รายการอ้างอิง

ภาษาไทย

ปรีชา พันธุมสินชัย. พจนานุกรมการบริหารการผลิตและสินค้าคงคลัง. กรุงเทพมหานคร: สำนักพิมพ์มหาวิทยาลัย รังสิต, 2539.

บรรเลง ศรนิล. ตารางงานโลหะ. กรุงเทพมหานคร: สำนักพิมพ์สถาบันเทคโนโลยีพระจอมเกล้าพระนครเหนือ, 2524.

ภาษาอังกฤษ

Anderson, J. R.; Corbett, A. T.; and Reiser, B.J. Essential LISP. Addison Wesley, 1987.

Bangalore, H. M. T. Production Technology. New Delhi: McGraw-Hill, 1980.

Bramley, A. N.; Micham, A. R.; and Owen, G. W. Advances in Manufacturing Technology X : Proceeding of the Twelfth National Conference on Manufacturing Research. Claverton Down, 1996.

Chang, T. C. Expert Process Planning for Manufacturing. Addison-Wesley, 1990.

Edney, R. C. Computer Aided Engineering for Mechnical Engineers. Hertfordshire: Prentice Hall, 1991.

Ethier, S. J.; and Ethier, C. A. AutoCAD for Success. New Jersey: Prentice-Hall, 1996.

Gerner, R., Middlebrook, M.; and Tanzillo, T. Maximizing AutoCAD R13. Autodesk Press, 1997.

Gindy, N.; Ratchev, T.; and Case, K. (1994). Process Capability Models for Equipment Selection. In Advances in Manufacturing Tech VIII :Proceeding of the Tenth National Conference on Manufacturing Research. Case, K.; and Newman, S. (eds.). Taylor & Francis: 244-248.

Halevi, G.; and Weill, R. D. Principle of Process Planning. Chapman & Hall, 1995.

Harkow, R. Essential AutoLISP. New York: Springer-Verlag, 1996.

Head, G. O.; and Head, J. D. 1000 AutoCAD Tips and Tricks. Ventana Press, 1993.

Kusiak, A. Intelligent Manufacturing System. New Jersey: Prentice Hall, 1990.

Linardakis, N.; and Mileham, A. R. (1994). A Strategy for Extracting Manufacturing Features for Prismatic Components from CAD. In Advances in Manufacturing Tech VIII : Proceeding of the Tenth National conference on Manufacturing Research. Case, K.; and Newman, S. (eds.). Taylor & Francis: 299-303.

McMahon, C.; and Browne, J. CAD CAM from principles to practice. Addison-Wesley, 1993

Pande, S.; and Desai, V. Expert CAPP System for Single Spindle Automats. International Journal of Production Research. 33(1995): 819-833.

Parsae, H. R.; and Jamshidi, M. Design and Implementation of Intelligent Manufacturing System. New Jersey: Prentice Hall, 1995

Razfar, R.; and Ridgway, K. (1994). A Knowledge Base System for the Selection of Cutting Tools and Cutting Condition for Milling. In Advances in Manufacturing Tech VIII : Proceeding of the Tenth National Conference on Manufacturing Research. Case, K. ; and Newman, S. (eds.). Taylor & Francis: 259-263.

Ransen, O. AutoCAD Programming in C/C++. West Sussex: John Wiley & sons, 1997

Rumana, N. An Object-Oriented Process Planning System for Rotational Components. Master's Thesis. AIT. Thailand. 1995.

Suh, S.; and Kang, J. Process Planning for Multi-Axis NC Machining of Free Surfaces International Journal of Production Research. 33(1995): 2723-2738.

Vallaponanthan, K. An Object-Oriented Computer-Aided Process Planning System for Prismatic Components. Master'sThesis. AIT. Thailand. 1995

Walsh, D.; Knight, R. L.; and Valaski, W. R. AutoCAD 13 Secrets. IDG Books, 1996.

Wong, T.; and Siu, S. A Knowledge-Based Approach to Automated Machining Process Selection and Sequencing. International Journal of Production Research. 33(1995): 3465-3484.

Zhoa, Y.; Ridgway, K.; and Ho, K. (1994). The Integration of CAD and A Knowledge Base Tool Selection System. In Advances in Manufacturing Tech VIII : Proceeding of the Tenth National Conference on Manufacturing Research. Case, K.,and Newman, S. (eds.). Taylor & Francis: 289-293.

Zhao, Z.; and Baines, R. (1994). Software Design for Sequencing Machining Operations. In Advances in Manufacturing Tech VIII : Proceeding of the Tenth National Conference on Manufacturing Research. Case, K.; and Newman, S. (eds.). Taylor & Francis: 309-313.

Zhang, H.; Huang, S. H.; and Mei, J. Operational Dimensioning and Tolerancing in Process Planning : Setup Planning. International Journal of Production Research (1996). 34: 1841-1858.

ภาคผนวก ก.

ตารางค่าคงที่สำหรับคำนวนค่า a_{min} (Halevi, 1995)

ตารางค่าคงที่สำหรับคำนวนค่า a_{min}

	Twist Drill	Core Drill	Insert Drill	Solid Carbide Drill	Reaming	Boring
1. พิกัดความเผื่อทางด้านขนาด (Dimension Tolerance, mm)	$0.05+0.015D^{0.7}$	$0.03+0.012D^{0.5}$	0.3	0.05	$0.004\ D^{0.6}$	$0.003D^{0.45}$
2. ระดับความเรียบผิวสูงสุด (μm)	12.5	6.25	12.5	12.5	1.575	12.5
3. ระดับความเรียบผิวต่ำสุด (μm)	3.125	1.575	3.125	2.25	0.4	0.8
4. ความร่วมศูนย์ (Straightness, mm)	$0.05+0.0006\left(\frac{L}{D}\right)^2$	0	0.03	0.02	0	0.01
5. ความเที่ยงความกลม (Roundness, mm)	0.1	0.05	0.07	0.04	0.01	0.01
6. ความขนาน (Parallelism, mm)	$0.08+0.0006\left(\frac{L}{D}\right)^2$	0	0.08	0.05	0	0.015
7. ความเที่ยงศูนย์ (Concentricity, mm)	$0.025D^{0.3}$	0	0.05	0.03	0	0.01
8. พิกัดความเผื่อเคลื่อนที่เป็นแนวหมุน (Circular runout, mm)	$0.05D^{0.3}$	0	0.1	0.06	0	0.02
9. Surface Integrity (mm)						
D<6	0.04	0.025	-	0.07	0	0.03
D<10	0.05	0.03	-			
D<18	0.06	0.04	0.06			
D<30	0.08	0.05	0.08			
D>30	0.10	0.06	0.1			
10. พิกัดความเผื่อของตำแหน่ง (Location Tolerance, mm)						
D<6	0.18	0	-	0.05	0	0.01
D<10	0.20		-			
D<18	0.22		0.22			
D<30	0.24		0.24			
D>30	0.28		0.28			
11. ความลึกการตัดเฉือนต่ำสุด	0.3	0.25	0.3	0.3	0.02	0.3

ตารางค่าคงที่สำหรับคำนวนค่า a_{min} (ต่อ)

	Twist Drill	Core Drill	Insert Drill	Solid Carbide Drill	Reaming	Boring
12. Deflection of tool per unit length (mm) D คือ ขนาดเส้นผ่านศูนย์กลาง						
D<6	0.0024	0.002	0	0	0	0
D<10	0.001	0.001				
D<18	0.001	0.001				
D<30	0.001	0.001				
D>30	0.000	0.000				

ก. การคำนวนค่า a_{min}

ค่าa_{min} เป็นค่าความลึกของการตัดเฉือนที่ต้องใช้ในการกำจัดความไม่แน่นอน (Inaccuracies) ของกระบวนการสร้างรูในลำดับก่อนหน้า โดยสามารถคำนวนจากผลรวมของความไม่แน่นอนต่างๆ 6 ตัวด้วยกัน ได้แก่

1. พิกัดความเผื่อทางด้านขนาด (Dimension Tolerance, mm) , I_1
2. ระดับความเรียบผิว (μm), I_2 โดย I_2 สามารถคำนวนได้จากสูตร $I_2 = 4R_a 10^{-3}$
3. Surface integrity (mm), I_3
4. ค่าพิกัดความเผื่อทางด้านรูปร่าง, I_4 การคำนวนจะอาศัยข้อมูลค่าพิกัดความเผื่อทางด้านรูปทรงชนิดต่างๆ ซึ่งได้แก่ ความร่วมศูนย์(II_1), ความเที่ยงความกลม(II_2), ความขนาน(II_3), ความเที่ยงศูนย์(II_4), และค่าพิกัดความเผื่อเคลื่อนที่ในแนวหมุน(II_5) โดยค่าแต่ละค่ามีผลกระทบซึ่งกันและกัน ค่า I_4 สามารถคำนวณได้จากสูตร $I_4 = \sqrt{II_1^2 + II_2^2 + II_3^2 + II_4^2 + II_5^2}$
5. ค่าพิกัดความเผื่อของตำแหน่ง และ Tool Deflection ค่าทั้งสองค่ามีผลกระทบซึ่งกันและกัน ดังนั้นจึงสามารถหาผลรวมโดยวิธีการทางสถิติโดยใช้สูตร $I_5 = \sqrt{II_6^2 + II_7^2}$

ดังนั้น a_{min} สามารถคำนวนจาก

$$a_{min} = I_1 + I_2 + I_3 + I_4 + I_5 + I_6$$

โดย I_6 คือ ค่าความลึกของการตัดเฉือนต่ำสุดของกระบวนการก่อนหน้า

ภาคผนวก ข.

การติดตั้ง และการใช้โปรแกรม

โปรแกรมคอมพิวเตอร์เพื่อช่วยวางแผนกระบวนการผลิตสำหรับกระบวนการสร้างรู และกระบวนการกัด ได้พัฒนาขึ้นโดยใช้โปรแกรมภาษา AutoLISP และทำงานบนโปรแกรม AutoCAD Release 13 สำหรับ Windows 95 โปรแกรมที่พัฒนามีส่วนช่วยผู้วางแผนกระบวนการผลิตในการเลือกชนิดกระบวนการผลิต เลือกเครื่องมือตัด และการกำหนดค่าสภาวะการตัดเฉือนและประมาณเวลาการตัดเฉือนสำหรับการสร้างรู และการกัดบนพื้นผิวชิ้นงานรูปหลายเหลี่ยม (Prismatic Component)

1. การติดตั้งโปรแกรม

โปรแกรมที่พัฒนาขึ้นประกอบไปด้วยแฟ้มข้อมูลชนิดต่างๆ เช่นแฟ้มข้อมูลโปรแกรม AutoLISP และแฟ้มข้อมูลรูปภาพสไลด์ เป็นต้น แฟ้มข้อมูลต่างๆจะถูกจัดเก็บอยู่ในไดเรกทอรี่ต่างๆดังนี้

- C:\CAPS ประกอบไปด้วยโปรแกรม AutoLISP ต่างๆที่พัฒนาขึ้น ได้แก่ *main.lsp*, *cus_menu.lsp*, *p_plan.lsp*, *process_cap.lsp* และ *cutting_tool.lsp* และแฟ้มข้อมูลกำหนดค่าติดตั้ง (Configuration File) ของโปรแกรม AutoCAD
- C:\CAPS\BLOCK จัดเก็บแฟ้มข้อมูลงานออกแบบลักษณะรูปร่างพิเศษ (**.DWG*)
- C:\CAPS\SLIDE จัดเก็บภาพสไลด์ต่างๆสำหรับเมนูรูปภาพ (**.SLD*)

การติดตั้งโปรแกรมต้องดำเนินการตามขั้นตอนต่อไปนี้

1.1 คัดลอกแฟ้มข้อมูลทั้งหมดในแผ่นข้อมูลลงใน HardDisk เช่น จาก A:\CAPS ไว้ใน C:\CAPS เป็นต้น

1.2 แก้ไข Support Path ของโปรแกรม AutoCAD โดยการเพิ่ม C:\CAPS ไว้ใน Support Part

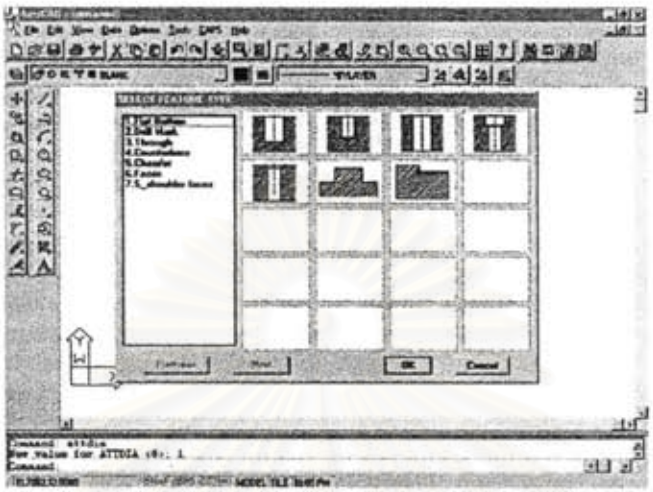
2. การใช้โปรแกรม

โปรแกรมที่ได้พัฒนาขึ้นจะใช้ Menu File ที่พัฒนาขึ้นใหม่ ได้แก่ ACADPP.MNS ซึ่งจะถูกเรียกใช้ทุกครั้งที่มีการเรียกโปรแกรม AutoCAD ขึ้นใหม่

