

AD-A210 148

AD-A210 148

Technical Document 1538
May 1989

The NOSC Code 911 Digital Dynamics Processor (DDP)

A Mildly Coupled Distributed-
Computing System

S. P. Murphy

Approved for public release; distribution is unlimited.

SDTIC
ELECTE
JUL 05 1989

CH

89 7 03 062

NAVAL OCEAN SYSTEMS CENTER

San Diego, California 92152-5000

E. G. SCHWEIZER, CAPT, USN
Commander

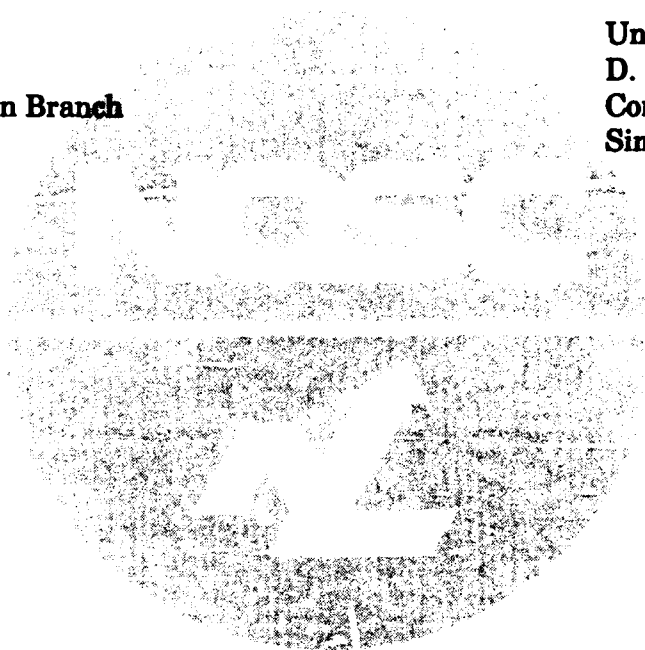
R. M. HILLYER
Technical Director

ADMINISTRATIVE INFORMATION

This document was prepared by Code 911 of the Naval Ocean Systems Center.

Released by
J. A. Mayr, Head
Real-Time Simulation Branch

Under authority of
D. L. Endicott, Head
Computer Sciences and
Simulation Division



JJ

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED			1b. RESTRICTIVE MARKINGS		
2a. SECURITY CLASSIFICATION AUTHORITY			3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.		
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE					
4. PERFORMING ORGANIZATION REPORT NUMBER(S) NOSC TD 1538			5. MONITORING ORGANIZATION REPORT NUMBER(S)		
6a. NAME OF PERFORMING ORGANIZATION Naval Ocean Systems Center		6b. OFFICE SYMBOL (If applicable) NOSC		7a. NAME OF MONITORING ORGANIZATION	
6c. ADDRESS (City, State and ZIP Code) San Diego, CA 92152-5000		7b. ADDRESS (City, State and ZIP Code)			
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Naval Sea Systems Command		8b. OFFICE SYMBOL (If applicable)		9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER	
8c. ADDRESS (City, State and ZIP Code) Torpedo Mk 48 Weapon System Project Office Washington, DC 20362		10. SOURCE OF FUNDING NUMBERS			
		PROGRAM ELEMENT NO. OMN		PROJECT NO. WT35	AGENCY ACCESSION NO. DN988 577
11. TITLE (Include Security Classification) THE NOSC Code 911 DIGITAL DYNAMICS PROCESSOR (DDP)—A Mildly Coupled Distributed-Computing System					
12. PERSONAL AUTHOR(S) S. P. Murphy					
13a. TYPE OF REPORT Interim		13b. TIME COVERED FROM July 1984 TO Apr 1989		14. DATE OF REPORT (Year, Month, Day) May 1989	
				15. PAGE COUNT 14	
16. SUPPLEMENTARY NOTATION					
17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)		
FIELD	GROUP	SUB-GROUP	torpedo system ASW undersea weapon development operational evaluation target echo		
19. ABSTRACT (Continue on reverse if necessary and identify by block number) This document describes the Naval Ocean Systems Center (NOSC), Code 911 Digital Dynamics Processor (DDP). The DDP is a multiprocessor system designed for high speed numerical calculation. This system implements a new variation of common memory in order to allow a far greater number of processors than currently on a Tightly Coupled Distributed System (TCDS).					
20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS			21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED		
22a. NAME OF RESPONSIBLE PERSON S. P. Murphy			22b. TELEPHONE (Include Area Code) (619) 553-2216		22c. OFFICE SYMBOL Code 911

