WATCHDOG SAFETY COMPUTER DESIGN AND IMPLEMENTATION

Roger D. Kilmer
Harry G. McCain
Maris Juberts
Steven A. Legowik

February 1984

To be Presented at the Robots 8 Conference
and Exposition, Detroit, Michigan,
June 4-7, 1984.

Abstract

WATCHDOG SAFETY COMPUTER DESIGN AND IMPLEMENTATION

There are many different aspects of safety to consider when utilizing a robot in an industrial application. In general, however, these can be categorized into the areas of personnel safety and equipment safety. This paper addresses the later category and presents one approach of providing equipment safety through the use of an auxiliary computer to monitor operations in the workstation. Such a computer system can be used to check robot operations during programming, automatic cycling, and debugging and repair to prevent unwanted conditions from occurring. The basic concepts, design and implementation of such an auxiliary computer on a robot operating in a machining workstation are described here.

1. Introduction

The problem of maintaining equipment safety in a workstation equipped with an industrial robot is a complex one to solve. Unlike other types of automated devices, a robot is not constrained in terms of what path in space it can travel or the velocity at which it can operate. This problem is not a severe one at present because most robots operate using pre-taught programs which define the path through which they can travel. The velocity is also a predefined parameter and most robot controllers have internal checks to limit the velocities to some maximum value. However, as robotic systems become more sophisticated in terms of using sensory feedback for real-time control of the robot path, the problem of equipment safety will increase. Also, most robotic systems are relatively new devices. As a result, there is little or no statistical information regarding failures or what preventive measures might be taken to prevent them. The only approach that can be taken is to examine the components which might be suspected of needing maintenance and to repair others after a failure has occurred. Thus, maintaining equipment safety will undoubtedly become more critical unless preventive measures are taken to minimize such problems.

One approach of addressing this problem is to use an auxiliary safety computer to monitor robot operations. The purpose of the safety computer is to prevent the robot from damaging itself or any of the equipment and sensors mounted on, and around the robot, in the event of a hardware malfunction, software bug, or operator error. To do this, the safety computer monitors a number of status and sensor inputs from the controller, the robot, and other sources. If the information that the safety computer monitors does not satisfy certain conditions, the safety computer stops the robot and notifies the operator.

2. Safety Computer Features

The safety computer, referred to as the Watchdog Safety Computer (WDSC), is a stand-alone microcomputer system used to monitor robot-related operations in the workstation. The purpose of this system is to detect operations which are outside the range of normal conditions and to stop the robot before a collision or any damage to the robot can occur. The initial implementation of this computer system is on a Cincinnati Milacron T3 robot installed in the Automated Manufacturing Research Facility (AMRF) at the National Bureau of Standards. In this application, the WDSC is used to monitor the individual joint and tool point motions of the T3 robot and various status signals from the hydraulic pump unit and the T3 controller. The WDSC measures the amount of rotation from a known "home" position, the rotational velocity and the rotational acceleration of each of the six robot joints. It compares these measurements with a set of maximum values, which are operator selectable, and if the maximum values are exceeded, halts the robot. The WDSC also performs a similar set of comparisons for the tool point velocity and acceleration. The hydraulic pump unit and T3 robot controller status checks are simple go/no-go tests. Figure 1 shows a block diagram of this implementation of the WDSC on the T3.

These maximum value and status checks are also made by the T3 controller, so that, in this case, the WDSC is a redundant system. One difference is that the individual joint limits are operator selectable in the WDSC, which is an added convenience during system testing. The redundancy actually goes one level lower in that another resolver was installed on each joint so that joint position data could be obtained independent of the existing T3 controller. In addition to the actual resolver installation, a resolver electronics system was designed and fabricated which provides joint position data for the six joints for up to three separate user systems. At present only two systems, the WDSC and the NBS Robot Control System, obtain T3 joint position data from this resolver electronics system.

Also incorporated in the WDSC is a forbidden volume check. These volumes are regions in the work volume, such as a machine tool, an operator's workstation, or the floor, into which the tool point of the robot must not enter. The volumes are formed by representing the surfaces of the object with from one to six planes. These planes are programmed into the WDSC by moving the tool point of the robot to three points near the surface of each plane. At each of these points, the WDSC computes the tool point location in world coordinates and from these, the location and orientation of the plane in space. For each plane, a safety margin is assigned which accounts for the distance required to stop the robot. If the tool point of the robot attempts to enter one of these forbidden volumes, the WDSC halts robot motion. In this case, the WDSC is not a redundant system since the T3 controller does not provide such forbidden volume checks.

As presently configured, the WDSC checks are performed only when the T3 controller is in the auto mode. This covers all automatic operations when the robot is performing a taught program and for all conditions when the robot is being controlled by the NBS Robot Control System. The WDSC, although operating, does not halt robot motions during manual and teach operations due to limitations in the T3 controller which only allow it to request a hydraulics shutdown. Stopping the robot by disabling the hydraulics is undesirable in many instances because the arm may settle down on an object and cause more damage than if the operator is allowed to correct the situation.

3. T3 Robot Description

The T3 robot is installed in a horizontal machining workstation in the AMRF. In this workstation, the T3 is primarily involved in material handling tasks. Figure 2 shows a layout of this workstation. Figure 3 is a photograph showing the T3, the roller-bed tables, and the horizontal machining center in the background.

The T3 is a six-axis servo-controlled robot powered by hydraulic actuators. It is manufactured by the Cincinnati Milacron Company located in Cincinnati, Ohio. All of the actuators, except the one on the elbow joint, are rotational actuators. The actuator on the elbow is a linear piston type actuator. All of the joints, including the elbow, are rotational and give the T3 six degrees of freedom. The actuators support and move the mass of the T3 when it is in operation and provide a rated lifting capacity of 100 pounds.

When the hydraulic power to the T3 is off, the upper arm of the T3 rests on a "shot pin" at the shoulder joint. The shot pin extends automatically when the hydraulic power is off, preventing the shoulder joint from settling past a point approximately 45 degrees above horizontal. If, however, the T3 is already below the shot pin, there is nothing to prevent the T3 from settling down against the floor. The T3 could cause a good deal of damage if it settled on the end-effector or any of the equipment around the robot.

There are two control inputs to the T3 controller which the safety computer can use to stop the robot. These two control inputs are Emergency Stop and Hold/Set. Both of these signals were intended to be used as manual shut off switches by the robot operator. A number of normally closed switches are connected in series between the controller's AC source and the control inputs. When the circuit is broken by one of the switches being pressed, the T3 controller performs the requested function. The Emergency Stop input turns off the hydraulic power to the T3 and is active any time that the hydraulics are enabled. The Hold/Set input causes the T3 to stop its motion without shutting off the hydraulics. The motion can be resumed by issuing a signal on the Hold/Clear input. However, the Hold/Set input only works when the T3 controller is running in the auto mode. A Hold/Set will not stop the robot when it is being run in manual or teach mode.

One of the major research topics being pursued in the AMRF is the development of a generic robot controller. To that end, the NBS Robot Control System (RCS) is being used to run the T3. The NBS RCS controls the T3 robot by sending commands to the T3 controller using an external communication link. This link is supported in the T3 system software by a program called Dynamic External Path Control (DEPC). DEPC allows the T3 to be controlled via a serial communication link with an external computer, in this case, the NBS RCS. (Note: This feature is functional only when the T3 controller is in the auto mode.) In the future, the T3 will be run exclusively through this control link. The NBS RCS will provide the high level control, and only the low level servo control will be left to the T3 controller. As a result, the T3 will be running in auto mode most of the time, except for brief periods of time when the T3 is being powered up or shut down, and thus can be stopped using a Hold/Set.

