A Robotic Application for Contamination Free Assembly

Randy M. Dumse

New Micros, Inc.
2100 N. HWY. 360
Suite 1607
Grand Prairie, Texas 75050

Abstract

Rockwell's Electro Static Gyro Navigation (ESGN) System is built around a one gram beryllium ball that functions as a gyro rotor. It is round within four millionths of an inch and is electrostatically suspended in a cavity with only two thousandths of an inch clearance between the ball and the walls. The presence of a detectable impurity during assembly could cause an electrostatic discharge that would destroy the gyro. Cleaning by conventional methods typically took 25 hours. This paper discusses the mechanical construction and FORTH programming of a robot arm that reduced the process to a 17 minute cycle.

Often times, technological advances in one area of endeavor cause the need for matching advances in entirely different areas. It was exactly this scenario that led to the development of the gyro cleaning and assembly robot used today at the Marine Systems Division of Autonetics, Rockwell International.

The Electro Static Gyro Navigation (ESGN) System produced by the Marine Systems Division is the finest inertial navigation system in use in the world today. In operation, this system allows the U.S. Navy's nuclear submarines to run submerged on patrol for six months at a time and then surface within feet of the intended position. Such accuracy in navigation places unparalleled tolerance requirements on the manufacturing and assembly process of the mechanical portions of the Electro Static gyro. In particular, the assembly process was found to be nearly impossible by conventional "human" operators. It was determined by the management of the ESGN program that only an automated cleaning process could yield suitable results. The development of that system is the subject of this paper.

At the heart of the ESGN System is a one gram beryllium ball that functions as the rotor of the gyroscope. This ball rotates between the two halves of an enclosing cavity suspended free from the walls by an electrostatic force field. The ball itself is manufactured to be round within four millionths of an inch. There are only two thousandths of an inch clearance between it and the cavity walls. Any particular material or oils clinging to the rotor can cause degradation of system accuracy. Additionally, total system failure can result from arc-over from the electrostatic field which can reach potentials of one million volts per inch. Clearly, the final assembly process must remove all detectable materials from the surface of the rotor and cavity.

Although the assembly personnel at Rockwell International's Autonetics Marine Systems Division are experienced and competent, manual cleaning of the gyro assembly proved unsatisfactory. The assembly personnel experienced a great deal of personal frustration and aggravation. Despite the clean room 1000 conditions and Class D 10000 work station precautions, the cleaning processes had to be repeated many times before a completed cavity assembly could pass inspection. It would take a determined technician at least four hours and, more typically, up to twenty-five hours to achieve a successful cleaning. (1)

There were several problems with human cleaning methods. In order to perform the scrubbing, manual surgical gloves were used to protect the technicians from the extreme cold jet streams of cleaning fluids and also to keep body oils off of the parts. It was found that even the finest gloves still slough off

*Received September 1983.

rubber particles that were sufficient to degrade the cleaning process. Cosmetic materials, oils and cloth fibers seemed always present when humans were involved. The hand assembly of the electrostatic gyro continued for four years until a special team was formed to automate the process. A search of the available robots at the time was made and none were found suitable for the cleaning task. All of the existing equipment was very expensive and generally suitable only for heavy industrial uses. The team decided to undertake the construction of a robot designed specifically for the cleaning task.

The results of their labors were most satisfactory. The robot arm they constructed ran the entire cleaning process of rotor and two cavities with an almost every time success in a cycle that takes only 17 minutes. Figure 1 depicts this robot in action. The entire development of the robot was done with the part time work of three mechanical engineers and the contract services of one programmer in only six months.