UNCLASSIFIED

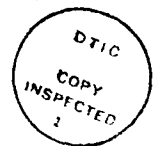
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

CONTENTS

BACKGROUND	1
INTRODUCTION	1
HARDWARE	1
SOFTWARE	3
IMPLEMENTATION	5
REFERENCES	7
BIBLIOGRAPHY	8

FIGURES

1. NOSC DDP system configuration	2
2. NOSC DDP repeater slave address/data paths	2
3. NOSC DDP processor slave data paths	4
4. NOSC DDP processor slave address paths	4



Accession For	
NTIS GRA&I	☒
DTIC TAB	☐
Unannounced	☐
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

BACKGROUND

The definition for distributive computing has many variations. Some of the more detailed definitions can be found in Weitzman (1980) and Sharp (1987). For the context of this report the definitions below will suffice:

Tightly Coupled Distributed System (TCDS). TCDS is a system in which several processors have access to a common block of memory. There are a number of large mainframes of this type (Univac 1100 series is one example). Each processor has its own address hardware for getting data and an overall priority scheme for all the processors to access that memory.

Loosely Coupled Distributed System (LCDS). LCDS is a system in which no common memory and data are shared by a network. Each processor must have a protocol facility for the network to access data/message packets on the network. Both Apollo and Sun market systems of this type.

This definition is generalized. A more exact version can be found in Liebowitz and Carson (1985).

INTRODUCTION

This paper describes the Naval Ocean Systems Center (NOSC), Code 911 Digital Dynamics Processor (DDP). The DDP is a multiprocessor system designed for high speed numerical calculation. This system implements a new variation of common memory in order to allow a far greater number of processors than found currently on a TCDS.

The conceptual design of the DDP was done in mid and late 1984 by Jack Zyphur. A variation of this design was implemented in hardware by Mitchell (1989) in early 1985. The first prototype was only two processor slaves, a repeater slave, and a master. Since the original goal of the DDP was to replace an EAI 8800 Analog computer, the first attempt at software for the DDP was a simulation language consisting of assembly language modules that would do the same functions as analog components. This attempt was later abandoned in favor of direct equation implementation using a "C" cross-compiler that ran on the master. In early 1987 Murphy programmed in a quaternion algorithm and the DDP proved operational in the NOSC Hybrid-Simulator. The DDP has since been up graded to six processor slaves running MC68030 CPUs at 25 MHz.

HARDWARE

The DDP's maximum configuration is 16 cages with 15 Processor Slaves (PSs) and one Repeater Slave (RS) per cage and one Master Processor controlling the system (figure 1). The system can be used with any number of PSs from 0 to 239. The RS is used primarily to buffer signals between the master and the multiple sets of cages (figure 2).