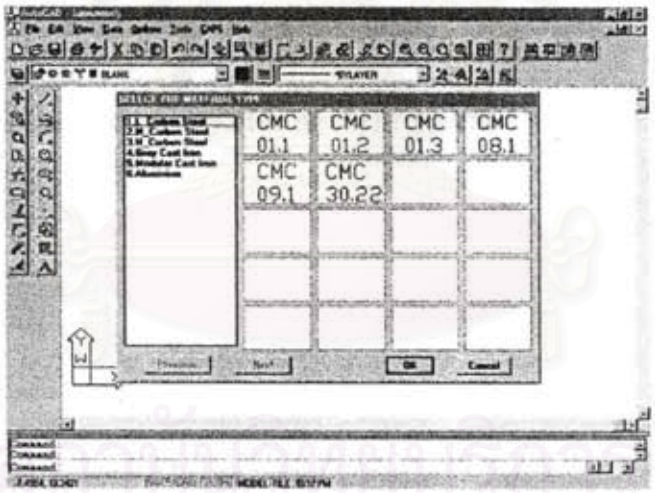
การทำงานของโปรแกรมเริ่มจากการเลือกเมนู Design Blank Part จากเมนูหลัก CAPS จากนั้นทำการตอบคำถามต่างๆที่จะปรากฏขึ้นบริเวณส่วนป้อนคำสั่ง (Command Menu) เพื่อกำหนดขนาดพื้นที่สำหรับงานออกแบบชิ้นงาน

หลังจากผู้วางแผนทำการวาดรูปชิ้นงานที่ต้องการแมชชีนเสร็จสิ้นแล้ว ผู้ออกแบบจะทำการเพิ่มลักษณะรูปร่างพื้นผิวต่างๆที่ต้องการแมชชีนลงในแบบ โดยเลือกเมนู Insert Feature จากเมนูหลัก

CAPS จากนั้นทำการเลือกลักษณะรูปร่างพื้นผิวต่างๆจากเมนูรูปภาพดังรูปที่ ข1 จากนั้นทำการตอบคำถามต่างที่ปรากฏขึ้นบริเวณส่วนป้อนคำสั่งเพื่อกำหนดคุณสมบัติของลักษณะรูปร่างที่ได้เลือกไว้แล้ว



รูปที่ ข1 เมนูรูปภาพสำหรับเลือกลักษณะรูปร่าง



รูปที่ ข2 เมนูรูปภาพสำหรับเลือกชนิดวัสดุชิ้นงาน

การวางแผนกระบวนการผลิตทำได้โดยเลือกเมนู Process Planning จากเมนูหลัก CAPS จากนั้นทำการกำหนดชนิดวัสดุจากเมนูรูปภาพดังรูปที่ ข2 หลังจากโปรแกรมทำการพิจารณาเลือกแผนกระบวนการผลิตที่เหมาะสมเสร็จสิ้น โปรแกรมจะถามชื่อชิ้นงานที่ทำการวางแผน จากนั้นทำการสร้างรายงานแผนกระบวนการผลิต แฟ้มข้อมูลรายงานแผนกระบวนการผลิตจะถูกจัดเก็บอยู่ในไดเรกทอรี่ C:\CAPS ซึ่งสามารถเรียกดูได้โดยโปรแกรม Notepad

ภาคผนวก ค.

รหัสคำสั่งโปรแกรม

1. ส่วนปรับปรุง ACADPP.MNS

```
***POP7
CAPSPop1_T        [&CAPS]
CAPSPop1_0        [&About CAPS...]^c^caboutpp;
CAPSPop1_1        [&Design Blank Part]^c^clayer make blank
                  ^c^ccolor 7 ^c^climit
CAPSPop1_2T       [&Insert Feature]^c^ccmdecho 0;^c^cattdia
                  1;$I=CAPS1 $I=*
CAPSPop1_3T       [&Process Planning]^c^c$I=CAPS2 $I=*;_.main;

**TB_MISCELLANEOUS
**CAPS
ID_caps           [_Toolbar("caps",_Floating,_Show,400,50,0)]
ID__0             [_Button("Edit current menu", ICON0041.bmp,
                  ICON_32_BLANK)]^c^c(MxaOpenFile (findfile
                  "AAA.TXT"))
ID_               [_Button("load Menu", ICON.bmp,
                  ICON_32_BLANK)]^c^c_menu
ID__1             [_Button("edit acad.lsp", ICON8467.bmp,
                  ICON_32_BLANK)]^c^c(MxaOpenFile (findfile
                  "ACAD.LSP"))
ID__2             [_Button("Load ACAD.lsp", ICON6334.bmp,
                  ICON_32_BLANK)]^c^c(load "ACAD.LSP")

***IMAGE
**CAPS1
[SELECT FEATURE TYPE]
[feat(blind_f,1.Flat Buttom)]^c^c_.layer _make hole color
     5;^c^c(setq insert 1);^c^c_drawhole;
[feat(blind_d,2.Drill Mark) ]^c^c_.layer _make hole color
     5;^c^c(setq insert 2);^c^c_drawhole;
[feat(thru,3.Through)       ]^c^c_.layer _make hole color
     5;^c^c(setq insert 3);^c^c_drawhole;
[feat(c_bore,4.Counterbore) ]^c^c_.layer _make hole color
     5;^c^c(setq insert 4);^c^c_drawhole;
[feat(c_sink,5.Chamfer)     ]^c^c_.layer _make hole color
     5;^c^c(setq insert 5);^c^c_drawhole;
[feat(facing,6.Faces)       ]^c^c_.layer _make face color
     5;^c^c_drawface;
[feat(facing_s,7.S_shoulder faces)]^c^c_.layer _make face
     color 5;^c^c_drawssface;
**CAPS2
[SELECT THE MATERIAL TYPE] ;ref coromil p.411
[mater(cmc01_1,1.L_Carbon Steel) ]    ^c^c(setq material (list
     "01_1" 125));
[mater(cmc01_2,2.M_Carbon Steel) ]    ^c^c(setq material (list
     "01_2" 150));
[mater(cmc01_3,3.H_Carbon Steel) ]    ^c^c(setq material (list
     "01_3" 170));
[mater(cmc08_1,4.Grey Cast Iron) ]    ^c^c(setq material (list
     "08_1" 180));
[mater(cmc09_1,5.Modular Cast Iron) ]^c^c(setq material
     (list "09_1" 160));
[mater(cmc30_22,6.Aluminium)      ]   ^c^c(setq material (list
     "30_22" 90));
```