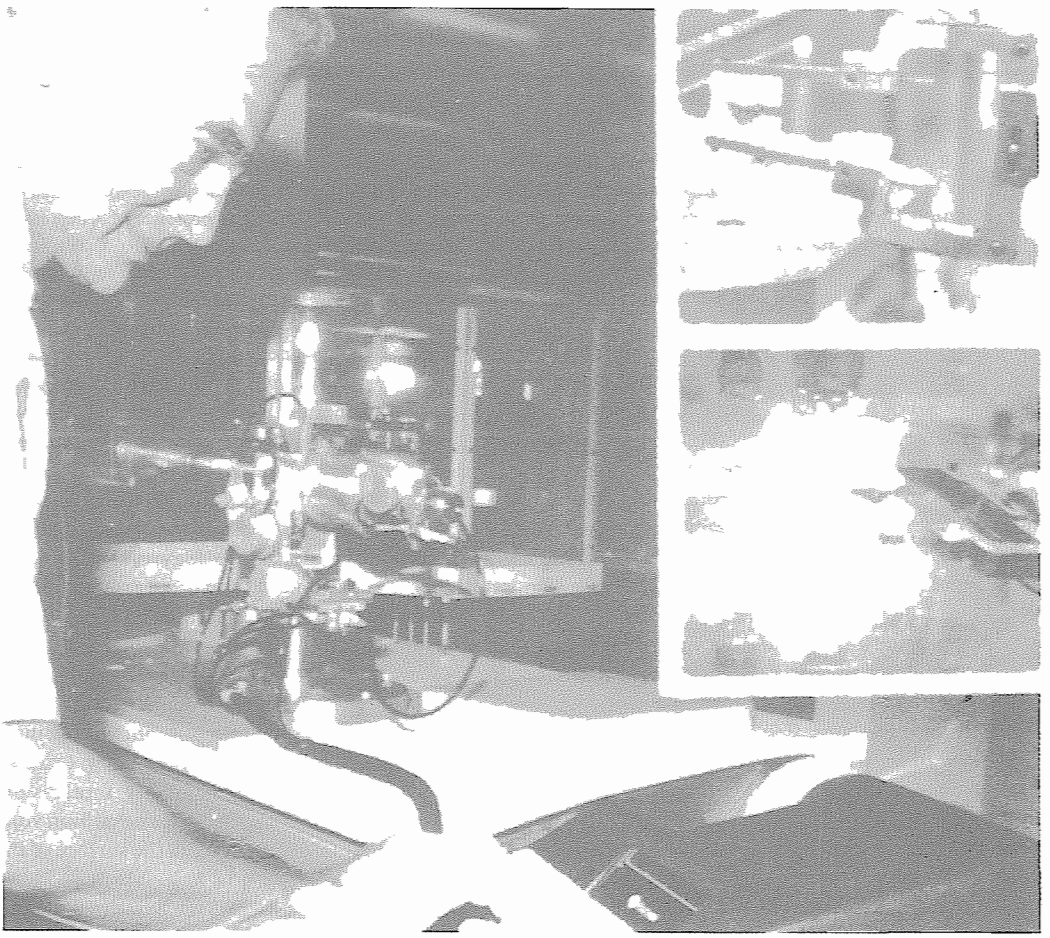

Figure 1. AIM 65 Microcomputer Controller Robot Gyro Cleaning Station
Insert 1: Gripper Holding Rotor
Insert 2: Cavity in Freon Spray

Mechanically, the design of the automated cleaning system held no mystery for the team. They conceived of a system composed of three areas; an open loading area, an enclosed dry area, and an enclosed wet or washing area.

The loading area was provided so the operator could load the rotor and cavity valves outside the areas in which the robot arm operated. The parts were placed on pedestals affixed to a "lazy susan" platform. The platform had three sides, each with four pedestals, one for the lower cavity half, one for the upper cavity half, one for the rotor ball and an assembly station for the finished parts. Rotating the platform one third of a revolution exposed a new set of pedestals to the operator to be cleared or loaded and positioned recently loaded parts within the reach of the robot arm inside the casing that housed the other two stations. The casing, made of clear plexiglass, allowed the robot arm to work in a slightly pressurized nitrogen atmosphere.

The casing was separated into two distinct halves by a plexiglass wall and swing door. The two halves were called the wet half, where the cleaning is done, and the dry half, where pick up and assembly is done.