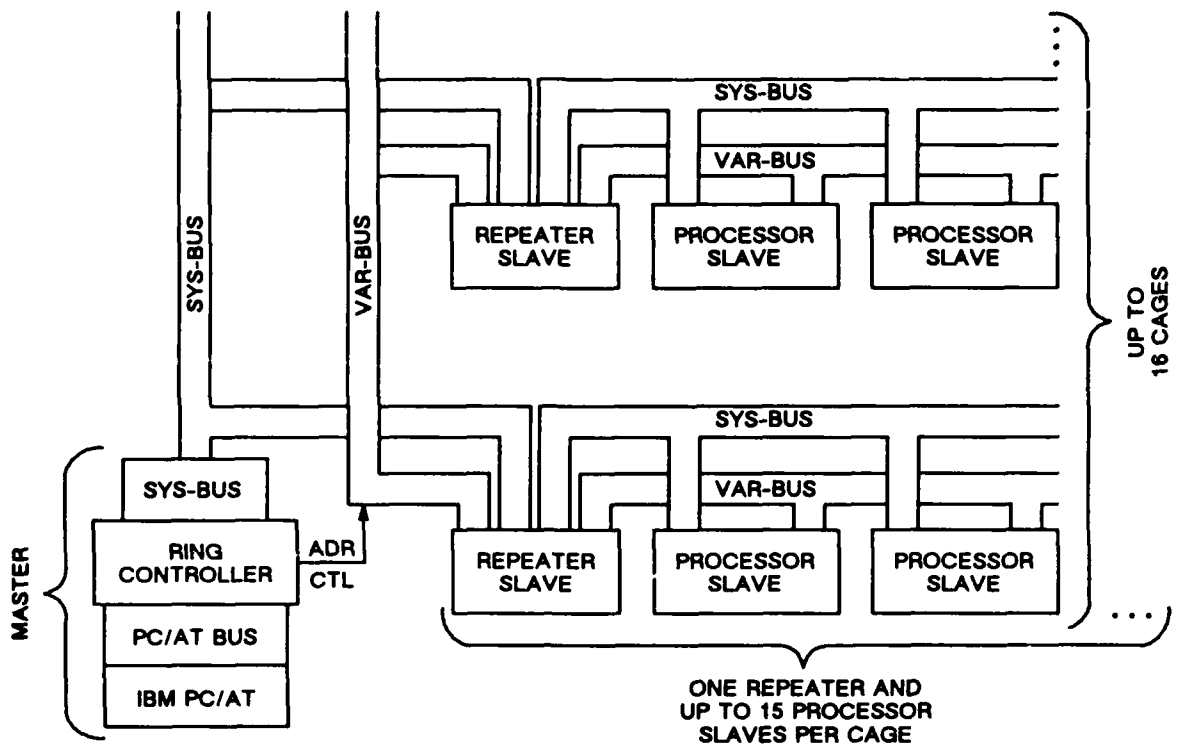


Figure 1. NOSC DDP system configuration.

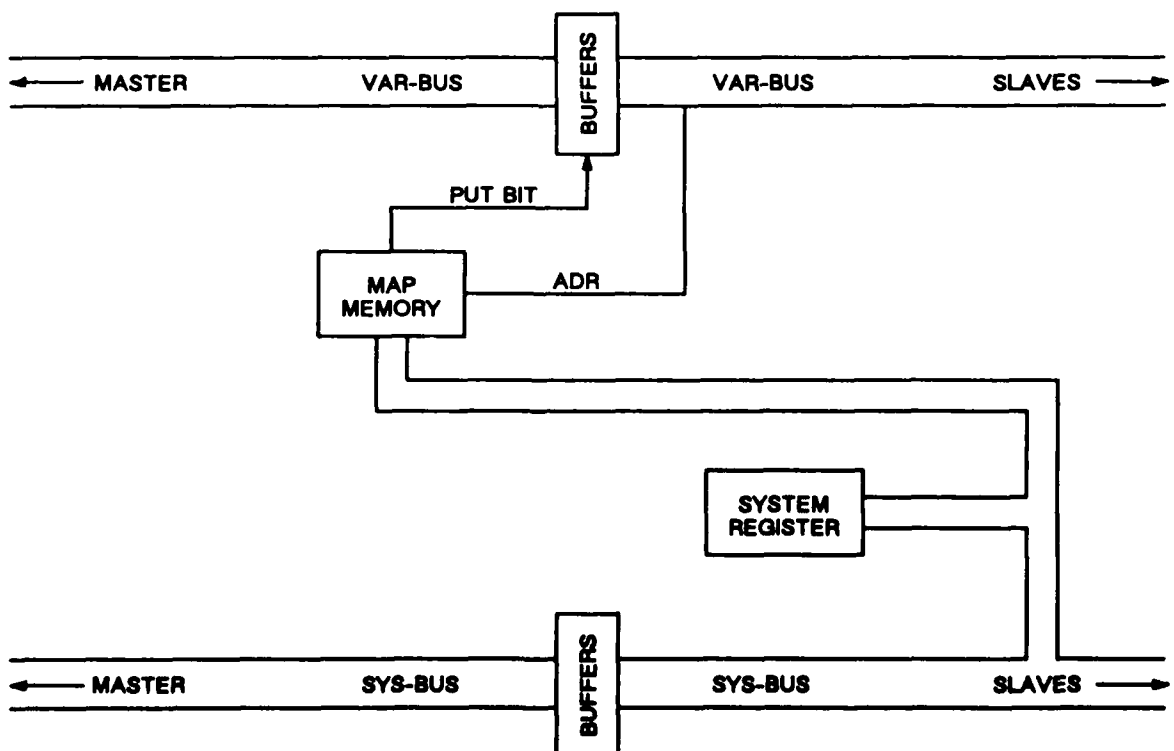


Figure 2. NOSC DDP repeater slave address/data paths.

Each cage has two 32-bit data buses, one for the system bus (S-bus) and one for the variable bus (V-bus) (figure 3). The PSs share common data with each other via the V-bus using a PUT-TAKE system.

