12/26/92

IN-63-CR

140696

P 154

FINAL REPORT FOR: A TELEROBOTIC DIGITAL CONTROLLER SYSTEM

CONTRACT NO: NAS5-30633

PREPARED BY:
The NAVTROL Company, Inc.
9204 Markville Drive
Dallas, Texas 75243

SEPTEMBER 30, 1992

PREPARED FOR:
NASA GODDARD SPACE FLIGHT CENTER
Greenbelt Road
Greenbelt, MD. 20771

PROPRIETARY NOTICE

SBIR RIGHTS NOTICE (APRIL 1985)

This SBIR data is furnished with SBIR rights under NASA Contract No. NAS5-30633. For a period of 2 years after acceptance of all items to be delivered under this contract the Government agrees to use this data for Government purposes only, and it shall not be disclosed outside the Government (including disclosure for procurement purposes) during such period without permission of the Contractor, except that, subject to the foregoing use and disclosure prohibitions, such data may be disclosed for use by support contractors. After the aforesaid 2-year period the Government has a royalty-free license to use, and to authorize others to use on its behalf, this data for Government purposes, but is relieved of all disclosure prohibitions and assumes no liability for unauthorized use of this data by third parties. This Notice shall be affixed to any reproductions of this data, in whole or in part.

The specific data to which this notice applies is contained on pages 2-12 through 2-18, 2-24 through 2-30, 2-32 through 2-40 and 4-1 through 4-10.

TABLE OF CONTENTS

TABLE OF CONTENTS (Cont.)

TABLE OF CONTENTS (Cont.)

LIST OF FIGURES

LIST OF TABLES

1 Introduction

This is the Final Report on NASA GSFC Contract NAS5-30633, which is an SBIR Phase II Contract for "Development of a Telerobotic Digital Controller System." This system is a network of joint mounted Dual Axes Digital Servo Controllers, providing control of various joints and end effectors of different Robotic systems. This report provides description of and User required information for the Digital Controller System Network (DSCN) and, in particular, the Dual Axis Digital Servo Controllers (DDSC), Model DDSC-2, developed to perform the controller functions. This highly capable, very flexible, programmable servo-controller, contained on a single, compact PCB measuring only 4.5" x 5.1" in size, is applicable to control systems of all types from sub arc second precision pointing to control of robotic joints and end effectors. This document concentrates on the robotic applications for the DDSC. This development was accomplished by The Navtrol Company, Inc., Dallas, Texas.

Each DDSC can provide precise control for both the Gripper and Nut Runner functions of an End Effector. Alternately, each DDSC can provide control of two joints of the Robot. Wiring within a robotic arm is minimized, as only four wires total, two power and two communications, are required for interconnection of as many as seven of these controllers.

Receiving, through its extensive I/O, analog or discrete information from End Effector mounted sensors, such as strain gauges or potentiometers, a DDSC generates PWM DC current, 0 to 8A, to control the two End Effector motors and provide independent force, velocity or position control for the two loops. It can use Hall device signals, intended for motor commutation, to derive motor shaft velocity and position information. One to seven controllers can operate autonomously or collectively, interchanging data and commands with a Supervisory computer over a single differential, high speed (5 MHz) serial interface. The DDSC requires only DC motor voltage (i.g. 28 VDC) for power and consumes, excluding motor power, only 3.7 watts. Sophisticated estimation and control algorithms provides smooth, precise, versatile control. The highly intelligent controller is able to perform transformations and derive functions while providing control.

Utilizing different sets of algorithms, the DDSC can utilize data from incremental or absolute angle encoders, resolvers, or potentiometers to provide control for two robotic joints. A Digital Servo Control Network (DSCN) of four of these controllers can provide control for a seven joint robotic arm plus an end effector. A DDSC can be mounted adjacent to or in between the joints which it controls. These controllers find application in industrial and military environments as well as space.

A network consisting of 5 DDSC controllers, a Master Controller board and a complete Supervisory Computer (Everex 386 Step computer with Math co-processor and monitor) was delivered, demonstrated and accepted at NASA GSFC. Simultaneous control of the 5 dual axis controllers with independent programs was demonstrated.

Delivery included servo development system software contained within the supervisory computer. This Development Monitoring system, referred to as DEMON, allows total monitoring of the controllers providing both graphical and alpha numeric displays. Adjustment of gains and other parameters in terms familiar to control engineers is easily accomplished through DEMON. Also facilitated is the commanding of various functions for each of the DDSC controller slaves. All of these capabilities were demonstrated to GSFC personnel.

Much earlier in the contract and prior to delivery of the above system, a "preliminary prototype" controller was delivered and demonstrated to simultaneously control NASA GSFC'S gripper and nut runner. All hardware and software deliveries required on the contract are complete.

Section 2 of this report describes in detail the Dual Digital Servo Controller (DDSC) which provides joint and/or end effector control and the Supervisory Controller which interfaces with the human operator. Section 3 provides a User's Guide for both hardware and software in applying the DDSC. Section 4 describes the robotic control algorithms. Also provided was a complete User's Manual for DEMON, NAVTROL's servo-development software system.

Section 5 provides test results for the program. The principle test results were the demonstration and the complete Acceptance Test ran with NASA GSFC as a witness. An Acceptance Test goes beyond what is usually required for a cost limited SBIR development program. It was not required by the contract. However, the Technical Director of the program requested it and Navtrol complied, although it extended the contract and resulted in expenditure of funds beyond that authorized by the contract. Section 5 discusses the Acceptance Test and presents some results which demonstrate the capability of the DDSC to provide excellent motor current control, precision control of gripper force and position and control of nut runner speed and torque.

Conclusions and recommendations are provided in Section 6. The primary conclusion is that the development was a success and the Telerobotic Digital Controller System is available for application. As part of a "Phase III" effort, Navtrol is now repackaging the DDSC within a aluminum case whose size is only 3" x 4 1/4" x 2 1/8". The power output stages have been redesigned to provide 15 amp continuous current and 20 amp. peak. These units will be delivered to Robotic Research Corporation for inclusion in robots to be delivered to the Oakridge, Tennessee, DOE facility. Navtrol has also presented this system to the University of Texas Automation and Robotics Research Institute for use on their "Stewart Platform" robot. UTA engineers have recommended use of this controller with their Stewart platform robot to both Air Force and Navy personnel.

There is no controller on the market that competes in size and capability with the controller developed on this program. Navtrol is justifiably proud of this development, as should be NASA GSFC.

2. DDSC and DSCN Descriptions
2.1 DDSC Description
2.1.1 Overall DDSC

Figure 2.1-1 illustrates the Dual Digital Servo Controller (DDSC) PC Board, approximately actual sized. Figure 2.1-2 is the DDSC functional diagram. Figure 2.1-3 illustrates PC Board functions utilized as a Joint Controller. Illustrated is the use of incremental angle encoders for position feedback. Resolvers, absolute angle encoders, potentiometers, or even brushless motor commutation signals can also be used. The "End Effector" model of the DDSC is physically the same as the Joint Controller but now comprises the functions illustrated in Figure 2.1-4. Principal features of the DDSC Controller are provided on Table 2.1-1.

Table 2.1-1: DDSC FEATURES

- PROVIDES COMPLETE SERVO CONTROL FOR TWO (2) AXES INCLUDING COORDINATE TRANSFORMATION.
- INCLUDES KALMAN FILTERING AND OPTIMAL CONTROL ALGORITHMS.
- CONTROLS TWO (2), THREE (3) PHASE BRUSHLESS DC TORQUE MOTORS, AT 28VDC, 8 AMPS. PWM POWER DRIVE OUTPUTS INCLUDED.
- ONLY 28VDC POWER REQUIRED. USES ONLY 3.8 WATTS AT 28 VOLTS (NO LOADS, MOTOR OFF)
- HIGH SPEED (5 MHz) SERIAL COMMUNICATION. REQUIRES ONLY ONE (1) DIFFERENTIAL PAIR OF WIRES.
- 24 S.E. OR 12 DIFF. ANALOG INPUTS TO TEN (10) BIT A/D.
- ANALOG INPUT GAINS ARE SOFTWARE PROGRAMMABLE, 1-128.
- POSITION F.B.: 1) DUAL INCREMENTAL ANGLE ENCODER I/F
  2) DUAL ABSOLUTE ANGLE ENCODER I/F
  3) DUAL ANGULAR RESOLVER I/F
  4) DUAL POTENTIOMETER I/F
- ± 5 VOLTS SUPPLIED FOR EXTERNAL USE.
- FOUR (4) DIFFERENTIAL AMPLIFIERS (EIGHT (8) INPUTS) FOR LOW SIGNAL APPLICATIONS. (i.g. Strain Gauges)
- TWO (2) 12 BIT D/A OUTPUTS
- TWENTY (20) DISCRETE INPUTS. TWO (2) TTL DISCRETE OUTPUTS
- FOUR (4) POWER DISCRETE OUTPUTS, 0.5A, 60V. OPEN DRAIN
- CARD SIZE = 4.5 X 5.1"

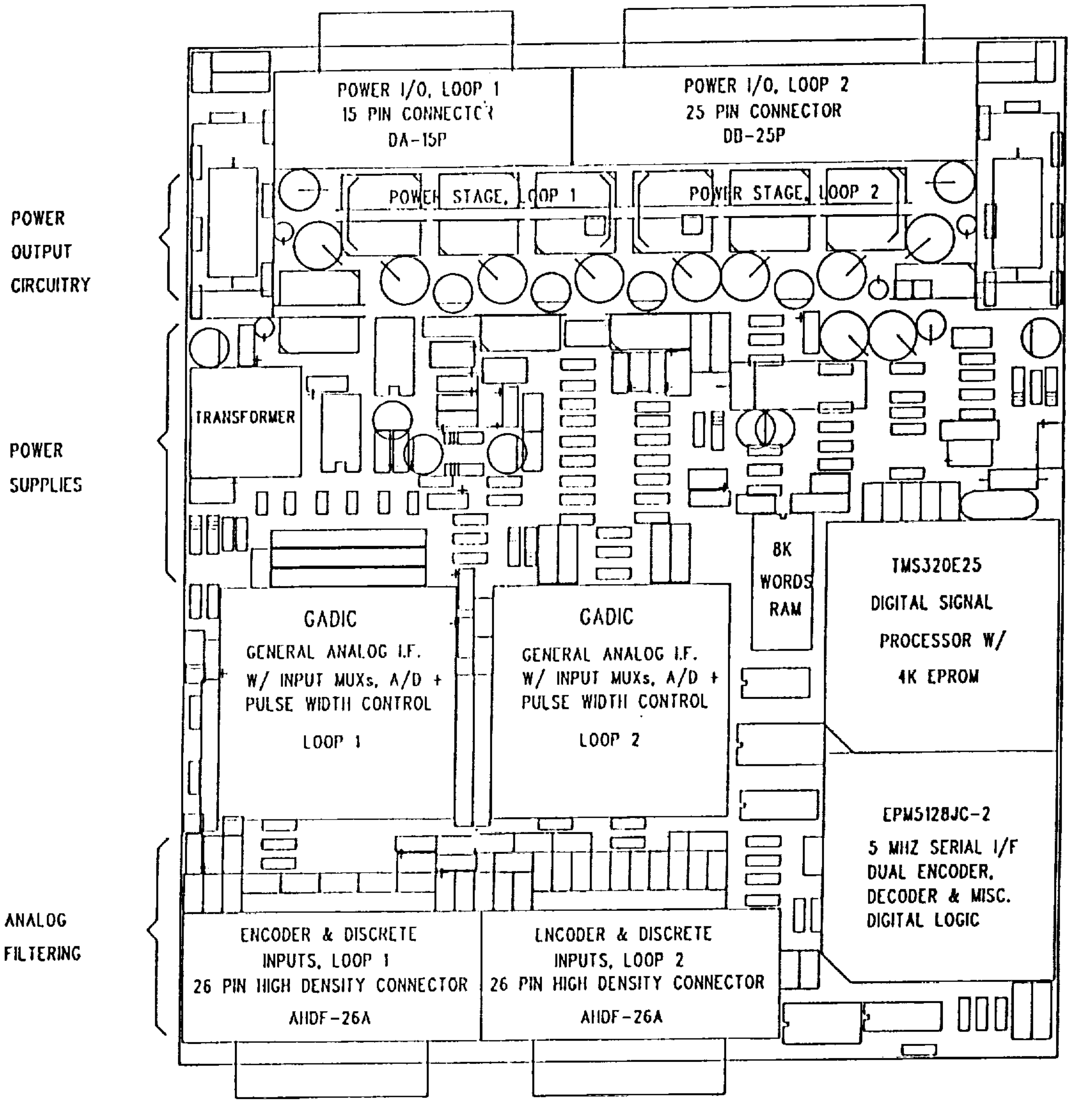


Figure 2.1-1: Dual Digital Servo Controller (DDSC) (Actual Size)

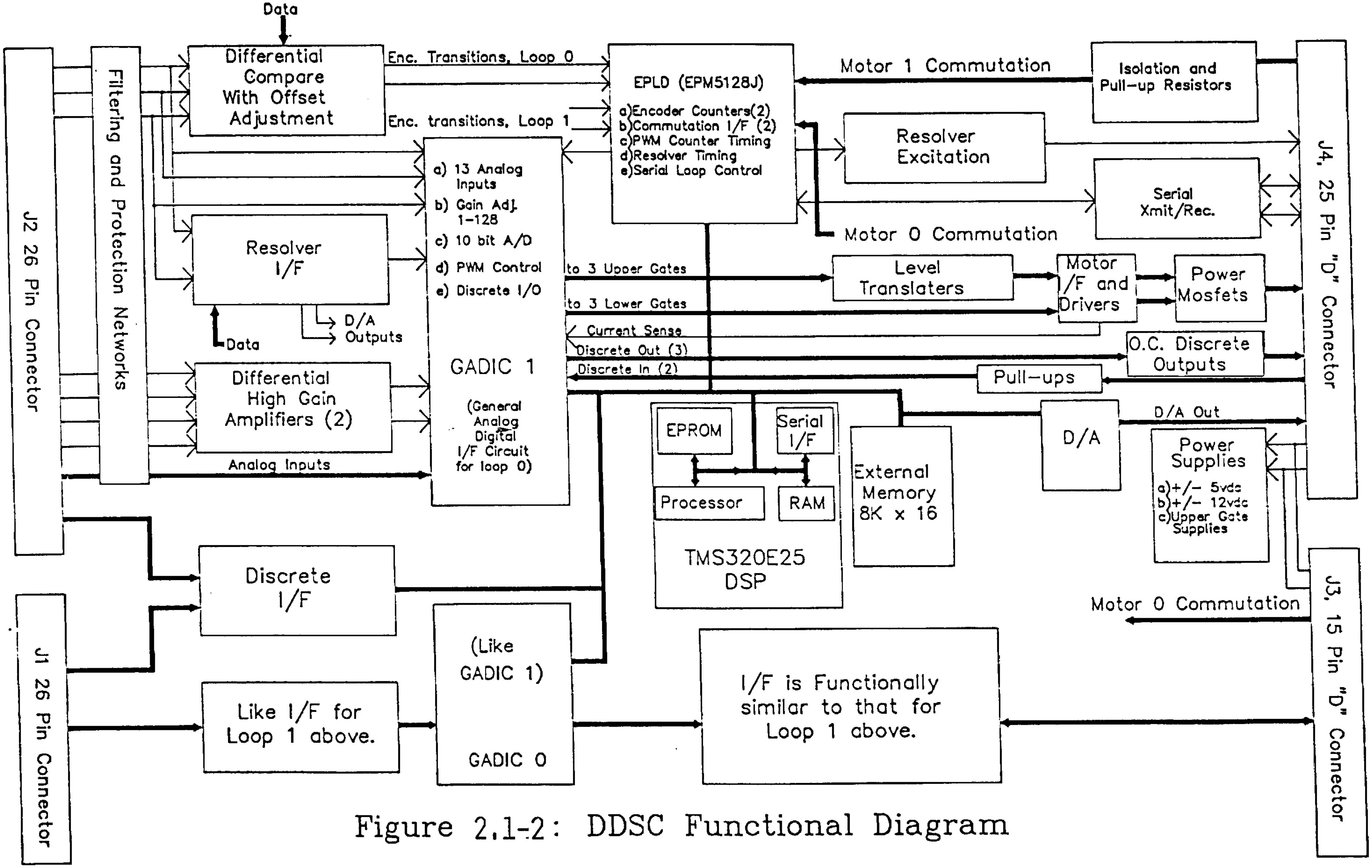


Figure 2.1-2: DDSC Functional Diagram

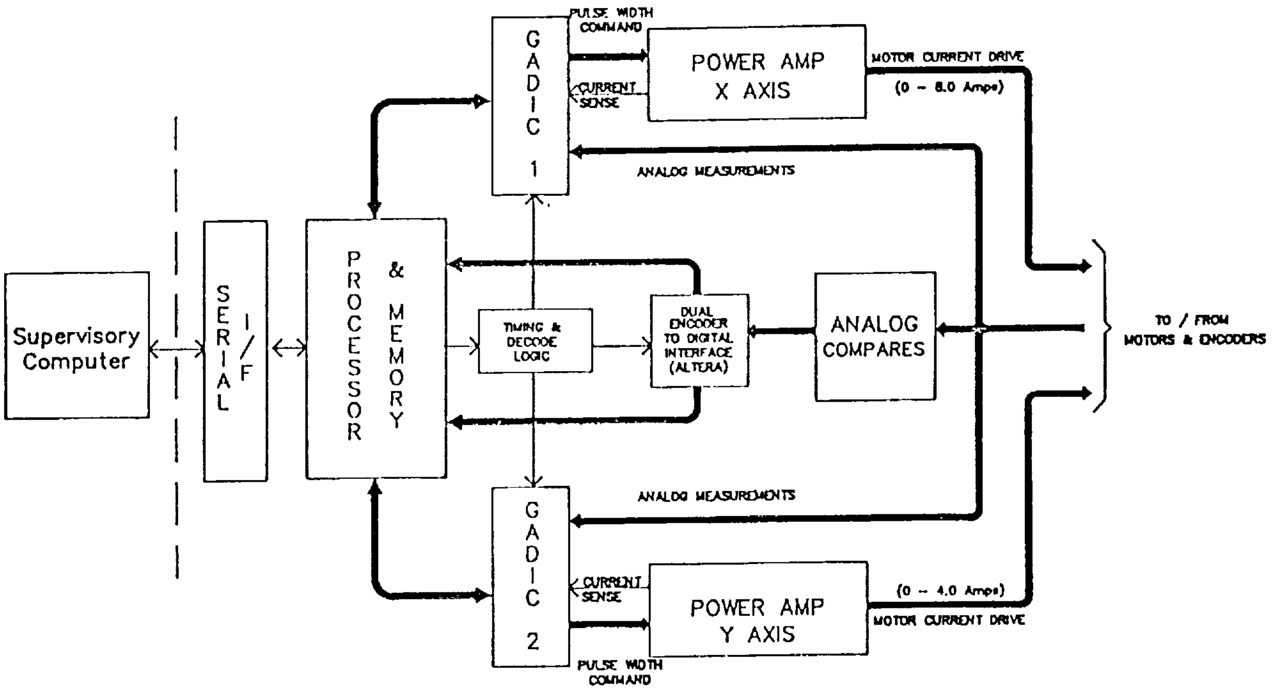


Figure 2.1-3
Joint Controller (With Encoder Feedback)

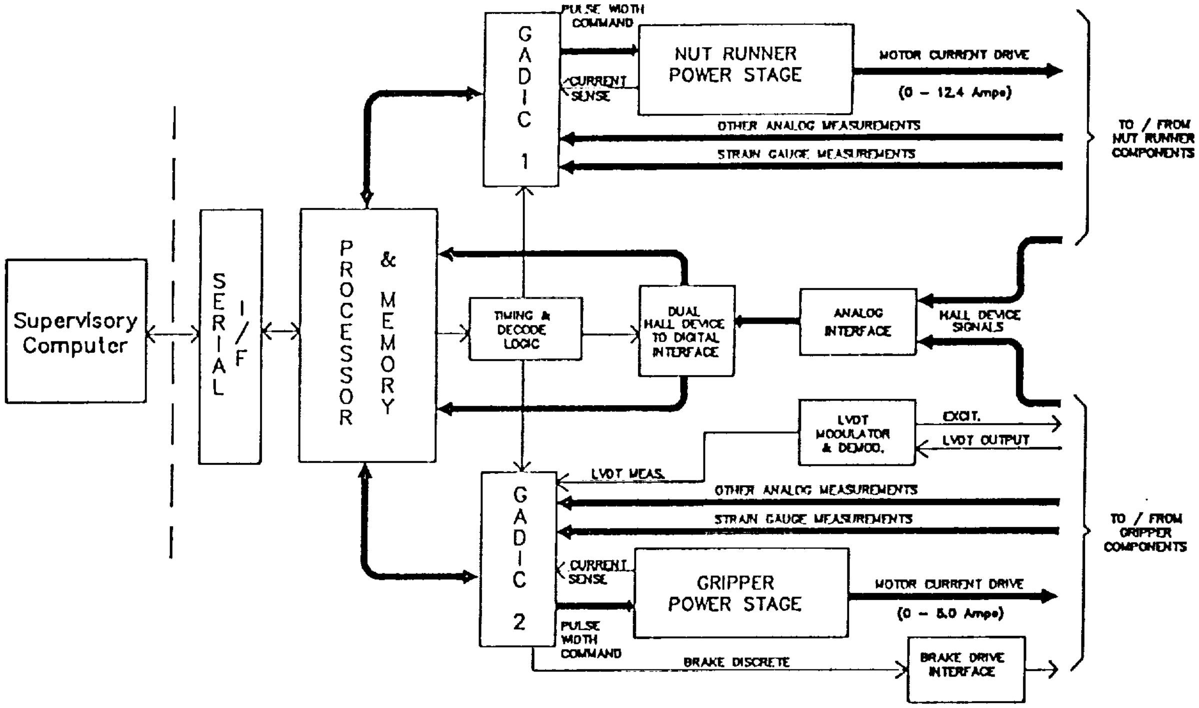


Figure 2.1-4
End Effector Controller Block Diagram

2.1.2 Major DDSC Components

2.1.2.1 TMS320E25 DSP

For computations the DDSC Servo Controllers utilize TMS320E25 Digital Signal Processors (DSP) by Texas Instruments. Digital Signal Processors have considerable computational advantage and are much more highly integrated than conventional micro processors. The TMS320E25 includes within a single chip, 4K words of EPROM program memory, 544 words of data memory, a hardware multiplier, 1-16 bit barrel shifter and a 32 bit accumulator. It also includes a serial I/O utilized for serial communications. Utilizing such a chip greatly reduces the hardware required to implement the Digital Servo Controllers.

2.1.2.2 General Analog/Digital Interface Circuit (GADIC)

Another IC, developed specifically for applications on the DDSC, is Navtrol's General Analog/Digital Interface Circuit, referred to as GADIC. This monolithic integrated circuit provides twelve (12) analog inputs, which can be allocated to any combination of differential or single ended analog inputs. Input gains are programmable, 1 to 128. Included is a nine (9) bit plus sign A/D converter. This A/D provides 6.6 micro second conversion time, including settling, when bypassing the multiplexed and programmable gain sections and 13.2 micro seconds when utilizing them. The GADIC also provides interface circuitry for a brushless DC motor including commutation logic and output power switch control. The chip provides programmable pulse width modulated (PWM) control and timing (10 bits plus sign) operating at 40,960 pulses/second. Automatic shutdown of the motor drive transistors is provided if the GADIC is not updated by the processor or if over current is sensed. "Fast bypass" circuitry provides conversion of current sense measurements in 4.4 micro seconds after hold, 6.6 µs with settling and hold times included. In addition, this IC provides three (3) discrete outputs and two (2) discrete inputs. Two (2) GADICs are utilized on each DDSC. Table 2.3-1 provides a concise listing of the GADIC's features.

2.1.2.3 Electrically Programmable Logic Device (EPLD)

The third major IC on Navtrol's DDSC is an EPM5128 Electrically Programmable Logic Device (EPLD), programmed to perform the digital functions required by the DDSC. In the incremental encoder version of the joint controller DDSC, sine/cosine analog outputs from two (2) incremental angle encoders provides the two (2) axis precision required. Logic for detecting

MULTIPLEXED A/D CONVERTER SYSTEM

- 12 multiplexed single ended (or 6 differential pair) analog inputs with cross connect, common mode and dual remote sense (channel A or B). ± 3.6 V input range (± 5 V maximum).
- Programmable gain (1, 2, 4, 8, 16, 32, 64 and 128).
- 9 bit High Speed (3 Phase Flash) A/D Converter with sign bit. (6.6 to 13.2 microsecond conversion time).
- Designed for automatic calibration (through software) of common mode, offsets, gain, etc.

BRUSHLESS DC MOTOR INTERFACE CIRCUITRY

- Programmable pulse width modulator control and timing (10 Bits + Sign), with effective switch operation at greater than 40,000 hertz (> 40 KHz).
- Commutation logic and drive transistors with an isolated Vss for the source of the N channel (lower) transistors. Over drive detection (open drain output) and shutdown of motor drive transistors.
- Automatic shut down if not updated by processor.
- Special circuitry for current sense "FAST-BYPASS" (bypasses input analog conditioning circuitry).

ADDITIONAL INPUT/OUTPUT

- Three discrete digital outputs and two discrete digital inputs (TTL or CMOS compatible).

GENERAL OPERATING SPECIFICATIONS

- ±5v operation with TTL and CMOS compatible I/O.
- Low power consumption. Approximately 20 ma @ 10 Mhz.
- Operates with 16 bit parallel data bus.
- Requires one (1) wait state to interface with TMS320X25.

Table 2.3-1 Features of Navtrol's GADIC

and counting the cycles of the sine output in eight (8) bit up-down counters is included within the EPLD. An encoder Index pulse sets the counter to zero or minus one (-1) as appropriate. The counter is only required to hold the pulses accumulated during one (1) sample period since reading the counter also clears it. This reduces counter size requirement. Continuous accumulation of the angle is in the DSP. The EPLD counter size, eight (8) bits, provides ± 128 counts. For 512 reads/seconds and an encoder with 16,384 lines, counter overflow occurs with a velocity of 1440 deg./second. With these parameters, system velocity must be kept below this value. Use of an encoder with less lines per cycle, a higher sampling (or read) rate of the counter or both would increase the system velocity allowed.

A second EPLD function provides the framing, clocking and control of the TMS320C25 processor serial port in such a way as to permit high speed (5 Mhz) asynchronous serial communication between DDSC slaves and a Master Controller located within a Supervisory Computer. The serial link synchronizes all controllers to the Master so that all data taking and all control actions are synchronized with each other. The EPLD provides the basic pulse width modulation timing to the GADIC IC described above. It contains address and enable decodes for the board and an adjustable wait state generator. Two (2) motor commutation (Hall devices) detection interfaces (six (6) discrete inputs) are included.

For different applications the EPLD is programmed accordingly. To control the Robotic End Effector the incremental angle encoder interface is replaced with circuitry for deriving position and velocity information from the motor commutation signals. This requires two (2) ten (10) bit counters which for each axis measures the time since the last transition of the commutation signal or the last read of the counter. Resolution of this time counter is 3.052 micro seconds. Also, two (2) eight (8) bit transition counters count the transitions of the commutation signals during each sample period. The time between transitions, the number of transitions per second and the accumulation of transitions are used to compute within the DSP motor angular velocity and relative position. With the exception of these substitutions the end effector EPLD is like the EPLD for the rotary joint.

Other alternate EPLD functions, which can be substituted for the incremental angle encoder function, are, 1) dual resolver 512 Hz excitation signal, providing proper timing and phasing, or 2) dual absolute encoder control. These functions are described in Section 2.5.

2.2 Navtrol's DSCN System

2.2.1 A Network of Controllers

As shown on Figure 2.2.1, the Digital Servo Controller Network consists of an interconnection of one (1) to seven (7) Digital Servo-Controllers (Slaves) in turn interconnected with a Master Controller through use of a high speed (5 MHz), single differential pair, serial communications bus. The highly structured bus, proprietary to NAVTROL, is described in Section 2.6.

This report emphasizes NAVTROL's DDSC servo controllers, developed on the SBIR Phase II program. However, descriptions of other DSCN components are presented here to provide some understanding of the environment in which the DDSC's operate.

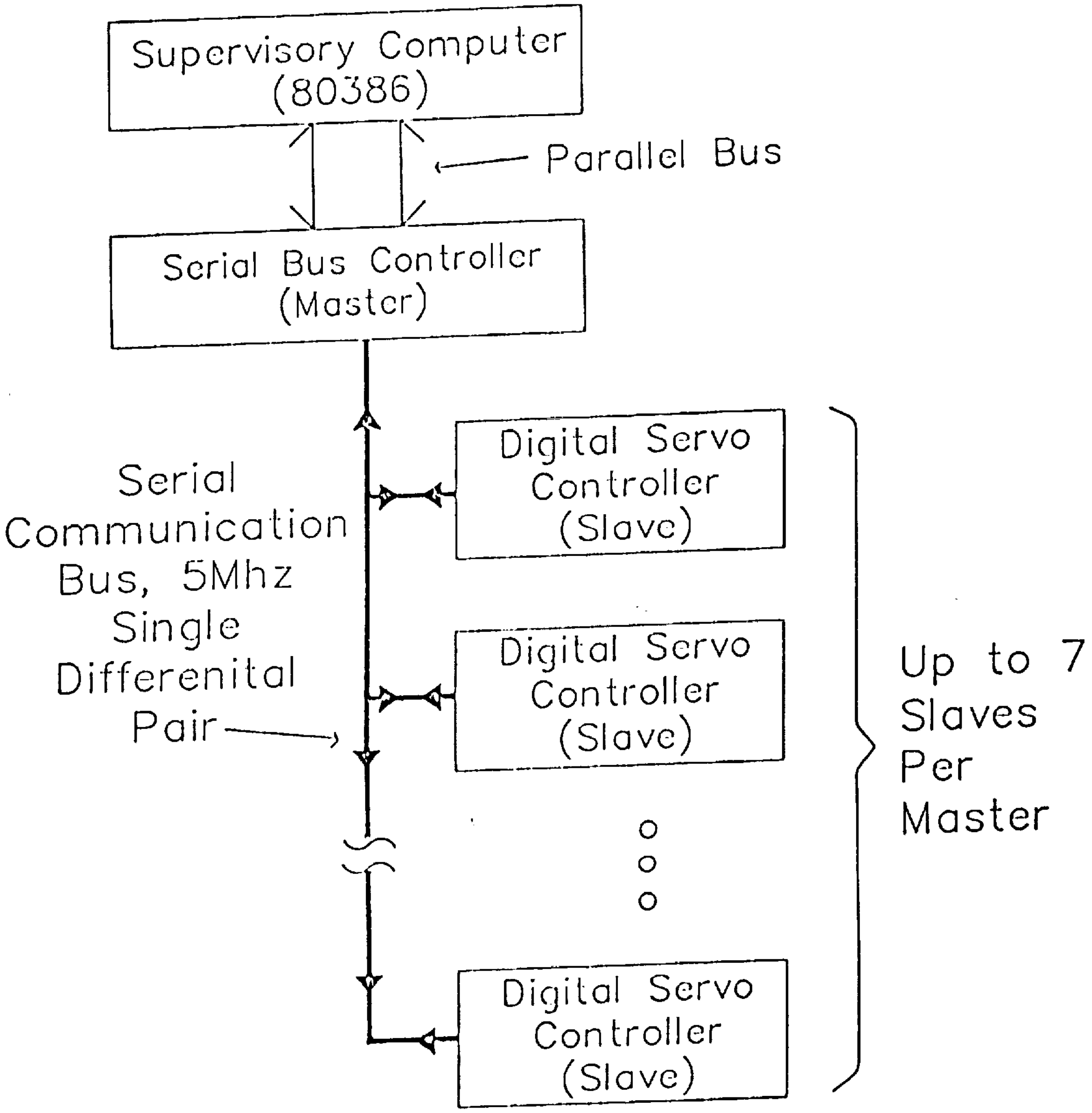
2.2.2 The Master Controller and Supervisory Computer

The Master Controller resides on the bus of an IBM PC AT 286 or 386 compatible computer. It controls communication to all DDSC slaves. Figure 2.2-2 illustrates the functions of the Master Controller board. Like the servo controller slaves, the Master Controller processor is also a TMS320C25, since it must repeat the same tasks, in controlling communication between the Servo Controllers and coordinating their activities, rapidly and repetitively. The Master Controller communicates over the AT parallel data bus to the IBM PC-AT 286 or 386 compatible Supervisory Computer.

The Supervisory Computer transforms commands from operators into information that can be communicated to and understood by the several Slave Controllers. It is a general purpose micro processor (AT286 or 386) since its tasks are diverse and it is the main interface between human operators and the Servo Controllers. For this reason it must be flexible in usage.

2.2.3 DEMON Servo Development System

Each of the seven (7) controllers can be monitored, parameters adjusted and functions controlled from the Supervisory Computer. The DEMON Servo-development software, whose capabilities are listed on Table 2.2-1, provides this capability in a user friendly environment.


Supervisory Computer (80386)
Parallel Bus
Serial Bus Controller (Master)
Serial Communication Bus, 5Mhz Single Differenital Pair
Digital Servo Controller (Slave)
Digital Servo Controller (Slave)
Digital Servo Controller (Slave)
Up to 7 Slaves Per Master

Figüre 2.2-1
DIGITAL SERVO CONTROLLER NETWORK (DSCN)

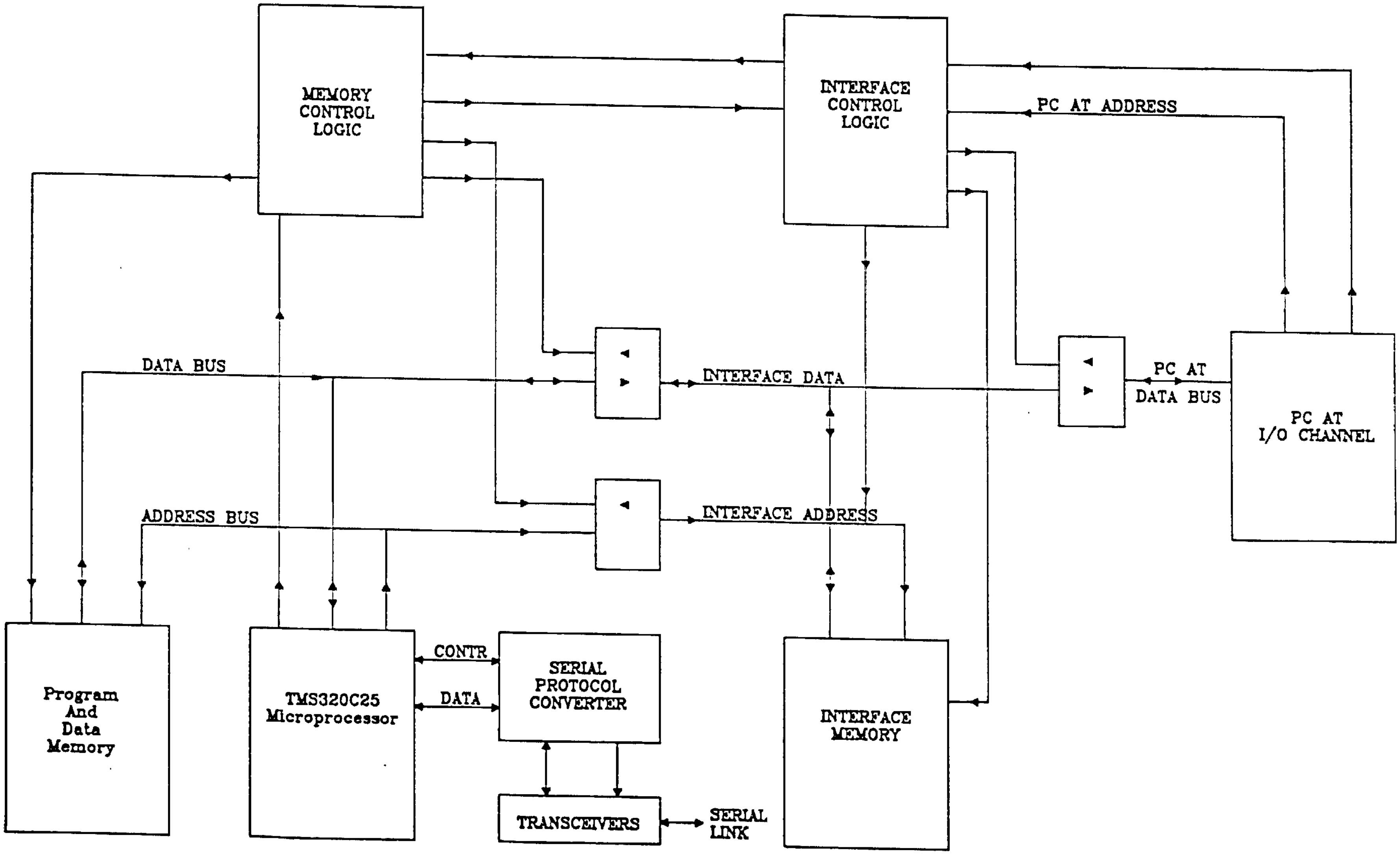


Figure 2.2-2: Functions of the Master Controller Board

TABLE 2.2-1: DEMON FEATURES

Software for System Development, Monitoring and Control

A. SYSTEM MONITORING

1. Data Collection

- Twelve (12) double precision variables can be collected sample by sample.
- "Rotary Collect" allows collection around an event whose time of occurrence is not predictable. (This capability is similar to that of a "Digital Logic Analyzer" except it is for analog or other signals within the monitored system.)
- Data collected can be saved on disk and/or compared with data taken at the same or other times.

2. Time Response Graphical Displays

- Any six (6) variables can be selected for graphs. Any three (3) variables can be displayed on one (1) screen. Other variable sets can be rapidly selected for plotting from present or previous data.
- Distinct grids permits fast assessments of parameter values.
- Zoom capability permit almost instantaneous enlargement of one (1) or all graphs about any point with respect to time, amplitude or both.
- Graph cursor provides rapid read out of data values to their full accuracy.
- Overlay capability allows two (2) graphs, even graphs taken at previous times and stored on disk, to "overlay" another graph. Full zoom and cursor capabilities also apply to the overlayed graphs.
- Graph parameters, including axis legends, graph titles and required scaling information, are pre-stored for all program variables. These are called automatically from other Tables and Files within DEMON on basis of the mnemonic for the variable or through its address.

3. X-Y Displays

- Provides for display of the interaction of any two (2) variables in the monitored system, in either real time or from previously collected data.
- A movable Pointer provides rapid read out of any graph location.
- Data selection, position of X and Y axes and scaling are all programmable.
- Zoom provides magnification about any curser position.
- "Trace" capability provides memory of the path taken for the X-Y variables.

B. SYSTEM ADJUSTING

- System gains, limits and other constants can be entered, monitored and adjusted from configurable tables.
- By use of interactive capability between tables, system gains can be calculated from various system parameters, such as sampling interval, time constants, etc., even when the relationship is complex.

C. SYSTEM CONTROL

- Load selected programs into controllers and save programs and data on disks.
- Provide Mode or Function Control of the System.
- Search or Scan Routines can be defined and Controlled from **DEMON.**

D. SYSTEM TESTING

- Diagnostic Tests on DSP Processor and/or Controller Memories.
- Simulated System to bypass hardware for testing and isolating problems.

