RC 18830 (82351) 4/15/93
Manufacturing 57 pages

Research Report

Determination of Dimensional C4 Solder Ball Parameters

K.B. Kirtley

IBM Research Division
T. J. Watson Research Center
Yorktown Heights, NY 10598

NOTICE

This report will be distributed outside of IBM up to one year after the IBM publication date.

IBM Research Division
Almaden • T.J. Watson • Tokyo • Zurich

Determination of Dimensional C4 Solder Ball Parameters

K. B. Kirtley

IBM Research
T. J. Watson Research Center
Yorktown Heights, NY 10598

Abstract: This research report describes a set of mathematical equations and a C program which may be used to determine whether or not C4 solder pads with a given height, diameter, and pad spacing can be inspected using the Oblique Viewing Microscope (OVM). The OVM is the optical front end for both the Pad Analysis System (PAS) and the Individual Chip Inspection System (ICIS). PAS and ICIS are used to detect low volume C4 solder pads on wafers and diced chips before the chips are joined to the substrate.

Introduction

Controlled Collapse Chip Connection (C4)[1,2] is an interconnect technology used to attach semiconductor chips to substrates. Developed during the 1960s by IBM, this process, which is also known as flip chip technology, is an alternative to wire bonding.

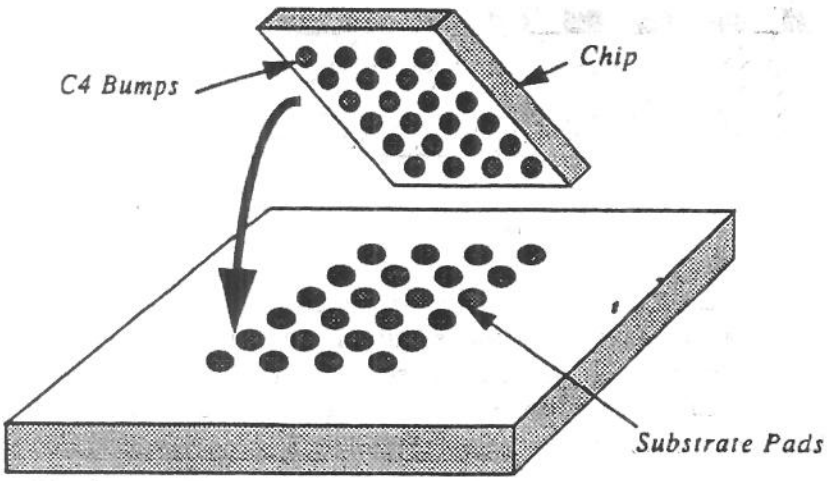

Figure 1. C4 technology

C4 technology allows one to place many more solder pads on a chip than can be placed with wire bonding. With C4, the entire surface of the chip can be used for the interconnect solder pads while with wire bonding, these interconnects are placed only around the perimeter of the chip.

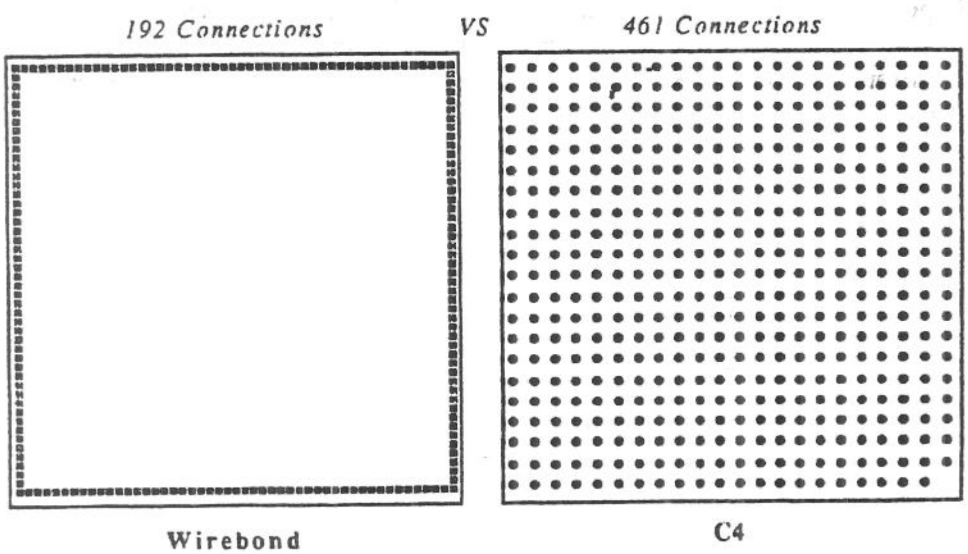

Figure 2. Interconnect density of wire bonding vs C4

The small lead-tin solder balls, which are also referred to as C4s, solder pads, and solder bumps, are deposited in an array on the top surface of the chip above the vias. To form a module, the chips are inverted and placed upside down on the substrate. The entire populated module is then passed through a furnace where the C4s melt and resolidify.

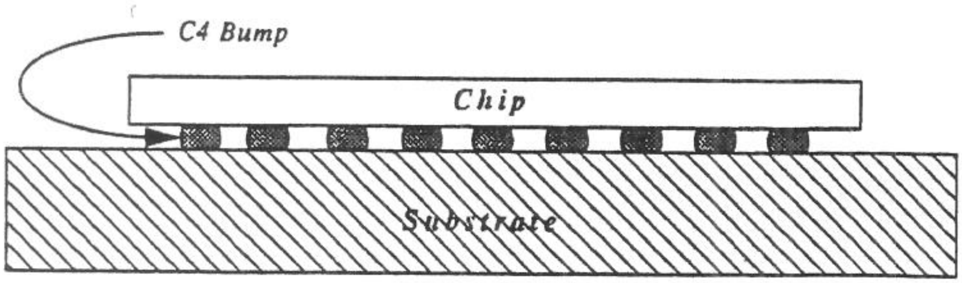

Figure 3. Chip joined to substrate with C4 solder pads

C4s form both the mechanical and electrical join between the chip and substrate. They must be of uniform size as a pad that is too small can lead to a failure either when the module is tested or after the machine is installed in the field. Since it is both costly and time consuming to locate and remove chips containing small C4s, there is strong motivation for detecting small pads before the chips are attached to the substrate.

In order to guarantee an acceptable chip quality level for the module build process, the chips are inspected for partial pad volume following electrical test and second reflow. The present standard is to reject any pad which has a volume less than 2/3 of the average volume of the adjacent pads[3]. This is also referred to as the "1/3 missing" criterion. Studies have shown that a percentage of missing volume is a better predictor of failure to join than either reduced pad height or reduced pad diameter.

Partial pad inspection was originally done by operators using stereoscopic microscopes and side lighting. In 1986 IBM replaced this tedious and error prone manual inspection with an internally developed automated machine vision system, the Pad Analysis System (PAS)[4-9].

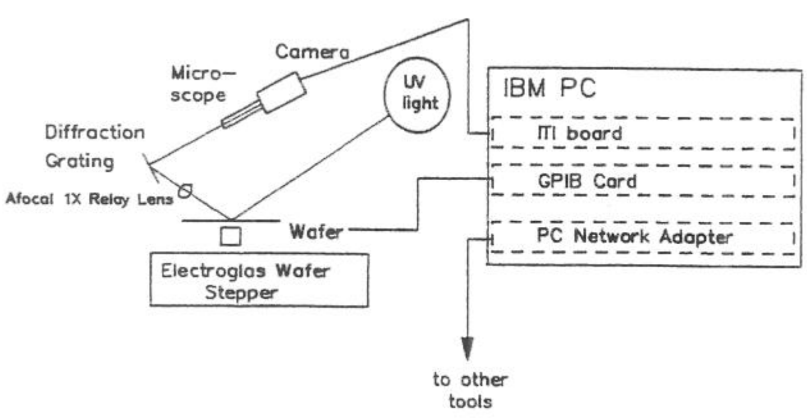

Figure 4. System diagram of the Pad Analysis System

This tool automatically loads and aligns each wafer under the optical front end, inspects all electrically good chip sites for partial pads, and then shares the results with other tools in the area. A similar tool, the Individual Chip Inspection System (ICIS), was released in 1991 to handle and inspect individual diced chips. Both tools are based on the Oblique Viewing Microscope (OVM)[10-15] which uses oblique illumination and observation to produce two dimensional images containing three dimensional volume information. Much as one can determine the height of a tree by measuring the tree's shadow and the angle of the sun, the height and diameter of the C4 pad can be found from OVM images which are digitized and then analyzed using internally developed image analysis algorithms developed specifically for this application[16-19].

This research report is a reference manual for engineers and scientists who must determine whether or not chips with a given C4 pad size and pad spacing can be inspected using the PAS/ICIS tools. Included in this report are a set of mathematical equations and a C program which may be used to determine the parameters for both the truncated sphere and the two-dimensional pad image produced by the Oblique Viewing Microscope. The Oblique Viewing Microscope, the Pad Analysis System, the Individual Chip Inspection System, and the associated algorithms have all been described elsewhere and the reader is referred to those references for further details.

C4 Solder Ball Parameters

Prior to the development of the Oblique Viewing Microscope, operators used stereoscopic microscopes to inspect chips for partial pads. Figure 5 shows a photograph of a bipolar chip which was taken using a standard optical microscope in which the illumination is normal to the surface of the chip. This image is similar to those seen by operators during manual inspection. The dark circular objects are the C4 pads and the brighter background is the surface of the chip. The images of the C4 pads are dark because most of the light hitting the pads is scattered off the curved surfaces. The bright spot at the center of many of these pad images is caused by light reflecting normal to the apex of the pad. Only the diameters of the pads can be easily observed and pancake shaped pads may not be detected. Operators can obtain additional information about actual pad volume through the use of side lighting and other visual cues such as pad glints and pad reflectivity.

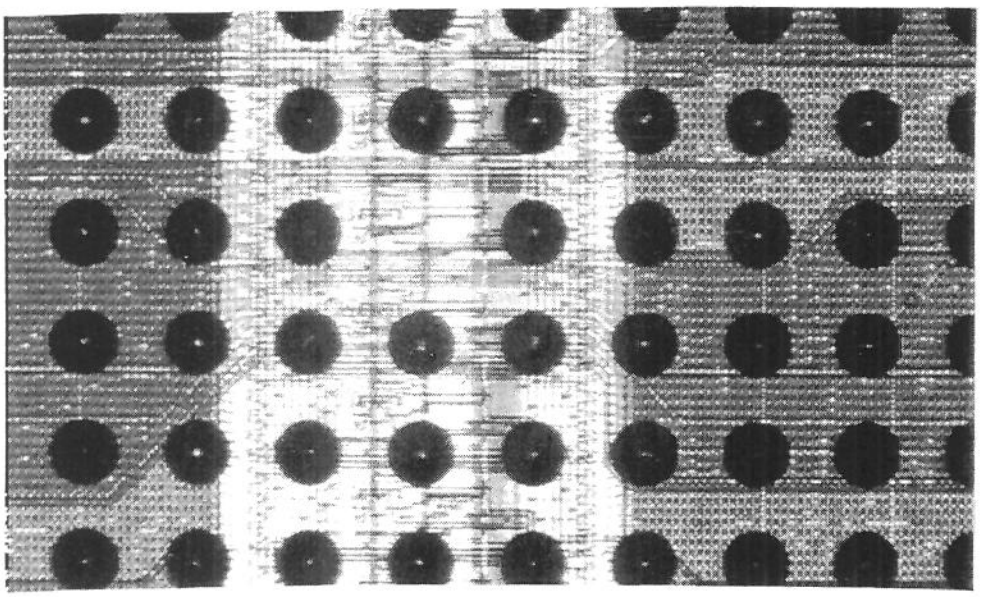

Figure 5. Bipolar chip imaged with bright field illumination

The invention of the Oblique Viewing Microscope provided a method by which volume information could be inferred from a two-dimensional image. Figure 6 shows a typical OVM image of a bipolar chip. The C4 pads are seen as dark oval shaped objects and the chip surface is the brighter background. To differentiate between the actual C4 pads, which are essentially truncated spheres, and the OVM images of the C4 pads, the latter will be referred to as "OVM C4 pad images" or just "pad images". In OVM images we see that features such as the chip surface, which are essentially two-dimensional in nature, are unchanged with the oblique illumination and oblique observation. However, we observe that images of 3-dimension features such as the pads are transformed in a way that is somewhat unexpected. Truncated spheres, which with bright field illumination appear as circular pad images, are reimaged by the OVM as a pair of overlapping ellipses. From formulas which appear later in the text, the

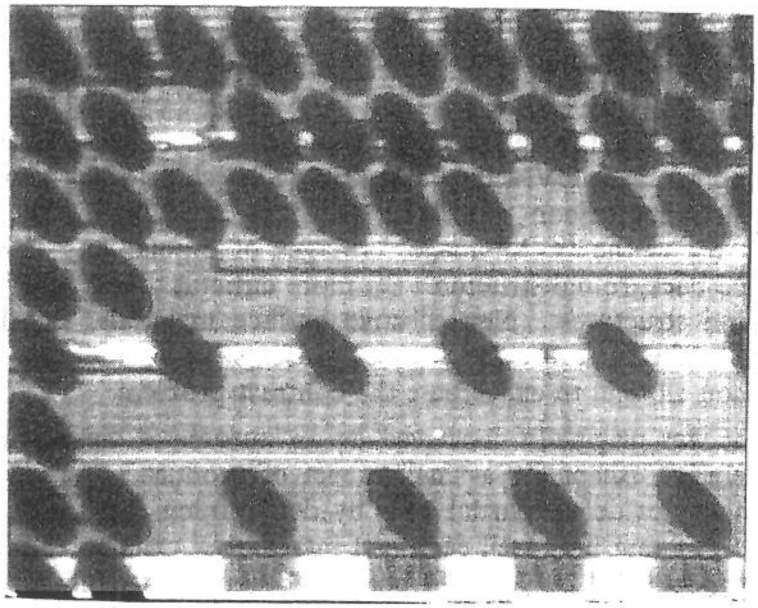

Figure 6. Bipolar chip imaged with the Oblique Viewing Microscope

height and diameter for each pad can in principle be calculated from the length and width of the elliptical image.

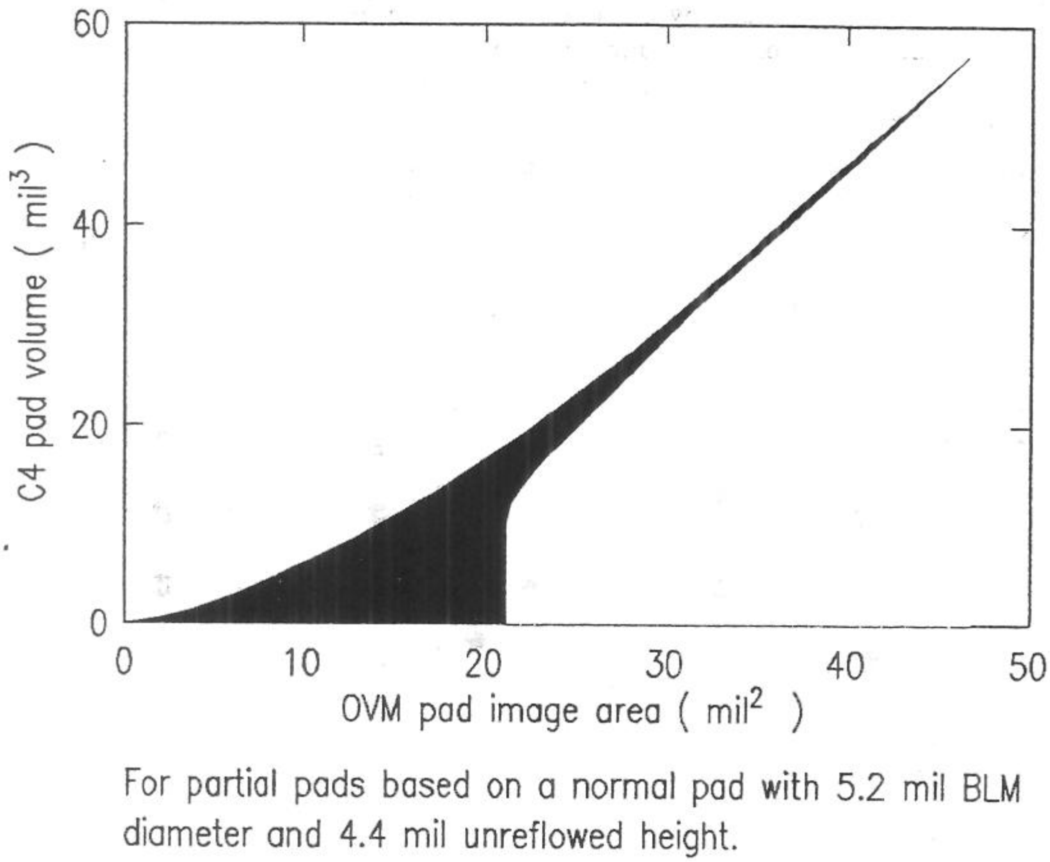

Figure 7. Relationship between pad volume and OVM pad image area

In order to properly analyze an OVM image for small pads, it is necessary to separate or segment the part of the image containing the pad from that of the background. This works best when the chip image has both good grey level contrast between the pad images and the background and good spatial separation between the pad images.