4. Safety Computer Hardware Configuration

The safety computer is a Multibus compatible microcomputer system. This system, shown in block diagram form in Figure 4, consists of a Multibus 9-slot chassis equipped with a single board computer using an 8086 CPU, a parallel I/O board with three 24 line parallel ports, and two EEPROM memory boards. All of these boards are commercially available and require no customizing prior to use in the safety computer.

As shown in Figure 4, the operator terminal is connected to the system through a serial port on the single board computer. The joint position data from the resolver electronics are brought into the system through a parallel port also located on the single board computer. The T3 status data and the Hold/Set and Emergency Stop output commands are transmitted through an optical isolator I/O panel to one of the ports on the parallel I/O board. The other block shown in this diagram is the NBS RCS. The hardware and the majority of the software for this interface have been implemented, but have not been completely tested and debugged at this time.

The safety computer is located in its own chassis so that it is isolated from other systems controlling the robot. In this configuration, the safety computer has sole control of the system bus, and thus avoids the problem of multi-processors using the same bus. This will increase system reliability and eliminate the possibility of the bus being tied up by another processor. If it shared the bus with the NBS RCS for example, a malfunction in the RCS could quite easily cause the safety computer to malfunction as well.

As shown in Figure 4, the safety computer is interfaced to the T3 controller through an optical isolator I/O panel. The safety computer receives a number of status indicators from the T3 controller through this interface. This is also used to send the Hold/Set and Emergency Stop signals to the T3 controller.

The T3 uses two different logic voltages, 120 VAC and 24 VDC, neither of which is compatible with the TTL logic used by the safety computer. The optical isolator I/O panel is used to convert between the T3's logic voltages and the safety computer's TTL logic. This panel has a total of 24 I/O lines which are configured (AC or DC voltage and input or output) by selecting appropriate driving or terminating modules. The modules convert from AC or DC to TTL, and vice versa. The interface between this panel and the safety computer is through the parallel I/O board. Both input data and output commands are transmitted over this interface.

5. Safety Computer Software Structure

The general structure of the safety computer software is illustrated in Figure 5. This figure shows the major components of the safety computer software structure and the interactions between these components.

The programming language used in developing software for the safety computer is FORTH. This software is divided into two major sections: the watchdog task and the operator task. The watchdog task is responsible for monitoring robot operations and stopping the robot to prevent any damage. The operator task provides an interface that allows the user to view the status of operations and alter the parameters of the safety computer. The two tasks timeshare on the same CPU.

The operator task runs the FORTH operating system and can be used to run programs that let the user examine or change parameters and to edit the watchdog program. Figure 6 shows the user terminal diplay for editing the velocity, acceleration and stopping threshold limits. The operator can select the limits by specifying a percentage of the maximum for each parameter shown in this display. This value is the percentage of a maximum value (either velocity, acceleration, or stopping threshold) that was determined empirically and is embedded in the software for checking these limits. If the operator does not select new limits, the default values of 50% are used. Figure 7 shows the display for examining the position, velocity and acceleration values for the T3.

The operator task handles all communications with the safety computer's terminal, and hence the user. The watchdog program is transparent to the user in the operator task except when the watchdog program sends messages to the user via the operator task. The terminal is slow relative to the watchdog program; it is only able to print out about one character every millisecond at 9600 baud. Thus, messages generated by the watchdog program are stored in a message queue until they can be printed out. When the watchdog program has put a message, or messages, into the message queue, it instructs the operator task to run a program that will list out the contents of the message queue at the terminal for the user to see.

The watchdog program, running in the watchdog task, is activated every 50 milliseconds (ms) by a timer interrupt. This interrupt is a signal to the watchdog program to check the status of the T3. The program examines the data it receives and decides whether the T3 is operating normally. If the data violate any of the T3's operational constraints, the watchdog program commands the T3 to stop. The watchdog program instructs the operator task to print out messages indicating the status of the watchdog program and any error conditions that it has detected.

The watchdog program is broken down into three functional blocks: input, computations, and output. Each of these functional blocks decomposes into a number of more specific tasks. The input block reads joint data from the resolver electronics and status data from the T3 controller. The computation block checks for timing and I/O errors, converts the data from the resolvers into joint angles, computes the position, velocity and acceleration of the T3, and checks for discrepancies in the T3's motion. The output block sends commands to the T3 controller (Hold/Set and Emergency Stop), and sends error and status messages to the user via a message queue. A top level flow diagram of the watchdog program is shown in Figure 8.

In order to explain how the watchdog program acquires its data while allowing the user to interact with the operator task, an understanding of the multitasking environment of FORTH is required. FORTH uses a round robin multitasking scheme. All of the tasks are linked together in a circular list, called the multitasking round robin. The CPU passes control around this linked list processing each task in turn. Each task is in one of two states, awake or asleep. When a task is asleep, it does not require any servicing by the CPU and control is passed on immediately to the next task in the round robin. If the task is awake, the program in the task resumes execution where it left off. The CPU continues executing the program until the program releases the CPU to return to the multitasking round robin. The CPU can be released explicitly by the program or implicitly by certain FORTH words. The words primarily responsible for releasing the CPU are those concerned with terminal I/O. Whenever a character, or string of characters, is sent or received from the terminal, the task is put to sleep and the CPU is released to service other tasks in the round robin. While the task is asleep, an interrupt routine accepts or transmits data to the terminal. When the transfer is complete, the interrupt wakes the task so that it can resume operation when its turn comes again in the round robin.

The CPU alternates back and forth between the watchdog and operator tasks. The operator releases the CPU when it is waiting for terminal input or when it is in the process of writing out to the terminal. The watchdog task releases the CPU when it has no new data to process. New data are read in from the resolvers every 50 ms by a routine triggered by a timer interrupt. When the interrupt routine has read in the resolver data, it increments the "new-data" flag. This indicates to the watchdog task that there are new data to be processed, and that the next time control of the CPU is passed to the watchdog task it should execute the watchdog program. When the watchdog program has been activated by the interrupt, it collects all the other data necessary to make its decisions. In the present version of the safety computer, these are the T3 status data.

When all of the data has been collected and the computations completed, the watchdog program decides whether to stop the T3 based on the current state of the safety computer and the operating conditions of the T3. The operating conditions of the T3 are given by the T3 controller status signals and the position, velocity, and acceleration values calculated from the resolver data.

The state of the safety computer is dependent upon previous decisions and actions and determines how the watchdog program will respond to the current input data. The five states of the safety computer are:

1) Manual/Auto -- The normal operating mode of the safety computer when the watchdog program has not detected any error conditions.

2) Hold/Set -- An error has been detected and the safety computer has issued a Hold/Set. The Hold/Set line is still active.

3) Hold/Set Idle -- An error has been detected and the watchdog program has issued a Hold/Set. The Hold/Set line is inactive. (The Hold/Set signal is momentary.)

4) Emergency Stop -- An Emergency Stop has been issued, because the T3 failed to stop after a Hold/Set was issued. The Emergency Stop line is still active.

5) Emergency Stop Idle -- An Emergency Stop has been issued, because the T3 failed to stop after a Hold/Set was issued. The Emergency Stop line is inactive. (The Emergency Stop signal is momentary.)