2.3 System Timing

As discussed in Section 2.2, Navtrol's Digital Servo Controller Network (DSCN) consists of an interconnection of 1 to 7 digital servo controllers (DDSC) slaves, in turn interconnected with the Master Controller. At present the system is set up such that the master communicates with each Slave 512 times per second. This same rate represents the basic control loop sampling frequency utilized within the DDSC's. Control loops with higher or lower multiples of this control frequency can be and are utilized within the DDSC. For example, the pulse width modulated motor current control loop operates at 10,960 samples/second, 20 times the basic control frequency. However, 512/second is the frequency of communication between each Slave and the Master Controller.

Figure 2.3-1 illustrates the timing utilized within each of the DDSC Slaves. In this figure the top line shows time in milliseconds, relative to the start of serial communications to a particular slave. The next line illustrates the time in which the serial communication occurs for that Slave. The control loop is synchronized to the serial communication. The pulse width interrupts for both loops, are timed relative to the serial communication for each Slave in a DSCN network. This synchronization of the pulse width interrupts occurs within each Slave's firmware contained within it's DSP's EPROM.

The sampling frequency for the pulse width loops is 10,960. per second. The actual pulse frequency for these loops is 4 times higher or 40,960 pulses per second. The pulse width interrupt times, 22.8 micro seconds for loop 0 and 23.8 micro seconds for loop 1, are sufficient to perform the control computations for the pulse width loops and, in addition, process additional A/D samples, as described in the following paragraphs.

It is not recommended that the User adjust or modify the communication or pulse width timing or algorithms. This is because of the interplay and critical timing between these two functions. Setting aside these two functions and the time that each requires, leaves approximately 840 micro seconds for applications oriented software programs. The time for this, as indicated on Figure 2.3-1, occurs between the end of the communication program and the start of routines in which data is collected and placed in buffers for use during serial communications. For example, words requested to be transferred back to the master are retrieved and stored in an array. There is also time remaining after the buffer routines for execution of a few instructions. However, in no case can the user's program extend past 1.953 ms, as this is the time that the serial communication with a particular Slave commences again and the processor must be ready to accept and recognize the address words from the master.

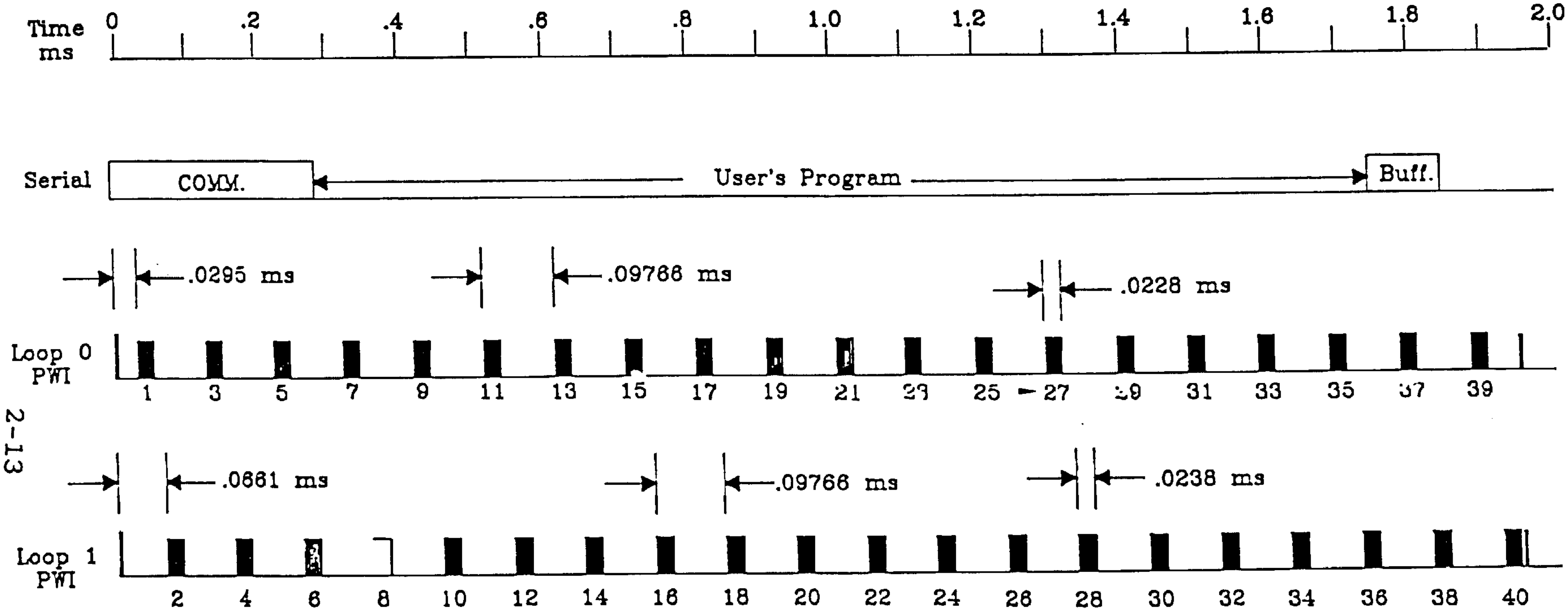


Figure 2.3–1: Control Loop Timing

If the PWM outputs are used, the user's program must accommodate them, either through carefully timed code or through interrupts occurring at the intervals indicated. The code for servicing the interrupts and controlling the motor current is within the DSP's EPROM and is automatically accessed by the interrupts. However, the interrupts can be disabled as described in Section 3.0.

In Figure 2.3-2 the control loop timing previously shown is expanded to show more timing detail during communication and shortly thereafter. First, the serial communication is broken down into its Read (S. to M.)/Write (M. to S.) segments. Functions of each of the segments are discussed in Section 2.6. Shown on Figure 2.3-2, but not shown on the previous diagram, are motor drive pulses for the two loops. These pulses occur at 4 times the rate of the pulse width interrupts, also shown on this expanded timing diagram. Note that the motor current pulses for one loop starts midway between the start of the pulses for the other loop. This reduces power supply current transients, since only one loop turns on at a time. In addition, pulse starts for DDSC slaves in a network are staggered with respect to other slaves so as not to occur at the same time. Again, all timing for all slaves is synchronized so that timing of events is repeatable and known. Note that pulse widths can vary from 0 to 100%.

Also illustrated on this diagram are the A/D samples obtained from GADIC0 and GADIC1, processed during the pulse width interrupts by the PWI software routines. The times shown for these samples are the times that the samples are actually taken, not the times when the A/D is read. The first sample of the two close side by side samples for each A/D is the measurement of motor current. The A/D for GADIC0 reads motor current for Loop 0 and GADIC1 reads it for Loop 1. The second sample, and another sample occurring simultaneously on the other GADIC, can be used for a multitude of applications. For a system utilizing an encoder, or encoders, the sine and cosine outputs from the incremental encoder, or encoders, are read simultaneously and used to determine the encoder angle as described in Section 2.5.1. Samples for the loops alternate, so that the first simultaneous set of samples is for Loop 0 while the next simultaneous set is used for Loop 1. In another application all the samples of GADIC0, except for motor current measurements, are used to perform demodulation functions for two resolvers, alternating between loops, as described in Section 2.5.2. The corresponding measurements on the other GADIC are used for strain gauge measurements in turn used for a torque loop. The torque loop will be sampled at 1024 samples per seconds or twice the basic control frequency. Navtrol is presently expanding the number of A/D samples taken during the communication cycle in order to, increase time utilization during the communication period.

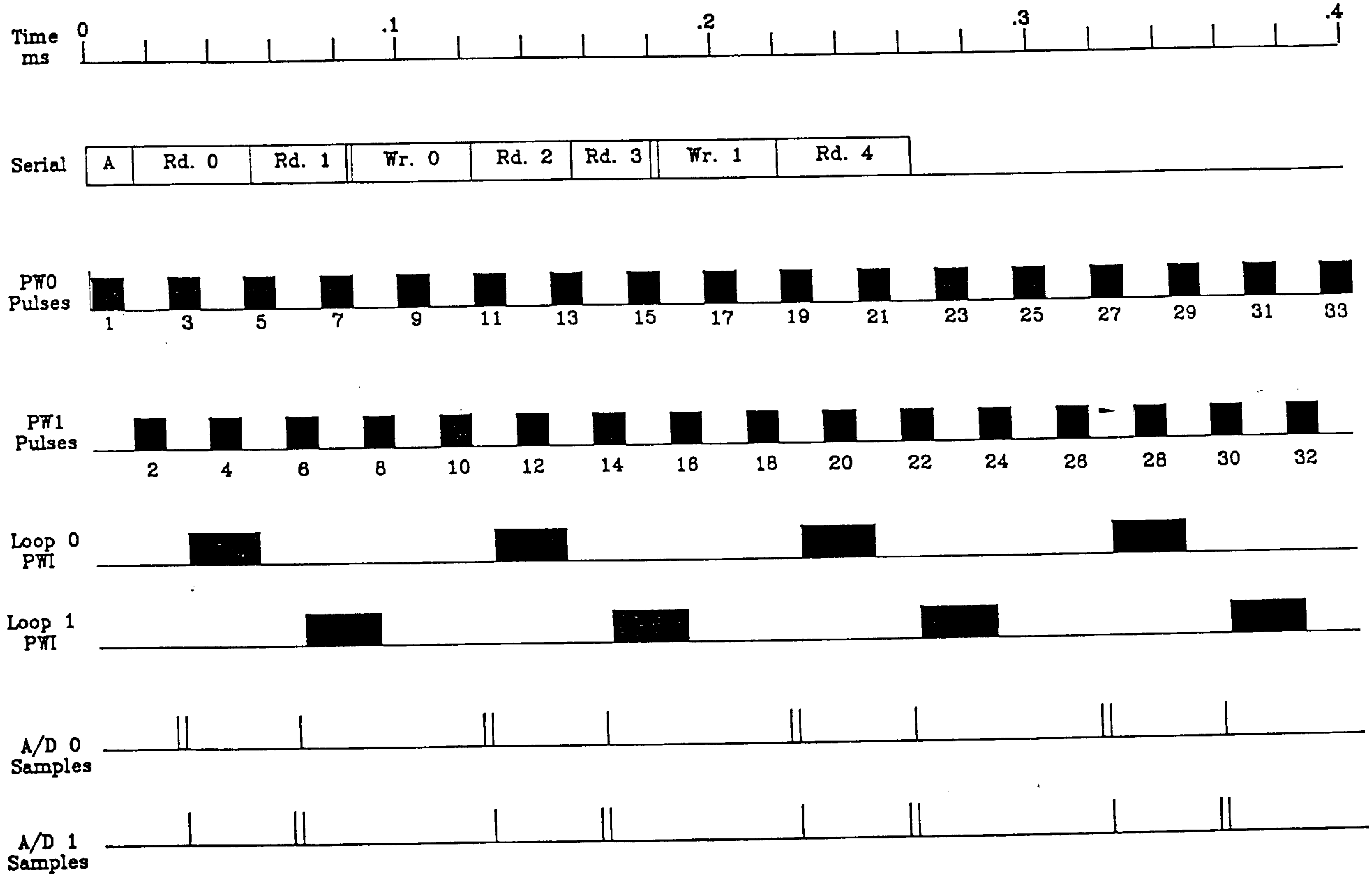


Figure 2.3–2: Expanded Control Loop Timing

2.4 Output Functions

2.4.1 PWM Motor Control

2.4.1.1 Overall Functional Description

Figure 2.4.1-1 is a functional diagram of one (1) of two (2) "Commutating Pulse Width Modulated (PWM) Power Amplifiers" utilized in the DDSC to control current for DC motors, either brush type or brushless. As indicated there, the components required for this function are the TMS320E25 Digital Signal Processor, the Digital Interface Gate Array, the General Analog/Digital Interface Circuit (GADIC), and Power Stages. Only the motor current measuring components in the power stages and the input to the GADIC are analog. Other functions, usually accomplished in analog circuitry in conventional pulse width modulated amplifiers, are accomplished digitally. Characteristics of the PWM amplifier are as follows:

1) PWM frequency = 40,960 Hz
2) PWM update Rate = 10,240 s/s
3) Current Rise Time = 0.1 ms (Linear Region)
4) Current delay = 0 to 0.1 ms
5) 8 A @ 50V (90V, optional) capability

The precisely controlled time of occurrence of the motor current pulses and motor current measurements with respect to the start of the control sampling period is illustrated on timing diagrams in Section 2.3.

Based on system torque requirements, a commanded motor current is defined in the TMS320E25. A measurement of motor current is obtained by passing a voltage proportional to motor current into a "three (3) stage flash" analog to digital converter, contained in the GADIC. The digital signal from the converter is then fed into the TMS320E25 Digital Signal Processor. Here, measured motor current is compared with commanded current and, based on motor voltage, motor resistance, motor inductance and other circuit parameters, the TMS320E25 calculates the desired pulse width. This timing command is then passed to the GADIC. The GADIC in turn controls the "on time" of the power transistors of the power stage, thereby controlling the current through the motor. Current command delay time can be as much as one sample period or 0.1 ms. Current Rise Time depends on applied voltage and motor inductance but can be as short as 1 sample period. In the linear region, rise time is 0.1 ms.

The Digital Gate Array on the DDSC contains the basic timing circuitry which provides a carefully synchronized 163,840 Hz signal, which in turn controls the sampling interval. Division by four (4) in the GADIC provides the pulse width modulation frequency

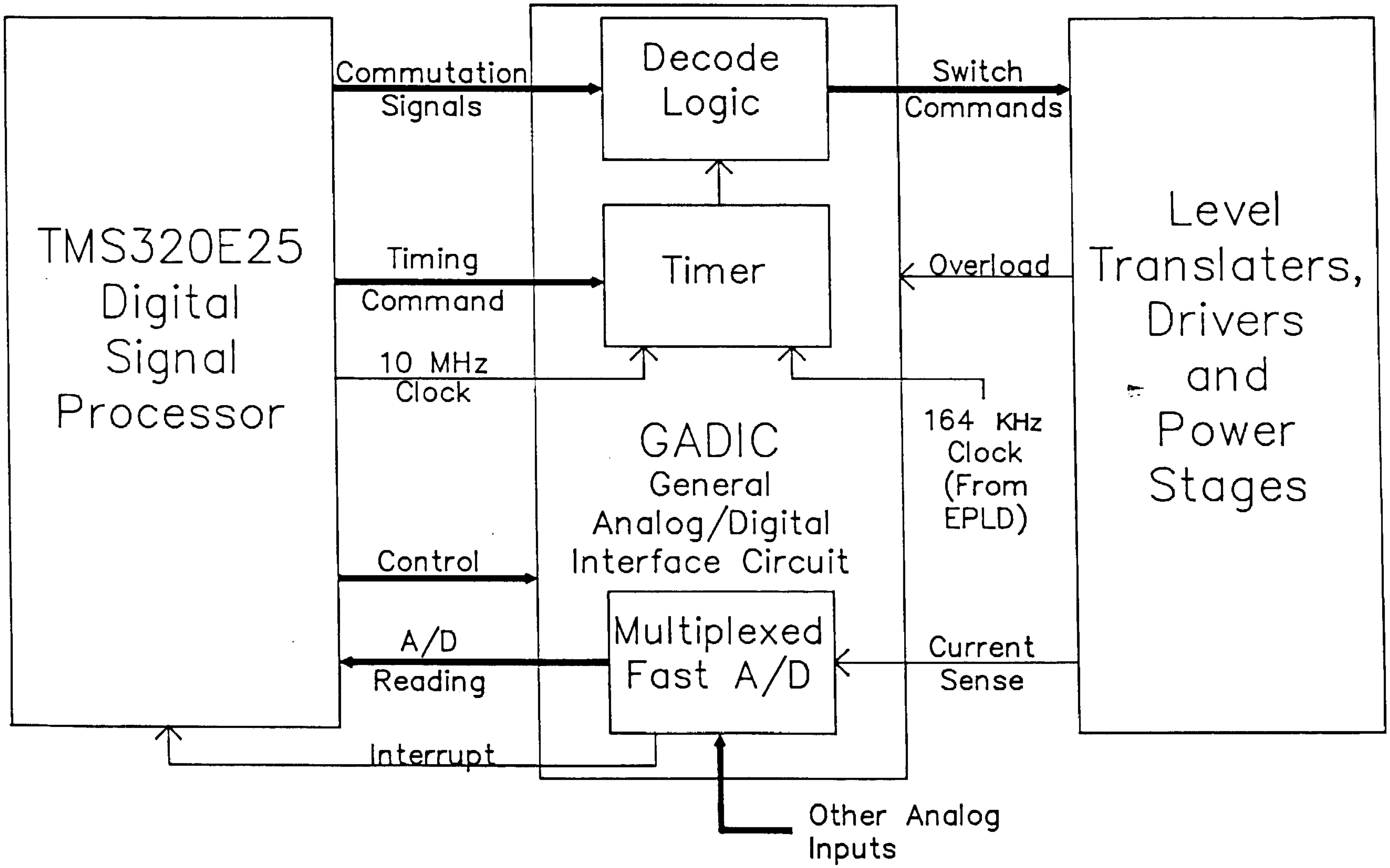


Figure 2.4.1—1: Commutated PWM Power Functions

of 40,960 pulses per second. Provisions are provided for synchronizing through the processor both the EPLD pulses and the GADIC PWM outputs with the serial communication timing. The GADIC also provides decoding logic for motor commutation signals received from the TMS320E25 Digital Signal Processor. The DSP derives the motor commutation selections from motor mounted Hall devices or angular measurements made by encoders or other devices not shown. The GADIC sends six (6) switch PWM commands to the power stage to control the three (3) upper and three (3) lower power switches.

If an overload in motor current is sensed, a discrete signal is sent to the GADIC, which then drops all switch commands to zero (0), both upper and lower. A discrete is also sent to the DSP. The DSP can then shut power down or re-command a current. Although not shown, the GADIC also contains circuitry to shut down the power stage if no new command is received from the TMS320E25 within four (4) pulse periods. This prevents driving the motors erroneously in the event of failure of the DSP or certain portions of the GADIC. The Processor checks the A/D circuitry and power stage for proper operation and can shut down the power stage either by setting the timing interval to zero (0) or, through the commutation signals to the GADIC, by selecting no switches to be turned on. Turning off all switches provides maximum braking to the motor.

2.4.1.2 Motor Driver Amplifier Power Stage

Figure 2.4.1-2 illustrates the PWM motor amp power stage. Figure 2.4.1-3 shows the electronic components utilized in the upper and lower switches and their respective drivers. Referring to Figure 2.4.1-2, the six (6) outputs from the GADIC are fed to six (6) individual drivers. The three (3) lower drivers are contained in two (2) very small surface mount IC's, the TSC4427, a dual driver IC manufactured by Teledyne. The three (3) upper stages make use of a similar driver IC, except it is inverting and in turn is driven by a NOR gate. A level shifter is required since the upper switches operate at the motor voltage, up to 50 VDC. (Operation up to 90 VDC is possible with changes in the components of the power stages.). The level shifter consists of a simple current amplifier which turns the high impedance CMOS drivers on with little utilization of power. Each of the drivers, both upper and lower, are connected to the gates of the power MOSFET transistors which they control. Not shown on the diagram is an isolated 12V power supply, which provides -12V with respect to the motor voltage (28 volts or ?) for control of the upper gates. The lower drivers are tied to 12 VDC with respect to Power ground.

The power bridge operates in the following manner. When current is required through a particular coil, a selected set of upper and lower switches is turned on. This directs the current

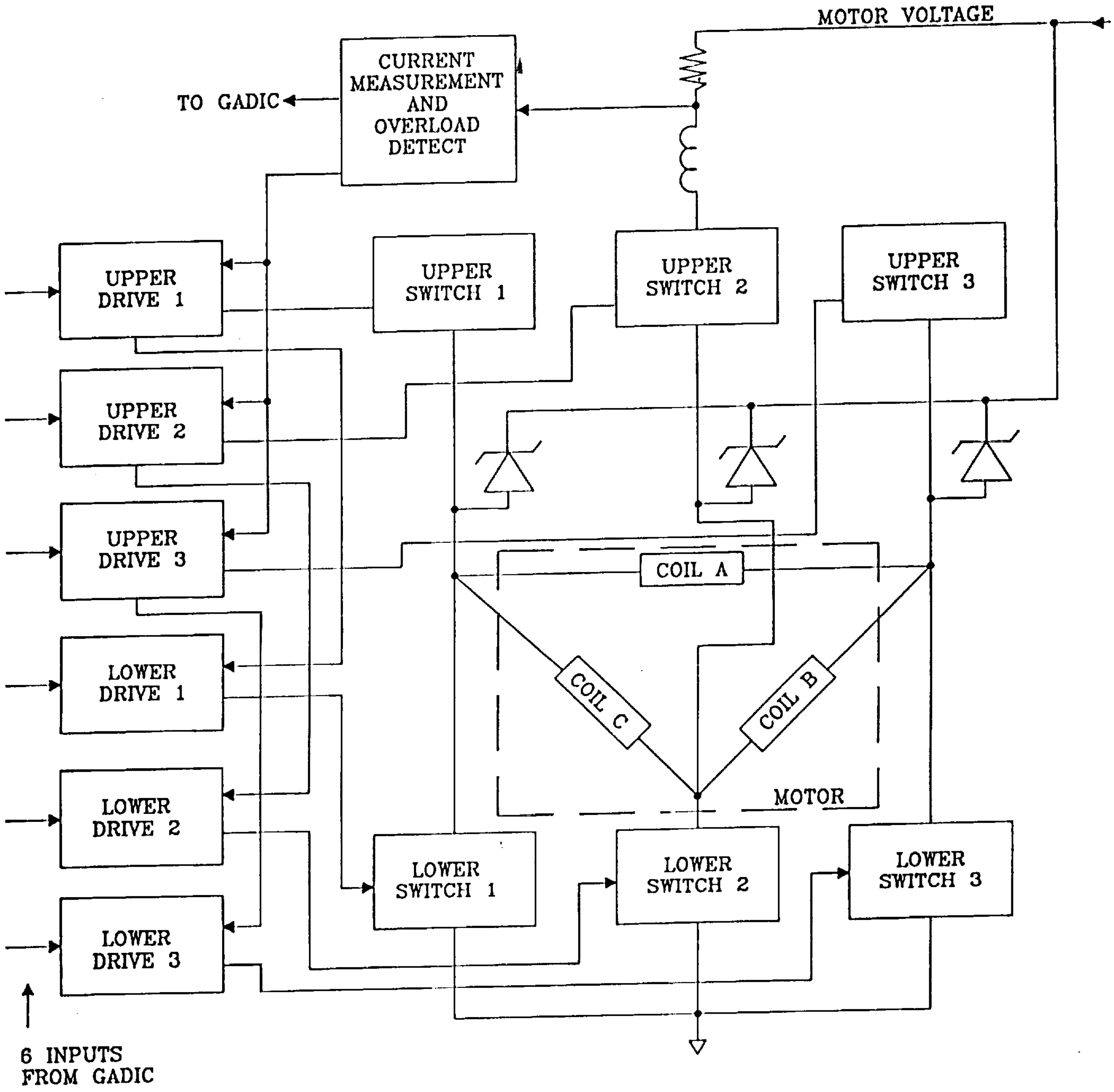


Figure 2.4.1-2: PWM POWER AMP POWER STAGES

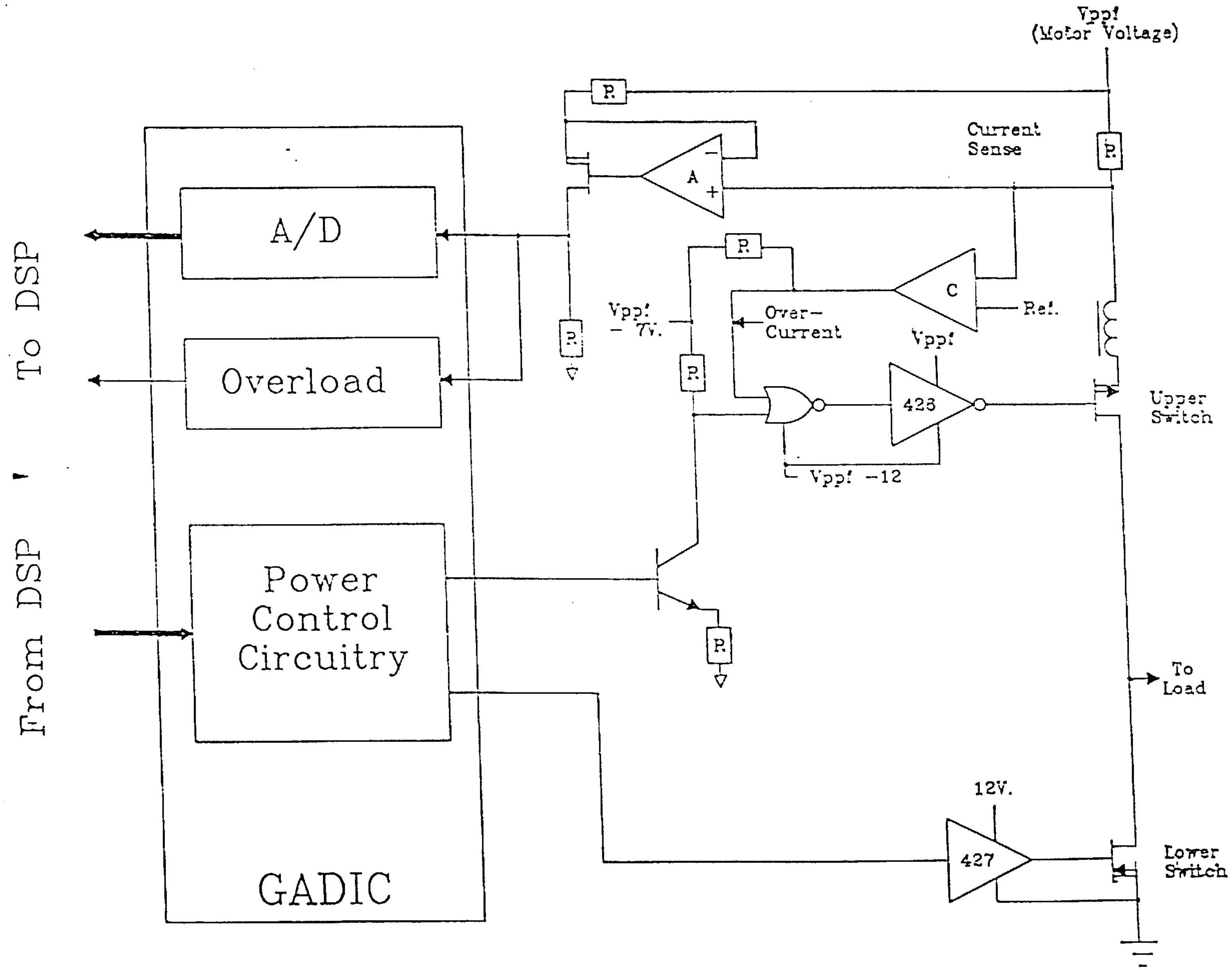


Figure 2.4.1−3: 1 of 3 Sets of Power Stages
(2 of 6 Switches)

in a particular direction through the motor coils in such a way as to maximize the amount of motor torque per unit of current. When the current reaches a desired level, as predicted from the motor coil parameters, the lower switch is turned off but the upper switch is left on. Because of coil inductance the current continues, completing its path through the Schottky diode directly above the previously on, lower transistor. Note that because the Schottky diode is connected to the motor voltage, the current continues to flow through the sense resistor. Because of coil resistance and other voltage drops within the loop, coil current will decrease in the off period. At the start of the, next pulse period the lower switch will again turn on for a period of time calculated to bring the average current to the desired value.

If the current is too high at the start of a pulse period both the upper and lower switches can be turned off, forcing the coil current to conduct from ground to the motor voltage, decreasing the current rapidly. Note that a Power Mosfet transistor has a diode in the back drive direction, allowing completion of the current path. During the time when both upper and lower switches are open, no measurement of current is made since the conduction path bypasses the sense resistor. This "full off" condition is timed the same as a "full on" or pulsed condition. Part way through the PW cycle, the upper switches are turned on and current measurements can be taken.

At the top of the power stage is a combined current and overload sensing resistor which generates a voltage proportional to current through the overall power bridge. The voltage across the sense resistor is amplified by a precision op amp, level translated by use of a small MOSFET transistor circuit and applied to the input of the GADIC for A/D conversion. When the current exceeds a preset value, the overload detecting circuit inhibits all three (3) upper drivers, turning them off immediately. It also sends a signal to the GADIC which in turn commands zero (0) current until the condition is corrected. A discrete is also sent to alert the DSP of this occurrence.

Logic in the GADIC prevents accidental turn on of a lower switch located directly below an activated upper switch. However, if through some failure the upper and lower transistor do turn on at the same time, the overload detection will shut the power stage down. Although failure detection is important, the primary motivation is to prevent failures of any kind from occurring, even if a motor lead is accidentally shorted to ground.

Although MOSFET power transistors are able to conduct in the back direction and act as diodes, the DDSC includes Schottky diodes for the upper transistors to prevent this mode of operation. This is because the internal diodes, especially in P-channel

MOSFET transistors, are slow recovering diodes. When a lower switch is turned off to modulate current amplitude, the inductive reactance of the load coils requires that current flow continue. If no external diode path with a lower impedance was provided,

then the current path would be through the internal diodes of the upper MOSFETS, located above the lower transistor which was previously on. When the lower transistor is again turned on, the slow recovery of the upper transistor diode results in a high surge of current which results in wasted power, EMI and, depending on how hard the transistors are being utilized, decreased reliability. By use of Schottky diodes, as shown on Figure 2.4.1-2, this back conduction can be prevented and a better behaved, better performing and more reliable circuit results.

Note that the voltage across resistor R_s flowing in the positive direction tends to bias the diode within the upper switch MOS transistors to an "on" state. For this reason, as well as to conserve power, the resistor R_s is selected so that only a 100 to 200 millivolts maximum occurs across it. With these low voltage levels, the lower voltage drop of the Schottky diodes will prevent back conduction in the upper power switches.

The diodes in the upper portion of the bridge are connected to the motor voltage to provide a path for the current, such that when the upper switch is on, but the lower switch is cut off, current will still flow through the current measuring resistor R_s. This is necessary for adequate current control at low current levels since the current must be measured during the time that the upper switch is off. This is because the short "on" pulse duration may not permit clean measurements during the on time.

Although the pulse rate is 40,960 pulse per second, pulse width time is commanded every fourth pulse or 10,960 times per second. Pulse width time is controlled by a 9.994 MHz clock providing 244 clocks per pulse. This would provide approximately eight (8) bits of resolution. However, by staggering the pulse widths over the four (4) pulse update interval, the equivalent of ten (10) bit resolution is obtained. Measurement of current through the sense resistor is also made only once every fourth pulse or 10,960 times per second. The measurement is made just before the next pulse is started, a region in which switching transients have had time to settle and rapid variations do not occur. This carefully timed sampling of the current bypasses problems due to transients which occur whenever the large MOSFET transistors are switched on and off. The new PW time based on the measurement is commanded before the next pulse following the pulse that occurs immediately following the measurement. The measurement is extrapolated to when the new commanded time takes effect to eliminate the effect of calculation delay.

The same power amplifier design can be used for a wide variety of applications, with only changes in the voltage and current capability of the power transistors, the Schottky diodes and other power stage components.

2.4.2 Other Output Functions

2.4.2.1 Power Discrete Outputs

Four open drain MOSFET outputs are provided for control of relays, actuators, or etc.. Each is capable of 0.5 amp at up to 50 volts. Inductive loads should be bypassed by diodes to prevent the inductive "kick" from damaging the open drain transistors.

2.4.2.2 D/A Outputs

Two twelve bit D/A outputs are provided at pins 11 and 24 of the 25 pin power connector. These D/A's are written to from the on board DSP. They can be used for monitoring on-board functions or for control of external devices.

2.4.3 Output DC Power Provisions

Commutation power (12 V.D.C., 50 ma) for Hall device excitation or other uses is provided on the 25 pin power/motor connector. Plus and minus 5 volts is provided on each of the signal connectors for use by external devices. Current on + 5V. should be limited to 250 ma and on - 5V. to 100 ma.

2.5 Input Functions

2.5.1 Incremental Encoder I/F

Navtrol's design for the interface circuitry between the DSP processor and an incremental angle encoder is illustrated in Figure 2.5.1-1. Here the sine and cosine tracks from the encoder, as well as the index, which were routed onto the PCB through the sine and cosine Special Inputs, utilizing the Special Inputs Return, are fed through compare circuitry and, in the EPLD, through appropriate logic into an up/down counter. Not shown are the input filters and the sine signal offset adjustment circuitry. In the EPLD, only the crossings of the sine wave when the cosine wave is high are counted. However in addition, two bits indicating the guadrant of the encoder cycles are included when the cycle count is read. This is used to increase the resolution of angle indicated by the count. Incremental encoders with in-phase and quadrature square wave outputs can also be properly interfaced by feeding these signals into the Sine and Cosine "Special Inputs." Digitally, these are treated exactly like the linear sine and cosine inputs.

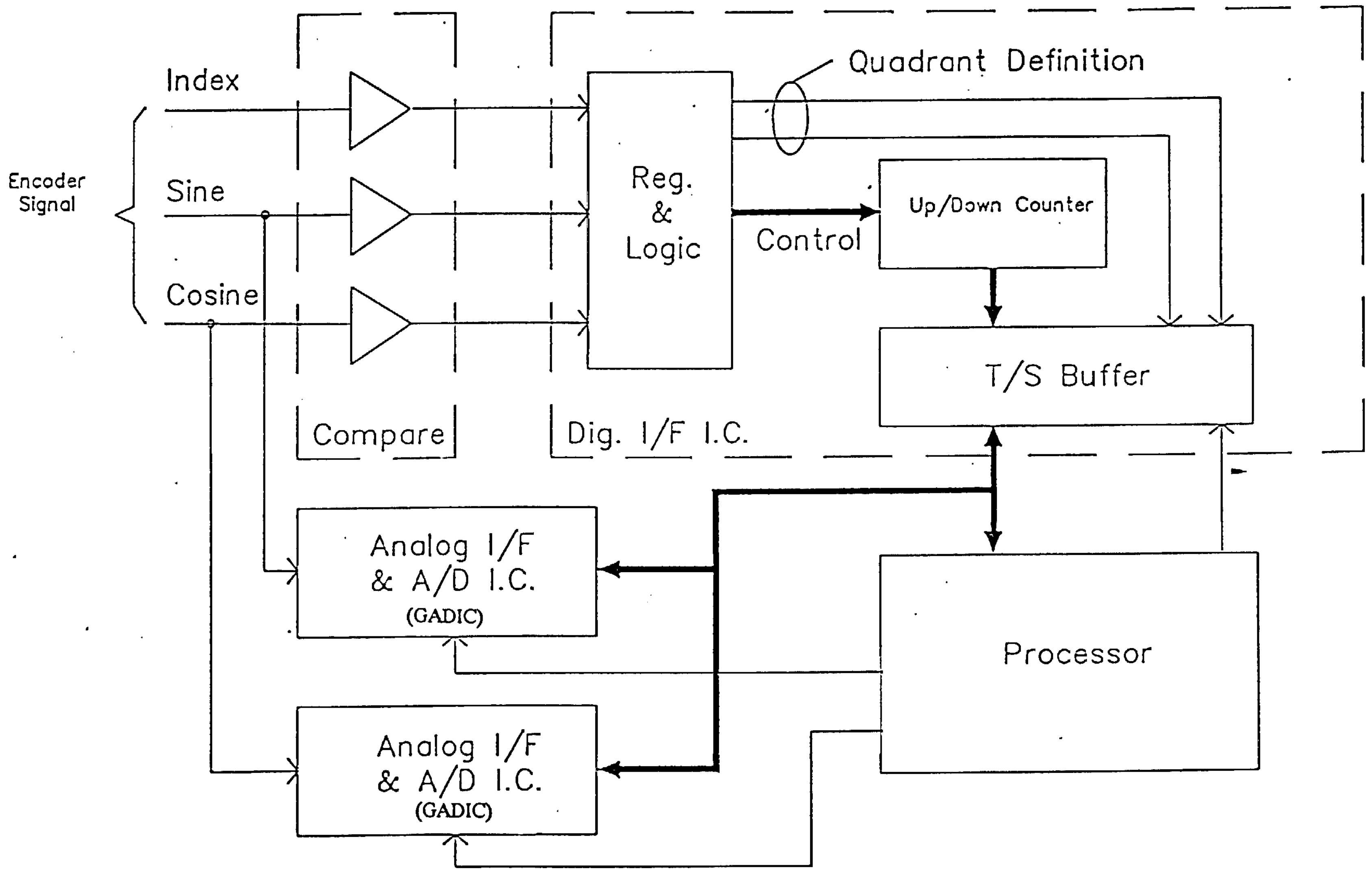
Using linear sine and cosine signals permits use of an approach in the DDSC to greatly increase the angular resolution. This approach is to convert these encoder sine and cosine signals into digital and feed them into the DSP. These are then converted into an angle and utilized to interpolate between the bits obtained by the up/down counter. Appropriate combination of both measurements, those from the EPLD up/down counter and the A/D converter, can provide a very high resolution measurement of the encoder angle. For an encoder with 8192 lines more than 26 bits of angular resolution is achieved. This resolution is fully utilized in the DDSC control algorithms to define velocity as well as angle, even when the motion is very slow.

2.5.2 Resolver I/F

2.5.2.1 Functional Description

The DDSC provides the capability for interfacing with 2 resolvers. Both excitation and detection functions are included. Figure 2.5.2-1 illustrates the resolver excitation functions. Within the onboard EPLD a counter circuit provides division by 1952 of the 10 MHz signal resulting in a 5120 Hz TTL signal. This signal is used to drive an electronic switch which in turn provides an output square wave, operating between +5 and -5 volts DC. This square wave is fed through a tuned second order filter, resulting in a sine wave excitation signal which is 14 volts peak to peak and can supply 20 milli amps or more. This should be sufficient to easily drive 2 resolvers at this frequency.

Within the EPLD the counter automatically reloads itself to perform the proper division resulting in the 5120 Hz signal.