The PUT-TAKE system consists of one 12-bit binary counter located on the master, and a Map and Variable Memory internal to each PS (figure 4). The counter provides a common address to the Map Memories on all the PSs simultaneously while each individual Variable Memory gets the address from the output of the Map Memory on its own card. When the system is initialized the master downloads a unique binary image into each PS's Map Memory. When the PS starts running, the counter continually loops through up to 4096 unique sets of address and control bits stored in Map Memory. Based on the state of the PUT and TAKE bits in Map Memory at the current counter location, the PS either puts data from Variable Memory to the V-bus, takes data from the V-bus and stores the data in Variable Memory, or remains passive. Normally only one PS will PUT to the V-bus at a time. Since PUT/TAKE has priority in accessing data in Variable Memory, the PS CPU can only access Variable Memory during IDLE times (neither PUT nor TAKE). This type of virtual common memory allows multiple PSs to get a current copy of the data at the same time (several TAKING at once).

The PUT-TAKE system gives each PS the impression it has 16K bytes (4096 x 32 bits) of memory in common with all the other PSs (previously defined as a TCDS). The PUT-TAKE system is implemented in hardware so there is no overhead cost to the PS's CPU as in the LCDS. For large lookup tables there is enough Program Memory on each PS so it can store its own copy of the table if need be. This leaves the V-bus solely for active data sharing. The V-bus takes 300 nsec to transfer an individual 32-bit value from one PS's Variable Memory to any or all of the other PSs' Variable Memories. If all of the 4096 variables are not used more critical variables can be repeated in Map Memory to provide a faster update rate. The data latency between a CPU's write into Variable Memory and another CPU's read of the new value is approximately 1 μ sec to 1 msec. The actual time depends on where in the V-bus counter cycle the write occurs, how many total variables are used, and the number of times the specific variable is repeated.

To handle communications with external devices each PS also has two 16-bit parallel input and two 16-bit parallel output ports for data I/O plus a serial port for console I/O. The two 16-bit input and output ports may be used in tandem to provide one 32-bit input parallel port and one 32-bit output parallel port if desired.

SOFTWARE

The DDP PSs have no built-in operating system and are controlled by the master. This leaves all the PSs' memory available for program and data and allows the PSs' CPU to do the one thing it should be doing, crunching numbers. This arrangement does place a little more responsibility on the programmer.

The programmer's task is to divide up the problem and assign each PS its portion. This job needs to be done with only one hardware consideration in