In each of these five operating modes, the watchdog program expects the data to meet certain requirements. The differentiation between Hold/Set and Hold/Set Idle, and Emergency Stop and Emergency Stop Idle, is the state of the output line. Both the Hold/Set and the Emergency Stop should be momentary signals on the control lines. While in Hold/Set or Emergency Stop mode, the signal line is being pulsed active. During the idle modes the signal is inactive. Thus, Hold/Set goes to Hold/Set Idle, and Emergency Stop goes to Emergency Stop Idle, after briefly activating the respective control line. The safety computer stays in the idle state until something (for example, a Hold/Clear or a manual reset of the safety computer) causes the operating mode of the safety computer to change again.

When the T3 is operating normally, the safety computer is in the Manual/Auto mode. When in the Manual/Auto mode the following are checked:

1) Joint velocities below maximum
2) Joint accelerations below maximum
3) Joint position within range
4) Tool point velocity below maximum
5) Tool point acceleration below maximum
6) Tool point outside all forbidden volumes

If any of these constraints is violated, the safety computer will issue a Hold/Set. Each of these error checks can be individually enabled or disabled by the user appropriately setting or resetting the corresponding enable flag. The velocities and accelerations are checked by comparing the current value to the respective maximum allowable limit. Exceeding this limit is considered an error condition. Similarly, joint position is compared to a range of allowable values. Moving the joint outside this range is considered an error condition. The current tool point location is compared to the list of forbidden volumes. Moving the tool point inside the safety margin of any of the planes which make up these volumes is considered an error condition.

When a Hold/Set is issued, the safety computer enters the Hold/Set operating mode. While in the Hold/Set mode, the safety computer checks to see if the T3 has actually stopped. Since the T3 is very massive and the response time of the safety computer is finite, the T3 will not come to rest immediately. To take this into account, the safety computer uses a stopping threshold to decide if the T3 has responded to a Hold/Set command. The stopping threshold specifies how far the T3 is allowed to travel after a Hold/Set has been issued. If the T3 travels farther than this stopping threshold, the safety computer assumes that the Hold/Set did not work and an Emergency Stop is issued. This check can be disabled by resetting the corresponding enable flag, "e-stop-en". This may be desirable during certain types of testing when an operator is present to manually hit the Emergency Stop switch if necessary. However, in most cases this will not be necessary because the stopping thresholds were empirically chosen so that there would be no false triggering.

In the Emergency Stop mode, the hydraulic power to the T3 is shut off. As a result, the T3 will slowly go limp, settling down on whatever is below it. With no power to the T3 there is no control, and therefore, no controlled motion. The only check for Emergency Stop is to examine the appropriate status line to see if the controller is responding properly. If it is not responding, there is no action that can be taken on the part of the safety computer, aside from notifying the operator.

6. Resolvers

The safety computer's main function is to monitor the motion of the T3. This is done by monitoring the position of the robot with six resolvers, which are mounted in the T3's Position Analog Units (PAUs). Each PAU contains a resolver and tachometer which provide joint position and velocity feedback to the T3 controller. An additional resolver was added to each PAU to provide an independent source of joint position data for the safety computer. These joint angles are used to compute the various motion parameters checked by the safety computer. Each resolver returns an analog signal which indicates the angular position of the resolver and thus, the position of the robot joint. This analog signal is converted to digital form by an R/D (resolver to digital) converter in the resolver electronics package designed and built at NBS. The R/D converter generates a 14 bit number which is representative of the resolver angle. A block diagram representation of the resolver electronics is shown in Figure 9.

Due to the gearing in the PAUs, the resolvers go through a number of revolutions while the joint travels through its range of rotation. The R/D converters only provide the angle of the resolver. To get the angle of the joint, the number of revolutions of the resolvers must be taken into account. This is done by the resolver electronics package. The resolver electronics has a three bit revolution counter for each resolver and adjusts the revolution counter appropriately when the R/D converter output rolls over. The three bits are combined with the 14 bit R/D output to produce a 17 bit result. In order for the resolver electronics to provide a consistent result, the revolution counters must be initialized every time the resolver electronics are powered up. This is done by moving the robot to a known "home" location and then setting each revolution counter to the correct revolution count. The revolution count at the home position is chosen so that the 17 bit resolver output will vary continuously over the range of the joint (i.e., so that it will not roll over in the middle of the joint's travel). This is done manually from the front panel of the resolver electronics.

Both the safety computer and the NBS RCS obtain data from the resolver electronics. These data are provided on request on a first come first serve basis. Since there is only one set of six R/D converters, the resolver electronics has circuitry that only allows one system to access the resolvers at a time. A system requests data by asserting its request line, and then waits for the resolver electronics to answer the request by asserting the acknowledge line. When the requesting system receives the acknowledge signal, it is free to read the resolvers. The resolvers are read, one at a time, by sending a three bit resolver address to each resolver channel and then reading the 17 bits of resolver data. All of the resolver data is latched at the moment the acknowledge is returned (forming a time coherent set). Then the resolver address is used to select which latch is read. The acknowledge line stays active during the whole process of reading the resolver data and prevents any other system from accessing the resolver electronics. After a preset amount of time (approximately 1 ms at present), the acknowledge line goes inactive and other systems can request the resolver data. The safety computer monitors the amount of time spent waiting for the resolver electronics to return the acknowledge signal. If it does not respond within a reasonable amount of time, the resolver reading routine aborts and does a Hold/Set on the robot.

The resolver reading routine is started by the 50 ms timer interrupt. In this way the interval between resolver samples is almost constant. The slight difference in intervals caused when both systems attempt to read the resolvers at the same time only results in a 2% error, which is acceptable for the velocity and acceleration calculations. The interrupt also sets the "new-data" flag which notifies the watchdog program of the 50 ms interrupt. The watchdog program converts the digital resolver readings into the six joint angles in radians, which are then used in the forward transform, and the velocity and acceleration calculations. The conversion formula is:

$$A = (R * B / G) + O \qquad (1)$$

where "A" is the joint angle, "R" is the integer resolver data, "B" is the radian bit weight, "G" is the gear ratio, and "O" is the resolver offset. The constants in the formula are chosen so that the joint angles produced are compatible with the forward transform. The radian bit weight is determined by the precision of the R/D converters, and is the same for all of the joints.

$$B = 2\ \text{pi} / 2 ** 14 = 0.000383495\ \text{rad/bit} \qquad (2)$$

The gear ratio accounts for the gearing in the PAU between the robot joint and the resolver. The gear ratios for the six joints are given in Table 1. The gear ratios in this table have a negative sign if the positive direction of rotation of the resolver and the forward transform rotation conventions do not agree. This is discussed further in the next section. The resolver offset is used to make the joint angles agree with the conventions of the forward transform. The exact values of the offsets depend on the physical orientation of the resolver with respect to the joint, and have to be recalculated any time this orientation is changed, e.g., when the PAU containing the resolver is removed or adjusted. The formula for calculating the resolver offset is:

$$O = H - (R * B / G) \qquad (3)$$

where "H" is the known angle at which "R" was read. This is done at the T3's "home" position. The T3 can be made to servo precisely to this "home" position where these readings are made.

Table 1. Gear ratios for the six joint resolvers.

