fw				e-
Vertical			300K		
Horizontal			600K		
Summing			800K		
CCD Node Sensitivity	Sv	0.8	1.2		μV/e-
Dark Signal*2 MPP mode	DS		200	600	e-/pixel/s
Readout Noise*3	Nr	15	20		e-rms
Dynamic Range	DR		15K		
Photo Response Non-Uniformity*4	PRNU		-	±10	%
Spectral Response Range	λ		400~1100		nm

*1 : Horizontal register saturation at Vertical Binning.
*2 : T=0℃. Dark Signal doubles for every 5 to 7℃
*3 : Operating frequency is 150kHz.
*4 : Measured at half of full well capacity. CTE is defined per pixel.

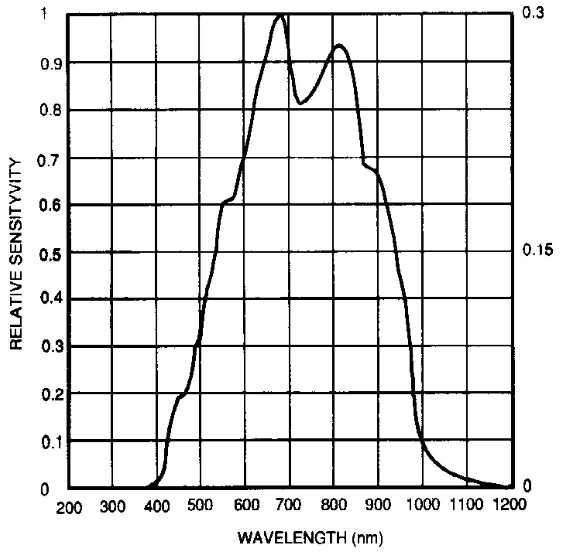
DR (Dynamic Range) $DR = \frac{\text{FULL WELL CAPACITY (Vertical)}}{\text{READOUT NOISE}}$

PRNU (Photo Response Non-Uniformity) $PRNU(\%) = \frac{\text{Noise}}{\text{Signal}} \times 100$

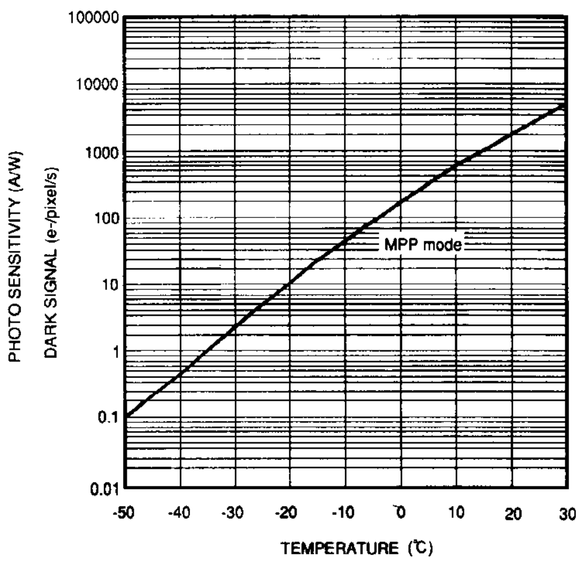
[Noise : Fixed Pattern Noise (peak to peak)]

HAMAMATSU

■Spectral Response (T=25℃)

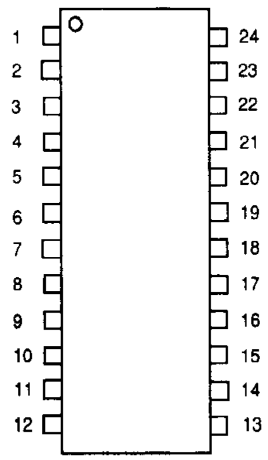


■Dark Signal vs Temperature



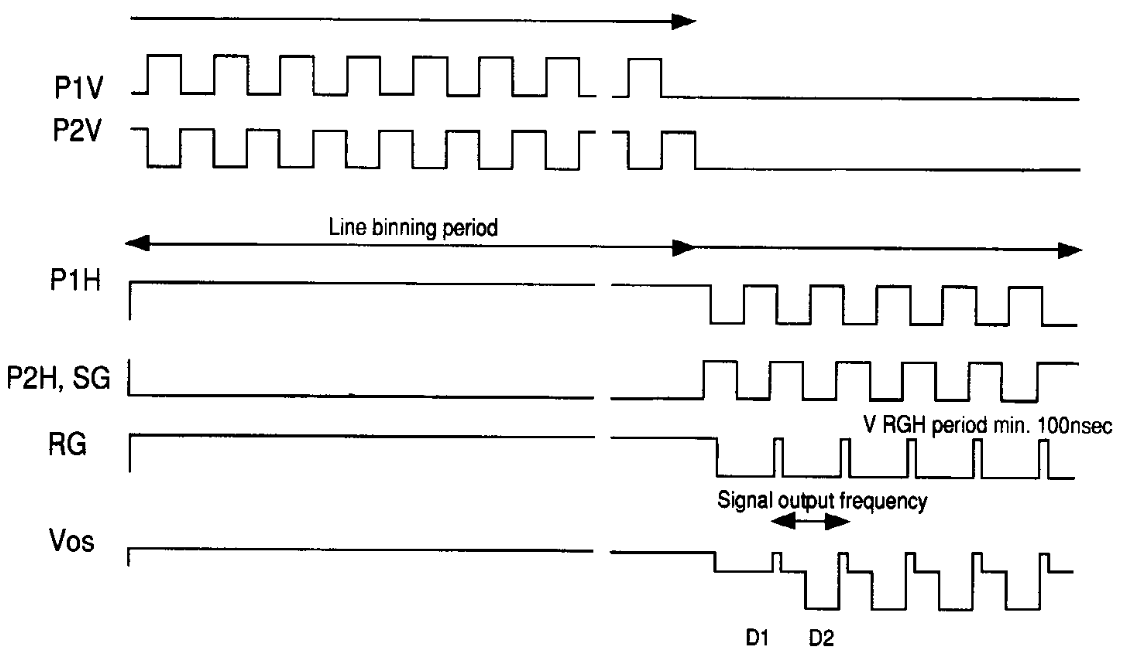
■Pinout

Pin No.	Symbol	Function	Remark
1	RG	Reset Gate	
2	RD	Reset Drain	
3	OS	Output Source	
4	OD	Output Transistor Drain	
5	OG	Output Gate	
6	SG	Summing Gate	
7	Th1	Thermistor	
8	Th2	Thermistor	
9	P2H	CCD Horizontal Register Clock-2	
10	P1H	CCD Horizontal Register Clock-1	
11	IG2H	Test Point (Horizontal Input Gate-2)	
12	IG1H	Test Point (Horizontal Input Gate-1)	
13	ISH	Test Point (Horizontal Input Source)	RD
14	P2V	CCD Vertical Register Clock-2	
15	P1V	CCD Vertical Register Clock-1	
16	NC		
17	NC		
18	P-	Peltier-	
19	P+	Peltier+	
20	SS	Substrate (GND)	
21	NC		
22	ISV	Test Point (Vertical Input Source)	RD
23	IG2V	Test Point (Vertical Input Gate-2)	
24	IG1V	Test Point (Vertical Input Gate-1)	

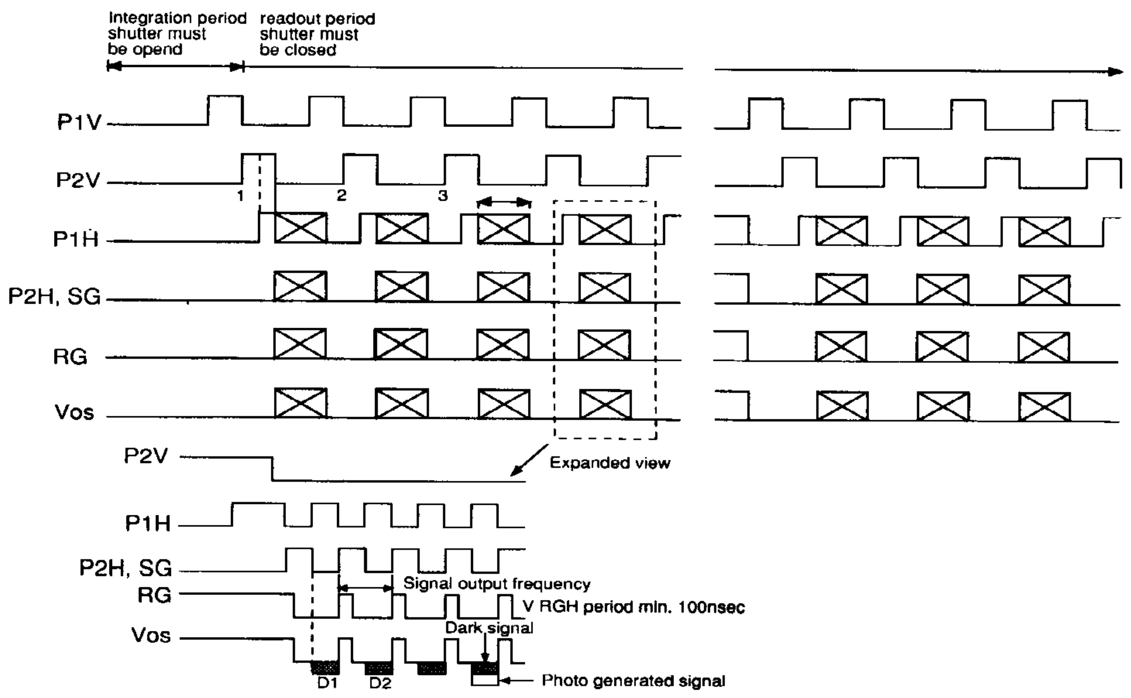


FFT-CCD Image Sensors for Scientific Applications S5469 Series

■Timing Chart (Line Binning)

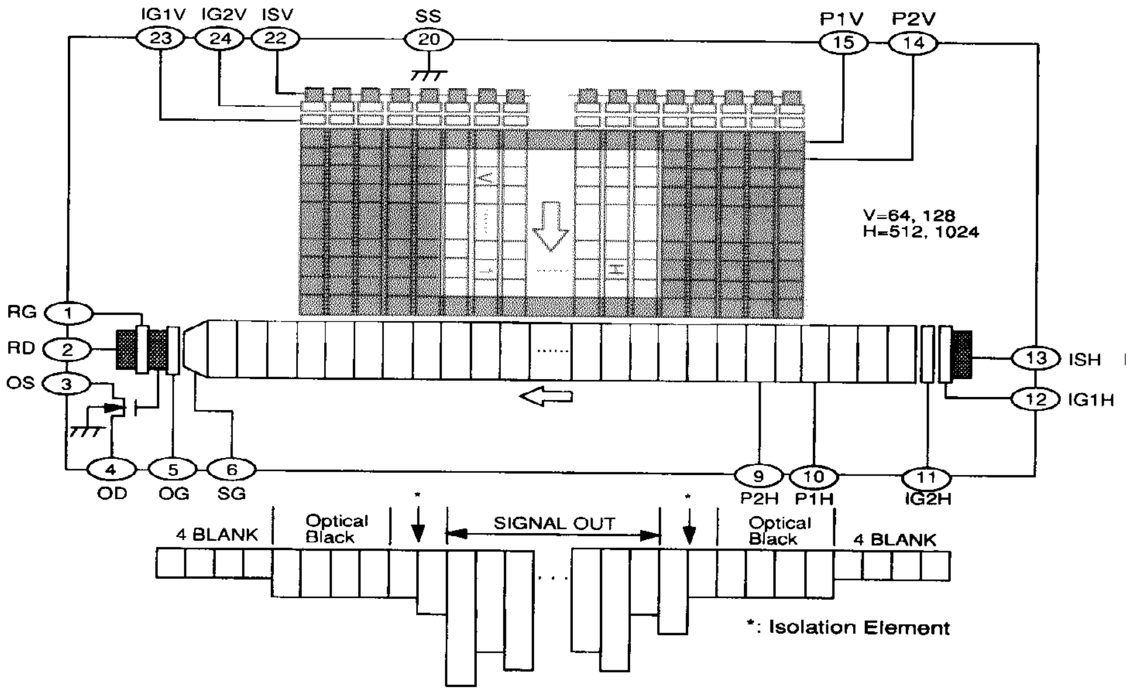


■Timing Chart (Area Scanning)



