

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

NON-CIRCULATING

NOTICE

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

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.

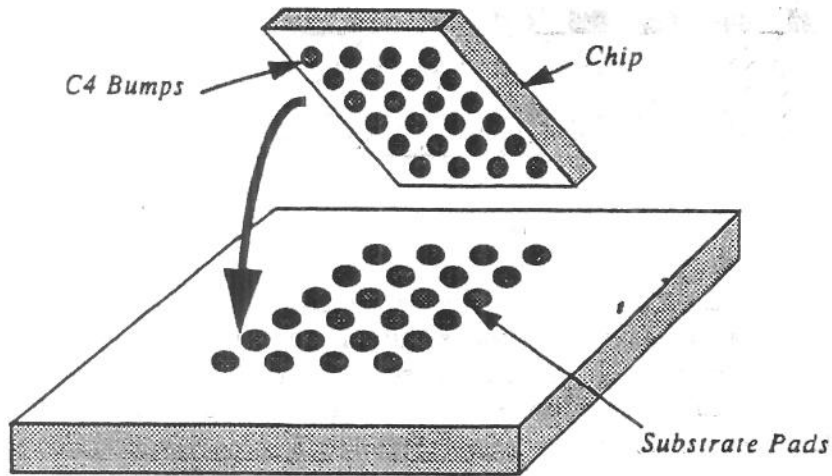


Figure 1. C4 technology

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.

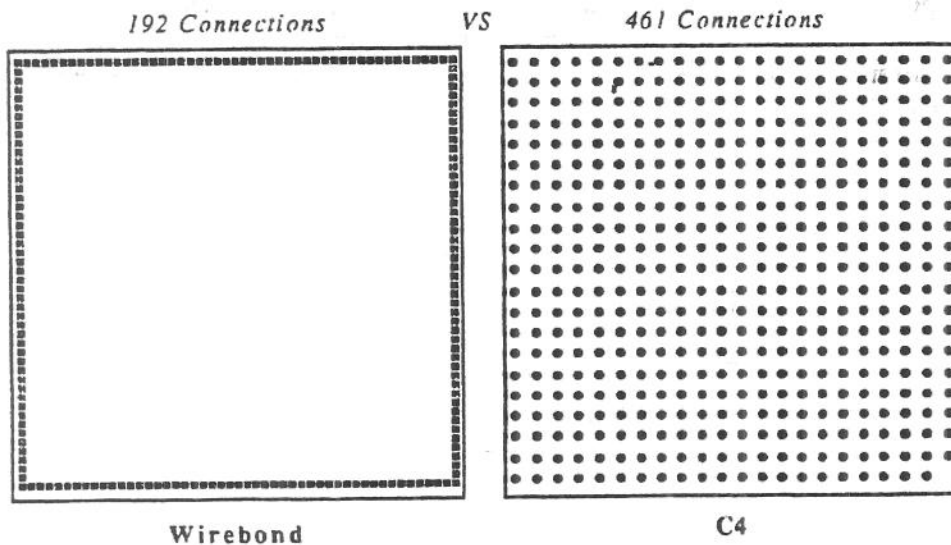


Figure 2. Interconnect density of wire bonding vs C4

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.

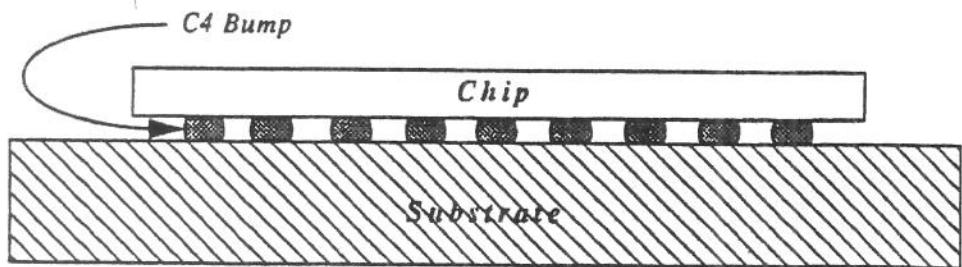


Figure 3. Chip joined to substrate with C4 solder pads

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 $\frac{2}{3}$ of the average volume of the adjacent pads³. 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.

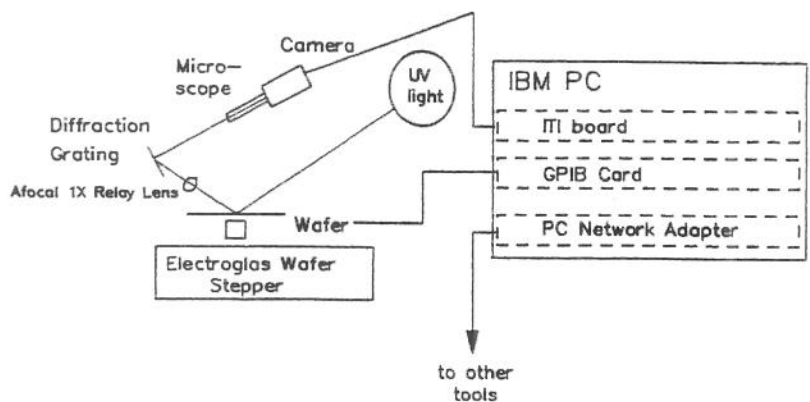


Figure 4. System diagram of the Pad Analysis System

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 application¹⁶⁻¹⁹.

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.

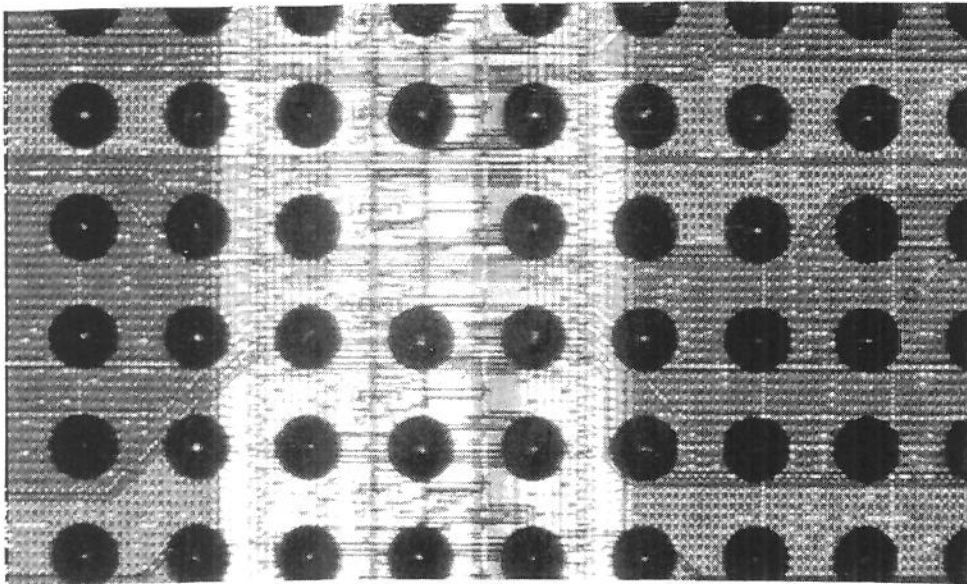


Figure 5. Bipolar chip imaged with bright field illumination

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

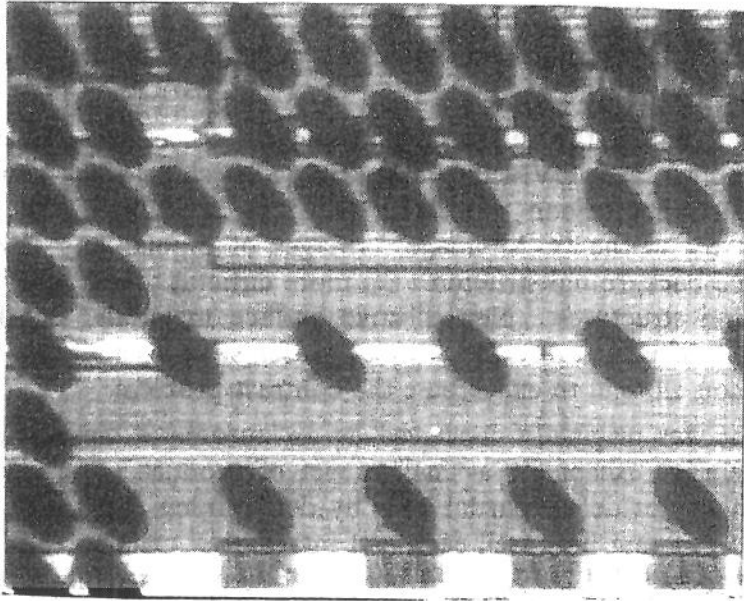
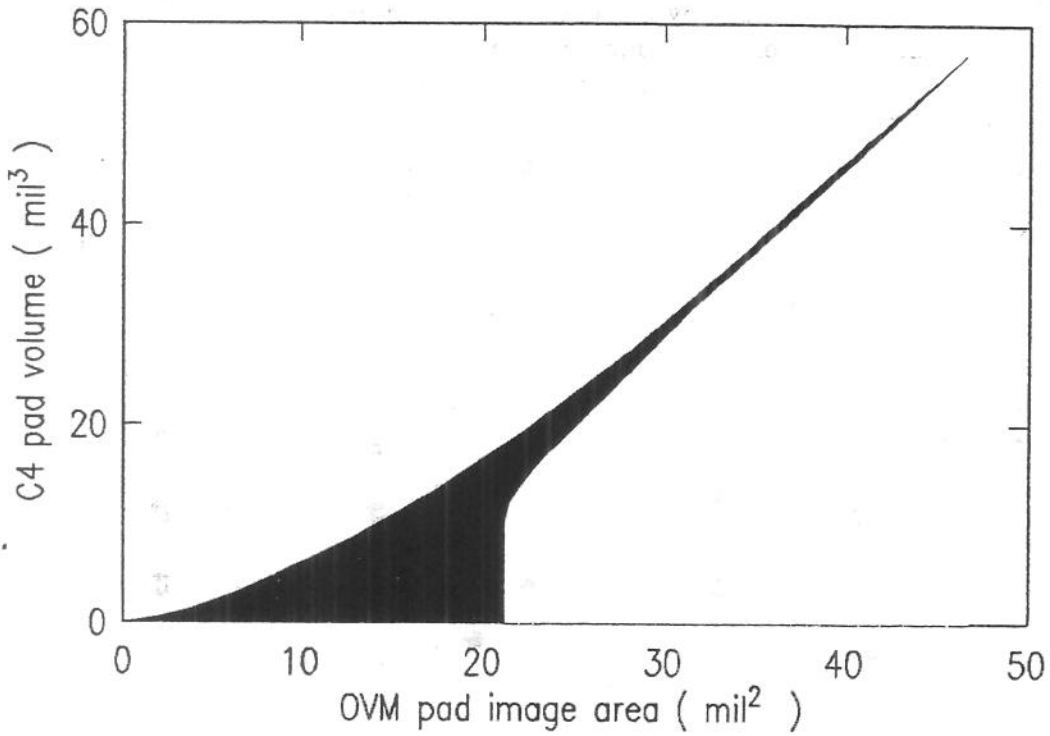