Image contrast is a function of the surface of the chip. Pads evaporated onto bare silicon wafers produce an excellent OVM image as do fiducial chips which contain no underlying circuitry. All other silicon processing tends to produce effects that decrease the degree of contrast in the OVM image. Images of bipolar and CMOS chips which have complicated background structures are more difficult to segment than chips with simple background structures. Nonreflective surface materials, shrinking ground rules, increased levels of personalization, uneven wafer surface, and very fine parallel structures in the silicon all tend to exacerbate the contrast problem.

Although the PAS tool was originally conceived as an automatic measurement tool, this goal has proved elusive except for a small subset of product. The current image analysis algorithm[19] does not detect partial pads by determining the dimensions of the pad but rather by determining how well each pad image area matches the ideal pad image for that chip. As seen in Figure 7, this is possible because with only a small error, the area of the pad image is directly proportional to the volume. This graph is based on formulas for volume and image pad area which are derived later in the text. The last point on the graph shows the relationship between pad image area and pad volume for a pad with a 5.2 mil BLM diameter and a 4.4 mil unreflowed height. All other points on the graph show the relationship between pad image area and volume for pads with a BLM less than or equal to 5.2 mils and an unreflowed height less than or equal to 4.4 mils. We can see that by detecting a reduction in pad image area, it is possible to determine the corresponding reduction in pad volume. The graph is multi-valued because pads with different dimension can have the same volume but will not have the same pad image area.

Another important issue is the actual arrangement of the C4s on the chip. For optimal low volume detection with the PAS system, the pads should be positioned so that there is no overlap between two adjacent pad images. When pad images overlap, it is difficult to discriminate where one pad image stops and the adjacent pad image begins. Although not specifically designed to handle overlapping pads, the current

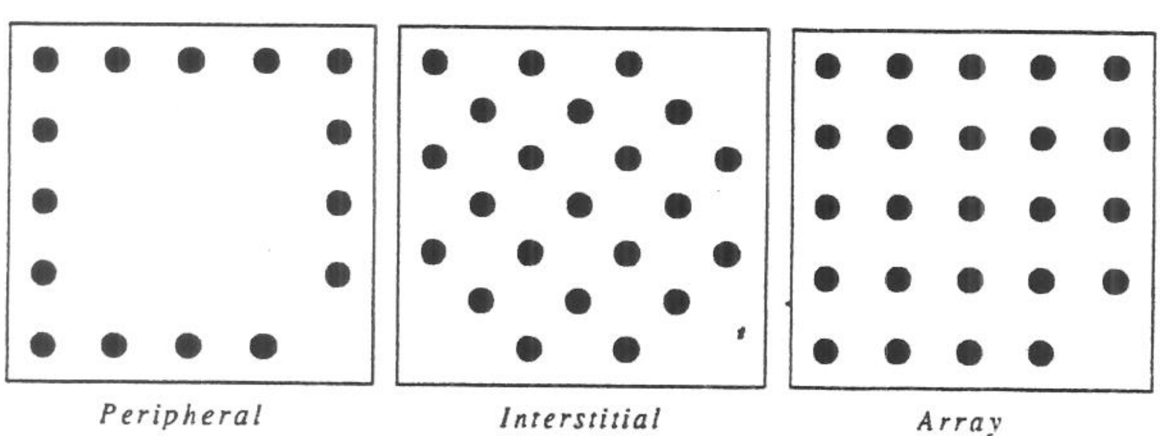

Figure 8. Sample C4 footprints

image analysis algorithm is still able to detect small pads when the amount of overlap is small. In addition, for chips which will be inspected with the ICIS tool, the pad must not be placed so close to the edge of the chip that the pad images extend off the edge of the chip[24].

The arrangement of the C4 solder pads on the surface of a chip is known as the C4 pad footprint. Several types of footprints already in use are shown in Figure 8. Prior to the use of the PAS/ICIS tools in manufacturing, minimum pad center to pad center spacings, also referred to as pitch, were established. Table 1 shows several pad dimensions and the associated minimum pitch.

Table 1. C4 Diameter and Pitch (in microns)

Nominal C4 Diameter	Minimum Pitch
150	300
125	250
100	225

These spacings were established for reasons other than the ability to inspect product with PAS/ICIS and they can be too small for certain configurations of pads. The critical dimension is the pad center to pad center spacing distance along a 45 degree diagonal as this is the orientation of the pad images when illuminated across the corner of the chip. For pads on a square grid array, the established minimum pitches allow sufficient space for the diagonal pad images but for interstitial footprints, the distance can be too small.

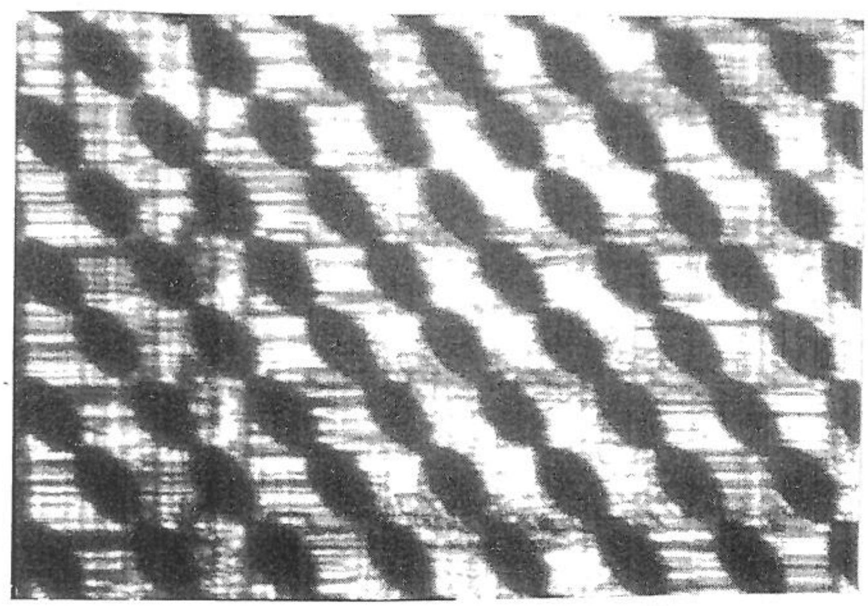

Figure 9. Product with overlapping pad images

There are several current footprints used for CMOS product which cause pad image overlap when the chip is imaged by the Oblique Viewing Microscope. For optimal PAS/ICIS performance, such footprints should be avoided. Improved performance would be expected if the inspection algorithm were redesigned to detect deviations in pad image overlap rather than deviations in pad image size.

In order to understand whether or not the images of the pads overlap, we must examine the size and shape of the unreflowed ball, the reflowed ball, and the OVM pad image. By knowing the size of either an unreflowed or reflowed C4 pad, we can calculate the size of the OVM image and determine whether or not the pad images overlap.

Currently C4 solder balls are created by evaporating a lead-tin mixture through a moly mask onto the surface of the wafer. Following terminal metals evaporation but before reflow, the pad has the shape of a truncated cone. If the following variables are defined for the unreflowed solder ball :

d_b = BLM diameter,
d_t = top diameter of the unreflowed solder ball,
h_u = height of unreflowed solder ball,
ϕ = internal base angle of pad, and
V = volume of the solder ball, then

$$V = \frac{\pi h_u \left(d_b^2 + d_b d_t + d_t^2\right)}{12}. \quad (1)$$

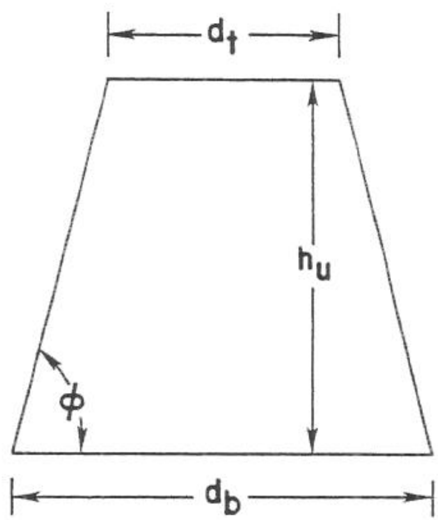

Figure 10. C4 solder ball post deposition but prior to reflow

The range of values for d_b and h_u are both called out in the Terminal Metals Specification[22]. d_t can be calculated from one of two formulas :

$$d_t = d_b(1 - .075h_u) \quad (2)$$

where h_u is measured in mils or

$$d_t = d_b - \frac{2h_u}{\tan\phi} \tag{3}$$

where ϕ is between 75 and 80 degrees.

Following electrical test and reflow, the solder volume will be the same as for the unreflowed ball, but now the pads are shaped like truncated spheres. If the following variables are defined for the reflowed ball :

d_e = equatorial diameter of reflowed solder ball,
d_v = viewed diameter of reflowed solder ball as seen by the OVM, and
h_r = height of reflowed solder ball, then

$$V = \pi h_r \left(\frac{d_e h_r}{2} - \frac{h_r^2}{3} \right). \tag{4}$$

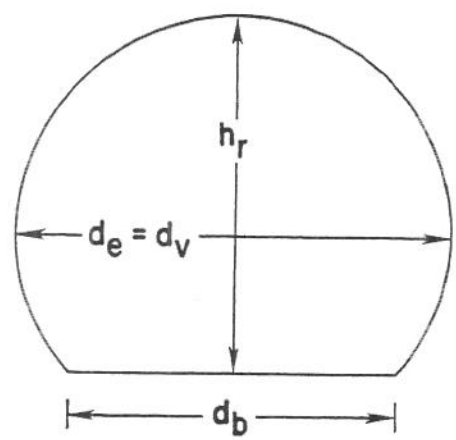

Figure 11. C4 solder ball after reflow

When the solder ball is a hemisphere or greater, the OVM viewed diameter, d_v, is the same as the equatorial diameter, d_e, and one may use the viewed diameter and Eq. (4) to calculate the volume. But if the truncated sphere is less than a hemisphere, then the OVM viewed diameter, d_v, is not equal to the equatorial diameter, d_e, but rather is equal to the BLM diameter, d_b. Although Eq. (4) is always true, it is useful to recalculate the volume in terms of the BLM diameter, d_b. From the Pythagorean theorem,

$$d_e = \frac{d_b^2}{4h_r} + h_r. \tag{5}$$

Substituting this relationship into Eq. (4) gives the volume in terms of the BLM diameter and the reflowed height :

$$V = \pi h_r \left(\frac{d_b^2}{8} + \frac{h_r^2}{6} \right). \tag{6}$$

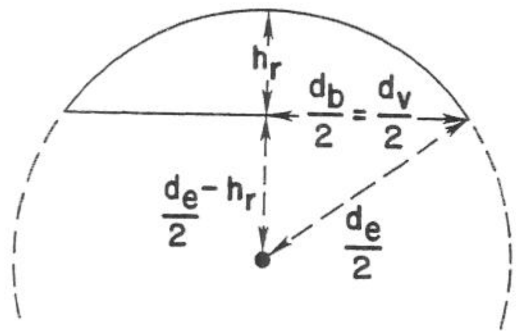

Figure 12. C4 solder ball after reflow showing relationship between d_b, d_e, and h_r.

Again, this relationship is always true, but for pads larger than a hemisphere, it is impossible to directly observe the BLM diameter, d_b.

Normally when a chip is placed under the Oblique Viewing Microscope, the solder balls produce an image in which each solder ball is seen as a pair of overlapping ellipses oriented at an angle of 45 degrees. One ellipse is the oblique observation of the back (unilluminated) side of the pad and the second is the shadow cast by that pad. The C4 pad images lie at 45 degrees because the light source is directed obliquely across the corner of the chip. This was done to allow, for a square grid of pads, the maximum space possible before overlap of pad images. The length of the pad image must be less than $\sqrt{2}$ times the pad spacing to avoid overlap.

Although we usually see the pads imaged as a pair of overlapping ellipses, the pad image actually depends of the degree of truncation of the sphere and can also be either a circle or a single ellipse. We will examine each case using the following definitions :

w = pad image width,
l = pad image length,
A = pad image area, and
θ = OVM illumination angle with respect to the normal.

Pads that are larger than a hemisphere produce a pair of ellipses. In this case the shadow width is equal to the equatorial diameter which is also equal to the viewed diameter :

$$w = d_e = d_v.$$

The shadow length and shadow area are both functions of the OVM illumination angle, the pad diameter, and the reflowed height :

$$l = \frac{d_e}{\cos\theta} + 2\tan\theta \left(h_r - \frac{d_e}{2} \right), \tag{7}$$

and

$$A = \frac{\pi\left(\frac{d_e}{2}\right)^2}{\cos\theta} + 2\left(h_r - \frac{d_e}{2}\right)\tan\theta\left[\frac{d_e^2}{4} - \left(h_r - \frac{d_e}{2}\right)^2 \sin^2\theta\right]^{1/2} +$$

$$\frac{d_e^2}{2\cos\theta} Arcsin\left[\frac{2\left(h_r - \frac{d_e}{2}\right)\sin\theta}{d_e}\right]. \tag{8}$$

When working with reflowed pad dimensions, a pad greater than a hemisphere is defined by a reflowed height to viewed diameter ratio of

$$.5 < \frac{h_r}{d_v} \leq 1.0$$

and when working with the pad image parameters width and length, a pad greater than a hemisphere is one defined by a pad image width to pad image length ratio of

$$\frac{1}{\cos\theta} < \frac{l}{w} \leq \frac{1+\sin\theta}{\cos\theta}.$$

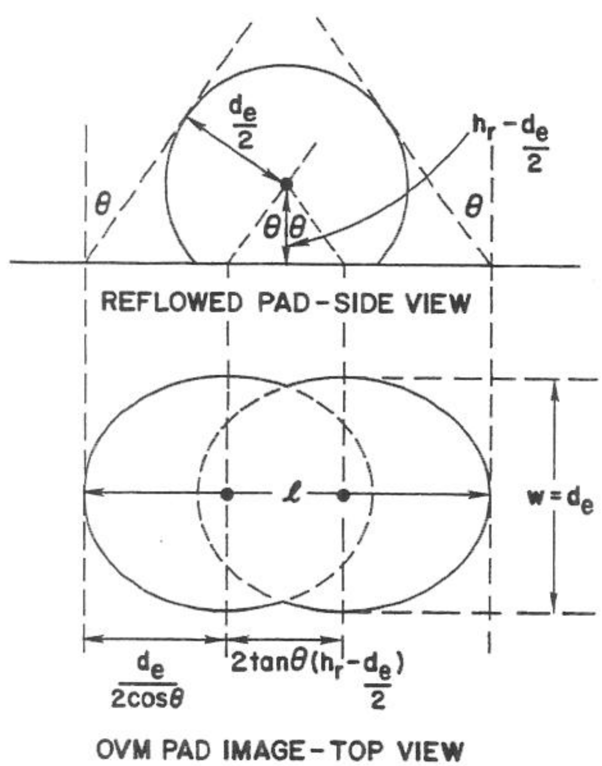

Figure 13. Pad and pad image for solder ball greater than hemisphere

Pads that are less than or equal to a hemisphere produce a single ellipse pad image. The exception to this is pad which are so flat that no shadow is cast. This situation is called the limiting value and is dealt with in the following section. For pads greater than this limiting value but less than a hemisphere

$$w = d_b = d_v,$$

$$l = \frac{d_e}{\cos\theta} + 2\tan\theta\left(h_r - \frac{d_e}{2}\right), \tag{7}$$

and

$$A = \frac{\pi w l}{4.}$$

Since it is impossible to measure the equatorial diameter, d_e, directly, it can be calculated from Eq. (5).