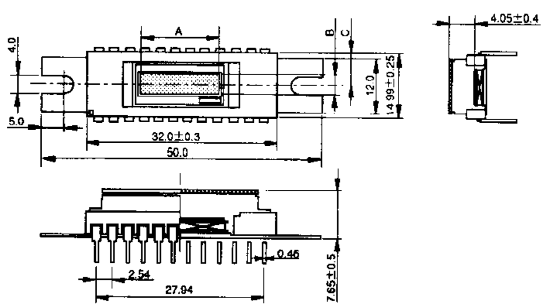
HAMAMATSU

Device Structure, Line Output Format

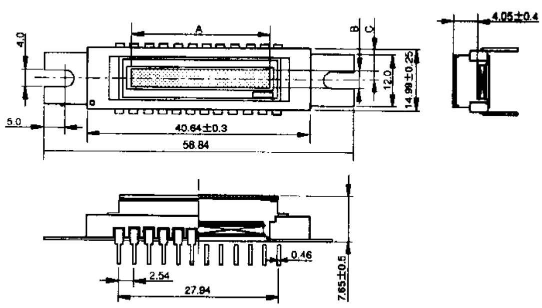


Dimensional Outlines (Unit : mm)

S5469-0906, 0907



S5469-1006, 1007



Sensors	Image Area (mm)		
	A	B	C
S5469-0906	12.288(H)	1.536(V)	7.495
S5469-0907	12.288(H)	3.072(V)	7.235
S5469-1006	24.576(H)	1.536(V)	7.495
S5469-1007	24.576(H)	3.072(V)	7.235

FFT-CCD Image Sensors for Scientific Applications S5469 Series

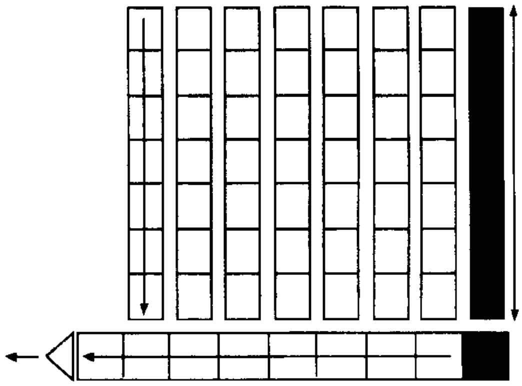
appendix 1:

FFT-CCD

(Full-Frame-Transfer Charge-Coupled-Device)

In FFT-CCD operation, signal charged generated inside silicon by input of light or X-ray are directly stored in the vertical CCD transfer line and are read out sequentially.Compared to the interline CCD having separate signal charge storage lines and charge transfer lines, the FFT-CCD offers advantages of 100% open area ratio, no insensitive regions and low readout noise.

Line Binning



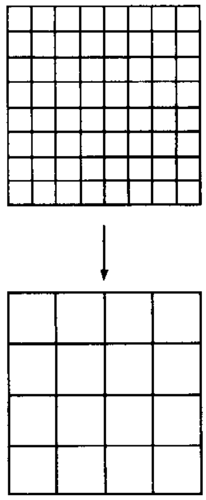
appendix 2:

Binning Operation

Line Binning Operation

By temporally transferring all the pixel signals in the vertical direction to the horizontal register, the signal charges are added and read out sequentially. This enables the signal charge in the vertical direction to be processed as one pixel, thereby allowing the same signal-charge processing as for linear image sensors. This is referred to as the line binning operation.

Pixel Binning
2×2 Summing Operation



Pixel Binning Operation

In addition to two-dimensional readout, using the summing gate and also controlling the vertical and horizontal transfer, this operation permits the addition of several frames of signal charges in the vertical and horizontal directions.This means that the apparent pixel size becomes larger and sensitivity improves accordingly. This operation is called pixel binning.

appendix 3:

MPP Mode

(Multi Pinned Phase)

As shown in the dark signal versus temperature characteristics, operating a CCD under specific drive conditions can reduce the dark signal generated within the storage time. This results in a long signal charge integration time.

reference:

J.Jenesick, T. Elliott, G. Fraschetti, S. Collins, "Charge-Coupled Device Pinning Technologies", SPIE Vol. 1071 Optical Sensors and Electronic Photography (1989), 153-169

HAMAMATSU

HAMAMATSU PHOTONICS K.K., Solid State Division
1126-1 Ichino-cho, Hamamatsu City, 435 Japan, Telephone:053-434-3311, Fax:053-434-5184, Telex:4225-185

U.S.A.: Hamamatsu Corporation:360 Foothill Road,P.O.Box 6910, Bridgewater, N.J. 08807-0910, Telephone:908-231-0960, Fax:908-231-1218
Germany: Hamamatsu Photonics Deutschland GmbH: Arzbergerstr. 10, D-82211 Herrsching am Ammersee, Telephone:49-8152-375-0, Fax:49-8152-2658
France: Hamamatsu Photonics France:Zone Orlytech-BT. 523 3, Allee du Cdt- Mouchotte, Paray Vieile Poste 91781Wissous Cedex, France Telephone:33-(1)49 75 56 80, Fax:33-(1)49 75 56 87
United Kingdom: Hamamatsu Photonics UK Limited: Lough Point, 2 Gladbeck Way, Windmill Hill, Middlesex EN2 7JA,England. Telephone:44-1-367-3560, Fax:44-1-367-6384
North Europe: Hamamatsu Photonics Norden AB: Färögatan 7, 164 40 Kista, Sweden, Telephone:46-8-703-29-50, Fax:46-8-750-58-95
Italy: Hamamatsu Photonics Italia S.R.L.: Via Monte Grappa 30 20020 Arese (Milano) Italy, Telephone:39-(2)935 81 733, Fax:39-(2)935 81 741
Spain: Hamamatsu Photonics Espana: Calle Sabadell,41-08191-Rubi, Barcelona, Spain, Telephone:34-3-699-6553, Fax:34-3-588-1966

Apr. 94
Printed in Japan
(1000) E

National Instruments Lab-PC+ Specifications

This appendix lists the specifications of the Lab-PC+. These specifications are typical at 25°C unless otherwise stated. The operating temperature range is 0° to 70°C.

Analog Input

Number of input channels	8 single-ended, 4 differential
Analog resolution	12 bits, 1 in 4,096
Relative accuracy (nonlinearity)	+-1.5 LSB maximum
Nonlinearity + quantization error (see explanation of specifications)	0.5 LSB typical
Differential nonlinearity	+-l LSB maximum (no missing codes) 0.5 LSB typical
Analog input range	5 V or 0 to +10 V, jumper-selectable
Common-mode ranges	
+-5 V analog input range (bipolar)	5 V[4]
0 to 10 V analog input range (unipolar)	0 to 10 V
Common-mode rejection at 60 Hz	75 dB typical at gain = 1 80 dB typical at gain = 100
Input signal gain	1, 2, 5, 10, 20, 50, 100, software-selectable
Measurement (gain) accuracy	
gain = 1 0.04% maximum	0.025% typical
Offset error	
(calibration performed at gain = 1)	trimmable to zero
Offset adjustment range, minimum	
Unipolar or bipolar ranges	+-47 LSB
Gain adjustment range, minimum	
Unipolar or bipolar ranges	93 LSB
System noise	0.3 LSB rms for gain = 1 0.6 LSB rms for gain = 100
Temperature coefficients	
Gain error	+-50 ppm/°C
Offset error	450 µV/°C + 10 µV/°C * gain
Input bias current	150 pA
Input impedance	0.1 Gohm in parallel with 45 pF

[4] 72 dB at 60 Hz for +-5 V common-mode range; 75 dB at 60 Hz for -2.5 to +-5 V common-mode range.

Input protection	+-45 V on all inputs (not ground)

Explanation of Analog Input Specif cations

Relative accuracy is a measure of the linearity of an ADC. However, relative accuracy is a tighter specification than a nonlinearity specification. Relative accuracy indicates the maximum deviation from a straight line for the analog input-to-digital output transfer curve. If an ADC has been calibrated perfectly, then this straight line is the ideal transfer function, and the relative accuracy specification indicates the worst deviation from the ideal that the ADC permits.

A relative accuracy specification of +-lLSB is roughly equivalent to (but not the same as) a +-1/2LSB nonlinearity or integral nonlinearity specification because relative accuracy encompasses both nonlinearity and variable quantization uncertainty, a quantity often mistakenly assumed to be exactly +-1/2LSB. Although quantization uncertainty is ideally +-1/2LSB, it can be different for each possible digital code and is actually the analog width of each code. Thus, it is more specific to use relative accuracy as a measure of linearity than it is to use what is normally called nonlinearity, because relative accuracy ensures that the sum of quantization uncertainty and A/D conversion error does not exceed a given amount.

Integral nonlinearity in an ADC is an often ill-defined specification that is supposed to indicate a converter's overall A/D transfer linearity. The manufacturers of the ADC chips used by National Instruments specify their integral nonlinearity by stating that the analog center of any code will not deviate from a straight line by more than +-1/2 LSB. This specification is misleading because, although the center of a particularly wide code may be found within +-1/2LSB of the ideal, one of its edges may be well beyond +-1 LSB; thus, the ADC would have a relative accuracy of that amount. National Instruments tests its boards to ensure that they meet all three linearity specifications defined in this appendix; specifications for integral nonlinearity are included primarily to maintain compatibility with a convention of specifications used by other board manufacturers. Relative accuracy, however, is much more useful.

Differential nonlinearity is a measure of deviation of code widths from their theoretical value of lLSB. The width of a given code is the size of the range of analog values that can be input to produce that code, ideally 1 LSB. A specification of +-1LSB differential nonlinearity ensures that no code has a width of 0 LSBs (that is, no missing codes) and that no code width exceeds 2 LSBs.

System noise is the amount of noise seen by the ADC when there is no signal present at the input of the board. The amount of noise that is reported directly (without any analysis) by the ADC is not necessarily the amount of real noise present in the system, unless the noise is >=0.5LSB rms. Noise that is less than this magnitude produces varying amounts of flicker, and the amount of flicker seen is a function of how near the real mean of the noise is to a code transition. If the mean is near or at a transition between codes, the ADC flickers evenly between the two codes, and the noise is seen as very nearly 0.5 LSB. If the mean is near the center of a code and the noise is relatively small, very little or no flicker is seen, and the noise is reported by the ADC asnearly 0 LSB. From the relationship between the mean of the noise and the measured rms magnitude of the noise, the character of the noise can be determined. National Instruments has determined that the character of the noise in the Lab-PC+ is fairly Gaussian, and so the noise specifications given are the amounts of pure Gaussian noise required to produce our readings.

Analog Data Acquisition Rates

Maximum sample rate	83.3 kHz (gain = 1) 71.4 kHz (all other gains)

Analog bandwidth (-3 dB)	400 kHz (gain = 1) 40 kHz (gain = 100)
Maximum multichannel scan rate	62.5 kHz (gain < 50) typical, 55.5 kHz worst case
(To settle within +-1.0 LSB)	20 kHz (gain = 100)

Analog Output

Number of output channels	Two single-ended
Analog resolution	12 bits, 1 part in 4,096
Relative accuracy (nonlinearity)	0.50 LSB maximum +-0.25 LSB typical
Differential nonlinearity	0.75 LSB maximum (monotonic over temperature) 0.25 LSB typical
Offset adjustment range, minimum	+-37 mV
Gain adjustment range, minimum	+-100 mV
Output voltage ranges	0 to +10 V, unipolar mode 5 V, bipolar mode, software-selectable
Current drive capability	+-2 mA
Output settling time	6 µsec for 10 V step to 0.012%
Output slew rate	10 V/µsec
Temperature coefficients	
Gain error	+10 ppm/°C
Voltage offset	+60 µV/°C
Output impedance	0.2 ohm typical

Explanation of Analog Output Specifications

Relative accuracy in a D/A system is the same as nonlinearity, because no uncertainty is added due to code width. Unlike an ADC, every digital code in a D/A system represents a specific analog value rather than a range of values. The relative accuracy of the system is therefore limited to the worst-case deviation from the ideal correspondence (a straight line), excepting noise. If a D/A system has been calibrated perfectly, then the relative accuracy specification reflects its worst-case absolute error.