2. รหัสโปรแกรมหลัก MAIN.LSP

```
;; MAIN PROGRAMMING FOR SELECTION PROCESS PLAN AND ESTIMATING
CUTTING CONDITION
(defun c:main (/ all_entity      x     y     I     j     a     b
                 c    d     e     f     g     s     en    enn
                 ang_list  entity_list      side  process     ang)
     (setq hbn (nth 1 material))
     (setq cmc (car material))
     (if (eq "failed" (load "proc_kb" "failed"))
       (alert "   Can't find required file PROC_KB.LSP   "))
     (if (eq "failed" (load "cut_tool" "failed"))
       (alert "   Can't find required file cut_tool.LSP   "))
     (princ "\nSearching all entity being performed ....")
     (setq y (ssget "x" '((8 . "HOLE,FACE") (0 . "INSERT"))))
     (if (/= y nil) (setq all_entity (list_entity y)))
     (setq e (length all_entity))
     (princ "\n\tNumber of entity = ")
     (princ e)
     (setq i 0)
     (setq ang_list '())
; GROUP THE ENTITY THAT HAVE THE SAME INSERT ANGLE
     (princ "\n\nSearching for number of setup being performed....")
     (repeat e
          (setq en (nth i all_entity))
          (setq enn (cdr (assoc 50 (entget en))))
          (if (/= ang_list nil)
          (progn
          (setq a (length ang_list))
          (setq j 0)
          (while (< j a)
          (setq ang (nth j ang_list))
          (if (/= (rtos enn 2 2) (rtos ang 2 2))
               (progn
               (if (= j (- a 1))
                    (progn
                    (setq ang_list (cons enn ang_list))
                    (setq j a)
                    (setq i (+ i 1))
                    ) ; END OF PROGN
                    (setq j (+ j 1))
               ); END OF IF
               ); END OF PROGN
               (progn
                    (setq i (+ i 1))
                    (setq j (+ a 1))
               ); END OF PROGN
          ); END OF IF
          ); END OF WHILE LOOP
          ); END OF THEN PROGN
          (progn
               (setq ang_list (cons enn ang_list))
               (setq i (+ i 1))
          ); END OF ELSE PROGN
          )); END OF IF AND END OF LOOP
```

```
      (REPEAT E)
            (setq b (length ang_list))
            (princ "\n\tNumber of setup = ") (princ b)
            (setq c 0)
            (setq process_list '())
            (while (< c b)
            (setq side '())
            (princ "\n\nExtract feature from face") (princ (+ c 1))
            (setq d (cons 50 (nth c ang_list)))
            (setq f (cons 8 "FACE"))
            (setq g (cons 8 "HOLE"))
            (setq s (ssget "x" (list d f)))
            (if (/= s nil) (setq side (append (list_entity s) side)))
            (setq s (ssget "x" (list d g)))
            (if (/= s nil) (setq side (append (list_entity s) side)))
            (if (/= side nil)     (setq process (process_select side)))
            (setq process_list (cons process process_list))
            (setq c (+ c 1))
            ); END OF WHILE
      (writeit cmc process_list); CALL FUNCTION "WRITEIT"
); END OF MAIN FUNCTION


;; FUNCTION FOR HELP PRINTING CUTTING CONDITION DATA
(defun print_ccon (con_list)
      (princ (car con_list))
      (princ (cadr con_list))
      (if (/= (nth 2 con_list) nil)
      (princ (nth 2 con_list)))
); END OF FUNCTION

; FUNCTION FOR LIST ENTITY NAME FROM THE SELECTION SET
(defun list_entity (ss / n i list_ent e entity )
      (setq n (sslength ss))
      (setq i 0)
      (setq list_ent '())
      (repeat n
            (setq e (entget (ssname ss i)))
            (setq i (+ i 1))
            (setq entity (cdr (assoc -1 e)))
            (setq list_ent (cons entity list_ent))
      ); END OF N LOOP
      (setq list_ent list_ent)
); END OF FUNCTION

; FUNCTION FOR PROCESS SELECTION FOR EACH SIDE
(defun process_select (side / c dummy e n)
      (setq n (length side))
      (if (/= n 0)
      (progn
      (setq selected '())
      (setq c 0)
      (setq dummy side)
      (repeat n
            (setq e (car dummy))
            (setq dummy (cdr dummy))
```

```
            (setq c (+ c 1))
            (cond
                  ((or (= (get_ent1 2 e) "FLAT_BUTTOM")
                  (= (get_ent1 2 e) "DRILL_MARK")
                  (= (get_ent1 2 e) "THROUGH")
                  (= (get_ent1 2 e) "COUNTER_BORE")) (setq selected
                        (append (hole_process e) selected)))
                  ((or (= (get_ent1 2 e) "CHAMFER")
                  (= (get_ent1 2 e) "FACE")
                  (= (get_ent1 2 e) "SSFACER")
                  (= (get_ent1 2 e) "SSFACEL")) (setq selected
                        (append (mill_process e) selected))))
            ); END OF N LOOP
            (setq selected selected)
            )
      ); END OF IF
); END OF FUNCTION

;; THIS FUNCTION IS FOR SELECT MILLING PROCESS.
(defun mill_process (ename / m_choice finish process feat width
                        depth lens dia ae len feat2 f process_list
                        process_data ent )
      (princ "\n\nProcess selection for :")
      (att_data ename)
      (setq feat (get_ent2 2 entdata))
      (setq width (get_ent2 41 entdata))
      (setq depth (get_ent2 151 entdata))
      (if (= depth "") (setq depth 3.0) (setq depth (atof depth)))
      (setq lens (get_ent2 150 entdata))
      (if (= lens "") (setq lens width) (setq lens (atof lens)))
      (if (= feat "FACE")
            (progn
            (if (> width lens)
            (progn
            (setq dia (* lens 1.2))
            (setq ae lens)
            (setq len width))
            (progn
            (setq dia (* width 1.2))
            (setq ae width)
            (setq len lens)))
            (princ "FACE  width ") (princ ae) (princ " length ")
            (princ len) (princ " depth ") (princ depth)
            (setq feat2 (list feat (fix ae) len depth cmc))
      )); END OF PROGN AND IF
      (if (or (= feat "SSFACEL") (= feat "SSFACER"))
            (progn
            (setq feat "SSFACE")
            (setq dia (* width 1.2))
            (setq len lens)
            (setq ae width)
            (setq depth (get_ent2 42 entdata))
            (princ "SSFACE  width ") (princ ae) (princ " length ")
            (princ len) (princ " depth ") (princ depth)
            (setq feat2 (list feat (fix ae) len depth cmc))
      ))
```

```
      (if (= feat "CHAMFER")
            (progn
            (setq ae (get_ent2 41 entdata))
            (setq len (atof (get_ent2 154 entdata)))
            (setq depth (get_ent2 42 entdata))
            (princ "Chamfer  diameter ") (princ (get_ent2 41
            entdata)) (princ " depth ") (princ (get_ent2 42
            entdata))
            (setq feat2 (list feat (fix ae) len depth cmc))
      ))

; VARIANT MODULE FOR HOLE MAKING
      (princ "\n\nQuery from existing process plan database : ")
      (getstring)
      (setq f (open "mill_p.dat" "r"))
      (read-line f)
      (while (setq process_list (read-line f))
      (setq process_data (read process_list))
      (setq ent (car process_data))
      (if (and (= feat (nth 0 ent))
      (= (fix ae) (nth 1 ent))
      (= len (nth 2 ent))
      (= depth (nth 3 ent))
      (= cmc (nth 4 ent)))
      (setq process (nth 1 process_data))); END OF IF
      ) ; END OF WHILE
      (if   (= process nil)
            (progn
            (princ "\n\nGenerative Process Planning")
            (setq m_choice (kb_mill entdata))     ; CALL KB_MILL
      (if (/= m_choice nil)
            (progn
            (princ "\nPossible process selected :")
            (setq len3 (length m_choice))
            (setq count3 0)
            (while (< count3 len3)
                  (progn
                        (princ "\n\t")
                        (princ (cadr (last (nth count3 m_choice))))
                        (princ "\t")
                        (princ (last (cadr (nth count3 m_choice))))
                        (princ " min.")
                        (setq count3 (+ count3 1))
                  ); END OF PROGN
                  )
      (setq finish (min_time m_choice))
; PRINT OUT PROCESS NAME OF THE MOST PRODUCTIVITY PROCESS
            (princ "\n\nMost productivity process is :")
            (princ "\n\t") (princ (cadr (last finish)))
            (setq process (list (cdr finish)))
            ); END OF THE FIRST PROGN
            (progn
            (princ "\nNo Available Process for :")
            (princ ename)
            (setq process nil)))
            (setq process process)
```

```
; BACK-UP EXISTING PROCESS PLAN
          (setq data (list feat2 process))
          (setq o (open "mill_p.dat" "a"))
          (print data o)
          (setq o (close o)))
     )
     (setq process process)
     )

;;  HOLE MAKING PROCESS
(defun hole_process (ename /  process_list f a_max min_doc n x
                       f_tool f_choice temp_process tempa finish
                       process process_data feat l d tf r tol
                       tol1 tol2 tol3 len ent feat2 data o)
     (princ "\n\nProcess selection for :")
     (att_data ename)
     (setq feat (get_ent2 2 entdata))
     (setq l (get_ent2 42 entdata))
     (setq d (get_ent2 41 entdata))
     (setq tf (get_ent2 150 entdata))
     (setq r (get_ent2 151 entdata))
     (if (= r "") (setq r 12.5) (setq r (atof r)))
     (setq tol (get_ent2 154 entdata))
          (if (= tol "") (setq tol1 200) (setq tol1 (atof tol)))
     (setq tol (get_ent2 155 entdata))
          (if (= tol "") (setq tol2 200) (setq tol2 (atof tol)))
     (setq tol (get_ent2 156 entdata))
          (if (= tol "") (setq tol3 200) (setq tol3 (atof tol)))
     (cond
          ((or (= feat "DRILL_MARK") (= feat "FLAT_BUTTOM") (=
          feat "COUNTER_BORE")) (setq len (+ 3 l (/ d 3))))
          ((= feat "THROUGH") (setq len (+ l 6 (/ d 3)))))

; PRINT FEATURE DETAIL
     (princ " ") (princ feat)
     (princ "  diameter ") (princ d)
     (princ "  depth ") (princ l)

; VARIANT MODULE FOR HOLE MAKING
     (princ "\n\nQuery from existing process plan database : ")
     (getstring)
     (setq f (open "hole_p.dat" "r"))
     (read-line f)
     (while (setq process_list (read-line f))
     (setq process_data (read process_list))
     (setq ent (car process_data))
     (if   (and (= feat (nth 0 ent))
          (= l (nth 1 ent))
          (= d (nth 2 ent))
          (= tf (nth 3 ent))
          (= r (nth 4 ent))
          (= tol1 (nth 5 ent))
          (= tol2 (nth 6 ent))
          (= tol3 (nth 7 ent))
          (= cmc (nth 8 ent)))
          (setq process (nth 1 process_data)));  END OF IF
```

```
); END OF WHILE
(if   (= process nil)
      (progn
      (princ "NONE")
      (princ "\n\nGenerative Process Planning")
      (getstring)
      (setq f_choice (finish_hole entdata))
      (if (/= f_choice nil)
            (progn
            (princ "\nPossible process selected :")
            (setq len2 (length f_choice))
            (setq count2 0)
            (while (< count2 len2)
            (progn
            (princ "\n\t")
            (princ (cadr (last (nth count2 f_choice))))
            (princ "\t")
            (princ (last (cadr (nth count2 f_choice))))
            (princ " min.")
            (setq count2 (+ count2 1))))
            (setq x (length f_choice))
            (setq n 0)
            (setq temp_process '())
; CHECK FOR ONE PASS PROCESS
      (while (< n x)
            (setq tempa (nth n f_choice))
            (if   (= (car tempa) 0)
            (setq temp_process (cons tempa temp_process)))
            (setq n (+ n 1)))
            (if (= temp_process nil)
            (setq finish (min_time f_choice))
            (setq finish (min_time temp_process)))
            (princ "\n\nMost productivity finishing process is :")
            (princ "\n\t") (princ (cadr (last finish)))
            (setq process (list (cdr finish))))
       (progn
            (princ "\nno available process for :")
            (princ ename)
            (setq process nil)))
;  ROUGH PROCESS
(if   (and (/= (car finish) 0) (/= process nil))
      (progn
      (setq a_max (caar finish))
      (setq min_doc (cadr (car finish)))
      (setq r_choice (rough_hole entdata a_max min_doc))
      (if (/= r_choice nil)
            (progn
            (setq x (length r_choice))
            (setq n 0)
            (setq temp_process '())
      ; CHECK FOR ONE PASS PROCESS
            (while (< n x)
            (setq tempa (nth n r_choice))
      (if (= (car tempa) 0)
            (setq temp_process (cons tempa temp_process)))
            (setq n (+ n 1)))
```

```
            (if (= temp_process nil)
                  (setq rough (min_time r_choice))
                  (setq rough (min_time temp_process)))
                  (setq process (cons rough process))))
             )
      )
            (setq process process)
      ; BACK-UP EXISTING PROCESS PLAN
            (setq feat2 (list feat l d tf r tol1 tol2 tol3 cmc))
            (setq data (list feat2 process))
            (setq o (open "hole_p.dat" "a"))
            (print data o)
            (setq o (close o))
            ); END OF GENERATIVE APPROACH SECTION
      )
      (setq process process)
);END OF FUNCTION

;SELECT PRODUCTIVE PROCESS FUNCTION
(defun min_time (choice / l n process tempa est_time best_time
                  best_tool)
      (setq l (length choice))
      (setq n 0)
      (setq best_time (last (cadr (nth 0 choice))))
      (setq best_tool (nth 0 choice))
      (while (< n l)
            (setq tempa (nth n choice))
            (setq est_time (last (cadr tempa)))
      (if (< est_time best_time)
            (progn
            (setq best_time est_time)
            (setq best_tool tempa)))
            (setq n (+ n 1))
      )
      (setq best_tool best_tool))

(defun att_data (ename)
      ; EXTRACT ALL ENTITY DATA OF THE CURRENT ENAME
      (setq entdata (entget ename))
      ; extract the entity type
      (setq enttype (cdr (assoc 0 entdata)))
      (setq count 150)
      ; repeat extract attribute information as long as
      (while (/= enttype "SEQEND")
      ; ENTTYPE DOES NOT EQUAL "SEQUED".SEQUED IS A MARKER
      ; THAT SIGNALS THE END OF THE ATTRIBUTES
      ; SETS THE SYMBOL ENAMEAT TO THE NEXT ENTITY NAME
      (setq enameat (entnext ename))
      ; SETS THE SYMBOL ATTDATA TO THE ENTITY DATA OF ENAMEAT
      (setq attdata (entget enameat))
      ; EXTRACT THE ENTITY DATA FOR ENTTYPE
      (setq enttype (cdr (assoc 0 attdata)))
      ; EXTRACT THE ENTITY DATA FOR TAGNAME
      (setq tagname (cdr (assoc 2 attdata)))
      ; EXTRACT THE ENTITY DATA FOR TAGVALU
      (setq tagvalu (cdr (assoc 1 attdata)))
```

```
      (setq a (cons count tagvalu))
      (if (/= enttype "SEQEND") (setq entdata (cons a entdata)))
      (setq count (+ count 1))
      ; SETS ENAME TO ENAMEAT
      (setq ename enameat)
      ))

; FUNCTION FOR EXTRACT ENTITY DATA FROM ENTDATA SYMBOL

;KNOW ENAME RETURN ASSOC
(defun get_ent1 (code ename / entdata)
      (setq entdata (entget ename))
      (cdr (assoc code entdata))
)

;KNOW ENTDATA RETURN ASSOC
(defun get_ent2 (code entdata)
(cdr (assoc code entdata))
)

(defun rtd (angg)
      (/ (* angg 180.0) pi))

(defun dtr (angg)
      (* pi (/ ang 180.0)))

(princ "\nMain.Lsp loaded.")
(princ)
```

3. รหัสโปรแกรม CUSMENU.LSP

```
;;; CUSTOMIZE AUTOCAD MENU
; FUNCTION FOR SET AUTOCAD ENVIRONMENT
(defun c:limit (/ width height)
      (setq width (getreal "\n>>Blank Width: "))
      (setq height (getreal "\n>>Blank Height: "))
      (command "limits" (list 0 0) (list width height))
      (command "zoom" "all"))
(defun c:drawhole (/ startpoint dia depth ang)
      (setq startpoint (getpoint "\n>>Start Point: "))
      (setq dia (getreal "\n>>Diameter: "))
      (setq depth (getreal "\n>>Depth: "))
      (setq ang (getreal "\n>>Approach direction: "))
      (cond
            ((= insert 1) (command "_.insert" "flat_buttom"
            startpoint dia depth ang))
            ((= insert 2) (command "_.insert" "drill_mark"
            startpoint dia depth ang))
            ((= insert 3) (command "_.insert" "through" startpoint
            dia depth ang))
            ((= insert 4) (command "_.insert" "counter_bore"
            startpoint dia depth ang))
            ((= insert 5) (command "_.insert" "chamfer" startpoint
            dia depth ang))))
```

```
(defun c:drawface (/ sp ep d r_ang ang)
      (setq sp (getpoint "\n>>Start Point: "))
      (setq ep (getpoint "\n>>End Point: "))
      (setq d (distance sp ep))
      (setq r_ang (angle sp ep))
      (setq ang (rtd (- r_ang pi)))
      (command "_.insert" "face" sp d d ang))

(defun f_slope (fp sp / x1 x2 y1 y2 slope)
      (setq x1 (car fp))
      (setq x2 (car sp))
      (setq y1 (cadr fp))
      (setq y2 (cadr sp))
      (if (/= (- x2 x1) 0)(setq slope (/ (- y2 y1) (- x2 x1)))
            (progn
            (if (> x2 x1) (setq slope 2.0)
            (setq slope 2.0)))))

(defun c:drawssface (/ sp ep tp anga angb d h angl angr dip)
      (setq sp (getpoint "\n>>Start Point: "))
      (setq ep (getpoint "\n>>Corner Point: "))
      (setq tp (getpoint "\n>>Third Point: "))
      (setq slope_a (f_slope ep sp))
      (setq slope_b (f_slope tp sp))
      (setq dip (- slope_b slope_a))
      (setq anga (angle sp ep))
      (setq angb (angle ep tp))
      (if (= angb 0) (setq angb (+ (* 2 pi) angb)))
      (setq d (distance sp ep))
      (setq h (distance ep tp))
      (setq angl (rtd (- anga pi)))
      (setq angr (rtd anga))
      (princ (list anga angb))
      (cond
            ((< dip 0) (command "_.insert" "ssfacel" ep d h angl))
            ((> dip 0) (command "_.insert" "ssfacer" ep d h angr)))
)

; FUNCTION FOR CONVERT RADIUS TO DEGREE
(defun rtd (angg)
      (/ (* angg 180.0) pi))

(defun drawchamfer (sp dia depth ang)
      (setq c_ang 30)
      (setq first (polar sp (atop ang) ((+ (/ dia 2) (* depth
      (atan (atop c_ang)))))))
      (setq second (polar first (+ pi (+ (atop ang) (* 2 (atop
      c_ang))))))
      (command "line" first second))

; FUNCTION FOR CONVERT DEGREE TO RADIUS
(defun atop (a)
      (/ (* a pi) 180))
```

```
; ABOUT FUNCTION
(defun c:aboutpp ()
      (alert "Computer Aided Process Planning\nfor Hole Making and
      Milling Process\nBy Kasan Pinweha GC816463"))

; IF LOADING PROCESS IS SUCCESSFUL THEN PRINT THE MASSAGE
(princ "\nCus_Menu.Lsp loaded.")
(princ)
```

