--- TABLE OF CONTENT ---

Content	Page
Nomenclature	i
List of Figures	iv
List of Tables	vi
Abstract	1
Introduction	2
Theory	3
Background	3
Assumptions	5
Paths	6
Surface Layer Effect	12
Dominance Index	14
Model and Computation	16
Results and Discussions	18
Shadow Factor	18
Mie Intensities for Bare and Coated Surfaces	21
Geometry Effects	26
Dominance Analysis	28
Conclusions	30
Acknowledgment	32
References	32

--- APPENDICES ---

Appendix	Page
A. Derivation of Shadow Factor for Bare Surfaces	A1 - A5
B. Derivation of Shadow Factor for Coated Surfaces	B1 - B3
C. Computer Program for Light Scattering from Particles on Reflective Surfaces	
C.1. Guide to Users	C1 - C6
C.2. Program Statements	C7 - C59

--- NOMENCLATURE ---

Symbol	Definition
a	Projected area on which incident light reflects from a surface to a particle or from which scattered light reflects from a surface to a detector
d	Particle diameter
i	One of the paths (A, B1, B2, or B3)
i(1)	Parallel component of the scattered light
i(2)	Perpendicular component of the scattered light
j	A range of particle size (0.1-1.0 μm in this research)
m	Major axis of an ellipse
n	Minor axis of an ellipse
n_1	Refractive index of the first medium encountered by the incident light (n_1=1.0003 for air)
n_2	Refractive index of the second medium encounteded by the incident light (n_2=1.458 for an oxide layer on silicon)
r	Radius of a spherical particle
t	Thickness of an oxide layer on a silicon wafer
A	Surface area (of an ellipse) on which a particle projects at a given incident/receiving angle
D	Detector location with coordinates (Xd,Yd,Zd)
D'	Image detector location with coordinates (Xd',Yd',Zd')
E	Total energy of the incident light
I	Mie intensity
I_T	Total Mie intensity received by a detector
I_i	Mie intensity for Path$_i$ (I_i=i(1)$_i$+i(2)$_i$), ($_i$=a, $_{b1}$, $_{b2}$, or $_{b3}$)
Ia	Mie intensity for Path A
I_{b1}	Mie intensity for Path B1
I_{b2}	Mie intensity for Path B2

I_{b3}	Mie intensity for Path B3
L	Half of the distance between two centers of the ellipses for a shadow factor in the case of a bare surface
L'	Half of the distance between two centers of the ellipses for a shadow factor in the case of an oxidized surface
P	Particle location with coordinates (Xp,Yp,Zp)
S	Light source location with coordinates (Xs,Ys,Zs)
S'	Image light source location with coordinates (Xs',Ys',Zs')
DI	Dominance Index
DP	Distance from a detector (D) to a particle (P)
SD	Distance from a light source (S) to a detector (D)
SF	Shadow factor
SF_b	Shadow factor of a bare surface
SF_c	Shadow factor of a coated surface
SF_i	Shadow factor for Path$_i$ ($_i$=a, $_{b1}$, $_{b2}$, or $_{b3}$)
SFa	Shadow factor for Path A
SF_{b1}	Shadow factor for Path B1
SF_{b1}'	Shadow factor for Path B1 when the incident light reflected by the air-oxide interface
$SF_{b1}"$	Shadow factor for Path B1 when the incident light reflected from the oxide-silicon interface
SF_{b2}	Shadow factor for Path B2
SF_{b2}'	Shadow factor for Path B2 when the scattered light reflected by the air-oxide interface
$SF_{b2}"$	Shadow factor for Path B2 when the scattered light reflected from the oxide-silicon interface
SF_{b3}	Shadow factor for Path B3
SP	Distance from a light source (S) to a particle (P)
(X,Y,Z)d,p,s,d',s'	Space coordinates of detector, particle, light source, image detector and image light source

α	Reflectivity of the air-oxide interface
β	Reflectivity of the oxide-silicon interface
τ_i	Reflectivity for Path$_i$ ($_i$=a, $_{b1}$, $_{b2}$, or $_{b3}$)
τ_a	Reflectivity for Path A
τ_{b1}	Reflectivity for Path B1
τ_{b2}	Reflectivity for Path B2
τ_{b3}	Reflectivity for Path B3
θ	A scattering angle
θ_a	A scattering angle for Path A (simple scatter)
θ_{b1}	A scattering angle for Path B1 (reflect-scatter)
θ_{b2}	A scattering angle for Path B2 (scatter-reflect)
θ_{b3}	A scattering angle for Path B3 (reflect-scatter-reflect)
ϕ	An incident angle (ϕ_i) or a receiving angle (ϕ_r)
ϕ'	A refractive angle, $\phi' = \sin^{-1}(n_1 \cdot \sin\phi / n_2)$

--- LIST OF FIGURES ---

Figure	Page
1. Four paths of light scattering from a sphere on a reflective surface	4
2. Definition of the scattering plane and scattering angle	6
3. Light reflected by an illuminating area on a surface	7
4. Particle illuminated by an image light source	10
5. Scattered light received by an image detector	10
6. Light scattering with an image light source and an image detector	11
7. Particle illuminated by light reflected by the air-oxide and oxide-silicon interfaces	13
8. Shadow factor for bare and oxidized silicon surfaces	18
9. Shadow factor vs coating thickness, incident angle = 15°, coated by silicon dioxide	20
10. Thickness/diameter ratio vs incident/receiving angle for shadow factor saturation	20
11. Relative Mie intensity vs particle size, Path A, incident angle=15°, scattering angle=165°	22
12. Relative Mie intensity vs particle size, Path B1, incident angle=15°, scattering angle=15°	22
13. Relative Mie intensity vs particle size, Path A, B1, and Total, incident angle=15°, receiving angle=0°	24
14. Relative Mie intensity vs particle size, all paths and total, incident angle=15°, receiving angle=5°	24
15. Total relative Mie intensity vs particle size, incident angle=15°, t=0, 0.1, 0.2, 0.4, 0.8 and 1.6 μm	26

16. Total relative Mie intensity vs particle size, function of particle position on a wafer 27

17. Dominance index vs incident angle, function of oxide thickness (t=0, 0.5, 1.0, 2.0, 3.0 and 4.0 μm) 29

--- LIST OF TABLES ---

Table	Page
1. Optical characteristics of light scattering Paths	12
2. Parameter values for Mie calculation	23
3. Comparison of scattering parameters among particles at different locations on a wafer	28

ABSTRACT

This report describes an analytical approach to light scattering from particles resting on reflective surfaces. The phenomenon of light scattering from particle-deposited surfaces depends on the optical characteristics of the particle, the surface, and the coated film individually and the combined optical effects among these three. Mie theory was used to compute scattering from the particle while ray tracing was used to estimate the influence of the surface. Four paths have been analyzed: (1) back scattering from a particle on which incident light impinges directly, (2) forward scattering from a particle of incident light reflected by a surface, (3) forward scattering from a particle, which then reflects from a surface to reach the detector, and (4) back scattering of reflected light which then reflects again from a surface to reach the detector. A "shadow factor" concept was developed and applied for numerical determination of the amount of light reflected from the surface for each path. Calculations were performed by a computer program using an IBM PC. The results of this study suggest a theory to quantitatively determine the light scattering from particles on reflective surfaces. Non-monotonic response for submicron particles was found. Particle location on the wafer is crucial and the presence of surface coatings always enhances response because of the shadow factor. A dominance index was developed to describe the relative magnitude of each path. Path B1 (reflect-

scatter) was proved the most important, having a dominance index 95 % at an incident angle of about 16° for the oxide thickness greater than 3 μm and 91% at an incident angle of about 21° for bare surfaces.

INTRODUCTION

Laser scanners are used in semiconductor industry for the detection of particulate contamination on wafer surfaces. Sizing of particles on contaminated surfaces is an important function for laser scanners. To size a particle, it is necessary for the instrument to know the relationship of light response (scattering intensity) with particle size. Mie theory is the well-known method to describe light scattering by spherical particles in free air (Van de Hulst, 1981). Unfortunately, Mie theory has not been extended to predict the light scattered by a spherical particle on a surface.

The current method for a laser scanner to measure the sizes of particles relies upon experimental calibration. This relates instrument response to light scattering by monodisperse polystyrene latex (PSL) spheres deposited on given surfaces under certain conditions. There is no theory available to predict the phenomenon of light scattering from spherical particles resting on reflective surfaces except an approximate model presented by Knollenberg (1986).

Therefore, the objective of this work was to apply Mie theory for a particle deposited on a reflective surface. This approach enables us to gain insights into the optical

interactions among the particle, the reflective surface, and any transparent oxide layer present and also to quantitatively determine the light scattering intensities against particle sizes.

THEORY

BACKGROUND

Light scattering from a single particle suspended in air is a complex phenomenon which was analyzed by Mie (1908) based on Maxwell's electromagnetic wave equations. The light intensities scattered by a spherical particle depend on the following parameters: (1) the size of a particle, (2) the wavelength of incident light, (3) the polarization of the incident light, (4) the scattering angle, (5) the refractive index of a particle, and (6) the refractive index of the suspending medium.

Since the Mie solution does not include the problem of light scattering from a particle attached to a boundary surface such as a wafer, we must seek an alternative approach. Knollenberg (1986), Lilienfeld (1986), and Locke and Donovan (1986) have discussed components for light scattering under this condition. Four paths for light scattering analyzed in this work include: (1) Light illuminates the particle directly and then scatters directly to the detector. This is a simple single backscatter interaction (Path A). (2) Light reflected by the boundary

surface illuminates the particle and then forward scatters to the detector. This represents a reflect-scatter interaction path (Path B1). (3) Incident light hits the particle, scatters to the surface and is then reflected to the detector. This is a path of scatter-reflect interaction (Path B2). (4) Combining (2) and (3), incident light reflects first, backscatter from the particle to the surface, and reflects again to reach the detector. This is a reflect-scatter-reflect path (Path B3). These four paths are schematically shown in Figure 1.

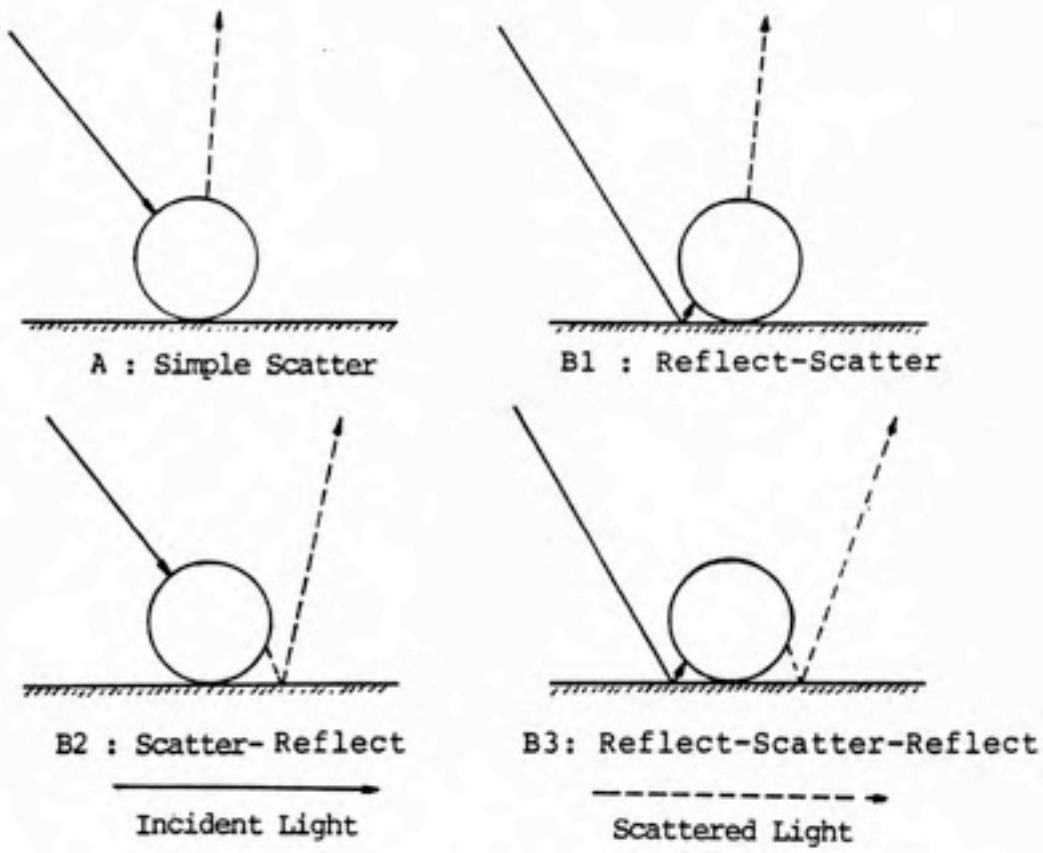

Figure 1 Four Paths of Light Scattering from a Sphere on a Reflective Surface

Other possible paths, for example, scattered light re-illuminates the particle, were not analyzed here for small contribution of scattered light due to multiple reflections and multiple scattering.

ASSUMPTIONS

To model light scattering due to these interactions, some assumptions are made here:

(1) The particle is spherical.

(2) The reflective surface is optically smooth.

(3) Each path acts independently. The incident and reflected waves do not reinforce or interfere.

(4) The total amount of scattered light that reaches the detector results from addition of that from each individual path, i.e., the scattered waves do not interfere.

(5) Light scattered by a particle is independent of where on the particle the light is incident. For example, a particle will scatter light in the same way if it is illuminated by unity area with light intensity of one, as it is illuminated by a half unit area with two units of light intensity.

(6) The projected area of a particle is small compared to the area illuminated by the light source.

(7) Scattered light is detected at a point.

(8) Some incident light will be reflected at the air-oxide interface when an oxide layer is present. Other light

will penetrate through and will partially reflect back to the upper surface of the oxide layer. No energy will be absorbed by the oxide. Single reflection from the oxide-silicon interface is assumed.

PATHS

Path A represents light scattering from a particle without regard to the effect of a surface. This is equivalent to the conventional light scattering from a particle in a homogeneous medium (Mie scattering).

The scattering angle, θ, is usually defined as the angle between the direction of the incident light and the scattered light. The scattering plane is the plane containing the line in the direction of the incident light

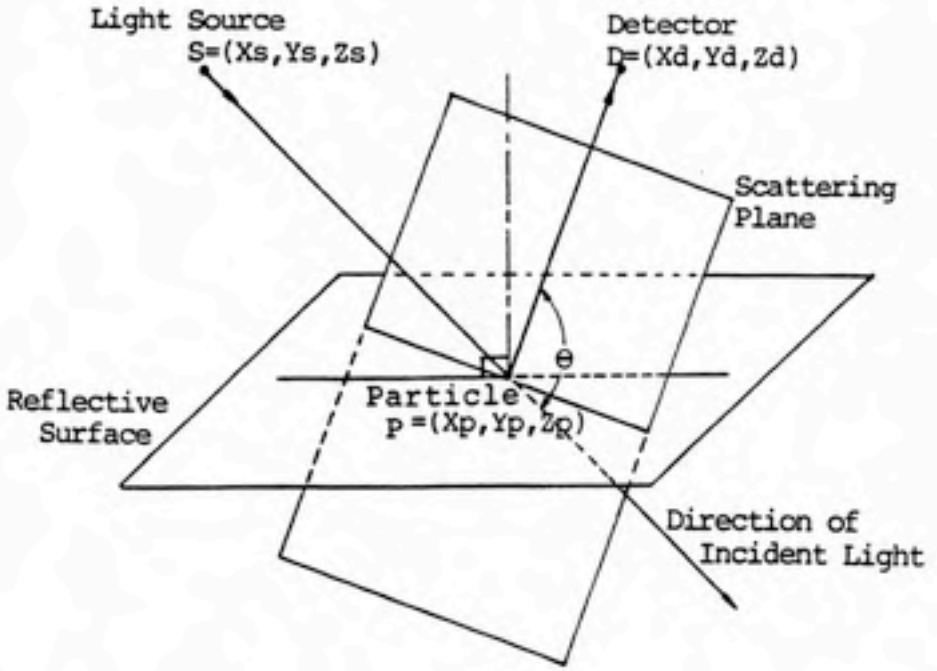

Figure 2 Definition of the Scattering Plane and Scattering Angle

and the line in the direction of the scattered light. Figure 2 shows a scattering plane and scattering angle. The scattering angle can be determined by the coordinates of the light source (S)-(Xs,Ys,Zs), the particle (P)-(Xp,Yp,Zp), and the detector (D)-(Xd,Yd,Zd). Equation (1) gives the dependence of the scattering angle on these coordinates.

$$\theta = \pi - \cos^{-1}\left(\frac{SD'-SP'-DP'}{-2 \cdot SP \cdot DP}\right) \qquad (1)$$

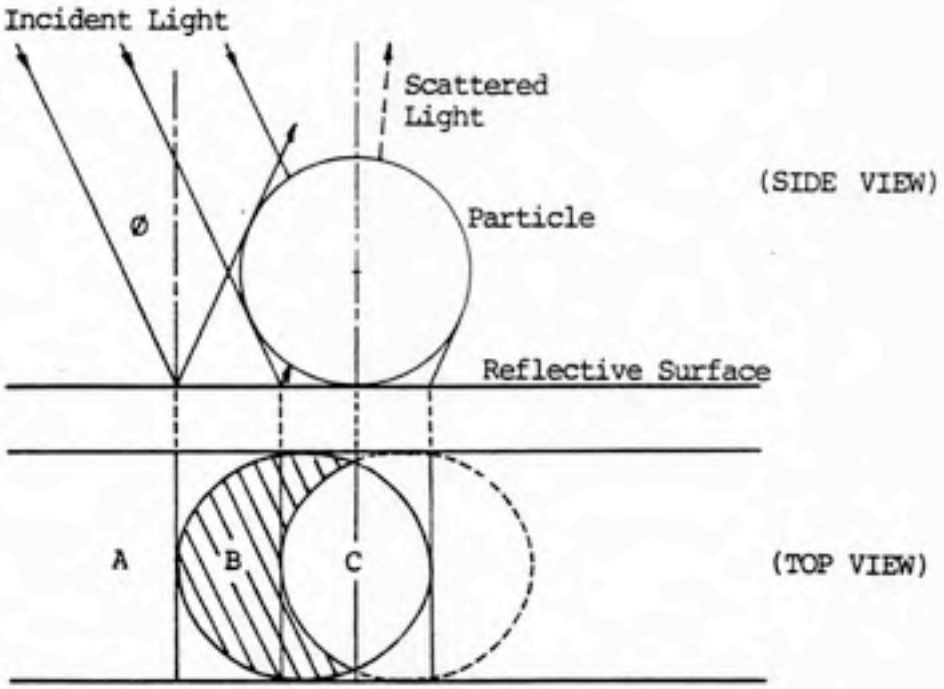

Figure 3 Light Reflected by an Illuminating Area on a Surface

The reflect-scatter path, Path B1 is the path by which the incident light hits the surface close enough to the particle to reflect onto the particle, and then be forward scattered by the particle into the detector. Question 1 is how much incident light will be reflected from the

reflective surface to the particle. As Figure 3 shows, if the reflected position of incident light, region A in figure 3, is too far away from the particle, the reflected light will not hit the particle. If the position, such as in region C, is too close to the particle, the incident light will be intercepted by the particle so that no reflected ray exists.

To numerically determine the amount of incident light that reflects to the particle, a shadow factor (SF) is defined as the area of the incident light on the reflecting plane that is reflected to the particle divided by the area of the particle projected onto the reflecting plane (equation (2))

$$SF = \frac{a}{A} \tag{2}$$

The shape of a spherical particle projected onto a surface at a given incident angle is an ellipse. a, the region B in figure 3, and A, the area of the ellipse, are determined by equations (3) and (4). Complete procedures for the derivations of a and A are listed in appendix A. Equations (3) and (4) summarize these results.

$$a = 2 \cdot r^2 \cdot (\sin\theta + \theta \cdot \sec\theta) \tag{3}$$

$$A = \pi \cdot r^2 \cdot \sec\theta \tag{4}$$

$$SF_b = (\sin(2\theta) + 2\theta)/\pi \tag{5}$$

By substituting a and A in equation (2), the shadow factor can be expressed in terms of θ as shown in equation

(5). It increases from 0 for $\emptyset$=0 (normal incidence) to 1 for $\emptyset$=π/2 (incident light parallel to the surface).

The second question to be concerned is scattering angle associated with Path B1 (θ_{b1}). It is different from θ_a. An easy way to understand the relationship between θ_{b1} and θ_a is to imagine an image light source caused by the reflective surface which is treated as a mirror (figure 4). Let the reflective surface be the XY plane in space. The X and Y coordinates (Xs',Ys') of the image light source is the same as those of a real light source (Xs,Ys). The Z coordinate (Zs') of the image light source is the negative value of Zs. Therefore, θ_{b1} can be calculated from equation (1) by substituting (Xs,Ys,Zs) with (Xs',Ys',Zs').