Differential nonlinearity in a D/A system is a measure of deviation of code width from 1 LSB. In this case, code width is the difference between the analog values produced by consecutive digital codes. A specification of +-1 LSB differential nonlinearity ensures that the code width is always greater than 0 LSBs (guaranteeing monotonicity) and is always less than 2 LSBs.

Digital I/O

Compatibility	TTL-compatible
Configuration	Three 8-bit ports (uses 8255A PPI)

Input logic low voltage	0.8 V maximum
Input logic high voltage	2.0 V minimum
Output logic low voltage at output current = 1.7 mA	0.45 V maximum
Output logic high voltage at output current = -200 IlA	2.4 V minimum
Input load current O<=Vin<=5 V	+-10 µA maximum
Darlington drive current (Ports B and C only) R_{EXT} = 750 ohm; V_{EXT} = 1.5 V	-1.0 mA minimum -4.0 mA maximum

Timing I/O

Configuration	Three 16-bit counter/timers (uses two 8253 STCs)
Compatibility	TTL-compatible inputs and outputs. Counter gate and clock inputs are pulled up with 4.7 kQ resistors onboard.
Input logic low voltage	0.8 V maximum
Input logic high voltage	2.2 V minimum
Output logic low voltage at output current = 1.6 mA	0.45 V maximum
Output logic high voltage at output current = -150 µA	2.4 V minimum

Specifications

Input load current 0 <= Vin < =5 V	[(5.0 - Vin) /10] mA
Input capacitance at 1 MHz	10 pF maximum
Base clock frequency	2 MHz +-0.01%

Power Requirements (from PC)

Power consumption	
+5 VDC	150 mA*
+12 VDC	70 mA
-12 VDC	45 mA

* Additional current up to 1 A can be drawn by the user through the 50-pin I/O connector.

Physical

Board dimensions	6.5 by 3.9 in.
I/O connector	50-pin keyed male ribbon cable connector

Operating Environment

Component temperature	0° to 70° C
Relative humidity	5% to 90% noncondensing

Storage Environment

Temperature	-55° to 150° C
Relative humidity	5% to 90% noncondensing

I/O Connector

This appendix contains the pinout and signal names for the I/O connector on the Lab-PC+. Figure B-1 shows the Lab-PC+ 50-pin I/O connector.

ACH0	1	2	ACH1
ACH2	3	4	ACH3
ACH4	5	6	ACH5
ACH6	7	8	ACH7
AISENSE/AIGND	9	10	DAC0 OUT
AGND	11	12	DAC1 OUT
DGND	13	14	PA0
PA1	15	16	PA2
PA3	17	18	PA4
PA5	19	20	PA6
PA7	21	22	PB0
PB1	23	24	PB2
PB3	25	26	PB4
PB5	27	28	PB6
PB7	29	30	PC0
PC1	31	32	PC2
PC3	33	34	PC4
PC5	35	36	PC6
PC7	37	38	EXTTRIG
EXTUPDATE*	39	40	EXTCONV*
OUTB0	41	42	GATB0
COUTB1	43	44	GATB1
CCLKB1	45	46	OUTB2
GATB2	47	48	CLKB2
+5 V	49	50	DGND

Figure B-1. Lab-PC+ I/O Connector Pin Assignments

Detailed signal specifications are included in Chapter 2, *Configuration and Installation*, and Appendix A, *Specifications*.

© National Instruments Corporation *B-1* *Lab-PC+ User Manual*

Theory of Operation

This chapter contains a functional overview of the Lab-PC+ and explains the operation of each functional unit making up the Lab-PC+. This chapter also explains the basic operation of the Lab-PC+ circuitry.

Functional Overview

The block diagram in Figure 3-1 shows a functional overview of the Lab-PC+ board.

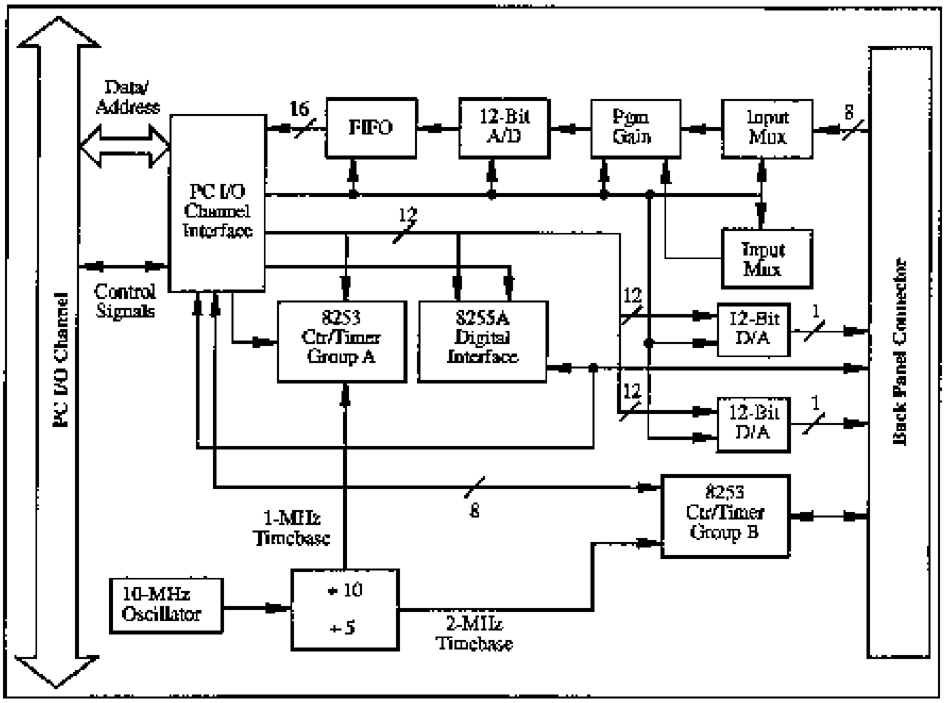
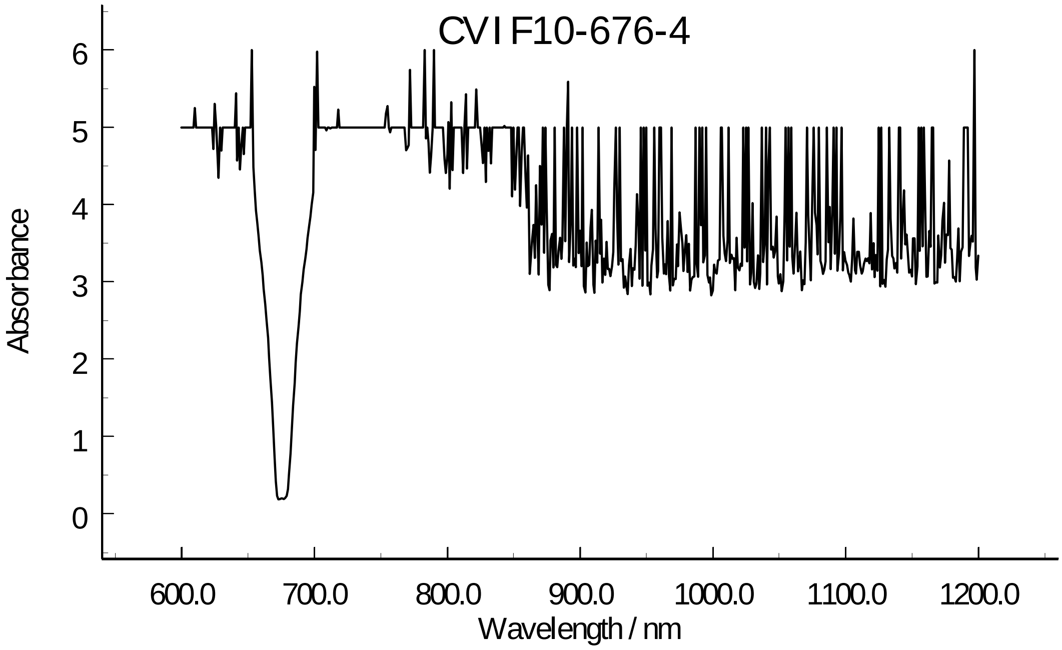


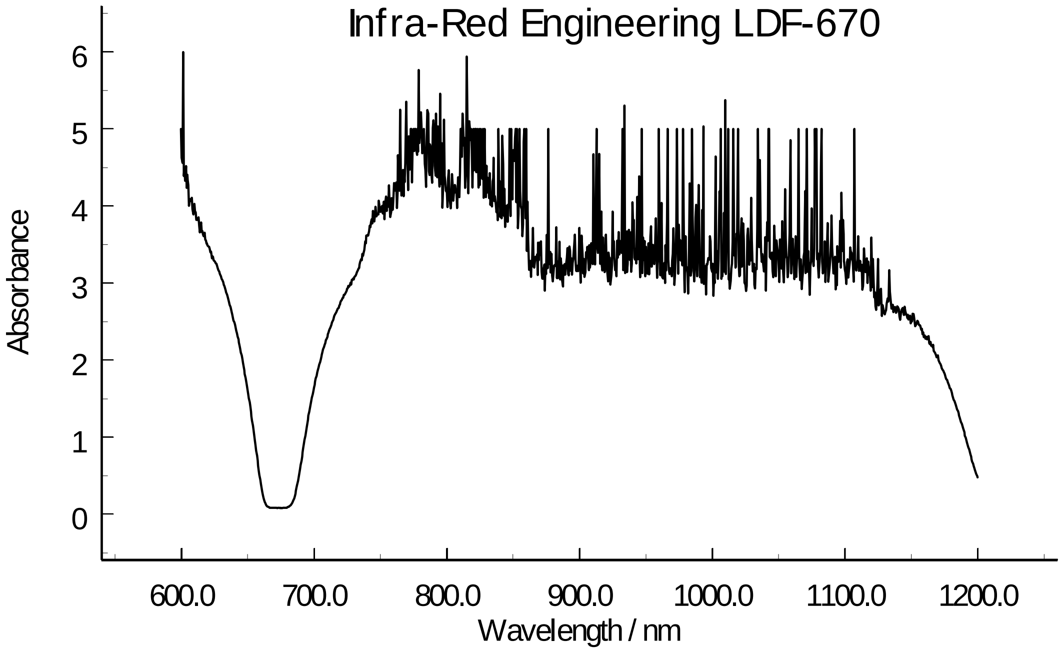
Figure 3-1. Lab-PC+ Block Diagram

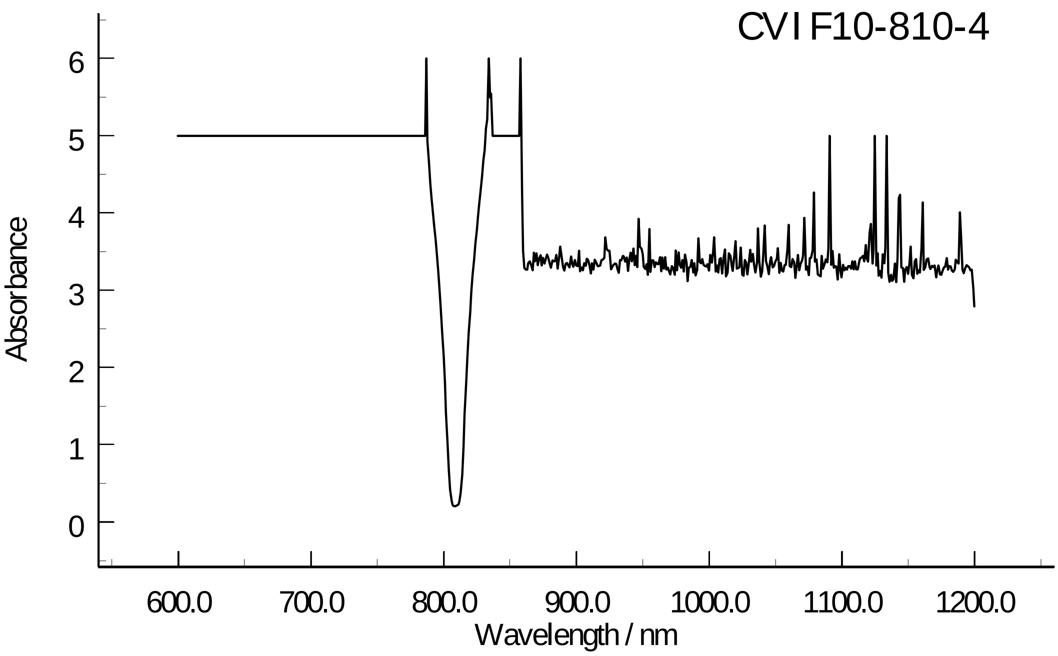
Appendix D: Measured Optical Component Performance