Encoder I/F Circuitry

FIGURE 2.5.1–1

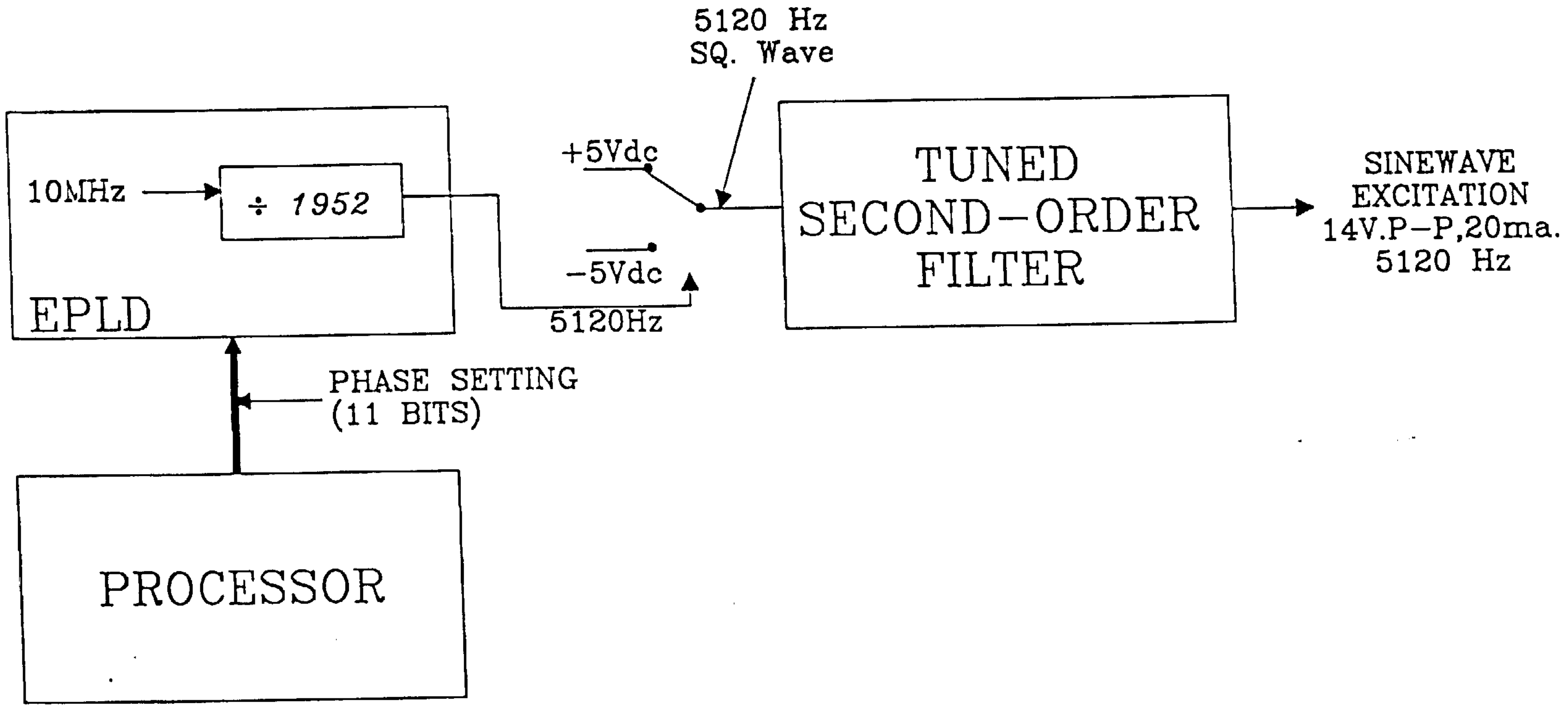


Figure 2.5.2−1: Resolver Excitation Functions

However, the counter can be loaded to a selected count from the processor, providing the capability for synchronizing the sine wave with other occurrences within the control loop. Specifically, the excitation can be synchronized such that the sample points for the received sine and cosine signals occur at the appropriate places. This is discussed further in paragraphs which follow.

The "Receive" portion of the resolver interface is illustrated on Figure 2.5.2-2. The sine-cosine signals from the two resolvers are fed into a multiplexer which selects signals either from one resolver or the other. The signals are then multiplied by ± 1 and fed into multiplying D/A converters. The SIGN amps and the MDAC's combine to provide multiplication such that sine A is multiplied by - (EA) and cosine A is multiplied by sine (EA). These two signals are then summed together resulting in the following relationships:

$$\mathrm{Sin}\,(EA) * \mathrm{Cos}\,A - \mathrm{Cos}\,(EA) * \mathrm{Sin}\,A = \mathrm{Sin}\,(EA - A)$$

$$\text{if} \quad EA \approx A, \text{ then } \mathrm{Sin}\,(EA - A) \approx (EA - A)$$

These equations can be easily verified using a handbook showing trigonometric identities. The term (EA) stands for estimated A and will converge to A utilizing estimation algorithms within the processor. The term (EA-A) represents the error signal for the estimator. This signal is then multiplied by 2 and fed into the GADIC. There it is multiplied by an additional gain factor of 32, fed through an A/D converter and from there into the processor.

Resolution of this conversion is 18.61 bits on the angle. This is calculated from the fact that 7 volts, assuming the resolver transformation ratio is equal to 1, represents an angle whose sine is 1. The LSB of the A/D is 7.03 milli volts. Angular resolution is therefore:

$$\mathrm{Sin}^{-1}\,[(.00703)\;/\;(7\ \mathrm{Volts} * 2 * 35)] = 9.00 \times 10^{-4}\ \mathrm{degrees}$$

Dividing this angle by 360 degrees indicates a rotational resolution of 18.61 bits. At this resolution the resolver interface may be non-monotonic. This is because the 12 bit D/A's are only accurate to 1/2 LSB, which represents an angle of approximately .0070 degrees. This computes to an angular resolution of 15.65 bits. Below this angular value, the combination of DAC settings and the GADIC A/D reading could result in different measured angles for the same angular inputs. However, if the DAC settings are held constant very small angular changes can be measured by the GADIC A/D very accurately. By use of a moving deadband between the estimated angle (EA) and a commanded angle used to compute the sine and cosine sent to the two DAC's, the additional resolution provided by the A/D can be used to provide smoother angular changes, more accurate velocity estimat-

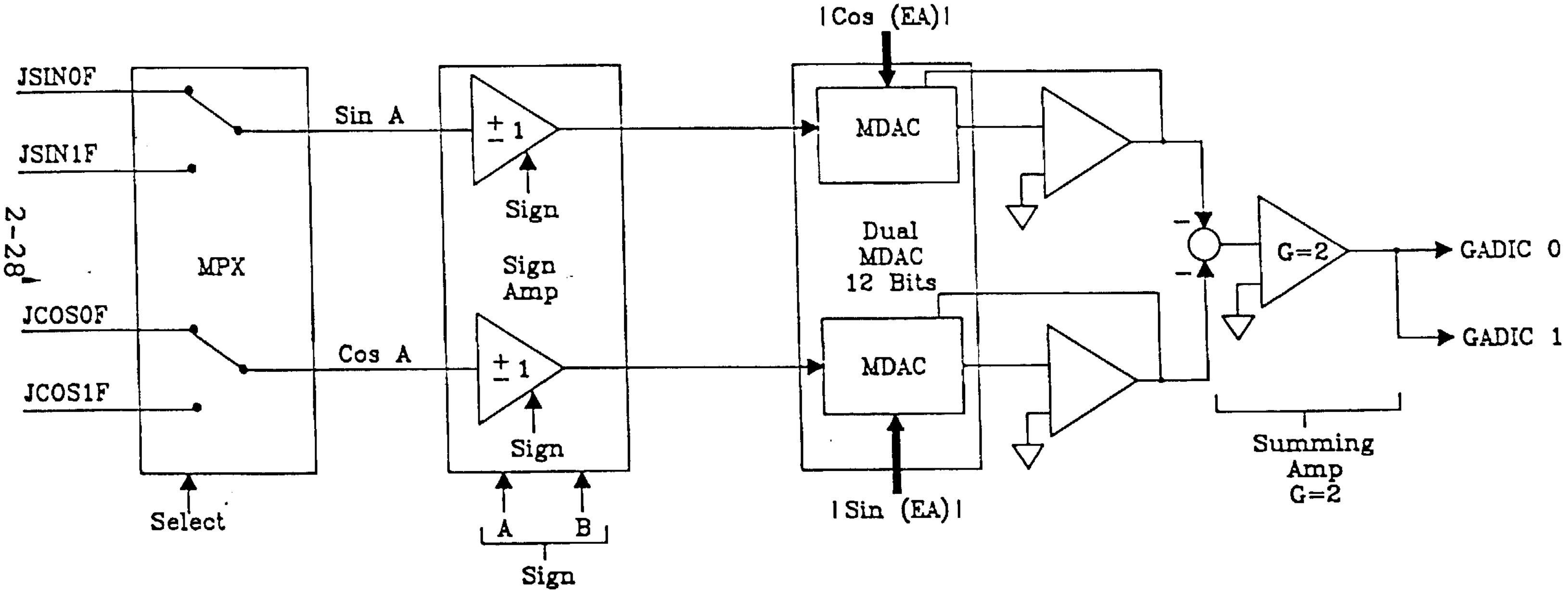


Figure 2.5.2-2: Resolver I/F Functional Diagram

ion, and better stability about small angles. No additional error is introduced if the deadband is kept small.

2.5.2.2 Demodulator Sampling Timing

Demodulation of the received resolver signal is accomplished digitally-not by the usual analog demodulation circuit. Figure 2.5.2-3 illustrates the demodulation scheme for 2 resolver "error angle" signals. As described in the previous section, the error signal from the summing amp is multiplied by a gain of 32 within the GADIC and then converted to digital with 9 bits plus sine resolution.

The loop selection multiplexer shown in Figure 2.5.2-2 selects signals from one loop or the other on command from the processor. In Figure 2.5.2-3 the first four samples are on the error signal from Loop 0 while the next 4 samples are on the error signal from Loop 1. The pattern then repeats itself such that the next four samples are again on Loop 0. A demodulated error signal results from combining the sampled signal in the matter defined by the relationship:

$$[(S01 + S02) - (S03 + S04)]$$

Is easy to see that a quadrature component of the signal (90 degree out of phase) is totally rejected as are all DC terms including bias offsets in the DAC circuitry and GADICs. For a particular loop 4 samples are accumulated for every other cycle, providing 20 samples total over the control sampling period, 1/512. These are then averaged providing the appropriate rejection of both higher and lower frequency interfering signals. The signals for Loop 1 are treated in an analogous way to likewise produce a very clean demodulated error signal. Note that each loop has 5 full cycles averaged over each control sample period.

One of the contributors to error signals within resolvers is the third harmonic which is produced by non linearities in the resolver "rotating transformer". The third harmonic is not removed by usual analog demodulation, although it is attenuated. This is easy to see if the received signal is imagined to be switched each 180 deg. and then filtered. In each half cycle an extra half cycle of the third harmonic is left over and affects the null detection in a resolver circuit. The third harmonic component is not large in the first place, and can be filtered before the demodulator.

In the approach used in the DDSC, the sampling timing is such as to greatly reduce the effect of third harmonics. This is illustrated in Figure 2.5.2-3 which shows the "in phase" third harmonic of the fundamental frequency. The sampling times could have been selected whereby the third harmonic was totally eliminated but this conflicts with other timing considerations for the system and since the third harmonic is already a small compon-

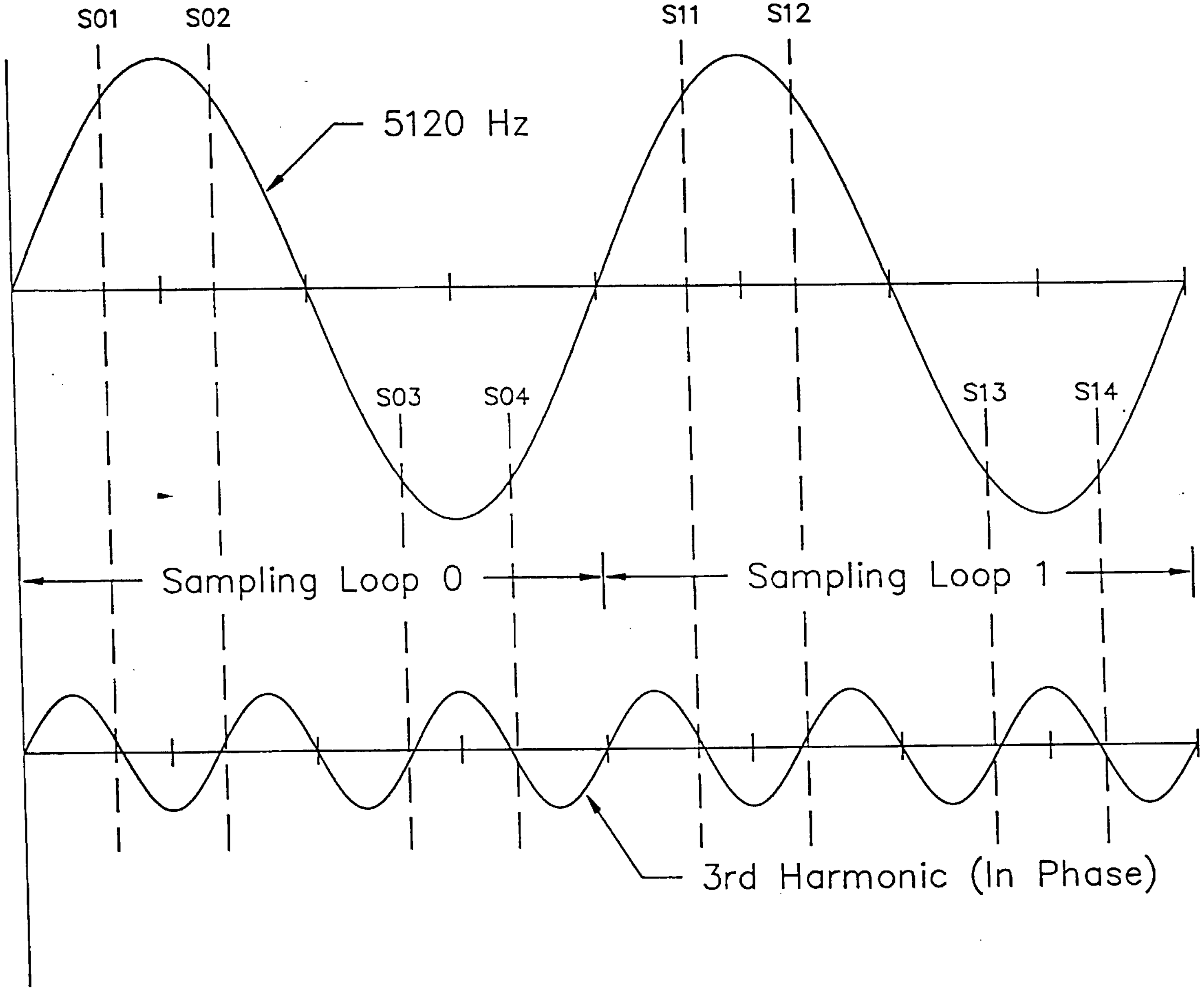


Figure 2.5.2-3 Demodulator Sampling Timing

ent, the attenuation provided by the sampling shown should be sufficient. A small amount of analog filtering is also provided on the board which reduces high frequency interference as well as the third harmonic. Results of tests thus far show the approach taken to be more than adequate.

2.5.3 Discrete/Absolute Encoder Inputs and Control

Sixteen discrete inputs are provided for data from absolute encoders or other devices. These sixteen inputs are tri-stated onto the data bus for use by the DSP. Two control lines to external devices (outputs) are provided, one for selection of one of two absolute encoders and one for "read enable" of the encoder. These two outputs are from registers within the EPLD which are set to the desired value from the DSP. If absolute encoders are not used these two controls can be used to access other sources of digital data for other "discrete output" uses as required.

2.5.4 High Gain Differential Inputs

Four high gain differential amplifiers are provided for use in amplifying strain gauges or other low level external signals. The impedance between the differential inputs is 20K, with 1 meg impedance from each side to ground. The gain of each of these four amplifiers is 100. The output from the amplifiers are fed into the GADIC for A/D conversion. Additional gain can be applied within the GADIC to the signals from these amplifiers. The positive inputs to the differential amps are also fed into the GADIC for A/D conversion. These define the common mode voltage of the high gain signal and can be used for diagnostic purposes.

2.5.5 Additional A/D Inputs

The pinouts for the two input connectors are defined in Section 3. On both these input connectors there are 10 analog inputs, including the two special inputs, sine and cosine, and the remote sense. The remote sense is intended to bring a referencing signal, such as signal ground, from some remote device. The special inputs are intended to be utilized for measurement of encoder or resolver sine and cosine respectively. Subtracting the four high gain differential inputs, discussed in the previous section, leaves four analog inputs on each connector for other uses. One of these inputs is used for identification of the board. This leaves seven additional analog inputs which are fed into the GADIC for gain adjustment and A/D conversion. These can be utilized for user purposes. The range of the A/D conversion is + or - 3.6 volts so that the inputs must be scaled accordingly. Gains of 1 to 128 can be provided within the GADIC.

2.6 Serial I/F to DDSC Slaves

2.6.1 Introduction

After researching the serial communication formats available today and not finding a suitable format to meet the needs of a multi-axis control system, Navtrol has developed a serial communications link that does. Referred to as NAVLink, this communications link for control systems was defined for the high speed, bi-directional transfer of data necessary for controlling multi-axis applications, such as highly complex robotic arms or manipulators. Important characteristics of this serial format are high data throughput, reliable transmission and reception of data, and the synchronization of the motion controllers at each joint of the system. The format supports transfers in both directions, synchronous with each sample period, to provide adequate commands to the motion controllers and feedback to the Supervisory computer for efficient operation of the system. Other features are described in Table 2.6.1-1. NAVLINK has been tested for cable lengths up to 200 feet.

1) Highly structured serial format with specific segments devoted to receiving and transmitting data.
2) Specific number of words assigned to each Slave for bi- directional transferral of data.
3) Specific time slots allocated for each Slave to perform communications within each sample period.
4) Detection of parity errors and retransmission of information lost due to these errors.
5) Synchronization of serial shifting and processor clocks with each word of data received.
6) Simple interface format to reduce overhead between the Master bus controller and the Supervisory computer.
7) Random selection of data to be returned from each Slave controller to the Master bus controller.
8) Modification of all data memory locations and capability to load and modify application specific programs.
9) Synchronization of applications programs within each slave performed by the Master bus controller.

Table 2.6.1-1 Features of NAVLINK

2.6.2 Implementation of NAVLINK

Implementation of NAVLINK on both the Master and Slave boards involves three (3) primary components: a TMS320E25 digital signal processor, an electrically programmable logic device (EPLD) IC, and a differential line driver/receiver. This configuration is shown in Figure 2.6.2-1. The differential line driver/receiver is used to transmit and receive the information across a twisted pair of wires. The TMS320E25 DSP provides the computational capability necessary for processing the serial data and, on this DDSC, controlling the motion of two axes of a system. In addition, the serial link makes use of the DSP's two (2) double buffered serial ports, one (1) transmit and one (1) receive. In the receive mode, an electrically programmable logic device, EPLD, synchronizes the serial receive clock with that of the master. It also derives framing signals to control the DSP serial receive port. EPLD checks parity and alerts the DSP to errors in the received word. In the transmit mode the EPLD times out the data words and inserts parity bits and framing signals, as required.

This serial communications link is configured as a serial bus controller (Master) and up to seven (7) high performance, programmable DDSC motion controllers (Slaves) that communicate along this serial bus. Figure 2.6.2-2, previously presented, shows the communications slots for each slave and the order that the Master services them. The Master can independently monitor or modify the data memory within each Slave under its control, as well as load and modify application specific programs to each controller. This serial link was designed for transfers of data synchronous with the sample period. Each word of data that is transmitted over this line must conform to a specific format described below.

2.6.3 Serial Word Format

Figure 2.6.3-1 shows the nineteen (19) bit data word format for Navtrol's NAVLINK. The first two (2) spaces contain special synchronization signals. When received these are used to align the DSP receive serial shift clock and generate a framing signal for the DSP, alerting it to begin shifting in data. Using the synchronized clock the DSP then clocks in the next sixteen (16) bits, which make up the data word being transferred. The most significant data bit is clocked in first. The final bit of the serial stream, the parity bit, is compared within the EPLD to determine if an odd number of logic "1" bits, including parity, were received. A flag is set or cleared to inform the DSP to use or not use the data word. **PARITY IS CHECKED AT THE RECEIVING END FOR EACH WORD TRANSFERRED ACROSS THE SERIAL LINE.** Words with parity errors are not used. Errors are flagged to allow later retransmission of the data lost due to these errors.

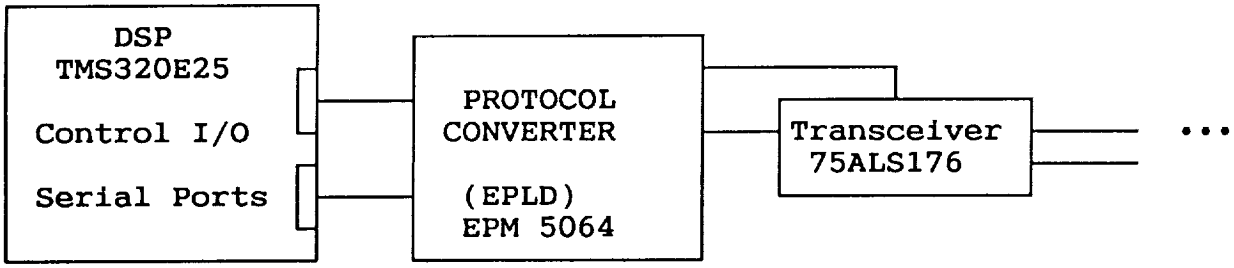
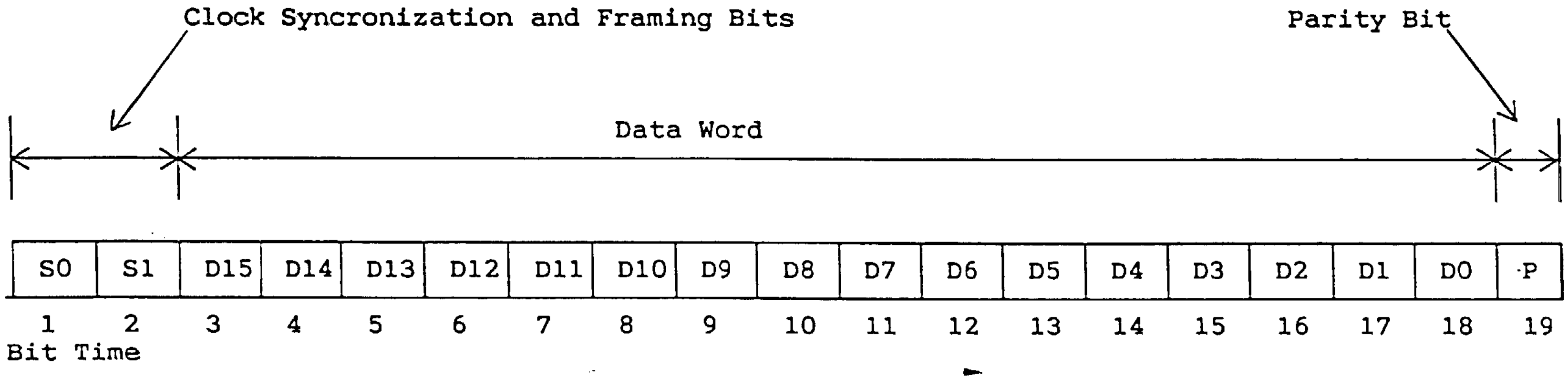


Figure 2.6.2-1 NAVLINK Terminal Configuration Block Diagram (Master and Each Slave)

S0	S1	S2	S3	S4	S5	S6	S C

S0 = SLAVE 0 COMMUNICATIONS SLOT
S1 = SLAVE 1 COMMUNICATIONS SLOT
S2 = SLAVE 2 COMMUNICATIONS SLOT
S3 = SLAVE 3 COMMUNICATIONS SLOT
S4 = SLAVE 4 COMMUNICATIONS SLOT
S5 = SLAVE 5 COMMUNICATIONS SLOT
S6 = SLAVE 6 COMMUNICATIONS SLOT
SC = SUPERVISORYCOMPUTER COMMUNICATIONS SLOT

Figure 2.6.2-2 Distribution Of Communications Slots



S0	S1	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	·P
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19

Bit Time

Figure 2.6.3–1

DATA FORMAT FOR NAVTROL'S ROBOTIC SERIAL LINK

2.6.4 Communications Slot Format And Description

The specific format for each communications slot is defined by Figure 2.6.4-1 which shows the distribution of reads and writes throughout the slot. (Note: "Read" refers to Slave to Master transfers; "Write" refers to Master to Slave transfers.) A total of sixty-eight (68) words are transferred in each slot and in each sample period for each Slave. The sample period (1/512) usually corresponds to the frequency which the Slave executes its control program, including communications. The communications portion of the software program in the Slave always occurs at the start of the sample period, followed by execution of the main control algorithms. Because of their high repetition rate, motor current PWM algorithms, by necessity, are intermingled with the communication routines. The Master initiates the communications cycle by transmitting the first of three (3) Slave address words, defined in Figure 2.6.4-2. A serial command word, defined in Figure 2.4.4-3, is sent to the Slave after the three (3) address words have been transmitted. This command word defines to the Slave the use to be made of data received later in the communications cycle. It defines whether the data in the two (2) "WRITE" slots are addresses or data and, if addresses, whether they are for read or write data and for which slot. When the Slave has determined from the Slave address words that it is "his turn" to communicate, he receives and stores the serial command word, then begins transmitting a string of eighteen (18) words to the Master. This string, the Read 0 and Read 1 segments, consists of the current status word for the slave and seventeen (17) data words. The addresses for these data words were defined in a previous communications cycle and are stored in an array in the Slave's memory. The data that is transmitted during these segments may come from the Slave's internal or external data memory.

After the Master has received the final word of the Read segments, it sends a string of ten (10) words to the Slave. This comprises the Write 0 segment. The contents of this string may be addresses for the data yet to be sent, the data to be stored in addresses previously sent, or addresses defining the source of data for the Read Segments 0, 1 and 2. The Slave determines the destination of this data by inspecting the serial command word received earlier. Note that to send new addresses and data to the Slave requires two (2) sample periods. When each word is received, parity is checked and valid data is stored into the appropriate address. Invalid data is discarded and a flag set to alert the Master.

After it has received the final word of the Write 0 segment, the Slave begins transmission of the Read 2 and Read 3 segments of the serial stream, which consists of fifteen (15) words of data. As was the case for the Read 0 and Read 1 segments, contents of this string are random data that have had their source addresses

ADDR	READ 0	READ 1	WRITE 0	READ 2	READ 3	WRITE 1	READ 4

ADDR	-	3 ADDRESS WORDS AND SERIAL COMMAND WORD (MASTER TO SLAVE)	4 WORDS
READ 0	-	STATUS AND RANDOM DATA READ SEGMENT 0 (SLAVE TO MASTER)	10 WORDS
READ 1	-	RANDOM DATA READ SEGMENT 1 (SLAVE TO MASTER)	8 WORDS
WRITE 0	-	RANDOM DATA WRITE SEGMENT 0 (MASTER TO SLAVE)	10 WORDS
READ 2	-	RANDOM DATA READ SEGMENT 2 (SLAVE TO MASTER)	8 WORDS
READ 3	-	RANDOM DATA READ SEGMENT 3 (SLAVE TO MASTER)	7 WORDS
WRITE 1	-	RANDOM DATA WRITE SEGMENT 1 (MASTER TO SLAVE)	10 WORDS
READ 4	-	PARITY ERRORS AND RANDOM DATA READ SEGMENT 4 (SLAVE TO MASTER)	11 WORDS
		TOTAL	--------- 68 WORDS

Figure 2.6.4-1 Serial Communications Slot Format

1	1	0	0	0	0	0	0	0	0	0	0	0	S	S	S

S = 3 BIT ADDRESS OF SLAVE
(SLAVE NUMBER 0 - 6)

Figure 2.6.4-2 Serial Address Word Format

0	0	0	0	0	0	0	0	R0	R1	R2	R3	R4	W0	W1	0

WHERE:

Rj = 1, j = 0, 1, 2 → Write 0 contents are new addresses for "Rj" data.

Rj - 0 -> No Operation

Rj = 1, j = 3, 4 → Write 1 contents are new Addresses for "Rj" data

Figure 2.6.4-3 Slave Serial Command Word Format

previously defined and transmitted by the Master and stored in the Slave's memory. Again, internal or external data memory may be the source for this data.

After reception of the final word of the Read 3 segment, the Master transmits to the Slave the Write 1 segment, a string of ten (10) words. Contents of this string may be addresses to define storage locations at the Slave for data, or the actual data to be stored in predefined addresses. This segment is the final transfer of information from Master to Slave for the cycle.

The last segment of the communication cycle is the Read 4 segment, which consists of eleven (11) words of random data, coming from internal or external data memory. Addresses for this data have been established in an earlier communication cycle as well. The final word of this segment is the "parity error word" for the communication cycle. These parity error flags are combined with similar flags in the Master and used by the Master to determine what data, if any, needs re-transmission and alert the Supervisory computer of communication problems.

After the Slave has sent the last word of the Read 4 segment, according to its commanded mode, it may begin execution of the application program resident in its memory. This application program can include servo-control algorithms, control function generation, coordinate transformations, etc. Upon completion of this program, the Slave enters an idle state and waits for the next communication cycle to begin.

The Master, after servicing a particular Slave, performs the same routine for each of the other Slaves under its control. After allowing time for the Supervisory Computer (a PC-AT or equivalent) to access the data received during the cycle or to set up data to be sent in the next cycle, the Master waits for its synchronization timer to begin the communication cycle again. This timer is presently configured for a 512 Hz sample rate.

2.7 Master Controller/Supervisory Computer Software Interface

The allocation of functions between the Master Controller and the Supervisory Computer coordinating system motion has been structured to relieve the Supervisory Computer of tedious tasks such as detection of parity errors and re-transmission of data lost due to these errors. Instead, the Master Controller performs these operations. By determining which segment of a particular Slave's communication slot contained an error, the Master will either retransmit to the Slave the data detected as erroneous or, in the case of a read error, delay transmission of new Read addresses to allow the Slave to send its data again.

The Master Controller also prevents overwriting of valid data within the Slave. Because each communications cycle contains data sent to each Slave, information received correctly by the Slaves must be protected from being overwritten by invalid data in the next cycle. To accomplish this, the Master sends to the Slave "harmless" dummy address and then follows this with dummy data to those addresses. These dummy parameters are not sent until transfer of valid addresses and data has been verified.

Addressing to a particular Slave unit is simplified for the Supervisory computer, as well. By storing the information necessary for communications into arrays within boundaries of sixteen (16) words and eight (8) words, the Supervisory Computer can set up or retrieve data for the Slave in question by simply adding the base address of the array to the Slave number or the shifted Slave number. Memory allocation of the different arrays within the data memory of the Master Controller board is defined in Table 2.7-1.

Commanding the Master serial bus controller has been simplified to reduce the overhead for the Supervisory Computer. By requiring the Supervisory Computer only to enter information into the appropriate arrays and provide a single command word, the time necessary for the serial bus controller and the Supervisory computer to communicate is greatly reduced. The format for this command word is given in Figure 2.7-1. The individual flags set within the command word are cleared as each operation is carried out and verified. Transmissions of new addresses for the Read segments, new addresses and data, etc., are processed until there are no commands pending. Then the Master clears this command word to notify the Supervisory Computer that the transfers have been completed.

3700h - 377Fh	Read 0 Address Arrays
3780h - 37FFh	Read 0 Data Arrays
3800h - 387Fh	Read 1 Address Arrays
3880h - 38FFh	Read 1 Data Arrays
3900h - 397Fh	Read 2 Address Arrays
3980h - 39FFh	Read 2 Data Arrays
3A00h - 3A7Fh	Read 3 Address Arrays
3A80h - 3AFFh	Read 3 Data Arrays
3B00h - 3B7Fh	Read 4 Address Arrays
3B80h - 3BFFh	Read 4 Data Arrays
3C00h - 3C7Fh	Write 0 Address Arrays
3C80h - 3CFFh	Write 0 Data Arrays
3D00h - 3D7Fh	Write 1 Address Arrays
3D80h - 3DFFh	Write 1 Data Arrays
3E00h - 3E7Fh	Previous Write 0 Buffers
3E80h - 3EFFh	Previous Write 1 Buffers
3F00h - 3F07H	Parity Error Words
3F08h - 3F0Fh	Slave Status Words
3F10h - 3F1Fh	Previous Slave Command Words
3F20h - 3F2Fh	Interface Command Words
3F30h - 3F3Fh	Write 0 Source Addresses
3F40h - 3F4Fh	Write 1 Source Addresses
3F50h - 3F5Fh	Slave Command Words
3F60h - 3F6Fh	Write 0 Buffer
3F70h - 3F7Fh	Write 1 Buffer
3FF0h - 3FFFh	Dummy Array

Table 2.7-1 Memory Allocation For Serial Arrays

							R_4	R_3	R_2	R_1	R_0	1_D	1_A	0_D	0_A

R0 – SEND READ 0 ADDRESSES

R1 – SEND READ 1 ADDRESSES

R2 – SEND READ 2 ADDRESSES

R3 – SEND READ 3 ADDRESSES

R4 – SEMD READ 4 ADDRESSES

0D – SEND WRITE 0 DATA

0A – SEND WRITE 0 ADDRESSES

1D – SEND WRITE 1 DATA

1A – SEND WRITE 1 ADDRESSES

Figure 2.7–2 Interface Command Word

2.8 Power Supplies

The DDSC requires only the motor voltage in order to function. All other power is generated on board. As shown in Figure 2.8-1 the motor voltage VPPf, is stepped down to approximately 12V though use of a switching regulator. As shown on Figure 2.8-2,this 12V is then chopped, applied to a transformer and used to generate the other voltages required on the board. Note that +2.5 volts, -7 volts and -12 volts are generated with respect to the motor voltage. These are used to supply the circuitry that drives the upper power transistors and measures the current through the motor load.

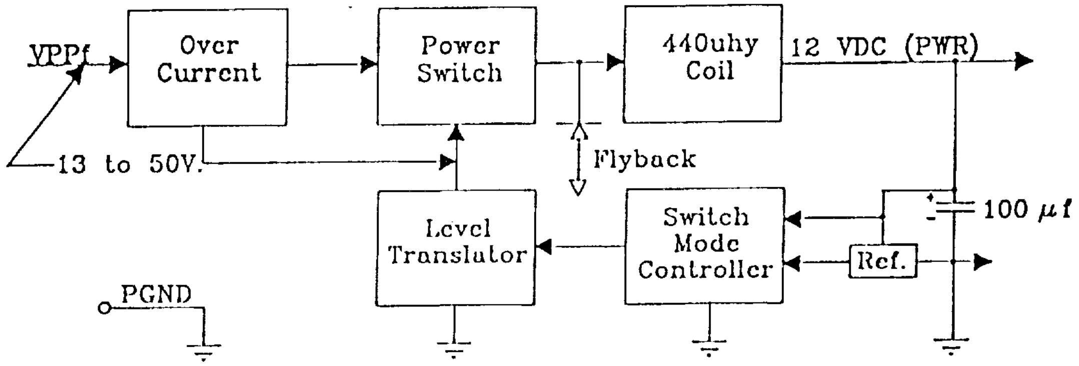


Figure 2.8-1 : Input Step Down Regulator

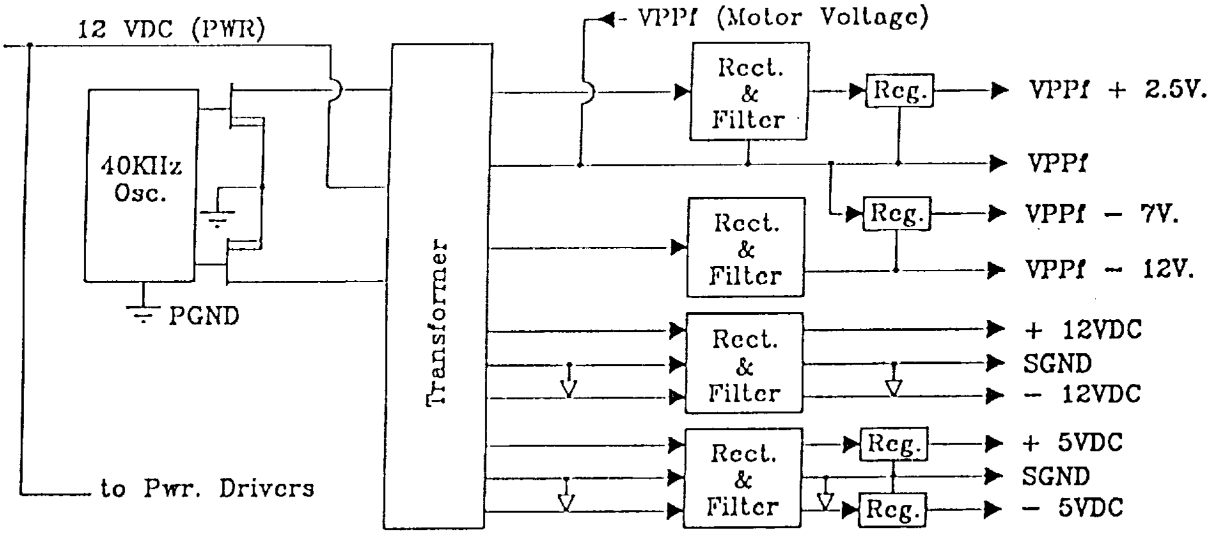


Figure 2.8-2 : Switching Voltage Generator

3. User's Guide for the DDSC

3.1 Hardware Interface Descriptions

3.1.1 Description of Signals and Pinouts for the DDSC

The Dual Digital Servo Controller (DDSC) was developed to be used for a variety of motion control and monitoring applications. The DDSC provides the capability to monitor and control two (2) independent axes of motion. These axes are referred to in this document as "Loop 0" and "Loop 1". Several of the input pins on the DDSC have been allocated for specific purposes, primarily for motion control functions. Other inputs are available to the user to provide optional monitoring of additional analog signals. Tables 3-1 and 3-2 list the various pins for the "Signal", or analog input connectors, and Table 3-3 presents the pins for the "Power", or motor output connectors on the DDSC. Note that while all of the connectors have inputs on them , all outputs are found on the "Power" connectors. All of the tables offer a brief description of each pin's function. Detailed descriptions of the various functions available on the DDSC are provided in subsequent sections. These further discussions will also define the configuration of the DDSC as it was delivered to NASA GSFC for control of a robotic gripper and nutrunner.

3.1.2 Basic Requirements for the DDSC

The DDSC contains all of the power supplies necessary for generation of the various voltages needed by the system. This simplifies the integration of the controller to a system by requiring only a single supply voltage and its return for system power. This supply voltage can range from 18 VDC to 45 VDC. Table 3-4 defines the connector pins associated with the system power and provides a brief description of each. The connector pins are rated at seven (7) Amps per pin; thus, for many applications not all pins are required from the current capability standpoint. However, using additional pins improve reliability and efficiency. It is recommended that at least two (2) power input pins and two (2) power return pins be used to assure operational reliability of the DDSC. If additional power supplies are used external to the DDSC (such as supplies for encoders or other sensors), a wire should be installed that connects the power ground of the DDSC and the ground of the additional supplies. Chassis ground should also be connected to the DDSC's power ground, if possible.

The other requirement for the DDSC to begin operation is a communications link with the Supervisory computer controlling it. The software within the DDSC is designed so that if a loss of communications occurs, the DDSC will go to and remain in an idle state (no motor commands) until communications are re-established

ANALOG INPUT CONNECTORS (DDSC - ROBOTIC)

J1 (Loop 0) = High Density "D-Sub" (26 pins)

Pin Number	Signal Description
1	Analog Input - Board ID.
2	Digital Output - Absolute Enc. Select
3	Analog Input - Channel 4 Non Invert[1]
4	Analog Input - High Gain Diff Pos 0
5	Analog Input - High Gain Diff Neg 0 or Channel 5 Inverting
6	Analog Input - High Gain Diff Pos 1 or Channel 5 Non Inverting
7	Analog Input - High Gain Diff Neg 0
8	Special Input - Sine[2]
9	Special Input - Cosine
10	Discrete Input - Limit Switch 0[3]
11	Diff. Comparator Input - Positive[4]
12	Diff. Comparator Input - Negative
13	Special Input - Discrete (Index)
14	Remote Sense (unused)
15	-5 VDC (100 ma Max.)
16	Ground
17	+5 VDC (100 ma Max.)
18	Special Inputs Return
19	Discrete Input - DIN 0[5]
20	Discrete Input - DIN 1
21	Discrete Input - DIN 2
22	Discrete Input - DIN 3
23	Discrete Input - DIN 4
24	Discrete Input - DIN 5
25	Discrete Input - DIN 6
26	Discrete Input - DIN 7

Table 3-1 Analog Input Connector Pinouts.

[1] Effective Input range for all analog inputs is -3.6v to +3.6v with a maximum of -5v to +5v.