Figure 6. Bipolar chip imaged with the Oblique Viewing Microscope

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



For partial pads based on a normal pad with 5.2 mil BLM diameter and 4.4 mil unreflowed height.

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 algorithm¹⁹ 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

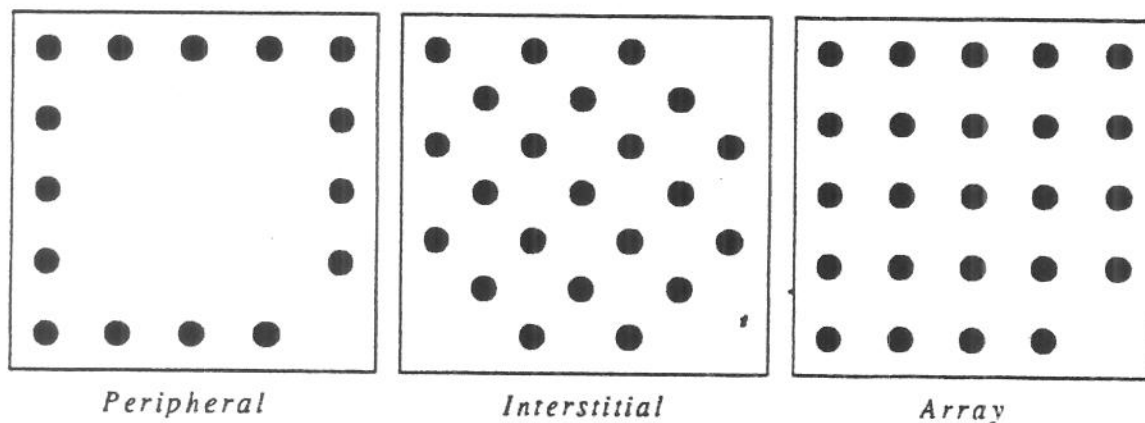


Figure 8. Sample C4 footprints

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 chip²⁴.

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.

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.

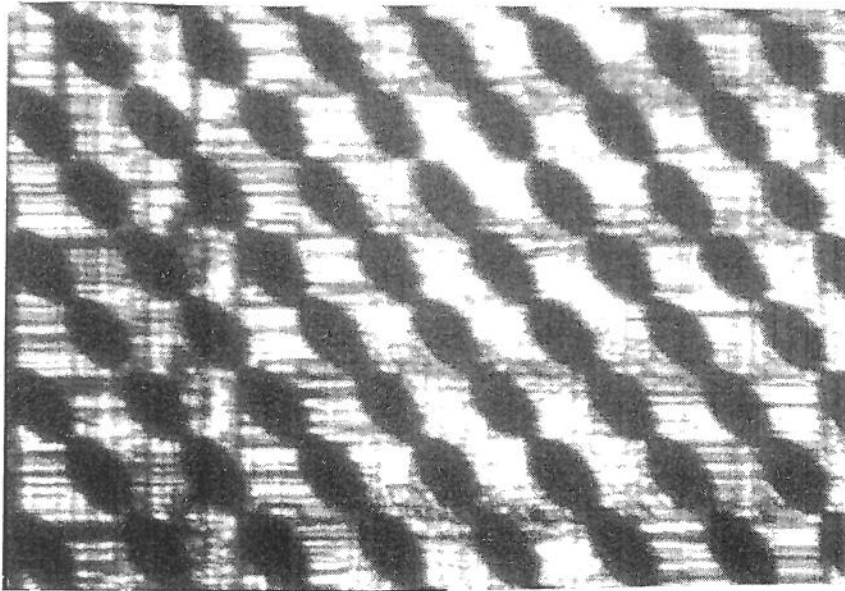


Figure 9. Product with overlapping pad images

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

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

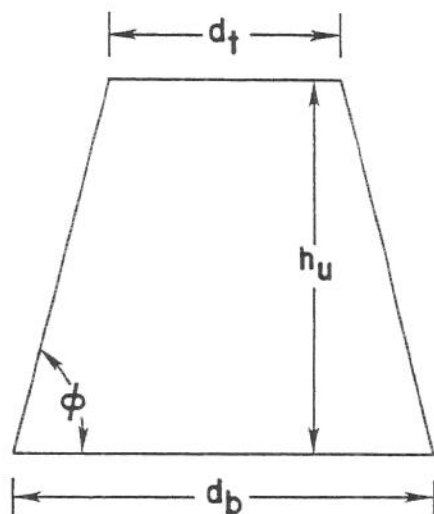


Figure 10. C4 solder ball post deposition but prior to reflow

The range of values for d_b and h_u are both called out in the Terminal Metals Specification²². 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

$$d_t = d_b - \frac{2h_u}{\tan \phi} \quad (3)$$

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

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

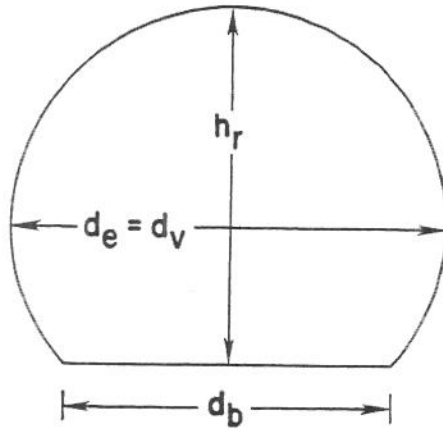


Figure 11. C4 solder ball after reflow

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,

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

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

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

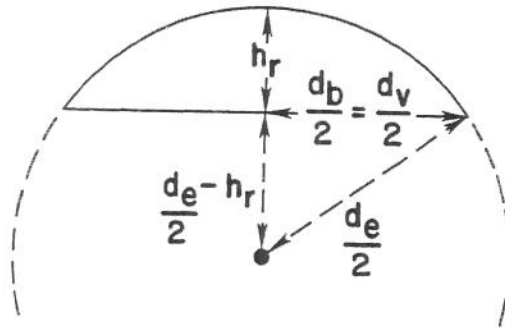


Figure 12. C4 solder ball after reflow showing relationship between d_b , d_e , and h_r .

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 :

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

and

$$A = \frac{\pi \left(\frac{d_e}{2}\right)^2}{\cos \theta} + 2 \left(h_r - \frac{d_e}{2}\right) \tan \theta \left[\frac{d_e^2}{4} - \left(h_r - \frac{d_e}{2}\right)^2 \sin^2 \theta \right]^{1/2} + \frac{d_e^2}{2 \cos \theta} \text{Arcsin} \left[\frac{2 \left(h_r - \frac{d_e}{2}\right) \sin \theta}{d_e} \right] \quad (8)$$