The third path by which light scattered from a particle on a reflective surface reaches the detector is the <u>scatter-reflect path (B2)</u>. This path has some characteristics similar to those of Path B1. A shadow factor is also applied in Path B2. The value of this factor is determined by the receiving angle ($\emptyset_r$), which is defined as the angle between the direction of scattered light and the direction normal to the reflective surface (see figure 5), of the detector using equation (5). The scattering angle of Path B2 (θ_{b2}) can be found by the image technique already discussed in the Path B1 analysis. An image detector, instead of an image light source, is illustrated in Figure 5 to determine θ_{b2}. It turns out that θ_{b2} is equal to θ_{b1}.

$$\theta_{b2} = \theta_{b1} \qquad (7)$$

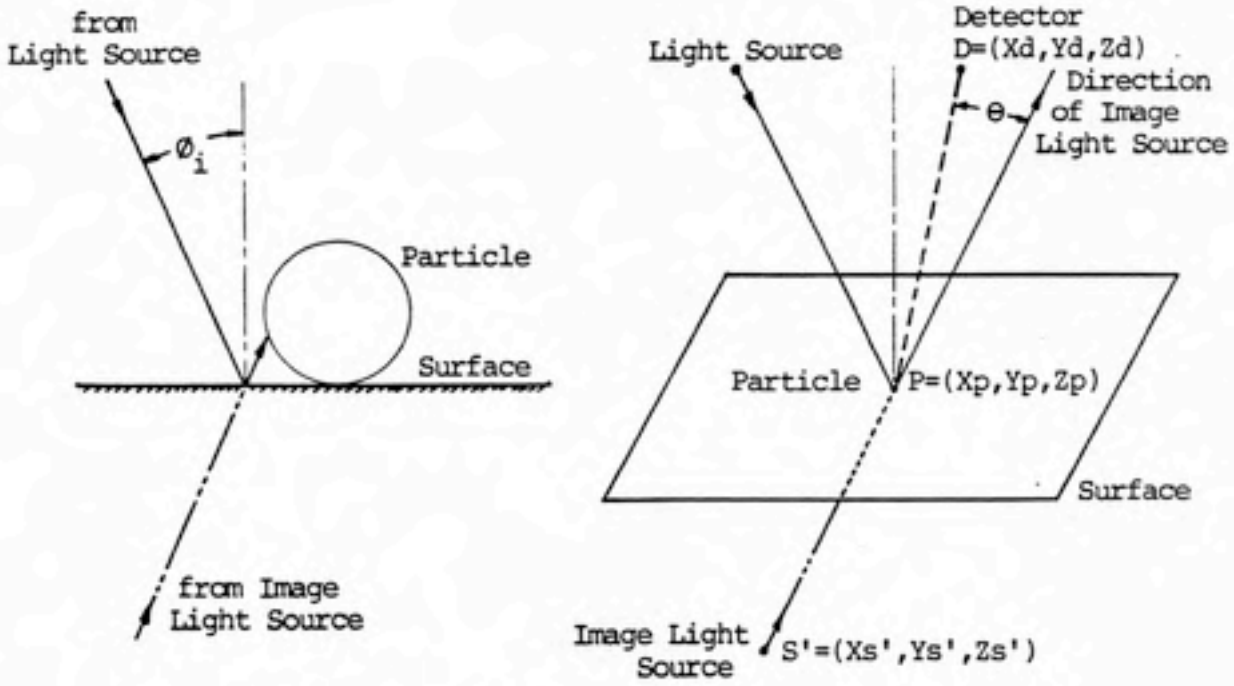

Figure 4 Particle Illuminated by an Image Light Source

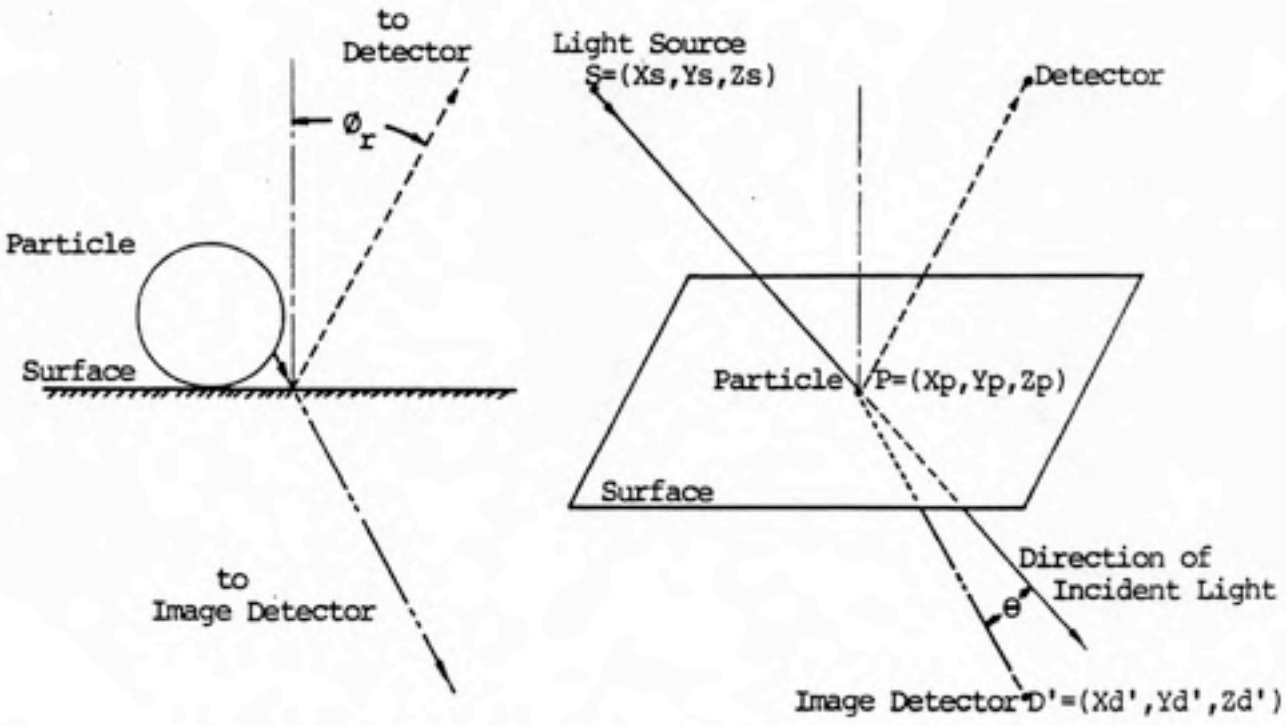

Figure 5 Scattered Light Received by an Image Detector

Combining the reflect-scatter and scatter-reflect paths, a reflect-scatter-reflect path (B3) is determined by two shadow factors, as discussed in the Path B1 and Path B2 analysis, and a new scattering angle (θ_{b3}). An image light source and an image detector are both applied to determine θ_{b3} (figure 6). It comes out to be the same value as that of Path A as shown in equation (8).

$$\theta_{b3} = \theta a \qquad (8)$$

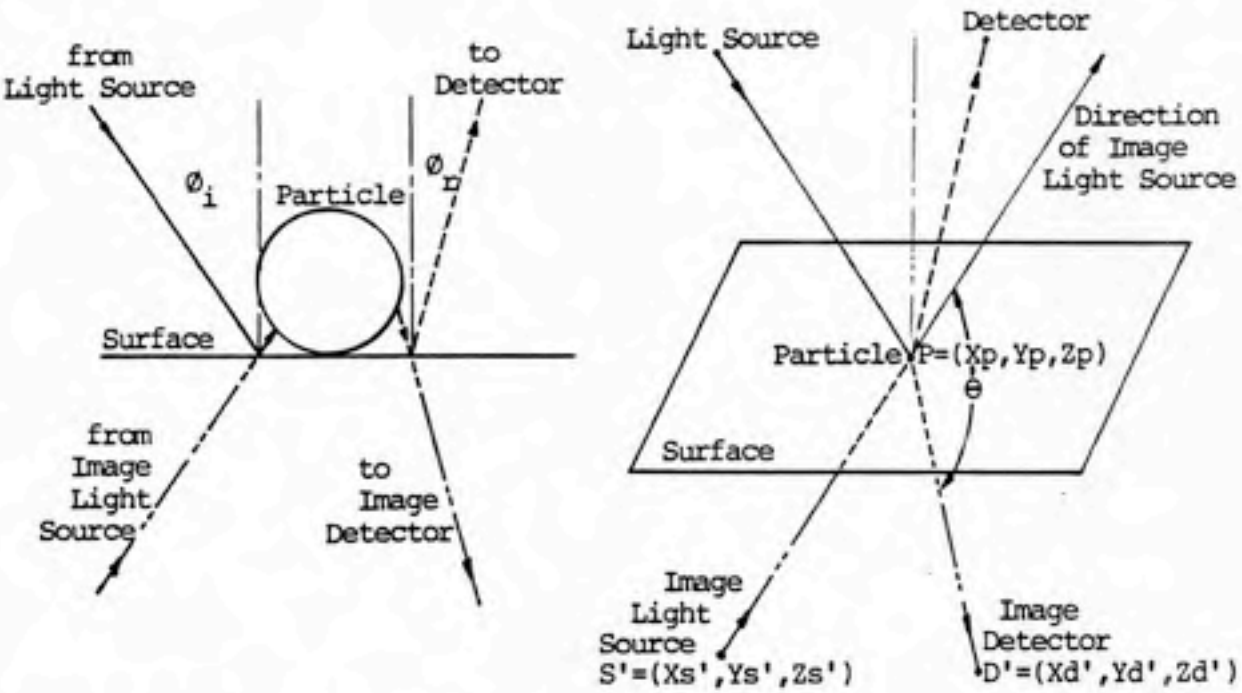

Figure 6 Light Scattering with an Image Light Source and an Image Detector

In summary, all four paths discussed above have different optical characteristics. Shadow factors are dependent on the incident angle ($\emptyset_i$) of the light source or the receiving angle ($\emptyset_r$) of the detector. Scattering angles are determined by the positions of the light source, the particle, and the detector as well as whether light has been

reflected or scattered. Table 1 summarizes the optical characteristics for all four mechanisms.

Table 1 Optical Characteristics of Light Scattering Paths

PATH		SHADOW FACTOR	SCATTERING ANGLE
Simple Scatter	A	No	θ_a
Reflect-Scatter	B1	SF_{b1} depends on incident angle ϕ_i	θ_{b1}
Scatter-Reflect	B2	SF_{b2} depends on receiving angle ϕ_r	$\theta_{b2} = \theta_{b1}$
Reflect-Scatter-Reflect	B3	SF_{b1} and SF_{b2}	$\theta_{b3} = \theta_a$

SURFACE LAYER EFFECT

An important effect on light scattering from particles deposited on wafers covered with oxide layers of various thickness has been experimentally reported (Locke and Donovan, 1986). When a light source illuminates a reflective surface covered with an oxide layer, part of the incident light is reflected at the air-oxide interface. The other part of the incident light penetrates the layer at the refractive angle, reaching the oxide-silicon interface. Light reflected at this interface passes back through the oxide to the air-oxide interface. This reflected component then penetrates the top of oxide layer and strikes the particle after another refraction. If the reflectivities at

the air-oxide interface and the oxide-silicon interface be α and ß respectively and E is the total energy of light illuminating the reflective surface, αE will be the amount of energy reflected at the air-oxide interface and ß(1-α)E will be the light energy reflected at the oxide-silicon interface.

The shadow factor for αE can be estimated by equation (5). The part ß(1-α)E has more possibility of

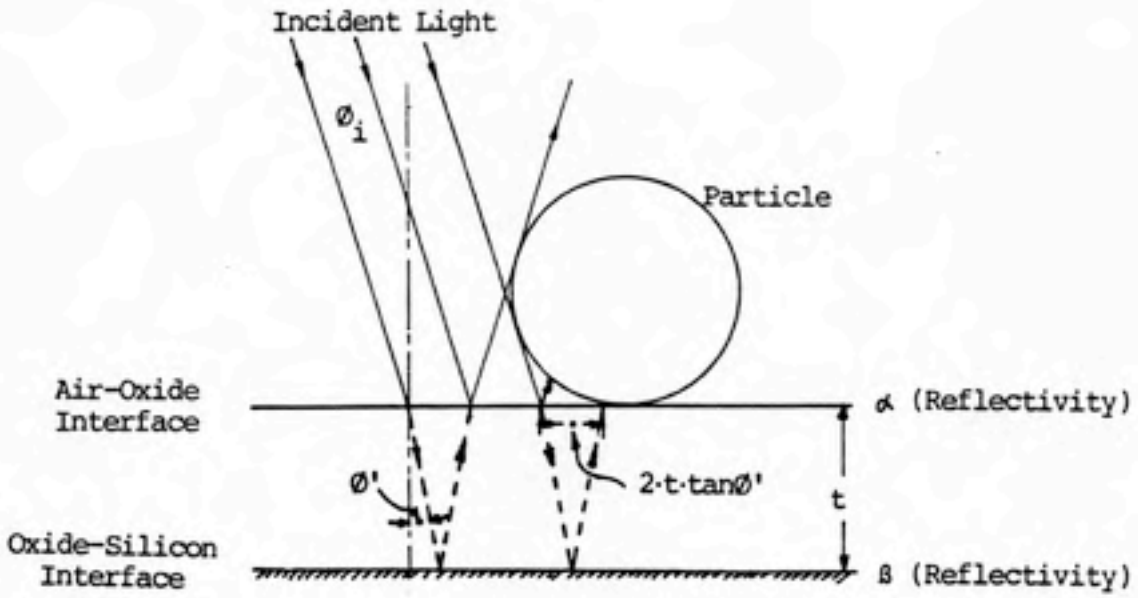

Figure 7 Particle Illuminated by Light Reflected by the air-oxide and oxide-silicon interfaces

hitting the particle for a distance of 2·t·tanØ' (see figure 7) moving toward inside of the particle. Therefore, the shadow factor for ß(1-α)E is greater than that for αE.

Appendix B shows the derivation of the shadow factor when an layer exists. Equations (9),(10) and (11) summarize this calculation.

$$L' = r \cdot \tan\theta + t \cdot \tan\theta' \tag{9}$$

$$m = r \cdot \sec\theta \tag{10}$$

$$SFc = \frac{2}{\pi}\left[\frac{L'}{m'}\sqrt{m'^2 - L'^2} + \sin^{-1}\left(\frac{L'}{m}\right)\right] \tag{11}$$

Equation (11) shows that the shadow factor for an oxidized surface is a function of particle size, the incident angle, refractive indices of air and oxide layer, and the thickness of oxide layer. When the oxide thickness is zero, equation (11) is the same as equation (5).

DOMINANCE INDEX

All paths may contribute to the total light intensity received by the detector. It is difficult to defined these contribution from each other. A dominance index (DI) has been designed to show the percent of total Mie intensity that belongs to a specific path. It is defined by the following equation.

$$DI_i = \frac{\sum_j (I_{i,j})}{\sum_i \sum_j (I_{i,j})} \tag{12}$$

i is the given path(A, B1, B2, or B3). j is the range of particle size of interest. The dominance index varies from zero to one. If a dominance index is zero for a particular

path, there is no contribution from this path of scattered light to the total response. If a dominance index is close to one, the corresponding path is the dominant path contributing to the total Mie intensity response.

MODEL AND COMPUTATION

On the basis of the assumptions and mechanisms discussed above, a mathematical model for the total Mie intensity received by a detector is set up in equation (13).

$$I_T = \sum_i (\tau_i \cdot SF_i \cdot I_i) \qquad (13)$$

τ_i is the reflectivity, SF_i is the shadow factor, and I_i is the Mie intensity of each individual mechanism.

In case of Path A for both bare and oxidized surfaces, $\tau_a \cdot SF_a$ is equal to one because of a total illumination in this mechanism. I_a is the Mie intensity at scattering angle θ_a.

For Path B1, in case of bare surface, τ_{b1} is the reflectivity of the surface and SF_{b1} is the shadow factor, determined by equation (5), with incident angle θ_i. When the surface is oxidized, $\tau_{b1} \cdot SF_{b1}$ becomes the form shown in equation (14).

$$\tau_{b1} \cdot SF_{b1} = \alpha \cdot SF_{b1}' + \beta \cdot (1-\alpha) \cdot SF_{b1}'' \qquad (14)$$

α, β are the reflectivities at air-oxide interface and oxide-silicon interface. SF_{b1}' is the shadow factor for the oxide layer. SF_{b1}'' is the shadow factor for the silicon surface.

For Path B2, in case of a bare surface, τ_{b2} is the reflectivity of the surface and SF_{b2} is the shadow factor, determined by equation (5), with receiving angle θ_r. When

the surface is oxidized, $\tau_{b2} \cdot SF_{b2}$ becomes the form shown in equation (15).

$$\tau_{b2} \cdot SF_{b2} = \alpha \cdot SF_{b2}' + \beta \cdot (1-\alpha) \cdot SF_{b2}'' \qquad (15)$$

In case of Path B3, SF_{b3} is equal to SF_{b1} times SF_{b2}. I_{b3} is the Mie intensity at scattering angle θ_{b3}.

A program written in Turbo Pascal has been developed to handle all calculations running on a personal computer. Appendix C lists the guide to users and the program statements. A command-driven menu was designed in this program to make it easy to use. This program has the capability of transferring data into Lotus-acceptable files for further data treatment such as graphing and statistical analysis. To reduce computation time, a 8087 math coprocessor is recommended to be installed in the PC. This may save 75% of the computer time to obtain results.

RESULTS AND DISCUSSIONS

SHADOW FACTOR

For the bare surface case, the shadow factor is a function of incident angle or receiving angle only. Figure 8 shows the relationship between the incident/receiving angle and the shadow factor. The shadow factor goes from zero for normal incidence/ receiving to one when light is parallel to the surface. At an incident angle of 15°, the angle used in the Wafer Inspection System (WIS-150) built by Aeronca Electronics, Inc., the shadow factor is 0.326.

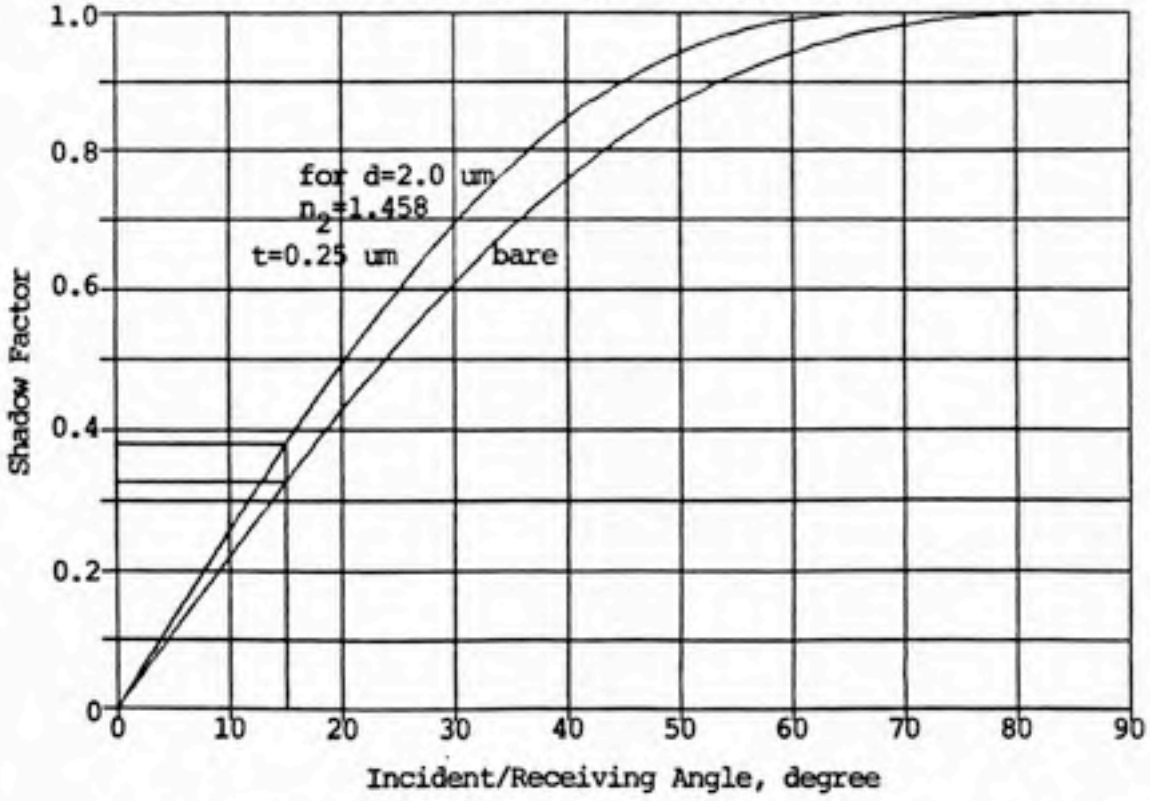

Figure 8 Shadow Factor for Bare and Oxidized Silicon Surfaces

When a surface is coated with a thin film, the shadow factor for light reflected from bottom surface becomes more complicated. Incident/receiving angle, refractive index and

thickness of the film, and particle size are the factors need to determine the magnitude of the shadow factor. As depicted in figure 8 the shadow factor increases from zero to one with increasing incident/receiving angles, given a coating thickness of 0.25 μm of silicon dioxide (refractive index=1.458) and a particle size of 2.0 μm. The curve for an oxidized surface for a fixed particle size shifts up-leftward from that for a bare surface. In other words, given the same incident/receiving angle, the shadow factor for a coated surface will greater than that for a bare surface. For example, the shadow factor is 0.379 for a particle of 2.0 μm diameter on a surface coated with silicon dioxide of 0.25 μm thickness at the incident angle of 15°; on a bare surface the shadow factor at this same inciden angle is 0.326.

Figure 9 shows how shadow factors vary with the thickness of the oxide layer as a function of particle sizes (0.5, 1.0, 2.0, and 4.0 μm) under a fixed incident/receiving angle of 15°. All curves start with the same point which corresponds to the shadow factor of a bare surface for a fixed incident/receiving angle (e.g. 15°). The curve for a large particle moves toward the right with decreasing slope. It is easily understood that a small particle resting on a surface coated with a given thickness will be more illuminated by a light beam reflected from the bottom surface than a large particle will be. For increasing coating thickness, the shadow factors for small particles

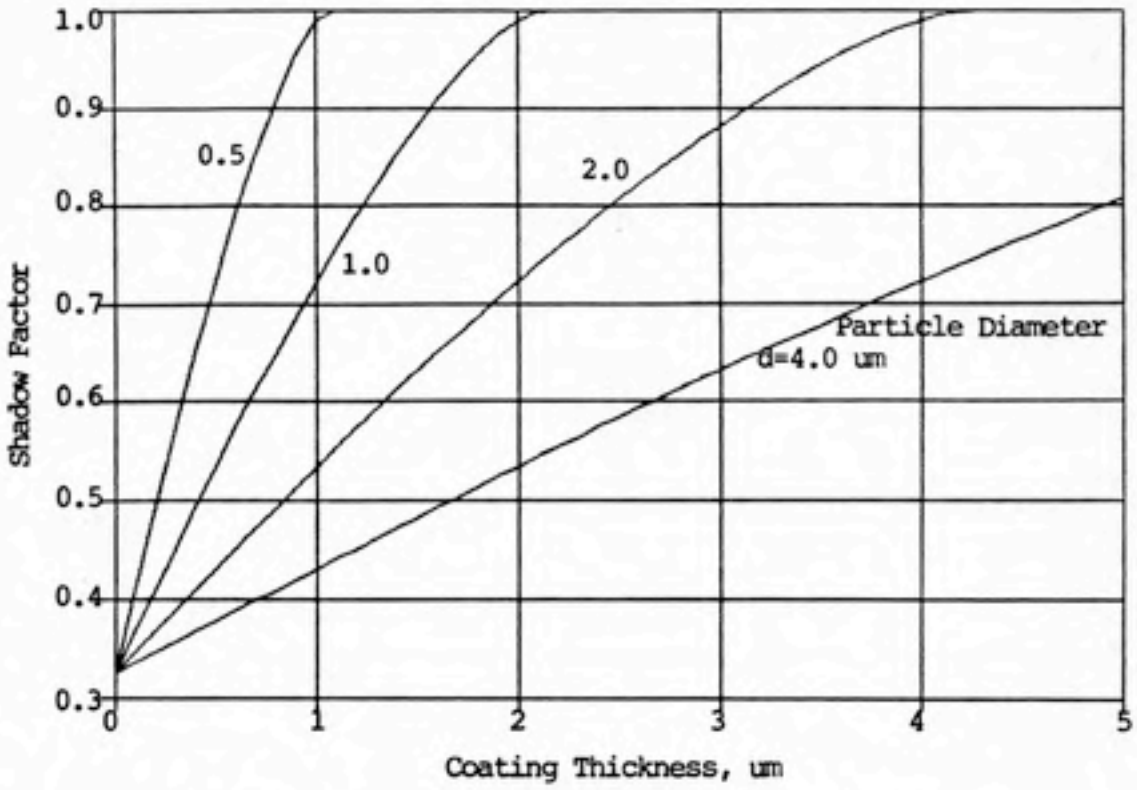

Figure 9 Shadow Factor vs Coating Thickness
Incident Angle = 15°, Coated by Silicon Dioxide

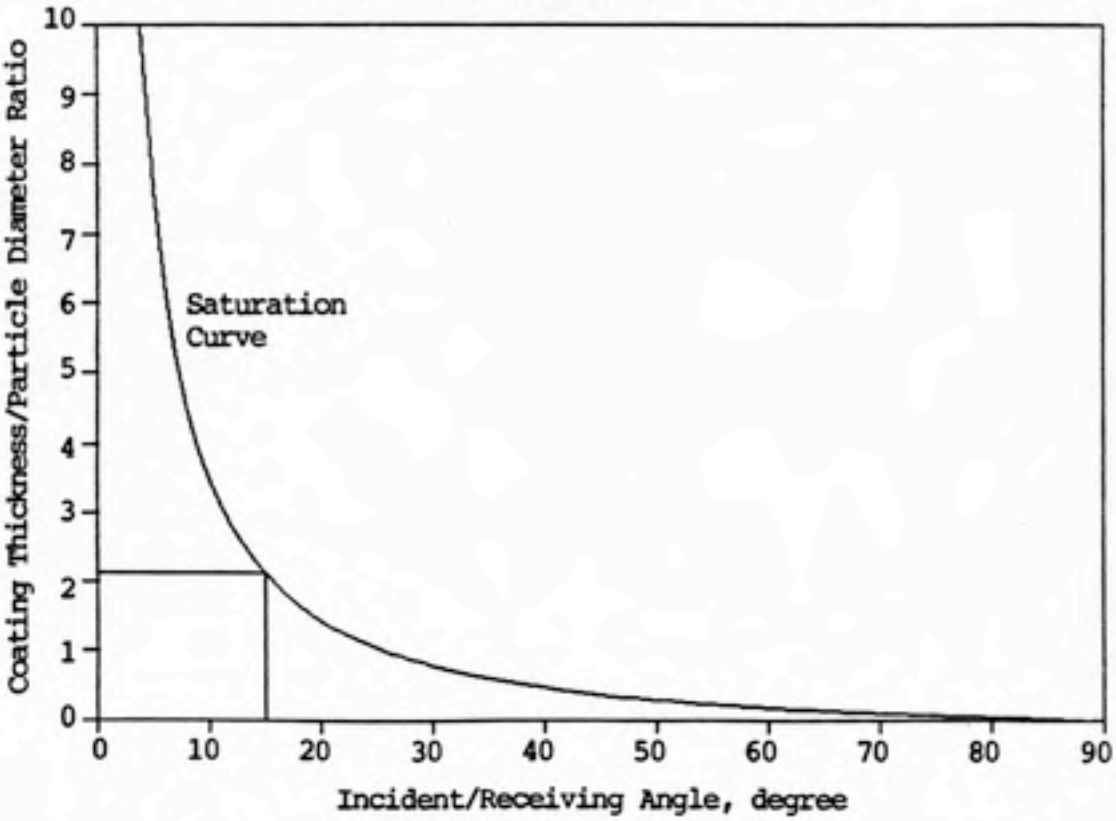

Figure 10 Thickness/Diameter Ratio vs Incident/Receiving Angle
for Shadow Factor Saturation

increase dramatically and reach a condition of saturation, i.e., the shadow factor is equal to one. The shadow factors for a large particle also reach saturation but at a slower rate. A ratio of coating thickness to particle diameter can be found above which saturation will occur given an incident/receiving angle and the refractive indices of the two mediums (air and oxide in this case). This is shown in appendix D and summarized in equation (16).

$$\frac{t}{d} = \frac{(1-\sin\theta)/\cos\theta}{2\sin\theta/\sqrt{(n_2^2/n_1^2)-\sin^2\theta}} \qquad (16)$$

For example, if an incident/receiving angle θ is 15°, n_1=1.0003, and n_2=1.458, the t/d for saturation will be 2.13. These relationships could be also found in figure 10. In summary, illuminating light reflected from the bottom surface (the oxide-silicon interface) becomes more important for a small particle than for a large particle.