Upon command from the operator the robot arm can begin the cleaning cycle. It first indexes to the position of the lower cavity, surrounds it with its grippers and grasps it firmly. It then lifts and transports the cavity to the wet half of the chamber. There, a freon jet stream is activated and the cavity passed through it to dislodge adhering contaminates. With each pass, the robot arm steps forward a fraction of an inch until the entire surface has been vigorously scrubbed. The cavity is then inverted and again stepped through the stream to clean the opposite side. One final step on the wet side is to rotate the scrubbed cavity over a jet stream of nitrogen to dry the part before being transported to the dry side of the chamber and lowered into the assembly pedestal.

The second phase of the cleaning process is that of the rotor itself. Much like the lower cavity before it, the rotor is transported to the wet side where it is passed through the freon stream repeatedly with small advancing steps until all the exposed surface is covered by the full impingement force of the high velocity stream. Critical areas of the rotor are covered, however, by the grippers so an intermediate set down of the rotor is instituted. The robot arm then reorients its grasp by rotating its grippers. It regrasps the rotor and repeats the wash cycle. A third wash cycle with the grippers slightly loosened completes the cleaning with a pseudo-random sweeping as the rotor floats free between the grippers. The rotor is dried in the nitrogen and then transported to the lower cavity and gently lowered into place.

Finally the upper cavity is cleaned in the same manner as the lower and placed on top of the two parts already assembled. The robot arm is retracted, allowing the platform to be rotated by the operator to recover the finished assembly. The cleaning program is then complete.

To have the facility to accomplish its task, the robot arm was designed with 5 axes of movement. Four of the axes were controlled by rotating stepping motors, the fifth by a linear actuator. As can be seen in Figure 2 these motors control the polar movements of the arm with respect to its fixed base. Movement can occur in horizontal or vertical axes. Also, the mechanical actuator, referred to as the gripper, can be extended, rotated, and opened or closed. Control of these five independent motions is all that is necessary to effect all the complex actions in the cleaning process.