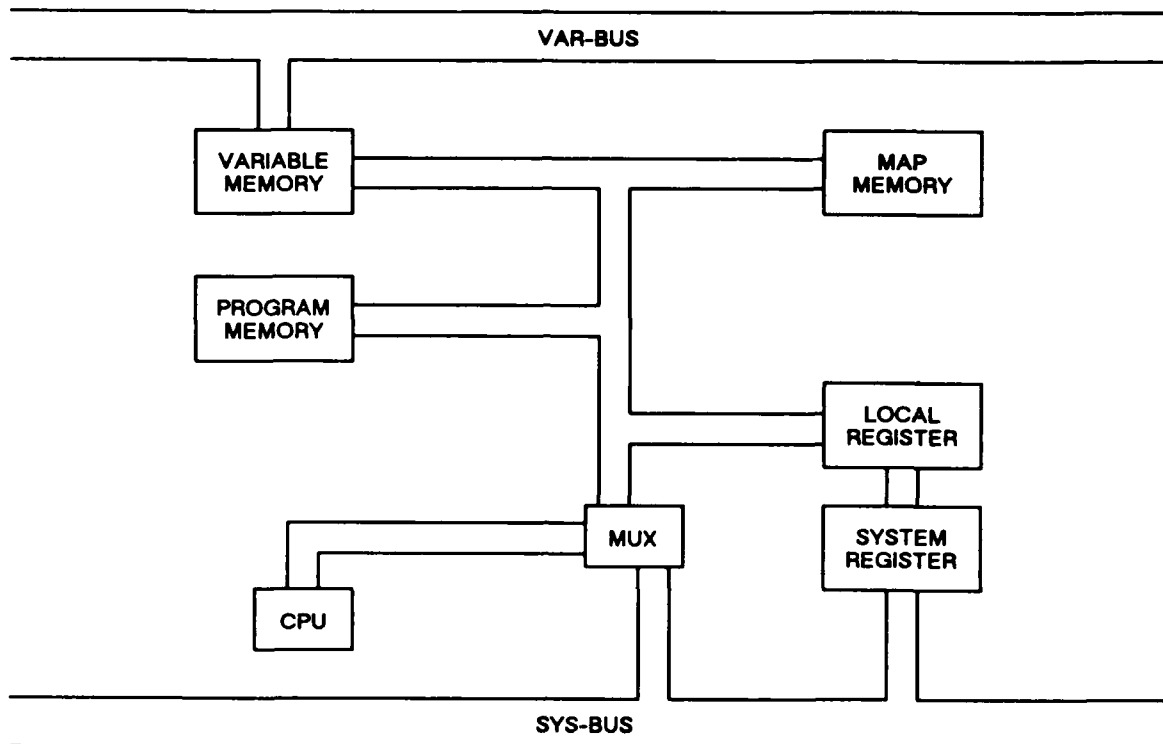


Figure 3. NOSC DDP processor slave data paths.

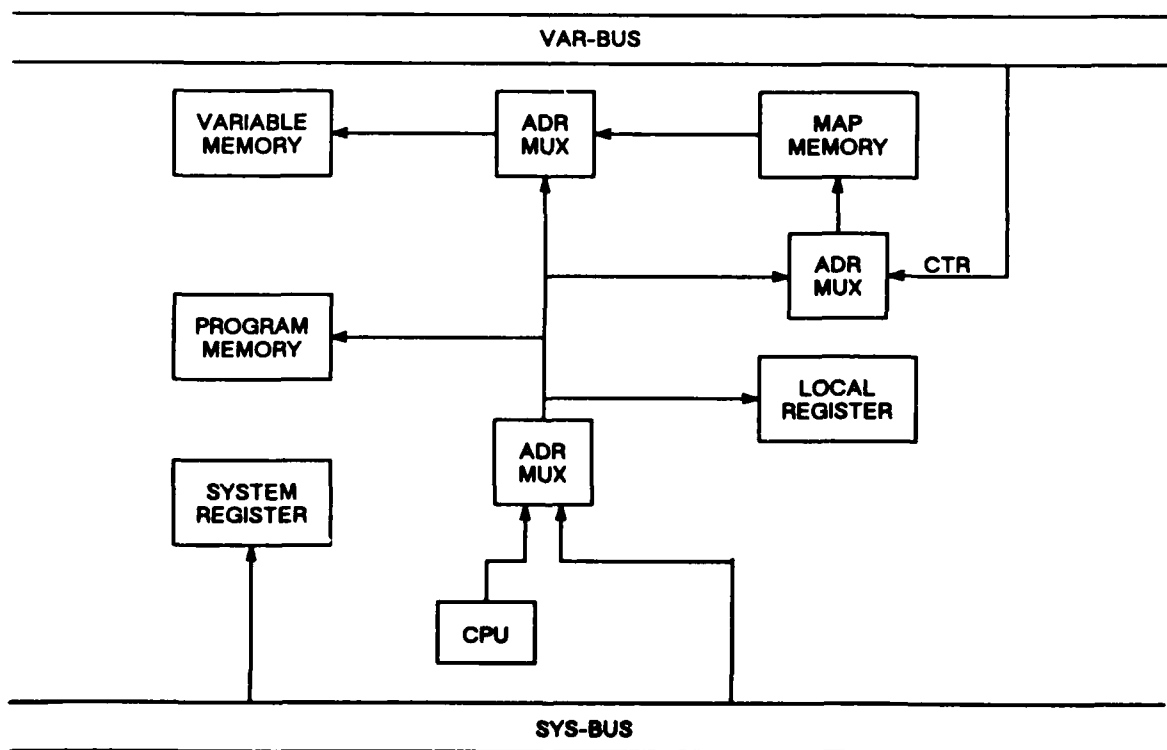


Figure 4. NOSC DDP processor slave address paths.

mind, minimize the number of variables on the V-bus. By doing this the programmer is minimizing the data latency and makes the virtual common memory look more like the ideal common memory.

There is a library of subroutines for the PSs that handles the low-level functions and some tools for debugging. This library has routines for getting data into and out of the I/O ports (16-bit, 32-bit, or serial), a deadman indicator (a register tickled repeatedly or else a LED lights for debugging), a programmable periodic interrupt that goes to all PSs (it can be used for syncing all the PS), and a special version of *printf*() that writes to the PS's serial port (also handy for debugging).