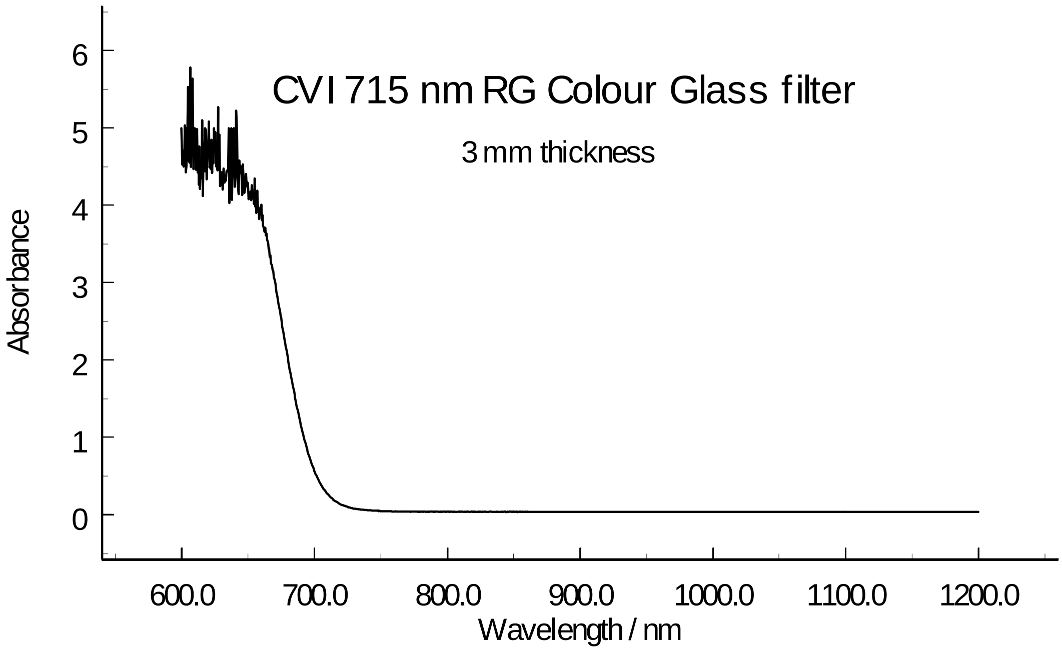
In this appendix, the measured optical absorbances of the filters mentioned in this thesis are presented. They were measured on a Perkin-Elmer Lamda 9 spectrophotometer, with a scan speed of 60 nm·m^{-1}, response time of 2 seconds, NIR sensitivity of 2, and a slit width of 0.5 nm.

The instrumental discontinuities evident at around 850 nm are irrelevant, as the traces were recorded only to ensure that the filter blocking action was constant over the light range detected by silicon detectors.




Infra-Red Engineering LDF-670
Absorbance
0
1
2
3
4
5
6
600.0
700.0
800.0
900.0
1000.0
1100.0
1200.0
Wavelength / nm


CVI F10-810-4
Absorbance
0
1
2
3
4
5
6
600.0
700.0
800.0
900.0
1000.0
1100.0
1200.0
Wavelength / nm

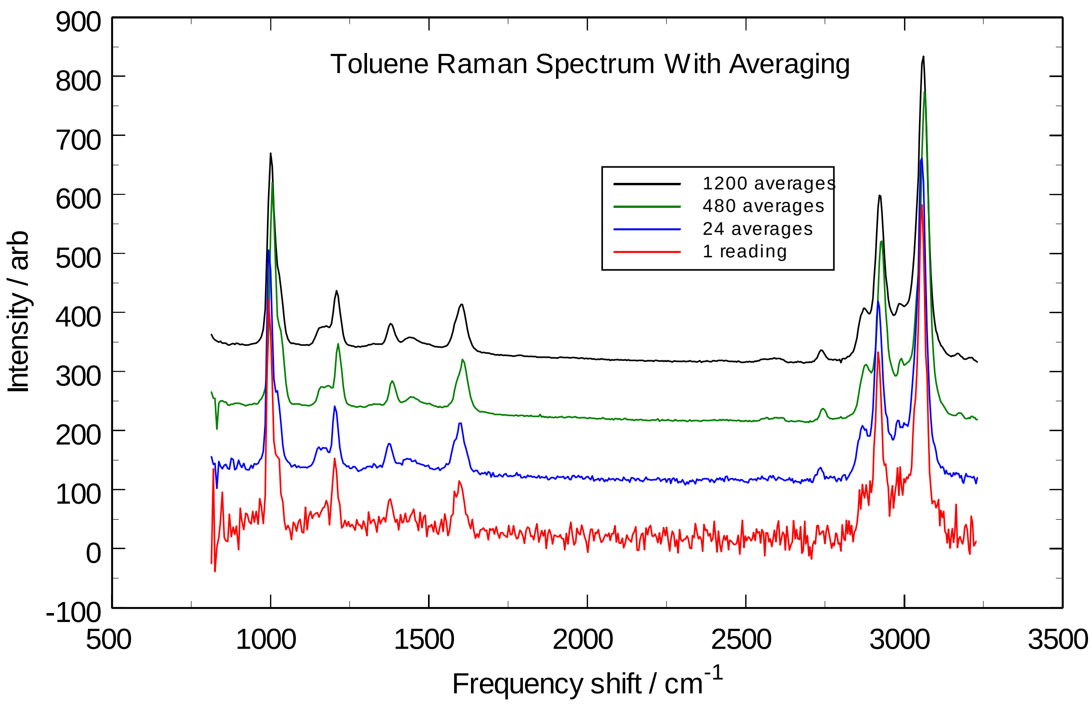

CVI 715 nm RG Colour Glass filter
3 mm thickness
Absorbance
0
1
2
3
4
5
6
600.0
700.0
800.0
900.0
1000.0
1100.0
1200.0
Wavelength / nm

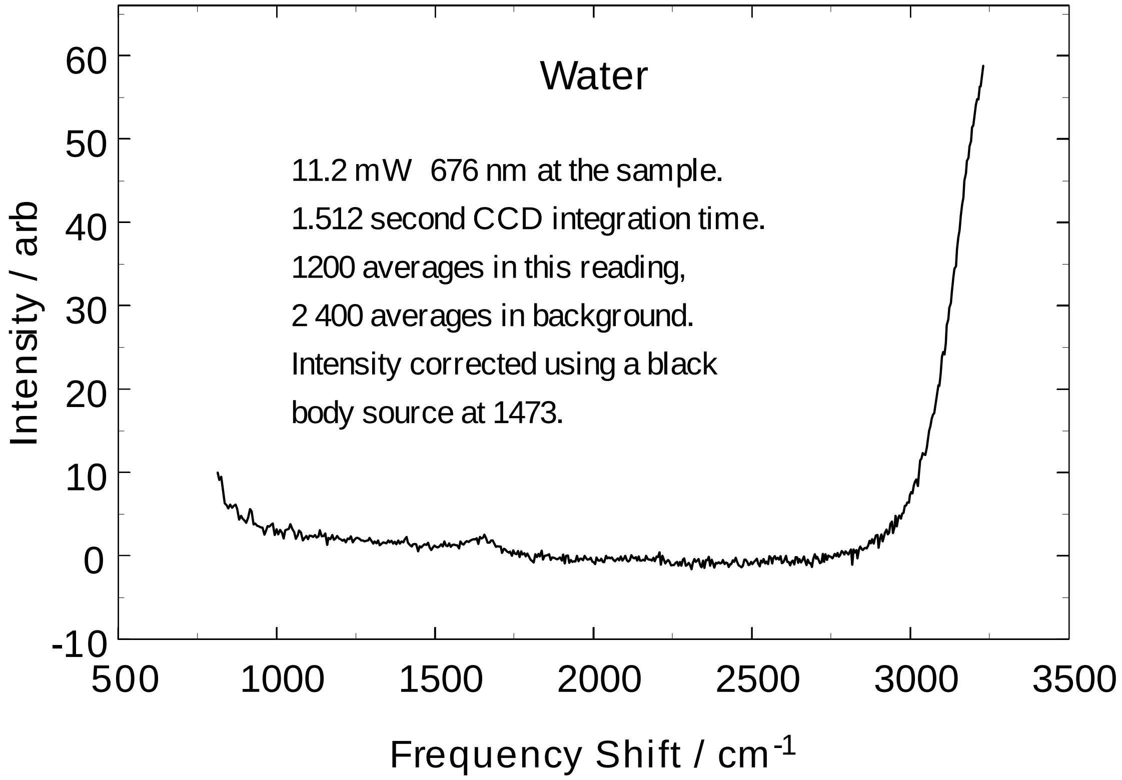
Appendix E : Raman Spectra Measured Using The Equipment Described in Chapter 3

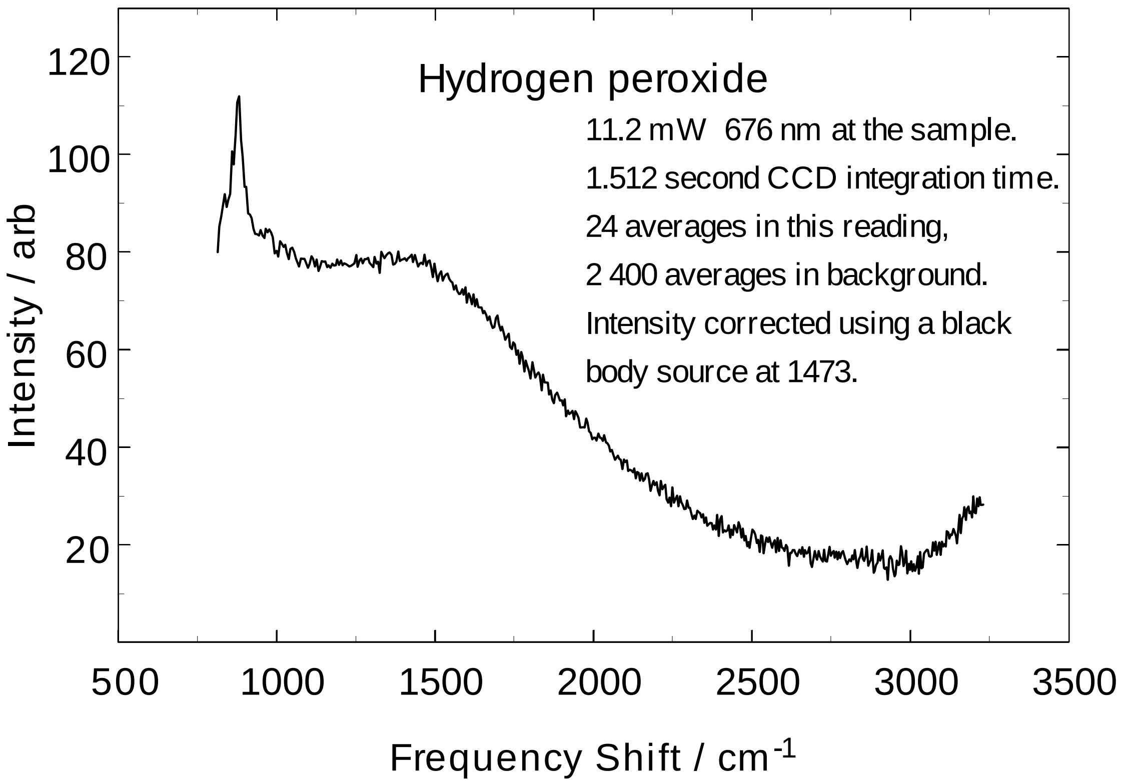
In this appendix, several Raman spectra are presented, measured under the conditions noted in each figure.