Joint Name	Gear Ratio
Base	8
Shoulder	-8
Elbow	-4
Pitch	4
Yaw	-4
Roll	4

7. Tool Point Tranformations

In addition to checking the motion of the individual joints, it is important to check the motion of the tool point, i.e., the point at the end of the robot gripper where an object is grasped. Since the motion of this point is a combination of the motion of all the joints, it is not measured directly but rather calculated. The formula used to compute the location of the tool point is called the forward transform. The forward transform is based on the physical geometry of the robot, including how it is jointed and how long the linkages are between the joints. The forward transform is a mapping from joint space to robot space. The joint space is defined by the joint angles of the robot, and the robot space is defined by the Cartesian coordinates of the tool point and the orientation given by three angles of rotation about the x, y, and z axes.

The forward transform can be considered as a combination of simple transformations. Each translation and rotation can be thought of as a transformation from one coordinate system to another. Each linkage of the robot is thought of as having its own coordinate system, and then the transformation to change from one coordinate system to another is a simple translation and rotation. By combining these simple transformations, a composite transformation that goes from the tool point coordinate system to the robot coordinate system can be constructed.

The present version of the safety computer only needs the position of the tool point, not the orientation. This can be obtained by applying the transformation just described to the tool point. This is the origin in the tool coordinate system. In the safety computer, the individual linkage transformations are applied to the tool point to transform the coordinates of this point from the tool coordinate system to the robot coordinate system.

Applying each of the rotations and translations in turn is more computationally efficient for a single point than constructing a transformation matrix from the product of the individual linkage transformations. This is not the case if several points need to be transformed or if orientation data are necessary.

In the interests of computational efficiency, the rotation and translation computations have been simplified as much as possible. Translations take the form of a simple addition, and rotations are specified around one of the three coordinate axes. The formula for rotating a point around an axis is given by:

$$U' = U * \mathrm{Cos}(angle) - V * \mathrm{Sin}(angle) \qquad (4)$$

$$V' = U * \mathrm{Sin}(angle) + V * \mathrm{Cos}(angle) \qquad (5)$$

Axis of rotation	U	V
x-axis	y	z
y-axis	z	x
z-axis	x	y

where the appropriate values are substituted for U and V depending upon the axis about which the point is being rotated. These simple transformations are sufficient to describe the articulation of the T3. The sequence of transformations necessary to transform a point from tool coordinates to robot coordinates is given in Table 2. The forward transform is written assuming that all the angles are zero when the T3 is stretched-out parallel to the floor. In practice, this configuration is physically impossible for the T3 to obtain due to the limited range of travel of the elbow joint.

The current version of the forward transform takes about 5 ms to execute for one point. This computation is done for every cycle of the watchdog program to calculate the position of the tool point.

8. Velocity and Acceleration Calculations

The velocities and accelerations used by the safety computer are computed in a relatively straightforward fashion. The velocity and acceleration are currently calculated using the following first order linear interpolations:

$$V2 = (P2 - P1) / T \qquad (6)$$

$$A2 = (V2 - V1) / T \qquad (7)$$

where "P1" and "P2" are the previous and current positions, "V1" and "V2" are the previous and current velocities, "A2" is the current acceleration, and "T" is the interval between samples, or 50 ms. The joint velocities and accelerations are signed scalar quantities, with the sign indicating the direction of rotational motion. The tool point velocity and acceleration are vector quantities specified by their components along the three coordinate axes. The computation for the vector quantities is essentially identical to that for the six joints. The calculations are done in a scalar fashion for each of the three coordinates (x, y, z), so that:

$$Vx2 = (Px2 - Px1) / T \tag{8}$$

$$Ax2 = (Vx2 - Vx1) / T \tag{9}$$

$$Vy2 = (Py2 - Py1) / T \tag{10}$$

$$Ay2 = (Vy2 - Vy1) / T \tag{11}$$

$$Vz2 = (Pz2 - Pz1) / T \tag{12}$$

$$Az2 = (Vz2 - Vz1) / T \tag{13}$$

The magnitude of the tool point velocity and acceleration vectors are used for comparison purposes.

These simple formulas work well as long as the measurements have sufficient precision and are relatively noise free. Fortunately, the resolvers fit this criterion nicely. The resolvers have almost no noise and enough precision to pick up very small changes in position. If this was not the case, some type of averaging would have to be done to reduce the effects of noise in the joint position data.

Table 2. Forward transform sequence - tool to robot coordinates.

Transformation	Description of Argument
x translation	tool point to roll joint length
x rotation	roll joint angle
x translation	roll joint to yaw joint length
z rotation	yaw joint angle
x translation	yaw joint to pitch joint length
y rotation	pitch joint angle
x translation	pitch joint to elbow joint length
y rotation	elbow joint angle
x translation	elbow joint to shoulder joint length
y rotation	shoulder joint angle
z rotation	base joint angle

9. Forbidden Volume Algorithms

The forbidden volume routine provides a method for the safety computer to impose restrictions on the positioning of the tool point. In a sense, it is the three dimensional analog of the joint limits. The forbidden volume routine allows the safety computer to prevent the robot from damaging itself by colliding with some object in its work space.

In essence, the forbidden volume routine uses a simple "world model" to predict when the robot is in danger of colliding with an object. This world model consists of a number of "forbidden volumes" which correspond to real objects in the work space. When the robot is about to run into something, it will cross the surface boundary of the object's volume. The volume can be defined larger than the object so that the robot reaches the surface boundary of the forbidden volume before colliding with the object. By defining the forbidden volume to be larger than the actual object, a safety margin is provided to allow for the time it takes the safety computer to perform its calculations, detect an error, and output the appropriate control signal to the T3, and for the robot to actually stop. This response time and the velocity of the robot will determine how far the robot will travel before being stopped, and thus, the necessary size of the safety margin. Since the distance traveled in this time is a function of the velocity of the robot, the approach taken in the safety computer is to make the safety margin for the forbidden volumes a function of velocity. Thus, the faster the T3 approaches a forbidden volume, the larger the safety margin will be to account for the longer distance required to stop motion.

A forbidden volume is defined as a collection of planes, or more accurately, as a conjunction of half spaces. A half space is bounded by a plane. Every point on one side of the plane is "inside" the half space, and every point on the other side is "outside" the half space. Thus, the forbidden volume is the region that is inside every one of a collection of half spaces. Using enough half planes in this manner, any convex object can be formed. (An object is convex if all of its surface planes meet in an outward pointing edge, like those of a cube.) If the object is not convex, it can be defined by a number of smaller convex objects. The actual complexity of the volumes defined in this manner is limited by memory space and available computation time of the safety computer. The processing of the preliminary forbidden volumes, which includes seven objects and a total of 17 planes, takes the forbidden volume routine about 12 ms. This fits easily into the 50 ms timing cycle.

The forbidden volumes are stored as a list of planes, given by their plane constants, and grouped into objects. Each plane is specified by the four plane constants in the equation:

$$A\,x + B\,y + C\,z = D \tag{14}$$

where "A", "B", "C", and "D" are the plane constants, and "x", "y", and "z" are the coordinates of points on the plane when they satisfy the equation.

The actual half space formula is specified by the inequality:

$$A\,x + B\,y + C\,z \leq D \tag{15}$$

Given a point (X,Y,Z), the point lies in the half space if it satisfies the half space inequality given above. The forbidden volume routines substitute the robot's tool point into the half space inequality. If it satisifies the equation, then the tool point is within the half plane. If it satisifies the equation for all of the object's half spaces, it is within the forbidden volume.