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

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

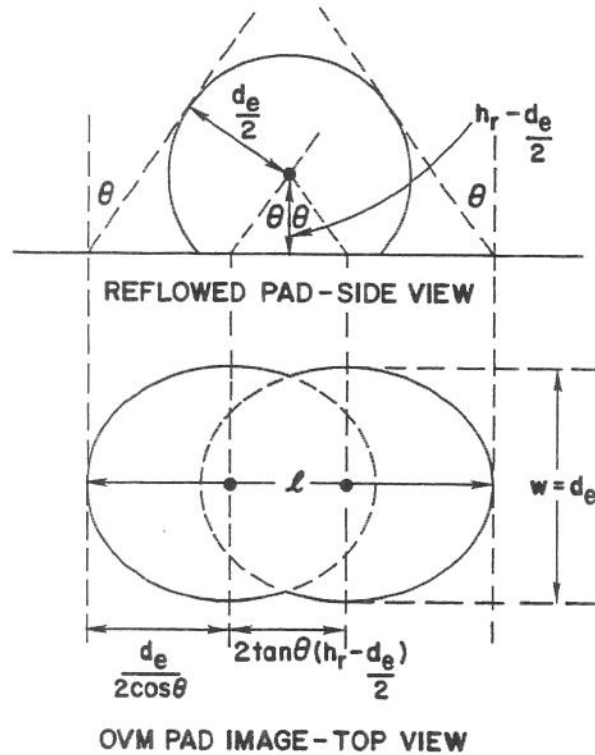


Figure 13. Pad and pad image for solder ball greater than hemisphere

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

$$w = d_b = d_v,$$

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

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

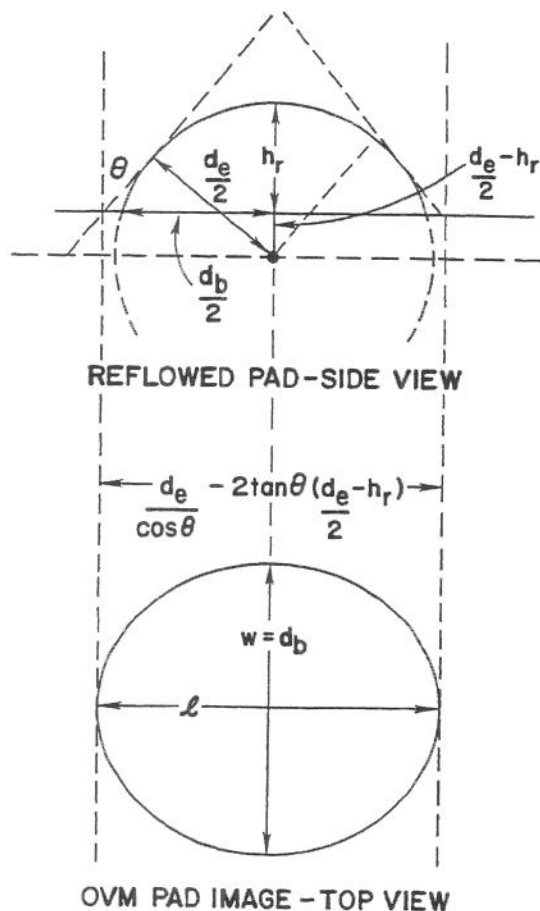


Figure 14. Pad and pad image for solder ball less than hemisphere

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

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

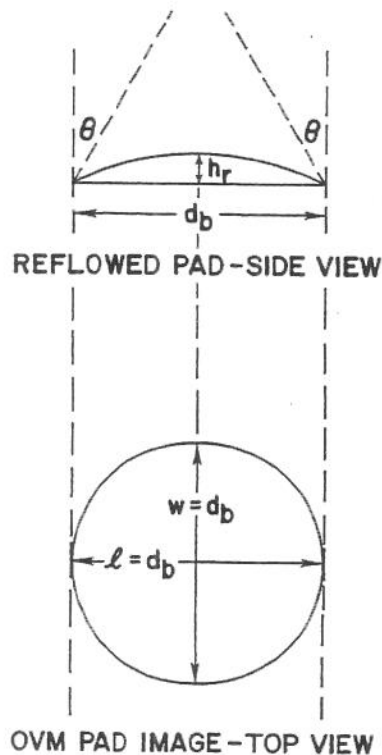


Figure 15. Pad and pad image for solder ball which does not cast a shadow

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

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.
- 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.
 - v : volume_mode - This mode determines which method is used to determine 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 specification²². 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 :

INPUT DATA FOR BLM DIAMETER AND
UNREFLOWED HEIGHT

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 lngh	reflow volume	image area	max diamt	max lngh
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 $y = h_u$.

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

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

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

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

(1)

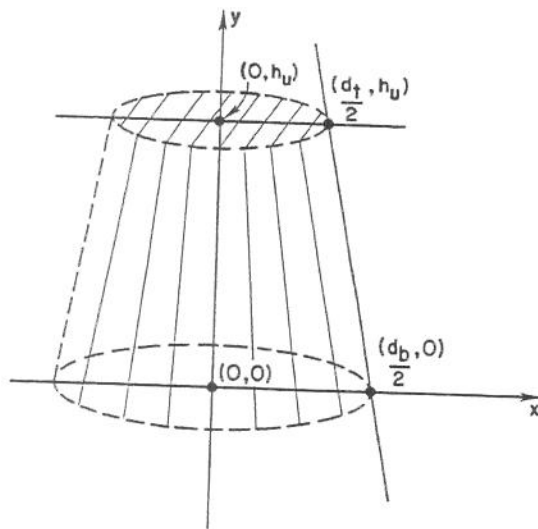


Figure 16. C4 solder ball post deposition and prior to reflow

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 $y = \frac{d_e}{2}$.

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

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

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

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

(4)

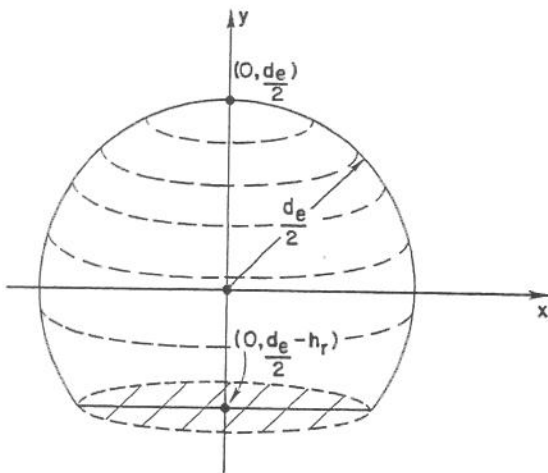


Figure 17. C4 solder ball after reflow

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 $x = -\tan \theta \left(h_r - \frac{d_e}{2}\right)$, and $x = \frac{d_e}{2 \cos \theta}$, and then multiplying by 4.

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

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

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

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

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

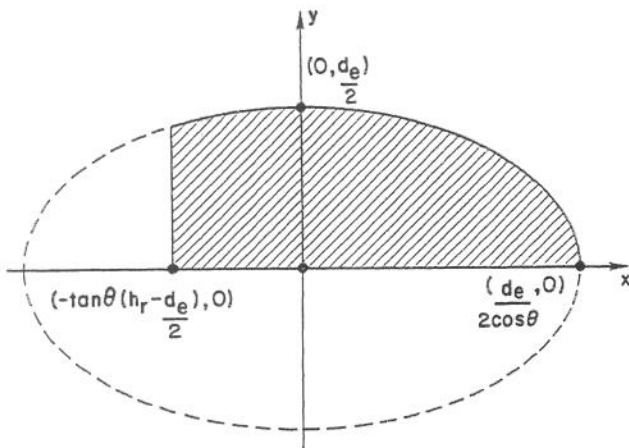


Figure 18. C4 solder ball as imaged by the OVM

Appendix B

```
1  /*****
2  /*                                     */
3  /*                                     */
4  /* C4_PARAM                           */
5  /*****
6  /*                                     */
7  /* Function : Determines parameters for unreflowed and reflowed C4 pad
8  /*           and parameters associated with the OVM pad image.
9  /*
10 /* Author   : K. B. Kirtley
11 /* Date    : 3/22/92
12 /*
13 /* RELEASE 1.0
14 /*
15 /* 03/30/93 : Added version number to main screen.
16 /* 03/19/93 : Added print statements to deal with situations in which
17 /*           entered parameters correspond to a pad that cannot
18 /*           physically exist (such as greater than a sphere).
19 /*
20 /* RELEASE 0.0 Pre-release version
21 /*
22 /* 02/10/93 : Fixed bug in unreflowed pad shape. Trap on top diameter
23 /*           less than zero.
24 /* 11/10/92 : Added ability to convert pad image parameters (length
25 /*           and width) to truncated sphere parameters
26 /* 11/04/92 : Added to ability to analyze truncated spheres with
27 /*           equal equatorial diameters but changing reflowed heights
28 /* 10/29/92 : Added ability to call functions from menu
29 /* 10/19/92 : Added area of OVM pad image
30 /*
31 /*****
32
33 /*****
34 /* INCLUDE FILES
35 /*****
36
37 #include <conio.h>
38 #include <graph.h>
39 #include <stdio.h>
40 #include <math.h>
41
42
43 /*****
44 /* # DEFINES
45 /*****
46
47 #define PI          3.14159
48
49 #define OFF         0
50 #define ON          1
51
52 #define TM_SPEC     0
53 #define SLOPE       1
54
55 #define UNREFLOWED  0
56 #define REFLOWED    1
57
58
59 /*****
60 /* GLOBAL VARIABLES
61 /*****
62
63 int file_mode;          /* 0 : don't write results to file */
64                         /* 1 : write results to file      */
65 FILE *out;             /*
66 char name [40];        /* DOS output file name
67
68 int volume_mode;       /* 0 : formula in specification
69                         /* top_BLM_diameter = BLM_diameter
70                         /* (1 - .075 * unreflowed_height)
71                         /* 1 : top_BLM_diameter = BLM_diameter */
```

```

72                                     /* - 2 * unreflowed height / tan */
73                                     /* (slope_angle) */
74 float slope_angle;                  /* interior slope angle for unreflowed */
75                                     /* pad */
76 float ovm_angle;                    /* OVM angle */
77
78 float padspacing;                   /* spacing between pads */
79
80 float cos_ovm, sin_ovm, tan_ovm;    /* sin, cos, and tan of ovm_angle */
81 float sqrt2;                        /* square root of 2 */
82
83 float rflw_volume;                  /* volume of pad calculated using */
84                                     /* reflowed pad parameters */
85 float BLM_diameter;                 /* BLM diameter */
86 float viewed_diameter;              /* reflowed viewed_diameter */
87 float equatorial_diameter;         /* diameter of sphere. if pad > than */
88                                     /* hemisphere, this will be the viewed */
89                                     /* diameter. if pad < hemisphere, */
90                                     /* this is spherical diameter as if */
91                                     /* pad were greater than hemisphere. */
92 float uflw_height;                  /* unreflowed height */
93 float rflw_height;                  /* reflowed height */
94 float pad_image_area;               /* area of ovm pad shadow image */
95
96 float pad_ratio_sphere;             /*
97 float pad_ratio_hemisphere;         /*
98 float pad_ratio_limit;              /* (1 - sin (theta)) / cos (theta) */
99                                     /* if reflowed height / viewed diam- */
100                                     /* eter / 2.0 is less than this value, */
101                                     /* ovm image is a circle and height is */
102                                     /* not known */
103 float shadow_ratio_sphere;          /*
104 float shadow_ratio_hemisphere;      /*
105 float shadow_ratio_limit;           /*
106
107 float shadow_length;                 /*
108 float max_shadow_length;            /*
109 float max_shadow_diameter;          /*
110
111 int print_index;                     /*
112
113
114 /*****
115 /* EXTERNAL FUNCTIONS */
116 /*****
117
118 int main (void);
119 int display_param (void);
120 int RefreshMenu (void);
121
122 int u_blm_uflw (void);
123 int r_blm_rflw (void);
124 int r_eqdm_rflw (void);
125 int r_viewed_rflw (void);
126 int r_pad_image (void);
127
128 int print_header (int);
129 int fprint_header (int);
130 int print_pad_parameters (int);
131 int fprint_pad_parameters (int);
132
133 int find_BLM_from_eqdiameter_rheight (void);
134 int find_eqdiameter_from_BLM_rheight (void);
135 int find_vdiameter_from_eqdiameter_rheight (void);
136
137 int find_shadow_length (void);
138 int find_rheight_from_slength_eqdiameter (void);
139 int find_eqdiameter_from_slength_rheight (void);
140
141 int find_rflw_volume (void);
142 int find_pad_image_area (void);
143
144
145 /*****
146 /* C4 PARAM PROGRAM */
147 /*****
148