[2] All Special Input signals, sine, cosine and discrete (Index), are referenced to the Special Inputs Return. All three signals are the "positive" inputs of separate differential comparators. There is a bias adjustment on the "negative" input of Special Signal (- Sine) to remove offsets. Special Input sine and cosine signals and their respective returns, after any adjustment for offset, are also routed to the GADIC for A/D conversion.

[3] Input range for limit switch inputs, 0 to +5 volts.

[4] Input range for Differential Compare is -5 to +5 volts.

[5] Input range for all Discrete Inputs is 0 to +5 volts.

J2 (Loop 1) = High Density "D-Sub" (26 pins)

Pin Number	Signal Description
1	Digital Output - Absolute Enc. Convt.
2	Analog Input - Channel 4 Non Invert[6]
3	Analog Input - Channel 4 Inverting
4	Analog Input - High Gain Diff Pos 0
5	Analog Input - High Gain Diff Neg 0 or Channel 5 Inverting
6	Analog Input - High Gain Diff Pos 1 or Channel 5 Non Inverting
7	Analog Input - High Gain Diff Neg 0
8	Special Input - Sine [7]
9	Special Input 1
10	Discrete Input - Limit Switch 0[8]
11	Diff. Comparator Input - Positive[9]
12	Diff. Comparator Input - Negative
13	Special Input - Discrete
14	Remote Sense (unused)
15	-5 VDC (100 ma Max.)
16	Ground
17	+5 VDC (100 ma Max.)
18	Special Input - Return
19	Discrete Input - DIN 0[10]
20	Discrete Input - DIN 1
21	Discrete Input - DIN 2
22	Discrete Input - DIN 3
23	Discrete Input - DIN 4
24	Discrete Input - DIN 5
25	Discrete Input - DIN 6
26	Discrete Input - DIN 7

Table 3-2 Analog Input Connector Pinouts.

[6] Effective Input range for all analog inputs is -3.6v to +3.6v with a maximum of -5v to +5v. For Loop 0 (J1-2) this pin is used for Absolute Encoder, the "Select".

[7] All Special Input signals, sine, cosine and discrete (Index), are referenced to the Special Inputs Return. All three signals are the "positive" inputs of separate differential comparators. There is a bias adjustment on the "negative" input of Special Input - Sine to remove offsets. Special Input sine and cosine signals and their respective returns, after adjustment for offset, are also routed to the GADIC for A/D conversion.

[8] Input range for limit switch inputs, 0 to +5 volts.

[9] Input range for Differential Compare is -5 to +5 volts.

[10] Input range for all Discrete Inputs is 0 to +5 volts.

POWER INPUT/OUTPUT CONNECTORS (DDSC - ROBOTIC)

J3 (J3 = 15 Pin "D-Sub")	Pin Number	J4 (J4 = 25 Pin "D-Sub")
Signal Ground	1	Signal Ground
Motor 0 - Phase 3[11]	2	Motor 1 - Phase 3
Motor 0 - Phase 2	3	+28 VDC Input
+12 VDC (100 ma Max)	4	Serial I/F (Positive)
Commutation 0 - Chan. A[12]	5	Power Ground
Motor 0 - Phase 1	6	D/A Output A[13]
Power Ground	7	Motor 1 - Phase 2
+28 VDC Input[14]	8	Discrete Output 00[15]
Motor 0 - Phase 3	9	Discrete Output 01
Motor 0 - Phase 2	10	Resolver Return
Commutation 0 - Chan. B	11	Motor 1 - Phase 1
Commutation 0 - Chan. C	12	Commutation 1 - Chan. C
Motor 0 - Phase 1	13	Commutation 1 - Chan. A
Power Ground	14	Motor 1 - Phase 3
+28 VDC Input	15	+28 VDC Input
	16	Serial I/F (Negative)
	17	Power Ground
	18	D/A Output B
	19	Motor 1 - Phase 2
	20	Discrete Output 10
	21	Discrete Output 11
	22	Resolver Excitation
	23	Motor 1 - Phase 1
	24	Power In (optional)
	25	Commutation 1 - Chan. B

[11] All Motor Phase outputs for Loop 0 are limited to 1 Amp, and 8 Amps for Loop 1; both loops @ 28 volts. These limitations are set by the sense resistors presently installed for the ROBOTIC PCB. They can be readily altered to 2, 4 or 8 Amps.

[12] All Motor Commutation Inputs must be limited to standard CMOS input voltage range (0 - +5 volts). These are normally open collector and pull up resistors must be provided.

[13] Both D/A output channels are of 12 bit resolution between +5v to -5v with approximately 15 mA source or sink output current capability. (TL084 op amp driver.)

[14] Input Power @ +28 VDC with 9.5 Amps maximum continuous output. (Depends on motor current required.)

[15] All four Discrete Outputs are 60v "open drain" MOSFET's. These outputs should be limited to a maximum of 400mA continuous maximum current.

Table 3-3 Power Output Connector Pinouts

Signal Name	Pin	Description
V++PWR	J3 - 8, 15 J4 - 3, 15	System power supply input. Voltage can range from 18 to 45 volts DC.
PWRGND	J3 - 7, 14 J4 - 5, 17	System power return. Also used to connect external and chassis grounds.
SYSPWR	J4 - 24	Power for system power supplies.

Table 3-4 DDSC Power Inputs

and the program and data memories of the DDSC are reinitialized. This serial link is designed for communication cables up to two hundred (200) feet in length or less. A twin-axial cable is used to connect the DDSC and the Serial Bus Controller or "Master", located in the PC-AT.

Please note that the "serial address", or Slave Number, of the DDSC must be set up before communications can properly commence. One of the analog inputs has been dedicated for selecting the serial address, or Slave number, of a DDSC. This input is read via the A/D converter and the voltage present determines the address that the Slave will respond to over the serial bus. This voltage ranges from 0 volts to -3.6 volts, and is set up by building a resistor network between ground and -5 volts. This resistor is installed external to the DDSC board on connector J1 between pins J1 - 1 and J1 - 16. This feature enables a DDSC that has experienced a failure to be easily replaced simply by disconnecting the defective board and installing a functional DDSC. Table 3-5 gives the required resistor for a particular Slave address. Addressing of the Slaves should always begin with 0, and increase as more Slaves are added to the serial bus, up to a maximum of seven (7) Slaves.

A signal ground is provided as a reference between the different controllers on the serial link. Line termination is also provided on both ends of the link. Table 3-6 lists the pins associated with the serial communications and also provides signal descriptions. Other details of the serial link are available in the Final Report on the DDSC and in the User's Manual for the serial link. The user should note that the serial communications takes about 265 uS per Slave to execute, with the sample rate of the DDSC set at 512 Hz (or 1953.1 uS). This leaves around 1688.1 uS to execute user specified algorithms (excluding PWM routines).

Slave Number	Resistor Value
0	0 ohms
1	1.5 K ohms
2	3.0 K ohms
3	4.7 K ohms
4	6.8 K ohms
5	10 K ohms
6	15 K ohms

All resistor values 5% tolerance

Table 3-5 Slave Address Selection

Signal Name	Pin	Description
SER+	J4 - 4	Positive differential input / output for serial data communications.
SER-	J4 - 16	Negative differential input / output for serial data communications.
SIGGND	J4 - 1	Signal Ground. Used to provide reference between DDSC and the Serial Communications Controller (Master).
SLVNUM	J1 - 1	Slave number selection input. Used to determine DDSC's address on serial bus. Input voltage ranges from 0 down to -3.6 VDC.

Table 3-6 Serial Communications Associated Inputs

3.1.3 Predefined Input Capabilities

As mentioned earlier, the DDSC has several inputs devoted to specific motion control functions. Special signal inputs and a special discrete are provided for <u>each</u> loop of the DDSC to interface with a variety of sensors (optical incremental encoder, rate gyro, etc.) The special signal inputs and the special discrete can be referenced to a common return line (special input signal return) or a remote sense line. The ability to adjust the offset for Special Input 0 for each Loop is provided on the DDSC by an eight (8) bit quad D/A converter. The special signal inputs and the special discrete each go to the positive inputs of differential comparators. The special signal return mentioned above becomes the negative inputs of these same comparators. The discrete outputs of the comparators become special inputs into the DDSC interface EPLD. The special signal inputs and their associated return are also routed into the A/D converter circuitry. The A/D has nine (9) bit plus sign resolution (-3.6 < Vin < +3.6 volts), and two (2) special signal inputs are capable of being sampled (gain = 1) at the PWM update rate (> 10 kHz) for each loop of the DDSC. Table 3-7 lists the special signal pins and presents a brief description of each input. For the robotics end effector controller, the special signal input and special discrete features are not used. However, a software selectable option to use the commutation feedback from the nutrunner for position and velocity measurement was developed and implemented in the interface EPLD.

The capability to monitor limit switches for each of two (2) axes is also provided. Each axis has a discrete input assigned for monitoring a limit switch. These inputs are read every sample period by the DDSC and stored for use by the program. These inputs are equivalent to a typical TTL input. Limit switches are typically used to indicate the end of a motor's excursion in both directions. All of the limit switch inputs are pulled up to +5 volts on the DDSC through 4.7 k-ohm resistors. The limit switch inputs are not used for the robotics end effector controller as delivered to NASA GSFC Robotics Branch.

Signal Name	Pin	Description
SPRET0	J1 - 18	Special input signal return for Loop 0. Buffered on-board to provide independent bias adjust for special inputs 0 and 1, as well as the special discrete.
SPDSC0	J1 - 13	Special discrete input for Loop 0. Referenced to Loop 0 special signal return (w/ bias adjust).
SPIN00	J1 - 8	* Special signal input 0 for Loop 0. Referenced to Loop 0 special signal return (w/ bias adjust).
SPIN01	J1 - 9	* Special signal input 1 for Loop 0. Referenced to Loop 0 special signal return (w/ bias adjust).
RSENS0	J1 - 14	Remote sense for Loop 0 analog inputs.
SPRET1	J2 - 12	Special input signal return for Loop 1. Buffered on-board to provide independent bias adjust for special inputs 0 and 1, as well as the special discrete.
SPDSC1	J2 - 13	Special discrete input for Loop 1. Referenced to Loop 1 special signal return (w/ bias adjust).
SPIN10	J2 - 8	* Special signal input 0 for Loop 1. Referenced to Loop 1 special signal return (w/ bias adjust).
SPIN11	J2 - 9	* Special signal input for Loop 1. Referenced to Loop 1 special signal return (w/ bias adjust).
RSENS1	J2 - 14	Remote sense for Loop 1 analog inputs.

* = Measurable input range is +/- 3.6 VDC. Absolute maximum input is +/- 5 VDC.

Table 3-7 Special Signal Inputs

Other Discrete inputs are available (eight (8) per Loop) for reading discrete signals such as enable and mode switches, status signals, etc. These inputs are accessed when a particular Loop reads its Discrete Input Interface. The numbering of the discrete inputs is an indication of the order of precedence for these bits. Table 3-8 lists the pins associated with discrete signal input. The table also lists the bit position in the data word for each input. These discrete inputs are pulled up to +5 volts through 10 k-ohm resistors. Please note that the commutation signals are read in through the interface EPLD, not the Discrete Interface.

Several other analog inputs on the DDSC are available to the user. By modifying the applications program, these inputs can be configured as single-ended (referenced to ground) or differential inputs. Variable gain (1 - 128) is also available for these inputs. Table 3-9 provides information for all analog inputs not defined previously. These input channels are capable of measuring voltages ranging between + 3.6 and - 3.6 volts (max rating: +/- 5 volts), with nine (9) bits plus sign resolution. These inputs also have filtering already in place on the board. Other options for collecting data through the A/D are described in detail in the discussion of the GADIC's in the Final Report of the DDSC.

Two (2) general purpose differential comparators (one (1) per loop) are also available, should they be needed. Table 3-10 provides the connector pin positions and descriptions for these inputs. The positive and negative inputs for these devices are also filtered in the same manner as the analog inputs described above. The output from these compares is read through the Discrete Input Interface into the DDSC's processor. The voltage range for these comparator inputs are -5 to +5 volts.

3.1.4 Predefined Output Capabilities

Since the Dual Digital Servo Controller was designed with motion control applications in mind, the ability to drive two (2) DC brush-type or brushless motors is provided on the board. These motors may have up to three (3) phases, with multiple pole pairs. Motor current is provided by pulse width modulated amplifiers. These outputs are to be connected to the appropriate windings of the motor to be controlled. Table 3-10 lists these motor outputs and their ratings. The width of the output pulses, as well as the commutation switch selection, are computed by the DDSC's processor. Parameters defining the actuator being controlled are entered through the monitor program DEMON, and the appropriate gains and constants for the pulse width amplifiers and commutation are generated and transferred to the DDSC.

Table 3-8 DDSC Discrete Inputs

Signal Name	Pin	Description
LIM00	J1 - 10	* Negative limit switch for Loop 0. Read on data bit 0 by the DSP.
LIM01	J1-11,12	* Positive limit switch for Loop 0. Read on data bit 1 by the DSP.
LIM10	J2 - 10	* Negative limit switch for Loop 1. Read on data bit 0 by the DSP.
LIM11	J2-11,12	* Positive limit switch for Loop 1. Read on data bit 1 by the DSP.
DISC00	J1 - 19	* Discrete input 0 for Loop 0. Read on data bit 0 by the processor.
DISC01	J1 - 20	* Discrete input 1 for Loop 0. Read on data bit 1 by the processor.
DISC02	J1 - 21	* Discrete input 2 for Loop 0. Read on data bit 2 by the processor.
DISC03	J1 - 22	* Discrete input 3 for Loop 0. Read on data bit 3 by the processor.
DISC04	J1 - 23	* Discrete input 4 for Loop 0. Read on data bit 4 by the processor.
DISC05	J1 - 24	* Discrete input 5 for Loop 0. Read on data bit 5 by the processor.
DISC06	J1 - 25	* Discrete input 6 for Loop 0. Read on data bit 6 by the processor.
DISC07	J1 - 26	* Discrete input 7 for Loop 0. Read on data bit 7 by the processor.
DISC10	J2 - 19	* Discrete input 0 for Loop 1. Read on data bit 8 by the processor.
DISC11	J2 - 20	* Discrete input 1 for Loop 1. Read on data bit 9 by the processor.
DISC12	J2 - 21	* Discrete input 2 for Loop 1. Read on data bit 10 by the processor.
DISC13	J2 - 22	* Discrete input 3 for Loop 1. Read on data bit 11 by the processor.

Table 3-8 DDSC Discrete Inputs (continued)

Signal Name	Pin	Description
DISC14	J2 - 23	* Discrete input 4 for Loop 1. Read on data bit 12 by the processor.
DISC15	J2 - 24	* Discrete input 5 for Loop 1. Read on data bit 13 by the processor.
DISC16	J2 - 25	* Discrete input 6 for Loop 1. Read on data bit 14 by the processor.
DISC17	J2 - 26	* Discrete input 7 for Loop 1. Read on data bit 15 by the processor.

* = Inputs pulled up on-board by 10 K resistor. Discrete inputs require either 0 or 5 VDC. All discrete inputs are read once per sample period.

Signal Name	Pin	Description
APOS04	J1 - 3	* Positive input for GADIC 0, mux. channel 4.
APOS05	J1 - 6	* Positive input for GADIC 0, mux. channel 5.
ANEG05	J1 - 5	* Negative input for GADIC 0, mux. channel 5.
APOS06	J1-4,5	* Positive input for GADIC 0, mux. channel 6 (from High Gain 0).
ANEG06	J1-6,7	* Negative input for GADIC 0, mux. channel 6 (from High Gain 1).
APOS14	J2 - 2	* Positive input for GADIC 1, mux. channel 4.
ANEG14	J2 - 3	* Negative input for GADIC 1, mux. channel 4.
APOS15	J2 - 6	* Positive input for GADIC 1, mux. channel 5.
ANEG15	J2 - 5	* Negative input for GADIC 1, mux. channel 5.
APOS16	J2-4,5	* Positive input for GADIC 1, mux. channel 6 (from High Gain 0).
ANEG16	J2-6,7	* Negative input for GADIC 1, mux. channel 6 (from High Gain 1).

* = Measurable analog input range is from - 3.6 volts to + 3.6 volts. Absolute maximum input is + or - 5 volts.

Table 3-9 DDSC Analog Inputs (Unused)

Signal Name	Pin	Description
CMP0P	J1 - 11	Loop 0 differential comparator positive input. The output of this compare is read via the GADIC 0 on data bit 1. Input voltage can range from - 5 volts to + 5 volts.
CMP0N	J1 - 12	Loop 0 differential comparator negative input.
CMP1P	J2 - 11	Loop 1 differential comparator positive input. The output of this compare is read via the GADIC 1 on data bit 1. Input voltage can range from - 5 volts to + 5 volts.
CMP1N	J2 - 12	Loop 1 differential comparator negative input.

Table 3-10 DDSC Comparator Inputs and Spare Inputs

Signal Name	Pin	Description
MOT01	J3 - 6, 13	Loop 0 motor output, phase 1.
MOT02	J3 - 3, 10	Loop 0 motor output, phase 2.
MOT03	J3 - 2, 9	Loop 0 motor output, phase 3.
MOT11	J4 - 3, 11	Loop 1 motor output, phase 1.
MOT12	J4 - 7, 19	Loop 1 motor output, phase 2.
MOT13	J4 - 2, 14	Loop 1 motor output, phase 3.

Loop 0 motor phases will presently supply about 1 amp of current max; Loop 1 will supply about 8 amps of current max. Do not exceed this rating or damage may occur to the DDSC.

Table 3-11 DDSC Motor Power Outputs

Two (2) channels of D/A output are available on the DDSC. These D/A converters have a range of -5 to +5 volts, with twelve (12) bits plus sign resolution. Four (4) discrete outputs are also provided, each with the capability of sinking 0.5 amps. These discrete outputs are open drain configuration, with a max voltage input of 60 volts. Reference voltages (+5, -5, Gnd) are provided for circuitry not on the DDSC. Table 3-12 lists the D/A, discrete and power outputs for the DDSC and defines their capabilities. Signal ground has also been provided at the connector, for a reference. These voltage outputs are to be used for references only, not for high current applications. To power circuits or sensors external to the DDSC, +12 volts is provided. The current rating for this voltage is 100 mA. A suggested use of the +12 volt supply would be to power the Hall effect devices used for motor commutation feedback or to power an encoder.

3.1.5 Configuration of NASA GSFC's DDSC (Robotics)

As delivered to NASA GSFC Robotics Branch, the DDSC was configured to interface with the gripper supplied by NASA GSFC and the nutrunner associated with the gripper. The serial address for the delivered DDSC has been preset to address 0 (Slave 0). The Gripper is controlled by Loop 0 of the DDSC; the nutrunner is controlled by Loop 1. Note that some of the Loop 1 analog inputs were needed to interface with the many strain gages on the gripper fingers. Also note that the two (2) loops are NOT interchangeable; that is, the connectors can not be switched and the system operated properly.

3.1.5.1 Description of End Effector Feedback

The gripper control feedback consists of a potentiometer to provide position information and numerous strain gages for force feedback data. Table 3-13 lists the pins utilized by the gripper control algorithms. Table 3-14 lists those DDSC pins associated with controlling the nutrunner. Figure 3-1 represents a block diagram of the gripper position and force control system. Figure 3-2 is a system block diagram of the nutrunner control.

The potentiometer for gripper position feedback was connected between +5 and -5 volts to provide increased resolution of the relative position of the gripper fingers. The potentiometer wiper voltage is measured as a single-ended (referenced to ground) input with gain = 1. The voltage measured is corrected for any A/D biases and then scaled and offset adjusted to create a position measurement for the gripper. This measurement is used for the positional control of the gripper.

Signal Name	Pin	Description
VOUTA	J4 - 6	D/A output chan. A. (-5 < V < +5).
VOUTB	J4 - 18	D/A output chan. B. (-5 < V < +5).

DESOUT0	J4 - 8	Discrete output chan. 0. (0.5 amps)
DESOUT1	J4 - 9	Discrete output chan. 1. (0.5 amps)
DESOUT2	J4 - 20	Discrete output chan. 2. (0.5 amps)
DESOUT3	J4 - 21	Discrete output chan. 3. (0.5 amps)

+12VDC	J3 - 4	External power output, + 12 VDC.
+5VDC	J1 - 17 J2 - 17	Reference voltage, +5 VDC, for external circuitry. Rated for 50 milliamps.
-5VDC	J1 - 15 J2 - 15	Reference voltage, -5 VDC, for external circuitry. Rated for 50 milliamps.
GROUND	J1 - 16 J2 - 16	Reference voltage, 0 VDC, for external circuitry. Keep isolated from motor power ground.

Table 3-12 Misc. DDSC Outputs and References

Signal Name	Pin	Description
POTFB	J1 - 9	Potentiometer feedback from gripper.
LNGFRCP	J1 - 5	Positive differential input from gripping force strain gage bridge for long finger.
LNGFRCN	J1 - 4	Negative differential input from gripping force strain gage bridge for long finger.
SHTFRCP	J2 - 5	Positive differential input from gripping force strain gage bridge for short finger.
SHTFRCN	J2 - 4	Negative differential input from gripping force strain gage bridge for short finger.
LATFRCP	J1 - 6	Positive differential input from lateral (side) force strain gage bridge.
LATFRCN	J1 - 7	Negative differential input from lateral (side) force strain gage bridge.
ROTFRCP	J2 - 6	Positive differential input from rotary (twist) force strain gage bridge.
ROTFRCN	J2 - 7	Negative differential input from rotary (twist) force strain gage bridge.
GRPMTR0	J3 - 6	Gripper motor output 0.
GRPMTR1	J3 - 3	Gripper motor output 1.

Table 3-13 Gripper Control Inputs / Outputs

Signal Name	Pin	Description
JCOM_A1	J4 - 13	* Commutation sensor input channel A.
JCOM_B1	J4 - 25	* Commutation sensor input channel B.
JCOM_C1	J4 - 12	* Commutation sensor input channel C.
MOT11	J4 - 11 J4 - 23	Loop 1 motor output phase 1. Note that two (2) pins are available and should be utilized for reliability.
MOT12	J4 - 7 J4 - 19	Loop 1 motor output phase 2. Note that two (2) pins are available and should be utilized for reliability.
MOT13	J4 - 2 J4 - 14	Loop 1 motor output phase 3. Note that two (2) pins are available and should be utilized for reliability.

* Commutation sensor inputs are "pulled up" to +5 volts through resistors (4.7 K ohm), and referenced to signal ground.

Table 3-14 Nutrunner Control Inputs / Outputs

The various strain gauges used for the force feedback are connected into bridges on a small PC board mounted at the gripper. This was done to reduce the susceptibility to noise of the high impedance strain gage inputs. The bridges were excited with +5 and -5 volts to provide maximum sensitivity. Null adjustments are provided on the remote PC board for removing any biases from each bridge output. The bridge outputs are differentially amplified (gain = 50) and then fed into the A/D converter, which has an additional gain of 8 or 16. The measured voltages are then scaled to produce force measurements for each gripper finger, the lateral (side) force, and the rotary (twist) force. The forces measured on the fingers are then combined to provide the measurement for the force control system.

As mentioned earlier, the position and velocity feedback for the nutrunner is derived from the nutrunner motor commutation signals. New techniques were developed that provide fairly accurate measurements of motor position and velocity, without the added expense and circuitry associated with an encoder.

3.1.5.2 Description of End Effector Control Outputs

Motor current capability of the gripper axis (Loop 0) has been limited to just over one (1) amp, as this is the maximum current the gripper motor will stand. Because the gripper motor is a brush-type motor, only two phases of the motor output section are necessary (phase 1 and 2). The direction of the motor is determined by the control algorithms.

For the nutrunner axis, the current capability was increased to just over eight (8) amps. This was to provide as much torque as possible at the nutrunner output. The nutrunner motor is a three (3) phase DC brushless type, so all motor output phases are utilized. Note that two (2) pins are available for each motor phase, and both pins should be connected to the motor with parallel wires. This will provide more reliable operation of the nutrunner motor, which draws considerably more current than the gripper motor.

The four (4) discrete outputs are not used by the end effector control algorithms. The two (2) channels of D/A converter output are also not utilized for the end effector controller. None of the sixteen (16 = 8 * 2) discrete inputs are used for the end effector controller as delivered.

3.2 DDSC Software Descriptions

3.2.1 Introduction

This section is presented to provide the User with information concerning the software features of the DDSC. This information includes memory maps for both hardware and software addresses, as well as subroutines embedded within the EPROM and general program structure. Segments of data and program memory available to the User will also be listed. The User should be familiar with the instruction set for the TMS320X25 Digital Signal Processor (DSP), as well as fixed point arithmetic and scaling techniques, in order to correctly implement a control or monitoring program in the DDSC. Note that the code contained within the **EPROM is not available for modification** by the User, and therefore should not be attempted.

After listing the program and data memory maps, the software interfaces with the various peripherals of the DDSC will be discussed. Examples of actual TMS320X25 code will be presented to provide further explanation. These examples will be general in nature, with options noted as necessary.

The format for presenting software addresses, data, and masks, will be in hexadecimal (or base 16). For clarity, these hex numbers will be presented as: **0NNNNh** or **0NNh.** These represent a sixteen (16) bit or eight (8) bit value, respectively. Where applicable, examples will be offered to help illustrate the feature being discussed. System software descriptions will have program memory addresses, data memory addresses, MSB values, etc. Software features that require special scaling or other operations will be further specified in the discussion of that feature.

3.2.2 Program Configuration for the DDSC

Programs developed for execution by the DDSC may perform a variety of tasks. These programs have certain constraints placed upon them, and these restrictions are presented here. Figure 3-1 illustrates the flow of a typical DDSC program. The typical program "order of events" for execution within the DDSC would be:

0) Reset / Power up Initialization
1) Start of serial communications - wait for proper serial address (synchronization of applications program).
2) Start execution of pulse width interrupts and service serial communications.

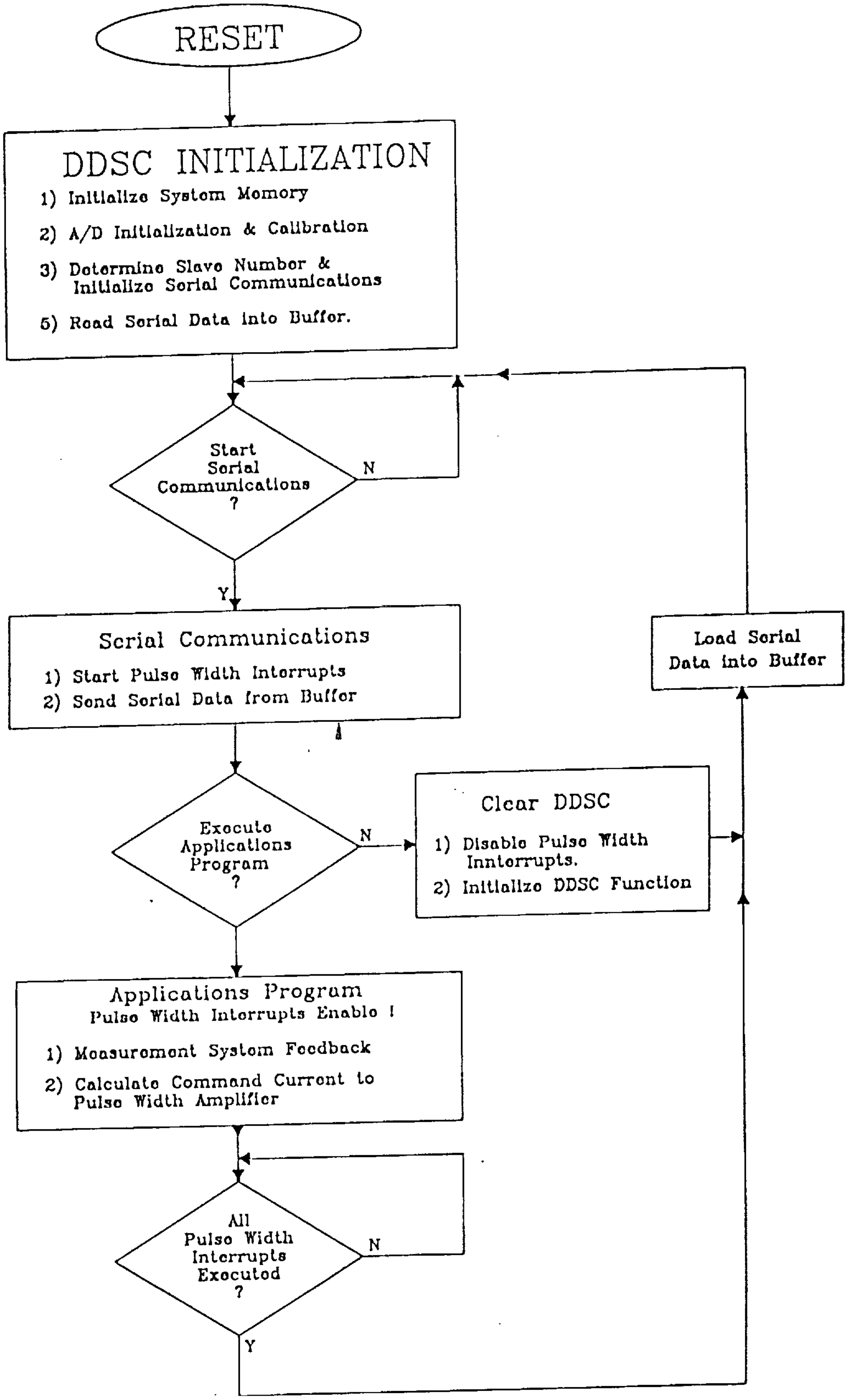

RESET
DDSC INITIALIZATION
1) Initialize System Memory
2) A/D Initialization & Calibration
3) Determine Slave Number & Initialize Serial Communications
5) Read Serial Data into Buffer.
Start Serial Communications ?
N
Y
Serial Communications
1) Start Pulse Width Interrupts
2) Send Serial Data from Buffer
Load Serial Data into Buffer
Execute Applications Program ?
N
Clear DDSC
1) Disable Pulse Width Innterrupts.
2) Initialize DDSC Function
Applications Program
Pulse Width Interrupts Enable !
1) Measurement System Feedback
2) Calculate Command Current to Pulse Width Amplifier
All Pulse Width Interrupts Executed ?
N
Y

Figure 3-1: DDSC Software Flowchart

3) End of serial communications.
   A) If "clear" mode, disable pulse width interrupts, fill serial buffer with required data and go to step 1.
   B) If "operate"mode, allow pulse width interrupts to occur and execute applications program.
4) Applications program - measure system feedback and calculate command current for pulse width algorithms.
5) End of applications program - fill serial buffer with required data and go to step 1.

Each "sample period", or iteration of the DDSC's program, always starts with servicing the serial communications. It is during this communications segment that the pulse width interrupts are enabled and servicing begins. The timing of the serial communications with the pulse width interrupts is very critical and required extensive time to complete. **User modifications of this timing or the serial communications software are not recommended.** After all of the data to be transferred from the DDSC to the Master has been sent via the serial interface, the DDSC determines whether to begin the applications program or remain in a "clear" state and wait to service the next communications cycle. Note that the serial communications is practically transparent to the User and requires little overhead from the applications program.

It is assumed that the applications program utilizes the pulse width interrupts, although this is not a necessary requirement. Should the DDSC's pulse width interrupts not be required by the applications program, the interrupts must be disabled to prevent timing variations. The interrupts are disabled by writing to the special configuration register (SPCNF) located within the GADIC. The address offset for the special configuration register is:

SPCNF : 004h.

If the pulse width interrupts are not needed, the interrupts for both loops must be disabled for proper timing to occur. The code to disable the pulse width interrupts is as follows (to disable the interrupts for Loop 0 only):

```
        LARP 0          ; point to auxiliary register 0
        LRLK AR0,(GADIC0+SPCNF)
        LACK 000h       ; turn off bypass interrupts
        SACL *          ; store data to GADIC
```

The User must also mask off the pulse width interrupts within the DSP processor, to prevent false interrupts from occurring.

The applications program itself has few necessary requirements from the perspective of the DDSC. If the pulse width interrupts are utilized, the command current must be calculated and stored to command the interrupts. Should only one (1) channel of the pulse width interrupts be required for a particular application, the unused Loop's command current **must** be set to zero (0). There are no timing constraints on the calculation of the command current, although it should occur at the same time each sample period to provide "smooth" system response. A suggested method of calculating these command currents would be to perform the Loop 0 system measurement and current calculation first, followed by the Loop 1 measurement and current computation.

At the end of the applications program, the data to be transferred by the serial communications must be collected into the serial data buffer. This is accomplished by calling the subroutine **SERfilbfr**, which is contained within the EPROM of the DDSC and will be discussed later. Execution of this subroutine requires a bit of time, so the User must allow sufficient time to permit the subroutine to complete. A suggested method of determining when to call this serial subroutine is to synchronize with the pulse width interrupts. Because the serial communications occurs at a rate of 512 Hz and the pulse width interrupts occur every 97.6 uS per loop, twenty (20) interrupts must occur per loop (total of forty (40) interrupts). To provide adequate time for the serial buffer filling routine to complete, it is suggested that the applications program sync with the final four (4) pulse width interrupts (two (2) per loop). Further discussion of the serial communications and its timing considerations are provided in a subsequent section. After filling the serial buffer with the required data, the program control must pass to the start of the serial communications, **SERgetsnc.** Addresses for these routines and program locations are also presented in the section concerning serial communications.

3.2.2.1 Applications Program Timing Considerations

The rate of execution for the DDSC's application program is presently set at 512 Hz (1.953 mS). This timing is controlled by the Master and reprogramming of both Master and slaves is necessary to change it. There is much interaction, not only between the Master and Slaves communication software routines, but also with the slaves PWM current control sub-routines. Adjustment of this timing moust be undertaken with considerable caution. With the present timing the time available for User defined algorithms to execute is approximately 840 uS (the sample period less time required for serial communications and both loops of pulse width interrupts). This may not seem like a large amount of time, but the TMS320X25 DSP can execute up to 8400 single cycle instructions during this period. (Most of the DSP's instructions are single cycle with 100 nS execution time). Should the time required for the applications program to execute exceed this value, the DDSC will miss a communications cycle and cause the controller to go into a clear state. If two (2) loops are being controlled, each loop will have about 420 uS to execute the necessary algorithms, which is sufficient for most applications. This time is distributed between the different channels of pulse width interrupts and is not available as a contiguous block.

3.2.3 DDSC System Memory Maps

The addresses for the hardware peripherals and the embedded software features are provided to the User for reference and to prevent any address contentions. Program and data memory addresses and allocation are also presented.

3.2.3.1 DDSC Hardware Addresses

The following is a list of the hardware peripherals found on the DDSC and their associated addresses, along with a brief description of each peripheral. Some of the signals listed are not available to the User, and should not be utilized. The addresses marked with an asterisk (*) are those signals considered "off limits" to the User, and should not be accessed.

Peripheral	Address		Description
COMMUT0	04000h	R	Loop 0 Commutation Input Register
COMMUT1	04400h	R	Loop 1 Commutation Input Register
QDAC	04800h	W	Quad D/A Converter (Bias Adjust)
CLR61	* 05400h	W	Clear for /61 clock (Pulse Width)
ENDREC	* 05800h	W	End of Receive Discrete (Serial)
XMTCLR	* 05800h	R	Transmit Clear Discrete (Serial)
SRESET	* 05C00h	R	Serial Hardware Reset (Serial)
ENC0	0C000h	R	Loop 0 Encoder Interface
XCOUNT0	0C000h	R	Loop 0 Transition Counter (COMENC)
ENC1	0C400h	R	Loop 1 Encoder Interface
XCOUNT1	0C400h	R	Loop 1 Transition Counter (COMENC)
TCOUNT0	0C800h	R	Loop 0 Time Counter (COMENC)
ECONF0	0C800h	W	Loop 0 Encoder Configuration Register
TCOUNT1	0CC00h	R	Loop 1 Time Counter (COMENC)
ECONF1	0CC00h	W	Loop 1 Encoder Configuration Register
GADIC0	0D000h	R/W	Loop 0 GADIC Base Address
GADIC1	0D400h	R/W	Loop 1 GADIC Base Address
DSCRTIN	0D800h	R	Discrete Input Interface
DACCOM	0DC00h	W	D/A Converter (12 bit) Interface

W = Write Only R = Read Only

COMENC = Commutation-based Encoder Interface

This list contains all of the hardware addresses used on the DDSC. Note that several of the addresses are merely base addresses (such as the GADICs) that have more than one (1) register associated with them. To generate the complete address for a register, add the register offset to the base address for the peripheral in question. Care should be taken to access a peripheral in the proper manner; i.e., do not write to a peripheral that requires a read and vice versa. Should there be any questions about these addresses or peripherals, please contact Navtrol for more information.

3.2.3.2 DDSC Software Addresses

Addresses of software routines within the DDSC are provided to allow the User to make use of these segments of code. Program memory addresses are presented along with a brief description of the routine. Data memory addresses are furnished with a description and other pertinent information (such as MSB values, initial conditions, etc.).

3.2.3.2.1 Program Memory Addresses

Program memory for the DDSC resides in two (2) locations: within the EPROM of the TMS320E25 and in the external RAM. The EPROM addresses are part of the embedded "operating system" of the DDSC, while the external RAM addresses are associated with the applications program developed by the User. Note that the User can only utilize the external RAM portion of the program memory map, as the EPROM must not be altered.

Internal program memory addresses and their descriptions are:

Routine	Address	Description
DELAY	0012Dh	Variable Time Delay Routine
FIXDAT	00131h	A/D Masking Routine
LOADDAC	0013Ah	D/A Service Routine
SERgetsnc	001A2h	Serial Communications Start
SERnxtsnc	004B9h	Serial Comm. "Clear" Routine
SERfilbfr	00529h	Serial Comm. Data Collection

These are the only routines that the User has access to within the EPROM. Other routines, such as the pulse width algorithms, system initialization, etc., are located within the EPROM, but should not be accessed directly by the User.

External program memory starts at address 02000h and extends to address 03AFFh (total of 6912 words available). The User should note that all applications programs must start at address 02000h.

3.2.3.2.2 Data Memory Addresses

Data memory for the DDSC is located throughout the internal RAM of the TMS320X25, as well as a portion of the external RAM. The internal data memory locations are used primarily to contain system variables and constants, while the external memory is used for tables and arrays. The following is a listing of the internal RAM storage locations and a brief description of each, along with other information, if applicable.

Mnemonic	Address	Description
PWSBUF	060h - 064h	Serial buffer for pulse width calc.
PWTMP0	065h - 069h	Loop 0 pulse width temporary storage
BRANCH	06Ah	Program Execution control variable
MSTAT0-2	06Bh - 06Dh	Status reg. storage for Math routines
STAT1-2	06Eh, 06Fh	Status reg. storage for interrupts
PWVAR0	070h - 072h	Loop 0 pulse width variables
PWVAR1	073h - 075h	Loop 1 pulse width variables
PWTMP1	076h - 07Ah	Loop 1 pulse width temporary storage
	07Bh	Not Used
I2VCT	07Ch	Interrupt 2 vector address storage
SFTVCT	07Dh	Software int. vector address storage
	07Eh, 07Fh	Not Used
JTABLE	00200h	Program function selection variable
STATUS	00201h	System status word.
SCOUNT	00202h	Serial Comm. iteration counter
ITERN	00203h	Applications prog. iteration counter
LOOP	00204h	Appl. prog. Loop index variable
LANGLE	00205h	"Last angle" variable
NANGLE	00206h	"Current angle" variable
TJTABL	00207h	Temporary func. select. variable
BFRPT0	00208h	Array Collection buffer pointer 0
BFRPT1	00209h	Array Collection buffer pointer 1
ADDR0	0020Ah	Array Collection source pointer 0
ADDR1	0020Bh	Array Collection source pointer 1
SRTMP0	0020Ch	Temporary storage for serial
SRTMP1	0020Dh	Temporary storage for serial
LSLIMM	0020Eh	Limiting Variable (MS word)
LSLIML	0020Fh	Limiting Variable (LS word)
ACMS	00210h	Accumulator storage for interrupts
ACLS	00211h	Accumulator Storage for interrupts
PRMS	00212h	Product reg storage for interrupts
PRLS	00213h	Product reg storage for interrupts
TREG	00214h	Temp register storage for interrupts
SVAR2	00215h	Aux. Reg. 2 storage for interrupts
SVAR3	00216h	Aux. Reg. 3 storage for interrupts
TMPADR	00217h	Temporary address storage for serial
LPCNT	00218h	Pulse Width Interrupt Counter...