The half spaces for the objects are generated using four points. Three points are sufficient to define the plane, and the fourth point is used to define which side of the plane is on the outside of the object. These points can be obtained conveniently by having the robot touch off the points defining the planes on the actual objects. The difference between the first and second, and the first and third, points are taken to produce two vectors which are oriented parallel to the surface of the plane:

$$POP1 = P1 - P0 \tag{16}$$

$$POP2 = P2 - P0 \tag{17}$$

where "P0", "P1", and "P2" are the three points on the plane, and "POP1" and "POP2" are the two vectors parallel to the plane. If the cross product of the two vectors parallel to the plane is taken, the resultant is a vector normal to the plane:

$$N = POP1 \times POP2 \tag{18}$$

where "N" is the vector normal to the plane. This normal vector is the basis for the coefficients "A", "B", and "C", which form the normalized normal vector for the plane. The fourth coefficient, "D", is found by taking the dot product between one of the points on the plane and the normal vector:

$$D = N \bullet P0 \tag{19}$$

or,

$$D = Nx\ P0x + Ny\ P0y + Nz\ P0z \tag{20}$$

which is the same form as the plane equation itself, with "Nx"="A", "Ny"="B", and "Nz"="C". Thus, "N" gives the values for the plane coefficients "A", "B", and "C". All of the coefficients are then scaled so that the normal vector becomes a unit vector. This is accomplished by dividing all the coefficients by the factor:

$$SQRT\ (A*A + B*B + C*C) \tag{21}$$

which is the magnitude of the normal. This has the effect of scaling the equation so that a simple addition of a constant to the parameter "D" will shift the plane by an equal amount in the direction of the normal. This allows a "safety margin" to be added directly onto "D" to shift the plane of the volume out from the actual surface. The final step in determining the coefficients is to take the orientation of the half space into account. The orientation is determined by the fourth "outside point." The outside point is substituted into the half plane inequality (Equation 15) to see if the half plane is oriented correctly. If the inequality is satisfied by the outside point then all of the coefficients must be negated to make the inequality agree with the known facts. The effect of negating the coefficients is to reverse the half space so that what was inside is now outside, and vice versa.

As mentioned earlier, the safety margin is defined as a function of velocity. This velocity dependence is accounted for by modifying the value of "D". At present this is done by taking the component of the velocity normal to the plane (obtained by taking the dot product of the velocity and the plane normal), scaling it, and adding the magnitude of the result to "D". This has the effect of adding a velocity dependent safety margin to all of the forbidden volumes, effectively causing the volumes to elongate in the direction of motion. The scaling factor is chosen empirically to provide enough stopping distance between the detection point and the actual volume to prevent a collision.

10. Conclusions

The safety computer was initially tested in November 1983. It is now used on a daily basis while other systems are being tested. Although the safety computer has undergone only a limited number of hours of operation, it has proved to be quite reliable and extremely useful during the development stages of other systems. This was particularly true for integration testing of the NBS RCS with other sensor systems, such as the NBS vision system, where movement of the robot was necessary to verify proper operation.

One of the primary requirements that influenced the design of the safety computer was that it must be easy to operate. This is particularly important since personnel who are not intimately familiar with the system will be using it while conducting tests on the T3. The procedures to start up the safety computer and to recover from an error condition are straightfoward. Self-explanatory prompts are displayed on the user terminal making it simple to modify the velocity, acceleration and stopping threshold parameters and to understand what error condition was detected.

A number of additions to the capabilities of the safety computer are already planned. A parallel communications link between the safety computer and the NBS RCS is in the process of being added. This link will enable the safety computer to verify that the RCS is operating properly, and vice versa. The RCS will also be able to send the safety computer the goal point, i.e., the point to which the robot is being sent. Thus, the safety commputer will be able to determine that the robot is about to enter a forbidden volume before it actually gets there. This may eliminate the need for a safety margin, or at least permit it to be reduced.

There are plans to increase the sophistication of the forbidden routines by including checks on other parts of the arm besides the tool point, such as the wrist. The current model of the robot used in these routines is a stick figure with pivot points corresponding to each joint. It may be possible to use a solid model of the robot if the computational time is not too great. By using a solid model, a protective envelope could be constructed around the robot arm to prevent sensors and other hardware mounted there from entering forbidden volumes and potentially being damaged. These and other enhancements will be tested in future versions of the safety computer.

11. Bibliography

1. Anon., Operating Manual for the Cincinnati Milacron T3 Industrial Robot, Publication No. 1-IR-79149, Cincinnati Milacron Company, Cincinnati, OH, June 1980.

2. Anon., Service Manual for the Cincinnati Milacron T3 Industrial Robot, Publication No. 3-IR-80147, Cincinnati Milacron Company, Cincinnati, OH, 1981.

3. Anon., CPU-Specific Documentation for polyFORTH on the Intel 8086/12, User's Supplement to the polyFORTH Reference Manual, FORTH, Inc., Hermosa Beach, CA, 1980.

4. Anon., Component Data Catalog, Intel Corporation, Santa Clara, CA, 1980.

5. Anon., The 8086 Family User's Manual, Numerics Supplement, Intel Corporation, Santa Clara, CA, 1980.

6. Barbera, A. J., Fitzgerald, M. L., Albus, J. S., and Haynes, L. S., RCS: The NBS Real-Time Control System, To be presented at the Robots 8 Conference and Exposition, Detroit, MI, June 4-7, 1984.

7. Kilmer, R. D., Safety Sensor Systems for Industrial Robots, Proceedings of the Robots VI Conference and Exposition, Detroit, MI, March 1-4, 1982.

8. Simpson, J. A., Hocken, R. J., and Albus, J. S., The Automated Manufacturing Research Facility of the National Bureau of Standards, Journal of Manufacturing Systems, <u>1</u> (1), pp. 17-32 (1982).

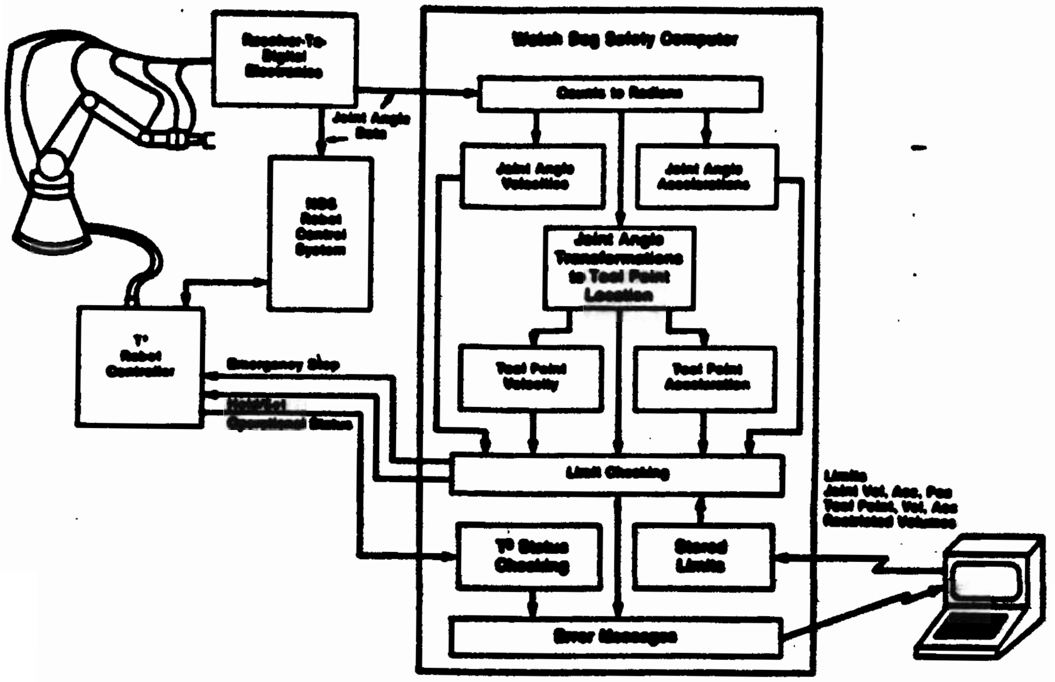

Resolver-To-Digital Electronics
Watch Dog Safety Computer
Counts to Radians
Joint Angle Data
NBS Robot Control System
Joint Angle Velocities
Joint Angle Accelerations
Joint Angle Transformations to Tool Point Location
T3 Robot Controller
Emergency Stop
Hold/Cnt Operational Status
Tool Point Velocity
Tool Point Acceleration
Limit Checking
Limits Joint Vel, Acc, Pos Tool Point, Vel, Acc Restricted Volumes
T3 Status Checking
Stored Limits
Error Messages

Figure 1. Watchdog Safety Computer implementation on the T3.