```

```

149 main ()
150 {
151     extern int volume_mode;
152     extern float slope_angle;
153     extern float ovm_angle;
154     extern float padspacing;
155     extern float cos_ovm, sin_ovm, tan_ovm;
156     extern float sqrt2;
157     extern float max_shadow_length;
158     extern float max_shadow_diameter;
159
160     char response;
161
162     /******
163     /* Put up first screen */
164     /******
165
166     _setvideomode (_TEXT80);
167     _clearscreen (0);
168
169     _settextposition (6,28);
170     printf ("--- C 4 _ P A R A M ---");
171     _settextposition (8,34);
172     printf ("Version 1.0");
173     _settextposition (10,15);
174     printf ("Program for calculating C4 solder ball parameters");
175     _settextposition (22,33);
176     printf ("K. B. Kirtley");
177     _settextposition (23,18);
178     printf ("IBM Research Division - Yorktown Heights, NY");
179     _settextposition (24,26);
180     printf ("(C) Copyright IBM Corp 1993");
181     _settextposition (15,28);
182     printf ("Hit any key to continue ");
183
184     getch ();
185
186
187     /******
188     /* default parameters */
189     /******
190
191     ovm_angle = (float) 51.66;
192     volume_mode = 1;
193     slope_angle = (float) 75.0;
194     padspacing = (float) (200.0 / 25.4);
195     file_mode = OFF;
196
197     sqrt2 = (float) sqrt ((double) 2.0);
198
199
200     /******
201     /* trig functions */
202     /******
203
204     cos_ovm = (float) cos ((double) (ovm_angle * (float) PI / (float) 180.0));
205     sin_ovm = (float) sin ((double) (ovm_angle * (float) PI / (float) 180.0));
206     tan_ovm = (float) tan ((double) (ovm_angle * (float) PI / (float) 180.0));
207
208     pad_ratio_limit = ((float) 1.0 - sin_ovm) / (float) 2.0 / cos_ovm;
209     pad_ratio_hemisphere = (float) .5;
210     pad_ratio_sphere = (float) 1.0;
211
212     shadow_ratio_limit = (float) 1.0;
213     shadow_ratio_hemisphere = cos_ovm;
214     shadow_ratio_sphere = cos_ovm / ((float) 1.0 + sin_ovm);
215
216
217     /******
218     /* calculate maximum length and width of pad image shadow before */
219     /* overlap between images */
220     /******
221
222     max_shadow_length = sqrt2 * padspacing;
223     max_shadow_diameter = max_shadow_length / (float) 2.0;
224
225

```

```

226 /*****
227 /* Put up C4 PARAM FUNCTION MENU and service request */
228 /*****
229
230 do
231 {
232     _setvideomode (_TEXT80);
233     _clearscreen (0);
234
235     RefreshMenu ();
236
237     response = (char) getche ();
238
239     switch (response)
240     {
241     case 'b' :
242         r_blm_rflw ();
243         break;
244
245     case 'd' :
246         display_param ();
247         break;
248
249     case 'e' :
250         break;
251
252     case 'i' :
253         r_pad_image ();
254         break;
255
256     case 'q' :
257         r_eqdm_rflw ();
258         break;
259
260     case 'u' :
261         u_blm_uflw ();
262         break;
263
264     case 'v' :
265         r_viewed_rflw ();
266         break;
267
268     default :
269         break;
270     }
271 } while (response != 'e');
272
273 return (0);
274 }
275
276
277 /*****
278 /* Print a new copy of the command menu */
279 /*****
280
281 RefreshMenu ()
282 {
283     _clearscreen (0);
284
285     _settextposition (2,18);
286     printf ("C 4 _ P A R A M   F U N C T I O N   M E N U");
287     printf ("\n\n");
288     printf ("SYSTEM PARAMETERS\n\n");
289     printf (" d : display/change current system parameters\n\n");
290     printf ("CALCULATE REFLOWED SOLDER BALL PARAMETERS ");
291     printf ("FROM THE FOLLOWING INITIAL CONDITIONS\n\n");
292     printf (" u : unreflowed pad - BLM diameter and unreflowed height\n\n");
293     printf (" b : reflowed pad - BLM diameter and reflowed height\n\n");
294     printf (" q : reflowed pad - equatorial diameter and reflowed height\n\n");
295     printf (" v : reflowed pad - viewed diameter and reflowed height\n\n");
296     printf (" i : reflowed pad - OVH pad image length and width\n\n");
297     printf (" e : EXIT program\n\n");
298     printf ("\n\n");
299     printf ("Enter choice : ");
300
301     return (0);
302 }

```



```

303
304
305 /*****
306 /* display parameters */
307 /*****
308
309 display_param ()
310 {
311     extern int file_mode;
312     extern int volume_mode;
313     extern float slope_angle;
314     extern float ovm_angle;
315     extern float padspacing;
316     extern float max_shadow_length;
317     extern float max_shadow_diameter;
318     extern float sqrt2;
319
320     char response;
321
322     do
323     {
324         _clearscreen (0);
325
326         _settextposition (2,24);
327         printf ("S Y S T E M P A R A M E T E R S");
328         _settextposition (5,14);
329         printf ("o : OVM angle : %5.2f degrees\n", ovm_angle);
330         _settextposition (6,14);
331         if (volume_mode == TM_SPEC)
332         {
333             printf ("v : volume_mode : terminal metals spec\n");
334         }
335
336         if (volume_mode == SLOPE)
337         {
338             printf ("v : volume_mode : slope angle\n");
339             _settextposition (7,14);
340             printf ("a : angle : %5.2f degrees\n", slope_angle);
341         }
342         _settextposition (8,14);
343         printf ("p : padspacing : %5.2f mils = %6.2f microns\n",
344             padspacing, padspacing * (float) 25.4);
345
346         _settextposition (9,14);
347         if (file_mode == OFF) printf ("f : file_mode : 0 - will not write output to disk");
348         if (file_mode == ON) printf ("f : file_mode : 1 - will write output to disk");
349
350         _settextposition (11,14);
351         printf ("e : EXIT this screen");
352
353         _settextposition (16,17);
354         printf ("Enter character to change system parameter : ");
355         response = (char) getche ();
356
357         switch (response)
358         {
359             case 'o' :
360                 do
361                 {
362                     _settextposition (18,17);
363                     printf ("Enter new OVM angle (>= 30 & <= 90 degrees) : ");
364                     scanf ("%f", &ovm_angle);
365                 } while (ovm_angle < (float) 30.0 || ovm_angle > (float) 90.0);
366
367                 /*****
368                 /* recalculate trig functions for new OVM angle */
369                 /*****
370
371                 cos_ovm = (float) cos ((double) (ovm_angle * (float) PI /
372                     (float) 180.0));
373                 sin_ovm = (float) sin ((double) (ovm_angle * (float) PI /
374                     (float) 180.0));
375                 tan_ovm = (float) tan ((double) (ovm_angle * (float) PI /
376                     (float) 180.0));
377                 pad_ratio_limit = ((float) 1.0 - sin_ovm) / (float) 2.0 / cos_ovm;
378
379                 break;

```

```

380
381 case 'v' :
382     do
383     {
384         _settextposition (18,15);
385         printf ("Enter %d for terminal spec or %d for slope angle : ",
386             TM_SPEC, SLOPE);
387         scanf ("%d", &volume_mode);
388     } while (volume_mode != TM_SPEC && volume_mode != SLOPE);
389
390     break;
391
392 case 'a' :
393     do
394     {
395         _settextposition (18,5);
396         printf ("Enter new angle for slope of unreflowed pad ");
397         printf ("(>= 45 & <= 90 degrees) : ");
398         scanf ("%f", &slope_angle);
399     } while (slope_angle < (float) 45.0 || slope_angle > (float) 90.0);
400
401     break;
402
403 case 'p' :
404     do
405     {
406         _settextposition (18,18);
407         printf ("Enter new pad spacing (25 - 1000 microns) : ");
408         scanf ("%f", &padspacing);
409     } while (padspacing < (float) 25 || padspacing > (float) 1000.0);
410
411     padspacing = padspacing / (float) 25.4;
412
413     /*
414     /* calculate maximum length and width of pad image shadow
415     /* before overlap between images
416     /*
417     /*
418     max_shadow_length = sqrt2 * padspacing;
419     max_shadow_diameter = max_shadow_length / (float) 2.0;
420
421     break;
422
423 case 'f' :
424     do
425     {
426         _settextposition (18,10);
427         printf ("Enter 0 to suppress writing to disk ");
428         printf ("or 1 to write to disk : ");
429         scanf ("%d", &file_mode);
430     } while (file_mode != OFF && file_mode != ON);
431
432     break;
433
434 case 'e' :
435     break;
436
437 default :
438     break;
439 }
440
441 } while (response != 'e');
442
443 return (0);
444 }
445
446
447 /*
448 /* unreflowed height and BLM diameter
449 /*
450
451 u_blm_uflw ()
452 {
453     extern int file_mode;
454     extern FILE *out;
455
456     extern float BLM_diameter;

