Addendum for Acromill 15.00

Objective

This document is to serve as an addendum until the new version of the AcroMill/AcroCut manual is released.

Differences between previous versions and version 15.00

Global parameters are now referenced by #1,#2,... instead of P1,P2....
Files can now be included into programs using the $I directive.
G50,G51 scale commands have been added.
G52 local co-ordinate system command has been added.
G53 machine co-ordinate system command has been added.
G54....G59 work co-ordinate system commands have been added.
G65 macro call command has been added.
G66 modal macro call command has been added.
G67 modal macro call cancel command has been added.
G69 rotation cancel command has been added.
G73,G74,G80...G89 can cycle commands have been added.
WHILE... END command has been added.
G31 Block Skip/Probing Cycle command has been added.
Support for upto 2 I/O expansion boards for a total of 256 I/O has now been added.
G90 and G91 modes can be mixed within a program.
Software now automatically recognizes ACR2000 or ACR8000 controller.
All the old programs that used to run in previous versions will still
run under version 15.00.
Exceptions are:

The SCALE command. This command is now implemented using the G51 command and allows Scaling centers to be specified. Scale commands should be changed to use G51 command instead. Scaling center is always an absolute position.

New Commands

"IF" Command

Syntax: IF <expression> {THEN}<stmt>
where <stmt> ::= (GOTO_STMT | RS274DSTMT | ASSIGN_STMT).

This command has been enhanced to allow the following variations.

IF (#1<400) THEN #5=6
IF (#4=5) THEN X10
IF (#10>#20) THEN GOTO 100

"WHILE" Command

Syntax: WHILE <expression> DO {<number>} <stmt> END {<number>}

This command is new and has the following variations

#1=1
#5=100
WHILE (#1<#5) DO
X(#1)
#1=#1+1
END

"$I" Include File Compiler Directive

Syntax: $I <filename>

This command allows inserting a program file as part of any other program file. Most common use of this file is to be able to keep most widly used programs or subroutines in a separate file. Then this file can be INCLUDED into any program that wishes to use these subroutines. This is how the CAN CYCLES are implemented. The CAN CYCLES are kept in a file called CYCLES.TXT. If the user does not need to use these can cycles, this file is not even used. If the cycles are to be used , the user can specify the command

$I CYCLES.TXT

as the very last command of the program. This will cause the Can Cycles to be linked to his file.

"G9" Exact Stop Mode

Syntax: G9 axes <Expression>

This command forces INPOSITION check after every move. This is useful for hole drilling programs where the tool is not contouring continuously. Move commands commanded with G9 decelerate at the end point and check for inposition. This mode is not necessary for G0 (Rapid) moves as in that mode, the inposition check is done automatically. This function is used when a sharp corner is required during a contouring cut. This mode is only valid for the block that has the G9 and is not modal. (See G61,G62,G63,G64 for modal information.)
Example:

G9 G1 X4 Y4

This will result in the axes to decelerate by the time they reach X4, Y4 and then perform inposition check.

"G10 L2" Offset Load

Syntax: G10 L (1 | 11) P <expression_1> Axes_Assign

This command allows entering of work coordinate offset numbers. The following lines of code illustrates its use.

```
G10L2P1X00Y0Z0  (COORDINATE #1 AT 0,0,0)
G10L2P2X05Y0Z0  (COORDINATE #2 AT 5,0,0)
G10L2P3X10Y0Z0  (COORDINATE #3 AT 10,0,0)
G10L2P4X10Y5Z0  (COORDINATE #4 AT 10,5,0)
G10L2P5X05Y5Z0  (COORDINATE #5 AT 5,5,0)
G10L2P6X00Y5Z0  (COORDINATE #6 AT 0,5,0)
```

"G20" or (G70) Inch Mode programming.

This command allows intrepeting programs as INCH mode programs. This command is modal. The control does not allow switching modes back and forth within a program.

"G21" or (G71) Metric Mode programming.

This command allows intrepeting programs as Metric mode programs. This command is modal. The control does not allow switching modes back and forth within a program.

"G27" Reference Position Return Check.

Syntax: G 27 {(Axis_Asign)*}
where Axis_Asign ::= axis <expression>

In this case, the machine is positioned to the specified Axis_Asign position. At this time, an input that is specified in the system parameters is checked. If the input is energized, the reference point return output is energized.

"G28" Return to Refernce position.