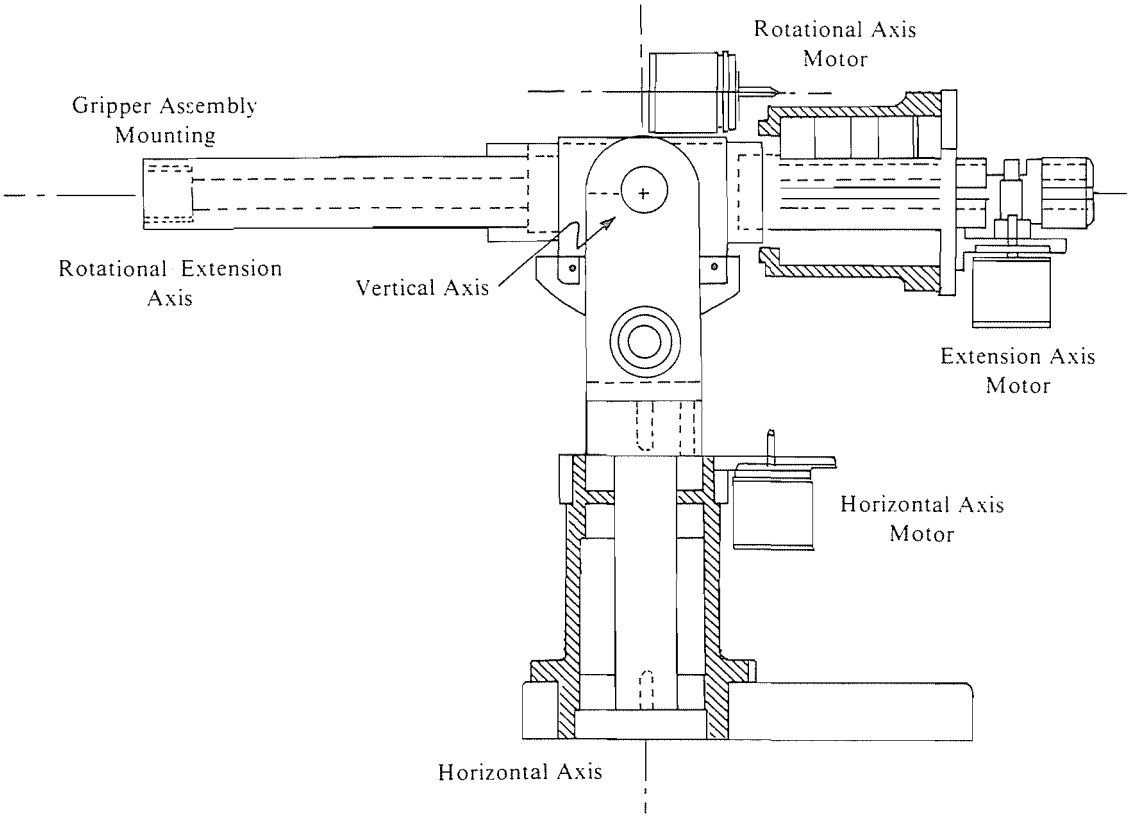

Figure 2. Single Arm Robot

Electrically the stepper motors are controlled by an SA 1027 integrated circuit. The microcomputer that hosts the FORTH program is an AIM-65. The outputs of the 6522 Versatile Interface Adapter from the AIM-65 are buffered by open collector level converts. Only two signals per motor are required, one for direction and another to initiate a step. Besides these, two other lines are buffered to operate the air solenoids that control the freon and nitrogen valves. See Figures 3, 4 and 5 for details.

This unusually successful project's outcome can be directly related to decisions made in the beginning by the project manager. One of those landmark decisions was to do the system programming in FORTH. It is notable that the project manager and his part-time assistants were mechanical engineers, with no programming experience. The project manager had heard of the reputation of FORTH in real time control of applications and under the advice of the contract programmer chose FORTH for the programming language. The contract programmer was a relatively newcomer to FORTH also, and this was his first project attempted in the language.