MIE INTENSITIES FOR THE BARE AND COATED SURFACES

Once the computer program for light scattering from particles on reflective surfaces has been set up, Mie intensities for the paths discussed above can be calculated given all the parameters needed in the program. The values for all parameters used in our calculation are given in Table 2.

Figure 11 shows curves of Mie intensity calculated for Path A which corresponds to a scattering angle of 165° (more backward scattering). A non-monotonic response

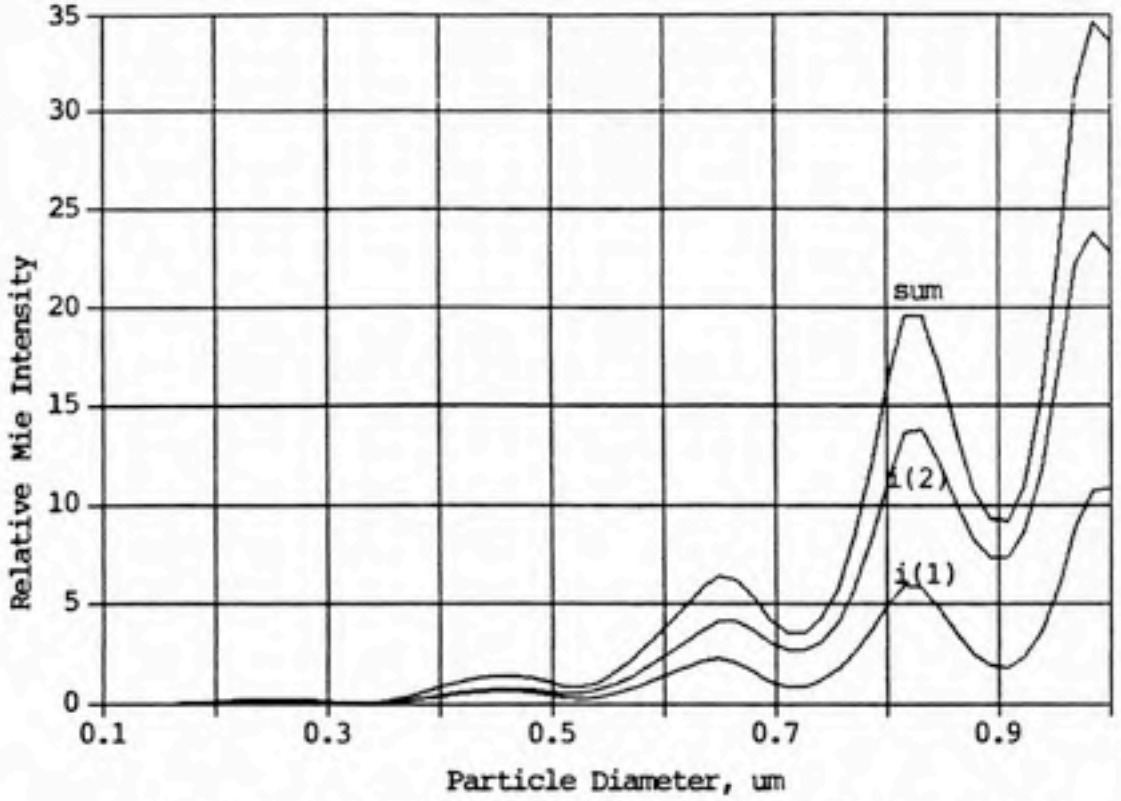

Figure 11 Relative Mie Intensity vs Particle Size, Path A
Incident Angle = 15°, Scattering Angle = 165°

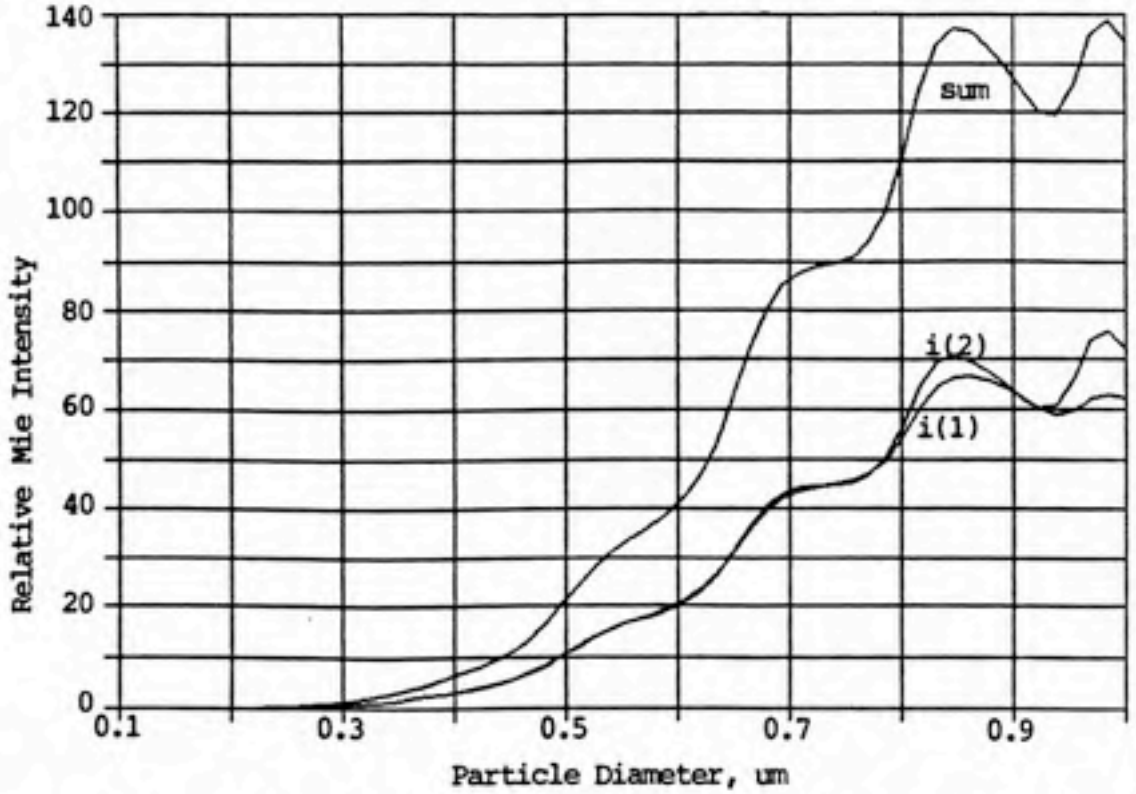

Figure 12 Relative Mie Intensity vs Particle Size, Path B1
Incident Angle = 15°, Scattering Angle = 15°

(oscillation) is found for particle sizes between 0.1 and 1.0 μm. Curve valleys occur at the particle sizes about 0.53, 0.71, and 0.90 μm.

Table 2. Parameter Values for Mie Calculation

Parameter	Value
Particle Refractive Index	1.586 + 0i (PSL spheres)
Wavelength of Incident light	0.6328 μm (Helium-neon laser)
Incident Angle	15°
Receiving Angle	0°
Refractive Index of Oxide Layer	1.458 (Silicon Dioxide)
Reflectivity of Air -Oxide Interface	0.3
Reflectivity of Oxide-Silicon Interface	0.9

Figure 12 is a similar plot for Path B1. This plot shows that the Mie intensity curve is close to monotonic for particle sizes less than 0.84 μm. Beyond 0.84 μm, the response becomes non-monotonic. The magnitude of Mie intensity from Path B1 is much greater than that from Path A. This implies that even through a shadow factor applied for Path B1 is only 0.326, Path B1 makes more contribution to the scattered light received by the detector. This is because of the small scattering angle (15°) for Path B1.

Figure 13 is a combined plot for Mie intensities from Paths A and B1, and their total for a bare surface. There

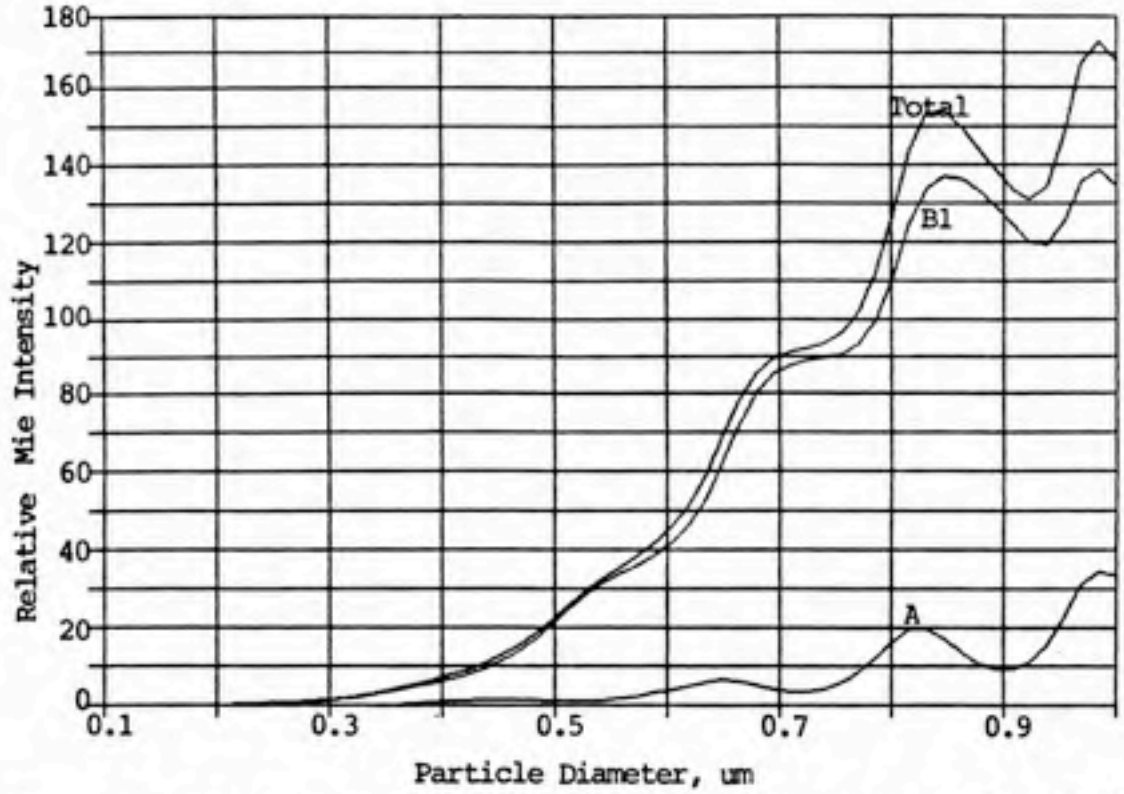

Figure 13 Relative Mie Intensity vs Particle Size, Path A, B1 and Total
Incident Angle = 15°, Receiving Angle = 0°

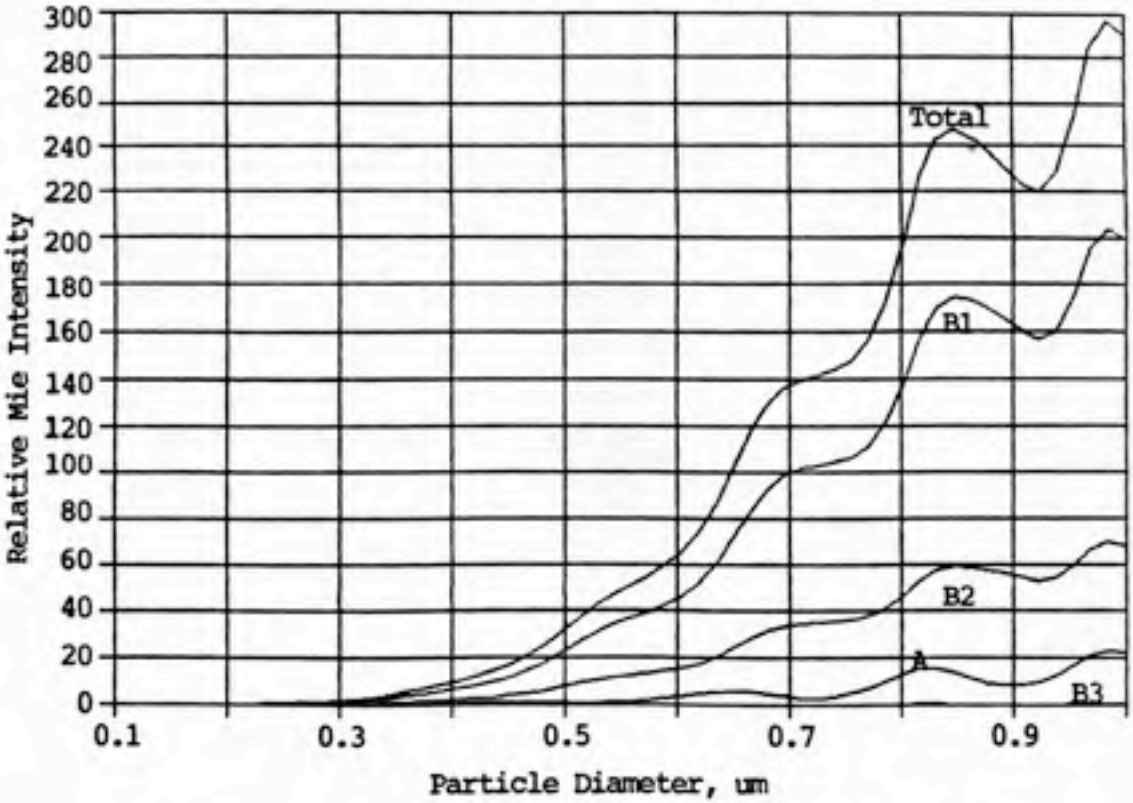

Figure 14 Relative Mie Intensity vs Particle Size, All Paths and Total,
Incident Angle = 15°, Receiving Angle = 5°

is no light scattering from Paths B2 and B3 because a normal receiving (θ_r=0) was used in the calculation. Figure 14 is a plot of another example at an incident angle 15° and a receiving angle 5°. It shows that the pattern of the curves is similar to that of the curve in figure 13. The curves of Path B2 and B3 are present because of the receiving angle is not 0. The magnitude for Path B3 is very small for its multiple reflections.

In general, the total Mie intensity has the same pattern as that from the dominant path such as B1 in these two examples. The path, such as B3, of multiple reflections or multiple scattering has almost negligible contribution to the total amount of Mie intensity.

As discussed before, the shadow factor increases with increasing coating thickness for a given particle size. This means that more of the light reflected from the bottom surface will be scattered by the particle. Figure 15 plots the total Mie intensity against particle size given a variety of oxide coating thicknesses (bare, 0.1, 0.2, 0.4, 0.8, and 1.6 μm). It shows that all Mie intensity curves have the same pattern, i.e., peaks and valleys occur at the same particle size. The only difference is the magnitude of the Mie intensity.

In summary, an oxide coating increases the amount of light illuminating a surface particle by Path B1, the reflect-scatter. It makes particles scatter more light to the detector.

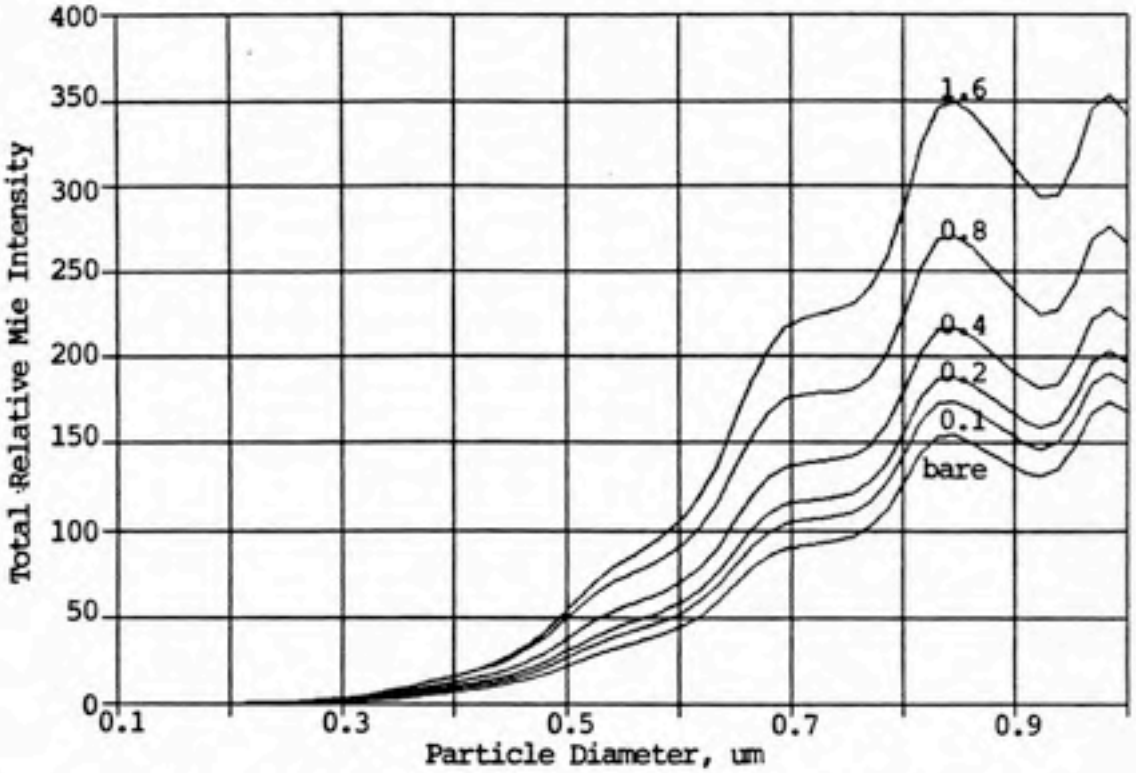

Figure 15 Total Relative Mie Intensity vs Particle Size
Incident Angle = 15°, t=0, 0.1, 0.2, 0.4, 0.8 and 1.6 um

GEOMETRY EFFECTS

A common way for a laser scanner to detect particles on a wafer is to scan the incident light beam across the wafer from one edge to the center and then to the other edge of the wafer. The wafer transportation system then advances the wafer a small distance for further inspection. The question to be discussed next here is the dependency of the Mie intensity response upon particle position along this scan line. Figure 16 depicts the Mie intensity response

against particle size as a function of particle position on a 3" diameter wafer. The curves shown in figure 16 all

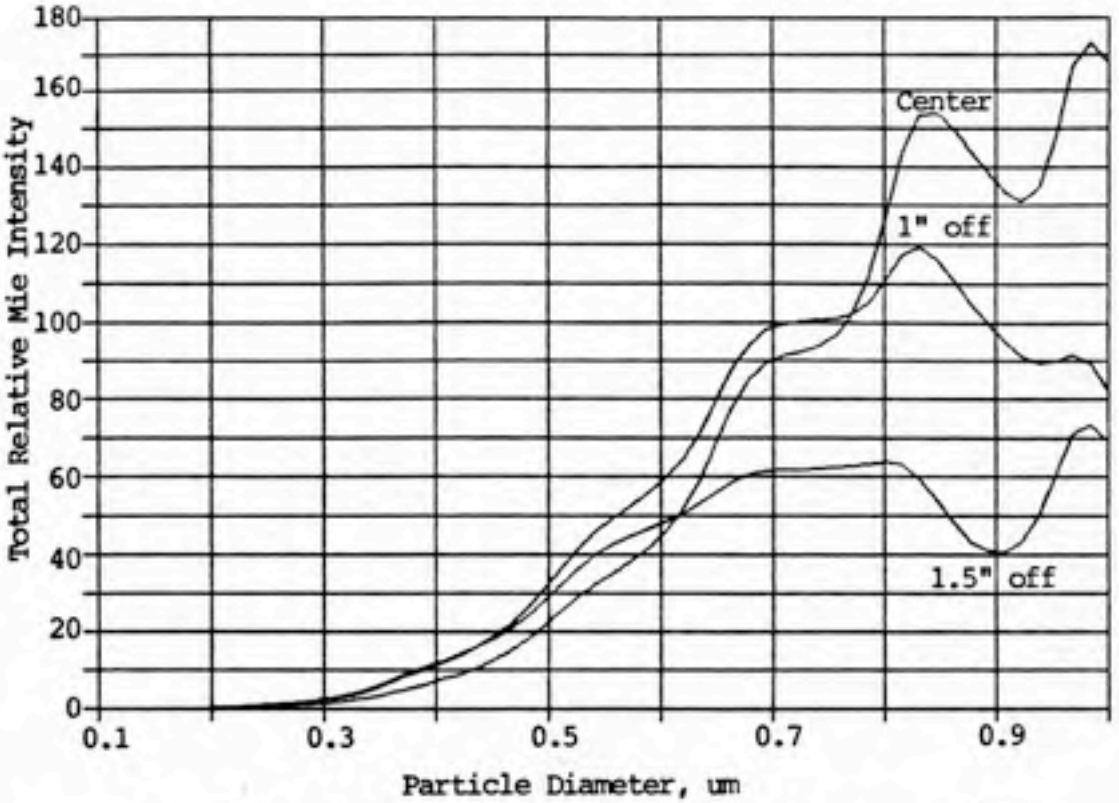

Figure 16 Total Relative Mie Intensity vs Particle Size, Function of Particle Position on a Wafer

have the same scattering parameters except for particle position: at the center of the wafer, 1" ; and 1.5" away from the center. These curves show a substantial difference in Mie intensity response when a particle deposits on a variety of the positions on the wafer. Table 3 lists the parameters changed because of the change of particle position. The substantial changes in all the angles and the shadow factors provide a good explanation as to why the response changes so much.