PARERR	00219h	Serial communications parity errors
SYNC	0021Ah	Serial Comm. address variable
COMMD	0021Bh	Serial Comm. command variable
SERBF1	0021Ch	Serial Comm. temp. storage
COLPTR	0021Dh	Serial Comm. pointer storage
ITMP0	0021Eh	Pulse Width temporary storage
DSCRTS	0021Fh	Discrete Input storage location
ADGAN0	00220h	Loop 0 system gain selection
ADGAN1	00221h	Loop 1 system gain selection
MXCNF0	00222h	Loop 0 system mux. configuration
MXCNF1	00223h	Loop 1 system mux. configuration
LIMSW0	00224h	Loop 0 limit switch storage
LIMSW1	00225h	Loop 1 limit switch storage
DOUT0	00226h	Loop 0 discrete output variable
DOUT1	00227h	Loop 1 discrete output variable
DADAT0	00228h	D/A command voltage Ch. A
DADAT1	00229h	D/A command voltage Ch. B
QDACA	0022Ah	Quad D/A Ch. A voltage
QDACB	0022Bh	Quad D/A Ch. B voltage
QDACC	0022Ch	Quad D/A Ch. C voltage
QDACD	0022Dh	Quad D/A Ch. D voltage
	0022Eh, 0022Fh	Not Used

IMEAS0	00230h	Loop 0 measured current
IMEAS1	00231h	Loop 1 measured current
MSAC0M	00232h	Loop 0 accumulated meas. current (MS)
MSAC0L	00233h	Loop 0 accumulated meas. current (LS)
MSAC1M	00234h	Loop 1 accumulated meas. current (MS)
MSAC1L	00235h	Loop 1 accumulated meas. current (LS)
MSFLG0	00236h	Loop 0 current bias measurement flag
MSFLG1	00237h	Loop 1 current bias measurement flag
KBIAS0	00238h	Loop 0 current bias filter gain
KBIAS1	00239h	Loop 1 current bias filter gain
IBIAS0	0023Ah	Loop 0 current bias measurement
IBIAS1	0023Bh	Loop 1 current bias measurement
IBIASC0	0023Ch	Loop 0 corrected current bias
IBIASC1	0023Dh	Loop 1 corrected current bias
	0023Eh, 0023Fh	Not Used
UIMTRC0	00350h	Loop 0 commanded current
	00351h - 0035Fh	Loop 0 pulse width constants
UIMTRC1	003D0h	Loop 1 commanded current
	003D1h - 003DFh	Loop 1 pulse width constants

As mentioned previously, external RAM is used for tables and arrays. <u>External RAM does not extend presently past address 03FFFh</u>. The following is a listing of the external RAM utilized for system operation (only the starting address of each table or array is given):

<u>Mnemonic</u>	<u>Address</u>	<u>Description</u>
A2DTBL0	03D20h	GADIC 0 A/D calibration constants
A2DTBL1	03DC0h	GADIC 1 A/D calibration constants
DACNST	03E60h	D/A constants (bias & scale factors)
ADSTRG	03E70h	A/D measurement storage array
SIGTBL	03F10h	A/D measurement pointer array
DACTBL	03F20h	D/A command storage array
ARRAY0	03F30h	Array collection storage 0
ARRAY1	03F58h	Array collection stoage 1
SERIAL	03F80h - 03FFFh	Serial communications arrays

3.2.4 DDSC Serial Communications

The DDSC has been designed with the capability of communicating in a serial format across a twisted pair of wires (or twin axial cable for electrically noisy enviroments) up to two hundred (200) feet in length or less. This serial link transfers data bi-directionally between the Master (serial bus controller) and the Slaves (individual DDSC's) at a clock rate of five (5) megahertz. The serial communications link provides the means to download application specific program and data memory files into the DDSC. The capability to monitor parameters within the DDSC, even during program execution, is also available through the serial communications link.

In addition to providing the throughput necessary for a high performance control network, this highly structured serial communications protocol was designed to reduce the amount of overhead required from the User's application program executing within the DDSC. Table 3-15 lists the program memory addresses associated with the serial communications for the DDSC.

3.2.4.1 Applications Program and Timing Considerations

The DDSC has embedded software which initializes the serial communications and places the DDSC into a "clear" state. As long as the DDSC is in this state, the serial communications will operate automatically. However, if the applications program is executed, the program must provide for servicing the serial communications in order for the DDSC to operate properly. Note that the serial data buffer must be filled each time before pausing for the DDSC serial address to be received. The subroutine to fill the serial buffer (**SERfilbfr**) must be called by the applications program before branching to the program synchronization code. This subroutine is typically called at the end of the program, after all control algorithms have been performed. To synchronize with the pulse width interrupts, the variable **LPCNT** (Data Page = 4, addr = 018h) should be monitored until the desired number of interrupts have completed. **LPCNT** is incremented at the end of each pulse width interrupt (both Loop 0 and Loop 1) and cleared to zero (0) at the start of each communications cycle. Since the interrupts occur every 97.6 uS for each loop (10.240 kHz) and the serial communications operates at a 512 Hz rate, twenty (20) interrupts per loop must occur between the start of each communications cycle (total of forty (40) interrupts (028h)). The amount of time necessary for the serial buffer filling routine to complete spans four (4) interrupts, two (2) per loop. The segment of TMS320X25 code to synchronize the serial buffer routine with the pulse width interrupts is as follows:

```
          LDPK 004h      ; data page pointer = 4

LOOP      LAC  LPCNT     ; load p.w. iteration counter
          SUBK 024h      ; all but 4 interrupts
          BGEZ SVCSER    ; jump if proper value

          IDLE           ; wait for next interrupt
          B    LOOP      ; check again

SVCSER    CALL SERfilbfr ; service serial buffer
```

Program Memory Addresses

SERgetsnc	001A2h	Start of serial communications (synchronization point).
SERnxtsnc	004B9h	End of program for "Clear" mode communications.
SERfilbfr	00529h	Serial communications buffer filling routine.

Table 3-15 DDSC Serial Communications Addresses

BFRPT0	00208h (Data Page 4, Address 08h)	Array 0 pointer buffer.
BFRPT1	00209h (Data Page 4, Address 09h)	Array 1 pointer buffer.
ADDR0	0020Ah (Data Page 4, Address 0Ah)	Array 0 collection address.
ADDR1	0020Bh (Data Page 4, Address 0Bh)	Array 1 collection address.
ARRAY0	03F30h (External RAM - 40 words)	Array 0 storage.
ARRAY1	03F58h (External RAM - 40 words)	Array 1 storage.

Table 3-16 DDSC Array Collection Variable Addresses

If the pulse width interrupts are not utilized by the applications program, the serial buffer routine should be called before branching to the start of serial communications (**SERgetsnc**). After filling the serial buffer, the program must branch to the start of serial communications **SERgetsnc**, which recognizes the DDSC's address and commences transmission of the data in the serial buffer. The array collection pointers (**BFRPT0** and **BFRPT1**) must also be reinitialized before pausing for synchronization of the program. Array collection is discussed in more detail in the following section.

All programs, whether in the "clear" or "operate" mode must branch to the serial address recognition / program synchronization routine (**SERgetsnc**). This section of software monitors the serial input register, checking if the word received from the serial interface matches the Slave synchronization word. The sync word contains the address for a particular Slave; the Slave commences serial communications upon recognizing its address from the Master.

3.2.4.2 Array Collection Variables

The capability to collect two (2) words of data during each Loop 1 pulse width interrupt cycle has been provided within the DDSC. This allows monitoring of program variables at a rate greater than 10 KHz. The addresses for the variables to be monitored are entered into specific data memory locations to indicate the requested data. The software associated with the pulse width interrupts then utilizes these addresses to read the specified locations and store the data into arrays for later collection or monitoring. The size of each of the two (2) arrays is now set at 40 words each, but the size of the arrays may be altered if necessary, although the size of each array must be as large or larger than the number of pulse width interrupts to occur during execution of the applications program. Refer to Table 3-16 for the addresses of the array collection variables. The array collection variables are initialized to the following values upon reset or power up initialization of the DDSC:

ADDR0 = **PWTIM0** (067h [Data Page 0, Address 067h])
ADDR1 = **PWTIM1** (078h [Data Page 0, Address 078h]).

To select a different variable for the array collection feature to operate on, enter the address for that variable into one of the above memory locations (**ADDR0** or **ADDR1**). Note that the array collection pointers (**BFRPT0** and **BFRPT1**) must be reloaded with the addresses of the storage arrays (**ARRAY0** and **ARRAY1**) each sample period before filling the serial data buffer.

3.2.5 DDSC Predefined Subroutines (EPROM)

Several subroutines have been developed for use with the DDSC. These routines are located within the EPROM of the TMS320E25 DSP and may be called from an applications program where needed.

3.2.5.1 DELAY

This subroutine provides a variable time delay function. The contents of auxiliary register AR0 are used to control a delay loop, with the total time determined by the following formula:

$$\text{delay} = (3 * [\text{AR0}]) + 9$$

where [AR0] represents the contents of register AR0. This subroutine has a minimum delay time ([AR0] = 0) of nine (9) machine cycles (each machine cycle requires 100 nS to complete). The maximum delay time is approximately 19.7 mS ([AR0] = 0FFFFh). Note that the auxiliary register pointer is altered when executing this subroutine. The address in program memory for **DELAY** is provided above in section 3.2.3.2.1. Input conditions are only that AR0 contain the delay count desired. There are no output conditions or return values.

3.2.5.2 FIXDAT

FIXDAT performs the masking and sign correction function for A/D measurements. No other conditioning is provided by this subroutine. Input conditions are that the raw A/D data to be masked be placed in the least significant word of the accumulator in the TMS320X25. The corrected data is returned in the same location (LS accumulator, with an MSB value of 115.2 volts). The time required to execute this subroutine is eleven (11) machine cycles (1.1 uS), which includes both the call to the subroutine and the return at the end of the routine. The address in program memory for **FIXDAT** is listed in section 3.2.3.2.1.

3.2.5.3 LOADDAC

This subroutine generates the properly formatted (magnitude and reference information) data to write to the D/A converter, and stores this data into the D/A storage array in external RAM. This data is then written to the D/A converter circuitry from the pulse width interrupt service routine. The D/A constants (scale factors and biases) are used to provide more accurate outputs of the D/A.

The input conditions for this routine are that the D/A command voltages be established in **dadat0** and **dadat1** (MSB = 2.5 volts) prior to executing **LOADDAC**. The reference selection for the D/A must also be entered into auxiliary register AR7 before calling this subroutine. Addresses for the D/A input variables are:

dadat0	:	00228h (Page 4, addr 028h)
dadat1	:	00229h (Page 4, addr 029h)

The D/A constants are located in external RAM. Each channel of the D/A has a set of these constants, which consist of a positive scale factor, a negative scale factor, and an offset or bias. The organization of these constants in memory is POSSF, NEGSF, BIAS for each channel's data. The address of the D/A constant table is:

DACNST : 03E60h.

The scale factors are initialized to 03FFFh (which represents a value of 1) and the biases to 00000h at power up or reset. These constants may be altered at any time via **DEMON** or by loading a data file which contains the D/A constants.

Execution time for **LOADDAC** is 127 machine cycles (12.7 uS), including the call and return from the subroutine. The address in program memory for this subroutine is listed in section 3.2.3.2.1.

3.2.6 Utilizing the GADIC

To perform control functions, analog data must be converted to digital and provided to the processor of the DDSC. This is accomplished through use of the General Analog Digital Interface Circuit (GADIC), two (2) of which are provided on the DDSC. Collection of analog data via the GADIC requires that the signal to be converted be correctly routed to the input of the A/D. The path of the signal is determined by selecting a particular input multiplexer channel and setting, if required, signal gain. A typical analog data collection operation would require the following sequence:

1) **Selection of multiplexer input.**
2) **Selection of gain (if needed).**
3) **Initiation of conversion.**
4) **Delay for conversion of data.**
5) **Restoration of multiplexer setting.**
6) **Restoration of gain (if needed).**
7) **Reading in converted data.**
8) **Conditioning of measurement.**

These steps should be implemented in the order shown to prevent invalid GADIC configurations. Although the multiplexer control register is written at different times from the conversion request, the changing of the multiplexer inputs does not occur until the actual conversion request is executed. The gain selection switches are altered when the gain value is written into the gain selection register. The steps for restoring the multiplexer and gain settings are required for programs which use the data collection options found in the DDSC's pulse width calculations. These system hardware selections will be described in each subsection as necessary. The steps associated with altering the gain may be deleted if there is no change from the system gain selection.

The output of the GADIC's A/D converter consists of nine (9) bits of data plus sign, and is in sign-magnitude format. The magnitude data bits are connected to the system data bus through D8 - D0. The sign bit is received through data bit D15. Because the maximum magnitude of the input voltage for the A/D is 3.6 volts **(-3.6 v to + 3.6 v)**, the most significant bit of the A/D data (bit D8) has a value of 1.8 volts (this results in an MSB value of 115.2 volts for a sixteen (16) bit data word with the A/D measurement in the 9 LSBs). Data bit 15 represents the sign of the measured data (0 = positive data, 1 = negative data). The remaining bits of the data word (bits D14 - D9) will contain random bits and should be masked off or cleared. Conversion time for the A/D varies with the type of function being performed. A general A/D conversion of the GADIC requires approximately 13.2 uS to complete.

The base addresses for each of the two (2) GADIC's on the DDSC, along with wait state information, are:

GADIC0 (Loop 0)	**0D000h**	**1 Wait State**
GADIC1 (Loop 1)	**0D400h**	**1 Wait State.**

The addresses of the individual registers within the GADIC will be presented as an offset to be added to the base address of the requested device. Table 3-17 lists all of the registers within the GADIC and presents their respective address offsets. This total address (base + offset) is used to actually access the specific register within the I.C. Wait states are used to allow sufficient time for the bus interface between the TMS320X25 and the GADIC. **Each wait state requires one machine cycle of the processor, approximately 100 nS (1e-7 second).** Wait states are executed for both read and write operations.

3.2.6.1 Analog Multiplexers

The GADIC has the capability of measuring many analog inputs, in either a single-ended mode or a differential mode. A remote sense capability and test voltages (for calibration purposes) are also available. The input multiplexers are altered through software, providing complete control of the data collection process. Each GADIC has two (2), eight (8) input multiplexers on the analog inputs (although only twelve (12) of the possible sixteen (16) inputs actually have external pin connections (on the GADIC), with the remaining four (4) inputs providing internal

Description	Mnemonic	Address Offset
WRITE REGISTERS		
Multiplexer Configuration Register	**MXCNF**	000h
Gain Control Register	**GAIN**	001h
Discrete Output Register	**DOUT**	003h
Special Configuration Register	**SPCNF**	004h
Pulse Width Mode Register	**PWMOD**	005h
Pulse Width Time Register	**PWTIM**	006h
Pulse Width Register Clear	**PWCLR**	008h
Command Current Sign Register	**SIGN**	00Ah
Commutation Register	**COMM**	00Bh
A/D Configuration Register	**CONF**	00Dh
READ ADDRESSES		
Internal Ground	**IGND**	000h
Analog Input 1	**INP1**	001h
Analog Input 2	**INP2**	002h
Analog Input 3	**INP3**	003h
Analog Input 4	**INP4**	004h
Analog Input 5	**INP5**	005h
Analog Input 6	**INP6**	006h
Test Voltages	**TEST**	007h
Fast A/D Conversion	**FREAD**	008h
Cross Connect Conversion	**XCONN**	009h
Common Mode Conversion (Negative)	**CMNEG**	00Ah
Common Mode Conversion (Positive)	**CMPOS**	00Bh
Discrete Input Register	**DIN**	00Ch

Table 3-17 GADIC Registers and Address Offsets

reference and calibration inputs). The outputs of these multiplexers become the positive and negative inputs of a differential amplifier. The output of this amplifier passes through the gain circuitry on to the A/D converter.

The User has complete control over the setting of the multiplexers through software, determining if the signal to be converted by the A/D is a differential input (an input into both multiplexers) or single-ended input (a single input into either multiplexer, usually referenced to ground). Table 3-18 defines the analog inputs available on the Loop 0 GADIC, their address offsets, and other pertinent information. Table 3-19 provides similar information for the Loop 1 GADIC. These address offsets apply to both the positive and negative input multiplexers (i.e., Analog Input 4 has a positive input and a negative input). The test voltages are a different value for each multiplexer and will be discussed in a later subsection.

Selection of a particular multiplexer input requires the modification of the Multiplexer Configuration Register (MXCNF), to specify the action of each multiplexer when a conversion is requested. The User has the option of making a new input selection for one (1) multiplexer while holding the other multiplexer input at its present setting. This scenario applies to both positive and negative multiplexers. Another option is to make new input selections for both multiplexers simultaneously. Selection of an embedded test voltage through the negative multiplexer channel is also available through MXCNF. The remote sense option utilizes the positive multiplexer channel for external signal references. The address offset of the multiplexer configuration register is:

MXCNF : 000h.

Modification of MXCNF requires four (4) bits of data to select a particular multiplexer setting. The location of these bits and the result of the different combinations are listed in Table 3-20. Note that the bits which control MXCNF are in bit positions D7 - D4. The four (4) LSB's of the data word have no effect on the multiplexer setting.

3.2.6.1.1 Differential Measurements

The sequence for measuring differential inputs is identical to the typical sequence presented in Section 3.2.6. This is due to both multiplexers changing simultaneously to the desired channels. To configure the GADIC for conversion of a differential input (input from each multiplexer), MXCNF must be written with 00Xh to provide the proper control of the multiplexers. The gain selection register should also be written before initiating the

GADIC 0 (U20) Input Signals and Address Assignments
for the Robotic DDSC

Pin Name	Pin Number	Designated Use	Current Assignment	Signal Origin	Read Address
-	-	Internal Ground	Internal Ground	AGND	D000
Pos1	37	Motor Sense 0 S.E.	Motor Current Sense Loop 0	Q10-S	D001[1]
Neg1	40	Diff. Ang.(Resolver)	DAC Output for Gripper Offset	U63-1	D001
Pos2	36	Sine 0 (Enc.)	(Available for assignment)	J1-9	D002
Neg2	41	Enc. 0 Return	(Available for assignment)	J1-18	D002
Pos3	35	Sine 1 (Enc. 1)	(Available for assignment)	J2-9	D003
Neg3	42	Enc. 1 Return	(Available for assignment)	J2-18	D003
Pos4	34	AP04 Aux Input	Gripper Pot. Voltage	J1-3	D004
Neg4	43	Board I.D. S.E.	Board I.D. Voltage	J1-1	D004
Pos5	33	H.G. Amp 2 In +	Gripper Twist Force S.G. In +	J1-8	D005
Neg5	45	H.G. Amp 1 In +	Gripper Long Finger S.G. In +	J1-5	D005
Pos6	32	H.G. Diff Amp 1 Out	Gripper Long Finger Force S.G.	U17-1	D006
Neg6	46	H.G. Diff Amp 2 Out	Gripper Twist Force S.G.	U17-7	D006
		Discrete Inputs			
In0	16	Input 0	(Available for assignment)	J1-10	D00C
In1	14	Compare 0 Output	Compare 0 is not assigned	J1-11,12	D00C

Abbreviations:
S.E. = Single Ended, Diff. = Difference or Differential, Ang.= Angle, Enc.= Angle Encoder
Aux = Auxiliary, I.D. = Identification, H.G. = High Gain, Amp = Amplfier
DAC = Digital/Analog converter, Pot = Potentiometer, S.G. = Strain Guage

[1] Setting of the GADIC "Mux Configuration Register" determines whether an analog multiplexer input is converted as a single ended (either positive or negative) or differential (differential of positive and negative) input. Both conversion types are read from the same address. (See Software Discussion for further detail.)

GADIC 1 (U26) Input Signals and Address Assignments
for the Robotic DDSC

Pin Name	Pin Number	Designated Use	Current Assignment	Signal Origin	Read Address
		A/D Inputs			
-	-	Internal Ground	Internal Ground	AGND	D400
Pos1	37	Motor Sense 1 S.E.	Motor Current Sense Loop 1	Q11-S	D401[1]
Neg1	40	Diff. Ang.(Resolver)	DAC Output for Use as Required	U63-1	D401
Pos2	36	Cosine 0 (Encoder)	(Available for assignment)	J2-9	D402
Neg2	41	Enc. 0 Return	(Available for assignment)	J2-18	D402
Pos3	35	Cosine 1 (Enc. 1)	(Available for assignment)	J1-9	D403
Neg3	42	Enc. 1 Return	(Available for assignment)	J1-18	D403
Pos4	34	AP04 Aux Input	Gripper Pot. Voltage	J2-3	D404
Neg4	43	AN04 Aux Input	(Available for assignment)	J2-1	D404
Pos5	33	H.G. Amp 2 In +	Gripper Lateral Force S.G.In +	J2-8	D405
Neg5	45	H.G. Amp 1 In +	Gripper Short Finger S.G. In +	J2-5	D405
Pos6	32	H.G. Diff Amp 1 Out	Gripper Short Finger Force S.G	U23-1	D406
Neg6	46	H.G. Diff Amp 2 Out	Gripper Lateral Force S.G.	U23-7	D406
		Discrete Inputs			
In0	16	Input 0	(Available for assignment)	J2-10	D40C
In1	14	Compare 1 Output	Compare 1 is not assigned	J2-11,12	D40C

Abreviations:
S.E. = Single Ended, Diff. = difference or differential, Ang = Angle
Enc. = Angle encoder, Aux = Auxiliary, H.G. = High Gain, Amp = Amplifier
DAC = Digital/Analog Converter, Pot = Potentiometer, S.G. = Strain Guage

[1] Setting of the GADIC "Mux Configuration Register" determines whether an analog multiplexer input is converted as a single ended (either positive or negative) or differential (differential of positive and negative) input. Both conversion types are read from the same address. (See Software Discussion for further detail.)

Data Bits				
D7	D6	D5	D4	"Set on Read" Mode
0	0	0	0	Set both multiplexers (differential)
0	1	0	0	Hold Positive ; Set Negative (S.E.)
1	0	0	0	Set Positive ; Hold Negative (S.E.)
1	1	0	0	Hold Positive ; Hold Negative
X	0	0	1	Set Positive ; Test Voltage 3
X	1	0	1	Hold Positive ; Test Voltage 3
0	X	1	0	Remote Sense ; Set Negative
1	X	1	0	Remote Sense ; Hold Negative
X	X	1	1	Remote Sense ; Test Voltage 3

X = Don't Care

S.E. = Single Ended

Table 3-20 GADIC Analog Input Multiplexer Control

Data Bits			
D6	D5	D4	GAIN VALUE
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

Table 3-21 GADIC Programmable Gain Control

conversion, if needed. After selecting the proper multiplexer control and gain, the conversion request is issued for the desired channel. An example of the code (TMS320X25) to initiate and read a differential input (Analog Input 6) is:

```
         LARP     0                    ; point to register 0
         LRLK     AR0,(GADIC+MXCNF)    ; mux conf. register addr
         LACK     000h                 ; select diff. inputs
         SACL     *                    ; store to GADIC
#        LRLK     AR0,(GADIC+GAIN)     ; gain sel. register addr
#        LACK     000h                 ; set gain = 1
#        SACL     *                    ; store to GADIC
         LRLK     AR0,(GADIC+INP6)     ; select input channel 6
         LAC      *                    ; Start conversion

         LARK     AR0,02Eh             ; set up 15 uS delay
         CALL     DELAY                ; time delay routine

         LARP     0                    ; point to register 0
         LRLK     AR0,(GADIC+MXCNF)    ; mux conf. register addr
         LACK     000h                 ; RESTORE SYSTEM MUX
         SACL     *                    ; store to GADIC
#        LRLK     AR0,(GADIC+GAIN)     ; gain sel. register addr
#        LACK     000h                 ; RESTORE SYSTEM GAIN
#        SACL     *                    ; store to GADIC
         LRLK     AR0,(GADIC+INP3)     ; RESTORE SYSTEM INPUT
         LAC      *                    ; load converted data
         SACL     DATA                 ; save data temporarily

     # = Optional
```

Note that modification of MXCNF is not necessary if the multiplexers are already configured, merely the request for start of conversion. However, the multiplexer control should be written to insure proper signal paths to the A/D converter. The system input selections and the system gain selection will be discussed later in this section. The time delay routine used was discussed in the section concerning software features of the DDSC (section 3.2.5.1).

3.2.6.1.2 Single-Ended Measurements

Initiation of a single-ended conversion (input from one (1) multiplexer, usually referenced to ground) is a bit more complex than a differential conversion. Because a single-ended conversion requires inputs from two (2) different multiplexer inputs, the multiplexer control register should be written first with a 00xh, to select the differential mode. After stepping the multiplexers to the desired reference channel (both multiplexers are altered), The multiplexer control register should be written with the desired command (step positive, hold negative; step negative, hold positive). If necessary, the gain selection should be written at this time. The final step is to initiate a conversion by requesting the multiplexer input channel to be measured. An example of the code to perform a single-ended conversion (positive Analog Input 5, referenced to internal ground) is:

```
          LARP     0                    ; point to register 0
          LRLK     AR0,(GADIC+MXCNF)    ; mux conf. register addr
          LACK     000h                 ; select diff. inputs
          SACL     *                    ; store to GADIC
+         LRLK     AR0,(GADIC+IGND)     ; internal ground address
          LAC      *                    ; select reference input
          LRLK     AR0,(GADIC+MXCNF)    ; mux conf. register addr
          LACK     080h                 ; step pos.; hold neg.
          SACL     *                    ; store to GADIC
#         LRLK     AR0,(GADIC+GAIN)     ; gain sel. register addr
#         LACK     000h                 ; set gain = 1
#         SACL     *                    ; store to GADIC
          LRLK     AR0,(GADIC+INP5)     ; select input channel 5
          LAC      *                    ; start conversion

          LARK     AR0,02Eh             ; set up 15 uS delay
          CALL     DELAY                ; time delay routine

          LARP     0                    ; point to register 0
          LRLK     AR0,(GADIC+MXCNF)    ; mux conf. register addr
          LACK     000h                 ; RESTORE SYSTEM MUX
          SACL     *                    ; store to GADIC
#         LRLK     AR0,(GADIC+GAIN)     ; gain sel. register addr
#         LACK     000h                 ; RESTORE SYSTEM GAIN
#         SACL     *                    ; store to GADIC
          LRLK     AR0,(GADIC+INP3)     ; RESTORE SYSTEM INPUT
          LAC      *                    ; load converted data
          SACL     DATA                 ; save data temporarily
```

+ = For this example, internal ground was used as the reference for the conversion. <u>Any</u> of the analog inputs can be used as the reference for a single ended conversion.

# = Optional.

The "dummy" read to select the reference input (internal ground, in this example) initiates a conversion of the A/D converter. However, the subsequent read of the signal in question (Analog Input 5) overrides the previous request and restarts the A/D with a new conversion. As mentioned in the previous example, modification of MXCNF is not required if the multiplexers are already set up. Note that the system selections should be restored following the conversion, should these settings be required.

3.2.6.1.3 System Multiplexer Settings

The system hardware settings are associated with the high speed data collection performed within the pulse width calculations. These settings define the analog input channels to be converted, as well as the gain to be applied to these signals. Actual values of these settings change, depending on which axis of pulse width calculations is pending execution. **These system settings are presented with the assumption that both axes of pulse width calculations are being used.**

At the start of execution of the User's applications program (the end of serial communications), the registers of the GADIC are configured with the default system settings. **The default system settings are for differential multiplexers, with Analog Input 3 as the analog channel selected.** These are the values to be restored if the GADIC multiplexer control register is altered, and the pulse width calculations are being used. If the pulse width calculations are not being utilized, there are no default system hardware settings, only the above listed initial conditions.

These default system settings are sent to the GADICs at the start of each communications cycle. The default multiplexer settings are found in the following variables:

mxcnf0	**:**	**00222h (Page 4, addr 022h)**
mxcnf1	**:**	**00223h (Page 4, addr 023h)**

The values found in these locations should not be altered unless the desired system settings change. When restoring the multiplexer setting after an A/D read, the contents of these locations should be written to the GADICs as follows:

```
        LDPK      004h                ; data page pointer = 4
        LARP      0                   ; point to register 0
        LRLK      AR0,(GADIC0+MXCNF)  ; load address pointer
        LAC       MXCNF0              ; load mux selection
        SACL      *,4,0               ; store data to GADIC
```

Note that the data is shifted left four (4) places when written to the GADIC. This is due to the fact that the significant data bits for the multiplexer configuration register are located in data bits D7-D4, with data bits D3-D0 not relevant.

3.2.6.2 Programmable Gain

The capability to vary the gain of the signal being routed to the A/D has been designed into the GADIC. The gain control register (GAIN) allows the User to specify one (1) of eight (8) gain selections, ranging from 1 to 128. The address offset for the gain control register is:

GAIN : 001h.

Alteration of the analog gain requires a three (3) bit data word be written into the gain control register. The location of these data bits and the resulting gain for each value is listed in Table 3-21. Note that the bits which control GAIN are in bit positions D6 - D4. The four (4) LSB's of the data word have no effect on the gain selection setting. **The default system gain setting is 2.** The default system gain selections are found in the variables:

adgan0 : 00220h (Page 4, addr 020h)
adgan1 : 00221h (Page 4, addr 021h).

These settings are transmitted to the GADICs in the same manner as the default multiplexer control selections described above. The following is an example of the TMS320X25 code to alter the gain control register:

```
          LARP      0                   ; point to register 0
          LRLK      AR0,(GADIC+GAIN)    ; gain sel. register addr
          LACK      000h                ; set gain = 1
          SACL      *                   ; store to GADIC
```

Note that the gain selection switches change when the gain data is written into the gain control register. If the gain is set too high, the voltage into the A/D converter will saturate at full scale (3.6 volts) and not yield useful information.

3.2.6.3 Calibration Options

The GADIC has several features inherent in its design to allow calibration of common mode, amplifier biases, etc., under software control. Although these options are not used frequently, they do provide the means to calibrate the analog data measurements.

3.2.6.3.1 Fast Convert

The fast convert feature allows the User to bypass the multiplexer and gain alteration steps and start an immediate conversion of the analog input currently selected with the present gain. Execution of a fast conversion requires 6.6 uS to complete. The address offset for initiating a fast A/D conversion is:

FREAD : 008h.

The following code illustrates how to perform a fast conversion of A/D data with the DDSC, using the present multiplexer and gain settings:

```
        LARP      0                      ; point to register 0
        LRLK      AR0,(GADIC+FREAD)      ; fast convert address
        LAC       *                      ; start conversion

        LARK      AR0,013h               ; set up 6.6 uS delay
        CALL      DELAY                  ; time delay routine

        LARP      0                      ; point to register 0
        LRLK      AR0,(GADIC+FREAD)      ; fast convert address
        LAC       *                      ; load converted data
        SACL      DATA                   ; save temporarily
```

The fast conversion feature does not alter the present multiplexer settings or the gain selection. The fast convert option is used, along with the default system hardware settings, for the high speed data collections from the pulse width calculations. This feature should not be used unless the multiplexer is not to be changed during the program.

3.2.6.3.2 Common Mode

The capability to measure the common mode voltage for both the positive and negative input multiplexers has been provided. This allows the User to compensate data for more accurate system measurements. When a common mode conversion is requested, the output of one (1) of the two (2) multiplexers is routed to both inputs of the differential amplifier. This signal then passes through the gain circuitry and is converted by the A/D. The address offsets for the common mode voltage measurements are:

CMNEG : 00Ah
CMPOS : 00Bh.

The following code is an example of the steps to execute a common mode measurement (positive channel):

```
          LARP      0                   ; point to register 0
          LRLK      AR0,(GADIC+CMPOS)   ; pos. common mode addr
          LAC       *                   ; start conversion

          LARK      AR0,02Eh            ; set up 15 uS delay
          CALL      DELAY               ; time delay routine

          LARP      0                   ; point to register 0
          LRLK      AR0,(GADIC+CMPOS)   ; pos. common mode addr
          LAC       *                   ; load converted data
          SACL      DATA                ; save data temporarily
```

Note that by using the common mode address for both starting the conversion and reading in the converted data, the multiplexer settings were not modified. The common mode conversion requires approximately 13.2 uS to complete.

3.2.6.3.3 Cross Connect

The capability to cross-connect, or "swap", the inputs to the differential amplifier has been designed into the GADIC. When a cross connect conversion is requested, the multiplexer outputs are reversed before entering the differential amplifier. This feature allows any amplifier biases due to signal polarity to be calibrated out. The address offset for initiating a cross connect conversion is:

XCONN : 009h.

The following TMS320X25 code is an example of the steps to execute a cross connect data conversion:

```
          LARP      0                   ; point to register 0
          LRLK      AR0,(GADIC+XCONN)   ; cross connect read addr
          LAC       *                   ; start conversion

          LARK      AR0,02Eh            ; set up 15 uS delay
          CALL      DELAY               ; time delay routine

          LARP      0                   ; point to register 0
          LRLK      AR0,(GADIC+XCONN)   ; cross connect read addr
          LAC       *                   ; load converted data
          SACL      DATA                ; save data temporarily
```

Note that the cross connect conversion does not alter the present gain or multiplexer settings. The cross connect conversion requires approximately 13.2 uS to complete.

3.2.6.3.4 Test Voltages

The GADIC was designed with various embedded test voltages for calibration purposes. By using these test voltages and the calibration conversions discussed previously, biases and scale factors for the A/D converter may be calculated to compensate actual data measurements.

Two (2) of these test voltages have already been mentioned earlier. These voltages are accessed through the input multiplexers (one (1) test voltage per multiplexer - channel 7). The test voltage from the positive multiplexer is -3.2 volts; the negative multiplexer test voltage is -3.15 volts. Both of these test voltages can be measured the same as any other multiplexer input (see "Differential Measurements" and "Single-Ended Measurements").

The third test voltage embedded within the GADIC is a -0.45 volt reference. This voltage is selected through the multiplexer control register (MXCNF) by setting bit D4 to a "1". When this bit is set, the -0.45 volt test voltage is routed to the negative input of the differential amplifier.

3.2.6.4 Timing Considerations

In order to maintain proper operation of the DDSC, there are certain timing restrictions that must be observed when making analog measurements. These timing conditions are associated with the pulse width calculations and the high speed data collection routines within the pulse width interrupts. Regularly spaced interrupts are used to initiate the pulse width calculations (approximately 10 kHz for each loop). For this discussion, it is assumed that both channels of pulse width calculations are being executed. If the pulse width calculations are not being used, there are no timing constraints on making analog measurements other than those imposed by the serial communications.

3.2.6.4.1 Predefined A/D Reads

As mentioned in the subsection concerning the input multiplexers, the DDSC has a predefined "system input setting". To reiterate, these settings are: **differential multiplexers reading Analog Input 3, gain = 2, for both GADICs**. The analog channel being read will alternate between Analog Input 3 (for Loop 0) and

Analog Input 2 (for Loop 1) depending on the next channel of pulse width calculations to execute. These predefined measurements are taken in every pulse width interrupt. **It is very important that the system settings be restored following any alterations.**

3.2.6.4.2 General A/D Reads

The DDSC has two (2) interrupt routines of pulse width calculations that must be executed. The start of these two routines or "loops" are separated in time by 36.6 uS, with Loop 0 occurring 36.6 uS before Loop 1. This was implemented to reduce the load on the system power supply by not allowing both channels of pulse width switches to "turn on" simultaneously. This technique also helps in reducing noise in the high speed collection measurements by sampling the "off" axis' data (i.e., Loop 0 interrupt routine measures Loop 1 variables, and vice versa).

Since each loop's pulse width calculation requires approximately 22 uS to complete, the time available between the end of the Loop 0 interrupt and the start of the Loop 1 interrupt is about 14 uS. This time is insufficient for setting up and completing a general A/D read of the GADIC. The time between the end of the Loop 1 interrupt and the start of the next Loop 0 interrupt is approximately 37 uS. This is adequate time to perform analog measurements. Therefore, **any general read of the GADIC's A/D converter must occur between the Loop 1 and Loop 0 pulse width interrupts**. To facilitate synchronizing the User's application program with the pulse width interrupts, a counter variable has been placed within the pulse width interrupt routines. This variable, **LPCNT**, is initialized to 0 at the start of the communications cycle for the DDSC. The interrupt counter also provides the means to determine and set program execution time. The counter is incremented after each interrupt (i.e., after the first Loop 0 interrupt, LPCNT = 1; after the first Loop 1 interrupt, LPCNT = 2; after the second Loop 0 interrupt, LPCNT = 3, etc.). **LPCNT has the value of 6 when the serial communications are completed (the start of the applications program).** The address of this pulse width iteration counter variable is :

LPCNT 0218h (Page 4, addr 018h).

By inserting a testing loop in the User's application program to check the value of this counter, the User can reliably execute an analog measurement between the pulse width interrupts. **A general A/D read must be synchronized with the pulse width interrupts for the high speed data collections to function properly.** The following TMS320X25 code is an example of this synchronization loop:

```
          LDPK      04h                 ; data page pointer = 4

SYNC
          LAC       LPCNT               ; load counter variable
          SUBK      010h                ; check for interrupt 16
          BGEZ      CONT                ; jump to continue

          IDLE                          ; wait for next interrupt
          B         SYNC                ; check again

CONT
          NOP                           ; A/D read here
```

This particular example is waiting until after the eighth (8th) iteration of the Loop 1 pulse width interrupt routine. Since general A/D conversions can occur only between the Loop 1 and Loop 0 interrupts, note that LPCNT must always be an even value when synchronizing for an A/D read. Remember that after the general A/D conversion has been completed, the system default hardware settings must be restored.

3.2.6.5 Utilizing A/D Measurements in the DDSC

The A/D measurements gathered from the GADIC may be used directly or conditioned to provide a greater degree of accuracy. To use the measurements directly, the measured data needs only to be converted from sign-magnitude format to 2's compliment format. This is accomplished by executing the following segment of TMS320X25 code after each A/D read:

```
          LAC       DATA                ; recall A/D data
          BGEZ      POSDAT              ; jump if positive data

          ANDK      001FFh              ; mask for 9 bits of data
          NEG                           ; change sign of data
          B         CNT00               ; continue

POSDAT
          ANDK      001FFh              ; mask for 9 bits of data
          B         CNT00               ; time equalization

CNT00
          SACL      MDATA               ; save masked data
```

This method is the quickest means of using the A/D measurements, requiring only ten (10) machine cycles to execute.

However, this method does not provide the most accurate measurement data. To achieve the greatest accuracy, a set of tables containing A/D scale factors and biases is available to condition the measurements gathered by the DDSC. Each individual GADIC has a separate table for each value of A/D gain, from 1 through 16. These constants are generated by a special calibration routine and stored in memory. This data is then saved to a User specified file for later use.