The master has utilities for clearing all the PSs, halting all the PSs' CPUs, downloading the binary images into Program and Map Memory, a C cross-compiler for programming the PSs, and a compiler that generates the binary image for Map Memory. The Map Memory compiler reads an ASCII file in which the programmer specifies which PSs PUT and TAKE which variables. The compiler checks for such things as unique names, undefined variables (TAKEN but not PUT), and warns when more than one PS is putting at a time. When the compilation is complete the compiler outputs the binary Map Memory image for each PS, a header file for each PS (for inclusion in that PS's C program), and some statistics on the V-bus loading. The header file for each PS defines the absolute Variable Memory locations for that PS's shared data. This header file is a helpful debugging aid in making sure each PS has all the V-bus variables defined.

Once a problem is implemented, the setup and loading of the DDP is accomplished by executing a batch job on the master. An operator can take over controlling the DDP, freeing the programmer to do other assignments.

IMPLEMENTATION

In the introduction, the DDP revision 1 proved successful by implementing a solution to the quaternion's parameters in the NOSC Hybrid-Simulator. Revision 2 handled a much greater problem: the real-time solution of the 6-Degree of Freedom (6-DOF) hydrodynamic equations of the Mk 50 torpedo. This solution included the quaternion algorithm previously written. This problem, to which an EAI 8800 Analog computer previously had been dedicated (the analog did not include the quaternions), was accomplished in six PSs. The problem was divided as follows:

- PS (1) X velocity and roll
- PS (2) Y velocity and yaw
- PS (3) Z velocity and pitch
- PS (4) Operator control interface and Simulator I/O
- PS (5) Quaternion states
- PS (6) Quaternion to Euler conversion and Simulator I/O

To help with interface into the hybrid-simulator, an analog cage was designed that contained A/D and D/A converters so that analog values could be read in and placed on the V-bus or data from the V-bus could be sent out as analog values.

The successful demonstration that the DDP could replace an analog system in real-time simulation was held on 4 Jan 1989.

Revision 3 of the DDP hardware is being setup to run two additional problems. The first task is the implementation of Mk 46 torpedo 6-DOF equations with quaternions, an autopilot, and engine simulation. The second task is to do the calculations to control an underwater passive acoustic simulation system.

REFERENCES

Liebowitz, Burt and John Carson, 1985. *Multiple Processory Systems for Real-Time Application*. Prentice-Hall.

Mitchell, Ray, 1989. *NOSC DDP Computer Reference Manual (REV 0)*.

Murphy, Steven, 1987. *Quaternion Solution to Euler Angles on the NOSC Ring Computer*. NOSC TN 1504.*

Sharp, John, 1987. *An Introduction to Distributed and Parallel Processing*. Blackwell Scientific Publication.

Weitzman, Cay, 1980. *Distributed Micro/Minicomputer Systems*. Prentice-Hall.

*NOSC Technical Notes (TNs) are working documents and do not represent an official policy statement of the Naval Ocean Systems Center. For further information, contact the author.

BIBLIOGRAPHY

- Booth, Grayce, 1981. *The Distributed System Environment: Some Practical Approaches*. McGraw-Hill.
- Bradford, R., 1980. "Distribution processing employing a loop network." *IBM Technical Disclosure Bulletin*, Vol. 23, No. 4.
- Brewer, David, 1984. *A Real-time Executive for Multiple-Computer Clusters*. Master's Thesis, Naval Postgraduate School.
- Donaldson, Hamish, 1979. *Designing a distributed processing system*. John Wiley & Sons.
- Harris, R., 1981. *Which is better: shared logic or distributive processing?* Word Processing and Information Systems Vol. 8, No. 1.
- Honeywell, Inc., 1978. *Technical Description of the Experimental Distributed Processor (XDP) System*. NOSC TN 406.*
- IEEE Computer Society, 1988. *IEEE 8th International Conference on Distributed Computing*. IEEE Press.
- Karpuls, Walter, 1987. *Multiprocessors and Array Processors*. Simulation Councils, Inc.
- LeLann, Gerard, 1983. *On Real-time Distributed Computing*. International Federation for Information Processing.
- McEntrie, Paul et al., 1984. *Distributed Computing: Concepts and Implementations*. IEEE Press.
- Morrison, Gordon, 1980. *A Distributed Computing Network for Real-Time Systems*. NUSC TD 5932.
- Warden, Gary and Stan Hodge, 1986, "Local Area Networks for Real-Time Simulations and Integration Support Facility Applications." *Proceedings of the IEEE 1986 National Aerospace and Electronics Conference*.

*NOSC Technical Notes (TNs) are working documents and do not represent an official policy statement of the Naval Ocean Systems Center. For further information, contact the author.