Again, when working with reflowed pad dimensions, a pad less than or equal to a hemisphere but greater than the limiting value is defined by a reflowed height to viewed diameter ratio of

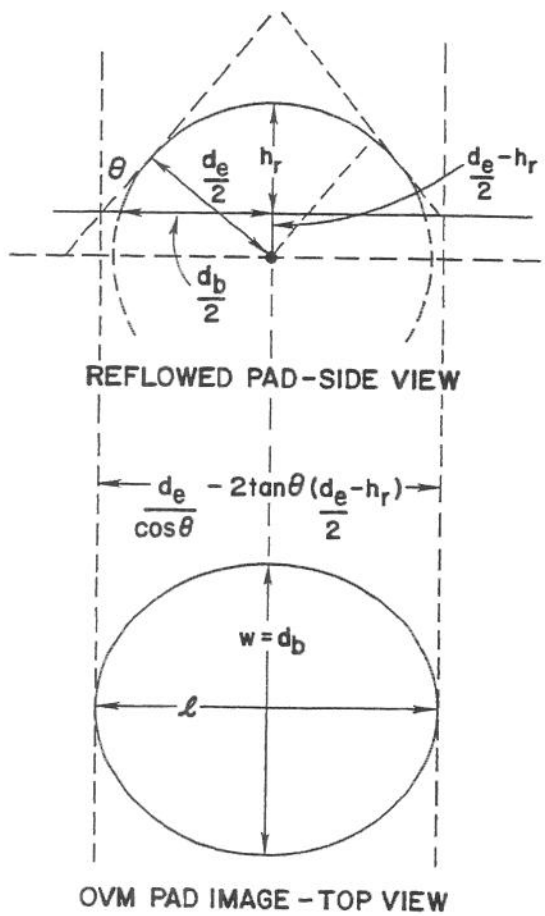

Figure 14. Pad and pad image for solder ball less than hemisphere

$$\frac{1 - \sin\theta}{2\cos\theta} < \frac{h_r}{d_v} \leq .5$$

and when working with the pad image width and length, a pad less than or equal to a hemisphere but greater than the pad limiting value is one defined by a pad image width to pad image length of

$$1 < \frac{l}{w} \leq \frac{1}{\cos\theta}.$$

Finally, when the pad image is a circle, the C4 pad itself is so small that no shadow is cast at all. In this case, the OVM image is the same as the conventional image. Here

$$l = w = d_b = d_v,$$

and

$$A = \frac{\pi d_b^2}{4.}$$

When working with reflowed pad dimensions, a pad less than or equal to the limiting case is defined by a reflowed height to viewed diameter ratio of

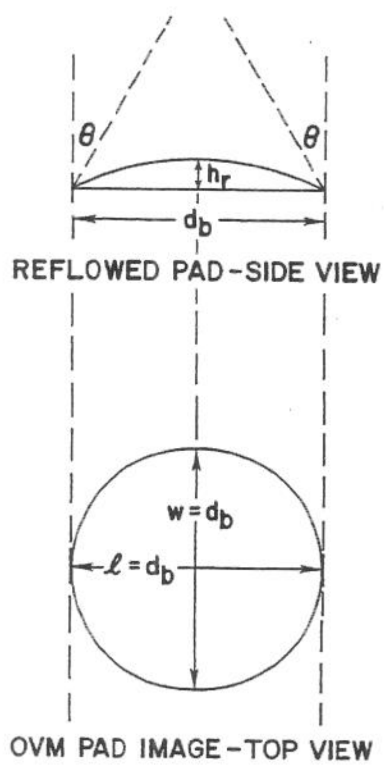

Figure 15. Pad and pad image for solder ball which does not cast a shadow

$$0 < \frac{h_r}{d_v} \leq \frac{1 - \sin\theta}{2cos\theta}.$$

and when working with the pad image width and length, a pad less than or equal to the limiting case is one defined by a pad image width to pad image length of

$$\frac{l}{w} = 1.$$

From these mathematical formulas, one can determine the relationship among the unreflowed pad, the reflowed pad, and the OVM image of the pad. The program that follows can be used to facilitate these calculations as well as determine whether or not the pads overlap.

Program : C4_PARAM

C4_PARAM is a program written in C which may be used to derive dimensional parameters associated with the unreflowed pad, reflowed pad, and OVM pad image. The program can also be used to determine whether or not the images of two solder balls located along a 45 degree line will overlap. Source code (Appendix B) has been included in this report so that the user may customize the program to his/her particular need. The program was compiled with Microsoft C version 5.1 and runs under DOS. Diskettes containing source and executable code are available from the author.

Running the program

1. Type "c4_param" to start program.
2. The program displays the title screen :

```
                    --- C 4 _ P A R A M ---

                          Version 1.0

          Program for calculating C4 solder ball parameters



                     Hit any key to continue



                          K. B. Kirtley
            IBM Research Division - Yorktown Heights, NY
                   (C) Copyright IBM Corp 1993
```

3. Hit any key to proceed to second screen.

4. The program displays the main menu screen.

```
              C 4 _ P A R A M   F U N C T I O N   M E N U

SYSTEM PARAMETERS

      d : display/change current system parameters

CALCULATE REFLOWED SOLDER BALL PARAMETERS FROM
THE FOLLOWING INITIAL CONDITIONS

      u : unreflowed pad - BLM diameter and unreflowed height

      b : reflowed pad - BLM diameter and reflowed height
      q : reflowed pad - equatorial diameter and reflowed height
      v : reflowed pad - viewed diameter and reflowed height

      i : reflowed pad - OVM pad image length and width

      e : EXIT program


Enter choice :
```

5. Enter 'd' to display the current system parameters.

```
                  S Y S T E M   P A R A M E T E R S

        o : OVM angle   = 51.66 degrees
        v : volume_mode = slope angle
        a : angle       = 75.00 degrees
        p : padspacing  = 7.87 mils = 200.00 microns
        f : file_mode   = 0 : will not write output to disk

        e : EXIT this screen


        Enter character to change system parameter
```

6. To change a system parameter, enter the character preceding the proper variable.
   a. o : OVM angle - All current PAS/ICIS systems have an OVM angle of 51.66 degrees to the normal. This value may be changed to theoretically test other possible angles. A decrease in the angle will produce a shorter pad shadow. Two other angles for which commercial components are available are 41 and 44 degrees.
   b. v : volume_mode - This mode determines which method is used to determined the angle ϕ as shown in Figure 10. There are two settings for this mode : ter-

minal metals spec or slope angle. The terminal metals mode uses the formula (Eq. 2) found in the terminal metals specification[22]. It is believed that this formula was derived empirically using Purdue (5.2 +/- .5 mil diameter) pads. Note that the formula works only if the unreflowed height is entered in mils. Slope angle mode allows the user to enter the angle ϕ directly. Measurements have shown ϕ to be between 75 and 80 degrees.

c. p : padspacing - This value is the distance between adjacent pads on a square grid. It is used to calculate the distance between pads along the diagonal. The default pad spacing value is set for 200 microns. Other known values include :

   Purdue, Olympic and ATx-1 = 250 microns
   ATx-4 = 225 microns

d. f : file_mode - The default file_mode is "will not write output to disk". To change this value, enter 'f' and the system will prompt for "0" for "will not write output to disk" or "1" for "write output to disk"

e. e : exit this screen - will return program to C4_PARAM FUNCTION MENU screen.

7. The options listed under CALCULATE REFLOWED SOLDER BALL PARAMETERS FROM THE FOLLOWING INITIAL CONDITIONS allow one to choose which initial condition will be requested.

   a. u : will prompt for a range of values for the BLM diameter and height of an unreflowed pad. The program will then calculate the volume, the reflowed parameters, and the OVM image parameters for each pair of input values.

   b. b : will prompt for a range of values for the BLM diameter and height of a reflowed pad. The program will then calculate the volume, the reflowed diameter, and the OVM image parameters for each pair of input values.

   c. q : will prompt for a range of values for the equatorial diameter and height of a reflowed pad. The program will then calculate the volume, the BLM diameter, and the OVM image parameters for each pair of input values.

   d. v : will prompt for a range of values for the viewed diameter and height of a reflowed pad. The program will then calculate the volume, the BLM diameter, the equatorial diameter, and the OVM image parameters for each pair of input values.

   e. i : will prompt for a range of values for the OVM pad image length and width. The program will then calculate the volume, the BLM diameter, the equatorial diameter, and the reflowed height for each pair of input values.

8. For example, after entering 'u', the following screen will appear :

```
I N P U T   D A T A   F O R   B L M   D I A M E T E R   A N D
        U N R E F L O W E D   H E I G H T

Requesting range of data for BLM diameter

Enter smallest BLM diameter (> 0) in mils                : 4.0
Enter largest BLM diameter (>= smallest) in mils         : 5.0
Enter BLM diameter increment (> 0) in mils               : 1.0


Requesting range of data for unreflowed height

Enter smallest unreflowed height (> 0) in mils           : 4.0
Enter largest unreflowed height (>= smallest) in mils    : 4.8
Enter unreflowed height increment (> 0) in mils          : 0.1
```

9. The results are displayed on the screen and also written to disk if file_mode was set on the system parameter screen.

urflw ht	reflw ht	BLM diamt	equat diamt	viewd diamt	shadw lngth	reflow volume	image area	max diamt	max lngth
4.00	2.75	4.00	4.20	4.20	8.41	28.14	29.18	5.57	11.14
4.10	2.76	4.00	4.21	4.21	8.45	28.40	29.38	5.57	11.14
4.20	2.78	4.00	4.22	4.22	8.49	28.65	29.56	5.57	11.14
4.30	2.79	4.00	4.22	4.22	8.52	28.88	29.74	5.57	11.14
4.40	2.80	4.00	4.23	4.23	8.55	29.10	29.90	5.57	11.14
4.50	2.81	4.00	4.23	4.23	8.58	29.31	30.06	5.57	11.14
4.60	2.82	4.00	4.24	4.24	8.61	29.50	30.20	5.57	11.14
4.70	2.83	4.00	4.24	4.24	8.64	29.68	30.33	5.57	11.14
4.80	2.84	4.00	4.25	4.25	8.66	29.84	30.46	5.57	11.14
4.00	3.24	5.00	5.17	5.17	10.00	49.68	42.38	5.57	11.14
4.10	3.27	5.00	5.18	5.18	10.06	50.31	42.76	5.57	11.14
4.20	3.29	5.00	5.19	5.19	10.12	50.91	43.14	5.57	11.14
4.30	3.31	5.00	5.20	5.20	10.18	51.50	43.49	5.57	11.14
4.40	3.33	5.00	5.21	5.21	10.23	52.06	43.83	5.57	11.14
4.50	3.35	5.00	5.22	5.22	10.29	52.59	44.16	5.57	11.14
4.60	3.37	5.00	5.22	5.22	10.34	53.11	44.47	5.57	11.14
4.70	3.39	5.00	5.23	5.23	10.38	53.60	44.77	5.57	11.14
4.80	3.40	5.00	5.24	5.24	10.43	54.08	45.06	5.57	11.14

Additional comments about the program :

1. To calculate parameters for only one pad size, enter the same value when the program prompts for the "smallest" and "largest" size. In this case, the program will not prompt for the "increment".

2. If the pad size or image size values entered result in a situation that cannot physically exist (such as a pad greater than a sphere), an appropriate message will print.

Acknowledgements

The author gratefully acknowledges the work of the entire PAS/ICIS team in Yorktown, East Fishkill, Boca Raton, Essonnes, and Burlington. Special thanks are extended to the current development team (R. Richter, J.-C. Chastang, R. Edmundson, W. Manthei, and J. Kolber) and to W. Giuliani and R. Richter for permission to reproduce Figures 1-4.

References

1. L. F. Miller, "Controlled Collapse Reflow Chip Joining," IBM J. Res. Develop. 13, 239 (1969).
2. L. S. Goldmann and P. A. Totta, "Area Array Solder Interconnections for VLSI," Solid State Technology 26, 91 (1983).
3. Engineering specification for reject criteria : Final Chip Group B (All Purdue, ATx/MLM 2.0/1.5 UM) Groundrules, 8633619, EC-C12197.
4. R. J. LeMaster, V. Brecher, J.-C. Chastang, K. B. Kirtley, "Automatic C4 Pad Analysis System," 1984 IBM Semiconductor Equipment Engineering Workshop, Danbury, CT (30 Oct 1984).
5. K. Kirtley, J.-C. Chastang, and R. LeMaster, "Automatic C4 Pad Analysis System," in Long Abstract Volume of International Conference on Advances in Image Processing and Pattern Recognition, V. Cappellini and R. Marconi, eds., Pisa (10-12 Dec 1985).
6. T. Naegele, "Staying Home: How Some U.S. Producers Fight Back," Electronics, 80 (25 Jun 1987).
7. B. E. Dom, K. B. Kirtley, R. Bonner, J.-C. Chastang, "C4 Pad Analysis System," in Technical Proceedings Semicon/East, Semiconductor Equipment and Material Institute, Mt. View, 116 (1987).
8. B. Dom, "Machine Vision Techniques for Integrated Circuit Inspection" in Machine Vision for Inspection and Measurement, H. Freeman, ed., Academic Press Inc., 257 (1989).
9. R. Richter and K. B. Kirtley, "The Pad Analysis System : A Successful Machine Vision System for Solder Ball Inspection," Machine Vision Applications, Architectures, and Systems Integration, B. C. Batchelor, S. S. Solomon, and F. M. Waltz, eds., Proc. SPIE 1823, 233-244 (1992).
10. T. Ross and A. Townsend, "Inspection Techniques for Solder Reflow Pad Height/Volume," IBM Technical Disclosure Bulletin 22, 4068 (1980).
11. J.-C. Chastang and R. F.Koerner, "Automated C4 Pad Volume Inspection System," TR 61.0011 (#38531), IBM, Yorktown Heights, NY (10 Apr 1981).
12. J.-C. Chastang, "Oblique Viewing Attachment for Microscope," Optical System Design and Production, Proc. SPIE 399, (1983).
13. J.-C. Chastang and R. F. Koerner, "Optical System for Oblique Viewing", US Patent 4,428,676 (31 Jan. 1984).
14. "Dynamic Chip Pad Volume Analysis Tool," IBM Technical Disclosure Bulletin vol. 27, 10A (1985).
15. "Rotational Oblique Viewing Microscope for Pad Analysis System," IBM Technical Disclosure Bulletin 33, 78 (1991).
16. W. E. Blanz, J. L. C. Sanz, and E. B. Hinkle, "Image analysis methods for solderball inspection in integrated circuit manufacturing," IEEE J. Robot. Autom. (USA). 4, no 2, 129 (1988).
17. K.B. Kirtley, "An Algorithm for Detecting Partial C4 Solder Pads Imaged by the Pad Analysis System," RC 15153, IBM, Yorktown Heights, NY (1989). (IBM Confidential)

18. "Image Analysis Algorithm for Detecting Defects in Repetitive Patterns," Research Disclosure, no 324, (1991).

19. "Image Analysis Algorithm for Detecting Defects in Repetitive Shapes in Single Images," Research Disclosure, no 340, (1992).