Syntax: G 28 {(Axis_Asign)*}
where Axis_Asign ::= axis <expression>

When this command is issued, the machine will first go to the Axis_Assign position and then return to the HOME (Reference point position) position. If this command is done before a manual HOME (reference position return) is done, the system will perform a HOME as if it was manually being done.

"G29" Return from reference position.

Syntax: G 29 {(Axis_Asign)*}
where Axis_Asign ::= axis <expression>

This command will move the machine to the point commanded by Axis_Asign but will position first to the intermediate point last commanded by the G28 command

"G30" Second Reference Point Return

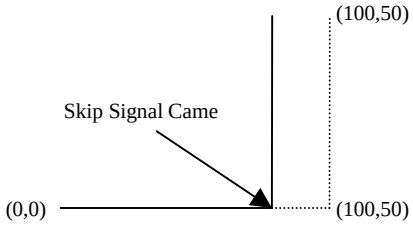
Syntax: G 30 P <expression> {(Axis_Asign)}
where Axis_Asign ::= axis <expression>,
the <expression> must equal to 2,3 or 4 and it specifies one of 4 reference points. If no P is specified, the default is 2.
This command positions the machine to the second reference position by first going to the commanded Axis_Asign point. The second reference position is set in the system parameters. This command only works if the machine was HOMED once after power up. G29 can still be used to return from the second reference point.

"G31" BLOCK SKIP on seeing Probe Input

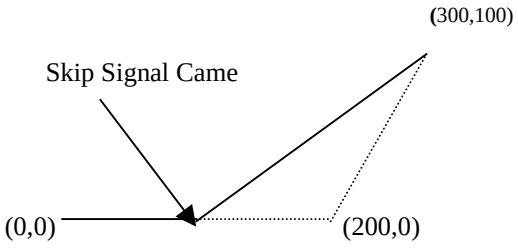
Syntax: G 31 {(Axis_Asign)*}
where Axis_Asign ::= axis <expression>

This command will cause the machine to start moving towards the Axis_Asign position. Before the machine reaches that point, if the BLOCK SKIP signal arrives, the remainder of the move is discarded and the machine will decelerate to a halt. Then the machine backs up to the precise location that the BLOCK SKIP signal was received at. At this time, the control will ignore the remainder of the command line and go on to the next block.

G31 G91 X100
Y50



G31 G90 X200
X300 Y100



At the instant that the Skip Signal comes, the encoder positions are stored in parameters #201,#202,#203,#204,#205,#206,#207,#208 for axis X,Y,Z……

This command is useful for Constant feed in Grinding machines and Tool Measurement with tactile sensor probe input. This is also useful for Digitizing molds.

AcroMill, AcroCut uses the INTCAP feature of the ACR2000/ACR8000 to implement the G31 probing cycle. System parameters in the AcroMill/AcroCut software allow the operator to specify what inputs the probe is wired to.

On the ACR8000
Input 24 is used to sample Axis0 and Axis1 (Usually X and Y)
Input 26 is used to sample Axis2 and Axis3 (Usually Z and A)
Input 28 is used to sample Axis4 and Axis5 (Usually B and C)
Input 30 is used to sample Axis6 and Axis7 (Usually U and V)

Note that given the above table, the user might have to jumper the probe input to more than one input to sample all the desired Axes.

On the ACR2000
Input 12 is Used to sample Axis0,Axis1,Axis2 and Axis3.
Input 13 is used to sample Axis0,Axis1,Axis2 and Axis3.
Input 14 is used to sample Axis0,Axis1,Axis2 and Axis3.
Input 15 is used to sample Axis0,Axis1,Axis2 and Axis3.

Note that given the above table, the user only has to wire the probe into one input to sample all the desired Axes.

The input settings are done in the System parameters. The triggering can be selected to happen either on the leading or the trailing edge.

G31 is a one shot command that is only active for one block. In this mode, the axis will try to complete the current command line. If the probe contact closes before the commands are complete, the control will immediately skip to the next line. At the instant that the probe hits, the control SAMPLES the encoder position of all the axes and stores it in GLOBAL paramaters #201, #202,#203,#204,#205,#206,#207,#208 for upto 8 Axes.

As an example the following program should be considered.

N100 F100 G1 X0Y0
N110 G31 X1
N120 Y1
N130 X0
N140 Y0

If the probe does not trigger, the above program will describe a SQUARE of 1 Unit on each side.

If on the other hand the probe hits when the axis is at 0.75 units in block N110, the axes will decelerate to a stop. Then the axes will move back to the exact point when the probe hit. (This compensates for the decelerating distance that the axes took to come to a halt).
At this point, the rest of the move in block N110 will be abandoned and execution will go on to block N120.
The net result is a RECTANGLE that is 0.75 units wide and 1 unit.