4. รหัสโปรแกรม CUT_TOOL.LSP

```
;TWIST_DRILL DATABASE (REF GTS PAGE220-221)
(defun t_drill (dia len l r hbn / tool_list t_data tool cut_con
                sel_tool f feed speed time)
      (setq f (open "t_drill.dat" "r"))
      (read-line f)
      (while (setq tool_list (read-line f))
      (setq t_data (read tool_list))
      (if (and (= dia (nth 1 t_data)) (<= len (nth 2 t_data)))
            (progn
            (setq tool (list (car t_data) "TwistDrill" dia (last
            t_data)))
            (setq feed (/ (* 2.83 (expt dia 0.6) (expt r 0.5) (-
                  1.09 (* 0.04 (/ l dia)))) hbn))
            (setq speed (/ (* 3.38 (expt dia 0.4) (expt (/ 160.0
                  hbn) 1.6)) (expt feed 0.6)))
            (setq time (/ (* pi dia len) (* speed feed 1000)))
            (setq cut_con (list len feed speed time))
            (setq sel_tool (list cut_con tool)))))
      (setq f (close f))
      (setq sel_tool sel_tool))

;INSERT_DRILL DATABASE ,IS = 68 (REF GTS PAGE186)
(defun i_drill (dia len l r hbn / tool_list t_data tool cut_con
                sel_tool f feed speed time)
      (setq f (open "i_drill.dat" "r"))
      (read-line f)
      (setq xxx 0)
      (while (= xxx 0)
            (setq tool_list (read-line f))
            (setq t_data (read tool_list))
            (if (and (>= dia (nth 1 t_data)) (<= dia (nth 2
                  t_data)) (<= len (nth 3 t_data)))
            (progn
            (setq tool (list (car t_data) "InsertDrill" dia (last
                  t_data)))
            (setq feed (* 0.3 (expt (/ dia 56.0) 0.424) (expt (/ r
                  12.5) 0.4)))
            (setq speed (/ (* 7.32 (expt dia 0.6)) (* (expt feed
                  0.3) (expt (/ hbn 220.0) 0.9))))
            (setq time (/ (* pi dia len) (* speed feed 1000)))
            (setq cut_con (list len feed speed time))
            (setq sel_tool (list cut_con tool))
            (setq xxx 1))))
      (setq f (close f))
```

```
      (setq sel_tool sel_tool))

;SOLID CARBIDE DRILL DATABASE (REF GTS PAGE174)
(defun sc_drill (dia len l r hbn / tool_list t_data tool cut_con
                 sel_tool f feed speed time)
      (setq f (open "sc_drill.dat" "r"))
      (read-line f)
      (while (setq tool_list (read-line f))
      (setq t_data (read tool_list))
      (if (and (= dia (nth 1 t_data)) (<= len (nth 2 t_data)))
            (progn
            (setq tool (list (car t_data) "SolidCarbideDrill" dia
                  (last t_data)))
            (setq feed (/ (* 2.83 (expt dia 0.6) (expt r 0.5) (-
                  1.09 (* 0.04 (/ 1 dia)))) hbn))
            (setq speed (/ (* 10.0 (expt dia 0.4) (expt (/ 160.0
                  hbn) 1.6)) (expt feed 0.6)))
            (setq time (/ (* pi dia len) (* speed feed 1000)))
            (setq cut_con (list len feed speed time))
            (setq sel_tool (list cut_con tool)))))
      (setq f (close f))
      (setq sel_tool sel_tool))

;CORE DRILL DATABASE (REF GTS PAGE 225)
(defun c_drill (dia len l r hbn / tool_list t_data tool cut_con
                 sel_tool f feed speed time)
      (setq f (open "c_drill.dat" "r"))
      (read-line f)
      (while (setq tool_list (read-line f))
      (setq t_data (read tool_list))
      (if (and (= dia (nth 1 t_data)) (<= len (nth 2 t_data)))
            (progn
            (setq tool (list (car t_data) "CoreDrill" dia (last
                  t_data)))
            (setq di (nth 3 t_data))
            (setq feed (/ (* 2.83 (expt dia 0.6) (expt r 0.5) (-
                  (* 0.04 (/ 1 dia)))) (* hbn (expt (/ (- dia
                  di) (* 2 dia)) 0.1))))
            (setq speed (/ (* 5.0 (expt dia 0.4) (expt (/ 160.0
                  hbn) 1.6)) (* (expt feed 0.6) (expt (/ (- dia di)
                  2.0) 0.1))))
            (setq time (/ (* pi dia len) (* speed feed 1000)))
            (setq cut_con (list len feed speed time))
            (setq sel_tool (list cut_con tool)))))
      (setq f (close f))
      (setq sel_tool sel_tool))

;BORING DATABASE (REF SANDVIK PAGE 209-213)
(defun f_boring (dia len l r hbn / tool_list t_data tool cut_con
                 sel_tool f feed speed time)
      (setq f (open "boring.dat" "r"))
      (read-line f)
      (setq xxx 0)
      (while      (= xxx 0)
            (setq tool_list (read-line f))
            (setq t_data (read tool_list))
```

```
            (if (and (>= dia (nth 1 t_data)) (<= dia (nth 2
                 t_data)) (<= l (nth 3 t_data)))
            (progn
            (setq tool (list (car t_data) "FineBoring" dia (last
                 t_data)))
            (setq feed (* 0.088 (expt r 0.5)))
            (setq speed (* (/ 90 (* (expt (+ 0.3 (* 0.11 (expt r
                 1.4))) 0.1) (expt feed 0.25))) (- 3.5 (* 2.5
                 (expt (/ hbn 150) 0.18)))))
            (setq time (/ (* pi dia l) (* speed feed 1000)))
            (setq cut_con (list l feed speed time))
            (setq sel_tool (list cut_con tool))
            (setq xxx 1))))
      (setq f (close f))
      (setq sel_tool sel_tool))

;ROUGH BORING DATABASE (REF SANDVIK PAGE 202)
(defun r_bore  (dia len l r hbn / tool_list t_data tool cut_con
                 sel_tool f feed speed time)
      (setq f (open "r_boring.dat" "r"))
      (read-line f)
      (setq xxx 0)
      (while     (= xxx 0)
            (setq tool_list (read-line f))
            (setq t_data (read tool_list))
            (if (and (>= dia (nth 1 t_data)) (<= dia (nth 2
                 t_data)) (<= len (nth 3 t_data)))
            (progn
            (setq tool (list (car t_data) "RoughBoring" dia (last
                 t_data)))
            (setq feed (* 0.088 (expt 12.5 0.5)))
            (setq speed (* (/ 90 (* (expt (+ 0.3 (* 0.11 (expt
                 12.5 1.4))) 0.1) (expt feed 0.25))) (- 3.5 (* 2.5
                 (expt (/ hbn 150) 0.18)))))
            (setq time (/ (* pi dia l) (* speed feed 1000)))
            (setq cut_con (list l feed speed time))
            (setq sel_tool (list cut_con tool))
            (setq xxx 1))))
      (setq f (close f))
      (setq sel_tool sel_tool))

;reaming database (ref GTS Page 276)
(defun ream (dia len l r hbn / tool_list t_data tool cut_con
              sel_tool f feed speed time)
      (setq f (open "reamer.dat" "r"))
      (read-line f)
      (while (setq tool_list (read-line f))
      (setq t_data (read tool_list))
      (if (and (= dia (nth 1 t_data)) (<= len (nth 2 t_data)))
            (progn
            (setq tool (list (car t_data) "Reamer" dia (last
                 t_data)))
            (setq di (- dia 0.2))
            (setq feed (* (/ 0.1 (expt (- dia di) 0.1)) (expt (/
                 220 hbn) 1.4) (expt (/ dia 3) 0.62)))
            (cond
```

```
        ((>= r 1.575) (setq feed (* feed (expt (/ r 3.125) 0.3))))
        ((< r 1.575) (setq feed (* 0.67 feed (expt (/ r 1.575) 0.15)
        (setq speed (* 27 (expt (/ 220 hbn) 0.7)))
        (setq time (/ (* pi dia len) (* speed feed 1000)))
        (setq cut_con (list len feed speed time))
        (setq sel_tool (list cut_con tool)))))
        (setq f (close f))
        (setq sel_tool sel_tool))

;COUNTER_BORE
(defun c_bore (dia len l r hbn)
     (setq tool (list (strcat "CB" (rtos dia 2 0)) "CounterBore"
          dia len))
     (setq feed (/ (* 2.83 (expt dia 0.6) (expt r 0.5) (- 1.09 (*
          0.04 (/ l dia)))) hbn))
     (setq speed (/ (* 3.38 (expt dia 0.4) (expt (/ 160.0 hbn)
          1.6)) (expt feed 0.6)))
     (setq time (/ (* pi dia l) (* speed feed 1000)))
     (setq cut_con (list l feed speed time))
     (setq sel_tool (list cut_con tool)))

(defun coro245 (dia len hbn ap cmc ae / x b a chipspace tool_list
                insert_coro245 t_data tool)

     (setq tool_list (list
     '("R245-050Q22-12L" L 3 50)      '("R245-063Q22-12L" L 4 63)
     '("R245-080Q27-12L" L 4 80)      '("R245-100Q32-12L" L 5 100)
     '("R245-125Q40-12L" L 6 125)     '("R245-160Q40-12L" L 7 160)
     '("R245-200Q40-12L" L 8 200)     '("R245-250Q40-12L" L 10 250)
     '("R245-050Q22-12M" M 4 50)      '("R245-063Q22-12M" M 5 63)
     '("R245-080Q27-12M" M 6 80)      '("R245-100Q32-12M" M 7 100)