The scattering angle and the shadow factor are two important parameters that affect the Mie intensity seen by a detector. A particle away from the center of the wafer will increase the scattering angle and so decrease the Mie

Table 3. Comparison of Scattering Parameters among Particles on Different Positions of a Wafer

Parameter	PARTICLE POSITION Center	1" off	1.5" off
Scattering angle of Path A,B3	165°	165.28°	165.61°
Scattering angle of Path B1,B2	15°	26.78°	36.04°
Incident angle, Øi	15°	18.49°	21.91°
Receiving angle, Ør	0°	11.31°	16.70°
Shadow factor for Øi	0.326	0.397	0.464
Shadow factor for Ør	0	0.248	0.361

intensity. The geometry among the incident light, the particle location, and the detector is very crucial for the determination of instrument response.

DOMINANCE ANALYSIS

To make one path dominate all others requires that all paths but one approach zero. A normal receiving angle (Ør=0) makes Paths B2 and B3 zero because no shadow factor exists. This was shown in the previous example. Similarly, normal incidence forces Paths B1 and B3 to become zero. The dominance index for Path B1 is 0.896 when the incident angle is 15° and the receiving angle is zero. Further analyses of

the dominance index versus the incident angle in figure 17 show that a maximum dominance index of 0.912 occurs at the incident angles about 20 - 22° in the case of a bare surface

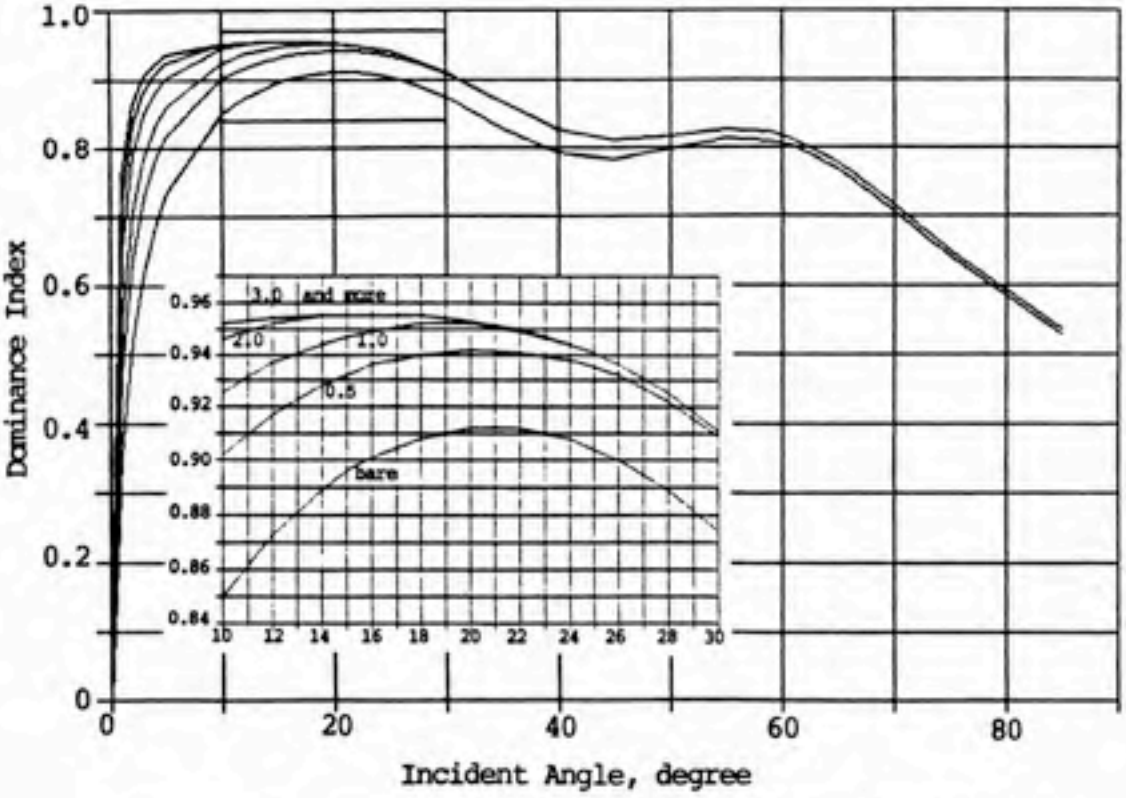

Figure 17 Dominance Index vs Incident Angle, Function of Oxide Thickness (t=0, 0.5, 1.0, 2.0, 3.0 and 4.0 um)

and the size range between 0.1 - 1.0 μm. If oxide coating thickness increases, the optimal incident angle which shows that the maximum dominance index shifts toward smaller values of incident angle until a saturation condition occurs with a maximum dominance index of 0.955 at size range of 14 - 18° (figure 17). These results give a good idea of how dominant one path can be and allow further instrument design efforts to be based on optimal incident angles.

CONCLUSIONS

Four independent paths by which light can be scattered from a particle on a reflective surface to a detector have been defined and analyzed. The shadow factor was introduced as a quantitative index to determine the amount of particle projected area illuminated by the reflected light. A model based on these concepts predicted that light scattered from particles on a reflective surface give a non-monotonic response for a particle size range of 0.1 - 1.0 μm. Particles having the same size deposited on different locations of the reflecting surface were predicted to generate different responses to a detector and were explained by the change of scattering angle and shadow factor. It is recommended that laser scanning be done with a constant scattering angle to obtain a consistent response. The coatings on a silicon, surface such as a non-absorbing oxide layer, have shown to enhance instrument response without changing the shape of the response curve. Coatings also led to an optimal dominance index for one of the paths -- more than 95% within the range of incident angle between 14 - 18°.

Further research should consider the effect of light interference on the shadow factor. The effect of multiple reflections within the coating medium and indeed between the incident light and that reflected from the bare silicon surface should be incorporated into the analysis.

Experiments should be conducted to verify the validity of this analytical approach.

Acknowledgment

This research was sponsored by Research Triangle Institute located in Research Triangle Park, North Carolina.

References

1. Knollenberg, R.G. The Importance of Media Refractive Index in Evaluating Liquid and Surface Microcontamination Measurements. Proceedings of the Institute of Environmental Sciences 1986 Annual Meeting, pp. 501-511.
2. Lilienfeld, P. Optical Detection of Particle Contamination on Surfaces: A Review. Aerosol Science and Technology, 5:145-165, 1986.
3. Locke, B.R., et al. The Detection of Polystyrene Latex Particles on Polished and Oxidized Silicon Wafers: An Instrument Characterization. Proceedings of the Institute of Environmental Sciences 1986 Annual Meeting, pp.487-492.
4. Tullis, B.J. Measuring and Specifying Particle Contamination by Process Equipment: Part III, Calibration. Microcontamination, 4, No. 1 January 1986, p 51-55, 86.
5. Van de Hulst, H.C. Light Scattering by Small Particles. Dover Publications, Inc., New York, 1981

Appendix A.

Derivation of Shadow Factor for Bare Surfaces

Recall the definition of shadow factor (SF) and equation (2) in p.9. SF is determined by two terms as the following.

$$SF = \frac{a}{A} \qquad (2)$$

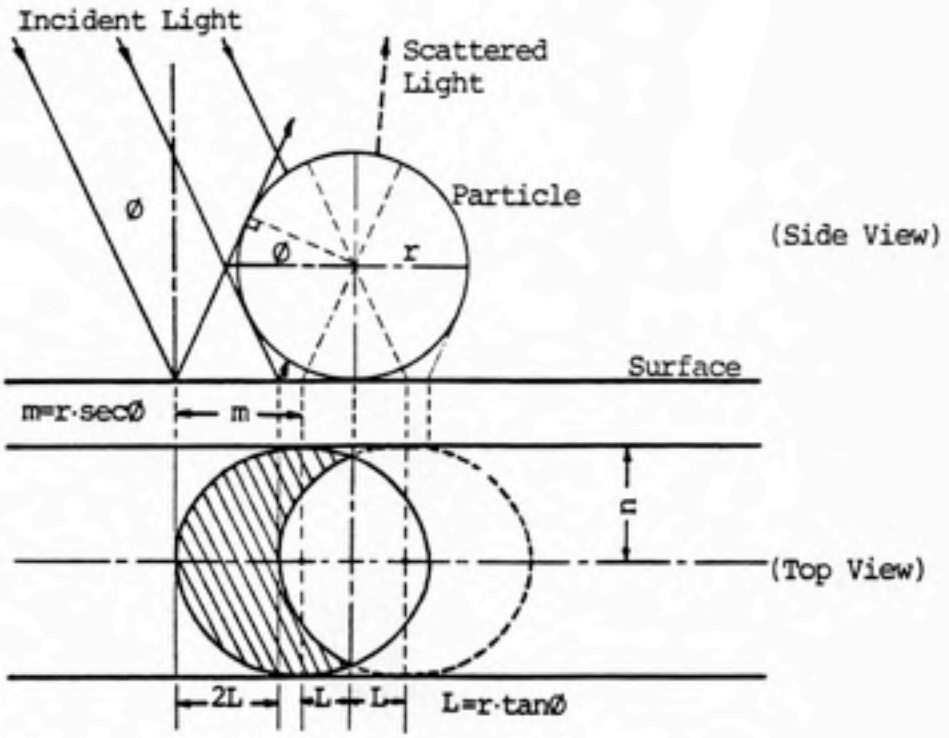

Figure A1 Light Reflected to a Particle on a Bare Surface

As shown in figure A1, a is the illuminated area and A is the total projected area of a particle illuminated by incident light given an incident angle Ø. Let m and n be the major and minor axes of an ellipse, then A can be expressed as equation (A.1).

$$A = \pi \cdot m \cdot n \qquad (A.1)$$

n is always equal to the radius (r) of the particle. m is a function of incident/receiving angle Ø (see figure A1). Thus

$$m = r \cdot \sec\emptyset \qquad (A.2)$$

$$n = r \qquad (A.3)$$

Substitution of equations (A.2) and (A.3) into equation (A.1) gives

$$A = \pi \cdot r^2 \cdot \sec\emptyset \qquad (4),(A.4)$$

To determine the illuminated area a, some transformations should be made. First of all, let the moon-like area (a) be divided by two equal horn-like areas (region X) as illustrated in figure A2. This gives

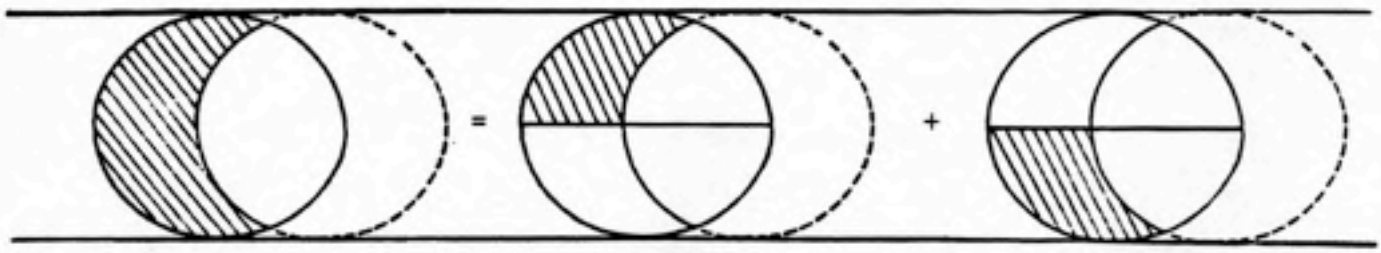

Figure A2

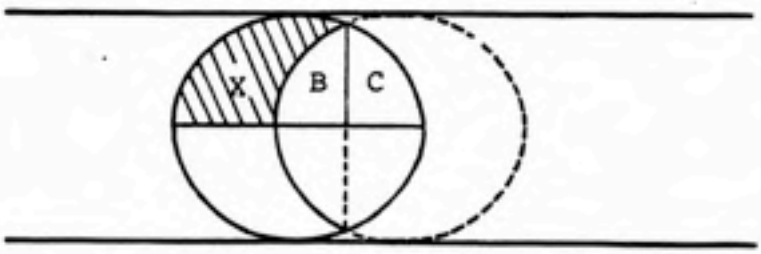

Figure A3

$$a = 2 \cdot (\text{region X}) \qquad (A.5)$$

The horn-like area, region X in figure A3, is equal to the area of region X plus B minus region C, as in equation (A.6), for the equivalence of regions B and C.

$$(\text{region X}) = (\text{region X+B}) - (\text{region C}) \qquad (A.6)$$

Region X+B is equal to a quarter of the area of an ellipse, region D, plus the region E as shown in figure A4. Thus

$$(\text{region X+B}) = (\text{region D}) + (\text{region E}) \qquad (A.7)$$

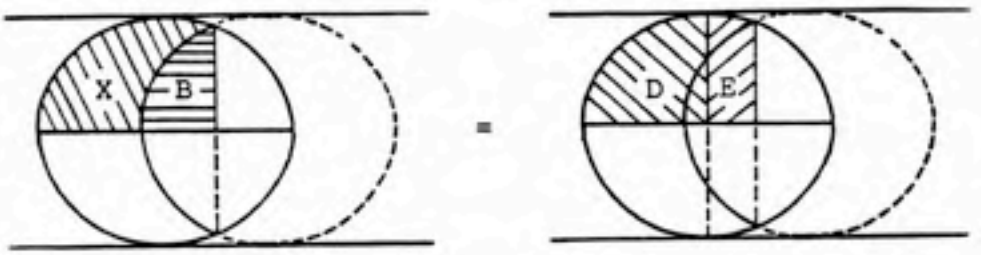

Figure A4

Region C is equal to region D minus E as in figure A5. This gives

$$(\text{region C}) = (\text{region D}) - (\text{region E}) \qquad (A.8)$$

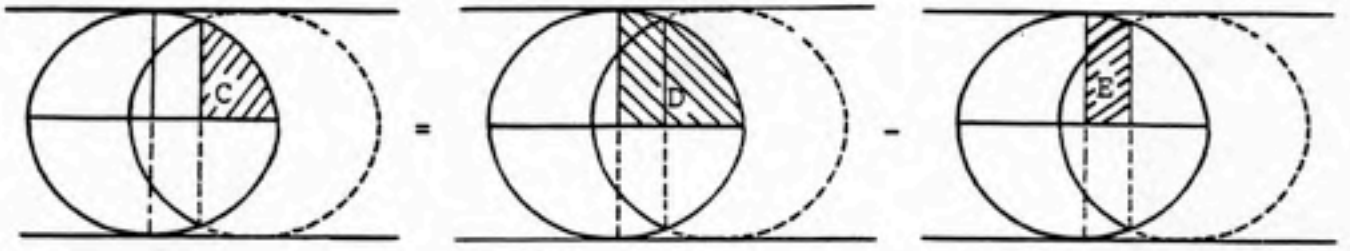

Figure A5

Substitution of equations (A.7) and (A.8) into equation (A.6) gives

$$(\text{region X}) = 2 \cdot (\text{region E}) \qquad (A.9)$$

Thus

$$a = 4.(\text{region E}) \tag{A.10}$$

The function of an ellipse given major and minor axes of m and n can be expressed as equation (A.11).

$$\frac{x^2}{m^2} + \frac{y^2}{n^2} = 1 \tag{A.11}$$

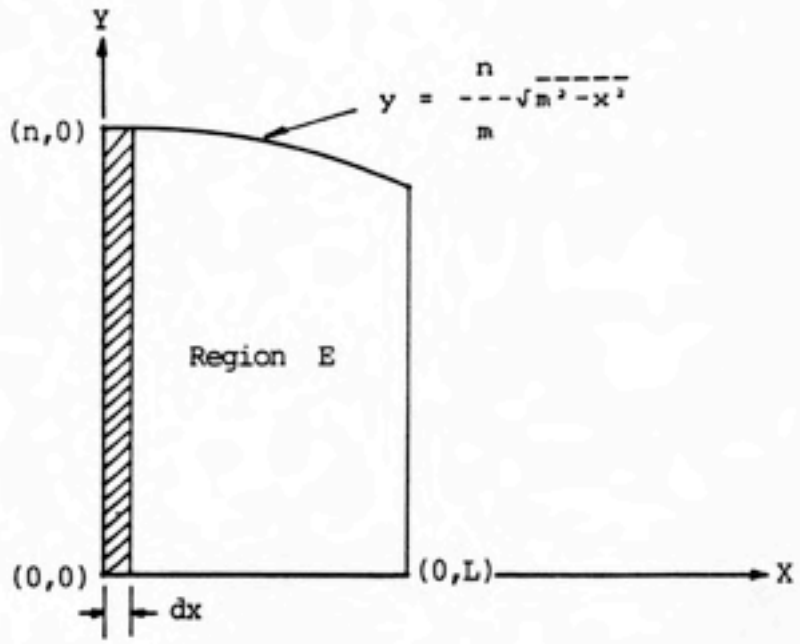

Figure A6

As figure A6 shows, the function of the curve (upper boundary) of region E is the elliptic function as the equation (A.12) which is rearranged from equation (A.11), ignoring the negative value of the square root.

$$y = \frac{n}{m}\sqrt{m^2-x^2} \tag{A.12}$$

Thus, area of region E, defined as a', can be taken an elliptic integral from zero to L. This gives

$$a' = \int_0^L y \cdot dx = \int_0^L \frac{n}{m}\sqrt{m^2-x^2} \cdot dx \tag{A.13}$$

Here L, as a function of θ, is a half of distance between the centers of two ellipses (see figure A1), and can be expressed as equation (A.14).

$$L = r \cdot \tan\theta \qquad (A.14)$$

Since

$$\int \sqrt{m^2 - x^2}\, dx = \frac{1}{2}\left[x \cdot \sqrt{m^2 - x^2} + m^2 \cdot \sin^{-1}\left(\frac{x}{m}\right) \right]$$

Therefore, equation (A.13) becomes

$$a' = \frac{n}{2 \cdot m}\left[L \cdot \sqrt{m^2 - L^2} + m^2 \cdot \sin^{-1}\left(\frac{L}{m}\right) \right] \qquad (A.15)$$

Substitution of equations (A.2), (A.3), and (A.14) into equation (A.15) gives

$$a' = \frac{r'}{2}\left(\sin\theta + \theta \cdot \sec\theta \right) \qquad (A.16)$$

Thus

$$a = 4 \cdot a' = 2 \cdot r^2 \cdot (\sin\theta + \theta \cdot \sec\theta) \qquad (3),(A.16)$$

Finally, shadow factor becomes

$$SF = \frac{a}{A} = \frac{2 \cdot r^2 \cdot (\sin\theta + \theta \cdot \sec\theta)}{\pi \cdot r^2 \cdot \sec\theta}$$

$$= (\sin(2\theta) + 2\theta)/\pi \qquad (5),(A.17)$$

A caution here is that equation (A.17) can be applied only in calculating shadow factor for bare surfaces.

Appendix B.

Derivation of Shadow Factor for Coated Surfaces

When light passes through a medium of refractive index n_1 and hitting a light transmissible medium of refractive index n_2 with angle $\emptyset$, a refractive angle $\emptyset'$ occurs based on the law of refraction. This gives

$$\sin\emptyset' = \frac{n_1}{n_2}\sin\emptyset \qquad \text{(B.1)}$$

Recall the situation that light penetrates through a coating, reflects from a bottom surface, passes upward

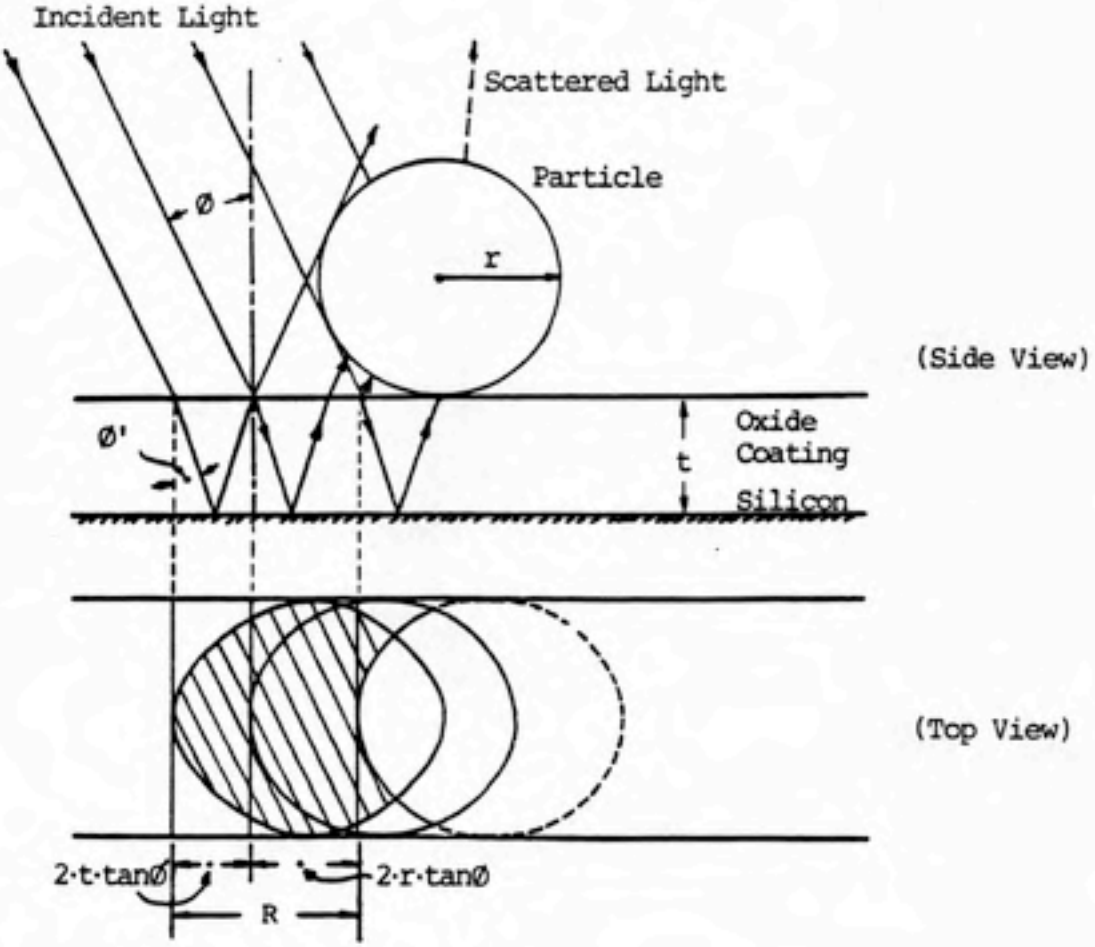

Figure B1

through the coating again, penetrates the top of the coating, and then hits the particle. This gives light more possibility to hit the particle for a distance of $2 \cdot t \cdot \tan\theta'$ moving toward inside of the particle. Figure B1 depicts this phenomenon. Consequently, the distance (R) between the centers of two ellipses becomes greater than that in the case of a bare surface. This gives R as

$$R = 2 \cdot r \cdot \tan\theta + 2 \cdot t \cdot \tan\theta' \qquad (B.2)$$

Here, t is the thickness of coating and θ' is the refractive angle which can be determined from equation (B.1).

The upper bound (L) of integration in equation (A.13) is a half of R. Let L' be the upper bound of integration in the case of an oxidized surface. Thus

$$L' = R/2 = r \cdot \tan\theta + t \cdot \tan\theta' \qquad (9),(B.3)$$

Recalling a' in equations (A.13) and (A.15) gives

$$a' = \int_0^{L'} \frac{n}{m} \sqrt{m^2 - x^2} \cdot dx$$

$$= \frac{n}{2 \cdot m} \left[L' \cdot \sqrt{m^2 - L'^2} + m^2 \cdot \sin^{-1}\left(\frac{L'}{m}\right) \right] \qquad (B.4)$$

Shadow factor becomes

$$SF = \frac{a}{A} = \frac{4 \cdot a'}{\pi \cdot m \cdot n}$$

$$= \frac{4 \cdot \frac{n}{2 \cdot m} \left[L' \cdot \sqrt{m^2 - L'^2} + m^2 \cdot \sin^{-1}\left(\frac{L'}{m}\right) \right]}{\pi \cdot m \cdot n}$$

$$= \frac{2}{\pi} \left[\frac{L'}{m^2} \sqrt{m^2 - L'^2} + \sin^{-1}\left(\frac{L'}{m}\right) \right] \qquad (B.5)$$

Equation (B.5) is the general equation for calculating shadow factors. When coating thickness is zero in the case of a bare surface, L'in equation (B.3) becomes identical to L in equation (A.14), and equation (B.5) can be simplified to equation (A.17).