At the instant that the probe hits the part, the system stores the axes positions into parameters #201....#208.

This allows the user to set offsets, or floating zeros to compensate for tool wear or fixture positions.

"G50" Scaling Cancel

Syntax: G 50

This command cancels the scaling initiated by the G51 command. This is a modal command.

"G51" Scaling On

Syntax: G 51 (Axis_Assign) {P <expression>}
{I <expression-1> J <expression-2> K <expression-3>}

Here Axis_Assign is the scaling center and P is used when scaling is to be applied equally to all axes. Instead of P, IJK can be used to apply unequal scaling to axes.

Note: if P is omitted, then scale_factor == the previous scaling factor. If IJK are omitted, then
<expression-1> == (var_ref_position #0)
<expression-2> == (var_ref_position #1)
<expression-3> == (var_ref_position #2)

This command allows scaling the part. The scaling for X and Y can be different if no cutter compensation is to be used. If the cutter compensation is to be used, the scaling for X and Y should be the same. The scaling center must be in absolute coordinates.

"G52" Local Coordinates System.

Syntax: G52 {Axis_Assign*}
where Axis_Asign ::= axis <expression>

When the local coordinates system is set, the all positions in the work coordinate systems 1….6 are shifted by the amount specified by the "Axis_Asign". If the Axis_Asign happens to be all zeroes….(X0Y0Z…) then the coordinate system is cancelled.

"G53"Machine Coordinate System.

Syntax: G53 {Axis_Assign*}
where Axis_Asign ::= axis <expression>
This command is a oneshot command that will position the machine in the Machine Coordinate system to the point specified by the Axis_Asign .

"G54….G59" Work Coordinate System.

Syntax: G <number> {Axis_Assign*}
where Axis_Asign ::= axis <expression>

These commands are six different work coordinate systems that allow shifting of the program based on values entered into the Work Coordinate parameters using the G10 L2 command.

"G61" Exact Stop Mode

Syntax: G61

This command initates a mode that makes the axes decelerate to a stop at the end of each G1 move and an inposition check performed. This mode is cancelled by a G64.

"G62" Automatic Corner Override

Syntax: G62
This command, when commanded during cutter compensation, will cause the axes to slow down on corners to the corner slow down speed set in the system parameters if the angle of the incedent lines is less than the threshold set in the system parameters. This mode is valid until the G61 (exact stop mode) or the G64 (cutting mode) or the G63 (tapping mode) is commanded.

"G63" Tapping Mode

Syntax: G63

When this command is commanded, the feedrate override is ignored and kept at 100%. No accel/decel is performed between blocks. This mode is valid until G61 (Exact stop mode) or G62 (automatic corner override mode) or G64 (Cutting mode) is commanded.

"G64" Cutting Mode

Syntax: G64

This command disables any acceleration or deceleration between blocks. This mode stays active until G61 (exact stop mode) , G62 (automatic corner override) or G63 (tapping mode) is commanded.

"G65" Macro Call

Syntax: G65 P <sub_number> { L <repeat_count>} {<var_assignment>}*
where <var_assignment> ::= variable <expression>

This is a new command and offers the following type of call structrue.

G65 P1000 L2 X1 Y2 Z3 A4 B5 C6 D7 E8 F9 G10 H11 I12 J13 K14

In this above example, Subroutine N1000 will be called 2 times with the following values loaded into the local variables.
#1 = 1
#2 = 2
#3 = 3
#4 = 4
#5 = 5
#6 = 6
#7 = 7
#8 = 8
#9 = 9
#10 = 10
#11 = 11
#12 = 12
#13 = 13
#14 = 14

This command will call the subroutine and loop L times only once.

N1000 (Sample Subroutine)
RET

"G66" Modal Macro Call

Syntax: G66 P <sub_number> { L <repeat_count>} {<var_assignment>}*
where <var_assignment> ::= variable <expression>

This command works identical to G65 with the exception that once the above setup Is completed, the subroutine call is evoked after every subsequent move block until the mode is cancelled by using the G67 command.

"G67" Modal Macro Call Cancel

Syntax: G67

This command cancels the modal macro G66 command.

"G68" Coordinate System Rotation

Syntax: G68 Axis_Assign1 AxisAssign_2 R <expression>

This command is modal and will cause subsequent program to be rotated about the center described by the Axis_Asign position. The rotation is in degrees. Positive rotation is CCW, negative rotation is CW.