     '("R245-125Q40-12M" M 8 125)     '("R245-160Q40-12M" M 10 160)
     '("R245-200Q40-12M" M 12 200)    '("R245-250Q40-12M" M 14 25)
     '("R245-050Q22-12H" H 5 50)      '("R245-063Q22-12H" H 6 63)
     '("R245-080Q27-12H" H 8 80)      '("R245-100Q32-12H" H 10 100)
     '("R245-125Q40-12H" H 12 125)    '("R245-160Q40-12H" H 16 160)
     '("R245-200Q40-12H" H 20 200)    '("R245-250Q40-12H" H 24 250)))

     ; TEMPLATE OF INSERT TABLE
     ; CMC , COROMANT GRADE , MATERIAL , GEOMETRY , VCO , CVC ,
     HEX
     ; REF. COROMANT PAGE 26,408
     (setq insert_coro245 (list
          '("01_1" "530" P L 444 -0.1864 0.08)
          '("01_2" "530" P L 408 -0.1864 0.08)
          '("01_3" "530" P L 451 -0.3376 0.08)
          '("01_1" "4030" P M 449 -0.2211 0.17)
          '("01_1" "4030" P H 449 -0.2211 0.25)
          '("01_2" "4030" P L 407 -0.2158 0.10)
          '("01_2" "4030" P M 407 -0.2158 0.17)
          '("01_2" "4030" P H 407 -0.2158 0.25)
          '("01_3" "4030" P L 373 -0.2243 0.10)
          '("01_3" "4030" P M 373 -0.2243 0.17)
          '("01_3" "4030" P H 373 -0.2243 0.25)
          '("30_22" "H13A" K L 806 -0.0781 0.08)
          '("08_1" "3020" K L 379 -0.2467 0.10)
```

```
          '("08_1" "3020" K M 379 -0.2467 0.17)
          '("08_1" "3020" K H 379 -0.2467 0.25)
          '("09_1" "3020" K L 224 -0.2256 0.10)
          '("09_1" "3020" K M 224 -0.2256 0.17)
          '("09_1" "3020" K H 224 -0.2256 0.25)))

;- DETERMINE THE CHIP SPACE,PITCH(COARSE(L)/CLOSE(M)/EXTRA CLOSE(H))
; LONG CHIPPING MATERIALS NEED MORE SPACE FOR THE CHIP THUS
COARSE PITCH IS SELECTED FOR LONG
; CHIPPING MATERIALS(E.G. LOW CARBON STEEL) AND CLOSE OR EXTRA
CLOSE PITCH IS SELECTED FOR
; HARDER MATERIALS (E.G. CAST IRON AND TITANIUM)
     (cond
          ((or (= cmc "01_2") (= cmc "01_3") (= cmc "01_1"))
          (setq chipspace 'M))
          ((= cmc "30_22") (setq chipspace 'L))
          ((or (= cmc "08_1") (= cmc "09_1")) (setq chipspace 'H))
     );END OF COND

; SELECT TOOL FROM TOOL_LIST BY MATCH CHIPSPACE AND DIAMETER
     (setq t_data (nth b tool_list))
     (if (and (= chipspace (nth 1 t_data)) (> (nth 3 t_data) dia))
          (progn
          (setq b (+ x 1))
          (setq a (length insert_coro245))
          (setq z (nth 2 t_data))
          (setq y 0)
     (while (< y a)
          (setq insert_data (nth y insert_coro245))
     (if (and (= cmc (car insert_data)) (= chipspace (nth 3
          insert_data)))
          (progn
          (setq y (+ a 1))
          (setq insert insert_data)
          (setq insert_type (nth 1 insert))
          (setq vco (nth 4 insert))
          (setq cvc (nth 5 insert))
          (setq hex (last insert))
          (setq dc  (last t_data))
          (setq dap (+ dc (* 2.0 (/ ap 1.0))))
          (setq vc  (cal_vc vco cvc hex ae dap))
          (setq n     (cal_n vc dap))
          (setq fz  (cal_fz hex 45))
          (setq vf  (cal_vf z n fz))
          (setq len (+ len 10 dc))
          (setq time (/ len vf))
          (setq tool (list (car t_data) "FaceMill_CORO245" (last
               t_data) nil  insert_type  z))
          (setq cut_con (list vc n fz vf time))
          (setq sel_tool (list cut_con tool)))
          (setq y (+ Y 1)))
     ) ; END OF WHILE
     ) ; END OF PROGN
          (setq b (+ b 1)))
     ) ; END OF WHILE
     (setq sel_tool sel_tool)) ; END OF FUNCTION
```

```
(defun coro290 (dia len hbn ap cmc ae / x b a chipspace tool_list
                insert_coro290 t_data tool)
     (setq tool_list (list
     '("R290-050Q22-15L" L 3 50)      '("R290-063Q22-15L" L 4 63)
     '("R290-080Q27-15L" L 4 80)      '("R290-100Q32-15L" L 5 100)
     '("R290-125Q40-15L" L 6 125)     '("R290-160Q40-15L" L 8 160)
     '("R290-200Q60-15L" L 10 200)    '("R290-250Q60-15L" L 12 250)
     '("R290-050Q22-15M" M 4 50)      '("R290-063Q22-15M" M 5 63)
     '("R290-080Q27-15M" M 6 80)      '("R290-100Q32-15M" M 6 100)
     '("R290-125Q40-15M" M 8 125)     '("R290-160Q40-15M" M 10 160)
     '("R290-200Q60-15M" M 12 200)    '("R290-250Q60-15M" M 16 250)
     '("R290-063Q22-15H" H 6 63)      '("R290-080Q27-15H" H 8 80)
     '("R290-100Q32-15H" H 10 100)    '("R290-125Q40-15H" H 12 125)
     '("R290-160Q40-15H" H 16 160)    '("R290-200Q60-15H" H 20 200)
     '("R290-250Q60-15H" H 25 250)))

; TEMPLATE OF INSERT TABLE
; CMC , COROMANT GRADE , MATERIAL , GEOMETRY , VCO , CVC , HEX
; REF. COROMANT PAGE 26,408
     (setq insert_coro290 (list
          '("01_1" "4030" P L 449 -0.2211 0.12)
          '("01_2" "4030" P L 407 -0.2158 0.12)
          '("01_3" "4030" P L 373 -0.2243 0.12)
          '("01_1" "4030" P M 449 -0.2211 0.17)
          '("01_2" "4030" P M 407 -0.2158 0.17)
          '("01_3" "4030" P M 373 -0.2243 0.17)
          '("01_1" "4030" P H 449 -0.2211 0.25)
          '("01_2" "4030" P H 407 -0.2158 0.25)
          '("01_3" "4030" P H 373 -0.2243 0.25)
          '("30_22" "H13A" K L 806 -0.0781 0.12)
          '("30_22" "H13A" K M 806 -0.0781 0.17)
          '("30_22" "H13A" K H 806 -0.0781 0.25)
          '("08_1" "3020" K L 379 -0.2467 0.12)
          '("08_1" "3020" K M 379 -0.2467 0.17)
          '("08_1" "3020" K H 379 -0.2467 0.25)
          '("09_1" "3020" K L 224 -0.2256 0.12)
          '("09_1" "3020" K M 224 -0.2256 0.17)
          '("09_1" "3020" K H 224 -0.2256 0.25)))

; DETERMINE THE CHIP SPACE ,PITCH (COARSE(L) / CLOSE(M) / EXTRA
CLOSE(H))
; LONG CHIPPING MATERIALS NEED MORE SPACE FOR THE CHIP THUS
COARSE PITCH IS SELECTED FOR LONG
; CHIPPING MATERIALS(E.G. LOW CARBON STEEL) AND CLOSE OR EXTRA
CLOSE PITCH IS SELECTED FOR
; HARDER MATERIALS (E.G. CAST IRON AND TITANIUM)

     (cond
          ((or (= cmc "01_2") (= cmc "01_3") (= cmc "01_1"))
          (setq chipspace 'M))
          ((= cmc "30_22") (setq chipspace 'L))
          ((or (= cmc "08_1") (= cmc "09_1")) (setq chipspace 'H))
     )
```

```
      (setq x (length tool_list))
      (setq b 0)
      (while (< b x)
            (setq t_data (nth b tool_list))
      (if (and (= chipspace (nth 1 t_data)) (> (nth 3 t_data) dia))
            (progn
            (setq b (+ x 1))
            (setq a (length insert_coro290))
            (setq z (nth 2 t_data))
            (setq y 0)
      (while (< y a)
            (setq insert_data (nth y insert_coro290))
            (nth 3 insert_data)))
(if (and (= cmc (car insert_data)) (= chipspace (nth 3
      insert_data)))
      (progn
            (setq y (+ a 1))
            (setq insert insert_data)
            (setq insert_type (nth 1 insert))
            (setq vco (nth 4 insert))
            (setq cvc (nth 5 insert))
            (setq hex (last insert))
            (setq dc  (last t_data))
            (setq dap dc)
            (setq vc  (cal_vc vco cvc hex ae dap))
            (setq n     (cal_n vc dap))
            (setq fz  (cal_fz hex 90))
            (setq vf  (cal_vf z n fz))
            (setq len (+ len 10 dc))
            (setq time (/ len vf))
; END OF FIND INSERT CODE
            (setq tool (list (car t_data) "FaceMill_CORO290" (last
                  t_data) nil  insert_type  z))
            (setq cut_con (list vc n fz vf time))
            (setq sel_tool (list cut_con tool)))
            (setq y (+ Y 1)))
      ) ; END OF WHILE
      ); END OF PROGN
      (setq b (+ b 1)))
      ) ; END OF WHILE
      (setq sel_tool sel_tool)
) ; END OF FUNCTION

; CHAMFER (CUTTING CONDITION FROM INSERT DRILL)
(defun chamfer (dia ang depth r hbn)
      (setq tool (list (strcat "chamfer" (rtos ang 2 0))
            "Chamfering" dia depth))
      (setq feed (* 0.3 (expt (/ dia 56.0) 0.424) (expt (/ r 12.5)
            0.4)))
      (setq speed (/ (* 7.32 (expt dia 0.6)) (* (expt feed 0.3)
            (expt (/ hbn 220.0) 0.9))))
      (setq time (/ (* pi dia depth) (* speed feed 1000)))
      (setq cut_con (list depth feed speed time))
      (setq sel_tool (list cut_con tool))
); END OF FUNCTION
```

```
(defun cal_vc (vco cvc hex ae dap)
      (* vco (expt 2.72 (/ (* cvc hex ae) (expt dap 0.5)))))

(defun cal_n (vc dap)
      (/ (* vc 1000.0) (* pi dap)))

(defun cal_fz (hex ang)
      (/ hex (sin (/ (* pi ang) 180.0))))

(defun cal_vf (z n fz)
      (* z n fz))
```