```

```

457 extern float uflw_height;
458 extern float rflw_height;
459
460 float BLM_diameter_small; /* BLM diameter - smallest value */
461 float BLM_diameter_big; /* BLM diameter - largest value */
462 float BLM_diameter_increment; /* BLM diameter - increment */
463
464 float uflw_top_diameter; /* diameter of top of unreflowed pad */
465
466 float uflw_height_small; /* unreflowed height - smallest value */
467 float uflw_height_big; /* unreflowed height - largest value */
468 float uflw_height_increment; /* unreflowed height - increment */
469
470 float uflw_volume; /* volume of pad calculated using */
471 /* unreflowed pad parameters */
472
473
474 /*****
475 /* Input parameters
476 *****/
477
478 _clearscreen (0);
479 _settextposition (2,12);
480 printf ("INPUT DATA FOR BLM DIAMETER AND");
481 _settextposition (3,24);
482 printf ("UNREFLOWED HEIGHT");
483
484 _settextposition (6,5);
485 printf ("Requesting range of data for BLM diameter");
486
487 do
488 {
489     _settextposition (8,5);
490     printf ("Enter smallest BLM diameter (> 0) in mils : ");
491     scanf ("%f", &BLM_diameter_small);
492 } while (BLM_diameter_small <= (float) 0.0);
493
494 do
495 {
496     _settextposition (9,5);
497     printf ("Enter largest BLM diameter (>= smallest) in mils : ");
498     scanf ("%f", &BLM_diameter_big);
499 } while (BLM_diameter_big < BLM_diameter_small);
500
501 if (BLM_diameter_small == BLM_diameter_big)
502 {
503     BLM_diameter_increment = (float) 1.0;
504 }
505 else
506 {
507     do
508     {
509         _settextposition (10,5);
510         printf ("Enter BLM diameter increment (> 0) in mils : ");
511         scanf ("%f", &BLM_diameter_increment);
512     } while (BLM_diameter_increment <= (float) 0.0);
513
514
515     _settextposition (13,5);
516     printf ("Requesting range of data for unreflowed height");
517
518     do
519     {
520         _settextposition (15,5);
521         printf ("Enter smallest unreflowed height (> 0) in mils : ");
522         scanf ("%f", &uflw_height_small);
523     } while (uflw_height_small <= (float) 0.0);
524
525     do
526     {
527         _settextposition (16,5);
528         printf ("Enter largest unreflowed height (>= smallest) in mils : ");
529         scanf ("%f", &uflw_height_big);
530     } while (uflw_height_big < uflw_height_small);
531
532     if (uflw_height_small == uflw_height_big)
533     {

```

```

534     uflw_height_increment = (float) 1.0;
535 }
536 else
537 {
538     do
539     {
540         _settextposition (17,5);
541         printf ("Enter unreflowed height increment (> 0) in mils      : ");
542         scanf ("%f", &uflw_height_increment);
543     } while (uflw_height_increment <= (float) 0.0);
544 }
545
546
547 /******
548 /* Write header to files                                     */
549 /******
550
551 if (file_mode == ON)
552 {
553     fprintf_header (UNREFLOWED);
554 }
555
556
557 /******
558 /* Write header to terminal                                 */
559 /******
560
561 print_header (UNREFLOWED);
562
563
564 /******
565 /* Process data                                           */
566 /******
567
568
569 for (BLM_diameter = BLM_diameter_small;
570     BLM_diameter <= BLM_diameter_big;
571     BLM_diameter += BLM_diameter_increment)
572 {
573     for (uflw_height = uflw_height_small;
574         uflw_height <= uflw_height_big;
575         uflw_height += uflw_height_increment)
576     {
577
578         /******
579         /* volume of pad based on formula for unreflowed pad    */
580         /******
581
582         if (volume_mode == 0)
583         {
584             /******
585             /* use formula in terminal metals specification    */
586             /******
587
588             uflw_top_diameter = BLM_diameter * ((float) 1.0 - (float)
589                 .075 * uflw_height);
590         }
591         else
592         {
593             /******
594             /* use formula based on interior angle              */
595             /******
596
597             uflw_top_diameter = BLM_diameter - (float) 2.0 *
598                 uflw_height / tan (slope_angle * PI / 180.0);
599         }
600
601         if (uflw_top_diameter >= (float) 0.0)
602         {
603             uflw_volume = (float) PI * uflw_height * (BLM_diameter * BLM_diameter
604                 + BLM_diameter * uflw_top_diameter + uflw_top_diameter *
605                 uflw_top_diameter) / (float) 12.0;
606
607
608             /******
609             /* calculate reflowed parameters from volume and  */
610             /* BLM_diameter                                     */

```

611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687

```
/* 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);
```



```

765 {
766     _settextposition (9,5);
767     printf ("Enter largest equatorial diameter (>= smallest) in mils : ");
768     scanf ("%f", &equatorial_diameter_big);
769 } while (equatorial_diameter_big < equatorial_diameter_small);
770
771 if (equatorial_diameter_small == equatorial_diameter_big)
772 {
773     equatorial_diameter_increment = (float) 1.0;
774 }
775 else
776 {
777     do
778     {
779         _settextposition (10,5);
780         printf ("Enter equatorial diameter increment (> 0) in mils      : ");
781         scanf ("%f", &equatorial_diameter_increment);
782     } while (equatorial_diameter_increment <= (float) 0.0);
783 }
784
785 _settextposition (13,5);
786 printf ("Requesting range of data for reflowed height");
787
788 do
789 {
790     _settextposition (15,5);
791     printf ("Enter smallest reflowed height (> 0) in mils      : ");
792     scanf ("%f", &rflw_height_small);
793 } while (rflw_height_small <= (float) 0.0);
794
795 do
796 {
797     _settextposition (16,5);
798     printf ("Enter largest reflowed height (>= smallest) in mils  : ");
799     scanf ("%f", &rflw_height_big);
800 } while (rflw_height_big < rflw_height_small);
801
802 if (rflw_height_small == rflw_height_big)
803 {
804     rflw_height_increment = (float) 1.0;
805 }
806 else
807 {
808     do
809     {
810         _settextposition (17,5);
811         printf ("Enter reflowed height increment (> 0) in mils      : ");
812         scanf ("%f", &rflw_height_increment);
813     } while (rflw_height_increment <= (float) 0.0);
814 }
815
816
817 /******
818 /* Write header to file
819 /******
820
821 if (file_mode == ON)
822 {
823     fprintf_header (REFLOWED);
824 }
825
826 /******
827 /* Write header to terminal
828 /******
829
830 print_header (REFLOWED);
831
832
833 /******
834 /* Process data
835 /******
836
837 for (equatorial_diameter = equatorial_diameter_small;
838     equatorial_diameter <= equatorial_diameter_big;
839     equatorial_diameter += equatorial_diameter_increment)
840 {
841     for (rflw_height = rflw_height_small;

```

```

842     rflw_height <= rflw_height_big;
843     rflw_height += rflw_height_increment)
844 {
845     if (rflw_height < equatorial_diameter)
846     {
847         /*
848         /* calculate BLM diameter
849         /*
850         /*
851         find_BLM_from_eqdiameter_rheight ();
852
853
854         /*
855         /* calculate viewed diameter
856         /*
857         /*
858         find_vdiameter_from_eqdiameter_rheight ();
859
860
861         /*
862         /* calculate pad volume
863         /*
864         /*
865         find_rflw_volume ();
866
867         /*
868         /* calculate pad shadow length
869         /*
870         /*
871         find_shadow_length ();
872
873         /*
874         /* calculate pad image area
875         /*
876         /*
877         find_pad_image_area ();
878
879
880         /*
881         /* print parameters to terminal
882         /*
883         /*
884         print_pad_parameters (REFLOWED);
885
886
887         /*
888         /* print parameters to file
889         /*
890         /*
891         if (file_mode == ON)
892         {
893             fprintf_pad_parameters (REFLOWED);
894         }
895     }
896     else
897     {
898         printf ("%5.2f ", rflw_height);
899         printf (" ");
900         printf ("%5.2f ", equatorial_diameter);
901         printf ("Pad parameters greater than a sphere\n");
902
903         if (file_mode == ON)
904         {
905             fprintf (out, "%5.2f ", rflw_height);
906             fprintf (out, " ");
907             fprintf (out, "%5.2f ", equatorial_diameter);
908             fprintf (out, "Pad parameters greater than a sphere\n");
909         }
910     }
911 }
912 }
913
914 printf ("\nHit any key to return to main menu");
915 getch ();
916
917 if (file_mode == ON) fclose (out);
918