The following is an example of the TMS320X25 code to correct a general A/D measurement using these constants. This sequence requires an additional 34 machine cycles to complete. The segment of code for masking the data presented above must be executed prior to implementing this method:

```
          LAC     MDATA          ; recall masked data
          BLZ     NEGDAT         ; jump if negative data

          LACK    000h
          SACL    TMP2           ; save positive offset
          B       CNT01          ; continue

NEGDAT
          LACK    008h           ; offset for neg data
          SACL    TMP2           ; save negative offset
          NOP                    ; time equalization
          NOP                    ; time equalization

CNT01
          LAC     MDATA,5        ; 115.2 v -> 3.6 v conv.
          ABS                    ; get magnitude of data
          SACH    TMP0,5         ; save table offset data
          ANDK    007E0h         ; mask for 6 LSB's
          SACL    TMP1           ; save LSB data for later

          LACK    007h           ; mask for 3 bits
          AND     TMP0           ; get table offset
          ADD     TMP2           ; + offset for sign
          SFL                    ; * 2 for table
          SACL    TMP2           ; save temporarily

          LAC     ADGAN,5        ; gain value for offset
          ADD     TMP2           ; + table offset
          ADLK    A2DTBL         ; + table base address
          SACL    TMP2           ; save pointer temp.

          LARP    0              ; point to register 0
          LAR     AR0,TMP2       ; load pointer

          LT      TMP1           ; load 6 LSB data
          MPY     *+             ; * scale factor
```

```
PAC                              ; product to accumulator
SFL                              ; * 2 for scale factor
ADDH      *                      ; + bias
SACH      CDATA                  ; save conditioned data
                                 ; MSB = 3.6 volts

LAC       CDATA,11               ; 3.6 v -> 115.2 v conv.
SACH      CDATA                  ; save conditioned data
                                 ; MSB = 115.2 volts
```

The MSB value for the scale factors in the table is 0.5. After multiplying the measured data by the scale factor, the result must be multiplied by 2 (value of the digital scale factor is approximately 0.5, so that real scale factor values of greater than 1 may be used). The MSB value of the bias data is 3.6 volts. Note that time equalization may be required for applications that are time sensitive. Storage for the A/D constant tables is reserved in the external RAM of the DDSC at the following addresses:

A2DTBL0	:	**03D20h**
A2DTBL1	:	**03DC0h**

where A2DTBL0 contains the constants associated with GADIC 0, and A2DTBL1 contains the constants for GADIC 1. There are sixteen (16) scale factors and sixteen (16) biases for each value of A/D gain. These values span the range of the A/D signals (-3.6 v to +3.6 v).

3.2.7 Special Signal Inputs

The Special Signal Inputs, through software, provide the means to collect data at a high rate and are not dedicated to a particular type of feedback device.

3.2.7.1 Analog Measurements

Each loop (axis) of the DDSC has two (2) Special Signal analog inputs associated with it (total of four (4)). Each channel is fed into a separate GADIC (Loop 0 utilizes analog input 2 of both GADICs; Loop 1 uses analog input 3), so that the A/D conversions can be executed as close to simultaneous as possible (actual offset between conversions of Special Signal inputs is approximately 300 nS). Note that the Special Signal measurements depend on the system multiplexer and gain settings described previously.

Conversion and collection of the Special Signal data occurs during the pulse width interrupt routines. At the start of each interrupt service routine, both of the DDSC's GADICs are commanded to execute a Fast Conversion. The default system hardware settings have defined the inputs and gains to use for these conversions.

While the conversion is taking place, the interrupt routine is performing the control calculations for the pulse width amplifier. After the Special Signal conversions are completed, the data is read from each GADIC, masked to clear the unused bits and converted from sign magnitude format to 2's compliment binary data. The data is then stored into an array for later use by the applications program. The address for this storage array is:

ADSTRG : 03E70h.

The data is stored in this array in the following manner: L0M0, L0M1, L1M0, L1M1, L0M0, L0M1, etc., where L refers to the Loop number of the pulse width interrupt and M refers to the measurement being taken in that interrupt (two (2) measurements per interrupt routine).

3.2.7.2 Special Discrete

The Special Signal Inputs have a discrete input for each loop of the DDSC. This digital signal is intended to be connected to the index or reference discrete of a feedback system. Status of the Special Signal discrete input can be checked by reading the encoder interface (discussed in a later section) and determining the state of data bit D15.

3.2.7.3 Special Signal Offset Adjustment

Offset adjustment has been supplied for the Special Signal 0 analog input for each loop in the form of an eight (8) bit D/A output (actually, the D/A contains four (4) individual D/As, with two (2) of the channels providing common mode offset for the GADICs and the remaining channels used for Special Signal Offset adjust). The output of the D/A is connected into a resistor network to provide minute steps in voltage to compensate the Special Signal input for biases. The address for this D/A converter is:

QDAC : 04800h

with the individual D/A addresses following in successive order. The data variables that contain the D/A commands are:

qdaca	:	0022Ah (Page 4, addr 02Ah)
qdacb	:	0022Bh (Page 4, addr 02Bh)
qdacc	:	0022Ch (Page 4, addr 02Ch)
qdacd	:	0022Dh (Page 4, addr 02Dh)

The data in **qdaca** and **qdacb** are used for Special Signal offset adjust for Loop 0 and 1, respectively. The contents of **qdacc** and **qdacd** are for adjusting the GADIC common mode bias. Data for these D/A variables has an MSB value of 320 volts. Upon power up or reset, these variables are initialized to a value of 2.5 volts (080h).

3.2.7.4 Comparator Input / Output

As mentioned previously, a "spare" or unused differential comparator is available for each loop of the DDSC. The digital outputs of these "spare" comparators are read into the processor through the discrete inputs on the GADICs. **The Loop 0 spare comparator output is read in from GADIC0 discrete input on data bit D1, while Loop 1 spare comparator output is read in from GADIC1 discrete input on data bit D1.** The voltage range for these comparator inputs are -5 to +5 volts.

3.2.8 Specialized Interfaces (EPLD)

Navtrol has developed special interfaces that allow the DDSC to adapt to numerous feedback devices. These interfaces are implemented within an Erasable Programmable Logic Device (EPLD) on the DDSC. Table 3-22 provides a brief description of the functions available within the EPLD. By configuring the DDSC at the start of execution of the program, an optical incremental encoder interface, a commutation based encoder interface or a resolver based encoder interface may be accessed. Each loop of the DDSC may be configured independently for any of the interfaces mentioned.

To initialize the DDSC and its encoder interfaces, the Encoder Interface configuration must be written. The data bits used and their purposes are as follows:

D0 - Sign of SINE (0 = normal, 1 = inverted)
D1 - Sign of COSINE (0 = normal, 1 = inverted)
D2 - Sign of INDEX (0 = normal, 1 = inverted)
D3 - Select (0 = Abs. Encoder, 1 = Resolver)
D4 - Abs. Encoder Select (0 = Loop 0, 1 = Loop 1)
D5 - Abs. Encoder Convert Select (0 = convert).

In addition to selecting between an absolute encoder or a resolver - type interface, data bit D3 also controls the excitation output for the resolver (0 = excitation off, 1 = excitation on). This feature allows the User to turn off the excitation when not needed, thus saving power. The inversion selection of the Special Signal states determines whether the signals are passed straight through to the encoder interface or if a 1's compliment of the input signals is to be performed. The addresses of the Encoder Interface configuration registers are:

ECONF0	**:**	**0C800h**
ECONF1	**:**	**0CC00h.**

ROBOTIC CONTROLLER
EPLD DECODE ADDRESSES & DESCRIPTIONS

Addr Used	Address Don't Care Bits	"READ" Functions	Data Bus Control	Active Data Bits
C000	0 - 9, 13	Read Commutation Transition Counter 0	EPLD	0-7, (8-15)*
C400	0 - 9, 13	Read Commutation Transition Counter 1	EPLD	0-7, (8-15)*
C800	0 - 9, 13	Read Commutation Time Counter 0	EPLD	0-9,(10-15)*
CC00	0 - 9, 13	Read Commutation Time Counter 1	EPLD	0-9,(10-15)*
D000	4 - 9, 13	Read GADIC 0 (addresses 0 - F)	GADIC0	0-8,15,(9-14)*
D400	4 - 9, 13	Read GADIC 1 (addresses 0 - F)	GADIC1	0-8,15,(9-14)*
D800	0 - 9, 13	Read Discrete Inputs	Discretes	0 - 15
DC00	0 - 9, 13	NOT A VALID READ ADDRESS	NA	NA
4000	0 - 9, 13	Read Commutation Inputs, Loop 0	EPLD	0-3, (4-15)*
4400	0 - 9, 13	Read Commutation Inputs, Loop 1	EPLD	0-3, (4-15)*
4800	0 - 9, 13	NOT A VALID READ ADDRESS	NA	NA
4C00	0 - 9, 13	End of Receive (Serial Control Sig.)	NA	NA
5000	0 - 9, 13	NOT A VALID READ ADDRESS	NA	NA
5400	0 - 9, 13	NOT A VALID READ ADDRESS	NA	NA
5800	0 - 9, 13	NOT A VALID READ ADDRESS	NA	NA
5C00	0 - 9, 13	Reset of Serial Controller	NA	NA

Addr Used	Address Don't Care Bits	"WRITE" Functions	Data Bus Control	Active Data Bits
C000	0 - 9, 13	Set Mode of EPLD Controller	CPU	0-1, (2-15)*
C800	0 - 9, 13	Reset (Load) Pulse Width Clock Gen.	CPU	(0 - 15)*
CC00	0 - 9, 13	Serial Control - Read Enable	CPU	(0 - 15)*

NOTE: (xx - yy)* indicates data bits which are "DON'T CARES" for writes or invalid for reads.

These configuration registers are used to independently set up each loop of the DDSC for a particular interface. The following is an example of the TMS320X25 code to implement this configuration (Loop 0 for an optical incremental encoder, non-inverted signals):

```
        LARP       0                     ; point to register 0
        LRLK       AR0,ECONF0            ; conf reg. address
        LACK       000h                  ; select configuration
        SACL       *                     ; store to register
```

Upon power up or reset, **the DDSC is initialized to interface with a commutation based encoder.** This configuration may be altered from the applications program, if necessary.

3.2.8.1 Optical Incremental Encoder

The DDSC's optical incremental encoder interface allows the DDSC to measure the angular position of a motor / axis to a great degree of accuracy. The Special Signal Inputs are utilized by this interface to provide position feedback information. The sine and cosine outputs of the encoder are measured and used to increase measurement resolution. The Special Signal discrete input should be connected to the index or reference output of the encoder.

The encoder sine and cosine signals are connected to the Special Signal Inputs, which are then routed to the Digital Interface IC (EPLD). Circuitry within the EPLD determines when a "line" of the encoder has been crossed and updates a counter in the appropriate direction. **This line counter is cleared each time it is read.** These counters are accessed through the Encoder Interface. The addresses, etc., of this interface are:

ENC0 (Loop 0)	:	0C000h
ENC1 (Loop 1)	:	0C400h.

The data word retrieved from the Encoder Interface contains information other than the counter value. A breakdown of the data bits of this interface word and their use is:

D15	Encoder Index (1 = reference present)
D14 - D13	Quadrature Information
D12 - D11	Discrete Inputs
D07 - D00	Counter Value (+/- 127).

The encoder index (D15) is the state of the Special Signal Discrete input since the last access of the encoder interface. If an index pulse had occurred since the last read of the encoder interface, it would have been latched into the EPLD and read into the processor. This discrete signal would also clear the line counter value to zero (0). The quadrature information (D14 - D13) is used to correct the counter value to correspond with the analog measurements of the sine and cosine. The two (2) bits of discrete inputs may be utilized for general purposes applications. The line counter is a signed, 2's compliment up/down counter. Data bit D7 determines the sign of the counter data, which depends on the direction of travel of the encoder. The remaining bits (D10 - D08) do not contain valid data and should be ignored.

3.2.8.2 Commutation Feedback Encoder

A software selectable option to use the commutation feedback from a brushless DC motor to derive position and velocity measurements has been developed and implemented in the interface EPLD. This encoder option provides a count of the commutation state transitions over one sample period. The position of the motor will be proportional to the count value (Each pole pair has six (6) transitions (changes) of the commutation sequence; total transitions for one (1) revolution of the motor equals the number of magnet pole pairs times six (6)). A timer count value indicating the time since the last transition is also available. With these inputs, the average velocity of the motor can be determined, along with the change in position. By using integration and extrapolation techniques, a fairly accurate position and velocity measurement are produced to provide feedback for the control algorithms. These measurements are not as smooth as the optical incremental encoder measurements, but should prove more than adequate for systems which contain a gear train or other speed reducer / torque multiplier. The addresses, etc., of the counters and timers are:

XCOUNT0 (Loop 0 Transition Counter)	**:**	**0C000h**
XCOUNT1 (Loop 1 Transition Counter)	**:**	**0C400h**

TCOUNT0 (Loop 0 Timer)	**:**	**0C800h**
TCOUNT1 (Loop 1 Timer)	**:**	**0CC00h.**

The transition counters are eight (8) bit, bi-directional counters. The count value is a 2's compliment number found in data bits D7 - D0, with D7 providing the sign information. The timer registers are ten (10) bit counters, with the time count found in data bits D9 - D0. This time value is always positive. **Both counters (transition and time) are cleared when the time register is read.** **Both registers should be accessed as close together as possible to prevent extra counts occurring, with the transition counter being read first.** The Special Signal inputs have no affect on this encoder interface configuration.

3.2.8.3 Resolver Interface

An interface for resolvers has been developed for the DDSC. This interface requires the use of the D/A converter and the Special Signal Inputs to produce a difference angle to be measured by the GADIC. The excitation signal for the resolver outputs is generated by the DDSC, with the same excitation signal used for both loops. The sine and cosine outputs from the resolver are fed into the Special Signal Inputs. These signals then pass through the D/A converter and are combined to create a difference angle. The difference angle produced is then routed into both GADICs via the negative 1 input channel. <u>Note that when the resolver interface is used, the D/A outputs are not available to the User.</u>

To generate the difference angle, The expected motor angle is calculated by adding the resolver bias angle (RVBIAS) to the predicted motor angle (w/ new control) and the result multiplied by the resolver "speed" (or number of pole pairs in the resolver). The sine and cosine of this expected motor angle are calculated, scaled (using the scale factors and biases stored in **DACNST**) and saved to data memory. **The sign of the cosine information must be inverted to accommodate the circuitry of the DDSC.** This D/A data must contain the proper reference selection (either Loop 0 or Loop 1), as well as the sign information of each signal. To select the reference for the resolver interface, one of the following values must be appended to the D/A data:

DARFS0 (Loop 0) : 00000h
DARFS1 (Loop 1) : 04000h.

The sign selection data must also be appended to the D/A data for proper operation. If the sign of the D/A data is positive, then no mask need be appended to the data. The masks to select a negative sign for each channel of the D/A are:

DACANG (Ch. A) : 01000h
DACBNG (Ch. B) : 02000h.

All of these masks must be appended to <u>both</u> channels of D/A data to correctly utilize the D/A converter. After the D/A data has been scaled and formatted, it must be stored into the D/A signal table (**DACTBL**). The data from this table is then written to the D/A circuitry by the pulse width interrupt routines. Data within the D/A signal table is ordered as follows (sixteen (16) data words):

L0S, L0C, L0S, L0C, L0S, L0C, L1S, L1C,
L1S, L1C, L1S, L1C, L1S, L1C, L0S, L0C

where **L** refers to the Loop number (Loop 0 or 1), **S** refers to the sine of the predicted angle, and **C** refers to the cosine of the predicted angle. The code within the pulse width interrupts determines which elements of the table to write to the D/A circuitry (two (2) elements are written per interrupts routine).

The difference angle input (NEG Ch. 1, both GADICs) is read from the pulse width interrupt routines and the data masked and sign corrected. The result is then stored into the A/D storage array (ADSTRG) in external RAM. Data is stored in this array in the following format:

L0G0P0, L0G1P0, L0G0P1, L0G1P1, L0G0N0, L0G1N0, L0G0N1, L0G1N1,
L1G0P0, L1G1P0, L1G0P1, L1G1P1, L1G0N0, L1G1N0, L1G0N1, L1G1N1,

and repeated four (4) more times for a total of eighty (80) data points per sample period (**L** refers to Loop number, **G** refers to the GADIC number, and **P** or **N** refers to either a positive or negative data point). Note that L0G0P0 and L0G1P0 are actually the same data point, with the exception that they are measured with different GADICs. To determine to difference angle (for Loop 0, for example), all of the negative data points for the loop in question from GADIC 0 are summed up and the result is subtracted from the sum of the similar positive data points from the same GADIC (this is a total of twenty (20) positive and twenty (20) negative samples). This result is then averaged and the A/D data is conditioned using the calibration constants described in Section 3.2.6.5. After multiplying by a conversion factor and dividing by the resolver "speed", the result is corrected for any gain in the A/D and the result saved temporarily. After performing the same tasks for the GADIC 1 measurements, the two (2) conditioned sums are then added together and averaged to produce the difference angle (DIFANG). Note that the resolver constants do not have specific addresses assigned to them; selection of data storage is left to the User.

3.2.8.4 High Gain Differential Inputs

There are a total of four (4) differentially amplified analog inputs on the DDSC, two (2) per Loop. These inputs can be used to monitor and measure signals from such feedback devices as strain gages, pressure transducers, etc. This signal is then read as a single-ended input (referenced to internal ground) through the A/D. For each Loop, these signals are read into the GADIC on positive and negative input channels 6.

3.2.9 Power Outputs (Motor Control)

The DDSC has the ability to drive two (2) DC brush-type or brushless motors is provided on the board. The brushless motors may have up to three (3) phases (delta or wye configuration). These motors can have multiple magnetic pole pairs, up to a total of 42. The width of the output pulses for the PWM amplifier are computed by the DDSC's processor during the pulse width interrupts. Parameters defining the motor being controlled, such as motor scale factor, motor resistance, motor inductance, etc., are entered through Navtrol's servo development program **DEMON**, and the appropriate gains and constants for the pulse width amplifiers are generated and transferred to the DDSC (remember to save off the gains and constants to the working data file for future use).

3.2.9.1 Command Current and Sign

The command current for the DDSC is to be calculated in the User's applications program. This value is then used by the pulse width interrupt to control the current through the motor. This command current is set up as a magnitude value; that is, it is always a positive value. Addresses, MSB values, etc. for the command current for both DDSC loops are listed in Table 3-23. Note that the MSB value for the current can be altered by replacing several resistors in the current sense circuitry. It is not mandatory that both Loops have the same MSB value for the pulse width current. However, the MSB value of the command current must match the MSB value for current within the pulse width calculations. **Upon reset or power up initialization and while the DDSC is in the "clear" mode, the command current is set to 0 for both loops.**

The sign of the command current is determined by the applications program and written to the GADIC pulse width sign register. The address offset for this register is:

SIGN : 00Ah.

Data bit 0 (D0) is the only bit necessary to configure this register. **For positive current commands, D0 is set to "1"; for negative current commands, D0 is cleared to "0".** The command current sign should be updated at the same time that the new command current is determined. The following is an example of the TMS320X25 code to convert the commanded system current to a magnitude, as well as determining and transmitting the sign of the current. **Note that the command current passed to the pulse width algorithms is always the positive magnitude of the actual command.**

```
        ZALH    UIMTRC              ; signed command cur.
        BLZ     NEGCUR              ; jump if negative

        SACH    UIMTRC              ; save positive cur.
        LACK    001h                ; set current positive
        SACL    SIGN
        B       CURCNT              ; continue
```

Mnemonic	Address	MSB Value
UIMTRC0	00350h (Page 6, Addr 050h)	0.516129 amps
UIMTRC1	003D0h (Page 7, Addr 050h)	4.129032 amps

Table 3-23 DDSC Current Command Variables

Mnemonic	Address	MSB Value
Last Measurements		
IMEAS0	00230h (Page 4, Addr 030h)	2.064516 amps
IMEAS1	00231h (Page 4, Addr 031h)	4.129032 amps
Accumulated Measurements		
MSACC0	MSW = 00232h, LSW = 00233h (Page 4, Addr 032h and 033h)	66.064512 amps
MSACC1	MSW = 00234h, LSW = 00235h (Page 4, Addr 034h and 035h)	132.129024 amps

Table 3-24 DDSC Current Measurement Variables

Mnemonic	Address	MSB Value
IBIAS0	0023Ah (Page 4, Addr 03Ah)	2.064516 amps *
IBIAS1	0023Bh (Page 4, Addr 03Bh)	4.129032 amps *
IBIASC0	0023Ch (Page 4, addr 03Ch)	2.064516 amps *
IBIASC1	0023Dh (Page 4, addr 03Dh)	4.129032 amps *
MSFLG0	00236h (Page 4, Addr 036h)	-
MSFLG1	00237h (Page 4, Addr 037h)	-

[MSFLGX = 0000 0000 0000 0000 b => Normal Operation]
[MSFLGX = XXXX XXXX XXXX XX11 b => Bias Measurement]

KBIAS0	00238h (Page 4, Addr 038h)	0.5
KBIAS1	00239h (Page 4, Addr 039h)	0.5

* - MSB Value and range are determined by selectable resistor values.

Table 3-25 DDSC Current Bias Variables

```
NEGCUR
          ABS                                ; get magnitude of data
          SACH       UIMTRC                  ; save current
          LACK       000h                    ; current negative
          SACL       SIGN
          NOP                                ; time equalization

CURCNT
          LARP       0                       ; point to register 0
          LRLK       AR0,(GADIC+SIGN)        ; register address
          ZALH       SIGN                    ; load sign data
          SACH       *                       ; store to GADIC
```

The time equalization was added so that each path through this seection of code (positive data or negative data) required the same amount of time to execute. This is important for applications that require very strict timing sequences.

3.2.9.2 Pulse Width Time and Mode

The time for the switches of the pulse width amplifier to "turn on" is calculated by the pulse width interrupt software. These routines measure the current presently in the motor windings, and calculate the pulse width time necessary to generate the correct current. The pulse width interrupt also determines if the current is increasing or decreasing, and sets a flag in the GADIC to indicate this information. The pulse width calculations are part of the embedded "operating system" code found in the DDSC's EPROM. The address offsets for the pulse width mode and time registers within the GADIC are:

PWMOD	**:**	**005h**
PWTIM	**:**	**006h.**

These GADIC registers (mode and time) should not be modified by the applications program; the pulse width calculations generate the data to be written into these registers.

3.2.9.3 Motor Current and Bias Measurement

The pulse width interrupt for each loop measures the current being supplied to the motor windings to provide closed loop control. The data retrieved from the GADIC for the current measurement is corrected for any bias that may be present in the current measurement. These measurements are saved into the DDSC's system memory as an "instantaneous" measurement and accumulated for averaging. The accumulated measurements may be averaged in the applications program to allow for smoother monitoring of the motor current. Table 3-24 lists the addresses, MSB values, etc., of the motor current measurements.

The DDSC also has the capability to measure the current bias for each Loop. Table 3-25 provides address and MSB value information for the current bias variables. Special flags have been incorporated with the pulse width calculations to generate a current bias measurement utilizing the current measurement taken in each pulse width interrupt. **To measure the current bias for a particular Loop, the measurement flag for that loop should be set to a non-zero value,** with data bits D1 and D0 set to 1's. If the flag is set to a non-zero value with data bits D1 and D0 not set to 1's, the code for measuring current bias is bypassed. **Upon power up or reset, these bias measurement flags are configured to allow calculation of the current bias value.**

After the measurement to be used for calculating the current bias is made, the measurement is passed through a filter to reduce the noise on the bias. The formula for this filter is:

IBIAS = (IBIAS * KBIAS) + ((1 - KBIAS) * IMEAS)

where IBIAS is the current bias and IMEAS represents the latest current measurement. This filter is implemented for both Loops of the DDSC. The filter constant, **KBIAS, is initialized upon power up or reset to a value of 0.99** (with MSB = 0.5, hex value of KBIAS is 07EB8h). The pulse width calculations utilize the variables IBIASC0 and IBIASC1 for correcting the measurements. The applications program should place the value for IBIAS0 and IBIAS1 into these locations, or correct the measured biases (IBIAS0, IBIAS1) to compensate for additional offsets.

3.2.9.4 Motor Commutation Control

Commutation of a motor connected to the DDSC is controlled by the GADICs on the DDSC. Logic within the GADIC takes into account the pulse width time, pulse width mode, and the commutation signals from the motor. The pulse width interrupts contain segments of code to read the commutation signal interface located within the EPLD and write this information to the commutation register in the GADIC. The addresses of these commutation interfaces (for each Loop) are:

COMMUT0	**:**	**04000h**
COMMUT1	**:**	**04400h.**

The address offset for the commutation register within the GADIC is listed in Table 3-17. The data read into the DDSC from the commutation interface is located in data bits D3 - D0. This data is organized as:

XXXX XXXX XXXX ABC0 b.

The A,B and C bits correspond with the commutation signals from the motor. Note that bit 0 must be zero for the commutation signals to be properly utilized in the GADIC. The applications code must read the commutation interface and store the data directly to the commutation register in the GADIC. The same bit positions listed above for the commutation interface are valid for the commutation register within the GADIC. Table 3-26 lists the valid commutation selections and their corresponding motor drive switch selection. Note that if a brush type motor is used and does not require electronic commutation, the commutation inputs for that axis must be hard-wired to indicate which step of the commutation sequence to use.

3.2.9.5 Timing Considerations

There are no critical timing constraints on the use of the pulse width amplifiers and their associated interrupts. The applications program must allow the pulse width interrupts to continue execution in order for the motor current control to function properly. A "dead-man" timer is included in each GADIC so that if the pulse width interrupt is not serviced regularly, the pulse width amplifier will shut down and not allow any current through the motor windings.

Commutation Bits			Switch Selection
A	B	C	
1	0	1	Upper Switch 1, Lower Switch 2
1	0	0	Upper Switch 1, Lower Switch 3
1	1	0	Upper Switch 2, Lower Switch 3
0	1	0	Upper Switch 2, Lower Switch 1
0	1	1	Upper Switch 3, Lower Switch 1
0	0	1	Upper Switch 3, Lower Switch 2

These selections are based on the assumption that the commanded current is positive and increasing.

Table 3-26 Commutation Sequence

The synchronization code used for making A/D measurements (section 3.2.6.4) may also be used to determine if the appropriate number of pulse width interrupts have occurred, indicating the end of the applications program. Note that the execution time for the DDSC program <u>must</u> be the same or less than the communications program executing within the Master.

The command current must be set up from the applications program, although there is no time restriction on this event. If the current is not updated each sample period, the pulse width interrupts will use the last entered current as the command. It is recommended that the current command be established at the same time each sample period to allow for smoother system operation.

3.2.10 Analog Output (D/A)

The DDSC contains two (2) independent channels of digital to analog (D/A) output. This variable voltage output can be used to control such mechanisms as galvanometers or servo valves, or may be used as a means to pass system information from the DDSC to external display devices, such as meters, bar graphs, etc. **The output amplifiers of the D/A (TL084 op amp) have the capability of sinking or sourcing up to 15 mA of current for each D/A channel.**

3.2.10.1 General Discussion

The D/A converters designed onto the DDSC have a range of -5 to +5 volts, with twelve (12) bits resolution. <u>The data to be sent to the D/A must be in sign-magnitude format</u>. The data representing the magnitude of the signal resides in data bits D11 - D0. The sign information is used to select the reference voltage for the D/A. As the range for the magnitude of the output is 5 volts and the D/A requires twelve (12) bits of data, **the MSB value for the D/A data (value of bit D14)** is 20 volts (value of bit D11 (MSB of actual data to be written to the D/A) is 2.5 volts). The addresses for the two (2) channels of D/A output and wait state information are:

DACA	**0DC00h**	**2 Wait States**
DACB	**0DC01h**	**2 Wait States.**

Note that the D/A circuitry requires an additional wait state that the other DDSC peripherals do not need. This is due to the slow access time for the D/A converter utilized. Each D/A will continue to output its specified voltage until the command is changed. A subroutine has been developed to generate the command data to be written to the D/A. This subroutine, **LOADDAC**, is embedded in the EPROM of the TMS320E25 DSP and may be called from an applications program. A further discussion of this subroutine is found in

Section 3.2.5.3.

3.2.10.2 Reference Selection

In order to provide bipolar voltage outputs for the D/A converters, the reference voltage for each D/A is independently switched between +5 and -5 volts. However, the outputs of the D/A's become the inverting inputs of the output amplifiers (TL084 type op amp). This brings about another inversion of the D/A signal. **Therefore, for positive output voltages from the D/A, the negative reference should be selected. Similarly, for a negative D/A output voltage, the positive reference should be designated.**

Selection of the D/A reference voltage is coincident with the writing of the D/A command voltage. An analog multiplexer and a set of analog switches determine the voltage used for the D/A reference. These components are part of the resolver measurement circuitry and will be discussed later. The reference selection data is written into a latch, the outputs of which are connected to the analog switches, with the output of the switches selecting either +5 volts or -5 volts as the reference for each D/A. The latch outputs also select a voltage on the analog multiplexer to serve as the source for the D/A reference signals. To select a voltage for the reference source, the analog multiplexer must be addressed properly. The bit positions and the results of their various combinations are:

ANMX0	**:**	**D14**	**ANMX1**	**:**	**D15**

D15	D14	Reference Source
0	0	Sine 0, Cosine 0
0	1	Sine 1, Cosine 1
1	0	+5 volts
1	1	0 volts

When a "1" is written to the reference selection bits of the latch, the positive reference voltage (+5 volts) is selected. A "0" selects the negative D/A reference (-5 volts). Each D/A channel has a separate reference selection bit, which are defined as:

DACARF : D12 (0 = negative, 1 = positive)
DACBRF : D13 (0 = negative, 1 = positive).

When these possible reference selections are considered with the inverting amplifiers on the D/A outputs, a "1" written to the reference latch produces a negative analog output voltage; a "0" generates a positive analog output voltage. An example of the code to determine the reference for and commanding the output for D/A channel B would be:

```
           LAC      DADAT1          ; load data (2's comp)
           BLZ      NEGDAT          ; jump if negative data

           ANDK     0FFFh           ; get 12 LS bits
           ORK      00h             ; set positive ref. DAC B
           SACL     DAOUT1          ; save D/A output data
           B        CONT            ; continue

NEGDAT
           ABS                      ; get positive magnitude
           ANDK     0FFFh           ; get 12 LS bits
           ORK      02000h          ; set negative ref. DAC B
           SACL     DAOUT1          ; save D/A output data
           NOP                      ; time equalization

CONT
           LARP     0               ; point to register 0
           LRLK     AR0,DACB        ; D/A channel B address
           LAC      DAOUT1          ; recall D/A data
           ORK      08000h          ; select +5 V ref. source
           SACL     *               ; store to GADIC
```

After the DDSC has been reset or powered up, the negative D/A references (positive voltage output) are selected for both channels of D/A, with the reference source selected to be +5 V and the D/A command voltage = 0 volts. Note that the reference source and sign information for both channels of the D/A are written at the same time as the D/A command. Hence, reference source and sign data must be appended to both D/A magnitude commands for proper operation.

The pulse width interrupts service routines contain code to read the D/A command data from a table in external RAM and write this information to the D/A interface. This allows the D/A outputs to be altered at a rate greater than 10 KHz, which is necessary for the resolver interface available on the DDSC. A more detailed discussion of this feature is found in the description of the subroutine **LOADDAC**.

3.2.11 Discrete Input / Output

The DDSC has been designed with the ability to monitor multiple discrete inputs. These inputs may be used to monitor such devices as limit switches (for end of travel indication), function selection, etc. The capability to control multiple discrete outputs has also been supplied. These outputs may be used to actuate solenoids, relays, brakes, etc.

3.2.11.1 Discrete Inputs

Each loop or axis of the DDSC has two (2) discrete inputs which are read into the processor through the GADIC. These inputs are usually reserved for monitoring limit switches. One input is used directly for limit switch monitoring. The other GADIC discrete input monitors the output of the spare comparator for each loop. All of these inputs are equivalent to a typical TTL input (0 to +5 volts). These inputs are read every sample period during the serial communications by the DDSC and stored into data memory by the program. The addresses for the limit switch input data in data memory are:

LIMSW0	**00224h**	**(Page 4, Address 024h)**
LIMSW1	**00225h**	**(Page 4, Address 025h).**

The data for the limit switch input is found in the LSB of the data word (D0).

Other Discrete inputs are available (eight (8) per Loop) on the DDSC for reading discrete signals such as enable and mode switches, status signals, etc. These discrete inputs may also be used to interface with an absolute encoder. These inputs are accessed when the Discrete Input Interface is read. The address of this input peripheral and other information is:

DSCRTIN	**0D800h**	**1 Wait State.**

The data that is read from the Discrete Input Interface is stored in the DDSC's memory at:

DSCRTS	**:**	**0021Fh (Page 4, addr 01Fh).**

Please note that the commutation signals for the motors are read in through the interface EPLD, not the Discrete Input Interface.

3.2.11.2 Discrete Outputs

Four (4) channels of discrete output are also provided on the DDSC, each with the capability of sinking 0.5 amps. These discrete outputs are an open drain configuration, with a maximum voltage input of 60 volts. These outputs are controlled through both of the GADICs on the DDSC. A variable has been set aside for each axis of discrete output. These variables and their addresses are:

DOUT0	**00226h**	**(Page 4, Address 026h)**
DOUT1	**00227h**	**(Page 4, Address 027h)**

Each GADIC has three (3) discrete outputs, although output 2 from each GADIC is not available. The usable outputs are accessed through data bits D1 - D0. The offset for the discrete output register in the GADIC is:

DOUT : 003h.

This offset is added to the base address of the GADIC to be written to produce the entire address. **The discrete output variables are initialized to 00h when the DDSC is powered up or reset.** This configuration sets all discrete outputs to a "0" or "off" state.

An example of the TMS320X25 code to turn on a discrete output on would be (turning on bit 1 on GADIC 1):

```
LDPK      004h                  ; data page pointer = 4
LACK      002h                  ; turn on bit 1
OR        DOUT1                 ; discrete output word
SACL      DOUT1                 ; save new discrete data
LARP      0                     ; point to register 0
LRLK      AR0,(GADIC1+DOUT)     ; address of register
SACL      *                     ; store data to GADIC
```

An example of the TMS320X25 software to turn off a discrete output would be (turning off bit 0 on GADIC 0):

```
LDPK      004h                  ; data page pointer = 4
LACK      006h                  ; turn off bit 0
AND       DOUT0                 ; discrete output word
SACL      DOUT0                 ; save new discrete data
LARP      0                     ; point to register 0
LRLK      AR0,(GADIC0+DOUT)     ; address of register
SACL      *                     ; store data to GADIC
```

Note that discrete output variable **DOUT0** is used for GADIC 0 and **DOUT1** is used for GADIC 1.

4. Robotic Control Algorithms

Control algorithms have been developed by Navtrol for control of robotic joints and end effectors, such as grippers and nutrunners. These are discussed in the following sections.

4.1 Gripper Control Algorithms

The algorithms developed for control of robotic grippers are designed to accommodate a variety of feedback devices. Position feedback from angle encoders, resolvers, or potentiometers can be utilized. For NASA GSFC's split-rail type parallel gripper, position feedback from a potentiometer and force feedback, utilizing strain gauges, are used. Both of these feedback techniques are discussed below as is interaction between them. Refer to Figure 4-1 for a block diagram of the gripper control algorithms.

4.1.1 Position Control

The control for opening and closing the gripper is accomplished in the "position" mode. This allows the end effector gripper to be opened or closed to a specific dimension without having an object in its grasp. Software limits are monitored in the DDSC to prevent the gripper from opening or closing too far and "jamming" the leadscrew assembly.

4.1.1.1 Position Feedback

To control the position (opening) of the gripper "fingers", opposing elements used to grip an object, feedback from a potentiometer is utilized. This potentiometer is connected between +5 VDC and -5 VDC to provide maximum sensitivity for measurements. At the time that the control program is first executed, the DDSC directly measures the voltage on the wiper of the potentiometer through each GADIC with a gain of one (1). The data from these measurements are then conditioned and combined. A scale factor and offset are used to convert the measurement from a voltage to a linear position measurement (**LNMEAS**) that can be used by the control calculations. The measured voltage is also adjusted for MSB value and sent to one (1) channel of the D/A converter on the DDSC. The output of the D/A converter is passed through a summing amplifier (gain = 2) and routed to an input on both of the GADICs. (This D/A has multiple uses. When used as part of the resolver I/F, the gain of two is required.)

To provide maximum resolution for the position measurement, a "difference" method is employed within the DDSC. Subsequent iterations of the gripper control program, utilizing the multiplexed inputs of the GADICs, measure the differential voltage between the potentiometer wiper and the output of the D/A converter.

1.