"G69" Rotation Cancel

Syntax: G69

CodeGeneration: rotation_cancel

Can Cycles

The following Can Cycles are kept in a file called CYCLES.TXT. If the user wants to use these cycles, this file must be included in the part program by using the following command at the end of the patricular file.

$I CYCLES.TXT

The above command should be on a line by itself and must be AFTER the main body of the user program, past the M2 (end of program) command.

The Can Cycles use labels in the range of N73000…… and higher. Therefore the user programs must not use any labels higher than say 60000 for any user program.

The general syntax is as follows:

G<cycle#>{AxisAssign1}{AxisAssign2}{AxisAssign3} {R <r-exp>}
{Q <q-exp>}{P <p-exp>} {L <l-exp>} {F<f-exp>}
where AxisAssigni = <exp-i> Axisi

"G73" High Speed Peck Drilling.

Example Blank: G73 X___Y___Z____R____Q____

"G74" Counter Tapping.

Example Blank: G74 X____Y____Z____R____P____

"G76" Fine Boring Cycle.

Example Blank: G76 X____Y____Z____R____P____Q____
This requires custom written Spindle Orient m code.

"G81" Spot Drill Cycle.

Example Blank: G81 X____Y____Z____R____

"G82" Counter Boring Drill Cycle.

Example Blank: G82 X____Y____Z____R____P____

"G83" Peck Drilling Cycle.

Example Blank: G83 X____Y____Z____R____P____Q____

"G84" Tapping Cycle.

Example Blank: G84 X____Y____Z____R____P____

"G85" Boring Cycle.

Example Blank: G85 X____Y____Z____R____

"G86" Boring Cycle.

Example Blank: G86 X____Y____Z____R____

"G87" Back Boring Cycle.

Example Blank: G87 X____Y____Z____R____Q____P____
This requires custom written Spindle Orient m code.

"G88" Boring Cycle.

Example Blank: G88 X____Y____Z____P____

"G89" Boring Cycle.

Example Blank: G89 X____Y____Z____R____P____

"G80" Cancel Can Cycle.

Syntax: G80

This cancels the can cycles.

"G98" Return to Initial level Select .

Syntax: G98

This specifies the return point for all can cycles to be the initial level. The drilling starts from the end point of the previous block. If the previous block ended at the R-Point, the drilling will start at the R-Point but return to the initial point at the end of the cycle.

"G99" Return to R-Point Select .

Syntax: G99

This specifies the return point for all subsequent can cycles to be the R-Point. The drilling will start from the end point of the previous block. If the previous block ended at the initial point, the drilling will start from there but return at the end of the cycle to the R-Point.