Appendix C.

COMPUTER PROGRAM FOR LIGHT SCATTERING FROM PARTICLES ON REFLECTIVE SURFACES

C.1 Guide to Users

This command-driven program, written in Turbo Pascal running on a personal computer, is a computation tool specially designed for light scattering from particles on reflective surfaces. It allows users to calculate Mie intensity as a function of particle size for Routes A, B1, B2 and B3. In addition to generating results given parameters, it can print data on the screen, on a printer as a format of report and on a file which is Lotus-acceptable.

This program is divided into several files of source program. These are a main program file, a text file and thirteen included files. Source programs should be compiled before running. These thirteen included files will be included in the main program during compiling. A command file, with extension of .COM, will be generated for executing (executing can also be done inside of Turbo Pascal). The text file will be used only when running this command file. The relationship among these files can be presented as figure C.1. To reduce the computation time, an 8087 math coprocessor is recommended to install in the personal computer. Compiling should be performed under the mode of Turbo Pascal with 8087 math coprocessor, i.e., using TURBO-87.COM instead of TURBO.COM. This optional usage can save about 75% of computation time.

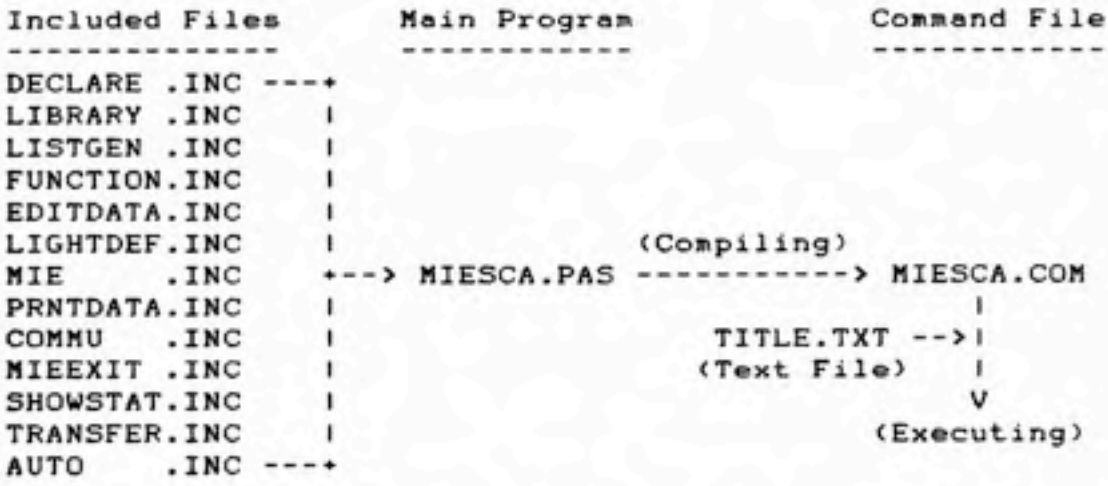

Figure C.1 The Relationship among all the files for Light Scattering from Particles on Reflective Surfaces

A command-driven style was designed in this program, because it is more convient to use. Main menu is given nine commands: AUTO, COMMUNICATE, EDIT, LIGHT, MIE, PRINT, QUIT, STATUS and TRANSFER, to control all the functions in this program. The function of each individual main command and its subcommands is shown in table C.1. There are two ways to use commands. One is moving high-light cursor to specific command by pressing directional keys and then hitting RETURN key. The other way is pressing the first character of the command you want, for example, pressing 'P' or 'p' for PRINT command. Some questions may be asked before executing a command for data needed in the program.

Table C.1 Main Commands, Subcommands, and their Functions

Main Command	Subcommand	Function
Auto	-	Automatically process a special program designed by users
Communicate	-	Transform data into Lotus-acceptable file allowing further treatments
Edit	-	Edit values for primary variables which all defaulted
Light	Combined	Define nonpolarized light
	Exit	Return to main command list
	Horizontal	Define parallel light only
	Vertical	Define perpendicular light
Mie	-	Make Mie theory calculations
Print	Printer	Make a report on the printer
	Exit	Return to main command list
	Screen	Print data on the screen
Quit	-	Terminate main program
Status	Dominant	Show dominance analysis
	Exit	Return to main command list
	Primary	Show primary variables
	Secondary	Show secondary variable
	Values-of-Shadow	Show values of shadow factors
Transfer	Load	Load a data file from a disk
	Save	Save data on a disk file
	Exit	Return to main command list

AUTO

AUTO is a special command designed for running a procedure which can be made by users who are familiar with Turbo Pascal language. These users can re-code the statements inside AUTO procedure of source program and re-compile the whole program. It gives users the results they want instead of repeating all the commands needed to run a series of data.

COMMUNICATE

Once data have been calculated, they may need to be transformed into a Lotus-acceptable file allowing further treatments such as statistical analysis and graphing. COMMUNICATE command provides this function. A drive name and a filename will be requested for saving the transformed data into a file which can be loaded by using Lotus commands of FILE-IMPORT-NUMBER.

EDIT

EDIT is a command to confirm or edit the default values for primary variables, which are defined as variables, have default values, are directly used in calculation for the results or for the other variables (secondary variables). Use directional keys to move highlight cursor to a particular variable. An 'Input a value :' question will appears at the left-top of the window for entering a new value. Press <RETURN> key will force program return to main

menu. Down and up arrow keys move cursor quicker. <Home> and <End> keys move cursor to the first and the last variables respectively.

LIGHT

LIGHT command is used to define whether a light source is polarized or nonpolarized. COMBINED subcommand allows to define a nonpolarized light source. EXIT subcommand returns the program to main menu. HORIZONTAL subcommand defines a light source with parallel (to scattering plane) component only. VERTICAL subcommand defines a perpendicular light source.

MIE

Once the primary variables have been confirmed or edited, the program is ready for making Mie calculations. MIE command starts the complicated calculation after a confirmation of yes.

PRINT

PRINT command is ready to use when finishing MIE command. PRINTER subcommand organizes data to a report format and prints them on a printer. EXIT subcommand returns the program to main menu. SCREEN subcommand prints data on the screen, dividing data into Routes A, B1, B2, B3 and total.

QUIT

To terminate the main program, use QUIT command followed by a confirmation of yes.

STATUS

STATUS command allows users to check the primary and secondary variables and look at the dominance indices and shadow factors. DOMINANT subcommand shows the results of dominance indices. EXIT subcommand returns the program to main menu. PRIMARY subcommand shows the current values of primary variables. SECONDARY subcommand shows the status of secondary variables. VALUES-OF-SHADOW subcommand lists all the shadow factors after a Mie calculation.

TRANSFER

TRANSFER command provides the program with abilities of transferring data in and out of disks. LOAD subcommand loads data from a file saved before into the program. SAVE subcommand saves data on a disk for further use. EXIT subcommand returns the program to main menu.

C.2 Program Statements

The program to calculate light scattering from particles on reflective surfaces is divided into fourteen parts. Each part was saved in each individual file. They are:

1. MIESCA.PAS (main program)
2. DECLARE.INC
3. LIBRARY.INC
4. LISTGEN.INC
5. FUNCTION.INC
6. EDITDATA.INC
7. LIGHTDEF.INC
8. MIE.INC
9. PRNTDATA.INC
10. COMMU.INC
11. MIEEXIT.INC
12. SHOWSTAT.INC
13. TRANSFER.INC
14. AUTO.INC

```
{ File Name : MIESCA.PAS ( main program ) }

Program MieScattering;

{$I Declare.Inc}

{$I Library.Inc}

{$I ListGen.Inc}

{$I Function.Inc}

{$I EditData.Inc}

{$I Lightdef.Inc}

{$I Mie.Inc}

{$I PrntData.Inc}

{$I Commu.Inc}

{$I MieExit.Inc}

{$I ShowStat.Inc}

{$I Transfer.Inc}

{$I Auto.Inc }

begin { main }
      Heading;
      ClrScr;
      ListGen;
      DefVar;
      while not exit do
      begin { while }
            border(1,4,79,25);
            case SelCom(1) of
                  'A' : Automa;
                  'C' : CommuLotus;
                  'E' : EditData;
                  'L' : LightDef;
                  'M' : Mie;
                  'P' : PrintData;
                  'Q' : MieExit;
                  'S' : ShowStatus;
                  'T' : Transfer;
                  else noise(1);
            end; { case }
      end; { while }
end. {main}
```

```
{ File Name : DECLARE.INC

  Purposes  : const section,
              type section,
              var section,
              default values
}
const
     MaxNameLen = 18;
     MaxMessLen = 70;
     NumComList = 7 ;( 1 : Main
                       2 : Light
                       3 : Print
                       4 : Print Route
                       5 : Communicate
                       6 : Status
                       7 : Transfer
                     )
     MaxListNum = 10;
     ComLevel   = 1;
     MaxList    = 60; { Maximum list for Mie calculation results }
     pi         = 3.141592654;

type
    TitleType = string[13]; { 13=3(' :')+MaxListLen('Communicate') }
    RemType = string[60];
    ComType = record
                  N  : string [MaxNameLen]; { Name }
                  X  : byte;                { x position }
                  Y  : byte;                { y position }
                  L  : byte;                { length of name }
                  M  : string [MaxMessLen]; { message }
                  P  : byte;                { previous command }
                  Nx : byte                 { next command }
              end;
    LightType = (Combined,Horizontal,Vertical);
    RouteType = (A,B1,B2,B3,Total);
    ValueType = array [0..MaxList] of Real;
    FNtype = string[14];

var
    ComList       : array [1..NumComList,1..MaxListNum] of ComType;
    Default       : array [1..NumComList] of byte;
    ListLen       : array [1..NumComList] of byte;
    Title         : array [1..NumComList] of TitleType;
    Exit          : boolean;
    ActCom,PreCom : byte;
    DeciNo        : integer;
    Light         : LightType;
    Route         : RouteType;
    RIpReal       : Real; { Real part of particle refractive index }
    RIpImag       : Real; { Imaginary part of particle index }
    Size          : ValueType; { size variable }
    MieIntensity1 : ValueType; { Mie intensity i(1) }
```

```
{ File Name : LIBRARY.INC

type          name                    function
----  -------------------- ------------------------------------------------

 p    ErrPrint(mess)        Print error message
 p    ReptCha(n,c)          Repeat characters
 p    ClrLine(n)            Clear a line
 p    ClRange(u,l)          Clear a range
 p    LightCell(x,y,l)      HighLight a defined cells
 p    NormScr               Turn the text mode  back to normal
 p    NorPrint(x,y,l,s)     Print a string in normal text mode
 p    KeyCheck(CheKey,ch)   Check the key pressed on the console
 p    Border(x1,y1,x2,y2)   Draw a border
 p    Noise(sel)            Make a variety of sounds
 p    MessLine(message)     Write a message on th 3rd line
----  -------------------- ------------------------------------------------
}
type
    strtype=string[80]; { string type }
    KeyType=(F1,F2,F3,F4,F5,F6,F7,F8,F9,F10,LA,RA,UA,DA,Home,EndKey,
             Spc,Resume,Tab,Return,NumKey,ChaKey,SymKey);

var
   CheKey : KeyType;
   ch     : char;

procedure ErrPrint(message:strtype);
begin { ErrPrint }
      textcolor(23);
      gotoxy(81-length(message),24);
      write(message);
      textcolor(15);
end; { ErrPrint }

procedure ReptCha(n,c:byte); { Repeat character given ASCII code by
                               n times }
var i:byte;
begin { ReptCha }
     for i:=1 to n do
     write(chr(c));
end; { ReptCha }

procedure ClrLine(n:byte); { clear a line given a line number }
begin { clrline }
      if n in [1..24]
         then begin
                 gotoxy(1,n);clreol;
              end
         else errprint('ClrLine parameter error');
end; { clrline }

procedure ClRange(u,l:byte); { clear a range given upper and lower bound }
var i:byte;
begin { clrange }
```

```
      if (u in [1..24]) and (l in [1..80])
         then begin
                 for i:=u to l do
                     begin { for }
                         gotoxy(1,i);
                         clreol;
                     end; { for }
              end
         else errprint('ClRange parameter(s) error');
end; { clrange }

procedure LightCell(x,y,l:byte); { HighLight the cells given the
                                   length=l at the beginning position of (x,y) }
begin { LightCell }
     if (x in [1..80]) and (y in [1..24]) and (l in [1..80])
        then begin
                gotoxy(x,y);
                textcolor(16);
                textbackground(7);
                reptcha(l,32);
                gotoxy(x,y);
             end
        else errprint('LightCell parameter(s) error');
end; { LightCell }

procedure NormScr;
begin { NormScr }
      textcolor(7);
      textbackground(1);
end; { NormScr }

procedure NorPrint(x,y,l:byte;s:strtype);
begin { NorPrint }
      gotoxy(x,y);
      NormScr;
      write(s);
      reptcha(l-length(s),32);
end; { NorPrint }

procedure KeyCheck(var CheKey:KeyType; var ch:char);
begin { CheKey }
      while  not keypressed do;
      if keyPressed then
      begin  { KeyPressed }
           read(Kbd,Ch);
           if (ch=#27) and KeyPressed then
              begin { extended codes }
                    read(kbd,ch);
                    case ch of
                         #59 : CheKey:=F1;
                         #60 : CheKey:=F2;
                         #61 : CheKey:=F3;
                         #62 : CheKey:=F4;
                         #63 : CheKey:=F5;
```

```
                        #64 : CheKey:=F6;
                        #65 : CheKey:=F7;
                        #66 : CheKey:=F8;
                        #67 : CheKey:=F9;
                        #68 : CheKey:=F10;
                        #75 : CheKey:=LA;
                        #77 : CheKey:=RA;
                        #72 : CheKey:=UA;
                        #80 : CheKey:=DA;
                        #71 : CheKey:=Home;
                        #79 : CheKey:=EndKey;
                      { #73 : CheKey:=PgUp; Not yet defined in KeyType
                        #81 : CheKey:=PgDn;                 "
                        #82 : CheKey:=Ins;                  "
                        #83 : CheKey:=Del;                  "}
                    end; { case }
                end { extended codes }
            else case ch of { single code }
                        #47 : CheKey:=Resume;
                        #32 : CheKey:=Spc;
                    {   #27 : CheKey:=Esc; }
                        #9  : CheKey:=Tab;
                        #13 : CheKey:=Return;
                        #48..#57 : CheKey:=NumKey;
                        #65..#90 : CheKey:=ChaKey;
                        #97..#122: CheKey:=ChaKey;
                        #33..#46 : CheKey:=SymKey;
                        #58..#64 : CheKey:=SymKey;
                        #91..#96 : CheKey:=SymKey;
                        #123..#127: CheKey:=SymKey;
                     { #8  : CheKey:=BackSp;          Not yet defined }
                end; { single code }
        end;  {KeyPressed}
end; { CheKey }

procedure Border(x1,y1,x2,y2:byte);
var i:byte;
begin { Border }
        TextColor(3);
        if (x2-x1<3) or (y2-y1<3) then errprint('Border parameter(s) error')
        else begin
                gotoxy(x1,y1);write(chr(201));
                reptcha(x2-x1-1,205);write(chr(187));
                for i:=y1+1 to y2-1 do begin
                                        gotoxy(x1,i);write(chr(186));
                                        gotoxy(x2,i);write(chr(186));
                                    end;
                gotoxy(x1,y2);write(chr(200));
                reptcha(x2-x1-1,205);write(chr(188));
            end;
      NormScr;
end; { Border }

procedure noise(sel:byte); { Sound a warning noise }
```

```
var i:byte;
begin { noise }
      case sel of
           0 : write(chr(7));
           1 : begin { noise 1 }
                     for i:=1 to 4 do
                         begin
                             sound(120+(i-1)*100);delay(40);
                             nosound;delay(2);
                         end;
               end; { noise 1 }
           2 : begin { noise 2 }
                     for i:=4 downto 1 do
                         begin
                              sound(120+(i-1)*100);delay(40);
                              nosound;delay(2);
                         end;
               end; { noise 2 }
           3 : begin { noise 3 - Symphony No. 5 Beethoven }
                     for i:=1 to 3 do
                         begin
                              sound(170);delay(150);
                              nosound;delay(10);
                         end;
                     sound(140);delay(800);nosound;
               end; { noise 3 }
           4 : begin { noise 4 - step }
                     sound(50);delay(50);
                     nosound;
               end; { noise 4 }
          else noise(1);
      end; { case }
end; { noise }

procedure MessLine(mess:strtype);
begin { MessLine }
      ClrLine(ComLevel+2);
      gotoxy(2,Comlevel+2);
      TextColor(14);
      write(mess);
      NormScr;
end; { MessLine }

procedure CR;
begin { CR }
      ClRange(ComLevel,ComLevel+2);
end; { CR }

procedure ComPrint(z:byte);
var i:byte;
begin { ComPrint }
      gotoxy(1,ComLevel);
      write(title[z]);
      for i:=1 to ListLen[z] do
```

```
          with ComList[z,i] do
          begin { with }
                gotoxy(x,y);
                write(N);
          end; { with }
end; { ComPrint }

procedure lightcom(a,b:byte);
begin { lightcom }
      with ComList[a,b] do
      begin
           lightcell(x,y,l);
           write(N);
           NormScr;
           MessLine(M);
      end;
end; { lightcom }

procedure darkcom(a,b:byte);
begin { darkcom }
      with ComList[a,b] do
      begin
      Norprint(x,y,l,n);
      end;
end; { darkcom }

function SelCom(ListNo:byte):Char;
var
    exit1, exit2, exit3 : boolean;
    idx : byte;

function FirstCha(a,b:byte):char;
var s:string[20];
begin { FirstCha}
      s:=ComList[a,b].N;
      delete(s,3,length(s)-2);
      delete(s,1,1);
      FirstCha:=upcase(s);
end; { FirstCha }

begin { SelCom }
      exit1:=false;
      while not exit1 do
      begin { while }
      CR;
      exit2:=false;
      ComPrint(ListNo);
      repeat
      darkcom(ListNo,PreCom);
      lightcom(ListNo,ActCom);
      PreCom:=ActCom;
      KeyCheck(CheKey,ch);
      with ComList[ListNo,ActCom] do
      case CheKey of
```

```
            RA,DA,Tab      : ActCom:=Nx; ( Next com )
            LA,UA          : ActCom:=P;  ( Prev com )
            Home           : ActCom:=1;  ( First com )
            EndKey         : ActCom:=listlen[ListNo]; ( Last com )
            Chakey,Return  : begin
                             ch:=upcase(ch);
                             exit1:=true;
                             exit2:=true;
                             if CheKey=Chakey
                                then begin
                                     SelCom:=ch;
                                     idx:=1;exit3:=false;
                                     repeat
                                     if ch=FirstCha(ListNo,idx)
                                        then begin
                                             ActCom:=idx;
                                             exit3:=true;
                                             end
                                        else idx:=idx+1;
                                      until (idx>Listlen[listNo]) or Exit3;
                                      darkcom(ListNo,PreCom);
                                      lightcom(ListNo,ActCom);
                                      end
                                else SelCom:=FirstCha(ListNo,ActCom);
                             end;
       end; ( case )
       until exit2;
       end; ( while )
end; ( SelCom )

procedure Heading;
type
 string12 = string[12];
procedure readtextfile( filename : string12);
var
   line     : string[100];
   infile   : text;
begin (readtextfile)
       TextColor(1); ( Blue )
       TextBackground(7); ( Gray )
       ClrScr;
       assign( infile, filename);
       reset( infile );
       while not eof(infile) do
       begin
            readln(infile, line);
            writeln(line);
       end;
       close( infile);
       NormScr;
end;(readtextfile)
begin(Heading)
     clrscr;
     readtextfile('title.txt');
```

```
      noise(1);
      delay(5000);
end;{Heading}

procedure finished(s:RemType);
begin { finished }
      noise(1);
      LightCell(79-Length(s),2,Length(s));
      write(s);
      delay(2000);
      NormScr;
end; { finished }
```

```
{ File Name : LISTGEN.INC }

procedure ListGen;
begin
{ 1 : Main }
       default[1]:=3;
       listlen[1]:=9;
       title[1]:='Command : ';
{ AUTO }
       with comlist[1,1] do
       begin
            N:=' Auto ';  X:=11;  Y:=ComLevel;  L:=6;
            M:='Automatically process a special program designed by users ';
            P:=9; Nx:=2;
       end; { AUTO }
{ Communicate }
       with comlist[1,2] do
       begin
             N:=' Communicate '  ;  X:=17;  Y:=ComLevel;  L:=13;
             M:='Transform data into Lotus (File Import) acceptable file  ';
             P:=1; Nx:=3;
       end; { Communicate }
{ Edit }
       with comlist[1,3] do
       begin
             N:=' Edit '  ;  X:=30;  Y:=ComLevel;  L:=6;
             M:='Edit the primary variables '; P:=2; Nx:=4;
       end; { Edit }
{ Light }
       with comlist[1,4] do
       begin
             N:=' Light '  ;  X:=36;  Y:=ComLevel;  L:=7;
             M:='Subcommand :  Combined  Exit  Horizontal  Vertical ';
             P:=3;Nx:=5;
       end; { light }
{ Mie }
       with comlist[1,5] do
       begin
            N:=' Mie ';  X:=43;  Y:=ComLevel;  L:=5;
            M:='Mie calculation ';  P:=4;  Nx:=6;
       end; { Mie }
{ Print }
       with comlist[1,6] do
       begin
             N:=' Print '  ;  X:=48;  Y:=ComLevel;  L:=7;
             M:='Subcommand :  Printer  Screen  Exit';P:=5;Nx:=7;
       end; { Print }
{ Quit }
       with comlist[1,7] do
       begin
             N:=' Quit '   ;  X:=55;  Y:=ComLevel;  L:=6;
             M:='Exit Mie Scattering program ';  P:=6;  Nx:=8;
       end; { Quit }
{ Status }
```

```
        with comlist[1,8] do
        begin
        N:=' Status ';  X:=61;  Y:=ComLevel;  L:=8;
        M:='Subcommand: Dominance Exit Primary Secondary Values_of_Shadow';
        P:=7;  Nx:=9;
        end; { Status }
{ Transfer }
        with comlist[1,9] do
        begin
              N:=' Transfer ';  X:=69;  Y:=ComLevel;  L:=11;
              M:='Subcommand :  Load  Save  Exit';  P:=8;  Nx:=1;
        end; { Transfer }
{ 2 : Light }
        default[2]:=1;
        listlen[2]:=4;
        title[2]:='LIGHT : ';
{ Combined }
        with comlist[2,1] do
        begin
             N:=' Combined ';  X:=9;  Y:=ComLevel;  L:=10;
             M:='Use non-polarized light ( combine i(1) and i(2) ) ';
             P:=4;Nx:=2;
        end; { Both }
{ Light-Exit }
        with comlist[2,2] do
        begin
             N:=' Exit ';  X:=19;  Y:=ComLevel;  L:=6;
             M:='Return to main command list ';  P:=1;  Nx:=3;
        end; { Light-Exit }
{ Horizontal }
        with comlist[2,3] do
        begin
             N:=' Horizontal ';  X:=25;  Y:=ComLevel;  L:=12;
             M:='Use parallel light i(1) only  ';  P:=2;  Nx:=4;
        end; { Horizontal }
{ Vertical }
        with comlist[2,4] do
        begin
             N:=' Vertical ';  X:=37;  Y:=ComLevel;  L:=10;
             M:='Use perpendicular light i(2) only  '; P:=3;  Nx:=1;
        end; { Vertical }
{ 3 : Print }
        default[3]:=2;
        listlen[3]:=3;
        title[3]:='PRINT : ';
{ Printer }
        with comlist[3,1] do
        begin
             N:=' Printer ';  X:=9;  Y:=ComLevel;  L:=9;
             M:='Print the results to the printer ';  P:=3;  Nx:=2;
        end; { Printer }
{ Screen }
        with comlist[3,2] do
        begin
```

```
            N:=' Screen ';   X:=18;  Y:=ComLevel;  L:=8;
            M:='Print the results on the screen ';  P:=1;  NX:=3;
        end; { screen }
{ Print-Exit }
        with comlist[3,3] do
        begin
            N:=' Exit ';  X:=26;  Y:=ComLevel;  L:=6;
            M:='Return to main command list ';  P:=2;  NX:=1;
        end; { Print-Exit }
{ 4 : Route for Print }
        default[4]:=1;
        listlen[4]:=6;
        title[4]:='ROUTE :  ';
{ A }
        with comlist[4,1] do
        begin
            N:=' A-Route_A ';  X:=9;  Y:=ComLevel;   L:=11;
            M:='Print the Mie intensities of route A ';P:=6;Nx:=2;
        end; { A }
{ B1 }
        with comlist[4,2] do
        begin
            N:=' B-Route_B1 ';  X:=20;  Y:=ComLevel;  L:=12;
            M:='Print the Mie intensities of route B1 ';P:=1;Nx:=3;
        end; { B }
{ B2 }
        with comlist[4,3] do
        begin
            N:=' C-Route_B2 ';  X:=32;  Y:=ComLevel;  L:=12;
            M:='Print the Mie intensities of route B2 ';P:=2;Nx:=4;
        end; { B2 }
{ B3 }
        with comlist[4,4] do
        begin
            N:=' D-Route_B3 ';  X:=44;  Y:=ComLevel; L:=12;
            M:='Print the Mie intensities of route B3 ';P:=3;Nx:=5;
        end; { B3 }
{ Print Route-Exit }
        with  comlist[4,5] do
        begin
            N:=' E-Exit ';  X:=56;  Y:=ComLevel;  L:=8;
            M:='Return to PRINT command list ';P:=4;Nx:=6;
        end; { Print Printer Route-Exit }
{ Total }
        with comlist[4,6] do
        begin
            N:=' F-Total ';  X:=64;  Y:=ComLevel;  L:=9;
            M:='Print the total Mie intensities  ';P:=5;Nx:=1;
        end; { All Route }
{ 5 : Communicate }
       default[5]:=2;
       listlen[5]:=3;
       title[5]:='COMMUNICATE :';
{ Chart }
```

```
        with comlist[5,1] do
        begin
             N:=' Chart ';   X:=16;  Y:=ComLevel;  L:=7;
             M:='Convert data to Chart acceptable file ';  P:=3;  Nx:=2;
        end; { Chart }
{ Lotus 1-2-3 }
        with comlist[5,2] do
        begin
             N:=' Lotus 1-2-3 ';  X:=23;  Y:=ComLevel;  L:=13;
             M:='Convert data to Lotus (File Import) acceptable file ';
             P:=1;Nx:=3;
        end; { Lotus }
{ Communicate-Exit }
        with comlist[5,3] do
        begin
             N:=' Exit ';  X:=36;  Y:=ComLevel;  L:=6;
             M:='Return to main command list ';  P:=2;  Nx:=1;
        end; { Communicate-Exit }
{ 6 : Status }
        default[6]:=3;
        listlen[6]:=5;
        title[6]:='STATUS : ';
{ Dominant }
        with comlist[6,1] do
        begin
             N:=' Dominance ';  X:=10;  Y:=ComLevel;  L:=11;
             M:='Show the dominance analysis of the Mie intensities ';
             P:=5;  Nx:=2;
        end; { Dominance }
{ Status - Exit }
        with comlist[6,2] do
        begin
             N:=' Exit ';  X:=21;  Y:=ComLevel;  L:=6;
             M:='Return to main command list ';  P:=1;  Nx:=3;
        end; { Status - Exit }
{ Primary }
        with comlist[6,3] do
        begin
             N:=' Primary ';  X:=27;  Y:=ComLevel;  L:=9;
             M:='Show the status of primary variables ';  P:=2;  Nx:=4;
        end; { Primary }
{ Secondary }
        with comlist[6,4] do
        begin
             N:=' Secondary ';  X:=36;  Y:=ComLevel;  L:=11;
             M:='Show the status of secondary variables ';  P:=3;  Nx:=5;
        end; { Secondary }
{ Values of Shadow }
        with comlist[6,5] do
        begin
             N:=' Values_of_Shadow ';  X:=47;  Y:=ComLevel;  L:=18;
             M:='Show the status of shadow values ';  P:=4;  Nx:=1;
        end; { Values of Shadow }
{ 7 : Transfer }
```

```
        default[7]:=1;
        listlen[7]:=3;
        title[7]:='TRANSFER : ';
{ Load }
        with comlist[7,1] do
        begin
             N:=' Load ';  X:=12;  Y:=ComLevel;  L:=6;
             M:='Load a data file ( .MIE ) from a disk ';  P:=3;  Nx:=2;
        end; { Load }
{ Save }
       with comlist[7,2] do
       begin
            N:=' Save ';  X:=18;  Y:=ComLevel;  L:=6;
            M:='Save data to a file ( .MIE ) in a disk ';  P:=1;  Nx:=3;
       end; { Save }
{ Transfer - Exit }
        with comlist[7,3] do
        begin
             N:=' Exit ';  X:=24;  Y:=ComLevel;  L:=6;
             M:='Return to main command list ';  P:=2;  Nx:=1;
        end; { Transfer - Exit }
end; { ListGen }
```

```
{ File Name : FUNCTION.INC }

procedure AngleCalc(Xd,Yd,Zd,Xp,Yp,Zp,Xs,Ys,Zs:real;var angle:real);
var
   { Xd,Yd,Zd : coordinate of detector }
   { Xp,Yp,Zp : coordinate of particle }
   { Xs,Ys,Zs : coordinate of light source }
   DS,DP,SP,
   coss : real;

begin { AngleCalc }
            DS:=sqrt(sqr(Xd-Xs)+sqr(Yd-Ys)+sqr(Zd-Zs));
            DP:=sqrt(sqr(Xd-Xp)+sqr(Yd-Yp)+sqr(Zd-Zp));
            SP:=sqrt(sqr(Xs-Xp)+sqr(Ys-Yp)+sqr(Zs-Zp));
            coss:=(sqr(DS)-sqr(DP)-sqr(SP))/(-2*DP*SP);
            angle:=arctan(sqrt(1/sqr(coss)-1));
            if coss<0 then angle:=180-(angle*180/pi)
            else angle:=angle*180/pi;
end; { AngleCalc }

function Tan(tann:real):real;
begin { Tan }
      Tan:=sin(tann)/cos(tann);
end; { tan }

function ArcSin(sinn:real):real;  { in radians }
var t:real;
begin { ArcSin }
      if (sinn>1) or (sinn<-1)
         then ErrPrint(' Parameter error in ArcSin function ')
         else begin
                  t:=ArcTan(sqrt(1/(1-sqr(sinn))-1));
                  if sinn<0 then ArcSin:=-t
                            else ArcSin:=t;
              end;
end; { ArcSin }

procedure Shadow(SA,Ds,RIs,Rp:real; var SV:real);
{ SA : Shadow angle, degree
  Ds : Thickness of coating surface, um
  RIs: Refractive Index of coating surface
  Rp : Radius of particle
  SV : Shadow value }
const
     RIa=1.0003; { Refractive Index of air }
var
   l,a,b,c:real;

begin { Shadow }
      SA:=SA*pi/180;
      Rp:=Rp*WL/pi/2;
      if Ds=0 then SV:=(sin(2*SA)+2*SA)/pi { Ds = 0 }
      else begin { Ds <> 0 }
                c:=ArcSin(RIa*sin(SA)/RIs);
```

```
                l:=Rp*tan(SA)+Ds*tan(c);
                a:=Rp/cos(aa);
                b:=Rp;
                if l>=a then SV:=1
                else SV:=2*(l*sqrt(a*a-l*l)/(a*a)+ArcSin(l/a))/pi;
            end; { Ds <> 0 }
end; { Shadow }

function Exist(FileSpec:FNtype):boolean;
var
   Fil:file;
begin { Exist }
      Assign(Fil,FileSpec);
      {$I-}
      Reset(Fil);
      {$I+}
      Exist:=(IOresult=0);
      close(Fil);
end; { Exist }
```

```
{ File Name : EDITDATA.INC }

procedure EditData;

label RTI;

const
     VarListLen = 19;
var
     x      : array [1..VarListLen] of byte;
     y      : array [1..VarListLen] of byte;
     Rem    : array [1..VarListLen] of RemType;
     RealNo : array [1..VarListLen] of real;
     IntNo  : array [1..VarListLen] of integer;
     exit   : boolean;

procedure ScreenEdit(Len:byte);
var
   i    : byte;
   quit : boolean;
   st   : RemType;

procedure GainData(x,y:byte;Rem:RemType;var RealNo:real;var IntNo:
                   integer;var CheKey:KeyType;var s:RemType);
label 11;

var x1,l:byte;
    ch:char;
    code:integer;

begin { GainData }
      code:=0;
      if IntNo=-99 then str(RealNo:6:3,s)
                   else str(IntNo,s);
      gotoxy(x,y);write(rem,' = ');
      x1:=x+length(rem)+3;
      lightcell(x1,y,8);
      if IntNo=-99 then write(RealNo:6:3)
                   else write(IntNo:6);
      NormScr;
      ClrLine(ComLevel+2);
  11: gotoxy(1,Comlevel+2);write('Input a value : ');
      KeyCheck(CheKey,ch);
      ClrLine(ComLevel+1);
      gotoxy(17,ComLevel+2);
      if (CheKey=NumKey) or (ch in ['.','-'])
      then begin { read number }
                write(ch);
                readln(s);s:=ch+s;
                if IntNo=-99 then val(s,RealNo,code)
                             else val(s,IntNo,code);
                if length(s)<6 then l:=length(s)
                               else l:=6;
                s:=copy(s,1,l);
```

```
                if code<>0 then begin (then)
                                        noise(1);
                                        gotoxy(1,ComLevel+1);
                                        write('Invalid character in',
                                              ' a number');
                                        goto 11;
                                    end ( then )
                                    else NorPrint(x1,y,8,s);
                Chekey:=RA;
            end; ( read number )
end; ( GainData )

procedure ResumeData(i:byte;r:real;int:integer);
begin ( ResumeData )
       case i of
            1 : Xs:=r;
            2 : Ys:=r;
            3 : Zs:=r;
            4 : Xp:=r;
            5 : Yp:=r;
            6 : Zp:=r;
            7 : Xd:=r;
            8 : Yd:=r;