20. R. Ray, Method and Apparatus for Inspecting Articles, US Patent 4,688,939 (25 Aug 1987)

21. R. Ray, Method and Apparatus for Inspection of Specular Three Dimensional Features, US Patent 5,058,178 (15 Oct 1991)

22. Engineering specification for terminal metals deposition : 5615562

23. East Fishkill PAS MPS : MPS 080150 PCN 063914, 8/24/89.

24. The formula for the minimum distance from the center of the pad to the edge of the chip to ensure that an overlapping pad image does not fall off the chip is

$$\frac{\left(h_r - \frac{d_e}{2}\right)\tan\theta + \left(\frac{d_e}{2}\right)\left(1 + \frac{1}{\cos^2\theta}\right)^{1/2}}{\sqrt{2}}.$$

Appendix A

Unreflowed Volume

The formula for the volume of the unreflowed solder ball can be found from the formula for the volume of the solid generated by revolving about the y axis the region bounded by the curve $y = \frac{-2h_u x}{(d_b - d_t)} + \frac{d_b h_u}{(d_b - d_t)}$, the x axis, and the line $y = h_u$.

$$V = \pi \int_0^{h_u} [g(y)]^2 dy$$

$$V = \pi \int_0^{h_u} \left(\frac{d_b}{2} - \frac{y(d_b - d_t)}{2h_u} \right)^2 dy$$

$$V = \pi \left(\frac{d_b^2 y}{4} - \frac{d_b(d_b - d_t)y^2}{4h_u} + \frac{(d_b - d_t)^2 y^3}{12h_u^2} \right) \Bigg|_0^{h_u}$$

$$V = \frac{\pi h_u \left(d_b^2 + d_b d_t + d_t^2\right)}{12} \qquad (1)$$

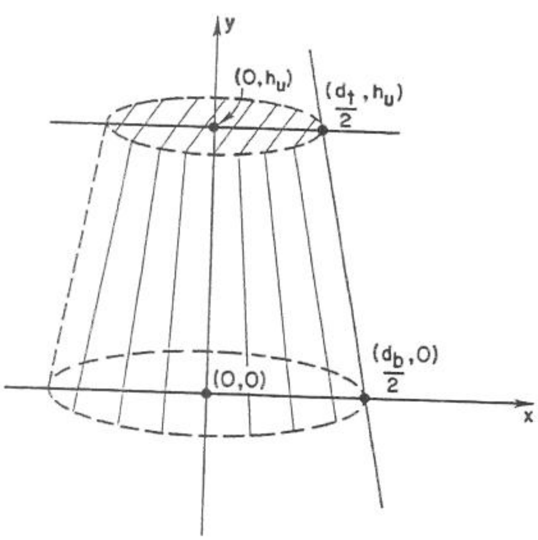

Figure 16. C4 solder ball post deposition and prior to reflow

Reflowed Volume

The formula for the volume of the reflowed solder ball can be found from the formula for the volume of the solid generated by revolving about the y axis the region bounded by the curve $y = \left(\frac{d_e^2}{4} - x^2\right)^{1/2}$, $y = \left(\frac{d_e}{2} - h_r\right)$, and $y = \frac{d_e}{2}$.

$$V = \pi \int_{\left(\frac{d_e}{2} - h_r\right)}^{\frac{d_e}{2}} [g(y)]^2 dy$$

$$V = \pi \int_{\frac{d_e}{2} - h_r}^{\frac{d_e}{2}} \left(\frac{d_e^2}{4} - y^2\right) dy$$

$$V = \pi \left(\frac{d_e^2 y}{4} - \frac{y^3}{3}\right)\Bigg|_{\frac{d_e}{2} - h_r}^{\frac{d_e}{2}}$$

$$V = \frac{\pi h_r \left(3 d_e h_r - 2h_r^2\right)}{6} \tag{4}$$

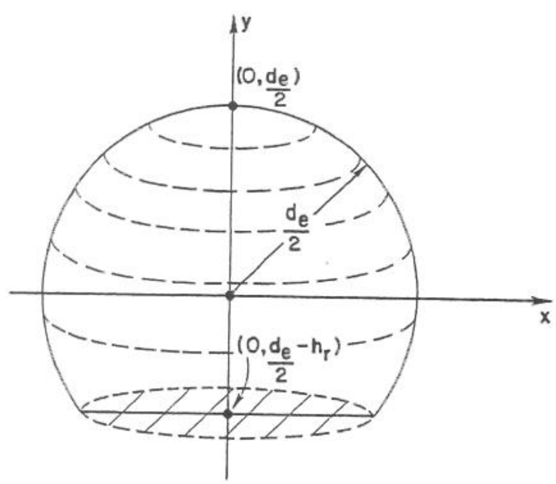

Figure 17. C4 solder ball after reflow

Volume and BLM diameter

If one knows the unreflowed variables, d_b and h_u, it is possible to calculate the reflowed variables d_e and h_r. The volume is first calculated using Equation (1). Since the volume will be the same after reflow, one can solve Eq. (6) for h_r. Eq. (6) is rewritten in terms of a cubic equation

$$h_r^3 + \frac{3d_b^2 h_r}{4} - \frac{6V}{\pi} = 0$$

Solving for h_r gives

$$h_r = \left[\frac{3V}{\pi} + \left(\frac{9V^2}{\pi} + \frac{d_b^6}{64} \right)^{1/2} \right]^{1/3} + \left[\frac{3V}{\pi} - \left(\frac{9V^2}{\pi} + \frac{d_b^6}{64} \right)^{1/2} \right]^{1/3} \quad (9)$$

And, as before,

$$d_e = \frac{d_b^2}{4h_r} + h_r \quad (5)$$

OVM Double Ellipse Pad Shadow Image Area

C4 pads greater than a hemisphere form a double ellipse OVM image. The area of this image can be found by calculating the area under the curve for one quarter of the double ellipse image and then multiplying by 4. The integration can be done most easily if the ellipse in question is centered at (0,0). The equation for this ellipse is

$$\frac{x^2}{\left(\frac{d_e}{2\cos\theta}\right)^2} + \frac{y^2}{\left(\frac{d_e}{2}\right)^2} = 1$$

We can find the area of the OVM pad image by finding the area of the region bounded by $y = \left(\left(\frac{d_e}{2}\right)^2 - \cos^2\theta x^2\right)^{\frac{1}{2}}$, the x axis, and the vertical lines $x = -\tan\theta\left(h_r - \frac{d_v}{2}\right)$, and $x = \frac{d_e}{2\cos\theta}$, and then multiplying by 4.

$$A = 4\int_{-\tan\theta\left(h_r - \frac{d_e}{2}\right)}^{\frac{d_e}{2\cos\theta}} \left(\left(\frac{d_e}{2}\right)^2 - \cos^2\theta x^2\right)^{\frac{1}{2}} dx$$

$$A = 4\cos\theta \int_{-\tan\theta\left(h_r - \frac{d_e}{2}\right)}^{\frac{d_e}{2\cos\theta}} \left(\left(\frac{d_e}{2\cos^2\theta}\right)^2 - x^2\right)^{\frac{1}{2}} dx$$

$$A = 4\cos\theta\left[\frac{x}{2}\left(\left(\frac{d_e}{2\cos\theta}\right)^2 - x^2\right)^{\frac{1}{2}} + \frac{1}{2}\left(\frac{d_e}{2\cos\theta}\right)^2 Arcsin\left(\frac{2x\cos\theta}{d_e}\right)\right]\Bigg|_{-\tan\theta\left(h_r - \frac{d_e}{2}\right)}^{\frac{d_e}{2\cos\theta}}$$

$$A = \frac{\pi\left(\frac{d_e}{2}\right)^2}{\cos\theta} + 2\left(h_r - \frac{d_e}{2}\right)\tan\theta\left[\frac{d_e^2}{4} - \left(h_r - \frac{d_e}{2}\right)^2\sin^2\theta\right]^{1/2} +$$

$$\frac{d_e^2}{2\cos\theta} Arcsin\left[\frac{2\left(h_r - \frac{d_e}{2}\right)\sin\theta}{d_e} \right] \qquad (8)$$

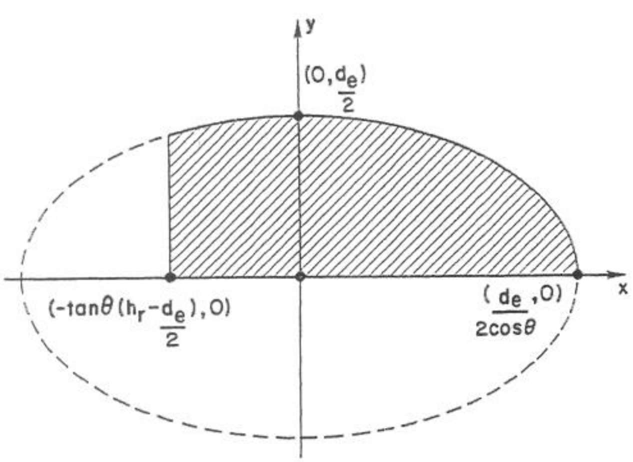

Figure 18. C4 solder ball as imaged by the OVM

Appendix B

```
/*******************************************************************************/
/*                    (C) Copyright IBM 1992, 1993                             */
/*******************************************************************************/
/*    C4_PARAM                                                                 */
/*******************************************************************************/
/*                                                                             */
/*    Function : Determines parameters for unreflowed and reflowed C4 pad      */
/*               and parameters associated with the OVM pad image.             */
/*                                                                             */
/*    Author   : K. B. Kirtley                                                 */
/*    Date     : 3/22/92                                                       */
/*                                                                             */
/*    RELEASE 1.0                                                              */
/*                                                                             */
/*    03/30/93 : Added version number to main screen.                          */
/*    03/19/93 : Added print statements to deal with situations in which       */
/*               entered parameters correspond to a pad that cannot            */
/*               physically exist (such as greater than a sphere).             */
/*                                                                             */
/*    RELEASE 0.0  Pre-release version                                         */
/*                                                                             */
/*    02/10/93 : Fixed bug in unreflowed pad shape.  Trap on top diameter      */
/*               less than zero.                                               */
/*    11/10/92 : Added ability to convert pad image parameters (length         */
/*               and width) to truncated sphere parameters                     */
/*    11/04/92 : Added to ability to analyze truncated spheres with            */
/*               equal equatorial diameters but changing reflowed heights      */
/*    10/29/92 : Added ability to call functions from menu                     */
/*    10/19/92 : Added area of OVM pad image                                   */
/*                                                                             */
/*******************************************************************************/

/*******************************************************************************/
/*   INCLUDE FILES                                                             */
/*******************************************************************************/

#include <conio.h>
#include <graph.h>
#include <stdio.h>
#include <math.h>


/*******************************************************************************/
/*   # DEFINES                                                                 */
/*******************************************************************************/

#define PI          3.14159

#define OFF         0
#define ON          1

#define TM_SPEC     0
#define SLOPE       1

#define UNREFLOWED 0
#define REFLOWED   1


/*******************************************************************************/
/*   GLOBAL VARIABLES                                                          */
/*******************************************************************************/

int file_mode;                          /* 0 : don't write results to file     */
                                        /* 1 : write results to file           */
FILE *out;                              /*                                     */
char name [40];                         /* DOS output file name                */

int volume_mode;                        /* 0 : formula in specification        */
                                        /*     top_BLM_diameter = BLM_diameter */
                                        /*     (1 - .075 * unreflowed height)  */
                                        /* 1 : top_BLM_diameter = BLM_diameter */
```

```
                                          /*     - 2 * unreflowed height / tan   */
                                          /*     (slope_angle)                    */
float slope_angle;                        /* interior slope angle for unreflowed */
                                          /*      pad                             */
float ovm_angle;                          /* OVM angle                           */

float padspacing;                         /* spacing between pads                */

float cos_ovm, sin_ovm, tan_ovm;          /* sin, cos, and tan of ovm_angle      */
float sqrt2;                              /* square root of 2                    */

float rflw_volume;                        /* volume of pad calculated using      */
                                          /*     reflowed pad parameters         */
float BLM_diameter;                       /* BLM diameter                        */
float viewed_diameter;                    /* reflowed viewed_diameter            */
float equatorial_diameter;                /* diameter of sphere. if pad > than   */
                                          /* hemisphere, this will be the viewed */
                                          /* diameter.  if pad < hemisphere,     */
                                          /* this is spherical diameter as if    */
                                          /* pad were greater than hemisphere.   */
float uflw_height;                        /* unreflowed height                   */
float rflw_height;                        /* reflowed height                     */
float pad_image_area;                     /* area of ovm pad shadow image        */

float pad_ratio_sphere;                   /*                                     */
float pad_ratio_hemisphere;               /*                                     */
float pad_ratio_limit;                    /* (1 - sin (theta)) / cos (theta)     */
                                          /* if reflowed height / viewed diam-   */
                                          /* eter / 2.0 is less than this value, */
                                          /* ovm image is a circle and height is */
                                          /* not known                           */
float shadow_ratio_sphere;                /*                                     */
float shadow_ratio_hemisphere;            /*                                     */
float shadow_ratio_limit;                 /*                                     */

float shadow_length;                      /*                                     */
float max_shadow_length;                  /*                                     */
float max_shadow_diameter;                /*                                     */

int print_index;                          /*                                     */


/******************************************************************************/
/*   EXTERNAL FUNCTIONS                                                       */
/******************************************************************************/

int main (void);
int display_param (void);
int RefreshMenu (void);

int u_blm_uflw (void);
int r_blm_rflw (void);
int r_equdm_rflw (void);
int r_viewed_rflw (void);
int r_pad_image (void);

int print_header (int);
int fprint_header (int);
int print_pad_parameters (int);
int fprint_pad_parameters (int);

int find_BLM_from_eqdiameter_rheight (void);
int find_eqdiameter_from_BLM_rheight (void);
int find_vdiameter_from_eqdiameter_rheight (void);

int find_shadow_length (void);
int find_rheight_from_slength_eqdiameter (void);
int find_eqdiameter_from_slength_rheight (void);

int find_rflw_volume (void);
int find_pad_image_area (void);


/******************************************************************************/
/*   C4_PARAM PROGRAM                                                         */
/******************************************************************************/
```

```
main ()
{
    extern int volume_mode;
    extern float slope_angle;
    extern float ovm_angle;
    extern float padspacing;
    extern float cos_ovm, sin_ovm, tan_ovm;
    extern float sqrt2;
    extern float max_shadow_length;
    extern float max_shadow_diameter;

    char response;

    /****************************************************************************/
    /*   Put up first screen                                                    */
    /****************************************************************************/

    _setvideomode (_TEXTC80);
    _clearscreen (0);

    _settextposition (6,28);
    printf ("--- C 4 _ P A R A M ---");
    _settextposition (8,34);
    printf ("Version 1.0");
    _settextposition (10,15);
    printf ("Program for calculating C4 solder ball parameters");
    _settextposition (22,33);
    printf ("K. B. Kirtley");
    _settextposition (23,18);
    printf ("IBM Research Division - Yorktown Heights, NY");
    _settextposition (24,26);
    printf ("(C) Copyright IBM Corp 1993");
    _settextposition (15,28);
    printf ("Hit any key to continue ");

    getch ();


    /****************************************************************************/
    /*   default parameters                                                     */
    /****************************************************************************/

    ovm_angle = (float) 51.66;
    volume_mode = 1;
    slope_angle = (float) 75.0;
    padspacing = (float) (200.0 / 25.4);
    file_mode = OFF;

    sqrt2 = (float) sqrt ((double) 2.0);


    /****************************************************************************/
    /*   trig functions                                                         */
    /****************************************************************************/

    cos_ovm = (float) cos ((double) (ovm_angle * (float) PI / (float) 180.0));
    sin_ovm = (float) sin ((double) (ovm_angle * (float) PI / (float) 180.0));
    tan_ovm = (float) tan ((double) (ovm_angle * (float) PI / (float) 180.0));

    pad_ratio_limit = ((float) 1.0 - sin_ovm) / (float) 2.0 / cos_ovm;
    pad_ratio_hemisphere = (float) .5;
    pad_ratio_sphere = (float) 1.0;

    shadow_ratio_limit = (float) 1.0;
    shadow_ratio_hemisphere = cos_ovm;
    shadow_ratio_sphere = cos_ovm / ((float) 1.0 + sin_ovm);


    /****************************************************************************/
    /*   calculate maximum length and width of pad image shadow before          */
    /*   overlap between images                                                 */
    /****************************************************************************/

    max_shadow_length = sqrt2 * padspacing;
    max_shadow_diameter = max_shadow_length / (float) 2.0;