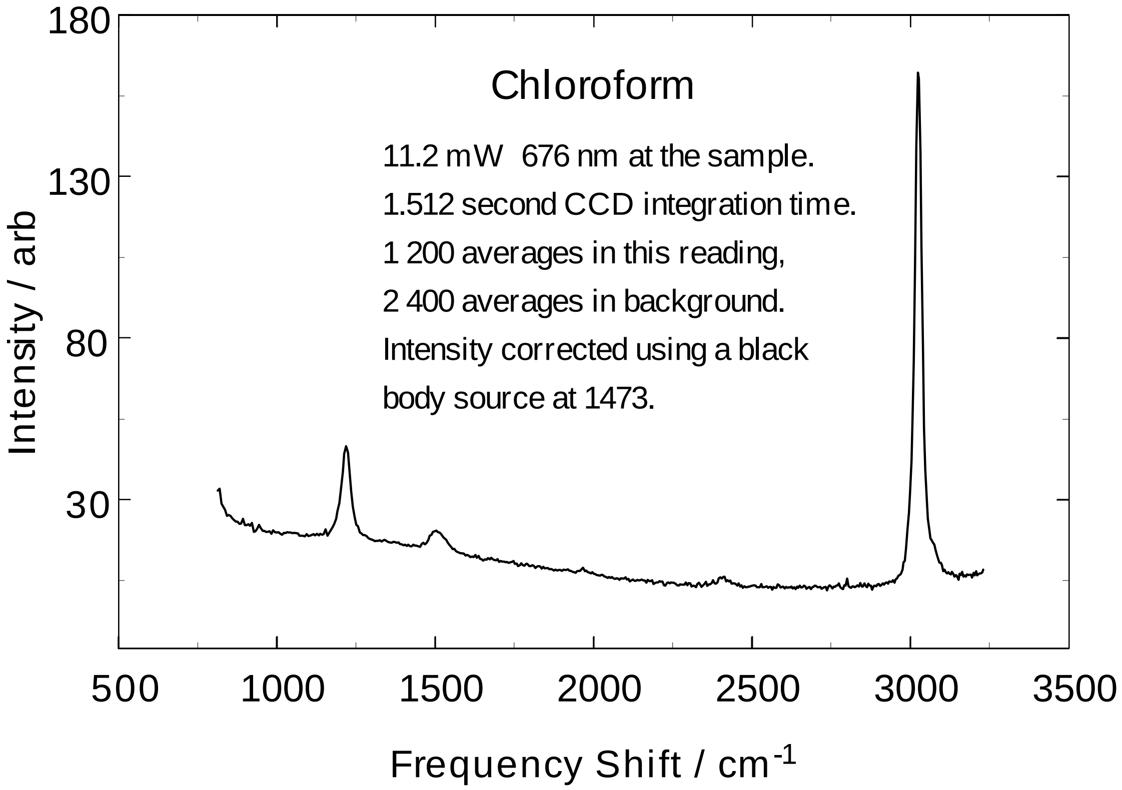
The spectra presented are:

- Toluene (1 - 1 200 readings averaged)
- Water, H_2O
- Hydrogen peroxide, H_2O_2
- Chloroform, $CHCl_3$
- Tetra-chloro ethene, $CCl_2.CCl_2$
- c-Hexane, C_6H_{12}
- n-Hexane, C_6H_{14}
- Iso-propyl alcohol (IPA), $CH_3.CHOH.CH_3$
- Methanol, CH_3OH
- Toluene, $C_6H_5CH_3$
- Xylene, $C_6H_4(CH_3)_2$
- Aspirin tablet
- Paracetamol tablet
- Panacryl Blue 5G dye fluorescence and absorbance

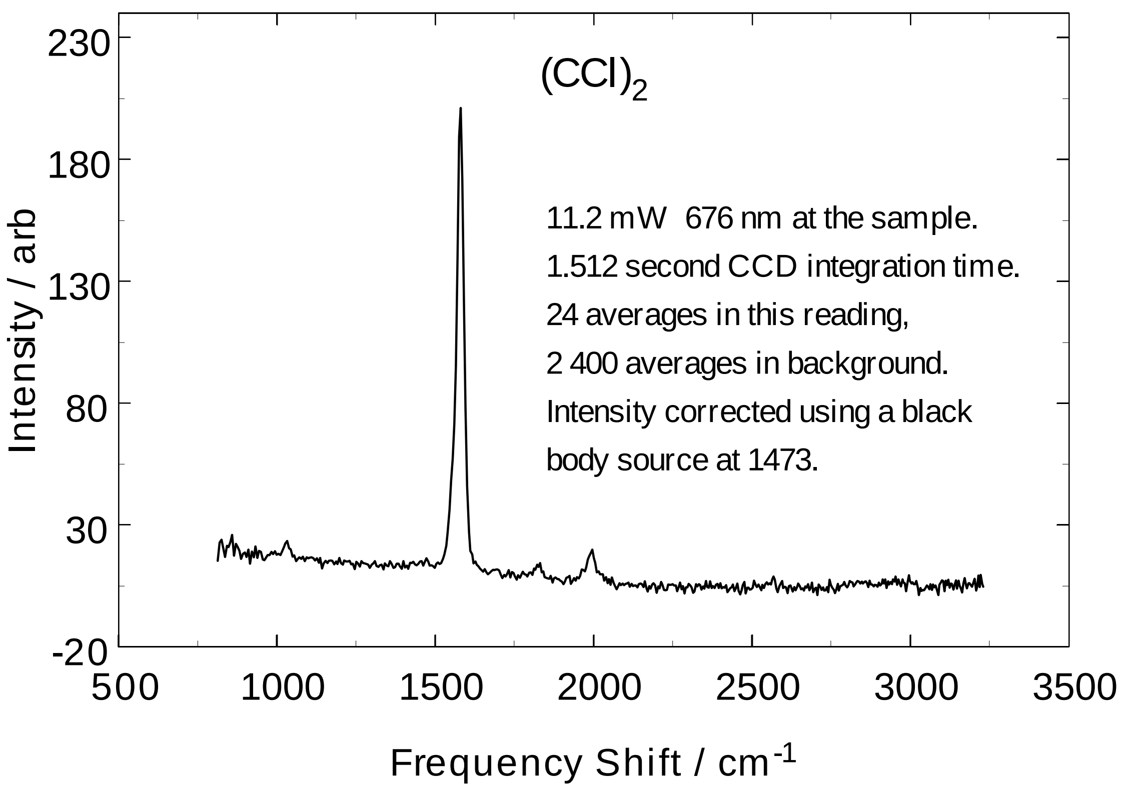



Water
11.2 mW 676 nm at the sample.
1.512 second CCD integration time.
1200 averages in this reading,
2 400 averages in background.
Intensity corrected using a black
body source at 1473.
Intensity / arb
60
50
40
30
20
10
0
-10
500
1000
1500
2000
2500
3000
3500
Frequency Shift / cm-1

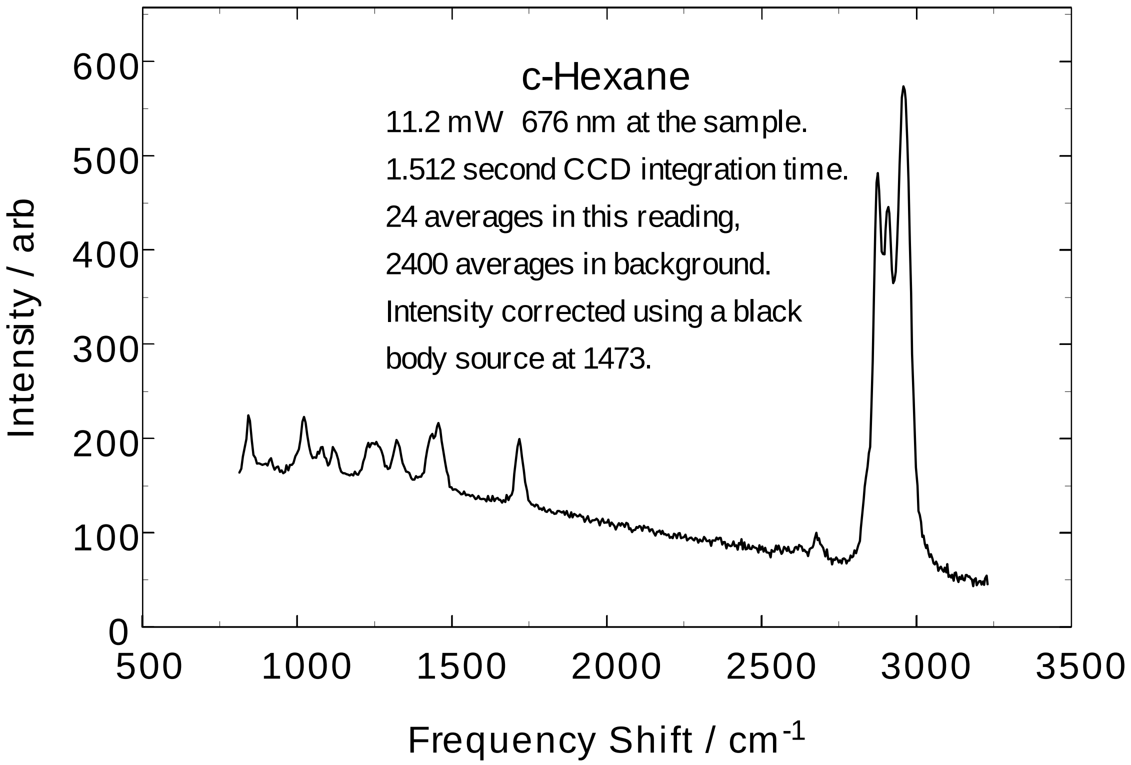

Hydrogen peroxide
11.2 mW 676 nm at the sample.
1.512 second CCD integration time.
24 averages in this reading,
2 400 averages in background.
Intensity corrected using a black
body source at 1473.
Intensity / arb
120
100
80
60
40
20
500
1000
1500
2000
2500
3000
3500
Frequency Shift / cm-1