5. รหัสคำสั่งโปรแกรม PROC_KB.LSP

```
;;; PROCESS KNOWLEDGE BASE

;; FINISHING PROCESS FOR HOLE MAKING
(defun finish_hole  (goal / a_min finish a_max min_doc feat l d r
                          tf len tol tol1 tol2 tol3)
      (setq finish '())
      (setq feat (get_ent2 2 goal))
      (setq l (get_ent2 42 goal))
      (setq d (get_ent2 41 goal))
      (setq tf (get_ent2 150 goal))
      (setq r (get_ent2 151 goal))
      (if (= r "") (setq r 12.5) (setq r (atof r)))
      (setq tol (get_ent2 154 goal))
      (if (= tol "") (setq tol1 200) (setq tol1 (atof tol)))
      (setq tol (get_ent2 155 goal))
      (if (= tol "") (setq tol2 200) (setq tol2 (atof tol)))
      (setq tol (get_ent2 156 goal))
      (if (= tol "") (setq tol3 200) (setq tol3 (atof tol)))
      (cond
            ((or (= feat "DRILL_MARK") (= feat "FLAT_BUTTOM")
            (= feat "COUNTER_BORE")) (setq len (+ 3 l (/ d 3))))
            ((= feat "THROUGH") (setq len (+ l 6 (/ d 3))))))

; COUNTER BORE
      (if   (= feat "COUNTER_BORE")
            (progn
            (setq tool (c_bore d len l r hbn))
      (if (/= tool nil)
            (progn
            (setq tool (cons 0 tool))
            (setq finish (cons tool finish)))
            (setq finish finish)))
            (setq finish finish))

; TWIST DRILL
      (if   (and
            (or (= feat "THROUGH") (= feat "DRILL_MARK"))
            (< (/ l d) 10)
            (> d 10.50)
```

```
            (<= d 38.00)
            (>= tol1 (+ 0.05 (* 0.0005 (expt (/ l d) 3))))
            (>= tol2 0.1)
            (>= tol3 (* 0.025 (expt d 0.3)))
            (or (and (>= r 3.125)
            (<= r 12.5))))
            (progn
            (setq tool (t_drill d len l r hbn))
            (if (/= tool nil)
            (progn
      ;FOR DECISION ON INTERMEDIATE PROCESS
            (setq tool (cons 0 tool))
            (setq finish (cons tool finish)))
            (setq finish finish)))
            (setq finish finish))

; INSERT DRILL
      (if   (and
            (or (= feat "THROUGH") (= feat "FLAT_BUTTOM")
            (= feat "DRILL_MARK"))
            (< (/ l d) 3.5)
            (>= d 16)
            (<= d 82)
            (>= tol1 0.03)
            (>= tol2 0.07)
            (>= tol3 0.05)
            (or (and (>= r 3.125)
            (<= r 12.5))))
            (progn
            (setq tool (i_drill d len l r hbn))
            (if (/= tool nil)
            (progn
      ;decision on intermediate process
            (setq tool (cons 0 tool))
            (setq finish (cons tool finish)))
            (setq finish finish)))
            (setq finish finish))

; SOLID CARBIDE DRILL
      (if   (and
            (or (= feat "THROUGH") (= feat "DRILL_MARK"))
            (< (/ l d) 2)
            (>= d 3)
            (<= d 20)
            (>= tol1 0.02)
            (>= tol2 0.04)
            (>= tol3 0.03)
            (or (and (>= r 2.25)
            (<= r 12.5))))
            (progn
            (setq tool (sc_drill d len l r hbn))
      (if (/= tool nil)
            (progn
      ;decision on intermediate process
            (setq tool (cons 0 tool))
            (setq finish (cons tool finish)))
```

```
            (setq finish finish)))
            (setq finish finish))

; CORE DRILL
      (if   (and
            (or (= feat "THROUGH") (= feat "FLAT_BUTTOM")
                  (= feat "DRILL_MARK"))
            (< (/ l d) 10)
            (> d 7.8)
            (<= d 31.0)
            (>= tol1 0)
            (>= tol2 0.05)
            (>= tol3 0)
            (or (and (>= r 1.575)
            (<= r 6.25))))
            (progn
            (setq tool (c_drill d len l r hbn))
      (if (/= tool nil)
            (progn
      (if (/= tf "core")
            (progn
            (setq a_max (* 0.35 d))
            (setq min_doc 0.25)
            (setq tool (cons (list a_max min_doc) tool)))
            (setq tool (cons 0 tool)))
            (setq finish (cons tool finish)))
            (setq finish finish)))
            (setq finish finish))

; FINE BORING
      (if   (and
            (or (= feat "THROUGH") (= feat "FLAT_BUTTOM")
                  (= feat "DRILL_MARK"))
            (< (/ l d) 10)
            (>= d 10)
            (<= d 266.5)
            (>= tol1 0.01)
            (>= tol2 0.01)
            (>= tol3 0.01)
            (or (and (>= r 0.8)
            (<= r 12.5))))
            (progn
            (setq tool (f_boring d len l r hbn))
      (if (/= tool nil)
            (progn
            (if (/= tf "core")
            (progn
            (setq a_max (+ 0.3 (* 0.11 (expt r 1.4))))
            (setq min_doc 0.3)
            (setq tool (cons (list a_max min_doc) tool)))
            (setq tool (cons 0 tool)))
            (setq finish (cons tool finish)))
            (setq finish finish)))
            (setq finish finish))
```

```
; REAMING
      (if   (and
            (or (= feat "THROUGH") (= feat "FLAT_BUTTOM")
                  (= feat "DRILL_MARK"))
            (< (/ l d) 15)
            (> d 7)
            (<= d 16)
            (>= tol1 0)
            (>= tol2 0.01)
            (>= tol3 0)
            (or (and (>= r 0.4)
            (<= r 1.575))))
            (progn
            (setq tool (ream d len l r hbn))
            (if (/= tool nil)
            (progn
      (if (/= tf "core")
            (progn
            (setq a_max (* (+ 0.01 (* 0.175 r)) (expt (/ 200 hbn)
                  0.4)))
            (setq min_doc 0.02)
            (setq tool (cons (list a_max min_doc) tool)))
            (setq tool (cons 0 tool)))
            (setq finish (cons tool finish)))
            (setq finish finish)))
      (setq finish finish))
); END OF FINISHING PROCESS FOR HOLE MAKING

;; ROUGHING PROCESS FOR HOLE MAKING
(defun rough_hole (n_goal a_max min_doc / amin tool len r l d
                        feat)
      (setq feat (get_ent2 2 n_goal))
      (setq l (get_ent2 42 n_goal))
      (setq d (get_ent2 41 n_goal))
      (setq r (get_ent2 151 n_goal))
      (if (= r "") (setq r 12.5) (setq r (atof r)))
            (cond
            ((or (= feat "DRILL_MARK") (= feat "FLAT_BUTTOM"))
                  (setq len (+ 3 l (/ d 3))))
            ((= feat "THROUGH") (setq len (+ l 6 (/ d 3)))))
; variable finish is list of possible finish process
      (setq rough '())

; TWIST DRILL
      (setq amin (cal_amin_twist d l min_doc))
      (if   (and
            (or (= feat "THROUGH") (= feat "DRILL_MARK"))
            (< (/ l d) 10)
            (>= d 10.5)
            (<= d 38)
            (> a_max amin))
            (progn
            (setq d_max (fix (- d (* 2 amin))))
            (setq tool (t_drill d_max len l r hbn))
            (if (/= tool nil)
            (progn
```

```
            (setq rough (cons tool rough)))
            (setq rough rough)))
            (setq rough rough))

; INSERT DRILL
      (setq amin (cal_amin_insert d l min_doc))
      (if   (and
            (or (= feat "THROUGH") (= feat "FLAT_BUTTOM") (= feat
                  "DRILL_MARK"))
            (< (/ l d) 3.5)
            (>= d 17.5)
            (<= d 100)
            (> a_max amin))
            (progn
            (setq d_max (fix (- d (* 2 amin))))
            (setq tool (i_drill d_max len l r hbn))
      (if (/= tool nil)
            (progn
            (setq rough (cons tool rough)))
            (setq rough rough)))
            (setq rough rough))

; SOLID CARBIDE DRILL
      (setq amin (cal_amin_solid d l min_doc))
      (if   (and
            (or (= feat "THROUGH") (= feat "DRILL_MARK"))
            (< (/ l d) 2)
            (>= d 5)
            (<= d 20)
            (> a_max amin))
            (progn
            (setq d_max (fix (- d (* 2 amin))))
            (setq tool (sc_drill d_max len l r hbn))
      (if (/= tool nil)
            (progn
            (setq rough (cons tool rough)))
            (setq rough rough)))
            (setq rough rough))

; CORE DRILL
      (setq amin (cal_amin_core d l min_doc))
      (if   (and
            (or (= feat "THROUGH") (= feat "FLAT_BUTTOM")
                  (= feat "DRILL_MARK"))
            (< (/ l d) 10)
            (>= d 7.8)
            (<= d 31)
            (> a_max amin))
            (progn
            (setq d_max (fix (- d (* 2 amin))))
            (setq tool (c_drill d_max len l r hbn))
      (if (/= tool nil)
            (progn
            (setq rough (cons tool rough)))
            (setq rough rough)))
            (setq rough rough))
```

```
; Rough boring
      (setq amin (cal_amin_rbore d l min_doc))
      (if   (and
            (or (= feat "THROUGH") (= feat "FLAT_BUTTOM")
                  (= feat "DRILL_MARK"))
            (< (/ l d) 10)
            (>= d 3)
            (<= d 266)
            (> a_max amin))
            (progn
            (setq d_max (- d (* 2 amin)))
            (setq tool (r_bore d_max len l r hbn))
      (if (/= tool nil)
            (progn
            (setq rough (cons tool rough)))
            (setq rough rough)))
            (setq rough rough))
); END OF ROUGH_KB_HOLE


; Calculating amin for twist drill
;
(defun cal_amin_twist (dia length min_doc / i1 i2 i3 i4 i5 i6 ii1
                        ii2 ii3 ii4 ii5 ii6 ii7)
      (setq i1 (+ 0.05 (* 0.015 (expt dia 0.7))))
      (setq i2 (* 4 12.5 (expt 10.0 -3)))
      (cond ((< dia 6.0)  (setq i3 0.04))
            ((< dia 10.0) (setq i3 0.05))
            ((< dia 18.0) (setq i3 0.06))
            ((< dia 30.0) (setq i3 0.08))
            ((> dia 30.0) (setq i3 0.10))
      )
      (setq ii1 (+ 0.05 (* 0.0005 (expt (/ length dia) 3.0))))
      (setq ii2 0.1)
      (setq ii3 (+ 0.08 (* 0.0006 (expt (/ length dia) 2.0))))
      (setq ii4 (* 0.025 (expt dia 0.3)))
      (setq ii5 (* 0.05 (expt dia 0.3)))
      (setq i4 (expt (+ (expt ii1 2.0) (expt ii2 2.0) (expt ii3
           2.0) (expt ii4 2.0) (expt ii5 2.0)) 0.5))
      (cond ((< dia 6.0)  (setq ii6 0.18))
            ((< dia 10.0) (setq ii6 0.20))
            ((< dia 18.0) (setq ii6 0.22))
            ((< dia 30.0) (setq ii6 0.24))
            ((> dia 30.0) (setq ii6 0.28))
      )
      (cond ((< dia 6.0)  (setq ii7 0.0024))
            ((< dia 10.0) (setq ii7 0.001))
            ((< dia 18.0) (setq ii7 0.001))
            ((< dia 30.0) (setq ii7 0.001))
            ((> dia 30.0) (setq ii7 0.000))
      )
      (setq i5 (expt (+ (expt ii6 2.0) (expt ii7 2.0)) 0.5))
      (setq i6 min_doc)
      (+ i1 i2 i3 i4 i5 i6)
)
```

```
; Calculating amin for solid carbide drill
(defun cal_amin_solid (dia length min_doc / i1 i2 i3 i4 i5 i6 ii1
                       ii2 ii3 ii4 ii5 ii6 ii7)
      (setq i1 0.05)
      (setq i2 (* 4 12.5 (expt 10 -3)))
      (setq i3 0.07)
      (setq ii1 0.02)
      (setq ii2 0.04)
      (setq ii3 0.05)
      (setq ii4 0.03)
      (setq ii5 0.06)
      (setq i4 (expt (+ (expt ii1 2) (expt ii2 2) (expt ii3 2)
           (expt ii4 2) (expt ii5 2)) 0.5))
      (setq ii6 0.05)
      (setq ii7 0)
      (setq i5 (expt (+ (expt ii6 2) (expt ii7 2)) 0.5))
      (setq i6 min_doc)
      (+ i1 i2 i3 i4 i5 i6)
)

; Calculating amin for insert drill
(defun cal_amin_insert (dia length min_doc / i1 i2 i3 i4 i5 i6
                        ii1 ii2 ii3 ii4 ii5 ii6 ii7)
      (setq i1 0.3)
      (setq i2 (* 4 12.5 (expt 10 -3)))
      (cond ((< dia 18) (setq i3 0.06))
           ((< dia 30) (setq i3 0.08))
           ((> dia 30) (setq i3 0.10))
      )
      (setq ii1 0.03)
      (setq ii2 0.07)
      (setq ii3 0.08)
      (setq ii4 0.05)
      (setq ii5 0.1)
      (setq i4 (expt (+ (expt ii1 2) (expt ii2 2) (expt ii3 2)
           (expt ii4 2) (expt ii5 2)) 0.5))
      (cond ((< dia 18) (setq ii6 0.22))
           ((< dia 30) (setq ii6 0.24))
           ((> dia 30) (setq ii6 0.28))
      )
      (setq ii7 0)
      (setq i5 (expt (+ (expt ii6 2) (expt ii7 2)) 0.5))
      (setq i6 min_doc)
      (+ i1 i2 i3 i4 i5 i6)
)

; Calculation amin for core drill
(defun cal_amin_core (dia length min_doc / i1 i2 i3 i4 i5 i6 ii1
                      ii2 ii3 ii4 ii5 ii6 ii7)
      (setq i1 (+ 0.03 (* 0.012 (expt dia 0.5))))
      (setq i2 (* 4 1.575 (expt 10 -3)))
      (cond ((< dia 6)  (setq i3 0.025))
           ((< dia 10) (setq i3 0.03))
           ((< dia 18) (setq i3 0.04))
           ((< dia 30) (setq i3 0.05))
           ((> dia 30) (setq i3 0.06))
```

```
      )
      (setq ii1 0)
      (setq ii2 0.05)
      (setq ii3 0)
      (setq ii4 0)
      (setq ii5 0)
      (setq i4 (expt (+ (expt ii1 2) (expt ii2 2) (expt ii3 2)
            (expt ii4 2) (expt ii5 2)) 0.5))
      (setq ii6 0)
      (cond ((< dia 6)   (setq ii7 0.002))
            ((< dia 10)  (setq ii7 0.001))
            ((< dia 18)  (setq ii7 0.001))
            ((< dia 30)  (setq ii7 0.001))
            ((> dia 30)  (setq ii7 0.000))
      )
      (setq i5 (expt (+ (expt ii6 2) (expt ii7 2)) 0.5))
      (setq i6 min_doc)
      (+ i1 i2 i3 i4 i5 i6)
)

(defun cal_amin_rbore (dia length min_doc / i1 i2 i3 i4 i5 i6 ii1
                       ii2 ii3 ii4 ii5 ii6 ii7)
      (setq i1 (* 0.003 (expt dia 0.45)))
      (setq i2 (* 4 0.8 (expt 10 -3)))
      (setq i3 0.03)
      (setq ii1 0.01)
      (setq ii2 0.01)
      (setq ii3 0.015)
      (setq ii4 0.01)
      (setq ii5 0.02)
      (setq i4 (expt (+ (expt ii1 2) (expt ii2 2) (expt ii3 2)
            (expt ii4 2) (expt ii5 2)) 0.5))
      (setq ii6 0.01)
      (setq ii7 0)
      (setq i5 (expt (+ (expt ii6 2) (expt ii7 2)) 0.5))
      (setq i6 min_doc)
      (+ i1 i2 i3 i4 i5 i6)
)

(defun kb_mill (goal / finish)
      (setq finish '())
      (setq feat (get_ent2 2 goal))
      (setq width (get_ent2 41 goal))
      (setq depth (get_ent2 151 goal))
      (if (= depth "") (setq depth 3) (setq depth (atof depth)))
      (setq lens (get_ent2 150 goal))
      (if (= lens "") (setq lens width) (setq lens (atof lens)))
      (if (= feat "FACE")
            (progn
     (if (> width lens)
            (progn
            (setq dia (* lens 1.2))
            (setq ae lens)
            (setq len width))
            (progn
            (setq dia (* width 1.2))
```

```
            (setq ae width)
            (setq len lens)))
            (princ len) (princ " depth ") (princ depth)
            )) ; END OF PROGN AND IF
      (if (or (= feat "SSFACEL") (= feat "SSFACER"))
            (progn
            (setq dia (* width 1.2))
            (setq len lens)
            (setq ae width)
            (setq depth (get_ent2 42 goal))
            (princ len) (princ " depth ") (princ depth)
            )) ; END OF PROGN AND IF

; CHAMFER
      (if (= feat "CHAMFER")
            (progn
            (setq tool (chamfer (get_ent2 41 goal) (atof (get_ent2
               154 goal)) (get_ent2 42 goal) 12.5 hbn))
      (if (/= tool nil)
            (progn
            (setq tool (cons 0 tool))
            (setq finish (cons tool finish)))
            (setq finish finish)))
            (setq finish finish))

; COROMILL 245 FOR FACE MILLING
      (if (and (= feat "FACE")
            (<= dia 250)
            (>= dia 32))
            (progn
;           (princ depth) (getstring)
            (setq tool (coro245 dia len hbn depth cmc ae))
      (if (/= tool nil)
            (progn
      ;DECISION ON INTERMEDIATE PROCESS
            (setq tool (cons 0 tool))
            (setq finish (cons tool finish)))
            (setq finish finish)))
            (setq finish finish))

; coromill 290 for square shoulder face milling and face milling
      (if (and (or (= feat "SSFACEL") (= feat "SSfACER"))
            (<= dia 250)
            (>= dia 40))
            (progn
            (setq depth (get_ent2 42 goal))
            (setq tool (coro290 dia len hbn depth cmc ae))
      (if (/= tool nil)
            (progn
      ; DECISION ON INTERMEDIATE PROCESS
            (setq tool (cons 0 tool))
            (setq finish (cons tool finish)))
            (setq finish finish)))
            (setq finish finish))
); END OF FUNCTION
```