FOLDOUT FRAME

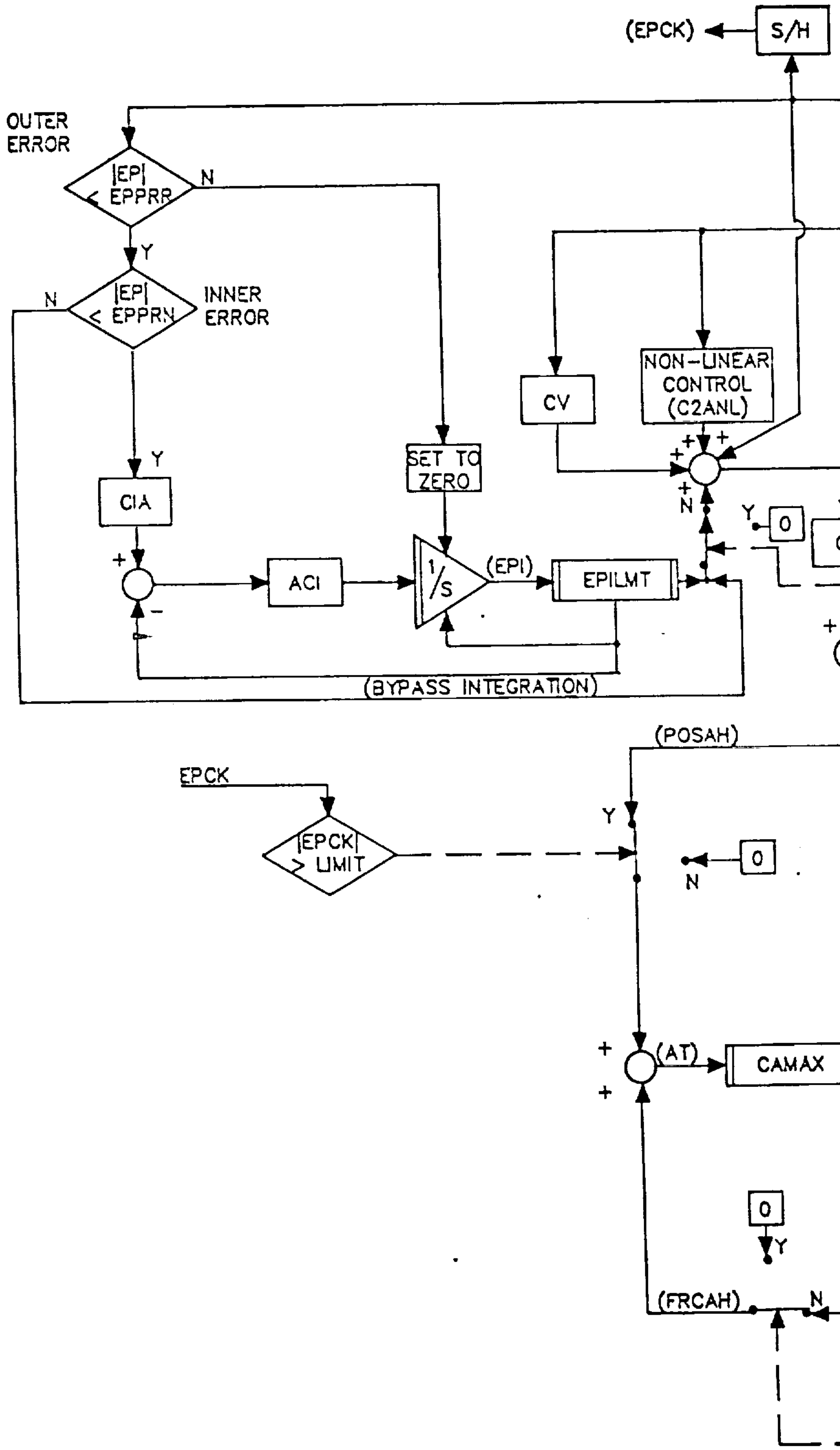
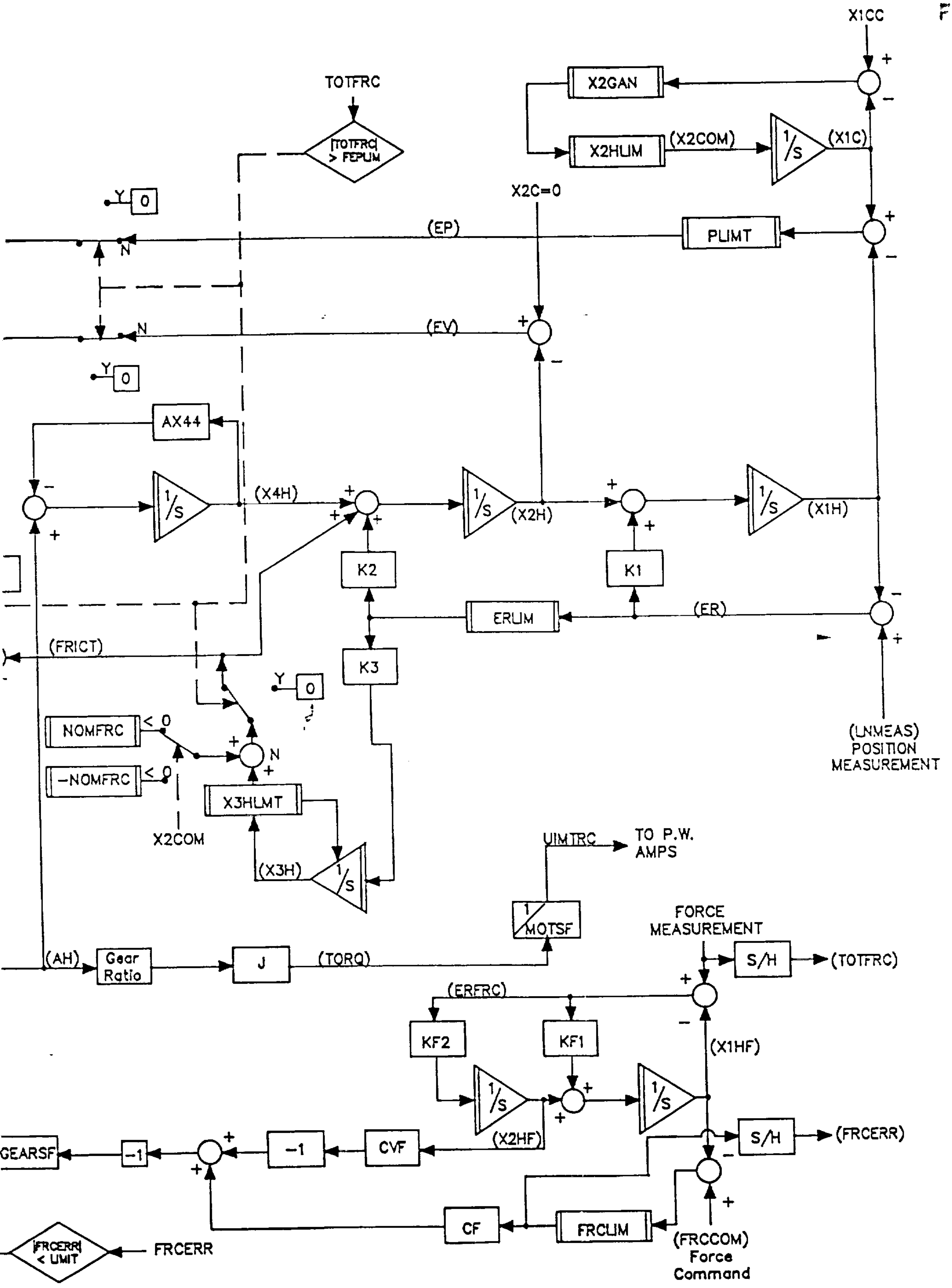


Figure 4–1: NA

2

FOLDOUT FRAME



A GRIPPER CONTROL SYSTEM DIAGRAM

A gain of sixteen (16) is used within the GADIC to Improve resolution of this difference measurement. Basic iteration rate for the control algorithm is 512 per second. However, during each sample period 17 measurements are made, one each pulse width cycle except those that occur during serial communications. In addition these measurements are made by each of the 2 GADICS, providing 34 measurements in all. The 17 measurements from each GADIC are then averaged, to increase accuracy of the measurements, and adjusted for GADIC scale factor and bias. These two averaged measurements, averaged together and added to the D/A command, represents the position (gripper opening) raw measurement. From this the corrected D/A command voltage is derived and sent to the D/A converter to generate a new voltage for the next sample period. Gripper scale factor and offset are then applied to the position measurement for further correction and to create the linear measurement (**LNMEAS**), in inches of opening, for gripper position.

4.1.1.2 Command Generation

To command the gripper to grasp or release an object, a commanded position (**X1CC**) is sent to the DDSC (Commanded position refers to the distance between the two (2) fingers of the gripper.) To grip, or close upon an object, the commanded position must be less than the size of the object to be gripped. This allows the force control algorithms to become active. Conversely, to release or open up from an object, the commanded position must be greater than the size of the object. The command for the position of the gripper is created by calculating a difference between the commanded position (**X1CC**) and the integrated position command (**X1C**) (see Figure 4-1). This difference is then passed through a gain (**X2GAN**) to create a "command" velocity (**X2COM**). Due to the backlash of the planetary gears and the "play" in the leadscrew assembly, velocity control of the gripper is not possible. Therefore, the commanded velocity (**X2C**) passed to the control calculations is always set to zero (0) The velocity derived from the position error is integrated to get a position (**X1C**), which is passed to the control algorithms and also used for the next sample period's command. The sign of the "command" velocity (**X2COM**) is used by the friction estimator described in the next section.

4.1.1.3 Position Control Calculations

After the position command for the gripper has been generated and the measurement of the present position has been made, the control algorithms are executed. These algorithms include the use of modern estimation theory to model the various states of the system, such that the estimated system may be compared to the actual system and corrections made to the model. The first step of these calculations is correcting the states of the system model using Kalman filtering. This technique also allows an estimate of the system velocity (**X2H**) and friction (**X3H**) to be generated from the position feedback. These estimates are used to stabilize the system and provide more precise control. The friction estimate (**FRICT**) is a simple model of the system friction, consisting of the friction state derived from the Kalman filter (**X3H**) and a "nominal" friction (**NOMFRC**) which is summed with the friction state value. The sign of the nominal friction is determined from the sign of the command velocity (**X2COM**), with the friction's sign being opposite that of the velocity (to oppose motion of the system). Following the Kalman filter calculations, prediction of the system states to the end of the next sample period without control applied is performed. This prediction, without control, is then used for the control calculations.

Determination of the various system errors is the next step in controlling the gripper position. Velocity and position errors (**EV**, **EP**) are derived by finding the difference between the command and the prediction for these different states. Integral control (**EPI**) is also performed to provide more accurate control of the gripper position. The limits which define the "dead-band" control (discussed in a moment) are used to determine what effect the integral control has on the control output. The modeled system friction (**FRICT**) is subtracted from the control acceleration, cancelling the friction effects in both the actual and modeled system. These three (3) errors plus friction are then used to create the control acceleration (**POSAH**) for the system.

To prevent the gripper from "hunting" or continuously controlling about the "play" in the mechanical assembly, a "dead-band" type control was implemented. This procedure uses two (2) limits, an "inner" (small) limit and an "outer" (large) limit. When the control program for the gripper is initially executed, the inner limit is set up within the dead-band limit variable for use by the controls. As the gripper moves, the calculated position error (**EPCK**) is constantly monitored, and when the magnitude of this error becomes less than the dead-band limit (the inner limit, in this case), the position derived control acceleration (**POSAH**) for the system is cleared. The dead-band limit variable is also altered, with the outer limit being stored there. The magnitude of the position error (**EPCK**) must become greater than the dead-band limit (outer limit) before the control acceleration is allowed to pass to the system. Once the error exceeds the dead-band limit and the control acceleration is allowed, the inner error is stored to the dead-band limit variable to prepare for the next time the gripper stops its motion.

After, it has been determined whether control acceleration is to be utilized, the control acceleration is summed with the acceleration (**FRCAH**) derived from the force control algorithms, which are discussed in the next section. The total acceleration command (**AT**) is then limited to the maximum system acceleration (**CAMAX**). The resultant acceleration (**AH**) is converted to a command torque (**TORQ**) by multiplying acceleration by system inertia (**J**) and the gear ratio of the gripper assembly. Command torque is then converted to a command current (**UIMTRC**) and the magnitude of this current will be used by the pulse width algorithms to drive the gripper motor. The sign of the command current is sent directly to the GADIC from the control algorithms. Prior to actual use of the new current command, correction of the pulse width constants for back EMF effects is performed using estimated velocity (**X2H**) and the voltage constant of the gripper motor. A final prediction is then performed using the new control acceleration (**AH**) to update the various states of the model.

4.1.2 Force Control

Control of the gripper motor when an object is being held is performed by the force control algorithms. The gripper can exert positive forces on an object (force from the fingers applied toward each other) or negative forces (force from the fingers applied away from each other). Several sets of strain gages were mounted on the gripper, providing the capability to measure the primary gripping force, the lateral or "sideways" force, and the rotary or "twisting" forces on the gripper. Although all of these force measurements are available for use within the DSP, only the measurements from the primary force strain gages are utilized. Estimated states representing gripper force and force rate of change are used to stabilize the control outputs.

4.1.2.1 Force Feedback

Measurement of the forces exerted by the gripper is accomplished through the strain gages mounted on the inside faces of the gripper fingers. The strain gages are connected into bridge circuits that are excited by +5 VDC and -5 VDC. Use of 10 V. across the bridge, compared to 5 V., increases the amplitudes of the force signals, and provides a better S/N ratio. The outputs of these bridge circuits are then fed into high gain (G=50) differential amplifiers on the DDSC. The outputs from these amplifiers are routed to analog inputs of the GADICs. An additional gain of eight is used within the GADIC to increase the resolution of the force measurements (**TOTFRC**). The lateral (**LATFRC**) and rotary (**ROTFRC**) force measurements are performed in the same manner, with the exception that the gain in the GADIC is sixteen (16) rather than eight (8).

The capability to measure the bias offset levels on the force feedback is available. These biases, if requested, are measured prior to the actual control measurements and the results are stored into memory. They are then subtracted from the control measurements to provide a more accurate reading of the forces on the gripper.

4.1.2.2 Force Control Calculations

After measuring the gripper forces, corrections are made to the modeled states of the force control. A force rate (**X2HF**) is determined through use of a Kalman estimator, and this rate is then used to provide stabilization of the force control output. Prediction of the various states without control applied is also performed, as in the gripper position control, and the force error (**FRCERR**) is calculated. This error is then used to generate the force related control acceleration (**FRCAH**) for the gripper motor, which is summed with the position derived acceleration (**POSAH**) to provide the total system acceleration (**AT**). A dead-band type control, very similar to the dead-band described in the position control discussion, is implemented to prevent the gripper from "chattering" due to mechanical play within the gripper. Note that the response of the gripper will differ depending on the object being grasped (i.e., a block of aluminum will have a different response than that of a tennis ball, etc.). Response plots of the gripper on various objects are found in the Results Section (5.0).

4.1.3 Force / Position Control Switching

To prevent the force-derived and position-derived controls from competing with and possibly cancelling each other, software "switches" have been incorporated into the control algorithms for the gripper. These switches are based on the measurement of gripper force (**TOTFRC**) and a User defined lower limit for this force. If the magnitude of the measured force is less than the prescribed limit, the force-derived acceleration (**FRCAH**) is cleared and the control of the gripper is due solely to the position-derived torque. Should the magnitude of the measured force exceed the limit, the position associated errors (position error (**EP**), velocity error (**EV**), and integrated position error (**EPI**), as well as the friction estimate (**FRICT**)) are cleared, thus disabling the position-derived control. These switches are checked each sample period, and appropriate action is taken when necessary.

4.2 Nutrunner Control Algorithms

Control of a nutrunner can also be performed by the DDSC. For this application measurement of motor position and speed was accomplished by monitoring the commutation signals from the motor. Figure 4-2 presents a block diagram of the control algorithms for the nutrunner. Control of the maximum output torque of the nutrunner is also performed by the DDSC to prevent damage due to a stalled motor.

FOLDOUT FRAME

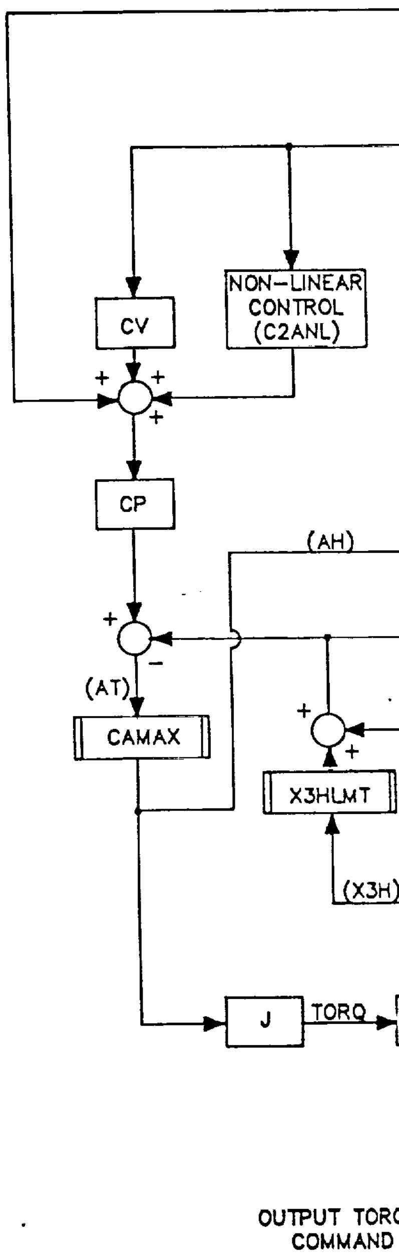
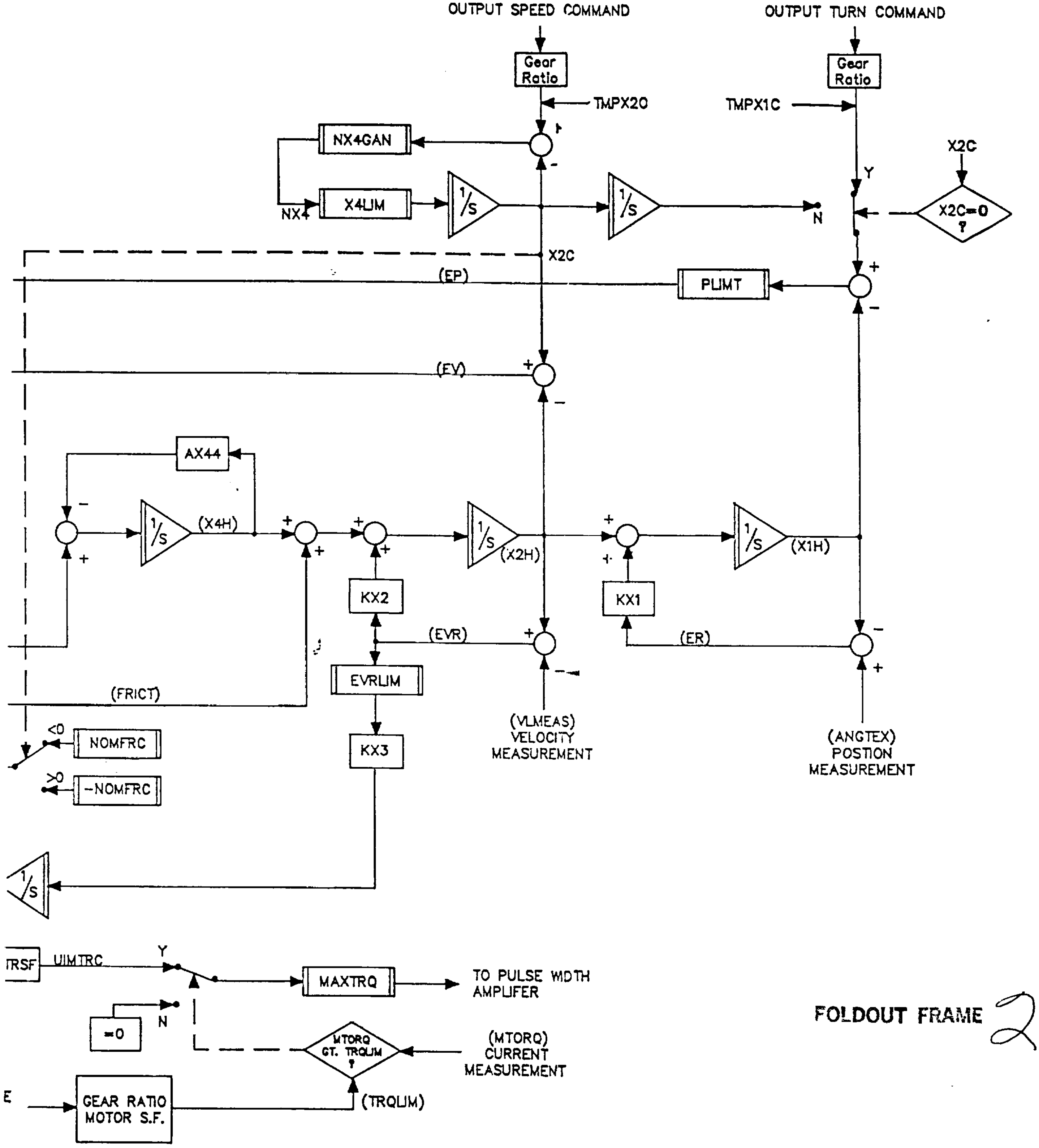


Figure 4



FOLDOUT FRAME 2

·2: NASA'S NUTRUNNER CONTROL SYSTEM DIAGRAM

4.2.1 Nutrunner Command Generation

There are multiple means of commanding the nutrunner through the DDSC. The output shaft of the nutrunner may be commanded to move a specific number of revolutions (position command) or to rotate at a fixed velocity (rate command). Note that all nutrunner commands to the DDSC are referenced to the output shaft of the nutrunner. Therefore, speed commands are entered with the units of "revolutions per minute" (RPM) and position commands use the units "revolutions". The number of turns completed by the nutrunner output shaft is monitored, through use of motor commutation signals, to determine when the command position has been achieved.

Rate or velocity control of the nutrunner is accomplished by converting the output speed command (**SPDCOM**) to a motor velocity command (**X2C**). A temporary velocity command (**TMPX2C**) is obtained by multiplying the output speed command by the gear ratio of the nutrunner and converting units from minutes to seconds (velocity of the nutrunner motor is measured in revolutions per second (rps)). The motor velocity command is then subtracted from the temporary velocity command and the difference is multiplied by a gain (**NX4GAN**) to generate a command acceleration (**NX4**). This acceleration is limited and then integrated to produce the motor velocity command (**X2C**). This produces a "ramp" of the nutrunner velocity command to start the shaft moving smoothly. The slope of the ramp may be controlled by varying the acceleration limit (**X4LIM**). The acceleration is calculated each sample period and integrated until the motor velocity command is equal to the temporary velocity command. If no output velocity command was specified, the nutrunner shaft will rotate at maximum velocity until the number of desired turns has been reached.

Whether an output velocity command (**SPDCOM**) was entered or not, the number of turns for the output shaft to rotate (**TRNCOM**) must be provided to the DDSC. This value is converted to a motor position command (**X1C**) by multiplying the output turn command by the gear ratio of the nutrunner. If there is no commanded velocity, this result is passed to the nutrunner controls directly. If a velocity has been specified, the integrated velocity is used as the motor position command (refer to Figure 5-2). The nutrunner position measurement (**ANGTEX**), whose discussion follows, is monitored until it and the motor position command are equal. When these two (2) variables are equal, the motor command velocity (**X2C**) is set to zero (0) and the nutrunner shaft is stopped.

4.2.2 Nutrunner Control Algorithms

The control calculations for the nutrunner are very similar to the control of the gripper described above. Different methods of measuring the position of the nutrunner motor are employed, as well as derivation of a velocity measurement. In addition, control of the nutrunner torque is performed to prevent damage to the nutrunner motor.

4.2.2.1 Nutrunner Feedback

The first step of control is measurement of position and velocity of the nutrunner motor. Techniques were developed to utilize the commutation signals from the motor to produce these measurements. Special counters within the DDSC's EPLD monitor the commutation inputs and count their transitions (changes in the states of the commutation signals). Additional counters increment with a fixed clock rate to indicate the elapsed time since the last transition of the commutation signals. For a given velocity, each transition count indicates a fixed portion of a revolution has occurred.

The basis for determining the position is that each pole pair of the motor has six (6) commutation states associated with it. The total number of transitions per revolution of the motor is simply six (6) times the number of pole pairs in the motor. The transition counts are accumulated and converted to revolutions to produce a position measurement (**ANGT**). The velocity measurement (**VLMEAS**) is also integrated and the result added to the accumulated angle to produce a more accurate position measurement (**ANGTEX**) to be used by the control algorithms. Note that this measurement is for the nutrunner motor, not the output shaft. To determine the number of revolutions of the shaft, divide the motor position measurement (**ANGTEX**) by the gear ratio of the nutrunner.

The transition counts and the timer counts are also used to generate a velocity measurement (**VLMEAS**). By dividing the angle data by the time for 1 period an average velocity (**AVGVEL**) is calculated for the sample period. The difference between the commanded torque (**TORQ**) and the friction estimate (**FRICT**) is also integrated and added as a correction and update to the average velocity to obtain the velocity measurement (**VLMEAS**), which is utilized by the control algorithms. As the motor slows down and no transitions are seen to occur, the velocity measurement is "decayed" until it eventually reaches zero (0).

4.2.2.2 Nutrunner Control Calculations

After the measurements of position and velocity have been made, the states that make up the model of the nutrunner motor are corrected through use of Kalman gains. A "simple" model of friction (**FRICT**) is generated the same way as for the gripper controls, consisting of the derived friction state (**X3H**) and the nominal friction value (**NOMFRC**). As in the gripper controls, the sign of the nominal friction is determined by the sign of the commanded velocity (sign of **NOMFRC** is opposite that of **X2C**). This friction is then applied to the various states of the model during the prediction without any control applied. After the predictions of the system states are completed, the control errors are calculated by comparing the estimated system states with their respective commands. The control acceleration is then computed from these errors in the same manner as the gripper algorithms, except that integral control is not performed. Friction (**FRICT**) is subtracted from the control acceleration (**AT**) and the result limited to form the command acceleration (**AH**). This command is then used to update the different states of the system model and, after conversion to a command torque (**TORQ**) and then a command current (**UIMTRC**), to control the pulse width calculations of the DDSC. A dead-band control is employed to prevent the motor from continuously controlling about discrete feedback. Correction of the pulse width constants for back EMF effects (based on motor speed) is also implemented for the nutrunner.

4.2.3 Nutrunner Output Torque Control

After the command current (**UIMTRC**) has been calculated, it is limited to a User defined level (**MAXTRQ**) before calculating the desired pulse width. This limit is used to prevent the nutrunner from "over torquing" the output shaft and possibly causing damage. A second limit, a variable torque limit, can be set to detect a "stalled" motor. This limit is entered into the DDSC as a torque (**TRQCOM**) in foot-pounds and the program converts this information into a current limit (**TRQLIM**) using the motor scale factor and the gear ratio of the nutrunner. When the commanded current exceeds this limit, a delay counter is started. As long as the commanded current is above the limit, the counter continues to increment, until it reaches a preset value that indicates a "stalled" motor. At that time, the current command to the motor is terminated and all commands to the nutrunner are cleared. Should the current fall below this "stall" limit, the delay counter is reset and monitoring continues. However, in no case does the current exceed that corresponding to MAXTRQ.

A major difference between the gripper control and the nutrunner control is commutation of the output switches. Since the gripper motor is a brushed-type motor, fixed commutation is used. However, the nutrunner motor is a brushless type motor and must have the commutation data written to the GADIC for proper operation. Because the speed of the nutrunner motor can be fairly high, the commutation data from the motor is written to the GADIC each pulse width interrupt, or 10240 times per second.

5 Test Results

5.1 Primary Accomplishment

The primary accomplishment on this program was the development of the Telerobotic Digital Controller System described in the preceding sections of this report. The small size and capability of the self-contained DDSC dual axis controller, combined with the ability to readily monitor and control seven of them in a network, is unique. There are no controllers or controller systems on the market that compete in size and capability with the DDSC and the network of controllers developed on this program.

5.2 Acceptance Test

The principle test results on this program were the demonstration and complete Acceptance Test ran at NASA GSFC and witnessed by NASA engineers. Acceptance testing was not required by the contract. However, the GSFC Technical Director for the program requested it and Navtrol complied. The Acceptance Test Plan was written by Navtrol based on anticipated performance, but before the units were complete and actual test results available. Problems were encountered in several areas that took time to correct even though they affected end performance very little. An example of this was very narrow microsecond spikes in motor current that affected motor control in no noticeable way. The cause of these spikes was not immediately known but was traced to very small variations in sample to sample timing. These timing variations were also innocuous in Robotic applications, but perhaps not for precision tracking. Anyway, the timing variations were corrected and the current spikes eliminated. Correction of this and similar minor problems resulted in an improved controller design and more confidence in it. However, it extended the contract and resulted in expenditure of funds beyond that authorized by the contract. The Acceptance Test Plan is provided in Appendix A.

5.2.1 Network Demonstration

The first thing the test plan required was a demonstration of basic network capabilities. All the tests were run with 6 slaves connected near the end of a cable approximately 200 foot long. The ability to download different programs and/or data to each DDSC slave was demonstrated. The ability to configure each DDSC slave and to control each or all slaves together was demonstrated. The ability to monitor each slave separately or several together and present the data in graphical or alpha numeric format was demonstrated. In summary, the capabilities previously defined for the network and for the DEMON software were demonstrated to the fullest extent.

5.2.2 Current Control Test

Simultaneous control of current into two dummy loads by a DDSC controller slave and for any of the slaves in the network was demonstrated. Steps and sine wave responses were ran for each of the two loops with current control demonstrated for 0 to 8 amps for Loop 1 and 0 to 4 amps for Loop 0. Responses of Loop 0 to steps of 0.2 amps and 3 amps are shown in Figures 5.2-1 and 5.2-2. Figures 5.2-3 and 5.2-4 show Loop 1 current step responses for 0.2 amps and 7 amps respectively.

5.2.3 Nutrunner Tests

For one of the slaves, simultaneous control of the GSFC gripper and GSFC nutrunner were demonstrated. The nutrunner tests, made while the gripper gripped a tennis ball with a controlled force, included controlling nutrunner speed from 5 to 60 rpms, both clockwise and counter clockwise. Figure 5.2-5 illustrates the capability of the DDSC to control nut runner velocities of 40, 50 and 60 rpm at the output shaft. The measured velocity shown on Figure 5.2-5 is that of the motor in revolutions per second. Gearing was such that output speed in rpm was 6/5 times motor speed in rps. Nut runner velocity control at only 5 rpm in both + and - directions is in both illustrated on Figure 5.2-6. Again, what is shown is motor revolution per second which resulted in the commanded speed at the output shaft. Although measured velocity may appear somewhat noisy, the variations are only about ± 1 RPM. Bear in mind that velocity measurement was derived from Hall device motor commutation signals, which provided the only feedback for control. The very discontinuous aspect of the measurements at the slow velocity plus variations in friction as the motor revolved resulted in the small variations shown. The variations are short lived and accuracy of the average speed at the output shaft was + or - 1%.

It was demonstrated that when the output torque, as determined by measurements of motor current, exceeded a previously selected level for a pre-setable delay time, the controller would drop the current to 0. In addition, the torque during the run would always remain below another pre-set level. This demonstrated the capability to tightened a nut to a certain torque and not exceed a "safe" torque which would not strip the threads or otherwise cause problems. The ability to command the nutrunner to traverse a specific angle at a commanded velocity was demonstrated, as shown in Figure 5.2-7. Angular accuracy was + or - 1 degree. Finally torque output verses commanded torque was checked by using a torque wrench. Because of the way the torque wrench operated, this turned out to be a fairly rough test, but was completed to everyone's satisfaction.