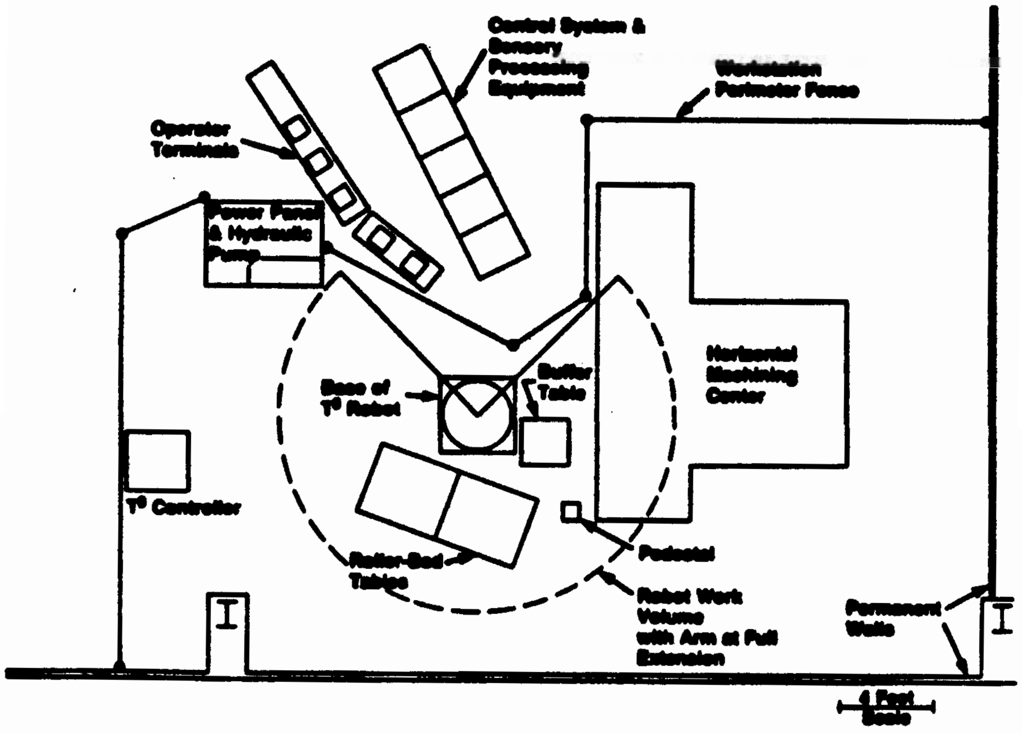

Control System & Sensory Processing Equipment
Workstation Perimeter Fence
Operator Terminals
Power Panel & Hydraulic Pump
Horizontal Machining Center
Base of T3 Robot
Buffer Table
T3 Controller
Pedestal
Roller-Bed Tables
Robot Work Volume with Arm at Full Extension
Permanent Walls
4 Feet Scale

Figure 2. Layout of the horizontal machining workstation in the AMRF.

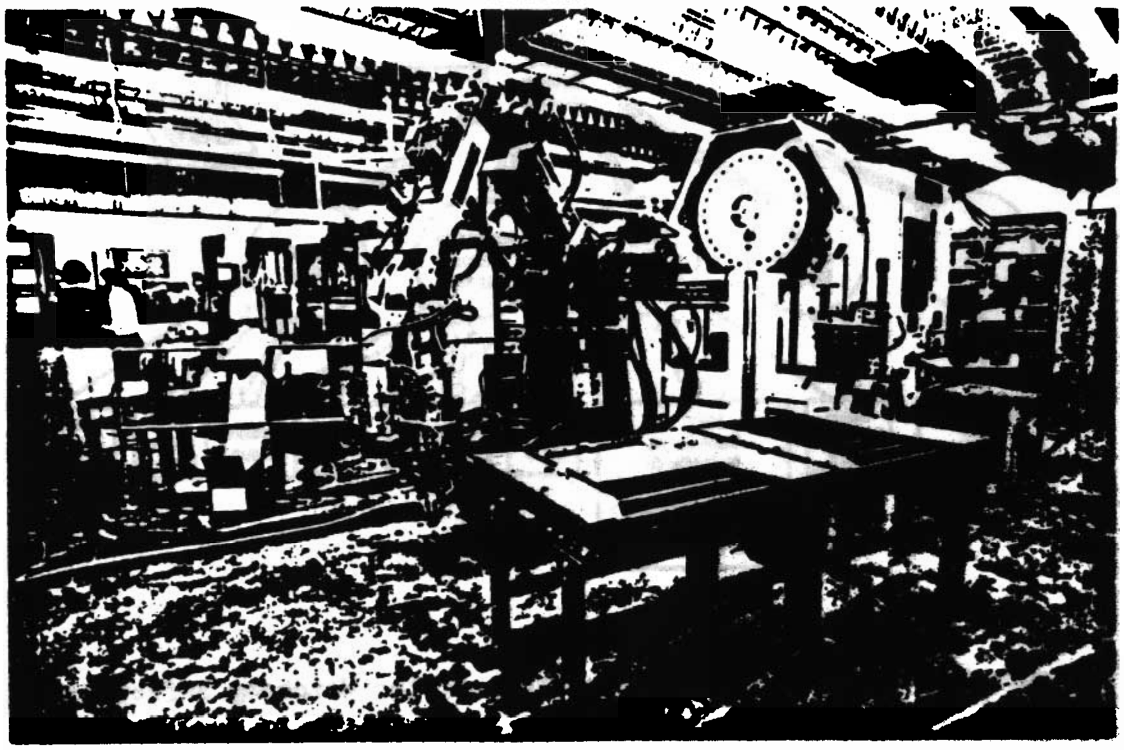


Figure 3. View of the T3 robot as installed in the AMRF.