            9 : Zd:=r;
            10: RIpReal:=r;
            11: RIpImag:=r;
            12: RIsReal:=r;
            13: MinAlpha:=r;
            14: MaxAlpha:=r;
            15: Intervals:=int;
            16: Dcs:=r;
            17: Refle1:=r;
            18: Refle2:=r;
            19: WL:=r;
       end; ( case )
end; ( ResumeData )

begin ( ScreenEdit )
      ClrLine(ComLevel);gotoxy(1,ComLevel);
      write('Edit . . . ');
      for i:=1 to Len do
      begin (for)
            gotoxy(x[i],y[i]);
            write(rem[i],' = ');
            if IntNo[i]=-99 then write(RealNo[i]:6:3)
                            else write(IntNo[i]:4);
      end; (for)
      i:=1;quit:=false;CheKey:=Return;st:='';
      repeat
            GainData(x[i],y[i],rem[i],RealNo[i],IntNo[i],CheKey,st);
            NorPrint(x[i]+length(rem[i])+3,y[i],8,st);
            ResumeData(i,RealNo[i],IntNo[i]);
            case CheKey of
                 DA     : if i+3>Len then i:=i+3-Len
```

```
                                    else i:=i+3;
                    Spc,
                    Tab,RA : if i=Len then i:=1
                                 else i:=i+1;
                    UA     : if i-3<1 then i:=i-3+Len
                                      else i:=i-3;
                    LA     : if i=1 then i:=Len
                                 else i:=i-1;
                    Home   : i:=1;
                    EndKey : i:=Len;
                    Return : quit:=true;
                    else noise(1);
                end; {case}
        until quit;
end; { ScreenEdit }

begin { EditData }
        if auto then goto RTI;
        x[1]:=5 ;y[1]:=6 ;Rem[1]:='X of light   ';RealNo[1]:=Xs;
        IntNo[1]:=-99;
        x[2]:=30;y[2]:=6 ;rem[2]:='Y of light   ';RealNo[2]:=Ys;
        IntNo[2]:=-99;
        x[3]:=55;y[3]:=6 ;rem[3]:='Z of light   ';RealNo[3]:=Zs;
        IntNo[3]:=-99;
        x[4]:=5 ;y[4]:=7 ;rem[4]:='X of particle';RealNo[4]:=Xp;
        IntNo[4]:=-99;
        x[5]:=30;y[5]:=7 ;rem[5]:='Y of particle';RealNo[5]:=Yp;
        IntNo[5]:=-99;
        x[6]:=55;y[6]:=7 ;rem[6]:='Z of particle';RealNo[6]:=Zp;
        IntNo[6]:=-99;
        x[7]:=5 ;y[7]:=8;rem[7]:='X of detector';RealNo[7]:=Xd;
        IntNo[7]:=-99;
        x[8]:=30;y[8]:=8;rem[8]:='Y of detector';RealNo[8]:=Yd;
        IntNo[8]:=-99;
        x[9]:=55;y[9]:=8;rem[9]:='Z of detector';RealNo[9]:=Zd;
        IntNo[9]:=-99;
        x[10]:=5;y[10]:=10;rem[10]:=
        'Refractive index of particle . . . . . Real part     ';
        RealNo[10]:=RIpReal;IntNo[10]:=-99;
        x[11]:=46;y[11]:=11;rem[11]:='Imaginary part';
        RealNo[11]:=RIpImag;IntNo[11]:=-99;
        x[12]:=5;y[12]:=12;rem[12]:=
        'Refractive index of coating surface . .(real part only)';
        RealNo[12]:=RIsReal;IntNo[12]:=-99;
        x[13]:=5;y[13]:=14;rem[13]:='Minimum size parameter (alpha) ';
        RealNo[13]:=MinApha;IntNo[13]:=-99;
        x[14]:=5;y[14]:=15;rem[14]:='Maximum size parameter (alpha) ';
        RealNo[14]:=MaxApha;IntNo[14]:=-99;
        x[15]:=5;y[15]:=16;rem[15]:='Number of intervral            ';
        RealNo[15]:=0;IntNo[15]:=Intervals;
        x[16]:=5;y[16]:=18;rem[16]:='Thickness of coating layer     ';
        RealNo[16]:=Dcs;IntNo[16]:=-99;
        x[17]:=5;y[17]:=19;rem[17]:='Reflectivity of bottom surface ';
        RealNo[17]:=Refle1;IntNo[17]:=-99;
```

```
      x[18]:=21;y[18]:=20;rem[18]:='coating layer  ';
      RealNo[18]:=Refle2;IntNo[18]:=-99;
      x[19]:=5;y[19]:=22;rem[19]:='Wave length (um)                ';
      RealNo[19]:=WL;IntNo[19]:=-99;
      ScreenEdit(VarListLen);
RTI:  AngleCalc(Xd,Yd,Zd,Xp,Yp,Zp,Xs,Ys,Zs,Theta);
      ThetaA:=180-Theta;
      AngleCalc(Xd,Yd,Zd,Xp,Yp,Zp,Xs,Ys,2*Zp-Zs,Theta);
      ThetaB1:=180-Theta;
      AngleCalc(Xd,Yd,2*Zp-Zd,Xp,Yp,Zp,Xs,Ys,Zs,Theta);
      ThetaB2:=180-Theta;
      AngleCalc(Xd,Yd,2*Zp-Zd,Xp,Yp,Zp,Xs,Ys,2*Zp-Zs,Theta);
      ThetaB3:=180-Theta;
      AngleCalc(Xp,Yp,Zd,Xp,Yp,Zp,Xs,Ys,Zs,ShadowAngleB1);
      AngleCalc(Xd,Yd,Zd,Xp,Yp,Zp,Xp,Yp,Zs,ShadowAngleB2);
      MieSecOK:=true;
      if not auto then
      begin
           ClRange(4,25);
           border(1,4,79,25);
           ActCom:=3;
      end;
end; { EditData }
```

```
{ File Name : LIGHTDEF.INC }

procedure LightMess(s:LightType);
begin { LightMess }
      gotoxy(60,4);
      write(' Light : ');
      case s of
           Combined   : write('Combined ');
           Horizontal : write('Horizontal ');
           Vertical   : write('Vertical ');
      end; { case }
end; { LightMess }

procedure LightDef;
label 1;
var
   exit:boolean;
begin { LightDef }
      exit:=false;
      ActCom:=default[2];
      PreCom:=ComList[2,ActCom].P;
      while not exit do
      begin { while }
            border(1,4,79,25);
            LightMess(Light);
            case SelCom(2) of
                 'C' : Light:=Combined;
                 'E' : begin
                            exit:=true;
                            goto 1;
                       end;
                 'H' : Light:=Horizontal;
                 'V' : Light:=Vertical;
                 else noise(1);
            end; { case }
    1:end; { while }
    ActCom:=4;
end; { LightDef }
```

```
{ File Name : MIE.INC }

procedure Mie;

{
Procedure to compute Mie functions for particles.
Original from Revised version -  work copy of February 26, 1986
    by Dr. Parker C. Reist.

Modified by Rong Chun Yu on May 14, 1986.

The basic formulation for these calculations are taken from the book
by Wichrasham entitled "Mie Theory Calculations". Computational routines
have been modified wherever possible to improve the accuracy of the
calculations.
}

var ch:char;

const
   Hi = 1E37;
   Lo =1E-37;
   Con = 57.29577951;
   Test = 1.0e-4;
   MaxIteration  =  200;

type
    ary = array[-1..MaxIteration] of real;

var
   XX,Y,SUM,SCAT,OLDSUM,I1,I2,R,S,L1,L2,L3,L4,
   YMIN,YMAX,QS,X2,M2,YR,YI,Z1,Z2,Z3 : REAL;
   N,U1,V1,BIGX,BIGY,PX,PY,OX,OY,LITTLEY,MARKER :INTEGER;
   PC,A_REAL,B_REAL,A_IM,B_IM       :REAL;
   AR,AI,OLDAR,OLDAI       :REAL;
   S1_REAL,S1_IM,S2_REAL,S2_IM  :REAL;
   PIE,PIE1,PIE2,TAU,TAU1,TAU2    :REAL;
   T,BR1,BR2,BI1,BI2,U,V                    :REAL;
   INCREMENT                    :REAL;
   Theta1,Apha1,RIpImag1:real;
   STEP,J:     INTEGER;

function Log(x:real):real;
begin
  if x=0 then Log:=-99
  else
  Log:= Ln(x)/Ln(10);
end; { Log }

function Power_10(x:real):real;
begin
   Power_10:= Exp(x*Ln(10));
end; { Power_10 }
```

```
procedure VADD(var a,b,c,d,e,f:real);
begin
    e:=a+c;
    f:=b+d;
end; { VADD }

procedure VSUB(var a,b,c,d,e,f:real);
begin
    e:=a-c;
    f:=b-d;
end; { VSUB }

procedure VMPY(var a,b,c,d,e,f:real);
begin
    e:=a*c-b*d;
    f:=a*d+b*c;
end; { VMPY }

procedure VDIV(var a,b,c,d,e,f:real);
begin
    e:=(a*c+b*d)/(c*c+d*d);
    f:=(b*c-a*d)/(c*c+d*d);
end; { VDIV }

procedure VINV(var a,b,e,f:real);
begin
    e:=a/(a*a+b*b);
    f:=-b/(a*a+b*b);
end; { VINV }

procedure wait;
begin { wait }
     TextColor(16);
     TextBackground(7);
     gotoxy(73,1);
     write(' Wait ');
     NormScr;
     gotoxy(3,5);
end; { wait }


PROCEDURE GET_A(VAR YR,YI,AR,AI:REAL;N:INTEGER);
VAR
   IMCOS,IMSIN,O1,O2,O3,O4,O5,O6 :REAL;
BEGIN
     IF N=0 THEN
     BEGIN
          IMCOS:= EXP(YI) + EXP(-YI);
          IMSIN:= EXP(YI) - EXP(-YI);
          L1:=IMCOS*SIN(YR);
          L2:=IMSIN*COS(YR);
          L3:=IMCOS*COS(YR);
          L4:=-IMSIN*SIN(YR);
          VDIV(L3,L4,L1,L2,AR,AI);
```

```
            OLDAR:=AR;
            OLDAI:=AI;
        END  ELSE
        BEGIN
            VINV(YR,YI,O1,O2);
            O1:=O1*N;
            O2:=O2*N;
            VSUB(O1,O2,OLDAR,OLDAI,O3,O4);
            VINV(O3,O4,O5,O6);
            AR:=-O1+O5;
            AI:=-O2+O6;
            OLDAR:=AR;
            OLDAI:=AI;
        END;
END (PROCEDURE GET_A);

PROCEDURE COMPUTE_A_AND_B(VAR  Apha,RIpReal,RIpImag1:REAL;N:INTEGER);
VAR
   AR,BR,AI,BI:REAL;
   J     :INTEGER;
   ETA_REAL,ETA_IM   :ARY;
   AO,AB,P,Q,R,S  :REAL;
   PR1,PI1,PR2,PI2,PR3,PI3,PR4,PI4,PR5,PI5,PR6,PI6,ER,EI  :REAL;

BEGIN
     J:=N;
     IF J=O THEN
     BEGIN
          ETA_REAL[-1]:=COS(Apha);
          ETA_IM[-1]:=-SIN(Apha);
          ETA_REAL[J]:=SIN(Apha);
          ETA_IM[J]:=COS(Apha);
          GET_A(YR,YI,AR,AI,J);
          A_REAL:=O.O;
          A_IM:=O.O;
          B_REAL:=O.O;
          B_IM:=O.O;
     END ELSE IF J>O THEN
     BEGIN
          (GET A_REAL, A_IM)
          ETA_REAL[J]:= ((2*J-1)/Apha)*ETA_REAL[J-1]-ETA_REAL[J-2];
          ETA_IM[J]:=((2*J-1)/Apha)*ETA_IM[J-1]-ETA_IM[J-2];
          YR:=Apha*RIpReal;
          YI:=Apha*RIpImag1;
          GET_A(YR,YI,AR,AI,J);
          VDIV(AR,AI,RIpReal,RIpImag1,PR1,PI1);
          PR2:=PR1+J/Apha;
          PI2:=PI1;
          ER:=ETA_REAL[J];
          EI:=O.O;
          VMPY(PR2,PI2,ER,EI,PR3,PI3);
          PR4:=PR3 - ETA_REAL[J-1];
          PI4:= PI3;
          VMPY(PR2,PI2,ETA_REAL[J],ETA_IM[J],PR5,PI5);
```

```
            VSUB(PR5,PI5,ETA_REAL[J-1],ETA_IM[J-1],PR6,PI6);
            VDIV(PR4,PI4,PR6,PI6,A_REAL,A_IM);
            (GET B_REAL, B_IM)
            VMPY(AR,AI,RIpReal,RIpImag1,PR1,PI1);
            PR2:=PR1+J/Apha;
            PI2:=PI1;
            ER:=ETA_REAL[J];
            EI:=0.0;
            VMPY(PR2,PI2,ER,EI,PR3,PI3);
            PR4:=PR3 - ETA_REAL[J-1];
            PI4:=PI3;
            VMPY(PR2,PI2,ETA_REAL[J],ETA_IM[J],PR5,PI5);
            VSUB(PR5,PI5,ETA_REAL[J-1],ETA_IM[J-1],PR6,PI6);
            VDIV(PR4,PI4,PR6,PI6,B_REAL,B_IM);
      END;
END (PROCEDURE GET_A_AND _B);

PROCEDURE GET_PIE_AND_TAU(VAR ANGLE:REAL;J:INTEGER);
BEGIN
      IF J=1 THEN
      BEGIN
            PIE:=1.0;
            TAU:=COS(ANGLE);
            PIE2:=PIE;
            TAU2:=TAU;
      END ELSE IF J=2 THEN
      BEGIN
            PIE:=3*COS(ANGLE);
            TAU:=3*COS(2*ANGLE);
            PIE1:=PIE;
            TAU1:=TAU;
      END ELSE IF J>2 THEN
      BEGIN
            PIE:=COS(ANGLE)*((2*J-1)/(J-1))*PIE1 - J/(J-1)*PIE2;
            TAU:=COS(ANGLE)*(PIE-PIE2)-(2*J-1)*SIN(ANGLE)*SIN(ANGLE)
                  *PIE1 + TAU2;
            PIE2:=PIE1;
            TAU2:=TAU1;
            PIE1:=PIE;
            TAU1:=TAU;
      END;(IF J>2 SECTION)
END (PROCEDURE GET_PIE_AND_TAU(ANGLE,N));

PROCEDURE COMPUTE_S_FACTORS(VAR ANGLE:REAL);
VAR
   S1R,S2R,S1I,S2I      :REAL;
   SUM1R,SUM2R,SUM1I,SUM2I  :REAL;
BEGIN
      N:=0;
      SUM1R:=1;
      SUM2R:=0;
      SUM1I:=0;
      SUM2I:=0;
      S1R:=1;
```

```
      IF ANGLE = 180/CON THEN ANGLE:= 179.999/con;
      IF ANGLE = 60/CON THEN  ANGLE:=59.99/CON;
(
THIS CORRECTION REMOVES GLITCH AT Theta=60 DEGREES AND Theta=180 DEGREES.
WATCH FOR PROBLEM IF LOOKING AT FINE DETAIL  AROUND 60 DEGREES -
CLOSER THAN 0.01 DEGREES, THEN MAY HAVE TO REVISE CORRECTION
)
      REPEAT
            COMPUTE_A_AND_B(Apha1,RIpReal,RIpImag1,N);
            GET_PIE_AND_TAU(angle,N);
            IF N>0 THEN
            BEGIN
                 S1R:= ((2*N+1)/(N*(N+1)))*(PIE*A_REAL + TAU*B_REAL);
                 S1I:= ((2*N+1)/(N*(N+1)))*(PIE*A_IM   + TAU*B_IM);
                 S2R:= ((2*N+1)/(N*(N+1)))*(PIE*B_REAL + TAU*A_REAL);
                 S2I:= ((2*N+1)/(N*(N+1)))*(PIE*B_IM   + TAU*A_IM);
                 IF N=1 THEN SUM1R:=0.0;
                 SUM1R:=SUM1R+S1R;
                 SUM1I:=SUM1I+S1I;
                 SUM2R:=SUM2R+S2R;
                 SUM2I:=SUM2I+S2I;
            END;
            N:=N+1;
      UNTIL ABS(((S1R+S1I+S2R+S2I)/(SUM1R+SUM1I+SUM2R+SUM2I)))<TEST;
      S1_REAL:=SUM1R;
      S2_REAL:=SUM2R;
      S1_IM  :=SUM1I;
      S2_IM  :=SUM2I;
END (PROCEDURE COMPUTE_S_FACTORS);

PROCEDURE MieCalcbySize;
label 1;
var
   RouteN:integer; ( 1-> Route A,
                     2-> Route B1,
                     3-> Route B2,
                     4-> Route B3 )
   Coef,TempSum,ToA,ToB1,ToB2,ToB3,ToAll:real;
   StepA,StepB1,StepB2,StepB3            :integer;

BEGIN
      if not MieSecOK
         then begin
                   write(chr(7));
                   MessLine('# # Variables not completely defined !'+
                            ' Execute <Edit> first.');
                   delay(3500);
                   goto 1;
              end;
      wait;
      RIpImag1:=-RIpImag;
      INCREMENT:=(MaxApha-MinApha)/(INTERVALS-1);
      YMIN:=HI;
      YMAX:=LO;
```

```
DominAP:=0;  DominB1P:=0;  DominB2P:=0;  DominB3P:=0;
DominAA:=0;  DominB1A:=0;  DominB2A:=0;  DominB3A:=0;
ToA:=0;  ToB1:=0;  ToB2:=0;  ToB3:=0;
StepA:=0;  StepB1:=0;  StepB2:=0;  StepB3:=0;
FOR  STEP:=1 TO INTERVALS DO
BEGIN
     gotoxy(73,3);write(STEP:2,' : ');
     Apha1:=MinApha+(STEP-1)*INCREMENT;
     Size[STEP]:=Apha1;
     Shadow(ShadowAngleB1,Dcs,RIsReal,Apha1,ShadowValueB11[step]);
     Shadow(ShadowAngleB1,0,RIsReal,Apha1,ShadowValueB12[step]);
     Shadow(ShadowAngleB2,Dcs,RIsReal,Apha1,ShadowValueB21[step]);
     Shadow(ShadowAngleB2,0,RIsReal,Apha1,ShadowValueB22[step]);
     YR:=RIpReal*Apha1;
     YI:=RIpImag1*Apha1;
     for RouteN:=1 to 4 do
     begin { for }
          gotoxy(78,3);write(RouteN:1);
          case RouteN of
               1:Theta1:=ThetaA/CON;
               2:Theta1:=ThetaB1/CON;
               3:Theta1:=ThetaB2/CON;
               4:Theta1:=ThetaB3/CON;
          end; { case }
          COMPUTE_S_FACTORS(Theta1);
          if Light=Vertical then MieIntensity1[STEP]:=0.0
             else MieIntensity1[STEP]:=S1_REAL*S1_REAL+S1_IM*S1_IM;
          if Light=Horizontal then MieIntensity2[STEP]:=0.0
             else MieIntensity2[STEP]:=S2_REAL*S2_REAL+S2_IM*S2_IM;
          IF MieIntensity1[STEP]>YMAX THEN YMAX:=MieIntensity1[STEP];
          IF MieIntensity1[STEP]<YMIN THEN YMIN:=MieIntensity1[STEP];
          IF MieIntensity2[STEP]>YMAX THEN YMAX:=MieIntensity2[STEP];
          IF MieIntensity2[STEP]<YMIN THEN YMIN:=MieIntensity2[STEP];
          S1_IM:=0.0;
          S2_IM:=0.0;
          S1_REAL:=0.0;
          S2_REAL:=0.0;
          case RouteN of
               1:begin
                       AIntensity1[STEP]:=MieIntensity1[STEP];
                       AIntensity2[STEP]:=MieIntensity2[STEP];
                       TempSum:=AIntensity1[STEP]+AIntensity2[STEP];
                       ToA:=ToA+TempSum;
                       if DominAP<TempSum
                          then begin
                                    DominAP:=TempSum;
                                    StepA:=STEP;
                               end;
                 end;
               2:begin
                       if Dcs<>0
                       then Coef:=Refle2*ShadowValueB12[STEP]+
                            (1.0-Refle2)*Refle1*ShadowValueB11[STEP]
                       else Coef:=Refle1*ShadowValueB12[STEP];
```

```
                        B1Intensity1[STEP]:=MieIntensity1[STEP]*Coef;
                        B1Intensity2[STEP]:=MieIntensity2[STEP]*Coef;
                        TempSum:=B1Intensity1[STEP]+B1Intensity2[STEP];
                        ToB1:=ToB1+TempSum;
                        if DominB1P<TempSum
                           then begin
                                   DominB1P:=TempSum;
                                   StepB1:=STEP;
                                end;
                     end;
                  3:begin
                        if Dcs<>0
                        then Coef:=Refle2*ShadowValueB22[STEP]+
                             (1.0-Refle2)*Refle1*ShadowValueB21[STEP]
                        else Coef:=Refle1*ShadowValueB22[STEP];
                        B2Intensity1[STEP]:=MieIntensity1[STEP]*Coef;
                        B2Intensity2[STEP]:=MieIntensity2[STEP]*Coef;
                        TempSum:=B2Intensity1[STEP]+B2Intensity2[STEP];
                        ToB2:=ToB2+TempSum;
                        if DominB2P<TempSum
                           then begin
                                   DominB2P:=TempSum;
                                   StepB2:=STEP;
                                end;
                     end;
                  4:begin
                        if Dcs<>0
                        then Coef:=(Refle2*ShadowValueB12[STEP]+
                             (1.0-Refle2)*Refle1*ShadowValueB11[STEP])*
                             (Refle2*ShadowValueB22[STEP]+
                             (1.0-Refle2)*Refle1*ShadowValueB21[STEP])
                        else Coef:=(Refle1*ShadowValueB12[STEP])*
                                   (Refle1*ShadowValueB22[STEP]);
                        B3Intensity1[STEP]:=MieIntensity1[STEP]*Coef;
                        B3Intensity2[STEP]:=MieIntensity2[STEP]*Coef;
                        TempSum:=B3Intensity1[STEP]+B3Intensity2[STEP];
                        ToB3:=ToB3+TempSum;
                        if DominB3P<TempSum
                           then begin
                                   DominB3P:=TempSum;
                                   StepB3:=STEP;
                                end;
                     end;
               end; { case }
          end; { for RouteN }
     end; { for STEP }
     if DominAP>0 then DominAP:=DominAP/(AIntensity1[StepA]+
                       AIntensity2[StepA]+
                       B1Intensity1[StepA]+B1Intensity2[StepA]+
                       B2Intensity1[StepA]+B2Intensity2[StepA]+
                       B3Intensity1[StepA]+B3Intensity2[StepA]);
     if DominB1P>0 then DominB1P:=DominB1P/(AIntensity1[StepB1]+
                        AIntensity2[StepB1]+
                        B1Intensity1[StepB1]+B1Intensity2[StepB1]+
```

```
                        B2Intensity1[StepB1]+B2Intensity2[StepB1]+
                        B3Intensity1[StepB1]+B3Intensity2[StepB1]);
      if DominB2P>0 then DominB2P:=DominB2P/(AIntensity1[StepB2]+
                        AIntensity2[StepB2]+
                        B1Intensity1[StepB2]+B1Intensity2[StepB2]+
                        B2Intensity1[StepB2]+B2Intensity2[StepB2]+
                        B3Intensity1[StepB2]+B3Intensity2[StepB2]);
      if DominB3P>0 then DominB3P:=DominB3P/(AIntensity1[StepB3]+
                        AIntensity2[StepB3]+
                        B1Intensity1[StepB3]+B1Intensity2[StepB3]+
                        B2Intensity1[StepB3]+B2Intensity2[StepB3]+
                        B3Intensity1[StepB3]+B3Intensity2[StepB3]);
      ToAll:=ToA+ToB1+ToB2+ToB3;
      DominAA:=ToA/ToAll;  DominB1A:=ToB1/ToAll;
      DominB2A:=ToB2/ToAll;  DominB3A:=ToB3/ToAll;
      if not Auto then finished('  End of Mie calculation  ');
    1:;
END; (PROCEDURE MieCalcbySize)

begin (Mie)
       if not auto then
       begin
            Clrline(ComLevel);
            gotoxy(1,ComLevel);write('COMMAND MIE . . . ');
            noise(1);
            messline('Ready to make Mie calculations? (Y/N) :  ');
            read(kbd,ch);write(upcase(ch));
            if upcase(ch)='Y' then  MieCalcbySize;
            ActCom:=5;
       end
       else MieCalcbySize;
       MieCalOK:=true;
end; ( Mie )
```

```
{ File Name : PRNTDATA.INC }

procedure PrintData;
type
    FunType = (PP,PS);
var
   exit:boolean;
   Fun : FunType;

procedure PSproc;
label 1,2,5,6,7,8,9;
type
    PaperType = string[132];
var
   step,nn,p,z,SkipN : integer;
   YN : char;
   tex:string[4];
   Date :string[8];
   Reporter,TempWord:PaperType;
   i1,i2,it,v1,v2,v3,v4,v5,v6,v7,v8,vt,siz:real;

function RepP(n,c:byte):PaperType;
var i:byte;
    t:PaperType;
begin { RepP }
      t:='';
      for i:=1 to n do
      t:=t+chr(c);
      RepP:=t;
end; { RepP }

procedure PrintTitle(I:byte);
begin { PrintTitle }
      writeln(LST,RepP(43,32),RepP(45,42));
      writeln(LST,RepP(43,32),
      '**  MIE  SCATTERING  CALCULATION  REPOPRT  **');
      writeln(LST,RepP(43,32),RepP(45,42),RepP(29,32),'Page : ',I:2);
      writeln(LST);
      writeln(LST,'Reporter : ',Reporter,RepP(106-Length(Reporter),32),
              'Date : ',Date);
      writeln(LST);
      writeln(LST,RepP(132,42));
      writeln(LST);
end; { PrintTitle }

begin { PSproc }
     if Fun=PS
     then begin { Print Screen }
     if Route=Total then tex:='';
     if Route=A then tex:='-A ';
     if Route=B1 then tex:='-B1';
     if Route=B2 then tex:='-B2';
     if Route=B3 then tex:='-B3';
     step:=0;
```

```
1: nn:=0;
   ClRange(4,25);
   border(1,4,79,25);
   gotoxy(3,5);
   write('Alpha          i(1)          i(2)      Total',tex);
   gotoxy(3,6);
   write('-----  ------------  ------------  ------------');
   for step:=step+1 to intervals do
   begin ( for )
         nn:=nn+1;
         gotoxy(3,nn+6);
         write(Size[step]:5:2);
         if Route=A then write('  ',AIntensity1[step]:12:4,'  ',
                         AIntensity2[step]:12:4,'  ',
                         AIntensity1[step]+AIntensity2[step]:12:4);
         if Route=B1 then write('  ',B1Intensity1[step]:12:4,'  ',
                          B1Intensity2[step]:12:4,'  ',
                          B1Intensity1[step]+B1Intensity2[step]:12:4);
         if Route=B2 then write('  ',B2Intensity1[step]:12:4,'  ',
                          B2Intensity2[step]:12:4,'  ',
                          B2Intensity1[step]+B2Intensity2[step]:12:4);
         if Route=B3 then write('  ',B3Intensity1[step]:12:4,'  ',
                          B3Intensity2[step]:12:4,'  ',
                          B3Intensity1[step]+B3Intensity2[step]:12:4);
         if Route=Total then
         begin ( Total )
               i1:=AIntensity1[step]+B1Intensity1[step]+
                   B2Intensity1[step]+B3Intensity1[step];
               i2:=AIntensity2[step]+B1Intensity2[step]+
                   B2Intensity2[step]+B3Intensity2[step];
               it:=i1+i2;
               write('  ',i1:12:4,'  ',i2:12:4,'  ',it:12:4);
         end; ( Total )
         if nn=17 then begin
                          MessLine('More data (Y/N)?  ');
                          read(kbd,YN);
                          write(upcase(YN));
                          if upcase(YN)='N' then goto 2
                                            else goto 1;
                       end; ( if )
   end; ( for )
 2:;
   end ( Print Screen )
   else begin ( Print Printer )
        noise(1);
        MessLine('Is your printer ready for 132 characters (Y/N) : ');
        gotoxy(50,3);read(kbd,YN);gotoxy(50,3);write(upcase(YN));
        if upcase(YN)<>'N' then
        begin ( ready to print out )
              gotoxy(3,6);write('Report Date (MM/DD/YY) : ________');
              gotoxy(28,6);readln(Date);
              gotoxy(3,8);
              write('Reporter : ______________________________');
              gotoxy(14,8);readln(Reporter);
```

```
5: gotoxy(3,10);
   write('Print out selection : 1 - Size parameter');
   gotoxy(25,11);write('2 - Size in um');
   gotoxy(23,12);write('? ');read(kbd,YN);write(YN);
   if YN='1' then tex:='PARA'
   else if YN='2' then tex:='(um)'
        else begin
                  noise(1);
                  goto 5;
             end;
   finished(' Wait ');
   PrintTitle(1);
   writeln(LST,RepP(52,32),'*** Primary Variables ***');
   writeln(LST,RepP(52,32),RepP(25,42));
   writeln(LST);
   writeln(LST,RepP(41,32),'Particle Refractive Index : ',
           RIpReal:5:3,' + ',RIpImag:5:3,' i');
   writeln(LST);
   writeln(LST,RepP(41,32),'Size Parameter : Maximum   -- ',
           MaxApha:7:4,' (= ',MaxApha*WL/pi:6:3,' um)');
   writeln(LST,RepP(58,32),'Minimum   -- ',MinApha:7:4,' (= ',
           MinApha*WL/pi:6:3,' um)');
   writeln(LST,RepP(58,32),'Intervals -- ',Intervals:2);
   writeln(LST);
   writeln(LST,RepP(41,32),'Coordinates    : Light Source -- ('

           Xs:7:3,',',Ys:7:3,',',Zs:7:3,')');
   writeln(LST,RepP(58,32),'Particle      -- (',Xp:7:3,',',
           Yp:7:3,',',Zp:7:3,')');
   writeln(LST,RepP(58,32),'Detector      -- (',Xd:7:3,',',
           Yd:7:3,',',Zd:7:3,')');
   writeln(LST);
   writeln(LST,RepP(41,32),'Incident Light  : Wave Length  -- '

           WL:7:4,' um');
   case Light of
        Combined   : TempWord:='None';
        Horizontal : TempWord:='Horizontal';
        Vertical   : TempWord:='Vertical';
   end; { case }
   writeln(LST,RepP(59,32),'Polarization --  ',TempWord);
   writeln(LST);
   writeln(LST,RepP(41,32),'Coating Layer   : Refractive',
           ' Index --  ',RIsReal:5:3);
   writeln(LST,RepP(59,32),'Thickness        -- ',
           Dcs:6:3,' um');
   writeln(LST,RepP(59,32),'Reflectance      --  ',
           Refle2:4:2);
   writeln(LST);
   writeln(LST,RepP(41,32),'Bottom Surface  : Reflectance',
           '     --   ',Refle1:4:2);
   writeln(LST);
   writeln(LST,RepP(132,42));
   for nn:=1 to 3 do
   writeln(LST);
   writeln(LST,RepP(52,32),'*** Secondary Variables ***');
```

```
  writeln(LST,RepP(52,32),RepP(27,42));
  writeln(LST);
  writeln(LST,RepP(41,32),'Route A  :  Scattering Angle -- ',
          ThetaA:6:2);
  writeln(LST);
  writeln(LST,RepP(41,32),'Route B1 :  Scattering Angle -- ',
          ThetaB1:6:2);
  writeln(LST,RepP(53,32),'Shadow Angle     -- ',
          ShadowAngleB1:6:2);
  writeln(LST);
  writeln(LST,RepP(41,32),'Route B2 :  Scattering Angle -- ',
          ThetaB2:6:2);
  writeln(LST,RepP(53,32),'Shadow Angle     -- ',
          ShadowAngleB2:6:2);
  writeln(LST);
  writeln(LST,RepP(41,32),'Route B3 :  Scattering Angle -- ',
          ThetaB3:6:2);
  writeln(LST,RepP(53,32),'Shadow Angle  1  -- ',
          ShadowAngleB1:6:2);
  writeln(LST,RepP(53,32),'Shadow Angle  2  -- ',
          ShadowAngleB2:6:2);
  writeln(LST);
  writeln(LST,RepP(132,42));
  writeln(LST);
  writeln(LST,RepP(48,32),'*** Dominance Analysis Index ***');
  writeln(LST);
  writeln(LST,RepP(52,32),'by Point          by Area');
  writeln(LST,RepP(52,32),'--------          -------');
  writeln(LST);
  writeln(LST,RepP(41,32),'Route A       ',DominAP:5:3,