```

```
    /*****************************************************************************/
    /*   Put up C4_PARAM FUNCTION MENU and service request                       */
    /*****************************************************************************/

    do
    {
       _setvideomode (_TEXTC80);
       _clearscreen (0);

       RefreshMenu ();

       response = (char) getche ();

       switch (response)
       {
          case 'b' :
             r_blm_rflw ();
             break;

          case 'd' :
             display_param ();
             break;

          case 'e' :
             break;

          case 'i' :
             r_pad_image ();
             break;

          case 'q' :
             r_equdm_rflw ();
             break;

          case 'u' :
             u_blm_uflw ();
             break;

          case 'v' :
             r_viewed_rflw ();
             break;

          default :
             break;
       }
    } while (response != 'e');

    return (0);
}


/*********************************************************************************/
/*   Print a new copy of the command menu                                        */
/*********************************************************************************/

RefreshMenu ()
{
    _clearscreen (0);

    _settextposition (2,18);
    printf ("C 4 _ P A R A M   F U N C T I O N   M E N U");
    printf ("\n\n");
    printf ("SYSTEM PARAMETERS\n\n");
    printf ("   d : display/change current system parameters\n\n");
    printf ("CALCULATE REFLOWED SOLDER BALL PARAMETERS ");
    printf ("FROM THE FOLLOWING INITIAL CONDITIONS\n\n");
    printf ("   u : unreflowed pad - BLM diameter and unreflowed height\n\n");
    printf ("   b : reflowed pad - BLM diameter and reflowed height\n");
    printf ("   q : reflowed pad - equatorial diameter and reflowed height\n");
    printf ("   v : reflowed pad - viewed diameter and reflowed height\n\n");
    printf ("   i : reflowed pad - OVM pad image length and width\n\n");
    printf ("   e : EXIT program\n");
    printf ("\n\n");
    printf ("Enter choice : ");

    return (0);
}
```

```
/*******************************************************************************/
/*   display parameters                                                        */
/*******************************************************************************/

display_param ()
{
    extern int file_mode;
    extern int volume_mode;
    extern float slope_angle;
    extern float ovm_angle;
    extern float padspacing;
    extern float max_shadow_length;
    extern float max_shadow_diameter;
    extern float sqrt2;

    char response;

    do
    {
        _clearscreen (0);

        _settextposition (2,24);
        printf ("S Y S T E M  P A R A M E T E R S");
        _settextposition (5,14);
        printf ("o : OVM angle   : %5.2f degrees\n", ovm_angle);
        _settextposition (6,14);
        if (volume_mode == TM_SPEC)
        {
            printf ("v : volume_mode : terminal metals spec\n");
        }

        if (volume_mode == SLOPE)
        {
            printf ("v : volume_mode : slope angle\n");
            _settextposition (7,14);
            printf ("a : angle       : %5.2f degrees\n", slope_angle);
        }
        _settextposition (8,14);
        printf ("p : padspacing  : %5.2f mils = %6.2f microns\n",
            padspacing, padspacing * (float) 25.4);

        _settextposition (9,14);
        if (file_mode == OFF) printf ("f : file_mode   : 0 - will not write output to disk");
        if (file_mode == ON) printf ("f : file_mode   : 1 - will write output to disk");

        _settextposition (11,14);
        printf ("e : EXIT this screen");

        _settextposition (16,17);
        printf ("Enter character to change system parameter : ");
        response = (char) getche ();

        switch (response)
        {
            case 'o' :
                do
                {
                    _settextposition (18,17);
                    printf ("Enter new OVM angle (>= 30 & <= 90 degrees) : ");
                    scanf ("%f", &ovm_angle);
                } while (ovm_angle < (float) 30.0 || ovm_angle > (float) 90.0);

                /******************************************************************/
                /*   recalculate trig functions for new OVM angle                 */
                /******************************************************************/

                cos_ovm = (float) cos ((double) (ovm_angle * (float) PI /
                    (float) 180.0));
                sin_ovm = (float) sin ((double) (ovm_angle * (float) PI /
                    (float) 180.0));
                tan_ovm = (float) tan ((double) (ovm_angle * (float) PI /
                    (float) 180.0));
                pad_ratio_limit = ((float) 1.0 - sin_ovm) / (float) 2.0 / cos_ovm;

                break;
```

```
            case 'v' :
               do
               {
                  _settextposition (18,15);
                  printf ("Enter %d for terminal spec or %d for slope angle : ",
                     TM_SPEC, SLOPE);
                  scanf ("%d", &volume_mode);
               } while (volume_mode != TM_SPEC && volume_mode != SLOPE);

               break;

            case 'a' :
               do
               {
                  _settextposition (18,5);
                  printf ("Enter new angle for slope of unreflowed pad ");
                  printf ("(>= 45 & <= 90 degrees) : ");
                  scanf ("%f", &slope_angle);
               } while (slope_angle < (float) 45.0 || slope_angle > (float) 90.0);

               break;

            case 'p' :
               do
               {
                  _settextposition (18,18);
                  printf ("Enter new pad spacing (25 - 1000 microns) : ");
                  scanf ("%f", &padspacing);
               } while (padspacing < (float) 25 || padspacing > (float) 1000.0);

               padspacing = padspacing / (float) 25.4;

               /********************************************************************/
               /*   calculate maximum length and width of pad image shadow         */
               /*   before overlap between images                                  */
               /********************************************************************/

               max_shadow_length = sqrt2 * padspacing;
               max_shadow_diameter = max_shadow_length / (float) 2.0;

               break;

            case 'f' :
               do
               {
                  _settextposition (18,10);
                  printf ("Enter 0 to suppress writing to disk ");
                  printf ("or 1 to write to disk : ");
                  scanf ("%d", &file_mode);
               } while (file_mode != OFF && file_mode != ON);

               break;

            case 'e' :
               break;

            default :
               break;
         }

   } while (response != 'e');

   return (0);
}


/*********************************************************************************/
/*   unreflowed height and BLM diameter                                          */
/*********************************************************************************/

u_blm_uflw ()
{
   extern int file_mode;
   extern FILE *out;

   extern float BLM_diameter;
```

```
    extern float uflw_height;
    extern float rflw_height;

    float BLM_diameter_small;           /* BLM diameter - smallest value       */
    float BLM_diameter_big;             /* BLM diameter - largest value        */
    float BLM_diameter_increment;       /* BLM diameter - increment            */

    float uflw_top_diameter;            /* diameter of top of unreflowed pad   */

    float uflw_height_small;            /* unreflowed height - smallest value                    */
    float uflw_height_big;              /* unreflowed height - largest value                     */
    float uflw_height_increment;        /* unreflowed height - increment                         */

    float uflw_volume;                  /* volume of pad calculated using      */
                                        /* unreflowed pad parameters           */


   /***************************************************************************/
   /*  Input parameters                                                       */
   /***************************************************************************/

   _clearscreen (0);
   _settextposition (2,12);
   printf ("I N P U T   D A T A   F O R   B L M   D I A M E T E R   A N D");
   _settextposition (3,24);
   printf ("U N R E F L O W E D   H E I G H T");

   _settextposition (6,5);
   printf ("Requesting range of data for BLM diameter");

   do
   {
      _settextposition (8,5);
      printf ("Enter smallest BLM diameter (> 0) in mils                : ");
      scanf ("%f", &BLM_diameter_small);
   } while (BLM_diameter_small <= (float) 0.0);

   do
   {
      _settextposition (9,5);
      printf ("Enter largest BLM diameter (>= smallest) in mils        : ");
      scanf ("%f", &BLM_diameter_big);
   } while (BLM_diameter_big < BLM_diameter_small);

   if (BLM_diameter_small == BLM_diameter_big)
   {
      BLM_diameter_increment = (float) 1.0;
   }
   else
   {
      do
      {
         _settextposition (10,5);
         printf ("Enter BLM diameter increment (> 0) in mils              : ");
         scanf ("%f", &BLM_diameter_increment);
      } while (BLM_diameter_increment <= (float) 0.0);
   }

   _settextposition (13,5);
   printf ("Requesting range of data for unreflowed height");

   do
   {
      _settextposition (15,5);
      printf ("Enter smallest unreflowed height (> 0) in mils          : ");
      scanf ("%f", &uflw_height_small);
   } while (uflw_height_small <= (float) 0.0);

   do
   {
      _settextposition (16,5);
      printf ("Enter largest unreflowed height (>= smallest) in mils : ");
      scanf ("%f", &uflw_height_big);
   } while (uflw_height_big < uflw_height_small);

   if (uflw_height_small == uflw_height_big)
   {
```

```
        uflw_height_increment = (float) 1.0;
    }
    else
    {
        do
        {
            _settextposition (17,5);
            printf ("Enter unreflowed height increment (> 0) in mils       : ");
            scanf ("%f", &uflw_height_increment);
        } while (uflw_height_increment <= (float) 0.0);
    }


    /****************************************************************************/
    /*  Write header to files                                                   */
    /****************************************************************************/

    if (file_mode == ON)
    {
        fprint_header (UNREFLOWED);
    }


    /****************************************************************************/
    /*  Write header to terminal                                                */
    /****************************************************************************/

    print_header (UNREFLOWED);


    /****************************************************************************/
    /*  Process data                                                            */
    /****************************************************************************/


    for (BLM_diameter = BLM_diameter_small;
        BLM_diameter <= BLM_diameter_big;
        BLM_diameter += BLM_diameter_increment)
    {
        for (uflw_height = uflw_height_small;
            uflw_height <= uflw_height_big;
            uflw_height += uflw_height_increment)
        {

            /**********************************************************************/
            /*   volume of pad based on formula for unreflowed pad                */
            /**********************************************************************/

            if (volume_mode == 0)
            {
                /******************************************************************/
                /*  use formula in terminal metals specification                  */
                /******************************************************************/

                uflw_top_diameter = BLM_diameter * ((float) 1.0 - (float)
                    .075 * uflw_height);
            }
            else
            {
                /******************************************************************/
                /*  use formula based on interior angle                           */
                /******************************************************************/

                uflw_top_diameter = BLM_diameter - (float) 2.0 *
                    uflw_height / tan (slope_angle * PI / 180.0);
            }

            if (uflw_top_diameter >= (float) 0.0)
            {
                uflw_volume = (float) PI * uflw_height * (BLM_diameter * BLM_diameter
                    + BLM_diameter * uflw_top_diameter + uflw_top_diameter *
                    uflw_top_diameter) / (float) 12.0;


                /******************************************************************/
                /*   calculate reflowed parameters from volume and                */
                /*   BLM_diameter                                                 */
```

```
/**********************************************************************/

{
   /*******************************************************************/
   /*   solve cubic equation for reflowed height                      */
   /*******************************************************************/

   double temp_double;
   double a_3, minusb_2;
   double A, B;

   minusb_2 = (double) ((float) 3.0 * uflw_volume / PI);
   a_3 = (double) (BLM_diameter * BLM_diameter / (float) 4.0);

   temp_double = minusb_2 * minusb_2 + a_3 * a_3 * a_3;
   temp_double = sqrt (temp_double);

   A = minusb_2 + temp_double;

   if (A < (double) 0)
      A = -pow (-A, (double) 1.0 / (double) 3.0);
   else
      A = pow (A, (double) 1.0 / (double) 3.0);

   B = minusb_2 - temp_double;

   if (B < (double) 0)
      B = - pow (-B, (double) 1.0 / (double) 3.0);
   else
      B = pow (B, (double) 1.0 / (double) 3.0);

   rflw_height = (float) (A + B);
}


/**********************************************************************/
/*   calculate equatorial diameter                                    */
/**********************************************************************/

find_eqdiameter_from_BLM_rheight ();


/**********************************************************************/
/*   calculate viewed diameter                                        */
/**********************************************************************/

find_vdiameter_from_eqdiameter_rheight ();


/**********************************************************************/
/*   calculate pad volume                                             */
/**********************************************************************/

find_rflw_volume ();


/**********************************************************************/
/*   calculate pad shadow length                                      */
/**********************************************************************/

find_shadow_length ();


/**********************************************************************/
/*   calculate pad image area                                         */
/**********************************************************************/

find_pad_image_area ();


/**********************************************************************/
/* Print values                                                       */
/**********************************************************************/

print_pad_parameters (UNREFLOWED);

```

```
                /*********************************************************************/
                /*  Print values                                                     */
                /*********************************************************************/

                if (file_mode == ON)
                {
                   fprint_pad_parameters (UNREFLOWED);
                }
             }
             else
             {
                printf ("%5.2f  ", uflw_height);
                printf ("        ");
                printf ("%5.2f  ", BLM_diameter);
                printf ("Calculated top diameter less than zero\n");

                if (file_mode == ON)
                {
                   fprintf (out, "%5.2f  ", uflw_height);
                   fprintf (out, "        ");
                   fprintf (out, "%5.2f  ", BLM_diameter);
                   fprintf (out, "Calculated top diameter less than zero\n");
                }
             }
          }
       }

   printf ("\nHit any key to return to main menu");
   getch ();

   if (file_mode == ON) fclose (out);

   return (0);
}

/*******************************************************************************/
/*  r_equdm_rflw                                                               */
/*******************************************************************************/

r_equdm_rflw ()
{
   extern int file_mode;
   extern FILE *out;

   extern float equatorial_diameter;
   extern float rflw_height;

   float equatorial_diameter_small;
   float equatorial_diameter_big;
   float equatorial_diameter_increment;

   float rflw_height_small;
   float rflw_height_big;
   float rflw_height_increment;        /* unreflowed height - increment                 */


   /***************************************************************************/
   /*  Input parameters                                                       */
   /***************************************************************************/

   _clearscreen (0);
   _settextposition (2,12);
   printf ("I N P U T  D A T A  F O R  E Q U A T O R I A L  D I A M E T E R");
   _settextposition (3,24);
   printf ("A N D  R E F L O W E D  H E I G H T");

   _settextposition (6,5);
   printf ("Requesting range of data for equatorial diameter");

   do
   {
      _settextposition (8,5);
      printf ("Enter smallest equatorial diameter (> 0) in mils        : ");
      scanf ("%f", &equatorial_diameter_small);
   } while (equatorial_diameter_small <= (float) 0.0);

   do
```

```
  {
     _settextposition (9,5);
     printf ("Enter largest equatorial diameter (>= smallest) in mils : ");
     scanf ("%f", &equatorial_diameter_big);
  } while (equatorial_diameter_big < equatorial_diameter_small);

  if (equatorial_diameter_small == equatorial_diameter_big)
  {
     equatorial_diameter_increment = (float) 1.0;
  }
  else
  {
     do
     {
        _settextposition (10,5);
        printf ("Enter equatorial diameter increment (> 0) in mils        : ");
        scanf ("%f", &equatorial_diameter_increment);
     } while (equatorial_diameter_increment <= (float) 0.0);
  }

  _settextposition (13,5);
  printf ("Requesting range of data for reflowed height");

  do
  {
     _settextposition (15,5);
     printf ("Enter smallest reflowed height (> 0) in mils              : ");
     scanf ("%f", &rflw_height_small);
  } while (rflw_height_small <= (float) 0.0);

  do
  {
     _settextposition (16,5);
     printf ("Enter largest reflowed height (>= smallest) in mils      : ");
     scanf ("%f", &rflw_height_big);
  } while (rflw_height_big < rflw_height_small);

  if (rflw_height_small == rflw_height_big)
  {
     rflw_height_increment = (float) 1.0;
  }
  else
  {
     do
     {
        _settextposition (17,5);
        printf ("Enter reflowed height increment (> 0) in mils            : ");
        scanf ("%f", &rflw_height_increment);
     } while (rflw_height_increment <= (float) 0.0);
  }


  /****************************************************************************/
  /*  Write header to file                                                    */
  /****************************************************************************/

  if (file_mode == ON)
  {
     fprint_header (REFLOWED);
  }

  /****************************************************************************/
  /*  Write header to terminal                                                */
  /****************************************************************************/

  print_header (REFLOWED);


  /****************************************************************************/
  /*  Process data                                                            */
  /****************************************************************************/

  for (equatorial_diameter = equatorial_diameter_small;
     equatorial_diameter <= equatorial_diameter_big;
     equatorial_diameter += equatorial_diameter_increment)
  {
     for (rflw_height = rflw_height_small;
```

```
            rflw_height <= rflw_height_big;
            rflw_height += rflw_height_increment)
        {
            if (rflw_height < equatorial_diameter)
            {
                /********************************************************************/
                /*   calculate BLM diameter                                         */
                /********************************************************************/

                find_BLM_from_eqdiameter_rheight ();


                /********************************************************************/
                /*   calculate viewed diameter                                      */
                /********************************************************************/

                find_vdiameter_from_eqdiameter_rheight ();


                /********************************************************************/
                /*  calculate pad volume                                            */
                /********************************************************************/

                find_rflw_volume ();

                /********************************************************************/
                /*  calculate pad shadow length                                     */
                /********************************************************************/

                find_shadow_length ();

                /********************************************************************/
                /*  calculate pad image area                                        */
                /********************************************************************/

                find_pad_image_area ();


                /********************************************************************/
                /*  print parameters to terminal                                    */
                /********************************************************************/

                print_pad_parameters (REFLOWED);


                /********************************************************************/
                /*  print parameters to file                                        */
                /********************************************************************/

                if (file_mode == ON)
                {
                    fprint_pad_parameters (REFLOWED);
                }
            }
            else
            {
                printf ("%5.2f  ", rflw_height);
                printf ("        ");
                printf ("%5.2f  ", equatorial_diameter);
                printf ("Pad parameters greater than a sphere\n");

                if (file_mode == ON)
                {
                    fprintf (out, "%5.2f  ", rflw_height);
                    fprintf (out, "        ");
                    fprintf (out, "%5.2f  ", equatorial_diameter);
                    fprintf (out, "Pad parameters greater than a sphere\n");
                }
            }
        }
    }

    printf ("\nHit any key to return to main menu");
    getch ();

    if (file_mode == ON) fclose (out);