ภาคผนวก ง.

ข้อมูลเครื่องมือตัด

1. Boring

```
("C4_39137A_08" 10.0 14.0 24 50.0)
("C4_39137A_11" 11.0 17.0 33.0 59.0)
("C4_39137A_14" 14.0 20.0 40.0 65.0)
("C4_39137A_17" 17.0 23.0 40.0 65.0)
("C4_39137A_20" 20.0 26.0 40.0 65.0)
("C3_39138A_024086A" 25.0 38.5 64.0 90.0)
("C4_39138A_033106A" 34.5 53.5 80.0 110.0)
("C5_39138A_046140A" 49.5 74.5 115.0 145.0)
("C6_39138A_065060A" 70.5 103.5 200.0 65)
("C3_39139A_024209A" 25.0 53.5 150.0 213.0)
("C5_39139A_033326A" 25.0 53.5 207.0 329.0)
("C5_39139A_046336A" 49.5 74.5 297.0 341.0)
("C6_39139A_065407A" 70.5 103.5 200.0 412.0)
("C6_39138A_096045A" 99.5 266.5 189.0 115.0)
("C8_39138A_096045A" 99.5 266.5 240.0 115.0)
```

2. Core Drill

```
("CD0780" 7.8 100 5.6 181)
("CD0800" 8.0 100 5.6 181)
("CD0870" 8.7 107 6.3 188)
("CD0880" 8.8 107 6.3 188)
("CD0900" 9.0 107 6.3 188)
("CD0940" 9.4 107 6.3 188)
("CD0970" 9.7 116 7.0 197)
("CD0980" 9.8 116 7.0 197)
("CD1000" 10.0 116 7.0 197)
("CD1010" 10.1 116 7.0 197)
("CD1030" 10.3 116 7.0 197)
("CD1050" 10.5 116 7.0 197)
("CD1075" 10.75 125 7.7 206)
("CD1100" 11.0 125 7.7 206)
("CD1175" 11.75 134 8.4 215)
("CD1200" 12.0 134 8.4 215)
("CD1220" 12.2 134 8.4 215)
("CD1275" 12.75 134 9.1 215)
("CD1300" 13.0 134 9.1 215)
("CD1375" 13.75 142 9.8 223)
("CD1400" 14.0 142 9.8 223)
("CD1475" 14.75 147 10.5 245)
("CD1500" 15.0 147 10.5 245)
("CD1575" 15.75 153 11.2 251)
("CD1600" 16.0 153 11.2 251)
("CD1675" 16.75 159 11.9 257)
("CD1700" 17.0 159 11.9 257)
("CD1775" 17.75 165 12.6 263)
("CD1800" 18.0 165 12.6 263)
("CD1870" 18.7 171 13.3 269)
("CD1900" 19.0 171 13.3 269)
("CD1970" 19.7 177 14.0 275)
("CD2000" 20.0 177 14.0 275)
("CD2070" 20.7 184 14.6 282)
("CD2100" 21.0 184 14.6 282)
("CD2170" 21.7 191 15.3 289)
("CD2200" 22.0 191 15.3 289)
("CD2270" 22.7 198 16.0 296)
("CD2300" 23.0 198 16.0 296)
("CD2370" 23.7 206 17.3 327)
("CD2400" 24.0 206 17.3 327)
("CD2470" 24.7 206 17.3 327)
("CD2500" 25.0 206 17.3 327)
```

```
("CD2570" 25.7 214 18.0 335)
("CD2600" 26.0 214 18.0 335)
("CD2670" 26.7 222 19.3 343)
("CD2700" 27.0 222 19.3 343)
("CD2770" 27.7 222 19.3 343)
("CD2800" 28.0 222 19.3 343)
("CD2870" 28.7 230 20.0 351)
("CD2900" 29.0 230 20.0 351)
("CD2970" 29.7 230 20.0 351)
("CD3000" 30.0 230 20.0 351)
("CD3060" 30.6 239 21.0 360)
("CD3100" 31.0 239 21.0 360)
```

3. Insert Drill

```
("ID1600" 16.0 19.0 50 123)
("ID1900" 19.0 22.0 60 143)
("ID2200" 22.0 24.0 60 143)
("ID2400" 24.0 27.0 75 158)
("ID2700" 27.0 30.0 85 168)
("ID3000" 30.0 34.0 100 183)
("ID3400" 34.0 39.0 100 183)
("ID3900" 39.0 42.0 120 203)
("ID4200" 42.0 45.0 135 218)
("ID4500" 45.0 48.0 135 218)
("ID4800" 48.0 53.0 135 218)
("ID5300" 53.0 55.0 150 233)
("ID5500" 55.0 60.0 165 248)
("ID6000" 60.0 66.0 165 248)
("ID6600" 66.0 69.0 165 248)
("ID6900" 69.0 73.0 190 273)
("ID7300" 73.0 79.0 190 273)
("ID7900" 79.0 82.0 190 273)
```

4. Boring

```
("C3_39168A_1_021068A" 18.0 32.0 128 81)
("C3_39168A_2_026084A" 30.0 38.0 128.0 97.0)
("C3_39168A_3_032034A" 37.0 47.0 128.0 50.0)
("C4_39168A_4_040041A" 46.0 56.0 160.0 65.0)
("C5_39168A_5_050044A" 55.0 70.0 200.0 70.0)
("C5_39168A_6_063045A" 69.0 84.0 200.0 75.0)
("C6_39168A_6_063045A" 69.0 84.0 252.0 75.0)
("C5_39168A_6_063045A" 83.0 101.0 200.0 86.0)
("C6_39168A_6_063045A" 83.0 101.0 252.0 86.0)
("C8_39168A_7_080060B" 99.0 150.0 320.0 100.0)
("C8_39168A_8_110080B" 148.0 270.0 320.0 125.0)
```

5. Reamer

```
("R070" 7.0 31 150)
("R075" 7.5 31 150)
("R080" 8.0 33 156)
("R085" 8.5 33 156)
("R090" 9.0 36 162)
("R095" 9.5 36 162)
("R100" 10.0 38 168)
("R105" 10.5 38 168)
("R110" 11.0 41 175)
("R115" 11.5 41 175)
("R120" 12.0 44 182)
("R125" 12.5 44 182)
("R130" 13.0 44 182)
("R135" 13.5 47 189)
("R140" 14.0 47 189)
```

```
("R145" 14.5 50 204)
("R150" 15.0 50 204)
("R155" 15.5 52 210)
("R160" 16.0 52 210)
```

6. **Solid Carbide Drill**

```
("SCD0300" 3.0 23 66)
("SCD0320" 3.2 23 66)
("SCD0330" 3.3 23 66)
("SCD0350" 3.5 23 66)
("SCD0370" 3.7 23 66)
("SCD0380" 3.8 29 74)
("SCD0400" 4.0 29 74)
("SCD0420" 4.2 29 74)
("SCD0450" 4.5 29 74)
("SCD0465" 4.65 29 74)
("SCD0480" 4.8 35 82)
("SCD0500" 5.0 35 82)
("SCD0510" 5.1 35 82)
("SCD0520" 5.2 35 82)
("SCD0550" 5.5 35 82)
("SCD0555" 5.55 35 82)
("SCD0580" 5.8 35 82)
("SCD0600" 6.0 35 82)
("SCD0630" 6.3 43 91)
("SCD0650" 6.5 43 91)
("SCD0660" 6.6 43 91)
("SCD0680" 6.8 43 91)
("SCD0700" 7.0 43 91)
("SCD0740" 7.4 43 91)
("SCD0750" 7.5 43 91)
("SCD0780" 7.8 43 91)
("SCD0800" 8.0 43 91)
("SCD0840" 8.4 49 103)
("SCD0850" 8.5 49 103)
("SCD0900" 9.0 49 103)
("SCD0930" 9.3 49 103)
("SCD0950" 9.5 49 103)
("SCD0980" 9.8 49 103)
("SCD1000" 10.0 49 103)
("SCD1020" 10.2 56 118)
("SCD1050" 10.5 56 118)
("SCD1100" 11.0 56 118)
("SCD1120" 11.2 56 118)
("SCD1150" 11.5 56 118)
("SCD1180" 11.8 56 118)
("SCD1200" 12.0 56 118)
("SCD1250" 12.5 60 124)
("SCD1300" 13.0 60 124)
("SCD1310" 13.1 60 124)
("SCD1350" 13.5 60 124)
("SCD1380" 13.8 60 124)
("SCD1400" 14.0 60 124)
("SCD1420" 14.2 63 133)
("SCD1450" 14.5 63 133)
("SCD1500" 15.0 63 133)
("SCD1510" 15.1 63 133)
("SCD1550" 15.5 63 133)
("SCD1580" 15.8 63 133)
("SCD1600" 16.0 63 133)
("SCD1650" 16.5 71 143)
```

```
("SCD1690" 16.9 71 143)
("SCD1700" 17.0 71 143)
("SCD1750" 17.5 71 143)
("SCD1770" 17.7 71 143)
("SCD1800" 18.0 71 143)
("SCD1850" 18.5 77 153)
("SCD1875" 18.75 77 153)
("SCD1890" 18.9 77 153)
("SCD1900" 19.0 77 153)
("SCD1950" 19.5 77 153)
("SCD2000" 20.0 77 153)
```

7. Twist Drill

```
("TD1050" 10.5 87 168)
("TD1060" 10.6 87 168)
("TD1100" 11.0 94 175)
("TD1110" 11.1 94 175)
("TD1150" 11.5 94 175)
("TD1160" 11.6 94 175)
("TD1170" 11.7 94 175)
("TD1175" 11.75 94 175)
("TD1180" 11.8 94 175)
("TD1190" 11.9 101 182)
("TD1191" 11.91 101 182)
("TD1200" 12.0 101 182)
("TD1250" 12.5 101 182)
("TD1300" 13.0 101 182)
("TD1340" 13.4 108 189)
("TD1349" 13.49 108 189)
("TD1380" 13.8 108 189)
("TD1390" 13.9 108 189)
("TD1400" 14.0 108 189)
("TD1410" 14.1 114 212)
("TD1420" 14.2 114 212)
("TD1460" 14.6 114 212)
("TD1508" 15.08 120 218)
("TD1550" 15.5 120 218)
("TD1560" 15.6 120 218)
("TD1600" 16.0 120 218)
("TD1610" 16.1 125 223)
("TD1650" 16.5 125 223)
("TD1680" 16.8 125 223)
("TD1690" 16.9 125 223)
("TD1730" 17.3 130 228)
("TD1740" 17.4 130 228)
("TD1746" 17.46 130 228)
("TD1780" 17.8 130 228)
("TD1790" 17.9 130 228)
("TD1800" 18.0 130 228)
("TD1830" 18.3 135 233)
("TD1840" 18.4 135 233)
("TD1850" 18.5 135 233)
("TD1860" 18.6 135 233)
("TD1900" 19.0 135 233)
("TD1905" 19.05 140 238)
("TD1910" 19.1 140 238)
("TD1950" 19.5 140 238)
("TD1960" 19.6 140 238)
("TD1970" 19.7 140 238)
("TD1975" 19.75 140 238)
("TD2000" 20.0 140 238)
```

```
("TD2010" 20.1 145 243)
("TD2020" 20.2 145 243)
("TD2025" 20.25 145 243)
("TD2020" 20.2 145 243)
("TD2060" 20.6 145 243)
("TD2064" 20.64 145 243)
("TD2070" 20.7 145 243)
("TD2120" 21.2 145 243)
("TD2130" 21.3 150 248)
("TD2140" 21.4 150 248)
("TD2190" 21.9 150 248)
("TD2200" 22.0 150 248)
("TD2210" 22.1 150 248)
("TD2260" 22.6 155 253)
("TD2270" 22.7 155 253)
("TD2280" 22.8 155 253)
("TD2330" 23.3 155 276)
("TD2340" 23.4 155 276)
("TD2350" 23.5 155 276)
("TD2400" 24.0 160 281)
("TD2410" 24.1 160 281)
("TD2460" 24.6 160 281)
("TD2470" 24.7 160 281)
("TD2520" 25.2 165 286)
("TD2530" 25.3 165 286)
("TD2580" 25.8 165 286)
("TD2590" 25.9 165 286)
("TD2640" 26.4 165 286)
("TD2650" 26.5 165 286)
("TD2775" 27.75 170 291)
("TD2800" 28.0 170 291)
("TD2975" 29.75 175 296)
("TD3000" 30.0 175 296)
("TD3025" 30.25 180 301)
("TD3175" 31.75 185 306)
("TD3200" 32.0 185 334)
("TD3250" 32.5 185 334)
("TD3500" 35.0 190 339)
("TD3550" 35.5 190 339)
("TD3600" 36.0 195 344)
("TD3650" 36.5 195 344)
```

ภาคผนวก จ.