```



```

919 | return (0);
920 | }
921
922 /*****
923 /* reflowed height and BLM diameter */
924 /*****
925
926 r_blm_rflw ()
927 {
928     extern int file_mode;
929     extern FILE *out;
930
931     extern float BLM_diameter;
932     extern float rflw_height;
933
934     float BLM_diameter_small;      /* BLM diameter - smallest value */
935     float BLM_diameter_big;        /* BLM diameter - largest value */
936     float BLM_diameter_increment;  /* BLM diameter - increment */
937     float rflw_height_small;       /* reflowed height - smallest value */
938     float rflw_height_big;         /* reflowed height - largest value */
939     float rflw_height_increment;   /* reflowed height - increment */
940
941
942     /*****
943     /* Input parameters */
944     /*****
945
946     _clearscreen (0);
947     _settextposition (2,12);
948     printf ("INPUT DATA FOR BLM DIAMETER AND");
949     _settextposition (3,26);
950     printf ("REFLOWED HEIGHT");
951
952     _settextposition (6,5);
953     printf ("Requesting range of data for BLM diameter");
954
955     do
956     {
957         _settextposition (8,5);
958         printf ("Enter smallest BLM diameter (> 0) in mils : ");
959         scanf ("%f", &BLM_diameter_small);
960     } while (BLM_diameter_small <= (float) 0.0);
961
962     do
963     {
964         _settextposition (9,5);
965         printf ("Enter largest BLM diameter (>= smallest) in mils : ");
966         scanf ("%f", &BLM_diameter_big);
967     } while (BLM_diameter_big < BLM_diameter_small);
968
969     if (BLM_diameter_small == BLM_diameter_big)
970     {
971         BLM_diameter_increment = (float) 1.0;
972     }
973     else
974     {
975         do
976         {
977             _settextposition (10,5);
978             printf ("Enter BLM diameter increment (> 0) in mils : ");
979             scanf ("%f", &BLM_diameter_increment);
980         } while (BLM_diameter_increment <= (float) 0.0);
981     }
982
983     _settextposition (13,5);
984     printf ("Requesting range of data for reflowed height");
985
986     do
987     {
988         _settextposition (15,5);
989         printf ("Enter smallest reflowed height (> 0) in mils : ");
990         scanf ("%f", &rflw_height_small);
991     } while (rflw_height_small <= (float) 0.0);
992
993     do
994     {
995         _settextposition (16,5);

```

```

996     printf ("Enter largest reflowed height (>= smallest) in mils : ");
997     scanf ("%f", &rflw_height_big);
998 } while (rflw_height_big < rflw_height_small);
999
1000     if (rflw_height_small == rflw_height_big)
1001     {
1002         rflw_height_increment = (float) 1.0;
1003     }
1004     else
1005     {
1006         do
1007         {
1008             _settextposition (17,5);
1009             printf ("Enter reflowed height increment (> 0) in mils      : ");
1010             scanf ("%f", &rflw_height_increment);
1011         } while (rflw_height_increment <= (float) 0.0);
1012     }
1013
1014     /******
1015     /* Write header to files
1016     /******
1017
1018     if (file_mode == ON)
1019     {
1020         fprintf_header (REFLOWED);
1021     }
1022
1023     /******
1024     /* Write header to terminal
1025     /******
1026
1027     print_header (REFLOWED);
1028
1029
1030     /******
1031     /* Process data
1032     /******
1033
1034     for (BLM_diameter = BLM_diameter_small;
1035          BLM_diameter <= BLM_diameter_big;
1036          BLM_diameter += BLM_diameter_increment)
1037     {
1038         for (rflw_height = rflw_height_small;
1039              rflw_height <= rflw_height_big;
1040              rflw_height += rflw_height_increment)
1041         {
1042             /******
1043             /* calculate equatorial diameter
1044             /******
1045
1046             find_eqdiameter_from_BLM_rheight ();
1047
1048
1049             /******
1050             /* calculate viewed diameter
1051             /******
1052
1053             find_vdiameter_from_eqdiameter_rheight ();
1054
1055             /******
1056             /* calculate pad volume
1057             /******
1058
1059             find_rflw_volume ();
1060
1061
1062             /******
1063             /* calculate pad shadow length
1064             /******
1065
1066             find_shadow_length ();
1067
1068
1069             /******
1070             /* calculate pad image area
1071             /******
1072

```

```

1073 | | find_pad_image_area ();
1074 | |
1075 | |
1076 | | /*****
1077 | | /* Print values */
1078 | | /*****
1079 | |
1080 | | print_pad_parameters (REFLOWED);
1081 | |
1082 | | if (file_mode == ON)
1083 | | {
1084 | | { fprint_pad_parameters (REFLOWED);
1085 | | }
1086 | | }
1087 | | }
1088 | |
1089 | | printf ("\nHit any key to return to main menu");
1090 | | getch ();
1091 | |
1092 | | if (file_mode == ON) fclose (out);
1093 | |
1094 | | return (0);
1095 | | }
1096 | |
1097 | | /*****
1098 | | /* reflowed height and reflowed diameter */
1099 | | /*****
1100 | |
1101 | | r_viewed_rflw ()
1102 | | {
1103 | |     extern int file_mode;
1104 | |     extern FILE *out;
1105 | |
1106 | |     extern float BLM_diameter;
1107 | |     extern float equatorial_diameter;
1108 | |     extern float viewed_diameter;
1109 | |     extern float rflw_height;
1110 | |
1111 | |     float viewed_diameter_small; /* reflowed v_diam - smallest value */
1112 | |     float viewed_diameter_big; /* reflowed v_diam - largest value */
1113 | |     float viewed_diameter_increment; /* reflowed v_diam - increment */
1114 | |     float rflw_height_small; /* reflowed height - smallest value */
1115 | |     float rflw_height_big; /* reflowed height - largest value */
1116 | |     float rflw_height_increment; /* reflowed height - increment */
1117 | |
1118 | |
1119 | |     /*****
1120 | |     /* Input parameters */
1121 | |     /*****
1122 | |
1123 | |     _clearscreen (0);
1124 | |     _settextposition (2,9);
1125 | |     printf ("INPUT DATA FOR VIEWED DIAMETER AND");
1126 | |     _settextposition (3,26);
1127 | |     printf ("REFLOWED HEIGHT");
1128 | |
1129 | |     _settextposition (6,5);
1130 | |     printf ("Requesting range of data for viewed diameter");
1131 | |
1132 | |     do
1133 | |     {
1134 | |         _settextposition (8,5);
1135 | |         printf ("Enter smallest viewed diameter (> 0) in mils : ");
1136 | |         scanf ("%f", &viewed_diameter_small);
1137 | |     } while (viewed_diameter_small <= (float) 0.0);
1138 | |
1139 | |     do
1140 | |     {
1141 | |         _settextposition (9,5);
1142 | |         printf ("Enter largest viewed diameter (>= smallest) in mils : ");
1143 | |         scanf ("%f", &viewed_diameter_big);
1144 | |     } while (viewed_diameter_big < viewed_diameter_small);
1145 | |
1146 | |     if (viewed_diameter_small == viewed_diameter_big)
1147 | |     {
1148 | |         viewed_diameter_increment = (float) 1.0;
1149 | |     }

```

```

1150     else
1151     {
1152     do
1153     {
1154         _settextposition (10,5);
1155         printf ("Enter viewed diameter increment (> 0) in mils      : ");
1156         scanf ("%f", &viewed_diameter_increment);
1157     } while (viewed_diameter_increment <= (float) 0.0);
1158     }
1159
1160     _settextposition (13,5);
1161     printf ("Requesting range of data for reflowed height");
1162
1163     do
1164     {
1165         _settextposition (15,5);
1166         printf ("Enter smallest reflowed height (> 0) in mils      : ");
1167         scanf ("%f", &rflw_height_small);
1168     } while (rflw_height_small <= (float) 0.0);
1169
1170     do
1171     {
1172         _settextposition (16,5);
1173         printf ("Enter largest reflowed height (>= smallest) in mils : ");
1174         scanf ("%f", &rflw_height_big);
1175     } while (rflw_height_big < rflw_height_small);
1176
1177     if (rflw_height_small == rflw_height_big)
1178     {
1179         rflw_height_increment = (float) 1.0;
1180     }
1181     else
1182     {
1183     do
1184     {
1185         _settextposition (17,5);
1186         printf ("Enter reflowed height increment (> 0) in mils      : ");
1187         scanf ("%f", &rflw_height_increment);
1188     } while (rflw_height_increment <= (float) 0.0);
1189     }
1190
1191
1192     /******
1193     /* Write header to files
1194     /******
1195
1196     if (file_mode == ON)
1197     {
1198         fprintf_header (REFLOWED);
1199     }
1200
1201     /******
1202     /* Write header to terminal
1203     /******
1204
1205     print_header (REFLOWED);
1206
1207
1208     /******
1209     /* Process data
1210     /******
1211
1212     for (viewed_diameter = viewed_diameter_small;
1213         viewed_diameter <= viewed_diameter_big;
1214         viewed_diameter += viewed_diameter_increment)
1215     {
1216         for (rflw_height = rflw_height_small;
1217             rflw_height <= rflw_height_big;
1218             rflw_height += rflw_height_increment)
1219         {
1220             /******
1221             /* check that dimensions are valid for sphere model
1222             /******
1223
1224             if (rflw_height < viewed_diameter)
1225             {
1226                 /******

```

```

1227      /* calculate volume checking for < hemisphere condition          */
1228      /******
1229
1230      if (rflw_height >= viewed_diameter / (float) 2.0)
1231      {
1232          equatorial_diameter = viewed_diameter;
1233          find_BLM_from_eqdiameter_rheight ();
1234      }
1235      else
1236      {
1237          BLM_diameter = viewed_diameter;
1238          find_eqdiameter_from_BLM_rheight ();
1239      }
1240
1241
1242      /******
1243      /* calculate pad volume                                          */
1244      /******
1245
1246      find_rflw_volume ();
1247
1248
1249      /******
1250      /* calculate pad shadow length                                  */
1251      /******
1252
1253      find_shadow_length ();
1254
1255
1256      /******
1257      /* calculate pad image area                                    */
1258      /******
1259
1260      find_pad_image_area ();
1261
1262
1263      /******
1264      /* Print values                                                */
1265      /******
1266
1267      print_pad_parameters (REFLOWED);
1268
1269
1270      /******
1271      /* Print values to file                                        */
1272      /******
1273
1274      if (file_mode == ON)
1275      {
1276          fprintf_pad_parameters (REFLOWED);
1277      }
1278      else
1279      {
1280          printf ("%5.2f ", rflw_height);
1281          printf (" ");
1282          printf (" ");
1283          printf ("%5.2f ", viewed_diameter);
1284          printf ("Pad parameters greater than a sphere\n");
1285
1286          if (file_mode == ON)
1287          {
1288              fprintf (out, "%5.2f ", rflw_height);
1289              fprintf (out, " ");
1290              fprintf (out, " ");
1291              fprintf (out, "%5.2f ", viewed_diameter);
1292              fprintf (out, "Pad parameters greater than a sphere\n");
1293          }
1294      }
1295      }
1296      }
1297      }
1298
1299      printf ("\nHit any key to return to main menu");
1300      getch ();
1301
1302      if (file_mode == ON) fclose (out);
1303