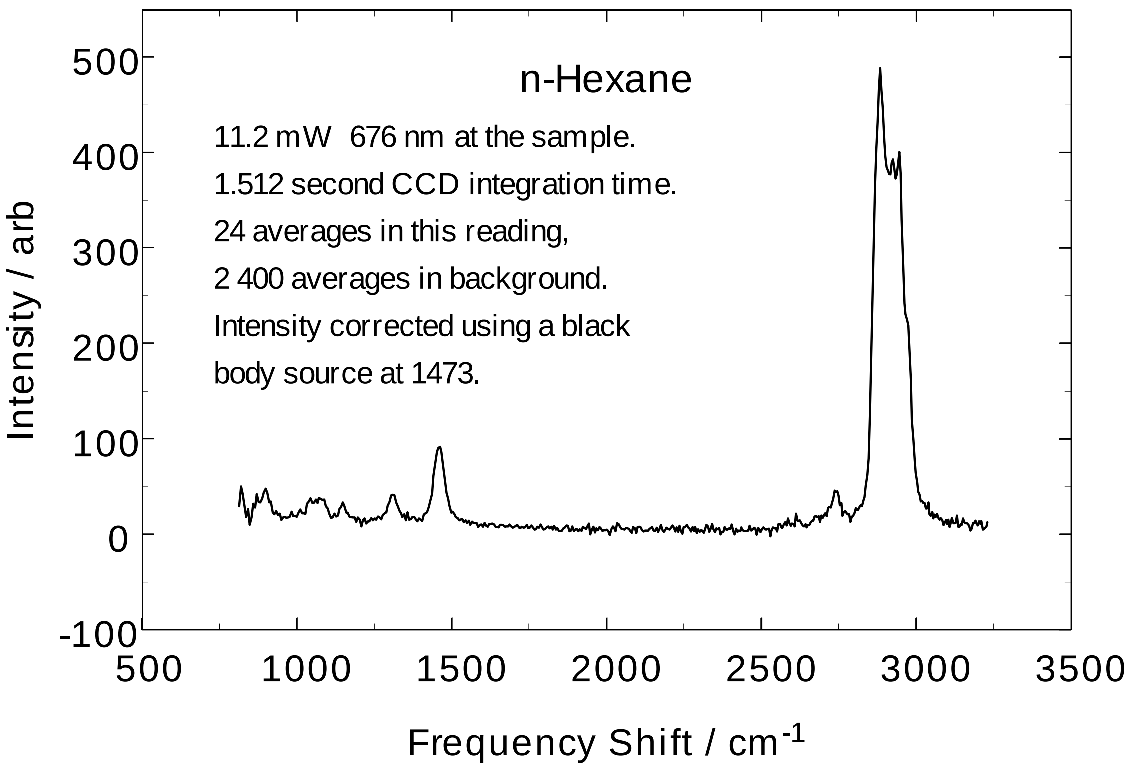


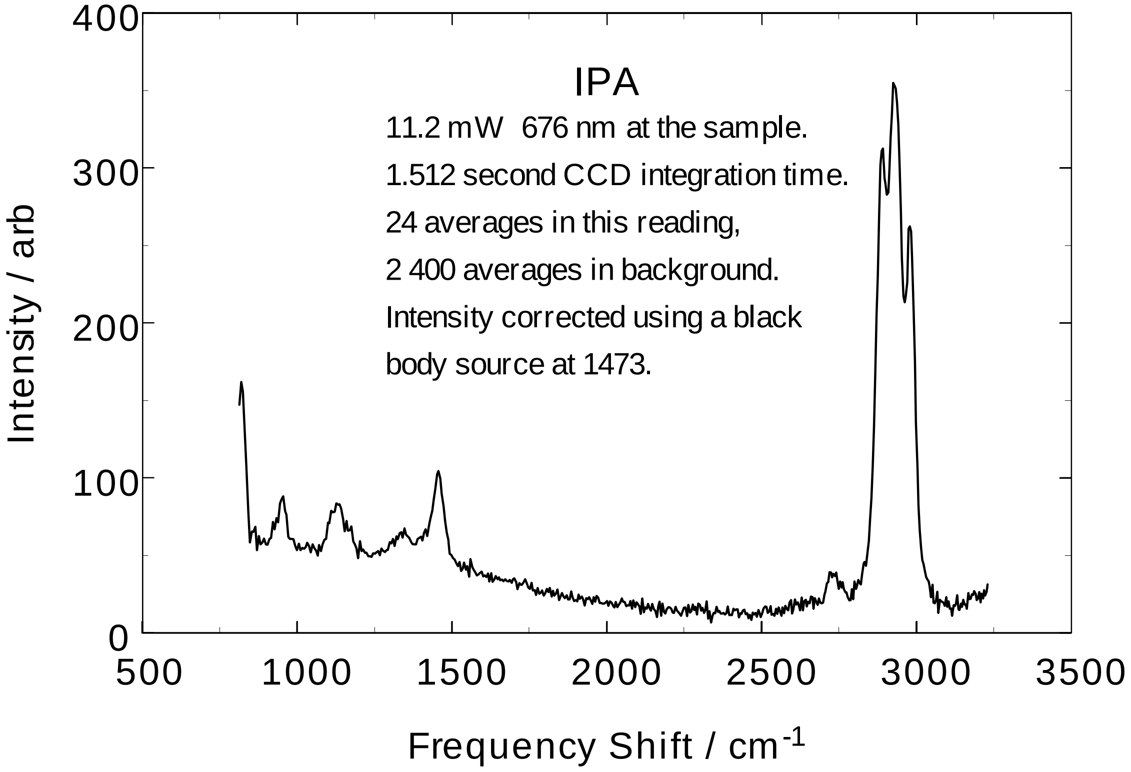
Chloroform
11.2 mW 676 nm at the sample.
1.512 second CCD integration time.
1 200 averages in this reading,
2 400 averages in background.
Intensity corrected using a black
body source at 1473.
Intensity / arb
180
130
80
30
500
1000
1500
2000
2500
3000
3500
Frequency Shift / cm-1

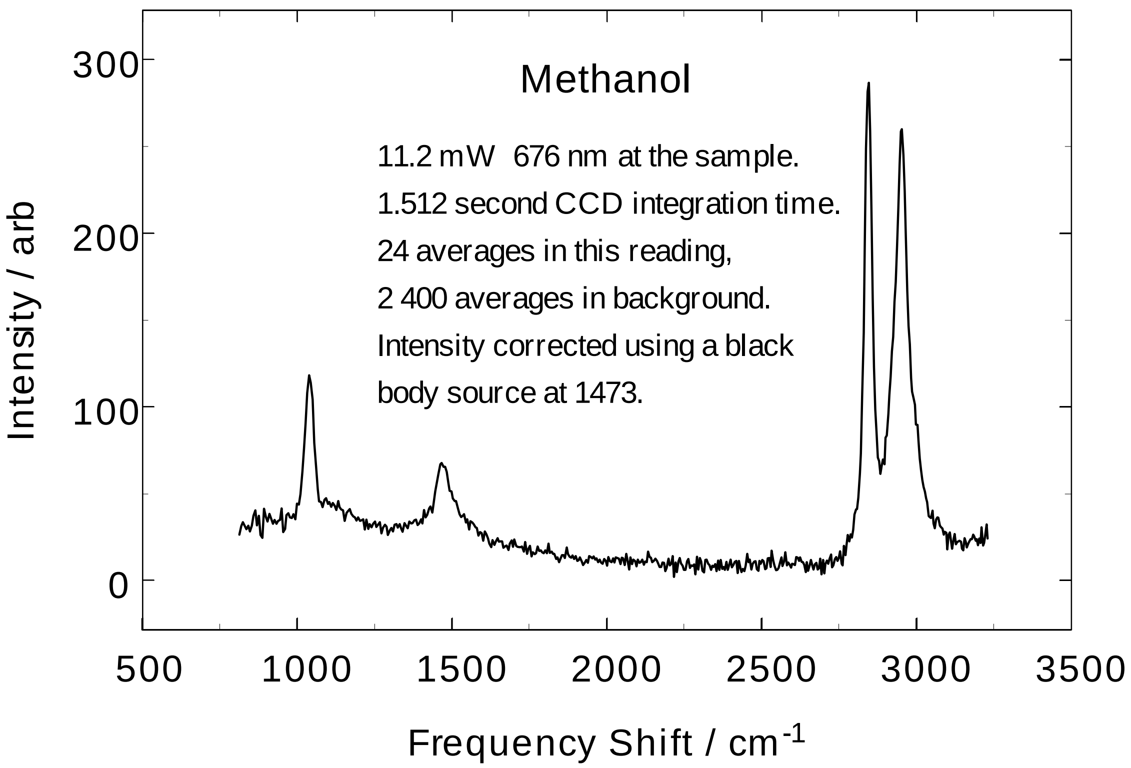


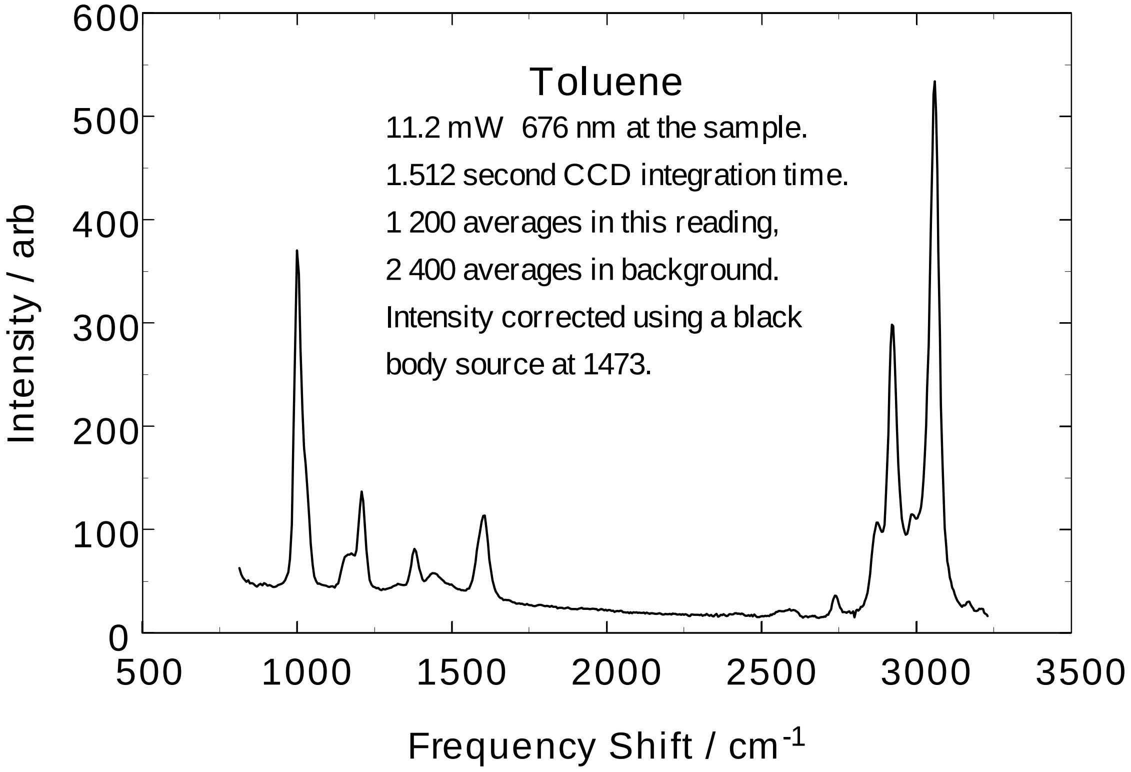
(CCl)2
11.2 mW 676 nm at the sample.
1.512 second CCD integration time.
24 averages in this reading,
2 400 averages in background.
Intensity corrected using a black
body source at 1473.
Intensity / arb
230
180
130
80
30
-20
500
1000
1500
2000
2500
3000
3500
Frequency Shift / cm-1

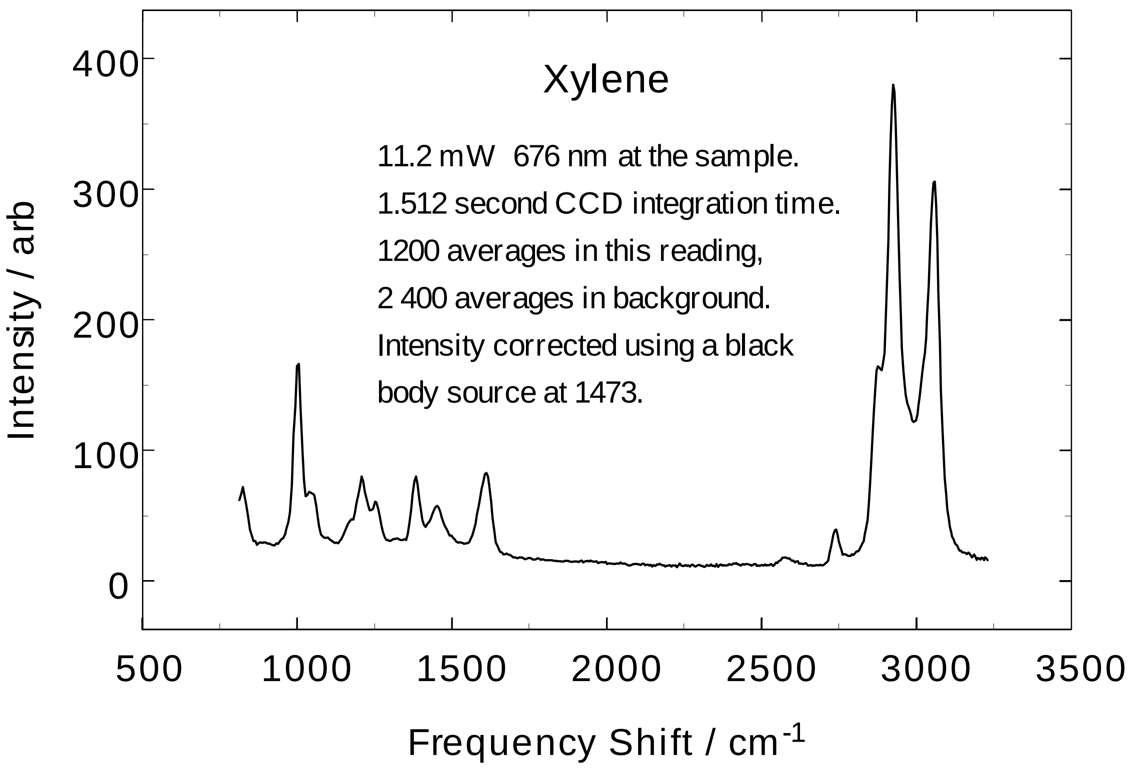

c-Hexane
11.2 mW 676 nm at the sample.
1.512 second CCD integration time.
24 averages in this reading,
2400 averages in background.
Intensity corrected using a black
body source at 1473.
Intensity / arb
0
100
200
300
400
500
600
500
1000
1500
2000
2500
3000
3500
Frequency Shift / cm-1