คำนิยามศัพท์เกี่ยวกับการวางแผนกระบวนการผลิต

AutoCAD ซอฟท์แวร์สำหรับเขียนแบบ 2 มิติที่เป็นที่นิยม และมักใช้ร่วมกับระบบคอมพิวเตอร์ช่วยในการออกแบบอื่นๆได้

AutoLISP เป็นโปรแกรมภาษาชั้นสูงสำหรับ AutoCAD โดยมีต้นกำเนิดมาจากโปรแกรมภาษา LISP AutoLISP ถูกใช้สำหรับเพิ่มความสามารถของโปรแกรม AutoCAD ให้เหมาะสมกับงานของผู้ใช้แต่ละคน

CAD (Computer Aided Design) เป็นเทคโนโลยีในการนำคอมพิวเตอร์มาช่วยในกระบวนการออกแบบ รวมทั้งวิเคราะห์งานออกแบบ เช่น Finite Element เป็นต้น

Cutting Tools เครื่องมือที่ใช้ในการตัดเฉือนเนื้อโลหะ เช่น มีดกัด ดอกสว่าน และมีดกลึง เป็นต้น

Feature ลักษณะรูปร่างทางเรขาคณิต หรือหน้าที่การทำงาน หรือคุณสมบัติของวัตถุที่ใช้ในการอธิบายการทำหน้าที่หรือพฤติกรรมของวัตถุนั้น

Finishing Tools เครื่องมือสำหรับใช้ตบแต่งให้วัสดุหรือชิ้นงานเสร็จเรียบร้อยสวยงามตามแบบชิ้นงานที่กำหนด

IGES (Initial Graphics Exchange Specification) เป็น Protocol มาตรฐานสำหรับถ่ายโอนแฟ้มข้อมูลสำหรับ CAD/CAM ระหว่างระบบคอมพิวเตอร์อื่นๆ

Machining Operation การทำให้เรียบ หรือการตัดเฉือนเนื้อโลหะออกด้วยการใช้เครื่องจักร เช่น เครื่องกัด เครื่องกลึง และเครื่องไส เป็นต้น

Machine Tool เครื่องจักรกลประเภทใดก็ตามที่ใช้ในงานช่าง เช่น เครื่องเจาะ เครื่องกลึง และเครื่องกัด เป็นต้น

Roughing Tools เครื่องมือสำหรับใช้กัดผิวหยาบ

Viewport ขอบเขตของรูปภาพที่สามารถแสดงผลบนหน้าจอแสดงผล

Wire Frame Graphics เป็นเทคนิคของระบบคอมพิวเตอร์ช่วยในการออกแบบสำหรับการแสดงภาพวัตถุ 3 มิติบนหน้าจอคอมพิวเตอร์โดยการแสดงเฉพาะเส้นขอบของวัตถุ

ภาคผนวก ฉ.

การดึงความรู้จากผู้เชี่ยวชาญ

การดึงความรู้จากผู้เชี่ยวชาญเป็นขั้นตอนที่สำคัญขั้นตอนหนึ่งของการพัฒนาระบบผู้เชี่ยวชาญ เทคนิคต่างๆที่นำมาใช้มักขึ้นอยู่กับชนิดของปัญหาที่กำลังศึกษา สำหรับการพัฒนาระบบคอมพิวเตอร์เพื่อช่วยในการวางแผนกระบวนการผลิตจะอาศัยการซักถามจากผู้เชี่ยวชาญในการวางแผนกระบวนการผลิตของบริษัทฯ โดยหัวข้อในการซักถามจะประกอบไปด้วย

- ตัวแปรต่างๆที่มีผลต่อการเลือกชนิดกระบวนการผลิต เครื่องมือตัด และกำหนดค่าสภาวะการตัดเฉือน
- ลักษณะการกำหนดรายละเอียดของชิ้นงานจากแบบทางวิศวกรรมที่ได้รับจากลูกค้า
- ข้อมูล เอกสาร และขั้นตอนที่จำเป็นในการวางแผนกระบวนการผลิต
- แผนกระบวนการผลิตเดิมของชิ้นงานที่ทำการศึกษา

นอกจากข้อมูลที่ได้รับจากการซักถามแล้ว ยังต้องศึกษาหาข้อมูลจากหนังสือคู่มือการตัดเฉือนโลหะ และแคตตาล็อคเครื่องมือตัดของบริษัทฯต่างๆที่มีการเลือกใช้ เพื่อสำหรับกำหนดรายละเอียดของข้อจำกัดต่างๆของกระบวนการผลิตแต่ละชนิด

ในการเลือกชนิดกระบวนการผลิตจะทำการวิเคราะห์ข้อมูลความสามารถของกระบวนการผลิตแต่ละชนิดและกำหนดเป็นกฎเกณฑ์ต่างๆและเขียนให้อยู่ในรูปแบบของ Production Rules ได้แก่ IF-THEN-ELSE และต้องทำการตรวจสอบความถูกต้องของกฎเกณฑ์ต่างๆกับผู้เชี่ยวชาญอีกครั้งหนึ่งก่อนนำไปใช้พัฒนาโปรแกรม โดยตัวอย่างกฎเกณฑ์ที่มีความสำคัญมีดังนี้

IF (The feature is a through hole), (diameter < 38 mm.), (length / diameter < 10), (3.125 μm< surface finish < 12.5 μm), (Roundness > 0.1 mm.) THEN (Select twist drill process) and (Initial process = none)

ถ้า ต้องการสร้างรูทะลุขนาดเส้นผ่านศูนย์กลางไม่เกิน 38 มม.

ระดับความเรียบผิวอยู่ระหว่าง 3.125 – 12.5 ไมโครเมตร

ขนาดความลึกต่อขนาดเส้นผ่านศูนย์กลางน้อยกว่า 10

ค่าความเที่ยงความกลมมากกว่า 0.1 มม.

ดังนั้น ใช้ดอกสว่าน (Twist Drill) ในการสร้างรู และไม่จำเป็นต้องมีกระบวนการก่อนหน้า

IF (The feature is a through hole or flat buttom hole), (16 mm. < diameter < 82 mm.), (length / diameter < 3.5), (3.125 μm< surface finish < 12.5 μm), (Roundness > 0.05 mm.) THEN (Select insert drill process) and (Initial process = none)

ถ้า ต้องการสร้างรูทะลุ หรือรูที่มีก้นเรียบ
ขนาดเส้นผ่านศูนย์กลางระหว่าง 16 – 82 มม.
ระดับความเรียบผิวอยู่ระหว่าง 3.125 – 12.5 ไมโครเมตร
ขนาดความลึกต่อขนาดเส้นผ่านศูนย์กลางน้อยกว่า 3.5
ค่าความเที่ยงความกลมมากกว่า 0.05 มม.
ดังนั้น ใช้ Insert Drill ในการสร้างรู และไม่จำเป็นต้องมีกระบวนการก่อนหน้า

IF (The feature is a through hole),(3 mm.<diameter<20mm.),(length / diameter < 2), (2.25 μm< surface finish < 12.5 μm), (Roundness > 0.04 mm.) THEN (Select solid carbide drill process) and (Initial process = none)

ถ้า ต้องการสร้างรูทะลุ หรือรูที่ไม่กำหนดขนาดก้นรู
ขนาดเส้นผ่านศูนย์กลางระหว่าง 3 – 20 มม.
ระดับความเรียบผิวอยู่ระหว่าง 2.25 – 12.5 ไมโครเมตร
ขนาดความลึกต่อขนาดเส้นผ่านศูนย์กลางน้อยกว่า 2
ค่าความเที่ยงความกลมมากกว่า 0.04 มม.
ดังนั้น ใช้ Solid Carbide Drill ในการสร้างรู และไม่จำเป็นต้องมีกระบวนการก่อนหน้า

IF (The feature is a through hole or flat buttom hole),(7 mm.<diameter<10mm.), (length / diameter < 10),(1.575μm< surface finish < 6.25 μm), (Roundness>0.05 mm.) THEN (Select core drill process) (cal. a_{max})

ถ้า ต้องการสร้างรูทะลุ หรือรูที่มีก้นเรียบ
ขนาดเส้นผ่านศูนย์กลางระหว่าง 7 – 10 มม.
ระดับความเรียบผิวอยู่ระหว่าง 1.575 – 6.25 ไมโครเมตร
ขนาดความลึกต่อขนาดเส้นผ่านศูนย์กลางน้อยกว่า 10
ค่าความเที่ยงความกลมมากกว่า 0.05 มม.
ดังนั้น ต้องทำการเตรียมรูก่อนและใช้ Core Drill ในการสร้างรู
คำนวนค่าความลึกการตัดเฉือนสูงสุดของ Core Drill

IF (The feature is a through hole or flat buttom hole),(10 mm.<diameter<266.5mm.), (length / diameter < 10), (0.8 μm< surface finish < 12.5 μm), (Roundness > 0.01 mm.) THEN (Select fine boring process) (cal. a_{max})

ถ้า ต้องการสร้างรูทะลุ หรือรูที่มีก้นเรียบ
ขนาดเส้นผ่านศูนย์กลางระหว่าง 10 – 266.5 มม.
ระดับความเรียบผิวอยู่ระหว่าง 0.8 – 12.5 ไมโครเมตร
ขนาดความลึกต่อขนาดเส้นผ่านศูนย์กลางน้อยกว่า 10
ค่าความเที่ยงความกลมมากกว่า 0.01 มม.
ดังนั้น ต้องทำการเตรียมรูก่อนและใช้ Fine Boring ในการสร้างรู
คำนวนค่าความลึกการตัดเฉือนสูงสุดของ Fine Boring

IF (The feature is a through hole or flat buttom hole),(7 mm.<diameter<16mm.), (length / diameter < 15), (0.4 μm< surface finish < 1.575 μm), (Roundness > 0.01 mm.) THEN (Select Reamer process) (cal. a_{max})

ถ้า ต้องการสร้างรูทะลุ หรือรูที่มีก้นเรียบ
ขนาดเส้นผ่านศูนย์กลางระหว่าง 7 – 16 มม.
ระดับความเรียบผิวอยู่ระหว่าง 0.4 – 1.575 ไมโครเมตร
ขนาดความลึกต่อขนาดเส้นผ่านศูนย์กลางน้อยกว่า 15
ค่าความเที่ยงความกลมมากกว่า 0.01 มม.
ดังนั้น ต้องทำการเตรียมรูก่อนและใช้ Reamer ในการสร้างรู
คำนวนค่าความลึกการตัดเฉือนสูงสุดของ Reamer

IF (The process is reamer) and (reamer a_{max} > twist drill a_{min}) THEN (Initial process is twist drill)

ถ้า สร้างรูโดย Reamer และ ค่าความลึกการตัดเฉือนสูงสุดของ Reamer มากกว่าค่าความลึกการตัดเฉือนต่ำสุดของ Twist Drill
ดังนั้น ทำการสร้างรูด้วย Twist Drill และตามด้วย Reamer

IF (The feature is face) THEN (Select coromill245 process) and (Diameter>1.2width)

ถ้า ต้องการกัดปาดหน้าเรียบ ดังนั้น ใช้เครื่องมือ Coromill245 ขนาดเส้นผ่านศูนย์กลางมากกว่า 1.2 เท่าของความกว้างที่ต้องการ

IF (The process is coromill245) and (Work piece material is cast iron) THEN (Select Insert grade 3020)

ถ้า ใช้เครื่องมือ Coromill245 และวัสดุชิ้นงานเป็นเหล็กหล่อ

ดังนั้น ใช้เม็ดมีดชนิด 3020

IF (The process is coromill245) and (Work piece material is aluminium) THEN (Select Insert grade H13A)

ถ้า ใช้เครื่องมือ Coromill245 และวัสดุชิ้นงานเป็นอลูมิเนียม

ดังนั้น ใช้เม็ดมีดชนิด H13A

IF (The feature is square shoulder face) THEN (Select coromill290 process) and (diameter > 1.2 width)

ถ้า ต้องการกัดปาดหน้ามีป่ามุมฉาก **ดังนั้น** ใช้เครื่องมือ Coromill290 ขนาดเส้นผ่านศูนย์กลางมากกว่า 1.2 เท่าของความกว้างที่ต้องการ

IF (The process is coromill290) and (Work piece material is steel) THEN (Select Insert grade 4030)

ถ้า ใช้เครื่องมือ Coromill290 และวัสดุชิ้นงานเป็นเหล็กเหนียว

ดังนั้น ใช้เม็ดมีดชนิด 4030

ประวัติผู้วิจัย

นายกสานต์ ปิ่นเวหาส์ เกิดวันที่ 5 ตุลาคม พ.ศ. 2516 ที่จังหวัดกรุงเทพมหานคร สำเร็จการศึกษาปริญญาตรีวิศวกรรมศาสตร์บัณฑิต สาขาวิชาวิศวกรรมการผลิต ภาควิชาวิศวกรรมการผลิต คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าพระนครเหนือ ในปีการศึกษา 2537 และเข้าศึกษาต่อในหลักสูตรวิศวกรรมศาสตร์มหาบัณฑิต ที่จุฬาลงกรณ์มหาวิทยาลัย เมื่อ พ.ศ. 2538