```

```

1304 } return (0);
1305 }
1306
1307
1308 /*****
1309  */
1310 /*****
1311  */
1312 r_pad_image ()
1313 {
1314     extern int file_mode;
1315     extern FILE *out;
1316
1317     extern float BLM_diameter;
1318     extern float viewed_diameter;
1319     extern float equatorial_diameter;
1320     extern float rflw_height;
1321
1322     float viewed_diameter_small;    /* reflowed v_diam - smallest value */
1323     float viewed_diameter_big;      /* reflowed v_diam - largest value */
1324     float viewed_diameter_increment; /* reflowed v_diam - increment */
1325     float shadow_length_small;
1326     float shadow_length_big;
1327     float shadow_length_increment;
1328
1329     float shadow_ratio;
1330
1331
1332     /*****
1333     */
1334     /*****
1335     */
1336     _clearscreen (0);
1337     _settextposition (2,9);
1338     printf ("INPUT DATA FOR OVM IMAGE");
1339
1340     _settextposition (6,5);
1341     printf ("Requesting range of data for OVM pad image diameter");
1342
1343     do
1344     {
1345         _settextposition (8,5);
1346         printf ("Enter smallest OVM pad image diameter (> 0) in mils : ");
1347         scanf ("%f", &viewed_diameter_small);
1348     } while (viewed_diameter_small <= (float) 0.0);
1349
1350     do
1351     {
1352         _settextposition (9,5);
1353         printf ("Enter largest OVM pad image diameter (>= smallest) in mils : ");
1354         scanf ("%f", &viewed_diameter_big);
1355     } while (viewed_diameter_big < viewed_diameter_small);
1356
1357     if (viewed_diameter_small == viewed_diameter_big)
1358     {
1359         viewed_diameter_increment = (float) 1.0;
1360     }
1361     else
1362     {
1363         do
1364         {
1365             _settextposition (10,5);
1366             printf ("Enter OVM pad image diameter increment (> 0) in mils : ");
1367             scanf ("%f", &viewed_diameter_increment);
1368         } while (viewed_diameter_increment <= (float) 0.0);
1369     }
1370
1371     _settextposition (13,5);
1372     printf ("Requesting range of data for OVM pad image length");
1373
1374     do
1375     {
1376         _settextposition (15,5);
1377         printf ("Enter smallest OVM pad image length (> 0) in mils : ");
1378         scanf ("%f", &shadow_length_small);
1379     } while (shadow_length_small <= (float) 0.0);
1380

```

```

1381 do
1382 {
1383     _settextposition (16,5);
1384     printf ("Enter largest OVM pad image length (>= smallest) in mils : ");
1385     scanf ("%f", &shadow_length_big);
1386 } while (shadow_length_big < shadow_length_small);
1387
1388 if (shadow_length_small == shadow_length_big)
1389 {
1390     shadow_length_increment = (float) 1.0;
1391 }
1392 else
1393 {
1394     do
1395     {
1396         _settextposition (17,5);
1397         printf ("Enter OVM pad image length increment (> 0) in mils : ");
1398         scanf ("%f", &shadow_length_increment);
1399     } while (shadow_length_increment <= (float) 0.0);
1400 }
1401
1402
1403 /******
1404 /* Write header to files
1405 /******
1406
1407 if (file_mode == ON)
1408 {
1409     fprintf_header (REFLOWED);
1410 }
1411
1412 /******
1413 /* Write header to terminal
1414 /******
1415
1416 print_header (REFLOWED);
1417
1418
1419 /******
1420 /* Process data
1421 /******
1422
1423 for (viewed_diameter = viewed_diameter_small;
1424     viewed_diameter <= viewed_diameter_big;
1425     viewed_diameter += viewed_diameter_increment)
1426 {
1427     for (shadow_length = shadow_length_small;
1428         shadow_length <= shadow_length_big;
1429         shadow_length += shadow_length_increment)
1430     {
1431         shadow_ratio = viewed_diameter / shadow_length;
1432
1433         /******
1434         /* Check for valid parameters
1435         /******
1436
1437         if (shadow_ratio > shadow_ratio_limit)
1438         {
1439             printf (" ");
1440             printf (" ");
1441             printf (" ");
1442             printf ("%5.2f ", viewed_diameter);
1443             printf ("%5.2f ", shadow_length);
1444             printf ("Shadow length less than shadow diameter\n");
1445
1446             if (file_mode == ON)
1447             {
1448                 fprintf (out, " ");
1449                 fprintf (out, " ");
1450                 fprintf (out, " ");
1451                 fprintf (out, "%5.2f ", viewed_diameter);
1452                 fprintf (out, "%5.2f ", shadow_length);
1453                 fprintf (out, "Shadow length less than shadow diameter\n");
1454             }
1455         }
1456         else if (shadow_ratio <= shadow_ratio_sphere)
1457         {

```

```

1458     printf ("      ");
1459     printf ("      ");
1460     printf ("      ");
1461     printf ("%5.2f ", viewed_diameter);
1462     printf ("%5.2f ", shadow_length);
1463     printf ("Shadow parameters greater than a sphere\n");
1464
1465     if (file_mode == ON)
1466     {
1467         fprintf (out, "      ");
1468         fprintf (out, "      ");
1469         fprintf (out, "      ");
1470         fprintf (out, "%5.2f ", viewed_diameter);
1471         fprintf (out, "%5.2f ", shadow_length);
1472         fprintf (out, "Shadow parameters greater than sphere\n");
1473     }
1474 }
1475 else
1476 {
1477     /******
1478     /* Check for CIRCLE
1479     /******
1480
1481     if (shadow_ratio == shadow_ratio_limit)
1482     {
1483         BLM_diameter = viewed_diameter;
1484         rflw_height = BLM_diameter * pad_ratio_limit;
1485         find_eqdiameter_from_BLM_rheight ();
1486     }
1487
1488     /******
1489     /* Check for ELLIPSE
1490     /******
1491
1492     else if (shadow_ratio >= shadow_ratio_hemisphere)
1493     {
1494         BLM_diameter = viewed_diameter;
1495         rflw_height = (shadow_length + sqrt ((double) shadow_length *
1496             shadow_length - BLM_diameter * BLM_diameter)) * cos_ovm /
1497             (float) 2.0 / ((float) 1.0 + sin_ovm);
1498         find_eqdiameter_from_BLM_rheight ();
1499     }
1500
1501     /******
1502     /* Must be double ELLIPSE
1503     /******
1504     else
1505     {
1506         equatorial_diameter = viewed_diameter;
1507         find_rheight_from_slengh_eqdiameter ();
1508         find_BLM_from_eqdiameter_rheight ();
1509     }
1510
1511     /******
1512     /* calculate pad volume
1513     /******
1514
1515     find_rflw_volume ();
1516
1517
1518     /******
1519     /* calculate pad image area
1520     /******
1521
1522     find_pad_image_area ();
1523
1524
1525     /******
1526     /* Print values
1527     /******
1528
1529     print_pad_parameters (REFLOWED);
1530
1531
1532     /******
1533     /* Print values to file
1534     /******

```



```

1535 |         |         |         |         |
1536 |         |         |         |         |         |
1537 |         |         |         |         |         |         |
1538 |         |         |         |         |         |         |         |
1539 |         |         |         |         |         |         |         |         |
1540 |         |         |         |         |         |         |         |         |
1541 |         |         |         |         |         |         |         |         |
1542 |         |         |         |         |         |         |         |         |
1543 |         |         |         |         |         |         |         |         |
1544 |         |         |         |         |         |         |         |         |
1545 |         |         |         |         |         |         |         |         |
1546 |         |         |         |         |         |         |         |         |
1547 |         |         |         |         |         |         |         |         |
1548 |         |         |         |         |         |         |         |         |
1549 |         |         |         |         |         |         |         |         |
1550 |         |         |         |         |         |         |         |         |
1551 |         |         |         |         |         |         |         |         |
1552 |         |         |         |         |         |         |         |         |
1553 |         |         |         |         |         |         |         |         |
1554 |         |         |         |         |         |         |         |         |
1555 |         |         |         |         |         |         |         |         |
1556 |         |         |         |         |         |         |         |         |
1557 |         |         |         |         |         |         |         |         |
1558 |         |         |         |         |         |         |         |         |
1559 |         |         |         |         |         |         |         |         |
1560 |         |         |         |         |         |         |         |         |
1561 |         |         |         |         |         |         |         |         |
1562 |         |         |         |         |         |         |         |         |
1563 |         |         |         |         |         |         |         |         |
1564 |         |         |         |         |         |         |         |         |
1565 |         |         |         |         |         |         |         |         |
1566 |         |         |         |         |         |         |         |         |
1567 |         |         |         |         |         |         |         |         |
1568 |         |         |         |         |         |         |         |         |
1569 |         |         |         |         |         |         |         |         |
1570 |         |         |         |         |         |         |         |         |
1571 |         |         |         |         |         |         |         |         |
1572 |         |         |         |         |         |         |         |         |
1573 |         |         |         |         |         |         |         |         |
1574 |         |         |         |         |         |         |         |         |
1575 |         |         |         |         |         |         |         |         |
1576 |         |         |         |         |         |         |         |         |
1577 |         |         |         |         |         |         |         |         |
1578 |         |         |         |         |         |         |         |         |
1579 |         |         |         |         |         |         |         |         |
1580 |         |         |         |         |         |         |         |         |
1581 |         |         |         |         |         |         |         |         |
1582 |         |         |         |         |         |         |         |         |
1583 |         |         |         |         |         |         |         |         |
1584 |         |         |         |         |         |         |         |         |
1585 |         |         |         |         |         |         |         |         |
1586 |         |         |         |         |         |         |         |         |
1587 |         |         |         |         |         |         |         |         |
1588 |         |         |         |         |         |         |         |         |
1589 |         |         |         |         |         |         |         |         |
1590 |         |         |         |         |         |         |         |         |
1591 |         |         |         |         |         |         |         |         |
1592 |         |         |         |         |         |         |         |         |
1593 |         |         |         |         |         |         |         |         |
1594 |         |         |         |         |         |         |         |         |
1595 |         |         |         |         |         |         |         |         |
1596 |         |         |         |         |         |         |         |         |
1597 |         |         |         |         |         |         |         |         |
1598 |         |         |         |         |         |         |         |         |
1599 |         |         |         |         |         |         |         |         |
1600 |         |         |         |         |         |         |         |         |
1601 |         |         |         |         |         |         |         |         |
1602 |         |         |         |         |         |         |         |         |
1603 |         |         |         |         |         |         |         |         |
1604 |         |         |         |         |         |         |         |         |
1605 |         |         |         |         |         |         |         |         |
1606 |         |         |         |         |         |         |         |         |
1607 |         |         |         |         |         |         |         |         |
1608 |         |         |         |         |         |         |         |         |
1609 |         |         |         |         |         |         |         |         |
1610 |         |         |         |         |         |         |         |         |
1611 |         |         |         |         |         |         |         |         |