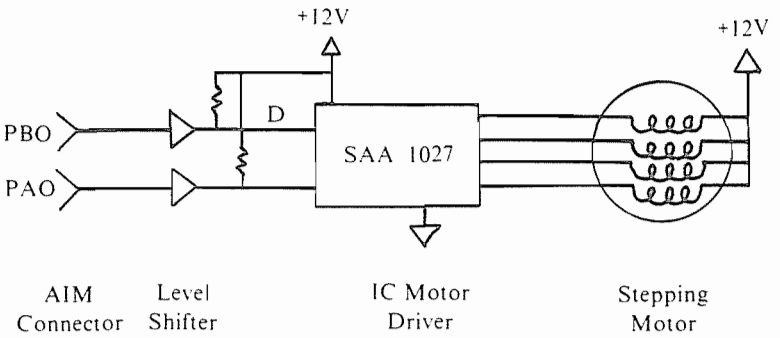

Figure 3. AIM 65 To Motor Drive Circuit

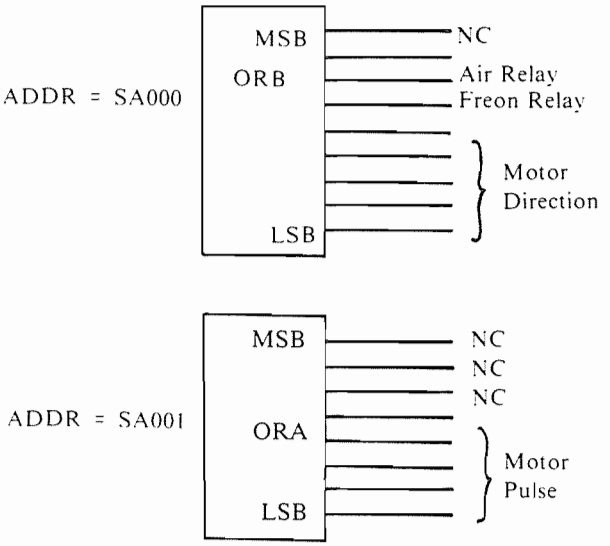

Figure 4. AIM 65 User R6522 Pin Assignment

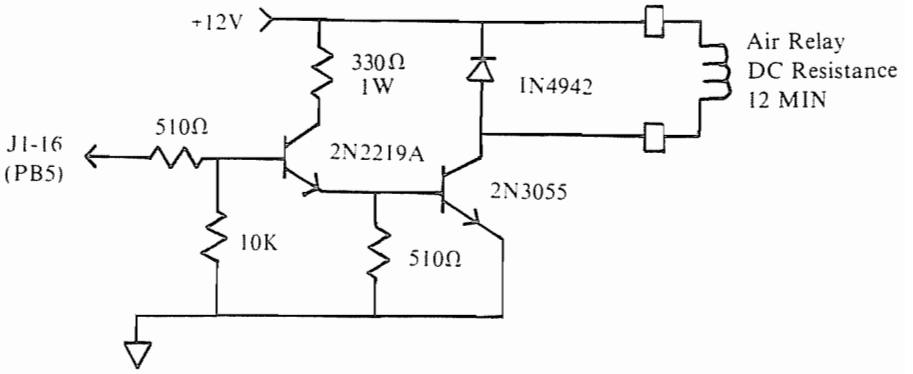

Figure 5. AIM 65 to Relay Driver

Perhaps the most remarkable point of all concerning this project is the manner in which the software was developed. At the request of the mechanical engineers, the contract programmer developed a series of words that defined basic primitive movements for each axis. This gave the engineers basic tools to manipulate the arm for testing purposes during construction. The programmer polished the basic set of words and assisted the mechanical engineers with the FORTH language. After that, it was actually the mechanical engineers that finished the rest of the programming. They quickly learned that any action they wished to achieve could be programmed by making new definitions based on the basic words provided. Beyond this ability to use the colon definition the engineers required no further instruction on use of the FORTH language.

The basic programming, shown in Listing 1, provides three new words for each possible axis of movement: one word for a relative movement in the positive direction, one word for a relative movement in minus direction and one that moves to an absolute position. A few other utilities and lower level primitives for the construction of the basic words complete the set.

Listing 2 shows the first level of words written by the mechanical engineers. Notice that each word uses only the existing basic words from Listing 1 and numbers. Listing 3 further shows how words defined by these engineers became the building blocks for higher level words.

Essentially, the contract programmer created a new language suited to the control of the robot arm. It is not easily recognized as such, though, because it exists as an extension of the AIM-65 FORTH language, that is, the language is added to an "open" FORTH system. Careful examination of the code generated by the mechanical engineers will show that only ":" and ";" and the extension words are required to have complete control over the robot functions. These words would have been just as effective in a closed implementation without the rest of FORTH. Another examination of Listing 1 is in order. It is a complete language written in FORTH that can be completely defined on two sheets of paper!

If more expressive definition names had been chosen for the basic words this new language could be easily learned. For instance, if SVP, which is an acronym for "swing vertical positive", had instead been named UP its intent would have been obvious. Similarly SHM could have been called LEFT and CLP called OPEN-GRIP. This programming certainly opens the door for the creative thinker to imagine new standards for robotics languages that could be generated from a FORTH base set.

In summary, there were a number of factors that made this project successful. In particular the choice of FORTH as the programming language can be cited. The use of FORTH as a development tool allowed interactive testing of the hardware before it reached its final configuration. The programming itself was tested incrementally. Words defined as lower level functions were easily combined as functions grew in complexity. Still those basic words were available for system alignment which was done on system power up manually. The FORTH operating system was complete enough to be resident in the target computer system, the AIM 65. No large expensive development system was needed and there was no waiting for machine time during development. If the intention had been to make this system for general marketing it would have been easy to use an even less expensive FORTH system with the language permanently extended. The current availability of the R65F11 FORTH-based microprocessor would lead to the obvious concept of a similar arm being built with control electronics installed in the base of the

unit.(2) The PROM programming abilities of the R65F11 would allow the user programs, such as those shown in Listings 2 and 3 to be permanently stored when the project programming was complete.

In all, it is difficult to say that the project could not have succeeded if FORTH was not chosen, but it is easy to say it would have been in no wise as quickly done. The possibility of simple languages for specific robotic applications under FORTH should now be sufficiently tantalizing to stimulate further work in the field.

References

1. William, Jeff and Dumse, R., "Labor Intensive Task Eliminated With An AIM 65 Microcomputer Controlled Robot", *WESCON Professional Program Papers* (1982).
2. Dumse, R., "Rockwell International Single Chip FORTH", *FORML Proceedings* (1982).