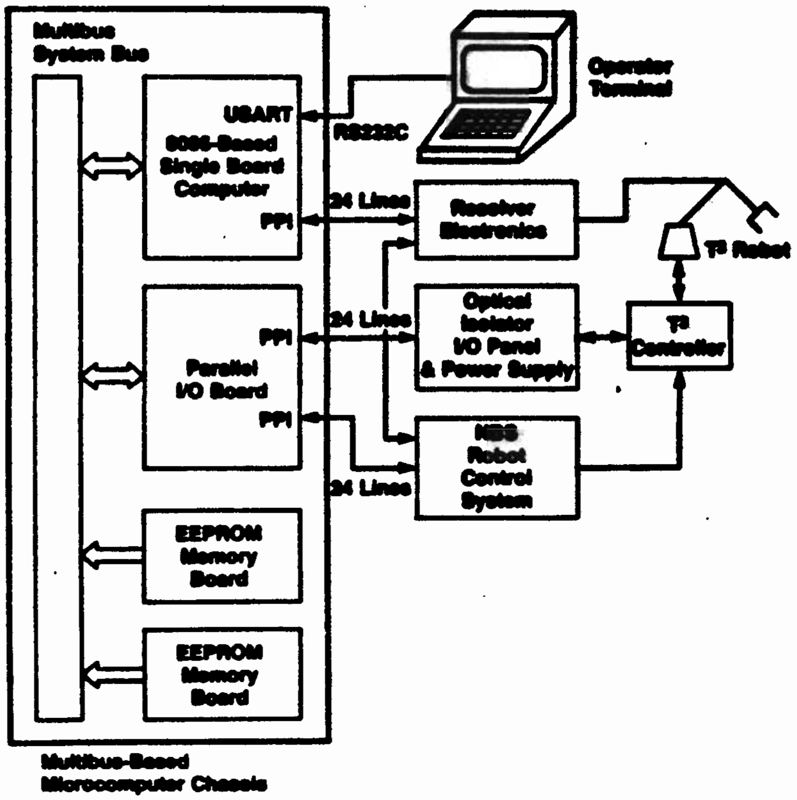


Figure 4. Watchdog Safety Computer hardware configuration.

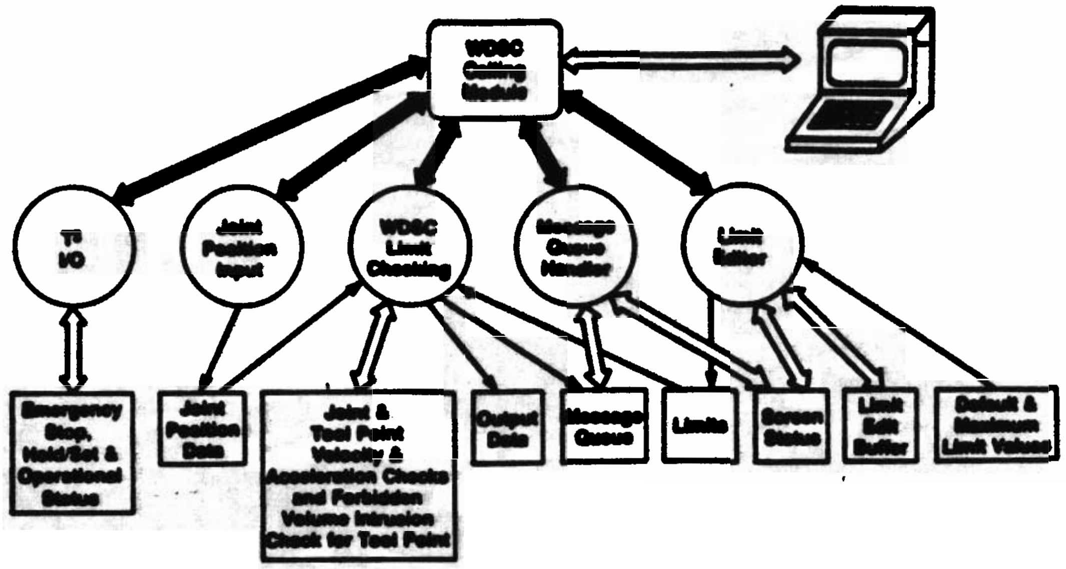
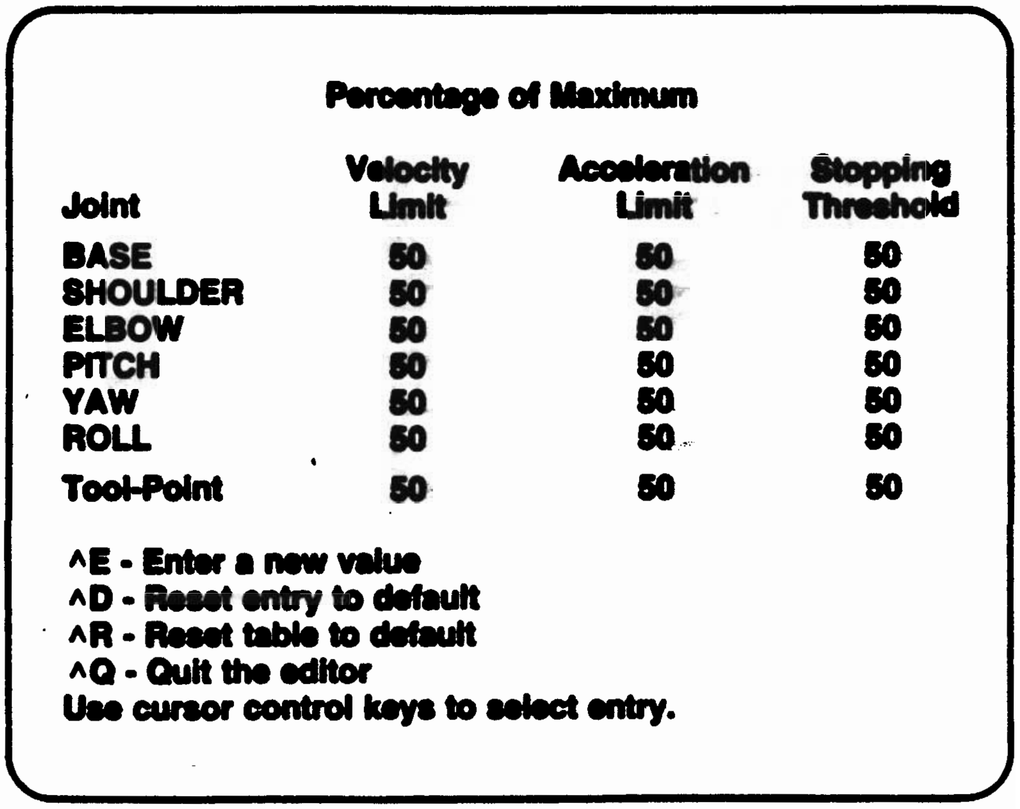


Figure 5. General software structure of the Watchdog Safety Computer.