Program Example:

```
(This program uses some of the new features available in v15.00 of AcroMill)

(Apart from the fixture offsets, the system now supports )
(Work Co-ordinates G54,G55,G56,G57,G58,G59 like the Fanuc 21M CNC)

(start with local coordinate system G54)
G54

(The following commands allow loading values into the work coordinate systems)
G10L2P1X00Y0Z0  (COORDINATE #1 AT 0,0,0)
G10L2P2X05Y0Z0  (COORDINATE #2 AT 5,0,0)
G10L2P3X10Y0Z0  (COORDINATE #3 AT 10,0,0)
G10L2P4X10Y5Z0  (COORDINATE #4 AT 10,5,0)
G10L2P5X05Y5Z0  (COORDINATE #5 AT 5,5,0)
G10L2P6X00Y5Z0  (COORDINATE #6 AT 0,5,0)

G54

(Return from reference point functions)

G28 X5 Y8 Z1 (Return to Reference Point#1 via X5 Y8 Z1)
M0          (OPTIONAL STOP)

G29 X0 Y0 Z0  (RETURN TO X0 Y0 Z0 VIA X5 Y8 Z1)
M0          (OPTIONAL STOP)

G30 X8 Y5 Z1 (RETURN HOME VIA X5 Y8 Z1)
M0

G31 X100Y100     (CHECK SKIP FUNCTION)

G92 X#101 Y#102  (THIS SHOULD SET FLOATING ZERO WHERE SKIP OCCURED)

X.5 Y.5 Z.5  (MOVE 1/2 INCH AWAY AND SET FLOATING ZERO)
G92X0Y0Z0

X.1
Y.1
X0
Y0
```

```
G53 X0 Y0 Z0  (MOVE BACK TO 0 0 IN MACHINE CO-ORDINATES AND CANCEL SET ZERO)
G92 X0 Y0 Z0

G52 X.75 Y.75 (CHECK LOCAL CO-ORDINATE SYSTEM)
G91 X.5
Y.5
X-.5
Y-.5
G52 X0 Y0

(DWELL FOR 2.5 SECONDS)
G4 F2.5
(TEST MACRO FIRST AND THEN MODAL MACRO)
X0Y0Z0

(Call the same subroutin with different work co-ordinates)
GOSUB 2000

X0Y0Z0
G1
GOSUB 1000

G55
X0Y0
G0
GOSUB 1000

G56
X0Y0
G1
GOSUB 1000

G57
X0Y0
G64
GOSUB 1000

G58
X0Y0
G64
GOSUB 1000

G59
X0Y0
GOSUB 1000

G54
X0Y0

X20Y0Z0
GOSUB 3000 L5 (STAIRCASE TEST TO CHECK L REPEATS)
X0Y0Z0

(CHECK ROTATION)

X15 Y5
```

```
G92X0Y0

G68 R45
GOSUB 1000

(CHECK SCALING ON TOP OF ROTATION)

G51 P2
GOSUB 1000
G52 P3
GOSUB 1000
G50
G69
(CANCEL ROTATION AND SCALING)
GOSUB 1000

(CHECK TOOL LENGTH Offset)
H1 X.5   Z0
H2 X1    Z0
H0 X1.5  Z0
H1 X2    Z0
H2 X2.5  Z0
H0 X3    Z0

M2

(THIS IS A TEST FOR MODAL MACRO)
N1000 G90
G66 P10000 D0.5 E8
X1
Y1
X0
G67
Y0
M99      (M99 IS THE SAME AS "RET")
N10000 G90
Z-#7
Z0
M99

N2000 G90
X2Y2Z0
G65 P10000 D.5
G91 X1
G65 P10000 D1
G91 Y1
G65 P10000 D1.5
G91 X-1
G65 P10000 D2
G91 Y-1
G90
RET

N3000 G91     (THIS WILL BE USED IN THE STARICASE TEST)
X1
Y1
```

```
G90
M99

(An example of how to use the WHILE loop command)

X0Y0
˝
P1=0
˝
WHILE (P1<1) DO
˝
X P1
˝
P1=P1+0.1
˝
END
˝



(The following lines will do the sin test for 0-360degrees)
˝
P1=0
˝
N100
˝
P2=Sin(P1)
˝
P1=P1+0.1
˝
X(P1) Y(P2)
˝
IF (P1<6.283) THEN GOTO 100
˝

˝
(The following lines will do the Cosine test for 0-360 degrees)
˝
P1=0
˝
N200
˝
P2=COS(P1)
˝
P1=P1+0.1
˝
X(P1) Y(P2+1)
IF P1<6.3 THEN GOTO 200

X0Y0

(The following lines will do the Tan test for 0-45 degrees)
P1=-.78
N300
P2=TAN(P1)
```

```
P1=P1+0.05
X(P1) Y(P2+1)
IF P1<0.78 THEN GOTO 300

X0Y0
(The following lines will do the ArcTan test from -1 to +1)
P1=-1
N400
P2=ATAN(P1)
P1=P1+0.05
X(P1) Y(P2+2)
IF P1<1 THEN GOTO 400



X0Y0

P1=-1.0
P2=-1.0
(This tests ABS command)
X(ABS(P1))
Y(ABS(P2))
X0
Y0
(This checks the Sqr command)
X(SQR(P1+P1))
Y(SQR(P1+P1))
X0
Y0

(This checks unary - command)

X(-SQR(P1))
Y(-SQR(P1))
X0
Y0

(This checks the Sqrt command. Axes should do 1.77245385 square)
P2= 3.14159265

X(SQRT(P2))
Y(SQRT(P2))
X0
Y0

(This checks the Exp command. Axes should do a 2.314049260 square)
X(EXP(P2)/10)
Y(EXP(P2)/10)
X0
Y0

(This checks the Ln command. Axes should do a 1.14472989 square)

X(LN(P2))
Y(LN(P2))
X0
```

```
Y0

(This checks out the INT and FRAC commands. Axis should do rectangle of)
(X3 and Y 0.14159265)

X(INT(P2))
Y(FRAC(P2))
X0
Y0

(This checks out MOD and DIV commands. Axis should do a rectangle of)
(X2 and Y3)
X(512 MOD 3)
Y(9 DIV 3)
X0
Y0
```