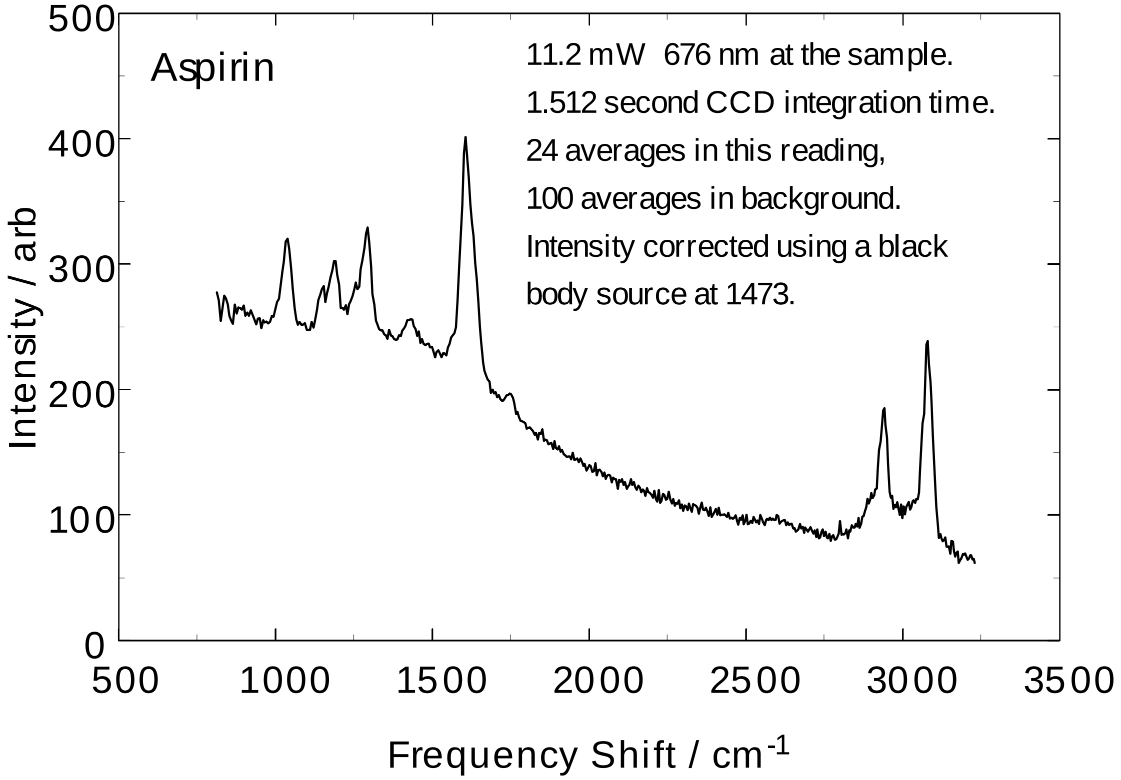

n-Hexane
11.2 mW 676 nm at the sample.
1.512 second CCD integration time.
24 averages in this reading,
2 400 averages in background.
Intensity corrected using a black
body source at 1473.
Intensity / arb
-100
0
100
200
300
400
500
500
1000
1500
2000
2500
3000
3500
Frequency Shift / cm-1

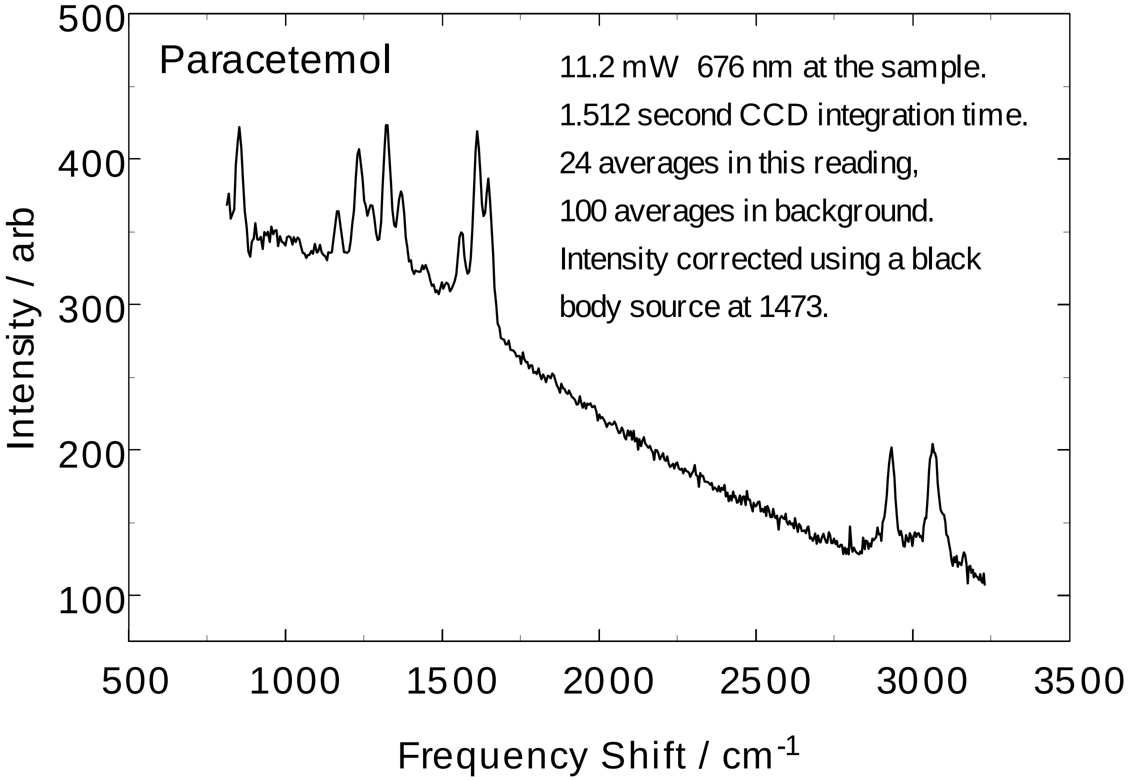

IPA
11.2 mW 676 nm at the sample.
1.512 second CCD integration time.
24 averages in this reading,
2 400 averages in background.
Intensity corrected using a black
body source at 1473.
Intensity / arb
0
100
200
300
400
500
1000
1500
2000
2500
3000
3500
Frequency Shift / cm-1

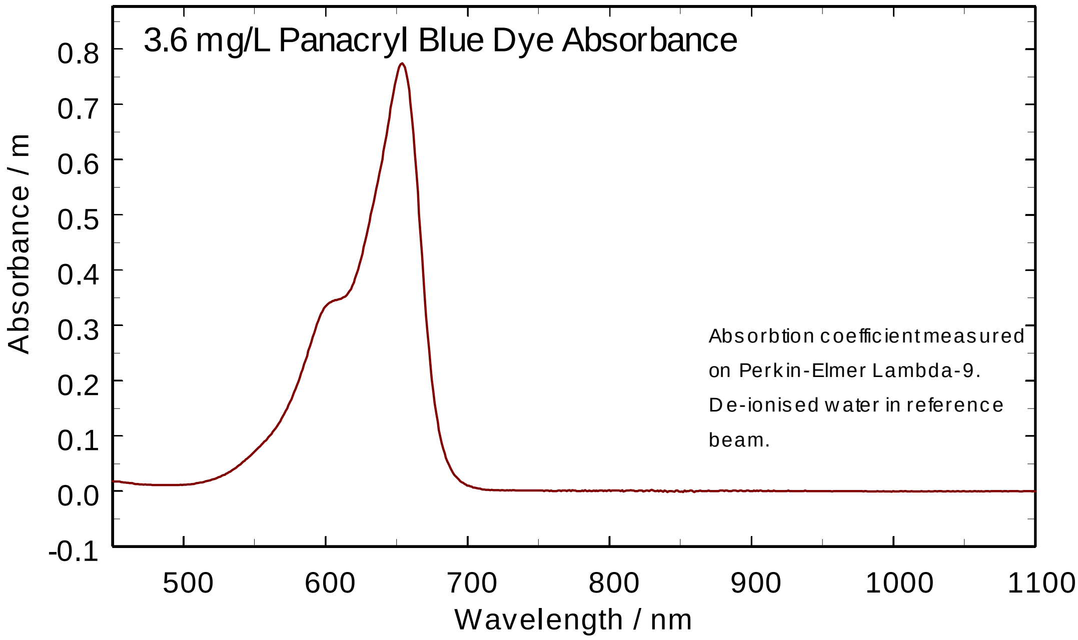

Methanol
11.2 mW 676 nm at the sample.
1.512 second CCD integration time.
24 averages in this reading,
2 400 averages in background.
Intensity corrected using a black
body source at 1473.
Intensity / arb
0
100
200
300
500
1000
1500
2000
2500
3000
3500
Frequency Shift / cm-1

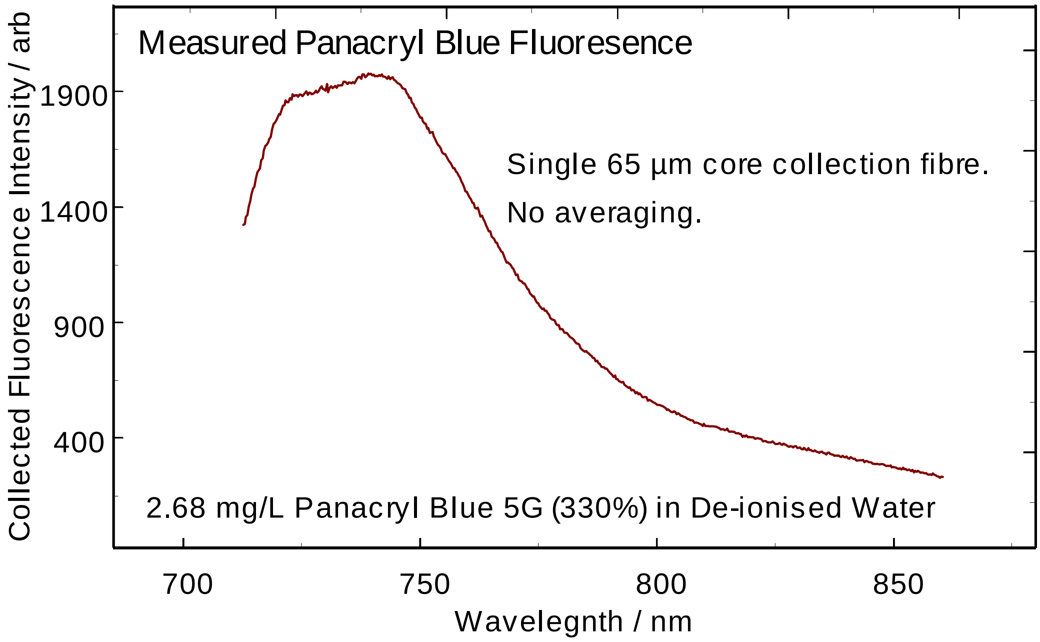

Toluene
11.2 mW 676 nm at the sample.
1.512 second CCD integration time.
1 200 averages in this reading,
2 400 averages in background.
Intensity corrected using a black
body source at 1473.
Intensity / arb
Frequency Shift / cm-1
0
100
200
300
400
500
600
500
1000
1500
2000
2500
3000
3500


Xylene
11.2 mW 676 nm at the sample.
1.512 second CCD integration time.
1200 averages in this reading,
2 400 averages in background.
Intensity corrected using a black
body source at 1473.
Intensity / arb
Frequency Shift / cm-1
0
100
200
300
400
500
1000
1500
2000
2500
3000
3500


Aspirin
11.2 mW 676 nm at the sample.
1.512 second CCD integration time.
24 averages in this reading,
100 averages in background.
Intensity corrected using a black
body source at 1473.
Intensity / arb
0
100
200
300
400
500
500
1000
1500
2000
2500
3000
3500
Frequency Shift / cm-1


Paracetemol
11.2 mW 676 nm at the sample.
1.512 second CCD integration time.
24 averages in this reading,
100 averages in background.
Intensity corrected using a black
body source at 1473.
Intensity / arb
100
200
300
400
500
500
1000
1500
2000
2500
3000
3500
Frequency Shift / cm-1


3.6 mg/L Panacryl Blue Dye Absorbance
Absorbance / m
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
-0.1
500
600
700
800
900
1000
1100
Wavelength / nm
Absorbtion coefficient measured
on Perkin-Elmer Lambda-9.
De-ionised water in reference
beam.