Figures and listing courtesy of Rockwell International.

Mr. Dumse graduated from the University of North Iowa in 1975 with a BA in physics. He developed the Rockwell Forth Chip and is interested in dedicated computer systems and microprocessor technology. As president of New Micros, he developed a board level version of the Rockwell chip, the "100 Squared".

Listing 1. Robot Arm Control FORTH Program

```
( Robot Arm Control in FORTH                                    By Dave Fincher )
                                                    ( Revised Version Oct. 23, 1981 )

HEX                          ( This part of the program is in hex )
0 VARIABLE ACT               ( ACT defines the active channel )
0100 VARIABLE TIME           ( Used for delay time )
A000 CONSTANT VIA            ( VIA address )
0 VARIABLE PEX               ( Position Variable, Extension Axis )
0 VARIABLE PSV               ( Position Variable, Vertical Axis )
0 VARIABLE PSH               ( Position Variable, Horizontal Axis )
0 VARIABLE PRO               ( Position Variable, Rotating Axis )
0 VARIABLE PCL               ( Position Variable, Clamp )

( Define word to initialize the VIA )
: T2L TIME C@ VIA 8 + C! ;          ( Ls byte of time to t21 of via )
: ACR 0 VIA B + C! ;                ( 0 to aux cntl register )
: DRA 1F VIA 3 + C! ;               ( 5 outputs from ora )
: DRB 7F VIA 2 + C! ;               ( 7 outputs from orb )
: ORA FF VIA 1+ C! ;                ( Set a outputs hi ( Motor drive trigger ) )
: ORB 0 VIA C! ;                    ( Set b outputs lo ( Motor direction and valves ) )
: I-VIA ACR DRA DRB ORA ORB T2L ;       ( All the above )

( Define the output control words )
: HI VIA 1+ DUP C@ OR SWAP C! ;     ( Sets the active channel pin )
                ( of ORA hi without affecting other pins )
: INV FF XOR ;  (changes stack 0's to 1's and 1's to 0's )
: LO VIA 1+ DUP C@ ACT C@ INV AND SWAP C! ;
                ( Sets active channel pin of ORA lo )
: T2H TIME 1+ C@ VIA 9 + C@ ;       ( MSB time goes to T2CH )
: RED VIA D + C@ 20 AND 0= NOT ;    ( Tests for T2 time-out )
: DELAY T2H BEGIN RED UNTIL ;       ( Loads T2H and waits for time-out )
: PULSE LO DELAY HI ;  ( Sets active trig lo, delays then sets trig hi  )

: SET-  ( Set the motor direction of the active channel to minus )
        ( without affecting other channels )
  VIA DUP C@ ACT C@ OR SWAP C! ;
: SET+    ( Similar to Set- )
  VIA DUP C@ ACT C@ INV AND SWAP C! ;
: CYCLE DO PULSE LOOP ;   ( Output N = stack pulses )

( Define Rotating Words )
: ROP     ( Drive the rotating axis positive by the delta on the stack )
    DUP   ( Dup the commanded (stack) position and use it to determine )
          ( direction of travel and save as the new value of PRO )
    8 ACT C!  ( Set active channel to 8 )
    SET+  ( Direction is plus )
  PRO @ + PRO!  ( Add delta to position register )
  0 CYCLE ;  ( Now count out pulses to the active trigger. )
  0 ACT C! ;  ( Set active channel to zero )
: ROM  ( Same as ROP except rotates minus )
  DUP 8 ACT C! SET- PRO @ SWAP - PRO ! 0 CYCLE 0 ACT C!
        PRO ! ABS 0 CYCLE 0  ACT C! ;
```