```

```

1612 /*****/
1613
1614 find_BLM_from_eqdiameter_rheight ()
1615 {
1616     extern float BLM_diameter;
1617     extern float equatorial_diameter;
1618     extern float rflw_height;
1619
1620     BLM_diameter = (float) 2.0 * sqrt ((double) (equatorial_diameter *
1621         rflw_height - rflw_height * rflw_height));
1622
1623     return (0);
1624 }
1625
1626
1627 /*****/
1628 /* Calculate equatorial diameter from BLM_diameter and rflw height */
1629 /*****/
1630
1631 find_eqdiameter_from_BLM_rheight ()
1632 {
1633     extern float equatorial_diameter;
1634     extern float BLM_diameter;
1635     extern float rflw_diameter;
1636
1637     equatorial_diameter = rflw_height + BLM_diameter * BLM_diameter /
1638         (float) 4.0 / rflw_height;
1639
1640     return (0);
1641 }
1642
1643
1644 /*****/
1645 /* Calculate equatorial_diameter from shadow length and rflw height */
1646 /*****/
1647
1648 find_eqdiameter_from_slength_rheight ()
1649 {
1650     extern float shadow_length;
1651     extern float equatorial_diameter;
1652     extern float rflw_height;
1653
1654     equatorial_diameter = (shadow_length / (float) 2.0 - rflw_height * tan_ovm)
1655         / pad_ratio_limit;
1656
1657     return (0);
1658 }
1659
1660
1661 /*****/
1662 /* Calculate rflw_height from equatorial_diameter and shadow length */
1663 /*****/
1664
1665 find_rheight_from_slength_eqdiameter ()
1666 {
1667     extern float rflw_height;
1668     extern float shadow_length;
1669     extern float equatorial_diameter;
1670
1671     rflw_height = (shadow_length / (float) 2.0 - equatorial_diameter *
1672         pad_ratio_limit) / tan_ovm;
1673
1674     return (0);
1675 }
1676
1677
1678 /*****/
1679 /* Calculate reflowed pad volume */
1680 /*****/
1681
1682 find_rflw_volume ()
1683 {
1684     extern float rflw_volume;
1685     extern float BLM_diameter;
1686     extern float rflw_height;
1687
1688     rflw_volume = ((float) 4.0 * rflw_height * rflw_height +

```

```

1689     (float) 3.0 * BLM_diameter * BLM_diameter) * PI *
1690     rflw_height / (float) 24.0;
1691
1692     return (0);
1693 }
1694
1695
1696 /*****
1697  /* Calculate OVH pad shadow length */
1698  *****/
1699
1700 find_shadow_length ()
1701 {
1702     extern float rflw_height;
1703     extern float BLM_diameter;
1704     extern float viewed_diameter;
1705     extern float equatorial_diameter;
1706     extern float shadow_length;
1707     float pad_ratio;
1708
1709
1710     pad_ratio = rflw_height / viewed_diameter;
1711
1712     if (pad_ratio <= pad_ratio_limit)
1713     {
1714         shadow_length = BLM_diameter;
1715     }
1716     else if (pad_ratio < pad_ratio_sphere)
1717     {
1718         shadow_length = (float) 2.0 * (rflw_height * tan_ovm + equatorial_diameter
1719             * pad_ratio_limit);
1720     }
1721     else
1722     {
1723         printf ("\nBall is sphere or larger based on pad_ratio.");
1724         printf ("\nHit any key to return to main menu");
1725         getch ();
1726         return (1);
1727     }
1728
1729     return (0);
1730 }
1731
1732
1733 /*****
1734  /* Calculate viewed_diameter from equatorial_diameter and rflw_height */
1735  *****/
1736
1737 find_vdiameter_from_eqdiameter_rheight ()
1738 {
1739     extern float viewed_diameter;
1740     extern float rflw_height;
1741     extern float equatorial_diameter;
1742
1743     if (rflw_height >= equatorial_diameter / (float) 2.0)
1744     {
1745         viewed_diameter = equatorial_diameter;
1746     }
1747     else
1748     {
1749         viewed_diameter = BLM_diameter;
1750     }
1751
1752     return (0);
1753 }
1754
1755
1756 /*****
1757  /* Write file header to hard disk */
1758  *****/
1759
1760 fprintf_header (int type)
1761 {
1762     extern char name [];
1763
1764     printf ("\n\nEnter name of output image file : ");
1765     scanf ("%s",name);

```

```

1766 |
1767 |   out = fopen (name,"w");
1768 |
1769 |   if (type == UNREFLOWED) fprintf (out, "urflw ");
1770 |   fprintf (out, "reflw BLM  equat viewd shadw reflow image  max  max\n");
1771 |   if (type == UNREFLOWED) fprintf (out, "ht  ");
1772 |   fprintf (out, "ht  diamt diamt diamt lngth volume area  diamt lngth\n\n");
1773 |
1774 |   return (0);
1775 | }
1776 |
1777 |
1778 | /*****
1779 | /*  Print file header to terminal
1780 | /*****
1781 |
1782 | print_header (int type)
1783 | {
1784 |     extern int print_index;          /* index for number of terminal lines */
1785 |
1786 |     print_index = 4;
1787 |
1788 |     _clearscreen (0);
1789 |
1790 |     if (type == UNREFLOWED) printf ("urflw ");
1791 |     printf ("reflw BLM  equat viewd shadw reflow image  max  max\n");
1792 |     if (type == UNREFLOWED) printf ("ht  ");
1793 |     printf ("ht  diamt diamt diamt lngth volume area  diamt lngth\n\n");
1794 |
1795 |     return (0);
1796 | }
1797 |
1798 |
1799 | /*****
1800 | /*  Print pad parameters to terminal
1801 | /*****
1802 |
1803 | print_pad_parameters (int type)
1804 | {
1805 |     extern int print_index;
1806 |
1807 |     extern float uflw_height;
1808 |     extern float rflw_height;
1809 |     extern float BLM_diameter;
1810 |     extern float equatorial_diameter;
1811 |     extern float viewed_diameter;
1812 |     extern float shadow_length;
1813 |     extern float rflw_volume;
1814 |     extern float pad_image_area;
1815 |     extern float max_shadow_diameter;
1816 |     extern float max_shadow_length;
1817 |
1818 |
1819 |     if (print_index == 22)
1820 |     {
1821 |         _settextposition (24, 0);
1822 |         printf ("Hit any key to continue to end of data");
1823 |         getch ();
1824 |         printf ("\n");
1825 |     }
1826 |
1827 |     if (type == UNREFLOWED) printf ("%5.2f ", uflw_height);
1828 |     printf ("%5.2f ", rflw_height);
1829 |     printf ("%5.2f ", BLM_diameter);
1830 |     printf ("%5.2f ", equatorial_diameter);
1831 |     printf ("%5.2f ", viewed_diameter);
1832 |     printf ("%5.2f ", shadow_length);
1833 |     printf ("%6.2f ", rflw_volume);
1834 |     printf ("%6.2f ", pad_image_area);
1835 |     printf ("%5.2f ", max_shadow_diameter);
1836 |     printf ("%5.2f\n", max_shadow_length);
1837 |
1838 |     print_index++;
1839 |
1840 |     return (0);
1841 | }
1842 |

```

```

1843 /*****
1844 /* print pad parameters to file on hard disk */
1845 /*****
1846
1847 fprint_pad_parameters (int type)
1848 {
1849     extern float uflw_height;
1850     extern float rflw_height;
1851     extern float BLM_diameter;
1852     extern float equatorial_diameter;
1853     extern float viewed_diameter;
1854     extern float shadow_length;
1855     extern float rflw_volume;
1856     extern float pad_image_area;
1857     extern float max_shadow_diameter;
1858     extern float max_shadow_length;
1859
1860     if (type == UNREFLOWED) fprintf (out, "%5.2f ", uflw_height);
1861     fprintf (out, "%5.2f ", rflw_height);
1862     fprintf (out, "%5.2f ", BLM_diameter);
1863     fprintf (out, "%5.2f ", equatorial_diameter);
1864     fprintf (out, "%5.2f ", viewed_diameter);
1865     fprintf (out, "%5.2f ", shadow_length);
1866     fprintf (out, "%6.2f ", rflw_volume);
1867     fprintf (out, "%6.2f ", pad_image_area);
1868     fprintf (out, "%5.2f ", max_shadow_diameter);
1869     fprintf (out, "%5.2f\n", max_shadow_length);
1870
1871     return (0);
1872 }

```

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