```

```c
    return (0);
}

/*****************************************************************************/
/*   reflowed height and BLM diameter                                        */
/*****************************************************************************/

r_blm_rflw ()
{
    extern int file_mode;
    extern FILE *out;

    extern float BLM_diameter;
    extern float rflw_height;

    float BLM_diameter_small;           /* BLM diameter - smallest value      */
    float BLM_diameter_big;             /* BLM diameter - largest value       */
    float BLM_diameter_increment;       /* BLM diameter - increment           */
    float rflw_height_small;            /* reflowed height - smallest value   */
    float rflw_height_big;              /* reflowed height - largest value    */
    float rflw_height_increment;        /* reflowed height - increment        */


    /*************************************************************************/
    /*  Input parameters                                                     */
    /*************************************************************************/

    _clearscreen (0);
    _settextposition (2,12);
    printf ("I N P U T  D A T A  F O R  B L M  D I A M E T E R  A N D");
    _settextposition (3,26);
    printf ("R E F L O W E D  H E I G H T");

    _settextposition (6,5);
    printf ("Requesting range of data for BLM diameter");

    do
    {
       _settextposition (8,5);
       printf ("Enter smallest BLM diameter (> 0) in mils            : ");
       scanf ("%f", &BLM_diameter_small);
    } while (BLM_diameter_small <= (float) 0.0);

    do
    {
       _settextposition (9,5);
       printf ("Enter largest BLM diameter (>= smallest) in mils     : ");
       scanf ("%f", &BLM_diameter_big);
    } while (BLM_diameter_big < BLM_diameter_small);

    if (BLM_diameter_small == BLM_diameter_big)
    {
       BLM_diameter_increment = (float) 1.0;
    }
    else
    {
       do
       {
          _settextposition (10,5);
          printf ("Enter BLM diameter increment (> 0) in mils           : ");
          scanf ("%f", &BLM_diameter_increment);
       } while (BLM_diameter_increment <= (float) 0.0);
    }

    _settextposition (13,5);
    printf ("Requesting range of data for reflowed height");

    do
    {
       _settextposition (15,5);
       printf ("Enter smallest reflowed height (> 0) in mils         : ");
       scanf ("%f", &rflw_height_small);
    } while (rflw_height_small <= (float) 0.0);

    do
    {
       _settextposition (16,5);
```

```
      printf ("Enter largest reflowed height (>= smallest) in mils : ");
      scanf ("%f", &rflw_height_big);
   } while (rflw_height_big < rflw_height_small);

   if (rflw_height_small == rflw_height_big)
   {
      rflw_height_increment = (float) 1.0;
   }
   else
   {
      do
      {
         _settextposition (17,5);
         printf ("Enter reflowed height increment (> 0) in mils       : ");
         scanf ("%f", &rflw_height_increment);
      } while (rflw_height_increment <= (float) 0.0);
   }

   /****************************************************************************/
   /*  Write header to files                                                   */
   /****************************************************************************/

   if (file_mode == ON)
   {
      fprint_header (REFLOWED);
   }

   /****************************************************************************/
   /*  Write header to terminal                                                */
   /****************************************************************************/

   print_header (REFLOWED);


   /****************************************************************************/
   /*  Process data                                                            */
   /****************************************************************************/

   for (BLM_diameter = BLM_diameter_small;
      BLM_diameter <= BLM_diameter_big;
      BLM_diameter += BLM_diameter_increment)
   {
      for (rflw_height = rflw_height_small;
         rflw_height <= rflw_height_big;
         rflw_height += rflw_height_increment)
      {
         /*********************************************************************/
         /*   calculate equatorial diameter                                   */
         /*********************************************************************/

         find_eqdiameter_from_BLM_rheight ();


         /*********************************************************************/
         /*   calculate viewed diameter                                       */
         /*********************************************************************/

         find_vdiameter_from_eqdiameter_rheight ();

         /*********************************************************************/
         /*  calculate pad volume                                             */
         /*********************************************************************/

         find_rflw_volume ();


         /*********************************************************************/
         /*  calculate pad shadow length                                      */
         /*********************************************************************/

         find_shadow_length ();


         /*********************************************************************/
         /*  calculate pad image area                                         */
         /*********************************************************************/

```

```
            find_pad_image_area ();


            /*********************************************************************/
            /*  Print values                                                     */
            /*********************************************************************/

            print_pad_parameters (REFLOWED);

            if (file_mode == ON)
            {
               fprint_pad_parameters (REFLOWED);
            }
        }
    }

    printf ("\nHit any key to return to main menu");
    getch ();

    if (file_mode == ON) fclose (out);

    return (0);
}

/*****************************************************************************/
/*   reflowed height and reflowed diameter                                   */
/*****************************************************************************/

r_viewed_rflw ()
{
    extern int file_mode;
    extern FILE *out;

    extern float BLM_diameter;
    extern float equatorial_diameter;
    extern float viewed_diameter;
    extern float rflw_height;

    float viewed_diameter_small;         /* reflowed v_diam - smallest value    */
    float viewed_diameter_big;           /* reflowed v_diam - largest value     */
    float viewed_diameter_increment;     /* reflowed v_diam - increment         */
    float rflw_height_small;             /* reflowed height - smallest value    */
    float rflw_height_big;               /* reflowed height - largest value     */
    float rflw_height_increment;         /* reflowed height - increment         */


    /*************************************************************************/
    /*  Input parameters                                                     */
    /*************************************************************************/

    _clearscreen (0);
    _settextposition (2,9);
    printf ("I N P U T  D A T A  F O R  V I E W E D  D I A M E T E R  A N D");
    _settextposition (3,26);
    printf ("R E F L O W E D  H E I G H T");

    _settextposition (6,5);
    printf ("Requesting range of data for viewed diameter");

    do
    {
       _settextposition (8,5);
       printf ("Enter smallest viewed diameter (> 0) in mils        : ");
       scanf ("%f", &viewed_diameter_small);
    } while (viewed_diameter_small <= (float) 0.0);

    do
    {
       _settextposition (9,5);
       printf ("Enter largest viewed diameter (>= smallest) in mils : ");
       scanf ("%f", &viewed_diameter_big);
    } while (viewed_diameter_big < viewed_diameter_small);

    if (viewed_diameter_small == viewed_diameter_big)
    {
       viewed_diameter_increment = (float) 1.0;
    }
```

```
    else
    {
        do
        {
            _settextposition (10,5);
            printf ("Enter viewed diameter increment (> 0) in mils        : ");
            scanf ("%f", &viewed_diameter_increment);
        } while (viewed_diameter_increment <= (float) 0.0);
    }

    _settextposition (13,5);
    printf ("Requesting range of data for reflowed height");

    do
    {
        _settextposition (15,5);
        printf ("Enter smallest reflowed height (> 0) in mils         : ");
        scanf ("%f", &rflw_height_small);
    } while (rflw_height_small <= (float) 0.0);

    do
    {
        _settextposition (16,5);
        printf ("Enter largest reflowed height (>= smallest) in mils : ");
        scanf ("%f", &rflw_height_big);
    } while (rflw_height_big < rflw_height_small);

    if (rflw_height_small == rflw_height_big)
    {
        rflw_height_increment = (float) 1.0;
    }
    else
    {
        do
        {
            _settextposition (17,5);
            printf ("Enter reflowed height increment (> 0) in mils        : ");
            scanf ("%f", &rflw_height_increment);
        } while (rflw_height_increment <= (float) 0.0);
    }


    /****************************************************************************/
    /*  Write header to files                                                   */
    /****************************************************************************/

    if (file_mode == ON)
    {
        fprint_header (REFLOWED);
    }

    /****************************************************************************/
    /*  Write header to terminal                                                */
    /****************************************************************************/

    print_header (REFLOWED);


    /****************************************************************************/
    /*  Process data                                                            */
    /****************************************************************************/

    for (viewed_diameter = viewed_diameter_small;
        viewed_diameter <= viewed_diameter_big;
        viewed_diameter += viewed_diameter_increment)
    {
        for (rflw_height = rflw_height_small;
            rflw_height <= rflw_height_big;
            rflw_height += rflw_height_increment)
        {
            /********************************************************************/
            /*   check that dimensions are valid for sphere model               */
            /********************************************************************/

            if (rflw_height < viewed_diameter)
            {
                /****************************************************************/
```

```
                /*   calculate volume checking for < hemisphere condition        */
                /*******************************************************************/

                if (rflw_height >= viewed_diameter / (float) 2.0)
                {
                   equatorial_diameter = viewed_diameter;
                   find_BLM_from_eqdiameter_rheight ();
                }
                else
                {
                   BLM_diameter = viewed_diameter;
                   find_eqdiameter_from_BLM_rheight ();
                }


                /*******************************************************************/
                /*  calculate pad volume                                           */
                /*******************************************************************/

                find_rflw_volume ();


                /*******************************************************************/
                /*  calculate pad shadow length                                    */
                /*******************************************************************/

                find_shadow_length ();


                /*******************************************************************/
                /*  calculate pad image area                                       */
                /*******************************************************************/

                find_pad_image_area ();


                /*******************************************************************/
                /*  Print values                                                   */
                /*******************************************************************/

                print_pad_parameters (REFLOWED);


                /*******************************************************************/
                /*  Print values to file                                           */
                /*******************************************************************/

                if (file_mode == ON)
                {
                   fprint_pad_parameters (REFLOWED);
                }
             }
             else
             {
                printf ("%5.2f  ", rflw_height);
                printf ("        ");
                printf ("        ");
                printf ("%5.2f  ", viewed_diameter);
                printf ("Pad parameters greater than a sphere\n");

                if (file_mode == ON)
                {
                   fprintf (out, "%5.2f  ", rflw_height);
                   fprintf (out, "        ");
                   fprintf (out, "        ");
                   fprintf (out, "%5.2f  ", viewed_diameter);
                   fprintf (out, "Pad parameters greater than a sphere\n");
                }
             }
          }
       }

    printf ("\nHit any key to return to main menu");
    getch ();

    if (file_mode == ON) fclose (out);

```

```
    return (0);
}


/*****************************************************************************/
/*   OVM pad image                                                           */
/*****************************************************************************/

r_pad_image ()
{
    extern int file_mode;
    extern FILE *out;

    extern float BLM_diameter;
    extern float viewed_diameter;
    extern float equatorial_diameter;
    extern float rflw_height;

    float viewed_diameter_small;        /* reflowed v_diam - smallest value    */
    float viewed_diameter_big;          /* reflowed v_diam - largest value     */
    float viewed_diameter_increment;    /* reflowed v_diam - increment         */
    float shadow_length_small;
    float shadow_length_big;
    float shadow_length_increment;

    float shadow_ratio;


    /*************************************************************************/
    /*  Input parameters                                                     */
    /*************************************************************************/

    _clearscreen (0);
    _settextposition (2,9);
    printf ("I N P U T  D A T A  F O R  O V M  I M A G E");

    _settextposition (6,5);
    printf ("Requesting range of data for OVM pad image diameter");

    do
    {
        _settextposition (8,5);
        printf ("Enter smallest OVM pad image diameter (> 0) in mils          : ");
        scanf ("%f", &viewed_diameter_small);
    } while (viewed_diameter_small <= (float) 0.0);

    do
    {
        _settextposition (9,5);
        printf ("Enter largest OVM pad image diameter (>= smallest) in mils : ");
        scanf ("%f", &viewed_diameter_big);
    } while (viewed_diameter_big < viewed_diameter_small);

    if (viewed_diameter_small == viewed_diameter_big)
    {
        viewed_diameter_increment = (float) 1.0;
    }
    else
    {
        do
        {
            _settextposition (10,5);
            printf ("Enter OVM pad image diameter increment (> 0) in mils       : ");
            scanf ("%f", &viewed_diameter_increment);
        } while (viewed_diameter_increment <= (float) 0.0);
    }

    _settextposition (13,5);
    printf ("Requesting range of data for OVM pad image length");

    do
    {
        _settextposition (15,5);
        printf ("Enter smallest OVM pad image length (> 0) in mils            : ");
        scanf ("%f", &shadow_length_small);
    } while (shadow_length_small <= (float) 0.0);