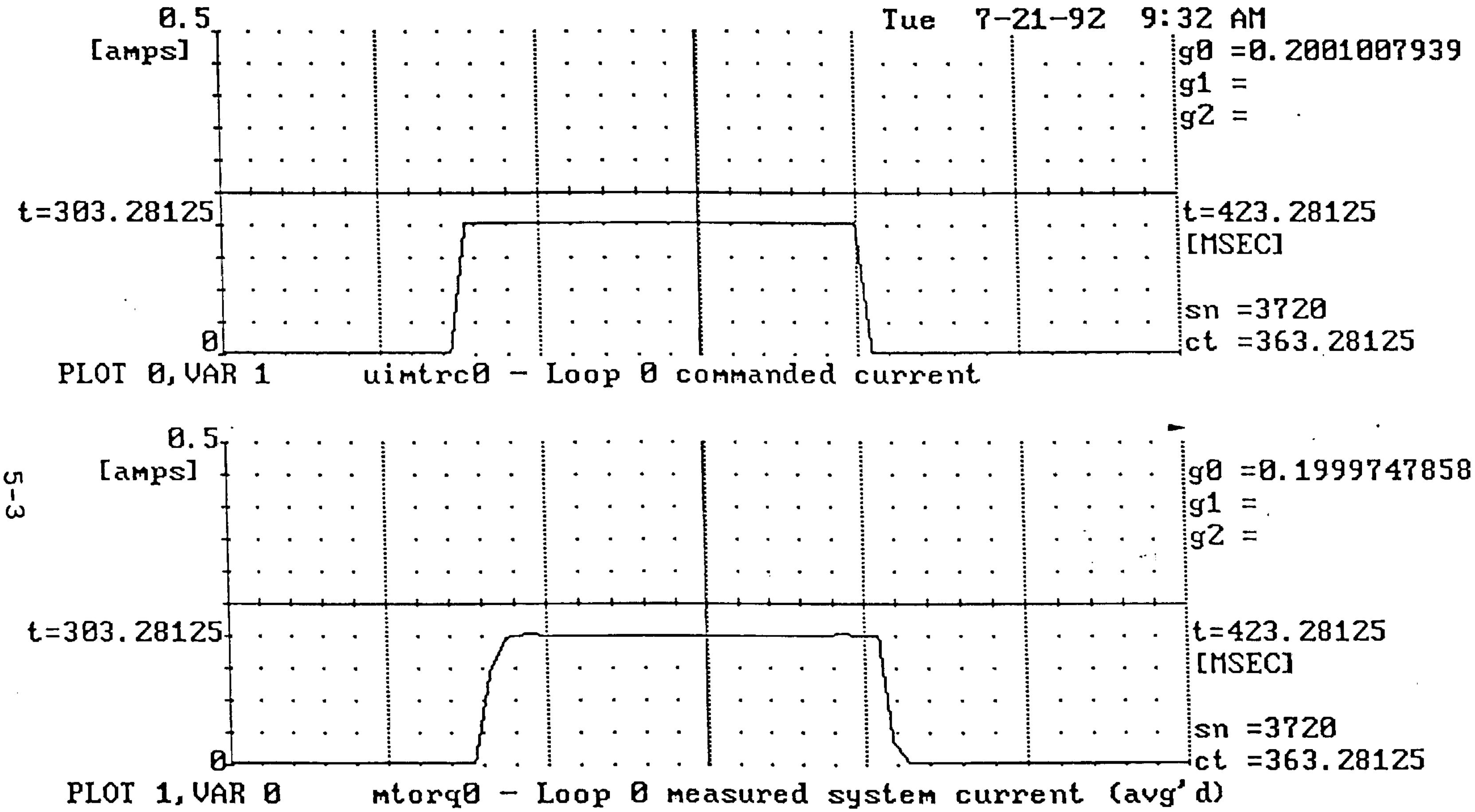


Figure 5.2-1: Loop 0 Current Step Response, 0.2 Amp

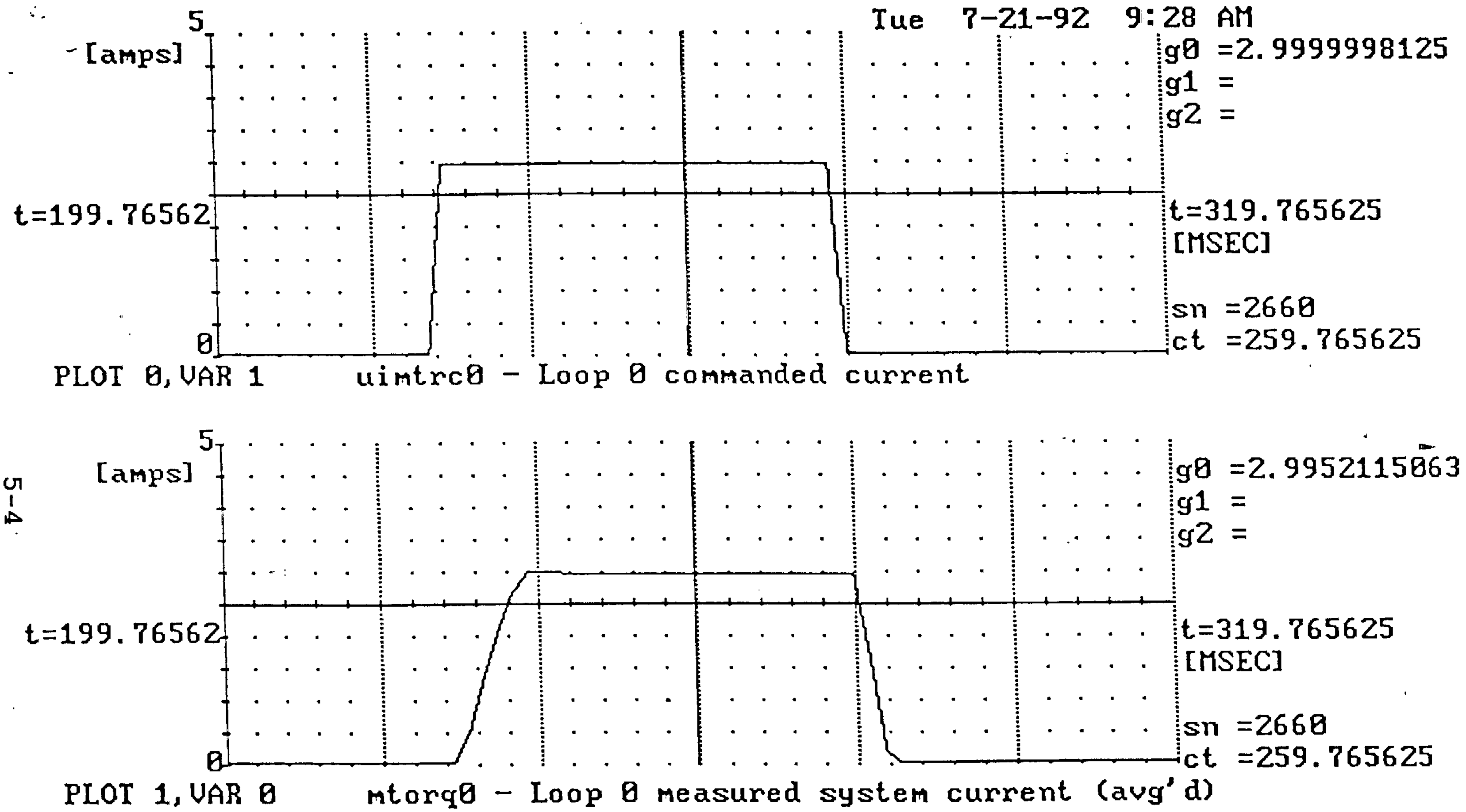


Figure 5.2-2: Loop 0 Current Step Response, 3 Amps

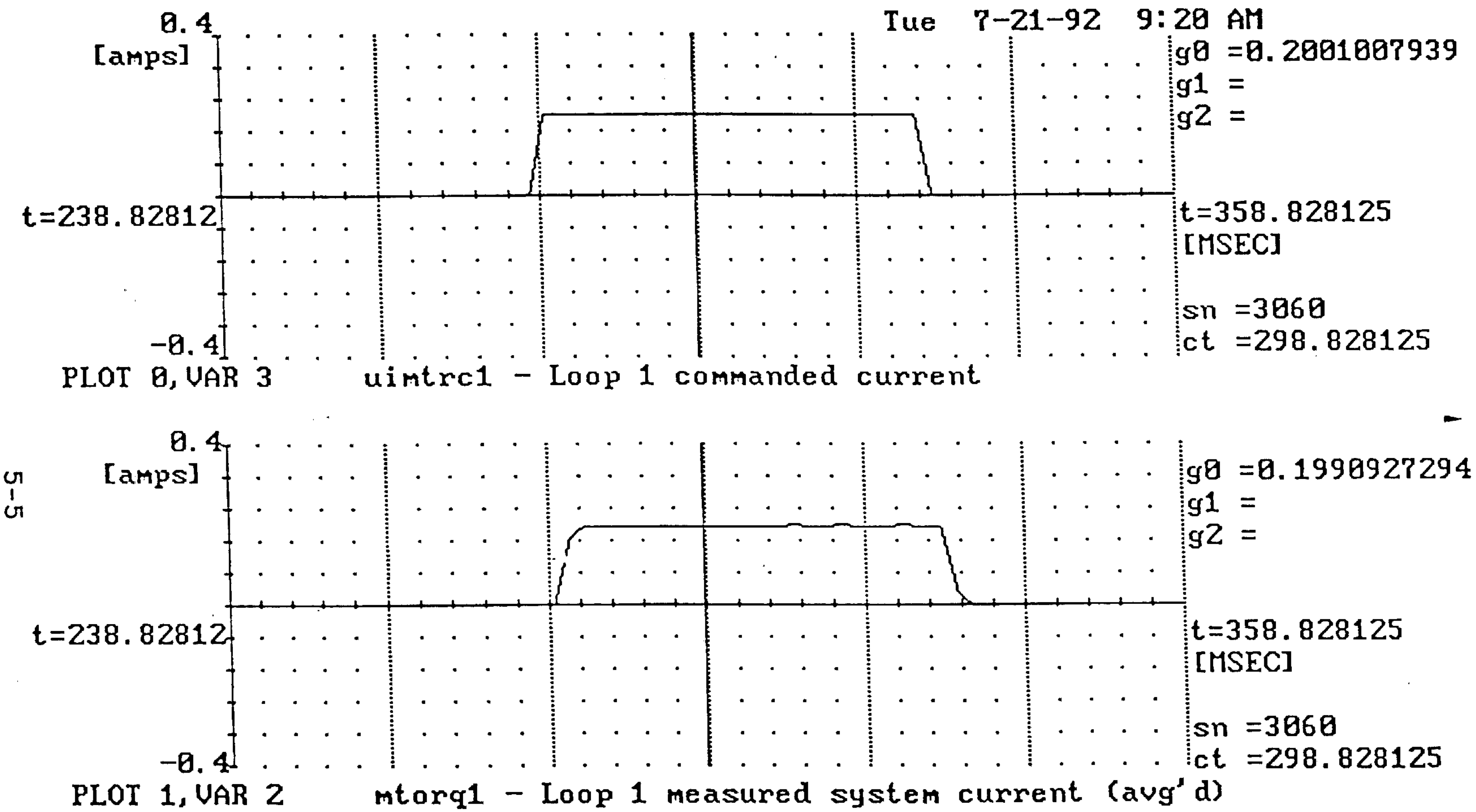


Figure 5.2-3: Loop 1 Current Step Response, 0.2 Amp

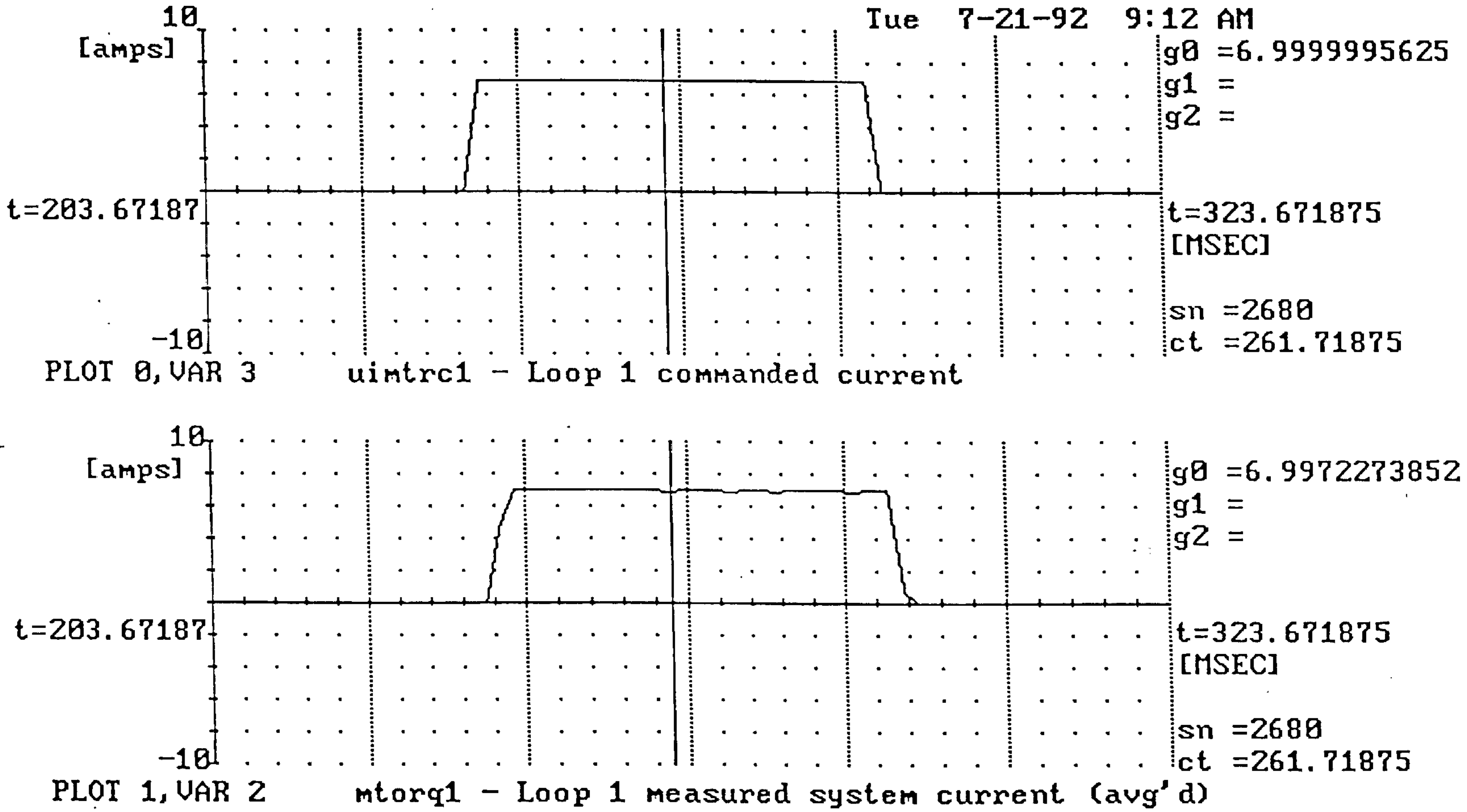


Figure 5.2-4: Loop 1 Current Step Response, 7 Amps

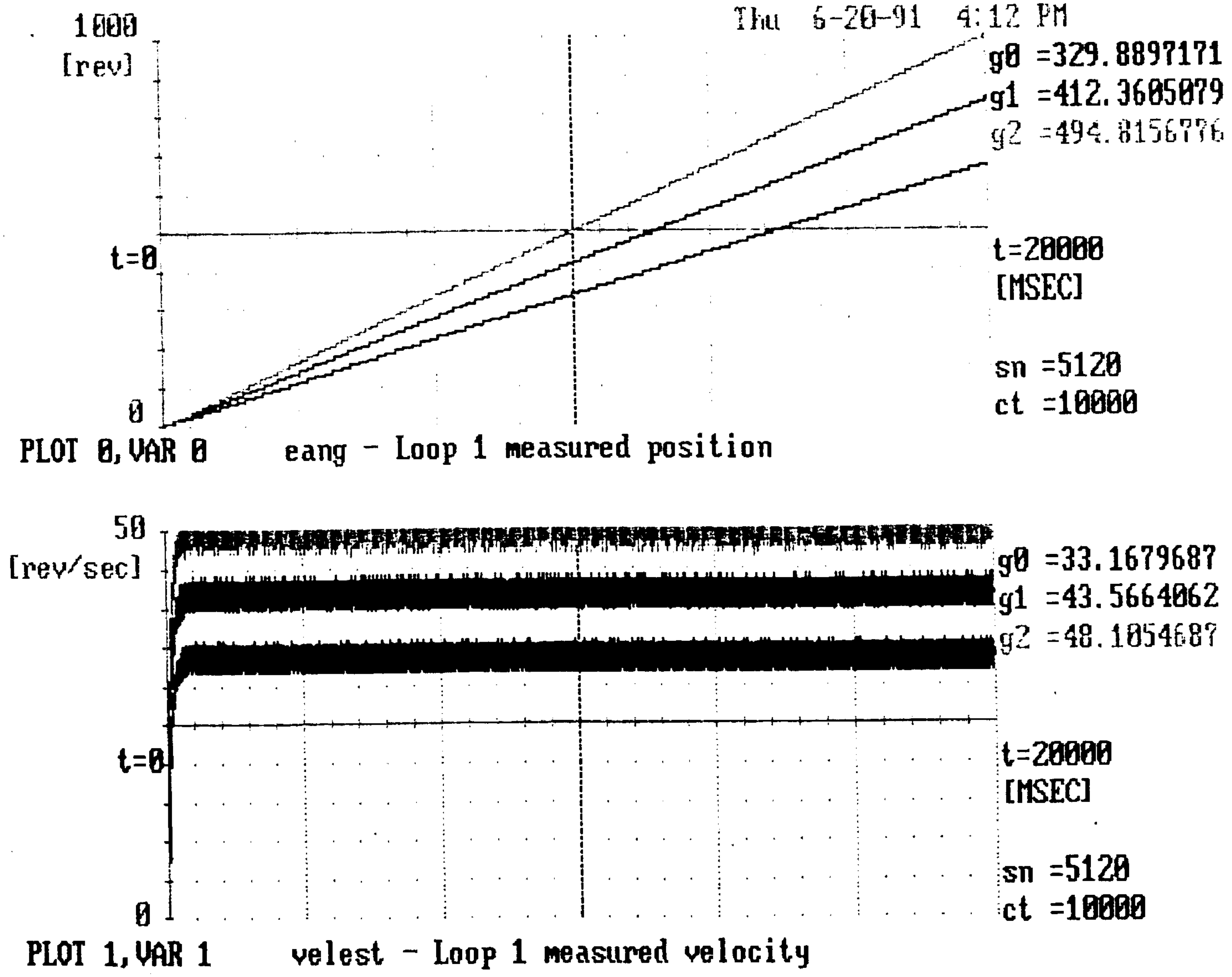


Figure 5.2–5: Nutrunner Velocity Control Runs

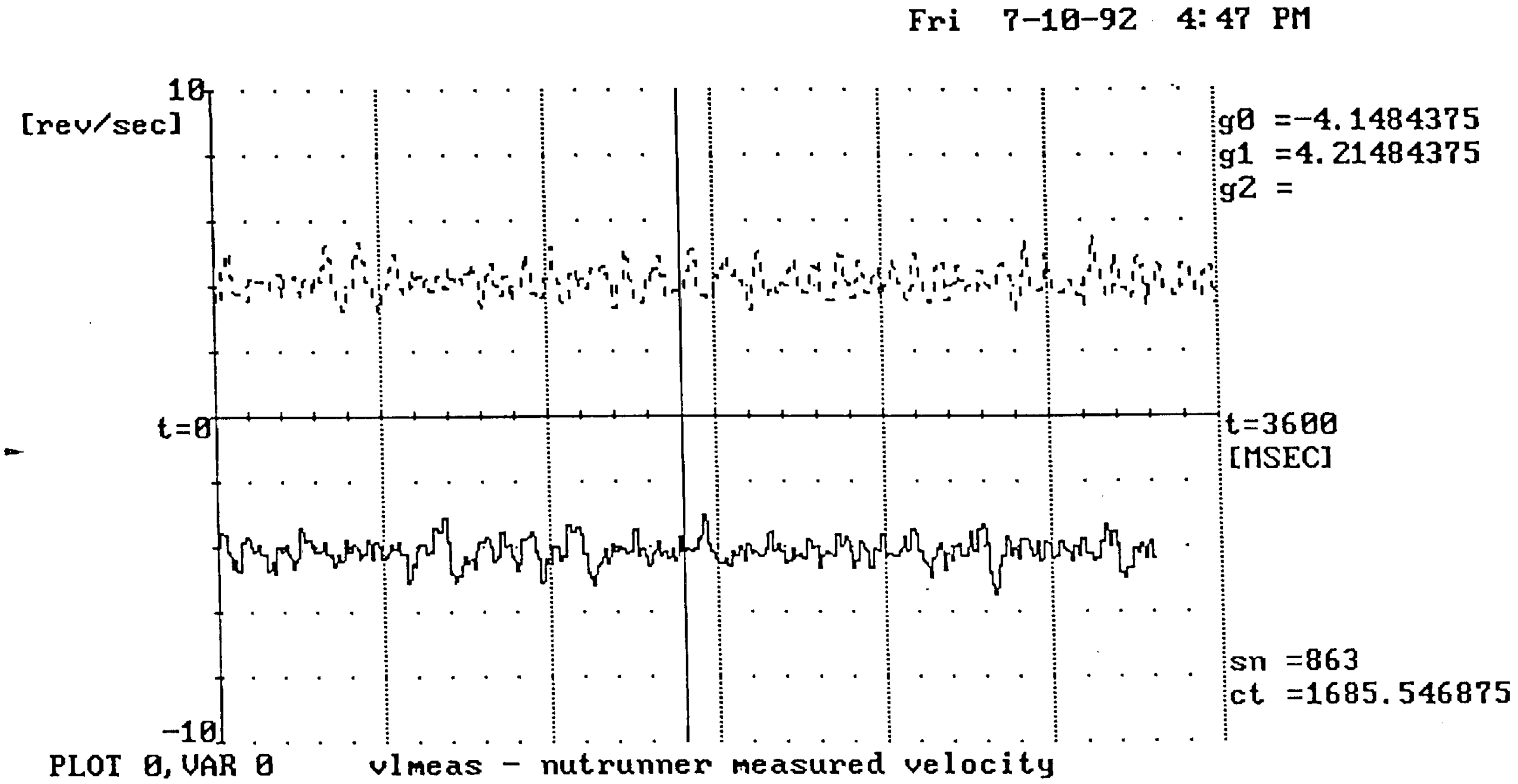


Figure 5.2-6: Nutrunner Velocity Control at + and - 5 RPM

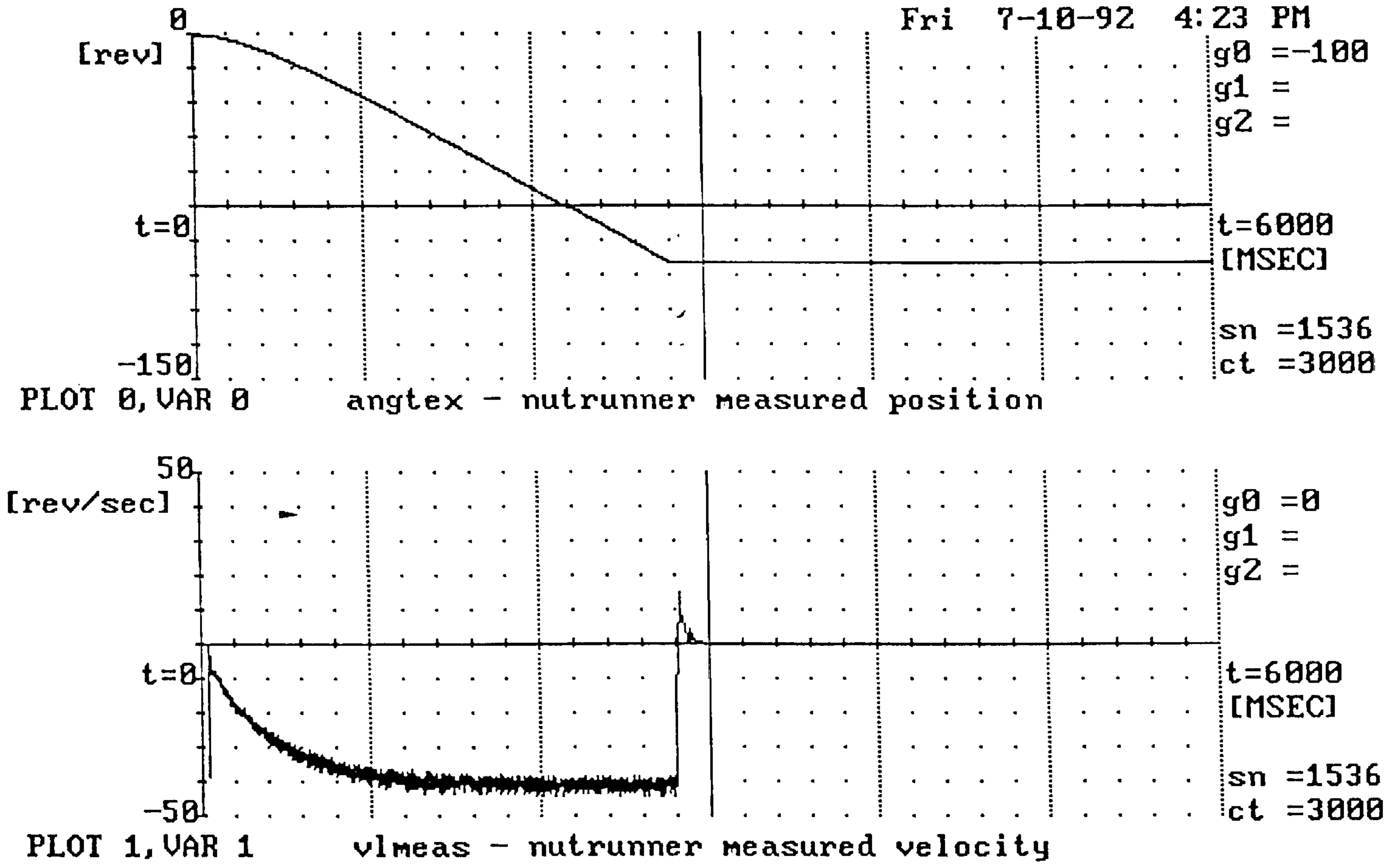


Figure 5.2-7: Nutrunner Response to a 100° Commanded Step in Position

5.2.4 Gripper Tests

Gripper tests were ran while the nut runner maintained its output shaft turning at 12 rpm. The first test include closing the gripper on a tennis ball with a force of 25 lbs.. It was demonstrated that the control was bi-lateral by squeezing the tennis ball with the hand and causing the jaws to open in order to maintain the 25 lb. force commanded. The test was repeated at various force levels. Force control in opening the jaw was demonstrated by forcing the jaws closed with one's hands while controlling the grippers opening force to 10, 20 and 30 lbs.. Control of the gripper jaw opening from closed to 5.9 inches was demonstrated.

The ability of the gripper to grip items made up of different material was demonstrated. The force and position response of the gripper in gripping an aluminum block, a tennis ball, and an aluminum can are all illustrated on Figure 5.2-8. When the gripper gripped the aluminum block the position stopped immediately on impact and the force increased rapidly. When the gripper encountered a tennis ball, the tennis ball compressed allowing the position to continue while the force increased at a more gradual rate than that which occurred for the aluminum block. For the aluminum can, force behaved erratically as the can crumpled under the advancing gripper arms. In all cases the transition from position control to force control was smooth and clean. No interference between the two control channels, for the gripper and nutrunner, was noted and satisfactory results for all tests were achieved.

One of the advantages of the DEMON Software Servo Development System is its ability to readily change system parameters to quickly optimize system performance. Figure 5.2-9 shows different bandwidth force responses for the Gripper in gripping a metal lock. The response at 25 Hz bandwidth is clearly smoother and more sure than that at 10 Hz. Although one might expect that the response at the wider bandwidth would be improved, the amount of improvement requires test results to determine, since control is not linear and unknown compliances are contained within the system.

5.3 Post Program Progress

Since the delivery of the complete system to NASA GSFC and acceptance there, development has continued on this product. Primarily, the further development was aimed toward packaging the unit into a rugged module which could readily be installed or removed for various applications. The specific application in mind was for two of Robotic Research Corp. (RRC) robots to be delivered to Oak Ridge National Laboratory at Oak Ridge, Tennessee. In addition to repackaging, additional capabilities have been added. One example was to improve the precision of force control.

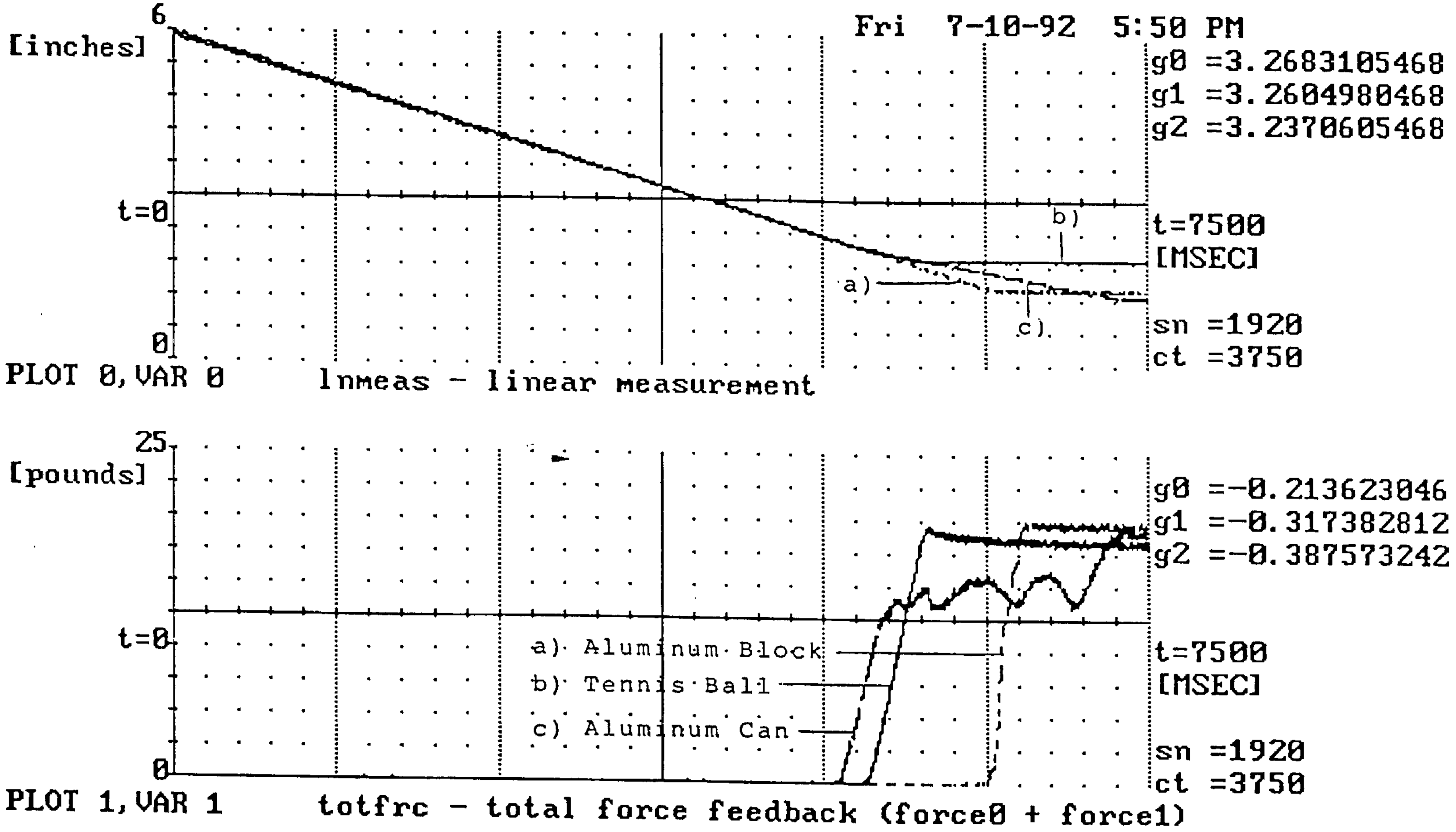


Figure 5.2–8: Gripper/Position Force Control Plots

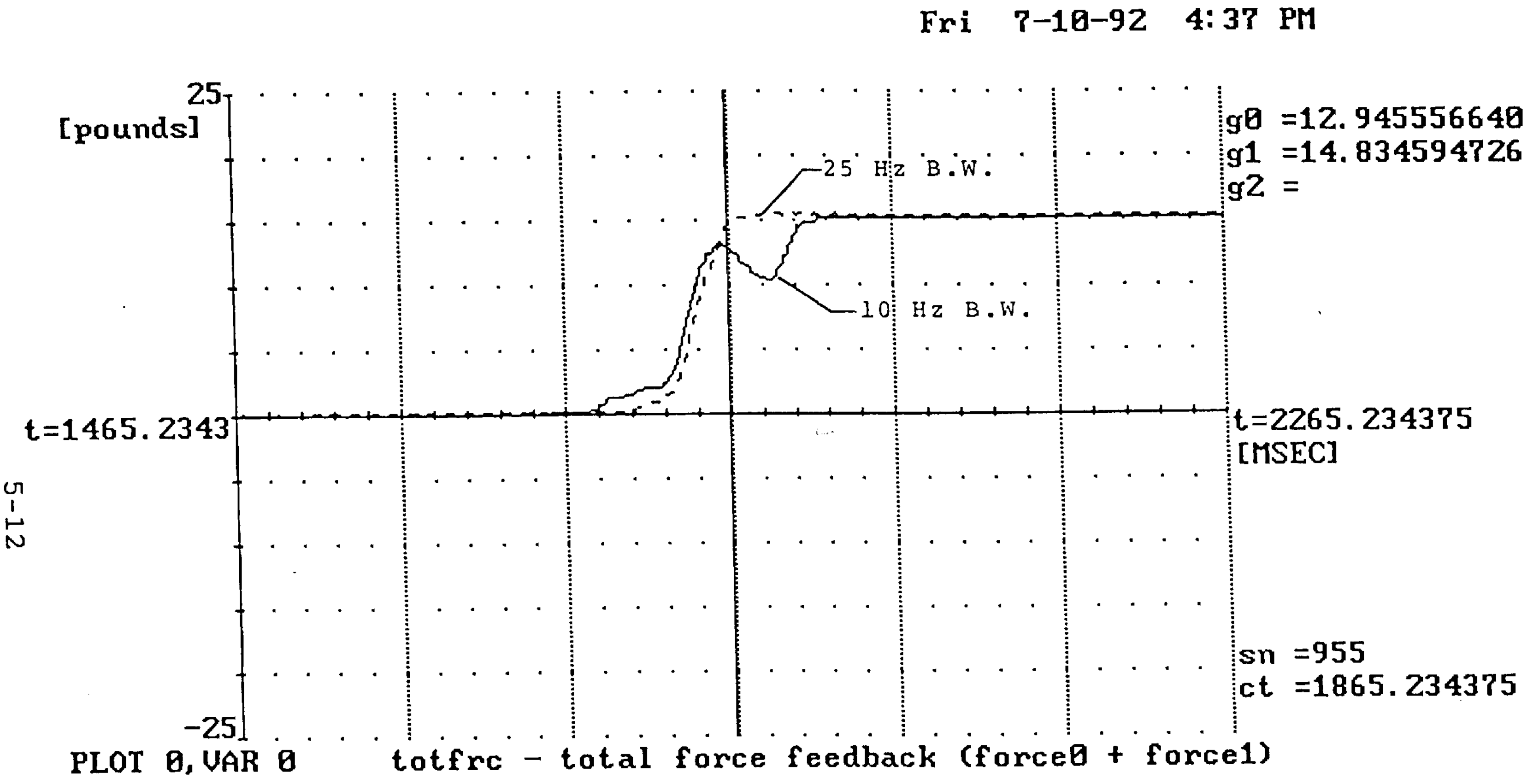


Figure 5.2–9: Plots of Gripper Force on a Metal Block
(Responses at 10 Hz and 25 Hz B.W.)

Figure 5.3-1 illustrates the module which is now being developed. This is a Dual Axis Controller capable of providing 15 to 20 amps at 90 volts. It is unique in size and capability and should find many and varied applications.

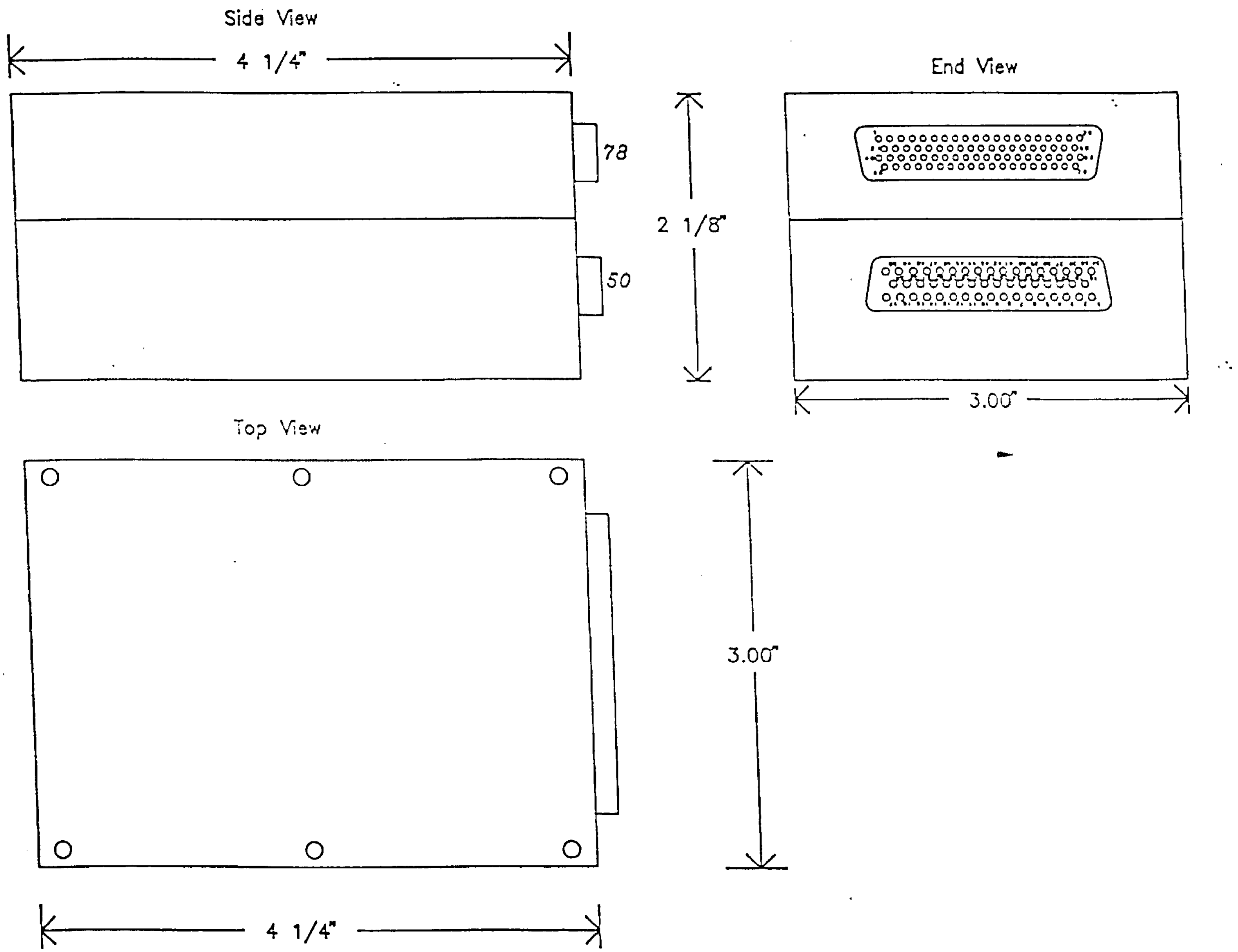


Figure 5.3–1: DDSC–3 Dual Axis Digital Servo Controller
(Not drawn to scale)

6 Conclusions and Recommendations

6.1 Conclusions

This SBIR Phase II development program for a Telerobotic Digital Controller System was a success. The small, compact, high capability, self contained DDSC Dual Axis Controller, combined with the ability to readily monitor and control seven of them in a network, is a unique and needed product. No controller or controller system on the market competes in size or capability with the DDSC controller developed on this program.

A primary objective of the SBIR program is that the Phase II development leads to commercial applications of the product. The productizing and commercialization of the DDSC, in the so called Phase III effort, is well underway. It is being repackaged as a very compact module and, as such, has already found commercial applications. It will be installed on two robots from Robotic Research Corporation (RRC) to be delivered to the Oak Ridge National Laboratory at Oak Ridge, Tennessee. The Automation and Robotic Research Institute (ARRI) a branch of the University of Texas at Arlington, has recommended use of a network of DDSC's for their "Stewart Platform" robots. Their program is presently in early development and if it goes forward and if the sponsor approves, Navtrol's DDSC controllers will be utilized. Other applications abound, and Navtrol feels sure that more customers will be found.

A second conclusion that this is a needed product in the evolution to higher capability and more precision robots. This was concluded by Navtrol before the development began. Discussions with RRC, ARRI and others has firmed up the conclusion that this system is the way to go for future robotics systems.

6.2 Recommendations

The controller developed on this program is nearly universal in application. Applications not only include robots of various types but also precision pointing and tracking system, such as precision laser pointing in outer space. It is recommended that NASA make use of the advances in technology accomplished on this program and paid for by them. It is further recommended that NASA sponsor the development of a flight version for use in future spacecraft.

APPENDIX A

ACCEPTANCE TEST PLAN FOR:

NAVTROL'S TELEROBOTIC DIGITAL CONTROLLER SYSTEM

ACCEPTANCE TEST PLAN FOR:
NAVTROL'S TELEROBOTIC DIGITAL CONTROLLER SYSTEM

1. INTRODUCTION

A Telerobotic Digital Controller System was developed by Navtrol under Phase I and Phase II SBIR development contracts from Goddard Space Flight Center. Seven dual axis servo controllers can be controlled from a single board within an AT computer. Another seven dual axis controllers can be controlled from another board located within the same AT computer. Each Dual Axis Controller is very compact measuring only 4.5 inches X 5.1 inches in size. Inner-board wiring is minimized as only 4 wires total, 2 power and 2 communication are required for inter-connections of the seven controllers. The controller is capable of controlling robotic joints and/or end effector functions depending on the down loaded program. Communication is maintained with all seven controllers at their basic sampling rate of 512 samples per second. Since all seven controllers can be monitored simultaneously the effect of control action in one joint on another joint is easily monitored and analyzed. Note that each of the seven controllers can be monitored, parameters adjusted and functions controlled interactively from the supervisory computer.

2. DEMONSTRATION OF BASIC NETWORK CAPABILITIES

These tests will be ran with 6 or more slaves connected near the end of a cable approximately 200 foot long. (Requirement was for a 150 foot cable.)

2.1. ABILITY TO DOWNLOAD INDIVIDUAL PROGRAMS AND/OR DATA TO

EACH DDSC SLAVE.

Two programs will be downloaded. One program will provide gripper and nut runner control while the other program will simulate joint control.

2.2. ABILITY TO CONFIGURE EACH DDSC SLAVE

Ability to adjust the filter and control gains for each individual slave will be demonstrated.

2.3. ABILITY TO CONTROL EACH SLAVE OR ALL SLAVES

Ability to command all slaves into STOW or CLEAR will be demonstrated. STOW is a predefined position. CLEAR is a non operative mode with power outputs set to zero. Ability to command each individual slave into either CLEAR, IDLE, or OPERATE will be demonstrated.

2.4. ABILITY TO MONITOR EACH INDIVIDUAL SLAVE SEPARATELY OR TOGETHER.

Ability to gather data from individual slaves or several slaves through time responses or multiple data points will be demonstrated.

3. <u>TORQUE (MOTOR CURRENT) TESTS</u>

Connect a controller to 2 dummy current load.s

3.1 Command 8A. and 4A. for the Nut Runner and Gripper channels respectively.

(a) Measure using DEMON. (Error less than 3%.)

(b) Measure with multimeter. (Error less than 3%.)

3.2 Repeat 3.1 with current at 0.8A and 0.4A.

3.3 Run sine wave (5 Hz) of current into both axis (6A and

3A peak) simultaneously and compare commanded and measured current.

3.4 Put 8A step into Nut Runner axis. Use array data to show response. Repeat for 4A..

3.5 Put 4A step into gripper axis. Use array data to show responses. Repeat for 2A..

4. GRIPPER AND NUT RUNNER TESTS

Demonstrate simultaneous operation of the GRIPPER and NUT RUNNER.

4.1 NUT RUNNER TESTS (Made while GRIPPER grips a tennis ball with 15 lbs. of force.)

4.1.1 Demonstrate control of NUT RUNNER speed from 6 to 60 rpm both clockwise and counter clockwise. Average speed accuracy ± 1%. Repeat with varying "small" disturbance torques applied.

4.1.2 With NUT RUNNER speed at 10 rpm clockwise and counter clockwise torque limit set at 7 and 2 ft. lbs.. Use hands to stop NUT RUNNER. Use DEMON to measure current and torque. Torque should equal the commanded torque, ± 10% or ± 0.5 ft. lbs, whichever is more. (Motor current can be controlled to ± 3% but friction is a larger unknown.) Current should drop to zero after "Set Current Time Delay." Higher torque limits can be checked only by using mechanical constraints such as vices or wrenches.

4.1.3 Repeat 4.1.2 at 6 ft. lbs. .

4.1.4 Command Nut Runner angular change of 720 degrees at a velocity of 50 rpm. Angular change should be 720 deg ± 1.0 degrees. Check other angular changes. All should be within + 1.0 degree. Measurements should be ± 0.5 degrees.

4.1.5 Using Torque Wrench measure torque outputs verses commanded torque. This is a static test with both Nut Runner and Torque Wrench mechanically restrained.

4.2 GRIPPER TESTS (Made while Nut Runner turns at 12 rpm with 7 ft. lbs. of torque capability.)

4.2.1 Close Gripper on tennis ball with force of 30 lbs. Measure force using the NASA installed strain gauges through DEMON. Accuracy should be better than ± 2 lbs.. Squeeze tennis ball. Jaws should open to maintain the commanded force. (Note: Accuracy of force measurement depends on accuracy of strain gauges and how well they are calibrated. Control of the force depends mostly on Gripper mechanical integrity.)

4.2.2 Repeat 4.2.1 with forces of 5, 10, 15, 20, and 25 lbs. Measure forces using the strain gauges through DEMON.

4.2.3 Repeat 4.2.2 using a load cell. Compare load cell output with DEMON presented results.

4.2.4 Try to keep the Gripper closed by restraining, mechanically or with your hands, the jaws from opening. Control the opening force to 10, 20 and 30 lbs. and observe strain gauge measurements on DEMON.

4.2.5 Control Gripper jaw opening from 0.25 inches to 5.90 inches. Accuracy should be ± 0.1 inch. (Potentiometer can be measured with a resolution better than 0.01 inch.)

4.2.6 Command Gripper jaw opening at 1.0 inches with a force of 10 lbs.

(1) Without constraints, jaws should close to 1.0 inches ± 0.1 inch.

(2) Force jaws open. Closing force should be 10 lbs ± 2 lbs.

4.2.7 Set gripper velocity to 0.4 "/s through DEMON. Command gripper jaw open. With DEMON, use time-tagged position measurements to determine gripper velocity. Velocity in the middle of the move should be within ± 0.1 "/s.

4.2.8 Repeat 4.2.7 with gripper velocity at 0.6 inches/sec. (Results of 4.2.7 and 4.2.8 depend primarily on gripper mechanical integrity.)

NASA
National Aeronautics and Space Administration

Report Documentation Page

1. Report No.
-

2. Government Accession No.
-

3. Recipient's Catalog No.
-

4. Title and Subtitle
Final Report for:
A Telerobotic Digital Controller System

5. Report Date
10/92

6. Performing Organization Code
-

7. Author(s)
Dr. Richard J. Brown

8. Performing Organization Report No.
-

10. Work Unit No.
-

9. Performing Organization Name and Address
The NAVTROL Company, Inc.
9204 Markville Drive
Dallas, TX 75243

11. Contract or Grant No.
NAS5-30633

13. Type of Report and Period Covered
Final Report 5/89-9/92

12. Sponsoring Agency Name and Address
National Aeronautics and Space Administration
Goddard Space Flight Center
Greenbelt, Md. 20771

14. Sponsoring Agency Code
-

15. Supplementary Notes

16. Abstract
This is the Final Report on an SBIR Phase II Contract for "Development of a Telerobotic Digital Controller System." This system is a network of joint mounted Dual Axes Digital Servo Controllers, providing control of various joints and end effectors of different Robotic systems. This report provides description of and user required information for the Digital Controller System Network (DSCN) and, in particular, the Dual Axis Digital Servo Controllers (DDSC), Model DDSC-2, developed to perform the controller functions. The DDSC can control 3 phase brushless or brush type DC motors, requiring up to 8 amps. Only four wires, two for power and 2 for serial communication, are required, except for local sensor and motor connections. This highly capable, very flexible, programmable servo-controller, contained on a single, compact PCB measuring only 4.5" x 5.1" in size, is applicable to control systems of all types from sub arc second precision pointing to control of robotic joints and end effectors. This document concentrates on the robotic applications for the DDSC.

17. Key Words (Suggested by Author(s))
Robotic, Telerobotic, Servo - Control, Digital Control, Distributed Control

18. Distribution Statement
Unclassified
SBIR Phase II
Limited Distribution

19. Security Classif. (of this report)
U

20. Security Classif. (of this page)
U

21. No. of pages
129

22. Price
-