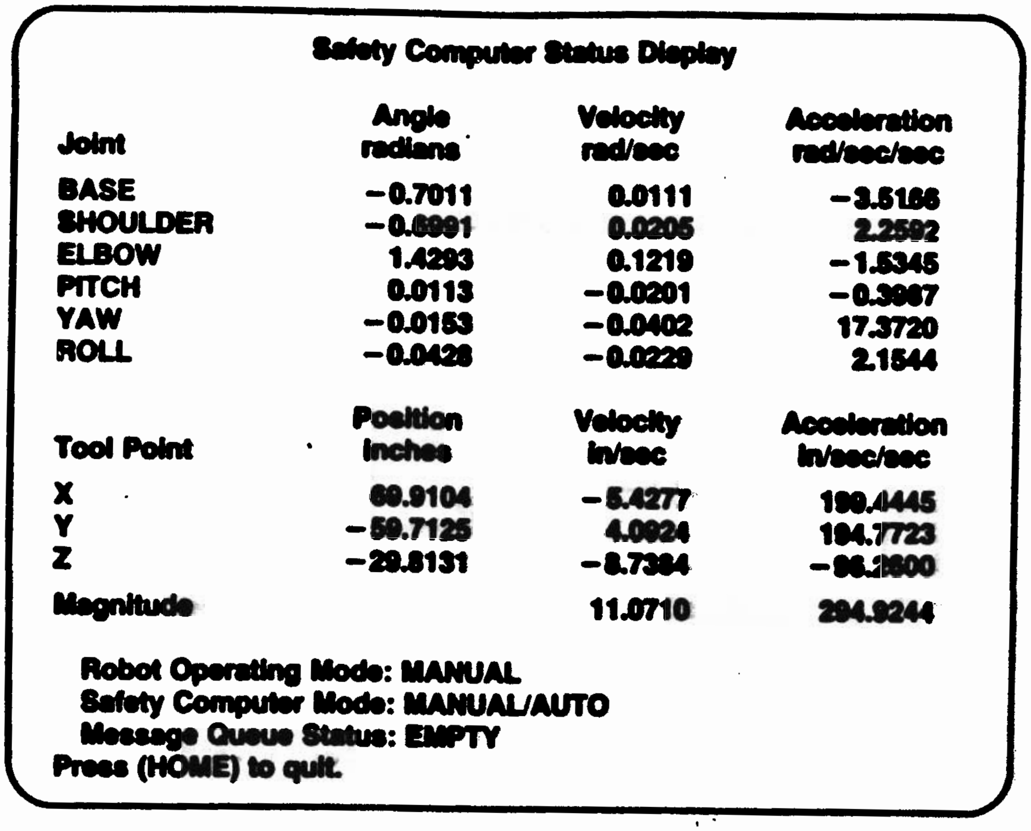


Percentage of Maximum

Joint	Velocity Limit	Acceleration Limit	Stopping Threshold
BASE	50	50	50
SHOULDER	50	50	50
ELBOW	50	50	50
PITCH	50	50	50
YAW	50	50	50
ROLL	50	50	50
Tool-Point	50	50	50

^E - Enter a new value
^D - Reset entry to default
^R - Reset table to default
^Q - Quit the editor
Use cursor control keys to select entry.

Figure 6. User terminal display for editing limits.



```
                      Safety Computer Status Display

                         Angle         Velocity      Acceleration
Joint                    radians       rad/sec       rad/sec/sec
BASE                     -0.7011        0.0111         -3.5166
SHOULDER                 -0.6991        0.0205          2.2592
ELBOW                     1.4293        0.1219         -1.5345
PITCH                     0.0113       -0.0201         -0.3967
YAW                      -0.0153       -0.0402         17.3720
ROLL                     -0.0428       -0.0229          2.1544

                         Position      Velocity      Acceleration
Tool Point               inches        in/sec        in/sec/sec
X                        69.9104       -5.4277        199.4445
Y                       -59.7125        4.0924        194.7723
Z                       -29.8131       -8.7384        -96.2600
Magnitude                              11.0710        294.9244

  Robot Operating Mode: MANUAL
  Safety Computer Mode: MANUAL/AUTO
  Message Queue Status: EMPTY
Press (HOME) to quit.
```

Figure 7. User terminal display for examining the motion of the T3.

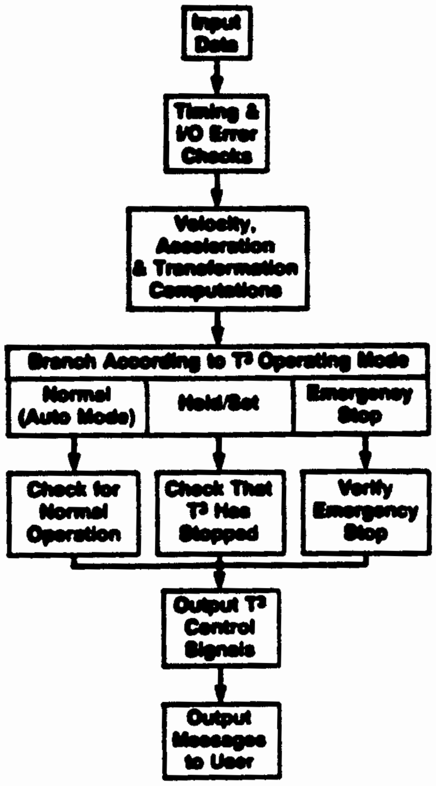

Input Data
Timing & I/O Error Checks
Velocity, Acceleration & Transformation Computations
Branch According to T³ Operating Mode
Normal (Auto Mode)
Hold/Set
Emergency Stop
Check for Normal Operation
Check That T³ Has Stopped
Verify Emergency Stop
Output T³ Control Signals
Output Messages to User

Figure 8. Block diagram of the watchdog program in the WDSC.

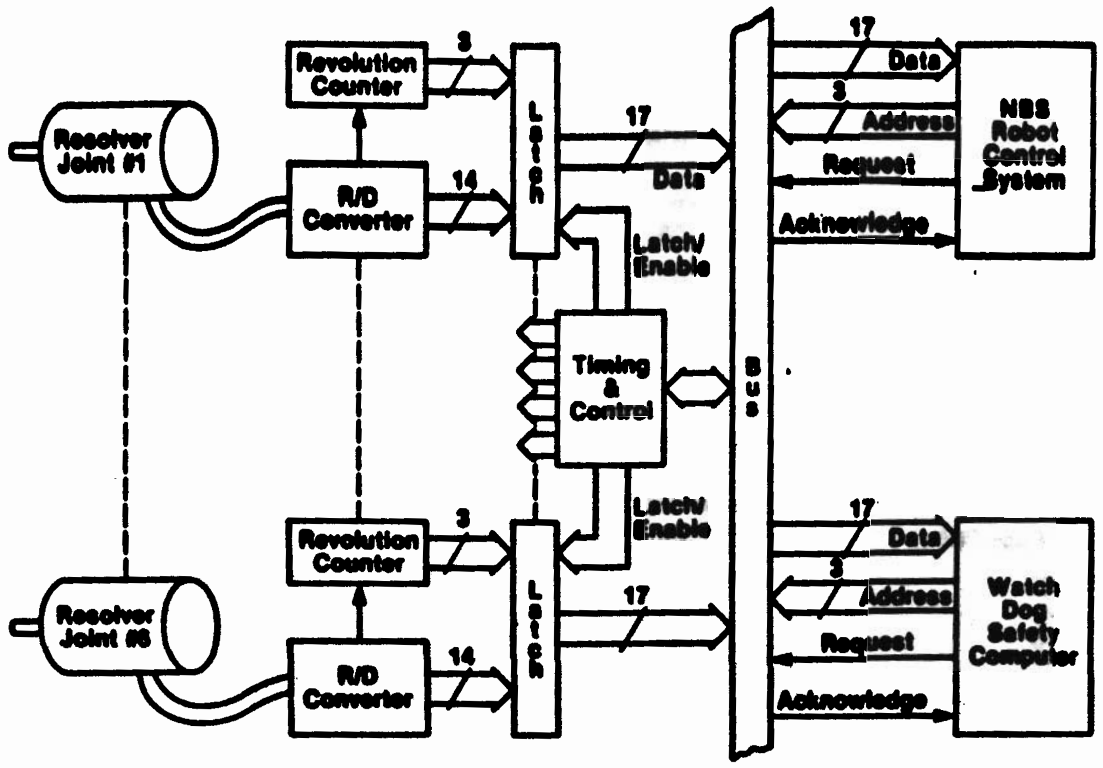


Figure 9. Block diagram of the resolver electronics.