```

```
do
{
    _settextposition (16,5);
    printf ("Enter largest OVM pad image length (>= smallest) in mils   : ");
    scanf ("%f", &shadow_length_big);
} while (shadow_length_big < shadow_length_small);

if (shadow_length_small == shadow_length_big)
{
    shadow_length_increment = (float) 1.0;
}
else
{
    do
    {
        _settextposition (17,5);
        printf ("Enter OVM pad image length increment (> 0) in mils         : ");
        scanf ("%f", &shadow_length_increment);
    } while (shadow_length_increment <= (float) 0.0);
}


/****************************************************************************/
/*  Write header to files                                                   */
/****************************************************************************/

if (file_mode == ON)
{
    fprint_header (REFLOWED);
}

/****************************************************************************/
/*  Write header to terminal                                                */
/****************************************************************************/

print_header (REFLOWED);


/****************************************************************************/
/*  Process data                                                            */
/****************************************************************************/

for (viewed_diameter = viewed_diameter_small;
    viewed_diameter <= viewed_diameter_big;
    viewed_diameter += viewed_diameter_increment)
{
    for (shadow_length = shadow_length_small;
        shadow_length <= shadow_length_big;
        shadow_length += shadow_length_increment)
    {
        shadow_ratio = viewed_diameter / shadow_length;

        /********************************************************************/
        /*   Check for valid parameters                                     */
        /********************************************************************/

        if (shadow_ratio > shadow_ratio_limit)
        {
            printf ("        ");
            printf ("        ");
            printf ("        ");
            printf ("%5.2f  ", viewed_diameter);
            printf ("%5.2f  ", shadow_length);
            printf ("Shadow length less than shadow diameter\n");

            if (file_mode == ON)
            {
                fprintf (out, "        ");
                fprintf (out, "        ");
                fprintf (out, "        ");
                fprintf (out, "%5.2f  ", viewed_diameter);
                fprintf (out, "%5.2f  ", shadow_length);
                fprintf (out, "Shadow length less than shadow diameter\n");
            }
        }
        else if (shadow_ratio <= shadow_ratio_sphere)
        {
```

```
        printf ("        ");
        printf ("        ");
        printf ("        ");
        printf ("%5.2f  ", viewed_diameter);
        printf ("%5.2f  ", shadow_length);
        printf ("Shadow parameters greater than a sphere\n");

        if (file_mode == ON)
        {
          fprintf (out, "        ");
          fprintf (out, "        ");
          fprintf (out, "        ");
          fprintf (out, "%5.2f  ", viewed_diameter);
          fprintf (out, "%5.2f  ", shadow_length);
          fprintf (out, "Shadow parameters greater than sphere\n");
        }
      }
      else
      {
        /*******************************************************************/
        /*   Check for CIRCLE                                              */
        /*******************************************************************/

        if (shadow_ratio == shadow_ratio_limit)
        {
          BLM_diameter = viewed_diameter;
          rflw_height = BLM_diameter * pad_ratio_limit;
          find_eqdiameter_from_BLM_rheight ();
        }

        /*******************************************************************/
        /*   Check for ELLIPSE                                             */
        /*******************************************************************/

        else if (shadow_ratio >= shadow_ratio_hemisphere)
        {
          BLM_diameter = viewed_diameter;
          rflw_height = (shadow_length + sqrt ((double) shadow_length *
             shadow_length - BLM_diameter * BLM_diameter)) * cos_ovm /
             (float) 2.0 / ((float) 1.0 + sin_ovm);
          find_eqdiameter_from_BLM_rheight ();
        }

        /*******************************************************************/
        /*   Must be double ELLIPSE                                        */
        /*******************************************************************/
        else
        {
          equatorial_diameter = viewed_diameter;
          find_rheight_from_slength_eqdiameter ();
          find_BLM_from_eqdiameter_rheight ();
        }

        /*******************************************************************/
        /*  calculate pad volume                                           */
        /*******************************************************************/

        find_rflw_volume ();


        /*******************************************************************/
        /*  calculate pad image area                                       */
        /*******************************************************************/

        find_pad_image_area ();


        /*******************************************************************/
        /*  Print values                                                   */
        /*******************************************************************/

        print_pad_parameters (REFLOWED);


        /*******************************************************************/
        /*  Print values to file                                           */
        /*******************************************************************/
```

```
                if (file_mode == ON)
                {
                    fprint_pad_parameters (REFLOWED);
                }
            }
        }
    }

    printf ("\nHit any key to return to main menu");
    getch ();

    if (file_mode == ON) fclose (out);

    return (0);
}


/*******************************************************************************/
/*  Calculate pad image area                                                   */
/*******************************************************************************/

find_pad_image_area ()
{
    extern float rflw_height;
    extern float viewed_diameter;
    extern float equatorial_diameter;
    extern float pad_image_area;
    extern float shadow_length;

    float pad_ratio;


    pad_ratio = rflw_height / viewed_diameter;

    if (pad_ratio <= pad_ratio_limit)
    {
        pad_image_area = (float) PI * BLM_diameter * BLM_diameter / (float) 4.0;
    }
    else if (pad_ratio <= pad_ratio_hemisphere)
    {
        pad_image_area = (float) PI * shadow_length * BLM_diameter / (float)
            4.0;
    }
    else if (pad_ratio < pad_ratio_sphere)
    {
        float equatorial_radius;
        float area1, area2, area3, area4;

        equatorial_radius = equatorial_diameter / (float) 2.0;

        area1 = (float) PI * equatorial_radius * equatorial_radius / cos_ovm;
        area3 = (float) 2.0 * equatorial_radius * equatorial_radius / cos_ovm;
        area4 = (float) asin ((double) (sin_ovm * (rflw_height -
            equatorial_radius) / equatorial_radius));
        area3 = area3 * area4;
        area2 = (float) 2.0 * (rflw_height - equatorial_radius) * tan_ovm *
            sqrt (equatorial_radius * equatorial_radius - (rflw_height -
            equatorial_radius) * (rflw_height - equatorial_radius) * sin_ovm *
            sin_ovm);

        pad_image_area = area1 + area2 + area3;
    }
    else
    {
        printf ("\nBall is sphere or larger based on pad_ratio.");
        printf ("\nHit any key to return to main menu");
        getch ();
        return (1);
    }

    return (0);
}


/*******************************************************************************/
/*  Calculate BLM_diameter from equatorial_diameter and rflw_height            */
```

```
/*******************************************************************************/

find_BLM_from_eqdiameter_rheight ()
{
    extern float BLM_diameter;
    extern float equatorial_diameter;
    extern float rflw_height;

    BLM_diameter = (float) 2.0 * sqrt ((double) (equatorial_diameter *
       rflw_height - rflw_height * rflw_height));

    return (0);
}


/*******************************************************************************/
/*  Calculate equaotrial_diameter from BLM_diameter and rflw_height           */
/*******************************************************************************/

find_eqdiameter_from_BLM_rheight ()
{
    extern float equatorial_diameter;
    extern float BLM_diameter;
    extern float rflw_diameter;

    equatorial_diameter = rflw_height + BLM_diameter * BLM_diameter /
       (float) 4.0 / rflw_height;

    return (0);
}


/*******************************************************************************/
/*  Calculate equaotrial_diameter from shadow length and rflw_height          */
/*******************************************************************************/

find_eqdiameter_from_slength_rheight ()
{
    extern float shadow_length;
    extern float equatorial_diameter;
    extern float rflw_height;

    equatorial_diameter = (shadow_length / (float) 2.0 - rflw_height * tan_ovm)
       / pad_ratio_limit;

    return (0);
}


/*******************************************************************************/
/*  Calculate rflw_height from equaotrial_diameter and shadow length          */
/*******************************************************************************/

find_rheight_from_slength_eqdiameter ()
{
    extern float rflw_height;
    extern float shadow_length;
    extern float equatorial_diameter;

    rflw_height = (shadow_length / (float) 2.0 - equatorial_diameter *
       pad_ratio_limit) / tan_ovm;

    return (0);
}


/*******************************************************************************/
/*  Calculate reflowed pad volume                                              */
/*******************************************************************************/

find_rflw_volume ()
{
    extern float rflw_volume;
    extern float BLM_diameter;
    extern float rflw_height;

    rflw_volume = ((float) 4.0 * rflw_height * rflw_height +
```

```
        (float) 3.0 * BLM_diameter * BLM_diameter) * PI *
        rflw_height / (float) 24.0;

    return (0);
}


/*********************************************************************************/
/*   Calculate OVM pad shadow length                                             */
/*********************************************************************************/

find_shadow_length ()
{
    extern float rflw_height;
    extern float BLM_diameter;
    extern float viewed_diameter;
    extern float equatorial_diameter;
    extern float shadow_length;
    float pad_ratio;


    pad_ratio = rflw_height / viewed_diameter;

    if (pad_ratio <= pad_ratio_limit)
    {
        shadow_length = BLM_diameter;
    }
    else if (pad_ratio < pad_ratio_sphere)
    {
        shadow_length = (float) 2.0 * (rflw_height * tan_ovm + equatorial_diameter
            * pad_ratio_limit);
    }
    else
    {
        printf ("\nBall is sphere or larger based on pad_ratio.");
        printf ("\nHit any key to return to main menu");
        getch ();
        return (1);
    }

    return (0);
}


/*********************************************************************************/
/*   Calculate viewed_diameter from equatorial_diameter and rflw_height          */
/*********************************************************************************/

find_vdiameter_from_eqdiameter_rheight ()
{
    extern float viewed_diameter;
    extern float rflw_height;
    extern float equatorial_diameter;

    if (rflw_height >= equatorial_diameter / (float) 2.0)
    {
        viewed_diameter = equatorial_diameter;
    }
    else
    {
        viewed_diameter = BLM_diameter;
    }

    return (0);
}


/*********************************************************************************/
/*   Write file header to hard disk                                              */
/*********************************************************************************/

fprint_header (int type)
{
    extern char name [];

    printf ("\n\nEnter name of output image file : ");
    scanf ("%s",name);
```

```
    out = fopen (name,"w");

    if (type == UNREFLOWED) fprintf (out, "urflw  ");
    fprintf (out, "reflw  BLM    equat  viewd  shadw  reflow  image   max    max\n");
    if (type == UNREFLOWED) fprintf (out, "ht     ");
    fprintf (out, "ht     diamt  diamt  diamt  lngth  volume  area    diamt  lngth\n\n");

    return (0);
}


/*******************************************************************************/
/*   Print file header to terminal                                             */
/*******************************************************************************/

print_header (int type)
{
    extern int print_index;             /* index for number of terminal lines  */

    print_index = 4;

    _clearscreen (0);

    if (type == UNREFLOWED) printf ("urflw  ");
    printf ("reflw  BLM    equat  viewd  shadw  reflow  image   max    max\n");
    if (type == UNREFLOWED) printf ("ht     ");
    printf ("ht     diamt  diamt  diamt  lngth  volume  area    diamt  lngth\n\n");

    return (0);
}


/*******************************************************************************/
/*   Print pad parameters to terminal                                          */
/*******************************************************************************/

print_pad_parameters (int type)
{
    extern int print_index;

    extern float uflw_height;
    extern float rflw_height;
    extern float BLM_diameter;
    extern float equatorial_diameter;
    extern float viewed_diameter;
    extern float shadow_length;
    extern float rflw_volume;
    extern float pad_image_area;
    extern float max_shadow_diameter;
    extern float max_shadow_length;


    if (print_index == 22)
    {
        _settextposition (24, 0);
        printf ("Hit any key to continue to end of data");
        getch ();
        printf ("\n");
    }

    if (type == UNREFLOWED) printf ("%5.2f  ", uflw_height);
    printf ("%5.2f  ", rflw_height);
    printf ("%5.2f  ", BLM_diameter);
    printf ("%5.2f  ", equatorial_diameter);
    printf ("%5.2f  ", viewed_diameter);
    printf ("%5.2f  ", shadow_length);
    printf ("%6.2f  ", rflw_volume);
    printf ("%6.2f  ", pad_image_area);
    printf ("%5.2f  ", max_shadow_diameter);
    printf ("%5.2f\n", max_shadow_length);

    print_index++;

    return (0);
}
```

```
/*********************************************************************************/
/*   print pad parameters to file on hard disk                                   */
/*********************************************************************************/

fprint_pad_parameters (int type)
{
    extern float uflw_height;
    extern float rflw_height;
    extern float BLM_diameter;
    extern float equatorial_diameter;
    extern float viewed_diameter;
    extern float shadow_length;
    extern float rflw_volume;
    extern float pad_image_area;
    extern float max_shadow_diameter;
    extern float max_shadow_length;

    if (type == UNREFLOWED) fprintf (out, "%5.2f  ", uflw_height);
    fprintf (out, "%5.2f  ", rflw_height);
    fprintf (out, "%5.2f  ", BLM_diameter);
    fprintf (out, "%5.2f  ", equatorial_diameter);
    fprintf (out, "%5.2f  ", viewed_diameter);
    fprintf (out, "%5.2f  ", shadow_length);
    fprintf (out, "%6.2f  ", rflw_volume);
    fprintf (out, "%6.2f  ", pad_image_area);
    fprintf (out, "%5.2f  ", max_shadow_diameter);
    fprintf (out, "%5.2f\n", max_shadow_length);

    return (0);
}
```

Appendix C

urflw ht	reflw ht	BLM diamt	equat diamt	viewd diamt	shadw lngth	reflow volume	image area	max diamt	max lngth
3.50	2.12	3.00	3.18	3.18	6.47	12.50	17.04	5.57	11.14
3.60	2.13	3.00	3.19	3.19	6.49	12.59	17.13	5.57	11.14
3.70	2.14	3.00	3.19	3.19	6.52	12.68	17.22	5.57	11.14
3.80	2.15	3.00	3.19	3.19	6.54	12.75	17.29	5.57	11.14
3.90	2.15	3.00	3.20	3.20	6.55	12.82	17.36	5.57	11.14
4.00	2.16	3.00	3.20	3.20	6.57	12.88	17.42	5.57	11.14
4.10	2.16	3.00	3.20	3.20	6.58	12.94	17.47	5.57	11.14
4.20	2.17	3.00	3.21	3.21	6.59	12.98	17.52	5.57	11.14
4.30	2.17	3.00	3.21	3.21	6.60	13.03	17.56	5.57	11.14
4.40	2.17	3.00	3.21	3.21	6.61	13.06	17.60	5.57	11.14
4.50	2.18	3.00	3.21	3.21	6.62	13.09	17.63	5.57	11.14
4.60	2.18	3.00	3.21	3.21	6.63	13.12	17.65	5.57	11.14
4.70	2.18	3.00	3.21	3.21	6.63	13.14	17.67	5.57	11.14
4.80	2.18	3.00	3.21	3.21	6.64	13.15	17.69	5.57	11.14
3.50	2.24	3.20	3.38	3.38	6.84	14.87	19.11	5.57	11.14
3.60	2.25	3.20	3.39	3.39	6.87	15.01	19.24	5.57	11.14
3.70	2.26	3.20	3.39	3.39	6.89	15.13	19.35	5.57	11.14
3.80	2.27	3.20	3.40	3.40	6.92	15.24	19.45	5.57	11.14
3.90	2.28	3.20	3.40	3.40	6.94	15.34	19.55	5.57	11.14
4.00	2.28	3.20	3.41	3.41	6.96	15.43	19.63	5.57	11.14
4.10	2.29	3.20	3.41	3.41	6.98	15.52	19.71	5.57	11.14
4.20	2.30	3.20	3.41	3.41	7.00	15.59	19.78	5.57	11.14
4.30	2.30	3.20	3.41	3.41	7.01	15.66	19.84	5.57	11.14
4.40	2.31	3.20	3.42	3.42	7.02	15.72	19.90	5.57	11.14
4.50	2.31	3.20	3.42	3.42	7.04	15.77	19.94	5.57	11.14
4.60	2.32	3.20	3.42	3.42	7.05	15.81	19.99	5.57	11.14
4.70	2.32	3.20	3.42	3.42	7.05	15.85	20.02	5.57	11.14
4.80	2.32	3.20	3.42	3.42	7.06	15.89	20.05	5.57	11.14
3.50	2.35	3.40	3.58	3.58	7.19	17.47	21.25	5.57	11.14
3.60	2.36	3.40	3.59	3.59	7.22	17.65	21.41	5.57	11.14
3.70	2.38	3.40	3.59	3.59	7.26	17.81	21.55	5.57	11.14
3.80	2.39	3.40	3.60	3.60	7.29	17.96	21.68	5.57	11.14
3.90	2.40	3.40	3.60	3.60	7.32	18.10	21.81	5.57	11.14
4.00	2.41	3.40	3.61	3.61	7.34	18.23	21.92	5.57	11.14
4.10	2.42	3.40	3.61	3.61	7.37	18.35	22.02	5.57	11.14
4.20	2.42	3.40	3.62	3.62	7.39	18.46	22.12	5.57	11.14
4.30	2.43	3.40	3.62	3.62	7.41	18.56	22.21	5.57	11.14
4.40	2.44	3.40	3.62	3.62	7.42	18.65	22.28	5.57	11.14
4.50	2.44	3.40	3.63	3.63	7.44	18.73	22.35	5.57	11.14
4.60	2.45	3.40	3.63	3.63	7.45	18.80	22.42	5.57	11.14
4.70	2.45	3.40	3.63	3.63	7.47	18.87	22.47	5.57	11.14
4.80	2.46	3.40	3.63	3.63	7.48	18.92	22.52	5.57	11.14
3.50	2.46	3.60	3.78	3.78	7.53	20.29	23.45	5.57	11.14
3.60	2.47	3.60	3.78	3.78	7.57	20.51	23.64	5.57	11.14