Measured Panacryl Blue Fluoresence
Collected Fluorescence Intensity / arb
1900
1400
900
400
Single 65 µm core collection fibre.
No averaging.
2.68 mg/L Panacryl Blue 5G (330%) in De-ionised Water
700
750
800
850
Wavelegnth / nm

Appendix F: Raman and Differential Thermal Analysis of Teflon AF-1600

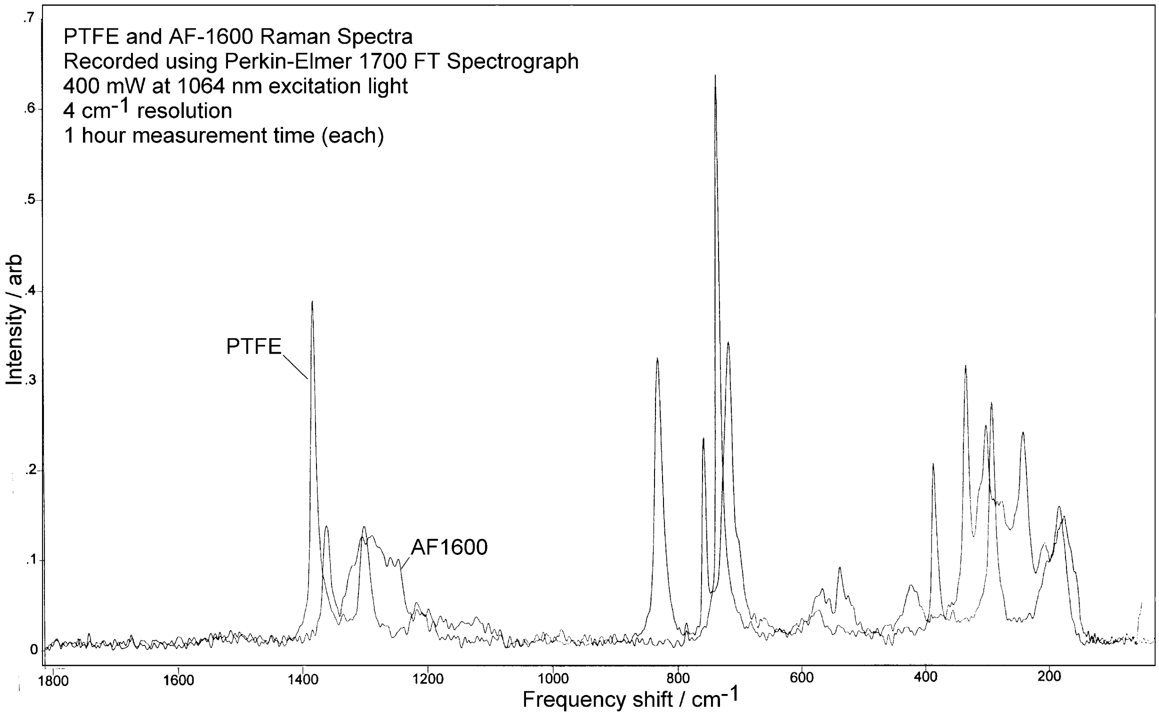


Figure F.1 The Raman spectra of Teflon AF 1600 and PTFE.

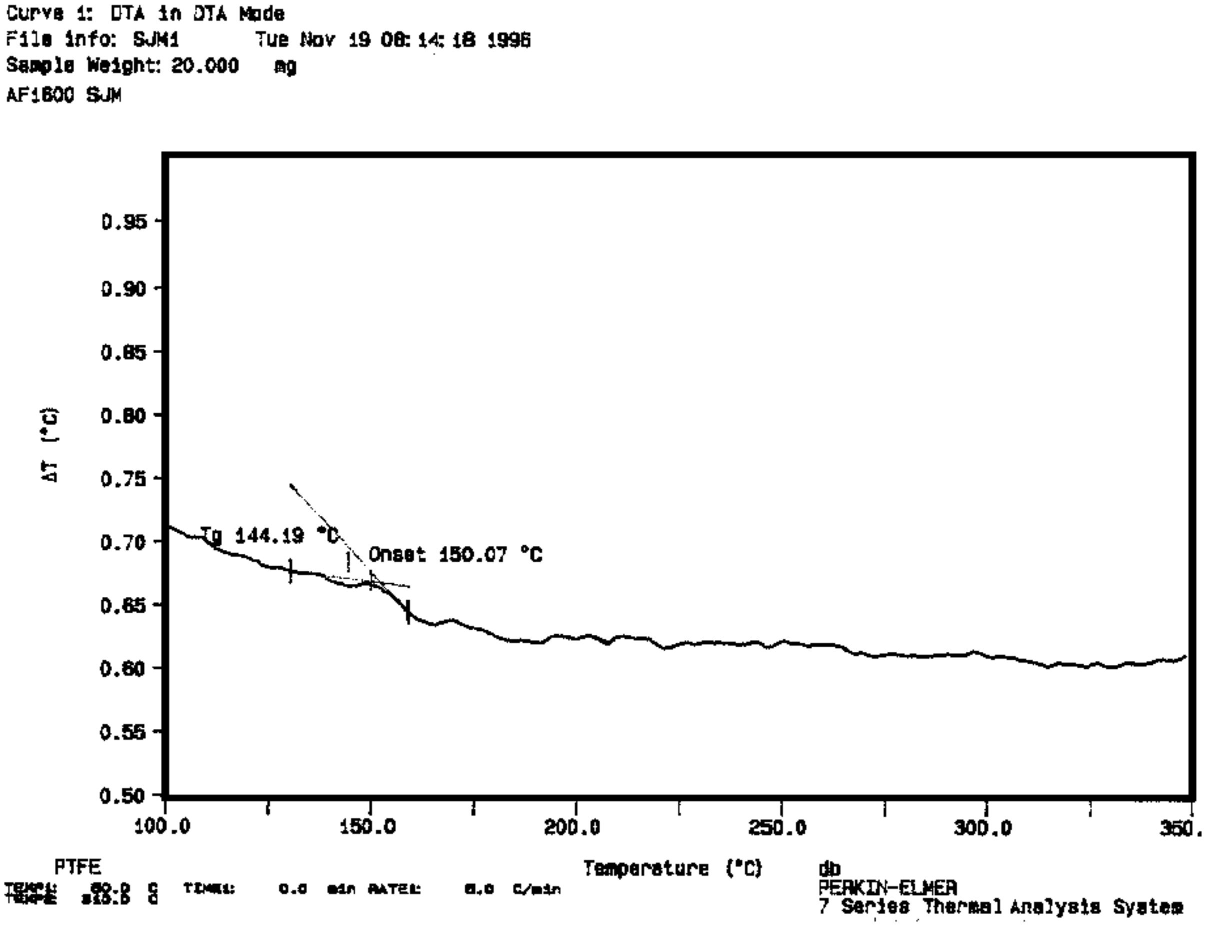


Figure F.2 DTA trace for the Teflon AF granules, as supplied by duPont, before any processing.

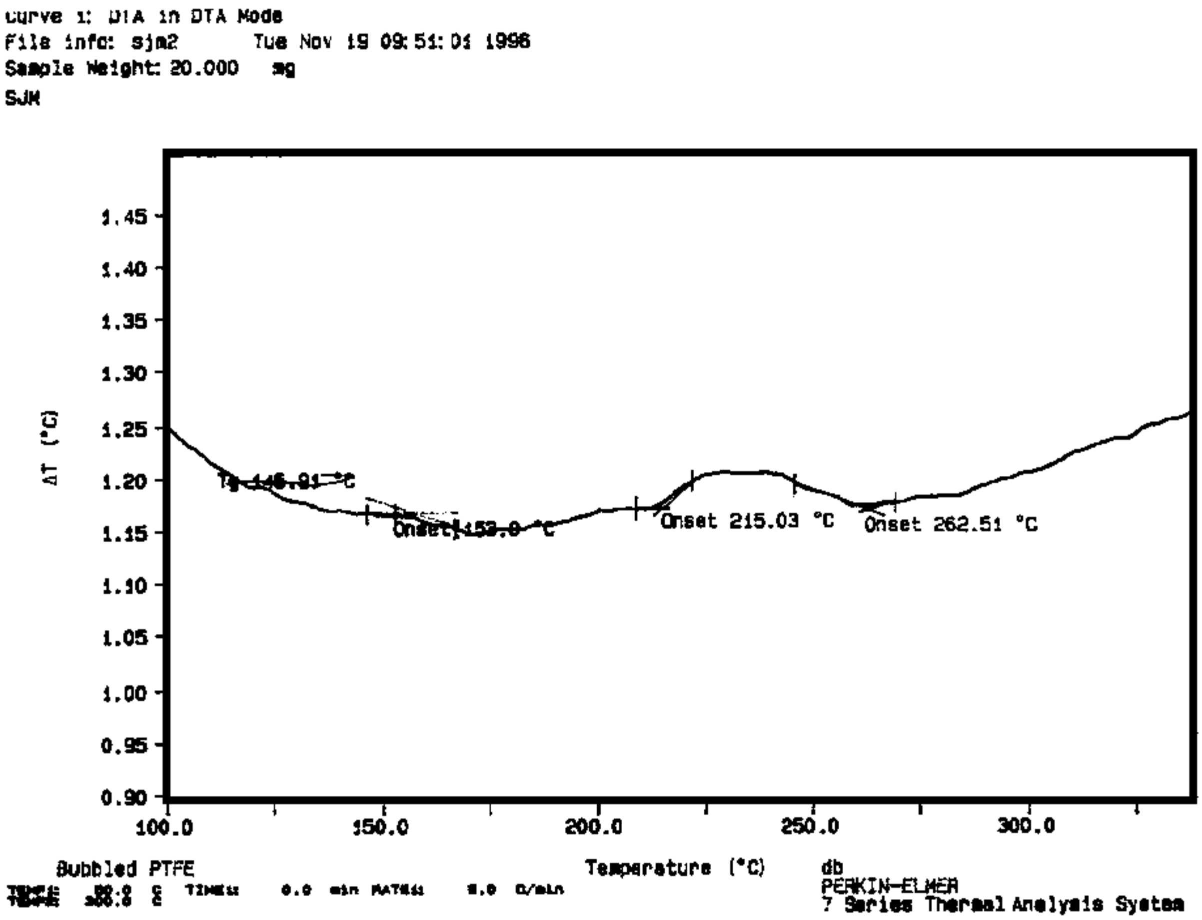


Figure F.3 DTA trace of Teflon AF deposited out of solution in Fluorinert FC75 and baked at 150ºC.