Listing 1. Continued.

```
: RO ( Drive the rotating axis to the location specified on the stack )
  DUP 8 ACT C! PRO @ - DUP SET+ 0< IF SET- THEN
  SWAP RPO ! ABS 0 CYCLE 0 ACT C! ;

( Define Clamp Words )
: CLP DUP 10 ACT C! SET+ PCL @ + PCL ! 0 CYCLE 0 ACT C! ;
: CLM DUP 10 ACT C! SET- PCL @ SWAP - PCL ! 0 CYCLE 0 ACT C! ;
: CL DUP 10 ACT C! PCL @ DUP SET+ 0< IF SET- THEN
 SWAP PCL ! ABS 0 CYCLE 0 ACT C! ;

( Define Extension Words )
: EXP DUP 1 ACT C! SET+ PEX @ + PEX ! 0 CYCLE 0 ACT C! ;
: EXM DUP  1 ACT  C! SET- PEX @ SWAP - PEX ! 0 CYCLE 0 ACT C!
: EX DUP 1 ACT C! PEX @ - DUP SET+ 0< IF SET- THEN
 SWAP PEX ! ABS 0 CYCLE 0 ACT C! ;

( Define Vertical Words )
: SVP DUP 2 ACT C! SET- PSV @ + PSV ! 0 CYCLE 0 ACT C! ;
: SVM DUP 2 ACT C! SET+ PSV @ SWAP - PSV ! 0 CYCLE 0 ACT C! ;
: SV DUP 2 ACT C! PSV @ - DUP SET- 0< IF SET+ THEN
 SWAP PSV ! ABS 0 CYCLE 0 ACT C! ;

( Define Horizontal Words )
: SHP DUP 4 ACT C! SET+ PSH @ + PSH ! 0 CYCLE 0 ACT C! ;
: SHM DUP 4 ACT C! SET- PSH @ SWAP - PSH ! 0 CYCLE 0 ACT C! ;
: SH DUP 4 ACT C! PSH @ - DUP SET+ 0 IF SET- THEN
 SWAP PSH ! ABS 0 CYCLE 0 ACT C! ;

( Define Valve Control Words )
: AIR 20 VIA C! ;
: SPRA 40 VIA C! ;
: OFF 0 VIA C! ;

( Define Delay and Speed Change Words )
DECIMAL     ( Decimal from here on )
: WAIT2 750 0 DO LOOP ;
: WAIT BEGIN WAIT2 1- -DUP 0= UNTIL ;  ( Loop and count down until stack )
                                       ( is zero. Delay = 0.1* stack )
: SP1 30000 TIME ! T2L ;   ( Lowest speed )
: SP2 17000 TIME ! T2L ;
: SP3 10000 TIME ! T2L ;
: SP4  5000 TIME ! T2L ;
: SP5  2400 TIME ! T2L ;   ( Highest speed )
I-VIA
OFF
FINIS
```

Listing 2. Overview of a Robotic Application

```
WORD DICTIONARY

                    GET FIRST CAVITY
: CAV 1 SP1 30 SV SP4 448 SH SP1 443 SH 0 SV SP3 25 CL;
                    MOVE CAVITY TO WASH
: MOV 1 SP1 30 SV SP4 -40 EX 1388 SH SP1 0 SV;
                    WASH CAVITY
: WAC SPRA SP2 140 EXP 30 SHM 30 0 DO SP3 60 SHP 4 EXM
      60 SHM 4 EXM LOOP 20 EX -40 EX 1388 SH OFF;
                    DRY
: DRY AIR SP4 1535 SH 1529 SH 300 WAIT;
                    SET FIRST CAVITY DOWN AT ASSEMBLY
: SET 1  SP4 200 EXP OFF 0 RO SP1 30 SV SP4 0 SH 0 EX SP1
      0 SV OCL;
```

Listing 3. Overview of a Robotic Application

```
WORD DICTIONARY

                    WASH FIRST CAVITY COMPLETE
: WIC               CAV 1       GET FIRST CAVITY
                    MOV 1       MOVE CAVITY TO WASH
          SP4       240 RO      ROTATE TO FIRST WASH ANGLE
                    WAC         WASH CAVITY
          SP4       1200 RO     ROTATE TO SECOND WASH ANGLE
                    WAC         WASH CAVITY
          SP4       1440 RO     ROTATE TO THIRD WASH ANGLE
          0EX       DRY         DRY
                    SET 1;      SET FIRST CAVITY AT ASSEMBLY
```

THE "WORD" WIC KEYED INTO THE COMPUTER IS ALL THAT IS NECESSARY TO PERFORM THE ENTIRE CAVITY WASH CYCLE!