3.70	2.49	3.60	3.79	3.79	7.61	20.73	23.81	5.57	11.14
3.80	2.50	3.60	3.80	3.80	7.65	20.93	23.98	5.57	11.14
3.90	2.51	3.60	3.80	3.80	7.68	21.11	24.13	5.57	11.14
4.00	2.53	3.60	3.81	3.81	7.71	21.28	24.28	5.57	11.14
4.10	2.54	3.60	3.81	3.81	7.74	21.44	24.41	5.57	11.14
4.20	2.55	3.60	3.82	3.82	7.76	21.59	24.53	5.57	11.14
4.30	2.55	3.60	3.82	3.82	7.79	21.73	24.65	5.57	11.14
4.40	2.56	3.60	3.83	3.83	7.81	21.86	24.75	5.57	11.14
4.50	2.57	3.60	3.83	3.83	7.83	21.97	24.85	5.57	11.14
4.60	2.58	3.60	3.83	3.83	7.85	22.08	24.94	5.57	11.14
4.70	2.58	3.60	3.84	3.84	7.87	22.17	25.01	5.57	11.14
4.80	2.59	3.60	3.84	3.84	7.88	22.26	25.09	5.57	11.14
3.50	2.56	3.80	3.97	3.97	7.86	23.32	25.70	5.57	11.14
3.60	2.58	3.80	3.98	3.98	7.91	23.61	25.92	5.57	11.14
3.70	2.60	3.80	3.99	3.99	7.95	23.87	26.14	5.57	11.14
3.80	2.61	3.80	3.99	3.99	7.99	24.13	26.34	5.57	11.14
3.90	2.63	3.80	4.00	4.00	8.03	24.36	26.52	5.57	11.14
4.00	2.64	3.80	4.01	4.01	8.07	24.59	26.70	5.57	11.14
4.10	2.65	3.80	4.01	4.01	8.10	24.79	26.86	5.57	11.14
4.20	2.66	3.80	4.02	4.02	8.13	24.99	27.02	5.57	11.14
4.30	2.67	3.80	4.02	4.02	8.16	25.17	27.16	5.57	11.14
4.40	2.68	3.80	4.03	4.03	8.19	25.34	27.29	5.57	11.14
4.50	2.69	3.80	4.03	4.03	8.21	25.50	27.42	5.57	11.14
4.60	2.70	3.80	4.04	4.04	8.24	25.64	27.53	5.57	11.14
4.70	2.71	3.80	4.04	4.04	8.26	25.78	27.64	5.57	11.14
4.80	2.72	3.80	4.05	4.05	8.28	25.90	27.73	5.57	11.14
3.50	2.66	4.00	4.16	4.16	8.18	26.58	28.00	5.57	11.14
3.60	2.68	4.00	4.17	4.17	8.23	26.93	28.26	5.57	11.14
3.70	2.70	4.00	4.18	4.18	8.28	27.26	28.51	5.57	11.14
3.80	2.72	4.00	4.19	4.19	8.33	27.57	28.75	5.57	11.14
3.90	2.73	4.00	4.20	4.20	8.37	27.86	28.97	5.57	11.14
4.00	2.75	4.00	4.20	4.20	8.41	28.14	29.18	5.57	11.14
4.10	2.76	4.00	4.21	4.21	8.45	28.40	29.38	5.57	11.14
4.20	2.78	4.00	4.22	4.22	8.49	28.65	29.56	5.57	11.14
4.30	2.79	4.00	4.22	4.22	8.52	28.88	29.74	5.57	11.14
4.40	2.80	4.00	4.23	4.23	8.55	29.10	29.90	5.57	11.14
4.50	2.81	4.00	4.23	4.23	8.58	29.31	30.06	5.57	11.14
4.60	2.82	4.00	4.24	4.24	8.61	29.50	30.20	5.57	11.14
4.70	2.83	4.00	4.24	4.24	8.64	29.68	30.33	5.57	11.14
4.80	2.84	4.00	4.25	4.25	8.66	29.84	30.46	5.57	11.14
3.50	2.76	4.20	4.36	4.36	8.48	30.06	30.35	5.57	11.14
3.60	2.78	4.20	4.37	4.37	8.54	30.47	30.65	5.57	11.14
3.70	2.80	4.20	4.37	4.37	8.60	30.87	30.94	5.57	11.14
3.80	2.82	4.20	4.38	4.38	8.65	31.25	31.21	5.57	11.14
3.90	2.84	4.20	4.39	4.39	8.70	31.60	31.47	5.57	11.14
4.00	2.85	4.20	4.40	4.40	8.75	31.95	31.72	5.57	11.14
4.10	2.87	4.20	4.41	4.41	8.79	32.27	31.95	5.57	11.14
4.20	2.89	4.20	4.41	4.41	8.83	32.58	32.17	5.57	11.14
4.30	2.90	4.20	4.42	4.42	8.87	32.87	32.38	5.57	11.14
4.40	2.91	4.20	4.43	4.43	8.91	33.14	32.58	5.57	11.14

4.50	2.93	4.20	4.43	4.43	8.94	33.40	32.76	5.57	11.14
4.60	2.94	4.20	4.44	4.44	8.97	33.64	32.93	5.57	11.14
4.70	2.95	4.20	4.44	4.44	9.00	33.87	33.10	5.57	11.14
4.80	2.96	4.20	4.45	4.45	9.03	34.09	33.25	5.57	11.14
3.50	2.85	4.40	4.55	4.55	8.78	33.76	32.75	5.57	11.14
3.60	2.87	4.40	4.56	4.56	8.85	34.25	33.09	5.57	11.14
3.70	2.90	4.40	4.57	4.57	8.91	34.71	33.42	5.57	11.14
3.80	2.92	4.40	4.58	4.58	8.97	35.16	33.73	5.57	11.14
3.90	2.94	4.40	4.58	4.58	9.02	35.59	34.03	5.57	11.14
4.00	2.96	4.40	4.59	4.59	9.07	36.00	34.31	5.57	11.14
4.10	2.97	4.40	4.60	4.60	9.12	36.39	34.58	5.57	11.14
4.20	2.99	4.40	4.61	4.61	9.17	36.76	34.84	5.57	11.14
4.30	3.01	4.40	4.62	4.62	9.21	37.12	35.08	5.57	11.14
4.40	3.02	4.40	4.62	4.62	9.25	37.45	35.31	5.57	11.14
4.50	3.04	4.40	4.63	4.63	9.29	37.77	35.53	5.57	11.14
4.60	3.05	4.40	4.64	4.64	9.33	38.08	35.73	5.57	11.14
4.70	3.06	4.40	4.64	4.64	9.36	38.36	35.93	5.57	11.14
4.80	3.08	4.40	4.65	4.65	9.40	38.63	36.11	5.57	11.14
3.50	2.94	4.60	4.74	4.74	9.07	37.67	35.19	5.57	11.14
3.60	2.96	4.60	4.75	4.75	9.14	38.24	35.57	5.57	11.14
3.70	2.99	4.60	4.76	4.76	9.21	38.79	35.94	5.57	11.14
3.80	3.01	4.60	4.77	4.77	9.27	39.32	36.29	5.57	11.14
3.90	3.03	4.60	4.78	4.78	9.33	39.83	36.63	5.57	11.14
4.00	3.05	4.60	4.79	4.79	9.39	40.31	36.95	5.57	11.14
4.10	3.07	4.60	4.80	4.80	9.44	40.77	37.26	5.57	11.14
4.20	3.09	4.60	4.80	4.80	9.49	41.22	37.55	5.57	11.14
4.30	3.11	4.60	4.81	4.81	9.54	41.64	37.83	5.57	11.14
4.40	3.13	4.60	4.82	4.82	9.59	42.05	38.10	5.57	11.14
4.50	3.15	4.60	4.83	4.83	9.63	42.43	38.35	5.57	11.14
4.60	3.16	4.60	4.83	4.83	9.67	42.80	38.59	5.57	11.14
4.70	3.18	4.60	4.84	4.84	9.71	43.15	38.82	5.57	11.14
4.80	3.19	4.60	4.85	4.85	9.75	43.48	39.04	5.57	11.14
3.50	3.02	4.80	4.93	4.93	9.36	41.81	37.68	5.57	11.14
3.60	3.05	4.80	4.94	4.94	9.43	42.47	38.10	5.57	11.14
3.70	3.08	4.80	4.95	4.95	9.50	43.10	38.51	5.57	11.14
3.80	3.10	4.80	4.96	4.96	9.57	43.72	38.90	5.57	11.14
3.90	3.13	4.80	4.97	4.97	9.64	44.30	39.28	5.57	11.14
4.00	3.15	4.80	4.98	4.98	9.70	44.87	39.64	5.57	11.14
4.10	3.17	4.80	4.99	4.99	9.76	45.41	39.99	5.57	11.14
4.20	3.19	4.80	5.00	5.00	9.81	45.93	40.32	5.57	11.14
4.30	3.21	4.80	5.01	5.01	9.86	46.43	40.64	5.57	11.14
4.40	3.23	4.80	5.01	5.01	9.92	46.91	40.94	5.57	11.14
4.50	3.25	4.80	5.02	5.02	9.96	47.37	41.23	5.57	11.14
4.60	3.27	4.80	5.03	5.03	10.01	47.81	41.51	5.57	11.14
4.70	3.28	4.80	5.04	5.04	10.05	48.23	41.77	5.57	11.14
4.80	3.30	4.80	5.04	5.04	10.09	48.63	42.02	5.57	11.14
3.50	3.11	5.00	5.12	5.12	9.63	46.17	40.20	5.57	11.14
3.60	3.14	5.00	5.13	5.13	9.71	46.92	40.67	5.57	11.14
3.70	3.16	5.00	5.14	5.14	9.79	47.65	41.12	5.57	11.14
3.80	3.19	5.00	5.15	5.15	9.86	48.35	41.56	5.57	11.14

3.90	3.22	5.00	5.16	5.16	9.93	49.03	41.98	5.57	11.14
4.00	3.24	5.00	5.17	5.17	10.00	49.68	42.38	5.57	11.14
4.10	3.27	5.00	5.18	5.18	10.06	50.31	42.76	5.57	11.14
4.20	3.29	5.00	5.19	5.19	10.12	50.91	43.14	5.57	11.14
4.30	3.31	5.00	5.20	5.20	10.18	51.50	43.49	5.57	11.14
4.40	3.33	5.00	5.21	5.21	10.23	52.06	43.83	5.57	11.14
4.50	3.35	5.00	5.22	5.22	10.29	52.59	44.16	5.57	11.14
4.60	3.37	5.00	5.22	5.22	10.34	53.11	44.47	5.57	11.14
4.70	3.39	5.00	5.23	5.23	10.38	53.60	44.77	5.57	11.14
4.80	3.40	5.00	5.24	5.24	10.43	54.08	45.06	5.57	11.14
3.50	3.19	5.20	5.31	5.31	9.90	50.74	42.77	5.57	11.14
3.60	3.22	5.20	5.32	5.32	9.98	51.60	43.28	5.57	11.14
3.70	3.25	5.20	5.33	5.33	10.07	52.42	43.78	5.57	11.14
3.80	3.28	5.20	5.34	5.34	10.14	53.22	44.26	5.57	11.14
3.90	3.30	5.20	5.35	5.35	10.22	54.00	44.72	5.57	11.14
4.00	3.33	5.20	5.36	5.36	10.29	54.74	45.16	5.57	11.14
4.10	3.36	5.20	5.37	5.37	10.36	55.46	45.59	5.57	11.14
4.20	3.38	5.20	5.38	5.38	10.42	56.16	46.00	5.57	11.14
4.30	3.41	5.20	5.39	5.39	10.48	56.83	46.40	5.57	11.14
4.40	3.43	5.20	5.40	5.40	10.54	57.48	46.78	5.57	11.14
4.50	3.45	5.20	5.41	5.41	10.60	58.10	47.14	5.57	11.14
4.60	3.47	5.20	5.42	5.42	10.66	58.70	47.49	5.57	11.14
4.70	3.49	5.20	5.43	5.43	10.71	59.27	47.83	5.57	11.14
4.80	3.51	5.20	5.43	5.43	10.76	59.83	48.15	5.57	11.14
3.50	3.26	5.40	5.50	5.50	10.16	55.54	45.38	5.57	11.14
3.60	3.30	5.40	5.51	5.51	10.25	56.50	45.93	5.57	11.14
3.70	3.33	5.40	5.52	5.52	10.34	57.43	46.47	5.57	11.14
3.80	3.36	5.40	5.53	5.53	10.42	58.33	46.99	5.57	11.14
3.90	3.39	5.40	5.54	5.54	10.50	59.21	47.50	5.57	11.14
4.00	3.42	5.40	5.55	5.55	10.57	60.06	47.99	5.57	11.14
4.10	3.45	5.40	5.56	5.56	10.65	60.87	48.45	5.57	11.14
4.20	3.47	5.40	5.57	5.57	10.72	61.67	48.91	5.57	11.14
4.30	3.50	5.40	5.58	5.58	10.78	62.43	49.34	5.57	11.14
4.40	3.52	5.40	5.59	5.59	10.85	63.17	49.77	5.57	11.14
4.50	3.54	5.40	5.60	5.60	10.91	63.89	50.17	5.57	11.14
4.60	3.57	5.40	5.61	5.61	10.97	64.58	50.56	5.57	11.14
4.70	3.59	5.40	5.62	5.62	11.02	65.24	50.94	5.57	11.14
4.80	3.61	5.40	5.63	5.63	11.08	65.88	51.30	5.57	11.14
3.50	3.34	5.60	5.69	5.69	10.42	60.56	48.02	5.57	11.14
3.60	3.37	5.60	5.70	5.70	10.51	61.63	48.62	5.57	11.14
3.70	3.41	5.60	5.71	5.71	10.60	62.67	49.21	5.57	11.14
3.80	3.44	5.60	5.72	5.72	10.69	63.68	49.77	5.57	11.14
3.90	3.47	5.60	5.73	5.73	10.77	64.67	50.32	5.57	11.14
4.00	3.50	5.60	5.74	5.74	10.85	65.62	50.85	5.57	11.14
4.10	3.53	5.60	5.75	5.75	10.93	66.54	51.36	5.57	11.14
4.20	3.56	5.60	5.76	5.76	11.00	67.44	51.86	5.57	11.14
4.30	3.59	5.60	5.77	5.77	11.07	68.31	52.34	5.57	11.14
4.40	3.61	5.60	5.78	5.78	11.14	69.15	52.80	5.57	11.14
4.50	3.64	5.60	5.79	5.79	11.21	69.96	53.25	5.57	11.14
4.60	3.66	5.60	5.80	5.80	11.27	70.74	53.68	5.57	11.14

4.70	3.68	5.60	5.81	5.81	11.33	71.50	54.09	5.57	11.14
4.80	3.70	5.60	5.82	5.82	11.39	72.23	54.49	5.57	11.14
3.50	3.41	5.80	5.88	5.88	10.66	65.79	50.70	5.57	11.14
3.60	3.45	5.80	5.89	5.89	10.76	66.98	51.35	5.57	11.14
3.70	3.48	5.80	5.90	5.90	10.86	68.15	51.98	5.57	11.14
3.80	3.52	5.80	5.91	5.91	10.95	69.27	52.59	5.57	11.14
3.90	3.55	5.80	5.92	5.92	11.04	70.37	53.19	5.57	11.14
4.00	3.58	5.80	5.93	5.93	11.12	71.44	53.76	5.57	11.14
4.10	3.61	5.80	5.94	5.94	11.21	72.47	54.32	5.57	11.14
4.20	3.64	5.80	5.95	5.95	11.28	73.47	54.85	5.57	11.14
4.30	3.67	5.80	5.96	5.96	11.36	74.45	55.38	5.57	11.14
4.40	3.70	5.80	5.97	5.97	11.43	75.39	55.88	5.57	11.14
4.50	3.73	5.80	5.98	5.98	11.50	76.31	56.37	5.57	11.14
4.60	3.75	5.80	5.99	5.99	11.57	77.20	56.84	5.57	11.14
4.70	3.78	5.80	6.00	6.00	11.63	78.06	57.30	5.57	11.14
4.80	3.80	5.80	6.01	6.01	11.70	78.89	57.74	5.57	11.14
3.50	3.48	6.00	6.07	6.07	10.91	71.25	53.42	5.57	11.14
3.60	3.52	6.00	6.08	6.08	11.01	72.57	54.12	5.57	11.14
3.70	3.56	6.00	6.09	6.09	11.11	73.85	54.79	5.57	11.14
3.80	3.59	6.00	6.10	6.10	11.21	75.10	55.45	5.57	11.14
3.90	3.63	6.00	6.11	6.11	11.30	76.32	56.09	5.57	11.14
4.00	3.66	6.00	6.12	6.12	11.39	77.50	56.71	5.57	11.14
4.10	3.70	6.00	6.13	6.13	11.48	78.66	57.31	5.57	11.14
4.20	3.73	6.00	6.14	6.14	11.56	79.77	57.89	5.57	11.14
4.30	3.76	6.00	6.15	6.15	11.64	80.86	58.46	5.57	11.14
4.40	3.79	6.00	6.16	6.16	11.72	81.92	59.00	5.57	11.14
4.50	3.81	6.00	6.17	6.17	11.79	82.95	59.53	5.57	11.14
4.60	3.84	6.00	6.18	6.18	11.86	83.94	60.05	5.57	11.14
4.70	3.87	6.00	6.19	6.19	11.93	84.91	60.54	5.57	11.14
4.80	3.89	6.00	6.20	6.20	12.00	85.85	61.03	5.57	11.14