                          '              ',DominAA:5:3);
  writeln(LST);
  writeln(LST,RepP(41,32),'Route B1      ',DominB1P:5:3,
                          '              ',DominB1A:5:3);
  writeln(LST);
  writeln(LST,RepP(41,32),'Route B2      ',DominB2P:5:3,
                          '              ',DominB2A:5:3);
  writeln(LST);
  writeln(LST,RepP(41,32),'Route B3      ',DominB3P:5:3,
                          '              ',DominB3A:5:3);
  writeln(LST);
  writeln(LST,RepP(132,42));
  for z:=1 to 23 do
  writeln(LST);
  p:=2;step:=0;SkipN:=0;
7:nn:=0;
  PrintTitle(p);
  writeln(LST,RepP(38,32),
          'SIZE        SHADOW        SHADOW        ',
          'SHADOW        SHADOW');
  writeln(LST,RepP(38,32),tex,
          '     FACTOR B11    FACTOR B12    ',
          'FACTOR B21    FACTOR B22');
  writeln(LST,RepP(38,32),'----      ----------    ',
```

```
          '----------   ----------   ----------');
    p:=p+1;
    for step:=step+1 to intervals do
    begin
      nn:=nn+1;SkipN:=SkipN+1;
      if YN='1' then siz:=size[step]
                else siz:=size[step]*WL/pi;
      writeln(LST,RepP(38,32),siz:5:2,RepP(7,32),
              ShadowValueB11[step]:6:3,RepP(7,32),
              ShadowValueB12[step]:6:3,RepP(7,32),
              ShadowValueB21[step]:6:3,RepP(7,32),
              ShadowValueB22[step]:6:3);
      if SkipN=5 then begin
                        writeln(LST);
                        SkipN:=0;
                        nn:=nn+1;
                      end;
      if nn=72 then begin
                      writeln(LST);
                      writeln(LST,RepP(132,42));
                      for z:=1 to 3 do
                      writeln(LST);
                      goto 8;
                    end;
    end;
    writeln(LST);
    writeln(LST,RepP(132,42));
    for z:=1 to 75-nn do
    writeln(LST);
8:  TempWord:='';
    for z:=1 to 4 do
    TempWord:=TempWord+(RepP(4,32)+'i(1)'+RepP(6,32)+
              'i(2)'+RepP(7,32)+'SUM ');
    step:=0;SkipN:=0;
6:  nn:=0;
    PrintTitle(p);
    writeln(Lst,' SIZE',RepP(10,32),'ROUTE  A',RepP(21,32),
            'ROUTE  B1',RepP(20,32),'ROUTE  B2',RepP(20,32),
            'ROUTE  B3',RepP(12,32),'TOTAL MIE');
    writeln(LST,' ',tex,TempWord,'  Intensity');
    writeln(LST,RepP(132,45));
    p:=p+1;
    for step:=step+1 to intervals do
    begin ( for )
          nn:=nn+1;SkipN:=SkipN+1;
          if YN='1' then siz:=size[step]
                    else siz:=size[step]*WL/pi;
          v1:=AIntensity1[step];
          v2:=AIntensity2[step];
          v3:=B1Intensity1[step];
          v4:=B1Intensity2[step];
          v5:=B2Intensity1[step];
          v6:=B2Intensity2[step];
          v7:=B3Intensity1[step];
```

```
                    v8:=B3intensity2[step];
                    vt:=v1+v2+v3+v4+v5+v6+v7+v8;
                    writeln(LST,siz:5:2,v1:9:3,' ',v2:9:3,' ',v1+v2:9:3,
                            v3:9:3,' ',v4:9:3,' ',v3+v4:9:3,
                            v5:9:3,' ',v6:9:3,' ',v5+v6:9:3,
                            v7:9:3,' ',v8:9:3,' ',v7+v8:9:3,' ',vt:10:3);
                    if SkipN=5 then begin
                                        writeln(LST);
                                        SkipN:=0;
                                        nn:=nn+1;
                                    end;
                    if nn=72 then begin
                                      writeln(LST);
                                      writeln(LST,RepP(132,42));
                                      for z:=1 to 3 do
                                      writeln(LST);
                                      goto 9;
                                  end;
                end; { for }
                writeln(LST);
                writeln(LST,RepP(132,42));
      9: end; { ready to print out }
    end; { Print Printer }
end; { PSproc }

procedure P_Route;
var
   exit:boolean;

begin { P_Route }
      exit:=false;
      ActCom:=default[4];
      PreCom:=ComList[4,ActCom].P;
      while not exit do
      begin { while }
            case SelCom(4) of
                 'A' : Route:=A;
                 'B' : Route:=B1;
                 'C' : Route:=B2;
                 'D' : Route:=B3;
                 'E' : exit:=true;
                 'F' : Route:=Total;
                 else noise(1);
            end; { case }
            if not exit then PSproc;
      end; { while }
      ActCom:=2;
      PreCom:=1;
end; { P_Route }

begin { PrintData }
      exit:=false;
      ActCom:=default[3];
      PreCom:=ComList[3,ActCom].p;
```

```
        while not exit do
        begin { while }
              ClrScr;
              border(1,4,79,25);
              case SelCom(3) of
                   'P' : begin
                              Fun:=PP;
                              PSproc;
                         end;
                   'S' : begin
                              Fun:=PS;
                              P_Route;
                         end;
                   'E' : exit:=true;
                   else noise(1);
              end; { case }
        end; { while }
        ClRange(5,23);
        ActCom:=6;
end; { PrintData }
```

```
{ File Name : COMMU.INC }

procedure CommuLotus;
label 1;
var
   R          : char;
   i          : integer;
   DriveName: string[1];
   FileName : string[8];
   FN         : FNType;
   DF         : Text;
   siz,v1,v2,v3,v4,v5,v6,v7,v8,vt,sf1,sf2,sf3,sf4:real;

begin { CommuLotus }
      ClrLine(3);
      gotoxy(1,3);write('Drive Name : _');
      gotoxy(14,3);read(DriveName);
      gotoxy(25,3);write('File Name : ________.PRN');
      gotoxy(37,3);readln(FileName);
      FN:=DriveName+':'+FileName+'.PRN';
      if Exist(FN) then
      begin
         finished(' File existed ');
         gotoxy(62,3);
         write('Replace (Y/N) : _');
         gotoxy(78,3);read(kbd,R);write(UpCase(R));
         if UpCase(R)<>'Y' then
            begin
                gotoxy(45,1);clreol;
                finished(' Save cancelled ');
                goto 1;
            end;
      end;
      gotoxy(79,3);
      Assign(DF,FN);Rewrite(DF);
      writeln(DF,'"Particle "','"Refractiv"','"e Index ="',RIpReal:5:3);
      writeln(DF,'"X ( p ) ="',Xp:6:3,'" "','"Y ( p ) ="',Yp:6:3,'" "',
              '"Z ( p ) ="',Zp:6:3);
      writeln(DF,'"X ( s ) ="',Xs:6:3,'" "','"Y ( s ) ="',Ys:6:3,'" "',
              '"Z ( s ) ="',Zs:6:3);
      writeln(DF,'"X ( d ) ="',Xd:6:3,'" "','"Y ( d ) ="',Yd:6:3,'" "',
              '"Z ( d ) ="',Zd:6:3);
      writeln(DF,'"Wave Leng"','"th (um) ="',WL:6:4);
      writeln(DF,'"Refractiv"','"e Index o"','"f Coating"','" Layer  ="',
              RIsReal:5:3);
      writeln(DF,'"Thickness"','" of Coati"','"ng Layer "','"( um )  ="',
              Dcs:6:4);
      writeln(DF,'"Reflectan"','"ce of Coa"','"ting Laye"','"r       ="',
              Refle2:4:2);
      writeln(DF,'"Reflectan"','"ce of Bot"','"tom Surfa"','"ce      ="',
              Refle1:4:2);
      writeln(DF,' ');
      writeln(DF,'"    Size "','"  i(1)-A "','"  i(2)-A "','" Total-A "',
              '" i(1)-B1 "','" i(2)-B1 "','" Total-B1"','" i(1)-B2 "',
```

```
            '" i(2)-B2 "',' Total-B2"',' i(1)-B3 "',' i(2)-B3 "',
            '" Total-B3"',' Tot-All "',
            '"S.F. B11"','"S.F. B12"','"S.F. B21"','"S.F. B22"');
      writeln(DF,'"    ---- "',' --------"',' --------"',' --------"',
            '" --------"',' --------"',' --------"',' --------"',
            '" --------"',' --------"',' --------"',' --------"',
            '" --------"',' --------"',' --------"',' --------"',
            '" --------"',' --------"');
      for i:=1 to intervals do
      begin
           siz:=size[i]*WL/pi;
           v1 :=AIntensity1[i];
           v2 :=AIntensity2[i];
           v3 :=B1Intensity1[i];
           v4 :=B1Intensity2[i];
           v5 :=B2Intensity1[i];
           v6 :=B2Intensity2[i];
           v7 :=B3Intensity1[i];
           v8 :=B3Intensity2[i];
           vt :=v1+v2+v3+v4+v5+v6+v7+v8;
           sf1:=ShadowValueB11[i];
           sf2:=ShadowValueB12[i];
           sf3:=ShadowValueB21[i];
           sf4:=ShadowValueB22[i];
          writeln(DF,siz:7:4,v1:9:4,v2:9:4,v1+v2:9:4,v3:9:4,v4:9:4,
                  v3+v4:9:4,v5:9:4,v6:9:4,v5+v6:9:4,v7:9:4,v8:9:4,
                  v7+v8:9:4,vt:10:4,sf1:6:3,sf2:6:3,sf3:6:3,sf4:6:3);
      end;
      close(DF);
      finished(' Lotus Translation completed ');
    1:;ActCom:=2;
end; ( CommuLotus )
```

```
{ File Name : MIEEXIT.INC }

procedure MieExit;
var c:char;
begin { MieExit }
      noise(1);
      messline('Confirm (Y/N) : ');
      read(kbd,c);write(upcase(c));
      if upcase(c)='Y' then
      begin
           ClrScr;
           writeln('Mie program terminated ! ');
           writeln;
           halt;
           end;
end; { MieExit }
```

```
{ File Name : SHOWSTAT.INC }

procedure ShowStatus;
var
   exit : boolean;

procedure ShowDomin;
begin { ShowDomin }
      if not MieCalOk then
      begin { Dominance analysis not yet defined }
            gotoxy(5,25);
            noise(1);
            write(' Dominance analysis have not been done yet ');
      end { Not done yet }
      else
      begin { Ready }
            ClRange(4,25);
            border(1,4,79,25);
            gotoxy(16,6);
            write('Dominance Analysis Index');
            gotoxy(16,7);
            write('by Point          by Area');
            gotoxy(16,8);
            write('--------          -------');
            gotoxy(5,10);
            write('Route A      ',DominAP:5:3,'             ',
                  DominAA:5:3);
            gotoxy(5,12);
            write('Route B1     ',DominB1P:5:3,'             ',
                  DominB1A:5:3);
            gotoxy(5,14);
            write('Route B2     ',DominB2P:5:3,'             ',
                  DominB2A:5:3);
            gotoxy(5,16);
            write('Route B3     ',DominB3P:5:3,'             ',
                  DominB3A:5:3);
      end; { Ready }
end; { ShowDomin }

procedure ShowPrimary;
begin { ShowPrimary }
      ClRange(4,25);
      border(1,4,79,25);
      gotoxy(3,6);
      write('Particle Refractive Index : ',RIpReal:5:3,' + ',
            RIpImag:5:3,' i');
      gotoxy(3,8);
      write('Size Parameter  : Maximum -- ',MaxApha:7:4,',  Minimum -- ',
            MinApha:7:4,',   Intervals -- ',Intervals:2);
      gotoxy(3,10);
      write('Coordinates     : Light Source -- (',Xs:7:3,',',Ys:7:3,',',
            Zs:7:3,')');
      gotoxy(21,11);
      write('Particle     -- (',Xp:7:3,',',Yp:7:3,',',Zp:7:3,')');
```

```
      gotoxy(21,12);
      write('Detector     -- (',Xd:7:3,',',Yd:7:3,',',Zd:7:3,')');
      gotoxy(3,14);
      write('Incident Light  : Wave Length  -- ',WL:7:4,' um');
      gotoxy(21,15);write('Polarization --  ');
      case Light of
           Combined   : write('Combined');
           Horizontal : write('Horizontal');
           Vertical   : write('Vertical');
      end; { case }
      gotoxy(3,17);
      write('Coating Layer   : Refractive Index --  ',RIsReal:5:3);
      gotoxy(21,18);
      write('Thickness        -- ',Dcs:6:3,' um');
      gotoxy(21,19);
      write('Reflectance      --  ',Refle2:4:2);
      gotoxy(3,21);
      write('Bottom Surface  : Reflectance      --  ',Refle1:4:2);
end; { ShowPrimary }

procedure ShowSecondary;
begin { ShowSecondary }
      ClRange(4,25);
      border(1,4,79,25);
      if not MieSecOk then
      begin { Secondary Variables not yet defined }
            gotoxy(5,25);
            noise(1);
            write(' Secondary variables have not been defined yet ');
      end { Not defined yet }
      else
      begin { Ready }
            gotoxy(3,6);
            write('Route A  :  Scattering Angle -- ',ThetaA:6:2);
            gotoxy(3,8);
            write('Route B1 :  Scattering Angle -- ',ThetaB1:6:2);
            gotoxy(15,9);
            write('Shadow Angle     -- ',ShadowAngleB1:6:2);
            gotoxy(3,11);
            write('Route B2 :  Scattering Angle -- ',ThetaB2:6:2);
            gotoxy(15,12);
            write('Shadow Angle     -- ',ShadowAngleB2:6:2);
            gotoxy(3,14);
            write('Route B3 :  Scattering Angle -- ',ThetaB3:6:2);
            gotoxy(15,15);
            write('Shadow Angle  1  -- ',ShadowAngleB1:6:2);
            gotoxy(15,16);
            write('Shadow Angle  2  -- ',ShadowAngleB2:6:2);
      end; { Ready }
end; { ShowSecondary }

procedure ShowShadow;

label 1,2;
```

```
var
   step,nn:integer;
   YN:char;

begin { ShowShadow }
      if not MieCalOk then
      begin { Shadow Values not yet defined }
            gotoxy(5,25);
            noise(1);
            write(' Shadow values have not been calculated yet ');
      end { Not calculated yet }
      else
      begin { Ready }
      step:=0;
   1: nn:=0;
      ClRange(4,25);
      border(1,4,79,25);
      gotoxy(3,5);
      write('Size(um)    SV-B11    SV-B12    SV-B21    SV-B22');
      gotoxy(3,6);
      write('--------    ------    ------    ------    ------');
      for step:=step+1 to intervals do
      begin { for }
           nn:=nn+1;
           gotoxy(4,nn+6);
           write(Size[step]*WL/pi:5:2,'      ',
                 ShadowValueB11[step]:6:2,'    ',
                 ShadowValueB12[step]:6:2,'    ',
                 ShadowValueB21[step]:6:2,'    ',
                 ShadowValueB22[step]:6:2);
           if nn=17 then begin
                         MessLine('More data (Y/N)?  ');
                         read(kbd,YN);
                         write(upcase(YN));
                         if upcase(YN)='N' then goto 2
                                           else goto 1;
                      end; { if }
      end; { for }
      end; { Ready }
    2:;
end; { ShowShadow }

begin { ShowStatus }
      exit:=false;
      ActCom:=default[6];
      PreCom:=ComList[6,ActCom].P;
      while not exit do
      case SelCom(6) of
           'D' : ShowDomin;
           'P' : ShowPrimary;
           'S' : ShowSecondary;
           'V' : ShowShadow;
           'E' : exit:=true;
```

```
            else noise(1);
        end; { case }
        ActCom:=8;
        ClRange(4,25);
end; { ShowStatus }
```

```
( File Name : TRANSFER.INC )

procedure Transfer;
type
    DataRecType = record
                      DataValue : real;
                  end;
var
   DataFile : file of DataRecType;
   DataRec  : DataRecType;
   I,J      : integer;
   DriveName: string[1]; ( A )
   FileName : string[8]; ( DATAFILE )
   FN       : FNtype; ( A:DATAFILE.MIE )
   exit     : boolean;

procedure TransferLoad;
label 1;
begin ( TransferLoad )
      ClrLine(3);
      gotoxy(1,3);write('Drive name : _');
      gotoxy(14,3);read(DriveName);
      gotoxy(25,3);write('file name : ________.MIE');
      gotoxy(37,3);read(FileName);
      FN:=DriveName+':'+FileName+'.MIE';
      if not Exist(FN) then
         begin ( file not existed )
            finished(' File not existed ');
            goto 1;
         end; ( file existed )
      Assign(DataFile,FN);Reset(DataFile);
      for I:=1 to 34 do ( Load all Primary & Secondary variables )
      begin ( for )
            read(DataFile,DataRec);
            with DataRec do
                 case I of
                      1:RIpReal:=DataValue;
                      2:RIpImag:=DataValue;
                      3:MaxApha:=DataValue;
                      4:MinApha:=DataValue;
                      5:Intervals:=trunc(DataValue); ( real -> integer )
                      6:Xs:=DataValue;
                      7:Ys:=DataValue;
                      8:Zs:=DataValue;
                      9:Xp:=DataValue;
                     10:Yp:=DataValue;
                     11:Zp:=DataValue;
                     12:Xd:=DataValue;
                     13:Yd:=DataValue;
                     14:Zd:=DataValue;
                     15:WL:=DataValue;
                     16:RIsReal:=DataValue;
                     17:Dcs:=DataValue;
                     18:Refle1:=DataValue;
```

```
                    19:ThetaA:=DataValue;
                    20:ThetaB1:=DataValue;
                    21:ThetaB2:=DataValue;
                    22:ThetaB3:=DataValue;
                    23:ShadowAngleB1:=DataValue;
                    24:ShadowAngleB2:=DataValue;
                    25:case trunc(DataValue) of
                            0 : Light:=Combined;
                            1 : Light:=Horizontal;
                            2 : Light:=Vertical;
                       end; { case }
                    26:Refle2:=DataValue;
                    27:DominAP:=DataValue;
                    28:DominAA:=DataValue;
                    29:DominB1P:=DataValue;
                    30:DominB1A:=DataValue;
                    31:DominB2P:=DataValue;
                    32:DominB2A:=DataValue;
                    33:DominB3P:=DataValue;
                    34:DominB3A:=DataValue;
                end; { Case }
end; { for }
for J:=0 to 12 do
for I:=1 to Intervals do
begin { for }
      read(DataFile,DataRec);
      case J of
           0:Size[I]:=DataRec.DataValue; { Sizes }
           1:AIntensity1[I]:=DataRec.DataValue;
             { Route A - i(1) }
           2:AIntensity2[I]:=DataRec.DataValue;
             { Route A - i(2) }
           3:B1Intensity1[I]:=DataRec.DataValue;
             { Route B1 - i(1) }
           4:B1Intensity2[I]:=DataRec.DataValue;
             { Route B1 - i(2) }
           5:B2Intensity1[I]:=DataRec.DataValue;
             { Route B2 - i(1) }
           6:B2Intensity2[I]:=DataRec.DataValue;
             { Route B2 - i(2) }
           7:B3Intensity1[I]:=DataRec.DataValue;
             { Route B3 - i(1) }
           8:B3Intensity2[I]:=DataRec.DataValue;
             { Route B3 - i(2) }
           9:ShadowValueB11[I]:=DataRec.DataValue;
             { B11 Shadow Value }
          10:ShadowValueB12[I]:=DataRec.DataValue;
             { B12 Shadow Value }
          11:ShadowValueB21[I]:=DataRec.DataValue;
             { B21 Shadow Value }
          12:ShadowValueB22[I]:=DataRec.DataValue;
             { B22 Shadow Value }
      end; { case }
end; { for }
```

```
      close(DataFile);
      finished(' Load completed ');
      MieSecOK:=true;
      MieCalOK:=true;
    1:ActCom:=3;
end; { TransferLoad }

procedure TransferSave;
label 1;
var
   R : char;
begin { TransferSave }
      if Auto then FN:=AutoFN
      else begin { Not Auto }
      ClrLine(3);
      gotoxy(1,3);write('Drive name : _');
      gotoxy(14,3);read(DriveName);
      gotoxy(25,3);write('file name : ________.MIE');
      gotoxy(37,3);read(FileName);
      FN:=DriveName+':'+FileName+'.MIE';
      if Exist(FN) then
         begin { file existed }
            finished(' File existed ');
            gotoxy(62,3);
            write('Replace (Y/N) : _');
            gotoxy(78,3);read(kbd,R);gotoxy(78,3);write(UpCase(R));
            if UpCase(R)<>'Y' then
            begin { Not Replace }
                  gotoxy(45,1);clreol;
                  finished(' Save cancelled ');
                  goto 1;
            end; { Not Replace }
         end; { file existed }
      end; { not Auto }
      Assign(DataFile,FN);Rewrite(DataFile);
      for I:=1 to 34 do { Save all Primary & Secondary variables }
      begin { for }
            with DataRec do
                 case I of
                      1:DataValue:=RIpReal;
                      2:DataValue:=RIpImag;
                      3:DataValue:=MaxApha;
                      4:DataValue:=MinApha;
                      5:DataValue:=Intervals; { real <- integer }
                      6:DataValue:=Xs;
                      7:DataValue:=Ys;
                      8:DataValue:=Zs;
                      9:DataValue:=Xp;
                     10:DataValue:=Yp;
                     11:DataValue:=Zp;
                     12:DataValue:=Xd;
                     13:DataValue:=Yd;
                     14:DataValue:=Zd;
                     15:DataValue:=WL;
```

```
                16:DataValue:=RIsReal;
                17:DataValue:=Dcs;
                18:Refle1;
                19:DataValue:=ThetaA;
                20:DataValue:=ThetaB1;
                21:DataValue:=ThetaB2;
                22:DataValue:=ThetaB3;
                23:DataValue:=ShadowAngleB1;
                24:DataValue:=ShadowAngleB2;
                25:case Light of
                        Combined   : DataValue:=0;
                        Horizontal : DataValue:=1;
                        Vertical   : DataValue:=2;
                   end; { case }
                26:DataValue:=Refle2;
                27:DataValue:=DominAP;
                28:DataValue:=DominAA;
                29:DataValue:=DominB1P;
                30:DataValue:=DominB1A;
                31:DataValue:=DominB2P;
                32:DataValue:=DominB2A;
                33:DataValue:=DominB3P;
                34:DataValue:=DominB3A;
            end; { Case }
        write(DataFile,DataRec);
end; { for }
for J:=0 to 12 do
for I:=1 to Intervals do
begin { for }
        case J of
            0:DataRec.DataValue:=Size[I]; { Sizes }
            1:DataRec.DataValue:=AIntensity1[I];
              { Route A - i(1) }
            2:DataRec.DataValue:=AIntensity2[I];
              { Route A - i(2) }
            3:DataRec.DataValue:=B1Intensity1[I];
              { Route B1 - i(1) }
            4:DataRec.DataValue:=B1Intensity2[I];
              { Route B1 - i(2) }
            5:DataRec.DataValue:=B2Intensity1[I];
              { Route B2 - i(1) }
            6:DataRec.DataValue:=B2Intensity2[I];
              { Route B2 - i(2) }
            7:DataRec.DataValue:=B3Intensity1[I];
              { Route B3 - i(1) }
            8:DataRec.DataValue:=B3Intensity2[I];
              { Route B3 - i(2) }
            9:DataRec.DataValue:=ShadowValueB11[I];
              { B11 Shadow value }
           10:DataRec.DataValue:=ShadowValueB12[I];
              { B12 Shadow value }
           11:DataRec.DataValue:=ShadowValueB21[I];
              { B21 Shadow value }
           12:DataRec.DataValue:=ShadowValueB22[I];
```

```
                    { B22 Shadow value }
              end; { case }
              write(DataFile,DataRec);
        end; { for }
        close(DataFile);
        finished(' Save completed ');
      1:ActCom:=3;
end; { TransferSave }

begin { Transfer }
        if not Auto then
        begin
              exit:=false;
              ActCom:=default[7];
              PreCom:=ComList[7,ActCom].P;
              while not exit do
              case SelCom(7) of
                    'L' : TransferLoad;
                    'S' : TransferSave;
                    'E' : exit:=true;
                    else noise(1);
              end; { case }
              ClRange(4,25);
              ActCom:=9;
        end
        else TransferSave;
end; { Transfer }
```

```
{ File Name : AUTO.INC }

procedure AUTOMA;

var
   a,b,c,d,e,f,g,h,i:real;
   idx,nn : integer;
   FN     : FNType;
   u,v    : ValueType;
            { u for central particle, v for edge particle }
   DF     : Text;
   id     : integer;

begin { AUTOMA }
      auto:=true;
      a:=Xs; b:=Ys; c:=Zs; d:=Xp; e:=Yp; f:=Zp; g:=Xd; h:=Yd; i:=Zd;
      ClrLine(1);
      gotoxy(1,1);write('COMMAND : AUTO ');
      for id:=5 to 6 do
      begin { id }
      case id of
           0:Dcs:=0.0;
           1:Dcs:=0.5;
           2:Dcs:=1.0;
           3:Dcs:=2.0;
           4:Dcs:=3.0;
           5:Dcs:=4.0;
           6:Dcs:=5.0;
      end;
      gotoxy(5,7);write('Coating Thickness = ',Dcs:3:1,' um');
      writeln(LST,'Coating Thickness = ',Dcs:3:1,' um');
      writeln(LST);
      writeln(LST,'Dominance Index by Area');
      writeln(LST,'Incident      B1         A  ');
      writeln(LST,'--------    ------    ------');
{ Range:10-30 degree }
      for idx:=1 to 11 do
      begin
           case idx of
                1 : Xs:=-0.8816;
                2 : Xs:=-1.0628;
                3 : Xs:=-1.2466;
                4 : Xs:=-1.4337;
                5 : Xs:=-1.6246;
                6 : Xs:=-1.8199;
                7 : Xs:=-2.0201;
                8 : Xs:=-2.2261;
                9 : Xs:=-2.4387;
                10: Xs:=-2.6585;
                11: Xs:=-2.8868;
           end;
{ Range:0-85 degree }
{
      for idx:=1 to 23 do
```

```
        begin
              case idx of
                    1 : Xs:=0;
                    2 : Xs:=-0.0873;
                    3 : Xs:=-0.1746;
                    4 : Xs:=-0.2620;
                    5 : Xs:=-0.3496;
                    6 : Xs:=-0.4374;
                    7 : Xs:=-0.8816;
                    8 : Xs:=-1.3397;
                    9 : Xs:=-1.8199;
                   10: Xs:=-2.3315;
                   11: Xs:=-2.8868;
                   12: Xs:=-3.5010;
                   13: Xs:=-4.1955;
                   14: Xs:=-5;
                   15: Xs:=-5.9588;
                   16: Xs:=-7.1407;
                   17: Xs:=-8.3214;
                   18: Xs:=-9.0202;
                   19: Xs:=-10.7225;
                   20: Xs:=-13.737;
                   21: Xs:=-18.66;
                   22: Xs:=-28.356;
                   23: Xs:=-57.15;
              end;
)
              EditData;
              gotoxy(5,9);write('Incident Angle = ',ThetaB1:6:2);
              gotoxy(16,15);
              write('* *  Program is RUNNING.  Do not interrupt !  * *');
              Mie;
              writeln(LST,'  ',ThetaB1:6:2,'    ',DominB1A:6:3,'    ',
                      DominAA:6:3);
        end; { for idx }
        writeln(LST);
        writeln(LST);
        end; { for id }
        auto:=false;
        Xs:=a; Ys:=b; Zs:=c; Xp:=d; Yp:=e; Zp:=f; Xd:=g; Yd:=h; Zd:=i;
        ActCom:=1;
        delay(2000);
        ClRange(4,25);
end; { AUTOMA }
```